Chapter 18. Theory

This chapter provides some theoretical background for the models, equations, and solution procedures used by Airpak. Information is presented in the following sections:

- Section 18.1: Governing Equations
- Section 18.2: Turbulence
- Section 18.3: Buoyancy-Driven Flows and Natural Convection
- Section 18.4: Radiation
- Section 18.5: Solution Procedures

18.1 Governing Equations

Airpak solves the Navier-Stokes equations for transport of mass, momentum, species, and energy when it calculates laminar flow with heat transfer. Additional transport equations are solved when the flow is turbulent (see Section 18.2) or when radiative heat transfer is included (see Section 18.4).

18.1.1 The Mass Conservation Equation

The equation for conservation of mass, or continuity equation, can be written as follows:

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \vec{v}) = 0 \qquad (18.1\text{-}1)$$

For an incompressible fluid, this reduces to

$$\nabla \cdot \vec{v} = 0 \qquad (18.1\text{-}2)$$

18.1.2 Momentum Equations

Transport of momentum in the ith direction in an inertial (non-accelerating) reference frame is described by [4]

$$\frac{\partial}{\partial t}(\rho\vec{v}) + \nabla \cdot (\rho\vec{v}\vec{v}) = -\nabla p + \nabla \cdot (\overline{\overline{\tau}}) + \rho\vec{g} + \vec{F} \tag{18.1-3}$$

where p is the static pressure, $\overline{\overline{\tau}}$ is the stress tensor(described below), and $\rho\vec{g}$ is the gravitational body force. $\vec{F}$ contains other source terms that may arise from resistances, sources, etc.

The stress tensor $\overline{\overline{\tau}}$ is given by

$$\overline{\overline{\tau}} = \mu\left[(\nabla\vec{v} + \nabla\vec{v}^{\mathrm{T}}) - \frac{2}{3}\nabla\cdot\vec{v}I\right] \tag{18.1-4}$$

where μ is the molecular viscosity and the second term on the right hand side is the effect of volume dilation.

18.1.3 Energy Conservation Equation

The energy equation for a fluid region can be written in terms of sensible enthalpy h (=$\int_{T_{\mathrm{ref}}}^{T} c_p dT$, where T_{ref} is 298.15 K) as

$$\frac{\partial}{\partial t}(\rho h) + \nabla \cdot (\rho h\vec{v}) = \nabla \cdot [(k + k_t)\nabla T] + S_h \tag{18.1-5}$$

where k is the molecular conductivity, k_t is the conductivity due to turbulent transport ($k_t = c_p\mu_t/\mathrm{Pr}_t$), and the source term S_h includes any volumetric heat sources you have defined.

In conducting solid regions, Airpak solves a simple conduction equation that includes the heat flux due to conduction and volumetric heat sources within the solid:

$$\frac{\partial}{\partial t}(\rho h) = \nabla \cdot (k\nabla T) + S_h \tag{18.1-6}$$

where ρ is density, k is conductivity, T is temperature, and S_h is the volumetric heat source.

Equation 18.1-6 is solved simultaneously with the energy transport equation, Equation 18.1-5, in the flow regions to yield a fully coupled conduction/convection heat transfer prediction.

18.1.4 Species Transport Equations

When you choose to solve conservation equations for species, Airpak predicts the local mass fraction of each species, Y_i, through the solution of a convection-diffusion equation for the ith species. This conservation equation takes the following general form:

$$\frac{\partial}{\partial t}(\rho Y_i) + \nabla \cdot (\rho \vec{v} Y_i) = -\nabla \cdot \vec{J}_i + S_i \qquad (18.1\text{-}7)$$

where S_i is the rate of creation by addition from user-defined sources. An equation of this form will be solved for $N-1$ species where N is the total number of fluid phase species present in the system.

Mass Diffusion in Laminar Flows

$\vec{J}_i$ is the diffusion flux of species i, which arises due to concentration gradients. Airpak uses the dilute approximation, under which the diffusion flux can be written as

$$\vec{J}_i = -\rho D_{i,m} \nabla Y_i \qquad (18.1\text{-}8)$$

Here $D_{i,m}$ is the diffusion coefficient for species i in the mixture.

Mass Diffusion in Turbulent Flows

In turbulent flows, Airpak computes the mass diffusion in the following form:

$$\vec{J}_i = -\left(\rho D_{i,m} + \frac{\mu_t}{\mathrm{Sc}_t}\right) \nabla Y_i \qquad (18.1\text{-}9)$$

where Sc_t is the turbulent Schmidt number, $\frac{\mu_t}{\rho D_t}$ (with a default setting of 0.7).

Treatment of Species Transport in the Energy Equation

For many multicomponent mixing flows, the transport of enthalpy due to species diffusion

$$\nabla \cdot \left[\sum_{i=1}^{n} (h_i) \vec{J}_i \right]$$

can have a significant effect on the enthalpy field and should not be neglected. In particular, when the Lewis number

$$\text{Le}_i = \frac{k}{\rho c_p D_{i,m}}$$

is far from unity, this term cannot be neglected. Airpak will include this term by default.

18.2 Turbulence

Four turbulence models are available in Airpak: the mixing-length zero-equation model, the indoor zero-equation model, the two-equation (standard k-ϵ) model, and the RNG k-ϵ model.

18.2.1 Zero-Equation Turbulence Models

Airpak provides two zero-equation turbulence models: the mixing-length model and the indoor model. These models are described below.

Mixing-Length Zero-Equation Turbulence Model

The mixing-length zero-equation turbulence model (also known as the algebraic model) uses the following relation to calculate turbulent viscosity, μ_t:

$$\mu_t = \rho \ell^2 S \tag{18.2-1}$$

The mixing length, ℓ, is defined as

$$\ell = \min(\kappa d, 0.09 d_{\max}) \tag{18.2-2}$$

where d is the distance from the wall and the von Kármán constant $\kappa = 0.419$.

S is the modulus of the mean rate-of-strain tensor, defined as

$$S \equiv \sqrt{2 S_{ij} S_{ij}} \tag{18.2-3}$$

with the mean strain rate S_{ij} given by

$$S_{ij} = \frac{1}{2}\left(\frac{\partial u_j}{\partial x_i} + \frac{\partial u_i}{\partial x_j}\right) \tag{18.2-4}$$

Indoor Zero-Equation Turbulence Model

The indoor zero-equation turbulence model was developed specifically for indoor airflow simulations [6]. It addresses the need of HVAC engineers for a simple but reliable turbulence model that can be used with modest desktop computing resources. It uses the following relationship to calculate the turbulent viscosity μ_t:

$$\mu_t = 0.03874 \rho v L \tag{18.2-5}$$

where v is the local velocity magnitude, ρ is the fluid density, L is defined as the distance from the nearest wall, and 0.03874 is an empirical constant.

Airpak determines the heat transfer at the boundary surfaces by computing a convective heat transfer coefficient:

$$h = \frac{\mu_{\text{eff}}}{\text{Pr}_{\text{eff}}} \frac{c_p}{\Delta x_j} \tag{18.2-6}$$

where c_p is the fluid specific heat, Pr_{eff} is the effective Prandtl number, and Δx_j is the grid spacing adjacent to the wall. μ_{eff} is the effective viscosity, given by

$$\mu_{\text{eff}} = \mu + \mu_t \tag{18.2-7}$$

where μ is the viscosity of the fluid.

This model is ideally suited for predicting indoor air flows that consider natural convection, forced convection, mixed convection, and displacement ventilation.

18.2.2 The Two-Equation (Standard k-ϵ) and RNG k-ϵ Models

This section presents the standard and RNG k-ϵ models. Both models have similar forms, with transport equations for k and ϵ. The major differences in the models are as follows:

- the method of calculating turbulent viscosity
- the turbulent Prandtl numbers governing the turbulent diffusion of k and ϵ
- the generation and destruction terms in the ϵ equation

This section describes the Reynolds-averaging method for calculating turbulent effects and provides an overview of the issues related to choosing an advanced turbulence model in Airpak. The transport equations, methods of calculating turbulent viscosity, and model constants are presented separately for each model. The features that are essentially common to both models follow, including turbulent production, generation due to buoyancy, and modeling heat transfer.

Reynolds (Ensemble) Averaging

The advanced turbulence models in Airpak are based on Reynolds averages of the governing equations. In Reynolds averaging, the solution variables in the instantaneous (exact) Navier-Stokes equations are decomposed into the mean (ensemble-averaged or time-averaged) and fluctuating components. For the velocity components:

$$u_i = \bar{u}_i + u_i' \tag{18.2-8}$$

where $\bar{u}_i$ and u_i' are the mean and instantaneous velocity components ($i = 1, 2, 3$).

Likewise, for pressure and other scalar quantities:

$$\phi = \bar{\phi} + \phi' \tag{18.2-9}$$

where ϕ denotes a scalar such as pressure or energy.

Substituting expressions of this form for the flow variables into the instantaneous continuity and momentum equations and taking a time (or ensemble) average (and dropping the overbar on the mean velocity, $\bar{u}$) yields the ensemble-averaged momentum equations. They can be written in Cartesian tensor form as:

$$\frac{\partial \rho}{\partial t} + \frac{\partial}{\partial x_i}(\rho u_i) = 0 \tag{18.2-10}$$

$$\frac{\partial}{\partial t}(\rho u_i) + \frac{\partial}{\partial x_j}(\rho u_i u_j) =$$

$$-\frac{\partial p}{\partial x_i} + \frac{\partial}{\partial x_j}\left[\mu\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} - \frac{2}{3}\delta_{ij}\frac{\partial u_l}{\partial x_l}\right)\right] + \frac{\partial}{\partial x_j}(-\rho\overline{u_i' u_j'}) \tag{18.2-11}$$

Equations 18.2-10 and 18.2-11 are called "Reynolds-averaged" Navier-Stokes (RANS) equations. They have the same general form as the instantaneous Navier-Stokes equations, with the velocities and other solution variables now representing ensemble-averaged (or time-averaged) values. Additional terms now appear that represent the effects of turbulence. These "Reynolds stresses", $-\rho\overline{u_i' u_j'}$, must be modeled in order to close Equation 18.2-11.

Choosing an Advanced Turbulence Model

This section provides an overview of the issues related to the advanced turbulence models provided in Airpak.

The Standard k-ϵ Model

The simplest "complete models" of turbulence are two-equation models in which the solution of two separate transport equations allows the turbulent velocity and length scales to be independently determined. The standard k-ϵ model in Airpak falls within this class of turbulence model and has become the workhorse of practical engineering flow calculations in the time since it was proposed by Launder and Spalding [15]. Robustness, economy, and reasonable accuracy for a wide range of turbulent flows explain its popularity in industrial flow and heat transfer simulations. It is a semi-empirical model, and the derivation of the model equations relies on phenomenological considerations and empiricism.

