V. Methods for Euler and Navier-Stokes Equations

by Bernhard Müller

Outline

The application of the cell-centered finite volume method is illustrated for the 2D Navier-Stokes equations. The finite volume discretization can be used for structured and unstructured grids. For the latter, vertex finite volume methods are preferred, because much less memory is needed to store the cell averages at nodes than in cells, if the cells are not quadrilaterals in 2D or not hexahedra in 3D.

The upwind discretization of the inviscid fluxes takes the direction into account, where the waves normal to the cell interface come from. Roe's approximate Riemann solver and scalar numerical dissipation are discussed in detail. Both methods allow the computation of shocks.

For contact discontinuities, Roe's approximate Riemann solver yields more accurate results than scalar numerical dissipation. For the viscous fluxes, central discretization is adequate.

The spatial discretization and the boundary conditions lead to a large system of ODEs. Its solution by explicit Runge-Kutta methods and implicit linear multistep methods is discussed for steady and unsteady flow applications. Convergence acceleration techniques are pointed out.

Algorithmic details of the cell-centered finite volume method on structured quadrilateral grids and of the dual-based vertex finite volume method on unstructured triangular grids are outlined.

1. Cell-Centered Finite Volume Method

1.1.Integral Form of 2D Navier-Stokes Equations

$$\int_\Omega \frac{\partial \mathbf{U}}{\partial t} dV + \int_\Gamma (\mathbf{F}_1 n_x + \mathbf{F}_2 n_y)\, dA = \int_\Omega \mathbf{F}_e\, dV. \qquad (1)$$

where $\mathbf{F}_j = \mathbf{F}_j^c - \mathbf{F}_j^v$, and

$$\mathbf{U} = \begin{pmatrix} \rho \\ \rho u \\ \rho v \\ \rho E \end{pmatrix}, \ \mathbf{F}_1^c = \begin{pmatrix} \rho u \\ \rho u^2 + p \\ \rho v u \\ \rho H u \end{pmatrix}, \ \mathbf{F}_2^c = \begin{pmatrix} \rho v \\ \rho u v \\ \rho v^2 + p \\ \rho H v \end{pmatrix}, \ \mathbf{F}_e = \begin{pmatrix} 0 \\ f_1 \\ f_2 \\ \mathbf{f} \cdot \mathbf{u} \end{pmatrix},$$

$H = E + \frac{p}{\rho}$ total enthalpy, $p = (\gamma - 1)(\rho E - \frac{1}{2}\rho(u^2 + v^2))$ pressure.

$$\mathbf{F}_1^v = \begin{pmatrix} 0 \\ \tau_{xx} \\ \tau_{yx} \\ \tau_{xx}u + \tau_{yx}v + k\frac{\partial T}{\partial x} \end{pmatrix}, \ \mathbf{F}_2^v = \begin{pmatrix} 0 \\ \tau_{xy} \\ \tau_{yy} \\ \tau_{xy}u + \tau_{yy}v + k\frac{\partial T}{\partial y} \end{pmatrix},$$

$\tau_{xx} = \frac{4}{3}\mu\frac{\partial u}{\partial x} - \frac{2}{3}\mu\frac{\partial v}{\partial y}$, $\tau_{xy} = \tau_{yx} = \mu(\frac{\partial u}{\partial y} + \frac{\partial v}{\partial x})$, $\tau_{yy} = \frac{4}{3}\mu\frac{\partial v}{\partial y} - \frac{2}{3}\mu\frac{\partial u}{\partial x}$.

For quadrilateral cell Ω_i with faces Γ_l, $l = 1, 2, 3, 4$:

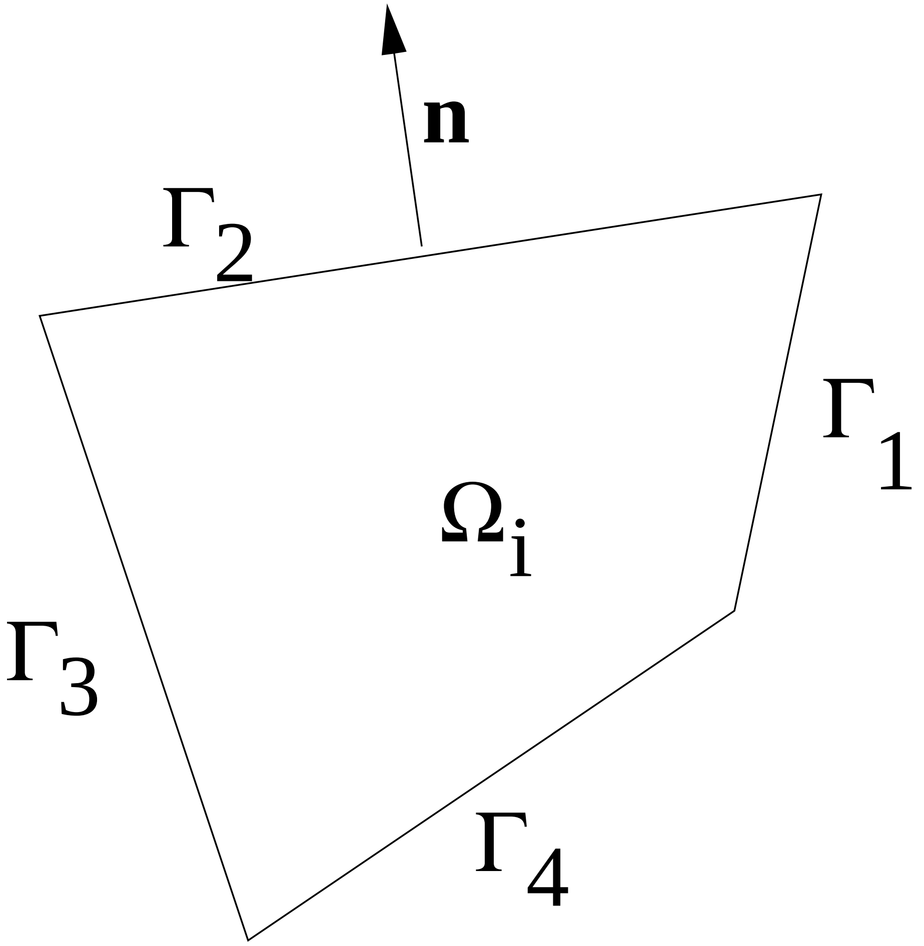
$$\frac{d\mathbf{U}_i}{dt}V_i + \sum_{l=1}^{4}\int_{\Gamma_l}(\mathbf{F}_1 n_x + \mathbf{F}_2 n_y)\, dA = \mathbf{F}_{e_i}V_i, \qquad (2)$$

with cell averaged conservative variables and source term

$$\mathbf{U}_i(t) = \frac{1}{V_i}\int_{\Omega_i}\mathbf{U}(\mathbf{x}, t)\, dV,$$

$$\mathbf{F}_{e_i}(t) = \frac{1}{V_i}\int_{\Omega_i}\mathbf{F}_e(\mathbf{U}(\mathbf{x}, t))\, dV \ ,$$

where $V_i = \int_{\Omega_i} dV$ area of quadrilateral cell Ω_i, equal to volume of corresponding 3D cell with unit length in z-direction.



If we express the flux over face l $\int_{\Gamma_l} \mathbf{F} \cdot \mathbf{n} dA$ by the scalar product of the face averaged flux tensor $\mathbf{F}_l$ and the face normal $\mathbf{S}_l = \int_{\Gamma_l} \mathbf{n} dA$, we can write the 2D Navier-Stokes equations (2) as

$$\frac{d\mathbf{U}_i}{dt} V_i + \sum_{l=1}^{4} \mathbf{F}_l \cdot \mathbf{S}_l \; = \; \mathbf{F}_{e_i} V_i, \tag{3}$$

1.2. Cell-Centered Finite Volume Method

We approximate $\mathbf{F} = \mathbf{F}_1 \mathbf{e}_1 + \mathbf{F}_2 \mathbf{e}_2$ at face Γ_l by means of the neighboring cell averages, e.g. the arithmetic average

$$\mathbf{F}_l \; = \; \frac{1}{2} (\mathbf{F}(\mathbf{U}_i) \; + \; (\mathbf{F}(\mathbf{U}_l)) \, , \tag{4}$$

where l denotes the neighboring cell adjacent to face l.

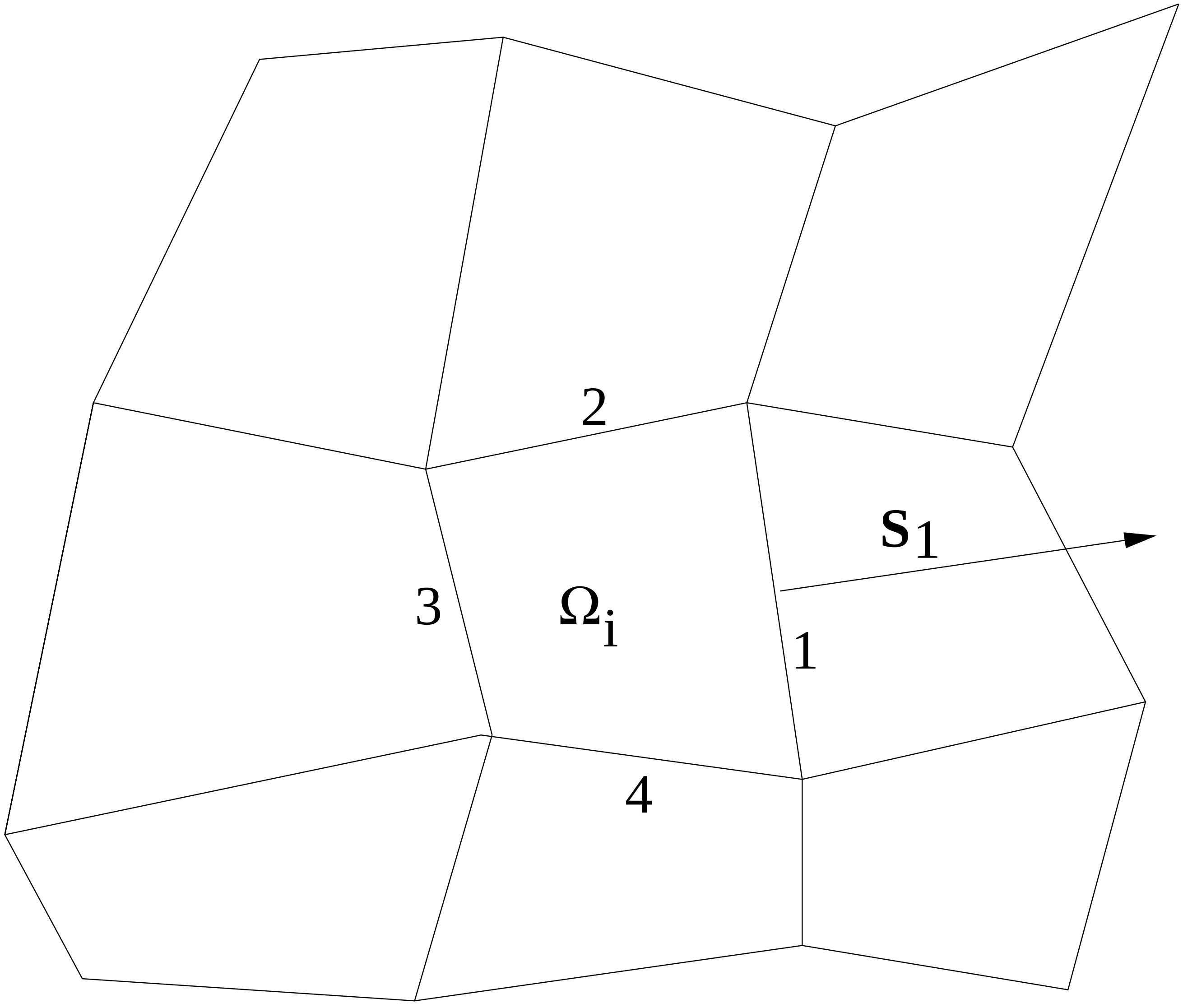
A nonlinear source term is approximated by $\mathbf{F}_{e_i} \approx \mathbf{F}_e(\mathbf{U}_i)$.

Thus, the cell-centered finite volume discretization of the

Navier-Stokes equations yields the following ODE using the same notation for the approximate cell averages as for the exact ones in (3):

$$\frac{d\mathbf{U}_i}{dt} V_i + \sum_{l=1}^{4} [\mathbf{F}_1 \, S_1 \, + \, \mathbf{F}_2 \, S_2]_l \; = \; \mathbf{F}_{e_i} \, V_i, \tag{5}$$

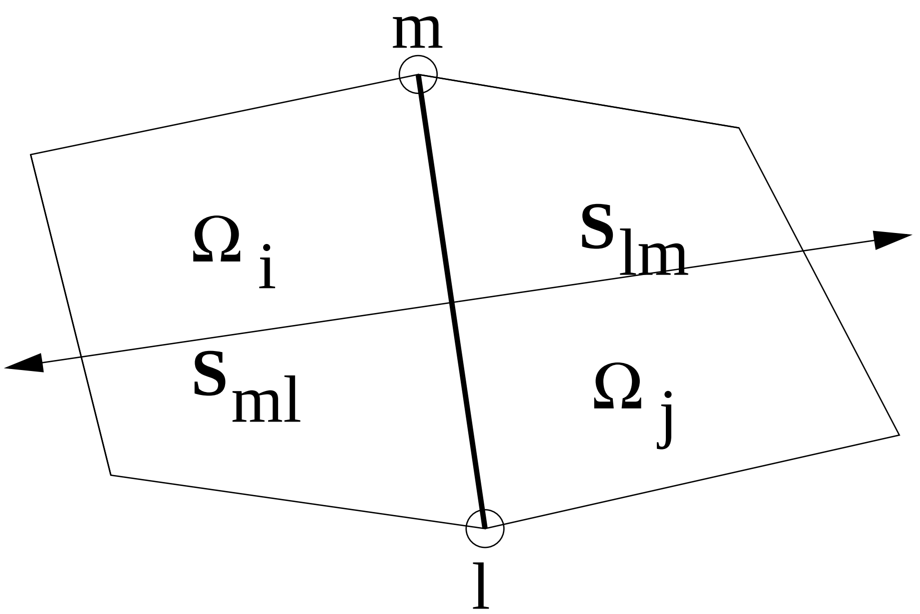
where $\mathbf{F}_{1_l}$ and $\mathbf{F}_{2_l}$ are functions of $\mathbf{U}_i$ and $\mathbf{U}_l$.


2
$\mathbf{s}_1$
3
Ω_i
1
4

Let the face Γ_{lm} be defined by the vertices $\mathbf{x}_l$ and $\mathbf{x}_m$. Assume that $\mathbf{x}_l$ and $\mathbf{x}_m$ lie on the boundary of Ω_i in counterclockwise direction. Then, the outer normal of face Γ_{lm} is given by

$$\mathbf{S}_{lm} = \begin{pmatrix} y_m - y_l \\ -(x_m - x_l) \end{pmatrix} .$$

For the neighboring cell Ω_j , the outer normal of face Γ_{ml} is $\mathbf{S}_{ml} = -\mathbf{S}_{lm}$. Thus, $(\mathbf{F} \cdot \mathbf{S})_{ml} = -(\mathbf{F} \cdot \mathbf{S})_{lm}$, i.e. the flux leaving control volume Ω_j over Γ_{ml} is equal to the flux entering control volume Ω_i over Γ_{lm} . Consequently, the net flux over all interior boundaries is zero guaranteeing conservativity.



