DEPARTMENT OF AEROSPACE ENGINEERING

Faculty Name	:	**Dr.A.Arun Negemiya, ASP/ Aero**	Academic Year	:	**2024-2025 (Even)**
Year & Branch	:	**III AEROSPACE**	Semester	:	**VI**
Course	:	**19ASB304 - Computational Fluid Dynamics for Aerospace Application**			

UNIT V – FLOW FIELD ANALYSIS AND TURBULENCE MODELS

Pressure and Velocity Corrections: SIMPLE and PISO Algorithms

Pressure-velocity coupling is achieved by using Equation 5.4-6 to derive an additional condition for pressure by reformatting the continuity equation (Equation 5.4-5). The pressure-based solver allows you to solve your flow problem in either a segregated or coupled manner. **ANSYS FLUENT** provides the option to choose among five pressure-velocity coupling algorithms: SIMPLE, SIMPLEC, PISO, Coupled, and (for unsteady flows using the non-iterative time advancement scheme (NITA)) Fractional Step (FSM). All the aforementioned schemes, except the "coupled" scheme, are based on the predictor-corrector approach. Note that **SIMPLE**, **SIMPLEC**, **PISO**, and **Fractional Step** use the pressure-based segregated algorithm, while **Coupled** uses the pressure-based coupled solver.

The pressure-velocity coupling schemes that are applicable when using the Eulerian multiphase model are **Phase-coupled SIMPLE**, **multiphase-coupled**, and **Full multiphase-coupled**.

Segregated Algorithms

SIMPLE

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 5.4-6

$$J_f^* = \hat{J}_f^* + d_f (p_{c0}^* - p_{c1}^*) \tag{5.4-7}$$

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

$$J_f = J_f^* + J_f' \tag{5.4-8}$$

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

$$J_f' = d_f (p_{c0}' - p_{c1}') \tag{5.4-9}$$

where p' is the cell pressure correction.

The SIMPLE algorithm substitutes the flux correction equations (Equations 5.4-8 and 5.4-9) into the discrete continuity equation (Equation 5.4-5) to obtain a discrete equation for the pressure correction p' in the cell:

$$a_P p' = \sum_{\text{nb}} a_{\text{nb}} p_{\text{nb}}' + b \tag{5.4-10}$$

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

$$b = \sum_{f}^{N_{\text{faces}}} J_f^* A_f \tag{5.4-11}$$

The pressure-correction equation (Equation 5.4-10) may be solved using the algebraic multigrid (AMG) method described in Section 5.6.3. Once a solution is obtained, the cell pressure and the face flux are corrected using

$$p = p^* + \alpha_p p' \tag{5.4-12}$$

$$J_f = J_f^* + d_f (p'_{c0} - p'_{c1}) \tag{5.4-13}$$

Here α_p is the under-relaxation factor for pressure (see Section 5.4.4 for information about under-relaxation). The corrected face flux, J_f, satisfies the discrete continuity equation identically during each iteration.

SIMPLEC

Many variants of the basic SIMPLE algorithm are available in the literature. In addition to SIMPLE, **ANSYS FLUENT** offers the SIMPLEC (SIMPLE-Consistent) algorithm. SIMPLE is the default, but many problems will benefit from the use of SIMPLECThe SIMPLEC procedure is similar to the SIMPLE procedure outlined above. The only difference lies in the expression used for the face flux correction, J'_f. As in SIMPLE, the correction equation may be written as

$$J_f = J_f^* + d_f (p'_{c0} - p'_{c1}) \tag{5.4-14}$$

However, the coefficient d_f is redefined as a function of $\overline{(a_P - \sum_{\text{nb}} a_{\text{nb}})}$. The use of this modified correction equation has been shown to accelerate convergence in problems where pressure-velocity coupling is the main deterrent to obtaining a solution.

Skewness Correction

For meshes with some degree of skewness, the approximate relationship between the correction of mass flux at the cell face and the difference of the pressure corrections at the adjacent cells is very rough. Since the components of the pressure-correction gradient along the cell faces are not known in advance, an iterative process similar to the PISO neighbor correction described below is desirable. After the initial solution of the pressure-correction equation, the pressure-correction gradient is recalculated and used to update the mass flux corrections. This process, which is referred to as "skewness correction", significantly reduces convergence difficulties associated with highly distorted meshes. The SIMPLEC skewness correction allows **ANSYS**

FLUENT to obtain a solution on a highly skewed mesh in approximately the same number of iterations as required for a more orthogonal mesh.

PISO

The Pressure-Implicit with Splitting of Operators (PISO) pressure-velocity coupling scheme, part of the SIMPLE family of algorithms, is based on the higher degree of the approximate relation between the corrections for pressure and velocity. One of the limitations of the SIMPLE and SIMPLEC algorithms is that new velocities and corresponding fluxes do not satisfy the momentum balance after the pressure-correction equation is solved. As a result, the calculation must be repeated until the balance is satisfied. To improve the efficiency of this calculation, the PISO algorithm performs two additional corrections: neighbor correction and skewness correction.

Neighbor Correction

The main idea of the PISO algorithm is to move the repeated calculations required by SIMPLE and SIMPLEC inside the solution stage of the pressure-correction equation. After one or more additional PISO loops, the corrected velocities satisfy the continuity and momentum equations more closely. This iterative process is called a momentum correction or "neighbor correction". The PISO algorithm takes a little more CPU time per solver iteration, but it can dramatically decrease the number of iterations required for convergence, especially for transient problems.