As the strengths and weaknesses of the standard k-ϵ model have become known, improvements have been made to the model to improve its performance. One of these variants is available in Airpak: the RNG k-ϵ model [22].

The RNG k-ϵ Model

The RNG k-ϵ model was derived using a rigorous statistical technique (called renormalization group theory). It is similar in form to the standard k-ϵ model, but includes the following refinements:

- The RNG model has an additional term in its ϵ equation that significantly improves the accuracy for rapidly strained flows.
- The effect of swirl on turbulence is included in the RNG model, enhancing accuracy for swirling flows.

- The RNG theory provides an analytical formula for turbulent Prandtl numbers, while the standard k-ϵ model uses user-specified, constant values.
- While the standard k-ϵ model is a high-Reynolds-number model, the RNG theory provides an analytically-derived differential formula for effective viscosity that accounts for low-Reynolds-number effects.

These features make the RNG k-ϵ model more accurate and reliable for a wider class of flows than the standard k-ϵ model.

Computational Effort: CPU Time and Solution Behavior

Due to the extra terms and functions in the governing equations and a greater degree of nonlinearity, computations with the RNG k-ϵ model tend to take 10–15% more CPU time than with the standard k-ϵ model.

Aside from the time per iteration, the choice of turbulence model can affect the ability of Airpak to obtain a converged solution. For example, the standard k-ϵ model is known to be slightly over-diffusive in certain situations, while the RNG k-ϵ model is designed such that the turbulent viscosity is reduced in response to high rates of strain. Since diffusion has a stabilizing effect on the numerics, the RNG model is more likely to be susceptible to instability in steady-state solutions. However, this should not necessarily be seen as a disadvantage of the RNG model, since these characteristics make it more responsive to important physical instabilities such as time-dependent turbulent vortex shedding.

The Two-Equation (Standard k-ϵ) Turbulence Model

The two-equation turbulence model (also known as the standard k-ϵ model) is more complex than the zero-equation model. The standard k-ϵ model [15] is a semi-empirical model based on model transport equations for the turbulent kinetic energy (k) and its dissipation rate (ϵ). The model transport equation for k is derived from the exact equation, while the model transport equation for ϵ is obtained using physical reasoning and bears little resemblance to its mathematically exact counterpart.

In the derivation of the standard k-ϵ model, it is assumed that the flow is fully turbulent, and the effects of molecular viscosity are negligible. The standard k-ϵ model is therefore valid only for fully turbulent flows.

Transport Equations for the Standard k-ϵ Model

The turbulent kinetic energy, k, and its rate of dissipation, ϵ, are obtained from the following transport equations:

$$\frac{\partial}{\partial t}(\rho k) + \frac{\partial}{\partial x_i}(\rho k u_i) = \frac{\partial}{\partial x_i}\left[\left(\mu + \frac{\mu_t}{\sigma_k}\right)\frac{\partial k}{\partial x_i}\right] + G_k + G_b - \rho\epsilon \tag{18.2-12}$$

and

$$\frac{\partial}{\partial t}(\rho\epsilon) + \frac{\partial}{\partial x_i}(\rho\epsilon u_i) =$$
$$\frac{\partial}{\partial x_i}\left[\left(\mu + \frac{\mu_t}{\sigma_\epsilon}\right)\frac{\partial \epsilon}{\partial x_i}\right] + C_{1\epsilon}\frac{\epsilon}{k}(G_k + C_{3\epsilon}G_b) - C_{2\epsilon}\rho\frac{\epsilon^2}{k} \tag{18.2-13}$$

In these equations, G_k represents the generation of turbulent kinetic energy due to the mean velocity gradients, calculated as described later in this section. G_b is the generation of turbulent kinetic energy due to buoyancy, calculated as described later in this section. $C_{1\epsilon}$, $C_{2\epsilon}$, and $C_{3\epsilon}$ are constants. σ_k and σ_ϵ are the turbulent Prandtl numbers for k and ϵ, respectively.

Modeling the Turbulent Viscosity

The "eddy" or turbulent viscosity, μ_t, is computed by combining k and ϵ as follows:

$$\mu_t = \rho C_\mu \frac{k^2}{\epsilon} \tag{18.2-14}$$

where C_μ is a constant.

Model Constants

The model constants $C_{1\epsilon}, C_{2\epsilon}, C_\mu, \sigma_k$, and σ_ϵ have the following default values [15]:

$$C_{1\epsilon} = 1.44, \quad C_{2\epsilon} = 1.92, \quad C_\mu = 0.09, \quad \sigma_k = 1.0, \quad \sigma_\epsilon = 1.3$$

These default values have been determined from experiments with air and water for fundamental turbulent shear flows including homogeneous shear flows and decaying isotropic grid turbulence. They have been found to work fairly well for a wide range of wall-bounded and free shear flows.

The RNG k-ϵ Model

The RNG-based k-ϵ turbulence model is derived from the instantaneous Navier-Stokes equations, using a mathematical technique called "renormalization group" (RNG) methods. The analytical derivation results in a model with constants different from those in the standard k-ϵ model, and additional terms and functions in the transport equations for k and ϵ. A more comprehensive description of RNG theory and its application to turbulence can be found in [7].

Transport Equations for the RNG k-ϵ Model

The RNG k-ϵ model has a similar form to the standard k-ϵ model:

$$\frac{\partial}{\partial t}(\rho k) + \frac{\partial}{\partial x_i}(\rho k u_i) = \frac{\partial}{\partial x_i}\left(\alpha_k \mu_{\rm eff} \frac{\partial k}{\partial x_i}\right) + G_k + G_b - \rho\epsilon \tag{18.2-15}$$

and

$$\frac{\partial}{\partial t}(\rho\epsilon) + \frac{\partial}{\partial x_i}(\rho\epsilon u_i) =$$
$$\frac{\partial}{\partial x_i}\left(\alpha_\epsilon \mu_{\rm eff} \frac{\partial \epsilon}{\partial x_i}\right) + C_{1\epsilon}\frac{\epsilon}{k}\left(G_k + C_{3\epsilon}G_b\right) - C_{2\epsilon}\rho\frac{\epsilon^2}{k} - R_\epsilon \tag{18.2-16}$$

In these equations, G_k represents the generation of turbulent kinetic energy due to the mean velocity gradients, calculated as described later in this section. G_b is the generation of turbulent kinetic energy due to buoyancy, calculated as described later in this section. The quantities α_k and α_ϵ are the inverse effective Prandtl numbers for k and ϵ, respectively.

Modeling the Effective Viscosity

The scale elimination procedure in RNG theory results in a differential equation for turbulent viscosity:

$$d\left(\frac{\rho^2 k}{\sqrt{\epsilon\mu}}\right) = 1.72\frac{\hat{\nu}}{\sqrt{\hat{\nu}^3 - 1 + C_\nu}}d\hat{\nu} \tag{18.2-17}$$

where

$$\begin{aligned} \hat{\nu} &= \mu_{\text{eff}}/\mu \\ C_\nu &\approx 100 \end{aligned}$$

Equation 18.2-17 is integrated to obtain an accurate description of how the effective turbulent transport varies with the effective Reynolds number (or eddy scale), allowing the model to better handle low-Reynolds-number and near-wall flows.

In the high-Reynolds-number limit, Equation 18.2-17 gives

$$\mu_t = \rho C_\mu \frac{k^2}{\epsilon} \tag{18.2-18}$$

with $C_\mu = 0.0845$, derived using RNG theory. It is interesting to note that this value of C_μ is very close to the empirically-determined value of 0.09 used in the standard k-ϵ model.

In Airpak, the effective viscosity is computed using the differential viscosity in Equation 18.2-17.

Calculating the Inverse Effective Prandtl Numbers

The inverse effective Prandtl numbers α_k and α_ϵ are computed using the following formula derived analytically by the RNG theory:

$$\left|\frac{\alpha - 1.3929}{\alpha_0 - 1.3929}\right|^{0.6321} \left|\frac{\alpha + 2.3929}{\alpha_0 + 2.3929}\right|^{0.3679} = \frac{\mu_{\rm mol}}{\mu_{\rm eff}} \qquad (18.2\text{-}19)$$

where $\alpha_0 = 1.0$. In the high-Reynolds-number limit ($\mu_{\rm mol}/\mu_{\rm eff} \ll 1$), $\alpha_k = \alpha_\epsilon \approx 1.393$.

The R_ϵ Term in the ϵ Equation

The main difference between the RNG and standard k-ϵ models lies in the additional term in the ϵ equation given by

$$R_\epsilon = \frac{C_\mu \rho \eta^3 (1 - \eta/\eta_0)}{1 + \beta\eta^3} \frac{\epsilon^2}{k} \qquad (18.2\text{-}20)$$

where $\eta \equiv Sk/\epsilon$, $\eta_0 = 4.38$, $\beta = 0.012$.

The effects of this term in the RNG ϵ equation can be seen more clearly by rearranging Equation 18.2-16. Using Equation 18.2-20, the last two terms in Equation 18.2-16 can be merged, and the resulting ϵ equation can be rewritten as

$$\rho \frac{D\epsilon}{Dt} = \frac{\partial}{\partial x_i}\left(\alpha_\epsilon \mu_{\rm eff} \frac{\partial \epsilon}{\partial x_i}\right) + C_{1\epsilon}\frac{\epsilon}{k}\left(G_k + C_{3\epsilon} G_b\right) - C^*_{2\epsilon}\rho\frac{\epsilon^2}{k} \qquad (18.2\text{-}21)$$

where $C^*_{2\epsilon}$ is given by

$$C^*_{2\epsilon} \equiv C_{2\epsilon} + \frac{C_\mu \rho \eta^3 (1 - \eta/\eta_0)}{1 + \beta\eta^3} \qquad (18.2\text{-}22)$$

In regions where $\eta < \eta_0$, the R term makes a positive contribution, and $C^*_{2\epsilon}$ becomes larger than $C_{2\epsilon}$. In the logarithmic layer, for instance, it can be shown that $\eta \approx 3.0$, giving $C^*_{2\epsilon} \approx 2.0$, which is close in magnitude to the value of $C_{2\epsilon}$ in the standard k-ϵ model (1.92). As a result, for weakly to moderately strained flows, the RNG model tends to give results largely comparable to the standard k-ϵ model.

In regions of large strain rate ($\eta > \eta_0$), however, the R term makes a negative contribution, making the value of $C^*_{2\epsilon}$ less than $C_{2\epsilon}$. In comparison with the standard k-ϵ model, the smaller destruction of ϵ augments ϵ, reducing k and eventually the effective viscosity. As a result, in rapidly strained flows, the RNG model yields a lower turbulent viscosity than the standard k-ϵ model.

Thus, the RNG model is more responsive to the effects of rapid strain and streamline curvature than the standard k-ϵ model, which explains the superior performance of the RNG model for certain classes of flows.

