ISNaS - incompressible flow solver
Mathematical manual

Report 93-96

Guus Segal
Kees Vuik
William Kuppen
Marcel Zijlema

TU Delft
Faculteit der Technische Wiskunde en Informatica
Faculty of Technical Mathematics and Informatics

Technische Universiteit Delft
Delft University of Technology

ISSN 0922-5641

Copies of these reports may be obtained from the bureau of the Faculty of Technical Mathematics and Informatics, Julianalaan 132, 2628 BL Delft, phone +3115784568.
A selection of these reports is available in PostScript form at the Faculty's anonymous ftp-site. They are located in the directory /pub/publications/tech-reports at ftp.twi.tudelft.nl

ISNaS - incompressible flow solver

Mathematical manual

Guus Segal
Kees Vuik
William Kuppen
Marcel Zijlema

version 1.0 23-12-1992

Contents

1 Introduction

In this manual we describe the mathematical techniques that are used in the ISNaS incompressible program. We do not give any derivation; for the mathematical theory we refer to the literature used.
This manual is meant for ISNaS developers only.

2 Some basic notations from tensor analysis

In the ISNaS incompressible code we are dealing with curvilinear boundary fitted grids. These grids are mapped (by an unknown transformation) onto a rectangular computational grid. Figure 2.1 gives a typical example of the mapping from physical (i.e. curvilinear) to computational grid. All computations are performed in the computational grid and hence the

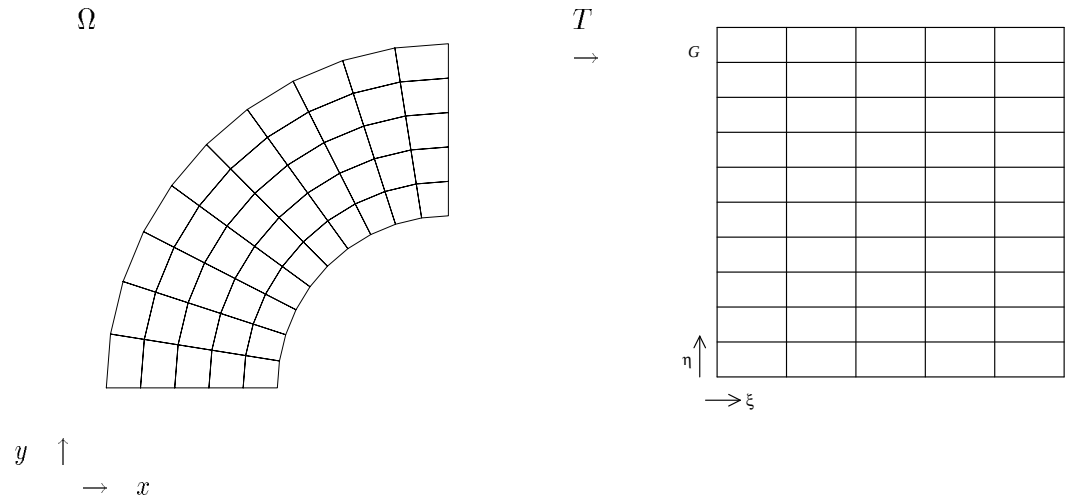


Figure 2.1: Boundary fitted co-ordinates and computational grid

differential equations are transformed from physical grid to computational grid. The resulting solution is transformed backwards.

In the sequel we shall use the following notations:

$\boldsymbol{x}$ = (x^1, x^2, x^3) is the Cartesian co-ordinate system,
$\boldsymbol{\xi}$ = (ξ^1, ξ^2, ξ^3) is the general co-ordinate system,
i.e. the co-ordinate system corresponding to the computational grid.

The mapping T from Cartesian to computational domain is given by

$$T : x^\alpha = x^\alpha(\xi^1, \xi^2, \xi^3) \tag{2.1}$$

We assume that the Jacobian J:

$$J = |\frac{\partial x^\alpha}{\partial \xi^\beta}| \tag{2.2}$$

is unequal to zero.

We define the covariant base vector $\boldsymbol{a}_{(\alpha)}$ as the tangent vector to the surface $\boldsymbol{x}(\boldsymbol{\xi}^\alpha)$, hence

$$\boldsymbol{a}_{(\alpha)} = \frac{\partial \boldsymbol{x}}{\partial \xi^\alpha} \tag{2.3}$$

The subscript α is placed between parentheses to emphasize that $\boldsymbol{a}_{(\alpha)}$ is not a component but one of the three base vectors $\boldsymbol{a}_1, \boldsymbol{a}_2, \boldsymbol{a}_3$.
Contravariant base vectors $\boldsymbol{a}^{(\alpha)}$ are defined as normal vectors to the $\boldsymbol{\xi}^\alpha$ = constant surfaces:

$$\boldsymbol{a}^{(\alpha)} = \boldsymbol{grad}\ \xi^\alpha \tag{2.4}$$

It can be shown that

$$\boldsymbol{a}^{(\alpha)} = \frac{1}{\sqrt{g}} \boldsymbol{a}_{(\beta)} \wedge \boldsymbol{a}_{(\gamma)} \quad \text{for } \alpha, \beta, \gamma \text{ cyclic} \tag{2.5}$$

where $\wedge$ denotes the outer product.

The correspondence between vector and tensor notation for a rank one tensor is expressed by

$$\boldsymbol{u} = U^\alpha \boldsymbol{a}_{(\alpha)} = U_\alpha \boldsymbol{a}^{(\alpha)} \tag{2.6}$$

For a tensor of rank two the correspondence between the two notations is given by, for example in the case of a mixed tensor:

$$\boldsymbol{U} = U^\alpha_\beta \boldsymbol{a}_{(\alpha)} \boldsymbol{a}^{(\beta)} \tag{2.7}$$

The covariant and contravariant components of a vector $\boldsymbol{u}$ can be obtained from

$$U_\alpha = \boldsymbol{a}_{(\alpha)} \cdot \boldsymbol{u}, \quad U^\alpha = \boldsymbol{a}^{(\alpha)} \cdot \boldsymbol{u} \tag{2.8}$$

For a rank two tensor we have for example

$$U^\alpha_\beta = \boldsymbol{a}^{(\alpha)} \cdot \boldsymbol{U} \cdot \boldsymbol{a}_{(\beta)} \tag{2.9}$$

The metric tensor The covariant and contravariant metric tensors $g_{\alpha\beta}$ and $g^{\alpha\beta}$ are defined as follows:

$$g_{\alpha\beta} = \boldsymbol{a}_{(\alpha)} \cdot \boldsymbol{a}_{(\beta)} \quad , \quad g^{\alpha\beta} = \boldsymbol{a}^{(\alpha)} \cdot \boldsymbol{a}^{(\beta)} \tag{2.10}$$

The name metric tensor is related to the fact that the length ds of a small line-segment satisfies

$$ds^2 = dx^\alpha dx^\alpha = g_{\alpha\beta} d\xi^\alpha d\xi^\beta \tag{2.11}$$

The determinant of $g_{\alpha\beta}$ is called g, and is given by

$$\sqrt{g} = \boldsymbol{a}_{(1)} \cdot (\boldsymbol{a}_{(2)} \wedge \boldsymbol{a}_{(3)}) \tag{2.12}$$

The two-dimensional version of (2.12) is given by

$$\sqrt{g} = a^1_{(1)} a^2_{(2)} - a^2_{(1)} a^1_{(2)} \tag{2.13}$$

By writing out the right-hand side one sees that

$$\sqrt{g} = J \tag{2.14}$$

The covariant derivative A covariant derivative is a tensor which reduces to a partial derivative of a vector field in Cartesian coordinates. For a scalar, the covariant derivative is the same as the partial derivative, and is denoted by

$$\mu_{,\alpha} = \frac{\partial \mu}{\partial \xi^{\alpha}} \tag{2.15}$$

The covariant derivative of a contravariant tensor of rank one is given by

$$U^{\alpha}_{,\beta} = \frac{\partial U^{\alpha}}{\partial \xi^{\beta}} + \{ \begin{matrix} \alpha \\ \gamma\beta \end{matrix} \} U^{\gamma} \tag{2.16}$$

where $\{ \begin{matrix} \alpha \\ \gamma\beta \end{matrix} \}$ is the so-called Christoffel symbol of the second kind given by

$$\{ \begin{matrix} \alpha \\ \gamma\beta \end{matrix} \} = \boldsymbol{a}^{(\alpha)} \frac{\partial \boldsymbol{a}_{(\gamma)}}{\partial \xi^{\beta}} = \frac{\partial \xi^{\alpha}}{\partial x^{\delta}} \frac{\partial^2 x^{\delta}}{\partial \xi^{\gamma} \partial \xi^{\beta}} = \{ \begin{matrix} \alpha \\ \beta\gamma \end{matrix} \} \tag{2.17}$$

It can be shown that

$$\{ \begin{matrix} \alpha \\ \gamma\beta \end{matrix} \} = \frac{1}{2} g^{\alpha\delta} \left(\frac{\partial g_{\delta\gamma}}{\partial \xi^{\beta}} + \frac{\partial g_{\delta\beta}}{\partial \xi^{\gamma}} - \frac{\partial g_{\gamma\beta}}{\partial \xi^{\delta}} \right) \tag{2.18}$$

The covariant derivative of a covariant tensor of rank one is given by the expression:

$$U_{\alpha,\beta} = \frac{\partial U_{\alpha}}{\partial \xi^{\beta}} - \{ \begin{matrix} \gamma \\ \alpha\beta \end{matrix} \} U_{\gamma} \tag{2.19}$$

It can be shown that

$$U^{\alpha}_{,\alpha} = \frac{1}{\sqrt{g}} \frac{\partial \sqrt{g} U^{\alpha}}{\partial \xi^{\alpha}} \tag{2.20}$$

The covariant derivative of a contravariant tensor of rank two is defined as follows:

$$T^{\alpha\beta}_{,\gamma} = \frac{\partial T^{\alpha\beta}}{\partial \xi^{\gamma}} + \{ \begin{matrix} \alpha \\ \delta\gamma \end{matrix} \} T^{\delta\beta} + \{ \begin{matrix} \beta \\ \delta\gamma \end{matrix} \} T^{\alpha\delta} \tag{2.21}$$

It can be shown that

$$T^{\alpha\beta}_{,\beta} = \frac{1}{\sqrt{g}} \frac{\delta \sqrt{g} T^{\alpha\beta}}{\partial \xi^{\beta}} + \{ \begin{matrix} \alpha \\ \gamma\beta \end{matrix} \} T^{\gamma\beta} \tag{2.22}$$

and

$$g^{\alpha\beta}_{,\beta} = 0 \tag{2.23}$$

The covariant derivative of a scalar density (i.e. a relative scalar of weight 1) is defined as

$$\rho_{,\alpha} = \frac{\partial \rho}{\partial \xi^{\alpha}} - \rho \{ \begin{matrix} \beta \\ \beta\alpha \end{matrix} \} \tag{2.24}$$

It can be shown that

$$\rho_{,\alpha} = \sqrt{g} \frac{\partial \rho / \sqrt{g}}{\partial \xi^{\alpha}} \tag{2.25}$$

Hence

$$\sqrt{g}_{,\alpha} = 0 \tag{2.26}$$

The volume element The infinitesimal volume element $d\Omega$ in d dimensions is given by

$$d\Omega = \sqrt{g} d\xi^1 d\xi^2 ... d\xi^d \tag{2.27}$$

The divergence theorem in vector and tensor notation Let $V \subset \Omega$, and let S be the boundary of V. The divergence theorem says, in vector notation,

$$\int_\Omega div\ \boldsymbol{u} dV = \oint_\Gamma \boldsymbol{u} \cdot \boldsymbol{d\Gamma} \tag{2.28}$$

Here $\boldsymbol{d\Gamma}$ stands for the vector $\boldsymbol{n} d\Gamma$, with n the outward unit normal on Γ, and $d\Gamma$ the (physical) surface element. In tensor notation the divergence theorem is given by

$$\int_\Omega U^\alpha_{,\alpha} d\Omega = \oint_\Gamma U^\alpha d\Gamma_\alpha \tag{2.29}$$

For a derivation and further references see van Kan et al (1991).

3 Discretization of the metric tensors

Since we assume that the transformation T is not explicitly known, but only implicitly by the mapping of the co-ordinates of the vertices, it is necessary to discretize the geometrical quantities mentioned in section 2. In the ISNaS incompressible code there are several ways of computing these quantities. The choice of which of these methods is used is defined by the input parameter geotype. In this section we shall treat the various possibilities as function of geotype. We distinguish between 2D and 3D.

3.1 2D-case

Geotype = 1

2D The following quantities are computed and stored in all points of the grid, i.e. the vertices, the centroids and the midside points:

$$\boldsymbol{a}_{(\alpha)}, g_{\alpha\beta}, \sqrt{g}, \{ {\alpha \atop \beta\gamma} \}$$

The quantities are computed in the following way:

Consider the p-cell with local numbering as shown in Figure 3.1. First $\boldsymbol{a}_{(1)}$ is computed in

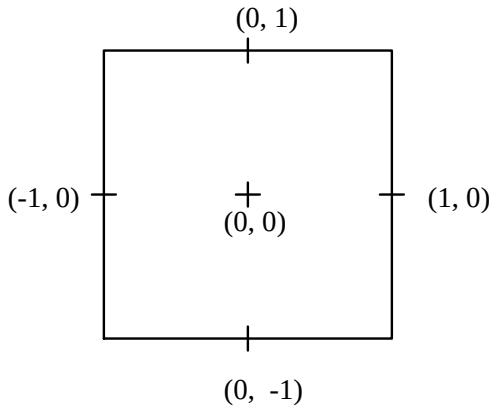


Figure 3.1: local numbering in P-cell

$(0, \pm 1)$ and $\boldsymbol{a}_{(2)}$ in $(\pm 1, 0)$ by:

$$\boldsymbol{a}_{(1)}(0, \pm 1) = \boldsymbol{x}(1, \pm 1) - \boldsymbol{x}(-1, \pm 1) \tag{3.1}$$

$$\boldsymbol{a}_{(2)}(\pm 1, 0) = \boldsymbol{x}(\pm 1, 1) - \boldsymbol{x}(\pm 1, -1) \tag{3.2}$$

Next $\boldsymbol{a}_{(1)}$ and $\boldsymbol{a}_{(2)}$ are computed in all points where they are not available by linear or bilinear interpolation, using the fewest number of interpolation points.

Hence:

$$\boldsymbol{a}_{(1)}(0,0) = \frac{1}{2}\{\boldsymbol{a}_{(1)}(0,1) + \boldsymbol{a}_{(1)}(0,-1)\} \tag{3.3}$$

$$\boldsymbol{a}_{(2)}(0,0) = \frac{1}{2}\{\boldsymbol{a}_{(2)}(1,0) + \boldsymbol{a}_{(2)}(-1,0)\} \tag{3.4}$$

$$\boldsymbol{a}_{(1)}(-1,0) = \frac{1}{4}\{\boldsymbol{a}_{(1)}(0,1) + \boldsymbol{a}_{(1)}(0,-1) + \boldsymbol{a}_1(-2,1) + \boldsymbol{a}_1(-2,-1)\} \tag{3.5}$$

$$\boldsymbol{a}_{(2)}(0,-1) = \frac{1}{4}\{\boldsymbol{a}_{(2)}(1,0) + \boldsymbol{a}_{(2)}(-1,0) + \boldsymbol{a}_2(-1,-2) + \boldsymbol{a}_2(1,-2)\} \tag{3.6}$$

$$\boldsymbol{a}_{(1)}(-1,-1) = \frac{1}{2}\{\boldsymbol{a}_{(1)}(-1,0) + \boldsymbol{a}_{(1)}(-1,-2)\} \tag{3.7}$$

$$\boldsymbol{a}_{(2)}(-1,-1) = \frac{1}{2}\{\boldsymbol{a}_{(2)}(0,-1) + \boldsymbol{a}_{(2)}(-2,-1)\} \tag{3.8}$$

etc.

From $\boldsymbol{a}_{(1)}$ and $\boldsymbol{a}_{(2)}$ we compute the $g_{\alpha\beta}$ in centroid by

$$g_{\alpha\beta} = \boldsymbol{a}_{(\alpha)} \cdot \boldsymbol{a}_{(\beta)} \tag{3.9}$$

and $g_{\alpha\beta}$ in all other points is computed by linear or bilinear interpolation from these centroid points.
For example:

$$g_{\alpha\beta}(-1,0) = \frac{1}{2}\{g_{\alpha\beta}(-2,0) + g_{\alpha\beta}(0,0)\} \tag{3.10}$$

$$g_{\alpha\beta}(-1,-1) = \frac{1}{4}\{g_{\alpha\beta}(0,0) + g_{\alpha\beta}(-2,0) + g_{\alpha\beta}(0,-2) + g_{\alpha\beta}(-2,-2)\} \tag{3.11}$$

etc.
Next $\sqrt{g}$ is computed in all points using the values of $g_{\alpha\beta}$ just computing by

$$\sqrt{g} = \sqrt{|det\ (g_{\alpha\beta})|} \tag{3.12}$$

To compute $\{\begin{smallmatrix}\alpha\\ \beta\gamma\end{smallmatrix}\}$ formula (2.18) is applied in all points of the elements.

So:

$$\begin{aligned}\{\begin{smallmatrix}1\\ 11\end{smallmatrix}\} &= \frac{1}{2}g^{1\delta}(\frac{\partial g_{\delta 1}}{\partial \xi^1} + \frac{\partial g_{\delta 1}}{\partial \xi^1} - \frac{\partial g_{11}}{\partial \xi^\delta}) \\ &= \frac{1}{2}g^{11}\frac{\partial g_{11}}{\partial \xi^1} + g^{12}(\frac{\partial g_{21}}{\partial \xi^1} - \frac{1}{2}\frac{\partial g_{11}}{\partial \xi^2}) \\ &= \frac{1}{2}\frac{g_{22}}{g}\frac{\partial g_{11}}{\partial \xi^1} - \frac{g_{12}}{g}(\frac{\partial g_{21}}{\partial \xi^1} - \frac{1}{2}\frac{g_{11}}{\partial \xi^2})\end{aligned} \tag{3.13}$$

$$
\begin{aligned}
\left\{ {1 \atop 12} \right\} &= \frac{1}{2} g^{1\delta} \left(\frac{\partial g_{\delta 1}}{\partial \xi^2} + \frac{\partial g_{\delta 2}}{\partial \xi^1} - \frac{\partial g_{12}}{\partial \xi^\delta} \right) \\
&= \frac{1}{2} g^{11} \left(\frac{\partial g_{11}}{\partial \xi^2} + \frac{\partial g_{12}}{\partial \xi^1} - \frac{\partial g_{12}}{\partial \xi^1} \right) + \frac{1}{2} g^{12} \left(\frac{\partial g_{21}}{\partial \xi^2} + \frac{\partial g_{22}}{\partial \xi^1} - \frac{\partial g_{12}}{\partial \xi^2} \right) \\
&= \frac{1}{2} \frac{g_{22}}{g} \frac{\partial g_{11}}{\partial \xi^2} - \frac{1}{2} \frac{g_{12}}{g} \frac{\partial g_{22}}{\partial \xi^1} \qquad (3.14)
\end{aligned}
$$

$$
\begin{aligned}
\left\{ {1 \atop 22} \right\} &= \frac{1}{2} g^{11} \left(\frac{\partial g_{12}}{\partial \xi^2} + \frac{\partial g_{21}}{\partial \xi^2} - \frac{\partial g_{22}}{\partial \xi^1} \right) + \frac{1}{2} g^{12} \left(\frac{\partial g_{22}}{\partial \xi^2} + \frac{\partial g_{22}}{\partial \xi^2} - \frac{\partial g_{22}}{\partial \xi^2} \right) \\
&= \frac{g_{22}}{g} \left(\frac{\partial g_{12}}{\partial \xi^2} - \frac{1}{2} \frac{\partial g_{22}}{\partial \xi^1} \right) - \frac{1}{2} \frac{g_{12}}{g} \frac{\partial g_{22}}{\partial \xi^2} \qquad (3.15)
\end{aligned}
$$

$$
\begin{aligned}
\left\{ {2 \atop 12} \right\} &= \frac{1}{2} g^{21} \left(\frac{\partial g_{11}}{\partial \xi^2} + \frac{\partial g_{12}}{\partial \xi^1} - \frac{\partial g_{12}}{\partial \xi^1} \right) + \frac{1}{2} g^{22} \left(\frac{\partial g_{21}}{\partial \xi^2} + \frac{\partial g_{22}}{\partial \xi^1} - \frac{\partial g_{12}}{\partial \xi^2} \right) \\
&= -\frac{1}{2} \frac{g_{21}}{g} \frac{\partial g_{11}}{\partial \xi^2} + \frac{1}{2} \frac{g_{11}}{g} \frac{\partial g_{22}}{\partial \xi^1} \qquad (3.16)
\end{aligned}
$$

$$
\begin{aligned}
\left\{ {2 \atop 11} \right\} &= \frac{1}{2} g^{21} \left(\frac{\partial g_{11}}{\partial \xi^1} + \frac{\partial g_{11}}{\partial \xi^1} - \frac{\partial g_{11}}{\partial \xi^1} \right) + \frac{1}{2} g^{22} \left(\frac{\partial g_{21}}{\partial \xi^1} + \frac{\partial g_{21}}{\partial \xi^1} - \frac{\partial g_{11}}{\partial \xi^2} \right) \\
&= -\frac{1}{2} \frac{g_{21}}{g} \frac{\partial g_{11}}{\partial \xi^1} + \frac{g_{11}}{g} \left(\frac{\partial g_{21}}{\partial \xi^1} - \frac{1}{2} \frac{\partial g_{11}}{\partial \xi^2} \right) \qquad (3.17)
\end{aligned}
$$

$$
\begin{aligned}
\left\{ {2 \atop 22} \right\} &= \frac{1}{2} g^{21} \left(\frac{\partial g_{12}}{\partial \xi^2} + \frac{\partial g_{12}}{\partial \xi^2} - \frac{\partial g_{22}}{\partial \xi^1} \right) + \frac{1}{2} g^{22} \left(\frac{\partial g_{22}}{\partial \xi^2} + \frac{\partial g_{22}}{\partial \xi^2} - \frac{\partial g_{22}}{\partial \xi^2} \right) \\
&= -\frac{g_{21}}{g} \left(\frac{\partial g_{12}}{\partial \xi^2} - \frac{1}{2} \frac{\partial g_{22}}{\partial \xi^1} \right) + \frac{1}{2} \frac{g_{11}}{g} \frac{\partial g_{22}}{\partial \xi^2} \qquad (3.18)
\end{aligned}
$$

In these expressions we have used that $g^{\alpha\beta}$ is the inverse of $g_{\alpha\beta}$, so

$$
\begin{bmatrix} g^{11} & g^{12} \\ g^{12} & g^{22} \end{bmatrix} = \frac{1}{g} \begin{bmatrix} g_{22} & -g_{12} \\ -g_{12} & g_{11} \end{bmatrix} \qquad (3.19)
$$

The derivatives $\partial \cdot / \partial \xi^\alpha$ are approximated by central differences using two neighbouring points.

Geotype = 2

The same quantities as for geotype = 1 are computed and stored in the same points. However, there are some minor differences, which result in a more accurate discretization of the differential equations.

The base vector $\boldsymbol{a}_{(\alpha)}$ are computed in exactly the same way as for geotype = 1, i.e. formulae (3.1) and (3.2) are applied.

The Jacobian $\sqrt{g}$ of the transformation in all points is computed from the base vectors in those points, using the expression:

$$\sqrt{g}_{(\xi,\eta)} = |a^1_{(1)}a^2_{(2)} - a^2_{(1)}a^1_{(2)}|_{(\xi,\eta)} \tag{3.20}$$

for all points.

In the same way $g_{\alpha\beta}$ is computed by (3.12) in all points.

With respect to the Christoffel symbols $\{ \begin{matrix} \alpha \\ \beta\gamma \end{matrix} \}$ not only the interpolation is canceled but also formula (2.18) is replaced by formula (2.17). The base vectors $\boldsymbol{a}^{(\alpha)}$ are computed by inversion of $\boldsymbol{a}_{(\alpha)}$, i.e.

$$\boldsymbol{a}^{(1)} = \frac{1}{\sqrt{g}}(\boldsymbol{a}^2_{(2)},\ -\boldsymbol{a}^1_{(2)}), \quad \boldsymbol{a}^{(2)} = \frac{1}{\sqrt{g}}(-\boldsymbol{a}^2_{(1)},\ a^1_{(1)}) \tag{3.21}$$

The derivatives are again computed by central differences based on 2 neighbouring points.

The formulae derived for the geometrical quantities can all be computed for the internal region. However, at the boundary some extra kind of extrapolation is necessary. In the present version of the flow solver the extrapolation has been taken care of by the introduction of virtual cells and hence virtual co-ordinates. See Figure 3.2.

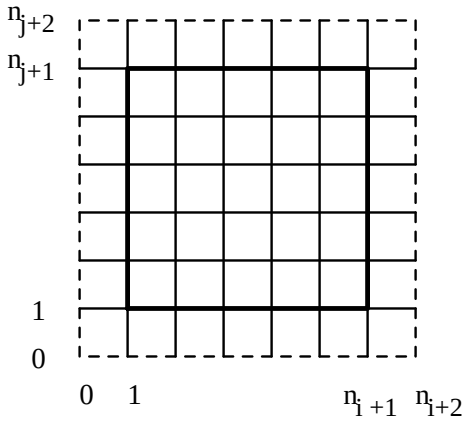


Figure 3.2: virtual cells surrounding the boundary of the region (computational space)

The co-ordinates of the virtual boundary are computed by linear extrapolation, for example

$$\boldsymbol{x}_{i,0} = 2\boldsymbol{x}_{i,1} - \boldsymbol{x}_{i,2} \tag{3.22}$$

The co-ordinates in the 4 vertex points are computed by taking the mean value of the linear extrapolation of the co-coordinates along the two virtual boundaries corresponding to this vertex.

For example

$$\boldsymbol{x}_{0,0} = \frac{1}{2}[(2x_{1,0} - x_{2,0}) + (2x_{0,1} - x_{0,2})] \tag{3.23}$$

The base vectors $\boldsymbol{a}_{(\alpha)}$ are computed in the centroids of all virtual cells and in the midside points of these cells. The metric tensor $g_{\alpha\beta}$ is computed in all non-virtual points as well as all virtual points that are not situated at the outer boundary of the virtual alls. The Christoffel symbols are only computed at the non-virtual points.

3.2 3D-case

The implementation here is only done for geotype = 2.

The covariant base vector $\boldsymbol{a}_{(\alpha)}$ is computed in the centre of the edges of a P-cell parallel to the ξ^{α}-axis, see Figure 3.3.

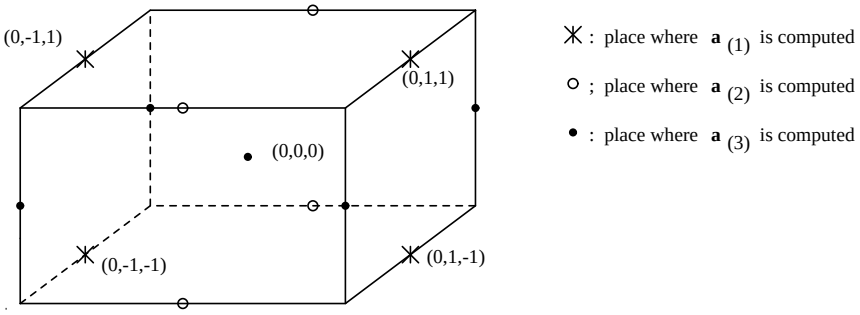


Figure 3.3: P-cell with local numbering and the places where $\mathbf{a}_{(1)}$, $\mathbf{a}_{(2)}$ and $\mathbf{a}_{(3)}$ are computed.