1.3. Computation of Flux Balance

Suppose we have a face-based data structure, i.e. for each interior face Γ_n, the numbers i and j of the adjacent left and right cells, respectively, and the outer normal $\mathbf{S}_n$ of Ω_i are given. Note that $-\mathbf{S}_n$ is the outer normal of Ω_j . After initializing the flux balance array `FB ( NCELL )` for all `NCELL` cells either by zero or the boundary flux, we can loop over all `NFACE` interior faces:

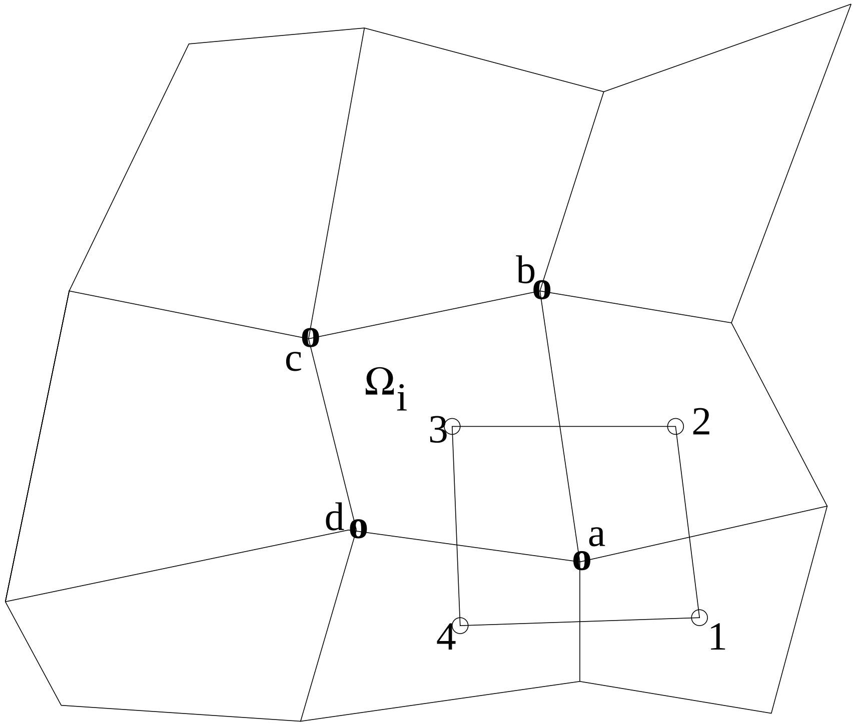
```
DO N=1,NFACE
I = number of left cell
J = number of right cell
S = face normal of left cell
FLUX = numerical flux ( U (I), U (J), S )
FB (I) = FB (I) + FLUX
FB (J) = FB (J) - FLUX
END DO
```

Thus, the flux balance for each cell can be computed without the need for storing or recomputing the fluxes.

1.4. Approximation of Derivatives

The x- and y-derivatives of u, v and T in (5) are usually approximated by using the Green-Gauss theorem, e.g.

$$\int_{\Omega} \boldsymbol{\nabla} T \, dV \;=\; \int_{\partial\Omega} T \mathbf{n} \, dA \,. \tag{6}$$



The derivatives at vertex a can be approximated by the volume average over the control volume Ω_a defined by the midpoints $\mathbf{x}_l$, $l = 1, 2, 3, 4$, of the adjacent cells

$$(\boldsymbol{\nabla} T)_a \approx \frac{1}{V_a} \int_{\Omega_a} \boldsymbol{\nabla} T \, dV \, .$$

With the Green-Gauss theorem (6), we get

$$(\boldsymbol{\nabla} T)_a \approx \frac{1}{V_a} \int_{\partial\Omega_a} T \mathbf{n} \, dA \, . \tag{7}$$

The surface integral is approximated by the trapezoidal rule

$$(\boldsymbol{\nabla} T)_a \approx \frac{1}{V_a} \sum_{l=1}^{4} \frac{1}{2}(T_l + T_{l+1}) \mathbf{S}_{l,l+1} \, , \tag{8}$$

where $\mathbf{S}_{l,l+1}$ is the normal of the face with the endpoints $\mathbf{x}_l$ and $\mathbf{x}_{l+1}$.

The viscosity and thermal conductivity are obtained by averaging, e.g.

$$\mu_a \approx \frac{1}{4}\sum_{l=1}^{4} \mu_l \, .$$

Thus, the viscous flux tensor $\mathbf{F}^v$ is available at the vertices a, b, c and d. Then, the viscous flux balance can be approximated by the trapezoidal rule

$$\int_{\partial\Omega_i} \mathbf{F}^v \cdot \mathbf{n} \, dA \; = \; \sum_{m=a}^{d} \frac{1}{2}(\mathbf{F}^v_m + \mathbf{F}^v_{m+1}) \cdot \mathbf{S}_{m,m+1} \qquad (9)$$

where $b = a + 1$, etc.

The central approximation (9) is 2nd order accurate on a smooth grid. Note that consistency may be lost on a non-smooth grid.

2. Numerical Flux Functions and Boundary Conditions

The shock thickness is proportional to (cf. Whitham (1974), p. 190)

$$\frac{4\mu}{3\rho_1}\frac{2\gamma}{\gamma+1}\frac{1}{(\mathbf{u}_1-\mathbf{u}_2)\cdot\mathbf{n}}\,,$$

where 1 and 2 denote the states upstream and downstream of the shock, respectively. E.g. for $\frac{\mu}{\rho_1} \approx 1.4\,10^{-5}\,\frac{m^2}{s}$ and a normal velocity jump of $\approx 100\frac{m}{s}$, the shock thickness is of the order of $1mm$. In general, the internal viscous structure of a shock wave needs not to be resolved, and a shock can be described as a discontinuity governed by the Euler equations.

For simple shock problems, e.g. blunt body flow, shock fitting can be used: the shock is treated as a moving discontinuity with the Rankine-Hugoniot relations satisfied across the shock.

In general, shock capturing must be used: the shock is computed as any other flow structure and evolves as steep gradient, which is smeared over a few grid points. The better the method, the less the smearing. Central discretizations (4) of the inviscid fluxes lead to the Gibb's phenomenon, i.e. oscillations near the shock.

We review here methods that are not only suitable for the capturing of shocks and contact discontinuities, but also of smooth flow. Even for smooth flow, unresolved wave numbers $> \frac{\pi}{\Delta x}$, grid discontinuities and boundary conditions can lead to numerical instability, unless numerical dissipation is employed. Numerical dissipation is inherent in upwind methods and must be added to central methods.

2.1. Riemann Solvers

In order to determine the flux at a cell interface, Godunov (1959) proposed to solve the Riemann problem for the Euler equations, i.e. the 1D Euler equations (obtained from (1) in 2D) with discontinuous initial data given by the cell averages in the cells on the left and right sides of the interface (here at $x = 0$) at the old time level:

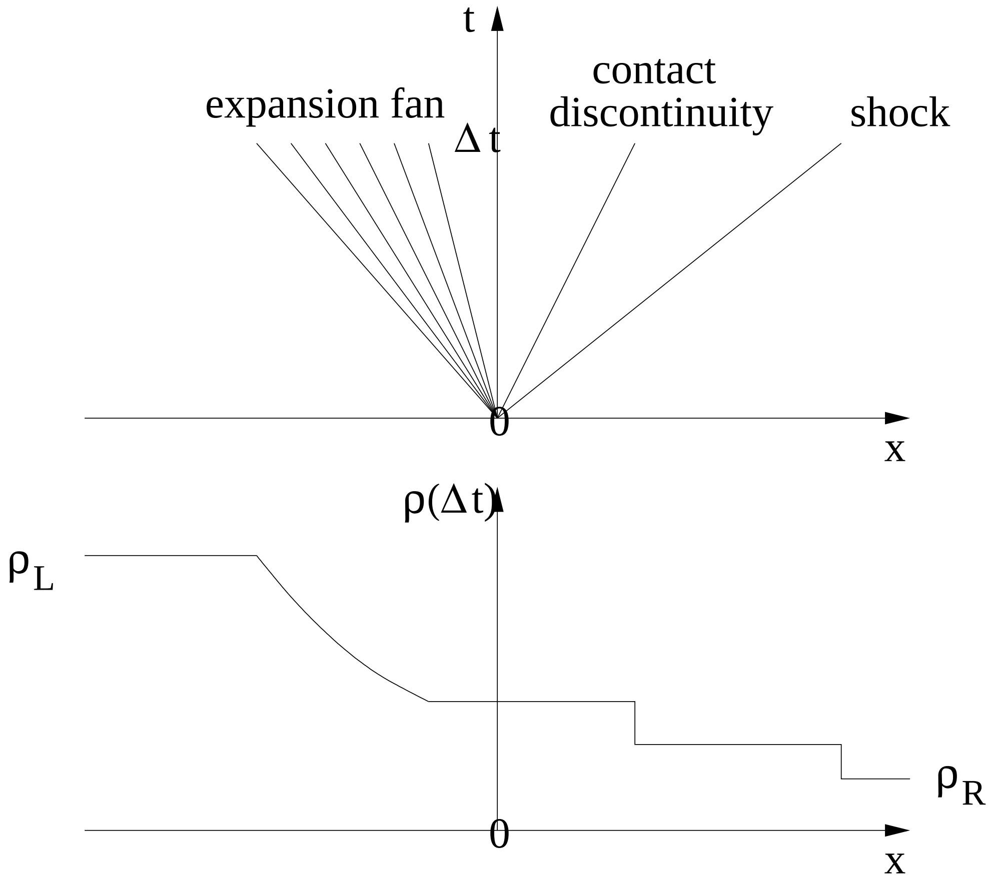
$$\frac{\partial \mathbf{U}}{\partial t} + \frac{\partial \mathbf{F}_1^c}{\partial x} = 0 \qquad (10)$$

$$\mathbf{U}(x,0) = \begin{cases} \mathbf{U}_L & x < 0 \\ \mathbf{U}_R & x > 0 \end{cases}$$

In the Godunov scheme, the Riemann problem (10) is solved exactly to yield $\mathbf{U}(x,t)$ for $t > 0$. The flux at the interface $x = 0$ is determined by $\mathbf{F}_1^c(\mathbf{U}(0,t))$.

Example of Riemann problem: shock tube problem (1D: $v = w = 0$)

$$u_L = u_R = 0\,, \quad \rho_L > \rho_R\,, \quad p_L > p_R$$



Approximate Riemann Solvers

Exact solution of Riemann problem

- expensive in terms of CPU time because of iterations,
- not needed, because accuracy limited by the following integration.

Popular approximate Riemann solvers:

Roe (1981) selects an appropriate linear equivalent Riemann problem and solves it exactly.

Osher (1982) approximates a shock by a compression fan, i.e. isentropic flow across shock assumed. The waves are coming to $(0, t)$ in the x-t-diagram.

Pandolfi (1984) approximates a shock by a compression fan (same as Osher). The waves are coming from the origin in the x-t-diagram.

2.2. Roe's Approximate Riemann Solver

Consider the quasi-linear form of (10)

$$\frac{\partial \mathbf{U}}{\partial t} + \mathbf{A}\frac{\partial \mathbf{U}}{\partial x} = 0\,, \tag{11}$$

where $\mathbf{A} = \frac{\partial \mathbf{F}_1^c}{\partial \mathbf{U}}$ is the Jacobian matrix of $\mathbf{F}_1^c$. Note that $\mathbf{A} = \mathbf{A}(\mathbf{U})$.

Define the linear problem

$$\frac{\partial \mathbf{U}}{\partial t} + \hat{\mathbf{A}}\frac{\partial \mathbf{U}}{\partial x} = 0\,, \tag{12}$$

where $\hat{\mathbf{A}} = \hat{\mathbf{A}}(\mathbf{U}_L, \mathbf{U}_R) = \mathit{constant}$, such that

- $\hat{\mathbf{A}}(\mathbf{U}_L, \mathbf{U}_R) \rightarrow \mathbf{A}(\mathbf{U}) \quad$ for $\mathbf{U}_L \rightarrow \mathbf{U}$ and $\mathbf{U}_R \rightarrow \mathbf{U}$,
- $\mathbf{F}_1^c(\mathbf{U}_R) - \mathbf{F}_1^c(\mathbf{U}_L) = \hat{\mathbf{A}}(\mathbf{U}_R - \mathbf{U}_L)$,
- $\hat{\mathbf{A}}$ has real eigenvalues and linear independent eigenvectors.

These conditions are satisfied for

$$\hat{\mathbf{A}} \; = \; \mathbf{A}(\hat{\mathbf{U}})\,, \tag{13}$$

where $\hat{\mathbf{U}}$ is determined by the Roe-averages

$$\hat{\rho} = \sqrt{\rho_L \rho_R}\,,\ \hat{\mathbf{u}} = \frac{\sqrt{\rho_L}\mathbf{u}_L + \sqrt{\rho_R}\mathbf{u}_R}{\sqrt{\rho_L} + \sqrt{\rho_R}}\,,\ \hat{H} = \frac{\sqrt{\rho_L}H_L + \sqrt{\rho_R}H_R}{\sqrt{\rho_L} + \sqrt{\rho_R}}\,. \tag{14}$$

We get the Roe-averaged speed of sound from $\hat{c}^2 = (\gamma - 1)(\hat{H} - \frac{1}{2}|\hat{\mathbf{u}}|^2)$.

Roe's approximate Riemann solver is exact, if $\mathbf{U}_L$ and $\mathbf{U}_R$ are connected by a single shock or by a single contact discontinuity.

We can express (11) in primitive variables $\mathbf{V} = (\rho, u, v, p)^T$ as

$$\frac{\partial \mathbf{U}}{\partial \mathbf{V}}\frac{\partial \mathbf{V}}{\partial t} \; + \; \mathbf{A}\frac{\partial \mathbf{U}}{\partial \mathbf{V}}\frac{\partial \mathbf{V}}{\partial x} \; = \; 0$$

or

$$\frac{\partial \mathbf{V}}{\partial t} \; + \; \mathbf{M}^{-1}\mathbf{A}\mathbf{M}\frac{\partial \mathbf{V}}{\partial x} \; = \; 0\,,$$

where in 3D $$\mathbf{M} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ u & \rho & 0 & 0 & 0 \\ v & 0 & \rho & 0 & 0 \\ w & 0 & 0 & \rho & 0 \\ \frac{|\mathbf{u}|^2}{2} & \rho u & \rho v & \rho w & \frac{1}{\gamma-1} \end{pmatrix} .$$

Comparing with the non-conservative Euler equations (cf. II. 2.3.), we see that $\mathbf{B}_1 = \mathbf{M}^{-1}\mathbf{A}\mathbf{M}$. Since $\mathbf{B}_1$ can be diagonalized, also $\mathbf{A}$ can:

$$\mathbf{R}^{-1}\,\mathbf{A}\,\mathbf{R} \;=\; \hat{\mathbf{\Lambda}}\,, \tag{15}$$

where here $\mathbf{R} = \mathbf{M}\mathbf{T}(\mathbf{e}_1)$. Thus, we can diagonalize $\hat{\mathbf{A}}$ and express (12) as 4 linear equations in 2D