Model Constants

The model constants $C_{1\epsilon}$ and $C_{2\epsilon}$ in Equation 18.2-16 have values derived analytically by the RNG theory. These values, used by default in Airpak, are

$$C_{1\epsilon} = 1.42, \quad C_{2\epsilon} = 1.68$$

Modeling Turbulent Production in the k-ϵ Models

From the exact equation for the transport of k, the term G_k, representing the production of turbulent kinetic energy, can be defined as

$$G_k = -\rho \overline{u'_i u'_j} \frac{\partial u_j}{\partial x_i} \tag{18.2-23}$$

To evaluate G_k in a manner consistent with the Boussinesq hypothesis,

$$G_k = \mu_t S^2 \tag{18.2-24}$$

where S is the modulus of the mean rate-of-strain tensor, defined as

$$S \equiv \sqrt{2S_{ij}S_{ij}} \qquad (18.2\text{-}25)$$

with the mean strain rate S_{ij} given by

$$S_{ij} = \frac{1}{2}\left(\frac{\partial u_j}{\partial x_j} + \frac{\partial u_i}{\partial x_j}\right) \qquad (18.2\text{-}26)$$

Effects of Buoyancy on Turbulence in the k-ϵ Models

When a non-zero gravity field and temperature gradient are present simultaneously, the k-ϵ models in Airpak account for the generation of k due to buoyancy (G_b in Equations 18.2-12 and 18.2-15), and the corresponding contribution to the production of ϵ in Equations 18.2-13 and 18.2-16.

The generation of turbulence due to buoyancy is given by

$$G_b = \beta g_i \frac{\mu_t}{\mathrm{Pr}_t}\frac{\partial T}{\partial x_i} \qquad (18.2\text{-}27)$$

where Pr_t is the turbulent Prandtl number for energy. For the standard k-ϵ model, the default value of Pr_t is 0.85. In the case of the RNG k-ϵ model, $\mathrm{Pr}_t = 1/\alpha$, where α is given by Equation 18.2-19, but with $\alpha_0 = 1/\mathrm{Pr} = k/\mu c_p$. The coefficient of thermal expansion, β, is defined as

$$\beta = -\frac{1}{\rho}\left(\frac{\partial \rho}{\partial T}\right)_p \qquad (18.2\text{-}28)$$

It can be seen from the transport equation for k (Equation 18.2-12 or 18.2-15) that turbulent kinetic energy tends to be augmented ($G_b > 0$) in unstable stratification. For stable stratification, buoyancy tends to suppress the turbulence ($G_b < 0$). In Airpak, the effects of buoyancy on the generation of k are always included when you have both a non-zero gravity field and a non-zero temperature (or density) gradient.

While the buoyancy effects on the generation of k are relatively well understood, the effect on ϵ is less clear. In Airpak, by default, the buoyancy effects on ϵ are neglected simply by setting G_b to zero in the transport equation for ϵ (Equation 18.2-13 or 18.2-16).

The degree to which ϵ is affected by the buoyancy is determined by the constant $C_{3\epsilon}$. In Airpak, $C_{3\epsilon}$ is not specified, but is instead calculated according to the following relation [10]:

$$C_{3\epsilon} = \tanh \left| \frac{v}{u} \right| \tag{18.2-29}$$

where v is the component of the flow velocity parallel to the gravitational vector and u is the component of the flow velocity perpendicular to the gravitational vector. In this way, $C_{3\epsilon}$ will become 1 for buoyant shear layers for which the main flow direction is aligned with the direction of gravity. For buoyant shear layers that are perpendicular to the gravitational vector, $C_{3\epsilon}$ will become zero.

Convective Heat Transfer Modeling in the k-ϵ Models

In Airpak, turbulent heat transport is modeled using the concept of Reynolds' analogy to turbulent momentum transfer. The "modeled" energy equation is thus given by the following:

$$\frac{\partial}{\partial t}(\rho E) + \frac{\partial}{\partial x_i}[u_i(\rho E + p)] = \frac{\partial}{\partial x_i}\left(k_{\text{eff}} \frac{\partial T}{\partial x_i}\right) + S_h \tag{18.2-30}$$

where E is the total energy and k_{eff} is the effective conductivity.

For the standard k-ϵ model, k_{eff} is given by

$$k_{\text{eff}} = k + \frac{c_p \mu_t}{\text{Pr}_t}$$

with the default value of the turbulent Prandtl number set to 0.85.

For the RNG k-ϵ model, the effective thermal conductivity is

$$k_{\text{eff}} = \alpha c_p \mu_{\text{eff}}$$

where α is calculated from Equation 18.2-19, but with $\alpha_0 = 1/\text{Pr} = k/\mu c_p$.

The fact that α varies with $\mu_{\text{mol}}/\mu_{\text{eff}}$, as in Equation 18.2-19, is an advantage of the RNG k-ϵ model. It is consistent with experimental evidence indicating that the turbulent Prandtl number varies with the molecular Prandtl number and turbulence [14]. Equation 18.2-19 works well across a very broad range of molecular Prandtl numbers, from liquid metals ($\text{Pr} \approx 10^{-2}$) to paraffin oils ($\text{Pr} \approx 10^3$), which allows heat transfer to be calculated in low-Reynolds-number regions. Equation 18.2-19 smoothly predicts the variation of effective Prandtl number from the molecular value ($\alpha = 1/\text{Pr}$) in the viscosity-dominated region to the fully turbulent value ($\alpha = 1.393$) in the fully turbulent regions of the flow.

18.3 Buoyancy-Driven Flows and Natural Convection

The importance of buoyancy forces in a mixed convection flow can be measured by the ratio of the Grashof and Reynolds numbers:

$$\frac{\text{Gr}}{\text{Re}^2} = \frac{\Delta \rho g h}{\rho v^2} \tag{18.3-1}$$

When this number approaches or exceeds unity, you should expect strong buoyancy contributions to the flow. Conversely, if it is very small, buoyancy forces may be ignored in your simulation. In pure natural convection, the strength of the buoyancy-induced flow is measured by the Rayleigh number:

$$\text{Ra} = \frac{g \beta \Delta T L^3 \rho}{\mu \alpha} \tag{18.3-2}$$

where β is the thermal expansion coefficient:

$$\beta = -\frac{1}{\rho}\frac{\partial \rho}{\partial T} \tag{18.3-3}$$

and α is the thermal diffusivity:

$$\alpha = \frac{k}{\rho c_p} \tag{18.3-4}$$

Rayleigh numbers less than 10^8 indicate a buoyancy-induced laminar flow, with transition to turbulence occurring over the range of $10^8 < \mathrm{Ra} < 10^{10}$.

Airpak uses either the Boussinesq model or the ideal gas law in the calculation of natural-convection flows, as described below.

18.3.1 The Boussinesq Model

By default, Airpak uses the Boussinesq model for natural-convection flows involving only one species. This model treats density as a constant value in all solved equations, except for the buoyancy term in the momentum equation:

$$(\rho - \rho_0)g \approx -\rho_0 \beta (T - T_0) g \tag{18.3-5}$$

where ρ_0 is the (constant) density of the flow, T_0 is the operating temperature, and β is the thermal expansion coefficient. Equation 18.3-5 is obtained by using the Boussinesq approximation $\rho = \rho_0(1 - \beta\Delta T)$ to eliminate ρ from the buoyancy term. This approximation is accurate as long as changes in actual density are small; specifically, the Boussinesq approximation is valid when $\beta(T - T_0) \ll 1$.

18.3.2 Incompressible Ideal Gas Law

In Airpak, if you choose to define the density using the ideal gas law for a single-fluid problem, Airpak will compute the density as

$$\rho = \frac{p_{\mathrm{op}}}{\frac{R}{M_w} T} \tag{18.3-6}$$

where R is the universal gas constant, M_w is the molecular weight of the fluid, and p_{op} is defined by you as the Oper. pressure in the Advanced problem setup panel (see Section 6.4.4). In this form, the density depends on the operating pressure (not on the local relative pressure field, local temperature field, or molecular weight).

If you choose to model species transport in Airpak, Airpak will compute the density as

$$\rho = \frac{p_{\rm op}}{RT \sum_i \frac{Y_i}{M_{w,i}}} \tag{18.3-7}$$

where Y_i is the mass fraction of species i and $M_{w,i}$ is the molecular weight of species i. In this form, the density depends only on the operating pressure.

Definition of the Operating Density

When the Boussinesq approximation is not used, the operating density, ρ_0, appears in the body-force term in the momentum equations as $(\rho - \rho_0)g$.

This form of the body-force term follows from the redefinition of pressure in Airpak as

$$p_s' = p_s - \rho_0 g x \tag{18.3-8}$$

The hydrostatic balance in a fluid at rest is then

$$p_s' = 0 \tag{18.3-9}$$

The definition of the operating density is thus important in all buoyancy-driven flows.

18.4 Radiation

Airpak provides two models that allow you to include radiation in your heat transfer simulations:

- Surface-to-surface (S2S) radiation model [21]
- Discrete ordinates (DO) radiation model [8, 19]

18.4.1 Introduction to Radiative Heat Transfer

The terms radiative heat transfer and thermal radiation are commonly used to describe heat transfer caused by electromagnetic (EM) waves. All materials continually emit and absorb EM waves, or photons. The strength and wavelength of emission depends on the temperature of the emitting material. At absolute zero K, no radiation is emitted from a surface. For heat transfer applications, wavelengths in the infrared spectrum are generally of greatest importance and are, therefore, the only ones considered in Airpak.

While both conduction and convection (the other basic modes of heat transfer) require a medium for transmission, radiation does not. Therefore, thermal radiation can traverse a long distance without interacting with a medium. Also, for most applications, conductive and convective heat transfer rates are linearly proportional to temperature differences. Radiative heat transfer rates, on the other hand, are (for the most part) proportional to differences in temperature raised to the fourth power.

Gray-Diffuse Radiation

Airpak's radiation models assume the surfaces to be gray and diffuse. Emissivity and absorptivity of a gray surface are independent of the wavelength. Also, by Kirchoff's law [16], the emissivity equals the absorptivity ($\epsilon = \alpha$). For a diffuse surface, the reflectivity is independent of the outgoing (or incoming) directions.

As stated earlier, for applications of interest, the exchange of radiative energy between surfaces is virtually unaffected by the medium that separates them. Thus, according to the gray-body model, if a certain amount of radiation (E) is incident on a surface, a fraction (ρE) is reflected, a fraction (αE) is absorbed, and a fraction (τE) is transmitted. Since for most indoor applications the surfaces in question are opaque to internally-generated thermal radiation (in the infrared spectrum), the surfaces can be considered opaque. The transmissivity, therefore, can be neglected. It follows, from conservation of energy, that $\alpha + \rho = 1$, since $\alpha = \epsilon$ (emissivity), and $\rho = 1 - \epsilon$.

