Chapter 11
Discretization of the Convection Term

Abstract So far, the discretization of the general steady diffusion equation has been formulated on orthogonal, non-orthogonal, structured, and unstructured grids. Another important term, the convection term represented by the divergence operator, is the focus of this chapter. Initially this term is discretized using a symmetrical linear profile similar to the one adopted for the discretization of the diffusion term. The shortcomings of this profile are delineated and a remedy is suggested through the use of an upwind profile. Even though it leads to physically plausible predictions, the upwind profile is shown to be highly diffusive generating results that are first order accurate. To increase accuracy, higher order profiles that are upwind biased are introduced. While reducing the discretization error, higher order profiles are shown to give rise to another type of error known as the dispersion error. Methods dealing with this error will be dealt with in the next chapter. Moreover, the flow field, which represents the driving catalyst of the convection term, is assumed to be known. The computation of the flow field will be the subject of later chapters.

11.1 Introduction

Although the convection term looks simple, its discretization presents a number of difficulties that have occupied researchers for more than three decades now. Their work has shed light on the problems that hindered its discretization and resulted in the development of a large number of convection schemes. The body of literature is so large that two chapters are devoted for the analysis of this term. In this chapter the basic concepts are introduced and High Order (HO) [1–3] upwind biased schemes are discussed. The bounding of the convective flux, which made possible the development of non-oscillatory convection schemes of high order of accuracy, denoted by High Resolution (HR) schemes, will be discussed in the next chapter.

For clarity, the presentation of new concepts will be initially introduced using a one dimensional grid, and then extended to multi dimensional non-orthogonal grids. The chapter starts with the discretization of the one-dimensional convection-diffusion

F. Moukalled et al., *The Finite Volume Method in Computational Fluid Dynamics*,
Fluid Mechanics and Its Applications 113, DOI 10.1007/978-3-319-16874-6_11

problem to highlight, through a stability criterion, the shortcomings of the central difference scheme [4, 5]. The upwind scheme [6] is shown to pass the stability test. The numerical diffusion error associated with low order schemes and the numerical dispersion error accompanying high order schemes are also explained. The treatment of the class of HR schemes will be discussed in the next chapter.

11.2 Steady One Dimensional Convection and Diffusion

Initially the derivations are performed for a very simple, one dimensional, steady convection-diffusion problem to learn as much as possible without being distracted by the complexity of the calculations. The governing equation for the case studied can be written as

$$\frac{d(\rho u\phi)}{dx} - \frac{d}{dx}\left(\Gamma^{\phi}\frac{d\phi}{dx}\right) = 0 \tag{11.1}$$

Luckily an analytical solution for the problem is available, and is used as a reference with which various numerical solutions are compared.

11.2.1 Analytical Solution

The continuity equation for this steady-state one dimensional problem of constant cross sectional area is given by

$$\frac{d(\rho u)}{dx} = 0 \tag{11.2}$$

implying that ρu is constant. Having this in mind and integrating with respect to x, Eq. (11.1) becomes

$$\rho u\phi - \Gamma^{\phi}\frac{d\phi}{dx} = c_1 \tag{11.3}$$

where c_1 is a constant of integration the value of which depends on the boundary conditions used. Rearranging, Eq. (11.3) is rewritten as

$$\frac{d\phi}{dx} = \frac{\rho u}{\Gamma^{\phi}}\phi - \frac{c_1}{\Gamma^{\phi}} \tag{11.4}$$

Through a change of variable, Eq. (11.4) is transformed to

$$\frac{d\Phi}{dx} = \frac{\rho u}{\Gamma^{\phi}}\Phi \tag{11.5}$$

where

$$\Phi = \frac{\rho u}{\Gamma^{\phi}}\phi - \frac{c_1}{\Gamma^{\phi}} \tag{11.6}$$

Separating variables and integrating, the solution to Eq. (11.5) is found to be

$$\frac{d\Phi}{\Phi} = \frac{\rho u}{\Gamma^{\phi}}dx \Rightarrow Ln(\Phi) = \frac{\rho u}{\Gamma^{\phi}}x + c_3 \Rightarrow \Phi = c_2 e^{\frac{\rho u}{\Gamma^{\phi}}x} \tag{11.7}$$

where c_2 is another constant of integration. Reverting back to the original variable, the general solution for ϕ is given by

$$\phi = \frac{c_2\Gamma^{\phi}e^{\frac{\rho u}{\Gamma^{\phi}}x} + c_1}{\rho u} \tag{11.8}$$

Thus the analytical solution between the two points W and E shown in Fig. 11.1 and subject to

$$\begin{cases} \phi = \phi_W \; at \; x = x_W \\ \phi = \phi_E \; at \; x = x_E \end{cases} \tag{11.9}$$

is obtained as

$$\frac{\phi - \phi_W}{\phi_E - \phi_W} = \frac{e^{Pe_L\frac{x-x_W}{L}} - 1}{e^{Pe_L} - 1} \tag{11.10}$$

where Pe_L is the Péclet number (based on the length L), which represents the ratio of the advective transport rate of ϕ to its diffusive transport rate, and is given by

$$Pe_L = \frac{\rho u L}{\Gamma^{\phi}} \quad \text{and} \quad L = x_E - x_W. \tag{11.11}$$

Equation (11.10) is evaluated for different values of Pe_L and results are displayed in Fig. 11.2. As shown, variations in ϕ between W and E change from a linear profile for pure diffusion problems to almost a step profile at high values of Pe_L.

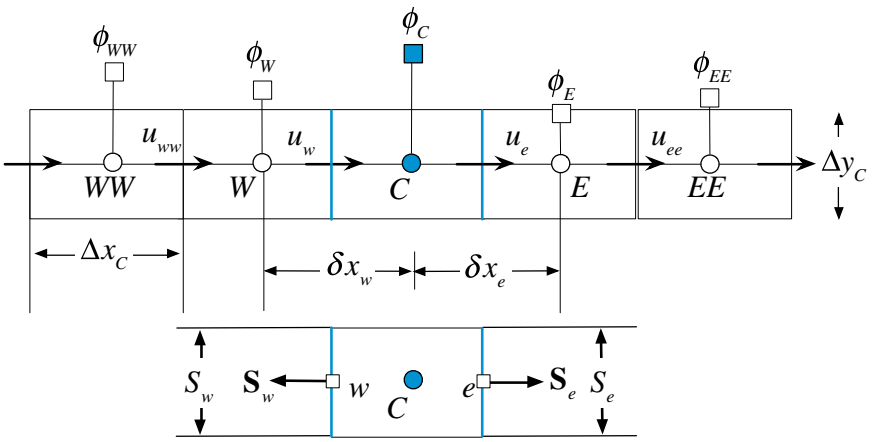

Fig. 11.1 Notation for a one dimensional grid system

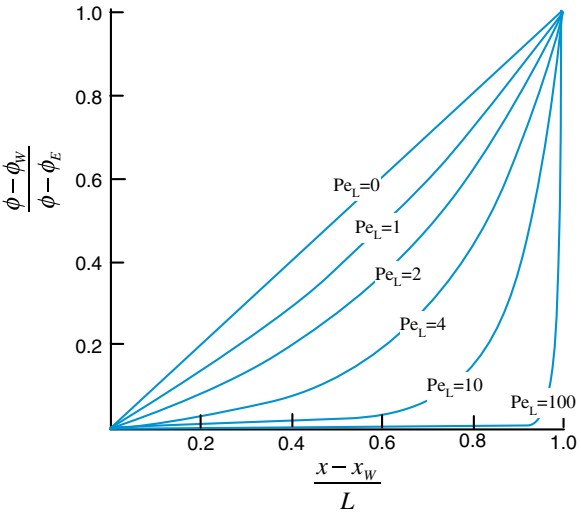

Fig. 11.2 Analytical solutions of the one dimensional convection and diffusion problem for various Pe_L

11.2.2 Numerical Solution

The discretization of Eq. (11.1) starts by integrating the conservation equation over the one dimensional element shown in Fig. 11.1 to yield

$$\int_{V_c} \left[\nabla \cdot (\rho \mathbf{v}\phi) - \nabla \cdot \left(\Gamma^{\phi} \nabla \phi\right)\right] dV = 0 \tag{11.12}$$

where $\mathbf{v} = u\mathbf{i}$ is the velocity vector. The conservation equation can be written in terms of the convection and diffusion fluxes as

$$\int_{V_c} \nabla \cdot \left(\mathbf{J}^{\phi,C} + \mathbf{J}^{\phi,D}\right) dV = 0 \quad \text{where} \quad \mathbf{J}^{\phi,C} = \rho \mathbf{v} \phi \quad \text{and} \quad \mathbf{J}^{\phi,D} = -\Gamma^{\phi} \nabla \phi. \tag{11.13}$$

Then, using the divergence theorem, the volume integral is transformed into a surface integral giving

$$\int_{V_c} \nabla \cdot \left(\mathbf{J}^{\phi,C} + \mathbf{J}^{\phi,D}\right) dV = \int_{\partial V_c} \left(\mathbf{J}^{\phi,C} + \mathbf{J}^{\phi,D}\right) \cdot d\mathbf{S} = \int_{\partial V_c} \left[\rho u \phi \mathbf{i} - \Gamma^{\phi} \frac{d\phi}{dx} \mathbf{i}\right] \cdot d\mathbf{S} = 0. \tag{11.14}$$

Replacing the surface integral by a summation of fluxes over the element faces, Eq. (11.14) becomes

$$\sum_{f \sim nb(C)} \left(\rho u \phi \mathbf{i} - \Gamma^{\phi} \frac{d\phi}{dx} \mathbf{i}\right)_f \cdot \mathbf{S}_f = 0. \tag{11.15}$$

Noting that the surface vectors on opposite sides of the element have opposite signs, the expanded form of Eq. (11.15) for a constant cross section is obtained as

$$\left[(\rho u \Delta y \phi)_e - \left(\Gamma^{\phi} \frac{d\phi}{dx} \Delta y\right)_e\right] - \left[(\rho u \Delta y \phi)_w - \left(\Gamma^{\phi} \frac{d\phi}{dx} \Delta y\right)_w\right] = 0 \tag{11.16}$$

The values of u at the cell faces are known, and those of the gradient can be discretized in the way previously described. The question is how to proceed in the discretization of the face values ϕ_e and ϕ_w in terms of the values at adjacent nodes. The method of specifying these face values is denoted in the literature by the "advection scheme".

11.2.3 A Preliminary Derivation: The Central Difference (CD) Scheme

At first sight the "obvious" answer would be a linear interpolation profile similar to the one used for the diffusion term. Hence, the value of ϕ at a given face, say the right face e, will be computed as

$$\phi(x) = k_0 + k_1 (x - x_C) \tag{11.17}$$

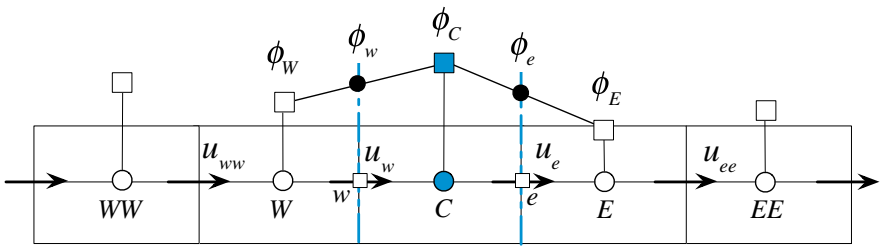

Fig. 11.3 The Central Difference scheme profile

where k_0 and k_1 are constants evaluated using the two nodes straddling face e. Thus using the fact that $\phi = \phi_E$ at $x = x_E$ and $\phi = \phi_C$ at $x = x_C$, Eq. (11.17) evaluated at $x = x_e$ gives

$$\phi_e = \phi_C + \frac{(\phi_E - \phi_C)}{(x_E - x_C)}(x_e - x_C). \tag{11.18}$$

This is basically the central difference scheme that can be obtained by a Taylor series expansion where terms involving derivatives of the second order and higher are neglected, which means it is second order accurate.

For the uniform grid shown in Fig. 11.3, Eq. (11.18) becomes

$$\phi_e = \frac{\phi_C + \phi_E}{2} \tag{11.19}$$

Example 1

Derive the value of ϕ_e for a central difference scheme using a Taylor expansion approach

Solution

The Taylor expansion about e can *be written as*

$$\phi_C = \underbrace{\phi(-\Delta x_C/2)}_{=\phi_e} + \left.\frac{\partial \phi}{\partial x}\right|_e (-\Delta x_C/2) + O\left(\Delta x_C^2\right)$$

$$= \phi_e - \frac{\phi_E - \phi_C}{\delta x_e}\frac{\Delta x_C}{2}$$

thus

$$\phi_e = \phi_C + \frac{\Delta x_C}{\delta x_e}\frac{\phi_E - \phi_C}{2}$$

for a uniform grid $\delta x_e = \Delta x_C$ *yielding*

$$\phi_e = \frac{\phi_C + \phi_E}{2}$$

Thus after the discretization of the diffusion term using a linear profile (Fig. 11.3), the term in the first square bracket of Eq. (11.16) becomes

$$\begin{aligned}(\rho u \Delta y \phi)_e - \left(\Gamma^\phi \frac{d\phi}{dx}\Delta y\right)_e &= (\rho u \Delta y)_e \frac{(\phi_E + \phi_C)}{2} - \left(\Gamma^\phi \frac{\Delta y}{\delta x}\right)_e (\phi_E - \phi_C) \\ &= FluxC_e\phi_C + FluxF_e\phi_E + FluxV_e\end{aligned} \tag{11.20}$$

where

$$\begin{aligned}FluxC_e &= \Gamma_e^\phi \frac{\Delta y_e}{\delta x_e} + \frac{(\rho u \Delta y)_e}{2} \\ FluxF_e &= -\Gamma_e^\phi \frac{\Delta y_e}{\delta x_e} + \frac{(\rho u \Delta y)_e}{2} \\ FluxV_e &= 0\end{aligned} \tag{11.21}$$

A similar expression for the term in the second square bracket of Eq. (11.16) can also be derived and is given by

$$\begin{aligned}-\left[(\rho u \Delta y \phi)_w - \left(\Gamma^\phi \frac{d\phi}{dx}\Delta y\right)_w\right] &= -\left[(\rho u \Delta y)_w \frac{(\phi_W + \phi_C)}{2} - \left(\Gamma^\phi \frac{\Delta y}{\delta x}\right)_w (\phi_C - \phi_W)\right] \\ &= FluxC_w\phi_C + FluxF_w\phi_W + FluxV_w\end{aligned} \tag{11.22}$$

where now

$$\begin{aligned}FluxC_w &= \Gamma_w^\phi \frac{\Delta y_w}{\delta x_w} - \frac{(\rho u \Delta y)_w}{2} \\ FluxF_w &= -\Gamma_w^\phi \frac{\Delta y_w}{\delta x_w} - \frac{(\rho u \Delta y)_w}{2} \\ FluxV_w &= 0\end{aligned} \tag{11.23}$$

Substitution of these values into the convection-diffusion equation yields

$$a_C\phi_C + a_E\phi_E + a_W\phi_W = 0 \tag{11.24}$$

where

$$\begin{aligned} a_E &= FluxF_e = -\Gamma_e^{\phi}\frac{\Delta y_e}{\delta x_e} + \frac{(\rho u \Delta y)_e}{2} \\ a_W &= FluxF_w = -\Gamma_w^{\phi}\frac{\Delta y_w}{\delta x_w} - \frac{(\rho u \Delta y)_w}{2} \\ a_C &= FluxC_e + FluxC_w = \left(\frac{(\rho u \Delta y)_e}{2} + \Gamma_e^{\phi}\frac{\Delta y_e}{\delta x_e}\right) + \left(-\frac{(\rho u \Delta y)_w}{2} + \Gamma_w^{\phi}\frac{\Delta y_w}{\delta x_w}\right) \end{aligned} \tag{11.25}$$

As the problem is one dimensional $\Delta y_e = \Delta y_w$ and, without loss of generality, can be set equal to 1. Moreover, from continuity u is constant and thus $(\rho u \Delta y)_e - (\rho u \Delta y)_w = 0$. Assuming a uniform diffusion coefficient, the coefficients of the discretized equation can be simplified to

$$\begin{aligned} a_E &= -\frac{\Gamma^{\phi}}{x_E - x_C} + \frac{(\rho u)_e}{2} \\ a_W &= -\frac{\Gamma^{\phi}}{x_C - x_W} - \frac{(\rho u)_w}{2} \\ a_C &= -(a_E + a_W) \end{aligned} \tag{11.26}$$

Substituting back in Eq. (11.24), the value for ϕ_C is found as

$$\frac{\phi_C - \phi_W}{\phi_E - \phi_W} = \frac{a_E}{a_E + a_W} \tag{11.27}$$

If the grid is assumed to be uniform, then the above equation can be written in terms of Pe_L as

$$\frac{\phi_C - \phi_W}{\phi_E - \phi_W} = \frac{1}{2}\left(1 - \frac{Pe_L}{2}\right) \tag{11.28}$$

where L is $(x_E - x_W)$, which is the size of two elements. The analytical solution for the problem can be obtained from Eq. (11.10) by setting $(x - x_W)/L$ to 0.5 and is given by

$$\frac{\phi_C - \phi_W}{\phi_E - \phi_W} = \frac{e^{\frac{Pe_L}{2}} - 1}{e^{Pe_L} - 1} \tag{11.29}$$

The two solutions are compared in Fig. 11.4 as Pe_L varies from −10 to +10. Figure 11.4 demonstrates that at low values of Pe_L the numerical and analytical

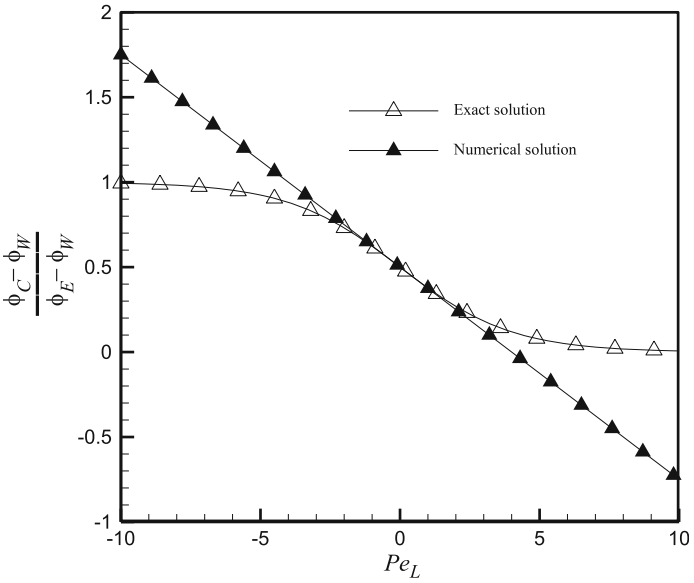

Fig. 11.4 Comparison of numerical (obtained with the CD scheme) and analytical solutions for the one dimensional convection and diffusion problem

solutions are very close to each other. However as Pe_L is increased beyond a certain value the central difference (CD) numerical solution greatly departs from the analytical solution and becomes unbounded experiencing unphysical behavior. Whereas the analytical solution approaches asymptotically the values 0 and 1 for positive and negative values of Pe_L, respectively, the CD solution decreases linearly from $+\infty$ to $-\infty$ as Pe_L increases from $-\infty$ to $+\infty$. This solution indicates that some of the assumptions used in the discretization of the equation are unrealistic or unphysical. What is the reason for this behavior?