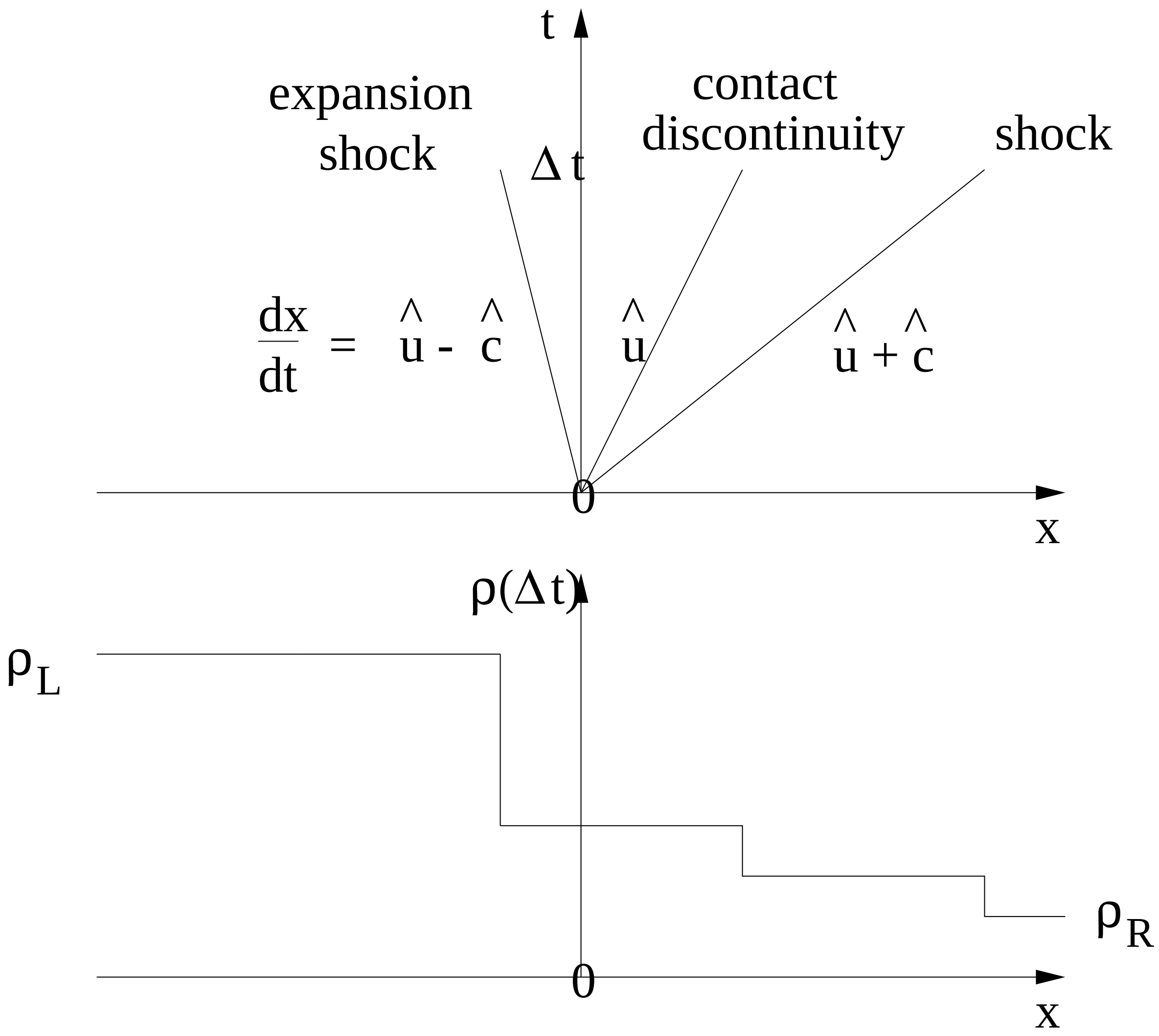
$$\frac{\partial W_j}{\partial t} + \hat{\lambda}_j \frac{\partial W_j}{\partial x} \;=\; 0\,, \quad j = 1, 2, 3, 4, \;, \tag{16}$$

where $\mathbf{W} = \hat{\mathbf{R}}^{-1}\mathbf{U} = \begin{pmatrix} -\frac{\hat{\rho}}{2\hat{c}}u + \frac{1}{2\hat{c}^2}p \\ \rho - \frac{1}{\hat{c}^2}p \\ \hat{\rho}v \\ \frac{\hat{\rho}}{2\hat{c}}u + \frac{1}{2\hat{c}^2}p \end{pmatrix}$ are the characteristic variables.

$\hat{\lambda}_1 = \hat{u} - \hat{c}\,,\ \hat{\lambda}_2 = \hat{\lambda}_3 = \hat{u}\,,\ \hat{\lambda}_4 = \hat{u} + \hat{c}$ are the constant wave speeds.

Let δW_j denote the strength of the j-th wave, i.e. the j-th component of $\hat{\mathbf{R}}^{-1}(\mathbf{U}_R - \mathbf{U}_L) = \hat{\mathbf{T}}^{-1}(\mathbf{e}_1)(\mathbf{V}_R - \mathbf{V}_L)$. Then, the exact solution of (16) at the interface can be expressed as

$$W_j(0,t) = \begin{cases} W_{j_L} = W_{j_R} - \delta W_j & if\ \hat{\lambda}_j > 0 \\ W_{j_R} = W_{j_L} + \delta W_j & if\ \hat{\lambda}_j < 0 \end{cases} .$$



Roe's approximate Riemann solver.

The flux difference can be written as

$$\mathbf{F}_1^c(\mathbf{U}_R) - \mathbf{F}_1^c(\mathbf{U}_L) \;=\; \hat{\mathbf{R}}\hat{\mathbf{\Lambda}}\hat{\mathbf{R}}^{-1}(\mathbf{U}_R - \mathbf{U}_L) \;=\; \sum_{j=1}^{4} \delta W_j \hat{\lambda}_j \hat{\mathbf{r}}_j$$

where $\hat{\mathbf{r}}_j$ is the jth right eigenvector of $\hat{\mathbf{A}}$, i.e. the jth column of $\hat{\mathbf{R}}$.

Thus, Roe expresses the flux at the interface by

$$\begin{aligned} \mathbf{F}_1^c(0,t) &= \mathbf{F}_1^c(\mathbf{U}_L) + \hat{\mathbf{A}}^-(\mathbf{U}_R - \mathbf{U}_L) && (17)\\ \mathbf{F}_1^c(0,t) &= \mathbf{F}_1^c(\mathbf{U}_R) - \hat{\mathbf{A}}^+(\mathbf{U}_R - \mathbf{U}_L) && (18)\\ \mathbf{F}_1^c(0,t) &= \frac{1}{2}[\mathbf{F}_1^c(\mathbf{U}_L) + \mathbf{F}_1^c(\mathbf{U}_R) - |\hat{\mathbf{A}}|(\mathbf{U}_R - \mathbf{U}_L)]\,, && (19) \end{aligned}$$

where (19) is obtained by averaging (17) and (18). $\hat{\mathbf{A}}^\pm = \hat{\mathbf{R}}\hat{\mathbf{\Lambda}}^\pm\hat{\mathbf{R}}^{-1}$ and $|\hat{\mathbf{A}}| = \hat{\mathbf{R}}|\hat{\mathbf{\Lambda}}|\hat{\mathbf{R}}^{-1}$, where $\hat{\mathbf{\Lambda}}^\pm = diag(\hat{\lambda}_j^\pm)$ with $\hat{\lambda}_j^\pm = \frac{1}{2}(\hat{\lambda}_j \pm |\hat{\lambda}_j|)$ and $|\hat{\mathbf{\Lambda}}| = diag(|\hat{\lambda}_j|)$.

The approximation of the expansion fan by an expansion shock violates the entropy condition and leads to a wrong flux at the interface, if the expansion fan straddles the time axis, i.e. if the expansion fan has a sonic condition $u - c = 0$. E.g. an expansion shock at the interface will not spread as an expansion fan, but will erroneously stay unchanged with Roe's approximate Riemann solver. We can enforce the entropy condition by introducing a sonic state for a sonic expansion fan, i.e. if $u - c = 0$ or $u + c = 0$, or by adding numerical dissipation to the acoustic waves, if $|u - c|$ or $|u + c|$ become small. The latter approach is employed by Harten's entropy fix: $|\hat{\lambda}_{1,4}| = |\hat{u} \mp \hat{c}|$ are replaced by $Q(\hat{\lambda}_{1,4}) = Q(\hat{u} \mp \hat{c})$, where

$$Q(\lambda) \;=\; \begin{cases} |\lambda| & if\ |\lambda| \geq \delta \\ \frac{1}{2}\left(\frac{\lambda^2}{\delta} + \delta\right) & if\ |\lambda| < \delta \end{cases} , \tag{20}$$

where e.g. $\delta = 0.1\hat{c}$. Note there is no change for a shock.

Application to 1D Euler equations

For a structured 1D grid with grid points $x_{i+1/2}$, $i = 0, 1, 2, ..., I$, $x_{i-1/2} < x_{i+1/2}$, and cells $\Omega_i = [x_{i-1/2}, x_{i+1/2}]$ with lengths $\Delta x_i = x_{i+1/2} - x_{i-1/2}$ (equal volume of corresponding control volume with $\Delta y = \Delta z = 1$) and normals $S_1 = 1$, the cell-centered finite volume discretization of the 1D Euler equations without source term can be expressed as, cf. (5):

$$\frac{d\mathbf{U}_i}{dt}\Delta x_i + \mathbf{F}^c_{1_{i+1/2}} - \mathbf{F}^c_{1_{i-1/2}} = 0 \, . \tag{21}$$

If we define $\mathbf{U}_L = \mathbf{U}_i$ and $\mathbf{U}_R = \mathbf{U}_{i+1}$, we can compute the flux $\mathbf{F}^c_{1_{i+1/2}}$ over the right interface $i + 1/2$ of cell i by Roe's approximate Riemann solver using either right hand side of (17), (18) or (19). Analogously, $\mathbf{F}^c_{1_{i-1/2}}$ is computed with $\mathbf{U}_L = \mathbf{U}_{i-1}$ and $\mathbf{U}_R = \mathbf{U}_i$. This upwind discretization is first-order accurate.

2.3. Total Variation Diminishing Schemes

We shall illustrate the extension of Roe's approximate Riemann solver to a second-order Total Variation Diminishing (TVD) method using the MUSCL approach and slope limiters. The approach can be used for any Riemann solver. Note that the TVD property and thus convergence can only be shown for 1D scalar hyperbolic conservation laws. However, TVD schemes have proved to work fine even for multidimensional nonlinear hyperbolic systems like the Euler equations.

With constant reconstruction of the flow variables inside a cell by the cell average, we obtain a first-order spatial discretization. Employing adjacent cell averages, we can obtain a linear or quadratic reconstruction and thus second- or third-order accurate schemes. With such reconstructions, we can determine the flow variables on either side of a cell interface more accurately. With a quadratic ansatz, van Leer's MUSCL (Monotone Upwind-centered Schemes for Conservation Laws) approach yields for the extrapolated primitive variables at the interface $i + 1/2$:

$$\mathbf{V}^L_{i+1/2} = \mathbf{V}_i + \frac{1}{2}[(1+\kappa)\delta_x \mathbf{V}_{i+1/2} + (1-\kappa)\delta_x \mathbf{V}_{i-1/2}]\frac{\Delta x_i}{2} \quad (22)$$

$$\mathbf{V}^R_{i+1/2} = \mathbf{V}_{i+1} - \frac{1}{2}[(1+\kappa)\delta_x \mathbf{V}_{i+1/2} + (1-\kappa)\delta_x \mathbf{V}_{i+3/2}]\frac{\Delta x_{i+1}}{2} \quad (23)$$

where $\delta_x \mathbf{V}_{i+1/2} = \frac{\mathbf{V}_{i+1} - \mathbf{V}_i}{x_{i+1} - x_i}$ approximates the x-derivative across the interface $i + 1/2$. The midpoint of cell i is $x_i = \frac{1}{2}(x_{i+1/2} + x_{i-1/2})$.

$\kappa = 1$ corresponds to central discretization, which leads to Gibb's phenomenon. But even the upwind schemes, e.g. the second-order pure upwind scheme with $\kappa = -1$, Fromm's second-order scheme with $\kappa = 0$, and the third-order upwind-biased scheme with $\kappa = \frac{1}{3}$, lead to oscillations at discontinuities. Therefore, the slopes are limited by slope limiters. Thereby, the slope across a discontinuity is set to zero, and the non-oscillatory first-order method is recovered, while in smooth flow regions higher accuracy is obtained.

Limiting the slopes in (22) and (23) yields the slope limited MUSCL approach of the primitive variables $\mathbf{V}$ (better than $\mathbf{U}$)

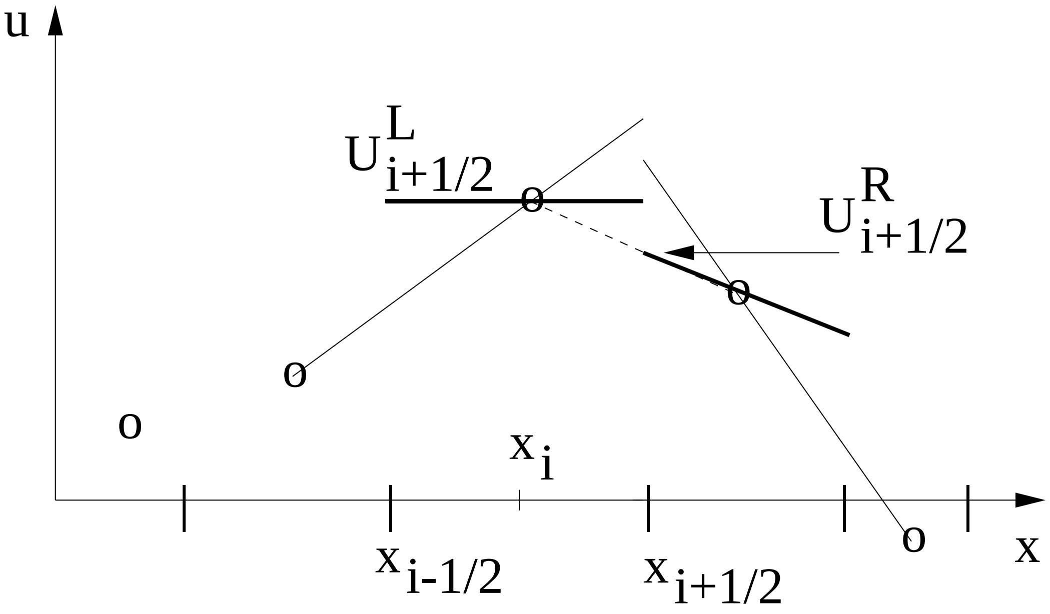
$$\begin{aligned} \mathbf{V}^L_{i+1/2} &= \mathbf{V}_i + \frac{1}{2}[(1+\kappa)\delta^+_x \mathbf{V}_i + (1-\kappa)\delta^-_x \mathbf{V}_i]\frac{\Delta x_i}{2} \qquad (24) \\ \mathbf{V}^R_{i+1/2} &= \mathbf{V}_{i+1} - \frac{1}{2}[(1+\kappa)\delta^-_x \mathbf{V}_{i+1} + (1-\kappa)\delta^+_x \mathbf{V}_{i+1}]\frac{\Delta x_{i+1}}{2} \qquad (25) \end{aligned}$$

where the limited slopes $\delta^{\pm}_x$ can be defined by different limiters, e.g.

$$\begin{aligned} \delta^+_x \mathbf{V}_i &= minmod(\delta_x \mathbf{V}_{i+1/2},\, \beta\, \delta_x \mathbf{V}_{i-1/2}) \;, \\ \delta^-_x \mathbf{V}_i &= minmod(\delta_x \mathbf{V}_{i-1/2},\, \beta\, \delta_x \mathbf{V}_{i+1/2}) \;, \end{aligned}$$

where $-1 \leq \kappa < 1$ and $1 \leq \beta \leq \frac{3-\kappa}{1-\kappa}$ are required for at least second-order TVD schemes. The popular minimum-modulus limiter is defined by

$$
\begin{aligned}
minmod(a,b) &= \begin{cases} a & \text{if } |a| \leq |b| \text{ and } ab > 0 \\ b & \text{if } |b| < |a| \text{ and } ab > 0 \\ 0 & \text{if } ab \leq 0 \end{cases} \\
&= sgn(a)\, max\{0, min\{|a|, sgn(a)b\}\}\,.
\end{aligned}
$$



Limited MUSCL approach for u at $x_{i+1/2}$ with $\kappa = -1$ and $\beta = 1$.

Knowing $\mathbf{V}^{L,R}_{i+1/2}$, we can compute $\mathbf{U}^{L,R}_{i+1/2}$, if we need it. While Roe's approximate Riemann solver (17), (18) or (19) yields a first-order spatial discretization with $\mathbf{F}^c_{1_{i+1/2}} = \mathbf{F}^c_{1_{i+1/2}}(\mathbf{U}_i, \mathbf{U}_{i+1})$ used in (21), we obtain a second-order (third-order for $\kappa = \frac{1}{3}$) spatial discretization (except at discontinuities) with

$$\mathbf{F}^c_{1_{i+1/2}} \;=\; \mathbf{F}^c_{1_{i+1/2}}(\mathbf{U}^L_{i+1/2}, \mathbf{U}^R_{i+1/2}) \tag{26}$$

used in (21), i.e. if Roe's approximate Riemann solver is applied with $\mathbf{U}_L = \mathbf{U}^L_{i+1/2}$ given by (24) and $\mathbf{U}_R = \mathbf{U}^R_{i+1/2}$ given by (25).