Radiative Transfer Equation

The radiative transfer equation (RTE) for an absorbing, emitting, and scattering medium at position $\vec{r}$ in the direction $\vec{s}$ is

$$\frac{dI(\vec{r},\vec{s})}{ds} + (a+\sigma_s)I(\vec{r},\vec{s}) = an^2\frac{\sigma T^4}{\pi} + \frac{\sigma_s}{4\pi}\int_0^{4\pi} I(\vec{r},\vec{s}\,')\ \Phi(\vec{s}\cdot\vec{s}\,')\ d\Omega' \tag{18.4-1}$$

where	$\vec{r}$	=	position vector
	$\vec{s}$	=	direction vector
	$\vec{s}\,'$	=	scattering direction vector
	s	=	path length
	a	=	absorption coefficient
	n	=	refractive index
	σ_s	=	scattering coefficient
	σ	=	Stefan-Boltzmann constant (5.672×10^{-8} W/m^2-K^4)
	I	=	radiation intensity, which depends on position ($\vec{r}$) and direction ($\vec{s}$)
	T	=	local temperature
	Φ	=	phase function
	Ω'	=	solid angle

$(a + \sigma_s)s$ is the optical thickness or opacity of the medium. The refractive index n is important when considering radiation in semi-transparent media. Figure 18.4.1 illustrates the process of radiation heat transfer.

18.4.2 The Surface-to-Surface Radiation Model

The default radiation model used in Airpak is the surface-to-surface radiation model. The energy flux leaving a given surface is composed of directly emitted and reflected energy. The reflected energy flux is dependent on the incident energy flux from the surroundings, which then can be expressed in terms of the energy flux leaving all other surfaces. The energy reflected from surface k is

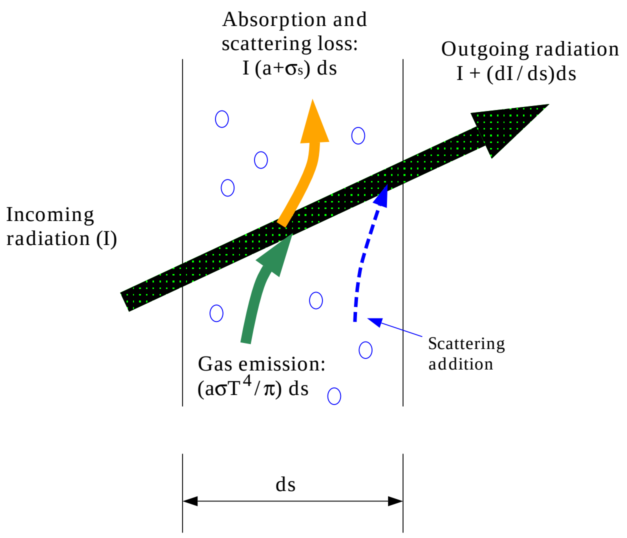
Absorption and scattering loss: I (a+σs) ds
Outgoing radiation I + (dI / ds)ds
Incoming radiation (I)
Scattering addition
Gas emission: (aσT4 / π) ds
ds

Figure 18.4.1: Radiation Heat Transfer

$$q_{\mathrm{out},k} = \epsilon_k \sigma T_k^4 + \rho_k q_{\mathrm{in},k} \tag{18.4-2}$$

where $q_{\mathrm{out},k}$ is the energy flux leaving the surface, ϵ_k is the emissivity, σ is the Boltzmann constant, and $q_{\mathrm{in},k}$ is the energy flux incident on the surface from the surroundings.

The amount of incident energy upon a surface from another surface is a direct function of the surface-to-surface "view factor," F_{jk}. The view factor F_{jk} is the fraction of energy leaving surface k that is incident on surface j. The incident energy flux q_{ik} can be expressed in terms of the energy flux leaving all other surfaces as

$$A_k q_{\mathrm{in},k} = \sum_{j=1}^{N} A_j q_{\mathrm{out},j} F_{jk} \tag{18.4-3}$$

where A_k is the area of surface k and F_{jk} is the view factor between surface k and surface j. For N surfaces, using the view factor reciprocity relationship gives

$$A_j F_{jk} = A_k F_{kj} \text{ for } j = 1, 2, 3, \ldots N \tag{18.4-4}$$

so that

$$q_{\text{in},k} = \sum_{j=1}^{N} F_{kj} q_{\text{out},j} \tag{18.4-5}$$

Therefore,

$$q_{\text{out},k} = \epsilon_k \sigma T_k^4 + \rho_k \sum_{j=1}^{N} F_{kj} q_{\text{out},j} \tag{18.4-6}$$

which can be written as

$$J_k = E_k + \rho_k \sum_{j=1}^{N} F_{kj} J_j \tag{18.4-7}$$

where J_k represents the energy that is given off (or radiosity) of surface k and E_k represents the emissive power of surface k. This represents N equations, which can be recast into matrix form as

$$\boldsymbol{K}\boldsymbol{J} = \boldsymbol{E} \tag{18.4-8}$$

where $\boldsymbol{K}$ is an $N \times N$ matrix, $\boldsymbol{J}$ is the radiosity vector, and $\boldsymbol{E}$ is the emissive power vector.

Equation 18.4-8 is referred to as the radiosity matrix equation. The view factor between two finite surfaces i and j is given by

$$F_{ij} = \frac{1}{A_i} \int_{A_i} \int_{A_j} \frac{\cos\theta_i \cos\theta_j}{\pi r^2} \delta_{ij} dA_i dA_j \tag{18.4-9}$$

where δ_{ij} is determined by the visibility of dA_j to dA_i. $\delta_{ij} = 1$ if dA_j is visible to dA_i and 0 otherwise.

18.4.3 The Discrete Ordinates (DO) Radiation Model

The discrete ordinates (DO) radiation model solves the radiative transfer equation (RTE) for a finite number of discrete solid angles, each associated with a vector direction $\vec{s}$ fixed in the global Cartesian system (x, y, z). The fineness of the angular discretization is set by default parameters that depend on the type of problem you are solving. The DO model transforms Equation 18.4-1 into a transport equation for radiation intensity in the spatial coordinates (x, y, z). The DO model solves for as many transport equations as there are directions $\vec{s}$. The solution method is identical to that used for the fluid flow and energy equations.

The implementation in Airpak uses a conservative variant of the discrete ordinates model called the finite-volume scheme [8, 19], and its extension to unstructured meshes [17].

The DO Equations

The DO model considers the radiative transfer equation (RTE) in the direction $\vec{s}$ as a field equation. Thus, Equation 18.4-1 is written as

$$\nabla \cdot (I(\vec{r}, \vec{s})\vec{s}) + (a + \sigma_s)I(\vec{r}, \vec{s}) = an^2 \frac{\sigma T^4}{\pi} + \frac{\sigma_s}{4\pi} \int_0^{4\pi} I(\vec{r}, \vec{s}\,')\ \Phi(\vec{s} \cdot \vec{s}\,')\ d\Omega' \tag{18.4-10}$$

Angular Discretization and Pixelation

Each octant of the angular space 4π at any spatial location is discretized into $N_\theta \times N_\phi$ solid angles of extent ω_i, called control angles. The angles θ and ϕ are the polar and azimuthal angles respectively, and are measured with respect to the global Cartesian system (x, y, z) as shown in Figure 18.4.2. The θ and ϕ extents of the control angle, $\Delta\theta$ and $\Delta\phi$, are constant.

When Cartesian meshes are used, it is possible to align the global angular discretization with the control volume face, as shown in Figure 18.4.3. For generalized unstructured meshes, however, control volume faces do

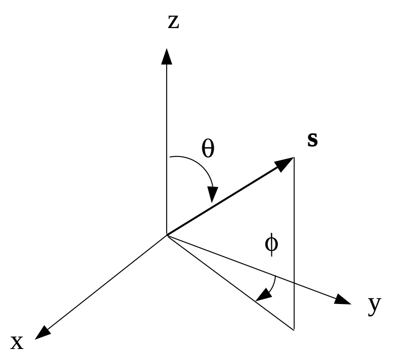

Figure 18.4.2: Angular Coordinate System

not in general align with the global angular discretization, as shown in Figure 18.4.4, leading to the problem of control angle overhang [17].

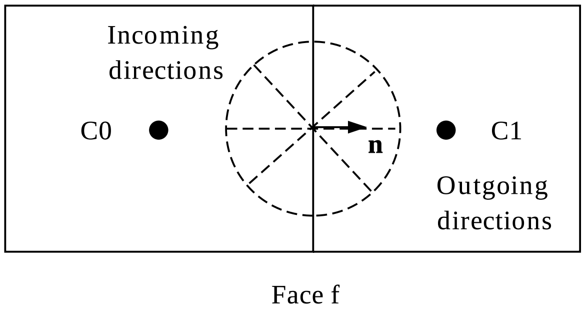

Figure 18.4.3: Face with No Control Angle Overhang

Essentially, control angles can straddle the control volume faces, so that they are partially incoming and partially outgoing to the face. Figure 18.4.5 shows a 3D example of a face with control angle overhang.

The control volume face cuts the sphere representing the angular space at an arbitrary angle. The line of intersection is a great circle. Control angle overhang may also occur as a result of reflection and refraction. It is important in these cases to correctly account for the overhanging fraction. This is done through the use of pixelation [17].

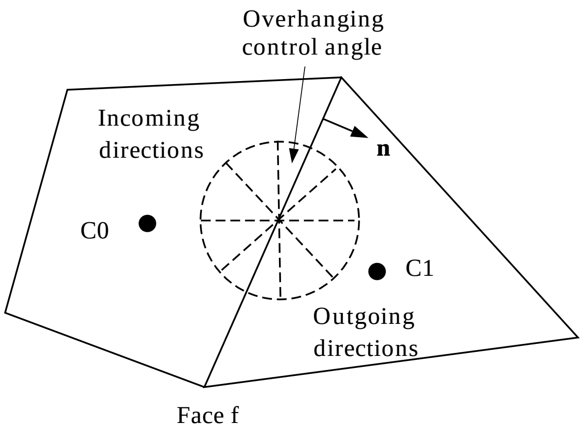

Figure 18.4.4: Face with Control Angle Overhang

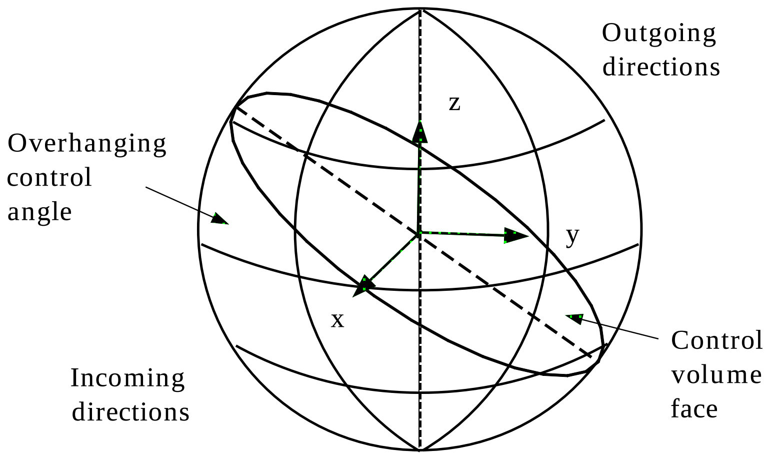

Figure 18.4.5: Face with Control Angle Overhang (3D)

Each overhanging control angle is divided into $N_{\theta_p} \times N_{\phi_p}$ pixels, as shown in Figure 18.4.6. The energy contained in each pixel is then treated as incoming or outgoing to the face. The influence of overhang can thus be accounted for within the pixel resolution. For problems involving gray-diffuse radiation, the default pixelation of 1×1 is usually sufficient. For problems involving symmetry boundaries, a pixelation of 3×3 is used.