As depicted in Fig. 11.5, whereas diffusion at point C is equally affected by conditions upstream and downstream of C (Fig. 11.5a), the advection process is a highly directional process transporting properties only in the direction of the flow (Fig. 11.5b). Therefore the linear profile approximation, which assigns equal weight to both the upwind and downwind nodes, is a good approximation for the diffusion term (Fig. 11.5a). However it cannot describe the directional preference of convection, for which a step profile is more appropriate (Fig. 11.5b), and is the cause of the problem.

The combined convection-diffusion zone of influence and the more relevant profile in this case is schematically depicted in Fig. 11.5c. This zone of influence approaches the diffusion region displayed in Fig. 11.5a and the advection region depicted in Fig. 11.5b at low and high values of the Péclet number, respectively.

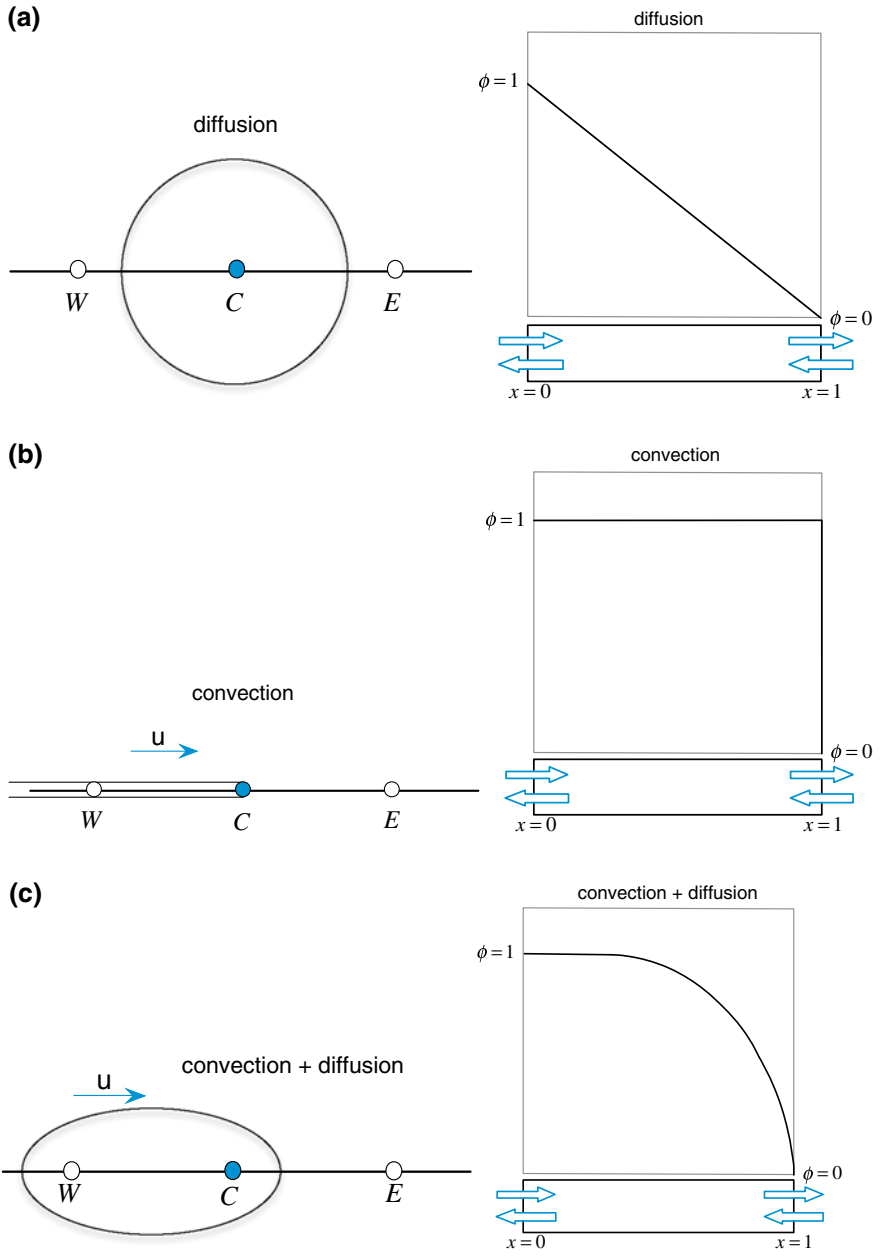

Fig. 11.5 Zone of influence of **a** diffusion, **b** convection, and **c** combined convection and diffusion terms and expected problem solution

Therefore, as long as diffusion is the dominant transfer mechanism, the use of a linear profile yields physical results. However, once convection overwhelms diffusion, unphysical results are obtained. The value of Péclet number at which this occurs can be easily calculated. Assuming the flow to be in the positive x direction, it is noted that there is a possibility for the a_E coefficient to become positive, thus leading to unphysical results (if the flow is in the negative x direction, then a_W may become positive) when,

$$-\Gamma_e^{\phi}\frac{\Delta y_e}{\delta x_e}+\frac{(\rho u\Delta y)_e}{2}>0\Rightarrow\frac{(\rho u)_e\delta x_e}{\Gamma_e^{\phi}}>2. \tag{11.30}$$

Defining the cell Péclet number as

$$Pe=\frac{\rho u\delta x}{\Gamma^{\phi}}. \tag{11.31}$$

which, for a uniform grid, is half Pe_L, then Eq. (11.30) can be rewritten as

$$Pe>2. \tag{11.32}$$

Thus for a cell Péclet number (Pe) larger than 2 the discretization process becomes inconsistent as now an increase in the neighboring value will lead to a decrease in the value at C. This in turn will lead to a further increase in the neighboring value, and the error is magnified.

This situation can be avoided by decreasing the grid size so that the cell Péclet number is smaller than 2. For many practical situations, however, the increase in storage and computational requirements may be too large to afford. Moreover for purely convected flows (e.g., Euler flows) this is simply not feasible. Therefore a remedy is needed.

11.2.4 The Upwind Scheme

When examining the discretization procedure described above, it is noticed that the reason for obtaining these positive coefficients is because of the adopted linear symmetric profile. A linear symmetric profile gives equal weights to the two nodes sharing the face with no directional preference, which is appropriate for non-directional phenomena with an elliptic type term, such as the diffusion term. It is simply not adequate for the convection term [4].

A scheme that is more compatible with the advection process is the upwind scheme [4, 6] schematically displayed in Fig. 11.6. The upwind scheme basically mimics the basic physics of advection in that the cell face value is made dependent on the upwind nodal value, i.e., dependent on the flow direction. In this case the cell face values for the configuration displayed in Fig. 11.6 are given by

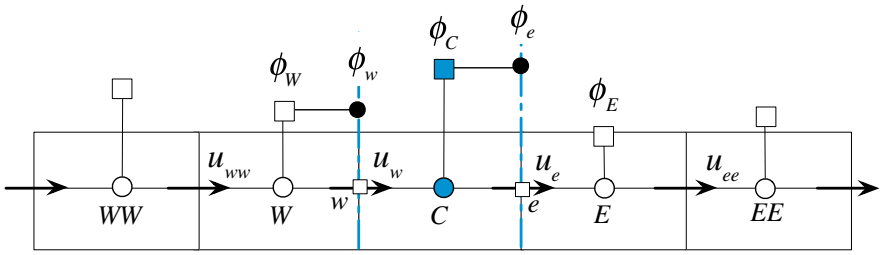

Fig. 11.6 The upwind scheme profile

$$\phi_e = \begin{cases} \phi_C & if \ \dot{m}_e > 0 \\ \phi_E & if \ \dot{m}_e < 0 \end{cases} \quad \text{and} \quad \phi_w = \begin{cases} \phi_C & if \ \dot{m}_w > 0 \\ \phi_W & if \ \dot{m}_w < 0 \end{cases} \tag{11.33}$$

where $\dot{m}_e$ and $\dot{m}_w$ are the mass flow rates at faces e and w given by

$$\begin{aligned} \dot{m}_e &= (\rho \mathbf{v} \cdot \mathbf{S})_e = (\rho u S)_e = (\rho u \Delta y)_e \\ \dot{m}_w &= (\rho \mathbf{v} \cdot \mathbf{S})_w = -(\rho u S)_w = -(\rho u \Delta y)_w \end{aligned} \tag{11.34}$$

Thus, the advection flux at face e can be written as

$$\begin{aligned} \dot{m}_e \phi_e &= \|\dot{m}_e, 0\| \phi_C - \|-\dot{m}_e, 0\| \phi_E \\ &= FluxC_e^{Conv} \phi_C + FluxF_e^{Conv} \phi_E + FluxV_e^{Conv} \end{aligned} \tag{11.35}$$

where

$$\begin{aligned} FluxC_e^{Conv} &= \|\dot{m}_e, 0\| \\ FluxF_e^{Conv} &= -\|-\dot{m}_e, 0\| \\ FluxV_e^{Conv} &= 0 \end{aligned} \tag{11.36}$$

In Eqs. (11.35) and (11.36), the term $\|a, b\|$ represents the maximum of a and b. Moreover, a similar relation can be derived for the advection flux at face w and is given by

$$\begin{aligned} \dot{m}_w \phi_w &= \|\dot{m}_w, 0\| \phi_C - \|-\dot{m}_w, 0\| \phi_W \\ &= FluxC_w^{Conv} \phi_C + FluxF_w^{Conv} \phi_W + FluxV_w^{Conv} \end{aligned} \tag{11.37}$$

where now

$$\begin{aligned} FluxC_w^{Conv} &= \|\dot{m}_w, 0\| \\ FluxF_w^{Conv} &= -\|-\dot{m}_w, 0\| \\ FluxV_w^{Conv} &= 0 \end{aligned} \tag{11.38}$$

Denoting the contribution of the diffusion flux with a superscript *Diff*, then substitution into Eq. (11.15) yields

$$\begin{aligned}\left(FluxC_e^{Conv} + FluxC_e^{Diff} + FluxC_w^{Conv} + FluxC_w^{Diff}\right)\phi_C \\ + \left(FluxF_e^{Conv} + FluxF_e^{Diff}\right)\phi_E + \left(FluxF_w^{Conv} + FluxF_w^{Diff}\right)\phi_W = 0\end{aligned} \tag{11.39}$$

which can be modified into the form

$$a_C\phi_C + a_E\phi_E + a_W\phi_W = 0 \tag{11.40}$$

with

$$\begin{aligned} a_E &= FluxF_e^{Conv} + FluxF_e^{Diff} \\ &= -\|-\dot{m}_e, 0\| - \Gamma_e^{\phi}\frac{S_e}{\delta x_e} \\ a_W &= FluxF_w^{Conv} + FluxF_w^{Diff} \\ &= -\|-\dot{m}_w, 0\| - \Gamma_w^{\phi}\frac{S_w}{\delta x_w} \\ a_C &= \sum_f \left(FluxC_f^{Conv} + FluxC_f^{Diff}\right) \\ &= \|\dot{m}_e, 0\| + \|\dot{m}_w, 0\| + \Gamma_e^{\phi}\frac{S_e}{\delta x_e} + \Gamma_w^{\phi}\frac{S_w}{\delta x_w} \\ &= -(a_E + a_W) + \underbrace{(\dot{m}_e + \dot{m}_w)}_{=0} \\ b_C &= -\sum_f \left(FluxV_f^{Conv} + FluxV_f^{Diff}\right) \\ &= 0 \end{aligned} \tag{11.41}$$

It is easily seen that the upwind scheme yields negative neighbor coefficients, and provided continuity is satisfied, the coefficient at the main node is given by,

$$a_C = -(a_W + a_E) \tag{11.42}$$

which guarantees the boundedness property.

Invoking the continuity constraint in Eq. (11.41) and assuming uniform grid and constant diffusion coefficient, the value for ϕ_C in terms of ϕ_E and ϕ_W is obtained from Eq. (11.40) and Eq. (11.41) as

$$\frac{\phi_C - \phi_W}{\phi_E - \phi_W} = \frac{2 + ||-Pe_L, 0||}{4 + ||-Pe_L, 0|| + ||Pe_L, 0||} = \frac{2 + ||-Pe_L, 0||}{4 + |Pe_L|} \tag{11.43}$$

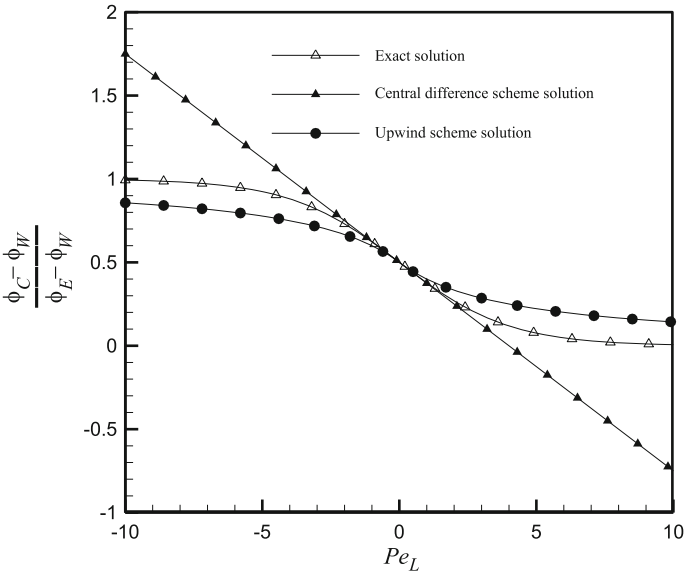

Fig. 11.7 Comparison of solutions obtained analytically and numerically using the upwind and central difference schemes for the one dimensional convection and diffusion problem

The ϕ_C profile generated using the upwind scheme (Eq. 11.43) is compared in Fig. 11.7 with similar ones obtained analytically (Eq. 11.29) and numerically via the central difference scheme (Eq. 11.28). At low Pe_L values, profiles indicate that the upwind scheme is not as accurate as central differencing. This is expected since the upwind profile is first order accurate while the linear profile is second order accurate. At high Pe_L values, the central difference scheme is unstable as its solution is unbounded and physically incorrect. On the other hand, even though the upwind scheme is not particularly accurate, it is physically correct.

Thus there appears to be a tradeoff between accuracy and stability. Using the upwind scheme, solutions are better behaved and bounded even at high Péclet numbers. This is however achieved at the cost of low accuracy. On the other hand, the second order central difference scheme becomes unstable beyond a certain value of Pe_L resulting in physically erroneous solutions. Both schemes seem to be infected by errors, one affecting accuracy while the other affecting stability. What are these errors? This will be discussed after introducing the downwind scheme.

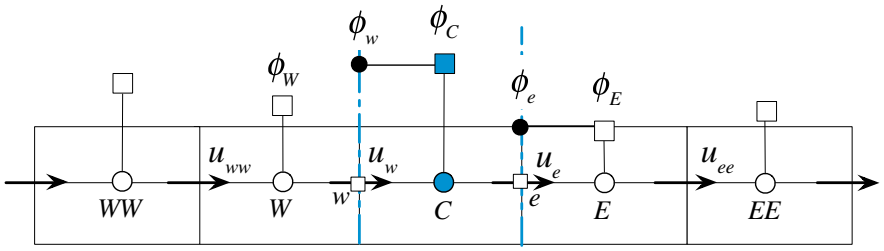

Fig. 11.8 The downwind scheme profile

11.2.5 The Downwind Scheme

It is interesting to see what happens if a scheme opposite to the upwind scheme is used, i.e., the downwind scheme [7, 8]. For this interpolation profile, displayed in Fig. 11.8, the value at the node on the downwind side of the interface is taken to represent the value at the interface. Thus, the values at faces e and w are calculated as

$$\phi_e = \begin{cases} \phi_E & if \quad \dot{m}_e > 0 \\ \phi_C & if \quad \dot{m}_e < 0 \end{cases} \quad \text{and} \quad \phi_w = \begin{cases} \phi_W & if \quad \dot{m}_w > 0 \\ \phi_C & if \quad \dot{m}_w < 0 \end{cases} \tag{11.44}$$

Using Eq. (11.44), the advection fluxes at the faces can be written as

$$\begin{aligned} \dot{m}_e\phi_e &= -\|-\dot{m}_e, 0\|\phi_C + \|\dot{m}_e, 0\|\phi_E \\ &= FluxC_e^{Conv}\phi_C + FluxF_e^{Conv}\phi_E + FluxV_e^{Conv} \\ \dot{m}_w\phi_w &= -\|-\dot{m}_w, 0\|\phi_C + \|\dot{m}_w, 0\|\phi_W \\ &= FluxC_w^{Conv}\phi_C + FluxF_w^{Conv}\phi_W + FluxV_w^{Conv} \end{aligned} \tag{11.45}$$

Substitution of the above values in the discretization equation, Eq. (11.15), yields

$$\begin{aligned} &\left(FluxC_e^{Conv} + FluxC_e^{Diff} + FluxC_w^{Conv} + FluxC_w^{Diff}\right)\phi_C \\ &\quad + \left(FluxF_e^{Conv} + FluxF_e^{Diff}\right)\phi_E + \left(FluxF_w^{Conv} + FluxF_w^{Diff}\right)\phi_W = 0 \end{aligned} \tag{11.46}$$

which can be modified into the form

$$a_C\phi_C + a_E\phi_E + a_W\phi_W = 0 \tag{11.47}$$

with

$$\begin{aligned}
a_E &= FluxF_e^{Conv} + FluxF_e^{Diff} \\
&= \|\dot{m}_e, 0\| - \Gamma_e^{\phi} \frac{S_e}{\delta x_e} \\
a_W &= FluxF_w^{Conv} + FluxF_w^{Diff} \\
&= \|\dot{m}_w, 0\| - \Gamma_w^{\phi} \frac{S_w}{\delta x_w} \\
a_C &= \sum_f \left(FluxC_f^{Conv} + FluxC_f^{Diff} \right) \\
&= - \|-\dot{m}_e, 0\| + \Gamma_e^{\phi} \frac{S_e}{\delta x_e} - \|-\dot{m}_w, 0\| + \Gamma_w^{\phi} \frac{S_w}{\delta x_w} \\
&= - (a_E + a_W) + \underbrace{(\dot{m}_e + \dot{m}_w)}_{=0} \\
b_C &= - \sum_f \left(FluxV_f^{Conv} + FluxV_f^{Diff} \right) \\
&= 0
\end{aligned} \tag{11.48}$$

Invoking the continuity constraint in Eq. (11.48) and assuming uniform grid and constant diffusion coefficient, the value for ϕ_C in terms of ϕ_E and ϕ_W is obtained as

$$\frac{\phi_C - \phi_W}{\phi_E - \phi_W} = \frac{2 - ||Pe_L, 0||}{4 - ||-Pe_L, 0|| - ||Pe_L, 0||} = \frac{2 - ||Pe_L, 0||}{4 - |Pe_L|} \tag{11.49}$$

Without plotting Eq. (11.49) it is clear that as $|Pe_L| \rightarrow 4$ the solution becomes completely unbounded confirming once more the analysis presented above. The downwind scheme may be beneficial when blended with other schemes to predict sharp interfaces [7, 8]. Nevertheless its introduction here will be exploited in the next section to give better insight into stability.

