Chapter 17
Turbulence Modeling

Abstract This chapter addresses some of the challenges that arise when solving turbulent flow problems. It is not intended to provide a comprehensive account on turbulence modeling, rather, the intention is simply to introduce the subject and focus on the implementation details of some of the most popular turbulence models. The presentation is limited to incompressible turbulent fluid flow and begins with a general introduction to turbulence modeling. Then the Reynolds stress tensor that originates from the adopted averaging procedure and the Boussinesq hypothesis used in modeling the Reynolds stresses are presented. This is followed by a review of the $k-\varepsilon$ and $k-\omega$ two-equation models. These are the most popular of the high Reynolds number and low Reynolds number turbulence models, respectively. The BSL and SST models are then introduced, both are derived by combining the $k-\varepsilon$ and $k-\omega$ models so as to address their respective weaknesses. Finally the treatment of the near wall region is presented in detail.

17.1 Turbulence Modeling

In deriving the Navier-Stokes equation in Chap. 3 no mention was made to whether the flow is laminar or turbulent. Whereas laminar flows are stable, turbulent flows are chaotic, diffusive causing rapid mixing, time-dependent, and involve three-dimensional vorticity fluctuations with a broad range of time and length scales [1]. Turbulence typically develops as an instability of laminar flows appearing at a certain critical Reynolds number. In the fluid, these instabilities are caused by the amplification of the perturbation due to the highly non-linear inertial terms.

The most accepted theory of turbulence is based on the "energy cascade" concept developed by Kolmogorov [2, 3]. According to this theory, turbulence is composed of eddies of different sizes with each one possessing a certain amount of energy that depends on its dimension. The larger eddies break up transferring their energy to smaller size eddies in a chain process by which the smaller newly formed eddies undergo similar breakup processes and transfer their energy to even smaller

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_17

eddies. This break up process continues until the smallest possible eddy size is reached. The smallest eddies are of scales at which the molecular viscosity is very effective at dissipating the turbulent kinetic energy as heat.

The smallest turbulent eddies are characterized by the Kolmogorov micro length (η) and time (t_η) scales given by

$$\eta = \left(\frac{\nu^3}{\varepsilon}\right)^{\frac{1}{4}} \quad t_\eta = \left(\frac{\nu}{\varepsilon}\right)^{\frac{1}{2}} \tag{17.1}$$

where ν is the molecular kinematic viscosity and ε the average rate of dissipation of turbulent kinetic energy that will be defined later. In addition, the size of the largest eddies, which is also known by the integral length scale, is defined as being proportional to the size of the geometry involved.

Based on the energy cascade concept a direct numerical solution of the Navier-Stokes equation for turbulent flows necessitates the use of a very small time step limited by a Courant number below 1 and a fine mesh ($\Delta x < \eta$) resulting in a large number of grid points (proportional to Re^3) to resolve the entire spectrum of temporal and spatial turbulent scales involved. This computationally demanding approach, which is denoted in the literature by Direct Numerical Simulation (DNS), has been used by few workers [4–7] in a limited number of simple studies. Due to its prohibitive computational cost the DNS approach cannot currently be employed to solve industrial problems. Future advances in computer technology may change the situation in favor of the DNS.

To reduce the large computational cost associated with a direct solution of the Navier-Stokes equation, statistical analyses can be used to simplify the resolution of turbulent flows. The time-dependent nature of turbulence together with its wide range of time scales suggest that statistical averaging techniques can be applied to approximate random fluctuations. Time averaging, however, leads to correlations that are not known a priori arising from the nonlinear terms in the equations of motion. Modeling these transpiring correlations constitutes the classic closure problem of turbulence modeling.

Following the statistical approach, workers have devised methods that are computationally less intensive than the DNS. One such method is the large eddy simulation (LES) [8–11], in which large scale turbulent structures are directly simulated whereas small turbulent scales are modeled using sub-grid scale models.

The key concept in the LES is to filter the Navier-Stokes equation to determine which scales to keep and which scales to discard. This is done by applying a spatial statistical filter of the form

$$\langle \mathbf{v}(\mathbf{x}, t) \rangle = \iiint F(\mathbf{x} - \lambda : \Delta)\mathbf{v}(\mathbf{x}, t) d^3\lambda \tag{17.2}$$

where the filter function retains values of $\mathbf{v}$ occurring on scales larger than the filter width Δ. The filter function F, is basically some function which is effectively zero

for values of **v** occurring at small scales. Here the symbol $\langle\ \rangle$ indicates a filtered variable.

Thus in the LES approach, turbulent structures of scales larger than the set minimum filter are not filtered. This has the advantage of directly resolving the larger eddies with the higher content of energy (anisotropic turbulence) while the smallest eddies are simply modeled. Accurate modeling of these eddies is possible since at the smaller scale (below the filter width) turbulence can be considered isotropic and independent of the flow type and boundary conditions. With the recent advances in computer technology the use of LES to solve industrial problems is gaining acceptance.

Nevertheless, currently the most popular approach for tackling industrial turbulent flow problems is the one based on solving the Reynolds Averaged Navier-Stokes (RANS) equations [12] where the statistical averaging is now based not on spatial averaging but on a proper time. The key approach is to decompose the flow variables into a time-mean value component and a fluctuating one (Fig. 17.1), substituting in the original equations, and time-averaging the obtained equations. Even though the name refers to the Navier-Stokes equation, the decomposition is applied to all governing equations. Two tracks for averaging the equations have been followed. The standard Reynolds averaging, which is used to derive the Reynolds-Averaged Navier-Stokes (RANS) equations [12], and the mass-weighted or Favre averaging technique employed with turbulent compressible flows and leading to the Favre-Averaged Navier-Stokes (FANS) equations [13]. Following either path, the intention is to model all scales of turbulent flow. Therefore with this approach the mesh size limitation is not as constraining as in the DNS and LES approaches.

In what follows the development of the RANS equations for an incompressible flow is presented.

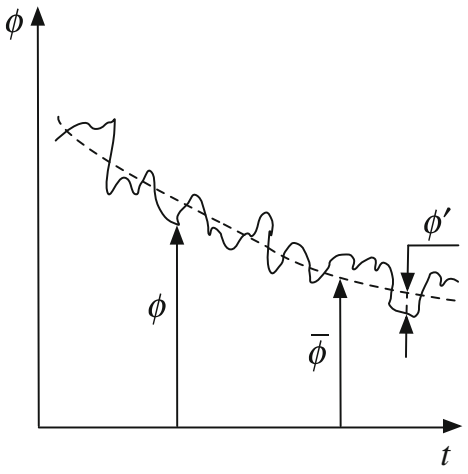

Fig. 17.1 Fluctuating and mean variable components

17.2 Reynolds Averaging

Let ϕ represents at time t and position $\mathbf{x}$ the instantaneous value of any of the flow variables involved ($\mathbf{v}$, p, e, h, T, ρ, etc.). Then, as shown in Fig. 17.1, it is decomposed into a mean value component $\overline{\phi}(\mathbf{x}, t)$ and a fluctuating component $\phi'(\mathbf{x}, t)$ such that

$$\phi(\mathbf{x}, t) = \overline{\phi}(\mathbf{x}, t) + \phi'(\mathbf{x}, t) \tag{17.3}$$

the mean value $\overline{\phi}$ is computed by any of the three Reynolds averaging techniques [12] presented below, of which the time averaging is the most widely used.

17.2.1 Time Averaging

Time averaging represents the average of a quantity over a time interval and is suitable for steady turbulent flows, that is flows which, on average, do not vary with time. If T is the interval over which averaging is performed, then $\overline{\phi}$, which depends only on location, is computed as

$$\overline{\phi}(\mathbf{x}) = \lim_{T\to\infty} \frac{1}{T} \int_{t}^{t+T} \phi(\mathbf{x}, t) dt \tag{17.4}$$

If $\overline{\phi}$ varies slowly with time in comparison with the time scale of turbulent fluctuations, the above equation is replaced by

$$\overline{\phi}(\mathbf{x}, t) = \frac{1}{T} \int_{t}^{t+T} \phi(\mathbf{x}, t) dt \tag{17.5}$$

17.2.2 Spatial Averaging

Spatial averaging represents the average of a quantity over a space interval or a volume V and is suitable for homogeneous turbulence. In this case $\overline{\phi}$, which only depends on time, is computed as

$$\overline{\phi}(t) = \lim_{V\to\infty} \frac{1}{V} \int_{V} \phi(\mathbf{x}, t) dV \tag{17.6}$$

17.2.3 Ensemble Averaging

Ensemble averaging, which is suitable for any type of turbulent flows including unsteady turbulent flows, represents the average of many identical quantities at a certain time. If the number of identical quantities is designated by N, then $\overline{\phi}$, which in this case is a function of space and time, is given by

$$\overline{\phi}(\mathbf{x},t) = \lim_{N\to\infty} \frac{1}{N} \sum_{i=1}^{N} \phi(\mathbf{x},t) \tag{17.7}$$

17.2.4 Averaging Rules

If ϕ and φ are two variables and ϕ' and φ' are their fluctuating components, then some of the averaging rules needed in deriving the RANS equations include the following:

$$\begin{aligned}
\overline{\phi'} &= 0 \\
\overline{\phi} &= \overline{\overline{\phi}} \\
\overline{\nabla\phi} &= \nabla\overline{\phi} \\
\overline{\phi+\varphi} &= \overline{\phi} + \overline{\varphi} \\
\overline{\overline{\phi}\varphi} &= \overline{\phi}\,\overline{\varphi} \\
\overline{\overline{\phi}\varphi'} &= 0 \\
\overline{\phi\varphi} &= \overline{\phi}\,\overline{\varphi} + \overline{\phi'\varphi'}
\end{aligned} \tag{17.8}$$

17.2.5 Incompressible RANS Equations

The incompressible Reynolds-averaged Navier–Stokes equations are based on time-averaged variables. Decomposing the velocity, pressure, and temperature fields into

$$\begin{aligned}
\mathbf{v} &= \overline{\mathbf{v}} + \mathbf{v}' \\
p &= \overline{p} + p' \\
T &= \overline{T} + T' \\
\overline{\mathbf{v}} &= \overline{u}\mathbf{i} + \overline{v}\mathbf{j} + \overline{w}\mathbf{k} \\
\mathbf{v}' &= u'\mathbf{i} + v'\mathbf{j} + w'\mathbf{k}
\end{aligned} \tag{17.9}$$

and substituting $\mathbf{v}$, p, and T by their decomposed expressions in the incompressible continuity, momentum, and energy equations given by Eqs. (3.13), (3.39), and (3.78),

respectively, assuming a Newtonian fluid, and taking the time average, these equations are transformed to

$$\overline{\nabla \cdot [\rho(\overline{\mathbf{v}} + \mathbf{v}')] = 0} \tag{17.10}$$

$$\overline{\frac{\partial}{\partial t}[\rho(\overline{\mathbf{v}} + \mathbf{v}')] + \nabla \cdot \{\rho(\overline{\mathbf{v}} + \mathbf{v}')(\overline{\mathbf{v}} + \mathbf{v}')\} = -\nabla(\overline{p} + p') + \nabla \cdot \left\{\mu\left[\nabla(\overline{\mathbf{v}} + \mathbf{v}') + (\nabla(\overline{\mathbf{v}} + \mathbf{v}'))^{\mathrm{T}}\right]\right\} + \mathbf{f}_b} \tag{17.11}$$

$$\overline{\frac{\partial}{\partial t}[\rho c_p(\overline{T} + T')] + \nabla \cdot \left[\rho c_p(\overline{\mathbf{v}} + \mathbf{v}')(\overline{T} + T')\right] = \nabla \cdot \left[k\nabla(\overline{T} + T')\right] + S^T} \tag{17.12}$$

The Reynolds averaged forms of Eqs. (17.10)–(17.12) are obtained as

$$\nabla \cdot [\rho\overline{\mathbf{v}}] = 0 \tag{17.13}$$

$$\frac{\partial}{\partial t}[\rho\overline{\mathbf{v}}] + \nabla \cdot \{\rho\overline{\mathbf{v}}\overline{\mathbf{v}}\} = -\nabla\overline{p} + \left[\nabla \cdot \left(\overline{\boldsymbol{\tau}} - \rho\overline{\mathbf{v}'\mathbf{v}'}\right)\right] + \overline{\mathbf{f}}_b \tag{17.14}$$

$$\frac{\partial}{\partial t}\left[\rho c_p\overline{T}\right] + \nabla \cdot \left[\rho c_p\overline{\mathbf{v}}\overline{T}\right] = \nabla \cdot \left[k\nabla\overline{T} - \rho c_p\overline{\mathbf{v}'T'}\right] + \overline{S^T} \tag{17.15}$$

The above Reynolds averaged equations are similar to the original conservation equations with the exception of the additional averaged products of the fluctuating components due to the non-linear terms. This introduces six new unknowns (the components of the tensor $-\rho\overline{\mathbf{v}'\mathbf{v}'}$, known as the Reynolds stress tensor $\boldsymbol{\tau}^R$) to the momentum equations and three new unknown turbulent heat fluxes $\left(\dot{\mathbf{q}}^R = -\rho c_p\overline{\mathbf{v}'T'}\right)$ to the energy equation. The expanded forms of the Reynolds stress tensor $\boldsymbol{\tau}^R$ and turbulent heat flux vector are given by

$$\boldsymbol{\tau}^R = -\rho\begin{pmatrix} \overline{u'u'} & \overline{u'v'} & \overline{u'w'} \\ \overline{u'v'} & \overline{v'v'} & \overline{v'w'} \\ \overline{u'w'} & \overline{v'w'} & \overline{w'w'} \end{pmatrix} \quad \dot{\mathbf{q}}^R = -\rho c_p\begin{bmatrix} \overline{u'T'} \\ \overline{v'T'} \\ \overline{w'T'} \end{bmatrix} \tag{17.16}$$

Consequently the set of RANS equations is not a closed set and to be able to solve it additional equations for the unknown Reynolds stress components are required. The process of calculating these Reynolds stresses is denoted in the literature by turbulence modeling. Attempting to develop such equations by using the original conservation equations results in additional unknowns (such as triple products of the fluctuating components) complicating the problem further. The Reynolds stress tensor comes from the non-linear convection term of both momentum and temperature confirming that the turbulence itself is the effect of a non-linear phenomena highly sensitive to any perturbation. Therefore any linear averaging of the equations, like the Reynolds averaging techniques, cannot reduce

the order of the problem. The complexity is actually recursive in that trying to develop additional equations for the triple products, quadruple products arise and so on. To overcome this problem, any turbulence model has to close the system of equations by expressing the non-linear fluctuating stress components only in terms of the mean components as described next.