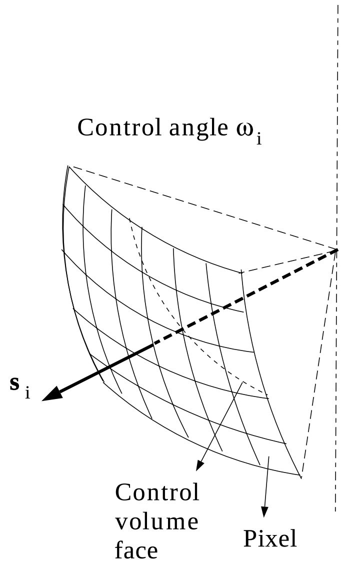

Figure 18.4.6: Pixelation of Control Angle

Boundary Condition Treatment at Gray-Diffuse Walls

For gray radiation, the incident radiation heat flux, $q_{\rm in}$, at the wall is

$$q_{\rm in} = \int_{\vec{s}\cdot\vec{n}>0} I_{\rm in} \vec{s} \cdot \vec{n} d\Omega \qquad (18.4\text{-}11)$$

The net radiative flux leaving the surface is given by

$$q_{\rm out} = (1 - \epsilon_w) q_{\rm in} + n^2 \epsilon_w \sigma T_w^4 \qquad (18.4\text{-}12)$$

where n is the refractive index of the medium next to the wall. The boundary intensity for all outgoing directions $\vec{s}$ at the wall is given by

$$I_0 = q_{\rm out}/\pi \tag{18.4-13}$$

Boundary Condition Treatment at Symmetry Boundaries

At symmetry boundaries, the direction of the reflected ray $\vec{s}_r$ corresponding to the incoming direction $\vec{s}$ is given by

$$\vec{s}_r = \vec{s} - 2\,(\vec{s}\cdot\vec{n})\,\vec{n} \tag{18.4-14}$$

Furthermore,

$$I_w(\vec{s}_r) = I_w(\vec{s}) \tag{18.4-15}$$

Boundary Condition Treatment at Flow Inlets and Exits

The net radiation heat flux at flow inlets and outlets is computed in the same manner as at walls, as described above. Airpak assumes that the emissivity of all flow inlets and outlets is 1.0 (black body absorption).

18.5 Solution Procedures

18.5.1 Overview of Numerical Scheme

Airpak will solve the governing integral equations for mass and momentum, and (when appropriate) for energy, species transport, and other scalars such as turbulence. A control-volume-based technique is used that consists of:

- Division of the domain into discrete control volumes using a computational grid.
- Integration of the governing equations on the individual control volumes to construct algebraic equations for the discrete dependent

variables ("unknowns") such as velocities, pressure, temperature, and conserved scalars.

- Linearization of the discretized equations and solution of the resultant linear equation system to yield updated values of the dependent variables.

The governing equations are solved sequentially (i.e., segregated from one another). Because the governing equations are non-linear (and coupled), several iterations of the solution loop must be performed before a converged solution is obtained. Each iteration consists of the steps illustrated in Figure 18.5.1 and outlined below:

1. Fluid properties are updated, based on the current solution. (If the calculation has just begun, the fluid properties will be updated based on the initialized solution.)
2. The u, v, and w momentum equations are each solved in turn using current values for pressure and face mass fluxes, in order to update the velocity field.
3. Since the velocities obtained in Step 2 may not satisfy the continuity equation locally, a "Poisson-type" equation for the pressure correction is derived from the continuity equation and the linearized momentum equations. This pressure correction equation is then solved to obtain the necessary corrections to the pressure and velocity fields and the face mass fluxes such that continuity is satisfied.
4. Where appropriate, equations for scalars such as turbulence, energy, species, and radiation are solved using the previously updated values of the other variables.
5. A check for convergence of the equation set is made.

These steps are continued until the convergence criteria are met.

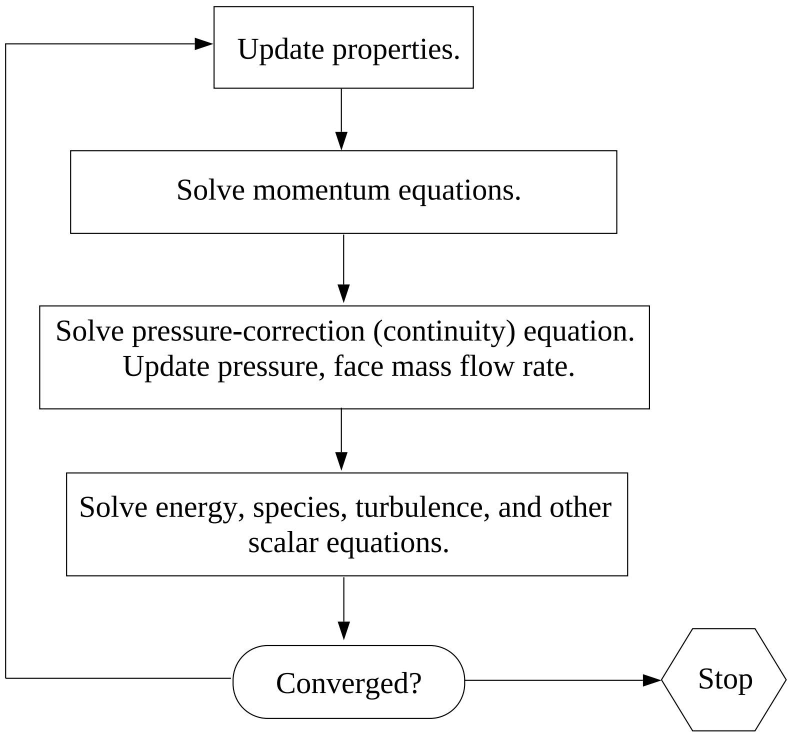
Update properties.
Solve momentum equations.
Solve pressure-correction (continuity) equation.
Update pressure, face mass flow rate.
Solve energy, species, turbulence, and other scalar equations.
Converged?
Stop

Figure 18.5.1: Overview of the Solution Method

Linearization

The discrete, non-linear governing equations are linearized to produce a system of equations for the dependent variables in every computational cell. The resultant linear system is then solved to yield an updated flow-field solution.

The manner in which the governing equations are linearized takes an "implicit" form with respect to the dependent variable (or set of variables) of interest. For a given variable, the unknown value in each cell is computed using a relation that includes both existing and unknown values from neighboring cells. Therefore each unknown will appear in more than one equation in the system, and these equations must be solved simultaneously to give the unknown quantities.

This will result in a system of linear equations with one equation for each cell in the domain. Because there is only one equation per cell, this is sometimes called a "scalar" system of equations. A point implicit (Gauss-Seidel) linear equation solver is used in conjunction with an algebraic multigrid (AMG) method to solve the resultant scalar system of equations for the dependent variable in each cell. For example, the x-momentum equation is linearized to produce a system of equations in which u velocity is the unknown. Simultaneous solution of this equation system (using the scalar AMG solver) yields an updated u-velocity field.

In summary, Airpak solves for a single variable field (e.g., p) by considering all cells at the same time. It then solves for the next variable field by again considering all cells at the same time, and so on.

18.5.2 Spatial Discretization

Airpak uses a control-volume-based technique to convert the governing equations to algebraic equations that can be solved numerically. This control volume technique consists of integrating the governing equations about each control volume, yielding discrete equations that conserve each quantity on a control-volume basis.

Discretization of the governing equations can be illustrated most easily by considering the steady-state conservation equation for transport of a scalar quantity ϕ. This is demonstrated by the following equation written in integral form for an arbitrary control volume V as follows:

$$\oint \rho\phi\,\vec{v}\cdot d\vec{A} = \oint \Gamma_\phi\,\nabla\phi\cdot d\vec{A} + \int_V S_\phi\,dV \qquad (18.5\text{-}1)$$

where

ρ	=	density
$\vec{v}$	=	velocity vector ($= u\,\hat{\imath} + v\,\hat{\jmath}$ in 2D)
$\vec{A}$	=	surface area vector
Γ_ϕ	=	diffusion coefficient for ϕ
$\nabla\phi$	=	gradient of ϕ ($= (\partial\phi/\partial x)\,\hat{\imath} + (\partial\phi/\partial y)\,\hat{\jmath}$ in 2D)
S_ϕ	=	source of ϕ per unit volume

Equation 18.5-1 is applied to each control volume, or cell, in the computational domain. The two-dimensional, triangular cell shown in Figure 18.5.2 is an example of such a control volume. Discretization of Equation 18.5-1 on a given cell yields

$$\sum_f^{N_{\text{faces}}} \vec{v}_f \phi_f \vec{A}_f = \sum_f^{N_{\text{faces}}} \Gamma_\phi \, (\nabla\phi)_n \vec{A}_f + S_\phi \, V \qquad (18.5\text{-}2)$$

where

N_{faces}	=	number of faces enclosing cell
ϕ_f	=	value of ϕ convected through face f
$\rho_f \vec{v}_f \cdot \vec{A}_f$	=	mass flux through the face
$\vec{A}_f$	=	area of face f, $\lvert A\rvert$ ($= \lvert A_x\hat{\imath} + A_y\hat{\jmath}\rvert$ in 2D)
$(\nabla\phi)_n$	=	magnitude of $\nabla\phi$ normal to face f
V	=	cell volume

The equations solved by Airpak take the same general form as the one given above and apply readily to multi-dimensional, unstructured meshes composed of arbitrary polyhedra.

Airpak stores discrete values of the scalar ϕ at the cell centers ($c0$ and $c1$ in Figure 18.5.2). However, face values ϕ_f are required for the convection terms in Equation 18.5-2 and must be interpolated from the cell center values. This is accomplished using an upwind scheme.