The $\boldsymbol{a}_{(1)}, \boldsymbol{a}_{(2)}$ and $\boldsymbol{a}_{(3)}$ are computed in the following way:

$$\boldsymbol{a}_{(1)}(0,i,j) = \boldsymbol{x}(1,i,j) - \boldsymbol{x}(-1,i,j) \tag{3.24}$$

$$\boldsymbol{a}_{(2)}(i,0,j) = \boldsymbol{x}(i,1,j) - \boldsymbol{x}(i,-1,j) \tag{3.25}$$

$$\boldsymbol{a}_{(3)}(i,j,0) = \boldsymbol{x}(i,j,1) - \boldsymbol{x}(i,j,-1) \tag{3.26}$$

where $i,j, \in \{-1,1\}$.

Just as the 2D-case we compute $\boldsymbol{a}_{(1)}, \boldsymbol{a}_{(2)}$ and $\boldsymbol{a}_{(3)}$ in all grid points where they are not available by a linear interpolation, using the fewest number of interpolation points.
So:

$$\boldsymbol{a}_{(1)}(-1,-1,-1) = \frac{1}{2}\{\boldsymbol{a}_{(1)}(-2,-1,-1) + \boldsymbol{a}_{(1)}(0,-1,-1)\} \tag{3.27}$$

$$\boldsymbol{a}_{(1)}(-1,0,-1) = \frac{1}{4}\{\boldsymbol{a}_{(1)}(-2,-1,-1) + \boldsymbol{a}_{(1)}(0,-1,-1) + \boldsymbol{a}_{(1)}(-2,1,-1) + \boldsymbol{a}_{(1)}(0,1,-1)\} \tag{3.28}$$

$$\boldsymbol{a}_{(1)}(-1,-1,0) = \frac{1}{4}\{\boldsymbol{a}_{(1)}(-2,-1,-1) + \boldsymbol{a}_{(1)}(0,-1,-1) + \boldsymbol{a}_{(1)}(-2,-1,1) + \boldsymbol{a}_{(1)}(0,-1,1)\} \tag{3.29}$$

$$\boldsymbol{a}_{(1)}(-1,0,0) = \frac{1}{8}\{\boldsymbol{a}_{(1)}(-2,-1,-1) + \boldsymbol{a}_{(1)}(0,-1,-1) + \boldsymbol{a}_{(1)}(-2,1,-1) + \boldsymbol{a}_{(1)}(0,1,-1) + \boldsymbol{a}_{(1)}(-2,-1,1) + \boldsymbol{a}_{(1)}(0,-1,1) + \boldsymbol{a}_{(1)}(-2,1,1) + \boldsymbol{a}_{(1)}(0,1,1)\} \tag{3.30}$$

etc.

The geometrical quantity $\sqrt{g}$ is computed for all gridpoints from the covariant base vectors; using the expression:

$$\begin{aligned}\sqrt{g}_{(i,j,k)} &= (a^1_{(1)}a^2_{(2)}a^3_{(3)} - a^1_{(1)}a^3_{(2)}a^2_{(3)} + \\ &\quad a^2_{(1)}a^3_{(2)}a^1_{(3)} - a^2_{(1)}a^1_{(2)}a^3_{(3)} + \\ &\quad a^3_{(1)}a^1_{(2)}a^2_{(3)} - a^3_{(1)}a^2_{(2)}a^1_{(3)})_{(i,j,k)}. \end{aligned} \tag{3.31}$$

The geometric tensors $g_{\alpha\beta}$ and $g^{\alpha\beta}$ are computed for all gridpoints by:

$$g_{\alpha\beta}(i,j,k) = (\boldsymbol{a}_{(\alpha)} \cdot \boldsymbol{a}_{(\beta)})_{(i,j,k)} \tag{3.32}$$

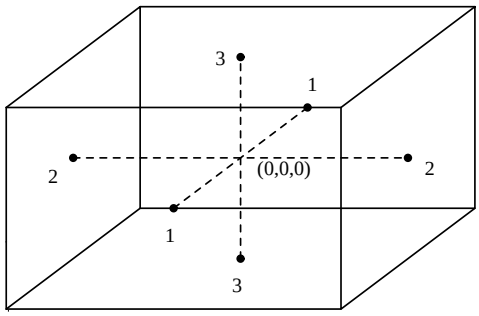
and

$$g^{\alpha\beta}(i,j,k) = (\frac{(-1)^{\alpha+\beta} det\ (G_{\alpha\beta})}{g})_{(i,j,k)} \tag{3.33}$$

where

$$G_{\alpha\beta} = \begin{matrix}\text{minor} \\ \alpha\beta\end{matrix} \begin{bmatrix} g_{11} & g_{12} & g_{13} \\ g_{21} & g_{22} & g_{23} \\ g_{31} & g_{32} & g_{33} \end{bmatrix} . \tag{3.34}$$

Christoffel symbols are computed by formula (2.17) for the centers of the faces of a p-cell, $\{ \begin{smallmatrix}1\\11\end{smallmatrix} \}, \{ \begin{smallmatrix}1\\12\end{smallmatrix} \}, ..., \{ \begin{smallmatrix}1\\33\end{smallmatrix} \}$ for the front and back face $\{ \begin{smallmatrix}2\\11\end{smallmatrix} \}, \{ \begin{smallmatrix}2\\12\end{smallmatrix} \}, ..., \{ \begin{smallmatrix}2\\33\end{smallmatrix} \}$ for the right and left face and $\{ \begin{smallmatrix}3\\11\end{smallmatrix} \}, ..., \{ \begin{smallmatrix}3\\33\end{smallmatrix} \}$ for the upper and lower face, see Figure 3.4 [1]



1: $\{ \begin{smallmatrix}1\\11\end{smallmatrix} \}, \{ \begin{smallmatrix}1\\12\end{smallmatrix} \}, ..., \{ \begin{smallmatrix}1\\33\end{smallmatrix} \}$

2: $\{ \begin{smallmatrix}2\\11\end{smallmatrix} \}, \{ \begin{smallmatrix}2\\12\end{smallmatrix} \}, ..., \{ \begin{smallmatrix}2\\33\end{smallmatrix} \}$

3: $\{ \begin{smallmatrix}3\\11\end{smallmatrix} \}, \{ \begin{smallmatrix}3\\12\end{smallmatrix} \}, ..., \{ \begin{smallmatrix}3\\33\end{smallmatrix} \}$

Figure 3.4: Places in the p-cell where the Christoffel symbols are computed.

[1] In 3D we don't need the Christoffel symbols in all grid points, because we use another formula for the deviatoric stress tensor ((4.16) instead of (4.2) with (2.16)).

The contravariant base vectors in formula (2.17) are computed by (2.5).
Just as in the 2D-case we introduce virtual cells to compute the geometrical quantities at the boundaries. See Figure 3.5.

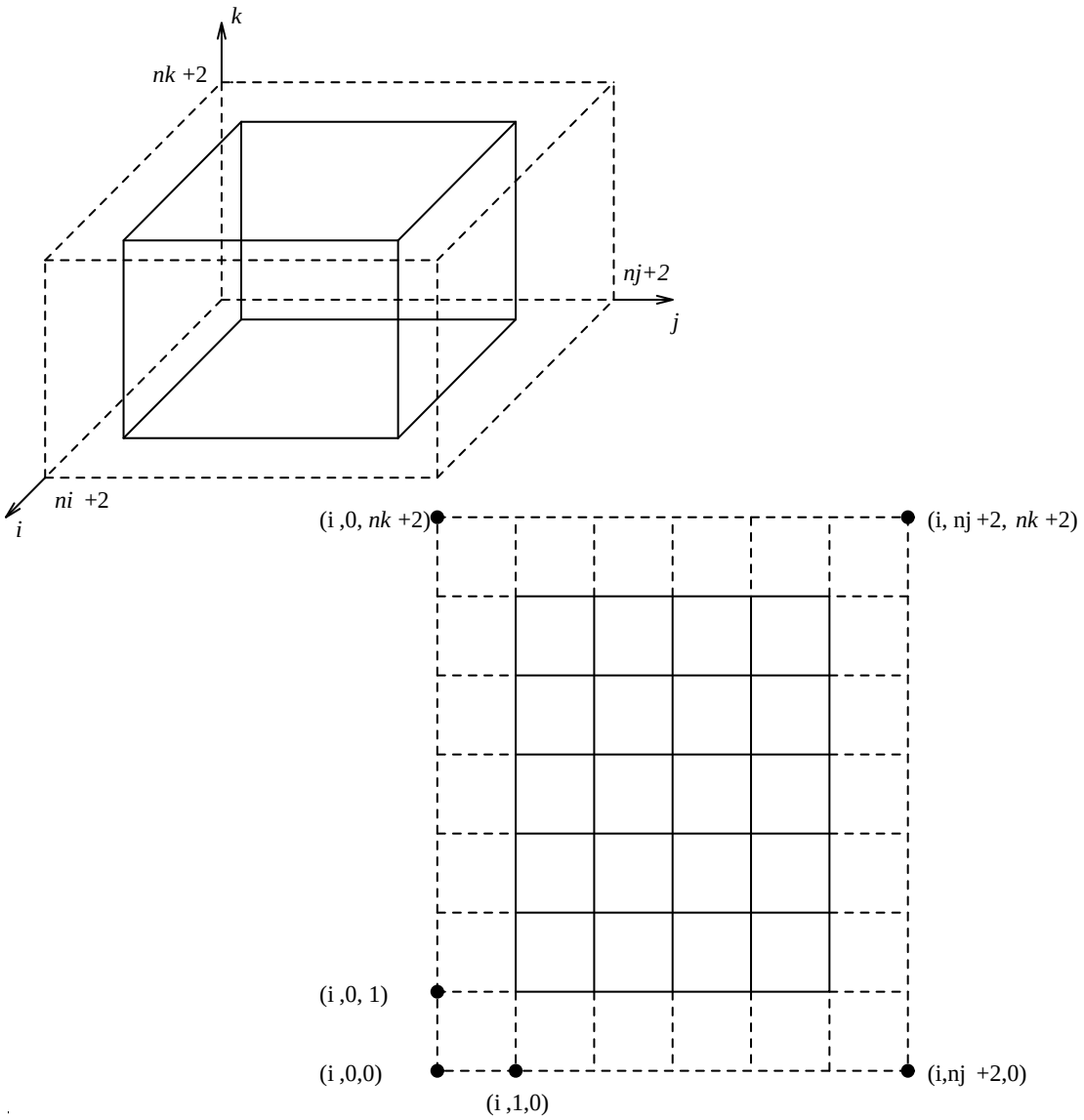


Figure 3.5: The virtual cells surrounding the boundary with a cross-section of the cube.

The co-ordinates of the virtual boundary are computed by a linear extrapolation, for example face $i = 0$

$$\boldsymbol{x}_{0,j,k} = 2\boldsymbol{x}_{1,j,k} - \boldsymbol{x}_{2,j,k} \ , \tag{3.35}$$

edge $i = 0$ and $j = 0$

$$\boldsymbol{x}_{0,0,k} = \frac{1}{2}[(2\boldsymbol{x}_{1,0,k} - \boldsymbol{x}_{2,0,k}) + (2\boldsymbol{x}_{0,1,k} - \boldsymbol{x}_{0,2,k})] \,, \tag{3.36}$$

vertex $i = 0$, $j = 0$ and $k = 0$

$$\boldsymbol{x}_{0,0,0} = \frac{1}{3}[(2\boldsymbol{x}_{1,0,0} - \boldsymbol{x}_{2,0,0}) + (2\boldsymbol{x}_{0,1,0} - \boldsymbol{x}_{0,2,0}) + (2\boldsymbol{x}_{0,0,1} - \boldsymbol{x}_{0,0,2})] \,. \tag{3.37}$$

4 Space discretization of the differential equations

In this section we describe the space discretization of the momentum equations and convection-diffusion equations in the inner region. Discretizations due to the boundary conditions are treated in section 5.

4.1 The momentum equations and continuity equation

The momentum equations in general co-ordinates read (see van Kan et al 1991; formula 5.2)

$$\frac{\partial}{\partial t}(\rho U^\alpha) + (\rho U^\alpha U^\beta)_{,\beta} + (g^{\alpha\beta}p)_{,\beta} - \tau^{\alpha\beta}_{,\beta} = \rho f^\alpha \tag{4.1}$$

with $\tau^{\alpha\beta}$ the deviatoric stress tensor given by

$$\tau^{\alpha\beta} = (\mu + \mu_t)(g^{\alpha\gamma}U^\beta_{,\gamma} + g^{\gamma\beta}U^\alpha_{,\gamma}). \tag{4.2}$$

Here U^α is the contravariant velocity, ρ the density, p the pressure, ρf^α some external force per unit volume, μ dynamic viscosity and μ_t denotes eddy-viscosity, which has to be specified. This specification is accomplished by a turbulence model. This will be presented in section 4.3.

In the present version all coefficients may depend on space, time and previous computed solutions. However, with respect to the density a correct implementation is only guaranteed for ρ is constant.

The continuity equation reads (see van Kan et al 1991; formula 5.1):

$$U^\alpha_{,\alpha} = 0 \tag{4.3}$$

As unknowns the fluxes $V^\alpha = \sqrt{g}U^\alpha$ are used.

Equations (4.2) and (4.3) are discretized by a finite volume method.

We distinguish between the 2D and the 3D case.

4.1.1 2D-case

The discretization of the continuity equation is straightforward. We use a staggered grid arrangement as plotted in Figure 4.1.
The continuity equation is integrated over a so-called pressure-cell. This yields:

$$V^1|^{(1,0)}_{(-1,0)} + V^2|^{(0,1)}_{(0,-1)} = 0, \tag{4.4}$$

where the local numbering of Figure 3.1 is used.

With respect to the discretization of the momentum equations we distinguish between the time-derivative, the convection term, the pressure gradient, the deviatoric stress tensor and the right-hand-side term.

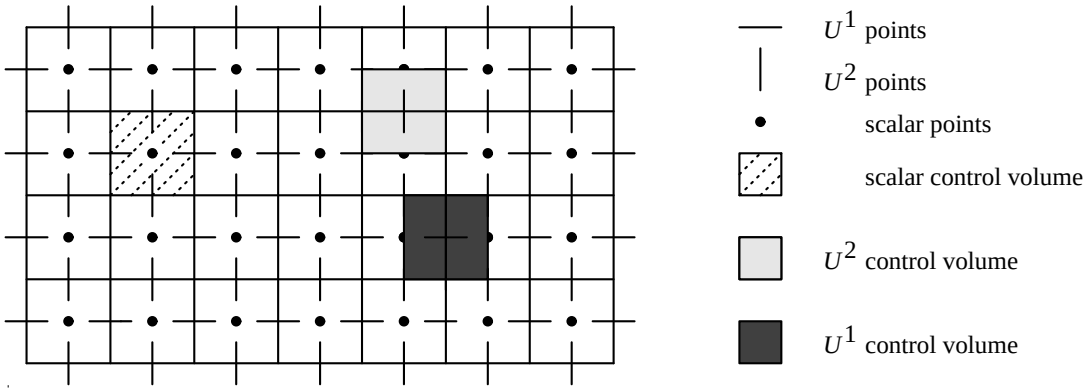


Figure 4.1: Arrangement of the unknowns for a staggered grid

The discretization of the time-derivative is given by formula (5.35) of van Kan et al (1991)):

$$\frac{\partial}{\partial t}(\rho V^\alpha|_{(0,0)}), \tag{4.5}$$

where (0, 0) is the center of a V^α-cell.
The discretization of the right-hand-side term is given by formula (5.34) of that report:

$$\rho f^\alpha \sqrt{g}|_{(0,0)} \tag{4.6}$$

In order to solve the so-called no flow problem, the discretization of the right-hand side has slightly been improved by taking

$$\rho\sqrt{g}(a_1^{(\alpha)} f_1 + a_2^{(\alpha)} f_2)|_{(0,0)}$$

See Segal (1993).
The discretization of the convective terms requires a linearization. At this moment only one type of linearization is available, the Newton linearization given by

$$V^\alpha V^\beta \approx V^\alpha \bar{V}^\beta + \bar{V}^\alpha V^\beta - \bar{V}^\alpha \bar{V}^\beta \tag{4.7}$$

where V^α is taken at the new time level and $\bar{V}^\alpha$ at the preceding one.

Apart from the linearization, the discretization of the convective terms is given by formulae (5.8) and (5.9) of van Kan et al (1991):

V^1-cell:

$$\frac{\rho}{\sqrt{g}}(V^1)^2|_{(-1,0)}^{(1,0)} + \frac{\rho}{\sqrt{g}}V^1V^2|_{(0,-1)}^{(0,1)} + \frac{\rho}{\sqrt{g}}\{{}^{1}_{\gamma\beta}\}V^\gamma V^\beta|_{(0,0)} \tag{4.8}$$

V^2-cell:

$$\frac{\rho}{\sqrt{g}}V^2V^1|_{(-1,0)}^{(1,0)} + \frac{\rho}{\sqrt{g}}(V^2)^2|_{(0,-1)}^{(0,1)} + \frac{\rho}{\sqrt{g}}\{{}^{2}_{\gamma\beta}\}V^\gamma V^\beta|_{(0,0)} \tag{4.9}$$

Unknowns not present at points where they are required, are computed by linear interpolation using the least number of neighbouring points possible.

The discretization of the deviatoric stress tensor is carried out according to formulae (5.23) to (5.25) in van Kan et al (1991):

V^1-cell:

$$-\sqrt{g}\mu(g^{11}U^1_{,1}+g^{12}U^1_{,2})|^{(1,0)}_{(-1,0)}-\sqrt{g}\mu(g^{11}U^2_{,1}+g^{22}U^1_{,2})|^{(0,1)}_{(0,-1)}-\{\,{}^{1}_{\gamma\beta}\}\tau^{\gamma\beta}\sqrt{g}|_{(0,0)} \quad (4.10)$$

V^2-cell:

$$-\sqrt{g}\mu(g^{11}U^2_{,1}+g^{22}U^1_{,2})|^{(1,0)}_{(-1,0)}-\sqrt{g}2\mu(g^{12}U^2_{,1}+g^{22}U^2_{,2})|^{(0,1)}_{(0,-1)}-\{\,{}^{2}_{\gamma\beta}\}\tau^{\gamma\beta}\sqrt{g}|_{(0,0)} \quad (4.11)$$

with $U^\alpha_{,\beta}$ given by formula (2.22) and $\tau^{\alpha\beta}$ by formula (4.2).

The derivatives $\frac{\partial U^\alpha}{\partial \xi^\beta}$ are computed by central differences, hence

$$\frac{\partial U^\alpha}{\partial \xi^1}|_{(\xi,\eta)} = U^\alpha|_{(\xi+1,\eta)} - U^\alpha|_{(\xi-1,\eta)} \quad (4.12)$$

$$\frac{\partial U^\alpha}{\partial \xi^2}|_{(\xi,\eta)} = U^\alpha|_{(\xi,\eta+1)} - U^\alpha|_{(\xi,\eta-1)}. \quad (4.13)$$

where for (ξ,η) the local numbering is used.
The same type of interpolation is used as for the convective terms. U^α is replaced by $V^\alpha/\sqrt{g}$ in the points where U^α is evaluated, although a better method might be to replace $\sqrt{g}U^\alpha_{,\beta}$ by $V^\alpha_{,\beta}$, since $\sqrt{g}_{,\beta}=0$.

Finally, the discretization of the pressure gradient is carried by formula (3.14) in Segal and Kassels (1991):

$$(p_{(1,0)}-p_{(-1,0)})(g^{\alpha 1}\sqrt{g})_{(0,0)}+(p_{(0,1)}-p_{(0,-1)})(g^{\alpha 2}\sqrt{g})_{(0,0)} \quad (4.14)$$

4.1.2 3D-case

First we show in Figure 4.2 the staggered grid arrangements of the unknowns together with the control volumes.

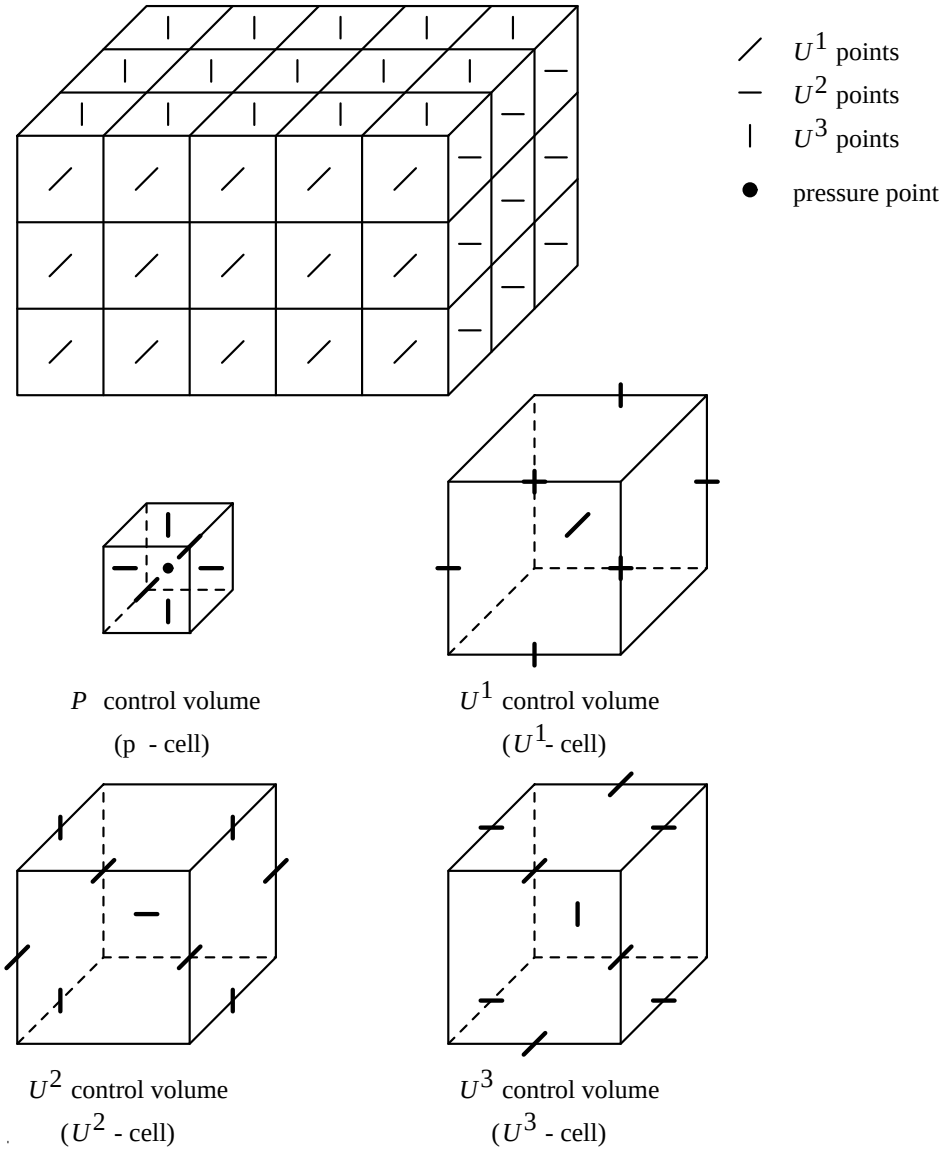


Figure 4.2: Arrangement of the unknowns for a staggered grid

If we integrate the continuity equation over a p-cell get

$$V^1|_{(-1,0,0)}^{(1,0,0)} + V^2|_{(0,-1,0)}^{(0,1,0)} + V^3|_{(0,0,-1)}^{(0,0,1)} = 0 \ , \tag{4.15}$$

this is a simple extension of formula (4.4).

Just as the 2D-case we splitup the momentum equation. The formulae (4.5) - (4.14) are all most the same in 3D. The main difference between 2D and 3D follows from the fact that not a discretization of formula (4.2) is used but from:

$$\tau^{\alpha\beta} = \mu(g^{\alpha\gamma}\frac{\partial U^\beta}{\partial \xi^\gamma} + g^{\gamma\beta}\frac{\partial U^\alpha}{\partial \xi^\gamma} - \frac{\partial g^{\alpha\beta}}{\partial \xi^\gamma}U^\gamma) \ . \tag{4.16}$$

This formula (4.16) follows directly from equation (4.2), (2.16), (2.21) and $g^{\alpha\beta}_{,\gamma} = 0$.
The discretization of the time derivative is given by:

$$\frac{\partial}{\partial t}((\rho V^\alpha)|_{(0,0,0)}) \ , \tag{4.17}$$

where $(0,0,0)$ is the centre of a V^α-cell.
The discretization of the right-hand-side term is given by

$$(\rho f^\alpha \sqrt{g})|_{(0,0,0)} \ . \tag{4.18}$$

The discretization of the convective terms is given by a straight forward extension of formulae (5.7) and (5.8a)-(5.8b) of Van Kan et al (1991):

$$\begin{aligned} V^\alpha - \text{ cell } : \quad & \\ & \frac{\rho}{\sqrt{g}}V^\alpha V^1|_{(-1,0,0)}^{(1,0,0)} + \frac{\rho}{\sqrt{g}}V^\alpha V^2|_{(0,-1,0)}^{(0,1,0)} + \frac{\rho}{\sqrt{g}}V^\alpha V^3|_{(0,0,-1)}^{(0,0,1)} + \\ & \frac{\rho}{\sqrt{g}}\{\begin{smallmatrix}\alpha\\ \gamma\beta\end{smallmatrix}\}V^\gamma V^\beta|_{(0,0,0)} \quad \text{for} \quad \alpha \in \{1,2,3\} \ . \end{aligned} \tag{4.19}$$

For all non-linear terms in (4.19) we used the Newton linearization given by

$$V^\alpha V^\beta \approx V^\alpha \bar{V}^\beta + \bar{V}^\alpha V^\beta - \bar{V}^\alpha \bar{V}^\beta \tag{4.20}$$

where V^α is taken at the new time level and $\bar{V}^\alpha$ at the old level.
Unknowns not present at points where they are required, are computed by linear interpolation using the fewest number of interpolation points.

