Toward an Optimal Solver for Time-spectral Solutions on Unstructured Meshes

Nathan L. Mundis * Dimitri J. Mavriplis †

Department of Mechanical Engineering, University of Wyoming, Laramie, Wyoming 82071-3295

The time-spectral method applied to the Euler and coupled aeroelastic equations theoretically offers significant computational savings for purely periodic problems when compared to standard time-implicit methods. However, attaining superior efficiency with time-spectral methods over traditional time-implicit methods hinges on the ability rapidly to solve the large non-linear system resulting from time-spectral discretizations which become larger and stiffer as more time instances are employed or the period of the flow becomes especially short (i.e. the maximum resolvable wave-number increases). In order to increase the efficiency of these solvers, and to improve robustness, particularly for large numbers of time instances, the Generalized Minimal Residual Method (GMRES) is used to solve the implicit linear system over all coupled time instances. The use of GMRES as the linear solver makes time-spectral methods more robust, allows them to be applied to a far greater subset of time-accurate problems, including those with a broad range of harmonic content, and vastly improves the efficiency of time-spectral methods. In previous work, a wave-number independent preconditioner that mitigates the increased stiffness of the time-spectral method when applied to problems with large resolvable wave numbers has been developed. This preconditioner, however, directly inverts a large matrix whose size increases in proportion to the number of time instances. As a result, the computational time of this method scales as the cube of the number of time instances. In the present work, this preconditioner has been reworked to take advantage of an approximate-factorization approach that effectively decouples the spatial and temporal systems. Once decoupled, the time-spectral matrix can be inverted in frequency space, where it has entries only on the main diagonal and therefore can be inverted quite efficiently. This new GMRES/preconditioner combination is shown to be over an order of magnitude more efficient than the previous wave-number independent preconditioner for problems with large numbers of time instances and/or large reduced frequencies.

I. Introduction

For problems with strong periodic content, such as turbomachinery flows or rotorcraft aerodynamics, time-spectral (TS) methods can be used to reduce the cost of computing the full, time-dependent solution for a given level of accuracy. In many cases, time-spectral methods using only a small number of time instances per period can provide equivalent or superior accuracy, at substantially reduced cost, compared to traditional time-implicit solutions using hundreds of time steps per period. In other cases, many time instances per period may be needed to resolve both high and low frequency periodic content simultaneously.

Both the time-spectral and harmonic-balance methods are derived using discrete Fourier analysis. These methods, developed by Hall,[1] McMullen,[2,3] and Gopinath,[4,5] transform the unsteady equations in the physical domain to a set of steady equations in the frequency domain and then use the time-discretization operator to transform the frequency content back into a discrete number of time instances that reside in the time domain. Each of these time instances is coupled to all other time instances through the time-discretization operator, and the entire system is solved as a single, large, steady-state problem. In the literature, the time-spectral method has been shown to be faster than dual-time stepping implicit methods using backwards difference time formulae for time periodic computations, such as turbomachinery flows,[2,5] oscillatory pitching airfoil/wing cases,[4,6] flapping wing,[7] helicopter rotor[8,9] and vortex shedding problems,[3] which all use few time instances. However, for problems with large numbers of time instances, it is not yet clear if the theoretical efficiency gains of TS methods over time-implicit methods can be realized.

In order to converge the time-spectral system with large numbers of time instances efficiently and robustly, the Krylov subspace, Generalized Minimal Residual method[10] is utilized. By using GMRES, the disparate time instances

*Post-doctoral Research Associate, AIAA Member; email: nmundis@uwyo.edu.

†Professor, AIAA Associate Fellow; email: mavripl@uwyo.edu.

are much more strongly coupled than they would be if a stationary iterative method such as the Jacobi method were used as the main, linear solver. However, time-spectral problems can still become difficult to converge when the maximum resolvable wave-number becomes large, which occurs as the reduced frequency of motion increases and/or the number of time instances used is raised. To remedy this convergence degradation, a preconditioning method that allows the pseudo-time step size to be independent of wave-number has been developed. This method inverts the complete spatial-temporal diagonal block at each preconditioning iteration, in contrast to a previously developed solver, which inverts only the uncoupled spatial-diagonal blocks for each time instance individually.[11,12] Further improvements are made by applying an approximate factorization algorithm (AF), effectively decoupling the effects of the spatial and temporal discretizations. The time-spectral approximate-factorization algorithm was previously developed by Thomas et al.[13] and Leffell et al.,[14,15] although the version used in the present work has been modified to solve the unstructured spatial discretization through an iterative block-colored Gauss-Seidel approach. Additionally, while the cited works use approximate factorization as the main linear solver, it is used as a preconditioner for the GMRES linear solver herein. A similar effort to use AF as a preconditioner for GMRES is currently being undertaken by Leffell et al.[16] The present work focuses on how this newly developed preconditioner improves the convergence of the time-spectral method for problems with high maximum resolvable wave-numbers. This solver is then extended to aeroelastic problems following the same fully-coupled approach used in previous work[12] but using the approximate-factorization preconditioner outlined above.

In the following sections, the necessary components of the time-spectral time discretization and the application of the new preconditioner paradigm to this system are presented. First, the governing equations and the base solver for the Euler equations are outlined. Then, the additions to the flow solver required to implement the time-spectral method are discussed. Next, the linearization of the time-spectral Euler equations and the solution of the resultant linear system using GMRES are examined. Subsequently, the newly improved approximate-factorization, wave-number independent preconditioner is explained. Then, this new solver/preconditioner combination is extended to aeroelastic problems. Following, results are presented for the new preconditioner. Additionally, the solver utilizing the approximate-factorization preconditioner is compared, in depth, to the previously best solver.[17] Finally, conclusions about the performance of the new solver are drawn, its limitations are outlined, and improvements for the future are planned.

II. Governing Equations

A. Base Solver

The Euler equations in conservative form can be written as follows:

$$\frac{\partial \mathbf{U}}{\partial t} + \nabla \cdot \mathbf{F}(\mathbf{U}) = 0 \tag{1}$$

where $\mathbf{U}$ represents the vector of conserved quantities (mass, momentum, and energy) and $\mathbf{F}(\mathbf{U})$ represents the convective fluxes. Integrating over a (moving) control volume $\Omega(t)$, the following is obtained:

$$\int_{\Omega(t)} \frac{\partial \mathbf{U}}{\partial t} dV + \int_{\partial\Omega(t)} (\mathbf{F}(\mathbf{U}) \cdot \tilde{\mathbf{n}}) dS = 0. \tag{2}$$

Using the following differential identity:

$$\frac{\partial}{\partial t} \int_{\Omega(t)} \mathbf{U} dV = \int_{\Omega(t)} \frac{\partial \mathbf{U}}{\partial t} dV + \int_{\partial\Omega(t)} \mathbf{U}(\dot{\mathbf{x}} \cdot \tilde{\mathbf{n}}) dS \tag{3}$$

where $\dot{\mathbf{x}}$ and $\tilde{\mathbf{n}}$ are the velocity and normal of the interface $\partial\Omega(t)$, respectively, equation (2) becomes:

$$\frac{\partial}{\partial t} \int_{\Omega(t)} \mathbf{U} dV + \int_{\partial\Omega(t)} (\mathbf{F}(\mathbf{U}) - \mathbf{U}\dot{\mathbf{x}}) \cdot \tilde{\mathbf{n}} dS = 0. \tag{4}$$

Considering $\mathbf{U}$ as cell averaged quantities, these equations are discretized in space as:

$$\frac{\partial}{\partial t}(V\mathbf{U}) + \mathbf{R}(\mathbf{U}, \dot{\mathbf{x}}(t), \tilde{\mathbf{n}}(t)) = 0 \tag{5}$$

where $\mathbf{R}(\mathbf{U}, \dot{\mathbf{x}}, \tilde{\mathbf{n}}) = \int_{\partial\Omega(t)} (\mathbf{F}(\mathbf{U}) - \dot{\mathbf{x}}\mathbf{U}) \cdot \tilde{\mathbf{n}} dS$ represents the discrete convective fluxes in arbitrary-Lagrangian-Eulerian (ALE) form and V denotes the control volume. In the discrete form, $\dot{\mathbf{x}}(t)$ and $\tilde{\mathbf{n}}(t)$ now represent the time varying velocities and surface normals of the control-volume boundary faces.

The Euler equations are discretized by a central difference finite-volume scheme with additional matrix-based artificial dissipation on hybrid meshes which may include triangles and quadrilaterals in two dimensions. Second-order accuracy, on smoothly varying meshes, is achieved using a two-pass construction of the artificial dissipation operator, which corresponds to an undivided biharmonic operator. A single unifying face-based data-structure is used in the flow solver for all types of elements. For a given face, the residual contribution of that face can be written as follows:

$$\mathbf{R}_{1st O,ik}(\mathbf{U},\dot{\mathbf{x}}(t),\tilde{\mathbf{n}}(t)) = (\mathbf{F}_i(\mathbf{U}_i)+\mathbf{F}_k(\mathbf{U}_k)-\mathbf{U}_{ik}\dot{\mathbf{x}}_{ik})\cdot\tilde{\mathbf{n}}\Delta S+\kappa\underline{\mathbf{T}}|\underline{\Lambda}|\underline{\mathbf{T}}^{-1}(\mathbf{U}_i-\mathbf{U}_k) \tag{6}$$

for first-order matrix dissipation, where $\underline{\mathbf{T}}$ is the right-eigenvector matrix, $\underline{\Lambda}$ is the eigenvalue matrix, and $\underline{\mathbf{T}}^{-1}$ is the left-eigenvector matrix of the convective fluxes. For second-order matrix dissipation, the residual on a face can be written as follows:

$$\mathbf{R}_{2ndO,ik}(\mathbf{U},\dot{\mathbf{x}}(t),\tilde{\mathbf{n}}(t)) = (\mathbf{F}_i(\mathbf{U}_i)+\mathbf{F}_k(\mathbf{U}_k)-\mathbf{U}_{ik}\dot{\mathbf{x}}_{ik})\cdot\tilde{\mathbf{n}}\Delta S-\kappa\underline{\mathbf{T}}|\underline{\Lambda}|\underline{\mathbf{T}}^{-1}(\mathbf{L}_i(\mathbf{U})-\mathbf{L}_k(\mathbf{U})) \tag{7}$$

where $\mathbf{L_p}(\mathbf{U})$ is the undivided Laplacian operator, taken as:

$$\mathbf{L}_p(\mathbf{U}) = \sum_{q=1}^{neighbors}(\mathbf{U}_q-\mathbf{U}_p) \tag{8}$$

In both cases, κ is an empirical constant with a typical value of $1/2$ for first-order matrix dissipation and $1/8$ for second-order matrix dissipation.