Upwinding means that the face value ϕ_f is derived from quantities in the cell upstream, or "upwind," relative to the direction of the normal velocity v_n in Equation 18.5-2. Airpak allows you to choose from two upwind schemes: first-order upwind, and second-order upwind. These schemes are described below.

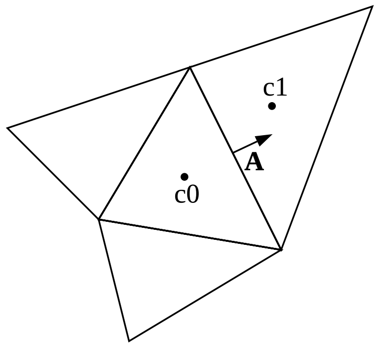

Figure 18.5.2: Control Volume Used to Illustrate Discretization of a Scalar Transport Equation

The diffusion terms in Equation 18.5-2 are central-differenced and are always second-order accurate.

First-Order Upwind Scheme

When first-order accuracy is desired, quantities at cell faces are determined by assuming that the cell-center values of any field variable represent a cell-average value and hold throughout the entire cell; the face quantities are identical to the cell quantities. Thus when first-order upwinding is selected, the face value ϕ_f is set equal to the cell-center value of ϕ in the upstream cell.

Second-Order Upwind Scheme

When second-order accuracy is desired, quantities at cell faces are computed using a multidimensional linear reconstruction approach [3]. In this approach, higher-order accuracy is achieved at cell faces through a Taylor series expansion of the cell-centered solution about the cell centroid. Thus when second-order upwinding is selected, the face value ϕ_f is computed using the following expression:

$$\phi_f = \phi + \nabla\phi \cdot \Delta\vec{s} \qquad (18.5\text{-}3)$$

where ϕ and $\nabla\phi$ are the cell-centered value and its gradient in the upstream cell, and $\Delta\vec{s}$ is the displacement vector from the upstream cell centroid to the face centroid. This formulation requires the determination of the gradient $\nabla\phi$ in each cell. This gradient is computed using the divergence theorem, which in discrete form is written as

$$\nabla\phi = \frac{1}{V} \sum_{f}^{N_{\text{faces}}} \tilde{\phi}_f \vec{A} \tag{18.5-4}$$

Here the face values $\tilde{\phi}_f$ are computed by averaging ϕ from the two cells adjacent to the face. Finally, the gradient $\nabla\phi$ is limited so that no new maxima or minima are introduced.

Linearized Form of the Discrete Equation

The discretized scalar transport equation (Equation 18.5-2) contains the unknown scalar variable ϕ at the cell center as well as the unknown values in surrounding neighbor cells. This equation will, in general, be non-linear with respect to these variables. A linearized form of Equation 18.5-2 can be written as

$$a_P \phi = \sum_{nb} a_{nb}\phi_{nb} + b \tag{18.5-5}$$

where the subscript nb refers to neighbor cells, and a_P and a_{nb} are the linearized coefficients for ϕ and ϕ_{nb}.

The number of neighbors for each cell depends on the grid topology, but will typically equal the number of faces enclosing the cell (boundary cells being the exception).

Similar equations can be written for each cell in the grid. This results in a set of algebraic equations with a sparse coefficient matrix. For scalar equations, Airpak solves this linear system using a point implicit (Gauss-Seidel) linear equation solver in conjunction with an algebraic multigrid (AMG) method which is described in Section 18.5.4.

Under-Relaxation

Because of the nonlinearity of the equation set being solved by Airpak, it is necessary to control the change of ϕ. This is typically achieved by under-relaxation, which reduces the change of ϕ produced during each iteration. In a simple form, the new value of the variable ϕ within a cell depends upon the old value, $\phi_{\rm old}$, the computed change in ϕ, $\Delta\phi$, and the under-relaxation factor, α, as follows:

$$\phi = \phi_{\rm old} + \alpha \Delta\phi \tag{18.5-6}$$

Discretization of the Momentum and Continuity Equations

In this section, special practices related to the discretization of the momentum and continuity equations and their solution are addressed. These practices are most easily described by considering the steady-state continuity and momentum equations in integral form:

$$\oint \rho\, \vec{v} \cdot d\vec{A} = 0 \tag{18.5-7}$$

$$\oint \rho \vec{v}\, \vec{v} \cdot d\vec{A} = - \oint p\boldsymbol{I} \cdot d\vec{A} + \oint \overline{\overline{\tau}} \cdot d\vec{A} + \int_V \vec{F}\, dV \tag{18.5-8}$$

where $\boldsymbol{I}$ is the identity matrix, $\overline{\overline{\tau}}$ is the stress tensor, and $\vec{F}$ is the force vector.

Discretization of the Momentum Equation

The discretization scheme described earlier in this section for a scalar transport equation is also used to discretize the momentum equations. For example, the x-momentum equation can be obtained by setting $\phi = u$:

$$a_P\, u = \sum_{nb} a_{nb}\, u_{nb} + \sum p_f \mathrm{A} \cdot \hat{\boldsymbol{\imath}} + S \tag{18.5-9}$$

If the pressure field and face mass fluxes were known, Equation 18.5-9 could be solved in the manner outlined earlier in this section, and a velocity field obtained. However, the pressure field and face mass fluxes are not known a priori and must be obtained as a part of the solution. There are important issues with respect to the storage of pressure and the discretization of the pressure gradient term; these are addressed later in this section.

Airpak uses a co-located scheme, whereby pressure and velocity are both stored at cell centers. However, Equation 18.5-9 requires the value of the pressure at the face between cells $c0$ and $c1$, shown in Figure 18.5.2. Therefore, an interpolation scheme is required to compute the face values of pressure from the cell values.

Pressure Interpolation Schemes

The default pressure interpolation scheme in Airpak is the body-force-weighted scheme. This scheme is good for high-Rayleigh-number natural convection flows. The body-force-weighted scheme computes the pressure values at the faces by assuming that the normal acceleration of the fluid resulting from the pressure gradient and body forces is continuous across each face. This works well if the body forces are known explicitly in the momentum equations (e.g., buoyancy calculations).

If buoyancy effects are not important in your model, it is recommended that you use the standard pressure scheme in Airpak. This interpolates the face pressure using momentum equation coefficients [20]. This procedure works well if the pressure variation between cell centers is smooth. When there are jumps or large gradients in the momentum source terms between control volumes, the pressure profile has a high gradient at the cell face, and cannot be interpolated using this scheme. If this scheme is used, the discrepancy shows up in overshoots/undershoots of cell velocity.

Another source of error for the standard pressure scheme is that Airpak assumes that the normal pressure gradient at the wall is zero. This is valid for boundary layers, but not in the presence of body forces or curvature. Again, the failure to correctly account for the wall pressure gradient is manifested in velocity vectors pointing in/out of walls.

If you require a more accurate solution to your problem, Airpak provides a second-order pressure interpolation scheme. This reconstructs the face pressure using the reconstructed gradient of pressure in a manner similar to the method used for second-order-accurate convection terms (see the section above on second-order upwind schemes for details). This scheme may provide some improvement over the standard scheme, but it may have some trouble if it is used at the start of a calculation and/or with a poor-quality mesh.

Discretization of the Continuity Equation

Equation 18.5-7 may be integrated over the control volume in Figure 18.5.2 to yield the following discrete equation

$$\sum_{f}^{N_{\text{faces}}} J_f A_f = 0 \qquad (18.5\text{-}10)$$

where J_f is the mass flow rate through face f, ρv_n.

As described in Section 18.5.1, the momentum and continuity equations are solved sequentially. In this sequential procedure, the continuity equation is used as an equation for pressure. However, pressure does not appear explicitly in Equation 18.5-10 for incompressible flows, since density is not directly related to pressure. The SIMPLE (Semi-Implicit Method for Pressure-Linked Equations) algorithm [18] is used for introducing pressure into the continuity equation. This procedure is outlined below.

In order to proceed further, it is necessary to relate the face values of velocity v_n to the stored values of velocity at the cell centers. Linear interpolation of cell-centered velocities to the face results in unphysical checker-boarding of pressure. Airpak uses a procedure similar to that outlined by Rhie and Chow [20] to prevent checkerboarding. The face value of velocity v_n is not averaged linearly; instead, momentum-weighted averaging, using weighting factors based on the a_P coefficient from equation 18.5-9, is performed. Using this procedure, the face flow rate J_f may be written as

$$J_f = \hat{J}_f + d_f\,(p_{c0} - p_{c1}) \tag{18.5-11}$$

where p_{c0} and p_{c1} are the pressures within the two cells on either side of the face, and $\hat{J}_f$ contains the influence of velocities in these cells (see Figure 18.5.2). The term d_f is a function of $\bar{a}_P$, the average of the momentum equation a_P coefficients for the cells on either side of face f.

Pressure-Velocity Coupling with SIMPLE

Pressure-velocity coupling is achieved by using Equation 18.5-11 to derive an equation for pressure from the discrete continuity equation (Equation 18.5-10). Airpak uses the SIMPLE (Semi-Implicit Method for Pressure-Linked Equations) pressure-velocity coupling algorithm. The SIMPLE algorithm uses a relationship between velocity and pressure corrections to enforce mass conservation and to obtain the pressure field.

If the momentum equation is solved with a guessed pressure field p^*, the resulting face flux J_f^* computed from Equation 18.5-11

$$J_f^* = \hat{J}_f^* + d_f\,(p_{c0}^* - p_{c1}^*) \tag{18.5-12}$$

does not satisfy the continuity equation. Consequently, a correction J_f' is added to the face flow rate J_f^* so that the corrected face flow rate J_f

$$J_f = J_f^* + J_f' \tag{18.5-13}$$

satisfies the continuity equation. The SIMPLE algorithm postulates that J_f' be written as

$$J_f' = d_f\,(p_{c0}' - p_{c1}') \tag{18.5-14}$$

where p' is the cell pressure correction.

The SIMPLE algorithm substitutes the flux correction equations (Equations 18.5-13 and 18.5-14) into the discrete continuity equation (Equation 18.5-10) to obtain a discrete equation for the pressure correction p' in the cell:

$$a_P\, p' = \sum_{nb} a_{nb}\, {p'}_{nb} + b \tag{18.5-15}$$

where the source term b is the net flow rate into the cell:

$$b = \sum_{f}^{N_{\rm faces}} J_f^* \tag{18.5-16}$$

The pressure-correction equation (Equation 18.5-15) may be solved using the algebraic multigrid (AMG) method described in Section 18.5.4. Once a solution is obtained, the cell pressure and the face flow rate are corrected using

$$p = p^* + \alpha_p\, p' \tag{18.5-17}$$

$$J_f = J_f^* + d_f\, (p'_{c0} - p'_{c1}) \tag{18.5-18}$$

Here α_p is the under-relaxation factor for pressure (see Equation 18.5-6 and related description for information about under-relaxation). The corrected face flow rate J_f satisfies the discrete continuity equation identically during each iteration.

