Canadian Aeronautics and Space Institute
11^{th} Aerodynamics Symposium
Toronto, 26-27 April 2005

An Unstructured Newton-Krylov Solver for the Compressible Navier-Stokes Equations

P. Wong
Graduate Student
peterson@oddjob.utias.utoronto.ca

D.W. Zingg
Professor
http://goldfinger.utias.utoronto.ca/∼dwz/

University of Toronto Institute for Aerospace Studies
4925 Dufferin Street, Toronto, Ontario
Canada, M3H 5T6

Abstract

We present our recent progress using the Newton-Krylov method to solve three-dimensional aerodynamic flows. An unstructured-grid approach is utilized to handle complex geometries that arise in practical industrial designs. The Spalart and Allmaras turbulence model is used to calculate turbulent eddy viscosities. We employ Newton's method to obtain the steady-state flow solution for a potentially fast convergence rate, with the linear system that arises solved by a preconditioned Krylov method. Issues with regard to preconditioning of the viscous operator in three dimensions are addressed. An incomplete factorization preconditioner applied to a matrix involving only the nearest neighboring terms is chosen based on numerical experiments. The capability of the current solver is demonstrated by numerical studies over the ONERA M6 wing as well as the DLR-F6 wing-body configuration.

1 Introduction

After many years of development, computational fluid dynamics (CFD) has become an important tool in aerodynamic analysis [1, 2]. It provides a reliable alternative to wind-tunnel and flight tests at a lower cost. The current technology is capable of performing Navier-Stokes calculations over complete three-dimensional aircraft configurations. These calculations require the use of a fine grid to capture the geometry, which leads to intensive memory usage as well as high computational times. Two drag-prediction workshops have been organized to assess the capabilities of current solvers when applied to such flows [3, 4, 5]. With parallelization, flow solutions can be obtained within a day using a grid up to three million nodes. However, it is inevident that grid convergence is achieved. Moreover, code-to-code variations are observed. It is believed that accurate drag prediction may require futher improvement in grid density and quality. As a result, research continues to develop more efficient and accurate algorithms to apply to these engineering geometries for routine industrial use.

The Newton-Krylov method is an efficient method to solve the Navier-Stokes equations [6]. This method has the potential for rapid convergence. Venkatakrishnan and Mavriplis [7] developed an unstructured approximate-Newton algorithm. The linear system is solved by a preconditioned generalized minimum residual (GMRES) method. Different preconditioners as well as orderings of the unknowns were studied. The algorithm was found to be competitive with a multigrid algorithm. Barth and Linton [8] presented a Newton solver on unstructured meshes. Matrix-free GMRES is used to solve the linear system. The work is extended to parallel computations as well as three-dimensional applications. Nielsen et al. [9] developed an unstructured Newton-Krylov algorithm for the Euler equations in two and three dimensions. The method is found to provide a fast asymptotic convergence rate. Mesh sequencing is found to be an effective startup strategy for the method. Anderson et al. [10] compared the performance of a Newton-Krylov method with a multigrid algorithm. Fast convergence was obtained using an inexact linear solve. They concluded that the performance of GMRES can greatly depend on the choice of parameters.

Blanco and Zingg [11] performed a study comparing quasi-Newton, standard Newton, and matrix-free Newton methods. They developed a fast solver on triangular grids using a matrix-free inexact-Newton approach together with an approximate-Newton startup strategy. Pueyo and Zingg [12] performed a parametric study of an inexact preconditioned matrix-free Newton-Krylov algorithm. Their optimized algorithm is found to converge faster and more reliably than an approximate Newton algorithm and an approximately-factored multigrid algorithm. Geuzaine et al. [13, 14] studied mesh sequencing as well as multigrid preconditioning with the Newton-Krylov method. Nemec and Zingg [15] applied the Newton-Krylov method to numerical optimization. The same approach is applied to solve the flow equations as well as the adjoint equations to calculate the objective function gradients. Their work is extended to multiblock structured grids using the Spalart-Allmaras (S-A) turbulence model. Chisholm and Zingg [16, 17] developed a strategy which provides effective and efficient startup with the Newton-Krylov algorithm. Their work is also extended to multi-block structured grids using the S-A turbulence model. Manzano et al. [18] applied the Newton-Krylov algorithm to three-dimensional inviscid flows using unstructured grids.

The purpose of this work is to extend the algorithm of Manzano et al. to turbulent flows using the S-A turbulence model on hybrid unstructured grids. The goal is to develop an efficient and robust algorithm for three-dimensional aerodynamic flows. Different aspects of the algorithm are studied and discussed in the paper, including preconditioning and startup strategy. The performance of the algorithm is demonstrated over a wing as well as a wing-body configuration.

2 Governing Equations

The governing equations are the Navier-Stokes equations. These equations describe the conservation of mass, momentum and total energy for a viscous compressible flow. For an arbitrary control volume Ω, the integral form of the equations can be written as:

$$\frac{\partial}{\partial t}\int_{\Omega} Q dV + \int_{\partial\Omega} \mathbf{F}\cdot\hat{\mathbf{n}} dS = \int_{\partial\Omega} \mathbf{G}\cdot\hat{\mathbf{n}} dS \quad (1)$$

with Q the set of conservative flow variables (density ρ, momentum components ρu, ρv, ρw, and total energy ρE). $\mathbf{F}$ is the inviscid flux tensor, and $\mathbf{G}$ is the flux tensor associated with viscosity and heat conduction. These quantities can be written as:

$$Q = \left[\begin{array}{ccccc} \rho & \rho u & \rho v & \rho w & \rho E \end{array}\right]^T \quad (2)$$

$$\begin{aligned}\mathbf{F} &= \begin{bmatrix} \rho u \\ \rho u^2 + p \\ \rho uv \\ \rho uw \\ u(\rho E + p) \end{bmatrix} \hat{\mathbf{i}} + \begin{bmatrix} \rho v \\ \rho uv \\ \rho v^2 + p \\ \rho vw \\ v(\rho E + p) \end{bmatrix} \hat{\mathbf{j}} \\ &+ \begin{bmatrix} \rho w \\ \rho uw \\ \rho vw \\ \rho w^2 + p \\ w(\rho E + p) \end{bmatrix} \hat{\mathbf{k}} \end{aligned} \quad (3)$$

$$\mathbf{G} = \begin{bmatrix} 0 \\ \tau_{xx} \\ \tau_{xy} \\ \tau_{xz} \\ f \end{bmatrix} \hat{\mathbf{i}} + \begin{bmatrix} 0 \\ \tau_{yx} \\ \tau_{yy} \\ \tau_{yz} \\ g \end{bmatrix} \hat{\mathbf{j}} + \begin{bmatrix} 0 \\ \tau_{zx} \\ \tau_{zy} \\ \tau_{zz} \\ h \end{bmatrix} \hat{\mathbf{k}}$$

$$\begin{aligned} f &= u\tau_{xx} + v\tau_{xy} + w\tau_{xz} - q_x \\ g &= u\tau_{yx} + v\tau_{yy} + w\tau_{yz} - q_y \\ h &= u\tau_{zx} + v\tau_{zy} + w\tau_{zz} - q_z \end{aligned} \quad (4)$$

