Finite volume method on unstructured grids

Praveen. C
praveen@math.tifrbng.res.in

Tata Institute of Fundamental Research
Center for Applicable Mathematics
Bangalore 560065
http://math.tifrbng.res.in/~praveen

April 1, 2013

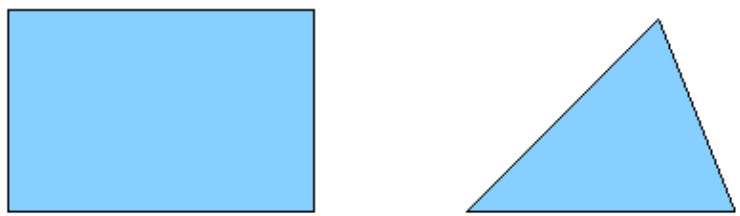
Mesh and Finite volumes

The basic idea of FVM is to divide domain Ω into a set of disjoint **finite volumes** and apply the conservation law on each finite volume. The division of Ω gives rise to the **mesh** or **grid**.

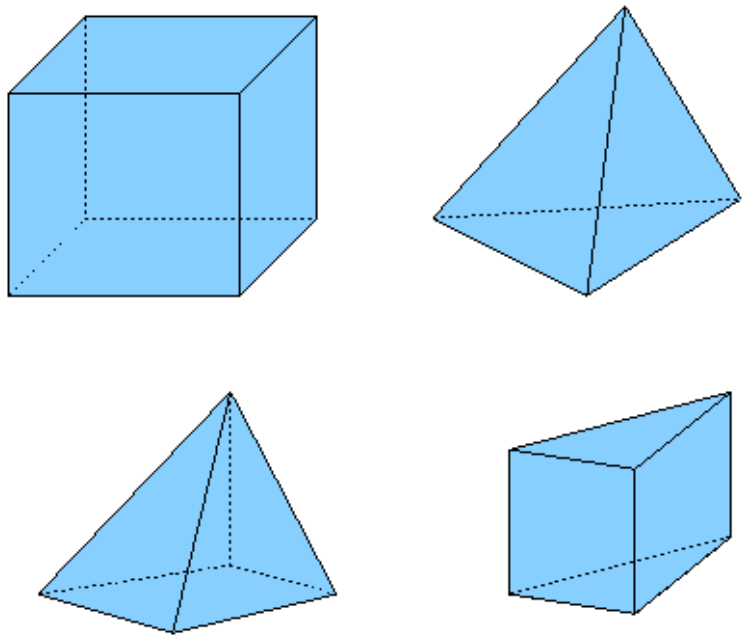
$$\Omega_i, \qquad i = 1, 2, \ldots, N_\Omega$$

$$\Omega = \bigcup_{i=1}^{N_\Omega} \Omega_i$$

For simplicity we take each Ω_i to be a polygonal cell.



Mesh and Finite volumes



The mesh can consist of different types of cells, e.g., triangular and quadrilateral cells in 2-D. Such a mesh is called **hybrid mesh**.

Usually the mesh is taken to be **conforming** in the sense that there are no hanging nodes. But it is possible to use grids with hanging nodes also, which can be advantageous when doing grid adaptation.

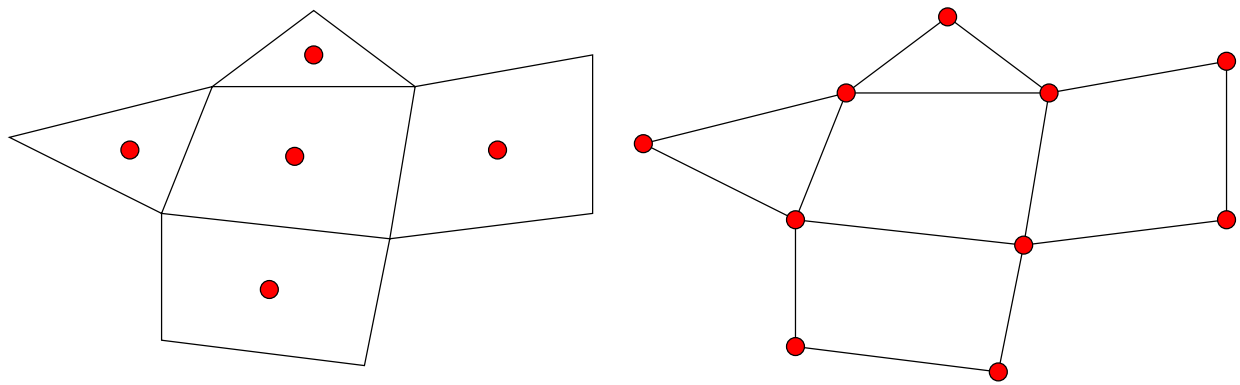
Mesh and Finite volumes

Let us denote the set of vertices in the mesh by

$$V = \{V_i : i = 1, 2, \ldots, N_V\}$$

Finite volumes

Once a mesh has been formed, we have to create the finite volumes on which the conservation law will be applied. This can be done in two ways, depending on where the solution is stored.



1. If the solution is stored at the center of each Ω_i, then Ω_i itself is the finite volume or cell, $C_i = \Omega_i$. This gives rise to the **cell-centered** finite volume scheme.
2. If the solution is stored at the vertices of the mesh, then around each vertex i we have to construct a cell C_i. This gives rise to the **vertex-centered** finite volume scheme.

Finite volumes

In either case we obtain a collection of disjoint finite volumes $\{C_i\}$, $i = 1, 2, \ldots, N_c$ such that

$$\Omega = \bigcup_{i=1}^{N_c} C_i$$

Some geometric information

Interior Face

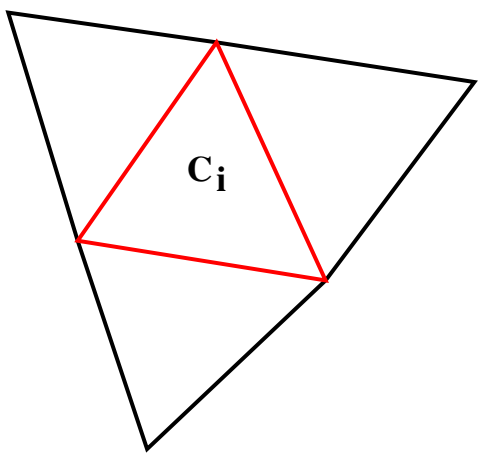
$$S_{ij} = \partial C_i \cap \partial C_j = \text{common face between } C_i \text{ and } C_j$$

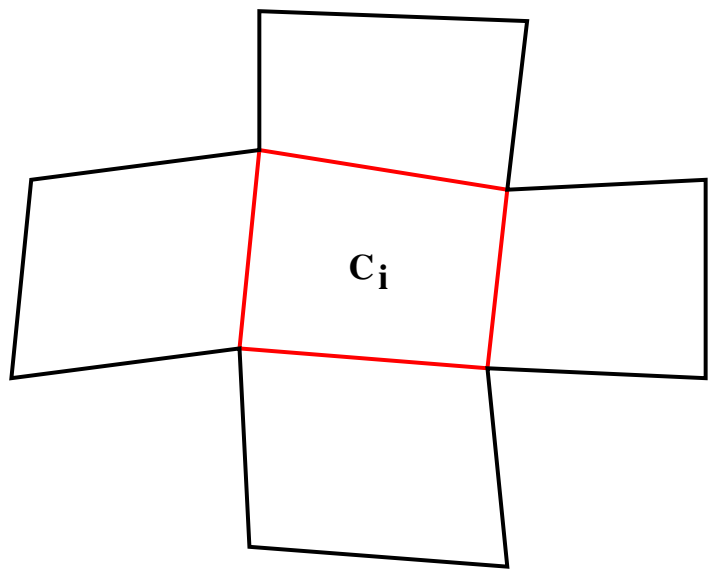
Boundary face

$$S_{ib} = \partial C_i \cap \partial \Omega = \text{face of } C_i \text{ on boundary of } \Omega$$

