A CFD METHOD FOR AXIAL THRUST LOAD PREDICTION OF CENTRIFUGAL COMPRESSORS

Zhen-Xue Han
Paul G. A. Cizmas
Department of Aerospace Engineering
Texas A&M University
College Station, Texas 77843-3141

Abstract

This paper presents the development of a numerical algorithm for the computation of axial thrust load on a centrifugal compressor. An unstructured flow solver has been developed for the computation of a hybrid, structured and unstructured grid. The computational domain of the impeller has been discretized using a structured mesh, while the computational domain on the back side of the wheel has been discretized using an unstructured mesh. The two grids are merged and a median dual-mesh is generated. The Navier-Stokes equations are discretized using a finite volume method. Roe’s flux-difference scheme is used for inviscid fluxes and directional derivatives along edges are used for viscous fluxes. The gradients at the mesh vertices are calculated using the Least-squares method. An explicit scheme is used
for time integration. Convergence is accelerated using a local time-step and implicit residual smoothing. The results of the numerical simulation include the axial thrust load of the centrifugal compressor. In addition, details of the leakage flow are presented.

**Nomenclature**

- $A$: Jacobian matrix
- $c$: Speed of sound
- $c_p$: Specific heat at constant pressure
- $E$: Source term
- $e$: Total internal energy per unit volume
- $\vec{F}$: Flux vector
- $\vec{G}$: Non-convective part of the flux vector
- $H$: Total internal enthalpy per unit mass
- $\hat{i}$: Unit vector in the $x$-direction
- $\hat{j}$: Unit vector in the $y$-direction
- $\hat{k}$: Unit vector in the $z$-direction
- $k$: Kinetic energy of turbulent fluctuations per unit mass
- $k(i)$: Set of vertices adjacent to node $i$
- $\hat{n}$: Outward unit normal
- $P$: Turbulent production
- $p$: Pressure
- $Pr$: Prandtl number
- $Q$: State vector
- $q$: Velocity module
- $\vec{q}$: Heat flux vector
- $R$: Residual or Riemann invariant
- $\vec{r}$: Spatial position vector
- $Re$: Reynolds number
$S$ Sutherland constant or surface area

$T$ Temperature

$U$ Normal relative velocity

$u$ Fluid velocity component in the $x$-direction

$u_\tau$ Friction velocity, $u_\tau = \sqrt{\tau_{wall}/\rho}$, where $\tau_{wall}$ is the viscous stress at wall

$\vec{u}$ Fluid velocity

$V$ Volume

$v$ Fluid velocity component in the $y$-direction

$\vec{v}$ Grid velocity

$V_g$ Normal grid velocity

$w$ Fluid velocity component in the $z$-direction

$y^+$ Non-dimensional number, $y^+ = \rho u_\tau y_1/\mu$, where $y_1$ is the height of the element adjacent to the wall

$\alpha$ parameter of $k - \omega$ turbulence model or limiter function

$\beta$ parameter of $k - \omega$ turbulence model

$\vec{\Gamma}$ Flux vector of turbulence model

$\gamma$ Ratio of specific heats

$\Delta$ Difference

$\delta_{ij}$ Dirac delta function

$\lambda$ Thermal conductivity

$\Phi$ State vector of turbulence model

$\varphi$ Generic variable

$\mu$ Viscosity

$\nabla$ Gradient

$\Pi$ Source term of turbulence model

$\rho$ Density

$\tau$ Viscous stress
σ  parameter of $k - \omega$ turbulence model

$\vec{\Omega}$  Angular velocity of the frame of reference

$\omega$  weighting factor or specific dissipation rate

Subscripts

$B$  Boundary

$g$  Grid

$i$  Node

$ij$  Edge

$inv$  Inviscid

$L$  Left

$n$  Surface normal direction

$R$  Right

$vis$  Viscous

$x$  Component in the $x$-coordinate

$y$  Component in the $y$-coordinate

$z$  Component in the $z$-coordinate

Superscripts

$*$  Total (or stagnation)

$m$  Iteration level

$n$  Time advancing level
INTRODUCTION

Accurate prediction of the flow field in axial- and centrifugal-flow compressors and turbines is necessary to properly estimate the axial thrust load. Several types of methods are available for predicting the axial thrust: empirical, analytical and numerical methods. Empirical formulae have been proposed for single-stage and multi-stage compressors with open, semi-open and closed impellers [12, 13, 19].

Analytical methods for predicting the axial thrust load have been developed based on the flow analysis in the gap between rotating and stationary walls. Two types of gap flows are analyzed: the axial gap flow between a rotating disk and a stator, such as that at the back of an impeller shroud, and the annular gap flow, such as that in an annular seal. This analytical method has been utilized to predict the axial thrust behavior in a rocket engine turbopump [11]. In another investigation, the transient axial thrust in a rocket engine turbopump has been predicted using a finite volume procedure that models the system using boundary nodes, internal nodes and branches [20]. Pressures, temperatures and species concentrations are computed at internal nodes by solving time-dependent mass, energy and species conservation equations.

The axial thrust of a rocket turbopump has been recently predicted based on a computational fluid dynamics (CFD) calculation [16]. The flow has been modeled using the Navier-Stokes equations with the Baldwin-Lomax turbulence model. A multi-block structured grid has been utilized to discretize the computational domain. In spite of a very good general agreement between the numerically simulated and experimentally measured pressure values on the impeller, the axial thrust prediction was off by a significant amount [16]. This difference was attributed to the incorrectly calculated pressure distribution in the back-face
seal cavity.

The approach used here to predict the axial load consists of numerically simulating the flow on the bladed and unbladed sides of the wheel. The flow simulation is completed by solving the Navier-Stokes equations with a $k - \omega$ turbulence model. To accurately calculate the axial load it is crucial that the flow on the bladed-side (or front-side) of the wheel be coupled to the flow on the back-side of the wheel.