For a Newtonian fluid in local thermodynamic equilibrium, Stokes relation is valid. The viscous stress tensor $\boldsymbol{\tau}$ can be related to the dynamic viscosity μ and the strain rate tensor using:

$$\begin{aligned} \boldsymbol{\tau} &= \mu \begin{bmatrix} 2u_x & u_y + v_x & u_z + w_x \\ v_x + u_y & 2v_y & v_z + w_y \\ w_x + u_z & w_y + v_z & 2w_z \end{bmatrix} \\ &- \frac{2}{3}\mu\left(u_x + v_y + w_z\right)\mathbf{I} \end{aligned} \quad (5)$$

where $\mathbf{I}$ is the unit tensor, and u_x denotes $\partial u/\partial x$ and so forth. The heat flux vector is given by Fourier's law $\mathbf{q} = -k\nabla T$. The thermal conductivity is related to the dynamic viscosity through the Prandtl number $Pr = c_p\mu/k$. Sutherland's law is used to calculate the dynamic viscosity. Assuming the fluid behaves as a thermally and calorically perfect gas, the pressure p can be written in terms of the flow variables to close the system:

$$p = (\gamma - 1)\left[\rho E - \frac{1}{2}\rho\left(u^2 + v^2 + w^2\right)\right] \quad (6)$$

3 Turbulence Modeling

We solve the Reynolds-averaged Navier-Stokes equations for turbulent flows. The Reynolds-stress tensor is modeled using the Boussinesq approximation and introducing an eddy-viscosity term. The turbulent eddy viscosity is modeled with the one-equation Spalart and

Allmaras turbulence model [19]. In differential form the model is written as:

$$\frac{\partial \tilde{\nu}}{\partial t} + (\mathbf{v} \cdot \nabla)\,\tilde{\nu} = c_{b1}(1 - f_{t2})\tilde{S}\tilde{\nu}$$
$$+\frac{1}{\sigma}\left[\nabla \cdot ((\nu + \tilde{\nu})\nabla\tilde{\nu}) \ + \ c_{b2}(\nabla\tilde{\nu})^2\right]$$
$$-\left[c_{w1}f_w - \frac{c_{b1}}{\kappa^2}f_{t2}\right]\left(\frac{\tilde{\nu}}{d}\right)^2$$
$$+f_{t1}\Delta U^2 \tag{7}$$

where $\mathbf{v}$ is the velocity vector. The model is solved in a form fully-coupled with the mean-flow equations. The terms on the right-hand side of the equation are the production term, the diffusion term, the destruction term, and the trip term respectively. The eddy viscosity ν_t is calculated from the working variable $\tilde{\nu}$, using:

$$\nu_t = \tilde{\nu} f_{v1}, \quad f_{v1} = \frac{\chi^3}{\chi^3 + c_{v1}^3}, \quad \chi = \frac{\tilde{\nu}}{\nu} \tag{8}$$

and ν denotes the kinematic viscosity. The vorticity-like term $\tilde{S}$ in the production term is calculated using:

$$\tilde{S} = S + \frac{\tilde{\nu}}{\kappa^2 d^2}f_{v2}, \quad f_{v2} = 1 - \frac{\chi}{1 + \chi f_{v1}} \tag{9}$$

where S = $|\nabla \times \mathbf{v}|$ is the magnitude of the vorticity, d is the distance to the closest wall, and κ is the von Kármán constant. The destruction term governs the dissipation of the eddy viscosity due to blocking effects of the wall. It contains a function f_w that models near-wall effects. The function is calculated by:

$$f_w = g\left(\frac{1 + c_{w3}^6}{g^6 + c_{w3}^6}\right)^{1/6}$$
$$g = r + c_{w2}(r^6 - r), \quad r = \frac{\tilde{\nu}}{\tilde{S}\kappa^2 d^2} \tag{10}$$

The model includes a trip term that models laminar-to-turbulent flow transition. Transition locations are not predicted and are specified by the user. The trip term includes two functions that are given by:

$$f_{t1} = c_{t1}g_t \exp\left[-c_{t2}\frac{\omega_t^2}{\Delta U^2}(d^2 + g_t^2 d_t^2)\right]$$
$$f_{t2} = c_{t3}\exp(-c_{t4}\chi^2)$$
$$g_t = \min\left(0.1, \ \frac{\Delta U}{\omega_t \Delta x}\right) \tag{11}$$

where ΔU is the norm of the velocity difference between a field point and the trip, ω_t is the magnitude of the vorticity at the trip, d_t is the distance to the closest trip, and Δx is the grid spacing at the trip. The flow can be assumed to be fully-turbulent by setting f_{t1} and f_{t2} to zero. This assumes transition occurs at the leading edge. Closure coefficients are given by:

$$c_{b1} = 0.1355, \quad \sigma = 2/3, \quad c_{b2} = 0.622,$$
$$\kappa = 0.41, \quad c_{w1} = c_{b1}/\kappa^2 + (1 + c_{b2})/\sigma,$$
$$c_{w2} = 0.3, \quad c_{w3} = 2, \quad c_{v1} = 7.1,$$
$$c_{t1} = 1, \quad c_{t2} = 2, \quad c_{t3} = 1.2, \quad c_{t4} = 0.5$$

Note that c_{t3} and c_{t4} are updated with values from newer versions of the model [20]. The wall boundary condition is $\tilde{\nu} = 0$. A value of $\nu_\infty/10$ is used as the free-stream condition for $\tilde{\nu}$, where ν_∞ is the kinematic viscosity in the free stream.

Ashford [21] proposed a modification to $\tilde{S}$ in the production term:

$$\tilde{S} = Sf_{v3} + \frac{\tilde{\nu}}{\kappa^2 d^2}f_{v2}, \quad f_{v2} = \left(1 + \frac{\chi}{c_{v2}}\right)^{-3}$$
$$f_{v3} = \frac{(1 + \chi f_{v1})(1 - f_{v2})}{\chi} \tag{12}$$

with c_{v2} = 5. The modification is found to produce better numerical properties [16] and is adopted in the current work.