For each cell C_i

$$N_i = \{C_j : C_i \text{ and } C_j \text{ have a common face } S_{ij}\}$$





Some geometric information

Remark: Instead of saying

$$C_j \in N_i$$

we will simply say that

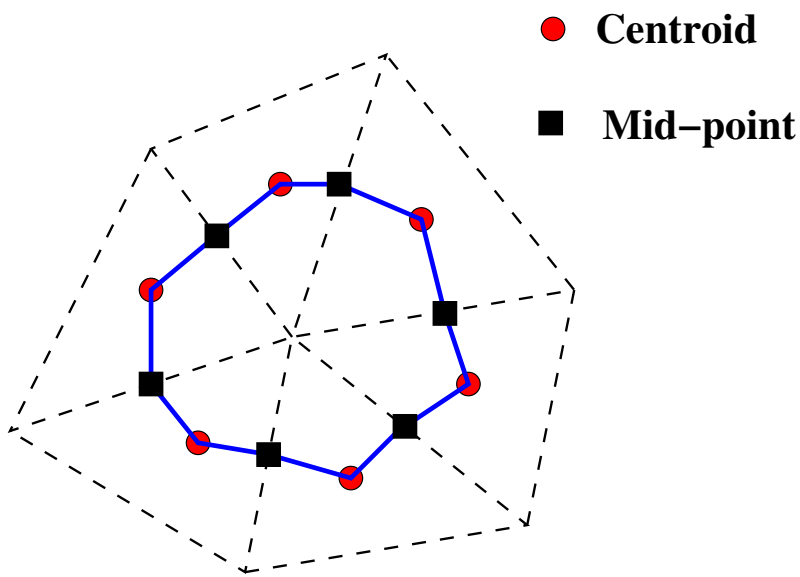
$$j \in N_i$$

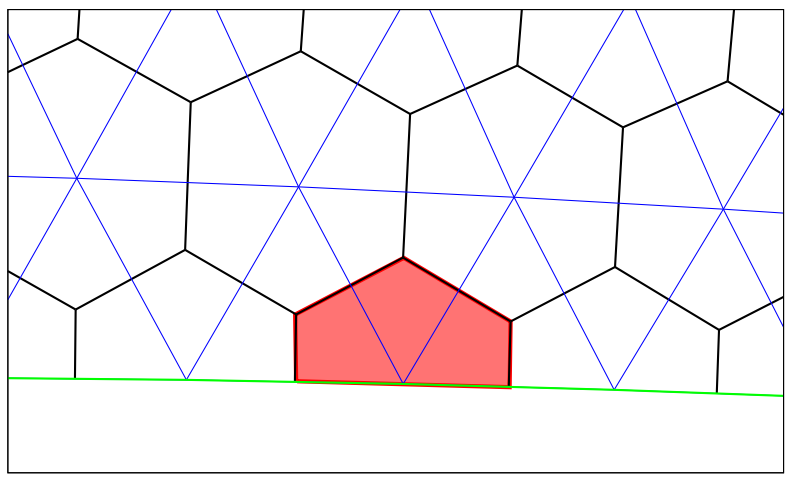
Vertex-centered finite volumes

There are several ways to define the finite volume around a vertex.

- Join the mid-point of the edges and cell centers. This leads to the **median dual** cell.
- (Triangular grids only) Join the circum-center of the triangles to the edge mid-points. For obtuse angled triangles, use the mid-point of largest side. This is called the **containment dual** cell. Useful in boundary layers.

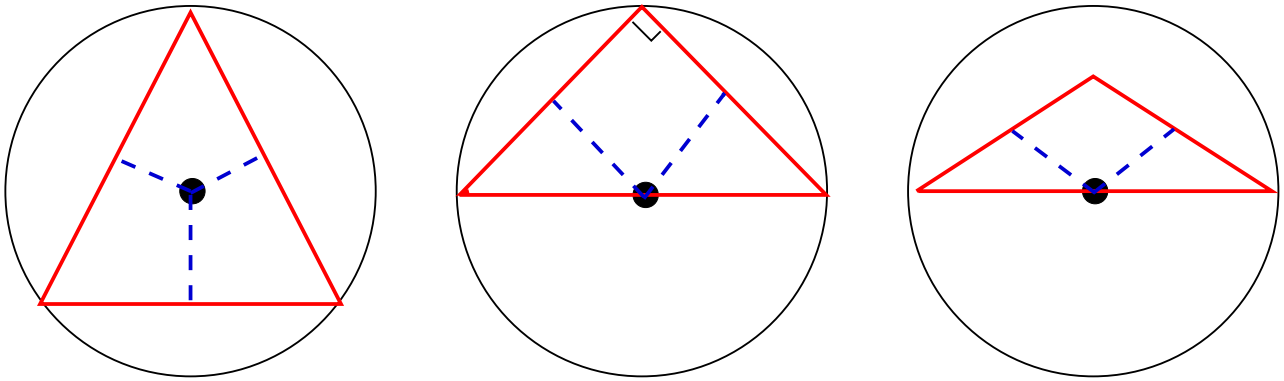
For vertices on the boundary, the finite volume is closed by a portion of the boundary edges.