11.3 Truncation Error: Numerical Diffusion and Anti-Diffusion

Truncation error occurs due to the approximate nature of the discretization process and is more easily analyzed for one dimensional situations on Cartesian meshes. The diffusion and anti-diffusion of the upwind, downwind, and central difference schemes are presented next.

11.3.1 The Upwind Scheme

Considering the equation discretized via the upwind scheme, the intention is to recover the integral equation while accounting for the truncation error. To do so, ϕ_C and ϕ_W are written as functions of ϕ_e and ϕ_w, respectively, for the case when the flow is assumed to be in the positive x direction. In this case the upwind scheme results in

$$\phi_e = \phi_C \quad \text{and} \quad \phi_w = \phi_W. \tag{11.50}$$

Thus the one dimensional convection diffusion equation discretized using the upwind scheme simplifies to

$$(\rho u \Delta y)_e \phi_C - (\rho u \Delta y)_w \phi_W - \left[\left(\Gamma^\phi \frac{d\phi}{dx}\Delta y\right)_e - \left(\Gamma^\phi \frac{d\phi}{dx}\Delta y\right)_w\right] = 0. \tag{11.51}$$

The one dimensional Taylor series expansion of ϕ_C with respect to its value at cell face e is given by

$$\begin{aligned}\phi_C &= \phi_e + \left(\frac{d\phi}{dx}\right)_e (x_C - x_e) + \frac{1}{2}\left(\frac{d^2\phi}{dx^2}\right)_e (x_C - x_e)^2 + \cdots \\ &= \phi_e - \left(\frac{d\phi}{dx}\right)_e (x_e - x_C) + \cdots\end{aligned} \tag{11.52}$$

and for a uniform grid as

$$\phi_C = \phi_e - \left(\frac{d\phi}{dx}\right)_e \frac{(\delta x)_e}{2} + \frac{1}{2}\left(\frac{d^2\phi}{dx^2}\right)_e \left(\frac{(\delta x)_e}{2}\right)^2 \cdots \tag{11.53}$$

A similar expression can be obtained for ϕ_W and is given by

$$\phi_W = \phi_w - \left(\frac{d\phi}{dx}\right)_w \frac{(\delta x)_w}{2} + \frac{1}{2}\left(\frac{d^2\phi}{dx^2}\right)_w \left(\frac{(\delta x)_w}{2}\right)^2 \cdots \tag{11.54}$$

Truncating second order terms and higher and substituting into the advection term, the left hand side of the discretized equation becomes

$$\begin{aligned}&(\rho u \Delta y)_e \phi_C - (\rho u \Delta y)_w \phi_W - \left[\left(\Gamma^\phi \frac{d\phi}{dx}\Delta y\right)_e - \left(\Gamma^\phi \frac{d\phi}{dx}\Delta y\right)_w\right] \\ &= (\rho u \Delta y)_e \left[\phi_e - \left(\frac{d\phi}{dx}\right)_e \frac{\delta x_e}{2}\right] - (\rho u \Delta y)_w \left[\phi_w - \left(\frac{d\phi}{dx}\right)_w \frac{\delta x_w}{2}\right] \\ &\quad - \left[\left(\Gamma^\phi \frac{d\phi}{dx}\Delta y\right)_e - \left(\Gamma^\phi \frac{d\phi}{dx}\Delta y\right)_w\right]\end{aligned} \tag{11.55}$$

which can be rearranged into

$$\begin{aligned}&(\rho u\Delta y)_e\phi_C-(\rho u\Delta y)_w\phi_W-\left[\left(\Gamma^\phi\frac{d\phi}{dx}\Delta y\right)_e-\left(\Gamma^\phi\frac{d\phi}{dx}\Delta y\right)_w\right]\\&=(\rho u\Delta y)_e\phi_e-(\rho u\Delta y)_w\phi_w\\&\quad-\left[\left(\Gamma^\phi+\rho u\frac{\delta x}{2}\right)_e\left(\frac{d\phi}{dx}\Delta y\right)_e-\left(\Gamma^\phi+\rho u\frac{\delta x}{2}\right)_w\left(\frac{d\phi}{dx}\Delta y\right)_w\right]\end{aligned}\tag{11.56}$$

It is clear now that the equation being solved has an added component of diffusion, which is called truncation error. The value of the numerical diffusion is equal to

$$\Gamma^\phi_{truncation}=\rho u\frac{\delta x}{2}\tag{11.57}$$

This truncation error, also known as stream wise diffusion, reduces the accuracy of the solution by altering the magnitude of the diffusion coefficient and consequently the equation to be solved. Thus the convection and diffusion equation has an effective modified value of the diffusion effects. On the other hand, this additional stream wise numerical diffusion is desirable as it stabilizes the solution by keeping it bounded and physically correct.

It is obvious that to reduce stream wise numerical diffusion a higher order approximation of the convection term is needed. However, as will be explained in a later section, this should be done in such a way that the solution remains bounded.

11.3.2 The Downwind Scheme

As with the upwind scheme, ϕ_C and ϕ_W are written as functions of ϕ_e and ϕ_w, respectively, for the case when the flow is assumed to be in the positive x direction. In this case the downwind scheme results in

$$\phi_e=\phi_E\quad\text{and}\quad\phi_w=\phi_C\tag{11.58}$$

and the discretized equation becomes

$$(\rho u\Delta y)_e\phi_E-(\rho u\Delta y)_w\phi_C-\left[\left(\Gamma^\phi\frac{d\phi}{dx}\Delta y\right)_e-\left(\Gamma^\phi\frac{d\phi}{dx}\Delta y\right)_w\right]=0\tag{11.59}$$

For a uniform grid, the one dimensional Taylor series expansions of ϕ_E and ϕ_C with respect to their values at cell faces e and w are given by

$$\begin{aligned}\phi_E &= \phi_e + \left(\frac{d\phi}{dx}\right)_e \frac{(\delta x)_e}{2} + \frac{1}{2}\left(\frac{d^2\phi}{dx^2}\right)_e \left(\frac{(\delta x)_e}{2}\right)^2 + \cdots \\ \phi_C &= \phi_w + \left(\frac{d\phi}{dx}\right)_w \frac{(\delta x)_w}{2} + \frac{1}{2}\left(\frac{d^2\phi}{dx^2}\right)_w \left(\frac{(\delta x)_w}{2}\right)^2 + \cdots\end{aligned} \tag{11.60}$$

Truncating second order terms and higher and substituting into the advection term, the left hand side of the discretized equation becomes

$$\begin{aligned}&(\rho u \Delta y)_e \phi_E - (\rho u \Delta y)_w \phi_C - \left[\left(\Gamma^\phi \frac{d\phi}{dx}\Delta y\right)_e - \left(\Gamma^\phi \frac{d\phi}{dx}\Delta y\right)_w\right] \\ &= (\rho u \Delta y)_e \phi_e - (\rho u \Delta y)_w \phi_w \\ &\quad - \left[\left(\Gamma^\phi - \rho u \frac{\delta x}{2}\right)_e \left(\frac{d\phi}{dx}\Delta y\right)_e - \left(\Gamma^\phi - \rho u \frac{\delta x}{2}\right)_w \left(\frac{d\phi}{dx}\Delta y\right)_w\right]\end{aligned} \tag{11.61}$$

For this profile, numerical diffusion has a negative sign and is equal to

$$\Gamma^\phi_{truncation} = -\rho u \frac{\delta x}{2} \tag{11.62}$$

which acts at decreasing the diffusion coefficient and in effect is an anti-diffusion error. Predictions using the downwind scheme are found to cause clipping of the advected profiles. In fact solutions to the one dimensional convection-diffusion problem generated using this scheme are more oscillatory than the CD scheme.

11.3.3 The Central Difference (CD) Scheme

The truncation error for the CD scheme is a little more involved to obtain with the finite volume method as computation of the gradient requires interpolated values at the element faces and not at nodes where they are available. Assuming the velocity is known everywhere and the grid is uniform with a size of Δx, the approximation is simply the one introduced in the calculation of $(\phi_e - \phi_w)$, which can be written as

$$\underbrace{(\phi_e - \phi_w)}_{Interpolated} = \frac{1}{2}(\phi_E + \phi_C) - \frac{1}{2}(\phi_C + \phi_W) = \underbrace{(\phi_e - \phi_w)}_{Exact} + TE \tag{11.63}$$

where *TE* refers to truncation error. To calculate this truncation error, the following Taylor series expansions with respect to the exact ϕ_e and ϕ_w values are needed:

$$\begin{aligned}
\phi_W &= \phi_w - \frac{\Delta x}{2}\phi'_w + \frac{\Delta x^2}{8}\phi''_w - \frac{\Delta x^3}{48}\phi'''_w + \frac{\Delta x^4}{384}\phi^{iv}_w - \frac{\Delta x^5}{3840}\phi^{v}_w + \cdots \\
\phi_C &= \phi_w + \frac{\Delta x}{2}\phi'_w + \frac{\Delta x^2}{8}\phi''_w + \frac{\Delta x^3}{48}\phi'''_w + \frac{\Delta x^4}{384}\phi^{iv}_w + \frac{\Delta x^5}{3840}\phi^{v}_w + \cdots \\
\phi_C &= \phi_e - \frac{\Delta x}{2}\phi'_e + \frac{\Delta x^2}{8}\phi''_e - \frac{\Delta x^3}{48}\phi'''_e + \frac{\Delta x^4}{384}\phi^{iv}_e - \frac{\Delta x^5}{3840}\phi^{v}_e + \cdots \\
\phi_E &= \phi_e + \frac{\Delta x}{2}\phi'_e + \frac{\Delta x^2}{8}\phi''_e + \frac{\Delta x^3}{48}\phi'''_e + \frac{\Delta x^4}{384}\phi^{iv}_e + \frac{\Delta x^5}{3840}\phi^{v}_e + \cdots
\end{aligned} \tag{11.64}$$

Using these expansions, the average values at the faces are obtained as

$$\begin{aligned}
\frac{1}{2}(\phi_E + \phi_C) &= \phi_e + \frac{\Delta x^2}{8}\phi''_e + \frac{\Delta x^4}{384}\phi^{iv}_e + \cdots \\
&\Rightarrow \frac{\Delta x^2}{4}\phi''_e = (\phi_C - 2\phi_e + \phi_E) - \frac{\Delta x^4}{192}\phi^{iv}_e + \cdots \\
\frac{1}{2}(\phi_C + \phi_W) &= \phi_w + \frac{\Delta x^2}{8}\phi''_w + \frac{\Delta x^4}{384}\phi^{iv}_w + \cdots \\
&\Rightarrow \frac{\Delta x^2}{4}\phi''_w = (\phi_W - 2\phi_w + \phi_C) - \frac{\Delta x^4}{192}\phi^{iv}_w + \cdots
\end{aligned} \tag{11.65}$$

Subtracting the above two terms given in Eq. (11.65), their difference is found to be

$$\begin{aligned}
\frac{1}{2}(\phi_E + \phi_C) - \frac{1}{2}(\phi_C + \phi_W) = (\phi_e - \phi_w) &+ \frac{\Delta x^2}{8}\left(\phi''_e - \phi''_w\right) \\
&+ \frac{\Delta x^4}{384}\left(\phi^{iv}_e - \phi^{iv}_w\right) + \cdots
\end{aligned} \tag{11.66}$$

Further, the expanded forms of the exact ϕ_e and ϕ_w with respect to ϕ_C are given by

$$\begin{aligned}
\phi_e &= \phi_C + \frac{\Delta x}{2}\phi'_C + \frac{\Delta x^2}{8}\phi''_C + \frac{\Delta x^3}{48}\phi'''_C + \frac{\Delta x^4}{384}\phi^{iv}_C + \frac{\Delta x^5}{3840}\phi^{v}_C + \cdots \\
\phi_w &= \phi_C - \frac{\Delta x}{2}\phi'_C + \frac{\Delta x^2}{8}\phi''_C - \frac{\Delta x^3}{48}\phi'''_C + \frac{\Delta x^4}{384}\phi^{iv}_C - \frac{\Delta x^5}{3840}\phi^{v}_C + \cdots
\end{aligned} \tag{11.67}$$

In addition, ϕ_E and ϕ_W can be expanded in terms of ϕ_C as

$$\begin{aligned}
\phi_E &= \phi_C + \Delta x\phi'_C + \frac{\Delta x^2}{2}\phi''_C + \frac{\Delta x^3}{6}\phi'''_C + \frac{\Delta x^4}{24}\phi^{iv}_C + \frac{\Delta x^5}{120}\phi^{v}_C + \cdots \\
\phi_W &= \phi_C - \Delta x\phi'_C + \frac{\Delta x^2}{2}\phi''_C - \frac{\Delta x^3}{6}\phi'''_C + \frac{\Delta x^4}{24}\phi^{iv}_C - \frac{\Delta x^5}{120}\phi^{v}_C + \cdots
\end{aligned} \tag{11.68}$$

Using the above two sets of equations, the following is obtained:

$$\begin{aligned}\phi_C - 2\phi_e + \phi_E &= \phi_C - 2\phi_C - \Delta x\phi'_C - \frac{\Delta x^2}{4}\phi''_C - \frac{\Delta x^3}{24}\phi'''_C - \frac{\Delta x^4}{192}\phi^{iv}_C - \frac{\Delta x^5}{1920}\phi^{v}_C + \cdots \\ &\quad + \phi_C + \Delta x\phi'_C + \frac{\Delta x^2}{2}\phi''_C + \frac{\Delta x^3}{6}\phi'''_C + \frac{\Delta x^4}{24}\phi^{iv}_C + \frac{\Delta x^5}{120}\phi^{v}_C + \cdots \\ &= \frac{\Delta x^2}{4}\phi''_C + \frac{\Delta x^3}{8}\phi'''_C + \frac{7\Delta x^4}{192}\phi^{iv}_C + \frac{15\Delta x^5}{1920}\phi^{v}_C + \cdots \\ \phi_W - 2\phi_w + \phi_C &= \frac{\Delta x^2}{4}\phi''_C - \frac{\Delta x^3}{8}\phi'''_C + \frac{7\Delta x^4}{192}\phi^{iv}_C - \frac{15\Delta x^5}{1920}\phi^{v}_C + \cdots\end{aligned} \tag{11.69}$$

Then $\left(\phi''_e - \phi''_w\right)$ is computed as

$$\frac{\Delta x^2}{8}\left(\phi''_e - \phi''_w\right) = \frac{\Delta x^3}{8}\phi'''_C + \frac{\Delta x^5}{128}\phi^{v}_C + \cdots \tag{11.70}$$

Substituting back, Eq. (11.66) is transformed to

$$\frac{1}{2}(\phi_E + \phi_C) - \frac{1}{2}(\phi_C + \phi_W) = \phi_e - \phi_w + \frac{\Delta x^3}{8}\phi'''_C + \frac{\Delta x^5}{128}\phi^{v}_C + \cdots \tag{11.71}$$

Dividing throughout by Δx, the truncation error associated with the calculation of the gradient is obtained as

$$TE = \frac{\Delta x^2}{8}\phi'''_C + \frac{\Delta x^4}{128}\phi^{v}_C + \cdots \tag{11.72}$$

which indicates that the method is second order accurate.

11.4 Numerical Stability

The confusion related to truncation error (first order for the physical solution and second order for the unphysical one) has led many workers to extrapolate that since central differencing of the diffusion term is so accurate, then central differencing of the convection term should also be similarly accurate. Of course, as seen in the previous sections, central differencing of the convection term can lead to unphysical solutions. The cause for this behavior is that the central difference scheme has no inherent *convective stability* when applied to derivatives of *odd* order like the convection term.

Leonard [9] advanced the *convective stability* concept as an explanation to oscillation in numerical solutions generated by applying the central difference scheme to convection dominated flows. Leonard used the general one dimensional unsteady convection-diffusion equation with constant velocity given by

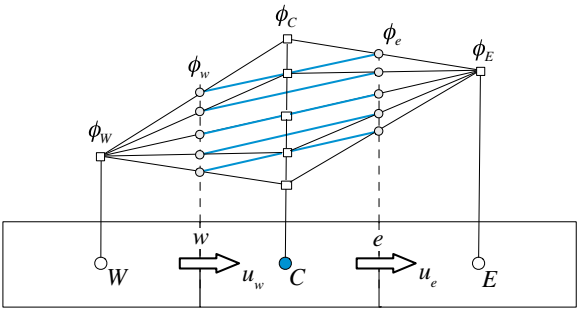

Fig. 11.9 Insensitivity of CD convection term to the values of $\boldsymbol{\phi}_C$

$$\frac{\partial(\rho\phi)}{\partial t} = -\frac{\partial(\rho u\phi)}{\partial x} + \frac{\partial}{\partial x}\left(\Gamma^{\phi}\frac{\partial\phi}{\partial x}\right) + Q^{\phi} \tag{11.73}$$

to describe this concept. When applying this equation over the element of centroid C shown in Fig. 11.9, its left hand side (*LHS*) represents the rate of change of ϕ_C within the control cell per unit time, and its right hand side (*RHS*) represents the net influx across the element surface and source terms within the element that are affecting the value of ϕ_C. If there were numerical errors in the *RHS* then the value of ϕ_C calculated from Eq. (11.73) would either increase or decrease depending on the scheme used in the discretization process. In an unstable scheme, a small deviation from the correct value of ϕ_C gives a corresponding increase/decrease in the net influx represented by the *RHS*. When an iterative procedure is used as part of the solution mechanism, an increase/decrease in the net influx will further increase/decrease the value of ϕ_C at each subsequent step of the iterative process. In a stable scheme this change in ϕ_C due to errors in the *RHS* should feed back negatively into the *RHS* as a self correction device. Clearly, for this kind of numerical stability, the *RHS* should satisfy

$$\frac{\partial(RHS)}{\partial\phi_C} < 0 \tag{11.74}$$

indicating that the sensitivity to ϕ_C of the combination of the modeled terms on the right hand side of Eq. (11.73) should be negative. In this case, an increase/decrease in ϕ_C will correspond to a decrease/increase in the influx, which will in turn pushes ϕ_C downward/upward towards its correct value. However, stability should not be confused with boundedness or accuracy. A stable numerical scheme could actually be unbounded giving rise to over/under shoots and oscillations/wiggles or be very diffusive and give results that are of low accuracy. The stability here refers to controlling the numerical error to remain bounded in order for it not to increase indefinitely as was the case with the central difference scheme where the normalized

value of ϕ_C (Fig. 11.4) was found to vary from $+\infty$ to $-\infty$ while varying Pe from $-\infty$ to $+\infty$, even though the correct value of ϕ_C should vary between 1 and 0. It so happens that the upwind scheme possesses both the boundedness and stability characteristics.