B. Time-spectral Method

If the flow is periodic in time, the variables $\mathbf{U}$ can be represented by a discrete Fourier series. The discrete Fourier transform of $\mathbf{U}$ in a period of T is given as follows:[4]

$$\widehat{\mathbf{U}}_k = \frac{1}{N}\sum_{n=0}^{N-1}\mathbf{U}^n e^{-ik\frac{2\pi}{T}n\Delta t} \tag{9}$$

where N is the number of time intervals and $\Delta t = T/N$. The Fourier inverse transform is then given as follows:

$$\mathbf{U}^n = \sum_{k=-\frac{N}{2}}^{\frac{N}{2}-1}\widehat{\mathbf{U}}_k e^{ik\frac{2\pi}{T}n\Delta t}. \tag{10}$$

It should be noted that $\frac{N}{2}$ is an integer division operation. Also note that this corresponds to a collocation approximation, i.e. the function $\mathbf{U}(t)$ is projected into the space spanned by the truncated set of complex exponential (spectral) functions, and the expansion coefficients (in this case the $\widehat{\mathbf{U}}_k$) are determined by requiring $\mathbf{U}(t)$ to be equal to its projection at N discrete locations in time, as given by equations (9) and (10). Differentiating equation (10) in time, the following is obtained:

$$\frac{\partial}{\partial t}(\mathbf{U}^n) = \frac{2\pi}{T}\sum_{k=-\frac{N}{2}}^{\frac{N}{2}-1} ik\widehat{\mathbf{U}}_k e^{ik\frac{2\pi}{T}n\Delta t}. \tag{11}$$

Substituting equation (9) into equation (11), the final form of the time-spectral time-derivative term is produced:[18,19]

$$\frac{\partial}{\partial t}(\mathbf{U}^n) = \sum_{j=0}^{N-1} d_n^j \mathbf{U}^j \tag{12}$$

where

$$d_n^j = \begin{cases} \frac{2\pi}{T}\frac{1}{2}(-1)^{n-j}\cot(\frac{\pi(n-j)}{N}) & n\neq j \\ 0 & n=j \end{cases} \tag{13}$$

for an even number of time instances and

$$d_n^j = \begin{cases} \frac{2\pi}{T}\frac{1}{2}(-1)^{n-j}\csc(\frac{\pi(n-j)}{N}) & n\neq j \\ 0 & n=j \end{cases} \tag{14}$$

for an odd number of time instances. Next, substitute equation (10) into equation (5), and require equation (5) to hold exactly at the same N discrete locations in time (i.e. multiply (5) by the Dirac delta test function $\delta(t-t^n)$ and integrate over all time), which yields the following time-spectral governing equation:

$$\sum_{j=0}^{N-1} d_n^j V^j \mathbf{U}^j + \mathbf{R}(\mathbf{U}^n, \dot{\mathbf{x}}^n, \tilde{\mathbf{n}}^n) = 0 \qquad n = 0,1,2,...,N-1. \tag{15}$$

This results in a system of N equations for the N time instances $\mathbf{U}^n$ which are all coupled through the summation over the time instances in the time derivative term. The spatial discretization operators remain unchanged in the time-spectral approach, with only the requirement that they be evaluated at the appropriate temporal location. Thus, the time-spectral method may be implemented without any modifications to an existing spatial discretization, requiring only the addition of the temporal discretization coupling term, although the multiple time instances must be solved simultaneously because of this coupling. It should be noted that an odd-even decoupling exists when even numbers of time instances are used, so this work uses odd numbers exclusively.

C. Fully Implicit Method

A common approach for solving the system of equations resulting from the time-spectral method (c.f. equation (15)) consists of adding a pseudo-time term as follows:

$$\frac{\partial}{\partial \tau}(V^n \mathbf{U}^n) + \sum_{j=0}^{N-1} d_n^j V^j \mathbf{U}^j + \mathbf{R}(\mathbf{U}^n, \dot{\mathbf{x}}^n, \tilde{\mathbf{n}}^n) = 0 \tag{16}$$

and time-stepping these equations until a pseudo-time steady state is achieved. However, for explicit pseudo-time stepping approaches, it has been shown that the pseudo-time step is limited by stability considerations according to the following restriction:[5]

$$\Delta\tau_n = CFL \frac{V^n}{\| \lambda \| + V^n k'} \tag{17}$$

where $\| \lambda \|$ is the spectral radius of the spatial discretization operator $\mathbf{R}(\mathbf{U}^n, \dot{\mathbf{x}}^n, \tilde{\mathbf{n}}^n)$ and k' represents the largest wave-number that can be resolved by the specified N time instances for the given period T:

$$k' = \begin{cases} \frac{\pi N}{T} & \text{if } N \text{ is even} \\ \frac{\pi (N-1)}{T} & \text{if } N \text{ is odd.} \end{cases} \tag{18}$$

This restriction on $\Delta\tau_n$ results in convergence degradation as the number of time instances is increased or the period decreased. The impact of this restriction can be eliminated by resorting to specific implicit approaches in pseudo-time. Such an approach has been derived in the literature[20] using a first-order backwards difference scheme in pseudo-time.

A more general strategy consists of devising a Newton approach for solving the fully coupled non-linear equations at all time instances given by equation (15) or (16). The Newton scheme takes the following form:

$$[\mathbf{A}]\,\Delta\mathbf{U} = -\sum_{j=0}^{N-1} d_n^j V^j \mathbf{U}^j - \mathbf{R}(\mathbf{U}^n, \dot{\mathbf{x}}^n, \tilde{\mathbf{n}}^n) \tag{19}$$

with the resulting Jacobian matrix given by:[20]

$$[\mathbf{A}] = \begin{bmatrix} \frac{V^0}{\Delta\tau_0}\underline{\mathbf{I}} + \underline{\mathbf{J}}_0 & V^1 d_0^1 \underline{\mathbf{I}} & \dots & V^{N-1} d_0^{N-1} \underline{\mathbf{I}} \\ V^0 d_1^0 \underline{\mathbf{I}} & \frac{V^1}{\Delta\tau_1}\underline{\mathbf{I}} + \underline{\mathbf{J}}_1 & \dots & V^{N-1} d_1^{N-1} \underline{\mathbf{I}} \\ \vdots & \vdots & \dots & \vdots \\ V^0 d_{N-1}^0 \underline{\mathbf{I}} & V^1 d_{N-1}^1 \underline{\mathbf{I}} & \dots & \frac{V^{N-1}}{\Delta\tau_{N-1}}\underline{\mathbf{I}} + \underline{\mathbf{J}}_{N-1} \end{bmatrix} \tag{20}$$

where a diagonal pseudo-time term can be included as shown for enhanced diagonal dominance of the Jacobian matrix and $\underline{\mathbf{I}}$ is the diagonal identity matrix for the entire spatial discretization of a single time instance. In the above matrix, $\underline{\mathbf{J}}_j$ corresponds to the Jacobian of the spatial-discretization operator evaluated at time instance j. For a first-order spatial discretization, $\underline{\mathbf{J}}_{j,1stO}$ is as follows:

$$\underline{\mathbf{J}}_{j,1stO} = \frac{\partial \mathbf{R}_{1stO}}{\partial \mathbf{U}}. \tag{21}$$

For a second-order spatial discretization, $\underline{\mathbf{J}}_{j,2ndO}$ contains many more entries, as each element of the mesh is not only influenced by its nearest neighbors, but also by the neighbors of its neighbors. The second-order Jacobian is derived as follows:

$$\underline{\mathbf{J}}_{j,2ndO} = \left.\frac{\partial \mathbf{R}_{2ndO}}{\partial \mathbf{U}}\right|_{L=constant} + \left.\frac{\partial \mathbf{R}_{2ndO}}{\partial \mathbf{L}}\right|_{U=constant} \cdot \frac{\partial \mathbf{L}}{\partial \mathbf{U}}. \tag{22}$$

For convenience, the time-spectral contribution to the matrix $[\mathbf{A}]$ can be written separately, as follows:

$$[\mathbf{D}_{TS}] = \begin{bmatrix} 0 & d_0^1\underline{\mathbf{I}} & \dots & d_0^{N-1}\underline{\mathbf{I}} \\ d_1^0\underline{\mathbf{I}} & 0 & \dots & d_1^{N-1}\underline{\mathbf{I}} \\ \vdots & \vdots & \ddots & \vdots \\ d_{N-1}^0\underline{\mathbf{I}} & d_{N-1}^1\underline{\mathbf{I}} & \dots & 0 \end{bmatrix} \tag{23}$$

such that $[\mathbf{A}] = \left[\frac{V^i}{\Delta\tau_i}\underline{\mathbf{I}} + \underline{\mathbf{J}}_i + V^j\underline{\mathbf{D}}_{TS}\right]$.