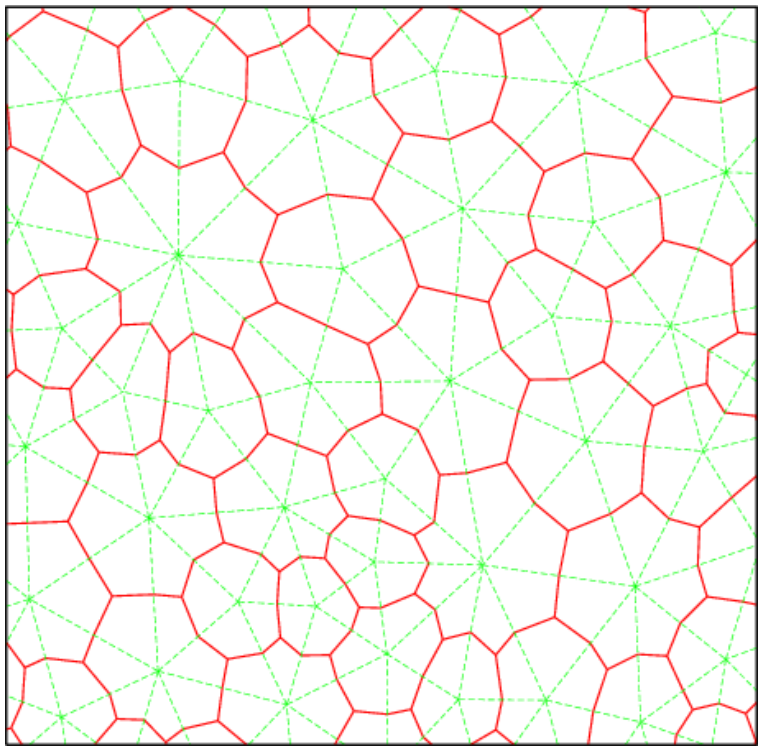




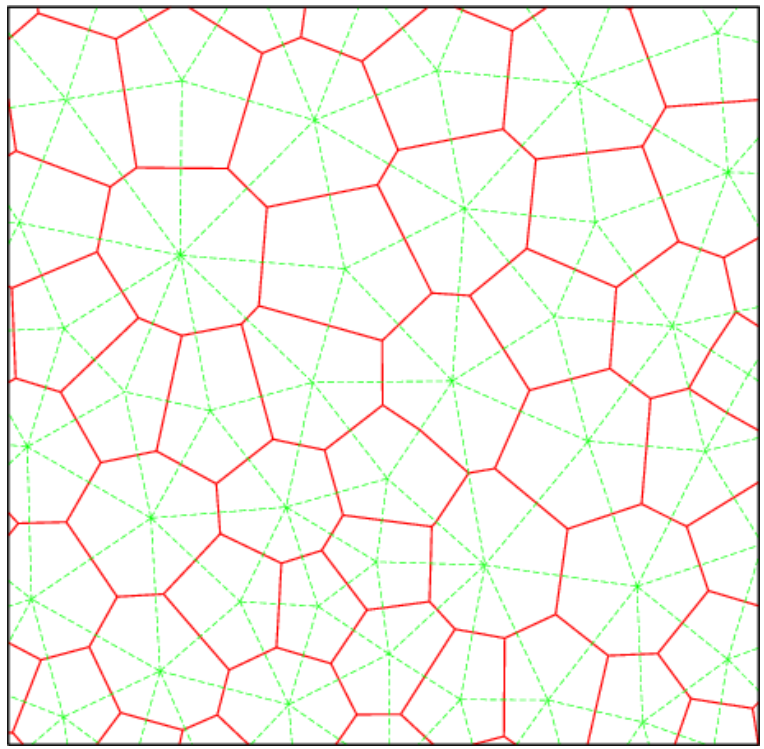
Vertex-centered finite volumes

containment dual



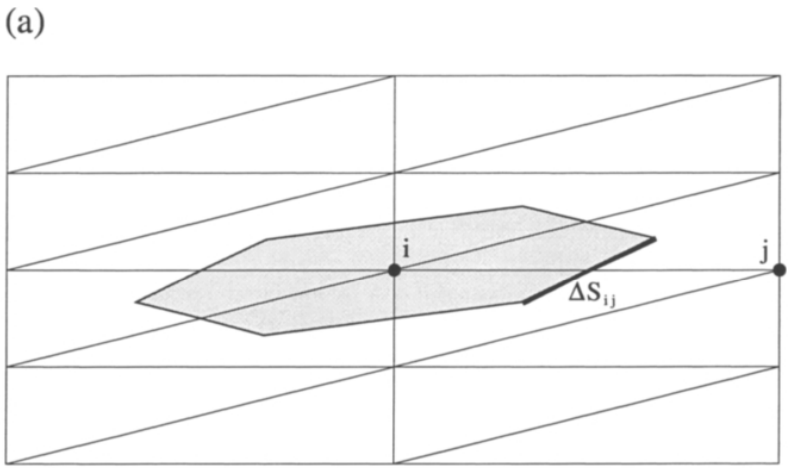


Median-dual cells



containment-dual cells

Vertex-centered finite volumes



Median-dual cells

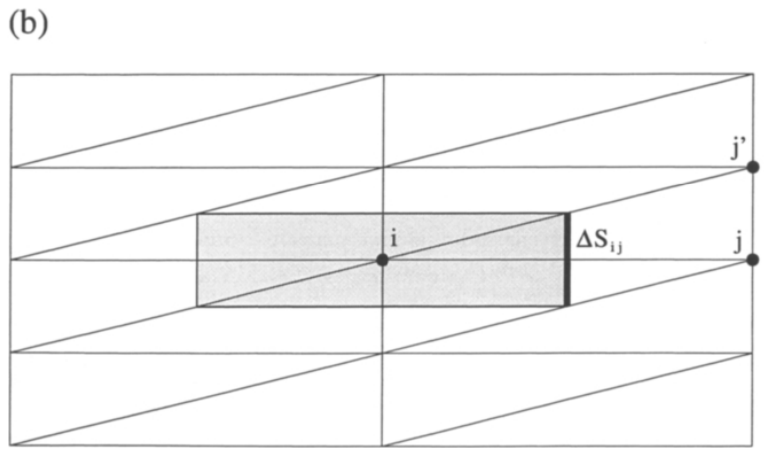


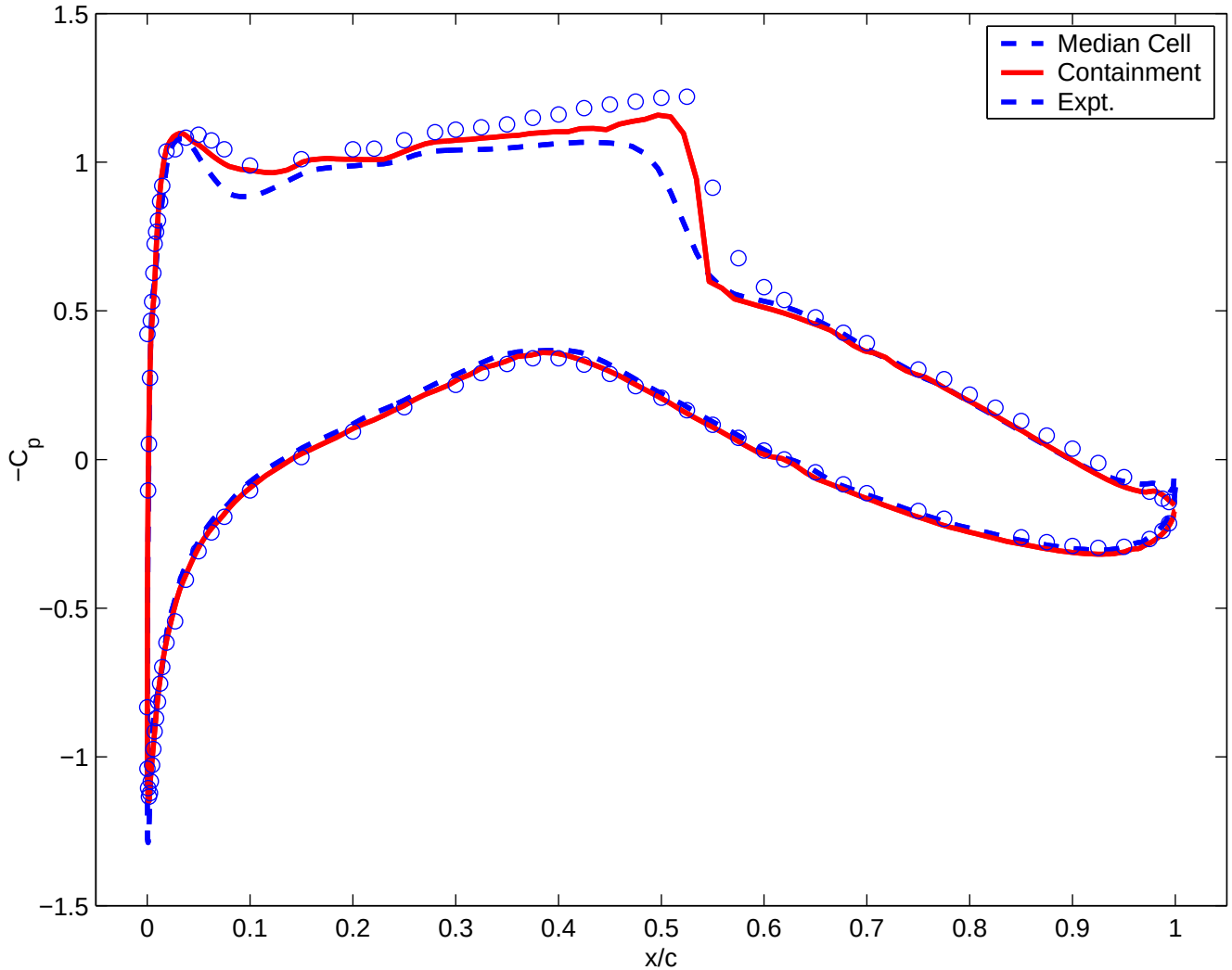
Figure 5.10: Comparison of median-dual (a) and containment-dual (b) control volumes for a stretched right-angle triangulation.

containment-dual cells

Vertex-centered finite volumes

Turbulent flow over RAE2822 airfoil: vertex-centered scheme

Mach = 0.729, α = 2.31 deg, Re = 6.5 million



FVM for Compressible NS equations

$$U_t + F_x + G_y = P_x + Q_y$$

Integrating over cell C_i

$$\int_{C_i} \frac{\partial U}{\partial t} \mathrm{d}V + \int_{C_i} (F_x + G_y) \mathrm{d}V = \int_{C_i} (P_x + Q_y) \mathrm{d}V$$