With this in mind, the general discretized form of the *RHS* of Eq. (11.73) is given by

$$RHS = -(\rho u \Delta y)_e \phi_e + (\rho u \Delta y)_w \phi_w + \left[\left(\Gamma^\phi \frac{\Delta y}{\delta x} \right)_e (\phi_E - \phi_C) - \left(\Gamma^\phi \frac{\Delta y}{\delta x} \right)_w (\phi_C - \phi_W) \right] + Q_C^\phi V_C \tag{11.75}$$

Thus using the central difference scheme, Eq. (11.75) becomes

$$RHS_{CD} = -(\rho u \Delta y)_e \frac{(\phi_E + \phi_C)}{2} + (\rho u \Delta y)_w \frac{(\phi_C + \phi_W)}{2} + \left[\left(\Gamma^\phi \frac{\Delta y}{\delta x} \right)_e (\phi_E - \phi_C) - \left(\Gamma^\phi \frac{\Delta y}{\delta x} \right)_w (\phi_C - \phi_W) \right] + Q_C^\phi V_C \tag{11.76}$$

Analyzing the central difference scheme using the above criteria, it is found that for the diffusion term the sensitivity is given by

$$\frac{\partial \left(RHS_{CD}^{Diff} \right)}{\partial \phi_C} = -2\Gamma^\phi \frac{\Delta y}{\delta x} \tag{11.77}$$

which is negative since Γ^ϕ is positive, indicating a stable scheme. However, for the convective term the sensitivity equation gives

$$\frac{\partial \left(RHS_{CD}^{Conv} \right)}{\partial \phi_C} = -\frac{1}{2}(\dot{m}_e + \dot{m}_w) \tag{11.78}$$

which is equal to zero for steady flows but not necessarily for unsteady flows. For unsteady situations, its value will be positive for decelerating flows. In a general flow, such regions act as wiggle sources and can easily lead to a total numerical catastrophe when the Péclet number is large enough. Even for steady flows, as its value is zero, it cannot feed back into the equation to act as a self correction device. Equation (11.78) also indicates that for steady flow the net convective flux computed with the CD scheme is independent of the value of ϕ_C. Therefore, as shown in Fig. 11.9, the different possible values of ϕ_C will result in the same net convective flux over the element of centroid C.

With the upwind scheme, the *RHS* can be obtained from Eq. (11.36) as

$$RHS_{Upwind} = -\|\dot{m}_e, 0\|\phi_C + \|-\dot{m}_e, 0\|\phi_E - \|\dot{m}_w, 0\|\phi_C + \|-\dot{m}_w, 0\|\phi_W + \left(\Gamma^\phi \frac{S}{\delta x}\right)_e (\phi_E - \phi_C) - \left(\Gamma^\phi \frac{S}{\delta x}\right)_w (\phi_C - \phi_W) + Q_C^\phi \tag{11.79}$$

The sensitivity is expected to be negative since the scheme was found to be stable for all Péclet number, indeed

$$\frac{\partial\left(RHS_{Upwind}^{Conv}\right)}{\partial\phi_C} = -\|\dot{m}_e, 0\| - \|\dot{m}_w, 0\| \tag{11.80}$$

which is negative or equal to zero for all flows. It will be equal to zero when both mass flow rates are negative, a situation that does not arise in a one dimensional situation of constant cross-sectional area. When added to the false diffusion introduced by the first order approximation, Eq. (11.80) indicates that the scheme is very stable. However, this stability is achieved at the expense of accuracy, as was demonstrated in the previous section.

For the downwind scheme the *RHS* is given by

$$RHS_{Downwind} = -\|\dot{m}_e, 0\|\phi_E + \|-\dot{m}_e, 0\|\phi_C - \|\dot{m}_w, 0\|\phi_W + \|-\dot{m}_w, 0\|\phi_C + \left(\Gamma^\phi \frac{S}{\delta x}\right)_e (\phi_E - \phi_C) + \left(\Gamma^\phi \frac{S}{\delta x}\right)_w (\phi_W - \phi_C) \tag{11.81}$$

The sensitivity is given by

$$\frac{\partial\left(RHS_{Downwind}^{Conv}\right)}{\partial\phi_C} = \|-\dot{m}_e, 0\| + \|-\dot{m}_w, 0\| \tag{11.82}$$

which is always positive or equal to zero for all flows. When this is added to the anti-diffusion effects it gives a highly unstable scheme.

11.5 Higher Order Upwind Schemes

The previous sections demonstrated that both the upwind and central difference schemes have severe limitations, the former because of its poor accuracy due to numerical diffusion, and the latter because of its instability also known as numerical dispersion error. These shortcomings have provoked a great deal of research to improve the accuracy and stability of advection schemes by using higher order

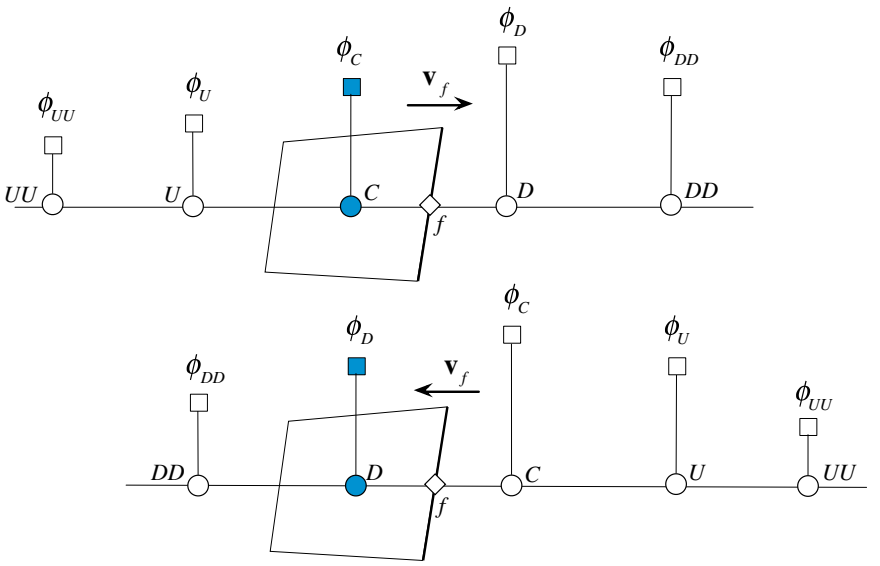

Fig. 11.10 A schematic showing the *UU*, *U*, *C*, *D* and *DD* node locations used in describing convection schemes

upwind biased interpolation profiles. These higher-order schemes aim at producing at least a second order accurate solutions, while being unconditionally stable.

To underline the conservation property which associates fluxes with cell faces, not nodes, in what follows *D*, *C*, and *U* will be used to denote the Downwind, Upwind, and far Upwind nodes at any particular face. In addition, *DD* and *UU* denote the nodes downstream and upstream of *D* and *U*, respectively. The corresponding values at these locations are denoted by $\phi_{DD}, \phi_D, \phi_C, \phi_U$ and ϕ_{UU}, respectively. This notation is displayed graphically in Fig. 11.10 for the cases when the velocity at the face is either positive ($\rightarrow$) or negative ($\leftarrow$).

11.5.1 Second Order Upwind Scheme

A second order scheme, requires the use of a linear profile, as was the case with the central difference scheme. However, instead of a symmetric profile, an upwind biased stencil is used [10]. As depicted in Fig. 11.11, the linear profile is constructed by employing the ϕ values at nodes *C* and *U*. Therefore the value at the face is actually calculated by extrapolation rather than interpolation.

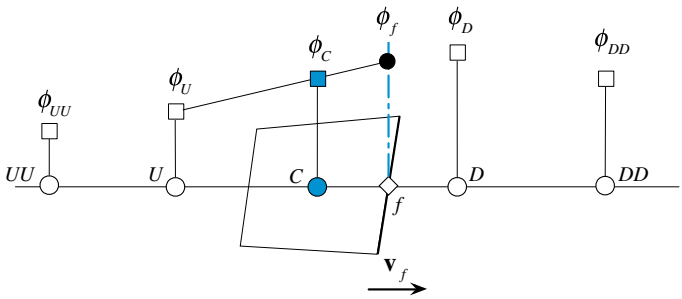

Fig. 11.11 The Second Order Upwind (SOU) scheme profile

11.5.2 The Interpolation Profile

Starting with the linear profile

$$\phi(x) = k_0 + k_1(x - x_C) \tag{11.83}$$

and fitting it to the nodal values at x_C and x_U where the ϕ values are ϕ_C and ϕ_U, respectively, the profile becomes

$$\phi(x) = \phi_C + \frac{\phi_C - \phi_U}{x_C - x_U}(x - x_C) \tag{11.84}$$

Using the above equation, the ϕ value at face f, shown in Fig. 11.11, is given by

$$\phi_f = \phi(x_f) = \phi_C + \frac{\phi_C - \phi_U}{x_C - x_U}(x_f - x_C) \tag{11.85}$$

which, for a uniform grid reduces to

$$\phi_f = \frac{3}{2}\phi_C - \frac{1}{2}\phi_U \tag{11.86}$$

11.5.3 The Discretized Equation

Using this profile to approximate the interface values in the discretized one dimensional convection diffusion equation (Eq. 11.16), the fluxes at the faces are obtained as

$$\begin{aligned}\dot{m}_e\phi_e &= \left(\frac{3}{2}\phi_C - \frac{1}{2}\phi_W\right)\|\dot{m}_e, 0\| - \left(\frac{3}{2}\phi_E - \frac{1}{2}\phi_{EE}\right)\|-\dot{m}_e, 0\| \\ \dot{m}_w\phi_w &= \left(\frac{3}{2}\phi_C - \frac{1}{2}\phi_E\right)\|\dot{m}_w, 0\| - \left(\frac{3}{2}\phi_W - \frac{1}{2}\phi_{WW}\right)\|-\dot{m}_w, 0\|\end{aligned} \tag{11.87}$$

Substitution of these fluxes in Eq. (11.16), the discretized form is transformed to

$$\begin{aligned}&\left(\frac{3}{2}\phi_C - \frac{1}{2}\phi_W\right)\|\dot{m}_e, 0\| - \left(\frac{3}{2}\phi_E - \frac{1}{2}\phi_{EE}\right)\|-\dot{m}_e, 0\| + \left(\frac{3}{2}\phi_C - \frac{1}{2}\phi_E\right)\|\dot{m}_w, 0\| \\ &- \left(\frac{3}{2}\phi_W - \frac{1}{2}\phi_{WW}\right)\|-\dot{m}_w, 0\| - \left[\left(\Gamma^\phi \frac{S}{\delta x}\right)_e (\phi_E - \phi_C) - \left(\Gamma^\phi \frac{S}{\delta x}\right)_w (\phi_C - \phi_W)\right] = 0\end{aligned} \tag{11.88}$$

which can be modified into the form

$$a_C\phi_C + a_E\phi_E + a_W\phi_W + a_{EE}\phi_{EE} + a_{WW}\phi_{WW} = 0 \tag{11.89}$$

where

$$\begin{aligned}a_E &= FluxF_e = -\Gamma_e^\phi \frac{S_e}{\delta x_e} - \frac{3}{2}\|-\dot{m}_e, 0\| - \frac{1}{2}\|\dot{m}_w, 0\| \quad a_{EE} = FluxF_{ee} = \frac{1}{2}\|-\dot{m}_e, 0\| \\ a_W &= FluxF_w = -\Gamma_w^\phi \frac{S_w}{\delta x_w} - \frac{3}{2}\|-\dot{m}_w, 0\| - \frac{1}{2}\|\dot{m}_e, 0\| \quad a_{WW} = FluxF_{ww} = \frac{1}{2}\|-\dot{m}_w, 0\| \\ a_C &= \sum_{f\sim nb(C)} FluxC_f \\ &= \Gamma_e^\phi \frac{S_e}{\delta x_e} + \Gamma_w^\phi \frac{S_w}{\delta x_w} + \frac{3}{2}\|\dot{m}_e, 0\| + \frac{3}{2}\|\dot{m}_w, 0\| \\ &= -(a_E + a_W + a_{EE} + a_{WW}) + (\dot{m}_e + \dot{m}_w)\end{aligned} \tag{11.90}$$

11.5.4 Truncation Error

Following a procedure similar to the one used with the central difference scheme, the truncation error is found to be

$$TE = -\frac{3}{8}\Delta x^2\phi_C''' - \frac{1}{4}\Delta x^3\phi_C^{iv} + \cdots \tag{11.91}$$

indicating second order accuracy.

11.5.5 Stability Analysis

To check the stability of the SOU scheme, the discretized convective fluxes are substituted in Eq. (11.75) resulting in the following $RHS_{convection}$ term:

$$RHS_{convection} = -\left(\frac{3}{2}\phi_C - \frac{1}{2}\phi_W\right)\|\dot{m}_e, 0\| + \left(\frac{3}{2}\phi_E - \frac{1}{2}\phi_{EE}\right)\|-\dot{m}_e, 0\| - \left(\frac{3}{2}\phi_C - \frac{1}{2}\phi_E\right)\|\dot{m}_w, 0\| + \left(\frac{3}{2}\phi_W - \frac{1}{2}\phi_{WW}\right)\|-\dot{m}_w, 0\| \tag{11.92}$$

The rate of change of this term with respect to ϕ_C is found as

$$\frac{\partial(RHS_{convection})}{\partial\phi_C} = -\frac{3}{2}\|\dot{m}_e, 0\| - \frac{3}{2}\|\dot{m}_w, 0\| \tag{11.93}$$

which is always negative indicating a stable scheme, i.e., the solution error will always remain under control. It should be kept in mind that this is true for the conditions stated in the derivations and not for the general case when the velocity is not constant.

11.5.6 The QUICK Scheme

The Quadratic Upstream Interpolation for Convective Kinematics (QUICK) scheme was developed by Leonard [2]. The method is based on interpolating the value of the dependent variable at each face of the element by using a quadratic polynomial biased toward the upstream direction, as shown in Fig. 11.12. The interpolated value is used to calculate the convective term in the governing equations for the dependent variable. The calculation of the values of the dependent variable at a cell face is detailed next.

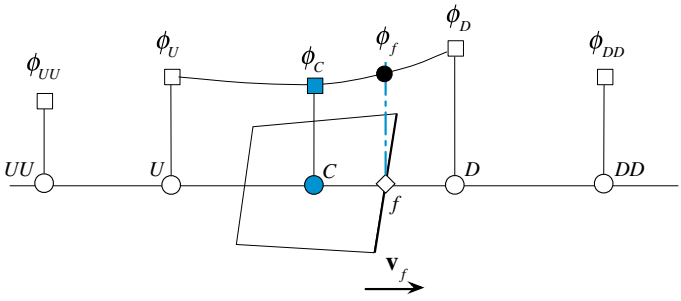

Fig. 11.12 The QUICK scheme profile

11.5.7 The Interpolation Profile

In its original form, the QUICK scheme required the use of a general multi dimensional second degree polynomial for the calculation of the unknown variable ϕ. However it is generally sufficient to treat the flow as locally one dimensional [11] and to use a second order one dimensional profile in each coordinate direction. For the one dimensional convection-diffusion problem under discussion, the value of ϕ is computed using

$$\phi = k_0 + k_1 x + k_2 x^2, \tag{11.94}$$

subject to

$$\phi = \begin{cases} \phi_U & at\ x = x_U \\ \phi_C & at\ x = x_C \\ \phi_D & at\ x = x_D \end{cases} \tag{11.95}$$

Applying the conditions given by Eq. (11.95), the profile for ϕ is found to be

$$\phi = \phi_U + \frac{(x - x_U)(x - x_C)}{(x_D - x_U)(x_D - x_C)}(\phi_D - \phi_U) + \frac{(x - x_U)(x - x_D)}{(x_C - x_U)(x_C - x_D)}(\phi_C - \phi_U) \tag{11.96}$$

For the case of a uniform grid, the value of ϕ at the cell face f reduces to

$$\phi_f = \frac{\phi_C + \phi_D}{2} - \frac{\phi_D - 2\phi_C + \phi_U}{8} \tag{11.97}$$

Thus, the convective fluxes at the element faces can be computed as

$$\begin{aligned} \dot{m}_e\phi_e &= \left(\frac{3}{4}\phi_C - \frac{1}{8}\phi_W + \frac{3}{8}\phi_E\right)\|\dot{m}_e, 0\| - \left(\frac{3}{4}\phi_E - \frac{1}{8}\phi_{EE} + \frac{3}{8}\phi_C\right)\|-\dot{m}_e, 0\| \\ \dot{m}_w\phi_w &= \left(\frac{3}{4}\phi_C - \frac{1}{8}\phi_E + \frac{3}{8}\phi_W\right)\|\dot{m}_w, 0\| - \left(\frac{3}{4}\phi_W - \frac{1}{8}\phi_{WW} + \frac{3}{8}\phi_C\right)\|-\dot{m}_w, 0\| \end{aligned} \tag{11.98}$$

Substituting back in the one dimensional convection-diffusion equation [Eq. (11.16)], the discretized form becomes

$$\begin{aligned} &\left(\frac{3}{4}\phi_C - \frac{1}{8}\phi_W + \frac{3}{8}\phi_E\right)\|\dot{m}_e, 0\| - \left(\frac{3}{4}\phi_E - \frac{1}{8}\phi_{EE} + \frac{3}{8}\phi_C\right)\|-\dot{m}_e, 0\| \\ &+ \left(\frac{3}{4}\phi_C - \frac{1}{8}\phi_E + \frac{3}{8}\phi_W\right)\|\dot{m}_w, 0\| - \left(\frac{3}{4}\phi_W - \frac{1}{8}\phi_{WW} + \frac{3}{8}\phi_C\right)\|-\dot{m}_w, 0\| \\ &- \left[\left(\Gamma^\phi \frac{S}{\delta x}\right)_e (\phi_E - \phi_C) - \left(\Gamma^\phi \frac{S}{\delta x}\right)_w (\phi_C - \phi_W)\right] = 0 \end{aligned} \tag{11.99}$$

which can be modified into the form

$$a_C\phi_C + a_E\phi_E + a_W\phi_W + a_{EE}\phi_{EE} + a_{WW}\phi_{WW} = 0 \tag{11.100}$$

with the coefficients given by

$$\begin{aligned}
a_E &= FluxF_e = -\Gamma_e^\phi \frac{S_e}{\delta x_e} - \frac{3}{4}\|-\dot{m}_e, 0\| + \frac{3}{8}\|\dot{m}_e, 0\| - \frac{1}{8}\|\dot{m}_w, 0\| \\
a_W &= FluxF_w = -\Gamma_w^\phi \frac{S_w}{\delta x_w} - \frac{3}{4}\|-\dot{m}_w, 0\| + \frac{3}{8}\|\dot{m}_w, 0\| - \frac{1}{8}\|\dot{m}_e, 0\| \\
a_{EE} &= FluxF_{ee} = \frac{1}{8}\|-\dot{m}_e, 0\| \quad a_{WW} = FluxF_{ww} = \frac{1}{8}\|-\dot{m}_w, 0\| \\
a_C &= \sum_{f\sim nb(C)} FluxC_f \\
&= \Gamma_e^\phi \frac{S_e}{\delta x_e} + \Gamma_w^\phi \frac{S_w}{\delta x_w} + \frac{3}{4}\|\dot{m}_e, 0\| - \frac{3}{8}\|-\dot{m}_e, 0\| + \frac{3}{4}\|\dot{m}_w, 0\| - \frac{3}{8}\|-\dot{m}_w, 0\| \\
&= -\left(a_E + a_W + a_{WW} + a_{EE}\right) + \left(\dot{m}_e + \dot{m}_w\right)
\end{aligned} \tag{11.101}$$

11.5.8 Truncation Error

Again following the procedure described above, the truncation error is found to be

$$TE = \frac{1}{16}\Delta x^3 \phi_C^{iv} - \frac{3}{128}\Delta x^4 \phi_C^{v} + \cdots \tag{11.102}$$

which is clearly third order accurate.