Returning to equation (19), each non-linear Newton iteration requires solving the linear system

$$[\mathbf{A}]\,\Delta\mathbf{U} = -\mathbf{R}_{\mathbf{TS}}(\mathbf{U}) \tag{24}$$

where $\mathbf{R}_{\mathbf{TS}}(\mathbf{U})$ represents the residual of the complete time-spectral system (i.e. right hand side of equation (19)), and $[\mathbf{A}]$ is the complete time-spectral Jacobian spanning all spatial and temporal degrees of freedom. Since direct inversion of $[\mathbf{A}]$ is generally intractable, an inexact Newton scheme can be formulated using an approximate representation of $[\mathbf{A}]$ which is simpler to invert. One possible simplification is to replace the exact spatial Jacobian in each diagonal block $\underline{\mathbf{J}}_i$ by the corresponding first-order Jacobian $\underline{\mathbf{J}}_{i,1stO}$ as is typically done for steady-state solvers. Another simplification consists of dropping all the off-diagonal terms representing the coupling between different time instances. When this is done, the linear system becomes decoupled among time instances and can be written as:

$$\left[\frac{V^i}{\Delta\tau_i}\underline{\mathbf{I}} + \underline{\mathbf{J}}_{i,1stO}\right]\Delta\mathbf{U}_i = -\mathbf{R}_{\mathbf{TS}}(\mathbf{U}) \tag{25}$$

for each time instance $i = 0, 1, 2, ..., N-1$. Alternatively, the diagonal blocks in the $[\mathbf{A}]$ matrix may be retained and the system solved in a block Jacobi fashion following:

$$\left[\frac{V^i}{\Delta\tau_i}\underline{\mathbf{I}} + \underline{\mathbf{J}}_{i,1stO}\right]\Delta\mathbf{U}_i^{l+1} = -\mathbf{R}_{\mathbf{TS}}(\mathbf{U}) - \sum_{j\neq i}\left[V^j d_i^j\underline{\mathbf{I}}\right]\Delta\mathbf{U}_j^l \tag{26}$$

where the block size corresponds to the entire spatial domain of each time instance and where l denotes the block Jacobi iteration index.

D. Block-colored Gauss-Seidel Linear Solver

As can be seen, both approaches require the inversion the first-order spatial Jacobian (augmented with a pseudo-time term) at each iteration. This may be accomplished using a suitable iterative solver such as block-colored Gauss-Seidel (BCGS). In this case, the block now corresponds to the $[4\times 4]$ block diagonal matrix for each two-dimensional cell of the mesh, and the iterative scheme can be written as follows, with the spatial and temporal off-diagonal terms retained:

$$\left[\frac{V^i}{\Delta\tau_i}\underline{\mathbf{I}} + \left[\underline{\mathbf{D}}_{i,1stO}\right]\right]\Delta\mathbf{U}_i^{l+1} = -\mathbf{R}_{\mathbf{TS}}(\mathbf{U}) - \sum_{j\neq i}\left[V^j d_i^j\underline{\mathbf{I}}\right]\Delta\mathbf{U}_j^l - \left[\underline{\mathbf{O}}_{i,1stO}\right]\Delta\mathbf{U}_i^l \tag{27}$$

where $\underline{\mathbf{D}}_{i,1stO}$ denotes the $[4\times 4]$ block matrix for the current cell in two dimensions, and $\underline{\mathbf{O}}_{i,1stO}$ refers to the off-diagonal blocks for neighboring mesh cells. Although this equation describes a block Jacobi iteration (at the mesh cell level), a block-colored Gauss-Seidel scheme can be recovered with a few simple modifications. First, the computational elements are divided into computational "colors" such that no two adjacent elements are the same color. This coloring allows the Gauss-Seidel method to be run in parallel. The BCGS method then updates (in parallel) all elements of each individual color (with the different colors updated in sequence), such that each update uses the newest information available for all other colors. The BCGS algorithm, written in residual form for a generic linear system is given in Algorithm (1).

Algorithm 1 : Block-colored Gauss-Seidel

1: Given $\underline{\mathbf{A}}\mathbf{x} = \mathbf{b}$
2: **for** i=1,...,ζ **do**
3: **for** j=1,...,n_{colors} **do**
4: **for** k=1,...,n_{e_j} **do**
5: Compute $\mathbf{r}_{i,k} = \mathbf{b} - \underline{\mathbf{A}}\mathbf{x}_{current}$
6: Compute $\Delta\mathbf{x}_{i,k} = \underline{\mathbf{D}}_k^{-1}\mathbf{r}_{i,k}$
7: Update $\mathbf{x}_{i+1,k} = \mathbf{x}_{i,k} + \Delta\mathbf{x}_{i,k}$
8: **end for**
9: **end for**
10: Compute $R_{L,i} = \|\mathbf{r}_i\|_2$
11: If satisfied Stop.
12: **end for**

In Algorithm (1), ζ is the maximum number of BCGS iterations allowed, n_{colors} is the number of colors into which the elements have been divided, and n_{e_j} is the number of elements having the jth color. Additionally, $\underline{\mathbf{D}}_k$ is the diagonal block of the Jacobian for element k, and is inverted directly using LU-decomposition. The specification $\mathbf{x}_{current}$ is used to indicate that some colors will have information from the previous iteration $\mathbf{x}_i$ while other colors might have already been updated during the current iteration $\mathbf{x}_{i+1}$; in other words, whether from the previous or current iteration, the most up-to-date information at the time of the evaluation of line 5 is used.

As noted earlier, the $\underline{\mathbf{A}}$ matrix given in lines 1 and 5 of the algorithm can either include or exclude the temporal off-diagonal terms given in equation (20). When these time-coupling terms are excluded, this corresponds to a BCGS solver with explicit time treatment, which is abbreviated "BCGS-EX." When the off-diagonal terms are included, implicit time treatment is used in the BCGS solver, and this method is abbreviated "BCGS-IM."

E. Generalized Minimal Residual Method

Despite the additional stability the above, implicit method affords, the time-spectral Euler equations still become difficult to solve as the number of time instances increases. With an increasing number of time instances, the Jacobian given in equation (20) proceeds farther and farther from diagonal dominance. To restore diagonal dominance, a decreasingly small pseudo-time-step size must be used. Thus, the potential efficiency gains of time-spectral methods over time-implicit methods begin to evaporate.

To regain the efficiency improvements afforded by time-spectral methods, a linear solver that does not require diagonal dominance of the Jacobian should be used. The Generalized Minimum Residual method is such a solver. A welcome byproduct of this solver choice is that GMRES also completely couples the various time instances. A flexible variant of the GMRES algorithm as described by Saad[10] is used. This flexible variant allows the use of an iterative method as preconditioner. The flexible GMRES (FGMRES) algorithm proceeds as given in Algorithm (2).

Algorithm 2 : Flexible GMRES

1: Given $\underline{\mathbf{A}}\mathbf{x} = \mathbf{b}$
2: Compute $\mathbf{r}_0 = \mathbf{b} - \underline{\mathbf{A}}\mathbf{x}_0$, $\beta = \|\mathbf{r}_0\|_2$, and $\mathbf{v}_1 = \mathbf{r}_0/\beta$
3: **for** j=1,...,n **do**
4: Compute $\mathbf{z}_j := \underline{\mathbf{P}}^{-1}\mathbf{v}_j$
5: Compute $\mathbf{w} := \underline{\mathbf{A}}\mathbf{z}_j$
6: **for** i=1,...,j **do**
7: $h_{i,j} := (\mathbf{w}, \mathbf{v}_i)$
8: $\mathbf{w} := \mathbf{w} - h_{i,j}\mathbf{v}_j$
9: **end for**
10: Compute $h_{j+1,j} = \|\mathbf{w}\|_2$ and $\mathbf{v}_{j+1} = \mathbf{w}/h_{j+1,j}$
11: Define $\underline{\mathbf{Z}}_m := [\mathbf{z}_1, \ldots, \mathbf{z}_m]$, $\underline{\bar{\mathbf{H}}}_m = \{h_{i,j}\}_{1\le i\le j+1; 1\le j\le m}$
12: **end for**
13: Compute $\mathbf{y}_m = argmin_y \|\beta\mathbf{e}_1 - \underline{\bar{\mathbf{H}}}_m\mathbf{y}\|_2$ and $\mathbf{x}_m = \mathbf{x}_0 + \underline{\mathbf{Z}}_m\mathbf{y}_m$
14: If satisfied Stop, else set $\mathbf{x}_0 \leftarrow \mathbf{x}_m$ and GoTo 1.

In this description, $\underline{\mathbf{A}}$ corresponds to the full and exact time-spectral Jacobian matrix $[\mathbf{A}]$ defined in equation (20)

with the second-order accurate spatial Jacobian, which may or may not be augmented with a pseudo-time-step term, $\mathbf{b}$ corresponds to the negative of the non-linear time-spectral residual $-\mathbf{R_{TS}}(\mathbf{U})$, and $\mathbf{x}$ is the non-linear update $\Delta\mathbf{U}$ to be computed.

Preconditioning is applied in line 4 of the algorithm. The BCGS linear solver is used for preconditioning in several different configurations as covered in the next subsection. A pseudo-time step must be applied to the BCGS system (i.e. added to $\underline{\mathbf{P}}$) to ensure diagonal dominance and convergence. To solve the minimization problem in line 13 of the algorithm, QR-factorization by means of Givens rotations is utilized.[10] Finally, line 14 of the algorithm indicates that this algorithm is, in fact, truncated, restarting GMRES using n Krylov vectors per restart.

Although Algorithm (2) shows the minimization problem outside the loop over Krylov vectors, in fact, this minimization is updated as each additional Krylov vector is added. This is done so that the current value of the linear residual is known for each iteration j. The FGMRES algorithm exits whenever this residual has either converged a specified amount or has converged to machine zero.