The discretization of the deviatoric stress tensor is carried out according to:

$$\begin{aligned} V^\alpha - \text{ cell } : \quad & \\ & -\sqrt{g}\tau^{\alpha 1}|_{(-1,0,0)}^{(1,0,0)} - \sqrt{g}\tau^{\alpha 2}|_{(0,-1,0)}^{(0,1,0)} - \sqrt{g}\tau^{\alpha 3}|_{(0,0,-1)}^{(0,0,1)} \\ & -(\{\begin{smallmatrix}\alpha\\ \gamma\beta\end{smallmatrix}\}\tau^{\gamma\beta}\sqrt{g})|_{(0,0,0)} \ , \end{aligned} \tag{4.21}$$

with $\tau^{\alpha\beta}$ given by formula (4.16).
The derivatives $\frac{\partial U^\alpha}{\partial \xi^\beta}$ in (4.16) are computed by central differences, thus:

$$\frac{\partial U^\alpha}{\partial \xi^\beta}|_{(i,j,k)} = U^\alpha|_{(i,j,k)+\Delta_\beta} - U^\alpha|_{(i,j,k)-\Delta_\beta} \tag{4.22}$$

where

$$\Delta_\beta = (\frac{1}{2}|\beta-2|\ |\beta-3|,\ |\beta-1|\ |\beta-3|,\ \frac{1}{2}|\beta-1|\ |\beta-2|)\ . \tag{4.23}$$

U^α is replaced by $V^\alpha/\sqrt{g}$. We make here also the remark that it might be better to replace $\sqrt{g}U^\alpha_{,\beta}$ by $V^\alpha_{,\beta}$. So use

$$\sqrt{g}\tau^{\alpha\beta} = \mu(g^{\alpha\gamma}\frac{\partial V^\alpha}{\partial \xi^\gamma} + g^{\gamma\beta}\frac{\partial V^\alpha}{\partial \xi^\gamma} - \frac{\partial g^{\alpha\beta}}{\partial \xi^\gamma}V^\gamma) \tag{4.24}$$

instead of formula (4.16) in (4.21).
Finally the discretization of the pressure term is carried out by a generalization of formula (3.14) in Segal and Kassels (1991):

V^α – cell :

$$(g^{\alpha 1}\sqrt{g})|_{(0,0,0)}(p_{(1,0,0)} - p_{(-1,0,0)}) + (\sqrt{g^{\alpha 2}}\sqrt{g})|_{(0,0,0)}(p_{(0,1,0)} - p_{(0,-1,0)}) + (g^{\alpha 3}\sqrt{g})|_{(0,0,0)}(p_{(0,0,1)} - p_{(0,0,-1)})\ . \tag{4.25}$$

4.2 The convection-diffusion equation

The convection-diffusion equation in Cartesian co-ordinates reads:

$$\frac{\partial c^* T}{\partial t} + \boldsymbol{u}\cdot\nabla(c^*T) + div\ (k\nabla T) + DT = f^*, \tag{4.26}$$

with k the matrix $\begin{pmatrix} k_{11} & k_{12} \\ k_{21} & k_{22} \end{pmatrix}$ and $k_{12} = k_{21}$.
Translated into general co-ordinates this equation becomes (see van Kan et al 1991, formula 5.4):

$$\frac{\partial c^* T}{\partial t} + (c^* U^\alpha T)_{,\alpha}) - (K^{\alpha\beta}T_{,\beta})_{,\alpha} + DT = f^*, \tag{4.27}$$

with $K^{\alpha\beta} = a^{(\alpha)}_\gamma a^{(\beta)}_\delta k^{\gamma\delta}$

The discretization of equation (4.27) is given by

$$\sqrt{g_{(0,0)}}\frac{\partial c^* T_{(0,0)}}{\partial t} + c^* V^1 T|^{(1,0)}_{(-1,0)} + c^* V^2 T|^{(0,1)}_{(0,-1)}$$
$$\sqrt{g}k^{1\beta}\frac{\partial T}{\partial \xi^\beta}|^{(1,0)}_{(-1,0)} - \sqrt{g}k^{2\beta}\frac{\partial T}{\partial \xi^\beta}|^{(0,1)}_{(0,-1)} + \sqrt{g}DT|_{(0,0)} = \sqrt{g}f^*|_{(0,0)} \tag{4.28}$$

With respect to interpolations and derivatives the same rules as for the momentum equations are applied.

4.3 Turbulence models

In section 4.1 we have stated that μ_t in (4.2) has to be determine by introducing a turbulence model. In this section we shall discussed two turbulence models.

4.3.1 The $k - L$ model

The standard one-equation model called the $k - L$ model uses a modelled transport equation for turbulent kinetic energy k, which is by definition positive, and an algebraic expression for a characteristic length scale L. The k equation in general co-ordinates reads

$$\frac{\partial \rho k}{\partial t} + (\rho U^\beta k)_{,\beta} - (\frac{\mu_t}{\sigma_k} g^{\beta\gamma} k_{,\gamma})_{,\beta} = R^{\alpha\beta} g_{\alpha\gamma} U^\gamma_{,\beta} - \rho c_d \frac{k^{3/2}}{L} \tag{4.29}$$

where $R^{\alpha\beta}$ is the Reynolds stress tensor

$$R^{\alpha\beta} = \mu_t (g^{\alpha\gamma} U^\beta_{,\gamma} + g^{\gamma\beta} U^\alpha_{,\gamma}) \tag{4.30}$$

Concerning source terms, the first term on the right hand side of (4.29) is the so-called production of turbulent kinetic energy and the last term is the dissipation rate of turbulent energy. Furthermore, c_d and σ_k are empirical constants, which values are 0.08 and 1.0, respectively. These values were proposed by Launder & Spalding, (1972).
It is necessary to find an algebraic expression for L, which depends on the flow geometry. This is an important disadvantage of the $k - L$ model, because, it is not very easy to find an expression for L for an arbitrary flow geometry. However, this model can be used for near-wall treatment, where little empirical information on the length scale distribution is available.
Finally, with k and L we can compute the eddy-viscosity

$$\mu_t = \rho \sqrt{k} L \tag{4.31}$$

4.3.2 The $k - \varepsilon$ model

The $k - \varepsilon$ model which is developed by Jones & Launder (1972) has been generally accepted as the standard turbulence model used in CFD codes. This two-equation model is presented here in full:

$$\mu_t = \rho c_\mu \frac{k^2}{\varepsilon} \tag{4.32}$$

$$\frac{\partial \rho k}{\partial t} + (\rho U^\beta k)_{,\beta} - (\frac{\mu_t}{\sigma_k} g^{\beta\gamma} k_{,\gamma})_{,\beta} = P - \rho\varepsilon \tag{4.33}$$

$$\frac{\partial \rho \varepsilon}{\partial t} + (\rho U^\beta \varepsilon)_{,\beta} - (\frac{\mu_t}{\sigma_\varepsilon} g^{\beta\gamma} \varepsilon_{,\gamma})_{,\beta} = \frac{\varepsilon}{k}(c_{1\varepsilon} P - c_{2\varepsilon} \rho\varepsilon) \tag{4.34}$$

where P is the production of turbulent kinetic energy

$$P = R^{\alpha\beta} g_{\alpha\gamma} U^\gamma_{,\beta} \tag{4.35}$$

c_μ	$c_{1\varepsilon}$	$c_{2\varepsilon}$	σ_k	σ_ε
0.09	1.44	1.92	1.0	1.3

Table 4.1: Constants for $k-\varepsilon$ model

and ε is the dissipation rate of turbulent energy, which is essentially positive. Furthermore, c_μ, $c_{1\varepsilon}$, and $c_{2\varepsilon}$ are empirical constants and σ_k and σ_ε are the turbulence Prandtl/Schmidt numbers for k and ε, respectively. The values of these contents are given in table 4.3.2 and are recommended by Launder & Spalding, (1974). An important remark is that turbulence models introduced above apply only to regions of flow with high Reynolds number. For a derivation and further references, see Zijlema, (1993).

4.3.3 2D implementation of turbulence models

The equations (4.29) or (4.33) and (4.34) can be considered as convection-diffusion equations. Hence, for the implementation of the $k-L$ and $k-\varepsilon$ model the equation (4.27) will be used. The evaluations of the functions c^*, $K^{\alpha\beta}$, D and f^* will be treated in section 6.3.
Before the transport equations have to be solved, the production term (4.35) must be evaluated. The discretization of the production term is carried out at the centre of a scalar cell. Since we use $V^\alpha = \sqrt{g}U^\alpha$ as unknowns, the covariant derivative of the contravariant velocity components must be expressed in terms of flux components. Thus

$$\sqrt{g}U^\alpha_{,\beta} = \frac{\partial V^\alpha}{\partial \xi^\beta} - \{ {\gamma \atop \gamma\beta} \} V^\alpha + \{ {\alpha \atop \beta\gamma} \} V^\gamma \tag{4.36}$$

The partial derivative of the flux component can be approximated by central differences. The same interpolation rules as for the momentum equations are applied. All geometrical quantities are evaluated at the centre of a scalar cell. Closest to a boundary, some derivatives $\partial V^\alpha / \partial \xi^\beta$ also contain virtual fluxes. These virtual quantities are expressed in internal fluxes by using linear extrapolation. For example, at lower boundary we get:

$$V^1_{i,-2} = 2V^1_{i,0} - V^1_{i,2} \tag{4.37}$$

In order to obtain non-negative solutions of the equations (4.33) and (4.34) a first order upwind scheme has to be used for the convection terms Zijlema, (1993). Consider the convection term of a convection-diffusion equation (4.27) in integral form:

$$\int_\Omega (c^* U^\alpha T)_{,\alpha} = \int_\Gamma c^* U^\alpha T d\Gamma_\alpha = c^* V^1 T|^{(1,0)}_{(-1,0)} + c^* V^2 T|^{(0,1)}_{(0,-1)} \tag{4.38}$$

Consider only the first part of the right hand side of (4.38). Since T is only given in the centre of a scalar cell, further approximation is needed. This can be done with a first order upwind scheme:

$$T_{(1,0)} \approx \frac{1}{2}[1 + \mathrm{sign}(V^1_{(1,0)})]T_{(0,0)} + \frac{1}{2}[1 - \mathrm{sign}(V^1_{(1,0)})]T_{(2,0)} \tag{4.39}$$

$$T_{(-1,0)} \approx \frac{1}{2}[1 - \mathrm{sign}(V^1_{(-1,0)})]T_{(0,0)} + \frac{1}{2}[1 + \mathrm{sign}(V^1_{(-1,0)})]T_{(-2,0)} \tag{4.40}$$

where the local numbering of figure (4.3) is used. The second part of the convective terms can be approximated the same way.

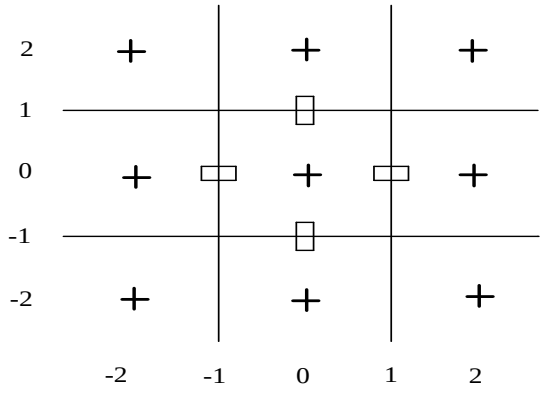


Figure 4.3: Local numbering for scalar cells

5 Implementation of the boundary conditions

In the present version of the ISNaS incompressible program, the following types of boundary conditions have been implemented.

Boundary conditions for the momentum equations:

Type 1: Velocity prescribed (Dirichlet boundary condition)

Type 2: Stress prescribed (Natural boundary condition)

Type 3: Normal stress and tangential velocity given (Semi natural flow)

Type 4: Tangential stress and normal velocity given (Slip boundary condition and also symmetry condition)

Boundary conditions for the convection-diffusion equations are:

Type 1: Scalar T prescribed (Dirichlet boundary condition)

Type 2: $\sigma T + (k\nabla T)\cdot \boldsymbol{n}$ prescribed (Robbins boundary condition)

In the next paragraphs we consider the various boundary conditions separately.

In 2D the notion normal and tangential vector have been defined in a somewhat strange way. For a user the normal vector is defined as the outward normal in the case of a counterclockwise direction and as the inward normal in the case of a clockwise direction. The tangential vector is defined in the direction of the outer boundary. In the program, however, the normal and tangential vector are always defined in the 1 or 2 direction in the computational grid. Hence at the boundary $\xi^2 = 0$, the normal direction is the ξ^2 direction and tangential direction the ξ^1 direction. At the boundary $\xi^1 = nx$, the normal direction is ξ^1 and the tangential direction ξ^2 etc.
In the sequel the internal definition will be used.

5.1 Prescribed velocities

5.1.1 2D implementation

In 2D, prescribed velocity given means in the present version $\boldsymbol{u}\cdot\boldsymbol{n}$ and $\boldsymbol{u}\cdot\boldsymbol{t}$ given. These quantities are transformed to contravariant components using the formulae (6.1) and (6.4) from van Kan et al. (1991):

$$U^n = \sqrt{g^{nn}}\boldsymbol{u}\cdot\boldsymbol{n} \tag{5.1}$$

$$U^t = \frac{1}{g_{tt}}(\sqrt{g_{tt}}\boldsymbol{u}\cdot\boldsymbol{t} - g_{nt}U^n) \tag{5.2}$$

Here the definitions of $\boldsymbol{n}$ and $\boldsymbol{t}$ as described above are meant both for the physical components as for the computational components.

The given normal velocity component (in computational space) is implemented by explicitly prescribing the velocity unknown at the boundary. In the program this is implemented by setting the corresponding main diagonal element equal to 1 and the off-diagonal elements in the corresponding rows to 0. The right-hand-side component corresponding to this unknown is made equal to the unknown itself.

The given velocity component U^t is implemented in the following way:

The matrix is built for all unknowns including all the "tangential" unknowns. The rows corresponding to the "tangential" unknowns closest to the boundary (see Figure 5.1) contain elements referring to virtual pressures and virtual "tangential" velocity unknowns. The virtual

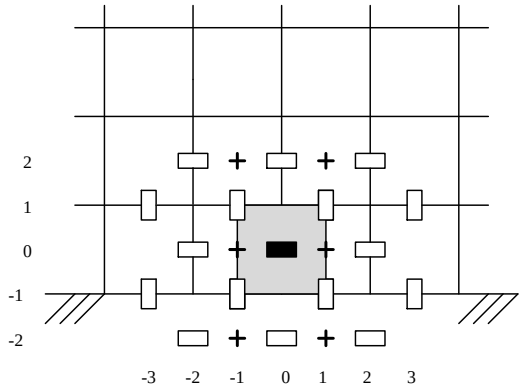


Figure 5.1: A "tangential" velocity cell at the boundary

quantities are expressed in internal unknowns and prescribed velocity components at the boundary using linear extrapolation. For example for the lower boundary (Figure 5.1) we get:

$$p_{i,-2} = 2p_{i,0} - p_{i,2} \tag{5.3}$$

$$V^1_{i,-2} = 2V^1_{i,-1} - V^1_{i,0}, \tag{5.4}$$

where $V^1_{i,-1}$ is the value of $\sqrt{g}U^t$ at the boundary point. The coefficient in the matrix corresponding to the virtual unknown multiplied by the expression (5.3) or (5.4) is transported to the right-hand side or the other matrix terms.

5.1.2 3D implementation

We give here two ways to describe the velocities in the physical domain:

(i) with Cartesian velocity components,

(ii) with normal and tangential components.

In the present version, only the first approach is implemented.

(i) If the Cartesian velocity components are prescribed we can compute the contravariant components in the following way:

$$U^{\alpha} = \boldsymbol{a}^{(\alpha)} \cdot \boldsymbol{u} \tag{5.5}$$

(ii) Normal and tangential components are prescribed on the plane where ξ^n is constant. The scalar products $\boldsymbol{u} \cdot \boldsymbol{n}$, $\boldsymbol{u} \cdot \boldsymbol{\tau}_1$ and $\boldsymbol{u} \cdot \boldsymbol{\tau}_2$ with the tangential vectors $\boldsymbol{\tau}_1$ and $\boldsymbol{\tau}_2$ are given by the user. See Figure 5.2.

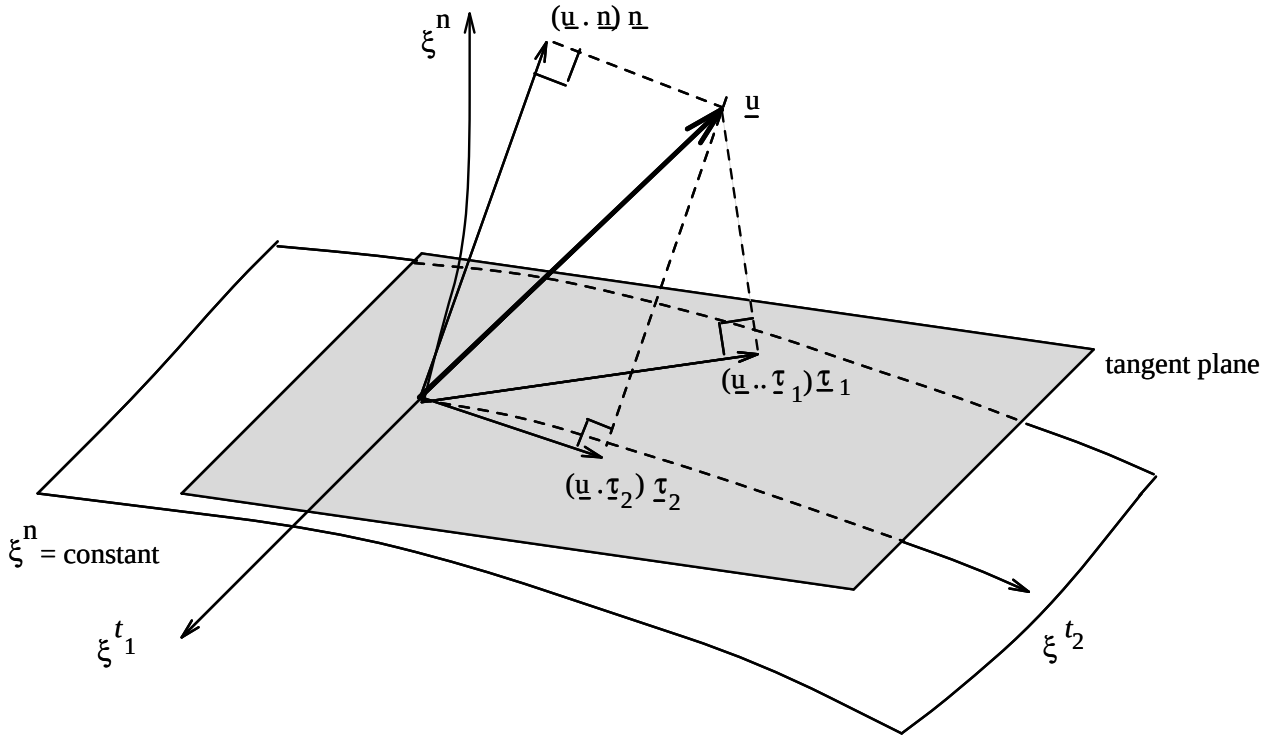


Figure 5.2: The scalar products $\underline{u} \cdot \underline{n}$, $\underline{u} \cdot \underline{\tau}_1$ and $\underline{u} \cdot \underline{\tau}_2$ and tangential vectors $\underline{\tau}_1$ and $\underline{\tau}_2$ are prescribed. Here is $\|\underline{n}\| = \|\underline{\tau}_1\| = \|\underline{\tau}_2\| = 1$.

From $\boldsymbol{u} \cdot \boldsymbol{n}$, $\boldsymbol{u} \cdot \boldsymbol{\tau}_1$, $\boldsymbol{u} \cdot \boldsymbol{\tau}_2$, $\boldsymbol{\tau}_1$ and $\boldsymbol{\tau}_2$ we can compute U^n, U^{t_1} and U^{t_2} by:

$$U^n = sign\,(\boldsymbol{a}^{(n)} \cdot \boldsymbol{n}) \sqrt{g^{nn}}\, \boldsymbol{u} \cdot \boldsymbol{n} \tag{5.6}$$

$$\begin{bmatrix} U^{t_1} \\ U^{t_2} \end{bmatrix} = \begin{bmatrix} g_{t_1 t_1} & g_{t_1 t_2} \\ g_{t_2 t_1} & g_{t_2 t_2} \end{bmatrix}^{-1} \left\{ \begin{bmatrix} \alpha_{11} & \alpha_{12} \\ \alpha_{21} & \alpha_{22} \end{bmatrix}^{-1} \begin{bmatrix} \boldsymbol{u} \cdot \boldsymbol{\tau}_1 \\ \boldsymbol{u} \cdot \boldsymbol{\tau}_2 \end{bmatrix} - U^n \begin{bmatrix} g_{nt_1} \\ g_{nt_2} \end{bmatrix} \right\} \tag{5.7}$$

where

$$\alpha_{i1} = \frac{\begin{vmatrix} \boldsymbol{\tau}_i \cdot \boldsymbol{a}_{(t_1)} & g_{t_1 t_2} \\ \boldsymbol{\tau}_i \cdot \boldsymbol{a}_{(t_2)} & g_{t_2 t_2} \end{vmatrix}}{\begin{vmatrix} g_{t_1 t_1} & g_{t_1 t_2} \\ g_{t_2 t_1} & g_{t_2 t_2} \end{vmatrix}}, \tag{5.8a}$$

$$\alpha_{i2} = \frac{\begin{vmatrix} g_{t_1 t_1} & \boldsymbol{\tau}_i \cdot \boldsymbol{a}_{(t_1)} \\ g_{t_2 t_1} & \boldsymbol{\tau}_i \cdot \boldsymbol{a}_{(t_2)} \end{vmatrix}}{\begin{vmatrix} g_{t_1 t_1} & g_{t_1 t_2} \\ g_{t_2 t_1} & g_{t_2 t_2} \end{vmatrix}}. \tag{5.8b}$$

Formula (5.7) makes no sense if the tangential vectors $\boldsymbol{\tau}_1$ and $\boldsymbol{\tau}_2$ are linear dependent, so they have to be linear independent.

The given U^n, U^{t_1} and U^{t_2} are implemented in almost the same way as in the 2D-case.

5.2 Stresses prescribed

5.2.1 2D implementation

In 2D stresses prescribed implies that normal and tangential stress components at the boundary are prescribed. Let S^{nn} and S^{nt} be the normal resp. tangential physical stress at the boundary, where the normal and tangential vector are defined as in 5.1.1.

From S^{nn} and S^{nt} we can compute σ^{nn} and σ^{nt} by

$$\sigma^{nn} = g^{nn} S^{nn}, \tag{5.9}$$

$$\sigma^{nt} = (\sqrt{g^{nn} g_{tt}} S^{nt} - g_{nt} \sigma^{nn}) / g_{tt}, \tag{5.10}$$

where $\sigma^{\alpha\beta}$ is defined by

$$\sigma^{\alpha\beta} = -g^{\alpha\beta} p + \tau^{\alpha\beta}. \tag{5.11}$$

An important remark is that in this formulation pressure and deviatoric stress tensor can not be separated, hence the discretization of both must be the same at the boundary. For that reason the discretization of the pressure at the boundary will be different from the one in the inner region.

Since no velocities are prescribed, it is necessary to consider finite volume cells around each velocity unknown, including the "normal" velocity points at the boundary.

Let us first consider the "tangential" boundary cell as sketched in Figure 5.1. The discretization of the convective terms, the right-hand side and the time derivative are exactly the same as for the inner cells, with the exception that virtual (tangential) velocities are eliminated by linear extrapolation as in formula (5.4).

The stress tensor (deviatoric part and pressure together) is discretized by:

$$\sqrt{g}\sigma^{11}|_{(-1,0)}^{(1,0)} + \sqrt{g}\sigma^{12}|_{(0,-1)}^{(0,1)} + \{{}^{1}_{\gamma\beta}\}\sigma^{\gamma\beta}\sqrt{g}|_{(0,0)} \tag{5.12}$$

In this expression $\sigma^{12}|_{(0,-1)}$ is given by formula (5.4). All other terms are treated in the usual way (except of course for the pressure).

With respect to the normal velocity unknown at the boundary a half cell is defined as in Figure 5.3. The discretization of the convective terms plus the stress tensor at the boundary

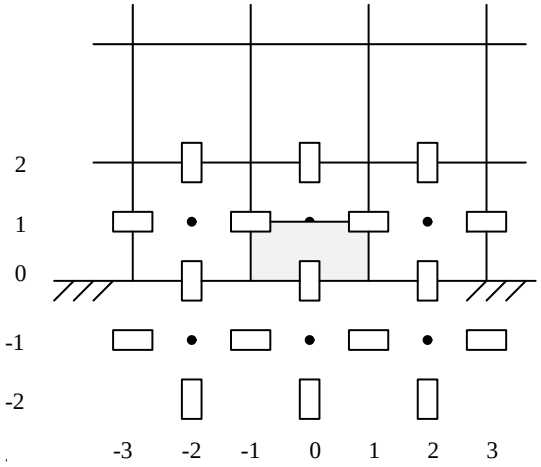


Figure 5.3: A "normal" velocity half cell at the boundary

is given by formula (6.14) from van Kan et al (1991):

$$\frac{1}{2}\sqrt{g}T^{21}|_{(-1,1)}^{(1,1)} + \sqrt{g}T^{22}|_{(0,0)}^{(0,1)} + \frac{1}{2}\sqrt{g}\{{}^{2}_{\gamma\beta}\}T^{\gamma\beta}|_{(0,0)}, \tag{5.13}$$

where

$$T^{\alpha\beta} = \rho U^{\alpha}U^{\beta} - \sigma^{\alpha\beta} \tag{5.14}$$

The discretization of the right-hand side gives

$$\frac{1}{2}\rho f^{\alpha}\sqrt{g}|_{(0,0)} \tag{5.15}$$

and of the time-derivative:

$$\frac{1}{2}\frac{\partial}{\partial t}(\rho V^{\alpha})|_{(0,0)} \tag{5.16}$$

The discretization of the convective terms is derived from (5.13) by substitution of

$$T^{\alpha\beta} = \rho U^{\alpha}U^{\beta} \tag{5.17}$$

and the approximation

$$V^{1}_{0,0} = \frac{1}{2}(V^{1}_{1,1} + V^{1}_{-1,1}) \tag{5.18}$$

The discretization of the stress tensor at the boundary is given by formulae (6.14), (6.15) of van Kan et al (1991):

$$RHS - \sqrt{g}\sigma^{22}|_{(0,1)} - \frac{1}{2}\sqrt{g}\{\begin{smallmatrix}2\\11\end{smallmatrix}\}\sigma^{11}|_{(0,0)}, \tag{5.19}$$

where RHS is defined by

$$\begin{aligned} RHS &= -\frac{1}{2}\sqrt{g}\sigma^{21}|_{(1,0)} + \frac{1}{2}\sqrt{g}\sigma^{21}|_{(-1,0)} + \sqrt{g}\sigma^{22}|_{(0,0)} \\ &\quad -\frac{1}{2}\sqrt{g}\{\begin{smallmatrix}2\\22\end{smallmatrix}\}\sigma^{22}|_{(0,0)} - \sqrt{g}\{\begin{smallmatrix}2\\12\end{smallmatrix}\}\sigma^{12}|_{(0,0)} \end{aligned} \tag{5.20}$$

The evaluation of $\sigma^{11}|_{(0,0)}$ introduces extra difficulties.
Following van Kan et al (1991), page 76, we use $p_{0,1}$ instead of $p_{0,0}$.
Furthermore $\frac{\partial U^1}{\partial \xi^2}|_{(0,0)}$ is computed at the preceding time-level, and $\frac{\partial U^1}{\partial \xi^1}|_{(0,0)}$ replaced by $\frac{\partial U^1}{\partial \xi^1}|_{(0,1)}$. Virtual velocities are not used. To compute $\frac{\partial U^1}{\partial \xi^2}|_{(0,0)}$ at the preceding time level, U^1 at the boundary is computed by linear extrapolation from inside, using two points.