11.5.9 Stability Analysis

To check the stability of the QUICK scheme, the discretized convective fluxes are substituted in Eq. (11.75) resulting in the following $RHS_{convection}$ term:

$$\begin{aligned}
RHS_{convection} = &-\left(\frac{3}{4}\phi_C - \frac{1}{8}\phi_W + \frac{3}{8}\phi_E\right)\|\dot{m}_e, 0\| + \left(\frac{3}{4}\phi_E - \frac{1}{8}\phi_{EE} + \frac{3}{8}\phi_C\right)\|-\dot{m}_e, 0\| \\
&-\left(\frac{3}{4}\phi_C - \frac{1}{8}\phi_E + \frac{3}{8}\phi_W\right)\|\dot{m}_w, 0\| + \left(\frac{3}{4}\phi_W - \frac{1}{8}\phi_{WW} + \frac{3}{8}\phi_C\right)\|-\dot{m}_w, 0\|
\end{aligned} \tag{11.103}$$

The rate of change of this term with respect to ϕ_C is found to be given by

$$\frac{\partial(RHS_{convection})}{\partial\phi_C} = -\frac{3}{8}\|\dot{m}_e, 0\| - \frac{3}{8}\|\dot{m}_w, 0\| - \frac{3}{8}(\dot{m}_e + \dot{m}_w) \tag{11.104}$$

which for a uniform velocity is always negative indicating a stable scheme. However this does not guarantee solution boundedness especially in the general case of nonuniform velocity.

11.5.10 The FROMM Scheme

The FROMM scheme [12] fits a linear profile between the far Upwind (U) and Downwind (D) nodes straddling the interface. As depicted in Fig. 11.13, instead of a symmetric profile, an upwind biased stencil is used to calculate the value of the dependent variable ϕ at face f. Based on the adopted profile, ϕ_U, ϕ_C, and ϕ_D are assumed to be collinear.

11.5.11 The Interpolation Profile

Starting with the linear profile

$$\phi(x) = k_0 + k_1(x - x_C) \tag{11.105}$$

and fitting it to the nodal values at x_D and x_U where the ϕ values are ϕ_D and ϕ_U, respectively, the profile becomes

$$\phi(x) = \phi_U + \frac{\phi_D - \phi_U}{x_D - x_U}(x - x_U) \tag{11.106}$$

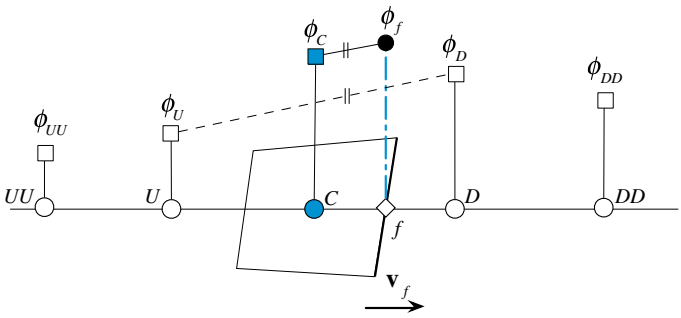

Fig. 11.13 The FROMM scheme profile

Using the above equation, the ϕ value at upwind node C is obtained as

$$\phi_C = \phi_U + \frac{\phi_D - \phi_U}{x_D - x_U}(x_C - x_U) \tag{11.107}$$

For the case of a uniform grid the above equation becomes

$$\phi_C = \frac{\phi_D + \phi_U}{2} \tag{11.108}$$

The required value at the face f, shown in Fig. 11.13, is given by

$$\phi_f = \phi(x_f) = \phi_U + \frac{\phi_D - \phi_U}{x_D - x_U}(x_f - x_U) = \phi_C + \frac{x_f - x_C}{x_D - x_U}(\phi_D - \phi_U) \tag{11.109}$$

which, for a uniform grid reduces to

$$\phi_f = \phi_C + \frac{\phi_D - \phi_U}{4} \tag{11.110}$$

The final expression for ϕ_f given in Eq. (11.109) was obtained by invoking Eq. (11.107).

11.5.12 The Discretized Equation

Using this profile to approximate the interface values in the discretized one dimensional convection diffusion equation (Eq. 11.16), the fluxes at the faces are obtained as

$$\begin{aligned}
\dot{m}_e\phi_e &= \left(\phi_C - \frac{1}{4}\phi_w + \frac{1}{4}\phi_E\right)\|\dot{m}_e, 0\| - \left(\phi_E - \frac{1}{4}\phi_{EE} + \frac{1}{4}\phi_C\right)\|-\dot{m}_e, 0\| \\
\dot{m}_w\phi_w &= \left(\phi_C - \frac{1}{4}\phi_E + \frac{1}{4}\phi_W\right)\|\dot{m}_w, 0\| - \left(\phi_W - \frac{1}{4}\phi_{WW} + \frac{1}{4}\phi_C\right)\|-\dot{m}_w, 0\|
\end{aligned} \tag{11.111}$$

Substitution of these fluxes in Eq. (11.16), the discretized form is transformed to

$$\begin{aligned}
&\left(\phi_C - \frac{1}{4}\phi_W + \frac{1}{4}\phi_E\right)\|\dot{m}_e, 0\| - \left(\phi_E - \frac{1}{4}\phi_{EE} + \frac{1}{4}\phi_C\right)\|-\dot{m}_e, 0\| \\
&+ \left(\phi_C - \frac{1}{4}\phi_E + \frac{1}{4}\phi_W\right)\|\dot{m}_w, 0\| - \left(\phi_W - \frac{1}{4}\phi_{WW} + \frac{1}{4}\phi_C\right)\|-\dot{m}_w, 0\| \\
&- \left[\left(\Gamma^\phi \frac{S}{\delta x}\right)_e (\phi_E - \phi_C) - \left(\Gamma^\phi \frac{S}{\delta x}\right)_w (\phi_C - \phi_W)\right] = 0
\end{aligned} \tag{11.112}$$

which can be modified into the form

$$a_C\phi_C + a_E\phi_E + a_W\phi_W + a_{EE}\phi_{EE} + a_{WW}\phi_{WW} = 0 \tag{11.113}$$

where

$$\begin{aligned}
a_E &= FluxF_e = -\Gamma_e^\phi \frac{S_e}{\delta x_e} + \frac{1}{4}\|\dot{m}_e, 0\| - \|-\dot{m}_e, 0\| - \frac{1}{4}\|\dot{m}_w, 0\| \\
a_W &= FluxF_w = -\Gamma_w^\phi \frac{S_w}{\delta x_w} + \frac{1}{4}\|\dot{m}_w, 0\| - \|-\dot{m}_w, 0\| - \frac{1}{4}\|\dot{m}_e, 0\| \\
a_{EE} &= FluxF_{ee} = \frac{1}{4}\|-\dot{m}_e, 0\| \quad a_{WW} = FluxF_{ww} = \frac{1}{4}\|-\dot{m}_w, 0\| \\
a_C &= \sum_{f\sim nb(C)} FluxC_f \\
&= \Gamma_e^\phi \frac{S_e}{\delta x_e} + \Gamma_w^\phi \frac{S_w}{\delta x_w} + \|\dot{m}_e, 0\| - \frac{1}{4}\|-\dot{m}_e, 0\| + \|\dot{m}_w, 0\| - \frac{1}{4}\|-\dot{m}_w, 0\| \\
&= -(a_E + a_W + a_{EE} + a_{WW}) + (\dot{m}_e + \dot{m}_w)
\end{aligned} \tag{11.114}$$

11.5.13 Truncation Error

Following a procedure similar to the one used with the central difference scheme, the truncation error can be derived to be of $O(\Delta x^2)$, i.e.,

$$TE = O\left(\Delta x^2\right) \tag{11.115}$$

indicating second order accuracy.

11.5.14 Stability Analysis

To check the stability of the FROMM scheme, the discretized convective fluxes are substituted in Eq. (11.75) resulting in the following $RHS_{convection}$ term:

$$\begin{aligned}
RHS_{convection} = &-\left(\phi_C - \frac{1}{4}\phi_W + \frac{1}{4}\phi_E\right)\|\dot{m}_e, 0\| + \left(\phi_E - \frac{1}{4}\phi_{EE} + \frac{1}{4}\phi_C\right)\|-\dot{m}_e, 0\| \\
&- \left(\phi_C - \frac{1}{4}\phi_E + \frac{1}{4}\phi_W\right)\|\dot{m}_w, 0\| + \left(\phi_W - \frac{1}{4}\phi_{WW} + \frac{1}{4}\phi_C\right)\|-\dot{m}_w, 0\|
\end{aligned} \tag{11.116}$$

The rate of change of this term with respect to ϕ_C is found as

$$\frac{\partial(RHS_{convection})}{\partial\phi_C} = -\frac{3}{4}\|\dot{m}_e, 0\| - \frac{3}{4}\|\dot{m}_w, 0\| - \frac{1}{4}(\dot{m}_e + \dot{m}_w) \tag{11.117}$$

which, for a constant velocity field, is always negative indicating a stable scheme, but not for the general case when the velocity is varying.

11.5.15 Comparison of the Various Schemes

By comparing the stability of the various schemes presented so far, it is clear that the most negative (with a coefficient of −3/2) is the one associated with the SOU scheme. This is followed by the upwind (with a coefficient of −1), then FROMM (with a coefficient of −3/4), after that QUICK(with a coefficient of −3/8), and finally the central difference scheme (with a coefficient of zero, i.e., a neutral sensitivity to changes in ϕ_C). The self corrective action discussed above is the sum of both convection and diffusion contributions. The false diffusion produced by the upwind scheme adds to its stability and even though its coefficient is −1, is the most stable scheme. This is demonstrated by the profiles presented in Fig. 11.14, which represent the solutions using the various numerical schemes of the convection-diffusion equation over a domain of length L = 1 and subject to the Dirichlet conditions of $\phi = 1$ at x = 0 and $\phi = 0$ at x = 1. While all solutions are stable at low Péclet number (i.e., Pe = 1, Fig. 11.14a), the CD, FROMM, and QUICK scheme solutions shown in Fig. 11.14b are seen to be wiggly at Pe = 10. At low Pe, the accuracy of the CD and QUICK schemes is comparable and their solutions are very close to the exact solution. The least accurate is the solution produced by the first order upwind scheme. The solution of the FROMM scheme is

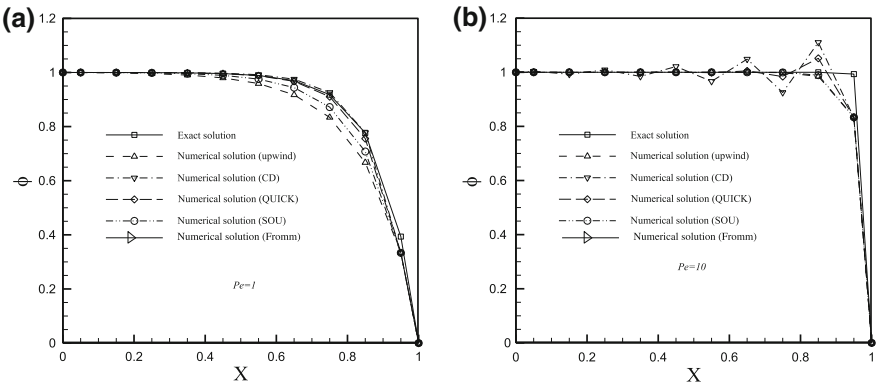

Fig. 11.14 Stability of solutions generated using several convective schemes at two values of cell Péclet number of **a** 1 and **b** 10

more accurate than the SOU scheme, which, in turn, is more accurate than the upwind scheme solution, while both the FROMM and SOU scheme solutions are less accurate than the QUICK scheme solution.

At high *Pe*, the behavior of the schemes changes. The only stable solutions are the ones obtained by the upwind and SOU schemes with their profiles being almost identical indicating that the SOU scheme is still highly diffusive. The CD scheme is wiggly over most of the domain. The FROMM and QUICK schemes, on the other hand, show wiggles but of smaller amplitudes. The reason for these wiggles is the imposed boundary condition at exit from the domain. As the phenomenon is convection dominated, it is affected by upstream values. At exit from the domain, the solution faces an imposed value that it has to satisfy, resulting in a large unexpected gradient causing the over and under shoots to occur. The diffusive upwind and SOU schemes being based on upstream values only, are smooth and do not show any sign of under/overshoots for all *Pe* values as the solution is independent of the imposed value at exit. However the SOU scheme is expected to give rise to oscillations (over/under shoots) in the presence of a high gradient in the domain like in the presence of a shock wave.

11.5.16 Functional Relationships for Uniform and Non-uniform Grids

The various interpolation profiles presented so far have been derived on a one dimensional Cartesian grid. Along a curvilinear coordinate axis, the same functional relationships can be used by replacing the Cartesian x-axis by a curvilinear ζ-axis, as shown in Fig. 11.15. For uniform grid, the functional relationships remain exactly the same, independent of whether a Cartesian or a curvilinear grid system is used. For non-uniform grid, the independent variable x appearing in the functional relationships should be replaced by the more general independent variable ζ, which represents distance along the coordinate axis. If O is the origin (Fig. 11.15), then ζ_U for example is calculated as

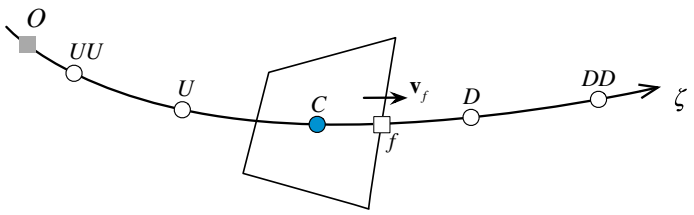

Fig. 11.15 Notation on a curvilinear coordinate axis ζ

Table 11.1 Functional relationships for several convection schemes on uniform and non-uniform grids

Scheme	Uniform grid	Non-uniform grid
Upwind	$\phi_f = \phi_C$	$\phi_f = \phi_C$
Downwind	$\phi_f = \phi_D$	$\phi_f = \phi_D$
CD	$\phi_f = 0.5(\phi_C + \phi_D)$	$\phi_f = \phi_C + \left(\frac{\phi_D - \phi_C}{\zeta_D - \zeta_C}\right)(\zeta_f - \zeta_C)$
SOU	$\phi_f = \frac{3}{2}\phi_C - \frac{1}{2}\phi_U$	$\phi_f = \phi_C + \left(\frac{\phi_C - \phi_U}{\zeta_C - \zeta_U}\right)(\zeta_f - \zeta_C)$
QUICK	$\phi_f = \frac{3}{4}\phi_C - \frac{1}{8}\phi_U + \frac{3}{8}\phi_D$	$\phi_f = \phi_U + \frac{(\zeta_f - \zeta_U)(\zeta_f - \zeta_C)}{(\zeta_D - \zeta_U)(\zeta_D - \zeta_C)}(\phi_D - \phi_U) + \frac{(\zeta_f - \zeta_U)(\zeta_f - \zeta_D)}{(\zeta_C - \zeta_U)(\zeta_C - \zeta_D)}(\phi_C - \phi_U)$
FROMM	$\phi_f = \phi_C + \frac{1}{4}(\phi_D - \phi_U)$	$\phi_f = \phi_C + \frac{\zeta_f - \zeta_C}{\zeta_D - \zeta_U}(\phi_D - \phi_U)$

$$\begin{aligned} \zeta_U &= \zeta_{UU} + (\zeta_U - \zeta_{UU}) \\ \zeta_{UU} &= \sqrt{(x_{UU} - x_O)^2 + (y_{UU} - y_O)^2 + (z_{UU} - z_O)^2} \\ \zeta_U - \zeta_{UU} &= \sqrt{(x_U - x_{UU})^2 + (y_U - y_{UU})^2 + (z_U - z_{UU})^2} \end{aligned} \tag{11.118}$$

Therefore distances are calculated by subdividing the curvilinear line into a number of straight lines and in general the following applies:

$$\zeta_1 - \zeta_2 = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2 + (z_1 - z_2)^2} \tag{11.119}$$

Adopting a general coordinate system, the functional relationships for the various schemes presented so far on uniform and non-uniform grids can be derived to be as shown in Table 11.1.

11.6 Steady Two Dimensional Advection

The steady two dimensional advection equation is given by

$$\nabla \cdot (\rho \mathbf{v} \phi) = 0. \tag{11.120}$$

Integrating over the two-dimensional element of volume V_C shown in Fig. 11.16, using the divergence theorem, and replacing the surface integral by a summation over the element faces, Eq. (11.120) becomes

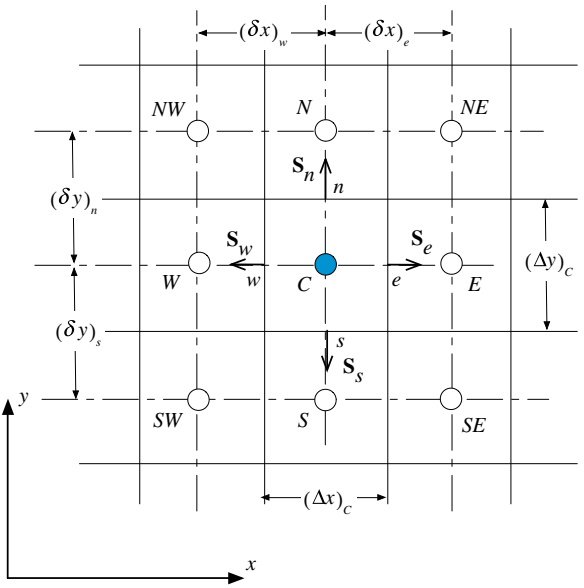

Fig. 11.16 Notation for a two dimensional Cartesian grid system

$$\sum_{f\sim nb(C)} \left(\int_f \mathbf{J}^{\phi,C} \cdot d\mathbf{S} \right) = 0 \tag{11.121}$$

Using a single Gaussian point for the face integral, the left hand side of Eq. (11.121) is transformed to

$$\sum_{f\sim nb(C)} \left(\int_f \mathbf{J}^{\phi,C} \cdot d\mathbf{S} \right) = \sum_{f\sim nb(C)} \left(\mathbf{J}_f^{\phi,C} \cdot \mathbf{S}_f \right) = \sum_{f\sim nb(C)} (\rho \mathbf{v}\phi)_f \cdot \mathbf{S}_f \tag{11.122}$$

Substitution of Eq. (11.122) in Eq. (11.121) yields

$$\sum_{f\sim nb(C)} (\rho \mathbf{v}\phi)_f \cdot \mathbf{S}_f = 0 \tag{11.123}$$

The full discretized form of Eq. (11.120) over a Cartesian grid is obtained as

$$\left(\overbrace{\rho u \Delta y \phi}^{\dot{m}_e} \right)_e - \left(\overbrace{\rho u \Delta y \phi}^{-\dot{m}_w} \right)_w + \left(\overbrace{\rho v \Delta x \phi}^{\dot{m}_n} \right)_n - \left(\overbrace{\rho v \Delta x \phi}^{-\dot{m}_s} \right)_s = 0 \tag{11.124}$$