17.3 Boussinesq Hypothesis

The direct modeling of the Reynolds stress tensor is based on the Boussinesq hypothesis [14–16], which in analogy with Newtonian flows assumes the Reynolds stress to be a linear function of the mean velocity gradients such that

$$\boldsymbol{\tau}^R = -\rho\overline{\mathbf{v}'\mathbf{v}'} = \mu_t\left\{\nabla\mathbf{v} + (\nabla\mathbf{v})^{\mathrm{T}}\right\} - \frac{2}{3}[\rho k + \mu_t(\nabla\cdot\mathbf{v})]\mathbf{I} \tag{17.17}$$

which for incompressible flows reduces to

$$\boldsymbol{\tau}^R = -\rho\overline{\mathbf{v}'\mathbf{v}'} = \mu_t\left\{\nabla\mathbf{v} + (\nabla\mathbf{v})^{\mathrm{T}}\right\} - \frac{2}{3}\rho k\mathbf{I} \tag{17.18}$$

where from now on the over bar is dropped from the averaged quantities to simplify the notation, k is the turbulent kinetic energy defined as

$$k = \frac{1}{2}\overline{\mathbf{v}'\cdot\mathbf{v}'} \tag{17.19}$$

and μ_t the turbulent eddy viscosity (in analogy with molecular viscosity), which is now flow, not fluid, dependent. With this approximation, the problem of calculating the Reynolds stress components is transformed into computing the turbulent kinetic energy and turbulent viscosity. For incompressible flows the term $-(2/3)\rho k\mathbf{I}$ in the Reynolds stress is usually combined with the pressure gradient term by defining a turbulent pressure p as

$$p \leftarrow p + \frac{2}{3}\rho k \tag{17.20}$$

thereby reducing the unknowns to μ_t alone, which is evaluated using a variety of turbulence models.

In a similar way, the turbulent thermal fluxes are calculated in analogy with Fourier's law such that

$$\dot{\mathbf{q}}^R = -\rho c_p\overline{\mathbf{v}'T'} = k_t\nabla T \tag{17.21}$$

where k_t is the turbulent thermal diffusivity calculated as will be explained later.

17.4 Turbulence Models

Several turbulence models based on the Boussinesq hypothesis have been developed to express the turbulent viscosity, μ_t, in terms of a velocity $(\sqrt{k})$ and length (ℓ) scales such that

$$\mu_t = \rho \ell \sqrt{k} \tag{17.22}$$

These models are grouped, into four main categories:

- Algebraic (Zero-Equation) Models
- One-Equation Models
- Two-Equation Models
- Second-Order Closure Models

None of the developed models is universally applicable to all flow conditions. Though each group has certain advantages and strengths.

The zero-equation models [17–19] use an algebraic equation to compute μ_t without the need to solve any differential equation. The one-equation models [20–22] require solving only one transport differential equation to compute the turbulent eddy viscosity. Two-equation turbulence models [23–35] necessitate the solution of two transport equations for the calculation of μ_t. The second-order closure models [36–41] are the most computationally expensive as separate transport equations are solved for the individual turbulent fluxes (6 equations).

The two-equation turbulence models are the most popular in terms of usage in the simulation of industrial applications, requiring the solution of two transport equations while delivering accurate enough predictions. The $k-\varepsilon$ model of Jones and Launder [23] was amongst the earliest two equation models and the most popular, while the $k-\omega$ model of Wilcox [28, 29] comes as a close second. Both models had undergone many modifications and improvements [26, 30], which have greatly extended their applicability.

17.5 Two-Equation Turbulence Models

17.5.1 Standard $k-\varepsilon$ Model

The well-known $k-\varepsilon$ model of Jones and Launder [23], known as the standard $k-\varepsilon$ model, is based on the Boussinesq approximation with the turbulent viscosity μ_t and thermal diffusivity k_t formulated as

$$\begin{aligned} \mu_t &= \rho C_\mu \frac{k^2}{\varepsilon} \\ k_t &= \frac{c_p \mu_t}{Pr_t} \end{aligned} \tag{17.23}$$

where ε is the rate of dissipation of turbulence kinetic energy per unit mass due to viscous stresses given by

$$\varepsilon = \frac{1}{2}\frac{\mu}{\rho}\overline{\left\{\nabla \mathbf{v}' + (\nabla \mathbf{v}')^T\right\} : \left\{\nabla \mathbf{v}' + (\nabla \mathbf{v}')^T\right\}} \tag{17.24}$$

In the model, the turbulent kinetic energy k and the turbulent energy dissipation rate ε are computed using

$$\frac{\partial}{\partial t}(\rho k) + \nabla \cdot (\rho \mathbf{v} k) = \nabla \cdot (\mu_{eff,k} \nabla k) + \underbrace{P_k - \rho\varepsilon}_{S^k} \tag{17.25}$$

$$\frac{\partial}{\partial t}(\rho \varepsilon) + \nabla \cdot (\rho \mathbf{v} \varepsilon) = \nabla \cdot (\mu_{eff,\varepsilon} \nabla \varepsilon) + \underbrace{C_{\varepsilon 1}\frac{\varepsilon}{k}P_k - C_{\varepsilon 2}\rho\frac{\varepsilon^2}{k}}_{S^\varepsilon} \tag{17.26}$$

where

$$\mu_{eff,k} = \mu + \frac{\mu_t}{\sigma_k} \quad \mu_{eff,\varepsilon} = \mu + \frac{\mu_t}{\sigma_\varepsilon} \tag{17.27}$$

with the turbulent Prandtl number (Pr_t) and other model constants assigned the following values: $C_{\varepsilon 1} = 1.44$, $C_{\varepsilon 2} = 1.92$, $C_\mu = 0.09$, $\sigma_k = 1.0$, $\sigma_\varepsilon = 1.3$ and $Pr_t = 0.9$.

The compact form of the production of turbulent energy term is given by

$$P_k = \boldsymbol{\tau}^R : \nabla \mathbf{v} \tag{17.28}$$

while its expanded form for incompressible flow can be obtained by multiplying Eq. (3.75) by μ_t.

In the derivation of the standard $k - \varepsilon$ model the flow is assumed to be fully turbulent and the effects of molecular viscosity to be negligible. Therefore the standard $k - \varepsilon$ model is a high Reynolds number turbulence model valid only for fully turbulent free shear flows that cannot be integrated all the way to the wall.

Modeling flows close to solid walls requires integration of the two equations over a fine grid in order to correctly capture the turbulent quantities inside the boundary layer as well as the corrections for low Reynolds number effects. A turbulence model that can be integrated all the way to the wall is denoted in the literature by a low Reynolds number turbulence model or a low Reynolds number version. Several so called low Reynolds number $k - \varepsilon$ models have been proposed over the years (see Patel et al. [42] and Wilcox [29] for a review). The idea behind their development is to damp the turbulent viscosity near the wall through the use of a damping function that tends towards zero as the distance to the wall decreases, i.e., as the wall is approached. Constants multiplying source terms in the turbulent dissipation equation are in some models also damped. All models share the same basic structure differing in the tuning of the damping functions and in some extra sources in the dissipation equation.

The only exception to this rule is the $k-\omega$ turbulence model of Wilcox [28, 29], which can be integrated all the way to the wall without the need to employ damping functions. Still the model can also be used as a high Reynolds number model.

One of the main drawback of the two-equation models is the so called stagnation point anomaly. In high strain regions the two-equation models tend to over predict the turbulence kinetic energy production P_k. This was originally recognized in stagnation point flows, but it is a more widespread anomaly. The problem is an overproduction of the turbulence kinetic energy for the case when a moderate level of k is subjected to a large rate of strain. This may be attributed to an underestimation of the sink term and/or an overestimation of the turbulent viscosity. These ideas could be merged together into a bound on the local turbulent time scale $t_s=k/\varepsilon$ (Medic and Durbin [43]). The first step is to reformulate the expression for the turbulent viscosity as

$$\mu_t = \rho C_\mu k t_s \tag{17.29}$$

Then using t_s, the ε equation is modified to

$$\frac{\partial}{\partial t}(\rho\varepsilon) + \nabla\cdot(\rho\mathbf{v}\varepsilon) = \nabla\cdot(\mu_{eff,\varepsilon}\nabla\varepsilon) + C_{\varepsilon 1}\frac{1}{t_s}P_k - C_{\varepsilon 2}\rho\frac{\varepsilon}{t_s} \tag{17.30}$$

Finally, to constrain the Reynolds stress tensor to be positive definite, a limiter is applied on t_s such that

$$\begin{aligned} t_s &= \min\left[\frac{k}{\varepsilon}, \frac{\alpha}{\sqrt{6}C_\mu S_t}\right], \\ \alpha &= 0.6 \\ S_t &= \sqrt{\mathbf{S}_t\cdot\mathbf{S}_t} \\ \mathbf{S}_t &= \frac{1}{2}\left(\nabla\mathbf{v} + \nabla\mathbf{v}^T\right) \end{aligned} \tag{17.31}$$

As a consequence, at a large rate of strain, P_k grows at the rate S_t rather than S_t^2.

17.5.2 The $k-\omega$ Model

It has already been mentioned how the family of $k-\varepsilon$ models are well behaved for free-shear flows while are likely to fail in predicting flows with adverse pressure gradient. Another class of models, for which the equation for ε is replaced by an equation for ω, where ω is the rate at which turbulence kinetic energy is converted into internal thermal energy per unit volume and time, is better capable of predicting separated flows.

The first complete turbulence model in this category was proposed by Kolmogorov [27]. In addition to the same equation for k, Kolmogorov developed a

second equation for ω. The reciprocal of ω serves as a local turbulence time scale while the turbulence length scale is given by $\sqrt{k}/\omega$. The $k-\omega$ model introduced next is proposed by Wilcox [29] as an evolution to the well-known $k-\omega$ model that Wilcox reported in [28].

The $k-\omega$ model of Wilcox [29] is similar in structure to the $k-\varepsilon$ model and is also based on the Boussinesq approximation. Two transport equations are solved to determine the two (large) scales of turbulence. The specific turbulence dissipation ω is defined as

$$\omega = \frac{\varepsilon}{C_\mu k} \tag{17.32}$$

The advantages of replacing the ε-equation by the ω-equation are: (i) the second is easier to integrate (more robust), (ii) it can be integrated through the sub-layer without the need for additional damping functions, and (iii) it performs better for flows with weak adverse pressure gradient. The conservation equations are written as

$$\frac{\partial}{\partial t}(\rho k) + \nabla \cdot (\rho \mathbf{v} k) = \nabla \cdot (\mu_{eff,k} \nabla k) + \underbrace{P_k - \beta^* \rho k \omega}_{S^k} \tag{17.33}$$

$$\frac{\partial}{\partial t}(\rho \omega) + \nabla \cdot (\rho \mathbf{v} \omega) = \nabla \cdot (\mu_{eff,\omega} \nabla \omega) + \underbrace{C_{\alpha 1} \frac{\omega}{k} P_k - C_{\beta 1} \rho \omega^2}_{S^\omega} \tag{17.34}$$

with the model constants assigned the values

$$C_{\alpha 1} = 5/9, C_{\beta 1} = 0.075, \beta^* = 0.09, \sigma_{k1} = 2, \sigma_{\omega 1} = 2, Pr_t = 0.9.$$

where

$$\begin{aligned} \mu_t &= \rho \frac{k}{\omega} \\ k_t &= \frac{\mu_t}{Pr_t} \\ \mu_{eff,k} &= \mu + \frac{\mu_t}{\sigma_{k_1}} \\ \mu_{eff,\omega} &= \mu + \frac{\mu_t}{\sigma_{\omega_1}} \end{aligned} \tag{17.35}$$

The major drawback of the Wilcox model is its sensitivity to the free stream [30] specified values, which leads to strong dependence of the solution on the arbitrary specification of the free stream ω. This dependence is not present in the $k-\varepsilon$ model.

17.5.3 The Baseline (BSL) $k-\omega$ Model

The Baseline (BSL) model developed by Menter [32] combines the $k-\varepsilon$ and $k-\omega$ models so as to take advantage of their respective strength, i.e. the robustness of the $k-\omega$ model near wall surfaces due to its simple low Reynolds number formulation and its ability to compute flows with weak adverse pressure gradients accurately, and the better performance of the $k-\varepsilon$ model near the boundary layer edge and away from walls, due to its insensitivity to the free stream values. The basis of this technique is the transformation of the $k-\varepsilon$ model to a $k-\omega$ formulation. This is an exact conversion, except for small contributions from the diffusion term due to the difference in the diffusion coefficients of the k and ε equations. The $k-\omega$ formulation of the $k-\varepsilon$ model is given by

$$\frac{\partial}{\partial t}(\rho k)+\nabla\cdot(\rho\mathbf{v}k)=\nabla\cdot(\mu_{eff,k}\nabla k)+\underbrace{P_k-\beta^*\rho k\omega}_{S^k} \tag{17.36}$$

$$\frac{\partial}{\partial t}(\rho\omega)+\nabla\cdot(\rho\mathbf{v}\omega)=\nabla\cdot(\mu_{eff,\omega}\nabla\omega)+\underbrace{C_{\alpha 2}\frac{\omega}{k}P_k-C_{\beta 2}\rho\omega^2+2\sigma_{\omega 2}\frac{\rho}{\omega}\nabla k\cdot\nabla\omega}_{S^\omega} \tag{17.37}$$

The differences between this formulation and the original $k-\omega$ model are in the additional cross-diffusion term appearing in the equation for ω and in the modeling constants that are given by

$$C_{\alpha 2}=0.4404, C_{\beta 2}=0.0828, \sigma_{k2}=1.0, \sigma_{\omega 2}=0.856 \text{ and } Pr_t=0.9.$$

The BSL $k-\omega$ model is derived by multiplying the $k-\omega$ (Eqs. 17.33 and 17.34) with a blending function F_1 and the $k-\omega$ formulation of the $k-\varepsilon$ model equations (Eqs. 17.36 and 17.37) by $(1-F_1)$, yielding the following equations for k and ω [32]:

$$\frac{\partial}{\partial t}(\rho k)+\nabla\cdot(\rho\mathbf{v}k)=\nabla\cdot(\mu_{eff,k}\nabla k)+\underbrace{P_k-\beta^*\rho k\omega}_{S^k} \tag{17.38}$$

$$\frac{\partial}{\partial t}(\rho\omega)+\nabla\cdot(\rho\mathbf{v}\omega)=\nabla\cdot(\mu_{eff,\omega}\nabla\omega)+\underbrace{\tilde{C}_\alpha\frac{\omega}{k}P_k-\tilde{C}_\beta\rho\omega^2+2(1-F_1)\sigma_{\omega 2}\frac{\rho}{\omega}\nabla k\cdot\nabla\omega}_{S^\omega} \tag{17.39}$$