4 Spatial Discretization

The spatial discretization follows that used by Mavriplis and Venkatakrishnan [22] for hybrid unstructured grids. A cell-vertex approach is utilized with centroidal-median-dual control volumes constructed around source-grid vertices. A finite-volume discretization is obtained by integrating the fluxes over the boundary of the control volume. The value of the flux at each control volume face is computed by averaging the fluxes in the two control volumes on either side of the face:

$$f_{ik} \simeq \frac{1}{2}\left[\mathbf{F}(Q_i) + \mathbf{F}(Q_k)\right] \cdot \vec{\mathbf{n}}_{ik} + D_{ik} \tag{13}$$

where f_{ik} is the inviscid numerical flux on the face ik with neighboring cells i and k, $\vec{\mathbf{n}}_{ik}$ is the area-weighted normal of the face ik, and D_{ik} is the dissipation operator.

Numerical dissipation is added for stability and resolving shocks. A matrix-dissipation scheme is used to discretize the convective flux [23]. It is constructed from the undivided Laplacian and biharmonic operators:

$$D_{ik} = -\frac{1}{2}|A_{ik}|\left[\varepsilon_{ik}^{(2)}(Q_k - Q_i) - \varepsilon_{ik}^{(4)}(L_k - L_i)\right]$$

$$L_i = \sum_k (Q_k - Q_i)$$

where

$$\varepsilon_i^{(2)} = \sum_k \kappa_2 \frac{|p_k - p_i|}{p_k + p_i}$$

and

$$\varepsilon_i^{(4)} = \max\left(0, \kappa_4 - \varepsilon_i^{(2)}\right) \qquad (14)$$

where ε_{ik} is calculated by averaging the values from the two neighboring cells i and k. Two parameters κ_2 and κ_4 control the addition of second- and fourth-difference dissipation. A pressure switch selects the second-difference operator in the presence of shocks, while the fourth-difference operator is used in areas of smooth flow. The Laplacian operator is denoted as L, and $|A|$ is the absolute value of the inviscid flux Jacobian. Small eigenvalues in the Jacobian may occur near stagnation points and sonic points using this approach. This affects convergence and can be avoided by introducing two parameters V_l and V_n [23]. Values of κ_2 = 2, κ_4 = 0.1, V_l = V_n = 0.25 are used in the current work. A centered scheme is utilized for the diffusive-flux term. The convective terms in the turbulence model are discretized using a first-order scheme, as suggested by Spalart and Allmaras [19].

Boundary conditions are enforced by extrapolating the solution to boundary faces and imposing the appropiate boundary conditions. They are handled in a fully-implicit manner in order to obtain fast convergence using Newton's method.

5 Newton-Krylov Algorithm

5.1 Newton iterations

After spatial discretization the steady-state governing equations become a system of nonlinear algebraic equations $\mathcal{R}(\mathcal{Q})$ = 0. We use Newton's method to obtain a solution of these equations. At each Newton iteration, we need to solve a linear system for the solution update.

$$\left(\frac{\partial \mathcal{R}}{\partial \mathcal{Q}}\right)^n \Delta\mathcal{Q}^n = -\mathcal{R}(\mathcal{Q}^n)$$

$$\mathcal{Q}^{n+1} = \mathcal{Q}^n + \Delta\mathcal{Q}^n \qquad (15)$$

This procedure is repeated until the solution satisfies some convergence tolerance. Robustness of the method can be improved by including a timestep and applying an implicit-Euler approach. The matrix of the linear system now becomes:

$$\mathcal{A}(\mathcal{Q}^n) = \frac{\mathcal{V}}{\Delta t^n} + \left(\frac{\partial \mathcal{R}}{\partial \mathcal{Q}}\right)^n \qquad (16)$$

where $\mathcal{V}$ is a diagonal matrix of cell volumes, and Δt^n is the timestep. When the timestep is increased towards infinity, Newton's method is approached.

5.2 The linear system

The linear system that arises every Newton iteration is large and sparse for practical problems. In addition, the matrix is non-symmetric due to the hyperbolic nature of the Navier-Stokes equations. Krylov subspace methods can be used to solve this class of problems. In particular, the generalized minimum residual method (GMRES) developed by Saad and Schultz [24] is found to be effective for aerodynamic applications. This method has the property of minimizing the 2-norm of the residual over all vectors in the Krylov subspace. A new search direction is constructed every iteration and is added to the subspace, thus progressively improving the solution. On the other hand, more search directions incur extra memory and computational costs. We found a non-restarted GMRES with 50 search directions to be sufficient for most cases.

Complete solving of the linear system is found to be unnecessary to obtain quadratic convergence [25]. An inexact Newton method can be utilized which leads to efficient algorithms by avoiding oversolving of the linear system. The linear system is solved until the solution satisfies a tolerance specified by a parameter η_n:

$$||\mathcal{R}(\mathcal{Q}^n) + \mathcal{A}(\mathcal{Q}^n)\Delta\mathcal{Q}^n|| \leq \eta_n ||\mathcal{R}(\mathcal{Q}^n)|| \qquad (17)$$

The GMRES algorithm allows a matrix-free implementation; the matrix of the linear system is not required explicitly. The matrix-vector product can be calculated using finite differences:

$$\mathcal{A}v \simeq \frac{\mathcal{R}(\mathcal{Q} + \epsilon v) - \mathcal{R}(\mathcal{Q})}{\epsilon} + \frac{\mathcal{V}}{\Delta t} v \qquad (18)$$

This allows quadratic convergence of Newton's method because the matrix of the linear system is a complete linearization of the residual vector. Moreover, this approach reduces memory usage and avoids some difficulties during linearization. We use a matrix-free stepsize of:

$$\epsilon ||v|| = \sqrt{10^{-10}} \qquad (19)$$

following recent results from Chisholm and Zingg [17].

5.3 Preconditioning

Preconditioning transforms the linear system (written as $\mathcal{A}x$ = b) to one which has the same solution, but is easier to solve by an iterative solver. This reduces the number of inner iterations required. The right-preconditioned

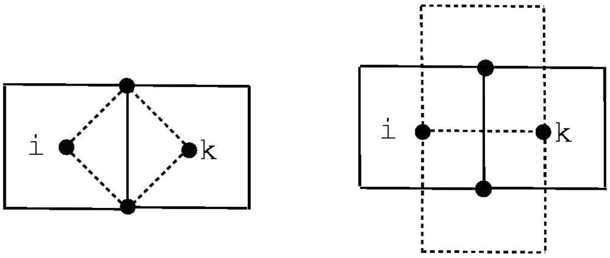

Figure 1: Calculation of the spatial derivatives by integration over (a) a diamond path, (b) a source-grid cell.