A pseudo-time step is used in the matrix $\underline{\mathbf{A}}$ in the FGMRES algorithm as well. By using this pseudo-time step within FGMRES itself the routine is modified positively as follows: an update that will avoid over-correction issues is used; the pseudo-time step is allowed to grow as the residual decreases so quadratic convergence can be retained; and the convergence rate of the FGMRES linear solver itself is increased (i.e. fewer Krylov vectors and/or restarts are required to converge the linear system) because the pseudo-time step term makes the Jacobian better conditioned for Krylov subspace methods. Thus, two different pseudo-time steps are used: one in the second-order Jacobian $\underline{\mathbf{A}}$ used by the FGMRES algorithm itself and another in the first-order Jacobian $\underline{\mathbf{P}}$ used in the BCGS portion of the preconditioner. Because the Jacobian used in the BCGS preconditioner must be diagonally dominant, the preconditioner pseudo-time step is almost always smaller and grows more slowly than the FGMRES pseudo-time step. Conversely, the pseudo-time step used in the FGMRES algorithm grows rapidly to infinity such that the diagonal term becomes vanishingly small and an exact Newton method is recovered after several orders of magnitude decrease in the non-linear residual.

1. *Preconditioning Methods for FGMRES*

As mentioned above, all preconditioning methods make use of the block-colored Gauss-Seidel solver in one form or another. The first preconditioning method uses BCGS with explicit treatment of time. In other words, Algorithm (1) is used, with a specified number of iterations ζ on each Krylov vector. Here the preconditioner matrix $\underline{\mathbf{P}}$ corresponds to the $[\mathbf{A}]$ matrix, given by equation (20), using the first-order spatial Jacobian with a pseudo-time step that is different, and smaller, than the pseudo-time step used in FGMRES. Additionally, this preconditioner matrix does not include the off-diagonal, time-coupling terms. FGMRES using this preconditioner is abbreviated as "GMRES-EX" in the remainder of this work.

The next preconditioner used is BCGS with implicit treatment of time. This preconditioner uses Algorithm (1) with a specified number of iterations ζ on each Krylov vector. The full $[\mathbf{A}]$ matrix, including the temporal off-diagonal time-spectral coupling terms, is used, but again, the first-order spatial Jacobian is used with a smaller pseudo-time step. This solver/preconditioner combination is abbreviated "GMRES-IM".

Both of the preconditioners presented thus far have a serious limitation: the pseudo-time step size needed to preserve diagonal-dominance of the Jacobian and an appropriately scaled update remains relatively small. In order to bypass this limitation, a preconditioner for FGMRES is formed as a defect correction method applied to the residual of equation (26) as:

$$\left[\frac{V^i}{\Delta\tau_{BCGS}}\underline{\mathbf{I}}+\underline{\mathbf{J}}_{i,1stO}\right](\Delta\mathbf{U}_i^{l+1}-\Delta\mathbf{U}_i^l)=-\mathbf{R_{TS}}(\mathbf{U})-\sum_{j\neq i}\left[V^j d_i^j\underline{\mathbf{I}}\right]\Delta\mathbf{U}_j^l-\left[\frac{V^i}{\Delta\tau_{FGMRES}}\underline{\mathbf{I}}+\underline{\mathbf{J}}_{i,1stO}\right]\Delta\mathbf{U}_i^l \tag{28}$$

where the right-hand-side corresponds to the residual of equation (26). A dual-iteration strategy is required for this preconditioner (thus the term defect-correction). Inner BCGS iterations are used to invert the left-hand side matrix providing an updated value for $\Delta\mathbf{U}_{i+1}^l$, which is then substituted into the right-hand side (RHS) terms, and the process is repeated, effectively driving the right-hand-side residual to zero after a number of outer iterations. The advantage of this approach comes from the fact that the pseudo-time step on the right-hand side residual is generally much larger than that required for stability of the BCGS iterative scheme; in fact, the FGMRES CFL number can be and is used as the defect correction step CFL number. This solver/preconditioner combination, which uses the first-order Jacobian on the RHS, is abbreviated "GMRES-DC1". The same defect-correction method can also use the second-order Jacobian on the RHS as follows:

$$\left[\frac{V^i}{\Delta\tau_{BCGS}}\underline{\mathbf{I}}+\underline{\mathbf{J}}_{i,1stO}\right](\Delta\mathbf{U}_i^{l+1}-\Delta\mathbf{U}_i^l)=-\mathbf{R_{TS}}(\mathbf{U})-\sum_{j\neq i}\left[V^j d_i^j\underline{\mathbf{I}}\right]\Delta\mathbf{U}_j^l-\left[\frac{V^i}{\Delta\tau_{FGMRES}}\underline{\mathbf{I}}+\underline{\mathbf{J}}_{i,2ndO}\right]\Delta\mathbf{U}_i^l, \tag{29}$$

which will be abbreviated as "GMRES-DC2".

All of the preconditioners outlined thus far require a pseudo-time step as defined by equation (17), which becomes smaller as the maximum wave-number resolvable by the time-spectral system increases. This wave-number increases, as can be seen from equation (18), as the length of the period of interest decreases or the number of time instances used increases. It was originally believed that the spatial spectral radius component in equation (17) would be dominant in almost all cases. Recent work, however, has indicated that the wave-number component dominates much more frequently than had been anticipated, especially for problems using large numbers of time instances such as those investigated in this and previous works.[11,12,14,15,21,22] As such, a wave-number independent preconditioner is derived by taking the GMRES-DC2 preconditioner, discussed above, and making one change: the $\underline{\mathbf{D}}_k$ given in line 6 of the BCGS algorithm, Algorithm (1) are changed to be the spatial-temporal diagonal blocks of the full, first-order accurate time-spectral Jacobian. These spatial-temporal diagonal blocks look similar to the time-spectral Jacobian given by equation (20), as follows:

$$\underline{\mathbf{D}}_{ST,k} = \begin{bmatrix} \frac{V_k^0}{\Delta\tau_0}\underline{\mathbf{I}} + \underline{\mathbf{D}}_{S,k,0} & V_k^1 d_0^1 \underline{\mathbf{I}} & \dots & V_k^{N-1} d_0^{N-1} \underline{\mathbf{I}} \\ V_k^0 d_1^0 \underline{\mathbf{I}} & \frac{V_k^1}{\Delta\tau_1}\underline{\mathbf{I}} + \underline{\mathbf{D}}_{S,k,1} & \dots & V_k^{N-1} d_1^{N-1} \underline{\mathbf{I}} \\ \vdots & \vdots & \dots & \vdots \\ V_k^0 d_{N-1}^0 \underline{\mathbf{I}} & V_k^1 d_{N-1}^1 \underline{\mathbf{I}} & \dots & \frac{V_k^{N-1}}{\Delta\tau_{N-1}}\underline{\mathbf{I}} + \underline{\mathbf{D}}_{S,k,N-1} \end{bmatrix} \tag{30}$$

where $\underline{\mathbf{D}}_{ST,k}$ is the full spatial-temporal diagonal block of the time-spectral Jacobian for spatial element k and $\underline{\mathbf{D}}_{S,k,j}$ and V_k^j are the spatial diagonal block and element volume, respectively, for spatial element k and time instance j. Thus, for the two-dimensional Euler equations, each $\underline{\mathbf{D}}_{ST,k}$ is of the size $[4N \times 4N]$. By inverting the spatial-temporal diagonal blocks in the BCGS algorithm, the pseudo-time-step size needed to restore diagonal dominance in the preconditioner becomes as follows:

$$\Delta\tau_n = CFL \frac{V^n}{\| \lambda \|} \tag{31}$$

which, as can be seen, has achieved the goal of wave-number independence. The solver/preconditioner combination which utilizes spatial-temporal diagonal-block inversion is abbreviated "GMRES-STI".

Although GMRES-STI accomplishes the goal of wave-number independence, it has been shown in previous work[17] that it involves the very costly direct inversion of a matrix whose size is $[4N \times 4N]$, where N is, as usual, the number of time instances. To form a more efficient preconditioner that still possess the essential property of wave-number independence, an approximate-factorization preconditioner is developed along the same lines as found in the literature.[13,14,15] The approximate-factorization algorithm factors the Jacobian into an approximate form as follows:

$$\begin{aligned} &\left[\underline{\mathbf{I}} + \Delta\tau_{BCGS}\underline{\mathbf{D}}_{TS}\right]\left[\frac{V^i}{\Delta\tau_{BCGS}}\underline{\mathbf{I}} + \underline{\mathbf{J}}_{i,1stO}\right](\Delta\mathbf{U}_i^{l+1} - \Delta\mathbf{U}_i^l) \\ &= -\mathbf{R}_{\mathbf{TS}}(\mathbf{U}) - \sum_{j\neq i}\left[V^j d_i^j \underline{\mathbf{I}}\right]\Delta\mathbf{U}_j^l - \left[\frac{V^i}{\Delta\tau_{FGMRES}}\underline{\mathbf{I}} + \underline{\mathbf{J}}_{i,2ndO}\right]\Delta\mathbf{U}_i^l = -\mathbf{R}_{DC} \end{aligned} \tag{32}$$

where the spatial and temporal contributions are relegated to separate matrices that can be inverted independently from the other and $\mathbf{R}_{DC}$ is the defect correction residual. This separation, however, is only approximate and introduces a factorization error, as follows:

$$\left[\underline{\mathbf{I}} + \Delta\tau_{BCGS}\underline{\mathbf{D}}_{TS}\right]\left[\frac{V^i}{\Delta\tau_{BCGS}}\underline{\mathbf{I}} + \underline{\mathbf{J}}_{i,1stO}\right] = \left[\frac{V^i}{\Delta\tau_{BCGS}}\underline{\mathbf{I}} + V^j\underline{\mathbf{D}}_{TS} + \underline{\mathbf{J}}_{i,1stO} + \Delta\tau_{BCGS}\underline{\mathbf{J}}_{i,1stO}\underline{\mathbf{D}}_{TS}\right] \tag{33}$$

where the last term on the right hand side is the factorization error, which should be small since it is proportional to $\Delta\tau_{BCGS}$. The approximate-factorization scheme is implemented in two steps; first, the temporal matrix is inverted to find an intermediate value $(\Delta\mathbf{U}_i^{l+1} - \Delta\mathbf{U}_i^l)^*$, as follows:

$$\Delta\Delta\mathbf{U}_i^* = (\Delta\mathbf{U}_i^{l+1} - \Delta\mathbf{U}_i^l)^* = \left[\underline{\mathbf{I}} + \Delta\tau_{BCGS}\underline{\mathbf{D}}_{TS}\right]^{-1}\left[-\mathbf{R}_{DC}\right]. \tag{34}$$