These equations are formally very similar to those of the standard $k-\omega$ model, however all their coefficients depend on the blending function F_1, in the form

$$\tilde{\Phi} = F_1\Phi_1 + (1 - F_1)\Phi_2 \tag{17.40}$$

with the constants of the original $k - \omega$ model used in Eq. (17.39) given by

$$C_{\alpha 1} = 0.5976, C_{\beta 1} = 0.075, \beta^* = 0.09, \sigma_{k1} = 2, \sigma_{\omega 1} = 2, Pr_t = 0.9.$$

The blending function F_1 depends on the solution variables and on the distance $d_\perp$ from the nearest wall and is given as

$$F_1 = \tanh(\gamma_1^4) \tag{17.41}$$

where

$$\begin{aligned} \gamma_1 &= \text{Min}\left(\text{Max}\left(\frac{\sqrt{k}}{\beta^*\omega(d_\perp)}, \frac{500\nu}{(d_\perp)^2\omega}\right), \frac{4\rho\sigma_{\omega 2}k}{CD_{k\omega}(d_\perp)^2}\right) \\ CD_{k\omega} &= \text{Max}\left(2\rho\sigma_{\omega 2}\frac{1}{\omega}\nabla k \cdot \nabla\omega, 10^{-10}\right) \end{aligned} \tag{17.42}$$

and

$$\begin{aligned} \mu_t &= \rho\frac{k}{\omega} \\ k_t &= \frac{\mu_t}{Pr_t} \\ \mu_{eff,k} &= \mu + \frac{\mu_t}{\tilde{\sigma}_k} \\ \mu_{eff,\omega} &= \mu + \frac{\mu_t}{\tilde{\sigma}_\omega} \end{aligned} \tag{17.43}$$

The BSL model has a similar performance as the $k - \omega$ model for boundary layer flows and is nearly identical to the $k - \varepsilon$ model for free shear flows. Its robustness is close to that of the $k - \omega$ model.

17.5.4 The Shear Stress Transport (SST) k − ω Model

Further modifications to the BSL model yields the Shear Stress Transport (SST) [32–35] model which, when compared to other eddy-viscosity models, has an improved adverse pressure gradient performance. The first modification is related to satisfying Bradshaw's assumption, which states that the principal shear stress and the turbulent kinetic energy in the boundary layer are linearly related via an equation of the form

$$\tau_{xy} = \rho a_1 k \tag{17.44}$$

On the other hand, the principal shear stress for conventional two-equation turbulence models can be computed as

$$\tau_{xy} = \mu_t \Omega = \rho \sqrt{\frac{\text{Production of } k}{\text{Dissipation of } k}} a_1 k \tag{17.45}$$

where Ω is the vorticity. In flows with adverse pressure gradient the ratio of production of turbulent kinetic energy to its dissipation rate could be much larger than one, thereby substantially violating Bradshaw's hypothesis. For Eq. (17.44) to be satisfied within the framework of eddy-viscosity models, Menter [35] modified the turbulent viscosity μ_t in the SST $k-\omega$ model by bounding it according to

$$\mu_t = \frac{\rho a_1 k}{\text{Max}(a_1 \omega, \sqrt{2} S_t F_2)} \tag{17.46}$$

where $a_1 = 0.31$, S_t is the magnitude of the strain rate defined in Eq. (17.31), and F_2 is given by [35]

$$F_2 = \tanh(\gamma_2^2) \text{ with } \gamma_2 = \text{Max}\left(2\frac{\sqrt{k}}{\beta^* \omega (d_\perp)}, \frac{500\nu}{(d_\perp)^2 \omega}\right) \tag{17.47}$$

Moreover, to maintain the original formulation of the eddy-viscosity for free shear layers, the same blending function approach as for the baseline model is also adopted in the SST $k-\omega$ model. The k and ω equations are given by Eq. (17.38) and Eq. (17.39), respectively.

The second modification is related to the production of turbulence kinetic energy P_k in the k equation (Eq. 17.38), which is replaced by $\tilde{P}_k$ given by

$$\tilde{P}_k = \min(P_k, c_1 \varepsilon) \tag{17.48}$$

where ε is obtained from Eq. (17.32) and the blending function F_1 is calculated as in the BSL model via Eqs. (17.41) and (17.42) with the coefficients computed by Eq. (17.40) using the following model constants:

$$C_{\alpha 1} = 0.5532, C_{\beta 1} = 0.075, \beta^* = 0.09, \sigma_{k1} = 2, \sigma_{\omega 1} = 2, c_1 = 10.$$
$$C_{\alpha 2} = 0.4403, C_{\beta 2} = 0.0828, \sigma_{k2} = 1.0, \sigma_{\omega 2} = 1.186, \text{ and } Pr_t = 0.9.$$

In addition, the turbulent thermal conductivity and effective turbulent viscosities for k and ω are computed as

$$\begin{aligned} k_t &= \frac{\mu_t}{Pr_t} \\ \mu_{eff,k} &= \mu + \frac{\mu_t}{\tilde{\sigma}_k} \\ \mu_{eff,\omega} &= \mu + \frac{\mu_t}{\tilde{\sigma}_\omega} \end{aligned} \tag{17.49}$$

17.6 Summary of Incompressible Turbulent Flow Equations

The incompressible time-averaged continuity, momentum, energy, turbulence kinetic energy, turbulence dissipation rate, and specific dissipation rate equations can be written, respectively, as

$$\nabla \cdot [\rho \mathbf{v}] = 0 \tag{17.50}$$

$$\frac{\partial}{\partial t}[\rho \mathbf{v}] + \nabla \cdot \{\rho \mathbf{v}\mathbf{v}\} = \nabla \cdot \{(\mu + \mu_t)\nabla \mathbf{v}\} + \mathbf{Q}^{\mathbf{v}} \tag{17.51}$$

$$\frac{\partial}{\partial t}(\rho c_p T) + \nabla \cdot [\rho c_p \mathbf{v} T] = \nabla \cdot \left[c_p \left(\frac{\mu}{Pr} + \frac{\mu_t}{Pr_t} \right) \nabla T \right] + Q^T \tag{17.52}$$

$$\frac{\partial}{\partial t}(\rho k) + \nabla \cdot [\rho \mathbf{v} k] = \nabla \cdot [\mu_{eff,k} \nabla k] + Q^k \tag{17.53}$$

$$\frac{\partial}{\partial t}(\rho \varepsilon) + \nabla \cdot [\rho \mathbf{v} \varepsilon] = \nabla \cdot [\mu_{eff,\varepsilon} \nabla \varepsilon] + Q^\varepsilon \tag{17.54}$$

$$\frac{\partial}{\partial t}(\rho \omega) + \nabla \cdot [\rho \mathbf{v} \omega] = \nabla \cdot [\mu_{eff,\omega} \nabla \omega] + Q^\omega \tag{17.55}$$

All these equations are similar in structure and can be written in the general form given by Eq. (3.93). As such, their discretization follows the general procedures presented in previous chapters. A summary of their discretized forms is given in the next section.

17.7 Discretization of the Turbulent Flow Equations

The discretization of the unsteady, convection, and diffusion terms appearing in the conservation equations for turbulent flows, Eqs. (17.50)–(17.55), follows the procedures described in previous chapters. Moreover the discretization of the momentum and energy equations lead to the same algebraic equations given by Eqs. (15.74) and (16.29), with molecular viscosity and thermal conductivity replaced by $(\mu + \mu_t)$ and $c_p(\mu/Pr + \mu_t/Pr_t)$, respectively. As the flow is incompressible, the terms involving $\nabla \cdot \mathbf{v}$ are set to zero in these equations. Moreover, at low speeds the dissipation terms involving Φ and Ψ in the energy equation being very small are usually neglected. Another difference is related to the implementation of boundary conditions at a wall, which will be explained later. The discretized equations for k, ε and ω are presented below assuming a first order Euler scheme for the discretization of the unsteady term and a high resolution scheme for the discretization of the convection term applied in the context of a deferred correction approach.

17.7.1 The Discretized Form of the k Equation

The final algebraic form of the turbulence kinetic energy equation can be written as

$$a_C^k k_C + \sum_{F \sim NB(C)} a_F^k k_F = b_C^k \tag{17.56}$$

where the coefficients are given by

$$\begin{aligned}
a_F^k &= -(\mu_{eff,k})_f \frac{E_f}{d_{CF}} - ||-\dot{m}_f, 0|| \\
a_C^k &= a_C^{\cdot} - \sum_{F \sim NB(C)} a_F^k + \sum_{f \sim nb(C)} \dot{m}_f + \begin{cases} \rho_C \dfrac{\varepsilon_C}{k_C} V_C & k-\varepsilon \text{ model} \\ \beta^* \rho_C \omega_C V_C & k-\omega \text{ models} \end{cases} \\
a_C^{\cdot} &= \frac{\rho_C V_C}{\Delta t} \\
a_C^{\circ} &= \frac{\rho_C^{\circ} V_C}{\Delta t} \\
b_C^k &= -\sum_{f \sim nb(C)} \dot{m}_f \left(k_f^{HR} - k_f^U \right) + a_C^{\circ} k_C^{\circ} + \begin{cases} (\tilde{P}_k)_C V_C & SST\ k-\omega \text{ model} \\ (P_k)_C V_C & otherwise \end{cases} \\
&\quad + \sum_{f \sim nb(C)} (\mu_{eff,k})_f (\nabla k)_f \cdot \mathbf{T}_f
\end{aligned} \tag{17.57}$$

The discretized form of the production of turbulent kinetic energy P_k is given by Eq. (16.27) with molecular viscosity replaced by turbulent viscosity. Moreover, similar to other variables, under relaxation of the turbulence kinetic energy equation is usually required.

17.7.2 The Discretized Form of the ε Equation

The final algebraic form of the turbulence dissipation rate equation can be written as

$$a_C^{\varepsilon} \varepsilon_C + \sum_{F \sim NB(C)} a_F^{\varepsilon} \varepsilon_F = b_C^{\varepsilon} \tag{17.58}$$

where the coefficients are given by

$$
\begin{aligned}
a_F^{\varepsilon} &= -\left(\mu_{eff,\varepsilon}\right)_f \frac{E_f}{d_{CF}} - \left\|-\dot{m}_f, 0\right\| \\
a_C^{\varepsilon} &= a_C^{\cdot} - \sum_{F\sim NB(C)} a_F^{\varepsilon} + \sum_{f\sim nb(C)} \dot{m}_f + C_{\varepsilon 2}\rho_C \frac{\varepsilon_C}{k_C} V_C \\
a_C^{\cdot} &= \frac{\rho_C V_C}{\Delta t} \\
a_C^{\circ} &= \frac{\rho_C^{\circ} V_C}{\Delta t} \\
b_C^{\varepsilon} &= \sum_{f\sim nb(C)} \left(\mu_{eff,\varepsilon}\right)_f (\nabla\varepsilon)_f \cdot \mathbf{T}_f - \sum_{f\sim nb(C)} \dot{m}_f (\varepsilon_f^{HR} - \varepsilon_f^{U}) + a_C^{\circ}\varepsilon_C^{\circ} + C_{\varepsilon 1}\frac{\varepsilon_C}{k_C}(P_k)_C V_C
\end{aligned}
\tag{17.59}
$$

Moreover, similar to other variables, under relaxation of the turbulence dissipation rate equation is usually required.

17.7.3 The Discretized Form of the ω Equation

The final algebraic form of the specific turbulence dissipation equation can be written as

$$
a_C^{\omega}\omega_C + \sum_{F\sim NB(C)} a_F^{\omega}\omega_F = b_C^{\omega} \tag{17.60}
$$

where the coefficients are given by

$$
\begin{aligned}
a_F^{\omega} &= -(\mu_{eff,\omega})_f \frac{E_f}{d_{CF}} - \|-\dot{m}_f, 0\| \\
a_C^{\omega} &= a_C^{\cdot} - \sum_{F\sim NB(C)} a_F^{\omega} + \sum_{f\sim nb(C)} \dot{m}_f + a_C^{add} \\
a_C^{\cdot} &= \frac{\rho_C V_C}{\Delta t} \\
a_C^{\circ} &= \frac{\rho_C^{\circ} V_C}{\Delta t} \\
b_C^{\omega} &= \sum_{f\sim nb(C)} (\mu_{eff,\omega})_f (\nabla\omega)_f \cdot \mathbf{T}_f - \sum_{f\sim nb(C)} \dot{m}_f (\omega_f^{HR} - \omega_f^{U}) + a_C^{\circ}\omega_C^{\circ} + b_C^{add}
\end{aligned}
\tag{17.61}
$$

The expressions for a_C^{add} and b_C^{add} depend on the formulation of the $k-\omega$ model used and the different expressions are given as follows:

Original formulation of the $k-\omega$ model

$$\begin{aligned} a_C^{add} &= C_{\beta 1}\rho_C\omega_C V_C \\ b_C^{add} &= C_{\alpha 1}\frac{\omega_C}{k_C}(P_k)_C V_C \end{aligned} \tag{17.62}$$

The $k-\omega$ formulation of the $k-\varepsilon$ model

$$\begin{aligned} a_C^{add} &= C_{\beta 1}\rho_C\omega_C V_C + ||-2\sigma_{\omega 2}\frac{\rho_C}{\omega_C^2}(\nabla k\cdot\nabla\omega)_C, 0||V_C \\ b_C^{add} &= C_{\alpha 1}\frac{\omega_C}{k_C}(P_k)_C V_C + ||2\sigma_{\omega 2}\frac{\rho_C}{\omega_C}(\nabla k\cdot\nabla\omega)_C, 0||V_C \end{aligned} \tag{17.63}$$

The BSL and SST formulation of the $k-\omega$ model

$$\begin{aligned} a_C^{add} &= \tilde{\beta}\rho_C\omega_C V_C + ||-2(1-F_1)_C\,\sigma_{\omega 2}\frac{\rho_C}{\omega_C^2}(\nabla k\cdot\nabla\omega)_C, 0||V_C \\ b_C^{add} &= \tilde{\alpha}\frac{\omega_C}{k_C}(P_k)_C V_C + ||2(1-F_1)_C\,\sigma_{\omega 2}\frac{\rho_C}{\omega_C}(\nabla k\cdot\nabla\omega)_C, 0||V_C \end{aligned} \tag{17.64}$$

The discretized form of $(\nabla k\cdot\nabla\omega)_C$ is computed as

$$(\nabla k\cdot\nabla\omega)_C = \left(\frac{\partial k}{\partial x}\right)_C\left(\frac{\partial\omega}{\partial x}\right)_C + \left(\frac{\partial k}{\partial y}\right)_C\left(\frac{\partial\omega}{\partial y}\right)_C + \left(\frac{\partial k}{\partial z}\right)_C\left(\frac{\partial\omega}{\partial z}\right)_C \tag{17.65}$$

Again under relaxation of the specific turbulence dissipation equation is usually required.