5.2.2 3D implementation

The normal and tangential stress components at the boundary are prescribed. Let S^{nn} be the normal stress component at the boundary and $S^{n\tau}$ the tangential stress component in the $\boldsymbol{\tau}$ direction (see Figure 5.4). So:

$$\boldsymbol{S}_{\xi^n = const} = S^{nn}\boldsymbol{n} + S^{n\tau}\boldsymbol{\tau} \tag{5.21}$$

where $\boldsymbol{n}$ is an unit normal vector and $\boldsymbol{\tau}$ is an unit tangential vector. From $S^{nn}\boldsymbol{n}$ and $S^{n\tau}\boldsymbol{\tau}$ we compute σ^{nn}, σ^{nt_1} and σ^{nt_2} the stresses in the computational domain. Just asin the 2D-case

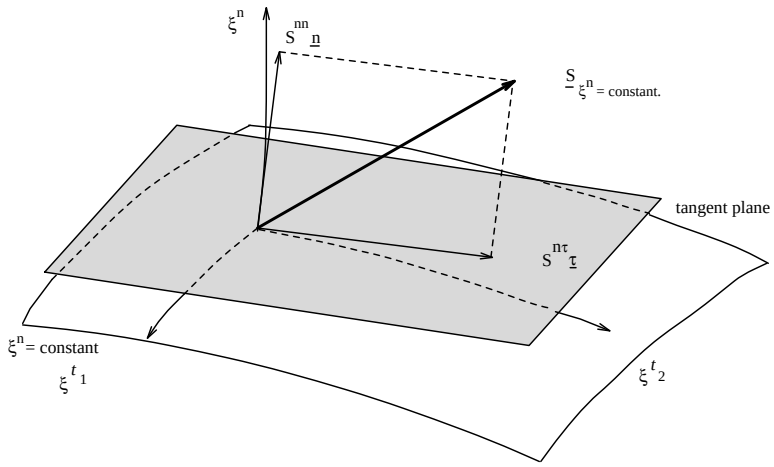


Figure 5.4: The normal and tangential stress in the physical domain at the boundary $\xi^n = constant$.

is:

$$\sigma^{nn} = g^{nn} S^{nn} \ , \tag{5.22}$$

σ^{nt_1} and σ^{nt_2} are computed by:

$$\begin{bmatrix} \sigma^{nt_1} \\ \sigma^{nt_2} \end{bmatrix} = sign\ (\boldsymbol{a}^{(n)} \cdot \boldsymbol{n}) \sqrt{g^{nn}} S^{n\tau} \begin{bmatrix} \alpha_1 \\ \alpha_2 \end{bmatrix} - \sigma^{nn} \begin{bmatrix} g_{t_1 t_1} & g_{t_1 t_2} \\ g_{t_2 t_1} & g_{t_2 t_2} \end{bmatrix}^{-1} \begin{bmatrix} g_{nt_1} \\ g_{nt_2} \end{bmatrix} \tag{5.23}$$

where

$$\alpha_1 = \frac{\begin{vmatrix} \boldsymbol{\tau} \cdot \boldsymbol{a}_{(t_1)} & g_{t_1 t_2} \\ \boldsymbol{\tau} \cdot \boldsymbol{a}_{(t_2)} & g_{t_2 t_2} \end{vmatrix}}{\begin{vmatrix} g_{t_1 t_1} & g_{t_1 t_2} \\ g_{t_2 t_1} & g_{t_2 t_2} \end{vmatrix}} .; \tag{5.24a}$$

$$\alpha_2 = \frac{\begin{vmatrix} g_{t_1 t_1} & \boldsymbol{\tau} \cdot \boldsymbol{a}_{(t_1)} \\ g_{t_2 t_1} & \boldsymbol{\tau} \cdot \boldsymbol{a}_{(t_2)} \end{vmatrix}}{\begin{vmatrix} g_{t_1 t_1} & g_{t_1 t_2} \\ g_{t_2 t_1} & g_{t_2 t_2} \end{vmatrix}} \tag{5.24b}$$

The stress $\sigma^{\alpha\beta}$ is defined by $\sigma^{\alpha\beta} = -g^{\alpha\beta} p + \tau^{\alpha\beta}$. At the boundary it is impossible to separate the pressure from te deviatoric stress tensor $\tau^{\alpha\beta}$. So the discretization of the pressure at the boundary will be different from the one in the inner region.

We have to consider three different cells closest to the boundary two "tangential" and one "normal" velocity cell.

Let us first consider the "tangential" cells. The two "tangential" cells closest to the boundary are considered differently from the ones in the inner region, because the stencil contain virtual unknowns (see Figure 5.5). The discretization of the convective terms, the right-hand side and the time derivative are the same as for the inner cells, with the exception that virtual velocities are eliminated by linear extrapolation. For example for the bottom boundary (see Figure 5.5) we get:

$$V^1_{i,j,-2} = 2V^1_{i,j,0} - V^1_{i,j,2} \tag{5.25}$$

$$V^2_{i,j,-2} = 2V^2_{i,j,0} - V^2_{i,j,2} \tag{5.26}$$

$$V^3_{i,j,-3} = 2V^3_{i,j,-1} - V^3_{i,j,1} \tag{5.27}$$

The stress tensor $\sigma^{\alpha\beta}$ is discretized in the following way for the V^α-cell:

$$\begin{aligned} &-\sqrt{g}\sigma^{\alpha 1}|^{(1,0,0)}_{(-1,0,0)} - \sqrt{g}\sigma^{\alpha 2}|^{(0,1,0)}_{(0,-1,0)} - \sqrt{g}\sigma^{\alpha 3}|^{(0,0,1)}_{(0,0,-1)} \\ &-\sqrt{g}\{{}^{\alpha}_{\gamma\beta}\}\sigma^{\gamma\beta}|_{(0,0,0)} \quad \text{for} \quad \alpha = 1, 2. \end{aligned} \tag{5.28}$$

Term $\sigma^{\alpha 3}|_{(0,0,-1)}$ is given by formula (5.23), if we are concerned with the bottom boundary. All other terms are treated in the usual way.

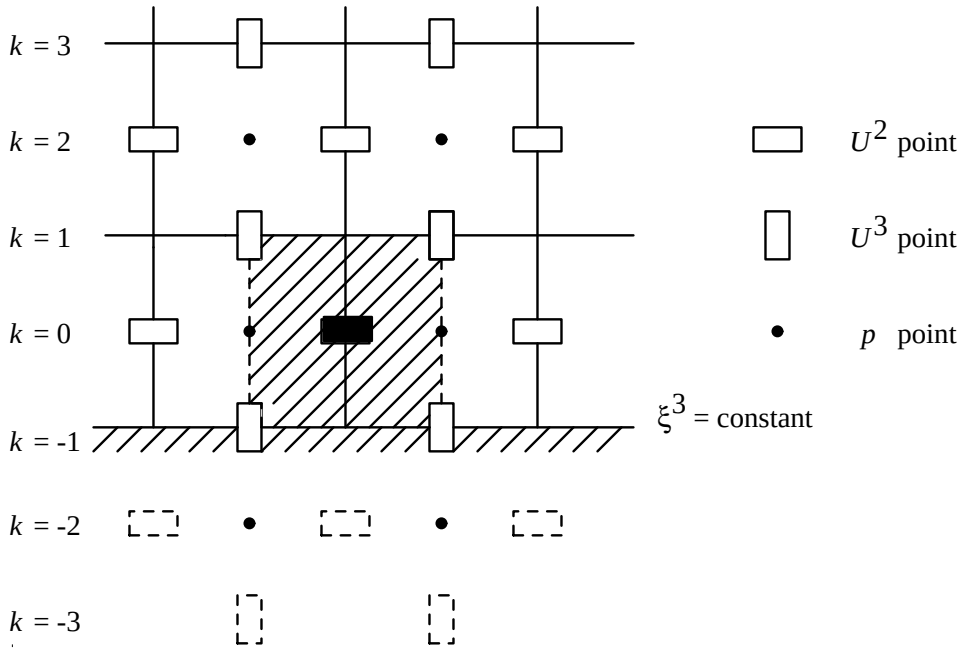


Figure 5.5: A cross-section ($\xi^1 = constant$) over an U^2-cell closest to the bottom of the region.

Since no normal velocity components are prescribed at the boundary we have to consider a finite volume around a "normal" velocity point at the boundary (see Figure 5.6). We will now consider the discretization for a "normal" velocity cell at the bottom boundary.
The discretization of the time-derivative gives:

$$\frac{1}{2}\frac{\partial}{\partial t}(\rho V^3)|_{(0,0,0)} \tag{5.29}$$

and of the right-hand side:

$$\frac{1}{2}(\rho f^3\sqrt{g})|_{(0,0,0)}\,. \tag{5.30}$$

The discretization of the convective terms is given by:

$$\begin{aligned}&\frac{1}{2}\frac{\rho}{\sqrt{g}}V^3V^1|^{(1,0,0)}_{(-1,0,0)} + \frac{1}{2}\frac{\rho}{\sqrt{g}}V^3V^2|^{(0,1,0)}_{(0,-1,0)} + \frac{\rho}{\sqrt{g}}V^3V^3|^{(0,0,1)}_{(0,0,0)}\\ &+\frac{1}{2}\frac{\rho}{\sqrt{g}}\{{}^{3}_{\gamma\beta}\}V^\gamma V^\beta|_{(0,0,0)}\end{aligned} \tag{5.31}$$

and the approximation:

$$V^1_{i,j,0} = V^1_{i,j,1} \tag{5.32}$$

$$V^2_{i,j,0} = V^2_{i,j,1}\,. \tag{5.33}$$

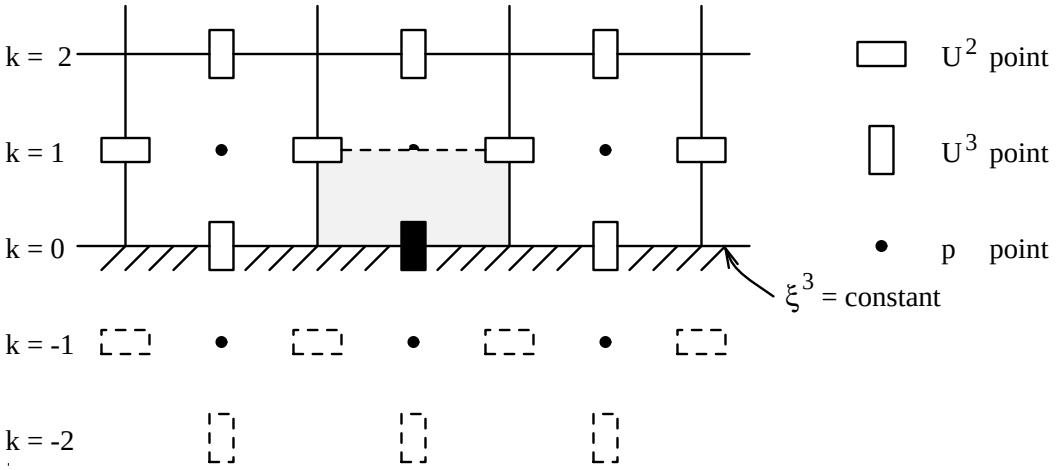


Figure 5.6: A cross-section ($\xi^1 = constant$) over an U^3-cell at the bottom.

If V^1 or V^2 are not present at $(i,j,1)$ then they are approximated by:

$$V^1_{i,j,1} = \frac{1}{2}(V^1_{i-1,j,1} + V^1_{i+1,j,1}) \tag{5.34}$$

$$V^2_{i,j,1} = \frac{1}{2}(V^2_{i,j-1,1} + V^2_{i,j+1,1})\ . \tag{5.35}$$

The discretization of the stress tensor at the boundary is given by the following formula:

$$\begin{aligned} &-\frac{1}{2}\sqrt{g}\sigma^{31}|^{(1,0,0)}_{(-1,0,0)} - \frac{1}{2}\sqrt{g}\sigma^{32}|^{(0,1,0)}_{(0,-1,0)} - \sqrt{g}\sigma^{33}|^{(0,0,1)}_{(0,0,0)} \\ &-\frac{1}{2}\sqrt{g}\{\begin{smallmatrix}3\\ \gamma\beta\end{smallmatrix}\}\sigma^{\gamma\beta}|_{(0,0,0)} \end{aligned} \tag{5.36}$$

or

$$\begin{aligned} RHS \;&-\; \sqrt{g}\sigma^{33}|_{(0,0,1)} - \frac{1}{2}\sqrt{g}\{\begin{smallmatrix}3\\ 11\end{smallmatrix}\}\sigma^{11}|_{(0,0,0)} \\ &-\; \sqrt{g}\{\begin{smallmatrix}3\\ 12\end{smallmatrix}\}\sigma^{12}|_{(0,0,0)} - \frac{1}{2}\sqrt{g}\{\begin{smallmatrix}3\\ 22\end{smallmatrix}\}\sigma^{22}|_{(0,0,0)} \end{aligned} \tag{5.37}$$

where RHS is given by:

$$\begin{aligned} RHS \;=\; &-\frac{1}{2}\sqrt{g}\sigma^{31}|^{(1,0,0)}_{(-1,0,0)} - \frac{1}{2}\sqrt{g}\sigma^{32}|^{(0,1,0)}_{(0,-1,0)} + \sqrt{g}\sigma^{33}|_{(0,0,0)} \\ &-\sqrt{g}\{\begin{smallmatrix}3\\ 13\end{smallmatrix}\}\sigma^{13}|_{(0,0,0)} - \sqrt{g}\{\begin{smallmatrix}3\\ 23\end{smallmatrix}\}\sigma^{23}|_{(0,0,0)} - \frac{1}{2}\sqrt{g}\{\begin{smallmatrix}3\\ 33\end{smallmatrix}\}\sigma^{33}|_{(0,0,0)}\ . \end{aligned} \tag{5.38}$$

The evaluation of $\sigma^{11}|_{(0,0,0)}$, $\sigma^{12}_{(0,0,0)}$ and $\sigma^{22}_{(0,0,0)}$ introduces some difficulties. First we need the pressure in point $(0,0,0)$, because we have to split up σ^{11}, σ^{12} and σ^{22}. Instead of $p_{0,0,0}$

we use $p_{0,0,1}$ just as in the 2D-case.
Secondly we need $\frac{\partial U^1}{\partial \xi^\alpha}$ and $\frac{\partial U^2}{\partial \xi^\alpha}$ at (0,0,0) for $\alpha = 1,2,3$. The derivatives $\frac{\partial U^1}{\partial \xi^1}|_{(0,0,0)}$, $\frac{\partial U^1}{\partial \xi^2}|_{(0,0,0)}$, $\frac{\partial U^2}{\partial \xi^1}|_{(0,0,0)}$ and $\frac{\partial U^2}{\partial \xi^2}|_{(0,0,0)}$ are replaced respectively by $\frac{\partial U^1}{\partial \xi^1}|_{(0,0,1)}$, $\frac{\partial U^1}{\partial \xi^2}|_{(0,0,1)}$, $\frac{\partial U^2}{\partial \xi^1}|_{(0,0,1)}$ and $\frac{\partial U^2}{\partial \xi^2}|_{(0,0,1)}$. In van Kan et al (1991), page 34, there are three strategies mentioned to compute $\frac{\partial U^1}{\partial \xi^3}|_{(0,0,0)}$ and $\frac{\partial U^2}{\partial \xi^3}|_{(0,0,0)}$. It seems reasonable if we use the same strategy as used in the 2D-case. That is, the derivatives $\frac{\partial U^1}{\partial \xi^3}|_{(0,0,0)}$ and $\frac{\partial U^2}{\partial \xi^3}|_{(0,0,0)}$ are computed at the preceding time level, U^1 and U^2 at the boundary are computed by linear extrapolation, using two points, so:

$$\frac{\partial U^\alpha}{\partial \xi^3}|_{(0,0,0)} = U^\alpha|_{(0,0,3)} - U^\alpha|_{(0,0,1)} \tag{5.39}$$

(at the preceding time level) for $\alpha = 1,2$.

5.3 Semi-natural outflow condition

5.3.1 2D implementation

With semi-natural outflow condition we mean tangential velocity and normal stress prescribed at the boundary, i.e.

$$\boldsymbol{u} \cdot \boldsymbol{t} \text{ and } S^{nn} \text{ given.} \tag{5.40}$$

Although $\boldsymbol{u} \cdot \boldsymbol{t}$ prescribed does in general not imply U^t prescribed we assume that instead of (5.40) the following boundary condition is given:

$$U^t \text{ and } S^{nn} \text{ given} \tag{5.41}$$

where S^{nn} is related to σ^{nn} by (5.9).

Boundary condition (5.41) influences both the tangential velocity cell as sketched in Figure 5.1, as the normal velocity half-cell sketched in Figure 5.3.

With respect to the tangential cell, the molecule is built in the same way as for the inner cells. The only difference is that virtual velocity components and virtual pressures are eliminated by linear extrapolation, i.e. by applying formulae (5.3) and (5.4).

The normal velocity half cell gives rise to the following discretization of the stress tensor (see van Kan et al 1991, formula (6.19)):

$$\begin{aligned} &-\frac{1}{2}\sqrt{g}\sigma^{21}|_{(-1,0)}^{(1,0)} - \sqrt{g}\sigma^{22}|_{(0,1)} - \frac{1}{2}\sqrt{g}(\{\begin{smallmatrix}2\\11\end{smallmatrix}\}\sigma^{11} + 2\{\begin{smallmatrix}2\\12\end{smallmatrix}\}\sigma^{12})|_{(0,0)} \\ &+\sqrt{g}\sigma^{22}|_{(0,0)} - \frac{1}{2}\sqrt{g}\{\begin{smallmatrix}2\\22\end{smallmatrix}\}\sigma^{22}|_{(0,0)}, \end{aligned} \tag{5.42}$$

where $\sigma^{22}|_{(0,0)}$ is given by (5.9). Virtual velocities are eliminated by linear extrapolation using formula (5.3). It must be remarked that the first term with respect to the pressure is

evaluated in the points (1,1) and (-1,1) instead of (1,0) resp. (-1,0).
The convective terms are evaluated by expanding

$$\frac{1}{2}\rho\sqrt{g}U^2U^\beta|_{(-1,0)}^{(1,0)} + \rho\sqrt{g}U^2U^\beta|_{(0,0)}^{(0,1)} + \frac{1}{2}\rho\sqrt{g}\{{}^{2}_{\gamma\beta}\}U^\gamma U^\beta|_{(0,0)} \tag{5.43}$$

using the standard inter- and extrapolations.

5.3.2 3D implementation

The tangential velocity and normal stress are prescribed at the boundary, i.e.:

$$\boldsymbol{u}\cdot\boldsymbol{\tau}_1,\ \boldsymbol{u}\cdot\boldsymbol{\tau}_2,\ \boldsymbol{\tau}_1,\ \boldsymbol{\tau}_2 \quad \text{and} \quad S^{nn} \quad \text{are given .} \tag{5.44}$$

From equation (5.7) it is clear that we can calculate U^{t_1} and U^{t_2} as g_{nt_1} and g_{nt_2} are zero, otherwise we have to make an assumption about U^{t_1} and U^{t_2}. In the remainder of this section we assume that

$$U^{t_1},\ U^{t_2} \quad \text{and} \quad S^{nn} \quad \text{are given ,} \tag{5.45}$$

at the boundary.
From S^{nn} and equation (5.22) follows the stress σ^{nn} at the boundary in the computational domain.

Boundary condition (5.45) influences the two "tangential" velocity cells (see Figure 5.5) and the "normal" velocity half-cell (see Figure 5.6).

The U^1 and U^2 "tangential" cells (bottom boundary) are built in a similar way as the inner cells. The only difference is that virtual velocity components and pressures are eliminated by linear extrapolation. For example for the bottom boundary we get:

$$V^\alpha_{i,j,-2} = 2V^\alpha_{i,j,-1} - V^\alpha_{i,j,0} \quad \text{for} \quad \alpha = 1,2 \tag{5.46}$$

$$V^3_{i,j,-3} = 2V^3_{i,j,-1} - V^3_{i,j,1} \tag{5.47}$$

$$p_{i,j,-2} = 2p_{i,j,0} - p_{i,j,2}\ . \tag{5.48}$$

The normal velocity half-cell at the bottom boundary. The discretization of the convective term gives

$$\begin{aligned}&\frac{1}{2}\frac{\rho}{\sqrt{g}}V^3V^1|_{(-1,0,0)}^{(1,0,0)} + \frac{1}{2}\frac{\rho}{\sqrt{g}}V^3V^2|_{(0,-1,0)}^{(0,1,0)} + \frac{\rho}{\sqrt{g}}V^3V^3|_{(0,0,0)}^{(0,0,1)}\\ &+\frac{1}{2}\frac{\rho}{\sqrt{g}}\{{}^{3}_{\gamma\beta}\}V^\gamma V^\beta|_{(0,0,0)}\ .\end{aligned} \tag{5.49}$$

Terms with the factor V^3V^3 are the only terms where we need a linearization procedure, since V^1 and V^2 are given at the boundary.
We use the following discretization of the stress tensor:

$$\begin{aligned}&-\ \frac{1}{2}\sqrt{g}\sigma^{31}|_{(-1,0,0)}^{(1,0,0)} - \frac{1}{2}\sqrt{g}\sigma^{32}|_{(0,-1,0)}^{(0,1,0)} - \sqrt{g}\sigma^{33}|_{(0,0,0)}^{(0,0,1)}\\ &-\ \frac{1}{2}\sqrt{g}\{{}^{3}_{\gamma\beta}\}\sigma^{\gamma\beta}|_{(0,0,0)}\ ,\end{aligned} \tag{5.50}$$

where $\sigma^{33}|_{(0,0,0)}$ is given by $(g^{33}S^{33})|_{(0,0,0)}$. So the right-hand side gets the contribution:

$$(-1 + \frac{1}{2}\{\begin{smallmatrix} 3 \\ 33 \end{smallmatrix}\})\sqrt{g}\sigma^{33}|_{(0,0,0)} \cdot \tag{5.51}$$

The virtual velocities introduced by formula (5.51) are eliminated by linear extrapolation. Just as in 2D is the pressure evaluated in the points (1,0,1), (-1,0,1), (0,1,1) and (0,-1,1) instead of (1,0,0), (-1,0,0), (0,1,0) and (0,-1,0).

5.4 Slip boundary condition

5.4.1 2D implementation

The slip boundary condition is equivalent to tangential stress as well as normal velocity component given. The treatment of this type of boundary conditions is the subject of Segal (1991). All formulae in this section are copied from that report.
The normal velocity given implies U^n given by the relation (5.1). The tangential stress given, however, does not automatically imply that a component of the contravariant stress tensor is prescribed. In fact it only implies a linear combination between stress components at the boundary through the relation

$$\sqrt{g^{nn}g_{tt}}S^{nt} = g_{nt}\sigma^{nn} + g_{tt}\sigma^{nt} \tag{5.52}$$

Since the normal velocity component is given no normal half cells are introduced.

The discretization of the convective terms implies the evaluation of virtual velocities by linear extrapolation. The discretization of the stress tensor is given by

$$-\sqrt{g}\sigma^{11}|_{(-1,0)}^{(1,0)} - \sqrt{g}\sigma^{12}|_{(0,-1)}^{(0,1)} - \{\begin{smallmatrix} 1 \\ \gamma\beta \end{smallmatrix}\}\sigma^{\gamma\beta}\sqrt{g}|_{(0,0)} \tag{5.53}$$

The linear extrapolation formula for the velocities is given by

$$V^1_{i,-2} = 2V^1_{i,0} - V^1_{i,2} \tag{5.54}$$

The term $(\sqrt{g}\sigma^{12})_{(0,-1)}$ is equal to:

$$\sqrt{g}\sigma^{12}|_{(0,-1)} = S^{nt}|_{(0,-1)} - \sqrt{g}\frac{g_{21}}{g_{11}}\sigma^{22}|_{(0,-1)}, \tag{5.55}$$

where the first term is given and the second term involves virtual velocities and pressures that can be eliminated by (5.54) resp. (5.3).

5.4.2 3D implementation

In this condition the normal velocity $\boldsymbol{u}\cdot\boldsymbol{n}$ and tangential stress $S^{n\tau}$ with $\boldsymbol{\tau}$ are prescribed. So we don't need "normal" velocity half-cells. We have only to consider the two "tangential"

cells. For the remainder of this section is the slip boundary condition given on the bottom boundary.

The discretization of the convective terms is given by:

$$\frac{\rho}{\sqrt{g}}V^{\alpha}V^{1}|^{(1,0,0)}_{(-1,0,0)} + \frac{\rho}{\sqrt{g}}V^{\alpha}V^{2}|^{(0,1,0)}_{(0,-1,0)} + \frac{\rho}{\sqrt{g}}V^{\alpha}V^{3}|^{(0,0,1)}_{(0,0,-1)}$$
$$+\frac{\rho}{\sqrt{g}}\{\begin{matrix}\alpha\\ \gamma\beta\end{matrix}\}V^{\gamma}V^{\beta}|_{(0,0,0)} \quad \text{for} \quad \alpha = 1,2\,. \tag{5.56}$$

Since $V^3 = \sqrt{g}U^3$ is prescribed at the bottom boundary we do not have to use a linear approximation for $V^{\alpha}V^3|_{(0,0,-1)}$ (see Segal (1991), formula (3.9)):

$$V^{\alpha}V^{3}|_{(0,0,-1)} = \frac{1}{2}(3V^{\alpha}|_{(0,0,0)} - V^{\alpha}|_{(0,0,2)})V^{3}|_{(0,0,-1)} \quad \text{for} \quad \alpha = 1,2\,. \tag{5.57}$$

The discretization of the stress tensor is given by:

$$-\sqrt{g}\sigma^{\alpha 1}|^{(1,0,0)}_{(-1,0,0)} - \sqrt{g}\sigma^{\alpha 2}|^{(0,1,0)}_{(0,-1,0)} - \sqrt{g}\sigma^{\alpha 3}|^{(0,0,1)}_{(0,0,-1)}$$
$$-\sqrt{g}\{\begin{matrix}\alpha\\ \gamma\beta\end{matrix}\}\sigma^{\gamma\beta}|_{(0,0,0)} \quad \text{for} \quad \alpha = 1,2\,. \tag{5.58}$$

The virtual velocities introduced by formula (5.58) are eliminated by linear extrapolation, i.e. using formula (5.25), (5.26) and (5.27).

The term $\sigma^{\alpha 3}|_{(0,0,-1)}$ in (5.58) is for $\alpha = 1$ (U^1-cell) equal to:

$$\sigma^{13}|_{(0,0,-1)} = \left(sign\ (\boldsymbol{a}^{(3)}\cdot\boldsymbol{n})\sqrt{g^{33}}S^{n\tau}\alpha_1 - \sigma^{33}\boldsymbol{e}_1^T \begin{bmatrix} g_{11} & g_{12}\\ g_{21} & g_{22}\end{bmatrix}^{-1}\begin{bmatrix} g_{31}\\ g_{32}\end{bmatrix}\right)|_{(0,0,-1)} \tag{5.59}$$

where $\boldsymbol{e}_1^T = [1\ \ 0]$, so:

$$\sigma^{13}|_{(0,0,-1)} = \left(sign\ (\boldsymbol{a}^{(3)}\cdot\boldsymbol{n})\sqrt{g^{33}}S^{n\tau}\alpha_1 - \sigma^{33}\frac{g_{22}g_{31} - g_{12}g_{32}}{g_{11}g_{22} - g_{12}g_{21}}\right)|_{(0,0,-1)}\,. \tag{5.60}$$

Term $\sigma^{\alpha 3}|_{(0,0,-1)}$ in (5.58) is for $\alpha = 2$ equal to:

$$\sigma^{23}|_{(0,0,-1)} = \left(sign\ (\boldsymbol{a}^{(3)}\cdot\boldsymbol{n})\sqrt{g^{33}}S^{n\tau}\alpha_2 - \sigma^{33}\boldsymbol{e}_2^T \begin{bmatrix} g_{11} & g_{12}\\ g_{21} & g_{22}\end{bmatrix}^{-1}\begin{bmatrix} g_{31}\\ g_{32}\end{bmatrix}\right)|_{(0,0,-1)} \tag{5.61}$$

where $\boldsymbol{e}_2^T = [0\ \ 1]$, so:

$$\sigma^{23}|_{(0,0,-1)} = \left(sign\ (\boldsymbol{a}^{(3)}\cdot\boldsymbol{n})\sqrt{g^{33}}S^{n\tau}\alpha_2 - \sigma^{33}\frac{g_{11}g_{32} - g_{21}g_{31}}{g_{11}g_{22} - g_{21}g_{12}}\right)|_{(0,0,-1)}\,. \tag{5.62}$$

The factors α_1 and α_2 are calculated with formula (5.24a)-(5.24b).
The $\sigma^{33}|_{(0,0,-1)}$ in (5.60) and (5.62) involves virtual velocities and pressures that can be eliminated by (5.25)-(5.27) and (5.48).

Before treating the boundary conditions for the scalar equations we shall first consider the special cases where we have a transition of one type of a boundary condition to another as well the case of a corner of the region.

5.5 Transition of types of boundary conditions

5.5.1 2D implementation

Just as in the preceding cases we restrict ourselves to the lower boundary in the computational domain. Let at the vertex point S we have two types of boundary conditions (see Figure 5.7).