The computational domain on the two sides of the wheel is reduced by taking into account flow periodicity caused by the wheel blades. A multi-block structured grid is used to discretize the computational domain on the front-side of the wheel. An unstructured grid is used to discretize the computational domain on the back-side of the wheel. The unstructured grid is utilized in order to properly discretize the somewhat irregular shape of the back-side of the wheel. The two grids are merged into an unstructured grid and the flow is computed using an unstructured mixed-grid Navier-Stokes solver. The difference of the pressure integral on the two sides of the wheel provides the axial thrust.

The next section briefly presents the governing equations of the flow model. The numerical method utilized to solve the governing equations is then presented. Results obtained for a typical compressor wheel are presented subsequently.

**PHYSICAL MODEL**

The flow in a centrifugal compressor is modeled using the integral form of the Navier-Stokes equations. The Navier-Stokes equations are written in terms of the absolute flow variables over a moving control volume that is rotating with a Cartesian reference frame. For turbomachinery flow simulations, using absolute flow variables gives better results than using relative flow variables, because the absolute far-field flow is uniform \[9, 4\]. Under
these conditions, the Navier-Stokes equations can be written in the following form

\[
\frac{\partial}{\partial t} \int_V Q dV + \oint_S \vec{F} \cdot \hat{n} dS = \int_V E dV, \tag{1}
\]

where \( t \) denotes time, \( V \) is the volume of an arbitrary control volume, \( \hat{n} \) is the outward unit normal vector of the surface area \( S \), \( Q \) is the state vector, \( \vec{F} \) is the flux vector and \( E \) is the source term. The state vector \( Q \) of conservative variables per unit volume is:

\[
Q = [\rho \ \rho u \ \rho v \ \rho w \ e]^T. \tag{2}
\]

Here \( e \) is the total internal energy per unit volume. The flux vector \( \vec{F} \) can be written as

\[
\vec{F} = Q(\vec{u} - \vec{v}) + \vec{G}, \tag{3}
\]

where \( \vec{u} \) is the fluid velocity and \( \vec{v} \) is the grid velocity. Consequently, the first term of the right-hand-side is the convection of \( Q \) relative to the surface element. The \( \vec{G} \) term represents the non-convective part of the flux. If \( \vec{\Omega} \) denotes the angular velocity of the frame of reference,
then \( \vec{v} = \vec{\Omega} \times \vec{r} \). The right hand side term \( E \) in equation (1) is the source term

\[
E = \begin{pmatrix}
0 \\
(\rho \vec{u} \times \vec{\Omega}) \cdot \hat{i} \\
(\rho \vec{u} \times \vec{\Omega}) \cdot \hat{j} \\
(\rho \vec{u} \times \vec{\Omega}) \cdot \hat{k} \\
0
\end{pmatrix}.
\]

(4)

The flux \( \vec{F} \) is divided into an inviscid part \( \vec{F}_{inv} \) and a viscous part \( \vec{F}_{vis} \)

\[
\vec{F} = Q(\vec{u} - \vec{v}) + \vec{G} = \vec{F}_{inv} - \vec{F}_{vis}.
\]

(5)

The component of \( \vec{F}_{inv} \) normal to the surface element is

\[
F_{inv} = \hat{n} \cdot \vec{F}_{inv} = \begin{pmatrix}
\rho U \\
\rho U u + p n_x \\
\rho U v + p n_y \\
\rho U w + p n_z \\
\rho U H + p V_g
\end{pmatrix}.
\]

(6)
In equation (6), \( H \) is the total enthalpy per unit mass \( H = (e + p)/\rho \), \( V_g \) is the velocity normal to the grid \( V_g = \vec{v} \cdot \hat{n} \) and \( U \) is the relative velocity normal to the grid \( U = (\vec{u} - \vec{v}) \cdot \hat{n} \).

The fluid is considered to be a calorically perfect gas.

The component of \( \vec{F}_{\text{vis}} \) normal to the surface element is

\[
F_{\text{vis}} = \hat{n} \cdot \vec{F}_{\text{vis}} = \begin{bmatrix}
0 \\
f_x \\
f_y \\
f_z \\
u f_x + v f_y + w f_z - q_n \\
\end{bmatrix} = \begin{bmatrix}
0 \\
\tau_{xx} n_x + \tau_{xy} n_y + \tau_{xz} n_z \\
\tau_{xy} n_x + \tau_{yy} n_y + \tau_{yz} n_z \\
\tau_{xz} n_x + \tau_{yz} n_y + \tau_{zz} n_z \\
u f_x + v f_y + w f_z - q_n \\
\end{bmatrix},
\]

(7)

where \( \tau_{xx}, \tau_{xy}, \tau_{xz}, \ldots, \tau_{zz} \) are viscous stresses. With the assumption of Newtonian fluid and Stokes’ hypothesis, the viscous stresses can be written as \( \tau_{ij} = -p \delta_{ij} + \mu (v_{i,j} + v_{j,i} - \).
2/3δ_{ij}v_{k,k}, (i, j, k = 1, 2, 3) using Einstein’s summation convention [18]. The viscosity $\mu$ is calculated using Sutherland formula $\mu/\mu_0 = (T/T_0)^{3/2} (T_0 + S)/(T + S)$ where $S$ is the Sutherland constant that is a characteristic of the gas. For air within temperature range from 170K to 1900K, the constant values are: $T_0 = 273K$, $S = 111.0K$, $\mu_0 = 1.716 \times 10^{-5} N \cdot s/m^2$.

The heat flux normal to the surface element is $q_n = \vec{q} \cdot \hat{n}$. Using Fourier’s law, the heat flux vector is $\vec{q} = -\lambda \nabla T$ where $\nabla T$ is the temperature gradient and $\lambda$ is the thermal conductivity. Using the definition of Prandtl number, $\lambda$ can be expressed as $\lambda = (\mu c_p)/Pr$.

