Computation of Incompressible Flows: SIMPLE and related Algorithms

Milovan Perić
CoMeT Continuum Mechanics Technologies GmbH
milovan@continuummechanicstechnologies.de
• Consider momentum equations, discretized using an implicit method; the algebraic equation at node P is:

\[ A_P^{u_i} u_{i,P}^{n+1} + \sum_l A_{l}^{u_i} u_{i,l}^{n+1} = Q_{u_i}^{n+1} - \left( \frac{\delta p^{n+1}}{\delta x_i} \right)_P \]

• The iterative solution method includes inner and outer iterations.

• Equations are solved one after another (sequential method).

• At the end of time step, all implicit terms are at the new time level...
Momentum equation solved first in $m$th outer iteration is:

$$A^u_{Pi} u^m_{i,P} + \sum_l A^u_{li} u^m_{i,l} = Q^{m-1}_{ui} - \left( \frac{\delta p^m}{\delta x_i} \right)^P$$

Velocities $u^m_{i}$ do not satisfy continuity equation – need to be corrected; pressure also needs to be updated:

$$u^{m*}_{i} = u^m_{i} + u' \quad \text{and} \quad p^{m*} = p^{m-1} + p'$$

Momentum equation for corrected variables:

$$A^u_{Pi} u^{m**}_{i,P} + \sum_l A^u_{li} u^{m**}_{i,l} = Q^{m-1}_{ui} - \left( \frac{\delta p^{m*}}{\delta x_i} \right)^P$$

Subtract the first equation:

$$u'_{i,P} = -\frac{1}{A^u_{Pi}} \left( \frac{\delta p'}{\delta x_i} \right)^P$$

Here an approximation is made: these velocities are not updated!
Now enforce continuity equation for $u_i^{m**}$ (incompressible flow):

$$\frac{\delta (\rho u_i^{m*})}{\delta x_i} + \frac{\delta (\rho u'_i)}{\delta x_i} = 0$$

This leads to the pressure-correction equation:

$$\frac{\delta}{\delta x_i} \left[ \frac{\rho}{A_P} \left( \frac{\delta p'}{\delta x_i} \right) \right]_P = \left[ \frac{\delta (\rho u_i^{m*})}{\delta x_i} \right]_P$$

Solve pressure-correction equation and correct velocities – now they satisfy the continuity equation.

Momentum equation will not be satisfied by $u_i^{m**}$ if all terms are updated, due to the introduced approximation.

Unless further corrections are performed, pressure correction needs to be under-relaxed:

$$p_i^{m*} = p_i^{m-1} + \alpha_p p'_i$$

Optimum relation: $\alpha_p = 1 - \alpha_u$
SIMPLEC-Algorithm – I

• In SIMPLE, one sets \( u^m = u^{m**} \) and \( p^m = p^{m*} \) and proceeds to the next outer iteration.

• Instead of neglecting the velocity correction at neighbor nodes, one can approximate its effect by assuming:

\[
\frac{u'_{i,P}}{A^u_i} \approx \frac{\sum_l A^u_i u'_{i,l}}{\sum_l A^u_i} \Rightarrow \sum_l \frac{A^u_i u'_{i,l}}{A^u_i} \approx u'_{i,P} \sum_l \frac{A^u_i}{A^u_i}
\]

• When \( u'^{m**}_i \) is introduced on both sides of the momentum equation for corrected velocity and pressure, we obtain:

\[
u'_{i,P} = -\frac{1}{A^u_i} \left( \frac{\delta p'}{\delta x_i} \right)_P - \frac{\sum_l \frac{A^u_i u'_{i,l}}{A^u_i}}{A^u_i}
\]

• By using the above approximation, we obtain a simpler relation:

\[
u'_{i,P} = -\frac{1}{A^u_i + \sum_l A^u_i} \left( \frac{\delta p'}{\delta x_i} \right)_P
\]

Enforce continuity and obtain pressure-correction equation; no under-relaxation for \( p' \).
In PISO, correction process from SIMPLE is continued:

\[ A_P^{ui} u_{i,P}^{m***} + \sum_l A_l^{ui} u_{i,l}^{m**} = Q_{u_i}^{m-1} - \left( \frac{\delta p^{m**}}{\delta x_i} \right)_P \]