Define **cell average value**

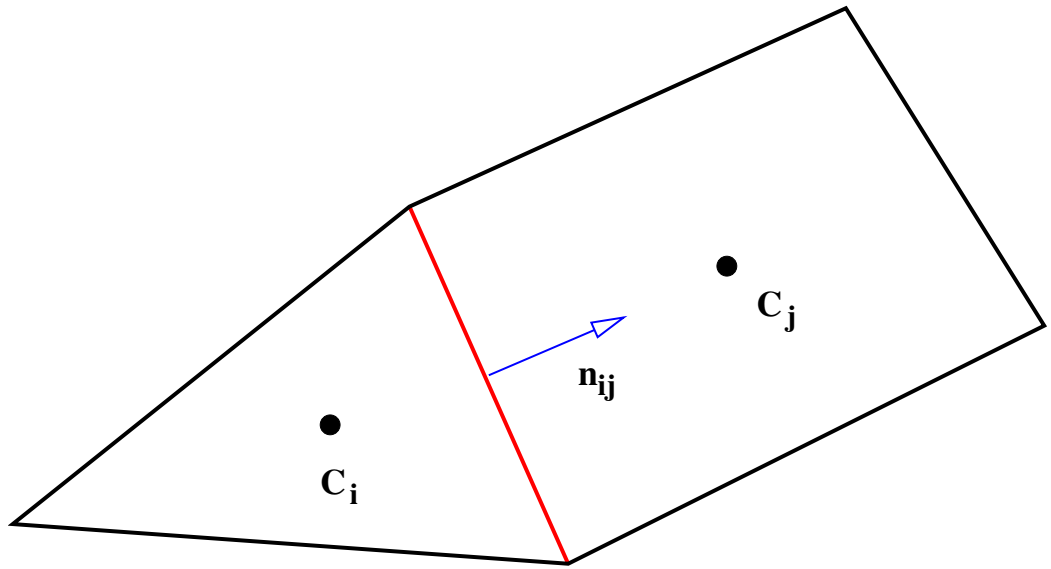
$$U_i(t) = \frac{1}{|C_i|} \int_{C_i} U(x, y, t) \mathrm{d}V$$

Using divergence theorem

$$|C_i| \frac{\mathrm{d}U_i}{\mathrm{d}t} + \int_{\partial C_i} (F n_x + G n_y) \mathrm{d}S = \int_{\partial C_i} (P n_x + Q n_y) \mathrm{d}S$$

FVM for Compressible NS equations

$$|C_i|\frac{\mathrm{d}U_i}{\mathrm{d}t} + \sum_{j\in N_i}\int_{S_{ij}}(Fn_x + Gn_y)\mathrm{d}S + \sum_{S_{ib}\in\partial\Omega}\int_{S_{ib}}(Fn_x + Gn_y)\mathrm{d}S$$
$$= \sum_{j\in N_i}\int_{S_{ij}}(Pn_x + Qn_y)\mathrm{d}S + \sum_{S_{ib}\in\partial\Omega}\int_{S_{ib}}(Pn_x + Qn_y)\mathrm{d}S$$



We have to approximate the flux integral by quadrature. For first order and second order accurate schemes, it is enough to use mid-point rule of

FVM for Compressible NS equations

integration.

$$\int_{S_{ij}} (Fn_x + Gn_y)\mathrm{d}S \approx (Fn_x + Gn_y)_{ij}|S_{ij}|$$

How to compute the flux ? We have two states U_{ij} and U_{ji} coming from cells C_i and C_j. We will use a numerical flux function of Godunov-type or flux vector splitting, etc

$$(Fn_x + Gn_y)_{ij} \approx H(U_{ij}, U_{ji}, n_{ij})$$

On boundary faces S_{ib} the flux is determined using appropriate boundary conditions

$$(Fn_x + Gn_y)_{ib} \approx H_b(U_{ib}, U_b, n_{ib})$$

The viscous fluxes are computed using central difference type approximations which we discuss later

$$(Pn_x + Qn_y)_{ij} \approx R_{ij}, \qquad (Pn_x + Qn_y)_{ib} \approx R_{ib}$$

FVM for Compressible NS equations

Finally we have the semi-discrete scheme

$$|C_i|\frac{\mathrm{d}U_i}{\mathrm{d}t} + \sum_{j\in N_i} H(U_{ij}, U_{ji}, n_{ij})|S_{ij}| + \sum_{S_{ib}\in\partial\Omega} H_b(U_{ib}, U_b, n_{ib})|S_{ib}|$$
$$= \sum_{j\in N_i} R_{ij}|S_{ij}| + \sum_{S_{ib}\in\partial\Omega} R_{ib}|S_{ib}|$$

We now have a system of ODE which we can integrate in time using various schemes like Runge-Kutta or implicit schemes.

First order finite volume

In this case, we assume that the solution inside each cell is a constant in space. Then on any interior face S_{ij} we have the two states

$$U_{ij} = U_i, \qquad U_{ji} = U_j$$

The convective flux is then approximated as

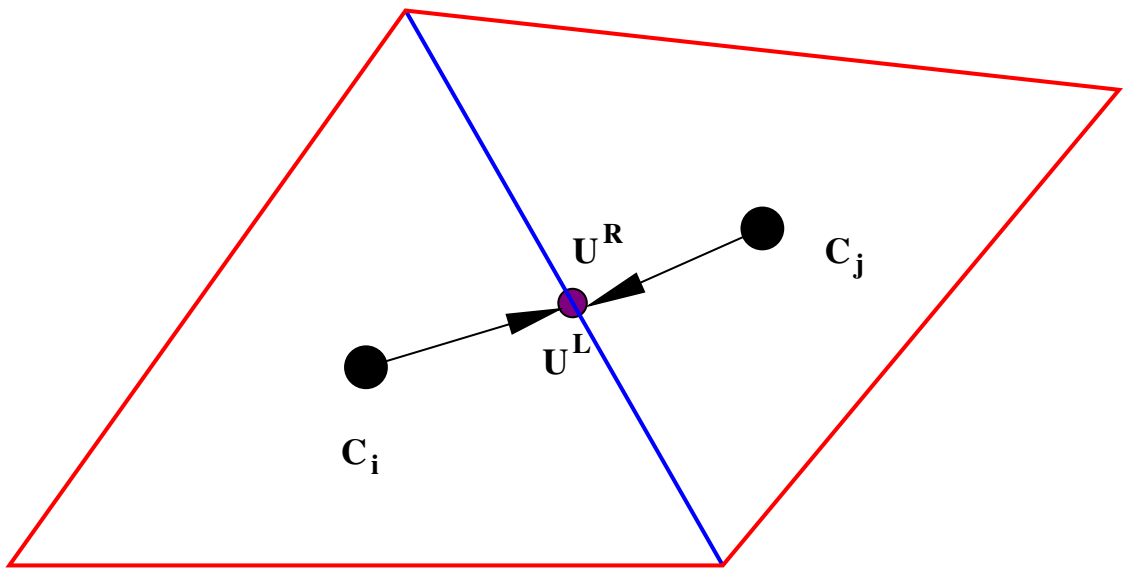
$$H(U_i, U_j, n_{ij})$$

which leads to a first order accurate scheme. If the numerical flux H is designed well, then these schemes are very stable, robust and have good properties like monotonicity and entropy condition. But they introduce too much error and lead to poor resolution of shocks, contact waves and vorticity.

Reconstruction

To achieve more than first order accuracy, we can reconstruct the solution inside each cell. The simplest approach is to perform piecewise linear reconstruction. The reconstruction can be performed on

- conserved variables
- primitive variables (ρ, u, v, p) or (T, u, v, p)
- characteristic variables



Reconstruction

Conserved variables allow satisfaction of conservation of reconstucted solution very easily.

Primitive variables make it easy to ensure positivity of density and pressure which leads to a more robust scheme.

Characteristics variables leads to more accurate schemes at a slightly more computational cost.