GMRES algorithm is based on solving

$$\mathcal{A}\mathcal{M}^{-1}u = b, u = \mathcal{M}x \tag{20}$$

with $\mathcal{M}$ as the preconditioner. The matrix $\mathcal{A}\mathcal{M}^{-1}$ should have a better condition number than the original matrix $\mathcal{A}$. In practice, an iterative solver will perform efficiently if the eigenvalues of $\mathcal{A}\mathcal{M}^{-1}$ are clustered around unity. An effective preconditioner $\mathcal{M}$ is chosen so that $\mathcal{M}^{-1}$ approximates $\mathcal{A}^{-1}$, while $\mathcal{M}^{-1}$ is easy to compute. This operation is performed every outer iteration.

Pueyo and Zingg [12] have constructed a preconditioner which works well for many aerodynamic flows. It is based on an incomplete-LU factorization (ILU(p)) of an approximate Jacobian after the reverse Cuthill-McKee (RCM) reordering of the unknowns. The parameter p controls the amount of fill. Increasing its value results in more accurate factors with extra storage and computational costs. The approximate Jacobian is constructed by a linearization of the flow equations with only second-difference dissipation. This improves the diagonal dominance of the matrix, and was found by Pueyo and Zingg to be more effective than the complete Jacobian. The coefficient of the dissipation term is calculated using

$$\varepsilon_p^{(2)} = \varepsilon^{(2)} + \sigma\varepsilon^{(4)} \tag{21}$$

with a parameter σ, where $\varepsilon^{(2)}$ and $\varepsilon^{(4)}$ are the coefficients of the dissipation term as defined in (14). The subscript p denotes the preconditioner. Chisholm and Zingg [16] have extended the approximate Jacobian from Pueyo and Zingg to incorporate the matrix-dissipation scheme. They suggest two parameters $V_{l,p}$ and $V_{n,p}$ to avoid overly small diagonal elements in the matrix. Hence the blend of scalar and matrix dissipation can be altered in the approximate Jacobian used to form the preconditioner. Values typically used are $V_{l,p}$ = $V_{n,p}$ = 0.6.

5.4 Preconditioning of the viscous term

The discretization of the viscous term produces a stencil involving the next-to-nearest neighboring terms. The inclusion of these terms in the preconditioner causes expensive ILU factorization and is found to be inefficient for three-dimensional cases. The baseline viscous term is calculated by:

$$\left(\int_{\partial\Omega} \mathbf{G}\cdot\hat{\mathbf{n}}dS\right)_i \simeq \sum_{ik} \mathbf{G}_{ik}\cdot\vec{\mathbf{n}}_{ik} \tag{22}$$

where $\mathbf{G}_{ik}$ = $\mathbf{G}(Q_{ik},\nabla Q_{ik})$ is the viscous flux on a face ik, with neighboring cells i and k. ∇Q is the gradient of the flow variables. This is calculated using:

$$\nabla Q_{ik} = \frac{1}{2}\left(\nabla Q_i + \nabla Q_k\right) \tag{23}$$

where

$$\nabla Q_i \simeq \frac{1}{V_i}\sum_{ik} Q_{ik}\vec{\mathbf{n}}_{ik} \tag{24}$$

and

$$Q_{ik} = \frac{1}{2}\left(Q_i + Q_k\right) \tag{25}$$

where V_i is the volume of cell i. Thus the viscous term involves next-to-nearest neighboring terms.

A study of several viscous operators that lead to a reduced stencil is performed. The first approach uses a truncated linearization in the preconditioning matrix, by setting:

$$\frac{\partial R_i}{\partial Q_{kk}} = 0 \tag{26}$$

where kk is a next-to-nearest neighbor of cell i. This approach only involves the nearest neighboring terms. It is referred as “distance-1 preconditioning” in the rest of the study.

The second approach approximates the gradient using an approximate-difference formula as suggested in references [26, 27]:

$$\nabla Q_{ik}\cdot\hat{\mathbf{n}}_{ik} \simeq \frac{Q_k - Q_i}{l_{ik}} \tag{27}$$

where l_{ik} is the distance between the centroids of cells i and k. This approach is efficient, and it has the same stencil as distance-1 preconditioning. However, this method is prone to inaccuracy on irregular grids when the line joining the centroids of cells i and k is not perpendicular to the face ik.

The third approach calculates the gradient on a face by integrating over a diamond-shaped control volume as developed by Coirier [28]. Flow variables at face vertices are approximated by averaging the surrounding grid nodes. This approach leads to the same stencil on triangular grids, but it has a larger stencil on structured grids when compared to the previous two methods.

The fourth approach calculates the gradient on a face by integrating over control volumes on the source grid [28]. This approach has the same stencil as the diamond-path approach. The last two approaches are illustrated in Figure 1. Extension of the viscous calculations to hybrid unstructured grids in three dimensions is straightforward.

5.5 Time-stepping strategy

Our startup strategy utilizes an implicit-Euler approach by introducing a timestep as given in (16). This improves both the stability of the nonlinear iterations and the conditioning of the linear system and thus results in a more robust procedure. On the other hand, the timestep affects the convergence rate. Therefore, it is important to choose a timestep that is both robust and efficient.

For the mean-flow equations, the local timestep following Pulliam [29] is utilized:

$$\Delta t_{flow} = \frac{\Delta t_{ref}}{1+\sqrt{V^{-1}}} \tag{28}$$

where V is the local cell volume. One way to calculate the reference timestep Δt_{ref} is to follow the switched evolution relaxation (SER) approach from Mulder and van Leer [30]:

$$\Delta t_{ref} = \alpha ||\mathcal{R}||_2^{-\beta} \tag{29}$$

where $||\mathcal{R}||_2$ is the residual norm. The idea is to increase the timestep inversely proportional to the residual norm, thus approaching Newton's method as the residual converges to zero. Other choices include the use of a constant value or a geometric series. These seem to be better choices for the startup stages due to their flexibility.

A spatially-varying timestep is used in the turbulence model following the approach of Chisholm and Zingg [16]. This approach prevents unstable solutions caused by negative values of $\tilde{\nu}$ by locally reducing the timestep. It allows larger timesteps to be used elsewhere in the domain. Moreover, this approach allows the use of a matrix-free implementation in the algorithm. The timestep is summarized as follows:

$$\Delta t_{turb} = \begin{cases} \Delta t_{flow} & \text{if } |\delta_e| < \delta_m \\ \\ |\Delta t_{limit}| & \text{otherwise} \end{cases} \tag{30}$$

where δ_e is an estimate of the solution update, and δ_m = $r\tilde{\nu}$ is the maximum allowable change specified by a parameter r. We use a value of r = 0.3. The estimate is determined using:

$$J_D \delta_e = -R \tag{31}$$

Case	M_∞	α°	Re
1	0.8395	3.06	11.7×10^6
2	0.8395	3.06	11.7×10^6
3	0.5	0.0	3.0×10^6

Table 1: Flow conditions.