Subtract equation for \( u_{i}^{m**} \) to obtain:

\[ u_{i,P}'' = \tilde{u}_{i,P}' - \frac{1}{A_P^{ui}} \left( \frac{\delta p''}{\delta x_i} \right)_P, \quad \tilde{u}_{i,P}' = -\sum_l A_l^{ui} u_{i,l}' \]

Enforce continuity on \( u_{i}^{m***} \) to obtain the second pressure-correction equation:

\[ \frac{\delta}{\delta x_i} \left[ \frac{\rho}{A_P^{ui}} \left( \frac{\delta p''}{\delta x_i} \right) \right]_P = \left[ \frac{\delta (\rho \tilde{u}_i')}{\delta x_i} \right]_P \]

The right-hand side can be computed because \( u' \) is available, but one needs the coefficient matrix from momentum equation (which is usually overwritten by the matrix for \( p' \)).
PISO-Algorithm – II

- The correction process can be continued by adding one more * and ‘; often 3-5 correctors are performed...
- No under-relaxation for pressure-correction is needed.
- PISO is seldom used for steady-state problems, but is often used for transient problems:
  - Momentum equations are solved only once, with mass fluxes and all deferred corrections based on solution from previous time step;
  - Several pressure corrections are applied, and linearized momentum equations are also explicitly updated.
  - If the non-linearity in momentum equations is not updated, PISO is not accurate enough; if it is updated, SIMPLE is more efficient...
- Another method was proposed by Patankar (seldom used): use SIMPLE only to correct velocities and enforce continuity...
SIMPLER-Algorithm

• Pressure is obtained by requiring that the corrected velocities satisfy the momentum equation without simplification:

\[ A_P^{u_i} u_i^{m} + \sum_l A_l^{u_i} u_{i,l}^{m} = Q_{u_i}^{m-1} - \left( \frac{\delta p^m}{\delta x_i} \right)_P \]

• The relation between corrected velocity and unknown pressure is:

\[ u_i^{m} = \frac{Q_{u_i}^{m-1} - \sum_l A_l^{u_i} u_{i,l}^{m}}{A_P^{u_i}} - \frac{1}{A_P^{u_i}} \left( \frac{\delta p^m}{\delta x_i} \right)_P = \hat{u}_i^{m} = \frac{1}{A_P^{u_i}} \left( \frac{\delta p^m}{\delta x_i} \right)_P \]

• By enforcing continuity again, pressure equation is obtained:

\[ \frac{\delta}{\delta x_i} \left[ \frac{\rho}{A_P^{u_i}} \left( \frac{\delta p^m}{\delta x_i} \right) \right]_P = \left[ \frac{\delta (\rho \hat{u}_i^{m})}{\delta x_i} \right]_P \]

The r.h.s. can be computed since \( u' \) is known from SIMPLE-step.
Boundary Conditions for Pressure-Correction

- When boundary velocity is specified, its correction is zero; this is equivalent to specified zero-gradient of pressure-correction (Neumann-condition).

- If velocity is specified at all boundaries (or treated as such within one outer iteration), pressure-correction equation has zero-gradient condition on all boundaries...

- The solution than only exists, if the sum of all source terms in pressure-correction equation is equal to zero!

- In FV-methods this is ensured if mass conservation is ensured by boundary velocities for the solution domain as a whole (mass fluxes at all inner CV-faces cancel out)...

- For incompressible flows, the solution of $p'$-equation is not unique (one can add a constant to it); $p' = 0$ is held at a reference point...
Comparison of SIMPLE and PISO – I

Computation of laminar flow around circular cylinder in a channel: Polyhedral grid in the vicinity of the cylinder. Periodic solution was first obtained using second-order discretization in space and time and SIMPLE-method; it was then continued with both SIMPLE and PISO for another 2 s.
Computed instantaneous pressure and velocity field at one time. Von Karman vortex street is obtained (Reynolds-number is 100).