2.4. Multidimensional Application

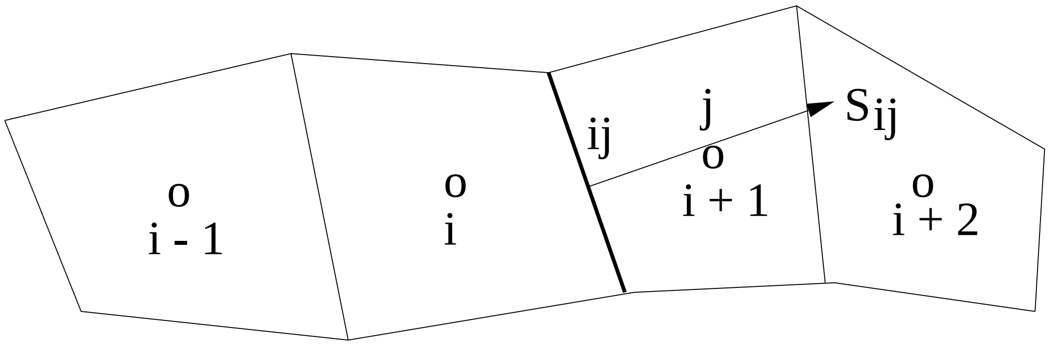
Suppose we want to compute the inviscid flux $\mathbf{F}^c \cdot \mathbf{S}$ in (5) at face ij separating cell i on the left and cell j on the right side using Roe's approximate Riemann solver. Using (19), we can express the Roe flux as:

$$(\mathbf{F}^c \cdot \mathbf{S})_{ij} \;=\; \frac{1}{2}[(\mathbf{F}^c(\mathbf{U}_L) + \mathbf{F}^c(\mathbf{U}_R)) \cdot \mathbf{S}_{ij} \;-\; |\hat{\mathbf{A}}_{ij}|(\mathbf{U}_R - \mathbf{U}_L)]\,, \qquad (27)$$

where $|\hat{\mathbf{A}}_{ij}| = \hat{\mathbf{R}}(\mathbf{n}_{ij})|\hat{\boldsymbol{\Lambda}}(\mathbf{S}_{ij})|\hat{\mathbf{R}}^{-1}(\mathbf{n}_{ij})$ with the outer unit normal $\mathbf{n}_{ij} = \frac{\mathbf{S}_{ij}}{|\mathbf{S}_{ij}|}$ and using the Roe-average (14). Remember $\hat{\mathbf{R}}(\mathbf{n}) = \hat{\mathbf{M}}\hat{\mathbf{T}}(\mathbf{n})$ and $\hat{\mathbf{R}}^{-1}(\mathbf{n})(\mathbf{U}_R - \mathbf{U}_L) = \hat{\mathbf{T}}^{-1}(\mathbf{n})(\mathbf{V}_R - \mathbf{V}_L)$.

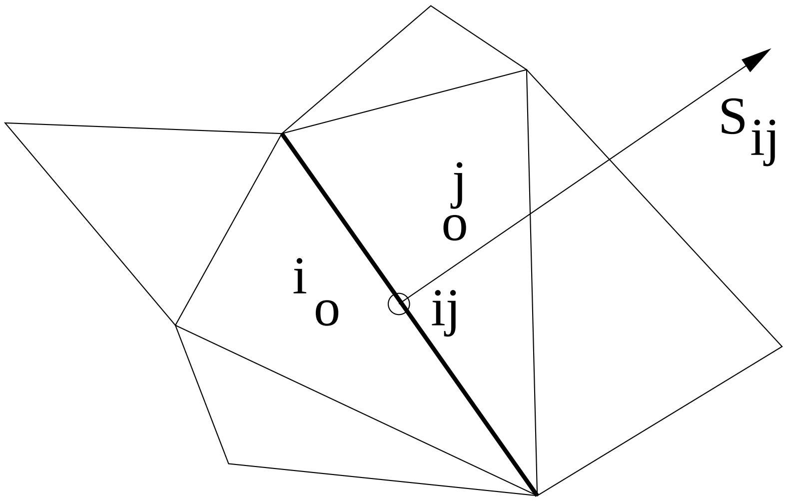
For $\mathbf{U}_L = \mathbf{U}_i$ and $\mathbf{U}_R = \mathbf{U}_j$ in (27), we obtain the first-order Roe flux.

To get a TVD scheme, we can employ the flux limited MUSCL approach (22) and (23). On a structured grid, we can use that ansatz simply in the normal direction, i.e. replace x by n.



Interface ij between cells i and j for structured grid.

On an unstructured grid, we can compute cell gradients $\boldsymbol{\nabla}\mathbf{V}_i$ using the Green-Gauss theorem (6) and approximating the surface integrals by the midpoint rule with the midpoint values on the cell interfaces approximated by the averages of the adjacent cells. Then, $\delta_x \mathbf{V}_{i+1/2}$ can be replaced by $(\mathbf{V}_j - \mathbf{V}_i)(\mathbf{x}_j - \mathbf{x}_i)/|\mathbf{x}_j - \mathbf{x}_i|^2$, $\delta_x \mathbf{V}_{i-1/2}$ by $\boldsymbol{\nabla}\mathbf{V}_i$, $\delta_x \mathbf{V}_{i+3/2}$ by $\boldsymbol{\nabla}\mathbf{V}_j$, $(\ )\frac{\Delta x_i}{2}$ by $(\ ) \cdot (\mathbf{x}_{ij} - \mathbf{x}_i)$ and $-(\ )\frac{\Delta x_{i+1}}{2}$ by $(\ ) \cdot (\mathbf{x}_{ij} - \mathbf{x}_j)$, where $\mathbf{x}_{ij}$ is the midpoint on face ij .



Interface ij between cells i and j for unstructured grid.

If the extrapolated variables are obtained by the slope limited MUSCL approach, Roe's approximate Riemann solver (27) is applied with $\mathbf{U}_L = \mathbf{U}^L_{i+1/2}$ and $\mathbf{U}_R = \mathbf{U}^R_{i+1/2}$.

The inviscid fluxes across the other interfaces of cell i in (5) are determined analogously.

2.5. Numerical Dissipation

In general, approximate Riemann solvers yield quite accurate results for the Euler and Navier-Stokes equations in a wide range of Mach numbers. If one is not interested in high accuracy for strong shocks, contact discontinuities or shear layers, one can save CPU time by replacing an approximate Riemann solver by central inviscid flux discretization plus numerical dissipation. Jameson (1981) proposes for a structured grid:

$$(\mathbf{F}^c \cdot \mathbf{S})_{i+1/2} \;=\; \frac{1}{2}\left[\,\mathbf{F}^c(\mathbf{U}_i) + \mathbf{F}^c(\mathbf{U}_{i+1})\,\right] \cdot \mathbf{S}_{i+1/2} \;+\; \mathbf{D}_{i+1/2}\,, \tag{28}$$

where the numerical dissipation is defined by

$$\begin{aligned} \mathbf{D}_{i+1/2} \;=\; -\lambda_{i+1/2}\,\big[\quad & \epsilon^{(2)}_{i+1/2} \quad (\mathbf{U}_{i+1} - \mathbf{U}_i) \;- \\ & \epsilon^{(4)}_{i+1/2} \quad (\mathbf{U}_{i+2} - 3\mathbf{U}_{i+1} + 3\mathbf{U}_i - \mathbf{U}_{i-1})\,\big] \end{aligned} \tag{29}$$

with the spectral radius of the Jacobian matrix of the inviscid flux

$$\lambda_{i+1/2} = |\frac{1}{2}(\mathbf{u}_i + \mathbf{u}_{i+1}) \cdot \mathbf{S}_{i+1/2}| + \frac{1}{2}(c_i + c_{i+1})|\mathbf{S}_{i+1/2}|$$

and the nonlinear coefficients of the first- and third-order numerical dissipation (finally of second and fourth differences)

$$\epsilon^{(2)}_{i+1/2} = \kappa^{(2)}\, max\{\nu_i, \nu_{i+1}\}\,,$$

$$\epsilon^{(4)}_{i+1/2} = max\{0, \kappa^{(4)} - \epsilon^{(2)}_{i+1/2}\}\,,$$

The pressure sensor

$$\nu_i \;=\; \frac{|p_{i+1} - 2p_i + p_{i-1}|}{p_{i+1} + 2p_i + p_{i-1}}$$

is relatively large near shocks and zero for linear pressure distributions. For transonic flow, typical values of the parameters are $\kappa^{(2)} \approx \frac{1}{4}$ and $\kappa^{(4)} \approx \frac{1}{256}$. If $\epsilon^{(2)}_{i+1/2} > \kappa^{(4)}$, the third-order numerical dissipation is turned off. Thus, near shocks, only the first-order numerical dissipation is active. That procedure corresponds to limiting in the MUSCL approach for TVD schemes.

Note that numerical dissipation introduces artificial viscosity. Since it scales with the grid spacing, it is reduced with grid refinement. In shear layers, we must make sure that the artificial viscosity is negligible compared with the true physical viscosity.

2.6. Boundary Conditions

2.6.1. Solid Wall

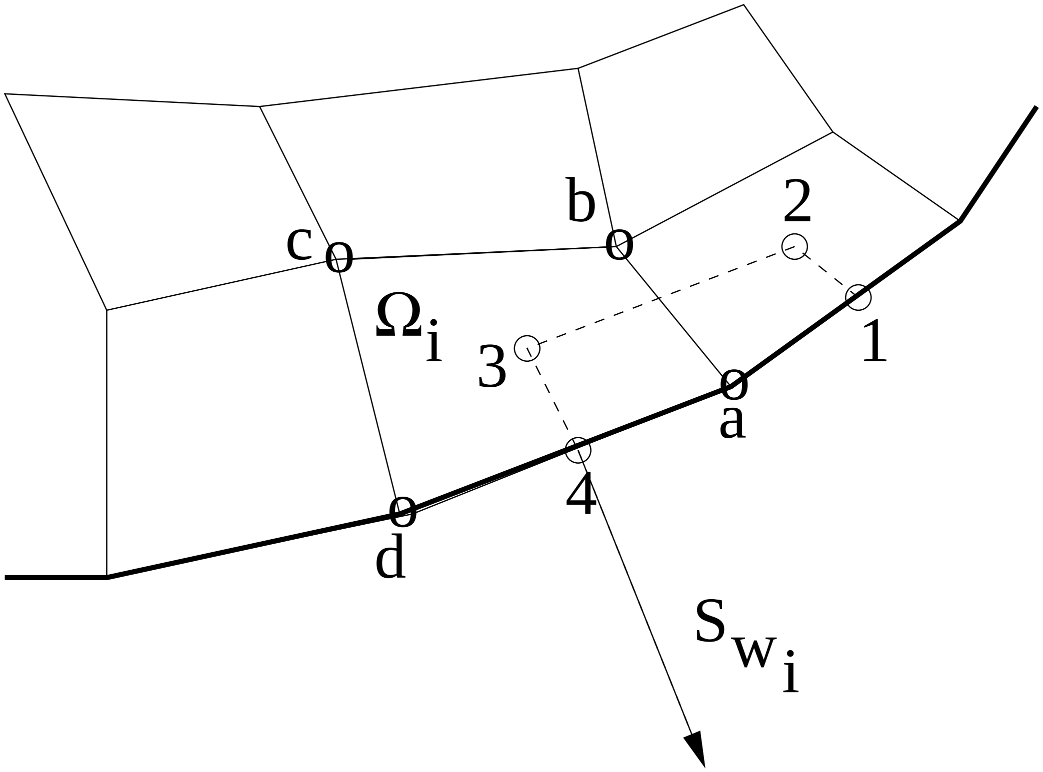
At a stationary solid wall next to cell i, only the average wall pressure p_{w_i} needs to be determined, because

$$\int_{\Gamma_{w_i}} \mathbf{F}^c \cdot \mathbf{n} \, dA \;=\; p_{w_i} \begin{pmatrix} 0 \\ \mathbf{S}_{w_i} \\ 0 \end{pmatrix} \tag{30}$$

where $\mathbf{S}_{w_i}$ is the outer wall face normal $\int_{\Gamma_{w_i}} \mathbf{n} dA$.

MacCormack: “All you need is p .“

Often, p_{w_i} can be approximated by the pressure p_i in cell i. The approximation is justified for the Euler equations, if the wall is plane, and for the Navier-Stokes equations, if the boundary layer approximation applies (i.e. $Re_{\infty L} \gg 1$ and no separation nor reattachment). Otherwise, the normal momentum equation for inviscid and viscous flow, respectively, has to be discretized.



Control volume 1234 to compute gradients at wall point a .

The gradients at the wall points a and d can be computed as indicated in 1.4. using the above control volume for a and the wall boundary conditions for viscous flow, e.g. $\mathbf{u}_w = 0$ and T_w given, on $4 - a - 1$. A simpler approximation is obtained from e.g. $\frac{\partial T_{w_i}}{\partial n} \approx \frac{T_4 - T_3}{|\mathbf{x}_4 - \mathbf{x}_3|} \frac{\mathbf{S}_{w_i}}{|\mathbf{S}_{w_i}|}$. If T_w is given, μ_w and κ_w can be computed directly. Otherwise, μ_w and κ_w can be approximated by adjacent cell values.

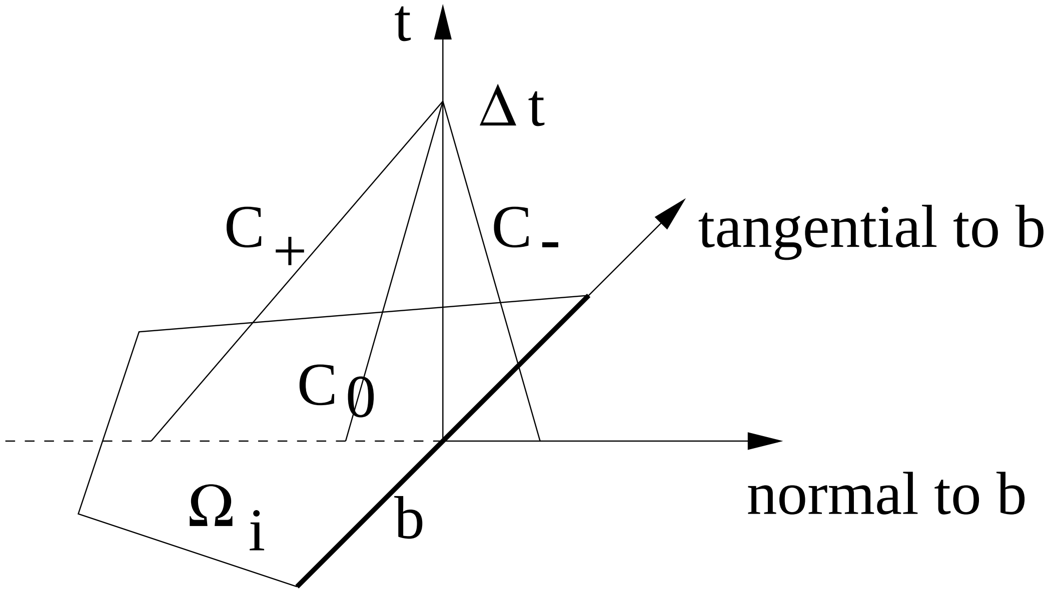
Thus, the viscous wall flux for a stationary wall can be computed from

$$\int_{\Gamma_{w_i}} \mathbf{F}^v \cdot \mathbf{n}\, dA \;=\; \left(\begin{array}{l} 0 \\ \boldsymbol{\tau}_{w_i} \cdot \mathbf{S}_{w_i} \\ k_{w_i} (\boldsymbol{\nabla} T)_{w_i} \cdot \mathbf{S}_{w_i} \end{array} \right) . \tag{31}$$

For an adiabatic wall, the wall heat flux is zero, i.e. the last component of (31) is zero.

2.6.2. Inflow and Outflow

The boundary conditions indicated in II. 2.6.2 can be used. Instead of employing the compatibility relations for the outgoing waves, one can often simply extrapolate the corresponding Riemann invariants.