**Turbulence Model**

The turbulence model used in this paper is a linear $k-\omega$ model developed by Wilcox [22]. This turbulence model also includes a transition model. The two equations that must be solved to determine the turbulent viscosity can be written in integral form as

$$\frac{\partial}{\partial t} \int_V \Phi dV + \oint_S [\vec{\Gamma}_{inv} - \vec{\Gamma}_{vis}] \cdot \hat{n} dS = \int_V \Pi dV$$ (8)
with

\[
\Phi = \begin{pmatrix}
\rho k \\
\rho \omega
\end{pmatrix}
\]

\[
\vec{\Gamma}_{inv} = \begin{pmatrix}
\rho k (\vec{u} - \vec{v}) \\
\rho \omega (\vec{u} - \vec{v})
\end{pmatrix}
\]

\[
\vec{\Gamma}_{vis} = \begin{pmatrix}
(\mu + \sigma^* \mu_T) \nabla k \\
(\mu + \sigma \mu_T) \nabla \omega
\end{pmatrix}
\]

\[
\Pi = \begin{pmatrix}
P - \beta^* \rho \omega k \\
\alpha P \omega / k - \beta \rho \omega^2
\end{pmatrix}
\]

\[
P = \mu_T \left\{ \left( \frac{\partial u}{\partial y} + \frac{\partial v}{\partial x} \right)^2 + \left( \frac{\partial u}{\partial z} + \frac{\partial w}{\partial x} \right)^2 + \left( \frac{\partial v}{\partial z} + \frac{\partial w}{\partial y} \right)^2 + 
\right.
\]

\[
+ 2 \left[ \left( \frac{\partial u}{\partial x} \right)^2 + \left( \frac{\partial v}{\partial y} \right)^2 + \left( \frac{\partial w}{\partial z} \right)^2 \right] \right\}
\]

Here $k$ is the kinetic energy of turbulent fluctuations per unit mass, $\omega$ is the specific dissipation rate, $\vec{\Gamma}_{inv}$ and $\vec{\Gamma}_{vis}$ are the turbulent inviscid flux and viscous flux, respectively. $P$ is the turbulent production, and $\mu_T$ is the turbulent viscosity given as $\mu_T = \alpha^* \rho k / \omega$. Turbulent heat conductivity, $\lambda_T$, is given by the relation $\lambda_T = \mu_T c_p / Pr_T$, where $Pr_T$ is the turbulent Prandtl number. To take into account turbulence, the values of viscosity $\mu$ and
heat conductivity $\lambda$ are replaced by $\mu + \mu_T$ and $\lambda + \lambda_T$, respectively. The values of the model parameters used in this paper are given in [22]. Note that the turbulence model equations (8) have the same form as the Navier-Stokes equations (1). This similarity will be used in the development of the numerical algorithm for solving the $k - \omega$ equations.

**NUMERICAL METHOD**

The governing equations are solved using a finite volume method. The computational domain is discretized using an unstructured mixed mesh. Time-marching is used to calculate the steady solution. This section presents the spatial discretization of the governing equations, the numerical implementation of the vector fluxes, the high order upwind scheme and the implementation of the boundary conditions.

**Spatial Discretization**

The computational domain is divided into a set of non-overlapping cells. These cells may be hexahedra, prisms and pyramids or an arbitrary combination of them. The cell-averaged variables are stored at the nodes of the grid, which are the vertices of the cells. The governing equations are discretized using mesh duals as control volumes. Figure 1 shows a subset of tetrahedra adjacent to node $i$. Let us consider the collection of all cells incident to vertex $i$. The dual-mesh at vertex $i$ is formed by connecting the cell centroids, face centroids and edge midpoints, and looping through the faces of all cells incident to this vertex $i$ and collecting the sub-tetrahedra associated to the vertex. This results in a polyhedron hull centered at the grid node. This type of dual mesh is called the median-dual [1].

The median dual-mesh is adopted in this paper because of its flexibility to handle unstructured mixed meshes. The volume of a dual mesh $i$ is associated with the grid node
i and can be obtained by adding the volumes of all sub-tetrahedra adjacent to this node. The surfaces of a dual mesh are associated with edges that are the connections of two cell centroids, \emph{i.e.}, two grid nodes. The area vector of a surface element associated with an edge is obtained by adding the area vectors of all sub-triangle surfaces that are sharing the midpoint of the edge during sub-tetrahedra collection procedure.

Figure 2 shows the face associated with dual mesh edge \((i, j)\) connecting nodes \(i\) and \(j\). The normal and area of the dual-mesh in Figure 2 can be found as \[21\]

\[
\vec{S}_{ij} = \sum_{k=\bigcup(i,j)} \frac{(\vec{r}_a \times \vec{r}_b)_k}{2} 
\]

\[
S_{ij} = \sqrt{(\vec{S}_{ij} \cdot \hat{i})^2 + (\vec{S}_{ij} \cdot \hat{j})^2 + (\vec{S}_{ij} \cdot \hat{k})^2}
\]

\[
\hat{n}_{ij} = \frac{\vec{S}_{ij}}{S_{ij}},
\]

where \(\bigcup(i,j)\) denotes the collection of the faces of the sub-tetrahedra surrounding edge \((i, j)\).

An edge-based data structure is utilized for the discretization of the governing equations (1) and (8) due to its flexibility and efficiency in handling different element types. The numerical scheme, however, is a node-based style because the solution is obtained at the nodes of the mesh. The surface integral of the inviscid and viscous fluxes is approximated by a sum over the faces of each control volume. The source terms are calculated using the control volume-averaged solution and the derivatives of flow variables at cell centroid.