Case	Geometry	Grid size
1	ONERA M6	179,000
2	ONERA M6	480,000
3	DLR-F6	431,000

Table 2: Geometry and grid size.

where J_D is the Jacobian and R is the right-hand side of the turbulence equation. The limiting timestep is calculated by:

$$\left(\frac{V}{\Delta t_{limit}} + J_D\right)\delta_m = -R \tag{32}$$

Further details about the local timestep can be found in the original work by Chisholm and Zingg [17].

6 Results

Three turbulent cases are studied. The first two are transonic flows over a wing. The third case is a subsonic flow over a wing-body configuration. Flow conditions are summarized in Table 1. The cases are assumed to be fully turbulent. All cases are run on a 1 GHz alpha EV68 processor at the high-performance advanced computing facility in the University of Toronto Institute for Aerospace Studies.

6.1 Grid generation

The ICEMCFD grid generator is utilized to generate the grids for the test cases. Prism layers are generated by extruding 15 layers of prism elements from the surface mesh using a growth ratio of 1.5. The offwall spacing is 10^{-6} times the chord at the wing root. The far-field boundary is specified at 12 wing-root chords from the wing. It is located at 12 times the length of the fuselage from the wing-body configuration.

The geometry and grid size are summarized in Table 2. A grid with 179,000 nodes is generated for the first case. Figure 2 shows the grid for the second case, with a close-up of the leading edge at the wing root. It is a finer grid with 480,000 nodes consisting of both tetrahedral and prismatic cells. The wing surface as well as the volume

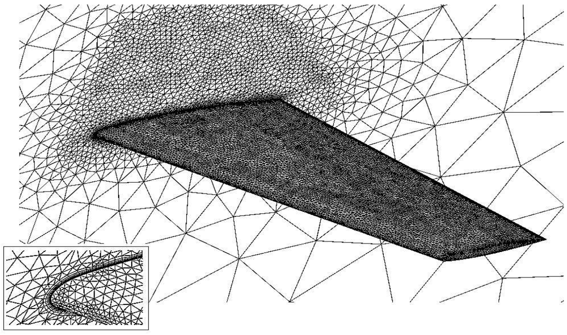

Figure 2: ONERA M6 wing grid with 480,000 nodes.

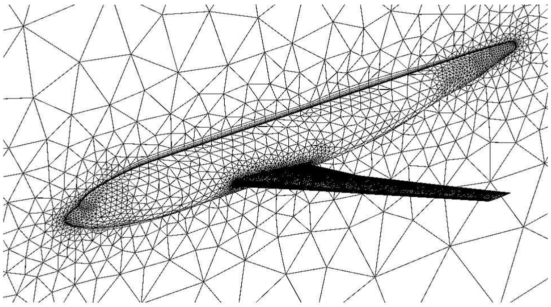

Figure 3: DLR-F6 wing-body grid with 431,000 nodes.

region above the wing are refined to provide a better solution of the shock wave. Figure 3 shows the grid with 431,000 nodes for the third case. None of these grids are expected to be sufficiently fine to achieve a low numerical error in drag.

6.2 Solver parameters

The linear system is solved using a matrix-free non-restarted version of GMRES with 50 Krylov vectors. A linear system tolerance of $\eta = 10^{-2}$ is used in this work, based on a study given in a later part of the paper. The preconditioner is ILU(1) based on an approximate Jacobian matrix after the reverse Cuthill-McKee reordering of the unknowns. Values of $\sigma = 10$, $V_{l,p} = V_{n,p} = 0.6$ are utilized in the approximate Jacobian.

Startup is initiated using a first-order scalar scheme before switching to the matrix-dissipation scheme. Switching is triggered when the mean-flow residual converges to 10^{-4}. The first-order scheme is defined with $\varepsilon^{(2)} = 1/4$, $\varepsilon^{(4)} = 0$, and $V_l = V_n = 1$, where $\varepsilon^{(2)}$ and $\varepsilon^{(4)}$ are the coefficients of the dissipation term as given in (14).

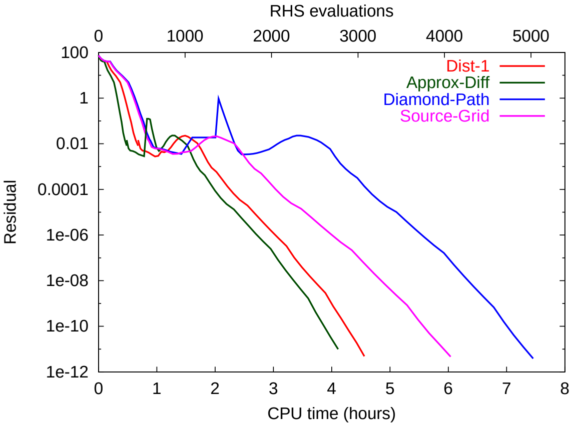

Figure 4: Case 1 convergence histories using different viscous-term calculations in the preconditioner.

One set of timestep parameters is used for the three cases in this work. We use $\Delta t_{ref} = 1$ for the first three iterations. After that, Δt_{ref} is set to 20 and the value is doubled every 5 iterations. To prevent the solution from becoming unstable with too large a timestep, the solution update is checked every Newton iteration. If non-physical flow quantities are encountered, (i.e. negative pressure or density), then the recent solution update is rejected and Δt_{ref} for the next iteration is halved. A similar safeguarding mechanism is used in the work by Smith et al. [26]. The same timestep sequence is used for the first-order stage as well as the matrix-dissipation stage.

A nonzero initial solution of $\tilde{\nu} = 10\nu_\infty$ is used for the turbulence model, as suggested by Chisholm and Zingg [16].

6.3 Preconditioning of the viscous term

Figure 4 depicts the convergence histories for Case 1 using four different calculations of the viscous term in the preconditioner. The baseline viscous calculation as given in (22) and (23) is used on the right-hand side; thus these cases all converge to the same solution. It is observed that distance-1 preconditioning and the approximate-difference formula have faster convergence than the other two approaches. The distance-1 viscous preconditioner is used in the rest of the study. Convergence to 10^{-12} for Case 1 using distance-1 viscous preconditioning is obtained in 4.5 hours or the equivalent of 3,000 residual evaluations. It requires 50 outer and 1,200 inner iterations in total.