Let us consider a subsonic outflow boundary b ($0 < \mathbf{u}_b \cdot \mathbf{n} < c_b$) adjacent to cell i with outer unit normal vector $\mathbf{n}$. Assume that the time step Δt is chosen such that the characteristics C_+ with $\frac{dn}{dt} = \mathbf{u}_b \cdot \mathbf{n} + c$ and C_0 with $\frac{dn}{dt} = \mathbf{u}_b \cdot \mathbf{n}$ come from cell i. Then the corresponding Riemann invariants can be approximated (remember we are neglecting tangential derivatives and source terms) by

$$\begin{aligned} \frac{p_b}{\rho_b^\gamma} &= \frac{p_i}{\rho_i^\gamma} \\ \mathbf{u}_b - (\mathbf{u}_b \cdot \mathbf{n})\mathbf{n} &= \mathbf{u}_i - (\mathbf{u}_i \cdot \mathbf{n})\mathbf{n} \qquad (32) \\ \mathbf{u}_b \cdot \mathbf{n} + \frac{2}{\gamma - 1} c_b &= \mathbf{u}_i \cdot \mathbf{n} + \frac{2}{\gamma - 1} c_i \,. \qquad (33) \end{aligned}$$

Using the boundary condition $p_b = p_a$, where p_a is the ambient pressure, we can determine $\mathbf{V}_b$ and thus compute the inviscid boundary flux $\mathbf{F}^c(\mathbf{U}_b)$. The viscous boundary flux $\mathbf{F}^v(\mathbf{U}_b)$ is usually neglected.

3. Time Discretization

The cell-centered finite volume discretization in cell i (5) can be expressed in semi-discrete form

$$\frac{d\mathbf{U}_i}{dt} = \mathbf{R}(t, \mathbf{U}_i, \mathbf{U}_j) , \tag{34}$$

where the residual

$$\mathbf{R}(t, \mathbf{U}_i, \mathbf{U}_j) = -\frac{1}{V_i} \left(\sum_{l=1}^{4} [\mathbf{F}_1 \, S_1 + \mathbf{F}_2 \, S_2]_l \right) + \mathbf{F}_{e_i}$$

is defined by the particular flux and source term discretizations. The fluxes and source term are functions of the cell average $\mathbf{U}_i$ and neighboring cell averages denoted by $\mathbf{U}_j$. Time dependent boundary conditions enter via the boundary fluxes and make the residual a function of time, too.

Thus, the finite volume discretization leads to a large system of ODEs

$$\frac{d\mathbf{U}}{dt} = \mathbf{R}(t, \mathbf{U}) \ , \tag{35}$$

where $\mathbf{U}$ denotes the vector with all the cell averages $\mathbf{U}_i$ in the computational domain. The i-th vector equation of (35) is defined by (34).

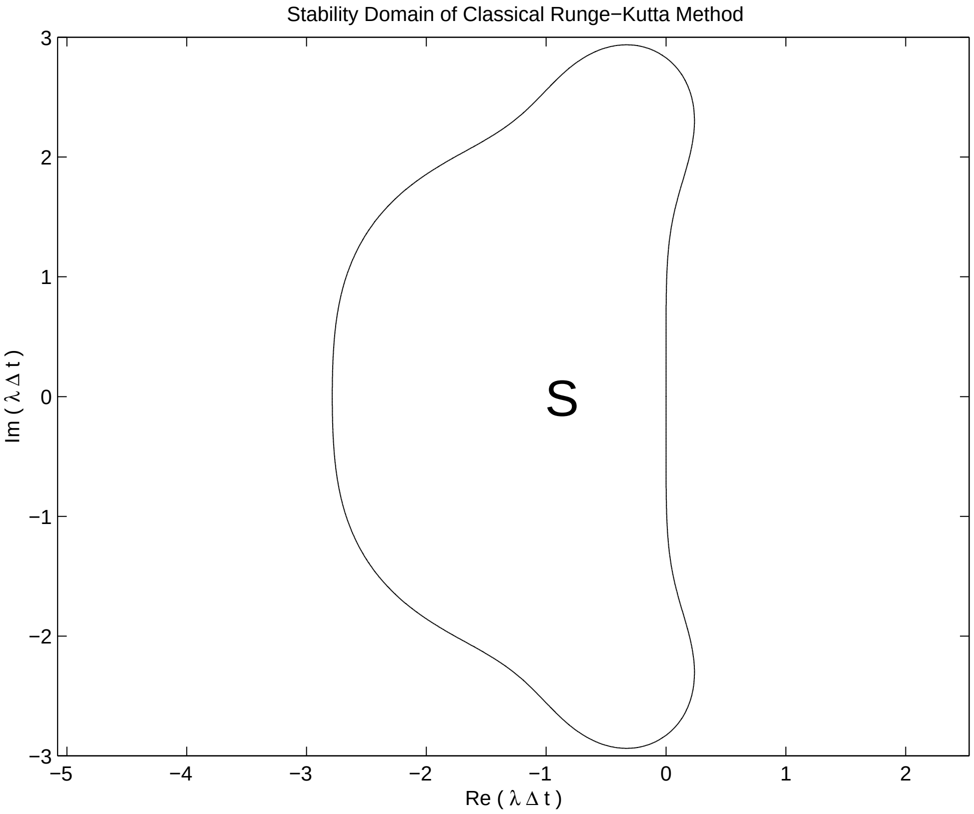
The system of ODEs (35) can be solved by any suitable ODE-solver. We shall outline examples of two popular classes: explicit Runge-Kutta methods and implicit linear multistep methods.

3.1. Explicit Runge-Kutta Methods

Unsteady non-stiff flow problems, i.e. the Euler and Navier-Stokes equations with moderate Mach and Reynolds numbers, can be integrated in time using the classical fourth-order Runge-Kutta method:

$$\begin{aligned}
\mathbf{U}^{(1)} &= \mathbf{U}^n + \tfrac{\Delta t}{2} \quad \mathbf{R}(t_n, \mathbf{U}^n)\,, \\
\mathbf{U}^{(2)} &= \mathbf{U}^n + \tfrac{\Delta t}{2} \quad \mathbf{R}(t_n + \frac{\Delta t}{2}, \mathbf{U}^{(1)})\,, \\
\mathbf{U}^{(3)} &= \mathbf{U}^n + \Delta t \quad \mathbf{R}(t_n + \frac{\Delta t}{2}, \mathbf{U}^{(2)})\,, \\
\mathbf{U}^{n+1} &= \mathbf{U}^n + \tfrac{\Delta t}{6} \quad (\,\mathbf{R}(t_n, \mathbf{U}^n) + 2\mathbf{R}(t_n + \frac{\Delta t}{2}, \mathbf{U}^{(1)}) + \,, \\
&\qquad\qquad 2\mathbf{R}(t_n + \frac{\Delta t}{2}, \mathbf{U}^{(2)}) + \mathbf{R}(t_n + \Delta t, \mathbf{U}^{(3)})\,)\,. \qquad (36)
\end{aligned}$$

When applied to the scalar test problem $\frac{du}{dt} = \lambda u$, $\lambda \in \mathbf{C}$ constant, the Runge-Kutta method (36) is stable, if $\lambda\Delta t \in S$, where S is the kidney shaped stability domain. If λ is real, the stability condition is $-2\sqrt{2} \leq \lambda\Delta t \leq 0$. If λ is imaginary , the stability condition is $-2\sqrt{2} \leq \lambda\Delta t \leq 2\sqrt{2}$.



A linear stability condition for the Navier-Stokes equations can be derived from the von Neumann stability analysis applied to the second-order central finite difference method (Müller (1990)):

$$\Delta t \;\leq\; \min_i \{\Delta t_i^c, \Delta t_i^v\}\,. \tag{37}$$

The inviscid time step Δt_i^c is defined by

$$\Delta t_i^c \;=\; 2\,CFL\frac{V_i}{\lambda_i^c} \tag{38}$$

with CFL depending on the Runge-Kutta method (see below) and λ_i^c determined by summing the spectral radii of the Jacobian matrices of the inviscid fluxes over the faces ij of the control volume Ω_i (4 faces for quadrilateral)

$$\lambda_i^c \;=\; \sum_{j=1}^{4}(|\mathbf{u}\cdot\mathbf{S}| + c|\mathbf{S}|)_{ij}\,.$$

The viscous time step Δt_i^v is approximated by

$$\Delta t_i^v \;=\; 2\,|RK|\frac{V_i}{\lambda_i^v} \tag{39}$$

with RK depending on the Runge-Kutta method (see below) and λ_i^v determined by summing the simplified spectral radii of the Jacobian matrices of the viscous fluxes over the faces ij of the control volume Ω_i (4 faces for quadrilateral)

$$\lambda_i^v \;=\; \frac{1}{4V_i}\sum_{j=1}^{4}(\nu|\mathbf{S}|^2)_{ij}$$

with $\nu = \frac{\mu}{\rho}\max\{\frac{4}{3},\frac{\gamma}{Pr}\}$. If we approximate the cell face values by the values in cell i, we obtain on a Cartesian grid:

$$\Delta t_i^c \;=\; CFL\left[\frac{|u_i|+c_i}{\Delta x}+\frac{|v_i|+c_i}{\Delta y}\right]^{-1},$$

$$\Delta t_i^v = |RK| \left[\nu_i \left(\frac{1}{\Delta x^2} + \frac{1}{\Delta y^2} \right) \right]^{-1} .$$

The stability bounds CFL and RK of the Runge-Kutta method are chosen such that the set

$$\{ z \in \mathbf{C} \,|\, RK \leq Re(z) \leq 0 \text{ and } |Im(z)| \leq CFL \}$$

is contained in the stability domain S of the Runge-Kutta method. E.g. for the classical Runge-Kutta method (36): $CFL = 2$ and $RK = -1.5$.

Steady state problems $\mathbf{R}(\mathbf{U}) = 0$ are often calculated by solving $\frac{d\mathbf{U}}{dt} = \mathbf{R}(\mathbf{U})$ for the steady state. Then, low-storage Runge-Kutta methods like the following one are preferred:

$$\begin{aligned}
\mathbf{U}^{(1)} &= \mathbf{U}^n + \tfrac{\Delta t}{4} \quad \mathbf{R}(\mathbf{U}^n)\,, \\
\mathbf{U}^{(2)} &= \mathbf{U}^n + \tfrac{\Delta t}{3} \quad \mathbf{R}(\mathbf{U}^{(1)})\,, \\
\mathbf{U}^{(3)} &= \mathbf{U}^n + \tfrac{\Delta t}{2} \quad \mathbf{R}(\mathbf{U}^{(2)})\,, \\
\mathbf{U}^{n+1} &= \mathbf{U}^n + \Delta t \quad \mathbf{R}(\mathbf{U}^{(3)})\,.
\end{aligned} \tag{40}$$

This four-stage Runge-Kutta method has the same stability domain as the classical one (36). (40) is fourth- and second-order accurate for linear and nonlinear ODEs, respectively. Many other Runge-Kutta methods may be found in the literature, e.g. a third-order one for TVD methods by Shu (SIAM J. Sci. Comput., 9(6):1073-1084, 1988).

Local Time Stepping

The convergence to steady state can be enhanced by local time stepping, i.e. by choosing the time step for each cell individually to get optimal local stability (e.g. good damping for multigrid):

$$\Delta t_i \;=\; \min\left\{\Delta t_i^c, \Delta t_i^v\right\}. \tag{41}$$

Here, the minimum of the local inviscid and viscous time steps is taken, whereas in (37) the minimum over the local time steps in all cells is taken. Instead of (41), one can use the geometric average to get a smoother transition between inviscid and viscous time steps:

$$\Delta t_i \;=\; \frac{\Delta t_i^c\,\Delta t_i^v}{\Delta t_i^c + \Delta t_i^v}\,.$$

A considerable convergence acceleration can be achieved with a multigrid method, cf. e.g. Wesseling (1992).

3.2. Implicit Linear Multistep Methods

To avoid restrictive time step conditions, implicit methods are preferred for stiff problems, e.g. large Reynolds number flows, and for steady state problems.

To solve (35) for the steady state, the implicit Euler method is suitable:

$$\frac{\mathbf{U}^{n+1} - \mathbf{U}^{n}}{\Delta t} = \mathbf{R}(\mathbf{U}^{n+1})\,. \tag{42}$$

For time dependent problems, the second-order backward differencing formula

$$\frac{\frac{3}{2}\mathbf{U}^{n+1} - 2\mathbf{U}^{n} + \frac{1}{2}\mathbf{U}^{n-1}}{\Delta t} = \mathbf{R}(t_{n+1}, \mathbf{U}^{n+1}) \tag{43}$$

is preferred over the second-order trapezoidal rule

$$\frac{\mathbf{U}^{n+1} - \mathbf{U}^{n}}{\Delta t} = \frac{1}{2}\left(\mathbf{R}(t_n, \mathbf{U}^{n}) + \mathbf{R}(t_{n+1}, \mathbf{U}^{n+1})\right), \tag{44}$$

because the former has a much larger stability domain.

The nonlinear equation (42) for $\mathbf{U}^{n+1}$ (analogously (43) and (44)) can be solved by Newton's method (often called the Newton-Raphson method) starting with $\mathbf{U}^{\nu=0} = \mathbf{U}^n$:

$$\left(\frac{1}{\Delta t} - \frac{\partial \mathbf{R}(\mathbf{U}^\nu)}{\partial \mathbf{U}}\right) \Delta \mathbf{U}^\nu \;=\; -\frac{\mathbf{U}^\nu - \mathbf{U}^n}{\Delta t} + \mathbf{R}(\mathbf{U}^\nu)\;,\; \nu = 0, 1, 2, \ldots \quad (45)$$

where $\Delta \mathbf{U}^\nu \;=\; \mathbf{U}^{\nu+1} - \mathbf{U}^\nu$, until a convergence criterion like $\|\Delta \mathbf{U}^\nu\| \le \epsilon$ is reached. Then, $\mathbf{U}^{\nu+1}$ is accepted as $\mathbf{U}^{n+1}$. The Jacobian matrix of the residual $\frac{\partial \mathbf{R}(\mathbf{U}^\nu)}{\partial \mathbf{U}}$ is usually simplified, e.g. for the Roe-flux in (21) by

$$\hat{\mathbf{A}}^{-\nu}_{i+1/2}(\Delta \mathbf{U}^\nu_{i+1} - \Delta \mathbf{U}^\nu_i) + \hat{\mathbf{A}}^{+\nu}_{i-1/2}(\Delta \mathbf{U}^\nu_i - \Delta \mathbf{U}^\nu_{i-1})\,.$$

Often, only the first iteration of Newton's method is computed, i.e. (45) for $\nu = 0$ only.

For each ν, (45) represents a sparse linear system. For 1D flow problems, the linear system has a small bandwidth and can be solved directly by LU factorization. For multidimensional flow problems, (45) can be solved by any suitable iterative method, cf. Saad (1996). Popular methods have been:

- Beam and Warming scheme (1978).
  The matrix is factorized into 1D systems in each grid direction. E.g. $(\mathbf{I}+\mathbf{L}_x+\mathbf{L}_y)\Delta\mathbf{U} \approx (\mathbf{I}+\mathbf{L}_x)(\mathbf{I}+\mathbf{L}_y)\Delta\mathbf{U}$. Each 1D system is solved directly.

- Lower-upper symmetric Gauss-Seidel (LU-SGS) method by Yoon and Jameson (1988). Symmetric Gauss-Seidel relaxation method starting in the lower left corner of the structured grid indices (i, j) and going back and forth to upper right corner on diagonals $i + j = constant$. In 2D, only 4×4 systems have to be solved.

- Generalized minimum residual (GMRES) method by Saad and Schultz (1986). Generalization of the conjugate gradient method. The solution of the linear system $Ax = b$ is sought in the Krylov subspace spanned by the residual $r_0 = b - Ax_0$ of the initial guess x_0 and a certain number of powers of A applied to r_0. Only matrix-vector and vector-vector operations are involved, but enough memory and preconditioning is needed.

To solve (43) for unsteady problems, Jameson (1991) introduces a dual time τ and replaces the corresponding left hand side of (45) by the dual time derivative. The resulting equation

$$\frac{d\mathbf{U}}{d\tau} = -\frac{\frac{3}{2}\mathbf{U} - 2\mathbf{U}^n + \frac{1}{2}\mathbf{U}^{n-1}}{\Delta t} + \mathbf{R}(t_{n+1}, \mathbf{U}) \qquad (46)$$

is solved for the steady state using all available convergence accelerations (local time stepping, Runge-Kutta method of type (40) with residual smoothing, multigrid) to determine the steady state solution of (46), i.e. $\mathbf{U}^{n+1}$, in $O(10)$ multigrid cycles for the unsteady Euler equations.