Adopting an upwind scheme in each coordinate direction by treating the flow as locally one dimensional, Eq. (11.124) becomes

$$a_C\phi_C + a_E\phi_E + a_W\phi_W + a_N\phi_N + a_S\phi_S = 0 \tag{11.125}$$

with the coefficients given by

$$\begin{aligned}
a_E &= FLuxF_e = -\|-\dot{m}_e, 0\| \qquad a_W = FLuxF_w = -\|-\dot{m}_w, 0\| \\
a_N &= FLuxF_n = -\|-\dot{m}_n, 0\| \qquad a_S = FLuxF_s = -\|-\dot{m}_s, 0\| \\
a_C &= \sum_{f\sim nb(C)} FluxC_f = \|\dot{m}_e, 0\| + \|\dot{m}_w, 0\| + \|\dot{m}_n, 0\| + \|\dot{m}_n, 0\| \\
&= -\underbrace{(a_E + a_W + a_N + a_S)}_{\sum_{F\sim NB(C)} a_F} + \dot{m}_e + \dot{m}_w + \dot{m}_n + \dot{m}_s
\end{aligned} \tag{11.126}$$

If the QUICK scheme is used, the discretized equation is changed into

$$\begin{aligned}
a_C\phi_C + a_E\phi_E + a_W\phi_W + a_{EE}\phi_{EE} + a_{WW}\phi_{WW} \\
+ a_N\phi_N + a_S\phi_S + a_{NN}\phi_{NN} + a_{SS}\phi_{SS} = 0
\end{aligned} \tag{11.127}$$

with its coefficients computed as

$$\begin{aligned}
a_E &= FluxF_e = -\frac{3}{4}\|-\dot{m}_e, 0\| + \frac{3}{8}\|\dot{m}_e, 0\| - \frac{1}{8}\|\dot{m}_w, 0\| \\
a_W &= FluxF_w = -\frac{3}{4}\|-\dot{m}_w, 0\| + \frac{3}{8}\|\dot{m}_w, 0\| - \frac{1}{8}\|\dot{m}_e, 0\| \\
a_{EE} &= FluxF_{ee} = \frac{1}{8}\|-\dot{m}_e, 0\| \qquad a_{WW} = FluxF_{ww} = \frac{1}{8}\|-\dot{m}_w, 0\| \\
a_N &= FluxF_n = -\frac{3}{4}\|-\dot{m}_n, 0\| + \frac{3}{8}\|\dot{m}_n, 0\| - \frac{1}{8}\|\dot{m}_s, 0\| \\
a_S &= FluxF_s = -\frac{3}{4}\|-\dot{m}_s, 0\| + \frac{3}{8}\|\dot{m}_s, 0\| - \frac{1}{8}\|\dot{m}_n, 0\| \\
a_{NN} &= FluxF_{nn} = \frac{1}{8}\|-\dot{m}_n, 0\| \qquad a_{SS} = FluxF_{ss} = \frac{1}{8}\|-\dot{m}_s, 0\| \\
a_C &= \sum_{f\sim nb(C)} FluxC_f = -\sum_{F\sim NB(C)} a_F + (\dot{m}_e + \dot{m}_w + \dot{m}_n + \dot{m}_s)
\end{aligned} \tag{11.128}$$

Both discretized equations are used to solve the pure advection of a step profile in an oblique velocity field problem. The physical domain is schematically depicted in Fig. 11.17a. It represents a square domain with a property ϕ being convected in a field with a velocity $\mathbf{v}(1,1)$. The value of ϕ is 1 on the left side of the domain and 0

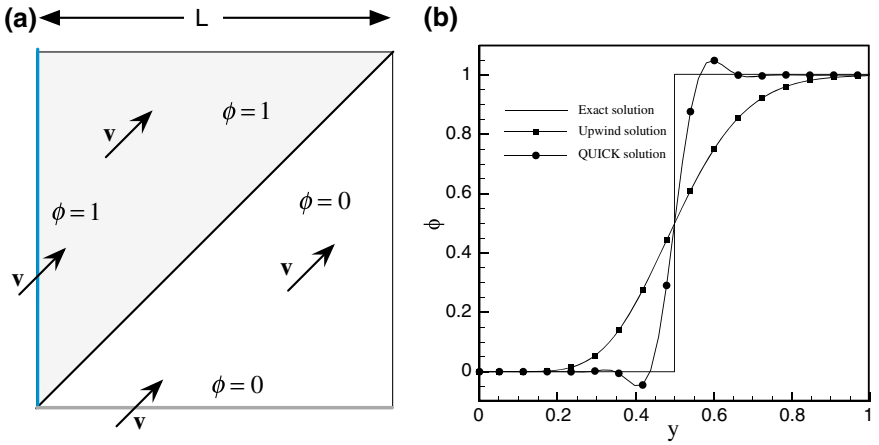

Fig. 11.17 **a** Physical domain and **b** ϕ profiles along the vertical centerline of the domain for pure advection of a step profile in an oblique velocity field

on the bottom. In the absence of any diffusion, the exact solution is $\phi = 1$ above the diagonal shown in Fig. 11.17a and $\phi = 0$ below it. Figure 11.17b compares the exact profile at $x = 0.5$ with similar ones obtained numerically using the upwind and QUICK schemes.

In comparison with the exact solution, the profile generated by the upwind scheme is smeared and highly inaccurate but very smooth. This inaccuracy is due to a new type of error known as cross-stream diffusion, which is caused by the one-dimensional interpolation profile used, i.e., it is due to treating the flow as locally one dimensional. The origin of cross-flow diffusion was identified by Patankar [4] and Stubley [13] as being a ***multi-dimensional*** phenomenon. It occurs only when the velocity field is not aligned with the grid. An approximate expression for the cross flow diffusion has been given by de Vahl Davis and Mallinson [14] for two dimensional flows as

$$\Gamma^{\phi}_{false} = \frac{\rho|\mathbf{v}|\Delta x \Delta y \sin(2\theta)}{4\left(\Delta y \sin^3(\theta) + \Delta x \cos^3(\theta)\right)} \tag{11.129}$$

where $|\mathbf{v}|$ is the velocity magnitude and θ the angle made by the velocity vector with the x coordinate axis. This error can be reduced by using higher order interpolation schemes as demonstrated by the profile generated with the QUICK scheme. The QUICK scheme profile is shown to be much sharper and more accurate than the upwind profile however it is infected with over/undershoots near the sharp gradient. As mentioned earlier this error is called the dispersion error, which causes the generation of maxima/minima in the solution domain and is a characteristic of all High Order (HO) schemes.

11.6.1 Error Sources

Based on the discussions in the previous sections the sources of numerical error in the discretization of the convective flux can be divided into numerical diffusion and numerical dispersion.

Numerical diffusion, which causes smearing of sharp gradients (Fig. 11.18a) can also be divided into stream wise and cross stream numerical diffusion. Stream wise numerical diffusion can be reduced by increasing the order of the interpolation profile, as shown in Fig. 11.18b, resulting in sharper profiles but inducing under/overshoots in the presence of large gradients. Cross-stream numerical diffusion, shown in Fig. 11.17b, is caused by the one dimensional nature of the assumed profile and can be reduced either by interpolating in the direction of the flow (i.e., multi-dimensional profiles) [15, 16] or by using one-dimensional higher order interpolation profiles (Fig. 11.18b).

Numerical dispersion error manifests itself through oscillation in the generated profile in the presence of large gradients rendering the solution unbounded. As shown in Figs. 11.18b and 11.19 and the results reported in Fig. 11.14b, it exists with all assumed interpolation profiles except the upwind scheme. It is the result of the unphysical behavior of the assumed interpolation profile.

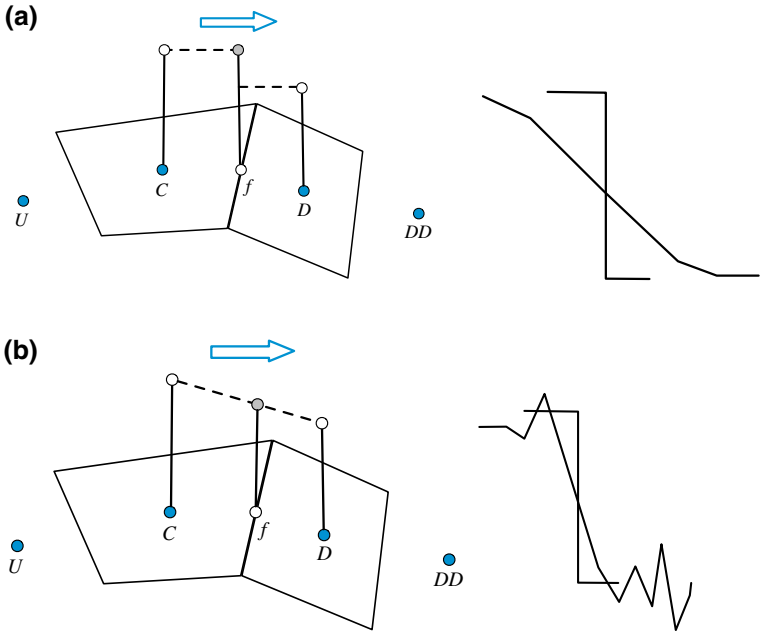

Fig. 11.18 **a** Smearing of sharp profiles by numerical diffusion and **b** wiggles in the computed profile due to numerical dispersion

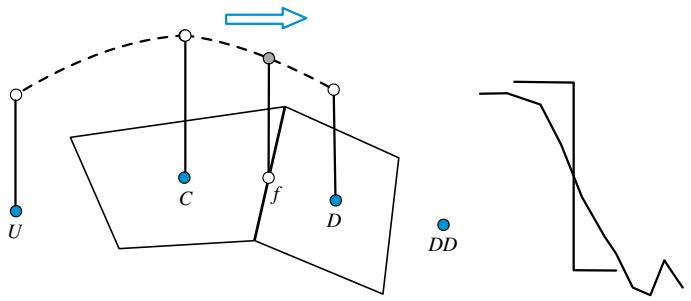

Fig. 11.19 Numerical dispersion error causing oscillations in the presence of a large gradient

An evaluation of this error can be obtained by using a simplified version of Eq. (11.73) in which diffusion and sources are neglecting. If further, velocity and density fields are considered constants, Eq. (11.73) simplifies to

$$\frac{\partial \phi}{\partial t} + u\frac{\partial \phi}{\partial x} = 0 \tag{11.130}$$

Assuming an exact solution of the form

$$\phi(x,t) = \phi(t)e^{jkx} \tag{11.131}$$

where j is the imaginary number defined by $j^2 = -1$. Then the exact value of the gradient becomes

$$\frac{\partial}{\partial x}\phi(x,t) = jk\phi(t)e^{jkx} = jk\phi(x,t). \tag{11.132}$$

With an interpolation profile, the numerical approximation of the gradient is written in terms of ϕ values at locations $-M\Delta x$, $(-M+1)\Delta x$, …, Δx, $2\Delta x$, …, $N\Delta x$ as

$$\frac{\partial}{\partial x}\phi(x,t) \approx \frac{1}{\Delta x}\sum_{n=-M}^{N} a_n \phi(x+n\Delta x,\, t) = \frac{1}{\Delta x}\sum_{n=-M}^{N} a_n \phi(t)\mathrm{e}^{jk(x+n\Delta x)} \tag{11.133}$$

which, upon substitution of the assumed solution, becomes

$$\frac{\partial}{\partial x}\phi(x,t) \approx \frac{1}{\Delta x}\sum_{n=-M}^{N} a_n \phi(t)e^{jk(x+n\Delta x)} = \frac{1}{\Delta x}\sum_{n=-M}^{N} a_n \mathrm{e}^{jkn\Delta x}\phi(x,t). \tag{11.134}$$

By comparing the exact and numerical solutions, it is found that

$$k = \frac{-j}{\Delta x}\sum_{n=-M}^{N} a_n \mathrm{e}^{jkn\Delta x}. \tag{11.135}$$

In general k is an imaginary number that can be written as

$$k = \mathrm{Re}(k) + j\,\mathrm{Im}(k) \tag{11.136}$$

where Re and Im refer to the real and imaginary part, respectively. Inserting k in the exact solution, the approximate solution is obtained as

$$\begin{aligned}\phi(x,t) &= \phi(t)e^{jkx} \\ &\approx \phi(t)e^{j[\mathrm{Re(k)+jIm(k)}]x} = \phi(t)\ \underbrace{\mathrm{e}^{j\mathrm{Re}(k)x}}_{\substack{\textit{Phase}\\ \textit{Dispersion}}}\ \underbrace{\mathrm{e}^{-\mathrm{Im}(k)x}}_{\substack{\textit{Amplitude}\\ \textit{Dissipation}}}\end{aligned} \tag{11.137}$$

Therefore, the numerical solution may include both diffusion (or dissipative) and dispersion errors. If k is real, only dispersion error occurs. However if k is complex, then both types of errors will arise. Based on this analysis, the value of k for the upwind and CD schemes can be checked. For the upwind scheme, the gradient is computed as

$$\begin{aligned}\frac{\partial\phi}{\partial x} &\simeq \frac{\phi_e - \phi_w}{\Delta x} = \frac{\phi_C - \phi_W}{\Delta x} = \frac{e^{jkx} - e^{jk(x-\delta x)}}{\Delta x}\phi(x,t) \\ &= \frac{1-\cos(k\delta x) + j\sin(k\delta x)}{\Delta x}\phi(x,t) \\ &= jk\phi(x,t) \Rightarrow k = \frac{\sin(k\delta x)}{\Delta x} - j\frac{1-\cos(k\delta x)}{\Delta x}\end{aligned} \tag{11.138}$$

It is clear that the upwind scheme gives rise to both types of errors. For the central difference scheme, the gradient is given by

$$\begin{aligned}\frac{\partial\phi}{\partial x} &\simeq \frac{\phi_e - \phi_w}{\Delta x} = \frac{\phi_E - \phi_W}{2\Delta x} = \frac{e^{jk(x+\delta x)} - e^{jk(x-\delta x)}}{2\Delta x}\phi(x,t) \\ &= \frac{1}{\Delta x} j\,\sin(k\delta x)\phi(x,t) = jk\phi(x,t) \Rightarrow k = \frac{\sin(k\delta x)}{\Delta x}\end{aligned} \tag{11.139}$$

Since k is imaginary, then only numerical dispersion error arises. This dispersion error causes oscillations and under/overshoots in the solution.

Having developed a better understanding of numerical dispersion, it is desirable to develop convective non-oscillatory schemes of high order of accuracy. This has kept researchers busy for an extended period of time until the bounding of the convective flux became understood. The development of such schemes will be detailed in the next chapter.

11.7 High Order Schemes on Unstructured Grids

As for structured grids, the functional relationships of HO schemes are defined as functions of the values at the U, C, and D nodes. While the C and D nodes are readily available for any interior face (Fig. 11.20a), defining the U node is not

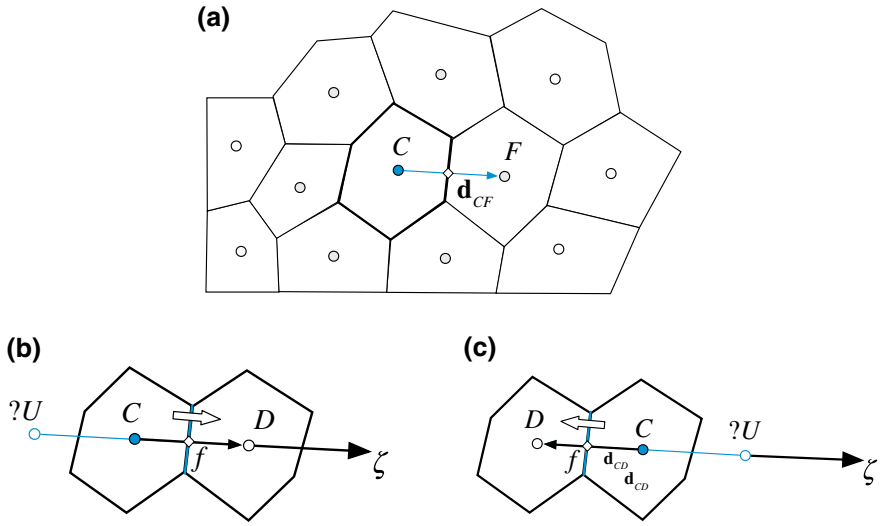

Fig. 11.20 **a** An element in an unstructured grid; the upwind node for HO and HR convection schemes in unstructured grids when the velocity at the element face is **b** negative or **c** positive

straightforward (Fig. 11.20b, c) in an unstructured grid. A simple way around this difficulty is to simply redefine HO schemes in terms of the gradients at the C and D nodes, or a combination of thereof. Another approach is to reconstruct a pseudo U node, which will be detailed in the next chapter and used in developing HR schemes.

11.7.1 Reformulating HO Schemes in Terms of Gradients

This approach relies on profiles developed over structured grids and is best explained by rewriting the QUICK [2] scheme using the new terminology and then generalizing results for any second order profile developed using three points. The functional relationship of the QUICK [2] scheme can be written as

$$\phi_f = \phi_C + \frac{1}{4}\left(\frac{\phi_D - \phi_U}{2}\right) + \frac{1}{4}(\phi_D - \phi_C) \tag{11.140}$$

By approximating the gradients at the locations C and f in the $\mathbf{d}_{CF}$ or ζ direction as shown in Fig. 11.19, then the gradients at the centroids C and f are computed as

$$\frac{\partial \phi_C}{\partial \zeta} = \frac{\phi_D - \phi_U}{2\Delta\zeta} \quad \frac{\partial \phi_f}{\partial \zeta} = \frac{\phi_D - \phi_C}{\Delta\zeta} \tag{11.141}$$

Using Eq. (11.141), Eq. (11.140) can be recast into

$$\phi_f = \phi_C + \frac{1}{2}\left(\frac{\phi_D - \phi_C}{2\Delta\zeta}\right)\frac{\Delta\zeta}{2} + \frac{1}{2}\left(\frac{\phi_D - \phi_C}{\Delta\zeta}\right)\frac{\Delta\zeta}{2} \tag{11.142}$$

or

$$\phi_f = \phi_C + \frac{1}{2}\frac{\partial\phi_C}{\partial\zeta}\frac{\Delta\zeta}{2} + \frac{1}{2}\frac{\partial\phi_f}{\partial\zeta}\frac{\Delta\zeta}{2} \tag{11.143}$$

Denoting the vector between C and f by $\mathbf{d}_{Cf}$, in vector form the above equation becomes

$$\phi_f = \phi_C + \frac{1}{2}\nabla\phi_C \cdot \mathbf{d}_{Cf} + \frac{1}{2}\nabla\phi_f \cdot \mathbf{d}_{Cf} \tag{11.144}$$

which is quite suitable for use in the context of unstructured grids since it only requires information related to gradient at the C and f locations. As long as the computation of these gradients is second order accurate, the way they are calculated becomes immaterial. This gives higher flexibility, as compared to the original formulation, over unstructured grids. Furthermore, Eq. (11.144) suggests that a profile based on three points can be written as

$$\phi_f = a\phi_C + b\nabla\phi_C \cdot \mathbf{d}_{Cf} + c\nabla\phi_f \cdot \mathbf{d}_{Cf} \tag{11.145}$$

where the constants a, b, and c are determined by equating ϕ_f to the profile obtained over structured grids. The general discretized form of Eq. (11.145) can be found once and then used in all subsequent derivations. This is derived by first substituting the approximate values of the gradients using Eq. (11.141) to yield

$$\begin{aligned}\phi_f &= a\phi_C + b\nabla\phi_C \cdot \mathbf{d}_{Cf} + c\nabla\phi_f \cdot \mathbf{d}_{Cf} \\ &= a\phi_C + b\frac{\phi_D - \phi_U}{2\Delta\zeta}\frac{\Delta\zeta}{2} + c\frac{\phi_D - \phi_C}{2\Delta\zeta}\frac{\Delta\zeta}{2}\end{aligned} \tag{11.146}$$

and then after some algebraic manipulations the final form is obtained as

$$\phi_f = \left(a - \frac{c}{2}\right)\phi_C + \left(\frac{b}{4} + \frac{c}{4}\right)\phi_D - \frac{b}{4}\phi_U \tag{11.147}$$

Using the above equation, the calculation of the a, b, and c coefficients is easily accomplished. For example the SOU profile in terms of the gradients can be found as follows:

$$\left.\begin{aligned}\phi_f &= \left(a-\frac{c}{2}\right)\phi_C + \left(\frac{b}{4}+\frac{c}{2}\right)\phi_D - \frac{b}{4}\phi_U \\ &= \frac{3}{2}\phi_C - \frac{1}{2}\phi_U\end{aligned}\right\} \Rightarrow \begin{cases}\dfrac{b}{4}=\dfrac{1}{2} \Rightarrow b=2 \\ \dfrac{b}{4}+\dfrac{c}{2}=0 \Rightarrow c=-1 \\ a-\dfrac{c}{2}=\dfrac{3}{2} \Rightarrow a=1\end{cases} \tag{11.148}$$

Thus the equivalent form of the SOU scheme is given by

$$\phi_f = \phi_C + \left(2\nabla\phi_C - \nabla\phi_f\right)\cdot \mathbf{d}_{Cf} \tag{11.149}$$

Following the same procedure, the gradient forms of the schemes presented earlier are found to be

Upwind scheme : $$\phi_f = \phi_C \tag{11.150}$$

Central difference : $$\phi_f = \phi_C + \nabla\phi_f \cdot \mathbf{d}_{Cf} \tag{11.151}$$

SOU scheme : $$\phi_f = \phi_C + \left(2\nabla\phi_C - \nabla\phi_f\right)\cdot \mathbf{d}_{Cf} \tag{11.152}$$

FROMM scheme : $$\phi_f = \phi_C + \nabla\phi_C \cdot \mathbf{d}_{Cf} \tag{11.153}$$

QUICK scheme : $$\phi_f = \phi_C + \frac{1}{2}\left(\nabla\phi_C + \nabla\phi_f\right)\cdot \mathbf{d}_{Cf} \tag{11.154}$$

Downwind scheme : $$\phi_f = \phi_C + 2\nabla\phi_f \cdot \mathbf{d}_{Cf} \tag{11.155}$$

A full chapter was devoted to the calculation of the gradient at the element centroid and faces and the reader is referred to Chap. 9 for the calculation of $\nabla\phi_C$ and $\nabla\phi_f$.