17.8 Boundary Conditions

17.8.1 Modeling Flow Near the Wall

As a turbulent flow approaches a wall its mean and fluctuating components of velocity, and consequently k, vanish creating large gradients. In addition, the very high turbulent stresses away from the wall, decrease in the near wall layer to values of magnitude comparable to those of the viscous stresses. Therefore if the near wall layer is to be resolved, a substantial number of grid points will be required.

Low Reynolds number turbulence models are capable of simulating the dampening effects of the wall but at the expense of using a very large number of grid

points. This is the unavoidable cost that has to be payed if accurate solutions of the flow in the near wall region is required.

On the other hand, the high Reynolds number turbulence approach, exemplified by the standard $k - \varepsilon$ model, avoids the need to resolve the near wall layer through the use of wall functions. In this method, theoretical profiles between the boundary surface and the first near-wall node are assumed and superimposed. Compared to the previous approach, wall functions reduce significantly the computational cost. The main disadvantage of this methodology however, is related to the validity of these profiles, which are only known and justified in near-equilibrium boundary layers. Details regarding this special treatment is explained next.

17.8.2 Standard Wall Functions

The wall functions approach is based on the universal flow profiles in the boundary layer along a wall, which can be divided into three regions [1] designated by the viscous sublayer ($0 < d^+ < 5$), the buffer sublayer ($5 < d^+ < 30$), and the inertial sublayer ($30 < d^+ < 200$), respectively, with the normalized distance to the wall d^+ defined as

$$d^+ = \frac{d_\perp u_\tau}{v} = y^+ \tag{17.66}$$

where $d_\perp$ is the normal distance to the wall, v is the kinematic viscosity ($= \mu/\rho$), and u_τ is the friction velocity expressed in terms of the wall shear stress τ_w as

$$u_\tau = \sqrt{\frac{|\tau_w|}{\rho}} \tag{17.67}$$

where $|\tau_w|$ is the magnitude of the wall shear stress. Measurements and direct numerical simulations have shown that turbulence is negligible in the viscous sublayer, viscous effects are small in the inertial sublayer, while both effects are important in the buffer layer [44] with the maximum turbulent production occurring at nearly $d^+ = 12$, with the location slightly dependent on the Reynolds number making modeling of the flow in the buffer region very difficult. Because of this, turbulence models avoid the buffer layer near a wall by placing the first internal grid point either in the viscous or the inertial sublayer. The practice of placing the first grid point in the viscous sublayer is adopted with low Reynolds number turbulence models while the other practice is used with high Reynolds number turbulence models. The empirical relations applicable in the viscous sublayer are given by [45]

$$\begin{aligned} u^+ &= d^+ \\ k^+ &= 0.1 d^{+2} \\ \varepsilon^+ &= 2\frac{k^+}{d^{+2}} = 0.2 \\ \omega^+ &= \frac{6}{C_{\beta 1} d^{+2}} \end{aligned} \tag{17.68}$$

where u^+, k^+, ε^+, and ω^+ are the normalized velocity parallel to the wall, normalized turbulence kinetic energy, normalized turbulence dissipation rate, and normalized turbulence frequency, respectively. In the general case of a moving wall with a velocity $\mathbf{v}_w$, these variables are defined as

$$\begin{aligned} u^+ &= \frac{|\mathbf{v} - \mathbf{v}_w|_{||}}{u_\tau} \\ k^+ &= \frac{k}{u_\tau^2} \\ \varepsilon^+ &= \frac{\varepsilon \nu}{u_\tau^4} \\ \omega^+ &= \frac{\omega \nu}{u_\tau^2} \end{aligned} \tag{17.69}$$

where $|\mathbf{v} - \mathbf{v}_w|_{||}$ is the magnitude of the velocity parallel to the wall.

A comparison of the above profiles with data obtained from direct numerical simulation reveals that velocity and dissipation rate remain in good agreement up to $d^+ = 10$, while the turbulence kinetic energy is over estimated at values of $d^+ > 5$. Finally in the low Reynolds turbulence formulation, the $k - \omega$ based models just require a boundary treatment that satisfies the model asymptotic values. On the other hand, in the $k - \varepsilon$ based models a damping function is added for the eddy viscosity equation that mimics the direct effect of molecular viscosity on the shear stress [25]. In the inertial sub-layer, the momentum profile is derived assuming a one-dimensional Couette flow with zero pressure gradient. Profiles for the turbulence quantities can be derived for a specific turbulence model. For the $k - \varepsilon$ and $k - \omega$ models these profiles are given by

$$\begin{aligned} u^+ &= \frac{1}{\kappa} Ln(d^+) + B \\ k^+ &= \frac{1}{\sqrt{C_\mu}} = \frac{1}{\sqrt{\beta^*}} \\ \varepsilon^+ &= \frac{\nu}{u_\tau \kappa d_\perp} \\ \omega^+ &= \frac{\nu}{u_\tau \kappa d_\perp \sqrt{\beta^*}} \end{aligned} \tag{17.70}$$

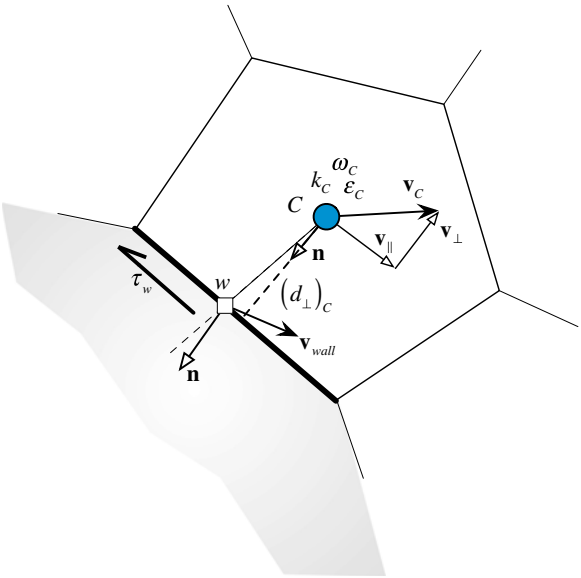

Fig. 17.2 A boundary control volume next to a wall

where the von Karman constant κ is assigned the value 0.41, $C_\mu = \beta^* = 0.09$ and $B = 5.25$. Data obtained from Direct Numerical Simulation (DNS) indicate excellent agreement for the velocity profile. On the other hand, turbulent quantities are less accurate.

As shown in Fig. 17.2, when solving a turbulent flow problem, modifications to the conservation equations are made at the first interior point C in the control volume next to the wall. The value of d^+ at that location, denoted by d_C^+, is first calculated to infer whether the point lies in the viscous or inertial sublayer. The value of d_C^+ is computed from the definition of d^+ and the value of u_τ obtained from the law of the wall as

$$\left.\begin{aligned} d_C^+ &= \frac{(d_\perp)_C u_\tau}{\nu} \\ k_C^+ = \frac{k_C}{u_\tau^2} &= \frac{1}{\sqrt{C_\mu}} \Rightarrow u_\tau = C_\mu^{1/4} k_C^{1/2} \end{aligned}\right\} \Rightarrow d_C^+ = \frac{C_\mu^{1/4} k_C^{1/2}}{\nu} (d_\perp)_C \qquad (17.71)$$

The transition from the viscous to the inertial layer is assumed to occur at a limiting value of d^+, denoted by d_{lim}^+. Different values for d_{lim}^+ are reported in different sources. All values however are between 11 and 12. A value of 11.06 is adopted here. This limiting value marks the intersection between the logarithmic and the linear profile. If $d_C^+ < d_{\text{lim}}^+$ then the grid point lies in the viscous sublayer, otherwise it is located in the inertial sublayer.

If the first grid point is in the viscous sublayer, then the flow is assumed to be laminar and the viscosity at the wall is set equal to the laminar viscosity μ and the shear stress is computed as for laminar flows. A fixed value of zero is imposed on the turbulence kinetic energy with the production of turbulence kinetic energy at the

first interior point modified by assuming that the shear stress is constant over the control volume with its value computed as

$$P_k \approx \tau_w \frac{\partial(\mathbf{v}_C - \mathbf{v}_w)_{||}}{\partial(d_\perp)}\bigg|_w = \mu \frac{(|\mathbf{v}_C - \mathbf{v}_w|_{||})^2}{(d_\perp)_C^2} \tag{17.72}$$

In the standard $k-\varepsilon$ model the dissipation rate of turbulence kinetic energy at the centroid of the first control volume next to the wall is computed by setting the laminar viscosity equal to the turbulent viscosity to yield

$$\varepsilon_C = \frac{C_\mu \rho k_C^2}{\mu} \tag{17.73}$$

whereas in the $k-\omega$ model the value of the turbulence frequency ω_C is computed from the analytical solution in the viscous sublayer as

$$\omega_C = \frac{6\nu}{C_{\beta 1}(d_\perp)_C^2} \tag{17.74}$$

If $d_C^+ > d_{\text{lim}}^+$ then the grid point is located in the inertial sublayer and the logarithmic wall functions are applied at the first interior point C. The implementation process involves computing the shear stress using the logarithmic wall function as

$$|\tau_w| = \rho u_\tau^2 = \frac{\rho u_\tau |\mathbf{v}_C - \mathbf{v}_w|_{||}}{\frac{1}{\kappa} Ln(d_C^+) + B} \Rightarrow \tau_w = -\frac{\rho u_\tau}{\frac{1}{\kappa} Ln(d_C^+) + B} (\mathbf{v}_C - \mathbf{v}_w)_{||} \tag{17.75}$$

where the fact that

$$\tau_w = -|\tau_w| \frac{(\mathbf{v}_C - \mathbf{v}_w)_{||}}{|\mathbf{v}_C - \mathbf{v}_w|_{||}} \tag{17.76}$$

has been used. This shear stress is used in solving the momentum equation either by invoking its value directly as a source term ($\tau_w S_b$) or via a modified viscosity at the wall μ_w computed as

$$|\tau_w| = \mu_w \frac{|\mathbf{v}_C - \mathbf{v}_w|_{||}}{(d_\perp)_C} = \frac{\rho u_\tau |\mathbf{v}_C - \mathbf{v}_w|_{||}}{\frac{1}{\kappa} Ln(d_C^+) + B} \Rightarrow \mu_w = \frac{\rho u_\tau (d_\perp)_C}{\frac{1}{\kappa} Ln(d_C^+) + B} \tag{17.77}$$

with the vector form of the wall shear stress expressed as

$$\tau_w = -\frac{\mu_w}{(d_\perp)_C} (\mathbf{v}_C - \mathbf{v}_w)_{||} \tag{17.78}$$

In either case the implementation follows the procedures described in Chap. 15.

It is worth noting that the shear stress can also be formulated in terms of the normalized quantities as

$$\begin{aligned} |\tau_w| &= \mu_{lam} \frac{|\mathbf{v}_C - \mathbf{v}_w|_{||}}{(d_\perp)_C} \frac{d^+}{u^+} \\ &= \tau_{lam} \frac{d^+}{u^+} \end{aligned} \tag{17.79}$$

In solving the turbulence kinetic energy equation, the value of k is assumed to prevail over the control volume (i.e., a zero gradient for k is used) with the term representing the production of the turbulence kinetic energy at the first interior point next to the wall modified by assuming that the shear stress is constant over the control volume with its value equal to that at the wall, while the velocity gradient is computed from the wall function as

$$\begin{aligned} u^+ = \frac{1}{\kappa} ln(d^+) + B \Rightarrow \frac{|\mathbf{v} - \mathbf{v}_w|_{||}}{u_\tau} &= \frac{1}{\kappa} ln\left(\frac{d_\perp u_\tau}{\nu}\right) + B \\ \Rightarrow \left.\frac{d\left(|\mathbf{v} - \mathbf{v}_w|_{||}\right)}{d(d_\perp)}\right|_w &= \frac{u_\tau}{\kappa (d_\perp)_C} \end{aligned} \tag{17.80}$$

Therefore, if the first interior point lies in the inertial sublayer, the production term in the turbulence kinetic energy equation is computed as

$$P_k = |\tau_w| \frac{u_\tau}{\kappa (d_\perp)_C} \tag{17.81}$$

It is also useful to formulate the production term P_k in terms of the normalized parameters. Starting with Eq. (17.72), the generic production term can be written as

$$\begin{aligned} P_k &\approx \tau_w \left.\frac{\partial (\mathbf{v}_C - \mathbf{v}_w)_{||}}{\partial (d_\perp)}\right|_w \\ &= \rho u_\tau^2 \left.\frac{\partial (\mathbf{v}_C - \mathbf{v}_w)_{||}}{\partial (d_\perp)}\right|_w \\ &= \rho u_\tau^2 \frac{\partial u^+}{\partial d^+} \frac{u_\tau^2}{\mu_{lam}} \\ &= \frac{\tau_{lam}^2}{\mu_{lam}} \left(\frac{d^+}{u^+}\right)^2 \frac{\partial u^+}{\partial d^+} \end{aligned} \tag{17.82}$$

In the $k - \varepsilon$ model the ε equation is not solved at the first interior point next to the wall. Rather its value is set by requiring the dissipation of turbulence kinetic energy to be equal to its production rate such that

$$\left.\begin{aligned} \rho\varepsilon_C = P_k = |\tau_w|\frac{u_\tau}{\kappa(d_\perp)_C} = \frac{\rho u_\tau^3}{\kappa(d_\perp)_C} \\ u_\tau = C_\mu^{1/4} k_C^{1/2} \end{aligned}\right\} \Rightarrow \varepsilon_C = \frac{C_\mu^{3/4} k_C^{3/2}}{\kappa(d_\perp)_C} \tag{17.83}$$

If the $k-\omega$ model is used with wall functions, the same procedure is used. For the k equation the same modified production term is obtained since $C_\mu = \beta^*$. The ω equation is not solved for the first interior point and its value is set again by requiring dissipation of turbulence kinetic energy to be equal to its production rate such that

$$\left.\begin{aligned} \rho\varepsilon_C = P_k = \rho\frac{C_\mu^{3/4} k_C^{3/2}}{\kappa(d_\perp)_C} \\ \omega_C = \frac{\varepsilon_C}{C_\mu k_C} \end{aligned}\right\} \Rightarrow \omega_C = \frac{k_C^{1/2}}{\kappa C_\mu^{1/4}(d_\perp)_C} \tag{17.84}$$

17.8.3 Improved Wall Functions

The above formulation is valid under local equilibrium conditions and results in a zero viscosity (Eq. 17.77) when τ_w vanishes, since $u_\tau = \sqrt{\tau_w/\rho}$, as is the case at reattachment and separating points. A generalization to the above formulation valid under local non-equilibrium conditions has been proposed for the $k-\varepsilon$ model by Launder and Spalding [46]. In their work, $\sqrt{k}$ is used as the characteristic turbulent velocity scale, instead of the friction velocity, through the identity

$$u^* = C_\mu^{1/4}\sqrt{k} \tag{17.85}$$

such that the wall viscosity and shear stress become

$$\begin{aligned} \mu_w &= \frac{\rho u^*(d_\perp)_C}{\frac{1}{\kappa}Ln(d_C^*) + B} \\ \tau_w &= \rho u_\tau u^* \\ u_\tau &= \frac{|\mathbf{v}_C - \mathbf{v}_w|_{||}}{\frac{1}{\kappa}Ln(d_C^*) + B} \end{aligned} \tag{17.86}$$

where d_C^* is defined using u^* as

$$d_C^* = \frac{(d_\perp)_C u^*}{\nu} \tag{17.87}$$

Equation (17.86) clearly indicates that the turbulent viscosity does not vanish when $\tau_w = 0$.