3.3. Time Derivative Preconditioning

The objective of the time derivative preconditioning is to modify the compressible flow equations such that the convergence to steady state is faster with the modified equations than with the original equations when using a time marching method for their numerical solution. The approach has been developed for low Mach number flow, for which the wave speeds are very disparate. We consider the 2D Euler equations

$$\frac{\partial \mathbf{U}}{\partial t} + \frac{\partial \mathbf{F}_1^c(\mathbf{U})}{\partial x} + \frac{\partial \mathbf{F}_2^c(\mathbf{U})}{\partial y} = 0 \tag{47}$$

where $\mathbf{U} = (\rho, \rho u, \rho v, \rho E)^T, \mathbf{F}_1^c = (\rho u, \rho u^2 + p, \rho v u, \rho H u)^T,$

$\mathbf{F}_2^c = (\rho v, \rho u v, \rho v^2 + p, \rho H v)^T.$

The eigenvalues of a linear combination of the Jacobian matrices $\mathbf{A}_1 = \frac{\partial \mathbf{F}_1^c}{\partial \mathbf{U}}$ and $\mathbf{A}_2 = \frac{\partial \mathbf{F}_2^c}{\partial \mathbf{U}}$, i.e. $\mathbf{C}(\mathbf{k}) = k_1 \mathbf{A}_1 + k_2 \mathbf{A}_2$ with $|\mathbf{k}| = 1$, are

$$\lambda_1 = \mathbf{u} \cdot \mathbf{k} - c\,, \quad \lambda_2 = \lambda_3 = \mathbf{u} \cdot \mathbf{k}, \quad \lambda_4 = \mathbf{u} \cdot \mathbf{k} + c \tag{48}$$

Thus, the hyperbolic system is ill-conditioned for $M \to 0$, because the characteristic condition number of $\mathbf{C}(\mathbf{k})$, i.e. $|\lambda_{max}/\lambda_{min}|$, reads

$$\kappa(\mathbf{C}(\mathbf{k})) = \frac{|\mathbf{u}\cdot\mathbf{k}| + c}{|\mathbf{u}\cdot\mathbf{k}|} \approx 1 + \frac{1}{M} \longrightarrow \infty \quad for \quad M \longrightarrow 0 \tag{49}$$

Thus, for low Mach number flows, the systems are ill-conditioned, when using implicit methods. For explicit methods, the stability condition requires very small time steps, and very many time levels are needed for steady state calculations of low Mach number flows.

In order to obtain a well-conditioned hyperbolic system without changing the steady state, the time derivative in (47) is multiplied by a preconditioning matrix $\mathbf{P}^{-1}$, cf. E. Turkel, “Preconditioning Techniques in CFD”, Annu. Rev. Fluid Mech. 1999, 31:385-416.

$$\mathbf{P}^{-1}\frac{\partial \mathbf{U}}{\partial t} + \frac{\partial \mathbf{F}_1^c}{\partial x} + \frac{\partial \mathbf{F}_2^c}{\partial y} \;=\; 0\,. \tag{50}$$

As an example, we consider the Weiss-Smith preconditioner

$$\mathbf{P}^{-1} = \frac{\partial \mathbf{U}}{\partial \mathbf{W}} \mathbf{P}_w^{-1} \frac{\partial \mathbf{W}}{\partial \mathbf{U}}\,, \tag{51}$$

where $d\mathbf{W} = \left(\frac{dp}{\rho c}, du, dv, dp - c^2 d\rho\right)^T$ are the symmetrizing variables and

$$\mathbf{P}_w^{-1} = \begin{pmatrix} M^{-2} & & & 0 \\ & 1 & & \\ & & 1 & \\ 0 & & & 1 \end{pmatrix}\,. \tag{52}$$

The eigenvalues of $\mathbf{PC}(\mathbf{k})$ are

$$\begin{aligned}
\lambda_1 &= \frac{1}{2}(1+M^2)\mathbf{u}\cdot\mathbf{k} - \left[\frac{1}{4}(1-M^2)^2(\mathbf{u}\cdot\mathbf{k})^2 + M^2c^2\right]^{\frac{1}{2}} \\
\lambda_2 &= \lambda_3 = \mathbf{u}\cdot\mathbf{k} \qquad (53) \\
\lambda_4 &= \frac{1}{2}(1+M^2)\mathbf{u}\cdot\mathbf{k} + \left[\frac{1}{4}(1-M^2)^2(\mathbf{u}\cdot\mathbf{k})^2 + M^2c^2\right]^{\frac{1}{2}}
\end{aligned}$$

The system (50) with (51) is well conditioned in the incompressible limit, because the characteristic condition number of $\mathbf{PC}(\mathbf{k})$ for $\mathbf{u}\cdot\mathbf{k} \neq 0$ behaves as

$$\kappa(\mathbf{PC}(\mathbf{k})) \longrightarrow \frac{\sqrt{5}+1}{\sqrt{5}-1} = 2.618 \quad for \quad M \longrightarrow 0 \qquad (54)$$

When employing the time derivative preconditioning, the Riemann solver, artificial viscosity and characteristic boundary conditions should be designed for the system

$$\frac{\partial \mathbf{U}}{\partial t} + \mathbf{P}\frac{\partial \mathbf{F}_1^c}{\partial x} + \mathbf{P}\frac{\partial \mathbf{F}_2^c}{\partial y} = 0 \tag{55}$$

and not for the original Euler equations, because the eigenvalues and eigenvectors are changed by time derivative preconditioning. Using the correct formulation of the numerical dissipation (either inherent in upwind schemes or added to central schemes) for (55), the numerical viscosity scales according to (54) and no longer according to (49). Thus, the accuracy of low Mach number computations is considerably increased, when solving the preconditioned system (55) or (50) with the correctly scaled numerical dissipation.

3.4. Perturbation Formulation

In non-reacting low speed flow, the changes in pressure, density and temperature are usually extremely small, e.g. $\frac{\Delta p}{p} = O(M)$ and $O(M^2)$ for unsteady and steady flow, respectively. The computation of such a small change, e.g. $\Delta p = p_R - p_L$, leads to cancellation, if $\frac{|\Delta p|}{p} < \epsilon_m$, where ϵ_m is the machine accuracy of the computer. $\epsilon_m = 2^{-24} \approx 10^{-7}$ and $\epsilon_m = 2^{-53} \approx 10^{-16}$ for IEEE single and double precision, resp.

In order to avoid cancellation for compressible low Mach number computations and in order to retain the conservative form of the governing equations, we introduce the perturbed conservative variables cf. J. Sesterhenn, B. Müller, H. Thomann, JCP 151, 597-615 (1999),

$$\mathbf{U}'(\mathbf{x}, t) \; = \; \mathbf{U}(\mathbf{x}, t) - \mathbf{U}_0, \tag{56}$$

where $\mathbf{U}_0 = (\rho_0, 0, \frac{p_0}{\gamma-1})^T$ is the vector of the conservative variables at stagnation conditions. We have the following relations

$$\begin{aligned}
\mathbf{u}' &= \frac{(\rho\mathbf{u})'}{\rho_0+\rho'}, \\
p' &= (\gamma-1)[(\rho E)' - \frac{1}{2}(\rho\mathbf{u})'\cdot\mathbf{u}'], \\
(\rho H)' &= (\rho E)' + p'.
\end{aligned}$$

The Euler equations in perturbation formulation read

$$\begin{aligned}
\frac{\partial\rho'}{\partial t} + \boldsymbol{\nabla}\cdot(\rho\mathbf{u})' &= 0\,, \\
\frac{\partial(\rho\mathbf{u})'}{\partial t} + \boldsymbol{\nabla}\cdot(\rho\mathbf{u})'\mathbf{u}' + \boldsymbol{\nabla}p' &= 0\,, \qquad (57) \\
\frac{\partial(\rho E)'}{\partial t} + \boldsymbol{\nabla}\cdot((\rho H)'\mathbf{u}' + (\rho H)_0\mathbf{u}') &= 0\,.
\end{aligned}$$

We recommend the perturbation formulation for steady and unsteady compressible low Mach number flow.

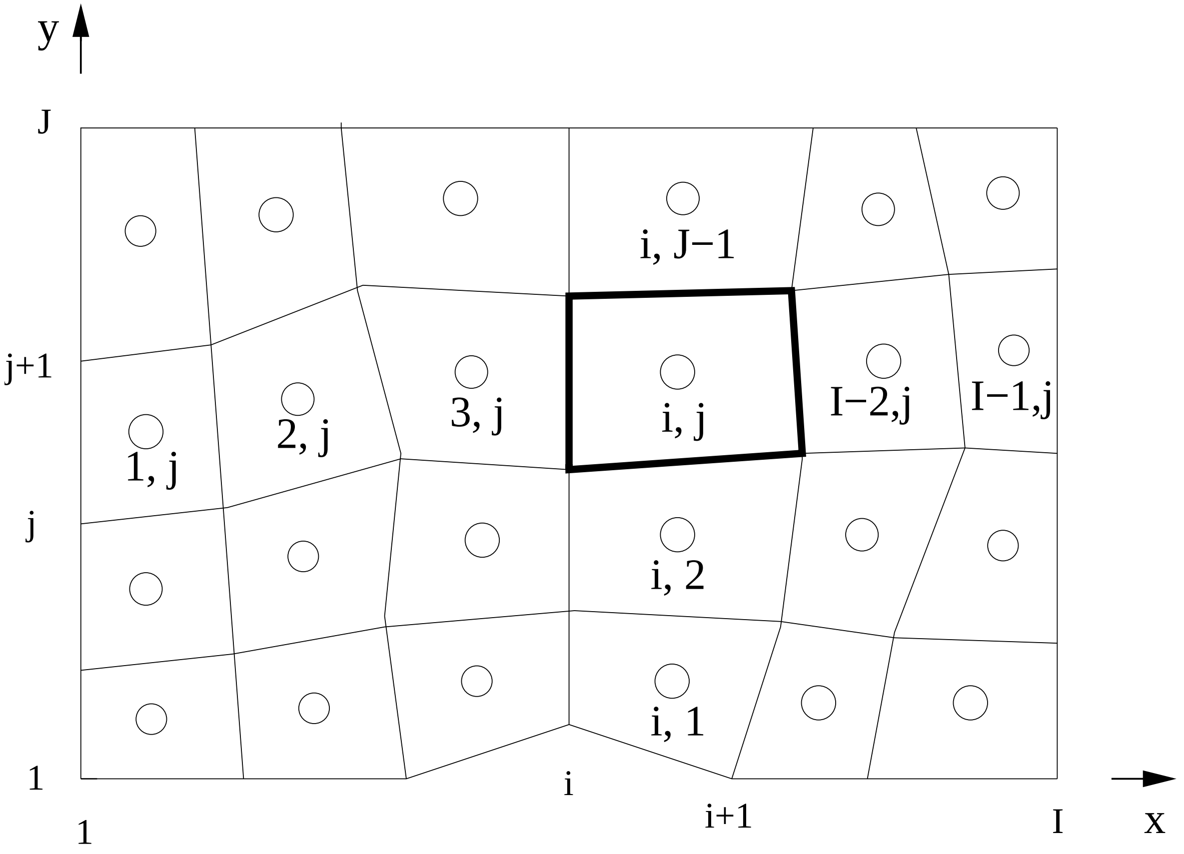
4. Finite Volume Methods on Structured and Unstructured Grids

Finite volume methods on structured grids are in general more accurate but less flexible for complex geometries and for grid adaptation than finite volume methods on unstructured grids. To retain accuracy in thin shear layers, e.g. boundary layers, unstructured grid methods usually use quadrilaterals in 2D and prisms or hexahedra in 3D shear layers. Outside shear layers, unstructured finite volume methods prefer triangles and tetrahedra, respectively.

4.1. Structured Finite Volume Method

We consider the cell-centered finite volume method discussed in section 1. on a structured grid. For a 2D structured grid, there exists a coordinate transformation from the physical x-y plane to the computational ξ-η plane, i.e. the i-j plane. In the figure below, $I = 7$

and $J = 5$ grid points in the i- and j-directions, respectively, constitute a 7×5 H-type grid for a bump.



Structured 7×5 H-type grid for a bump.

4.1.1. Metric Terms

Suppose the coordinates of the grid points $\mathbf{x}_{i,j}$, $i = 1, ..., I$, $j = 1, ..., J$, have been provided by a grid generation program. The face normals **SI** in the i-direction are determined by (cf. p. 9 and figure above)

$$\mathbf{SI}_{i,j} = \begin{pmatrix} y_{i,j+1} - y_{i,j} \\ -(x_{i,j+1} - x_{i,j}) \end{pmatrix}, \; i = 1, ..., I, \; j = 1, ..., J-1.$$

Similarly, the face normals **SJ** in the j-direction are determined by

$$\mathbf{SJ}_{i,j} = \begin{pmatrix} y_{i,j} - y_{i+1,j} \\ -(x_{i,j} - x_{i+1,j}) \end{pmatrix}, \; i = 1, ..., I-1, \; j = 1, ..., J.$$

The areas $\mathbf{V}$ of the quadrilateral cells are computed as

$$\mathbf{V}_{i,j} = 0.5\left|(x_{i+1,j+1} - x_{i,j})(y_{i,j+1} - y_{i+1,j}) - (y_{i+1,j+1} - y_{i,j})(x_{i,j+1} - x_{i+1,j})\right|, \\ i = 1, ..., I-1,\ j = 1, ..., J-1.$$

Thus, with two loops, one over i and one over j, the metric terms of a 2D structured grid can be determined. The formula for 3D are given in the appendix.

4.1.2. Numerical Fluxes

With the cell averages $\mathbf{U}_{i,j},\ i = 1, ..., I-1,\ j = 1, ..., J-1$, at time level n and the face normals, the numerical fluxes in the interior can be determined. The left MUSCL extrapolation (24) at the face $\Gamma_{i+1/2,j}$ with normal $\mathbf{SI}_{i+1,j}$, cf. the structured grid figure above, needs $\mathbf{U}_{i-1,j}$, $\mathbf{U}_{i,j}$, $\mathbf{U}_{i+1,j}$. Thus, at all interior faces in the i-direction, except for $\Gamma_{1+1/2,j}$ with normal $\mathbf{SI}_{2,j}$, the left MUSCL extrapolation (24) can be computed. At $\Gamma_{1+1/2,j}$ with normal $\mathbf{SI}_{2,j}$, we simply use the zeroth order extrapolation $\mathbf{U}^L_{1+1/2,j} = \mathbf{U}_{1,j}$. Similarly, the right MUSCL extrapolation (25) at the face $\Gamma_{i+1/2,j}$ with normal $\mathbf{SI}_{i+1,j}$ needs $\mathbf{U}_{i,j}$, $\mathbf{U}_{i+1,j}$, $\mathbf{U}_{i+2,j}$ and can therefore be determined at all interior faces in the i-direction, except for $\Gamma_{I-3/2,j}$, where $\mathbf{U}^R_{I-3/2,j} = \mathbf{U}_{I-1,j}$ is used. Then, an approximate Riemann solver can be used to compute the inviscid flux $\mathbf{FI}^c_{i+1,j}$ over face $\Gamma_{i+1/2,j}$ with normal $\mathbf{SI}_{i+1,j}$. Roe's approximate Riemann solver (27) yields

$$\mathbf{FI}^c_{i+1,j} \;=\; 0.5[(\mathbf{F}^c(\mathbf{U}^L_{i+1/2,j})+\mathbf{F}^c(\mathbf{U}^R_{i+1/2,j}))\cdot\mathbf{SI}_{i+1,j} - |\hat{\mathbf{A}}_{i+1,j}|(\mathbf{U}^R_{i+1/2,j}-\mathbf{U}^L_{i+1/2,j})]\,, \tag{58}$$

where $|\hat{\mathbf{A}}_{i+1,j}|$ is computed with the Roe-average (14) of $\mathbf{U}^L_{i+1/2,j}$ and $\mathbf{U}^R_{i+1/2,j}$ and the face normal $\mathbf{SI}_{i+1,j}$.

The inviscid flux $\mathbf{FJ}^c_{i,j+1}$ over face $\Gamma_{1,j+1/2}$ with normal $\mathbf{SJ}_{i,j+1}$ can be determined analogously.