If \(Q_i\) denotes the volume-averaged flow variable \(Q\) over the volume \(V_i\), equation (1) can be rewritten in a semi-discrete form as

\[
\frac{\partial Q_i}{\partial t} V_i = - \int_{S_i} \vec{F} \cdot \hat{n} dS + E_i V_i.
\]
The surface integral term on right hand side of this equation can be approximated as

$$\oint_S \mathbf{F} \cdot \mathbf{n} dS = \sum_{j=k(i)} (F_{\text{inv}} - F_{\text{vis}})_{ij} \cdot S_{ij},$$  \hspace{1cm} (14)$$

where \(k(i)\) is the set of vertices adjacent to node \(i\), \((F_{\text{inv}} - F_{\text{vis}})_{ij}\) is the flux normal to the dual-mesh cell face and \(S_{ij}\) is the cell face surface area. Both the surface area and the flux are associated with the dual mesh face that is associated with the edge \((i, j)\).

The turbulence model equations (8) have the same form as the Navier-Stokes equations (1). Consequently, the spatial discretization of the turbulence model equations is similar to the discretization of the Navier-Stokes equations, shown in equations (13) and (14).

**Vector Fluxes Implementation**

Different approaches are used in this paper to calculate the viscous and inviscid fluxes. To evaluate the viscous flux in equation (14), the velocity and temperature derivatives are required at edge midpoints. For this edge-based data structure, the gradients of a generic variable \(\varphi\) at edge midpoints could be simply computed by averaging the nodal values

$$\overline{(\nabla \varphi)_{ij}} = \frac{1}{2} \left( (\nabla \varphi)_i + (\nabla \varphi)_j \right).$$  \hspace{1cm} (15)$$

The disadvantage of this approach is that decoupling occurs, particularly for hexahedra and prismatic cells. It is possible to prevent decoupling by taking into account the directional derivatives along an edge \([3]\)

$$\nabla \varphi_{ij} = \overline{(\nabla \varphi)_{ij}} - \left[ \overline{(\nabla \varphi)_{ij} \cdot \frac{\Delta \mathbf{r}}{|\Delta \mathbf{r}|}} \cdot \frac{\varphi_j - \varphi_i}{|\Delta \mathbf{r}|} \right] \frac{\Delta \mathbf{r}}{|\Delta \mathbf{r}|}.$$  \hspace{1cm} (16)$$
The same approach as outlined above is also employed for the discretization of the viscous term of the turbulence model equations (8). The algorithm utilized to calculate the gradients at the grid node is presented in a following section.

The inviscid flux in equation (14) is calculated using Roe’s flux-difference splitting scheme [17]. The interface flux for the finite volume formulation is calculated in each of the three coordinate directions as the solution of a locally one-dimensional Riemann problem normal to the cell interface [15]

$$\frac{\partial Q}{\partial t} + \frac{\partial F}{\partial n} = 0 \quad (17)$$

$$Q = Q_L \quad n < 0$$

$$Q = Q_R \quad n > 0,$$

where $Q_L$ and $Q_R$ are the left and right fluid states and $n$ is the coordinate normal to the cell interface. Roe’s procedure consists of constructing the solution to the approximate linearized problem $\frac{\partial Q}{\partial t} + A\frac{\partial Q}{\partial n} = 0$, where the Jacobian matrix $A = \frac{\partial F}{\partial Q}$ is evaluated at a Roe-averaged state such that the Hugoniot-Rankine shock jump condition $F_R - F_L = A(Q_R - Q_L)$ is satisfied.

The interface inviscid flux can then be written as

$$(F_{\text{inv}})_{ij} = \frac{1}{2} \left[ F(Q_R) + F(Q_L) - |A|(Q_R - Q_L) \right], \quad (18)$$

where the absolute value of the Roe-averaged matrix $|A|$ indicates that the matrix is evaluated
using the absolute values of the eigenvalues. Note that the eigenmodes of \( \overline{\lambda} \) are \( \lambda \)

\[
\begin{align*}
\lambda_1 &= \lambda_2 = \lambda_3 = \lambda_{\overline{U}} = \overline{U} \\
\lambda_4 &= \lambda_{\overline{U}+\overline{\tau}} = \overline{U} + \overline{\tau} \\
\lambda_5 &= \lambda_{\overline{U}-\overline{\tau}} = \overline{U} - \overline{\tau}
\end{align*}
\]

(19)

and the Roe-averaged primitive variables are given by the formulae

\[
\begin{align*}
\overline{\rho} &= \sqrt{\rho_R \rho_L} \\
\overline{u} &= (u_L \sqrt{\rho_L} + u_R \sqrt{\rho_R})/(\sqrt{\rho_L} + \sqrt{\rho_R}) \\
\overline{v} &= (v_L \sqrt{\rho_L} + v_R \sqrt{\rho_R})/(\sqrt{\rho_L} + \sqrt{\rho_R}) \\
\overline{w} &= (w_L \sqrt{\rho_L} + w_R \sqrt{\rho_R})/(\sqrt{\rho_L} + \sqrt{\rho_R}) \\
\overline{H} &= (H_L \sqrt{\rho_L} + H_R \sqrt{\rho_R})/(\sqrt{\rho_L} + \sqrt{\rho_R}) \\
\overline{U} &= \overline{u} n_x + \overline{v} n_y + \overline{w} n_z - V_g.
\end{align*}
\]

(20)

It is computationally more efficient to write the flux \( F \) as \([15], [21]:\)

\[
(F_{inv})_{ij} = \frac{1}{2} \left[ F(Q_R) + F(Q_L) - |\Delta F_{\overline{U}}| - |\Delta F_{\overline{U}+\overline{\tau}}| - |\Delta F_{\overline{U}-\overline{\tau}}| \right],
\]