Cylinder is not at the center of the channel, so the vortex shedding is not symmetric...
Comparison of SIMPLE and PISO – III

Simulation continued using transient SIMPLE and time step 0.005 s (red line), and with PISO using 4 time steps: 0.005 s (blue line), 2 times smaller (0.0025 s; gold line), 4 times smaller (0.00125 s; green line) and 8 times smaller (0.000625 s, black line).

PISO-solutions are converging towards SIMPLE-solution, but with 1st-order... SIMPLE performed 4 outer iterations per time step, PISO 4-5 corrector steps...
SIMPLE-Algorithm for Polyhedral Grids – I

- Second order discretization in space and time is assumed (midpoint rule, linear interpolation, central differences).
- Pressure is treated in a conservative way: pressure forces computed at CV-faces.
- The net pressure force can be expressed via pressure gradient using Gauss-rule:

\[ Q_{i,P}^p = - \int_S p \mathbf{i}_i \cdot \mathbf{n} \, dS = - \int_V \frac{\partial p}{\partial x_i} \, dV \]

\[ \sum_k p_k S_k^i \approx \left( \frac{\partial p}{\partial x_i} \right)_p \Delta V \Rightarrow \left( \frac{\partial p}{\partial x_i} \right)_p = \frac{\sum_k p_k S_k^i}{\Delta V} \]

- At the start of a new outer iteration, momentum equations are solved to obtain \( u_i^{m*} \).
For the discretized continuity equation in a FV-method, one needs to compute mass fluxes through CV-faces.

Simple linear interpolation with a colocated variable arrangement leads to problems (oscillatory solutions).

The usual solution to this problem is the so-called “Rhie-Chow” correction, which is added to interpolated velocity:

\[
v_{n,k}^* = (v_{n,k}^* - (\Delta V)_k \left( \frac{1}{A_P} \right)_k \left[ \left( \frac{\delta p^m - 1}{\delta n} \right)_k - \left( \frac{\delta p^m - 1}{\delta n} \right)_k \right]
\]

\[
v_{n,k} = (\mathbf{v} \cdot \mathbf{n})_k \quad – \text{velocity component normal to CV-face.}
\]

The term in brackets represents the difference between pressure derivative computed at the face and the average of pressure derivatives computed at CV-centroids.

It is proportional to the third derivative of pressure multiplied by mesh spacing squared.
SIMPLE-Algorithm for Polyhedral Grids – III

• Averaged gradient at CV-face can be computed by averaging pressure gradients computed at CV-centroids:
  \[
  \left( \frac{\delta p^m - 1}{\delta n} \right)_k = \overline{\nabla p}_k \cdot \mathbf{n}_k
  \]

• The derivative at CV-face is computed as in diffusion fluxes...

• The mass fluxes are computed using interpolated velocities:
  \[
  \dot{m}^*_m = \int_{S_k} \rho v \cdot \mathbf{n} \, dS \approx \left( \rho v^*_n S \right)_k
  \]

• These fluxes do not satisfy continuity equation – there is an imbalance:
  \[
  \sum_k \dot{m}^*_m = \Delta \dot{m}_p
  \]

• The mass fluxes need to be corrected, as explained earlier...
Velocity correction at CV-face is proportional to the gradient of pressure-correction (SIMPLE-approximation):

\[ (v'_n)_k \approx - (\rho \Delta V S)_k \left( \frac{1}{A^v_n} \right)_k \left( \frac{\delta p'}{\delta n} \right)_k \]

Since pressure-correction tends to zero as outer iterations converge, additional approximations are possible – neglecting non-orthogonality:

\[ \left( \frac{\partial \phi}{\partial n} \right)_k \approx \frac{\phi_{N_k} - \phi_P}{|r_{N'_k} - r_{P'}|} + \left[ \frac{(\nabla \phi)_{N_k} \cdot (r_{N'_k} - r_{N_k}) - (\nabla \phi)_P \cdot (r_{P'} - r_P)}{|r_{N'_k} - r_{P'|}} \right]^{\text{old}} \]

\[ \left( \frac{\delta p'}{\delta n} \right)_k \approx \frac{p'_{N_k} - p'_P}{(r_{N_k} - r_P) \cdot n} \]

The rest is neglected, but could be taken into account in another corrector...
Now enforce continuity equation (incompressible flow):

\[ \sum_k m_k^m + \sum_k m_k'^{} = 0 \quad \text{with} \quad m_k' \approx (\rho v_n' S)_k \]

The result is a pressure-correction equation:

\[ A_P^' p_P^' + \sum_l A_l^' p_l^' = -\Delta m_P \]

Solve pressure-correction equation to obtain \( p' \).