The viscous fluxes $\mathbf{FI}^v_{i+1,j}$ and $\mathbf{FJ}^v_{i,j+1}$ can be computed as described in section 1.4. If a shear layer is aligned in the i-direction, e.g. approximately parallel to the grid line $j = 1$ for a boundary layer, the viscous variations in the i-direction can often be neglected, because they are much smaller than those in the j-direction. That viscous flux approximation is called thin layer approximation. Then, the gradients at face $\Gamma_{i,j+1/2}$ with normal $\mathbf{SJ}_{i,j+1}$ can be approximated by a simplification of (8):

$$(\boldsymbol{\nabla} T)_{i,j+1/2} \;\approx\; \frac{2}{V_{i,j} + V_{i,j+1}} (T_{i,j+1} - T_{i,j}) \mathbf{SJ}_{i,j+1} \,. \tag{59}$$

With the viscosity average $\mu_{i,j+1/2} \;\approx\; 0.5(\mu_{i,j} + \mu_{i,j+1})$, the viscous flux in the j-direction is approximated by

$$\mathbf{FJ}^v_{i+1,j} \;= \mathbf{F}^v_{i,j+1/2} \cdot \mathbf{SJ}_{i,j+1} \,. \tag{60}$$

The inviscid and viscous flux approximations are added to yield the numerical fluxes in the i- and j-directions, i.e. $\mathbf{FI} = \mathbf{FI}^c + \mathbf{FI}^v$ and $\mathbf{FJ} = \mathbf{FJ}^c + \mathbf{FJ}^v$.

The numerical fluxes at the boundaries are computed as discussed in section 2.6.

Then, the flux balance of cell i, j, cf. pp. 10-11, is computed as

$$\mathbf{FB}_{i,j} \; = \mathbf{FI}_{i+1,j} - \mathbf{FI}_{i,j} + \mathbf{FJ}_{i,j+1} - \mathbf{FJ}_{i,j} \tag{61}$$

by looping over $i = 1, ..., I-1$ and $j = 1, ..., J-1$.

The ODE system (34) becomes

$$\frac{d\mathbf{U}_{i,j}}{dt} \; = \; \mathbf{R}(t, \mathbf{U}_{i,j}, \mathbf{U}_{i\pm1,j}, \mathbf{U}_{i\pm2,j}, \mathbf{U}_{i,j\pm1}, \mathbf{U}_{i,j\pm2}) \; , \tag{62}$$

with the residual $\mathbf{R}(t, \mathbf{U}_{i,j}, \mathbf{U}_{i\pm1,j}, \mathbf{U}_{i\pm2,j}, \mathbf{U}_{i,j\pm1}, \mathbf{U}_{i,j\pm2}) \; = -\frac{1}{V_{i,j}}\mathbf{FB}_{i,j}$.

4.2. Unstructured Finite Volume Method

The cell-centered finite volume method outlined for quadrilateral cells and structured grids in the previous sections can be applied for polygonal cells and unstructured grids, because the number of faces of a cell can be arbitrary and the numerical flux approximation only depends on adjacent cell averages.

However, vertex (i.e. node-centered) finite volume methods have been preferred for triangular unstructured grids, because for a $I \times J$ structured grid the unknowns at only IJ vertices (i.e. nodes) need to be stored, while the cell averages in $2(I-1)(J-1)$ triangles need to be stored for a cell-centered finite volume method. In 3D, the vertex finite volume methods need about 5 to 7 times less memory than the cell-centered finite volume method, because there are about $5-7$ more tetrahedra than vertices in 3D tetrahedral grids.

For a grid, the corresponding dual grid can be defined, cf. figure 1.

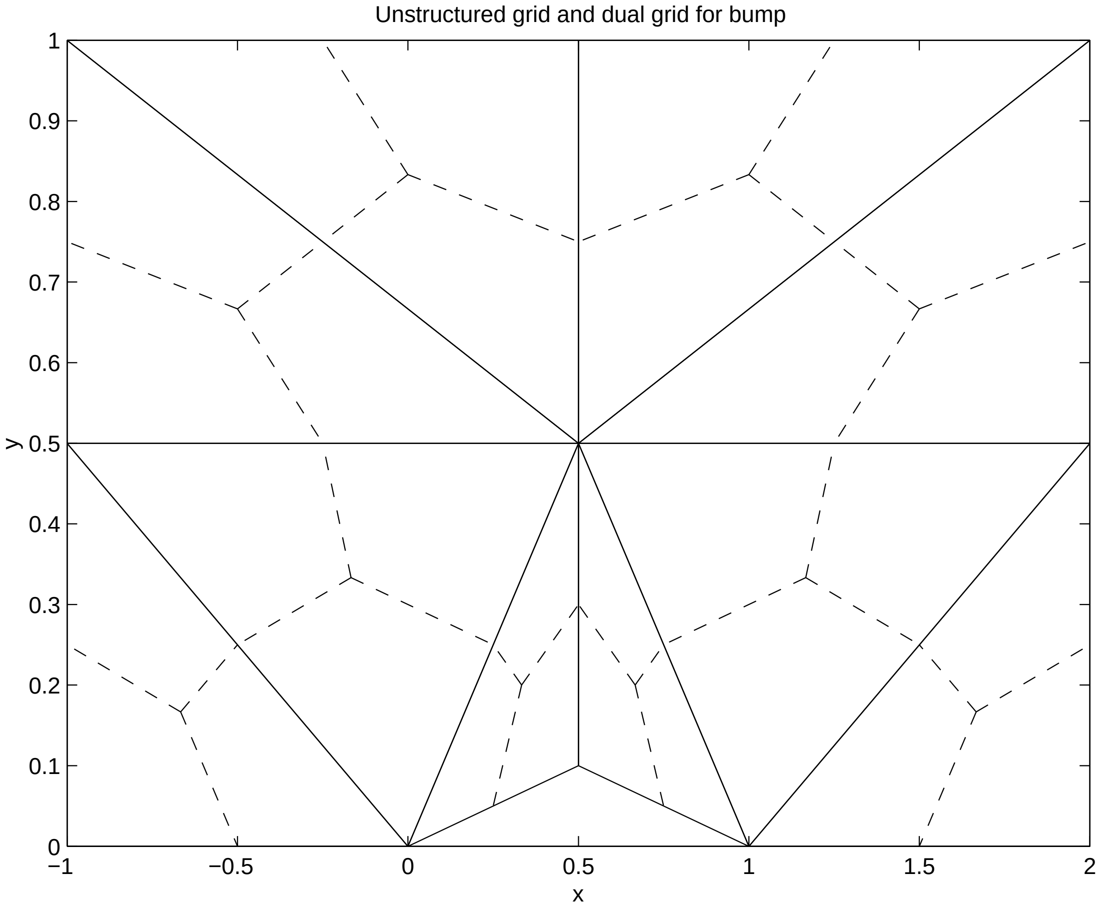


Figure 1: Unstructured grid (solid lines) and dual grid ($- - -$).

The median dual is often chosen, because it is defined elementwise by the centroids of adjacent cells and the midpoints of adjacent edges, cf. figure 2.

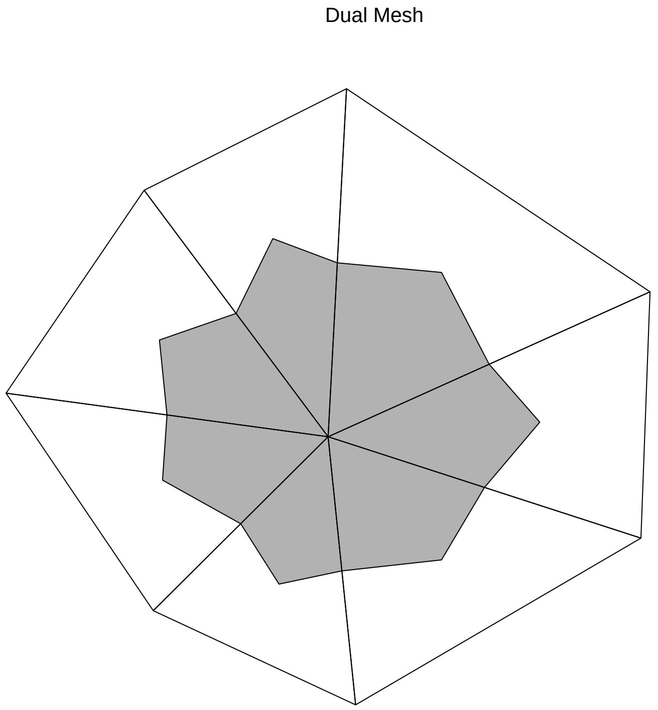


Figure 2: Median dual.

For the edge ij from vertex i to vertex j of the grid, the corresponding face ij of the dual grid consists of the union of a left and right line (union of several surfaces in 3D) with the face normal $\mathbf{S}_{ij}$, cf. figure 3.

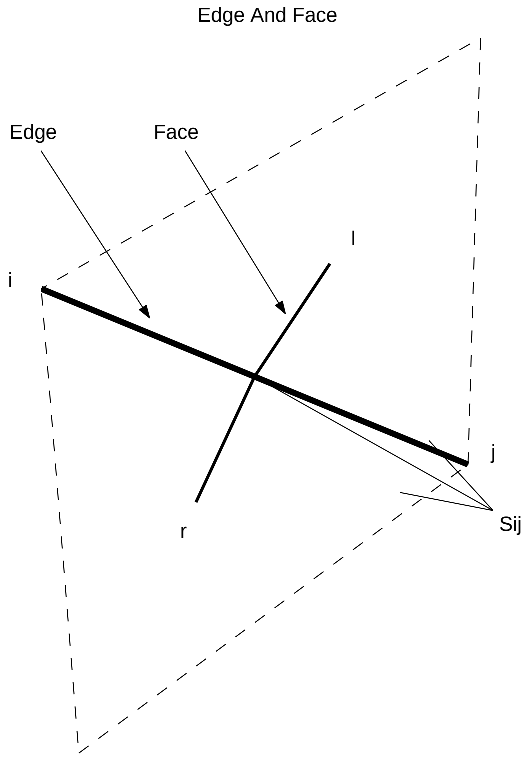


Figure 3: Edge_{ij}, face_{ij} and face normal $\mathbf{S}_{ij}$.

The computational work of a finite volume discretization is proportional to the number of faces. Since a face of the dual grid corresponds to an edge of the grid and the number of edges is equal to the number of faces (except for the boundaries) for a 2D grid, the computational work is about the same for vertex and cell-centered finite volume methods in 2D. In 3D grids, however, there are about twice as many faces than edges, cf. T.J. Barth, in AGARD-R-787, May 1992, pp. 6-1 - 6-61. Therefore, the computational work for the cell-centered finite volume method is about twice as much as for the vertex finite volume method. On the other hand, the cell-centered finite volume method is expected to be more accurate, because its grid cells (tetrahedra) are smaller than the dual grid cells (polyhedra).

4.2.1. Edge-Based Data Structure

Here, we consider the dual-based vertex finite volume method in 2D. We define an edge-based data structure, cf. T.J. Barth, in

AGARD-R-787, May 1992, pp. 6-1 - 6-61. The integer array `ND(NEDGE,4)` for `NEDGE` edges contains, cf. D.J. Mavripilis and A. Jameson, AIAA J., Vol. 28, No. 8, Aug. 1990, pp. 1415-1425, and figure 3

```
i = ND(n,1) % number of origin vertex
j = ND(n,2) % number of destination vertex
l = ND(n,3) % number of left cell
r = ND(n,4) % number of right cell
```

where `n = 1,...,NEDGE` is the edge number.

The element connectivity `NEL(NELEMENT,3)` for `NELEMENT` elements (i.e. cells) is employed as for standard finite element methods, cf. C. Johnson's book (1987), pp.45-46, and figure 4

```
i = NEL(m,1) % number of first vertex
j = NEL(m,2) % number of second vertex
l = NEL(m,3) % number of third vertex
```

where `m = 1,...,NELEMENT` is the element number. Once the first vertex is chosen, the second and third vertices of the element are determined in counter-clockwise direction.

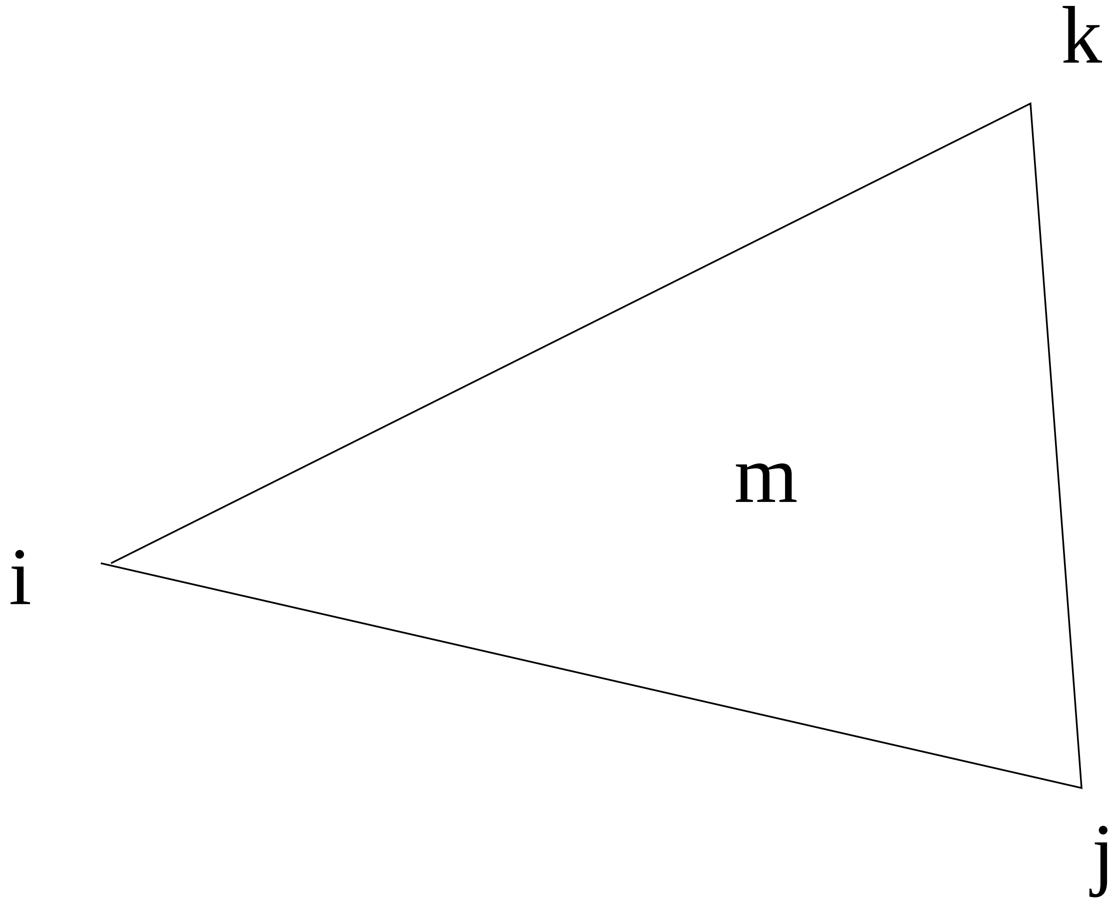


Figure 4: Element m and its vertices i, j, k.

4.2.2. Metric Terms

The face normals **S** of the dual grid associated to the edges of the grid (figure 3) and the areas V of the polygonal cells (figure 2) can be computed by looping over all edges. We assume that the boundary edges are oriented such that the interior is to the left and that the number to the right, i.e. the exterior, is flagged with 0. The control volume associated with a boundary point is shown in figure 5.

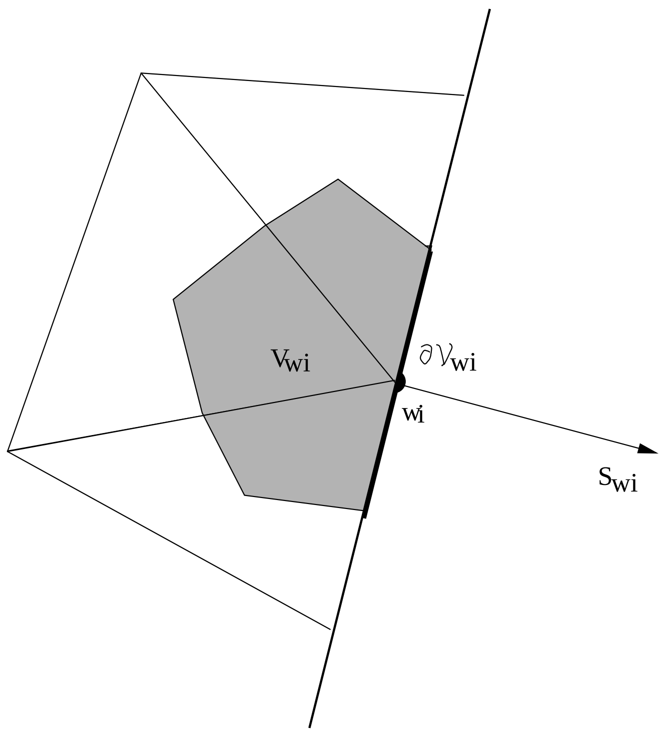


Figure 5: Control volume at boundary point w_i.