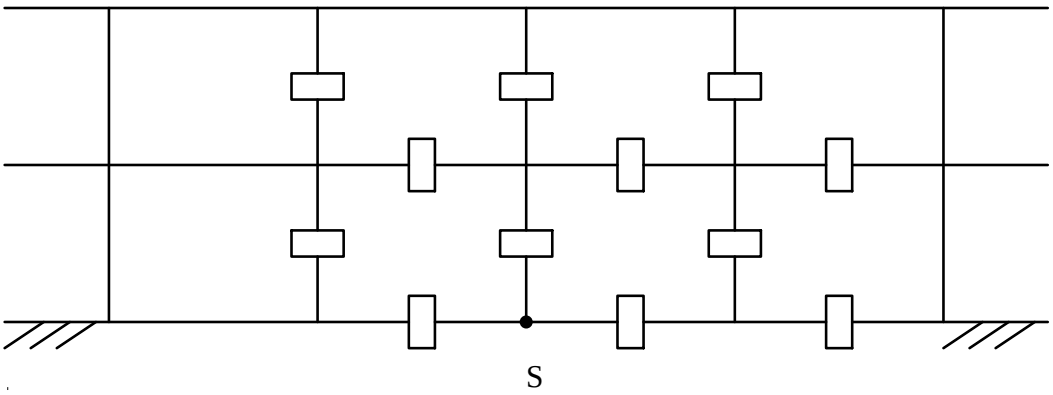


Figure 5.7: transition point S (vertex of cell). At the left of point S the type of boundary condition differs from the one on the right.

We shall only consider the boundary condition at the left of point S but in relation to the boundary condition at the right. In all cases the most restrictive boundary conditions, i.e. the one that influences the velocity most directly will be applied. Let us first consider Dirichlet boundary conditions at the left of S. Since Dirichlet boundary conditions are the most restrictive, it is assumed that also in point S the velocity is prescribed. All points left of S are treated in the usual way. The only special treatment is required for the tangential cell just above point S (Figure 5.8).

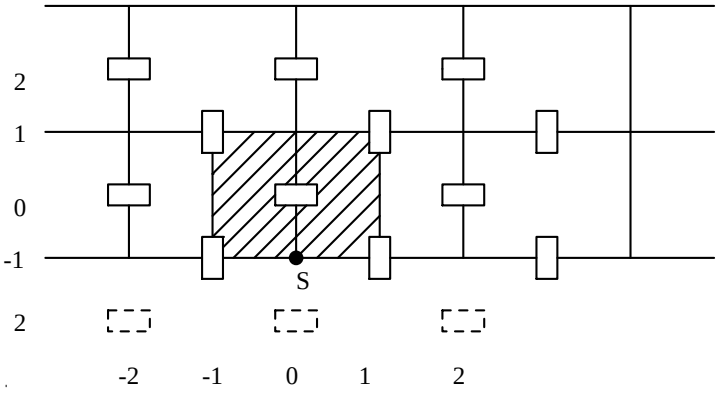


Figure 5.8: Tangential cell just above transition point S.

The molecule corresponding to $V^1_{(0,0)}$ contains 3 virtual points, 2 of which can be eliminated

by linear extrapolation using the boundary conditions.
The only special point is $V^1_{(2,-2)}$. If at the right side of S we have a boundary condition of type 3 this point may be treated in the usual way. However, if boundary conditions of type 2 or 4 are prescribed $V^1_{(2,-2)}$ must be eliminated by the linear extrapolation:

$$V^1_{(2,-2)} = 2V^1_{(2,0)} - V^1_{(2,2)} \tag{5.63}$$

Now suppose that we have a boundary condition of type 2 at the left side of S and a boundary condition of type 1, 3 or 4 at the right side of S. It is sufficient to consider the tangential cell sketched in Figure 5.8 and the normal half cell left of point S sketched in Figure 5.9.

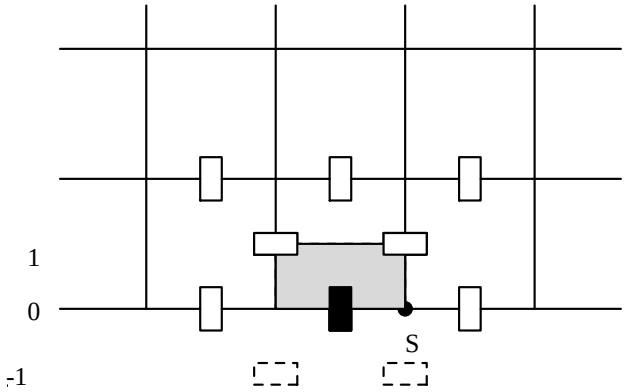


Figure 5.9: Normal half cell left of the transition point S

All other cells are treated in the usual way. Let us first consider the tangential cell of Figure 5.8. If at the right side of point S we have a boundary condition of type 1 or type 3, the tangential velocity in point S is given and the cell is treated as if corresponding to the right side. In the case of boundary conditions of type 4 the cell may be treated in the usual way.
With respect to the normal half cell we can proceed as usual. Since no virtual velocities appear no special treatment is necessary.

In the case that we have a boundary condition of type 3 at the left side of S we also distinguish between the tangential cell of Figure 5.8 and the normal half cell of Figure 5.9. With respect to the tangential cell the procedure described in 5.3.1 can be applied without any restriction. With respect to the normal half cell we may also proceed in the standard way, i.e. apply formula (5.42). This procedure leads to virtual unknowns which may be eliminated in the usual way. There is no need to use extra information if velocities are given at the right of point S.

Finally with respect to boundary conditions of type 4 it is sufficient to consider the tangential cell of Figure 5.8. If at the right side of point S boundary conditions of type 1 or type 3 are given (i.e. u_t prescribed) these boundary conditions prevail. In the case of boundary condition of type 2 at the right side of S, no special action is necessary.

5.5.2 3D implementation

Just as in the 2D case we restrict ourselves to only one boundary, the bottom boundary in the computational domain. First we assume that only one boundary condition type is prescribed on a "bottom boundary"-face of a p-cell, see Figure 5.10.

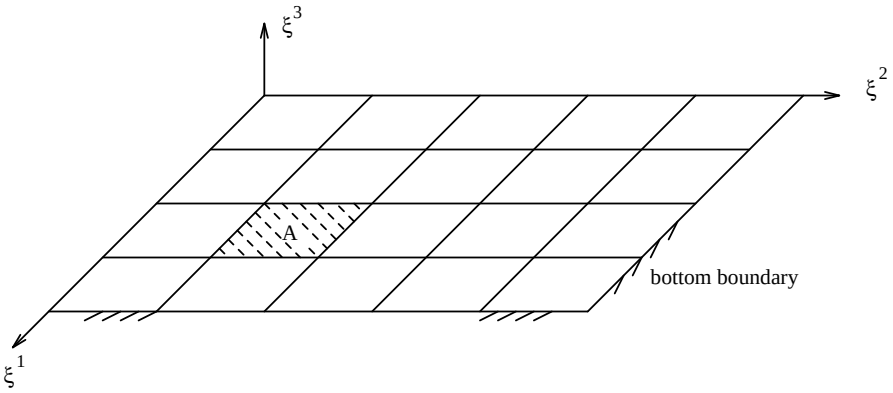


Figure 5.10: There is only one boundary condition type prescribed on A, the "bottom boundary"-face of a p-cell.

It is clear that the only thing that has to be treated very carefully is the extrapolations of the virtual points, see Figure 5.11.

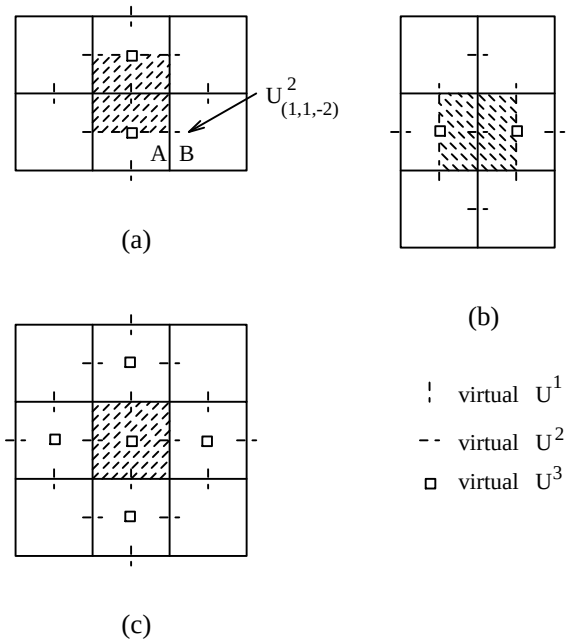


Figure 5.11: A cross-section of the bottom boundary and an (a) U^1-cell, (b) U^2-cell and (c) U^3-cell, with the positions of the virtual unknowns.

For the extrapolation of for instance $V^2_{(1,1,-2)}$ in a U^1-cell (see Figure 5.11(a)) we have to consider the boundary conditions in the two "bottom boundary"-faces A and B. If boundary condition type 1 (Dirichlet) or type 3 (semi-natural outflow) is prescribed in one of the two "bottom boundary"-faces then the following extrapolation is used:

$$V^2_{(1,1,-2)} = 2V^2_{(1,1,-1)} - V^2_{(1,1,0)} \,, \tag{5.64}$$

otherwise

$$V^2_{(1,1,-2)} = 2V^2_{(1,1,0)} - V^2_{(1,1,2)} \,. \tag{5.65}$$

It is clear that a similar procedure can be used for all virtual velocities in the "tangential" cells and "normal" half cell.

5.6 Treatment of boundary conditions at the corners of the region

With respect to the corners of the region, it is necessary to consider the boundary conditions carefully, because unknowns may be not present anymore. Let us investigate the four boundary conditions in this special case.

5.6.1 2D

For simplicity we restrict ourselves to the case of a boundary condition at the right side of the lower boundary in the computational region.

In the case of Dirichlet boundary conditions (type 1), no points at the right of the left boundary appear, hence no special precautions are necessary.

In the case of boundary conditions of type 2 we have to distinguish between "tangential" cells and normal half cells. Only tangential cells of tangential velocities not lying on another boundary are considered. As a consequence the last tangential cell is at a distance 1 from the boundary and no special treatment is necessary.
With respect to the normal half cell sketched in Figure 5.12 we have to be more careful.

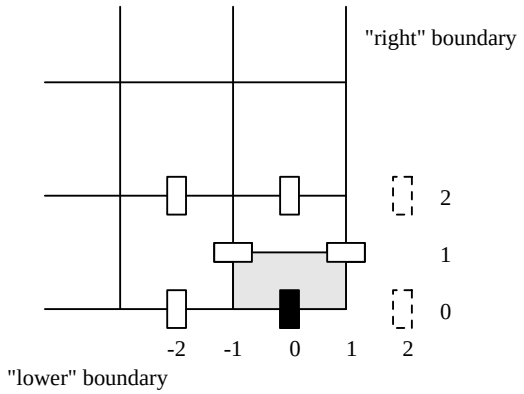


Figure 5.12: "normal" half-cell at the intersection of "lower" and right boundary.

The discretization of the convective terms using formulae (5.13), (5.17) and (5.18) introduces virtual velocities in the points (2,0) and (2,2). These virtual velocities are eliminated in the standard way by linear extrapolation using the value of V^2 at the right boundary if available and otherwise using the values $V^2_{(0,i)}$ and $V^2_{(-2,i)}$. Hence even if V^2 is given at the right boundary, we still use the interpolated values. This approach simplifies the treatment of the boundary conditions.

The stress tensor in this cell as treated in formulae (5.19), (5.20) does not introduce virtual unknowns at the right of the right boundary. Hence this part does not require a special treatment.

With respect to boundary conditions of type 3 and type 4 the standard procedure may be followed, provided virtual velocities are eliminated in the usual way. This is the case both for the tangential cells and the normal half cells.

5.6.2 3D

In the 3D-case we have to consider two kinds of corners, edges and vertices.

Edges We restrict ourselves to the case of boundary conditions at the edge left under ($\xi^2 = 0$ and $\xi^3 = 0$) in the computational domain. There are $4^2 = 16$ possible combinations to prescribe the boundary conditions, but only $\begin{pmatrix} 4+2-1 \\ 2 \end{pmatrix} = 10$ are really different, see Table 5.1.

combination	boundary condition type	
	left boundary ($\xi^2 = 0$)	bottom boundary ($\xi^3 = 0$)
(i)	1	1
(ii)	1	2
(iii)	1	3
(iv)	1	4
(v)	2	2
(vi)	2	3
(vii)	2	4
(viii)	3	3
(ix)	3	4
(x)	4	4

Table 5.1: Combinations of the boundary conditions for the edge left under ($\xi^2 = 0$ and $\xi^3 = 0$.)

Combination (i)

Since the normal velocities are given at the boundaries we have only to consider the "tangential" U^1-cell, see Figure 5.13.

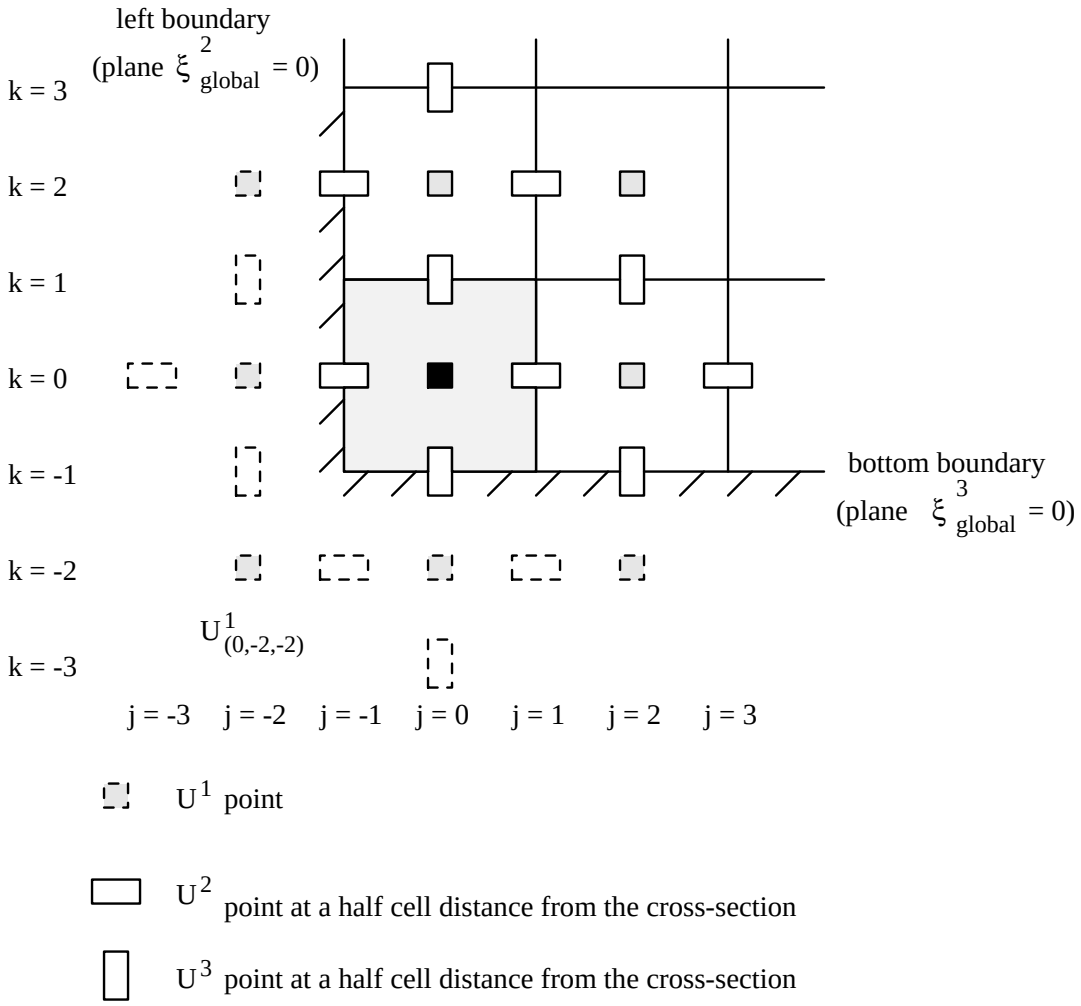


Figure 5.13: Cross-section over the "tangential" cell near the edge left under.

The virtual velocities are eliminated in the usual way [1] one exception that is for $V^1_{(0,-2,-2)}$.

[1]

$$
\begin{aligned}
V^{\alpha}_{(i,j,-2)} &= 2V^{\alpha}_{(i,j,-1)} - V^{\alpha}_{(i,j,0)} \quad \text{for} \quad \alpha = 1,2 \quad \text{and} \quad j \neq -2 \\
V^{\alpha}_{(i,-2,k)} &= 2V^{\alpha}_{(i,-1,k)} - V^{\alpha}_{(i,0,k)} \quad \text{for} \quad \alpha = 1,3 \quad \text{and} \quad k \neq -2 \\
V^{2}_{(i,-3,0)} &= 2V^{2}_{(i,-1,0)} - V^{2}_{(i,1,0)} \\
V^{3}_{(i,0,-3)} &= 2V^{3}_{(i,0,-1)} - V^{3}_{(i,0,1)} \ .
\end{aligned}
$$

This virtual quantity can be extrapolated in the following way:

$$V^1_{(0,-2,-2)} = 4V^1_{(0,-1,-1)} - 2V^1_{(0,-1,0)} - 2V^1_{(0,0,-1)} + V^1_{(0,0,0)} \tag{5.66}$$

or

$$V^1_{(0,-2,-2)} = 2V^1_{(0,-1,-1)} - V^1_{(0,0,0)} \cdot \tag{5.67}$$

Both equations have the same order of accuracy.

Combination (ii)

Here we have to consider the "tangential" cell and the "normal" half-cell at the bottom boundary, see Figures 5.13 and 5.14.

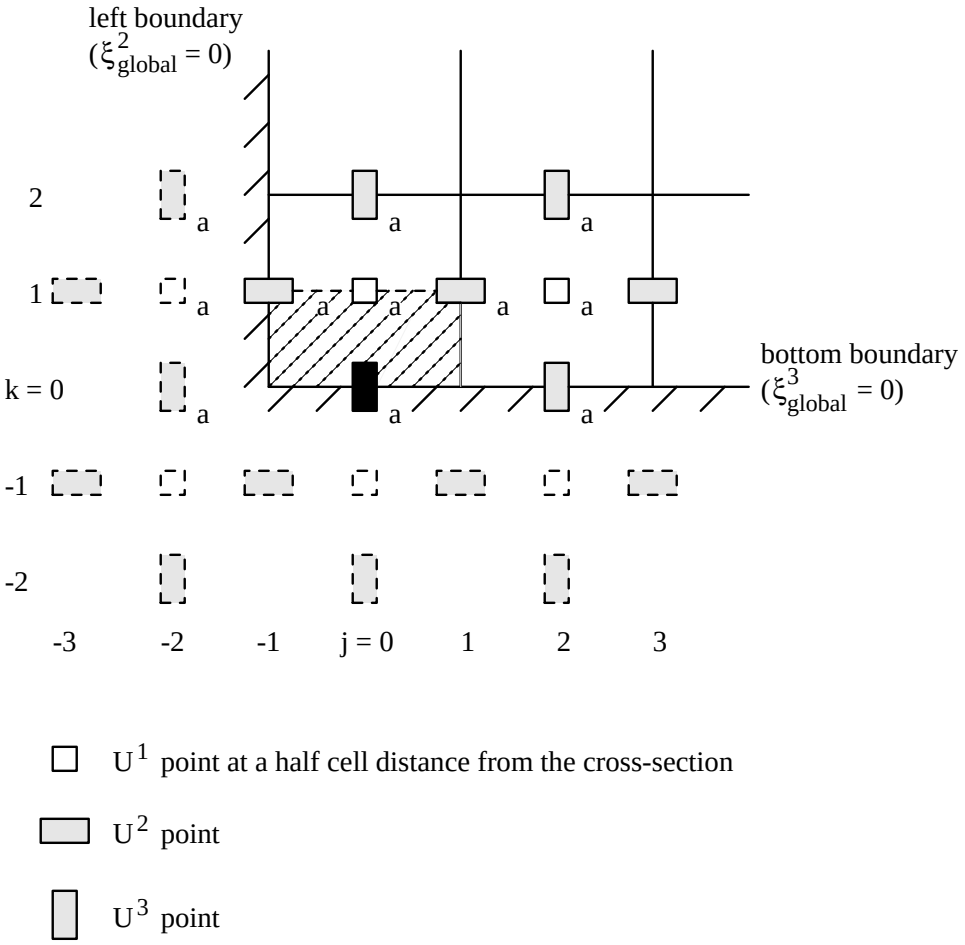


Figure 5.14: Cross-section over a "normal" U^3 half-cell.

"Tangential" U^1-cell.

The approach is here al most the same as the one prescribed in paragraph 5.2.2. Although there are more virtual unknowns they can be eliminated in the usual way. The $V^1_{(0,-2,-2)}$

velocity forms an exception in this case. We can only use equation (5.67) for the elimination of $V^1_{(0,-2,-2)}$, since V^1 is not prescribed at the bottom boundary.

"Normal" U^3 half-cell.
Only the velocities marked an a in Figure 5.14 appear in the discretization, since σ^{31}, σ^{32} and σ^{33} respectively U^1, U^2 and U^3 are given at the bottom boundary respectively left boundary. The approach is almost identical to the one given in paragraph 5.2.2, only the terms $\frac{1}{2}\frac{\rho}{\sqrt{g}}V^3V^2|_{(0,-1,0)}$ and $\frac{\partial U^1}{\partial \xi^2}|_{(0,0,-1)}$ are treated in a different way. The first term is completely known since V^2 and V^3 are prescribed at the left boundary. So this term can be transported to the right-hand side.

The derivative $\frac{\partial U^1}{\partial \xi^2}|_{(0,0,1)} = \frac{1}{2}(U^1_{(0,2,1)} - U^1_{(0,-2,1)})$, introduced by the stress tensor can be computed using the standard elimination rules, so:

$$\frac{\partial U^1}{\partial \xi^2}|_{(0,0,1)} = \frac{1}{2}(U^1_{(0,2,1)} - (2U^1_{(0,-1,1)} - U^1_{(0,0,1)}) \tag{5.68}$$

where

$$U^1_{(0,i,j)} = \frac{1}{2}(U^1_{(-1,i,j)} + U^1_{(1,i,j)}) \; . \tag{5.69}$$

Combination (iii)
The U^1, U^2 and σ^{33} are given at the bottom boundary and U^1, U^2 and U^3 at the left boundary, so it is necessary to consider besides the "tangential" cell the "normal" U^3 half-cell.
The "tangential" cell can be treated in almost the same way as the "tangential" cell in combination (i), see also paragraph 5.2.2.
The discretization for the "normal" U^3 half-cell is almost given in paragraph 5.2.2. But now there are more virtual unknowns and some of them: $V^1_{(0,-2,-1)}, V^2_{(0,-3,-1)}$ and $V^3_{(0,-2,-2)}$ can not be eliminated in the usual way. The virtual unknowns $V^1_{(0,-2,-1)}$ and $V^1_{(0,-3,-1)}$ and $V^3_{(0,-2,-2)}$ (see Figure 5.14) can be eliminated by using one of the following equations:

$$V^1_{(i,-2,-1)} = 4V^1_{(i,-1,0)} - 2V^1_{(i,0,0)} - 2V^1_{(i,-1,1)} + V^1_{(i,0,1)} \tag{5.70}$$

for $i = -1$ or 1,

$$V^2_{(0,-3,-1)} = 4V^2_{(0,-1,0)} - 2V^2_{(0,1,0)} - 2V^2_{(0,-1,1)} + V^2_{(0,1,1)} \; , \tag{5.71}$$

and

$$V^3_{(0,-2,-2)} = 4V^3_{(0,0,0)} - 2V^3_{(0,2,0)} - 2V^3_{(0,0,2)} + V^3_{(0,2,2)} \; . \tag{5.72}$$

Combination (iv)
Here we have only to consider the "tangential" cell, since the "normal" velocities are given (U^2 at the left boundary and U^3 at the bottom boundary). For the treatment of the "tangential" cell, we refer to paragraph 5.4.2. All virtual unknowns (see Figure 5.13) exept of $V^1_{(0,-2,-2)}$ are eliminated in the usual way. For $V^1_{(0,-2,-2)}$ we can use formula (5.67).

Combination (v)

The stresses σ^{21}, σ^{22} and σ^{23} are prescribed at the left boundary and σ^{31}, σ^{32} and σ^{33} at the botom boundary. So we have to consider two "normal' half-cells and the "tangential" U^1-cell. See paragraph 5.2.2, for the treatment of the "tangential" U^1 cell (Figure 5.15) and the "normal" half-cells (Figure 5.14). However, in this case there are some differences:
Firstly, there are more virtual unknowns, but they caus no extra problems, since they can be eliminated by using the standard rules.
Secondly, the discretization of the stress tensor $\sigma^{\alpha\beta}$ for the "tangential" U^1-cell produces a difference. In formula (5.28) not only $\sigma^{13}|_{(0,0,-1)}$ is given, but also $\sigma^{12}_{(0,0,-1)}$.

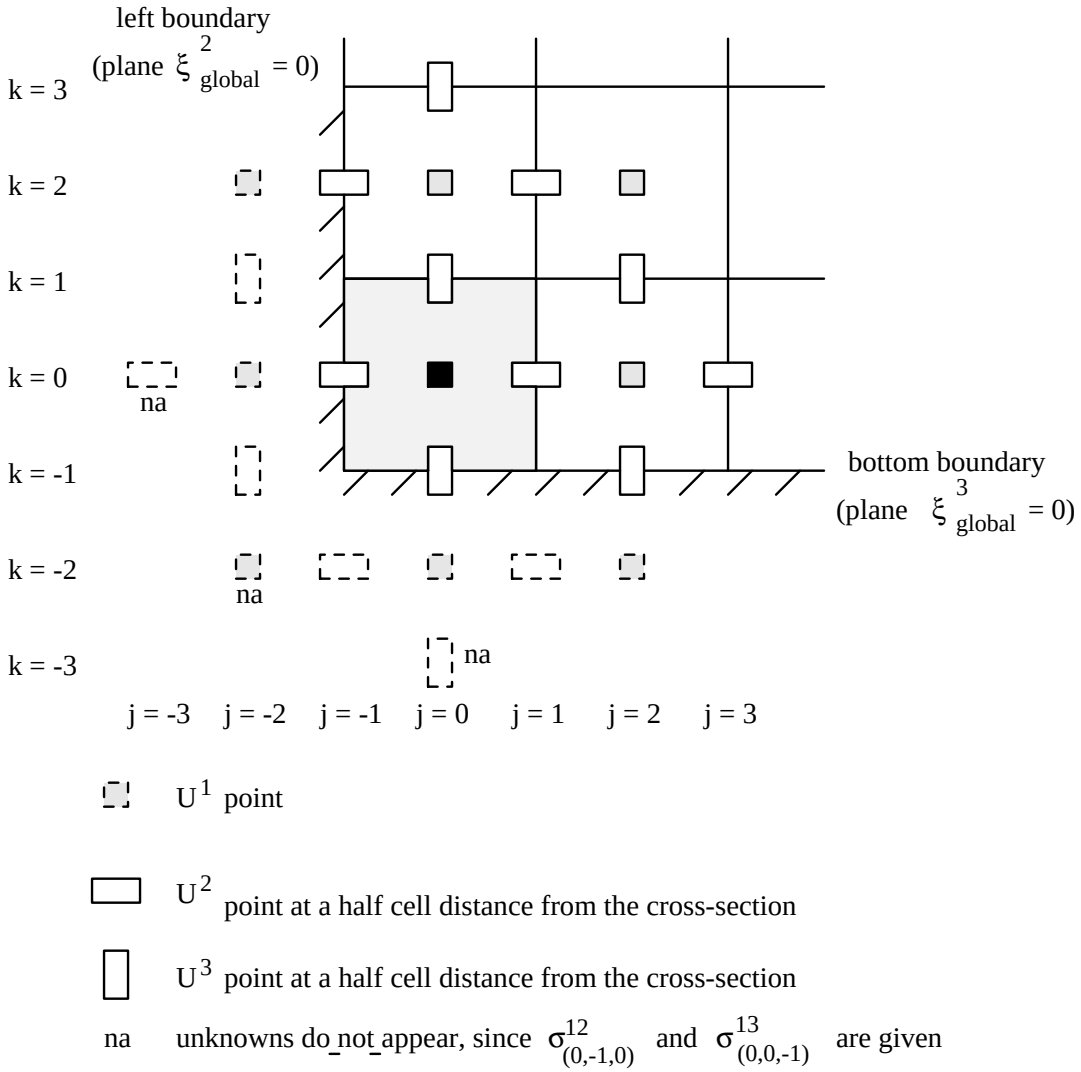


Figure 5.15: Cross-section over the "tangential" U^1-cell.