18.5.3 Time Discretization

In Airpak the time-dependent equations must be discretized in both space and time. The spatial discretization for the time-dependent equations is identical to the steady-state case (see Section 18.5.2). Temporal discretization involves the integration of every term in the differential equations over a time step Δt. The integration of the transient terms is straightforward, as shown below.

A generic expression for the time evolution of a variable ϕ is given by

$$\frac{\partial \phi}{\partial t} = F(\phi) \tag{18.5-19}$$

where the function F incorporates any spatial discretization. If the time derivative is discretized using backward differences, the first-order accurate temporal discretization is given by

$$\frac{\phi^{n+1} - \phi^n}{\Delta t} = F(\phi) \tag{18.5-20}$$

where

ϕ	=	a scalar quantity
$n+1$	=	value at the next time level, $t + \Delta t$
n	=	value at the current time level, t

Once the time derivative has been discretized, a choice remains for evaluating $F(\phi)$: in particular, which time level values of ϕ should be used in evaluating F?

One method (the method used in Airpak) is to evaluate $F(\phi)$ at the future time level:

$$\frac{\phi^{n+1} - \phi^n}{\Delta t} = F(\phi^{n+1}) \tag{18.5-21}$$

This is referred to as “implicit” integration since ϕ^{n+1} in a given cell is related to ϕ^{n+1} in neighboring cells through $F(\phi^{n+1})$:

$$\phi^{n+1} = \phi^n + \Delta t F(\phi^{n+1}) \tag{18.5-22}$$

This implicit equation can be solved iteratively by initializing ϕ^i to ϕ^n and iterating the equation

$$\phi^i = \phi^n + \Delta t F(\phi^i) \tag{18.5-23}$$

until ϕ^i stops changing (i.e., converges). At that point, ϕ^{n+1} is set to ϕ^i.

The advantage of the fully implicit scheme is that it is unconditionally stable with respect to time step size.

18.5.4 Multigrid Method

This section describes the mathematical basis of the multigrid approach used in Airpak.

Approach

Airpak uses a multigrid scheme to accelerate the convergence of the solver by computing corrections on a series of coarse grid levels. The use of this multigrid scheme can greatly reduce the number of iterations and the CPU time required to obtain a converged solution, particularly when your model contains a large number of control volumes.

The Need for Multigrid

Implicit solution of the linearized equations on unstructured meshes is complicated by the fact that there is no equivalent of the line-iterative methods that are commonly used on structured grids. Since direct matrix inversion is out of the question for realistic problems and “whole-field” solvers that rely on conjugate-gradient (CG) methods have robustness problems associated with them, the methods of choice are point implicit solvers like Gauss-Seidel. Although the Gauss-Seidel scheme rapidly removes local (high-frequency) errors in the solution, global (low-frequency) errors are reduced at a rate inversely related to the grid size. Thus, for a large number of nodes, the solver “stalls” and the residual reduction rate becomes prohibitively low.

Multigrid techniques allow global error to be addressed by using a sequence of successively coarser meshes. This method is based upon the principle that global (low-frequency) error existing on a fine mesh can be represented on a coarse mesh where it again becomes accessible as local (high-frequency) error: because there are fewer coarse cells overall, the global corrections can be communicated more quickly between adjacent cells. Since computations can be performed at exponentially decaying expense in both CPU time and memory storage on coarser meshes, there is the potential for very efficient elimination of global error. The fine-grid relaxation scheme or “smoother”, in this case either the point-implicit Gauss-Seidel or the explicit multi-stage scheme, is not required to be particularly effective at reducing global error and can be tuned for efficient reduction of local error.

The Basic Concept in Multigrid

Consider the set of discretized linear (or linearized) equations given by

$$A\,\phi_e + b = 0 \tag{18.5-24}$$

where ϕ_e is the exact solution. Before the solution has converged there will be a defect d associated with the approximate solution ϕ:

$$A\,\phi + b = d \tag{18.5-25}$$

We seek a correction ψ to ϕ such that the exact solution is given by

$$\phi_e = \phi + \psi \tag{18.5-26}$$

Substituting Equation 18.5-26 into Equation 18.5-24 gives

$$\begin{aligned} A\,(\phi + \psi) + b &= 0 && (18.5\text{-}27) \\ A\,\psi + (A\,\phi + b) &= 0 && (18.5\text{-}28) \end{aligned}$$

Now using Equations 18.5-25 and 18.5-28 we obtain

$$A\,\psi + d = 0 \tag{18.5-29}$$

which is an equation for the correction in terms of the original fine level operator A and the defect d. Assuming the local (high-frequency) errors have been sufficiently damped by the relaxation scheme on the fine level, the correction ψ will be smooth and therefore more effectively solved on the next coarser level.

Restriction and Prolongation

Solving for corrections on the coarse level requires transferring the defect down from the fine level (restriction), computing corrections, and then transferring the corrections back up from the coarse level (prolongation). We can write the equations for coarse level corrections ψ^H as

$$A^H \, \psi^H + R \, d = 0 \tag{18.5-30}$$

where A^H is the coarse level operator and R the restriction operator responsible for transferring the fine level defect down to the coarse level. Solution of Equation 18.5-30 is followed by an update of the fine level solution given by

$$\phi^{\text{new}} = \phi + P \, \psi^H \tag{18.5-31}$$

where P is the prolongation operator used to transfer the coarse level corrections up to the fine level.

Unstructured Multigrid

The primary difficulty with using multigrid on unstructured grids is the creation and use of the coarse grid hierarchy. On a structured grid, the coarse grids can be formed simply by removing every other grid line from the fine grids and the prolongation and restriction operators are simple to formulate (e.g., injection and bilinear interpolation).

Multigrid Cycles

A multigrid cycle can be defined as a recursive procedure that is applied at each grid level as it moves through the grid hierarchy. Three types of multigrid cycles are available in Airpak: the V, W, and flexible ("flex") cycles.

The V and W Cycles

Figures 18.5.3 and 18.5.4 show the V and W multigrid cycles (defined below). In each figure, the multigrid cycle is represented by a square, and then expanded to show the individual steps that are performed within the cycle. You may want to follow along in the figures as you read the steps below.

For the V and W cycles, the traversal of the hierarchy is governed by three parameters, β_1, β_2, and β_3, as follows:

1. β_1 "smoothings", (sometimes called pre-relaxation sweeps), are performed at the current grid level to reduce the high-frequency components of the error (local error).

   In Figures 18.5.3 and 18.5.4 this step is represented by a circle and marks the start of a multigrid cycle. The high-wave-number components of error should be reduced until the remaining error is expressible on the next coarser mesh without significant aliasing.

   If this is the coarsest grid level, then the multigrid cycle on this level is complete. (In Figures 18.5.3 and 18.5.4 there are 3 coarse grid levels, so the square representing the multigrid cycle on level 3 is equivalent to a circle, as shown in the final diagram in each figure.)

   ! In Airpak, β_1 is zero (i.e., pre-relaxation is not performed).

2. Next, the problem is "restricted" to the next coarser grid level using the appropriate restriction operator.

   In Figures 18.5.3 and 18.5.4, the restriction from a finer grid level to a coarser grid level is designated by a downward-sloping line.

3. The error on the coarse grid is reduced by performing β_2 multigrid cycles (represented in Figures 18.5.3 and 18.5.4 as squares). Commonly, for fixed multigrid strategies β_2 is either 1 or 2, corresponding to V-cycle and W-cycle multigrid, respectively.

4. Next, the cumulative correction computed on the coarse grid is "interpolated" back to the fine grid using the appropriate prolongation operator and added to the fine grid solution.

   In Figures 18.5.3 and 18.5.4 the prolongation is represented by an upward-sloping line.

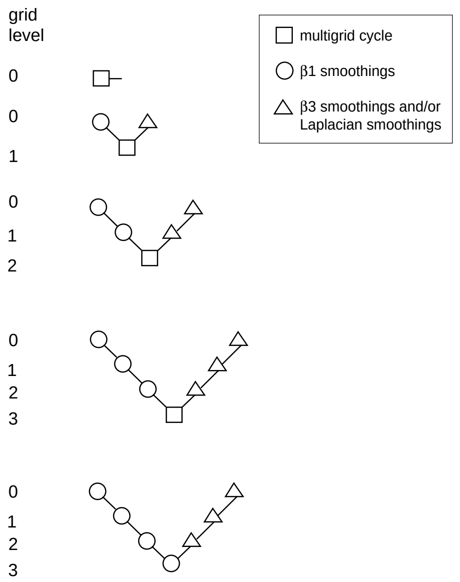

Figure 18.5.3: V-Cycle Multigrid

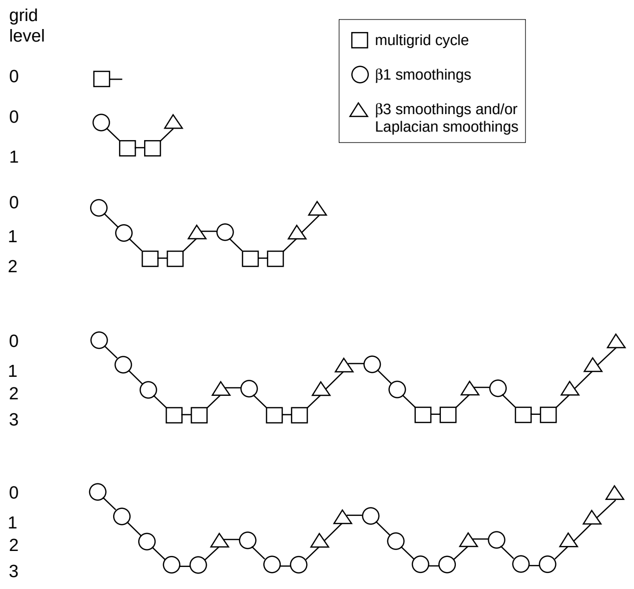

Figure 18.5.4: W-Cycle Multigrid

The high-frequency error now present at the fine grid level is due to the prolongation procedure used to transfer the correction.

5. In the final step, β_3 "smoothings" (post-relaxations) are performed to remove the high-frequency error introduced on the coarse grid by the β_2 multigrid cycles.

   In Figures 18.5.3 and 18.5.4, this relaxation procedure is represented by a single triangle.

The Flexible Cycle

For the flexible cycle, the calculation and use of coarse grid corrections is controlled in the multigrid procedure by the logic illustrated in Figure 18.5.5. This logic ensures that coarser grid calculations are invoked when the rate of residual reduction on the current grid level is too slow. In addition, the multigrid controls dictate when the iterative solution of the correction on the current coarse grid level is sufficiently converged and should thus be applied to the solution on the next finer grid. These two decisions are controlled by the parameters α and β shown in Figure 18.5.5, as described in detail below. Note that the logic of the multigrid procedure is such that grid levels may be visited repeatedly during a single global iteration on an equation. For a set of 4 multigrid levels, referred to as 0, 1, 2, and 3, the flex-cycle multigrid procedure for solving a given transport equation might consist of visiting grid levels as 0-1-2-3-2-3-2-1-0-1-2-1-0, for example.