Let us denote by W the set of variables that are going to be reconstructed. Using the reconstruction process, obtain two states W_{ij} and W_{ji} at each face S_{ij} and compute the flux

$$H(W_{ij}, W_{ji}, n_{ij})$$

Gradient-based reconstruction: Cell/vertex-centered case

Assume that we have the gradient of W at the cell centers. Then inside each cell C_i the reconstructed solution is

$$W(x,y) = W_i + (r - r_i) \cdot \nabla W_i, \qquad r = (x,y), \qquad r_i = (x_i, y_i)$$

At the mid-point $r = r_{ij}$ of face S_{ij} we obtain two states

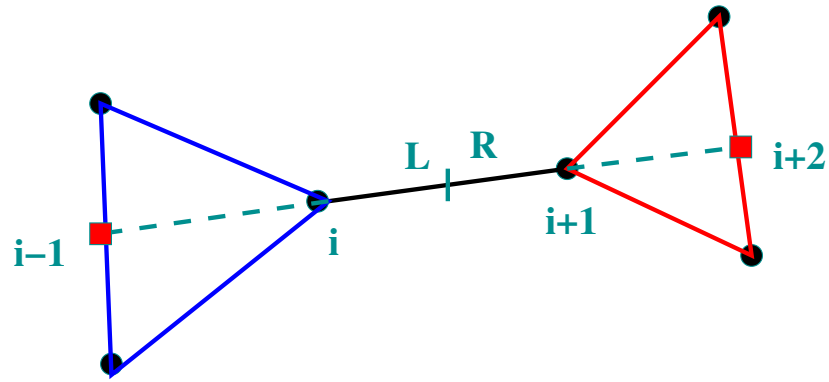
$$W_{ij} = W_i + (r_{ij} - r_i) \cdot \nabla W_i, \qquad W_{ji} = W_j + (r_{ij} - r_j) \cdot \nabla W_j$$

In order to ensure monotone solutions, a limiter function is calculated on each cell and the limited reconstructed values are

$$W_{ij} = W_i + \phi_i (r_{ij} - r_i) \cdot \nabla W_i, \qquad W_{ji} = W_j + \phi_j (r_{ij} - r_j) \cdot \nabla W_j$$

Since W has several components, the limiter function is computed for each component. Popular limiters are min-max limiter of Barth-Jespersen and Venkatakrishnan limiter which are explained later.

MUSCL-type Reconstruction: vertex-centered case



$$U^L = U_i + \frac{1}{2}\text{Limiter}\left[(U_{i+1} - U_i), \frac{|P_iP_{i+1}|}{|P_iP_{i-1}|}(U_i - U_{i-1})\right]$$

or, using vertex-gradients

$$U^L = U_i + \frac{1}{2}\text{Limiter}\left[(U_{i+1} - U_i), (\vec{P}_{i+1} - \vec{P}_i) \cdot \nabla U_i\right]$$

Van-albada limiter

$$\text{Limiter}(a, b) = \frac{(a^2 + \epsilon)b + (b^2 + \epsilon)a}{a^2 + b^2 + 2\epsilon}, \quad \epsilon \ll 1$$

MUSCL-type Reconstruction: vertex-centered case

(See Lohner, Section 10.4.2) This extends the MUSCL idea to unstructured grids.

$$W_{ij} = W_i + \frac{1}{4}[(1-k)\Delta_i^- + (1+k)(W_j - W_i)]$$

$$W_{ji} = W_j - \frac{1}{4}[(1-k)\Delta_j^+ + (1+k)(W_j - W_i)]$$

where the forward and backward difference operators are given by

$$\Delta_i^- = W_i - W_{i-1} = 2r_{ij} \cdot \nabla W_i - (W_j - W_i)$$

$$\Delta_j^+ = W_{j+1} - W_j = 2r_{ij} \cdot \nabla W_j - (W_j - W_i)$$

where

$$r_{ij} = r_j - r_i$$

In 1-D case, $k = 1/3$ gives third order accurate scheme.

MUSCL-type Reconstruction: vertex-centered case

With limiter

$$W_{ij} = W_i + \frac{s_i}{4}[(1 - ks_i)\Delta_i^- + (1 + ks_i)(W_j - W_i)]$$

$$W_{ji} = W_j - \frac{s_j}{4}[(1 - ks_j)\Delta_j^+ + (1 + ks_j)(W_j - W_i)]$$

$$s_i = L(\Delta_i^-, W_j - W_i), \qquad s_j = L(\Delta_j^+, W_j - W_i)$$

Limiter function, van-Albada type limiter

$$L(a, b) = \max\left[0, \frac{2ab + \epsilon}{a^2 + b^2 + \epsilon}\right]$$

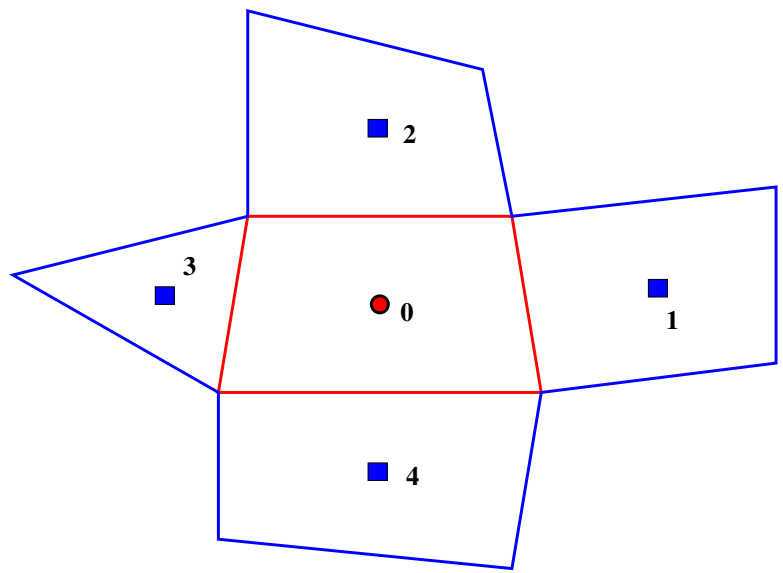
Gradient computation: least squares

Given data

$$W_0, W_1, W_2, \ldots, W_n$$

at positions

$$r_0, r_1, r_2, \ldots, r_n, \qquad r_i = (x_i, y_i)$$



Gradient computation: least squares

Compute $\nabla W_0 = (a, b)$. From Taylor formula

$$\begin{aligned} W_i &= W_0 + (r_i - r_0) \cdot \nabla W_0 + \mathcal{O}\left(h^2\right) \\ &= W_0 + (x_i - x_0)a + (y_i - y_0)b + \mathcal{O}\left(h^2\right) \end{aligned}$$

Define

$$\Delta W_i = W_i - W_0, \qquad \Delta x_i = x_i - x_0, \qquad \Delta y_i = y_i - y_0$$

Neglecting higher order terms, we have an over-determined system of equations

$$\Delta W_i = \Delta x_i a + \Delta y_i b, \qquad i = 1, 2, \ldots, n$$

Let us determined (a, b) by solving the weighted minimization problem

$$\min_{a,b} \sum_{i=1}^{n} \omega_i \left[\Delta W_i - \Delta x_i a - \Delta y_i b\right]^2$$

Gradient computation: least squares

The weight function is usually chosen to be of the form

$$\omega_i = \frac{1}{|r_i - r_0|^p}, \qquad p = 0 \text{ or } p = 2$$

Conditions for extremum are

$$\frac{\partial}{\partial a} \sum_{i=1}^{n} \omega_i \left[\Delta W_i - \Delta x_i a - \Delta y_i b\right]^2 = 2 \sum_{i=1}^{n} \omega_i \left[-\Delta W_i \Delta x_i + (\Delta x_i)^2 a + \Delta x_i \Delta y_i b\right] = 0$$

$$\frac{\partial}{\partial b} \sum_{i=1}^{n} \omega_i \left[\Delta W_i - \Delta x_i a - \Delta y_i b\right]^2 = 2 \sum_{i=1}^{n} \omega_i \left[-\Delta W_i \Delta y_i + \Delta x_i \Delta y_i a + (\Delta y_i)^2 b\right] = 0$$

We obtained two coupled equations

$$(\sum_i \omega_i \Delta x_i^2) a + (\sum_i \omega_i \Delta x_i \Delta y_i) b = \sum_i \omega_i \Delta W_i \Delta x_i$$

$$(\sum_i \omega_i \Delta x_i \Delta y_i) a + (\sum_i \omega_i \Delta y_i^2) b = \sum_i \omega_i \Delta W_i \Delta y_i$$

Gradient computation: least squares

In matrix form

$$\begin{bmatrix} \sum_i \omega_i \Delta x_i^2 & \sum_i \omega_i \Delta x_i \Delta y_i \\ \sum_i \omega_i \Delta x_i \Delta y_i & \sum_i \omega_i \Delta y_i^2 \end{bmatrix} \begin{bmatrix} a \\ b \end{bmatrix} = \begin{bmatrix} \sum_i \omega_i \Delta W_i \Delta x_i \\ \sum_i \omega_i \Delta W_i \Delta y_i \end{bmatrix}$$

Assume that the points do not all lie on the same straight line. Then the matrix on the left has determinant

$$(\sum_i \omega_i \Delta x_i^2)(\sum_i \omega_i \Delta y_i^2) - (\sum_i \omega_i \Delta x_i \Delta y_i)^2 > 0$$

and hence is invertible.