5.7 Boundary conditions for the convection-diffusion equations

The boundary conditions for the convection-diffusion type equations are much easier to implement than the boundary conditions for the velocity components.

In the case of Dirichlet boundary conditions it is sufficient to use linear extrapolation to eliminate virtual scalars. So for example in Figure 5.16 we use the following formulae:

$$T_{.,0} = 2T_{.,1/2} - T_{.,1} \tag{5.73}$$

for normal boundary points and

$$T_{0,0} = \frac{1}{2}(2T_{1,0} - T_{2,0}) + \frac{1}{2}(2T_{0,1} - T_{0,2}) \tag{5.74}$$

for the corner points. In the case of a Robbins boundary condition we follow van Kan et al

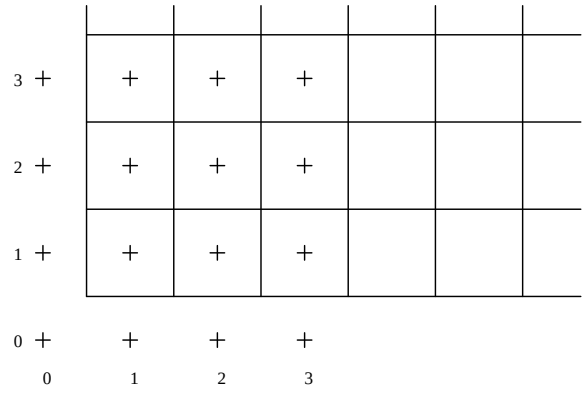


Figure 5.16: Cells for scalar quantities and corresponding virtual points.

(1991). This means that the Robbins boundary condition

$$k^{\alpha\beta}T_{,\beta}n_\alpha = b - \sigma T \tag{5.75}$$

is substituted in the diffusive term:

$$\begin{aligned}\int_\Omega -(k^{\alpha\beta}T_{,\beta})_{,\alpha}d\Omega &= -\int_\Gamma k^{\alpha\beta}T_{,\beta}n_\alpha d\Gamma \\ &= -\int_{\Gamma\backslash\Gamma_b} k^{\alpha\beta}T_{,\beta}n_\alpha d\Gamma - \int_{\Gamma_b}(b-\sigma T)d\Gamma\ ,\end{aligned} \tag{5.76}$$

where Γ_b is the boundary at which the robbins boundary condition is given. Virtual scalars T are eliminated in the usual way.

5.8 Wall functions

5.8.1 Introduction

In this section we shall consider the boundary conditions for the momentum and turbulence transport equations along a solid wall.

In the case of laminar flows at the wall the no-slip conditions are directly applied. The situation is much more complicated if turbulent flow is calculated. The reason for this is twofold:

- in the near–wall region there are very steep gradients of the flow properties and so the need for fine mesh is required to resolve the wall layer properly in a numerical scheme;
- there is an important role of molecular viscosity near the wall and hence the "high Reynolds number" version of the $k - \varepsilon$ model can not be applied in the near–wall region.

However, specification of the boundary conditions right at the wall is not necessary because empirical laws of sufficient generality are available. One of these laws is the "law of the wall", which can be used to provide near–wall boundary conditions for the momentum and turbulence transport equations, rather than on the wall itself. Other advantage of the "law of the wall" is that its allows some additional empirical information in special cases, for example the roughness of the wall.
More details can be found in Zijlema (1993).

5.8.2 Boundary conditions for the momentum equations

We shall consider the control volumes adjacent to the wall. Figure (5.17) shows a scalar point P whose associated volume is bounded on the south side by the wall.

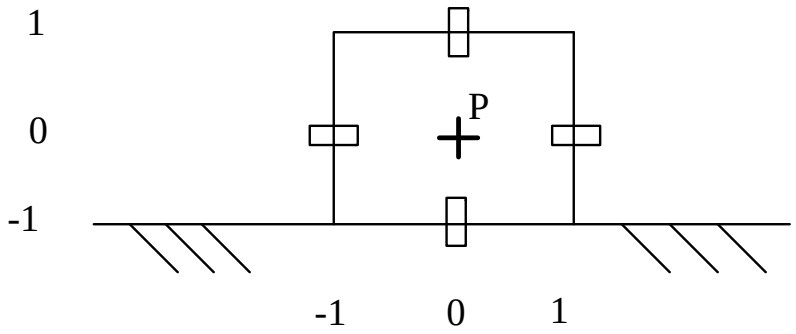


Figure 5.17: A scalar cell adjacent to a wall

The following boundary conditions for the momentum equations will be used:

$$\boldsymbol{u} \cdot n = 0 \text{ and } S^{nt} = \tau_w \tag{5.77}$$

where $\boldsymbol{u} \cdot \boldsymbol{n}$ is the physical velocity component normal to the wall, S^{nt} is the physical shear stress tangent to the wall and τ_w is the wall shear stress. This type of boundary conditions is the same as discussed in section 5.4.
The wall shear stress τ_w can be computed in point P from the "law of the wall"

$$\tau_w = \frac{\rho c_\mu^{1/4} \kappa \sqrt{k_P}}{\ln(E Y_P^+)} \boldsymbol{u} \cdot \boldsymbol{t}_P \tag{5.78}$$

where

$$Y_P^+ \equiv \frac{\rho c_\mu^{1/4} Y_P \sqrt{k}}{\mu} \tag{5.79}$$

and Y_P is the normal distance of the point P from the wall, $\boldsymbol{u} \cdot \boldsymbol{t}$ is the physical velocity component along the wall, κ is the Von Kármán constant (≈ 0.4) and E is a roughness parameter, approximately equal to 9.0 for a smooth wall. In the above $\boldsymbol{u} \cdot \boldsymbol{t}$ and k are evaluated at the previous iteration. However, a distinction is first made between the first computational points associated with scalar volumes, which lie in the viscous region and those which lie in the turbulent region. The limiting value is $Y_v^+ = 11.3$ which is determined by matching the linear and logarithmic velocity profiles. Thus, for $Y_P^+ > 11.3$, τ_w is calculated from (5.78). For $Y_P^+ < 11.3$ the wall shear stress is given by

$$\tau_w = \mu \frac{\boldsymbol{u} \cdot \boldsymbol{t}_P}{Y_P} \tag{5.80}$$

The tangential velocity along the wall expressed in terms of contravariant components reads:

$$\boldsymbol{u} \cdot \boldsymbol{t} = \frac{g_{t\alpha}}{\sqrt{g_{tt}}} U^\alpha \tag{5.81}$$

The contravariant velocity components U^α at point P is computed by linear interpolation using the neighbouring points. Finally, the normal distance Y_P is evaluated as follows, see Figure (5.18):

$$\cos\phi = \frac{AB.BC}{|AB|.|BC|}$$
$$\sin\phi = \sqrt{1 - \cos^2\phi}$$
$$Y_P = \frac{1}{2}|BC| \sin\phi$$

where $A = \boldsymbol{x}_{(i,j)}$, $B = \boldsymbol{x}_{(i+1/2,j)}$ and $C = \boldsymbol{x}_{(i+1/2,j+1/2)}$. The co-ordinates of points B and C are obtained by linear interpolation in the obvious way.

5.8.3 Boundary conditions for the turbulence equations

Boundary conditions are required for k and ε to solve the $k - L$ or $k - \varepsilon$ equations. The turbulent energy k at the first grid point away from the wall is calculated by the transport equation (4.33). However, to ensure an accurate numerical representation of near-wall effects,

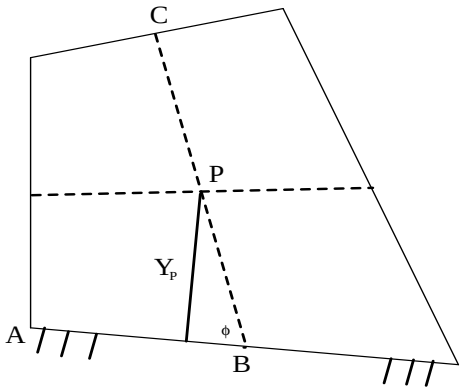


Figure 5.18: Calculation of distance between near wall point and the wall

a special care is needed in evaluating the source terms, i.e. the production and dissipation rate, of k-equation in wall-adjacent cells. Let us consider the production term of the k-equation. Since the near-wall flow is assumed of the Couette type, the dominant contribution to the production is

$$P = \tau_w \frac{\partial \boldsymbol{u} \cdot \boldsymbol{t}}{\partial \boldsymbol{n}} \tag{5.82}$$

To discretize this term we use the midpoint rule. The volume-integrated production of k may be approximated by

$$\int_{\Omega_h} P d\Omega \approx \tau_w \frac{\boldsymbol{u} \cdot \boldsymbol{t}_{(0,0)}}{Y_P} \sqrt{g}_{(0,0)} \tag{5.83}$$

where the local numbering of Figure (5.17) is used, and τ_w is evaluated according to Y_P^+ value. The dissipation term in the k equation is integrated in the same way using the following relation for ε:

$$\varepsilon = \frac{c_\mu^{3/4} k^{3/2}}{\kappa Y} \tag{5.84}$$

See Launder & Spalding (1974). This assumption leads to

$$\int_{\Omega_h} \rho \varepsilon d\Omega \approx \rho c_\mu^{3/4} k_P^{3/2} \frac{\ln(E Y_P^+)}{\kappa Y_P} \sqrt{g}_{(0,0)} \tag{5.85}$$

when $Y_P^+ > 11.3$. If $Y_P^+ < 11.3$ then

$$\int_{\Omega_h} \rho \varepsilon d\Omega \approx \rho c_\mu^{3/4} k_P^{3/2} \frac{Y_P^+}{Y_P} \sqrt{g}_{(0,0)} \tag{5.86}$$

The boundary condition $\partial k / \partial \boldsymbol{n} = 0$ for k-equation has been applied at the wall.
Finally, the ε-equation is not solved in wall-adjacent cells, because of its inapplicability there. Instead, the expression (5.84) is used for the evaluation of dissipation rate in the first computational point.

6 Time-discretization

6.1 Introduction

After application of the space discretization of momentum and transport equations and the implementation of the boundary conditions as described in chapter 5, the discretized Navier-Stokes equations in the time-domain read:

$$\boldsymbol{M}\dot{\boldsymbol{V}} + \boldsymbol{S}(\boldsymbol{V}) + \boldsymbol{G}\boldsymbol{P} = \boldsymbol{F} \ , \tag{6.1}$$

$$\boldsymbol{D}\boldsymbol{V} = \boldsymbol{0} \ , \tag{6.2}$$

$$\boldsymbol{M}_i\dot{\boldsymbol{T}}_i + \boldsymbol{S}_i\boldsymbol{T}_i = \boldsymbol{F}_i \ . \tag{6.3}$$

In (6.1) - (6.3) $\boldsymbol{V}$ denotes the vector of velocity unknowns, $\boldsymbol{P}$ the vector of pressure unknowns and $\boldsymbol{T}_i$ the vector of the i^{th} scalar unknowns.

The matrix $\boldsymbol{M}$ is a diagonal matrix containing the value of ρ in the centroids on the diagonal. The expression $\boldsymbol{S}(\boldsymbol{V})$ represents the discretization of the deviatoric stress tensor and the convective terms. $\boldsymbol{G}\boldsymbol{P}$ represents the discretization of the pressure gradient, and $\boldsymbol{F}$ the discretization of the source terms.

The equation $\boldsymbol{D}\boldsymbol{V} = \boldsymbol{0}$ denotes the discretized continuity equation.
Equation (6.3) stands for the general discretized convection-diffusion equation, where again $\boldsymbol{M}_i$ is a diagonal matrix, and $\boldsymbol{F}_i$ the discretization of the source term.
$\boldsymbol{S}_i\boldsymbol{T}_i$ represents the discretization of the convective and diffusive terms. In fact this term may also be non-linear, but in our program it is treated as if it is linear.
The first important decision that has been made is that momentum equations and continuity equation are coupled, but that all scalar convection-diffusion equations are decoupled from the momentum equations and the other convection-diffusion equations.

In fact this means that for each time step first the momentum and continuity equation are solved and then each scalar equation separately in the sequence given by their index number. So per time-step:

Solve $\boldsymbol{V}$ and $\boldsymbol{P}$

Solve T_1
Solve T_2
$\vdots$

The time discretization is performed with a standard technique for the solution of ordinary differential equations. At this moment only one type of time-solver is present: the so-called θ method.

6.2 The θ-method

The standard θ-method applied to (6.1), (6.2) reads:

$$\begin{aligned} \boldsymbol{M}\frac{\boldsymbol{V}^{n+1}-\boldsymbol{V}^{n}}{\Delta t} &+ \theta\boldsymbol{S}(\boldsymbol{V}^{n+1})+(1-\theta)\boldsymbol{S}(\boldsymbol{V}^{n})+\theta\boldsymbol{G}\boldsymbol{P}^{n+1}+(1-\theta)G\boldsymbol{P}^{n} \qquad (6.4)\\ &= \theta\boldsymbol{F}^{n+1}+(1-\theta)\boldsymbol{F}^{n} \\ D\boldsymbol{V}^{n+1} &= \boldsymbol{0}\ , \quad 0\le\theta\le 1\ . \qquad (6.5) \end{aligned}$$

$n+1$ denotes the new and n the preceding time-level.
To solve (6.3), (6.4) it is necessary to linearize the term $\boldsymbol{S}(\boldsymbol{V}^{n+1})$. In ISNaS, the convective terms are linearized by a Newton linearization as given in formula (4.7). Coefficients that depend on the solution, like for example the viscosity, are evaluated at the preceding time-level.

The θ-method is unconditionally stable for $0.5\le\theta<1$. In the range $0\le\theta<0.5$ a time-step restriction is necessary. At this moment $\theta<0.5$ has not been tested.

Practical implementation:

Instead of solving (6.4), (6.5) immediately, we introduce an intermediate level $n+\theta$ by:

$$\begin{aligned} \boldsymbol{V}^{n+\theta} &= \theta\boldsymbol{V}^{n+1}+(1-\theta)\boldsymbol{V}^{n} \\ \boldsymbol{P}^{n+\theta} &= \theta\boldsymbol{P}^{n+1}+(1-\theta)\boldsymbol{P}^{n} \qquad (6.6)\\ \boldsymbol{F}^{n+\theta} &= \theta\boldsymbol{F}^{n+1}+(1-\theta)\boldsymbol{F}^{n} \end{aligned}$$

If we assume that $S(\boldsymbol{V})$ is linearized, i.e. can be written as $\boldsymbol{S}(\boldsymbol{V}^{n+1})\simeq\boldsymbol{A}(\boldsymbol{V}^{n})+\boldsymbol{B}(\boldsymbol{V}^{n})\boldsymbol{V}^{n+1}$, then (6.4) reduces to:

$$\begin{aligned} \boldsymbol{M}\frac{\boldsymbol{V}^{n+1}-\boldsymbol{V}^{n}}{\Delta t} &+ \theta\boldsymbol{B}(\boldsymbol{V}^{n})\boldsymbol{V}^{n+1}+(1-\theta)\boldsymbol{B}(\boldsymbol{V}^{n})\boldsymbol{V}^{n}+\theta\boldsymbol{G}\boldsymbol{P}^{n+1}+(1-\theta)\boldsymbol{G}\boldsymbol{P}^{n} \\ &+ \boldsymbol{A}(\boldsymbol{V}^{n})=\theta\boldsymbol{F}^{n+1}+(1-\theta)\boldsymbol{F}^{n} \qquad (6.7) \end{aligned}$$

Substitution of (6.6) into (6.7) and (6.5) gives

$$\boldsymbol{M}\frac{\boldsymbol{V}^{n+\theta}-\boldsymbol{V}^{n}}{\theta\Delta t}+\boldsymbol{B}(\boldsymbol{V}^{n})\boldsymbol{V}^{n+\theta}+\boldsymbol{G}\boldsymbol{P}^{n+\theta}=\boldsymbol{F}^{n+\theta}-\boldsymbol{A}(\boldsymbol{V}^{n}) \qquad (6.8)$$

$$\boldsymbol{D}\boldsymbol{V}^{n+\theta}=0 \qquad (6.9)$$

From (6.6) it then follows that:

$$\begin{aligned} \boldsymbol{V}^{n+1} &= \frac{1}{\theta}(\boldsymbol{V}^{n+\theta}-(1-\theta)\boldsymbol{V}^{n}) \qquad (6.10)\\ \boldsymbol{P}^{n+1} &= \frac{1}{\theta}(\boldsymbol{P}^{n+\theta}-(1-\theta)\boldsymbol{P}^{n}) \end{aligned}$$

Once the momentum equations have been solved for t^{n+1}, each of the scalar convection-diffusion equations is solved for one time step. Exactly the same θ method with practical implementation is used as for the momentum equations. Quantities already computed, like the velocity are substituted in these equations, thus improving the stability.

6.3 Time discretization of turbulence equations

For each time step first the momentum and continuity equations are solved and then each scalar equation separately. Finally, the turbulence equations are solved. It should be noted that for all equations the eddy viscosity μ_t is evaluated at old time level. This is also true for the momentum equations.

Normally, the equations for k and ε are coupled, in other words, k appears in the ε equation and vice versa. Bearing in mind that each convection-diffusion equation is solved separately, the equations for k and ε will be treated as decoupled equations, in the following way: for each time step first the equation for k is solve using the updated velocity components U^α and non-updated turbulence quantities, i.e. k, ε and μ_t. The same holds for the equation for ε, which is solved after k.

The right hand side of equations (4.33)-(4.34) can rise considerable problems, because it represents a function of the solution U^α, k and ε and is non-linear. We are compelled to use fully-implicit scheme enabling such a linearization. In the present solver, a Newton linearization is use. The dissipation term in k-equation in both $k-L$ and $k-\varepsilon$ model is evaluated as follows

$$\varepsilon = \rho c_\mu \frac{k^2}{\mu_t} \tag{6.11}$$

This non-linear expression is linearized on the following way:

$$\rho c_\mu \left(\frac{2\, k^{\text{new}}\, k^{\text{old}} - (k^{\text{old}})^2}{\mu_t}\right) \tag{6.12}$$

The same holds for the destruction term in the ε-equation:

$$\frac{2\, \varepsilon^{\text{new}}\, \varepsilon^{\text{old}} - (\varepsilon^{\text{old}})^2}{k^{\text{old}}} \tag{6.13}$$

Finally, the functions c^*, $K^{\alpha\beta}$, D and f^* for both equations are given by:

- k-equation:

$$c^* = \rho$$

$$K^{\alpha\beta} = g^{\alpha\beta}\, \frac{\mu_t^{\text{old}}}{\sigma_k}$$

$$D = 2\rho^2 c_\mu\, \frac{k^{\text{old}}}{\mu_t^{\text{old}}}$$

$$f^* = P + \rho^2 c_\mu\, \frac{(k^{\text{old}})^2}{\mu_t^{\text{old}}}$$

- ε-equation:

$$c^* = \rho$$

$$K^{\alpha\beta} = g^{\alpha\beta}\,\frac{\mu_t^{\text{old}}}{\sigma_\varepsilon}$$

$$D = 2c_{2\varepsilon}\rho\,\frac{\varepsilon^{\text{old}}}{k^{\text{old}}}$$

$$f^* = \frac{\varepsilon^{\text{old}}}{k^{\text{old}}}\,(c_{1\varepsilon}P + c_{2\varepsilon}\rho\varepsilon^{\text{old}})$$

On the grounds of the foregoing considerations the fully-implicit time scheme, i.e. $\theta = 1$, must be used in the present solver.

It can be shown that the discrete equations for k and ε, which is a result of applying upwind scheme for convection terms and Newton linearization for non-linear source terms, yield positive values for k and ε Zijlema (1993).

7 Pressure correction

7.1 Introduction

An essential difficulty in the solution of the coupled momentum equations and continuity equation (6.1), (6.2) or its (time discretized form (for example (6.8), (6.9), is the absence of the pressure in the continuity equation. If we consider the system of equations as one large system of linear equations to be solved, this means that in the part corresponding to the continuity equations we have zeros at the main diagonal. Formally equations (6.8), (6.9) may be written as:

$$\begin{bmatrix} \boldsymbol{S} & \boldsymbol{G} \\ \boldsymbol{D} & \boldsymbol{0} \end{bmatrix} \begin{bmatrix} \boldsymbol{V}^{n+\theta} \\ \boldsymbol{P}^{n+\theta} \end{bmatrix} = \begin{bmatrix} \boldsymbol{F}_1^{n+\theta} \\ \boldsymbol{F}_2^{n+\theta} \end{bmatrix} , \tag{7.1}$$

where $\boldsymbol{F}_2^{n+\theta}$ is only non-zero if non-zero Dirichlet boundary conditions for the velocity are prescribed.

The solution of systems of equations of the form (7.1) is in general more difficult for a linear solver than the solution of equations arising from the discretization of standard convection-diffusion equations. There are several ways to solve this problem. One of the possible ways is to perturb the continuity equation. This leads to methods like the penalty method or Uzawa iterations. An alternative way to solve the problem is formed by projection methods. In these methods first the pressure at the new level is estimated, for example by the old pressure, and then the momentum equations are solved yielding an intermediate velocity field. By projecting this velocity onto the space of divergence-free vector fields a new velocity and pressure may be computed. An important representant of this class is the so-called pressure-correction method, which will be treated in 7.2.

7.2 The pressure-correction method

The pressure-correction method as implemented in the ISNaS incompressible code is the one described in van Kan et al (1991). Starting point is the θ-method formulated by (6.4), (6.5) or the variant (6.8), (6.9).

Following van Kan et al (1991) we define an intermediate velocity $\boldsymbol{V}^*$ by:

$$\boldsymbol{M}\frac{\boldsymbol{V}^* - \boldsymbol{V}^n}{\theta\Delta t} + \boldsymbol{B}(\boldsymbol{V}^n)\boldsymbol{V}^* + \boldsymbol{G}\boldsymbol{P}^n = \boldsymbol{F}^{n+\theta} - \boldsymbol{A}(\boldsymbol{V}^n) \tag{7.2}$$

$\boldsymbol{V}^*$ must be such that the boundary conditions at $t = t^n + \theta\Delta t$ are satisfied. In the case of prescribed normal velocities this means that the corresponding rows in the matrix G contain zeros.

Subtraction of (7.2) from (6.8) gives

$$M\frac{\boldsymbol{V}^{n+\theta} - \boldsymbol{V}^*}{\theta\Delta t} = -\boldsymbol{G}(\boldsymbol{P}^{n+\theta} - \boldsymbol{P}^n) , \tag{7.3}$$

where the term $\boldsymbol{B}(V^n)(\boldsymbol{V}^{n+\theta} - \boldsymbol{V}^*)$ has been neglected.
Application of (6.9) to (7.3) gives

$$-\boldsymbol{D}\boldsymbol{V}^* = -\theta\Delta t \boldsymbol{D}\boldsymbol{M}^{-1}\boldsymbol{G}(\boldsymbol{P}^{n+\theta} - \boldsymbol{P}^n)\,, \tag{7.4}$$

which is a Laplacian-type equation for the pressure correction. Once $\boldsymbol{P}^{n+\theta}$ has been computed $\boldsymbol{V}^{n+\theta}$ follows from (7.3):

$$\boldsymbol{V}^{n+\theta} = \boldsymbol{V}^* - \theta\Delta t \boldsymbol{M}^{-1}\boldsymbol{G}(\boldsymbol{P}^{n+\theta} - \boldsymbol{P}^n) \tag{7.5}$$

Remark: the matrix $-\boldsymbol{D}\boldsymbol{M}^{-1}\boldsymbol{G}$ is in general non-symmetrical.

8 The linear solver

8.1 Introduction

The discretization of the incompressible Navier-Stokes equations in general curvilinear coordinates is described in the foregoing sections. The space discretization consists of a finite volume technique on a structured grid. The motivation for these choices is that we want to solve large two and three dimensional problems. In these problems it is important to obtain fast iterative methods to solve the discretized equations. This is easier using a finite volume technique instead of a finite element technique. Finally the structured grid enables us to develop a good implementation of the methods on vector computers.

The linear systems to be solved are Vuik (1992), Vuik (1993):

the momentum equations

$$M^{n+1}u^{n+1} = f^{n+1} \quad , \quad u^{n+1} = \begin{pmatrix} u_1^{n+1} \\ u_2^{n+1} \\ u_3^{n+1} \end{pmatrix} \quad ,$$

the pressure equation

$$P\Delta p^{n+1} = g^{n+1} \quad , \quad \Delta p^{n+1} = p^{n+1} - p^n \quad ,$$

and eventually one or more transport equations:

transport equations

$$\begin{aligned} C_1^{n+1} \quad c_1^{n+1} \quad &= \quad d_1^{n+1} \quad , \\ &\vdots \\ C_k^{n+1} \quad c_k^{n+1} \quad &= \quad d_k^{n+1} \quad . \end{aligned}$$

Suppose n_i is the number of grid points in the x_i-direction, where we take $n_3 = 1$ for a 2-dimensional problem. The pressure and transport matrices have $n_1 \cdot n_2 \cdot n_3$ rows and columns. The dimension of the momentum matrix is $2 \cdot n_1 \cdot n_2$ in 2-dimensional problems and $3 \cdot n_1 \cdot n_2 \cdot n_3$ in 3-dimensional problems.

For the structure of the matrices in 2-dimensions we refer to Vuik (1992) and Vuik (1993). In the 3-dimensional case the nonzero structure is symmetric for all matrices. In 3 dimensions the structure of the pressure equation is given in Figure 8.1.
Note that the nonzero structure is symmetric. The momentum matrix can be partitioned in the following form:

$$\begin{pmatrix} M_{11} & M_{12} & M_{13} \\ M_{21} & M_{22} & M_{23} \\ M_{31} & M_{32} & M_{33} \end{pmatrix} \begin{pmatrix} u_1 \\ u_2 \\ u_3 \end{pmatrix} = \begin{pmatrix} f_1 \\ f_2 \\ f_3 \end{pmatrix} \quad .$$

The structure of M_{ii}, $i = 1, 2, 3$ is the same as for the pressure equations. The off-diagonal blocks contain 16 non zero diagonals. The non zero structure of the momentum matrix is non symmetric. To illustrate this we give M_{12} and M_{21} in Figures 8.2 and 8.3 and note the non

zero structure of M_{12} is not equal to the non zero structure of M_{21}^T.

In the following table we summarize the number of non zero elements in some matrices.

	2D	3D
pressure matrix	$9 \cdot n_1 \cdot n_2$	$19 \cdot n_1 \cdot n_2 \cdot n_3$
momentum matrix	$13 \cdot 2 \cdot n_1 \cdot n_2$	$51 \cdot 3 \cdot n_1 \cdot n_2 \cdot n_3$

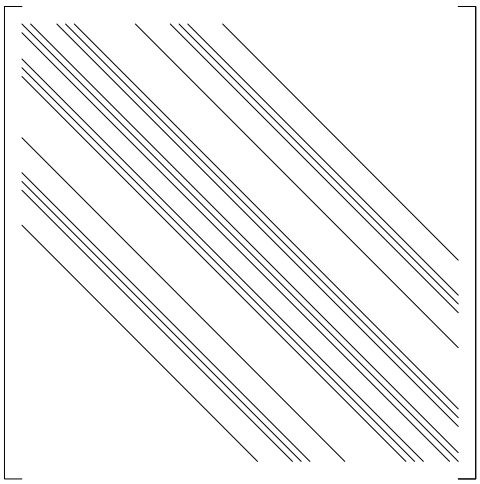


Figure 8.1: The pressure matrix P

In three dimensions the momentum matrix is much larger than the pressure matrix. The ratio in 2D is equal to $\frac{13 \cdot 2}{9} = 3$ whereas the ration in 3D is equal to $\frac{51 \cdot 3}{19} = 8$.