(21)
where

\[ |\Delta F_{\text{U}}| = |\lambda_{\text{U}}| \left( \Delta \rho - \frac{\Delta \rho}{c^2} \right) \begin{bmatrix} 1 \\ \bar{u} \\ \bar{v} \\ \bar{w} \\ \bar{q}^2/2 \end{bmatrix} + \begin{bmatrix} 0 \\ \Delta u - \Delta \bar{U} n_x \\ \Delta v - \Delta \bar{U} n_y \\ \Delta w - \Delta \bar{U} n_z \end{bmatrix} + \bar{\rho} \begin{bmatrix} \pi \Delta u + \bar{\pi} \Delta v + \bar{\pi} \Delta w - \bar{U} \Delta \bar{U} \end{bmatrix} \]
\[ |\Delta F_{\bar{U}^{-}}| = |\lambda_{\bar{U}^{-}}| \left( \frac{\Delta p - \overline{pc} \Delta U}{2c^2} \right) \begin{bmatrix} 1 \\ \bar{u} - \overline{cn}_x \\ \bar{v} - \overline{cn}_x \\ \bar{w} - \overline{cn}_x \\ \overline{H} - \overline{cU} \end{bmatrix} \] (23)

\[ |\Delta F_{\bar{U}^{+}}| = |\lambda_{\bar{U}^{+}}| \left( \frac{\Delta p + \overline{pc} \Delta U}{2c^2} \right) \begin{bmatrix} 1 \\ \bar{u} + \overline{cn}_x \\ \bar{v} + \overline{cn}_x \\ \bar{w} + \overline{cn}_x \\ \overline{H} + \overline{cU} \end{bmatrix}. \] (24)

The operator \( \Delta() = ()_R - ()_L \) defines the jump over the discontinuity surface and \( \overline{q}^2 \) is calculated from \( \overline{q}^2 = \overline{p}^2 + \overline{u}^2 + \overline{w}^2 \).

In this paper, the turbulence model equations (8) are solved separately from the Navier-Stokes equations (1). The inviscid, or better said convective, fluxes in equation (8) are given
in the same manner as above and have the following expression

\[
(\Gamma_{inv})_{ij} = \frac{1}{2} [\Gamma(Q_R, \Phi_R) + F(Q_L, \Phi_L) - \\
-|\Delta \Gamma_{\bar{\tau}}| - |\Delta \Gamma_{\bar{\tau}+\tau}| - |\Delta \Gamma_{\bar{\tau}-\tau}|],
\]

where

\[
|\Delta \Gamma_{\bar{\tau}}| = |\lambda_{\bar{\tau}}| \left\{ \begin{array}{l}
\lambda
\end{array} \right\}
\]

\[
\left( \Delta \rho - \frac{\Delta p}{c^2} \right) \left[ \begin{array}{c}
\frac{k}{\rho}
\end{array} \right] + \frac{\rho}{\Delta \omega} \left[ \begin{array}{c}
\Delta k
\Delta \omega
\end{array} \right],
\]

(26)

\[
|\Delta \Gamma_{\bar{\tau}+\tau}| = |\lambda_{\bar{\tau}+\tau}| \left\{ \begin{array}{l}
\lambda
\end{array} \right\}
\]

\[
\left( \Delta \rho + \frac{\rho p \Delta U}{2c^2} \right) \left[ \begin{array}{c}
\frac{k}{\rho}
\end{array} \right] \left[ \begin{array}{c}
\frac{1}{\omega}
\end{array} \right],
\]

(27)

\[
|\Delta \Gamma_{\bar{\tau}-\tau}| = |\lambda_{\bar{\tau}-\tau}| \left\{ \begin{array}{l}
\lambda
\end{array} \right\}
\]

\[
\left( \Delta \rho - \frac{\rho p \Delta U}{2c^2} \right) \left[ \begin{array}{c}
\frac{k}{\rho}
\end{array} \right] \left[ \begin{array}{c}
\frac{1}{\omega}
\end{array} \right],
\]

(28)

**High-order Upwind Scheme**

Assuming that the flow variables are piecewise constant in each cell, the numerical method outlined above is first-order accurate. With the assumption of piecewise constant variables, the left and right fluid states used in the equations (18) to (25) are \(Q_L = Q_i\) and \(Q_R = Q_j\). Here \(Q_i\) and \(Q_j\) are cell-averaged fluid states associated with cells \(V_i\) and \(V_j\), respectively.

Assuming that the flow variables have a piecewise linear distribution over all cells, the numerical method becomes second-order accurate in space. The piecewise linear function in each cell is reconstructed using the cell-averaged values of \(Q\). The left and right fluid states
are determined using linear reconstruction. To produce a monotone solution, a limiter is applied to the linearly reconstructed fluid states.

As shown in [2], a piecewise linear redistribution of the cell-averaged flow variables can be represented by

\[
Q(x, y, z) = Q(x_0, y_0, z_0) + \nabla Q_0 \cdot \Delta \vec{r};
\]  

(29)

where \(\Delta \vec{r}\) is the vector from point 0 to any point \((x, y, z)\) in the cell, and \(\nabla Q_0\) represents the solution gradient in the cell. Note that this equation is simply the first-order accuracy Taylor approximation plus a higher-order correction.

Using the piecewise linear redistribution (29), the left and right fluid states \(Q_R\) and \(Q_L\) are found to be:

\[
Q_L = Q_i + \frac{1}{2} \nabla Q_i \cdot \Delta \vec{r}
\]

(30)

\[
Q_R = Q_j - \frac{1}{2} \nabla Q_j \cdot \Delta \vec{r},
\]

where \(\Delta \vec{r} = \vec{r}_j - \vec{r}_i\), \(\nabla Q_i\) and \(\nabla Q_j\) are gradients of \(Q\) at the end nodes \(i\) and \(j\) of an edge \((i, j)\), respectively. The methodology of computing the gradients of \(Q\) at nodes is presented in the following section.