The time-spectral matrix is inverted by making use of a convenient property of spectral discretizations: in frequency space, spectral matrices and their derivatives are strictly diagonal. Thus, the right hand side of the previous equation

is first transformed into frequency space using a fast-Fourier transform (FFT) library and then the update is found as follows:

$$(\widehat{\Delta\Delta\mathbf{U}_i})_k^* = \frac{1}{1+i\omega k\Delta\tau_{BCGS}}\left(-\widehat{\mathbf{R}_{DC}}\right)_k \tag{35}$$

where the i in the denominator is is the imaginary unit $i=\sqrt{-1}$ and k is the wave number. Next, this intermediate state in frequency space is transformed back into physical space using the FFT library. Then, the spatial matrix is inverted and applied to this intermediate state as follows to arrive at the update $(\Delta\mathbf{U}_i^{l+1}-\Delta\mathbf{U}_i^l)$:

$$\Delta\Delta\mathbf{U}_i = (\Delta\mathbf{U}_i^{l+1}-\Delta\mathbf{U}_i^l) = \left[\frac{V^i}{\Delta\tau_{BCGS}}\underline{\mathbf{I}}+\underline{\mathbf{J}}_{i,1stO}\right]^{-1}\left(\Delta\mathbf{U}_i^{l+1}-\Delta\mathbf{U}_i^l\right)^* \tag{36}$$

where the spatial Jacobian is inverted using the BCGS solver described in Algorithm (1), where the matrix inverted is the spatial Jacobian for a single time instance uncoupled from the other time instances. In works by other authors,[13,14,15] the spatial Jacobian is generally inverted before the temporal Jacobian; however, in the present work, it was found that by inverting the temporal Jacobian first, the resulting spatial system is better conditioned for the iterative BCGS algorithm used to invert the spatial Jacobian. For this reason, the inversion order of the AF solver in the present work is reversed compared to these other works. It can be shown that if exact inverses of both temporal and spatial AF Jacobians are assumed, the order in which these matrices are inverted does not affect the resultant solution. Thus ends the presentation of the approximate-factorization preconditioner, whose combination with FGMRES is abbreviated "GMRES-AF". Table (1) summarizes the various solvers and solver/preconditioner combinations presented in the current and past works[11,12,17] and the abbreviations used to describe them.

Table 1. Summary of Solvers and Preconditioners

Linear Solver	Preconditioner	Abbreviation
Time-explicit BCGS	N/A	BCGS-EX
Time-implicit BCGS	N/A	BCGS-IM
FGMRES	Time-explicit BCGS	GMRES-EX
FGMRES	Time-implicit BCGS	GMRES-IM
FGMRES	Def. Corr., BCGS-IM, 1stO Jac.	GMRES-DC1
FGMRES	Def. Corr., BCGS-IM, 2ndO Jac.	GMRES-DC2
FGMRES	Def. Corr., Spatial-Temporal Inv.	GMRES-STI
FGMRES	Def. Corr., Approx. Factorization	GMRES-AF

F. Structural Equations and Solver

The aeroelastic model is based on the response of an airfoil with two degrees of freedom, namely pitch and plunge as shown in Figure (1). The equations of motion for such a system can be summarized as:

$$\begin{aligned} m\ddot{h}+S_\alpha\ddot{\alpha}+K_h h &= -L \\ S_\alpha\ddot{h}+I_\alpha\ddot{\alpha}+K_\alpha\alpha &= M_{ea} \end{aligned} \tag{37}$$

where the variables correspond to the following quantities:

m:	mass of airfoil	S_α:	static imbalance
I_α:	sectional moment of inertia of airfoil	K_h:	plunging spring coefficient
K_α:	pitching spring coefficient	h:	vertical displacement (positive downward)
α:	angle-of-attack	L:	sectional lift of airfoil
M_{ea}:	sectional moment of airfoil about elastic axis		

Non-dimensionalizing time in equations (37) by the uncoupled natural frequency of pitch yields:

$$[M]\ddot{\mathbf{q}}+[K]\mathbf{q}=\mathbf{F} \tag{38}$$

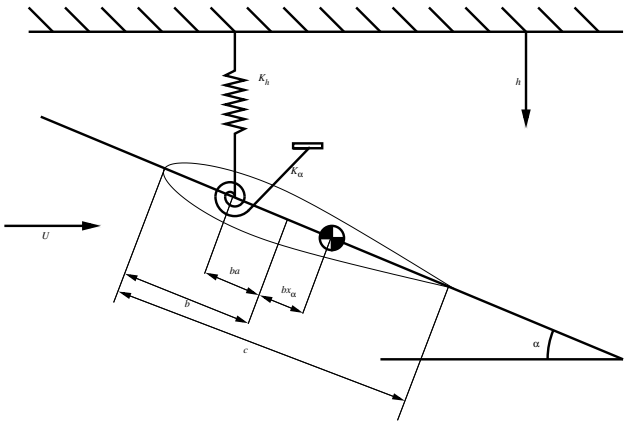


Figure 1. Two degree of freedom 2-D aeroelastic problem schematic

where the non-dimensional mass matrix $[M]$ and stiffness matrix $[K]$ are as follows:

$$[M] = \begin{bmatrix} 1 & x_\alpha \\ x_\alpha & r_\alpha^2 \end{bmatrix}, \quad [K] = \begin{bmatrix} \left(\frac{\omega_h}{\omega_\alpha}\right)^2 & 0 \\ 0 & r_\alpha^2 \end{bmatrix} \tag{39}$$

The corresponding non-dimensional load and displacement vectors are defined by the following relations:

$$\mathbf{F} = \frac{1}{\pi\mu k_c^2} \begin{bmatrix} -C_l \\ 2C_m \end{bmatrix}, \quad \mathbf{q} = \begin{bmatrix} \frac{h}{b} \\ \alpha \end{bmatrix}, \quad \ddot{\mathbf{q}} = \frac{\partial^2 \mathbf{q}}{\partial \check{t}^2} \tag{40}$$

The newly introduced quantities are give in the following table:

b:	semichord of airfoil	k_c:	reduced frequency of pitch, $k_c = \frac{\omega_\alpha c}{2U_\infty}$
μ:	airfoil mass ratio, $\mu = \frac{m}{\pi\rho b^2}$	ω_h:	uncoupled natural frequency of plunge
ω_α:	uncoupled natural frequency of pitch	x_α:	structural parameter defined as $\frac{S_\alpha}{mb}$
r_α^2:	structural parameter defined as $\frac{I_\alpha}{mb}$	$\check{t}$:	structural time, $\check{t} = \omega_\alpha t$
C_l, C_m:	section lift coefficient and section moment coefficient about the elastic axis		

The reduced frequency k_c is typically written in terms of the flutter velocity V_f as follows:

$$k_c = \frac{\omega_\alpha c}{2U_\infty} = \frac{1}{V_f\sqrt{\mu}} \tag{41}$$

where c is the chord length of the airfoil, U_∞ is the free stream velocity, and the flutter velocity V_f is defined as

$$V_f = \frac{U_\infty}{\omega_\alpha b\sqrt{\mu}}. \tag{42}$$

The natural pitch frequency is found by solving equation (42) for ω_α in terms of the prescribed flutter velocity V_f.

1. Transformation of Structural Equations into First-derivative Form:

The aeroelastic equations as shown in Equation (38) are second-derivative partial differential equations. A transformation to first-derivative equations is used in order to facilitate the solution procedure. The transformation used is:

$$\begin{aligned} \mathbf{r_1} &= \mathbf{q} \\ \mathbf{r_2} &= \dot{\mathbf{r}}_\mathbf{1} \end{aligned} \tag{43}$$

The resulting first-derivative equations are then:

$$\begin{aligned} \dot{\breve{\mathbf{r}}}_1 &= \mathbf{r}_2 \\ \dot{\breve{\mathbf{r}}}_2 &= -[M]^{-1}[K]\mathbf{r}_1 + [M]^{-1}\mathbf{F} \end{aligned} \tag{44}$$

and in matrix notation:

$$\begin{bmatrix} \dot{\breve{\mathbf{r}}}_1 \\ \dot{\breve{\mathbf{r}}}_2 \end{bmatrix} + \begin{bmatrix} 0 & -[I] \\ [M]^{-1}[K] & 0 \end{bmatrix} \begin{bmatrix} \mathbf{r}_1 \\ \mathbf{r}_2 \end{bmatrix} = \begin{bmatrix} 0 \\ [M]^{-1}\mathbf{F} \end{bmatrix} \tag{45}$$

$$\dot{\breve{\mathbf{r}}} + [\Psi]\mathbf{r} = \Phi. \tag{46}$$

The matrix $[\Psi]$ is constant and can be precomputed and stored for use during the time integration process. The time derivative term $\dot{\breve{\mathbf{r}}}$ can be discretized using second-order (or first-order) accurate backward difference formulae similar to the time derivative term in the flow equations, as follows:

$$\frac{3\mathbf{r}^{n+1} - 4\mathbf{r}^n + \mathbf{r}^{n-1}}{2\Delta\breve{t}} + [\Psi]\mathbf{r}^{n+1} = \Phi \tag{47}$$

or the time derivative can be discretized using the time-spectral method similarly to equation (15) for purely periodic problems:

$$\sum_{j=0}^{N-1} \breve{d}_n^j \mathbf{r}^j + [\Psi]\mathbf{r}^n = \Phi \quad n = 0, 1, 2, ..., N-1. \tag{48}$$

It should be noted that the time step $\Delta\breve{t}$ appearing in the denominator of the discretized version of the structural equations is different from the time step of the flow equations, and their relation is $\Delta\breve{t} = \omega_\alpha \Delta t$, where Δt is the non-dimensional time step used for the flow equations. Similarly, the period used to calculate the $\breve{d}_n^j$ and $\breve{\phi}$ in time-spectral is the period in the structural time frame. A breve ($\breve{}$) is placed above all quantities that use structural time for clarity. If one wanted to assume that the torsional stiffness were infinite, the torsional equation would need to be removed from the system and the plunge equation non-dimensionalized in a different way.