Similar to the standard wall functions case, the value of d_C^* is first computed. If the value of $d_C^* < d_{\text{lim}}^* = 11.06$ then the first point is in the viscous sublayer and the procedure described for the standard wall functions is used.

If the value of $d_C^* > d_{\text{lim}}^*$ the first interior grid point is located in the inertial sublayer and the wall viscosity is computed via Eq. (17.86). Using this viscosity the wall shear stress is obtained and implemented as explained earlier. To find k_C, the k-conservation equation is solved. The procedure described with the standard wall functions for calculating the production and dissipation terms in the near wall control volume, assumed that the values of k_C and ε_C prevail over the entire control volume. Since the P_k and $\boldsymbol{\varepsilon}$ values vary drastically across the near-wall cell, evaluating them at the cell centre in discretizing the k equation leads to inaccurate approximations. To improve predictions, Launder and Spalding [46] suggested suitable approximations for their cell-average values. Starting with the production term, its average value is computed as

$$\begin{aligned}\overline{P}_k &= \frac{1}{(d_\perp)_C}\int_0^{(d_\perp)_C} P_k d(d_\perp) \\ &= \frac{1}{(d_\perp)_C}\int_0^{(d_\perp)_C} |\tau_w|\frac{d(|\mathbf{v}-\mathbf{v}_w|_{||})}{d[(d_\perp)]}d(d_\perp) = \frac{1}{(d_\perp)_C}|\tau_w||\mathbf{v}_C-\mathbf{v}_w|_{||} = \mu_w\frac{\left[|\mathbf{v}_C-\mathbf{v}_w|_{||}\right]^2}{(d_\perp)_C^2}\end{aligned} \tag{17.88}$$

Invoking Eq. (17.86), the average production term is found to be

$$\overline{P}_k = \frac{\rho C_\mu^{1/4}\sqrt{k_C}}{(d_\perp)_C\left(\frac{1}{\kappa}Ln(d_C^*)+B\right)}\left[|\mathbf{v}_C-\mathbf{v}_w|_{||}\right]^2 \tag{17.89}$$

Then, the volume integral of the production term is obtained as

$$\int_V P_k dV = \overline{P}_k V_C \tag{17.90}$$

To calculate an average turbulence dissipation rate, the integrated value of dissipation is set equal to the production rate resulting in

$$\int_0^{(d_\perp)_C}\rho\varepsilon d(d_\perp) = \int_0^{(d_\perp)_C} P_k d(d_\perp) = \int_0^{(d_\perp)_C}|\tau_w|\frac{d\left(|\mathbf{v}-\mathbf{v}_w|_{||}\right)}{d[(d_\perp)]}d(d_\perp) \tag{17.91}$$

The integral is evaluated by expressing the shear stress and velocity gradients as

$$|\tau_w| = \rho u_\tau^2$$
$$\frac{d\left(|\mathbf{v}-\mathbf{v}_w|_{||}\right)}{d[(d_\perp)]} \approx \frac{\Delta\left(|\mathbf{v}-\mathbf{v}_w|_{||}\right)}{\Delta[(d_\perp)]} = \frac{|\mathbf{v}_C-\mathbf{v}_w|_{||}}{(d_\perp)_C} = \frac{u_\tau}{(d_\perp)_C}\left(\frac{1}{\kappa}Ln(d_C^*)+B\right) \tag{17.92}$$

Thus

$$\begin{aligned}\int_0^{(d_\perp)_C} \rho\varepsilon d(d_\perp) &= \int_0^{(d_\perp)_C} \rho u_\tau^2 \frac{u_\tau}{(d_\perp)_C}\left(\frac{1}{\kappa}Ln(d_C^*)+B\right)d(d_\perp) \\ &= \rho u_\tau^3\left(\frac{1}{\kappa}Ln(d_C^*)+B\right) = \rho C_\mu^{3/4}k^{3/2}\left(\frac{1}{\kappa}Ln(d_C^*)+B\right)\end{aligned} \tag{17.93}$$

and the average turbulence dissipation rate is computed as

$$\overline{\varepsilon_C} = \frac{1}{\rho(d_\perp)_C}\int_0^{(d_\perp)_C} \rho\varepsilon d(d_\perp) = \frac{C_\mu^{3/4}k_C^{3/2}}{(d_\perp)_C}\left(\frac{1}{\kappa}Ln(d_C^*)+B\right) \tag{17.94}$$

while its volume integral is found to be

$$\int_V \rho\varepsilon dV = \rho\overline{\varepsilon_C}V_C \tag{17.95}$$

As for the standard wall functions, the ε equation is not solved at the first interior point next to the wall and its value is set using Eq. (17.83). Moreover, similar equations can be developed for use with the $k-\omega$ model by transforming ε into ω via Eq. (17.32).

17.8.4 Scalable Wall Functions

The wall functions approach is most accurate when the first grid point in the near wall region lies in the inertial sublayer at a normalized distance $d^* > d_{low}^*$ where, depending on the numerical formulation, $d_{low}^* \approx 20$. This represents a serious limitation in situations where the boundary layer is very thin as it cannot be resolved with a coarse near-wall grid. The scalable wall function approach developed in [47, 48] overcomes this hurdle by slightly modifying the calculation of the wall shear stress by redefining u^+ as

$$u^+ = \frac{1}{\kappa}\ln\tilde{d}^* + B \tag{17.96}$$

where

$$\tilde{d}^* = \max(d^*, d^*_{\text{lim}}) \quad d^*_{\text{lim}} \approx 11.06 \tag{17.97}$$

With this formulation, the definition of $\tilde{d}^*$ becomes independent of the grid spacing as it prevents the first grid point from being located in the viscous sublayer leading to consistent results for grids of arbitrary refinement. The simulation error introduced is related to not accounting for the viscous sublayer, which is the case for all wall function formulations. It should be clarified however that this error can be significant for flows with relatively low Reynolds number. The implementation of the scalable wall functions is straightforward whereby the procedures described above remains unchanged with the exception of replacing d^* or d^+ by, respectively, $\tilde{d}^*$ or $\tilde{d}^+$.

This approach is usually used when the details of the boundary layer is not of interest. If the purpose of using a fine grid near the wall is to examine the details of the boundary layer then a low Reynolds number model should be used as explained next.

17.8.5 Wall Boundary Conditions for Low Reynolds Number Models

With low Reynolds number turbulence models a fine grid resolution must be used in the near wall region in order to properly resolve the viscous sublayer allowing laminar flow boundary conditions to be applied. Therefore, the boundary condition to be used at the wall in the momentum equation is the no slip condition presented in Chap. 15 with no need to be repeated here.

The $k-\omega$ and $k-\varepsilon$ turbulence models and their variants are examples of this type. For all these models, the asymptotic boundary conditions applicable for k, ε and ω are set to

$$\begin{aligned} k_w &\to 0 \\ \varepsilon_w &\to 2\frac{vk}{(d_\perp)^2_C} \\ \omega_w &\to 10\frac{6v}{C_{\beta 1}(d_\perp)^2_C} \end{aligned} \tag{17.98}$$

The value for ω_w given by Eq. (17.98), which is intended to be applied at the first interior grid point next to the wall, is valid for grid points located at d^+ or $d^* < 2.5$ with 15–20 nodes needed in that region for grid independent solutions. The factor of 10 is introduced into Eq. (17.98) based on the recommendation of Menter as it eliminates the need to specify ω at any other internal point beyond the one next to the wall.

17.8.6 Automatic Near-Wall Treatment

With the wall functions method the normalized distance to the wall is first calculated to infer wether the first interior point lies in the viscous or inertial sublayer. In the viscous sublayer the flow is treated as laminar. Because the standard $k-\varepsilon$ model is valid in the fully turbulent regions and does not include damping functions to model viscous effects it introduces error into the solution. To eliminate this error care should be exercised to make sure that the first interior point is in the log region. This however eliminates the influence of the viscous sublayer, which may be significant on the solution. Therefore it is desirable to have the option of resolving the viscous sublayer without the stringent requirement of a very fine grid near the wall. This is possible with the $k-\omega$ model and its variants as the ω equation can be integrated all the way to the wall without the need for any additional damping functions. The idea is to develop a method that switches automatically between the low and the high Reynolds number formulation based on the value of the normalized distance to the wall. This task is achievable by the ω equation because of its known analytical solutions in the viscous and inertial sublayers. For that purpose, both solutions are blended smoothly according to [34]

$$\omega = \left[\omega_{vis}^2 + \omega_{\log}^2\right]^{0.5} \tag{17.99}$$

where

$$\omega_{vis} = \frac{6\nu}{C_{\beta 1} d_\perp^2} \quad \omega_{\log} = \frac{u^*}{\kappa d_\perp \sqrt{\beta^*}} = \frac{(u^*)^2}{\kappa \nu d^* \sqrt{\beta^*}} \tag{17.100}$$

The shear stress in the momentum equation is computed as

$$|\tau_w| = \rho u_\tau u^* = \rho \frac{u^*}{u^+} |\mathbf{v}_C - \mathbf{v}_w|_{||} \tag{17.101}$$

with the values of the velocities u_τ and u^* in the near wall region calculated using

$$u_\tau^{vis} = \sqrt{\frac{\mu}{\rho} \frac{|\mathbf{v}_C - \mathbf{v}_w|_{||}}{(d_\perp)_C}} \quad u_\tau^{\log} = \frac{|\mathbf{v}_C - \mathbf{v}_w|_{||}}{\frac{1}{\kappa}\ln d_C^+ + B} \quad u_\tau = \left[\left(u_\tau^{vis}\right)^4 + \left(u_\tau^{\log}\right)^4\right]^{0.25} \tag{17.102}$$

$$u_{vis}^* = \sqrt{\frac{\mu}{\rho} \frac{|\mathbf{v}_C - \mathbf{v}_w|_{||}}{(d_\perp)_C}} \quad u_{\log}^* = \beta^{*1/4} k^{1/2} \quad u^* = \left[\left(u_{vis}^*\right)^4 + \left(u_{\log}^*\right)^4\right]^{0.25} \tag{17.103}$$

In solving the k equation, the gradient at the wall is set to zero and the production in the near wall cell is altered to

$$P_k = |\tau_w| \frac{u_\tau}{\kappa (d_\perp)_C} = \rho \left(\frac{u^*}{u^+}\right)^2 \frac{\partial u^+}{\partial d^+} \left(|\mathbf{v}_C - \mathbf{v}_w|_{||}\right)^2 \tag{17.104}$$

Based on the grid spacing, this blending approach permits a smooth shift of the wall treatment from a viscous sublayer to a wall function.