To ensure that the linearly reconstructed fluid states produce a monotone solution, a limiter function is introduced into equation (30) [5]

\[
Q_L = Q_i + \frac{1}{2} \alpha [(1 - \eta)\nabla Q_i \cdot \Delta \vec{r}, \eta(Q_j - Q_i)]
\]

(31)

\[
Q_R = Q_j - \frac{1}{2} \alpha [(1 - \eta)\nabla Q_j \cdot \Delta \vec{r}, \eta(Q_j - Q_i)]
\]
where $\eta = \frac{1}{3}$ and $\alpha[\cdot, \cdot]$ is the limiter function

$$\alpha[a, b] = [1 + \text{sign}(1, ab)][(a^2 + \epsilon)b^2 + (b^2 + \epsilon)a^2]$$

$$2(a^2 + b^2 + 2\epsilon).$$

(32)

$\epsilon$ is a very small number to prevent division by zero in smooth regions of the flow. The limiter may reduce the solution accuracy in regions of large gradients, in order to avoid the generation of new extrema.

The convective flux of the turbulence equations (8) is approximated by a first-order upwind discretization, which produces a robust and efficient numerical scheme.

**Vertex Gradients Calculation**

Several techniques are available for calculating solution gradients at vertices of the mesh. Two simple and easy to implement approaches are Green-Gauss and Least-squares methods. The Least-squares method is more complex than Green-Gauss method, but it possesses desirable properties for general unstructured meshes [1]. The Least-squares method is used in this paper.

**Time Integration**

The semi-discrete form (13) of the Navier-Stokes equations (1) and of the turbulence model equations (8) can be written for a grid node $i$ as

$$\frac{\partial Q_i}{\partial t} V_i = R_i,$$

(33)

with

$$R_i = - \oint_{S_i} \vec{F} \cdot \hat{n} dS + E_i V_i,$$

(34)
where $R_i$ is the residual.

To obtain a steady-state solution, this equation must be integrated in time. In this paper, the turbulence model equations are uncoupled from the Navier-Stokes equations. The Navier-Stokes equations are solved first, assuming the turbulent viscosity $\mu_T$ locally constant and equal to the value from the previous time step. Subsequently, the turbulence model equations are solved, $k$ and $\omega$ are advanced in time, and $\mu_T$ is updated. Both sets of equations are integrated in time using the same explicit, four-stage Runge-Kutta scheme

\[
\begin{align*}
Q_i^{(0)} &= Q_i^n \\
Q_i^{(1)} &= Q_i^{(0)} + \alpha_1 \frac{\Delta t_i}{V_i} R(Q_i^{(0)}) \\
Q_i^{(2)} &= Q_i^{(0)} + \alpha_2 \frac{\Delta t_i}{V_i} R(Q_i^{(1)}) \\
Q_i^{(3)} &= Q_i^{(0)} + \alpha_3 \frac{\Delta t_i}{V_i} R(Q_i^{(2)}) \\
Q_i^{(4)} &= Q_i^{(0)} + \alpha_4 \frac{\Delta t_i}{V_i} R(Q_i^{(3)}) \\
Q_i^{n+1} &= Q_i^{(4)},
\end{align*}
\]

where $\Delta t_i$ is the time step at node $i$, superscript $n$ is time-stepping level, $\alpha_1$, $\alpha_2$, $\alpha_3$ and $\alpha_4$
are stage coefficients

\[ \alpha_1 = 0.1668 \]
\[ \alpha_2 = 0.3028 \]
\[ \alpha_3 = 0.5276 \]
\[ \alpha_4 = 1.0. \]

In this scheme, the inviscid operator is evaluated at each stage and, for computational efficiency, the dissipative operator is evaluated only on the first stage and third stage.

For steady-state computations, convergence is accelerated using a local time-step and implicit residual smoothing [14]. The time-step calculation is based strictly on inviscid stability considerations for each node [6]

\[ \Delta t_i \leq CFL \frac{V_i}{\sum_{j=k(i)} \frac{1}{2} (|U|+c)_{k(i)} S_{ij}} }, \]

(36)

where \( U \) is the normal relative velocity, \( c \) is the speed of sound, and \( CFL \) is the Courant number.

The time step can be further increased by residual smoothing, which increases the support of the scheme [10]. Residual smoothing is essentially a Laplacian filtering of the numerical values of the residuals. The following equation is solved for the new value of the residual:

\[ \bar{R}_i = R_i + \epsilon \nabla^2 \bar{R}_i, \]

(37)

where \( \bar{R}_i \) is the Laplacian filtered value of \( R_i \). The undivided Laplacian, \( \nabla^2 \bar{R}_i \), can be
represented on an unstructured mesh as:

\[ \nabla^2 \mathbf{R}_i = \sum_{j=k(i)} (\mathbf{R}_j - \mathbf{R}_i). \]  \hspace{1cm} (38)

The resulting implicit equation for \( \mathbf{R}_i \) is solved by Jacobian iteration:

\[ \mathbf{R}_i^{(m)} = \mathbf{R}_i + \epsilon \sum_{j=k(i)} \mathbf{R}_j^{(m-1)} \frac{1}{1 + \epsilon \sum_{j=k(i)} 1}. \]  \hspace{1cm} (39)

Typically, two Jacobian iterations are performed with \( \epsilon = 0.5 \).

**Boundary Conditions**

Boundary conditions are applied as conditions on the flux at boundary surfaces as opposed to being applied directly to state variables. This approach, called “weak condition” [21], avoids singularities at mesh points located at corners. To close the dual-meshes at boundary points a face-based, as opposed to an edge-based, data structure is utilized. The boundary face-based data structure and the adjacent edge-based data structure are used to evaluate the fluxes and gradients of the flow field. The boundary surface elements are discretized in the same manner as the elements of interior grid. The vector normal to the boundary dual-mesh face is found in the same manner as the normal to a dual mesh face in the interior of the grid.