Preconditioner	Storage	i-it	t_f	t_s	$t_f + t_s$
ILU(0)	1.0	33	10	226	236
ILU(1)	2.0	24	29	173	202
ILU(2)	3.8	12	96	101	197
ILU(3)	6.4	10	269	106	375
ILU(4)	9.8	9	638	121	759
ILUT(10^{-3},20)	1.3	42	548	370	918
ILUT(10^{-3},80)	2.3	21	1,176	219	1,395
ILUT(10^{-3},160)	3.3	15	2,065	197	2,262
ILUT(10^{-5},20)	1.4	42	2,802	397	3,199
ILUT(10^{-5},80)	2.9	20	7,365	251	7,616
ILUT(10^{-5},160)	4.6	14	14,971	230	15,201

Table 3: Memory, cost and effectiveness to reduce the inner residual by two orders of magnitude for different preconditioners.

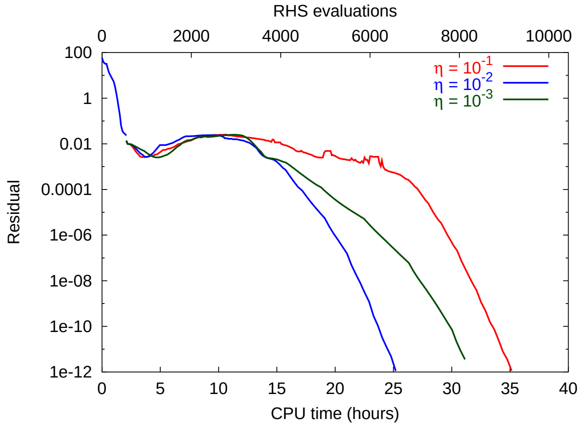

Figure 5: Case 2 convergence histories using different tolerances in the linear solver.

6.4 Incomplete factorization

The drop-tolerance strategy ILUT is studied and compared to ILU(p) preconditioning. Table 3 tabulates memory, cost and effectiveness to reduce the linear residual by two orders-of-magnitude for several preconditioners. The study is performed on Case 1. The linear system that arises when the non-linear residual is 10^{-4} is studied. In the table, i-it is the number of inner iterations, t_f is the time to factorize the matrix, and t_s is the time to solve the system. In this study, ILU(p) is found to be more efficient than ILUT. The ILU(p) preconditioner with $p = 1$ is found to be the best choice for this case based on both memory and cost considerations.

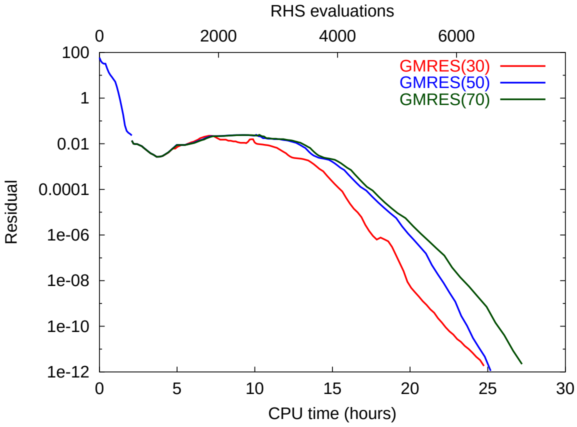

Figure 6: Case 2 convergence histories using different sizes of the Krylov subspace.

Convergence criterion	CPU time (hours)
0.5% of C_L	17.2
0.1% of C_L	18.8
0.01% of C_L	20.6
0.5% of C_D	16.9
0.1% of C_D	18.6
0.01% of C_D	20.5

Table 4: Convergence data for the lift and drag coefficients for Case 2.

6.5 Convergence results

The distance-1 viscous formulation is chosen to construct the preconditioning matrix in the rest of the study. ILU(1) is used as the preconditioner. Figure 5 shows the convergence for Case 2 using different linear system tolerances. Convergence to 10^{-12} is obtained in 25 hours or the equivalent of 6,000 residual evaluations for this half-million-node case using a linear system tolerance of $\eta = 10^{-2}$. It requires 148 outer and 2,300 inner iterations. The use of a larger inner tolerance of 10^{-1} is found to produce a longer startup stage with an increased number of outer iterations, while a smaller inner tolerance of 10^{-3} leads to slower asymptotic convergence with an increased number of inner iterations.

Figure 6 shows the convergence using different sizes of the Krylov subspace. The use of GMRES(30) is found to converge faster than GMRES(50) during startup, but leads to a slower asymptotic convergence.

Convergence of lift and drag coefficients with $\eta = 10^{-2}$ and GMRES(50) is given in Figure 7. The time required to converge the force coefficients to some specified tolerances is summarized in Table 4. It requires 17

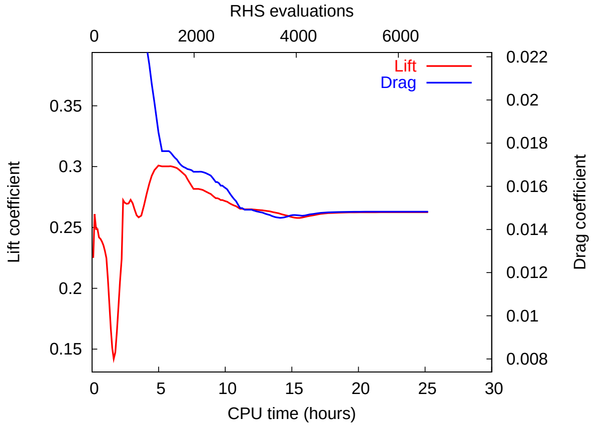

Figure 7: Convergence of lift and drag coefficients for Case 2.

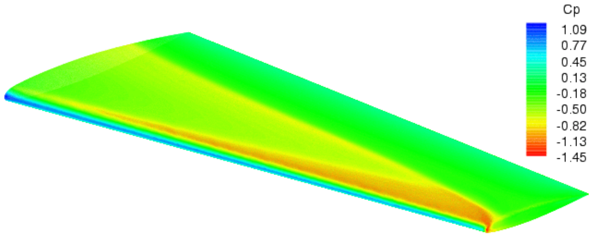

Figure 8: Pressure contours over the ONERA M6 wing at $M_\infty = 0.8395$, $\alpha = 3.06°$, and $Re = 11.7 \times 10^6$.

hours to converge to within 0.5% of the converged lift and drag coefficients, which are 0.263 and 0.0148 respectively. Figure 8 shows the pressure contours over the wing. The pressure coefficients at different wingspan locations are compared to experimental data in Figure 11.

Figure 9 shows the convergence for the third case over the wing-body configuration. Convergence to 10^{-12} is obtained in 20 hours with the equivalent of 6,000 residual evaluations. It requires 165 outer and 2,000 inner iterations in total. The pressure contours over the wing-body configuration are shown in Figure 10.

7 Conclusions

A Newton-Krylov algorithm is presented for turbulent aerodynamic flows. Convergence to 10^{-12} for half-million-node three-dimensional cases can be obtained in 20-25 hours on a single processor.