2. *FGMRES Aeroelastic Solver:*

The fluid and structure equations are coupled together implicitly; thus, for each iteration, fluid and structure residuals are found using either the BDF2 or the TS equations as discussed above. A Jacobian is found for the entire fluid/structure system. This Jacobian not only couples the fluid residuals to the flow variables and the structural residuals to the structural variables, but also the fluid residuals to the structural variables and the structural residuals to the flow variables. Starting with the flow Jacobian (of a single time instance in the case of the time-spectral method), the coupled Jacobian will have as many additional rows and columns as there are structural variables. Thus, a flow Jacobian that is of size $[[4 \times N_c] \times [4 \times N_c]]$ where N_c is the number of elements or cells in the computational domain of the flow, would become a $[[4 \times N_c + N_s] \times [4 \times N_c + N_s]]$ matrix for the implicitly coupled fluid/structure system, where N_s is the number of structural variables.

While the above represents the size of the Jacobian if all entries were stored, it should be readily apparent that a matrix of this size could not be stored on modern computers for the sizes of computational meshes that are typically used. Instead, only the non-zero blocks of the Jacobian are stored. The first-order Jacobian is stored as an array of its diagonal blocks, which is $[4 \times 4 \times N_c]$, and an array of its off-diagonal blocks, which are stored on the faces and is $[2 \times 4 \times 4 \times N_f]$ where the 2 represents the opposite sides of each face and N_f is the number of faces. The full second-order Jacobian is not stored explicitly; rather, the evaluation of exact Jacobian-vector products is enabled as required at each linear solver iteration by storing three sets of diagonal and off-diagonal blocks as described by equation (22), which are then used to assemble Jacobian-vector products.

The changes in the structural residuals with respect to the flow variables occur entirely as a function of the lift and moment coefficient variations with the flow variables, thus:

$$\frac{\partial \mathbf{R_s}}{\partial \mathbf{U}} = f\left(\frac{\partial C_L}{\partial \mathbf{U}}, \frac{\partial C_m}{\partial \mathbf{U}}\right). \tag{49}$$

This portion of the Jacobian $\frac{\partial \mathbf{R_s}}{\partial \mathbf{U}}$, above, only has non-zero values in cells which include a face that lies on the surface of the airfoil and only these non-zero values are stored.

The changes in the flow residuals with respect to the structural variables are a function only of the change in the grid motion in response to variations in the structural variables. That is, the structural variables influence both the grid speeds and the orientation of the face normal vectors:

$$\frac{\partial \mathbf{R_f}}{\partial \mathbf{r}} = f\left(\frac{\partial \dot{\mathbf{x}}}{\partial \mathbf{r}}, \frac{\partial \tilde{\mathbf{n}}}{\partial \mathbf{r}}\right). \tag{50}$$

In the present work, mesh motion is achieved by translating and rotating the mesh rigidly. Thus, every face in the mesh changes speed and orientation when the mesh is rotated and translated. Because every face is affected, $\frac{\partial \mathbf{R_f}}{\partial \mathbf{r}}$ has the same size as the diagonal blocks of the first-order Jacobian, $[4 \times 4 \times N_c]$. For three-dimensional, fixed-wing or deforming-blade aeroelastic analysis, where rigid mesh motion is not feasible, a deforming mesh capability would be required. This deforming mesh motion will add additional dependencies on cell volume and face area in the above portion of the coupled Jacobian. Additionally, since all face-normals change orientation when a rigid mesh rotates, the above component of the Jacobian has non-zero values for all cells. This would not necessarily be the case for a deforming mesh. It should also be noted that an additional dependence of the structural residuals on the structural variables is included in this framework. This dependence arises as a result of the grid motion since the coefficient of lift is itself dependent on the orientation of the face normals on the airfoil surface:

$$\frac{\partial \mathbf{R_s}}{\partial \mathbf{r}} = f\left([\Psi], \frac{\partial C_L}{\partial \tilde{\mathbf{n}}} \frac{\partial \tilde{\mathbf{n}}}{\partial \mathbf{r}}\right). \tag{51}$$

This final piece of the full fluid/structure Jacobian retains its $[4 \times 4]$ size.

The fully-coupled linear system described in the preceding paragraphs and equations is solved for each non-linear iteration using the GMRES-AF solver, as described above, but with the following modifications. First, both the first-order Jacobian used in the BCGS preconditioner and the second-order Jacobian used in both the defect-correction step and in the FGMRES algorithm contain the full fluid/structure coupling given by $\partial \mathbf{R_f}/\partial \mathbf{U}$, $\partial \mathbf{R_s}/\partial \mathbf{U}$, $\partial \mathbf{R_f}/\partial \mathbf{r}$, and $\partial \mathbf{R_s}/\partial \mathbf{r}$ as described above. These additions are included in all levels of preconditioning. It has been found that for more difficult aeroelastic problems, convergence can stagnate if structural coupling is neglected from the preconditioner, as had been done in previous work.[22] By including the fluid/structure coupling in the preconditioner, this problem has been eliminated.

Finally, just as is the case with the flow Jacobian, a pseudo-time term is added to the diagonal-block of the structural contribution to the Jacobian. This pseudo-time term has the same form as that of the flow, given by equation (17); however, the value for the spectral radius $\| \lambda \|$ that couples the fluid and structure together is not found analytically. Through trial and error, a value of $\| \lambda \|$ that maximizes convergence efficiency is used.

G. Implementation

The various time instances in the time-spectral approach are coupled and must be solved simultaneously. This coupling can be implemented serially, whereby a single time-instance is solved at any given moment, and then transmits its update to the next time instance, which is then solved and the process repeated sequentially until all time instances have been updated. However, since the coupling occurs as a source term in the residual, each individual time instance may be solved in parallel with the other time instances. This introduces an additional dimension for achieving parallelism compared to time-implicit computations, where progress in the time dimension is necessarily sequential. In this implementation, two levels of parallelism are introduced, the first in the spatial dimension, and the second in the time dimension where the various time instances are solved by spawning multiple instances of the spatial solver on a parallel computing cluster. The implementation uses MPI for parallelism in the time dimension. This additional parallelism in time may prove to be particularly enabling with the advent of rapidly expanding massively parallel computing clusters, particularly for cases where parallelism in the spatial dimension has been exhausted (perhaps due to the adequacy of moderate grid sizes).

One of the drawbacks of the chosen time-parallelism strategy for time-spectral methods is that each time instance must broadcast its entire solution field to all other time instances, which can result in a significant amount of communication. Various strategies for communicating the different time instances to all processors have been investigated. Currently, a round-robin approach is implemented, where each processor sends its time instance to a single neighboring processor. The received time instance is added to the time derivative source term on the local processor, and then passed on to the next processor. By repeating this procedure $N-1$ times, where N is the number or time instances,

the complete time derivative involving summations from all time instances is accumulated without the requirement of creating a local temporary copy of all the additional time-instance solution vectors or performing any communication intensive broadcast operations.

The aforementioned communication strategy works in all cases except the spatial-temporal diagonal-block inversion step of the GMRES-STI solver and the temporal-inversion part of the GMRES-AF solver, where it is necessary not just to include the contributions of other time instances in the residual, but in the spatial-temporal-diagonal and approximate-temporal blocks of the Jacobian as well. As such for GMRES-STI, after the Jacobian has been calculated during each non-linear iteration and before the GMRES linear solve begins, the $[4 \times 4]$ diagonal blocks from all time instances of the two-dimensional spatial Jacobian for a single grid cell are all transmitted to a selected MPI process on which the spatial-temporal diagonal block, of size $[(4 \times N) \times (4 \times N)]$ is assembled and inverted. The spatial-cells are distributed as evenly as possible among the N computational cores available, one core each for the N time instances. How evenly the cells can be distributed among the N cores, however, is complicated by the use of the BCGS preconditioner, which necessitates that each cell color, in fact, be split as evenly as possible. Evenly splitting each color is more difficult than evenly splitting the entire grid because, since a typical grid composed of triangles and quadrilaterals is split into four colors, the actual groups being split are only around one-fourth the grid size. A similar process is used to reduce the pseudo-time step for each cell (since local pseudo-time stepping is used) to find its minimum value across all time instances for the approximate-temporal matrices in the GMRES-AF approach.

In addition to the transmission of the spatial-temporal diagonal blocks among the cores during each non-linear iteration, the BCGS residual vector must also be transmitted to its corresponding MPI process for each BCGS preconditioning iteration when using the GMRES-STI solver. Once all the residual vectors are assembled on the correct processes, the BCGS solution vector is found from this residual vector and the spatial-temporal diagonal block, which was LU-decomposed earlier, at the start of the current non-linear iteration. Then, the BCGS solution vector is divided and each piece returned to the MPI process on which its time instance usually resides. As can be seen, the GMRES-STI solver has more than twice as much communication per preconditioner iteration as the GMRES-DC1 and GMRES-DC2 solvers. Additionally, it requires the inversion of a $[(4 \times N) \times (4 \times N)]$ matrix for every 2D spatial grid cell during each non-linear iteration. It turns out this matrix inversion step becomes the most time consuming step in the GMRES-STI solver for large numbers of time instances.

On the other hand, for the GMRES-AF solver, the inverted approximate-temporal block is only used once for each call of the approximate-factorization preconditioner, instead of for every BCGS iteration. Additionally, since the time instances are decoupled after the inverted approximate-temporal block has been applied, no further communication occurs among the time instances for the current iteration of the approximate-factorization preconditioner. As such, the GMRES-AF scheme has less communication per preconditioner iteration than all other preconditioners except GMRES-EX.