The tangency flow condition is imposed by requiring that the flux through the wall is zero, so that \( \mathbf{u} \cdot \hat{n} = 0 \), where \( \hat{n} \) is the normal to the boundary dual mesh. The flux normal
to the boundary face is

\[
F_{B_i} = \begin{pmatrix}
0 \\
p_i \cdot \hat{n}_{B_i} \cdot \hat{i} \\
p_i \cdot \hat{n}_{B_i} \cdot \hat{j} \\
p_i \cdot \hat{n}_{B_i} \cdot \hat{k} \\
-pV_g
\end{pmatrix}.
\]

The expression of the normal flux is valid for both viscous and inviscid flow. In the viscous flow case, the no-slip boundary condition is applied after each time step, such that \( \vec{u} = \vec{v} \) for moving walls and \( \vec{u} = 0 \) for stationary walls.

The inflow/outflow boundary flux is defined in terms of the fluid state at the boundary node \( i \) and is specified as \([21]\):

\[
F_{B_i} = \begin{cases}
-F(Q_{-\infty}) \cdot \hat{n}_{B_i} & \text{if } \vec{U}_i \cdot \hat{n}_{B_i} < -c_i \\
F(Q_i, Q_{-\infty}) \cdot \hat{n}_{B_i} & \text{if } \vec{U}_i \cdot \hat{n}_{B_i} \in [-c_i, c_i] \\
F(Q_i) \cdot \hat{n}_{B_i} & \text{if } \vec{U}_i \cdot \hat{n}_{B_i} > c_i.
\end{cases}
\]

The flux normal to the boundary face imposed for supersonic inflow (\( i.e., \vec{U}_i \cdot \hat{n}_{B_i} < -c_i \)) is computed using the state vector at upstream infinity. The flux normal to the boundary face imposed for subsonic inflow and outflow (\( i.e., \vec{U}_i \cdot \hat{n}_{B_i} \in [-c_i, c_i] \)) is computed using
an intermediate state calculated from the conditions at cell $i$ and upstream infinity. The intermediate state is calculated using Riemann invariants. The flux imposed for supersonic outflow (i.e., $\vec{U}_i \cdot \hat{n}_{B_i} > c_i$) is computed using the state vector at cell $i$.

The exit boundary conditions for the leakage flow are specified similarly to the exit boundary conditions for the impeller flow. A constant static pressure is specified at the inner diameter of the back plate, at the exit of the leakage flow. The pressure distribution in the radial direction is obtained by integrating the radial equilibrium equation.

At periodic boundaries, the scalar state variables and velocity vectors, eventually rotated, of each periodic node pair are set equal after each time stepping. The turbulent flow variables, $k$ and $\omega$, are specified at the inlet boundary and extrapolated at the outlet boundary. The value of $k$ and the gradient of $\omega$ are set to zero at walls.

**NUMERICAL RESULTS**

This section presents the results of the flow simulation necessary to calculate the axial thrust in a centrifugal compressor. To calculate the axial thrust on the compressor wheel, the computational domain includes not only the impeller but also the back side of the wheel. The simulation of the flow between the back side of the wheel and the backplate is necessary to calculate the leakage flow. Because of the periodicity, the flow is calculated in only one passage of the impeller and the corresponding back side of the wheel. A detail of the impeller is shown in Fig. 3 while the entire computational domain is shown in Fig. 4.

The impeller has 12 blades, an exterior diameter of 94 mm, a tip clearance of 0.75 mm and an angular velocity of 100,000 rpm. The inlet flow is axial. The inlet temperature is 20 deg Celsius and the inlet total pressure is 101,325 Pa.
Computational Grid

The computational grid of the impeller, shown in Fig. 5, reveals that this is a multi-block, structured mesh. An $H$-grid block discretizes the computational domain upstream of the blade, a $C$-grid is utilized around the airfoil, an $H$-grid discretizes the blade wake region and an $O$-grid discretizes the blade tip region. The computational domain on the back side of the wheel is discretized by an unstructured grid. The unstructured grid has been used because of irregular shape of the computational domain on the back side of the wheel. An advancing-front/local-reconnection unstructured grid generation algorithm is used to generate a two-dimensional triangle-quadrilateral mixed mesh in the meridional plane of the leakage flow channel, as shown in Fig. 6. The two-dimensional grid is then stacked in the tangential direction and generates a three-dimensional prism-hexahedron mixed mesh for the leakage flow channel. The structured mesh of the impeller and the unstructured mesh of the leakage flow are merged into an unstructured grid.

Accuracy of Numerical Results

The numerical algorithm has been validated against experimental data and the results of these comparisons are presented in [7]. Since experimental data are not yet available for this compressor, to validate the accuracy of numerical results it is necessary to show that the results are independent of the grid that discretizes the computational domain. Four meshes have been used to assess that the results are grid independent. The coarsest grid has approximately 183k nodes in the impeller mesh and $3,100 \times 30$ nodes in the leakage flow mesh. The number of meridional planes has been kept constant and equal to 30 for all four grids. The $y^+$ number for the coarsest grid is less than 3.5. The number of nodes of the finer meshes is shown in Table 1.
To verify that the solution is grid independent, the mass flow rates of the leakage flow and the axial thrust loads of the impeller are compared using the four meshes. The mass flow rate of the leakage flow and the axial thrust have been chosen as indicators of the grid convergence because these two variables depend on both the flow through the impeller and the flow on the back side of the wheel. The variation of the leakage mass flow rate as a function of the grid size is shown in Fig. 7. The mass flow rate is non-dimensionalized by the mass flow rate corresponding to the coarsest grid.

The axial thrust of the impeller is calculated by integrating the pressure variation on both sides of the wheel. The variation of the axial thrust as a function of the grid size is shown in Fig. 7. Both the leakage flow and the axial thrust indicate that the grid must exceed 600,000 nodes in order to obtain grid independent results.

The computation has been done on an Linux workstation with an Alpha 21164 processor. The computational time required for this simulation is approximately $1.2 \cdot 10^{-4}$ sec/iteration/grid point. Convergence is reached in approximately 2,000 to 5,000 iterations, depending on the flow conditions.