So in 3D a momentum matrix times vector is 8 times as expensive as a pressure matrix times vector.

The momentum matrix and the transport matrix depend on the time t. In many problems the pressure matrix is independent of the time. However, this property of the pressure matrix is not used in the current implementation.

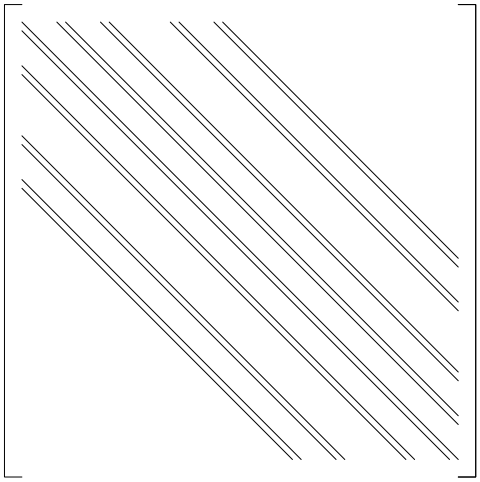


Figure 8.2: The momentum-matrix M_{12}

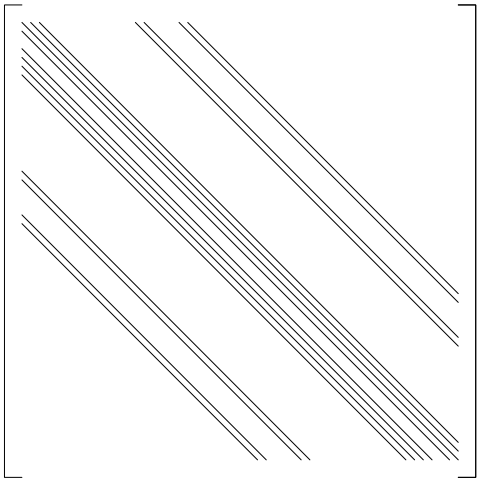


Figure 8.3: The momentum-matrix M_{21}

8.2 Survey of iterative methods

The systems given in Section 8.1 are solved with iterative methods of CG-type. All the methods used in ISNaS can be applied to unsymmetric matrices. The methods used in ISNaS are:

LSQR
This is a stable implementation of CG applied to the normal equations Paige et al (1982).

CGS
CGS is an iterative method based on the Bi-Lanczos algorithm Sonneveld (1989).

GMRES
An iterative method, which computes an approximation with a minimal residual Saad et al (1986).

GMRESR
A method based on GMRES, but in general cheaper with respect to work and memory Van der Vorst et al (1993). In Table 8.2 we summarize the properties of the iterative methods. This table only gives an indication of the properties. So in many experiments the results

properties	bad ⟵		good ⟶	
memory	GMRES	GMRESR	CGS	LSQR
robustness	CGS	GMRES	GMRESR	LSQR
CPU-time	LSQR	GMRES	CGS	GMRESR

Table 8.2: Properties of the iterative methods

agree with Table 8.2. However, for specific problems the results may be different.
Stopping criteria
For iterative methods it is necessary to specify a stopping criterion. In general the norm of the residual: $\|r_k\|_2 = \|b - Ax_k\|_2$ is easy to obtain. So all our stopping criteria are based on $\|r_k\|_2$. For the different equations we recommend different stopping criteria. For the details we refer to Vuik (1992), p. 8 for the momentum equations, Vuik (1992), p.13 for the pressure equation, and Vuik (1992), p. 15 for a transport equation.
Starting vector
Finally we have to choose a starting vector for the iterative methods. Since we solve the systems for every timestep, the solution of the foregoing timestep is in general a good starting vector. For the details we refer to Vuik (1992), p. 6, 7 for the momentum equations, Vuik (1992), p. 13 for the pressure equation, and Vuik (1992), p. 15 for a transport equation.

8.3 Preconditioning

In many applications, iterative methods are combined with a preconditioner Meijerink et al (1977). It is a well known fact that a good preconditioner is very important in order to obtain fast iterative methods. The preconditioners used in ISNaS are based on incomplete LU decompositions. In such a preconditioner, one constructs a lower triangular matrix L and an upper triangular matrix U, where L and U have a prescribed nonzero pattern, and LU is a good approximation of A. The iterative methods can be applied to

$$U^{-1}L^{-1}Ax = U^{-1}L^{-1}b\,, \tag{8.1}$$

$$AU^{-1}L^{-1}y = b\,, \tag{8.2}$$

or

$$L^{-1}AU^{-1}y = L^{-1}b\,. \tag{8.3}$$

We call equation (8.1) a preconditioned system and equation (8.2) a postconditioned system. The final equation is only used in combination with the Eisenstat implementation Eisenstat (1981). In general, the convergence behaviour of a Krylov type iterative method depends on the eigenvalue distribution of the matrix. In the three equations given above the eigenvalues of the product-matrices are the same. So the convergence behaviour is approximately the same when we use (8.1), (8.2), or (8.3). A small advantage of a postconditioned system is that the norm of a residual is not influenced by the matrices L and U (compare Vuik (1992), p. 12, 13).

Below we give a short description of the preconditioners used in ISNaS.

Diagonal scaling
A diagonal preconditioner is obtained by choosing $L = I$ and $U = diag\ (A)$. This is a cheap preconditioner with respect to memory and can be used in combination with vector and parallel computers. For most problems the gain in the number of iterations is small.

ILUD
For this preconditioner we construct $LD^{-1}U$ as an approximation of A. To obtain L, D, and U we use the following rules Van der Vorst (1981):

- $diag\,(L) = diag\,(U) = D$;
- the off-diagonal parts of L and U are equal to the corresponding parts of A;
- $diag\,(LD^{-1}U) = diag\,(A)$.

The third rule can be replaced by the following:

$$\text{rowsum}\,(LD^{-1}U) = \text{rowsum}\,(A) \quad \text{for every row}\,,$$

which leads to the MILU preconditioning. We always use an average of ILUD and MILUD. This preconditioner is also cheap with respect to memory. It costs two extra vectors, one

for D and the other one for D^{-1}. Using the Eisenstat implementation we are able to save one matrix vector product per iteration. In the ISNaS program ILUD preconditioner means application of the iterative method to (8.3) and not to (8.1), which is done in the other preconditioners. Multiplication with L^{-1} and U^{-1} leads to recurrencies. So these parts do not run in vector speed on a vector computer.

ILU
This preconditioner is only used for the pressure and transport equations. The matrices L and U are constructed such that LU approximates A and satisfies the following rules:

- $diag\,(L) = I$;
- the structure of L and U is comparable to the structure of A;
- if $a_{ij} \neq 0$ then $(LU)_{ij} = a_{ij}$.

Again the last rule for $i = j$ can be repaced by

- rowsum (LU) = rowsum (A),

which leads to MILU. We always use an average of ILU and MILU. The convergence behaviour of an iterative method combined with MILU is in general beter than a combination with MILUD. A disadvantage is that extra memory space is needed to store L and U. The amount of extra memory is the same as the amount of memory to store A. Furthermore it is impossible to save a matrix vector product per iteration (compare the Eisenstat implementation). From our experiments we conclude that if the memory space is available than it is better to use MILU.

Memory space
During the solution of the pressure or transport equation the memory space of the momentum matrix is available. For this reason we always use the MILU preconditioner to solve the pressure and transport equations, and the MILUD preconditioner for the momentum equations.

Vectorization
Due to the recurrencies, the multiplication of L^{-1} or U^{-1} for MILUD or MILU runs in scalar speed on a vector computer. In the ISNaS program the loops are rewritten in such a way that they run in vectorspeed Ashcraft (1988). Note that the rewritten loops use indirect adressing and are much shorter than the original loops. On the Convex C3840 this leads to good results.

8.4 Concluding remarks

Not all the combinations described in the foregoing sections are implemented. In Van Nooyen (1993) all the implemented combinations are summarized.

9 Post-processing

The ISNaS incompressible program computes the fluxes V^α in the midside points of the cells and the scalars in the centroids. For post-processing purposes these quantities are needed in the vertices of the cells. For that reason it is necessary to interpolate (or at the boundary extrapolate) the computed values to the vertex points. Numerical examples have shown that a straighforward interpolation in the computational space is not accurate enough. For that reason a weighted approach, taking into account the distances in physical space, is necessary. In the following sections we consider the interpolation and backtransformation applied both for scalar quantities and for the fluxes.

9.1 Interpolation of scalars in 2D

The scalar unknowns are positioned in the centroids of the cells. In order to interpolate these values to the vertices a weighted mean value of the four surrounding cells is used. Figure 9.1 sketches a typical example. In this figure point i, j is the vertex in which the interpolated

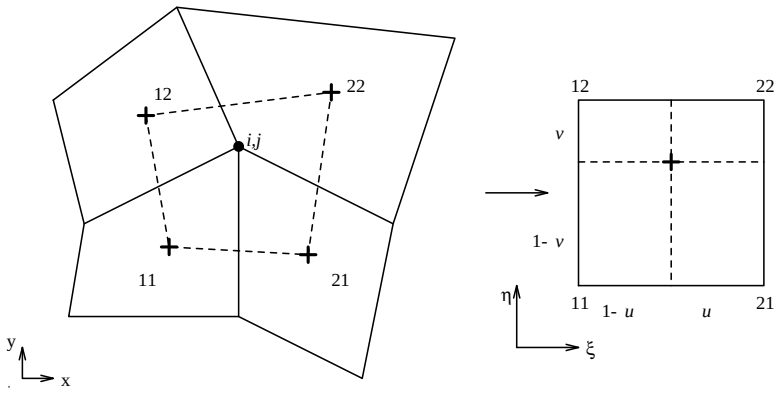


Figure 9.1: Vertex i, j with four surrounding cells and mapping of quadrilateral formed by centroids on to a square.

values must be computed. This point is part of 4 cells with centroids 11, 21, 12 and 22. In order to compute the interpolated value, the quadrilateral spanned by the 4 centroids is mapped onto a unit square (0, 1) × (0,1) by a bilinear mapping as is usual in finite elements. So

$$\begin{bmatrix} x \\ y \end{bmatrix} = \sum_{i,j=1}^{2} \lambda_i(\xi)\lambda_j(\eta) \begin{bmatrix} x \\ y \end{bmatrix}_{ij} , \tag{9.1}$$

with $\lambda_1(\xi) = 1 - \xi, \;\; \lambda_2(\xi) = \xi$

The value of the scalar in (x, y) is computed by

$$\text{Scalar } (x, y) = \sum_{i,j=1}^{2} \lambda_i(\xi)\lambda_j(\eta) \,\text{scalar } (x_{ij}, y_{ij}) \tag{9.2}$$

To evaluate (9.2) it is necessary to know the value of (ξ, η) in point (x, y). This value can be computed from (9.1) by solving this system of non-linear equation with a Newton-Raphson method.
Define

$$\begin{bmatrix} F_1(\xi,\eta) \\ F_2(\xi,\eta) \end{bmatrix} = \sum_{i,j=1}^{2} \lambda_i(\xi)\lambda_j(\eta) \begin{bmatrix} x \\ y \end{bmatrix}_{ij} - \begin{bmatrix} x \\ y \end{bmatrix} \tag{9.3}$$

The Newton-Raphson method can be written as:

$$(\xi,\eta)^1 = (1/2, 1/2)$$

$$\begin{bmatrix} \partial F_1/\partial\xi & \partial F_1/\partial\eta \\ \partial F_2/\partial\xi & \partial F_2/\partial\eta \end{bmatrix}^n \left(\begin{bmatrix} \xi \\ \eta \end{bmatrix}^{n+1} - \begin{bmatrix} \xi \\ \eta \end{bmatrix}^n = - \begin{bmatrix} F_1 \\ F_2 \end{bmatrix}^n \quad n = 1, 2, ... \right. \tag{9.4}$$

Since Newton is a fast converging process, the maximal number of iterations is restricted to 5. At this moment the iterations is stopped if $\|\boldsymbol{\xi}^{n+1} - \boldsymbol{\xi}^n\| < 0.001$.

From (9.3) it follows that:

$$\begin{aligned} \boldsymbol{F}(\xi,\eta) &= (1-\xi)(1-\eta)\boldsymbol{x}_{11} + \xi\eta\boldsymbol{x}_{21} + (1-\xi)\eta\boldsymbol{x}_{12} - \boldsymbol{x} \\ &= \boldsymbol{x}_{11} + (\boldsymbol{x}_{11} + \boldsymbol{x}_{22} - \boldsymbol{x}_{21} - \boldsymbol{x}_{12})\xi\eta + (\boldsymbol{x}_{21} - \boldsymbol{x}_{11})\xi + (\boldsymbol{x}_{12} - \boldsymbol{x}_{11})\eta - \boldsymbol{x} \end{aligned} \tag{9.5}$$

and

$$\frac{\partial \boldsymbol{F}}{\partial \xi} = (\boldsymbol{x}_{11} + \boldsymbol{x}_{22} - \boldsymbol{x}_{21} - \boldsymbol{x}_{12})\eta + \boldsymbol{x}_{21} - \boldsymbol{x}_{11} \tag{9.6}$$

$$\frac{\partial \boldsymbol{F}}{\partial \eta} = (\boldsymbol{x}_{11} + \boldsymbol{x}_{22} - \boldsymbol{x}_{21} - \boldsymbol{x}_{12})\xi + \boldsymbol{x}_{12} - \boldsymbol{x}_{11} \tag{9.7}$$

With respect to the boundary points it is not longer possible to use an interpolation. In that case an extrapolation is used. Figure 9.2 shows the four points that are used to compute the value at an under boundary.

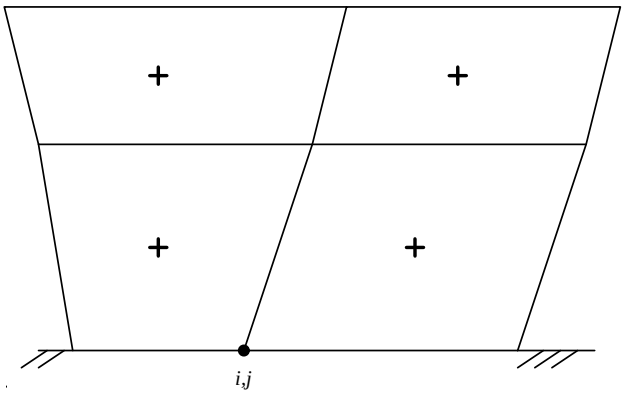


Figure 9.2: Cells that are used to extrapolate the scalar value at the under boundary i, j.

9.2 Interpolation of the velocity in 2D

The interpolation of the velocity is performed in three steps.

In the first step the Cartesian velocity is computed in the cell centre. First the fluxes V^1 and V^2 are averaged according to

$$V^1_{(0,0)} = (V^1_{(1,0)} + V^1_{(-1,0)})/2 \ , \tag{9.8}$$
$$V^2_{(0,0)} = (V^2_{(0,1)} + V^2_{(0,-1)})/2 \ , \tag{9.9}$$

see Figure 9.3 for the notations. Next the fluxes are transformed to Cartesian velocity com-

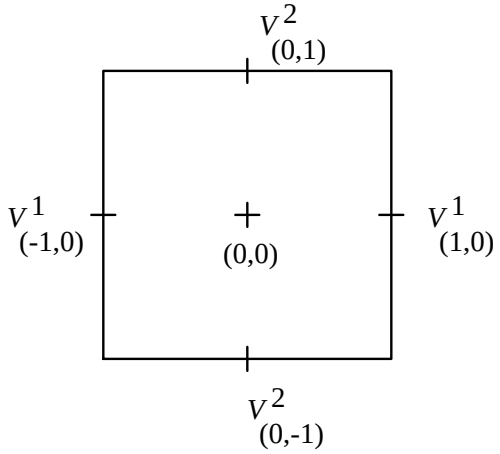


Figure 9.3: Cell with fluxes and centroid

ponents using:

$$V^\alpha = \sqrt{g} U^\alpha \ ,$$
$$\boldsymbol{u} = U^\alpha \boldsymbol{a}_\alpha \ ,$$

hence

$$\boldsymbol{u} = (V^1 \boldsymbol{a}_{(1)} + V^2 \boldsymbol{a}_{(2)})/\sqrt{g} \tag{9.10}$$

In the second step each of the components is interpolated to the vertices by exactly the procedure described for scalars in 9.1.

Finally at the boundary essential boundary conditions, if present, are substituted in order to avoid unnecessary interpolation erors.

9.3 Computation of the stream function

A special scalar that is computed, is the stream function ψ. Since in fact ISNaS incompressible computes the fluxes, the steam function computation is straightforward.

At present we assume $\psi = 0$ at the vertex point (1,1). The values in the other vertices are computed by summation:

For $j := 1(1)nj$ do

$\psi_{1,j+1} := \psi_{1,j} + V^1_{1,j}$

$i := 1(1)ni$ do

$\psi_{i+1,j+1} := \psi_{i,j+1} + V^2_{i,j+1}$

References

[1] C.C. Ashcraft and R.G. Grimes,
"On vectorizing incomplete factorization and SSOR preconditioners",
SIAM J. Sci. Stat. Comput., **9**, pp. 122-151, (1988).

[2] S.C. Eisenstat,
"Efficient implementation of a class of preconditioned conjugate gradient methods",
SIAM J. Sci. Stat. Comput., **2**, pp. 1-4, (1981).

[3] Kay, C. David,
"Schaum's outline of theory and problems of tensor calculus",
McGraw-Hill book company, (1988).

[4] W.P. Jones and B.E. Launder,
"The prediction of laminarization with a two-equation model of turbulence",
Int. J. of Heat and Mass Transfer, **15**, pp. 301-314, (1972).

[5] J.J.I.M. van Kan, C.W. Oosterlee, A. Segal and P. Wesseling,
"Discretization of the incompressible Navier-Stokes equations in general coordinates using contravariant velocity components",
Report 91-09, Faculty of Technical Mathematics and Informatics, Delft University of Technology, (1991).

[6] B.E. Launder and D.B. Spalding,
"Lectures in mathematical models of turbulence",
Academic Press, London, 1972.

[7] B.E. Launder and D.B. Spalding,
"The numerical computation of turbulent flows",
Comp. meth. in Appl. Mech. and Eng., **3**, pp. 269-289, (1974).

[8] J.A. Meijerink and H.A. van der Vorst,
"An iterative solution method for linear systems of which the coefficient matrix is a symmetric M-matrix",
Math. of Comp., **31**, pp. 148-162, (1977).

[9] R.R.P. van Nooyen,
ISNAS user manual, version 1.0 a, (1993).

[10] C.C. Paige and M.A. Saunders,
"LSQR: an algorithm for sparse linear equations and sparse least squares",
ACM Trans. Math. Soft., **8**, pp. 43-71, (1982).

[11] Y. Saad and M.H. Schultz,
"GMRES: a generalized minimal residual algorithm for solving non symmetric linear systems",
SIAM J. Statist. Comput., **7**, pp. 856-869, (1986).

[12] Guus Segal,
"The treatment of slip boundary conditions for the incompressible Navier-Stokes equations in general co-ordinates",
Report 91-22, Faculty of Technical Mathematics and Informatics, Delft University of Technology, (1991).

[13] Guus Segal and Kees Kassels,
"Some 2D test examples for the ISNaS incompressible code",
Report 91-44, Faculty of Technical Mathematics and Informatics, Delft University of Technology, (1991)

[14] Guus Segal,
"The no flow problem",
to appear.

[15] P. Sonneveld,
"CGS: a fast Lanczos type solver for nonsymmetric linear systems",
SIAM J. Sci. Stat. Comput., **10**, pp. 36-52, (1989).

[16] H.A. van der Vorst,
"Iterative solution methods for certain spares linear systems with a non-symmetric matrix arising from PDE problems",
J. Comp. Physics, **44**, pp. 1-19, (1981).

[17] H.A. van der Vorst and C. Vuik,
"GMRES: a family of nested GMRES methods",
J. Num. Lin. Alg. Appl., to appear (1993),
Report 91-80, Faculty of Technical Mathematics and Informatics, Delft University of Technology, (1991).

[18] C. Vuik,
"Solution of the discretized incompressible Navier-Stokes equations with the GMRES method",
Int. J. Num. Math. in Fluids, **16**, pp. 507-523, (1993)

[19] C. Vuik,
"Termination criteria for GMRES-like methods to solve the discretized incompressible Navier-Stokes equations",
Report 92-50, Faculty of Technical Mathematics and Informatics, Delft University of Technology, (1992).

[20] M. Zijlema,
"Finite volume discretization of the $k-\varepsilon$ turbulence model in general coordinates",
Technical report, to be published, (1993).

Appendices

In these appendices we prove equations (5.6), (5.7), (5.22) and (5.23).

A Proof of (5.6) and (5.7)

We have to prove (see Figure A.1):

$$U^n = sign(\boldsymbol{a}^{(n)} \cdot \boldsymbol{n}) \sqrt{g^{nn}}\, \boldsymbol{u} \cdot \boldsymbol{n} \tag{A.1}$$

and

$$\begin{bmatrix} U^{t_1} \\ U^{t_2} \end{bmatrix} = \begin{bmatrix} g_{t_1 t_1} & g_{t_1 t_2} \\ g_{t_2 t_1} & g_{t_2 t_2} \end{bmatrix}^{-1} \left\{ \begin{bmatrix} \alpha_{11} & \alpha_{12} \\ \alpha_{21} & \alpha_{22} \end{bmatrix}^{-1} \begin{bmatrix} \boldsymbol{u} \cdot \boldsymbol{\tau}_1 \\ \boldsymbol{u} \cdot \boldsymbol{\tau}_2 \end{bmatrix} - U^n \begin{bmatrix} g_{nt_1} \\ g_{nt_2} \end{bmatrix} \right\} \tag{A.2}$$

where

$$\alpha_{i1} = \frac{\begin{vmatrix} \boldsymbol{\tau}_i \cdot \boldsymbol{a}_{(t_1)} & g_{t_1 t_2} \\ \boldsymbol{\tau}_i \cdot \boldsymbol{a}_{(t_2)} & g_{t_2 t_2} \end{vmatrix}}{\begin{vmatrix} g_{t_1 t_1} & g_{t_1 t_2} \\ g_{t_2 t_1} & g_{t_2 t_2} \end{vmatrix}} \tag{A.3}$$

and

$$\alpha_{i2} = \frac{\begin{vmatrix} g_{t_1 t_1} & \boldsymbol{\tau}_i \cdot \boldsymbol{a}_{(t_1)} \\ g_{t_2 t_1} & \boldsymbol{\tau}_i \cdot \boldsymbol{a}_{(t_2)} \end{vmatrix}}{\begin{vmatrix} g_{t_1 t_1} & g_{t_1 t_2} \\ g_{t_2 t_1} & g_{t_2 t_2} \end{vmatrix}} \tag{A.4}$$

First (A.1):

$$\boldsymbol{u} \cdot \boldsymbol{n} = \boldsymbol{u} \cdot \frac{\boldsymbol{a}^{(n)}}{\|\boldsymbol{a}^{(n)}\|} = \frac{U^n}{\sqrt{g^{nn}}}$$

so

$$U^n = \sqrt{g^{nn}}\, \boldsymbol{u} \cdot \boldsymbol{n} \ .$$

This formula is true if $\boldsymbol{a}^{(n)}$ and $\boldsymbol{n}$ have the same direction else we have to use:

$$U^n = -\sqrt{g^{nn}}\, \boldsymbol{u} \cdot \boldsymbol{n} \ .$$

□

Formula (A.2):
The tangential vector $\boldsymbol{\tau}_i$, given by the user can be decomposed in the following way:

$$\boldsymbol{\tau}_i = \alpha_{i1} \boldsymbol{a}_{(t_1)} + \alpha_{i2} \boldsymbol{a}_{(t_2)} \ , \tag{A.5}$$

see Figure A.2.