17.8.7 Near-Wall Heat Transfer

Similar to velocity profiles, near wall temperature profiles should also be corrected when high Reynolds number turbulence models are used. These profiles are obtained by an adjusted Reynolds analogy from the velocity profiles by altering the log-law to provide an equation that relates the value of temperature at the near wall node T_C, to that at the wall T_w, and to the wall heat flux q_w. The procedure starts by defining a normalized temperature T^+ as

$$T^+ = \frac{T_w - T}{T^*} \tag{17.105}$$

where T^* is the modified friction temperature given by

$$T^* = \frac{q_w}{\rho c_p u^*} \tag{17.106}$$

and other variables are as defined earlier. Then combining the above two equations, the normalized temperature equation becomes

$$T^+ = \frac{\rho c_p u^*}{q_w}(T_w - T) \tag{17.107}$$

For the standard wall functions formulation, if the first interior point is located in the viscous sublayer then the normalized temperature is calculated as

$$T^+ = Pr\, d^* \tag{17.108}$$

whereas if it lies in the inertial sublayer then it is computed using the law of the wall as

$$T^+ = 2.12Ln(d^*) + \beta(Pr) \tag{17.109}$$

where

$$\beta(Pr) = (3.85Pr^{1/3} - 1.3)^2 + 2.12Ln(Pr) \tag{17.110}$$

and Pr is the laminar Prandtl number. For the scalable wall functions formulation the same equation is used with d^+ replaced by $\tilde{d}^+$, as explained earlier. With the automatic near wall treatment approach, the equation suggested by Kader [49], which blends the viscous sublayer with the law of the wall, is used. In fact this

approach may also be used with the standard and scalable wall functions approaches. According to this formula the normalized temperature is computed as

$$T^+ = Pr\, d^* e^{-\Gamma} + [2.12Ln(1 + d^*) + \beta(Pr)]e^{-1/\Gamma} \tag{17.111}$$

with the blending function Γ given by

$$\Gamma = \frac{0.01(Pr\, d^*)^4}{1 + 5Pr^3 d^*} \tag{17.112}$$

Moreover, Eq. (17.107) can be interpreted in two different ways depending on the physical boundary condition imposed on temperature. For a given wall heat flux boundary condition, the numerical condition for T_w will be a fixed value, iteratively updated, given by

$$\begin{aligned} T_w &= T_C + \frac{q_w T_C^+}{\rho c_p u^*} \\ &= T_C + \frac{q_w T_C^+}{\rho c_p C_\mu^{1/4} \sqrt{k_C}} \end{aligned} \tag{17.113}$$

If, instead, temperature is imposed on the boundary, then the temperature gradient must be held fixed numerically and calculated as

$$\begin{aligned} q_w &= \frac{\rho c_p u^*}{T_C^+}(T_w - T_C) \\ &= \frac{\rho c_p C_\mu^{1/4} \sqrt{k_C}}{T_C^+}(T_w - T_C) \end{aligned} \tag{17.114}$$

17.8.8 Other Boundary Conditions

Beside walls, conditions at other boundaries are needed. This include inlet, outlet, and symmetry boundary conditions. At inlet to a domain the values of the turbulence kinetic energy and dissipation rate, which are usually unknown, are required. If the values are known from measurements then they should be used, otherwise they must be estimated. The easiest way is to assign the values for k and ε. Values may also be specified via the turbulence intensity I defined as

$$I = \frac{\sqrt{\overline{\mathbf{v}' \cdot \mathbf{v}'}}}{\sqrt{\mathbf{v} \cdot \mathbf{v}}} \tag{17.115}$$

from which the turbulence kinetic energy is obtained as

$$k = \frac{1}{2} I^2 (\mathbf{v} \cdot \mathbf{v}) \tag{17.116}$$

A value of turbulence intensity between 1 and 10 % is usually used with values less than 1 % considered low while values higher than 10 % considered high. The values of the turbulence dissipation rate ε and turbulence frequency ω are computed from Eqs. (17.23) and (17.32) by specifying a turbulence length scale whose value is dependent on the largest eddy dimension but it is usually set to one-tenth of the width of a shear layer or the domain size, with their expressions given by

$$\begin{aligned} \varepsilon &= C_\mu \frac{k^{3/2}}{\ell} \\ \omega &= \frac{k^{1/2}}{\ell} \end{aligned} \tag{17.117}$$

The value for ε and ω may also be computed from knowledge of k and the turbulent to laminar viscosity ratio using

$$\begin{aligned} \varepsilon &= C_\mu \rho \frac{k^2}{\mu} \left(\frac{\mu}{\mu_t} \right) \\ \omega &= \rho \frac{k}{\mu} \left(\frac{\mu}{\mu_t} \right) \end{aligned} \tag{17.118}$$

At an outlet and a symmetry boundary, the treatment for k, ε and ω is similar to that for the general scalar variable ϕ presented in previous chapters and is deemed unnecessary to be repeated.

17.9 Calculating Normal Distance to the Wall

In the BSL and SST turbulence models, the normal distance to the nearest wall $d_\perp$ is needed over the entire domain to determine the interface between the computation regions of $k - \varepsilon$ and $k - \omega$, as reflected by Eq. (17.42) for the BSL model and (17.47) for the SST model. Search procedures to compute $d_\perp$ are computationally expensive in three-dimensional situations even for fixed grids. The situation gets worse with a moving grid as the search has to be repeated at every time step. This has forced workers to introduce approximations in the calculation of $d_\perp$ that incur large errors.

To avoid the expensive search procedures, techniques based on solving differential equations for $d_\perp$ have been developed. These methods are based on solving a Poisson, Eikonal, or Hamilton-Jacobi equations [50–53]. The attraction in these equations is their composition which involves gradient and/or Laplacian operators

that are readily available in CFD solvers. This renders the approach easy to implement, stable, and economical especially with moving grids.

Adopting a Poisson-like approach, the following differential equation for a variable ϕ is solved:

$$\nabla^2\phi = -1 \tag{17.119}$$

subject to

$$\begin{cases} \phi = 0 & \text{on walls} \\ \nabla\phi \cdot \mathbf{n} = 0 & \text{elsewhere} \end{cases} \tag{17.120}$$

The normal distance to the nearest wall is computed using the predicted value of ϕ and its gradient as

$$\begin{aligned} d_\perp &= -|\nabla\phi| + \sqrt{|\nabla\phi|^2 + 2\phi} \\ &= -\sqrt{\left(\frac{\partial\phi}{\partial x}\right)^2 + \left(\frac{\partial\phi}{\partial y}\right)^2 + \left(\frac{\partial\phi}{\partial z}\right)^2} + \sqrt{\left(\frac{\partial\phi}{\partial x}\right)^2 + \left(\frac{\partial\phi}{\partial y}\right)^2 + \left(\frac{\partial\phi}{\partial z}\right)^2 + 2\phi} \end{aligned} \tag{17.121}$$

While solving a turbulent flow problem, Eq. (17.119) is discretized on the grid network generated for solving the problem using the finite volume method and following the procedures described in Chap. 8. During the solution procedure, a converged solution for Eq. (17.119) is first obtained, from which the normal distances to the wall are computed, prior to solving the turbulent flow problem. Referring to Fig. 17.3, and recalling that the term $gDiff_f$ is defined as

$$gDiff_f = \frac{E_f}{d_{CF}} \tag{17.122}$$

the discretized form of Eq. (17.119) can be written as

$$a_C\phi_C + \sum_{F\sim NB(C)} a_F\phi_F = b_C \tag{17.123}$$

where

$$\begin{aligned} a_F &= FluxF_f = -gDiff_f \\ a_C &= \sum_{f\sim NB(C)} FluxC_f = -\sum_{f\sim NB(C)} FluxF_f = \sum_{f\sim NB(C)} gDiff_f \\ b_C &= V_C + \sum_{f\sim nb(C)} \left((\nabla\phi)_f \cdot \mathbf{T}_f\right) \end{aligned} \tag{17.124}$$

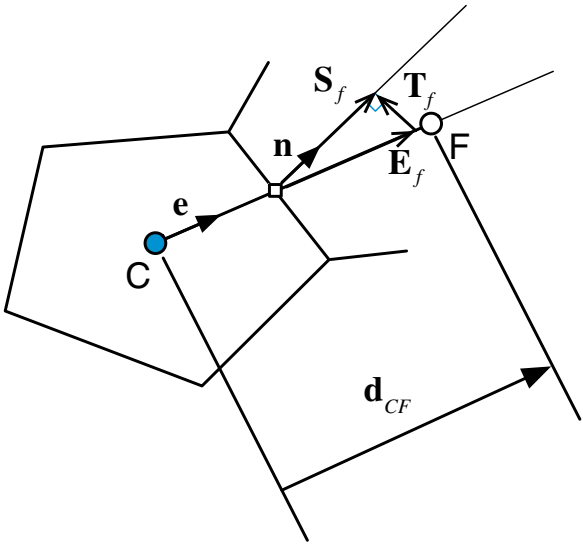

Fig. 17.3 A two-dimensional control volume with its geometrical quantities

Moreover, the Dirichlet and Von Neumann boundary conditions described by Eq. (17.120) are treated as explained in Chap. 8.

Following the solution for Eq. (17.119), the normal distance to the wall at all cell centroids in the domain are calculated using Eq. (17.121) with the gradient calculated as detailed in Chap. 9.

17.10 Computational Pointers

OpenFOAM® [54] implements several LES and RANS turbulence models for both compressible and incompressible flows. The root directory for all models is denoted by "FOAM_SRC/turbulenceModels". Incompressible turbulence models are located in the sub-directory "FOAM_SRC/turbulenceModels/incompressible" within which the three sub-sub-directories "LES" (refers to the large eddy simulation approach), "RAS" (refers to the Reynolds-averaged Navier-Stokes approach), and "turbulenceModel" reside. The first two sub-sub-directories "LES" and "RAS" define the special features of the LES and RAS models, respectively, while in "turbulenceModel" the abstract base classes of both incompressible RAS and LES models are defined. The base class defines a series of abstract virtual functions that have to be specified for any derived class as shown in Listing 17.1.

```
        //- Return the turbulence viscosity
        virtual tmp<volScalarField> nut() const = 0;

        //- Return the effective viscosity
        virtual tmp<volScalarField> nuEff() const = 0;

        //- Return the turbulence kinetic energy
        virtual tmp<volScalarField> k() const = 0;
…
        //- Return the turbulence kinetic energy dissipation rate
        virtual tmp<volScalarField> epsilon() const = 0;

        //- Return the Reynolds stress tensor
        virtual tmp<volSymmTensorField> R() const = 0;
```

Listing 17.1 Script used to define virtual functions

The base class also defines the normal distance to the wall, which is a useful quantity for turbulence models that can be used with all derived classes. The statement used for that is given in Listing 17.2 as

```
        //- Return the near wall distances
        const nearWallDist& y() const
        {
            return y_;
        }
```

Listing 17.2 Statement used to define the normal distance to the wall

To better understand the code structure that defines the OpenFOAM® turbulence models, in the following the $k - \varepsilon$ model with the Spalding wall functions and the SST $k - \omega$ model are used as examples.

The model definition can be found in the directory "FOAM_SRC/turbulenceModels/incompressible/RAS" where all the RANS turbulence models are placed and defined. For the RANS models, OpenFOAM® defines an additional non virtual base class named RASModel (deriving the turbulenceModel class) from which all models are derived (Listing 17.3).

```
class RASModel
:
    public turbulenceModel,
    public IOdictionary
{
…
        //- Allow omegaMin to be changed
        dimensionedScalar& omegaMin()
        {
            return omegaMin_;
        }
        //- Const access to the coefficients dictionary
        virtual const dictionary& coeffDict() const
        {
                return coeffDict_;
        }
        //- Return the effective viscosity
        virtual tmp<volScalarField> nuEff() const
        {
            return tmp<volScalarField>
            (
                new volScalarField("nuEff", nut() + nu())
            );
        }
```

Listing 17.3 Script used to define the non virtual base class RASModel

This class is just a wrapper mainly for bounding the values of the effective viscosity and turbulence quantities, as well as a dictionary definition.

17.10.1 The $k - \varepsilon$ Model

The "kEpsilon" class implements the standard version of the $k - \varepsilon$ model given by Eqs. (17.25) and (17.26). It defines the necessary constants and set of variables used in the model, as shown in Listing 17.4.

```
class kEpsilon
:
    public RASModel
{
protected:

    // Protected data
        // Model coefficients
            dimensionedScalar Cmu_;
            dimensionedScalar C1_;
            dimensionedScalar C2_;
            dimensionedScalar sigmaEps_;
        // Fields
            volScalarField k_;
            volScalarField epsilon_;
            volScalarField nut_;
```

Listing 17.4 Script defining the $k - \varepsilon$ model

It is selectable from the "RASProperties" dictionary with the name "kEpsilon" defined under the "TypeName" of the class as (Listing 17.5)

```
    //- Runtime type information
    TypeName("kEpsilon");
```

Listing 17.5 Statement used to select the $k-\varepsilon$ model

Being derived from the base virtual class "turbulenceModel", all its virtual base functions have to be defined as follows (Listing 17.6).

```
   //- Return the effective stress tensor including the laminar stress
        virtual tmp<volSymmTensorField> devReff() const;

   //- Return the source term for the momentum equation
      virtual tmp<fvVectorMatrix> divDevReff(volVectorField& U) const;
```

Listing 17.6 Statements used to define the effective stress tensor and the diffusion field in the momentum equation

The "divDevReff" function returns the diffusion contribution in the momentum equation including the Reynolds stress, i.e.,

$$\text{divDevReff}(\mathbf{v}) = \left[\nabla \cdot \left(\overline{\boldsymbol{\tau}} - \rho\overline{\mathbf{v}'\mathbf{v}'}\right)\right] \tag{17.125}$$

In this case the type of data returned by the function (Listing 17.7) is in the form of an fvMatrix defined as

```
tmp<fvVectorMatrix> kEpsilon::divDevReff(volVectorField& U) const
{
    return
    (
      - fvm::laplacian(nuEff(), U)
      - fvc::div(nuEff()*dev(T(fvc::grad(U))))
    );
}
```

Listing 17.7 Implicit and explicit contributions of the Laplacian term

in which the Laplacian of the velocity field is split into an implicit and an explicit matrix contribution. It is important to mention that the divDevReff term depends only on the velocity gradient field and the total or effective viscosity (nuEff()). This

means that OpenFOAM® implements the wall shear stress contribution by modifying only the turbulent viscosity at the wall; thus implementing Eq. (17.77).

The assembly and solution of the turbulence model equations are defined via the "correct()" member function as in Listing 17.8.

```
void kEpsilon::correct()
{
…
```

Listing 17.8 Assembly and solution of the turbulence model

Here OpenFOAM® first assembles and then solves the ε equation before the k equation, using the script in Listing 17.9 as

```
  volScalarField G(GName(), nut_*2*magSqr(symm(fvc::grad(U_))));
 // Update epsilon and G at the wall
  epsilon_.boundaryField().updateCoeffs();
  // Dissipation equation
  tmp<fvScalarMatrix> epsEqn
  (
      fvm::ddt(epsilon_)
    + fvm::div(phi_, epsilon_)
    - fvm::laplacian(DepsilonEff(), epsilon_)
   ==
      C1_*G*epsilon_/k_
    - fvm::Sp(C2_*epsilon_/k_, epsilon_)
  );
  epsEqn().relax();
  epsEqn().boundaryManipulate(epsilon_.boundaryField());
  solve(epsEqn);
  bound(epsilon_, epsilonMin_);
```

Listing 17.9 Assemble and solve the ε equation

where the G field defines the global production term previously defined in Eq. (17.28) as P_k. Based on the wall functions approach, it is necessary before solving the ε equation, to modify the value of ε at all centroids of cells attached to the wall using Eqs. (17.81) and (17.83). In OpenFOAM® the values of ε and P_k are altered at the wall by defining a special wall boundary condition for the field ε with the modification forced through the function "epsilon_.boundaryField(). updateCoeffs();" with the dedicated boundary definition for the dissipation rate ε found under the directory "FOAM_SRC/turbulenceModels/incompressible/RAS/derivedFvPatchFields/wallFunctions/epsilonWallFunctions/epsilonWallFunction".