Skewness Correction

For meshes with some degree of skewness, the approximate relationship between the correction of mass flux at the cell face and the difference of the pressure corrections at the adjacent cells is very rough. Since the components of the pressure-correction gradient along the cell faces are not known in advance, an iterative process similar to the PISO neighbor correction described above is desirable. After the initial solution of the pressure-correction equation, the pressure-correction gradient is recalculated and used to update the mass flux corrections. This process, which is referred to as "skewness correction", significantly reduces convergence difficulties associated with highly distorted meshes. The PISO skewness correction allows **ANSYS FLUENT** to obtain a solution on a highly skewed mesh in approximately the same number of iterations as required for a more orthogonal mesh.

Skewness - Neighbor Coupling

For meshes with a high degree of skewness, the simultaneous coupling of the neighbor and skewness corrections at the same pressure correction equation source may cause divergence or a lack of robustness. An alternate, although more expensive, method for handling the neighbor and skewness corrections inside the PISO algorithm is to apply one or more iterations of skewness correction for each separate iteration of neighbor correction. For each iteration of the classical PISO algorithm, this technique allows a more accurate adjustment of the face mass flux correction according to the normal pressure correction gradient.

Fractional-Step Method (FSM)

In the FSM, the momentum equations are decoupled from the continuity equation using a mathematical technique called operator-splitting or approximate factorization. The resulting solution algorithm is similar to the segregated solution algorithms described earlier. The formalism used in the approximate factorization allows you to control the order of splitting error. Because of this, the FSM is adopted in **ANSYS FLUENT** as a velocity-coupling scheme in a non-iterative time-advancement (NITA) algorithm (Section 5.4.5).

Coupled Algorithm

As previously mentioned, the pressure-based solver allows you to solve your flow problem in either a segregated or coupled manner. Using the coupled approach offers some advantages over the non-coupled or segregated approach. The coupled scheme obtains a robust and efficient single-phase implementation for steady-state flows, with superior performance compared to the segregated solution schemes. This pressure-based coupled algorithm offers an alternative to the density-based and pressure-based segregated algorithm with SIMPLE-type pressure-velocity coupling. For transient flows, using the coupled algorithm is necessary when the quality of the mesh is poor, or if large time steps are used.

The pressure-based segregated algorithm solves the momentum equation and pressure correction equations separately. This semi-implicit solution method results in slow convergence.

The coupled algorithm solves the momentum and pressure-based continuity equations together. The full implicit coupling is achieved through an implicit discretization of pressure gradient

terms in the momentum equations, and an implicit discretization of the face mass flux, including the Rhie-Chow pressure dissipation terms.

In the momentum equations (5.4-3), the pressure gradient for component k is of the form

$$\sum_f p_f A_k = -\sum_j a^{u_k p} p_j \qquad (5.4\text{-}15)$$

Where $a^{u_k p}$ is the coefficient derived from the Gauss divergence theorem and coefficients of the pressure interpolation schemes (Equation 5.4-4). Finally, for any ith cell, the discretized form of the momentum equation for component u_k is defined as

$$\sum_j {a_{ij}}^{u_k u_k} u_{kj} + \sum_j {a_{ij}}^{u_k p} p_j = {b_i}^{u_k} \qquad (5.4\text{-}16)$$

In the continuity equation, Equation 5.4-5, the balance of fluxes is replaced using the flux expression in Equation 5.4-6, resulting in the discretized form

$$\sum_k \sum_j {a_{ij}}^{p u_k} u_{kj} + \sum_j {a_{ij}}^{pp} p_j = {b_i}^{p} \qquad (5.4\text{-}17)$$

As a result, the overall system of equations (5.4-16 and 5.4-17), after being transformed to the δ-form, is presented as

$$\sum_j [A]_{ij} \vec{X}_j = \vec{B}_i \qquad (5.4\text{-}18)$$

where the influence of a cell i on a cell j has the form

$$A_{ij} = \begin{bmatrix} a_{ij}^{pp} & a_{ij}^{pu} & a_{ij}^{pv} & a_{ij}^{pw} \\ a_{ij}^{up} & a_{ij}^{uu} & a_{ij}^{uv} & a_{ij}^{uw} \\ a_{ij}^{vp} & a_{ij}^{vu} & a_{ij}^{vv} & a_{ij}^{vw} \\ a_{ij}^{wp} & a_{ij}^{wu} & a_{ij}^{wv} & a_{ij}^{ww} \end{bmatrix} \qquad (5.4\text{-}19)$$

and the unknown and residual vectors have the form

$$\vec{X}_j = \begin{bmatrix} p_i' \\ u_i' \\ v_i' \\ w_i' \end{bmatrix} \quad (5.4\text{-}20)$$

$$\vec{B}_i = \begin{bmatrix} -r_i^p \\ -r_i^u \\ -r_i^v \\ -r_i^w \end{bmatrix} \quad (5.4\text{-}21)$$

Note that Equation 5.4-18 is solved using the coupled AMG, which is detailed in Section 5.6.3.

Limitations

The pressure-based coupled algorithm is not compatible with

- The non-iterative time advancement solver (NITA)
- Periodic mass-flow boundary conditions
- The fixed velocity option