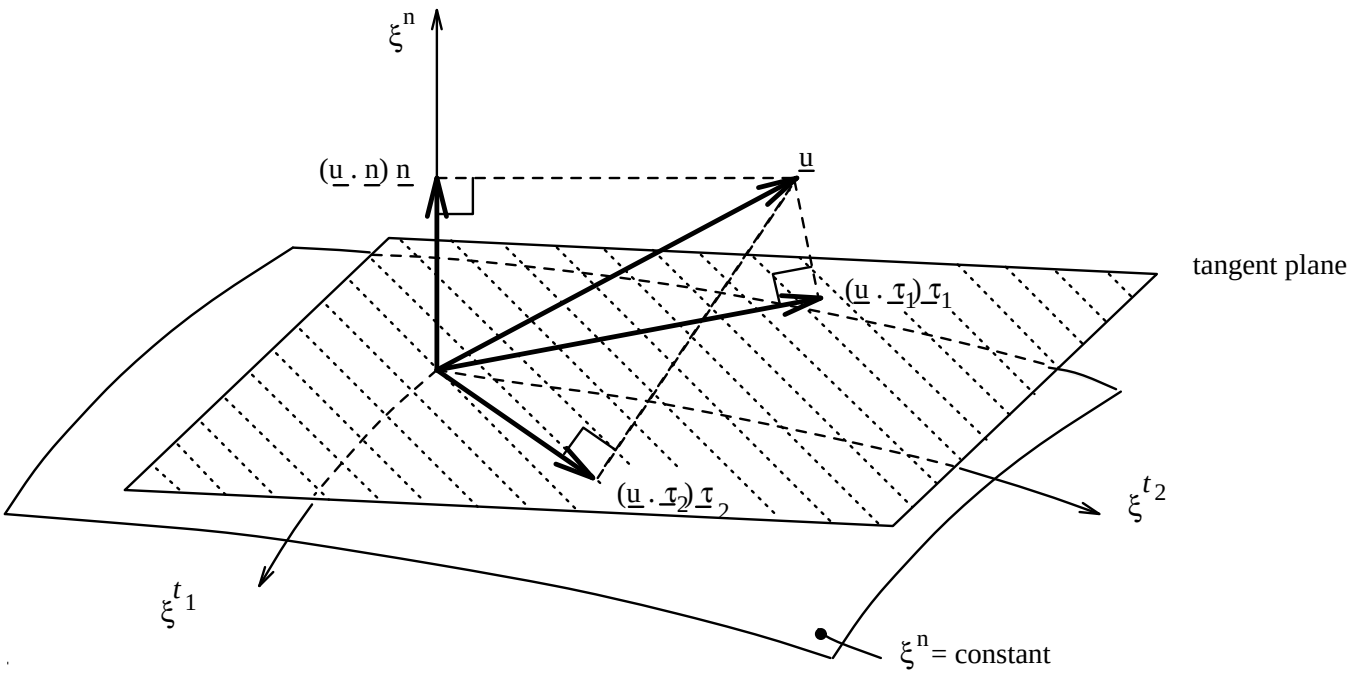


Figure A.1: Normal and tangential velocity components are given by the user ($\|\underline{n}\| = \|\underline{\tau}_1\| = \|\underline{\tau}_2\| = 1$)

The calculation of α_{ij}:
From Figure A.2 it is clear that

$$\begin{aligned} \boldsymbol{a}_{(t_1)} \cdot \alpha_{i2}\boldsymbol{a}_{(t_2)} &= \boldsymbol{a}_{(t_1)} \cdot (\boldsymbol{\tau}_i - \alpha_{i1}\boldsymbol{a}_{(t_1)}) \\ \boldsymbol{a}_{(t_1)} \cdot \boldsymbol{a}_{(t_1)}\alpha_{i1} + \boldsymbol{a}_{(t_1)} \cdot \boldsymbol{a}_{(t_2)}\alpha_{i2} &= \boldsymbol{a}_{(t_1)} \cdot \boldsymbol{\tau}_i \end{aligned} \tag{A.6}$$

and

$$\begin{aligned} \boldsymbol{a}_{(t_2)} \cdot \alpha_{i1}\boldsymbol{a}_{(t_1)} &= \boldsymbol{a}_{(t_2)} \cdot (\boldsymbol{\tau}_i - \alpha_{i2}\boldsymbol{a}_{(t_2)}) \\ \boldsymbol{a}_{(t_2)} \cdot \boldsymbol{a}_{(t_1)}\alpha_{i1} + \boldsymbol{a}_{(t_2)} \cdot \boldsymbol{a}_{(t_2)}\alpha_{i2} &= \boldsymbol{a}_{(t_2)} \cdot \boldsymbol{\tau}_i \ . \end{aligned} \tag{A.7}$$

Frome (A.6), (A.7) and Cramer's rule we obtain.

$$\alpha_{i1} = \frac{\begin{vmatrix} \boldsymbol{a}_{(t_1)} \cdot \boldsymbol{\tau}_i & \boldsymbol{a}_{(t_1)} \cdot \boldsymbol{a}_{(t_2)} \\ \boldsymbol{a}_{(t_2)} \cdot \boldsymbol{\tau}_i & \boldsymbol{a}_{(t_2)} \cdot \boldsymbol{a}_{(t_2)} \end{vmatrix}}{\begin{vmatrix} \boldsymbol{a}_{(t_1)} \cdot \boldsymbol{a}_{(t_1)} & \boldsymbol{a}_{(t_1)} \cdot \boldsymbol{a}_{(t_2)} \\ \boldsymbol{a}_{(t_2)} \cdot \boldsymbol{a}_{(t_1)} & \boldsymbol{a}_{(t_2)} \cdot \boldsymbol{a}_{(t_2)} \end{vmatrix}} = \frac{\begin{vmatrix} \boldsymbol{a}_{(t_1)} \cdot \boldsymbol{\tau}_i & g_{t_1 t_2} \\ \boldsymbol{a}_{(t_2)} \cdot \boldsymbol{\tau}_i & g_{t_2 t_2} \end{vmatrix}}{\begin{vmatrix} g_{t_1 t_1} & g_{t_1 t_2} \\ g_{t_2 t_1} & g_{t_2 t_2} \end{vmatrix}} \tag{A.8}$$

and

$$\alpha_{i2} = \frac{\begin{vmatrix} \boldsymbol{a}_{(t_1)} \cdot \boldsymbol{a}_{(t_1)} & \boldsymbol{a}_{(t_1)} \cdot \boldsymbol{\tau}_i \\ \boldsymbol{a}_{(t_2)} \cdot \boldsymbol{a}_{(t_1)} & \boldsymbol{a}_{(t_2)} \cdot \boldsymbol{\tau}_i \end{vmatrix}}{\begin{vmatrix} \boldsymbol{a}_{(t_1)} \cdot \boldsymbol{a}_{(t_1)} & \boldsymbol{a}_{(t_1)} \cdot \boldsymbol{a}_{(t_2)} \\ \boldsymbol{a}_{(t_2)} \cdot \boldsymbol{a}_{(t_1)} & \boldsymbol{a}_{(t_2)} \cdot \boldsymbol{a}_{(t_2)} \end{vmatrix}} = \frac{\begin{vmatrix} g_{t_1 t_1} & \boldsymbol{a}_{(t_1)} \cdot \boldsymbol{\tau}_i \\ g_{t_2 t_1} & \boldsymbol{a}_{(t_2)} \cdot \boldsymbol{\tau}_i \end{vmatrix}}{\begin{vmatrix} g_{t_1 t_1} & g_{t_1 t_2} \\ g_{t_2 t_1} & g_{t_2 t_2} \end{vmatrix}} \tag{A.9}$$

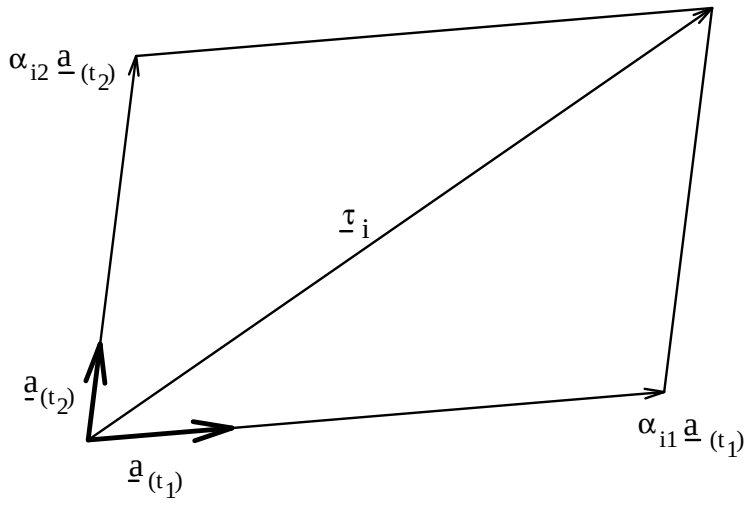


Figure A.2: Decomposition of $\underline{\tau}_i$.

Back to formula (A.5):

$$\begin{aligned} \boldsymbol{u}\cdot\boldsymbol{\tau}_i &= \boldsymbol{u}\cdot(\alpha_{i1}\boldsymbol{a}_{(t_1)}+\alpha_{i2}\boldsymbol{a}_{(t_2)}) \\ &= \alpha_{i1}\boldsymbol{u}\cdot g_{pt_1}\boldsymbol{a}^{(p)}+\alpha_{i2}\boldsymbol{u}\cdot g_{pt_2}\boldsymbol{a}^{(p)} \\ &= (\alpha_{i1}g_{pt_1}+\alpha_{i2}g_{pt_2})U^p\ , \end{aligned} \tag{A.10}$$

so:

$$\begin{aligned} \boldsymbol{u}\cdot\boldsymbol{\tau}_1 &= (\alpha_{11}g_{t_1t_1}+\alpha_{12}g_{t_1t_2})U^{t_1}+(\alpha_{11}g_{t_2t_1}+\alpha_{12}g_{t_2t_1})U^{t_2}+ \\ &\quad (\alpha_{11}g_{nt_1}+\alpha_{12}g_{nt_2})U^n\ , \end{aligned} \tag{A.11}$$

$$\begin{aligned} \boldsymbol{u}\cdot\boldsymbol{\tau}_2 &= (\alpha_{21}g_{t_1t_1}+\alpha_{22}g_{t_1t_2})U^{t_1}+(\alpha_{21}g_{t_2t_1}+\alpha_{22}g_{t_2t_2})U^{t_2}+ \\ &\quad (\alpha_{21}g_{nt_1}+\alpha_{22}g_{nt_2})U^n\ . \end{aligned} \tag{A.12}$$

Formula (A.11) and (A.12) in matrix notation gives:

$$\begin{bmatrix} \alpha_{11}g_{t_1t_1}+\alpha_{12}g_{t_1t_2} & \alpha_{11}g_{t_2t_1}+\alpha_{12}g_{t_2t_2} \\ \alpha_{21}g_{t_1t_1}+\alpha_{22}g_{t_1t_2} & \alpha_{21}g_{t_2t_1}+\alpha_{22}g_{t_2t_2} \end{bmatrix} \begin{bmatrix} U^{t_1} \\ U^{t_2} \end{bmatrix} = \begin{bmatrix} \boldsymbol{u}\cdot\boldsymbol{\tau}_1-(\alpha_{11}g_{nt_1}+\alpha_{12}g_{nt_2})U^n \\ \boldsymbol{u}\cdot\boldsymbol{\tau}_2-(\alpha_{21}g_{nt_1}+\alpha_{22}g_{nt_2})U^n \end{bmatrix}$$

or:

$$\begin{bmatrix} \alpha_{11} & \alpha_{12} \\ \alpha_{21} & \alpha_{22} \end{bmatrix} \begin{bmatrix} g_{t_1t_1} & g_{t_1t_2} \\ g_{t_2t_1} & g_{t_2t_2} \end{bmatrix} \begin{bmatrix} U^{t_1} \\ U^{t_2} \end{bmatrix} = \begin{bmatrix} \boldsymbol{u}\cdot\boldsymbol{\tau}_1 \\ \boldsymbol{u}\cdot\boldsymbol{\tau}_2 \end{bmatrix} - U^n \begin{bmatrix} \alpha_{11} & \alpha_{12} \\ \alpha_{21} & \alpha_{22} \end{bmatrix} \begin{bmatrix} g_{nt_1} \\ g_{nt_2} \end{bmatrix}$$

Hence:

$$\begin{bmatrix} U^{t_1} \\ U^{t_2} \end{bmatrix} = \begin{bmatrix} g_{t_1 t_1} & g_{t_1 t_2} \\ g_{t_2 t_1} & g_{t_2 t_2} \end{bmatrix}^{-1} \left\{ \begin{bmatrix} \alpha_{11} & \alpha_{12} \\ \alpha_{21} & \alpha_{22} \end{bmatrix}^{-1} \begin{bmatrix} \boldsymbol{u} \cdot \boldsymbol{\tau}_1 \\ \boldsymbol{u} \cdot \boldsymbol{\tau}_2 \end{bmatrix} - U^n \begin{bmatrix} g_{nt_1} \\ g_{nt_1} \end{bmatrix} \right\}^{1}$$

where α_{ij} is given by (A.8) and (A.9). □

[1] This formula can be modified in the following way:

$$\begin{bmatrix} U^{t_1} \\ U^{t_2} \end{bmatrix} = \begin{bmatrix} g_{t_2 t_2} & g_{t_1 t_2} \\ -g_{t_2 t_1} & g_{t_1 t_1} \end{bmatrix} \left\{ \begin{bmatrix} \beta_{11} & \beta_{12} \\ \beta_{21} & \beta_{22} \end{bmatrix}^{-1} \begin{bmatrix} \underline{u} \cdot \underline{\tau_1} \\ \underline{u} \cdot \underline{\tau_2} \end{bmatrix} - \frac{U^n}{g_{t_1 t_1} g_{t_2 t_2} - g_{t_1 t_2} g_{t_2 t_1}} \begin{bmatrix} g_{nt_1} \\ g_{nt_2} \end{bmatrix} \right\}$$

where

$$\beta_{i1} = \begin{vmatrix} \underline{\tau_i} \cdot \underline{a}_{(t_1)} & g_{t_1 t_2} \\ \underline{\tau_i} \cdot \underline{a}_{(t_2)} & g_{t_2 t_2} \end{vmatrix}$$

and

$$\beta_{12} = \begin{vmatrix} g_{t_1 t_1} & \underline{\tau_i} \cdot \underline{a}_{(t_1)} \\ g_{t_2 t_1} & \underline{\tau_i} \cdot \underline{a}_{(t_2)} \end{vmatrix}$$

B Proof of (5.22) and (5.23)

We have to prove (see Figure B.1):

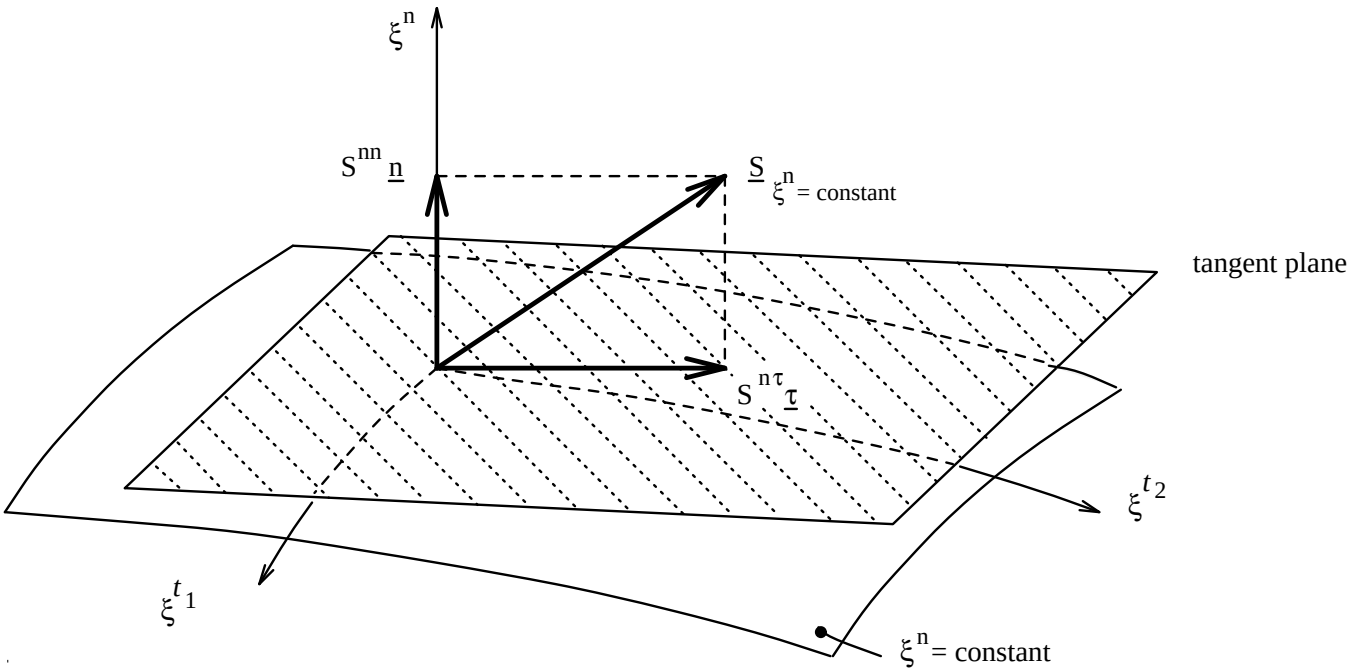


Figure B.1: The normal and tangential stress in the physical domain at the boundary $\xi^n = constant$.

$$\sigma^{nn} = g^{nn} S^{nn} \tag{B.1}$$

and

$$\begin{bmatrix} \sigma_{nt_1} \\ \sigma_{nt_2} \end{bmatrix} = sign(\boldsymbol{a}^{(n)} \cdot \boldsymbol{n}) \cdot \sqrt{g^{nn}} S^{n\tau} \begin{bmatrix} \alpha_1 \\ \alpha_2 \end{bmatrix} - \sigma^{nn} \begin{bmatrix} g_{t_1t_1} & g_{t_1t_2} \\ g_{t_2t_1} & g_{t_2t_2} \end{bmatrix}^{-1} \begin{bmatrix} g_{nt_1} \\ g_{nt_2} \end{bmatrix}^{2} \tag{B.2}$$

[2]This formula can also be modified in the following way:

$$\begin{bmatrix} \sigma^{nt_1} \\ \sigma^{nt_2} \end{bmatrix} = \frac{1}{g_{t_1t_1} \cdot g_{t_2t_2} - g_{t_1t_2} \cdot g_{t_2t_1}} \left\{ sign(a^{(n)} \cdot \underline{n}) \sqrt{g^{nn}} S^{n\tau} \begin{bmatrix} \beta_1 \\ \beta_2 \end{bmatrix} - \sigma^{nn} \begin{bmatrix} g_{t_2t_2} & -g_{t_1t_2} \\ -g_{t_2t_1} & g_{t_1t_1} \end{bmatrix} \begin{bmatrix} g_{nt_1} \\ g_{nt_2} \end{bmatrix} \right\}$$

where

$$\beta_1 = \begin{vmatrix} \underline{\tau} \cdot \underline{a}_{(t_1)} & g_{t_1t_2} \\ \underline{\tau} \cdot \underline{a}_{(t_2)} & g_{t_2t_2} \end{vmatrix}$$

and

$$\beta_2 = \begin{vmatrix} g_{t_1t_1} & \underline{\tau} \cdot \underline{a}_{(t_1)} \\ g_{t_2t_1} & \underline{\tau} \cdot \underline{a}_{(t_2)} \end{vmatrix} .$$

where

$$\alpha_1 = \frac{\begin{vmatrix} \boldsymbol{\tau}\cdot\boldsymbol{a}_{(t_1)} & g_{t_1t_2} \\ \boldsymbol{\tau}\cdot\boldsymbol{a}_{(t_2)} & g_{t_2t_3} \end{vmatrix}}{\begin{vmatrix} g_{t_1t_1} & g_{t_1t_2} \\ g_{t_2t_1} & g_{t_2t_2} \end{vmatrix}}\ , \tag{B.3}$$

$$\alpha_2 = \frac{\begin{vmatrix} g_{t_1t_1} & \boldsymbol{\tau}\cdot\boldsymbol{a}_{(t_1)} \\ g_{t_2t_1} & \boldsymbol{\tau}\cdot\boldsymbol{a}_{(t_2)} \end{vmatrix}}{\begin{vmatrix} g_{t_1t_1} & \boldsymbol{\tau}\cdot\boldsymbol{a}_{(t_1)} \\ g_{t_1t_1} & g_{t_1t_2} \\ g_{t_2t_1} & g_{t_2t_2} \end{vmatrix}}\ . \tag{B.4}$$

We start with formula (3.17) from Kay (1988):

$$\boldsymbol{S}_{\xi^n=constant} = \bar{\sigma}^{ij}(\boldsymbol{n}\cdot\boldsymbol{e}_i)\boldsymbol{e}_j \tag{B.5}$$

where $\bar{\sigma}^{ij}$ is stress tensor in the physical domain.
From $\underline{S}_{\xi^n=constant} = S^{nn}\boldsymbol{n} + S^{n\tau}\boldsymbol{\tau}$ and (B.5) we get:

$$\bar{\sigma}^{ij}(\boldsymbol{n}\cdot\boldsymbol{e}_i)\boldsymbol{e}_j = S^{nn}\boldsymbol{n} + S^{n\tau}\boldsymbol{\tau} \tag{B.6}$$

so

$$(\bar{\sigma}^{ij}(\boldsymbol{n}\cdot\boldsymbol{e}_j)\boldsymbol{e}_j)\cdot g^{(n)} = (S^{nn}\boldsymbol{n} + S^{n\tau}\boldsymbol{\tau})\cdot\boldsymbol{a}^{(n)} = S^{nn}\boldsymbol{n}\cdot\boldsymbol{a}^{(n)}\ . \tag{B.7}$$

It should be noticed that S^{nn} and $S^{n\tau}$ are not tensors.
The normal vector $\boldsymbol{n}$ pointing in the outside direction of the domain is equal to: $\frac{1}{\|\boldsymbol{a}^{(n)}\|}\boldsymbol{a}^{(n)}$ if $\boldsymbol{a}^{(n)}$ is pointing in the outside direction and $-\frac{1}{\|a^{(n)}\|}\boldsymbol{a}^{(n)}$ otherwise. Formally we can write:

$$\boldsymbol{n} = \frac{sign(\boldsymbol{a}^{(n)}\cdot\boldsymbol{n})}{\|\boldsymbol{a}^{(n)}\|}\boldsymbol{a}^{(n)}\ . \tag{B.8}$$

From (B.7) and (B.8) we get:

$$\begin{aligned}
&(\bar{\sigma}^{ij}(\boldsymbol{a}^{(n)}\cdot\boldsymbol{e}_i)\boldsymbol{e}_j)\cdot\boldsymbol{a}^{(n)} = S^{nn}\boldsymbol{a}^{(n)}\cdot\boldsymbol{a}^{(n)} \\
&\bar{\sigma}^{ij}(\boldsymbol{a}^{(n)})_i(\boldsymbol{a}^{(n)})_j = S^{nn}g^{nn} \\
&\bar{\sigma}^{ij}\frac{\partial\xi^n}{\partial x^i}\frac{\partial\xi^n}{\partial x^j} = g^{nn}S^{nn}
\end{aligned}$$

so

$$\sigma^{nn} = g^{nn}S^{nn}\ .$$

□

Formula (B.2):
The tangential vector $\boldsymbol{\tau}$ is just as in the previous proof equal to:

$$\boldsymbol{\tau} = \alpha_1\boldsymbol{a}_{(t_1)} + \alpha_2\boldsymbol{a}_{(t_2)} \tag{B.9}$$

where

$$\alpha_1 = \frac{\begin{vmatrix} \boldsymbol{\tau}\cdot\boldsymbol{a}_{(t_1)} & g_{t_1t_2} \\ \boldsymbol{\tau}\cdot\boldsymbol{a}_{(t_2)} & g_{t_2t_1} \end{vmatrix}}{\begin{vmatrix} g_{t_1t_1} & g_{t_1t_2} \\ g_{t_2t_1} & g_{t_2t_2} \end{vmatrix}}$$

and

$$\alpha_2 = \frac{\begin{vmatrix} g_{t_1t_1} & \boldsymbol{\tau}\cdot\boldsymbol{a}_{(t_1)} \\ g_{t_2t_1} & \boldsymbol{\tau}\cdot\boldsymbol{a}_{(t_2)} \end{vmatrix}}{\begin{vmatrix} g_{t_1t_1} & g_{t_1t_2} \\ g_{t_2t_1} & g_{t_2t_2} \end{vmatrix}} .$$

From (B.7), (B.8) and (B.9) it follows that:

$$\begin{aligned}
&(\bar{\sigma}^{ij}(\frac{sign(\boldsymbol{a}^{(n)}\cdot\boldsymbol{n})}{\|\boldsymbol{a}^{(n)}\|}\boldsymbol{a}^{(n)}\cdot\boldsymbol{e}_i)\boldsymbol{e}_j)\cdot\boldsymbol{a}_{t_l)} = S^{n\tau}(\alpha_1\boldsymbol{a}_{(t_1)}+\alpha_2\boldsymbol{a}_{(t_2)})\cdot\boldsymbol{a}_{(t_l)} \\
&sign(\boldsymbol{a}^{(n)}\cdot\boldsymbol{n})\bar{\sigma}^{ij}(\boldsymbol{a}^{(n)})_i(\boldsymbol{a}_{(t_l)})_j = \|\boldsymbol{a}^{(n)}\|S^{n\tau}(\alpha_1\boldsymbol{a}_{(t_1)}\cdot\boldsymbol{a}_{(t_l)}+\alpha_2\boldsymbol{a}_{(t_2)}\cdot\boldsymbol{a}_{t_l)}) \\
&sign(\boldsymbol{a}^{(n)}\cdot\boldsymbol{n})\bar{\sigma}^{ij}(\boldsymbol{a}^{(n)})_i g_{pt_l}(\boldsymbol{a}^{(p)})_j = \sqrt{g^{nn}}S^{n\tau}(\alpha_1 g_{t_1t_l}+\alpha_2 g_{t_2t_l}) \\
&sign(\boldsymbol{a}^{(n)}\cdot\boldsymbol{n})g_{pt_l}\bar{\sigma}^{ij}\frac{\partial\xi^n}{\partial x^i}\frac{\partial\xi^p}{\partial x^j} = \sqrt{g^{nn}}S^{n\tau}(\alpha_1 g_{t_1t_1}+\alpha_2 g_{t_2t_l})
\end{aligned}$$

so:

$$sign(\boldsymbol{a}^{(n)}\cdot\boldsymbol{n})g_{pt_l}\sigma^{np} = \sqrt{g^{nn}}S^{n\tau}(\alpha_1 g_{t_1t_l}+\alpha_2 g_{t_2t_l}) , \tag{B.10}$$

where $\sigma^{\alpha\beta}$ is the stress tensor in the computational domain. For $l = 1,2$ we get:

$$sign(\boldsymbol{a}^{(n)}\cdot\boldsymbol{n})\{g_{t_1t_1}\sigma^{nt_1}+g_{t_2t_1}\sigma^{nt_2}+g_{nt_1}\sigma^{nn}\} = \sqrt{g^{nn}}S^{n\tau}(\alpha_1 g_{t_1t_1}+\alpha_2 g_{t_2t_1})$$

and

$$sign(\boldsymbol{a}^{(n)}\cdot\boldsymbol{n})\{g_{t_1t_2}\sigma^{nt_1}+g_{t_2t_2}\sigma^{nt_2}+g_{nt_2}\sigma^{nn}\} = \sqrt{g^{nn}}S^{n\tau}(\alpha_1 g_{t_1t_2}+\alpha_2 g t_2t_2)$$

or:

$$\begin{bmatrix} g_{t_1t_1} & g_{t_2t_1} \\ g_{t_1t_2} & g_{t_2t_2} \end{bmatrix}\begin{bmatrix} \sigma^{nt_1} \\ \sigma^{nt_2} \end{bmatrix} + \sigma^{nn}\begin{bmatrix} g_{nt_1} \\ g_{nt_2} \end{bmatrix} = sign(\boldsymbol{a}^{(n)}\cdot\boldsymbol{n})\sqrt{g^{nn}}S^{n\tau}\begin{bmatrix} g_{t_1t_1} & g_{t_2t_1} \\ g_{t_1t_2} & g_{t_2t_2} \end{bmatrix}\begin{bmatrix} \alpha_1 \\ \alpha_2 \end{bmatrix}$$

so:

$$\begin{bmatrix} \sigma^{nt_1} \\ \sigma^{nt_2} \end{bmatrix} = sign(\boldsymbol{a}^{(n)}\cdot\boldsymbol{n})\sqrt{g^{nn}}S^{n\tau}\begin{bmatrix} \alpha_1 \\ \alpha_2 \end{bmatrix} - \sigma^{nn}\begin{bmatrix} g_{t_1t_1} & g_{t_2t_1} \\ g_{t_1t_2} & g_{t_2t_2} \end{bmatrix}^{-1}\begin{bmatrix} g_{nt_1} \\ g_{nt_2} \end{bmatrix}$$

hence:

$$\begin{bmatrix} \sigma^{nt_1} \\ \sigma^{nt_2} \end{bmatrix} = sign(\boldsymbol{a}^{(n)}\cdot\boldsymbol{n})\sqrt{g^{nn}}S^{n\tau}\begin{bmatrix} \alpha_1 \\ \alpha_2 \end{bmatrix} - \sigma^{nn}\begin{bmatrix} g_{t_1t_1} & g_{t_1t_2} \\ g_{t_2t_1} & g_{t_2t_2} \end{bmatrix}^{-1}\begin{bmatrix} g_{nt_1} \\ g_{nt_2} \end{bmatrix} . \tag{B.11}$$

By using Cramers rule in formula (B.11) we get:

$$\begin{bmatrix} \sigma^{nt_1} \\ \sigma^{nt_2} \end{bmatrix} = \frac{1}{g_{t_1t_1}g_{t_2t_1} - g_{t_1t_2}g_{t_2t_1}} \left\{ sign(\boldsymbol{a}^{(n)} \cdot \boldsymbol{n})\sqrt{g^{nn}} S^{n\tau} \begin{bmatrix} \beta_1 \\ \beta_2 \end{bmatrix} - \sigma^{nn} \begin{bmatrix} g_{t_2t_2} & -g_{t_1t_2} \\ -g_{t_2t_1} & g_{t_1t_1} \end{bmatrix} \begin{bmatrix} g_{nt_1} \\ g_{nt_2} \end{bmatrix} \right\} \tag{B.12}$$

where

$$\beta_1 = \begin{vmatrix} \boldsymbol{\tau} \cdot \boldsymbol{a}_{(t_1)} & g_{t_1t_2} \\ \boldsymbol{\tau} \cdot \boldsymbol{a}_{(t_2)} & g_{t_2t_2} \end{vmatrix} \tag{B.13}$$

and

$$\beta_2 = \begin{vmatrix} g_{t_1t_1} & \boldsymbol{\tau} \cdot \boldsymbol{a}_{(t_1)} \\ g_{t_2t_1} & \boldsymbol{\tau} \cdot \boldsymbol{a}_{(t_2)} \end{vmatrix} . \tag{B.14}$$

□