**Leakage Flow**

One goal of this paper is to calculate the axial thrust load on a centrifugal compressor. This task requires not only the calculation of the flow through the impeller but also the calculation of flow on the back side of the wheel. At a pressure ratio, $p_{\text{exit static}}/p_{\text{inlet total}}$, of 2.22 and an exit static pressure of the leakage flow of 101,325 Pa, the numerical simulation predicts a total-to-total pressure ratio of 3.44 and a total-to-total efficiency of the impeller of 78%. The predicted mass flow rate is 0.49 kg/s, the leakage flow is 2.8% of the total flow and the axial thrust load is 249 N.
The rest of this section presents the flow features of the back side of the wheel. Figure 8 shows the pressure on the back surface of the wheel. Note that although the geometry of the leakage channel is symmetric, the flow field is not axisymmetric, due to the blade wakes.

The relative Mach number distribution is shown in a meridional plane that intersects the leakage flow channel at two different circumferential locations. For the first circumferential location, the meridional plane divides equally the channel between two adjacent blades. Figure 9 shows the Mach number distribution and Fig. 11 shows the velocity vectors. For the second circumferential location, the meridional plane intersects the blade trailing edge. Figure 10 shows the Mach number distribution in the meridional surface behind the blade trailing edge. Velocity vectors behind the impeller blade at the inlet in the back flow are shown in Fig. 12. Note that at this circumferential location, the leakage mass flow returns into the impeller. The numerical results show that 22% of the leakage flow returns into the impeller flow. Because of the reversal of the leakage flow, the impeller flow and the back-plate flow must be solved coupled to avoid the boundary condition problems at the junction between the impeller and the leakage flow.

CONCLUSIONS

The development of a numerical algorithm for computation of axial thrust load on a centrifugal compressor has been presented. To calculate the axial thrust load, both the flow through the impeller and the leakage flow on the back side of the wheel have been simulated. An unstructured flow solver has been developed for the computation of a hybrid, structured and unstructured grid. The computational domain of the impeller has been discretized using a structured mesh, while the computational domain on the back side of the wheel has been discretized using an unstructured mesh. The two grids have been merged and the Navier-
Stokes equations have been discretized using a median dual-mesh. Eddy viscosity has been calculated using a linear $k - \omega$ turbulence model that also includes a transition model. The discretization of the governing equations has been done using a finite volume method. Roe’s flux-difference scheme has been used for inviscid fluxes. Directional derivatives along edges have been used for viscous fluxes. The gradients at the mesh vertices have been calculated using the Least-squares method. An explicit scheme has been used for time integration. Convergence has been accelerated using a local time-step and implicit residual smoothing.

The CFD method proposed herein for axial thrust prediction has two advantages compared to the previous method [16]: an unstructured grid that better discretizes the leakage flow region and a better turbulence model.

Since experimental results were not yet available for this compressor, the numerical simulation has been validated by verifying that the solution is grid independent. The results of the numerical simulation include the predicted axial thrust load and the details of the leakage flow. The variation of the leakage mass flow rate as a function of the circumferential location has been predicted. The numerical results have also shown that for certain circumferential locations, part of the leakage flow returns into the impeller flow. This requires that the impeller flow and the back-plate flow be solved coupled, to avoid boundary conditions problems at the junction between the impeller and the leakage flow.

Further investigations are necessary to capture the variation of axial thrust load and leakage flow with the operating point and the boundary conditions for the leakage flow. An investigation of the influence of turbulence modeling in the back-plate region on the axial thrust is also necessary.
ACKNOWLEDGMENTS

This work has been funded by the Turbomachinery Research Consortium. The authors would like to thank Mr. Gerald LaRue of Honeywell for his support during the project.

References


Table 1: Computational grid size.

<table>
<thead>
<tr>
<th>Grid</th>
<th>Impeller Grid</th>
<th>Leakage Grid</th>
<th>Total</th>
<th>$y^+$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>183,471</td>
<td>3100 × 30</td>
<td>276,471</td>
<td>3.5</td>
</tr>
<tr>
<td>2</td>
<td>265,471</td>
<td>3890 × 30</td>
<td>382,171</td>
<td>3.5</td>
</tr>
<tr>
<td>3</td>
<td>396,483</td>
<td>4450 × 30</td>
<td>529,983</td>
<td>2.5</td>
</tr>
<tr>
<td>4</td>
<td>501,583</td>
<td>5950 × 30</td>
<td>680,083</td>
<td>2.5</td>
</tr>
</tbody>
</table>
Figure 1: Subset of tetrahedra adjacent to node \( i \).
Figure 2: Dual-mesh surface element associated with edge $ij$. 
Figure 3: Detail of the impeller.
Figure 4: Computational grid.
Figure 5: Computational grid of the impeller.
Figure 6: Meridional section through the grid on the back side of the wheel.
Figure 7: Variation of calculated axial thrust and leakage flow with grid size.
Figure 8: Pressure contours on back side of wheel.
Figure 9: Mach number in meridional plane (mid-passage).

Figure 10: Mach number in meridional plane (blade plane).

Figure 11: Detail of leakage flow (mid-passage).

Figure 12: Detail of leakage flow (blade plane).
Captions sheet

Table 1. Computational grid size.

Figure 1. Subset of tetrahedra adjacent to node $i$.

Figure 2. Dual-mesh surface element associated with edge $ij$.

Figure 3. Detail of the impeller.

Figure 4. Computational grid.

Figure 5. Computational grid of the impeller.

Figure 6. Meridional section through the grid on the back side of the wheel.

Figure 7. Variation of calculated axial thrust and leakage flow with grid size.

Figure 8. Pressure contours on back side of wheel.

Figure 9. Mach number in meridional plane (mid-passage).

Figure 10. Mach number in meridional plane (blade plane).

Figure 11. Detail of leakage flow (mid-passage).

Figure 12. Detail of leakage flow (blade plane).