The inclusion of the next-to-nearest neighboring terms in the viscous operator causes preconditioning to become impractical for three-dimensional applications. Four approaches are suggested as alternatives and are found to be viable options. The distance-1 viscous preconditioning as well as the approximate-difference approach are selected based on efficiency considerations. The ILU(p) and ILUT preconditioners are studied; the former is found to be more efficient.

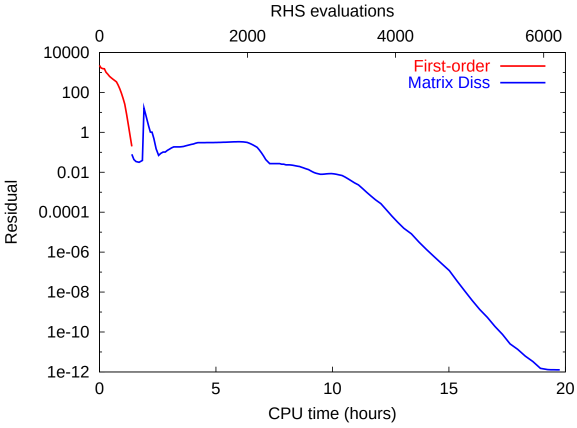

Figure 9: Case 3 convergence history.

Current results have motivated further research to improve the efficiency of the current algorithm. Future work includes investigation of preconditioning and startup strategies and the choice of solver parameters. The algorithm will also be extended to parallel to further reduce computational time. The improved algorithm will be applied to computations on finer grids to produce more accurate flow solutions.

8 Acknowledgments

This research was supported by Bombardier Aerospace and an OGS grant of the Government of Ontario. The authors would like to thank Prof. Jason Lassaline and Todd Chisholm for many useful discussions.

References

[1] Johnson, F. T., Tinoco, E. N., and Yu, N. J., "Thirty Years of Development and Application of CFD at Boeing Commercial Airplanes, Seattle," *AIAA Paper 2003-3439*, 2003.

[2] Nelson, T. E. and Zingg, D. W., "Fifty Years of Aerodynamics: Successes, Challenges, and Opportunities," *CAS Journal*, Vol. 50, No. 1, 2004, pp. 61–84.

[3] Lee-Rausch, E. M., Frink, N. T., Mavriplis, D. J., Rausch, R. D., and Milholen, W. E., "Transonic Drag Prediction on a DLR-F6 Transport Configura-

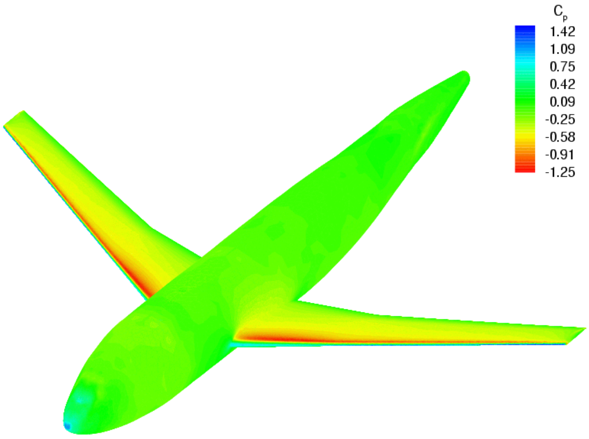

Figure 10: Pressure contours over the DLR-F6 wing-body configuration at M_∞ = 0.5, α = 0°, and Re = 3 × 10^6.

tion Using Unstructured Grid Solvers," *AIAA Paper 2004-0554*, 2004.

[4] May, G., van der Weide, E., Jameson, A., Sriram, and Martinelli, L., "Drag Prediction of the DLR-F6 Configuration," *AIAA Paper 2004-0396*, 2004.

[5] Luo, H., Baum, J. D., and Löhner, R., "High-Reynolds Number Viscous Flow Computations Using an Unstructured-Grid Method," *AIAA Paper 2004-1103*, 2004.

[6] Knoll, D. A. and Keyes, D. E., "Jacobian-free Newton-Krylov methods: a Survey of Approaches and Applications," *Journal of Computational Physics*, Vol. 193, 2004, pp. 357–397.

[7] Venkatakrishnan, V. and Mavriplis, D. J., "Implicit Solvers for Unstructured Meshes," *Journal of Computational Physics*, Vol. 105, 1992, pp. 83–91.

[8] Barth, T. J. and Linton, S. W., "An Unstructured Mesh Newton Solver for Compressible Fluid Flow and its Parallel Implementation," *AIAA Paper 95-0221*, 1995.

[9] Nielsen, E. J., Anderson, W. K., Walters, R. W., and Keyes, D. E., "Application of Newton-Krylov Methodology to a Three-Dimensional Unstructured Euler Code," *AIAA Paper 95-1733*, 1995.

[10] Anderson, W. K., Rausch, R. D., and Bonhaus, D. L., "Implicit/Multigrid Algorithms for Incompressible Turbulent Flows on Unstructured Grids," *Journal of Computational Physics*, Vol. 128, 1996, pp. 391–408.

[11] Blanco, M. and Zingg, D. W., "Fast Newton-Krylov Method for Unstructured Grids," *AIAA Journal*, Vol. 36, No. 4, 1998, pp. 607–612.

[12] Pueyo, A. and Zingg, D. W., "Efficient Newton-Krylov Solver for Aerodynamic Computations," *AIAA Journal*, Vol. 36, No. 11, 1998, pp. 1991–1997.

[13] Geuzaine, P., Lepot, I., Meers, F., and Essers, J. A., "Multilevel Newton-Krylov Algorithms for Computing Compressible Flows on Unstructured Meshes," *AIAA Paper 99-3341*, 1999.

[14] Geuzaine, P., "Newton-Krylov Strategy for Compressible Turbulent Flows on Unstructured Meshes," *AIAA Journal*, Vol. 39, No. 3, 2000, pp. 528–531.

[15] Nemec, M. and Zingg, D. W., "Newton-Krylov Algorithm for Aerodynamic Design Using the Navier-Stokes Equations," *AIAA Journal*, Vol. 40, No. 6, 2002, pp. 1146–1154.

[16] Chisholm, T. and Zingg, D. W., "A Newton-Krylov Algorithm for Turbulent Aerodynamic Flows," *AIAA Paper 2003-0071*, 2003.

[17] Chisholm, T. and Zingg, D. W., "Start-up Issues in a Newton-Krylov Algorithm for Turbulent Aerodynamic Flows," *AIAA Paper 2003-3708*, 2003.

[18] Manzano, L. M., Lassaline, J. V., Wong, P., and Zingg, D. W., "A Newton-Krylov Algorithm for the Euler Equations Using Unstructured Grids," *AIAA Paper 2003-0274*, 2003.