Based on the wall function model this class has to update the values of the two variables ε and P_k. This operation is performed by the function "calculate" in which,

after defining all necessary variables, a loop cycle changes the corresponding values (with G in Listing 17.10 corresponding to the P_k production term). The "w" variable is just a weight factor to take into account boundary cells in contact with more than one wall (i.e., corners). For standard faces "w" has a value of one.

```
label cellI = patch.faceCells()[faceI];
scalar w = cornerWeights[faceI];

epsilon[cellI] += w*Cmu75*pow(k[cellI], 1.5)/(kappa_*y[faceI]);

G[cellI] +=
    w
   *(nutw[faceI] + nuw[faceI])
   *magGradUw[faceI]
   *Cmu25*sqrt(k[cellI])
   /(kappa_*y[faceI]);
```

Listing 17.10 Calculating ε and modifying the production of turbulence kinetic energy in the near wall cells

It is important to note that while G or P_k is a source term, ε is a field that is solved by a transport equation. To impose in a cell a value previously calculated, the matrix has to be manipulated at the right location, in order to return the correct value.Thus the class "epsilonWallFunctionFvPatchScalarField" is derived, as shown in Listing 17.11, from the class "fixedInternalValueFvPatchField" according to

```
class epsilonWallFunctionFvPatchScalarField
:
    public fixedInternalValueFvPatchField<scalar>
{
…
```

Listing 17.11 Creating the class "epsilonWallFunctionFvPatchScalarField" from the class "fixedInternalValueFvPatchField"

with class "fixedInternalValueFvPatchField", shown in Listing 17.12, being just a wrapper class containing a special function to manipulate the matrix by imposing the expected values of the variable in a list of cells.

```
template<class Type>
class fixedInternalValueFvPatchField
:
    public zeroGradientFvPatchField<Type>
…
template<class Type>
void Foam::fixedInternalValueFvPatchField<Type>::manipulateMatrix
(
    fvMatrix<Type>& matrix
)
{

    // Apply the patch internal field as a constraint in the matrix
    matrix.setValues(this->patch().faceCells(), this->patchInternalField());
}
```

Listing 17.12 The functionality of the "fixedInternalValueFvPatchField" class

This function is called form form the "kEpsilon::correct()" class after assembling the matrix with "epsEqn().boundaryManipulate(epsilon_.boundaryField());".

Once the ε equation is solved, OpenFOAM® proceeds to assembling and solving the k equation, as shown in Listing 17.13.

```
    // Turbulent kinetic energy equation
    tmp<fvScalarMatrix> kEqn
    (
        fvm::ddt(k_)
      + fvm::div(phi_, k_)
      - fvm::laplacian(DkEff(), k_)
     ==
        G
      - fvm::Sp(epsilon_/k_, k_)
    );
    kEqn().relax();
    solve(kEqn);
    bound(k_, kMin_);
```

Listing 17.13 Assemble and solve the k equation

In this case no additional manipulation is required as the production term P_k is already changed at the wall and the boundary condition for k is just a zero gradient ("zeroGradient") type.

After calculating the k and ε values, the turbulent eddy viscosity ν_t ($=\mu_t/\rho$) is updated and then corrected at wall boundaries according to Eq. (17.77). This is accomplished using the following statements in Listing 17.14:

```
    // Re-calculate viscosity
    nut_ = Cmu_*sqr(k_)/epsilon_;
    nut_.correctBoundaryConditions();
```

Listing 17.14 Updating the turbulent eddy viscosity at walls

The "volScalarField" nut (ν_t) is defined in the constructor of the kEpsilon class shown in Listing 17.15 as

```
nut_
(
    IOobject
    (
        "nut",
        runTime_.timeName(),
        mesh_,
        IOobject::NO_READ,
        IOobject::AUTO_WRITE
    ),
    autoCreateNut("nut", mesh_)
)
```

Listing 17.15 Script used to define the turbulent eddy viscosity

in which the function "autoCreateNut" is used. This function is defined in the file "backwardsCompatibilityWallFunctions.C", which is placed in the directory "FOAM_SRC/src/turbulenceModels/incompressible/RAS/backwardsCompatibility/wallFunctions". The "autoCreateNut" acts in order to create the (ν_t) object by reading the file and the related boundary types only if it is already present inside the working directory (Listing 17.16).

```
if (nutHeader.headerOk())
{
  return tmp<volScalarField>(new volScalarField(nutHeader, mesh));
}
else
{
```

Listing 17.16 "IF" statement for checking the definition of the nut file

For the case when the file "nut" is not found, the standard Spalding wall function is applied using the boundary patch class definition "nutkWallFunction FvPatchScalarField", through the following statement (Listing 17.17):

```
if (isA<wallFvPatch>(bm[patchI]))
 {
  nutBoundaryTypes[patchI] =
  nutkWallFunctionFvPatchScalarField::typeName;
 }
```

Listing 17.17 Application of the standard Spalding wall function

The class "nutkWallFunctionFvPatchScalarField" described in Listing 17.18 is located in the "FOAM_SRC/turbulenceModels/incompressible/RAS/derivedFvPatchFields/wallFunctions/nutWallFunctions" directory and inherits a base class named "nutWallFunctionFvPatchScalarField", i.e.,

```
class nutkWallFunctionFvPatchScalarField
:
    public nutWallFunctionFvPatchScalarField
{
…
```

Listing 17.18 Class definition

The "nutWallFunctionFvPatchScalarField" in Listing 17.19 is a base class that wraps the changing of the boundary value of the eddy viscosity "nut" (ν_t) by defining the related "updateCoeffs" function as

```
void nutWallFunctionFvPatchScalarField::updateCoeffs()
{
    if (updated())
    {
        return;
    }

    operator==(calcNut());

    fixedValueFvPatchScalarField::updateCoeffs();
}
```

Listing 17.19 The updateCoeffs() function definition

where the "calcNut" function is defined as pure virtual and it has to be delineated from the derived class. Based on that the derived class "nutkWallFunction FvPatchScalarField" implements the "calcNut" function according to Spalding assumption and Eq. (17.77) as Listing 17.20.

```
label faceCellI = patch().faceCells()[faceI];

scalar yPlus = Cmu25*y[faceI]*sqrt(k[faceCellI])/nuw[faceI];

if (yPlus > yPlusLam_)
{
    nutw[faceI] = nuw[faceI]*(yPlus*kappa_/log(E_*yPlus) - 1.0);
}
```

Listing 17.20 Calculate eddy viscosity at the wall according to Spalding wall function

Once the turbulent eddy viscosity is updated based on the wall function value the momentum shear stress is then correctly evaluated.

17.10.2 The SST $k - \omega$ Model

The "kOmegaSST" class implements the version of the SST $k - \omega$ model described by Eqs. (17.38) through (17.49). The set of variables and constants used in the model are defined by the script shown in Listing 17.21.

```
class kOmegaSST
:
    public RASModel
{

protected:

    // Protected data

        // Model coefficients
            dimensionedScalar alphaK1_;
            dimensionedScalar alphaK2_;

            dimensionedScalar alphaOmega1_;
            dimensionedScalar alphaOmega2_;

            dimensionedScalar gamma1_;
            dimensionedScalar gamma2_;

            dimensionedScalar beta1_;
            dimensionedScalar beta2_;

            dimensionedScalar betaStar_;

            dimensionedScalar a1_;
            dimensionedScalar b1_;
            dimensionedScalar c1_;

            Switch F3_;


        //- Wall distance field
        //  Note: different to wall distance in parent RASModel
        wallDist y_;

        // Fields

            volScalarField k_;
            volScalarField omega_;
            volScalarField nut_;
```

Listing 17.21 Script defining the SST $k - \omega$ model

```
// Protected Member Functions

    tmp<volScalarField> F1(const volScalarField& CDkOmega) const;
    tmp<volScalarField> F2() const;
    tmp<volScalarField> F3() const;
    tmp<volScalarField> F23() const;

    tmp<volScalarField> blend
    (
        const volScalarField& F1,
        const dimensionedScalar& psi1,
        const dimensionedScalar& psi2
    ) const
    {
        return F1*(psi1 - psi2) + psi2;
    }

    tmp<volScalarField> alphaK(const volScalarField& F1) const
    {
        return blend(F1, alphaK1_, alphaK2_);
    }

    tmp<volScalarField> alphaOmega(const volScalarField& F1) const
    {
        return blend(F1, alphaOmega1_, alphaOmega2_);
    }

    tmp<volScalarField> beta(const volScalarField& F1) const
    {
        return blend(F1, beta1_, beta2_);
    }

    tmp<volScalarField> gamma(const volScalarField& F1) const
    {
        return blend(F1, gamma1_, gamma2_);
    }
```

Listing 17.21 (continued)

Additional private member functions (*blend*, *alphaK*, *alphaOmega,* etc.) are now defined in order to represent the blended coefficients appearing in Eq. (17.39). Functions F1 and F2 describe the variables given by Eqs. (17.41) and (17.47), respectively. Function F3 does not appear in Menter's original model and is a modification for rough walls introduced by Hellsten [55].

As shown in Listing 17.22, the model is selectable from the "RASProperties" dictionary with the name "kOmegaSST" defined under the "TypeName" of the class as

```
//- Runtime type information
TypeName("kOmegaSST");
```

Listing 17.22 Statement used to select the SST $k - \omega$ model

The kOmegaSST class is derived from the base virtual class "turbulenceModel", and as such all its virtual base functions are specialized accordingly (Listing 17.23).

```
   //- Return the effective stress tensor including the laminar stress
        virtual tmp<volSymmTensorField> devReff() const;

   //- Return the source term for the momentum equation
      virtual tmp<fvVectorMatrix> divDevReff(volVectorField& U) const;

        //- Return the source term for the momentum equation
        virtual tmp<fvVectorMatrix> divDevRhoReff
        (
            const volScalarField& rho,
            volVectorField& U
        ) const;
```

Listing 17.23 Specialization of the pure virtual functions defined in the base class "RASModel"

The implementation of the model is detailed in "FOAM_SRC/turbulence Models/incompressible/RAS/kOmegaSST/kOmegaSST.C". As in the kEpsilon class the *correct* function solves the full set of equations of the SST turbulence model. First the ω equation is setup and solved using the script shown in Listing 17.24.

```
void kOmegaSST::correct()
{
    RASModel::correct();

    const volScalarField S2(2*magSqr(symm(fvc::grad(U_))));
    volScalarField G(GName(), nut_*S2);

    // Update omega and G at the wall
    omega_.boundaryField().updateCoeffs();

    const volScalarField CDkOmega
    (
        (2*alphaOmega2_)*(fvc::grad(k_) & fvc::grad(omega_))/omega_
    );

    const volScalarField F1(this->F1(CDkOmega));

    // Turbulent frequency equation
    tmp<fvScalarMatrix> omegaEqn
    (
        fvm::ddt(omega_)
      + fvm::div(phi_, omega_)
      - fvm::laplacian(DomegaEff(F1), omega_)
     ==
        gamma(F1)*S2
      - fvm::Sp(beta(F1)*omega_, omega_)
      - fvm::SuSp
        (
            (F1 - scalar(1))*CDkOmega/omega_,
            omega_
        )
    );
    omegaEqn().relax();
    omegaEqn().boundaryManipulate(omega_.boundaryField());
    solve(omegaEqn);
    bound(omega_, omegaMin_);
```

Listing 17.24 Assemble and solve the ω equation

In the script, the G field represents the production term P_k defined in Eq. (17.28), while the CDkOmega and F1 fields are evaluated based on Eqs. (17.41), (17.42), and (17.47).

As in the kEpsilon class the values of ω and P_k are modified at the wall following the wall functions approach by defining a wall boundary condition class for ω. Modifications to the field are then enforced through the function "omega_.boundary Field().updateCoeffs();". The boundary definition for the eddy frequency ω can be found under the directory "FOAM_SRC/turbulenceModels/incompressible/RAS/derivedFvPatchFields/wallFunctions/omegaWallFunctions/omegaWallFunction".

Based on the wall functions model, this class has to update the values of the two variables ω and P_k. This operation is performed, as shown in Listing 17.25, by the usual "calculate" function using Eqs. (17.81), (17.99) and (17.100) (with G corresponding to the P_k production term).

```
label cellI = patch.faceCells()[faceI];
scalar w = cornerWeights[faceI];

scalar omegaVis = 6.0*muw[faceI]/(rhow[faceI]*beta1_*sqr(y[faceI]));

scalar omegaLog = sqrt(k[cellI])/(Cmu25*kappa_*y[faceI]);

omega[cellI] += w*sqrt(sqr(omegaVis) + sqr(omegaLog));


G[cellI] +=
    w
   *(mutw[faceI] + muw[faceI])
   *magGradUw[faceI]
   *Cmu25*sqrt(k[cellI])
   /(kappa_*y[faceI]);
```

Listing 17.25 Calculating ω and modifying the production of turbulence kinetic energy in the near wall cells

To impose the previously calculated cell value, the matrix has to be manipulated following the same procedure used with the "kEpsilon" model. Once the ω equation is solved, OpenFOAM® proceeds to assembling and solving the k equation, as shown in Listing 17.26.

```
    // Turbulent kinetic energy equation
    tmp<fvScalarMatrix> kEqn
    (
        fvm::ddt(k_)
      + fvm::div(phi_, k_)
      - fvm::laplacian(DkEff(), k_)
     ==
        G
      - fvm::Sp(epsilon_/k_, k_)
    );
    kEqn().relax();
    solve(kEqn);
    bound(k_, kMin_);
```

Listing 17.26 Assemble and solve the k equation

In this case no additional manipulation is required as the production term P_k is already modified at the wall and the boundary condition for k is just a zero gradient ("zeroGradient") type.

Once the k and ω values are calculated the turbulent eddy viscosity $\nu_t(=\mu_t/\rho)$ is updated using the script in Listing 17.27, which is in accordance with Eq. (17.46).

```
    // Re-calculate viscosity
    nut_ = a1_*k_/max(a1_*omega_, b1_*F23()*sqrt(S2));
    nut_.correctBoundaryConditions();
```

Listing 17.27 Updating the turbulent eddy viscosity

The Menter reliability constraint is applied to the eddy viscosity where the constant b1_ takes the value of 1.0, while the F23 function, described in Listing 17.28, returns by default the proper F2 function defined by Eq. (17.47).

```
tmp<volScalarField> kOmegaSST::F23() const
{
    tmp<volScalarField> f23(F2());
    if (F3_)
    {
        f23() *= F3();
    }
    return f23;
}
```

Listing 17.28 The F23 function definition

Finally the eddy viscosity is corrected at wall boundaries according to Eq. (17.77) using the same procedure as in the kEspilon class in which the Spalding wall function is applied by the class "nutkWallFunctionFvPatchScalarField".