Choice of stencil: We need atleast two neighbouring cells to apply least squares method. We can choose the face neighbouring cells N_i which is usually sufficient to apply least squares. If necessary one can add extra cells by using the neighbours of neighbours.

Gradient computation: least squares

Remark: The LS derivative formula is exact for linear polynomials. The derivatives obtained by this approach are in general first order accurate, i.e.,

$$W_x - W_x^{\text{exact}} = \mathcal{O}(h)$$

On smooth and symmetric stencils, we can obtain close to second order accuracy.

Remark: The least squares method gives accurate gradient estimates. But on highly stretched grids, it can lead to unstable schemes. The use of distance based weight alleviates the problem to some extent.

Remark: This idea can be easily extended to three dimensions. We can also use quadratic reconstruction where we retain terms involving second derivatives in the Taylor expansion. This gives derivatives which are second order accurate (exact for quadratic polynomials) and we also obtain second derivatives which are first order accurate.

Gradient computation: Green-Gauss

Green theorem applied to cell C_i

$$\int_{C_i} \nabla W \mathrm{d}V = \int_{\partial C_i} W n \mathrm{d}S$$

$$\nabla W_i = \frac{1}{|C_i|} \sum_{j \in N_i} \frac{1}{2}(W_i + W_j) n_{ij} |S_{ij}|$$

This formula has to be modified for boundary cells.

min-max limiter

The basic idea is that the reconstructed states W_{ij} must remain between the minimum and maximum values in the stencil of C_i. Define

$$W_i^m = \min_{j \in N_i}(W_j, W_i), \qquad W_i^M = \max_{j \in N_i}(W_j, W_i)$$

Then we want to choose the largest value of $0 \leq \phi_i \leq 1$ so that

$$W_i^m \leq W_i + \phi_i(r_{ij} - r_i) \cdot \nabla W_i \leq W_i^M, \qquad \forall C_j \in N_i$$

Define

$$\Delta_{ij} = (r_{ij} - r_i) \cdot \nabla W_i$$

$$\phi_{ij} = \begin{cases} \min\left(1, \frac{W_i^M - W_i}{\Delta_{ij}}\right) & \text{if } \Delta_{ij} > 0 \\ \min\left(1, \frac{W_i^m - W_i}{\Delta_{ij}}\right) & \text{if } \Delta_{ij} < 0 \\ 1 & \text{otherwise} \end{cases}$$

min-max limiter

Then

$$\phi_i = \min_{j \in N_i} \phi_{ij}$$

For scalar conservation laws, one can show that the FV scheme with a monotone flux satisfies local maximum principle and hence is stable in maximum norm.

For Euler equations, this leads to a very robust scheme but it is not very accurate since it can clip smooth extrema also.

Moreover, the limiter is not a smooth function due to use of min and max functions. This leads to slow convergence to steady state solutions and in fact we do not obtain convergence to machine zero in most cases.

Venkatakrishnan limiter

This is a smooth modification of the minmax limiter which has good convergence properties for steady state problems.

$$\phi_{ij} = \begin{cases} L(W_i^M - W_i, \Delta_{ij}) & \text{if } \Delta_{ij} > 0 \\ L(W_i^m - W_i, \Delta_{ij}) & \text{if } \Delta_{ij} < 0 \\ 1 & \text{otherwise} \end{cases}$$

$$L(a,b) = \frac{a^2 + 2ab + \omega}{a^2 + 2b^2 + ab + \omega}$$

Then

$$\phi_i = \min_{j \in N_i} \phi_{ij}$$

We have essentially replaced the min function in min-max limiter with the above smooth function. There is a paramater ω which is chosen as

$$\omega = (Kh)^3$$

Venkatakrishnan limiter

Here h represents the cell size. One can take

$$h = (\text{area of cell})^{1/2}, \qquad h = (\text{volume of cell})^{1/3}$$

or take h to be average cell size in the grid. The value of K affects the amount of limiting, with $K = 0$ giving highest amount of limiting.

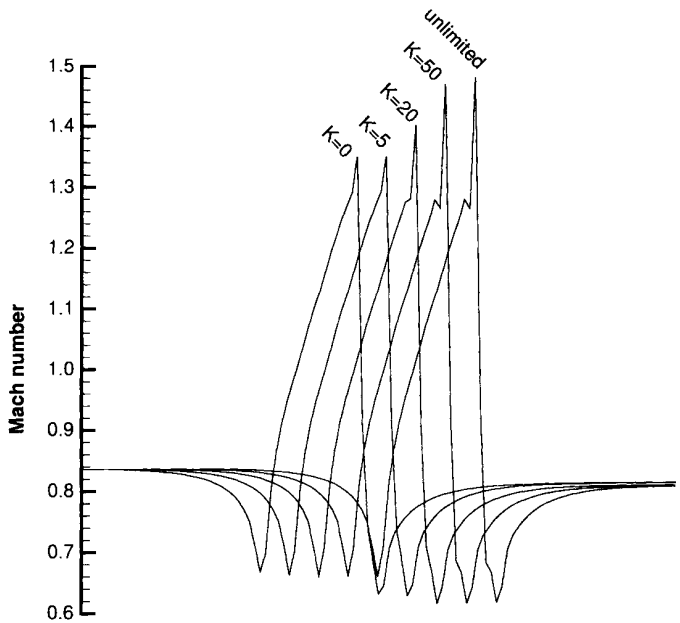


Figure 5.16: Effect of the constant K in Venkatakrishnan's limiter, given by Eq. (5.67), on the solution for an inviscid flow past a circular arc.

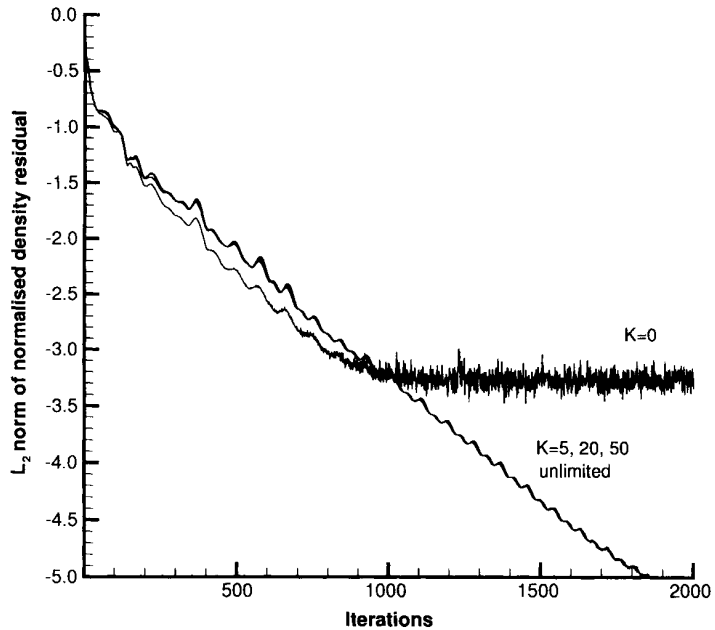
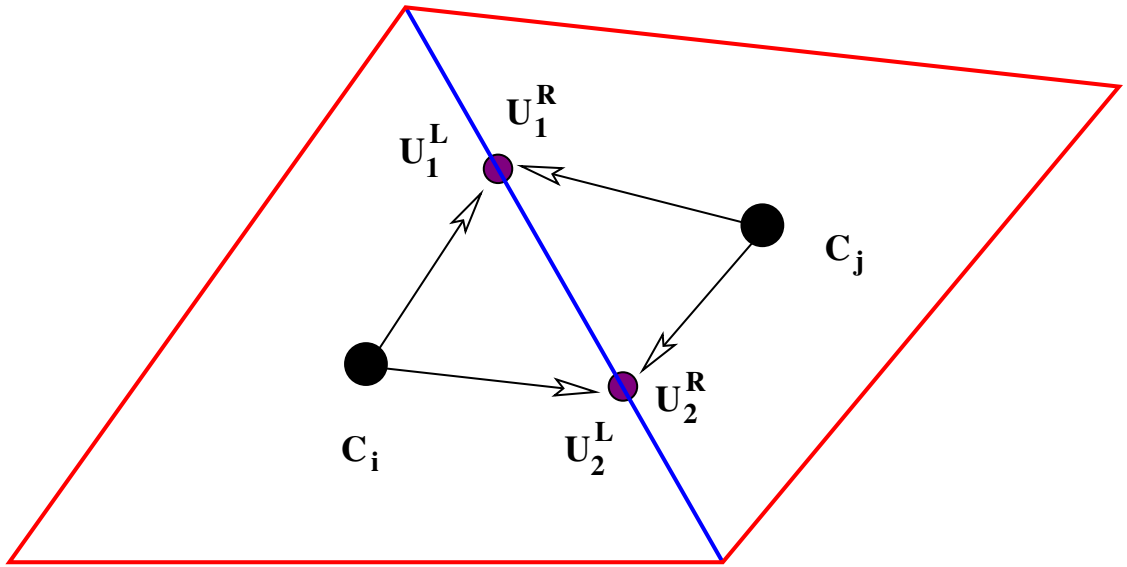