11.8 The Deferred Correction Approach

The Deferred Correction (DC) procedure of Khosla and Rubin [17] is a compacting technique that enables the use of HO schemes in codes initially written for low order schemes without violating any of the stability rules. The approach is applicable on any type of structured or unstructured grid systems. The method is based on writing the convection flux at a cell face f calculated using a HO scheme as

$$\dot{m}_f \phi_f^{HO} = \underbrace{\dot{m}_f \phi_f^U}_{implicit} + \underbrace{\dot{m}_f\left(\phi_f^{HO} - \phi_f^U\right)}_{explicit} \tag{11.156}$$

where the superscripts U and HO refer to the Upwind and High Order scheme, respectively. By expressing the convection flux in this way, the first term on the right hand side (RHS) is implicitly evaluated by expressing it in terms of nodal

values, while the second term on the RHS is evaluated explicitly using the latest available ϕ values, i.e., values from the previous iteration in an iterative solution procedure. In terms of nodal values, Eq. (11.156) is expressed as

$$\begin{aligned}\dot{m}_f\phi_f^{HO} &= \| \dot{m}_f, 0 \| \phi_C - \| -\dot{m}_f, 0 \| \phi_F + \left(\dot{m}_f\phi_f^{HR} - \| \dot{m}_f, 0 \| \phi_C + \| -\dot{m}_f, 0 \| \phi_F\right) \\ &= \underbrace{FluxC_f\phi_C + FluxF_f\phi_F}_{implicit} + \underbrace{FluxV_f}_{explicit}\end{aligned} \tag{11.157}$$

where

$$\begin{aligned}FluxC_f &= \| \dot{m}_f, 0 \| \\ FluxF_f &= - \| -\dot{m}_f, 0 \| \\ FluxV_f &= \dot{m}_f\phi_f^{HR} - FluxC_f\phi_C - FluxF_f\phi_F\end{aligned} \tag{11.158}$$

Substituting the convection flux given by Eqs. (11.157) and (11.158) in Eq. (11.156) and rearranging, the final form of the algebraic equation is obtained as

$$a_C\phi_C + \sum_{F\sim NB(C)} a_F\phi_F = b_C \tag{11.159}$$

where

$$\begin{aligned}a_F &= FluxF_f = - \| -\dot{m}_f, 0 \| \\ a_C &= \sum_{f\sim nb(C)} FluxC_f = \sum_{f\sim nb(C)} \| \dot{m}_f, 0 \| = \sum_{f\sim nb(C)} \left(\dot{m}_f + \| -\dot{m}_f, 0 \|\right) \\ &= - \sum_{F\sim NB(C)} a_F + \sum_{f\sim nb(C)} \dot{m}_f\end{aligned} \tag{11.160}$$

and

$$\begin{aligned}b_C &= Q_C^{\phi}V_C - \underbrace{\sum_{f\sim nb(C)} FluxV_f}_{b_C^{DC}} \\ &= Q_C^{\phi}V_C - \underbrace{\sum_{f\sim nb(C)} \dot{m}_f\left(\phi_f^{HO} - \phi_f^{U}\right)}_{b_C^{DC}}\end{aligned} \tag{11.161}$$

the b_C^{DC} represents the extra source term arising due to the DC procedure. Moreover, the DC technique results in an equation for which the coefficient matrix is always diagonally dominant since it is formed using the upwind scheme.

Example 2
Derive the coefficients for a deferred correction implementation of the QUICK scheme

Solution
The conservation equation is discretized into the following algebraic equation in every cell:

$$a_C\phi_C + \sum_{F\sim NB(C)} a_F\phi_F = b_C$$

For the quick scheme the value at a cell face is computed as

$$\phi_f = \frac{3}{4}\phi_C - \frac{1}{8}\phi_U + \frac{3}{8}\phi_D$$

The coefficients of the algebraic system of equations in a deferred correction approach are based on the UPWIND convection scheme. As such these coefficients are given by

$$a_F = -\|-\dot{m}_f, 0\|$$
$$a_C = -\sum_{F\sim NB(C)} a_F + \sum_{f\sim nb(C)} \dot{m}_f$$

The difference between the UPWIND and QUICK schemes is explicitly accounted for in the source term which is modified to

$$b_C = S_C^{\phi}V_C - \sum_{f\sim nb(C)} \dot{m}_f\left(\frac{3}{4}\phi_C - \frac{9}{8}\phi_U + \frac{3}{8}\phi_D\right)$$

This compacting procedure is simple to implement, however as the difference between the cell face values calculated with the upwind scheme and that calculated with the HO scheme becomes larger, the convergence rate diminishes.

11.9 Computational Pointers

11.9.1 uFVM

The assembly of the convection term in uFVM is performed in cfdAssembleConvectionTerm, where the assembly for the upwind scheme is performed first for interior faces (cfdAssembleConvectionTermInterior), then for boundary faces by looping over the various boundary patches (cfdAssembleConvectionTermInletBC, cfdAssembleConvectionTermOutletBC, etc.).

The core of the interior faces assembly routine is shown in Listing 11.1. The owner and neighbor indices for the interior faces are first retrieved and an upwind index is defined (pos). Then the coefficients for the upwind scheme are computed.

```
theMdotName = ['Mdot' theFluidTag];
mdotField = cfdGetMeshField(theMdotName,'Faces');

mdot_f = mdotField.phi(iFaces);

iOwners = [theMesh.faces(iFaces).iOwner];
iNeighbours = [theMesh.faces(iFaces).iNeighbour];
pos = zeros(size(mdot_f));
pos((mdot_f>0))=1;

theFluxes.FLUXC1f(iFaces,1) = mdot_f.*pos;
theFluxes.FLUXC2f(iFaces,1) = mdot_f.*(1-pos);
theFluxes.FLUXVf(iFaces,1) = 0;
```

Listing 11.1 Convection scheme class declaration

The implementation of High Order schemes is performed after the upwind assembly using the deferred correction method. The assembly of the QUICK scheme (cfdAssembleConvectionTermDCQUICK) is shown in Listing 11.2.

```
iUpwind = pos.*iOwners + (1-pos).*iNeighbours;
%get the upwind gradient at the interior faces
phiGradCf = phiGrad(iUpwind,:,iComponent);
%interpolated gradient to interior faces
iOwners = [theMesh.faces(iFaces).iOwner]';
iNeighbours = [theMesh.faces(iFaces).iNeighbour]';
pos = zeros(size(mdot_f));
pos(mdot_f>0)=1;
%
phiGradf                                                                    =
cfdInterpolateGradientsFromElementsToInteriorFaces('Average:Corrected'
,phiGrad,phi);
rc = [theMesh.elements(iUpwind).centroid]';
rf = [theMesh.faces(iFaces).centroid]';
rCf = rf-rc;
%
corr = mdot_f .* dot(phiGradCf'+phiGradf',rCf')'*0.5;
%
theFluxes.FLUXTf(iFaces) = theFluxes.FLUXTf(iFaces) +  corr;
```

Listing 11.2 Assembly of the QUICK scheme

The deferred correction value is computed for all interior faces as

$$\dot{m}_f\left(\phi_f^{HR}-\phi_f^{UPWIND}\right)=\dot{m}_f\left(\underbrace{\phi_C+\frac{1}{2}\left(\nabla\phi_C+\nabla\phi_f\right)\cdot\mathbf{d}_{Cf}}_{QUICK}-\underbrace{\phi_C}_{UPWIND}\right)$$
$$=\dot{m}_f\frac{1}{2}\left(\nabla\phi_C+\nabla\phi_f\right)\cdot\mathbf{d}_{Cf} \tag{11.162}$$

11.9.2 OpenFOAM®

In OpenFOAM® [18] the convection term can be evaluated either explicitly using **fvc::div(mDot, phi)** or implicitly via the **fvm::div(mDot,phi)** function. The **fvc::div(mDot,phi)** returns a field in which the divergence of phi is evaluated in each cell. The field is added to the right hand side of the system of equations. The **fvm::div(mDot,phi)** returns the **fvMatrix**, a matrix of coefficients that are evaluated based on the linearization of the face fluxes. The matrix of coefficients is then added to the left hand side of the system of equations.

The scripts of **fvm::div** and **fvc::div** functions can be found in the file **FOAM_SRC/finiteVolume/finiteVolume/convectionSchemes/gaussConvectionScheme/gaussConvectionSheme.C.** As already stated, the convection class is based on the type of interpolation scheme at the face and accordingly the declaration of the class is performed as displayed in Listing 11.3.

```
template<class Type>
class gaussConvectionScheme
:
    public fv::convectionScheme<Type>
{
    // Private data

        tmp<surfaceInterpolationScheme<Type> > tinterpScheme_;
```

Listing 11.3 Convection scheme class declaration

In the above declaration the private member **tinterpScheme_** describes the face interpolation type on which the divergence operator is based. Additionally to the above mentioned function, the **gaussConvectionScheme** class defines two auxiliary functions named **interpolate** and **flux**, as shown in Listing 11.4, which wrap the **surfaceInterpolation** class used to perform interpolation from volume fields to surface fields.

```
template<class Type>
tmp<GeometricField<Type, fvsPatchField, surfaceMesh> >
gaussConvectionScheme<Type>::interpolate
(
    const surfaceScalarField&,
    const GeometricField<Type, fvPatchField, volMesh>& vf
) const
{
    return tinterpScheme_().interpolate(vf);
}

template<class Type>
tmp<GeometricField<Type, fvsPatchField, surfaceMesh> >
gaussConvectionScheme<Type>::flux
(
    const surfaceScalarField& faceFlux,

    const GeometricField<Type, fvPatchField, volMesh>& vf
) const
{
    return faceFlux*interpolate(faceFlux, vf);
}
```

Listing 11.4 The **gaussConvectionScheme** class that defines the **interpolate** and **flux** functions

For the explicit discretization of convection, the operator defines a field where the divergence of the cell face fluxes are stored. The code used by the **fvc::div** operator for the explicit evaluation of the divergence of a field over a specific volume is as follows (Listing 11.5):

```
template<class Type>
tmp<GeometricField<Type, fvPatchField, volMesh> >
gaussConvectionScheme<Type>::fvcDiv
(
    const surfaceScalarField& faceFlux,
    const GeometricField<Type, fvPatchField, volMesh>& vf
) const
{
    tmp<GeometricField<Type, fvPatchField, volMesh> > tConvection
    (
        fvc::surfaceIntegrate(flux(faceFlux, vf))
    );

    tConvection().rename
    (
        "convection(" + faceFlux.name() + ',' + vf.name() + ')'
    );

    return tConvection;
}
```

Listing 11.5 Script used for the explicit evaluation of the divergence operator

The calculation of the divergence of a field involves the following three steps:

1. Evaluating values at the faces of the element.
2. Multiplying the value at the face with the mass flux at the face (i.e., **faceFlux**).
3. Summing the contributions of all cell faces and dividing the sum by the cell volume.

First the face value of the generic variable **vf** is evaluated. Then the estimate obtained is multiplied by the corresponding face mass flux value using auxiliary functions. Finally the sum over the faces of the cell is performed using the **surfaceIntegrate** function. The implementation of this function can be found under **FOAM_SRC/finiteVolume/finiteVolume/fvc/fvcSurfaceIntegrate.C** and its script is given in Listing 11.6.

```
template<class Type>
void surfaceIntegrate
(
    Field<Type>& ivf,
    const GeometricField<Type, fvsPatchField, surfaceMesh>& ssf
)
{
    const fvMesh& mesh = ssf.mesh();

    const labelUList& owner = mesh.owner();
    const labelUList& neighbour = mesh.neighbour();

    const Field<Type>& issf = ssf;

    forAll(owner, facei)
    {
        ivf[owner[facei]] += issf[facei];
        ivf[neighbour[facei]] -= issf[facei];
    }

    forAll(mesh.boundary(), patchi)
    {
        const labelUList& pFaceCells =
            mesh.boundary()[patchi].faceCells();

        const fvsPatchField<Type>& pssf = ssf.boundaryField()[patchi];

        forAll(mesh.boundary()[patchi], facei)
        {
            ivf[pFaceCells[facei]] += pssf[facei];
        }
    }

    ivf /= mesh.V();
}
```

Listing 11.6 Script of the **surfaceIntegrate** function

The implicit discretization of the convection term is performed using the **fvm::div** operator. It does so by implicitly expressing the dependent variable value at the face as function of the owner and neighbor values according to

$$\phi_f = \underbrace{\varpi}_{\substack{owner \\ coefficient}} \phi_O + \underbrace{(1-\varpi)}_{\substack{neighbour \\ coefficient}} \phi_N \tag{11.163}$$

where subscripts O and N refer to owner and neighbor, respectively, and ϖ represents the weight assigned to the contribution of the owner to the value at the face. The way the weight is calculated will be described in the next chapter. The coefficients of ϕ_O and ϕ_N are then used to assemble the matrix of coefficients. The script used to perform implicit discretization of the divergence operator reads (Listing 11.7)

```
template<class Type>
tmp<fvMatrix<Type> >
gaussConvectionScheme<Type>::fvmDiv
(
    const surfaceScalarField& faceFlux,
    const GeometricField<Type, fvPatchField, volMesh>& vf
) const
{
    tmp<surfaceScalarField> tweights = tinterpScheme_().weights(vf);
    const surfaceScalarField& weights = tweights();

    tmp<fvMatrix<Type> > tfvm
    (
        new fvMatrix<Type>
        (
            vf,
            faceFlux.dimensions()*vf.dimensions()
        )
    );
    fvMatrix<Type>& fvm = tfvm();
…
```

Listing 11.7 Implicit calculation of the divergence of a field

where in a first place an **fvMatrix** is defined and then, as shown in Listing 11.8, it is properly filled with the corresponding coefficients as

```
    fvm.lower() = -weights.internalField()*faceFlux.internalField();
    fvm.upper() = fvm.lower() + faceFlux.internalField();
    fvm.negSumDiag();
```

Listing 11.8 Properly assembling the weights contributions to coefficients

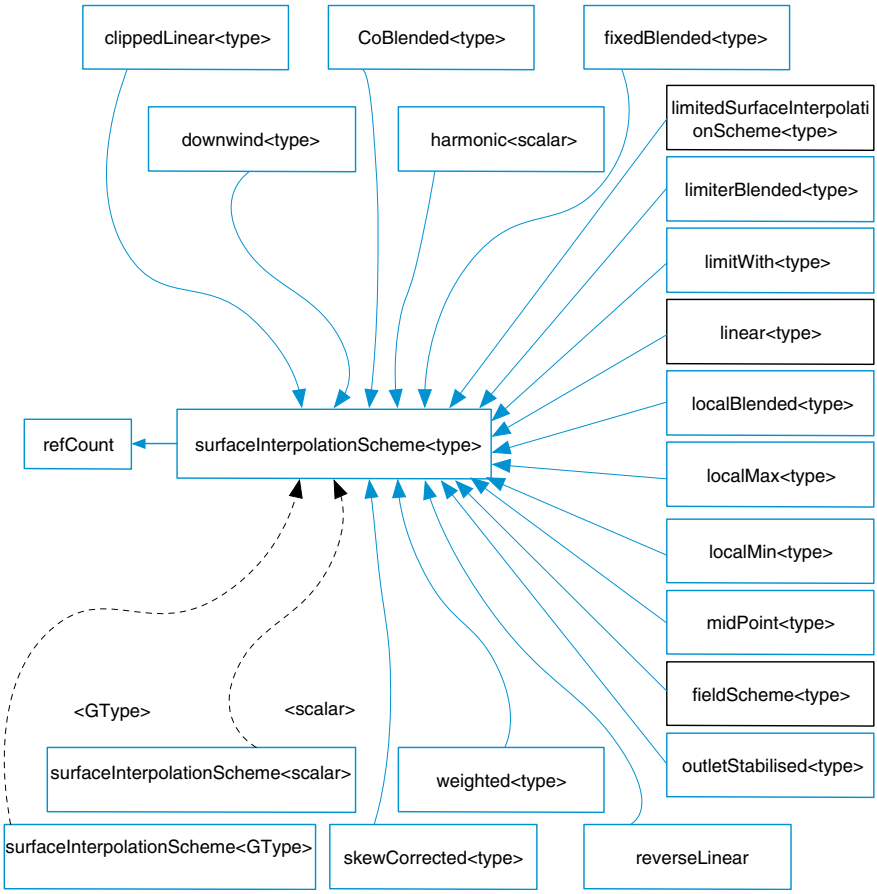

Fig. 11.21 **UML** Graph for the surfaceInterpolationScheme class, where a box with a black border denotes a documented struct or class for which not all inheritance/containment relations are shown

where the upper and lower vectors are now defined using the interpolation weights denoted **tweights**, as described above. Further, Eq. (11.163) indicates that the implementation of the implicit operator in OpenFOAM® is based on the so called "downwind" discretization (as explained later) in which all schemes are discretized in a fully implicit manner independently of the order of accuracy and without using a deferred correction approach. This technique is also similar to the downwind weighing factor method that will be described in the next chapter.