III. Computational Results

A. Flow-only results

A two-dimensional inviscid flow test case is constructed with the forced pitching oscillation of a NACA-0012 airfoil at a Mach number of 0.755 and a mean incidence $\alpha_0 = 0.016$ degrees. The pitching motion is prescribed about the quarter chord of the airfoil as follows:

$$\alpha(t) = \alpha_0 + \alpha_A \sin(\omega t). \tag{52}$$

The reduced frequency k_c is equal to 0.0814 and the pitching amplitude α_A is equal to 2.51 degrees. This test case corresponds to the AGARD test case No. 5.[23] Three different unstructured meshes are utilized in this paper: a coarse mesh consisting of 4,471 nodes and 8,747 cells, a medium mesh of 16,181 nodes and 31,992 cells, and a fine mesh of 60,826 nodes and 120,938 cells. Figures (2)-(4) show the near field of these three computational meshes.

Figure (5a) shows the comparison of the lift coefficient versus non-dimensional time between a reference solution obtained using a second-order accurate time-implicit solver with $\Delta t = T/4096$ (where T denotes the period of airfoil motion) and the time-spectral method with $N = 3, 7, 15$, and 47 time instances on the coarse mesh. For this case, the lift computed by the time-spectral method with even 1 harmonic or 3 time instances shows reasonable agreement with the reference solution.

Figure (5b) shows the comparison of the moment coefficient versus non-dimensional time for the same reference and time-spectral solutions. The moment history contains multiple harmonics and thus is not captured accurately with $N = 3$ in the TS method; in fact $N = 47$ time instances are needed to produce near-exact agreement. Figure (5c) shows the comparison of the drag coefficient versus non-dimensional time for those same solutions. It should be noted that,

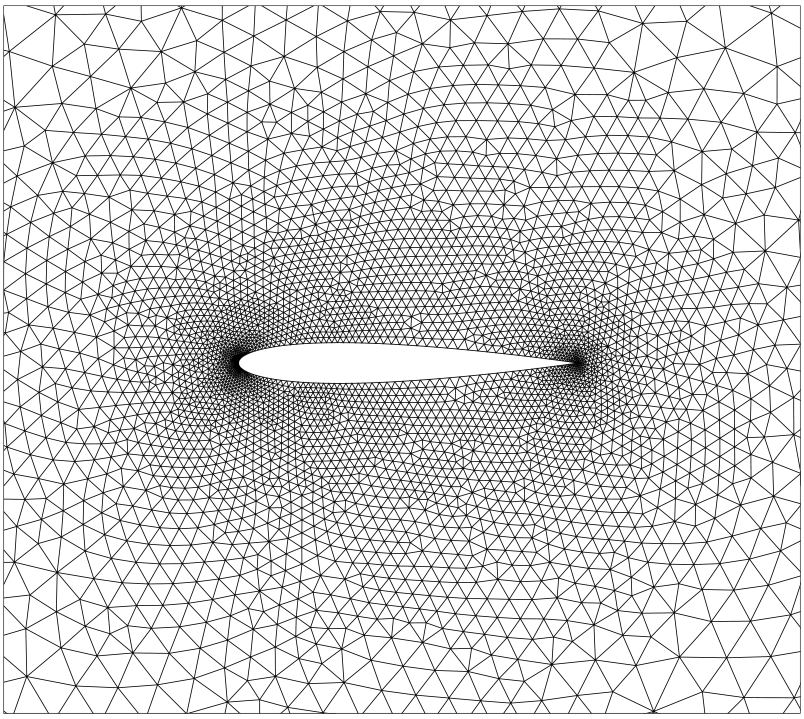


Figure 2. Near field on the coarse mesh for the NACA-0012 airfoil

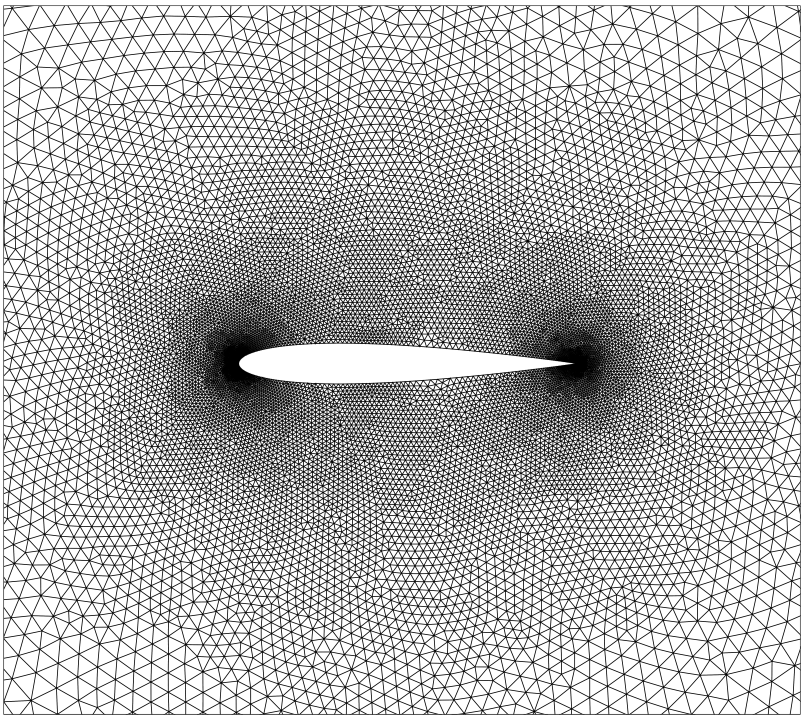


Figure 3. Near field on the medium mesh for the NACA-0012 airfoil

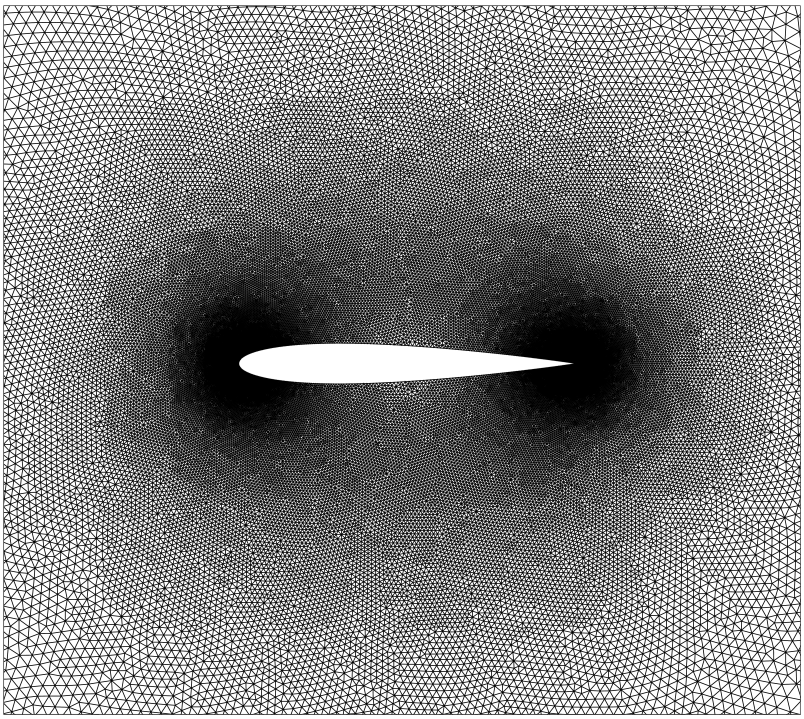


Figure 4. Near field on the fine mesh for the NACA-0012 airfoil

since the results use the Euler equations, the drag shown is pressure drag and does not contain viscous effects. The drag coefficient shows good agreement for the $N = 47$ solution. The most difficult areas of the moment and pressure drag curves to resolve occur around $t = 7$s and $t = 28$s, where the $N = 47$ and reference solutions show a slight reversal. It can be seen that the results of the TS method converge to the reference solution as the number of time instances increases.

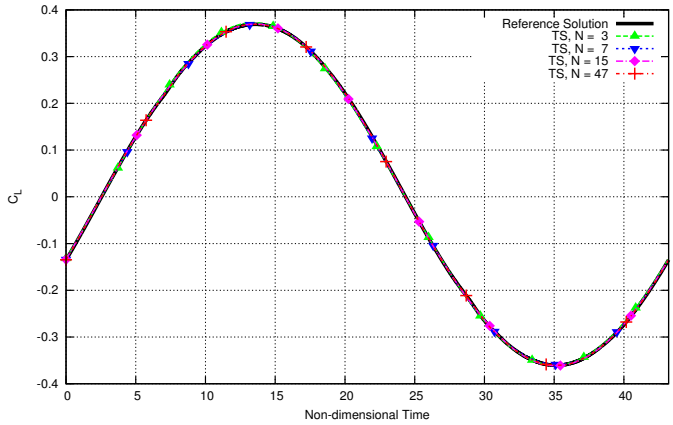
1. Coarse-mesh efficiency comparison

As mentioned above, by inverting the temporally-coupled diagonal blocks of the Jacobian, instead of just the uncoupled, spatial diagonal blocks, as had been done previously in the BCGS preconditioner,[11] the temporal restrictions on the CFL number can be eliminated. This results in solution convergence that, in terms of non-linear iterations, Krylov vectors, and preconditioner iterations, is independent of the number of time instances used as well as the reduced frequency of the motion. All of the wall-clock times presented in this section are for solutions utilizing one computational core per time instance.