Figure 5.17: Effect of the constant K in Venkatakrishnan's limiter on the convergence for an inviscid flow past a circular arc.

Venkatakrishnan limiter

- Performance depends on K
- Oscillations are not completely avoided
- May fail with strong strongs, hypersonic flows
- Works best when used with non-dimensional variables

Third order scheme



Quadratic reconstruction in cell C_i

$$\begin{aligned}\tilde{U}(x,y) = U_i &+ a_i(x-x_i) + b_i(y-y_i) \\ &+ c_i(x-x_i)^2 + d_i(x-x_i)(y-y_i) + e_i(y-y_i)^2\end{aligned}$$

Third order scheme

2-point Gauss quadrature for flux

$$F_{ij} = \omega_1 F(U_1^L, U_1^R, \hat{n}_{ij}) + \omega_2 F(U_2^L, U_2^R, \hat{n}_{ij})$$

For more, see Barth, VKI Lecture notes, 1994.

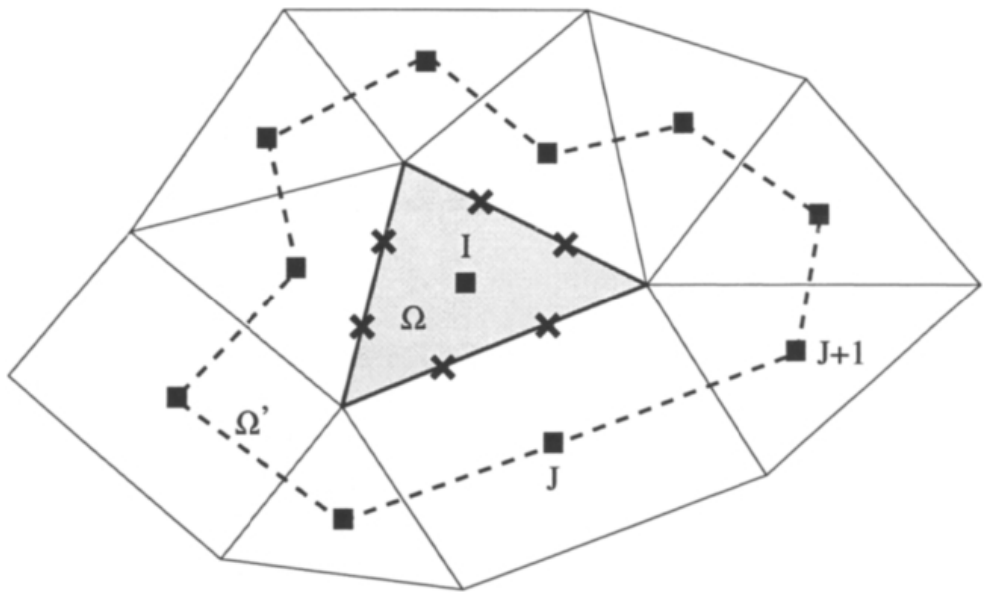


Figure 5.14: Stencil of the quadratic reconstruction method due to Delanaye [62], [63] in 2D (filled rectangles). Dashed line represents the integration path of the Green-Gauss gradient evaluation (control volume Ω'). Crosses denote the quadrature points for integration of the fluxes.

Viscous flux

Viscous flux requires gradients of velocity and temperature at the mid-point of the cell face. There are two options

- Compute gradient at cell centers and average to obtain gradient at face. These gradients are anyway already computed for reconstruction purpose. It can be used in cell-centered or vertex-centered schemes.
- Compute gradient at face center directly. This approach can be used in the cell centered case. It requires additional least squares or Green-Gauss technique to be applied at each face. Hence it requires more computations.
- Vertex-centered grids: A finite element type approach can also be used which leads to a compact stencil for viscous fluxes.

Viscous flux: Using cell gradients

Assume that gradients of W at cell center are available using least squares or Green-Gauss technique. To compute viscous flux at face S_{ij}, obvious thing to do is

$$\nabla W_{ij} = \overline{\nabla W}_{ij} = \frac{1}{2}(\nabla W_i + \nabla W_j)$$

and then compute viscous flux using these gradients. But this leads to **odd-even decoupling** problem. To avoid this we approximate gradient as

$$\nabla W_{ij} = \overline{\nabla W}_{ij} + \left(\frac{W_j - W_i}{|r_j - r_i|} - \overline{\nabla W}_{ij} \cdot \hat{r}_{ij} \right) \hat{r}_{ij}$$

where

$$\hat{r}_{ij} = \frac{r_j - r_i}{|r_j - r_i|} = \text{unit vector from } i \text{ to } j$$

Viscous flux: Using cell gradients

This definition is consistent in the sense that it gives correct directional derivative

$$\begin{aligned}\nabla W_{ij} \cdot \hat{r}_{ij} &= \overline{\nabla W}_{ij} \cdot \hat{r}_{ij} + \left(\frac{W_j - W_i}{|r_j - r_i|} - \overline{\nabla W}_{ij} \cdot \hat{r}_{ij} \right) \hat{r}_{ij} \cdot \hat{r}_{ij} \\ &= \frac{W_j - W_i}{|r_j - r_i|}\end{aligned}$$

This corrected scheme avoids the odd-even decoupling problem.

Remark: The stencil for this scheme is large since it involves secondary neighbours.

Viscous flux: using face gradient approach

This is used in case of cell-centered schemes. There are several approaches.

One possibility is to

- interpolate W from cell centers to the vertices.
- compute gradient using Green-Gauss theorem on the face centered control volume

Second approach: Compute gradient at the vertices using Green-Gauss formula. Then average gradient to find gradient at face center. (Jameson's vertex-centroid scheme)

Another approach due to Frink also makes use of vertex values

$$(x_j - x_i)W_x + (y_j - y_i)W_y = W_j - W_i$$

$$(x_2 - x_1)W_x + (y_2 - y_1)W_y = W_2 - W_1$$

This coupled system can be solved to obtain the derivatives W_x and W_y.

Viscous flux: FEM approach

This approach can be used for vertex-centered scheme on triangular and tetrahedral grids. The boundary ∂C_i of dual cell C_i contains portions located inside different triangles. On each triangle T we know the solution at its three vertices i, j, k and we can approximate the gradient using Green-Gauss theorem

$$\nabla W_T = \frac{1}{|T|}\left[\frac{W_i + W_j}{2} N_{ij} + \frac{W_j + W_k}{2} N_{jk} + \frac{W_k + W_i}{2} N_{ki}\right]$$

$$N_{ij} = \begin{bmatrix} +(y_j - y_i) \\ -(x_j - x_i) \end{bmatrix} = \text{normal to edge } ij$$

The flux across the face inside triangle T is computed using ∇W_T. Thus the stencil of the scheme involves only the nearest neighbours.

Boundary conditions

There are

- natural boundaries: solid wall
- artificial boundaries: inlet, outlet, farfield, symmetry plane, etc.