As described above for both explicit and implicit discretization methods, the divergence operator for the calculation of the convection term is based on the face interpolation and the calculation of the weights. OpenFOAM®performs these tasks in a base class denoted by **surfaceInterpolationScheme**. From this base class a large number of interpolation schemes are derived and implemented, as shown in the **UML** graph depicted in Fig. 11.21. For better understanding, additional details about this class are given next.

As mentioned above, the **fvc::div** and **fvm::div** operators use the **surfaceInterpolationScheme** class to perform the needed tasks for each discretization method. In this class, the functions used to calculate the values at the faces (vf) and the weights are displayed in Listings 11.9 and 11.10, respectively, and where **tinterpScheme_** (Listing 11.10) is a **surfaceInterpolationScheme** object.

```
template<class Type>
tmp<GeometricField<Type, fvsPatchField, surfaceMesh> >
gaussConvectionScheme<Type>::interpolate
(
    const surfaceScalarField&,
    const GeometricField<Type, fvPatchField, volMesh>& vf
) const

{
    return tinterpScheme_().interpolate(vf);
}
```

Listing 11.9 The interpolation function where the value at the face (vf) is computed

and

```
template<class Type>
tmp<fvMatrix<Type> >
gaussConvectionScheme<Type>::fvmDiv
(
    const surfaceScalarField& faceFlux,
    const GeometricField<Type, fvPatchField, volMesh>& vf
) const
{
    tmp<surfaceScalarField> tweights = tinterpScheme_().weights(vf);
    const surfaceScalarField& weights = tweights();
```

Listing 11.10 The function used to calculate the weights

The definition of the class can be found in **FOAM_SRC/finiteVolume/interpolation/surfaceInterpolation/surfaceInterpolationScheme/SurfaceInterpolationScheme.H** where the two main functions (Listings 11.11 and 11.12), necessary for the calculation of the divergence operator are defined.

```
//- Return the face-interpolate of the given cell field
//  with explicit correction
virtual tmp<GeometricField<Type, fvsPatchField, surfaceMesh> >
interpolate(const GeometricField<Type, fvPatchField, volMesh>&) const;
```

Listing 11.11 Definition of the main function for face interpolation

```
//- Return the interpolation weighting factors for the given field
virtual tmp<surfaceScalarField> weights
(
    const GeometricField<Type, fvPatchField, volMesh>&
) const = 0;
```

Listing 11.12 Definition of the main function for weight factors calculation

The **interpolate** function in Listing 11.9 (here OpenFOAM® adopts the same function name but for a different implementation) is used with the explicit operator **fvc::div** and is defined as a normal virtual function meaning that a derived class may or may not adopt it. For most of the derived classes this function is not selected and the definition in the **surfaceInterpolationScheme** class is based on a modified form of Eq. (11.163) written as

$$\phi_f = \underbrace{\varpi}_{\substack{owner\\coefficient}} \phi_O + \underbrace{(1-\varpi)}_{\substack{neighbour\\coefficient}} \phi_N = \phi_N + \varpi(\phi_O - \phi_N) \tag{11.164}$$

To implement Eq. (11.164), OpenFOAM® uses auxiliary functions with names similar to the ones used earlier in the **surfaceInterpolationScheme** class. First an **interpolate** class with a single argument is instantiated from the divergence operator. Then inside the **interpolate** function a new **interpolate** function with two arguments is introduced along with a new **weights** function, using the script shown in Listing 11.13.

```
template<class Type>
tmp<GeometricField<Type, fvsPatchField, surfaceMesh> >
surfaceInterpolationScheme<Type>::interpolate
(
    const GeometricField<Type, fvPatchField, volMesh>& vf
) const
{
…

    tmp<GeometricField<Type, fvsPatchField, surfaceMesh> > tsf
        = interpolate(vf, weights(vf));
…
```

Listing 11.13 The new interpolate function with two arguments

The new **interpolate** function (the one with the two arguments) compute the face value according to Eq. (11.164). The script used to perform this task is given by (Listing 11.14)

```
//- Return the face-interpolate of the given cell field
//  with the given weighting factors
template<class Type>
tmp<GeometricField<Type, fvsPatchField, surfaceMesh> >
surfaceInterpolationScheme<Type>::interpolate
(
    const GeometricField<Type, fvPatchField, volMesh>& vf,
    const tmp<surfaceScalarField>& tlambdas
)
{
…
    const surfaceScalarField& lambdas = tlambdas();

    const Field<Type>& vfi = vf.internalField();
    const scalarField& lambda = lambdas.internalField();

    const fvMesh& mesh = vf.mesh();
    const labelUList& P = mesh.owner();
    const labelUList& N = mesh.neighbour();

   GeometricField<Type, fvsPatchField, surfaceMesh>& sf = tsf();

    Field<Type>& sfi = sf.internalField();


    for (register label fi=0; fi<P.size(); fi++)
    {
        sfi[fi] = lambda[fi]*(vfi[P[fi]] - vfi[N[fi]]) + vfi[N[fi]];
    }
…
```

Listing 11.14 Script used to compute face values according to Eq. (11.164)

where lambda represents the weight ϖ in Eq. (11.164).

As briefly introduced above, the **weights** function is now a pure virtual function and it defines the weights of the interpolation. Contrary to the **interpolate** function, the **weights** function being pure virtual has to be redefined for every derived class (C^{++} requirement). Looking again at Fig. 11.20, all derived classes have to define the **weights** function. The redefinition must be performed by constructing the proper weights corresponding to the scheme adopted.

For the case of the central difference scheme the **weights** function is defined in a class named **linear<Type>** with its script given by (Listing 11.15),

```
//Member Functions

//- Return the interpolation weighting factors
virtual tmp<surfaceScalarField> weights
(
   const GeometricField<Type, fvPatchField, volMesh>&
) const
{
   return this->mesh().surfaceInterpolation::weights();
}
```

Listing 11.15 Script for calculating the weights of the central difference scheme

where **surfaceInterpolation::weights()** returns the distance weights based on the mesh, thus defining the central difference discretization.

Another example is the *downwind* scheme defined by Eq. 11.44, and for which the **weights** function is given by (Listing 11.16)

```
//- Return the interpolation weighting factors
virtual tmp<surfaceScalarField> weights
(
    const GeometricField<Type, fvPatchField, volMesh>&
) const
{
    return neg(faceFlux_);
}
```

Listing 11.16 Script for calculating the weights of the downwind scheme

where the **neg** function returns 0 for positive fluxes and 1 for negative ones, which is the opposite of the **pos** function used with the upwind scheme.

11.10 Closure

The chapter dealt with the discretization of the convection term. The difficulties associated with the use of a linear symmetrical profile were discussed and remedies suggested. The upwind scheme, although stable and generates physically plausible results, is highly diffusive smearing sharp gradients and producing results that are first order accurate. Upwind-biased HO convection schemes were shown to improve accuracy but to produce a new type of error known as the dispersion error, which manifest itself in the solution through wiggles, oscillations, and over/under shoots across sharp gradients. No attempt was made to address this error in this chapter. Among other issues, this will form the subject of the next chapter that further expands on the discretization of the convection term.

11.11 Exercises

Exercise 1

(a) Find third order accurate expressions, i.e., $O(\Delta x^3)$, of ϕ_U, ϕ_C, and ϕ_D by using a Taylor series expansion about the face f for the one dimensional uniform grid shown in Fig. 11.22.
(b) Derive a high-order scheme using a linear combination $(a\phi_U + b\phi_C + c\phi_D)$ of these expressions in such a way as to eliminate the first and second order derivatives from the final expression for ϕ_f with the additional condition that $a + b + c = 1$.
(c) Prove that the scheme you just derived is third order accurate in its representation of the convection operator.

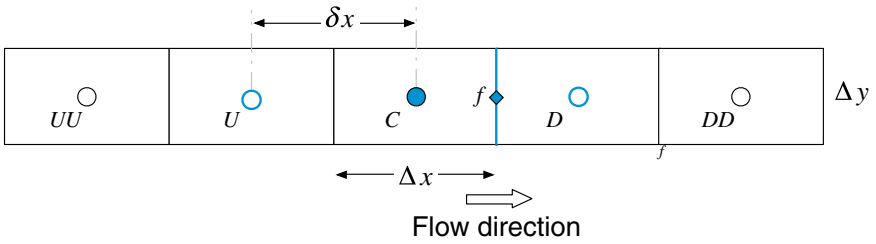

Fig. 11.22 A uniform one dimensional grid system

Exercise 2

The QUICK scheme fits a quadratic function to three nodal values to estimate the value of a scalar at a cell face, according to

$$\phi_f = -\frac{1}{8}\phi_U + \frac{3}{4}\phi_C + \frac{3}{8}\phi_D$$

(a) For a uniform cartesian two dimensional grid write down the expressions for ϕ_e, ϕ_w, ϕ_n and ϕ_s in terms of the values at neighboring nodes, assuming that the velocity components u and v are known, constant, and positive.
(b) Neglecting diffusion and assuming a uniform source Q^ϕ per unit volume, derive an algebraic discretization of the ϕ scalar conservation equation given by

$$\nabla.(\rho \mathbf{v}\phi) = Q^\phi$$

in the form

$$a_C\phi_C + \sum_{F\sim NB(C)} a_F\phi_F = b_C$$

assuming that a cell is of unit depth, with the area of its e, w, n, and s faces denoted by S_e, S_w, S_n, and S_s, respectively, and its volume by V_C.

(c) Splitting the QUICK expression for ϕ_f into the form "Upwind differencing" + "deferred correction" so that the coefficients become similar to those of the upwind scheme, then moving the deferred correction to the source term, write an expression for this source term.

Exercise 3

In a steady two dimensional situation, the variable ϕ is governed by

$$\nabla.(\rho\mathbf{v}\phi) = Q^{\phi}$$

where $\rho = 1$, $Q^{\phi} = 15 - 3\phi$.

The flow field is such that $\mathbf{v} = \mathbf{li} + 4\mathbf{j}$ everywhere and $\Delta x = \Delta y = 1$. The domain is discretized using the orthogonal grid shown in Fig. 11.23 with the values of ϕ given at the inlet boundaries as shown in the figure. Using the finite volume approach and the Second Order Upwind convection scheme to

(a) derive the algebraic equations for the four control volumes, and
(b) compute the values of ϕ_1, ϕ_2, ϕ_3 and ϕ_4 using 3 iterations of a simple Gauss-Siedel type solver.

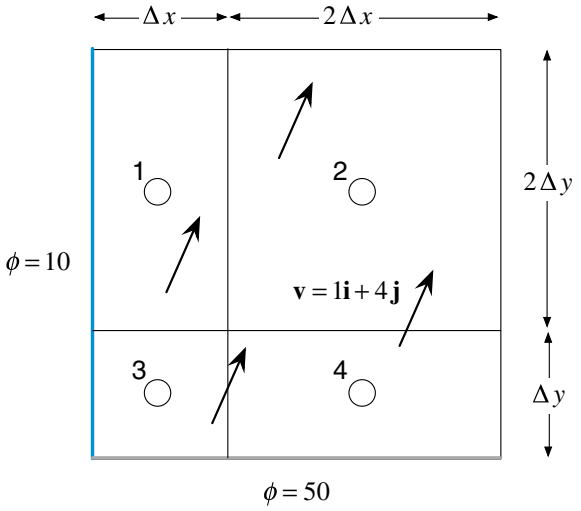

Fig. 11.23 A two dimensional configuration discretized using a non-uniform Cartesian grid for the advection of a scalar ϕ in the presence of a source term

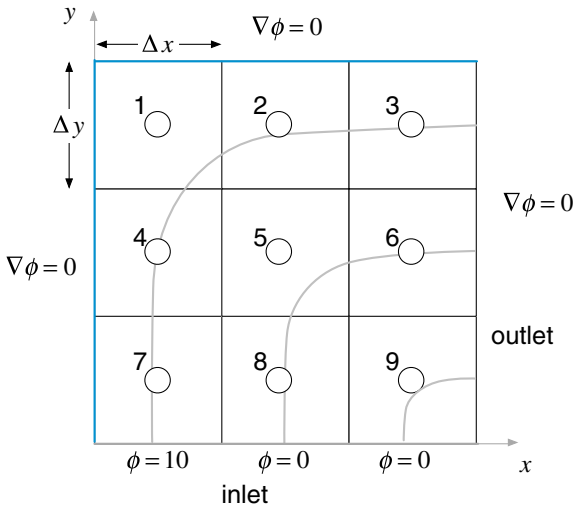

Fig. 11.24 Convection of a two dimensional scalar field

Exercise 4

Consider the steady transport of a scalar ϕ in the domain shown in Fig. 11.24. The governing conservation equation is given by

$$\nabla.(\rho \mathbf{v}\phi) = 0$$

where $\rho = 1$, $\mathbf{v} = 2yx^2\mathbf{i} - 2xy^2\mathbf{j}$, and $\Delta x = \Delta y = 1/3$.

(a) Using the UPWIND scheme, discretize the equation over the computational domain and find the value of ϕ at each element centroid.
(b) Using the QUICK scheme, applied via a deferred correction approach, discretize the equation over the computational domain and find the value of ϕ at each element centroid.

Exercise 5 (OpenFOAM®)

(a) Using Doxygen [19], list all derived classes of the *surfaceInterpolationScheme<Type>* class (these classes implement the virtual *interpolate* function).
(b) Describe the weights function of the class midPoint<Type>, downwind <Type> and linear<Type>.
(c) Write a class that inherits the *surfaceInterpolationScheme<Type>* class and implements the interpolate function using a geometric average.

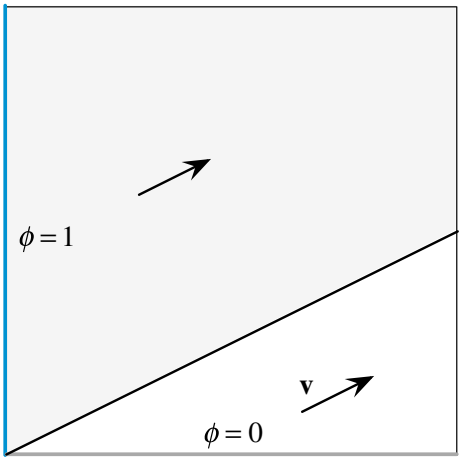

Fig. 11.25 Advection of a step profile in an oblique velocity field

Exercise 6 (OpenFOAM®, uFVM)
The advection of a step profile in an oblique velocity field, **v** = 2**i** + **j**, shown in Fig. 11.25 is governed by

$$\nabla . (\rho \mathbf{v} \phi) = 0$$

For different grid sizes, setup the problem and solve it in OpenFOAM® and uFVM using the following advection schemes assuming unit dimensions in x and y directions, and compare results with the exact solution $(\rho = 1)$:

(a) UPWIND
(b) QUICK
(c) SOU

Exercise 7 (OpenFOAM®, uFVM)
The Smith-Hutton test governed by

$$\nabla . (\rho \mathbf{v} \phi) = 0$$

and illustrated in Fig. 11.26, involves the pure advection of a step profile in a rotational velocity field described as

$$\mathbf{v} = 2y\left(1 - x^2\right)\mathbf{i} - 2x\left(1 - y^2\right)\mathbf{j}$$

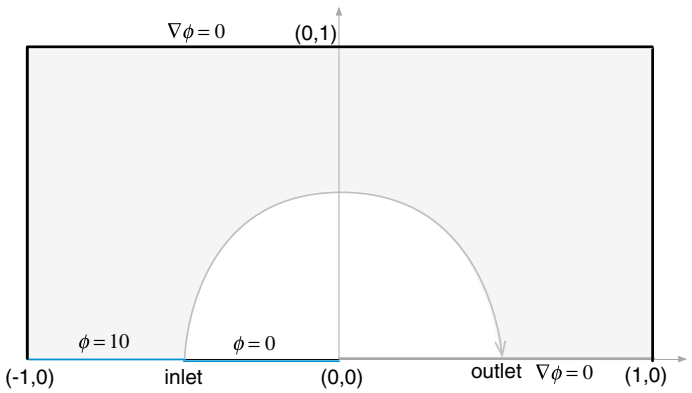

Fig. 11.26 Advection of a step profile in a two dimensional rotational velocity field

For different grid sizes, solve the test in openFOAM® and uFVM using the following advection schemes, and compare results with the exact solution ($\rho = 1$):

(a) UPWIND
(b) QUICK
(c) SOU

References

1. Spalding DB (1972) A novel finite difference formulation for differential expressions involving both first and second derivatives. Int J Numer Meth Eng 4:551–559
2. Leonard BP (1979) A stable and accurate convective modelling procedure based on quadratic upstream interpolation. Comput Methods Appl Mech Eng 19:59–98
3. Raithby GD (1976) A critical evaluation of upstream differencing applied to problems involving fluid flow. Comput Methods Appl Mech Eng 9:75–103
4. Patankar SV (1980) Numerical heat transfer and fluid flow. Hemisphere Publishing Corporation, McGraw-Hill
5. Patankar SV, Baliga BR (1978) A new finite-difference scheme for parabolic differential equations. Numer Heat Transf 1:27–37
6. Courant R, Isaacson E, Rees M (1952) On the solution of nonlinear hyperbolic differential equations by finite differences. Commun Pure Appl Math 5:243–255
7. Moukalled F, Darwish M (2012) Transient schemes for capturing interfaces of free-surface flows. Numer Heat Transf Part B Fundam 61(3):171–203
8. Darwish M, Moukalled F (2006) Convective schemes for capturing interfaces of free-surface flows on unstructured grids. Numer Heat Transf Part B Fundam 49(1):19–42
9. Leonard BP (1979) A survey of finite difference of opinion on numerical muddling of the incomprehensible defective confusion equation. In: Hughes TJR (ed) Finite element methods for convection dominated flows, AMD-34, ASME

10. Shyy W (1985) A study of finite difference approximations to steady state convection dominate flow problems. J Comput Phys 57:415–438
11. Leonard BP, Leschziner MA, McGuirk J (1978) Third order finite-difference method for steady two-dimensional convection. In: Taylor C, Morgan K, Brebbia CA (eds) Numerical methods in laminar and turbulent flows. Pentech Press, London, pp 807–819
12. Fromm JE (1968) A method for reducing dispersion in convective difference schemes. J Comput Phys 3:176–189
13. Stubley GD, Raithby GD, Strong AB (1980) Proposal for a new discrete method based on an assessment of discretization errors. Numerical Heat Transfer 3:411–428
14. de Vahl Davis G, Mallinson GD (1972) False diffusion in numerical fluid mechanics. University of New South Wales, School of Mechanical and Industrial Engineering (Report 1972/FMT/1)
15. Raithby GD (1976) Skew upstream differencing schemes for problems involving fluid flow. Comput Methods Appl Mech Eng 9:153–164
16. Darwish M, Moukalled F (1996) A new route for building bounded skew-upwind schemes. Comput Methods Appl Mech Eng 129:221–233
17. Khosla PK, Rubin SG (1974) A diagonally dominant second-order accurate implicit scheme. Comput Fluids 2:207–209
18. OpenFOAM (2015) Version 2.3.x. http://www.openfoam.org
19. OpenFOAM Doxygen (2015) Version 2.3.x. http://www.openfoam.org/docs/cpp/