The areas V of the median dual cells associated with the vertices are first set to zero. Then, we loop over all edges, compute the left parts of the face normals and for interior edges also the right parts. The area contributions associated with the edge ij, i.e. two triangles defined by vertex i - midpoint of edge ij - centroid l and centroid r, respectively, are assembled. The coordinates of grid point np are given by $GP(np, 1:2)$.

```
V = 0
for n=1,NEDGE
i = ND(n,1)
j = ND(n,2)
nl = ND(n,3)
nr = ND(n,4)
%Centroid, 3 points of left triangle.
p1 = GP(NEL(nl,1),:)  % coordinates of first vertex
p2 = GP(NEL(nl,2),:)  % coordinates of second vertex
p3 = GP(NEL(nl,3),:)  % coordinates of third vertex
```

```
cl = [(p1(1)+p2(1)+p3(1))/3, (p1(2)+p2(2)+p3(2))/3]
% coordinates of centroid
%Midpoint m on edge i j.
pi = GP(i,:)  % coordinates of vertex origin i
pj = GP(j,:)  % coordinates of vertex destination j
pm = 0.5*(GP(i,:)+(GP(j,:))  % coordinates of midpoint m
%Face normal, left part.
Sl = [cl(2)-pm(2), -(cl(1)-pm(1))]
S(n) = Sl
%Area of triangle i - m - l assembled.
Vl=0.5*|(pm(1)-pi(1))*(cl(2)-pm(2))-(pm(2)-pi(2))*(cl(1)-pm(1))|
V(i) = V(i) + Vl
V(j) = V(j) + Vl
```

```
IF (nr.NE.0) THEN
% similar for right triangle
...
Sr=...
S(n) = S(n) + Sr
Vr = ...
V(i) = V(i) + Vr
V(j) = V(j) + Vr
END IF
END
```

4.2.3. Numerical Fluxes

For a first-order method, the inviscid flux balance is computed by looping over all edges, because the edges of the grid correspond to the faces of the dual grid. The program of p. 11 becomes:

```
DO n=1,NEDGE
i = ND(n,1)
j = ND(n,2)
$S_{ij}$ = S(n)
$FLUX_{ij}$ = flux ( U (i), U (j), $S_{ij}$ )
FB (i) = FB (i) + $FLUX_{ij}$
FB (j) = FB (j) - $FLUX_{ij}$
END DO
```

The boundary conditions are considered in the boundary fluxes, which are added to the flux balances at the boundary vertices. As numerical flux function, we can for example choose Roe's approximate Riemann solver (27). A simpler but less accurate flux approximation is based on the Lax-Friedrichs method. For unstructured grids, the Lax-Friedrichs numerical flux can be computed as, cf. D. Kröner's book (1997), p.182:

$$(\mathbf{F}^c \cdot \mathbf{S})_{ij} \;=\; \frac{1}{2}[(\mathbf{F}^c(\mathbf{U}_L) + \mathbf{F}^c(\mathbf{U}_R)) \cdot \mathbf{S}_{ij} \;-\; \frac{\rho(\hat{\mathbf{A}}_{ij})}{CFL}(\mathbf{U}_R - \mathbf{U}_L)]\,, \quad (63)$$

where $\rho(\hat{\mathbf{A}}_{ij}) = |\hat{\mathbf{u}}_{ij} \cdot \mathbf{S}_{ij}| + \hat{c}|\mathbf{S}_{ij}|$ denotes the spectral radius of $\hat{\mathbf{A}}_{ij}$ and $CFL = \frac{\Delta t_i^c}{2V_i} \sum_{j \in N(i)} \rho(\hat{\mathbf{A}}_{ij})$ is the CFL number, cf. p. 48. $N(i)$ is the set of vertices, which have a common edge with vertex i. Instead of the Roe-average, the arithmetic averages may be taken.

Second-order accuracy can be obtained by a linear reconstruction, cf. M. Nilsing's Master's thesis, Uppsala University, Jan. 2002. With van Leer's MUSCL approach

$$\begin{aligned} U_{ij}^- &= U_i + \mathbf{R}_{\frac{1}{2}ij} \cdot \nabla U_i \\ U_{ij}^+ &= U_j - \mathbf{R}_{\frac{1}{2}ij} \cdot \nabla U_j \end{aligned} \qquad (64)$$

where $\mathbf{R}_{\frac{1}{2}ij}$ is the position vector on edge$_{ij}$ from vertex i to the midpoint of edge$_{ij}$ and $-\mathbf{R}_{\frac{1}{2}ij}$ is the corresponding position vector from vertex j. U_{ij}^{-} is the linear extrapolation of U from vertex i and U_{ij}^{+} is the linear extrapolation of U from vertex j. See fig. 3 and 6.

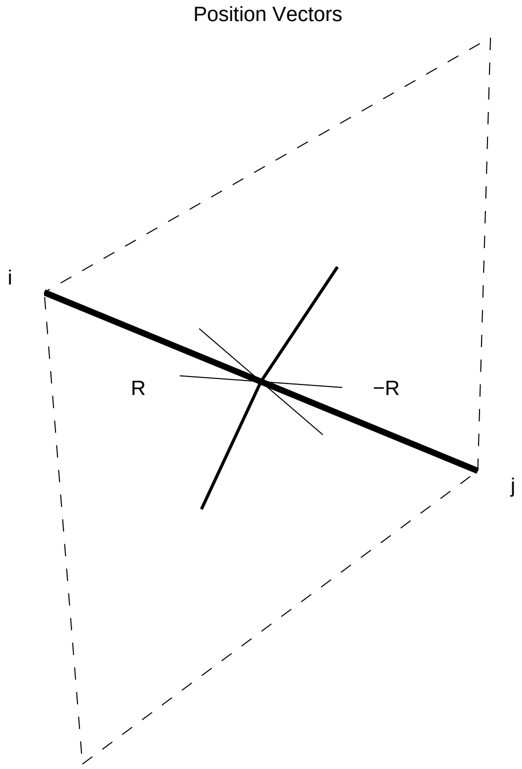


Figure 6: Position vector $\mathbf{R} = \mathbf{R}_{\frac{1}{2}ij}$ and $-\mathbf{R} = -\mathbf{R}_{\frac{1}{2}ij}$.

The x- and y-derivatives are approximated with the Green-Gauss theorem:

$$\nabla u \approx \frac{1}{V}\int_{\Omega} \nabla u dV = \frac{1}{V}\int_{\partial\Omega} u\mathbf{n} dA \tag{65}$$

and the trapezoidal rule:

$$\nabla U_i \approx \frac{1}{V_i}\sum_{j\in N(i)} \frac{1}{2}(U_i + U_j)\mathbf{S}_{ij} \ . \tag{66}$$

To suppress oscillations at discontinuities, the approximated values must not exceed maximum or minimum values of adjacent cells. Therefore, the slopes are limited by slope limiters. Those applied to (64) yield:

$$\begin{aligned} U_{ij}^{-} &= U_i + \Phi_i \mathbf{R}_{\frac{1}{2}ij} \cdot \nabla U_i \\ U_{ij}^{+} &= U_j - \Phi_j \mathbf{R}_{\frac{1}{2}ij} \cdot \nabla U_j, \end{aligned} \tag{67}$$

where the limiter Φ is defined as Barth-Jesperson limiter, cf. T.J.

Barth and D.C. Jesperson, AIAA-89-0366, 1989:

$$\Phi_{ij} = \begin{cases} min\left(1, \frac{u_i^{max} - u_i}{\bar{u}_{ij} - u_i}\right), & \text{if } \bar{u}_{ij} - u_i > 0 \\ min\left(1, \frac{u_i^{min} - u_i}{\bar{u}_{ij} - u_i}\right), & \text{if } \bar{u}_{ij} - u_i < 0 \\ 1, & \text{if } \bar{u}_{ij} - u_i = 0 \end{cases} \qquad \begin{array}{r} u_i^{max} = max(u_i, u_j), \quad j \in N(i) \\ u_i^{min} = min(u_i, u_j), \quad j \in N(i) \end{array} \tag{68}$$

where $\Phi_i = min(\Phi_{ij})$, $j \in N(i)$, and $\bar{u}_{ij} = u_{ij}^-$ from the linear reconstruction (64). The Venkatakrishnan limiter, cf. V. Venkatakrishnan, JCP 118 (1995), pp. 120-130, is defined as:

$$\Phi_{ij} = \begin{cases} \frac{\Delta_{max}^2 + \epsilon_i^2 + 2\Delta_- \Delta_{max}}{\Delta_{max}^2 + 2\Delta_-^2 + \Delta_- \Delta_{max} + \epsilon_i^2}, & \text{if } \Delta_- > 0 \\ \frac{\Delta_{min}^2 + \epsilon_i^2 + 2\Delta_- \Delta_{min}}{\Delta_{min}^2 + 2\Delta_-^2 + \Delta_- \Delta_{min} + \epsilon_i^2}, & \text{if } \Delta_- < 0 \\ 1 & \text{if } \Delta_- = 0 \end{cases} \qquad \begin{array}{rcl} \Delta_- &=& \bar{u}_{ij} - u_i \\ \Delta_{max} &=& u_i^{max} - u_i \\ \Delta_{min} &=& u_i^{min} - u_i \\ \epsilon_i^2 &=& (k\bar{\Delta}_i)^3 \end{array} \tag{69}$$

where $\bar{\Delta}_i$ is the local grid spacing. The control volume is approximated as a circle and then $\bar{\Delta}_i$ is defined as its diameter. k is a constant and signifies a threshold: oscillations below this value are allowed to exist in the solution whereas large values imply no limiting at all. Thus, as k is increased from 0, the convergence will improve until an unstable solution occurs. Here $k = 5$. $\bar{u}_{ij}$, u_i^{min},u_i^{max} and Φ_i are defined as in the Barth-Jesperson limiter.

The solution variables are reconstructed componentwise and the primitive variables are used in the reconstruction step. The conservative variables can reduce the accuracy at contact discontinuities and therefore the use of the primitive variables is recommended.

The gradients for the viscous flux approximation can be computed by averaging the gradients computed at the vertices with

$$\boldsymbol{\nabla} U_{ij} = \frac{1}{2}(\boldsymbol{\nabla} U_i + \boldsymbol{\nabla} U_j)\ . \tag{70}$$

However, this approximation leads to a wide stencil with poor accuracy (double spacings) and stability (odd-even decoupling) characteristics. Therefore, more compact viscous flux approximations are preferred, cf. A. Haselbacher, J.J. McGuirk, G.J. Page, AIAA J., Vol. 37, No. 2, Febr. 1999, pp. 177-184, and A. Haselbacher, J. Blazek, AIAA J., Vol. 38, No. 10, Oct. 2000, pp. 2094-2102. We can get a compact approximation by applying the Green-Gauss theorem in each element, i.e. each cell of the grid and not the dual grid, cf. figure 7, e.g. the gradient of temperature T:

$$\boldsymbol{\nabla} T_a \approx \frac{1}{V_a}\int_{\Omega_a} \boldsymbol{\nabla} T dV = \frac{1}{V_a}\int_{\partial\Omega_a} T\mathbf{n} dA\ . \tag{71}$$

Using the trapezoidal rule, the cell-averaged gradient becomes:

$$\boldsymbol{\nabla} T_a \approx \frac{1}{V_a} \sum_{lm=ij}^{ki} \frac{1}{2}(T_l + T_m)\mathbf{S}_{lm}, \tag{72}$$

where the sum is over the faces ij, jk, ki of cell a. The viscosity is averaged as $\mu_a \approx \frac{1}{3}\sum_{m=i}^{k} \mu_m$, where the sum is over the vertices i, j, k of cell a. The thermal conductivity k is approximated similarly.

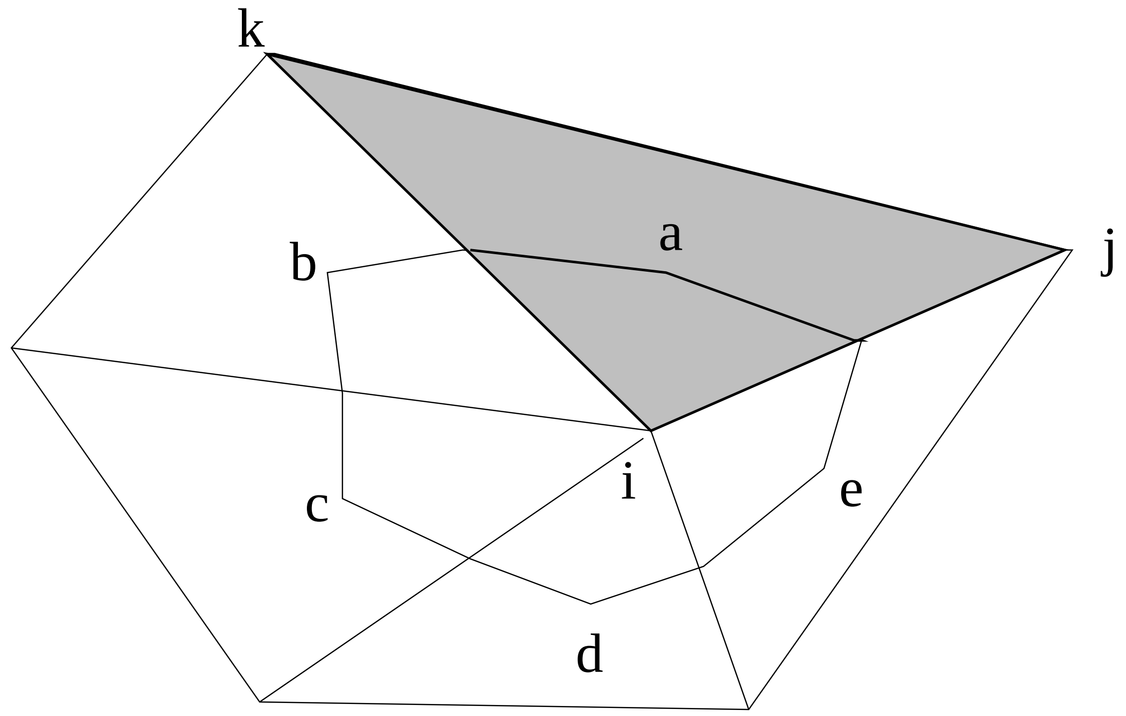


Figure 7: Median dual i and elements a to e.

The viscous flux function at face ij of the dual grid, associated with edge ij of the grid, is approximated by e.g.

$$k_{ij}\boldsymbol{\nabla} T_{ij} \approx \frac{1}{2}(k_l \boldsymbol{\nabla} T_l + k_r \boldsymbol{\nabla} T_r) ,$$

where l and r denote the cells to the left and right of the edge ij, respectively, cf. section 4.2.1. In figure 7, $l = a$ and $r = e$ for edge ij. Then, the viscous flux, here the heat flux over the median dual Ω_i, can be approximated by

$$\int_{\partial\Omega_i} k\boldsymbol{\nabla} T \, dA \approx \sum_{l\in E(i)} (k_l \boldsymbol{\nabla} T_l) \cdot \mathbf{S}_l , \tag{73}$$

where $E(i)$ denotes the edges, which have vertex i in common.

High order reconstructions of structured and unstructured finite volume methods can be obtained by essentially non-oscillatory (ENO) and weighted essentially non-oscillatory (WENO) schemes, cf. C.-W. Shu, pp. 439-582, and R. Abgrall, T. Sonar, O. Friedrich, and G. Billet, pp. 1-67, in T.J. Barth and H. Decononck (Eds.), “High-Order Methods for Computational Physics, Springer-Verlag, Berlin, 1999.