Correct velocities (mass fluxes) at CV-faces; they now satisfy continuity equation.

Correct also velocities at CV-centroids...

Correct pressure by adding only a fraction of \( p' \) (for steady-state flows, 0.1 to 0.3; for transient flows, 0.3 to 0.9, depending on time-step size and grid non-orthogonality).
SIMPLE-Algorithm for Polyhedral Grids – VI

- Specified velocity at boundary => zero-gradient in pressure-correction...
- Specified pressure at boundary => velocity extrapolated and corrected...

\[
\begin{align*}
\nu_{n,k}^m &= (\nu_{n,k}^m)_k - (\Delta V)_k \left( \frac{1}{A_P^{v_n}} \right)_k \left[ \left( \frac{\delta p_n^{m-1}}{\delta n} \right)_k - \left( \frac{\delta p_n^{m-1}}{\delta n} \right)_k \right] \\

\end{align*}
\]

Computed at boundary

Cell-center value

Extrapolated to cell face

\[
\begin{align*}
(v'_n)_k &\approx - (\rho \Delta V \cdot S)_k \left( \frac{1}{A_P^{v_n}} \right)_k \left( \frac{\delta p'}{\delta n} \right)_k \\

\left( \frac{\delta p'}{\delta n} \right)_k &\approx \frac{p'_{N_k} - p'_P}{(r_{N_k} - r_P) \cdot \mathbf{n}}
\end{align*}
\]

\( N_k \) – boundary node
Pressure-Based Methods for Compressible Flows – I

• Many methods are designed specifically for compressible flow.
• These methods are usually not efficient when Ma ~ 0…
• Special methods (e.g. pre-conditioning) are used to make methods work also for weakly compressible flows…
• SIMPLE-type methods were originally developed for incompressible flows…
• They can be extended to compressible flows – and they work well (used in most commercial and public codes)…
• One needs to solve also the energy equation…
• ... and use equation of state to obtain density once pressure and temperature are updated.
Pressure-Based Methods for Compressible Flows – II

- The energy equation in terms of enthalpy:

\[
\frac{\partial}{\partial t} \int_V \rho h \, dV + \int_S \rho h v \cdot n \, dS = \int_S \lambda \nabla T \cdot n \, dS + \int_V (v \cdot \nabla p + S : \nabla v) \, dV + \frac{\partial}{\partial t} \int_V p \, dV
\]

Viscous part of stress tensor, \( S = \tau_{ij} i_i i_j \)

- The viscous stresses have now an additional contribution:

\[
\tau_{ij} = \mu \left( \frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} \right) - \frac{2}{3} \mu \delta_{ij} \nabla \cdot v
\]

- For an ideal gas, \( h = c_p T \) and the equation of state is: \( \rho = \frac{p}{RT} \)

- The continuity equation now has the unsteady term and mass flux is non-linear, since both density and velocity are variable:

\[
\frac{\partial}{\partial t} \int_V \rho \, dV + \int_S \rho v \cdot n \, dS = 0
\]
Pressure-Based Methods for Compressible Flows – III

- The first step is the solution of momentum equations – as for incompressible flows.

- The major changes are in the discretized continuity equation (for the sake of simplicity, implicit-Euler method assumed):

  \[
  \frac{\Delta V}{\Delta t} (\rho^{m-1} - \rho^n)_P + \sum_k \dot{m}_k^{m*} = \Delta \dot{m}_P
  \]

- Mass fluxes have to be corrected to satisfy continuity; for this, both velocity and density need to be corrected. Velocity correction is as before:

  \[
  (v'_n)_k \approx - (\rho \Delta V S)_k \left( \frac{1}{A_P^n} \right)_k \left( \frac{\delta p'}{\delta n} \right)_k
  \]