[19] Spalart, P. R. and Allmaras, S. R., "A One-Equation Turbulence Model for Aerodynamic Flows," *AIAA Paper 92-0439*, 1992.

[20] Spalart, P. R. and Allmaras, S. R., "A One-Equation Turbulence Model for Aerodynamic Flows," *La Recherche Aérospatiale*, No. 1, 1994, pp. 5–21.

[21] Ashford, G. A., *An Unstructured Grid Generation and Adaptive Solution Technique for High Reynolds Number Compressible Flows*, Ph.D. thesis, University of Michigan, 1996.

[22] Mavriplis, D. J. and Venkatakrishnan, V., "A Unified Multigrid Solver for the Navier-Stokes Equations on Mixed Element Meshes," *AIAA Paper 95-1666*, 1995.

[23] Swanson, R. C. and Turkel, E., "On Central-Difference and Upwind Schemes," *J. Comp. Phys.*, Vol. 101, 1992, pp. 292–306.

[24] Saad, Y. and Schultz, M. H., "GMRES: A Generalized Minimum Residual Algorithm For Solving Nonsymmetric Linear Systems," *SIAM J. Sci. Stat. Computing*, Vol. 7, 1986, pp. 856–869.

[25] Eisenstat, S. C. and Walker, H. F., "Choosing the Forcing Terms in an Inexact Newton Method," *SIAM J. Sci. Comput.*, Vol. 17, No. 1, 1996, pp. 16–32.

[26] Smith, T. M., Hooper, R. W., Ober, C. C., Lorber, A. A., and Shadid, J. N., "Comparison of Operators for Newton-Krylov Method for Solving Compressible Flows on Unstructured Meshes," *AIAA Paper 2004-0743*, 2004.

[27] Mavriplis, D. J., "On Convergence Acceleration Techniques for Unstructured Meshes," *AIAA Paper 98-2966*, 1998.

[28] Coirier, W. J., *An Adaptively-Refined, Cartesian, Cell-Based Scheme for the Euler and Navier-Stokes Equations*, Ph.D. thesis, University of Michigan, 1994.

[29] Pulliam, T. H., "Efficient Solution Methods for the Navier-Stokes Equations," Tech. rep., Lecture Notes for the von Kármán Inst. for Fluid Dynamics Lecture Series: Numerical Techniques for Viscous Flow Computation in Turbomachinery Bladings, Brussels, Belgium, Jan. 1986.

[30] Mulder, W. A. and van Leer, B., "Experiments with Implicit Upwind Methods for the Euler Equations," *Journal of Computational Physics*, Vol. 59, 1985, pp. 232–246.

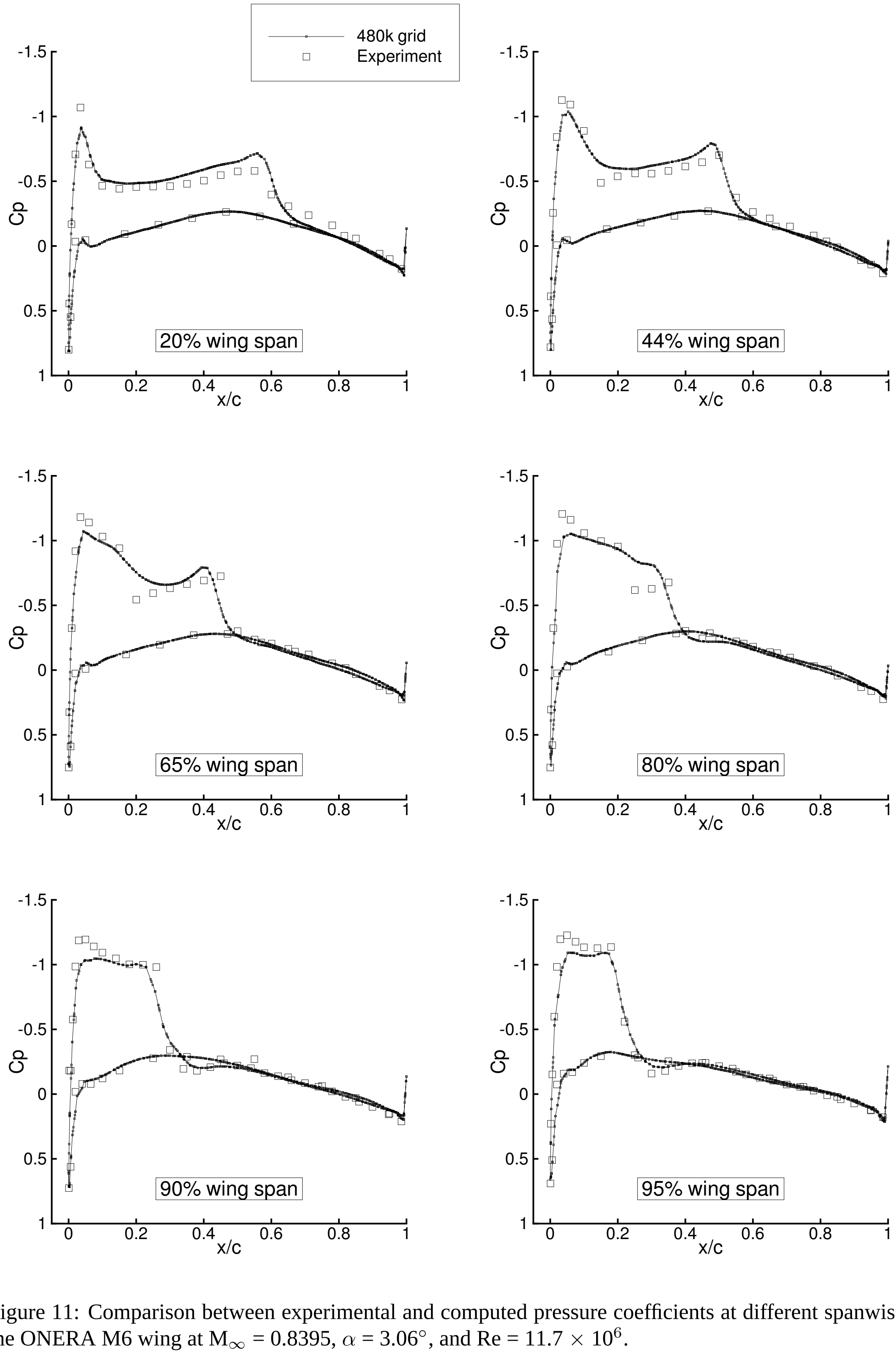

Figure 11: Comparison between experimental and computed pressure coefficients at different spanwise locations for the ONERA M6 wing at $M_\infty = 0.8395$, $\alpha = 3.06°$, and Re = 11.7×10^6.