Natural boundary conditions are known from the PDE problem itself. Artificial boundary conditions have to be cooked up to lead to a stable scheme. Characteristics are useful to know how much information has to be specified.

In the finite volume method, boundary conditions are implemented through fluxes (mostly).

Solid wall: inviscid flux

The condition is that normal velocity must be zero. Fluid cannot enter into a solid surface. This is true of both inviscid and viscous flows. In inviscid flows, there can be a tangential component of velocity but this has to be determined from the numerical scheme. If normal velocity is zero, the flux is

$$\begin{bmatrix} 0 \\ pn_x \\ pn_y \\ 0 \end{bmatrix}$$

Cell-centered case: First order scheme, use the pressure in the adjacent cell to compute the flux. To get higher order accuracy, one can extrapolate the pressure from the cell center to the face mid-point.

Vertex-centered case: We already know pressure on the boundary face. Use it to calculate the flux.

Solid wall: inviscid flux using ghost cells

We can introduce a ghost cell inside the solid surface and set the values in the ghost cell to appropiately recover the flux.

Inviscid flows:

- The density and pressure in ghost cell are set to be same as in the real cell.
- The tangential velocity is same as in real cell but normal velocity is reversed.

Viscous flows:

- Adiabatic wall: Density and pressure are set same as in real cell.
- Isothermal wall: Pressure is same as in real cell, density is computed using specified temperature and known pressure.
- All velocity components are reversed in the ghost cell.

Now we have two states at the boundary face and we can use a numerical flux function to compute the flux. It is important to ensure that mass and energy fluxes are zero.

Solid wall: Viscous flux

We know the gradients of velocity at the solid wall face by some procedure. We use this to calculate shear stress and the viscous fluxes due to shear stress.

If we have adiabatic wall, then the heat flux is set to zero. For isothermal wall, heat flux is calculated from Fourier law and added to energy equation.

Vertex-centered case: We have unknowns on the solid wall. It is usual practice to set the velocity at these location to zero after updating the solution. For isothermal wall, the temperature is set to the specified wall temperature.

Supersonic inflow and outflow

At supersonic inflow, all wave speeds are positive. So the flux must be determined from the inflow conditions.

Similarly, at a supersonic outflow, the flux is determined from the state in the boundary cell.

Subsonic inflow

We have three positive and one negative eigenvalue.

$$u_n - a, u_n, u_n < 0 \qquad u_n + a > 0$$

So we have to specify three pieces of information on the inflow side and the remaining information must be taken from the interior cell. One choice is to specify velocity and pressure or density from the inflow conditions and the remaining is taken from the interior cell.

Alternately one can use local characteristic variables.

Alternately, we can use an upwind numerical flux like Steger-Warming which is already based on characteristic splitting to compute the flux at inflow face

$$H_{sw}(U_0, U_\infty, n) = A^+(U_0, n)U_0 + A^-(U_\infty, n)U_\infty$$

Subsonic outflow

We have three positive and one negative eigenvalue.

$$u_n - a < 0 \qquad u_n, u_n, u_n + a > 0$$

So we have to specify three pieces of information on the interior side and the remaining information must be taken from the exterior of the domain. Usually, one knows the pressure on the outlet side so this can be specified. The density and velocity is taken from the interior cell.

Alternately one can use local characteristic variables. This is explained in the far-field boundary conditions.

Alternately, we can use an upwind numerical flux like Steger-Warming to compute the flux at outflow face

$$H_{sw}(U_0, U_{out}, n) = A^+(U_0, n)U_0 + A^-(U_{out}, n)U_{out}$$

U_{out} is determined using the known outlet pressure and remaining values are taken from the interior cell.

Farfield conditions

This is typical of aerospace applications like flow around an airfoil or aircraft. The real domain contains the whole exterior of the airfoil but for computational purpose, we have to use a finite domain. It is important to place the farfield domain sufficiently far away from the lifting bodies.

Characteristic approach: (See Wesseling, section 12.4) We look at a one dimensional problem in the direction of the outward normal ignoring the variation in other directions. Then one can find characteristic variables (Riemann invariants) and corresponding speeds as

$$
\begin{aligned}
W_1 &= u_n - \frac{2a}{\gamma - 1}, & \lambda_1 &= u_n - a \\
W_2 &= S, & \lambda_2 &= u_n \\
W_3 &= u_t, & \lambda_3 &= u_n \\
W_4 &= u_n + \frac{2a}{\gamma - 1}, & \lambda_4 &= u_n + a
\end{aligned}
$$

Farfield conditions

At the far-field face we calculate W_f as

$$W_{f,i} = \begin{cases} W_{0,i} & \lambda_i \geq 0 \\ W_{\infty,i} & \lambda_i < 0 \end{cases}, \qquad i = 1, \ldots, 4$$

Once W_f is obtained then we can calculate the state (ρ, u, v, p) at the face from which the flux can be obtained.

$$u_{n,f} = \frac{1}{2}(W_{f,1} + W_{f,4}), \quad a_f = \frac{\gamma - 1}{4}(W_{f,4} - W_{f,1})$$

$$u_{t,f} = W_{f,3}, \qquad S_f = \frac{p_f}{\rho_f^\gamma} = W_{f,2}$$

Remark: At some outflow boundaries, one usually knows the outlet pressure, maybe the atmospheric pressure. In that case, the known pressure can be used.

Farfield conditions

Remark: There are other sets of characteristic variables, see Lohner, section 8.5 and Blazek, section 8.3.1

Characteristic-based flux approach: Alternately, we can use an upwind numerical flux like Steger-Warming which is already based on characteristic splitting to compute the flux at farfield face

$$H_{sw}(U_0, U_\infty, n) = A^+(U_0, n)U_0 + A^-(U_\infty, n)U_\infty$$

Remark: For lifting problems like airfoils in 2-D, the far-field domain may have to be placed about 50-100 chord lengths away from the airfoil in order to minimize the effect of artificial boundary conditions. One can use point vortex model to correct the far-field conditions which allow placing the outer boundary closer to the airfoil, see Blazek, Chap. 8. It is good to do some study on the effect of outer boundary position on lift and drag coefficients.

Farfield conditions

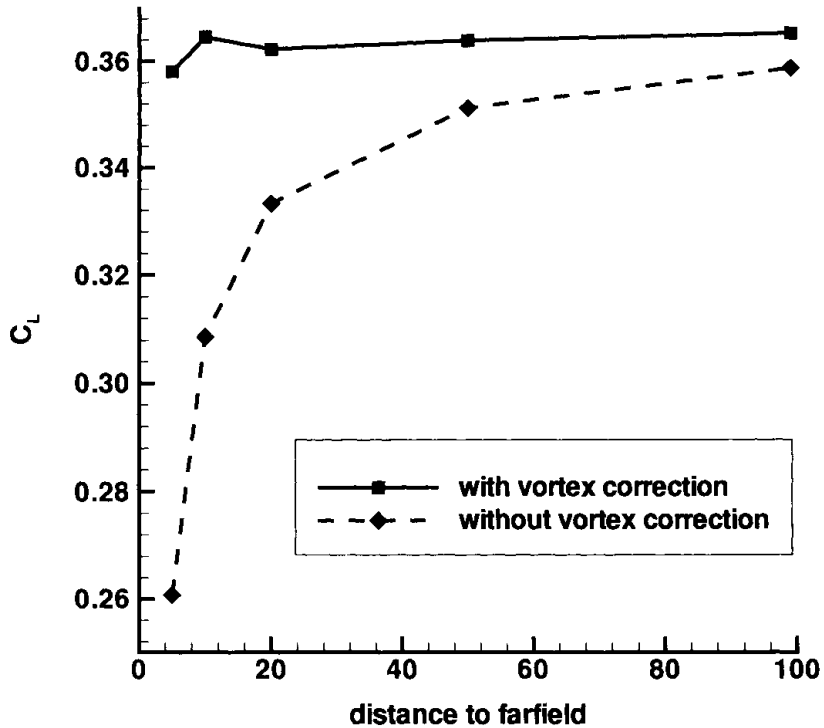


Figure 8.7: Effects of distance to the farfield boundary and of single vortex on the lift coefficient. NACA 0012 airfoil, $M_\infty = 0.8$, $\alpha = 1.25°$.