The main difference between the flexible cycle and the V and W cycles is that the satisfaction of the residual reduction tolerance and termination criterion determine when and how often each level is visited in the flexible cycle, whereas in the V and W cycles the traversal pattern is explicitly defined.

The Residual Reduction Rate Criteria

The multigrid procedure invokes calculations on the next coarser grid level when the error reduction rate on the current level is insufficient, as defined by

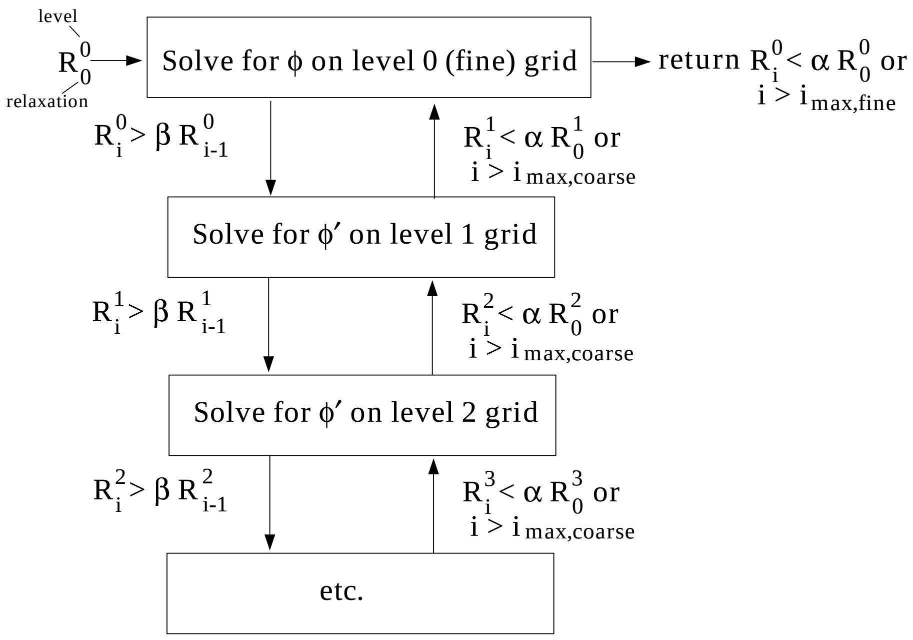
level
R$^0_0$
relaxation
Solve for ϕ on level 0 (fine) grid
return R^0_i < α R$^0_0$ or i > i$_{max,fine}$
R^0_i > β R$^0_{i-1}$
R^1_i < α R$^1_0$ or i > i$_{max,coarse}$
Solve for ϕ′ on level 1 grid
R^1_i > β R$^1_{i-1}$
R^2_i < α R$^2_0$ or i > i$_{max,coarse}$
Solve for ϕ′ on level 2 grid
R^2_i > β R$^2_{i-1}$
R^3_i < α R$^3_0$ or i > i$_{max,coarse}$
etc.

Figure 18.5.5: Logic Controlling the Flex Multigrid Cycle

$$R_i > \beta R_{i-1} \tag{18.5-32}$$

Here R_i is the absolute sum of residuals (defect) computed on the current grid level after the ith relaxation on this level. The above equation states that if the residual present in the iterative solution after i relaxations is greater than some fraction, β (between 0 and 1), of the residual present after the $(i-1)$th relaxation, the next coarser grid level should be visited. Thus β is referred to as the residual reduction tolerance, and determines when to "give up" on the iterative solution at the current grid level and move to solving the correction equations on the next coarser grid. The value of β controls the frequency with which coarser grid levels are visited. The default value is 0.1. A larger value will result in less frequent visits, and a smaller value will result in more frequent visits.

The Termination Criteria

Provided that the residual reduction rate is sufficiently rapid, the correction equations will be converged on the current grid level and the result applied to the solution field on the next finer grid level.

The correction equations on the current grid level are considered sufficiently converged when the error in the correction solution is reduced to some fraction, α (between 0 and 1), of the original error on this grid level:

$$R_i < \alpha R_0 \tag{18.5-33}$$

Here, R_i is the residual on the current grid level after the ith iteration on this level, and R_0 is the residual that was initially obtained on this grid level at the current global iteration. The parameter α, referred to as the termination criterion, has a default value of 0.1. Note that the above equation is also used to terminate calculations on the lowest (finest) grid level during the multigrid procedure. Thus, relaxations are continued on each grid level (including the finest grid level) until the criterion of this equation is obeyed (or until a maximum number of relaxations has been completed, in the case that the specified criterion is never achieved).

Restriction, Prolongation, and Coarse-Level Operators

The multigrid algorithm in Airpak is referred to as an "algebraic" multigrid (AMG) scheme because, as we shall see, the coarse level equations are generated without the use of any geometry or re-discretization on the coarse levels; a feature that makes AMG particularly attractive for use on unstructured meshes. The advantage is that no coarse grids have to be constructed or stored, and no fluxes or source terms need be evaluated on the coarse levels. This approach is in contrast with FAS (sometimes called "geometric") multigrid in which a hierarchy of meshes is required and the discretized equations are evaluated on every level. In theory, the advantage of FAS over AMG is that the former should perform better for non-linear problems since non-linearities in the system are carried down to the coarse levels through the re-discretization; when using AMG, once the system is linearized, non-linearities are not "felt" by the solver until the fine level operator is next updated.

AMG Restriction and Prolongation Operators

The restriction and prolongation operators used here are based on the additive correction (AC) strategy described for structured grids by Hutchinson and Raithby [11]. Inter-level transfer is accomplished by piecewise constant interpolation and prolongation. The defect in any coarse level cell is given by the sum of those from the fine level cells it contains, while fine level corrections are obtained by injection of coarse level values. In this manner the prolongation operator is given by the transpose of the restriction operator

$$P = R^T \tag{18.5-34}$$

The restriction operator is defined by a coarsening or "grouping" of fine level cells into coarse level ones. In this process each fine level cell is grouped with one or more of its "strongest" neighbors, with a preference given to currently ungrouped neighbors. The algorithm attempts to collect cells into groups of fixed size, typically two or four, but any number can be specified. In the context of grouping, strongest refers to the neighbor j of the current cell i for which the coefficient A_{ij} is largest. For sets of coupled equations A_{ij} is a block matrix and the measure of its magnitude is simply taken to be the magnitude of its first element. In addition, the set of coupled equations for a given cell are treated together and not divided amongst different coarse cells. This results in the same coarsening for each equation in the system.

AMG Coarse Level Operator

The coarse level operator A^H is constructed using a Galerkin approach. Here we require that the defect associated with the corrected fine level solution must vanish when transferred back to the coarse level. Therefore we may write

$$R\, d^{\text{new}} = 0 \tag{18.5-35}$$

Upon substituting Equations 18.5-25 and 18.5-31 for d^{new} and ϕ^{new} we have

$$\begin{aligned} R\,[A\,\phi^{\text{new}} + b] &= 0 \\ R\left[A\left(\phi + P\,\psi^H\right) + b\right] &= 0 \end{aligned} \tag{18.5-36}$$

Now rearranging and using Equation 18.5-25 once again gives

$$\begin{aligned} R\,A\,P\,\psi^H + R\,(A\,\phi + b) &= 0 \\ R\,A\,P\,\psi^H + R\,d &= 0 \end{aligned} \tag{18.5-37}$$

Comparison of Equation 18.5-37 with Equation 18.5-30 leads to the following expression for the coarse level operator:

$$A^H = R\,A\,P \tag{18.5-38}$$

The construction of coarse level operators thus reduces to a summation of diagonal and corresponding off-diagonal blocks for all fine level cells within a group to form the diagonal block of that group's coarse cell.

18.5.5 Solution Residuals

During the solution process you can monitor the convergence dynamically by checking residuals. At the end of each solver iteration, the residual sum for each of the conserved variables is computed and stored, thus recording the convergence history. This history is also saved in the data file. The residual sum is defined below.

On a computer with infinite precision, these residuals will go to zero as the solution converges. On an actual computer, the residuals decay to some small value ("round-off") and then stop changing ("level out"). For "single precision" computations (the default for workstations and most computers), residuals can drop as many as six orders of magnitude before hitting round-off. Double precision residuals can drop up to twelve orders of magnitude. Guidelines for judging convergence can be found in Section 14.10.5.

After discretization, the conservation equation for a general variable ϕ at a cell P can be written as

$$a_P \phi_P = \sum_{nb} a_{nb} \phi_{nb} + b \qquad (18.5\text{-}39)$$

Here a_P is the center coefficient, a_{nb} are the influence coefficients for the neighboring cells, and b is the contribution of the constant part of the source term S_c in $S = S_c + S_P \phi$ and of the boundary conditions. In Equation 18.5-39,

$$a_P = \sum_{nb} a_{nb} - S_P \qquad (18.5\text{-}40)$$

The residual R^ϕ computed by Airpak is the imbalance in Equation 18.5-39 summed over all the computational cells P. This is referred to as the "unscaled" residual. It may be written as

$$R^\phi = \sum_{\text{cells } P} \left| \sum_{nb} a_{nb} \phi_{nb} + b - a_P \phi_P \right| \qquad (18.5\text{-}41)$$

In general, it is difficult to judge convergence by examining the residuals defined by Equation 18.5-41 since no scaling is employed. This is especially true in enclosed flows such as natural convection in a room where there is no inlet flow rate of ϕ with which to compare the residual. Airpak scales the residual using a scaling factor representative of the flow rate of ϕ through the domain. This "scaled" residual is defined as

$$R^\phi = \frac{\sum_{\text{cells } P} \left| \sum_{nb} a_{nb} \phi_{nb} + b - a_P \phi_P \right|}{\sum_{\text{cells } P} \left| a_P \phi_P \right|} \qquad (18.5\text{-}42)$$

For the momentum equations the denominator term $a_P \phi_P$ is replaced by $a_P v_P$, where v_P is the magnitude of the velocity at cell P.

The scaled residual is a more appropriate indicator of convergence, and is the residual displayed by Airpak.

For the continuity equation, the unscaled residual is defined as

$$R^c = \sum_{\text{cells } P} |\text{rate of mass creation in cell P}| \tag{18.5-43}$$

The scaled residual for the continuity equation is defined as

$$\frac{R^c_{\text{iteration } N}}{R^c_{\text{iteration } 5}} \tag{18.5-44}$$

The denominator is the largest absolute value of the continuity residual in the first five iterations.