17.10.3 simpleFoamTurbulent

The simpleFoamTurbulent solver is an extension, for turbulent flow simulations, of the simpleFoamImproved solver described in Chap. 15. As previously described, turbulence affects the diffusion terms of the transport equations. Its effects are embodied into the equations through the effective viscosity, which is the sum of both laminar and eddy viscosities.

In order to include turbulence in the OpenFOAM® solver, the virtual base class *RASModel* is invoked and few modifications are introduced. The first one is to substitute the constant laminar transport properties with the variable turbulent ones

through the turbulence model. Therefore in the "*createFields.H*" file, the object turbulence of type RASModel is instantiated, as shown in Listing 17.29.

```
Info<< "Reading transportProperties\n" << endl;

singlePhaseTransportModel laminarTransport(U,mdotf);

autoPtr<incompressible::RASModel> turbulence
(
    incompressible::RASModel::New(U, mdotf , laminarTransport)
);
```

Listing 17.29 Turbulence model definition

For the definition of the RASModel, it is necessary to activate the *single PhaseTransportModel* class that defines the general transport model for the laminar viscosity (from dictionary it could be set as constant or as function of temperature like in the Sutherland's model), the velocity, and the mass flux. As described earlier, these quantities are necessary to define the transport equations of the turbulent quantities. To be noticed in Listing 17.29 is the definition of object *turbulence* as "*autoPtr*", which basically can be treated as a standard pointer in C^{++}.

The second main modification, as depicted in Listing 17.30, is to the momentum equation where diffusion is now computed using the *divDevReff(U) term* previously described.

```
fvVectorMatrix UEqn
(
        fvm::ddt(U)
      + fvm::div(mdotf, U)
      + fvm::SuSp(-fvc::div(mdotf),U)
      + turbulence->divDevReff(U)
);
```

Listing 17.30 Momentum equation detail

The last main modification displayed in Listing 17.31 is in the main solver file shown below.

```
…
#       include "UEqn.H"
#       include "ppEqn.H"

        turbulence->correct();
…
```

Listing 17.31 Main file modification: turbulence model solution

The addition of the statement *turbulence->correct()* activates, each time it is called, the solution of the turbulence model equations allowing the calculation of the eddy viscosity, which is used in the momentum equations.

17.11 Closure

The extra step needed to model incompressible turbulent flows was introduced. The $k-\varepsilon$, $k-\omega$, and some of their variants were discussed. Modeling of the near wall region using wall functions was detailed. This concludes the developments intended to be discussed in this book. The next chapter will discuss the implementation of boundary conditions in OpenFOAM® and uFVM.

17.12 Exercises

Exercise 1
Given the following turbulent intensities and integral length scales, calculate the corresponding $k-\omega-\varepsilon$ values (Table 17.1):

Table 17.1 Data for exercise 1

I	0.01	0.05	0.1	0.25	0.5
ℓ	0.0001	0.1	0.2	1	10
k					
ε					
ω					

Exercise 2
Given the following turbulent intensities and viscosity ratios calculate the corresponding $k-\omega-\varepsilon$ values (Table 17.2):

Table 17.2 Data for exercise 2

I	0.01	0.05	0.1	0.25	0.5
μ_t/μ	0.1	1	10	100	1000
k					
ε					
ω					

Exercise 3
Modify the standard $k-\varepsilon$ turbulence model described above to include the realizability constraint described in Eqs. (17.29–17.31).

Exercise 4
Formulate the realizability constraint of the standard $k-\omega$ turbulence model.

Exercise 5
Starting with the BSL $k-\omega$ model, show that, if the F_1 function is identically one, the ω-equation corresponds, with minor simplifications, to the standard ε equation of the $k-\varepsilon$ turbulence model.

Exercise 6
Using a Newton-Raphson linearization (i.e., $y(x) \approx y(0) + xy'(x)$), formulate the diagonal and the source term coefficients of Eq. (17.63).

Exercise 7
Formulate Eqs. (17.83) and (17.84) based on turbulent dimensionless quantities, i.e.,

$$\varepsilon = \varepsilon(u^*, d^+, k, \nu)$$
$$\omega = \omega(u^*, d^+)$$

Exercise 8
Develop an OpenFOAM® application that implements normal distance to the wall using Eqs. (17.119), (17.120), (17.121).

Exercise 9
Experiments in roughened surfaces indicate that near rough walls the classical logarithmic law of the wall has a different intercept. This is due to a higher wall shear stress, which shifts the logarithmic velocity profile downward. With the assumption that the following formula is applicable:

$$u^+ = \frac{1}{\kappa} Ln(d^+) + B - \Delta B$$

where $\Delta B = \frac{1}{\kappa} Ln(1 + h_s^+)$ (h_s^+ is the equivalent sand grain roughness), find an equivalent d_{eff}^+ formulation, valid for both smooth and rough walls, allowing the logarithmic law of the wall to be re-expressed in its classical form, i.e.,

$$u^+ = \frac{1}{\kappa} Ln(d_{eff}^+) + B$$

Exercise 10 (OpenFOAM®)
Using the Doxygen documentation [56], list all derived classes of the base class *RASModel* for incompressible flows (i.e., turbulence models available in OpenFOAM®).

Exercise 11 (OpenFOAM®)
Identify the turbulent quantities that are resolved in each incompressible turbulence model defined in Exercise 10.

Exercise 12 (OpenFOAM®)
Using the Doxygen documentation, list all derived classes of the base class *nutWallFunctionFvPatchScalarField.*

Exercise 13 (OpenFOAM®)
The base class *nutWallFunctionFvPatchScalarField* defines an additional virtual base function *virtual tmp<scalarField>yPlus () const = 0.* Describe the function definition for each of the derived classes, commenting on the differences.

Exercise 14 (OpenFOAM®)
Compare the implementation in "FOAM_SRC/turbulenceModels/incompressible/RAS/derivedFvPatchFields/wallFunctions/epsilonWallFunctions/epsilonLowReWallFunction" with the Low Reynolds number models formulation given by Eq. (17.98).

References

1. Tennekes H, Lumley JL (1972) A first course in turbulence. MIT Press, Cambridge. ISBN 978-0-262-20019-6
2. Kolmogorov AN (1941) The local structure of turbulence in incompressible viscous fluids at very large Reynolds numbers. Doklady AN SSSR 30:299–303
3. Kolmogorov AN (1941) Dissipation of energy in isotropic turbulence. Dokl Akad Nauk SSSR 32:19–21.é
4. Moser RD, Kim J, Mansour NN (1999) Direct numerical simulation of turbulent channel flow up to $Re_\tau = 590$. Phys Fluids 11(4):943–945
5. Scardovelli R, Zaleski S (1999) Direct numerical simulation of free-surface and interfacial flow. Ann Rev Fluid Mech. 31:567–603
6. Le H, Moin P, Kim J (1997) Direct numerical simulation of turbulent flow over a backward-facing step. J Fluid Mech 330:349–374
7. Choi H, Moin P, Kim J (1993) Direct numerical simulation of turbulent flow over Riblets. J Fluid Mech 255:503–539
8. Leonard A (1974) Energy cascade in large-eddy simulations of turbulent fluid flows. Adv Geophys A 18:237–248
9. Sagaut P (2006) Large eddy simulation for incompressible flows-an introduction. Springer, Berlin
10. Ferziger JH (1995) Large eddy simulation. In: Hussaini MY, Gatski T (eds) Simulation and modeling of turbulent flows. Cambridge University Press, New York
11. Nieuwstadt FTM, Mason PJ, Moeng C-H, Schuman U (1991) Large eddy simulation of the convective boundary layer: a comparison of four computer codes. In: Durst F et al (eds) Turbulent shear flows, 8th edn. Springer, Berlin
12. Reynolds O (1895) On the dynamical theory of incompressible viscous fluids and the determination of the criterion. Philos Trans Royal Soc London A 186:123–164
13. Favre A (1965) Equations des Gas Turbulents Compressibles. Journal de Mecanique 4 (3):361–390
14. Boussinesq J (1877) Essai sur la théorie des eaux courantes. Mémoires présentés par divers savants à l'Académie des Sciences 23(1):1–680
15. Schlichting H (1968) *Boundary-layer theory,* 6th edn. Chapter XIX. McGraw Hill
16. Schmitt FG (2007) About Boussinesq's turbulent viscosity hypothesis: historical remarks and a direct evaluation of its validity. Comptes Rendus Mécanique 335(9 and 10):617–627
17. Prandtl L (1925) Uber die ausgebildete Turbulenz. ZAMM 5:136–139

18. Baldwin BS, Lomax H (1978) Thin-Layer approximation and algebraic model for separated turbulent flows. AIAA Paper, Huntsville, pp 78–257
19. Cebeci T, Smith AMO (1974) Analysis of turbulent boundary layers. Ser Appl Math Mech, vol XV, Academic Press, Waltham
20. Baldwin BS, Barth TJ (1990) A one-equation turbulence transport model for high reynolds number wall-bounded flows. NASA TM-102847
21. Goldberg UC (1991) Derivation and testing of a one-equation model based on two time scales. AIAA J 29(8):1337–1340
22. Spalart PR, Allmaras SR (1992) A one-equation turbulence model for aerodynamic flows. AIAA Paper, Reno, pp 92–439
23. Jones WP, Launder BE (1972) The prediction of laminarization with a two-equation model of turbulence. Int J Heat Mass Transf 15:301–314
24. Launder BE, Sharma BI (1974) Application of the energy dissipation model of turbulence to the calculation of flow near a spinning disk. Lett Heat Mass Transfer 1(2):131–138
25. Chien K-Y (1982) Predictions of channel and boundary-layer flows with a low-reynolds-number turbulence model. AIAA J 20(1):33–38
26. Myong HK, Kasagi N (1990) A new approach to the improvement of k-ε turbulence model for wall-bounded shear flows. JSME Int J 33:63–72
27. Kolmogorov AN (1942) Equations of turbulent motion of an incompressible fluid. Izvestia Acad Sci USSR Phys 6(1 and 2):56–58
28. Wilcox D (1988) Reassessment of the scale-determining equation for advanced turbulence models. AIAA J 26(11):1299–1310
29. Wilcox DC (1998) Turbulence modeling for CFD, 2nd edn. DCW Industries, US
30. Menter FR (1992) Influence of freestream values on $k - \omega$ turbulence model predictions. AIAA J 30(6):1657–1659
31. Menter FR (1993) Zonal two-equation k − ω turbulence model for aerodynamic flows. AIAA Paper, Orlando, pp 1993–2906
32. Menter F (1994) Two-equation eddy-viscosity turbulence models for engineering applications. AIAA J 32(8):1598–1605
33. Menter FR, Kuntz M, Langtry R (2003) Ten years of industrial experience with the SST turbulence model, 4th edn. Turbulence, Heat and Mass Transfer, Antalya, pp 73–86
34. Menter FR, Carregal Ferreira J, Esch T, Konno B (2003) The SST turbulence model with improved wall treatment for heat transfer predictions in gas turbines. In: Proceedings of the international gas turbine congress, Tokyo, IGTC2003-TS-059
35. Menter FR (2009) Review of the shear-stress transport turbulence model experience from an industrial perspective. Int J Comput Fluid Dyn 23(4):305–316
36. Daky BJ, Harlow FH (1970) Transport equations in turbulence. Phys Fluids 13:2634–2649
37. Fu S, Launder BE, Tselepidakis DP (1987) Accommodating the effects of high strain rates in modelling the pressure-strain correlation. Report no. TFD/87/5, Mechanical Engineering Department, Manchester Institute of Science and Technology, England
38. Gibson MM, Launder BE (1986) Ground effects on pressure fluctuations in the atmospheric boundary layer. J Fluid Mech 86(Pt. 3):491–511
39. Gibson MM, Younis BA (1986) Calculation of swirling jets with a reynolds stress closure. Phys Fluids 29:38–48
40. Wilcox DC, Rubesin MW (1980) Progress in turbulence modeling for complex flow fields including effects of compressibility. NASA TP-1517
41. Wilcox DC (1988) Multiscale model for turbulent flows. AIAA J 26(11):1311–1320
42. Patel VC, Rodi W, Scheuerer G (1985) Turbulence models for near-wall and low reynolds number flows: a review. AIAA J 23(9):1308–1319
43. Medic G, Durbin PA (2002) Toward improved prediction of heat transfer on turbine blades. ASME J Turbomach 124(2):187–192
44. Sahay A, Sreenivasan KR (1999) The wall-normal position in pipe and channel flows at which viscous and turbulent shear stresses are equal. Phys Fluids 11(10):3186–3188

45. Bredberg J (2000) On the wall boundary condition for turbulence models. Department of Thermo and Fluid Dynamics, Chalmers University of Technology, Internal report 00/4, Goteborg
46. Launder BE, Spalding DB (1974) The numerical computation of turbulent flows. Comput Methods Appl Mech Eng 3:269–289
47. Grotjans H, Menter F(1998) Wall function for general application cfd codes. In: Computational fluid dynamics 1998, Proceedings fourth European CFD Conference ECCOMAS, Wiley, Chichester
48. Menter F, Esch T (2001) Elements of industrial heat transfer prediction. In: Proceedings 16th Brazilian congress of mechanical engineering (COBEM), pp 117–127
49. Kader BA (1981) Temperature and concentration profiles in fully turbulent boundary layers. Int J Heat Mass Transf 24:1541–1544
50. Tucker PG (2003) Differential equation-based wall distance computation for DES and RANS. J Comput Phys 190:229–248
51. Sethian JA (1999) Fast marching methods. SIAM Rev 41(2):199–235
52. Tucker PG, Rumsey CL, Spalart PR, Bartels RE, Biedron RT (2004) Computations of wall distances based on differential equations. AIAA Paper 2004–2232
53. Xu J-L, Yan C, Fan J-J (2011) Computations of wall distances by solving a transport equation. Appl Math Mech 32(2):141–150
54. OpenFOAM (2015) Version 2.3.x. http://www.openfoam.org
55. Hellsten A (1998) Some improvements in menter's k-omega-SST turbulence model. In: 29th AIAA fluid dynamics conference, AIAA-98-2554
56. OpenFOAM Doxygen (2015) Version 2.3.x. http://www.openfoam.org/docs/cpp/