- Density correction follows from equation of state:

  \[
  \rho'_k \approx \left( \frac{\partial \rho}{\partial p} \right)_T p'_k = \frac{p'_k}{RT_k}
  \]
The corrected mass flux can be expressed as:

$$(m^* + m')_k = (\rho^{m-1} + \rho')_k (v^*_n + v'_n)_k S_k .$$

One can neglect the product of density and velocity correction (as it tends to zero faster than other terms):

$$m'_k \approx (\rho^{m-1} v'_n S)_k + (\rho' v^*_n S)_k$$

The corrected continuity equation now reads:

$$\frac{\Delta V}{\Delta t} \rho'_p + \sum_k m'_k + \Delta m_p = 0$$

By substituting expressions for velocity and density correction, one again obtains a pressure-correction equation...

... which differs significantly from one for incompressible flow.
Pressure-Based Methods for Compressible Flows – V

- The pressure-correction equation is no longer of Poisson-type; it now resembles a transport equation with convective and diffusive fluxes...

- Diffusive part comes from velocity correction in face mass flux (proportional to pressure gradient)...

- Convective part and time derivative come from density correction (directly proportional to pressure at face or $P$)...

- Pressure now has to be specified on part of the boundary, and as initial condition...

- The ratio of convective to diffusive contribution is proportional to $Ma^2$, so the method adapts automatically to the type of flow and works at both limits ($Ma = 0$ and $Ma$ very high).
Pressure-Based Methods for Compressible Flows – VI

- When Mach-number is high, density correction plays the dominant role – diffusive part is negligible (like solving continuity equation for density)...

- When Mach-number is very low, density correction becomes small and the method behaves as for incompressible flow.

- This method also works for acoustics applications (propagation of pressure waves at low Mach-number)...

- When shocks are present, interpolation of density to CV-face may need special attention (linear interpolation may lead to oscillations)...

- Local blending of upwind-approximations, gradient-limiter, or special TVD, ENO or WENO-schemes are used...
Non-uniform and uniform grids (from 8 x 8 CV to 256 x 256 CV)
Linear interpolation and central-difference differentiation with midpoint-rule integrals; SIMPLE-algorithm, FV-method
Estimation of discretization errors for the strength of primary and secondary vortex by Richardson-extrapolation: 2nd-order convergence for both quantities and both grids – but errors much smaller on non-uniform grid.
Non-uniform grids (from 10 x 10 CV to 160 x 160 CV).

Upwind, Central, and cubic interpolation with midpoint-rule integrals; cubic interpolation with Simpson-rule integrals.

Centerline velocity profiles: high-order interpolation causes oscillations on coarse grids, but much higher accuracy when grids are fine enough...
Effects of under-relaxation factors on convergence of outer iterations for the steady-state lid-driven cavity flow at $Re = 1000$, $32 \times 32$ CV uniform grid: staggered variable arrangement (left) and colocated variable arrangement (right)
Effects of grid fineness and under-relaxation factors on convergence of outer iterations for the steady-state lid-driven cavity flow at Re = 1000, 32 x 32 CV and 64 x 64 CV uniform grid, staggered and colocated variable arrangement.

The fine the grid, the more important it is to use optimal under-relaxation.
Drag coefficients for the 2D flow around a cylinder in a channel as functions of grid size: steady flow at Re = 20 (left) and the maximum drag coefficient in a periodic unsteady flow at Re = 100 (right). Also shown are extrapolated values using Richardson-extrapolation.
Variation of drag and lift in a periodic unsteady flow at $Re = 100$, computed on three grids. The difference between solutions on consecutive grids reduces by a factor 4, as expected of a 2nd-order discretization method.
Stagnation-Point Flow

Linear interpolation between two cell-centers provides value at $k'$-location:

$$\phi_{k'} = \frac{|r_k - r_P|}{|r_{N_k} - r_P|} \phi_{N_k} + \frac{|r_{N_k} - r_k|}{|r_{N_k} - r_P|} \phi_P$$

Using

$$\phi_k \approx \phi_{k'}$$

Using

$$\phi_k \approx \phi_{k'} + \left( \nabla \phi \right)_{k'} \cdot (r_k - r_{k'})$$