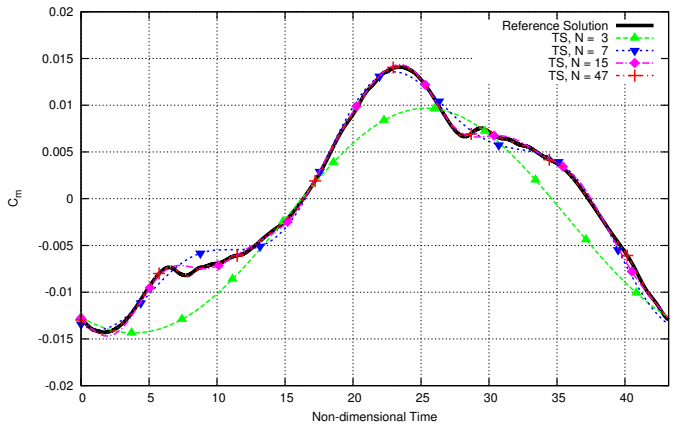
The AGARD test case No. 5 is examined using the coarse computational mesh in Figure (2). Figure (6) shows the convergence of this test case for the GMRES-STI solver by plotting the non-linear residual versus the number of non-linear iterations, cumulative preconditioning iterations, cumulative Krylov vectors, and wall-clock time per time instance, in that order for the four subfigures. Table (2) summarizes this same data for the converged solutions, except it includes gross wall-clock time instead of wall-clock time per time instance. It should be noted that the solutions in this section have all been converged to a convergence tolerance of 3×10^{-12} and all use the same GMRES CFL number growth strategy which allows the CFL number to reach infinity before convergence is attained

As can be seen in this figure and table, solutions for all numbers of time instances use about the same number of non-linear iterations, cumulative preconditioner iterations, and cumulative Krylov vectors. However, the wall-clock time used increases with increasing numbers of time instances. This wall-clock time trend results from $O(N^3)$ expense of the direct inversion of the spatial-temporal diagonal blocks. That being said, the wave-number independence of this preconditioner is clearly demonstrated by the near constant number of BCGS iterations used regardless of the number of time instances.

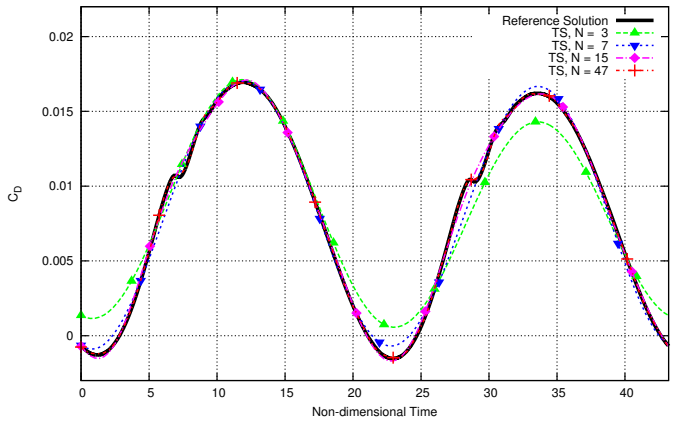
When the GMRES-AF solver is applied to this same problem with the same numbers and more time instances, as shown in Figure (7) and Table (3), all numbers of time instances use about the same number of preconditioner



(a)



(b)



(c)

Figure 5. Comparison of computed lift coefficient (a), moment coefficient (b), and pressure drag coefficient (c) using the time-spectral method to a reference, time-implicit solution for the AGARD5 test case on the coarse mesh

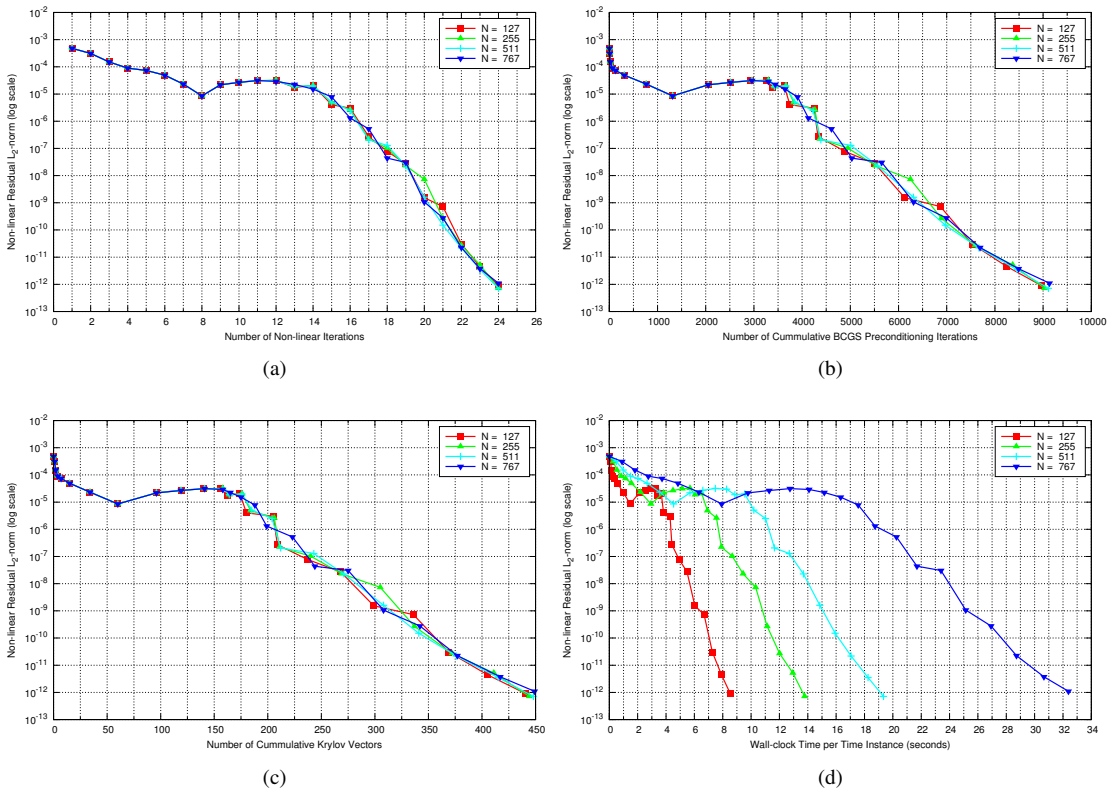


Figure 6. Comparison of convergence for the AGARD test case No. 5 with the indicated numbers of time instances using the GMRES-STI solver on the coarse grid with Non-linear Iterations (a), BCGS Iterations (b), and Krylov vectors (c), and wall-clock time per time instance (d) on the x-axis

Table 2. Convergence of the AGARD 5 Test Case using the GMRES-STI Solver on the Coarse Mesh

Num. Time Inst.	Non-linear Iter.	BCGS Iter.	Krylov Vectors	Wall-clock Time (s)	BCGS CFL
15	24	8,905	437	137	2.5
31	24	8,850	435	245	2.5
63	24	8,951	440	458	2.5
127	24	8,970	441	1,085	2.5
255	24	9,029	444	3,514	2.5
511	24	9,109	448	9,881	2.5
767	24	9,131	449	24,840	2.5

iterations and Krylov vectors, in addition to non-linear iterations. This result is because, like with the GMRES-STI solver, the pseudo-time step size used by the GMRES-AF solver is independent of the number of time instances and is fully determined by the spectral radius of the spatial discretization. It should be noted that a maximum of ten defect-correction steps is allowed for each Krylov vector for all numbers of time instances. Wall-clock time, on the other hand, still increases for increasing numbers of time instances, but the wall-clock time per time instance is nearly constant between 95 and 511 time instances. Additionally, the wall-clock time used is lower than that used by the GMRES-STI solver for all numbers of time instances considered, and it also increases less rapidly when time instances are added with the GMRES-AF solver as compared to the GMRES-STI solver.

Table 3. Convergence of the AGARD 5 Test Case using the GMRES-AF Solver on the Coarse Mesh

Num. Time Inst.	Non-linear Iter.	BCGS Iter.	Krylov Vectors	Wall-clock Time (s)	BCGS CFL
15	24	9,138	384	84.4	3.0
31	24	9,059	380	107	3.0
63	24	9,515	400	181	3.0
127	23	8,915	374	291	3.0
255	24	9,563	401	575	3.0
511	24	9,659	405	1,151	3.0
767	24	9,707	407	1,752	3.0
1,023	24	9,659	405	3,024	3.0
1,535	24	9,635	404	5,058	3.0
2,047	24	9,467	397	7,786	3.0

To verify these same trends hold for higher wave-numbers, the same test case is run except at twice the reduced frequency of the AGARD 5 test case, i.e. $k_c = 0.1628$ for the second test case, with all other test-case parameters remaining unchanged. Tables (4) and (5) convey the convergence data for this test case using the GMRES-STI and GMRES-AF solvers, respectively. As can be seen, the same trends are apparent at this higher reduced frequency.

Table 4. Convergence of the $2k_c$ Test Case using the GMRES-STI Solver on the Coarse Mesh

Num. Time Inst.	Non-linear Iter.	BCGS Iter.	Krylov Vectors	Wall-clock Time (s)	BCGS CFL
15	21	7,528	369	121	2.5
31	21	7,784	382	218	2.5
63	21	8,064	396	419	2.5
127	21	7,945	390	964	2.5
255	21	7,864	386	3,118	2.5
511	21	7,864	386	8,618	2.5
767	21	7,824	384	19,794	2.5

To further examine how the cumulative preconditioner iterations and wall-clock time needed to converge a solution change in response to reduced frequency and number of time instances for the two solvers under consideration in the present work, Figures (8) and (9) plot the number of preconditioning iterations and wall-clock time per time instance versus the number of time instances for four reduced frequencies, from the AGARD5 test case frequency to eight times that frequency. Figure (8) supports the conclusion that the GMRES-STI solver is wave-number independent at all frequencies. While the spatial-diagonal blocks are inverted only once per non-linear iteration, they are inverted directly, a process that uses $O\left(N^3\right)$ operations where N is the number of time instances. Given that the number of computational cores used scales as $O\left(N\right)$, i.e. one core per time instance, an increase in the number of time instances results in $O\left(N^2\right)$ work per computational core during the diagonal-block inversion step, assuming perfect scaling. This scaling is the primary reason why the wall-clock time to convergence per time instance exhibits an increasing slope with increasing numbers of time instances, as shown in Figure (9). To prove that wall-clock time scales like

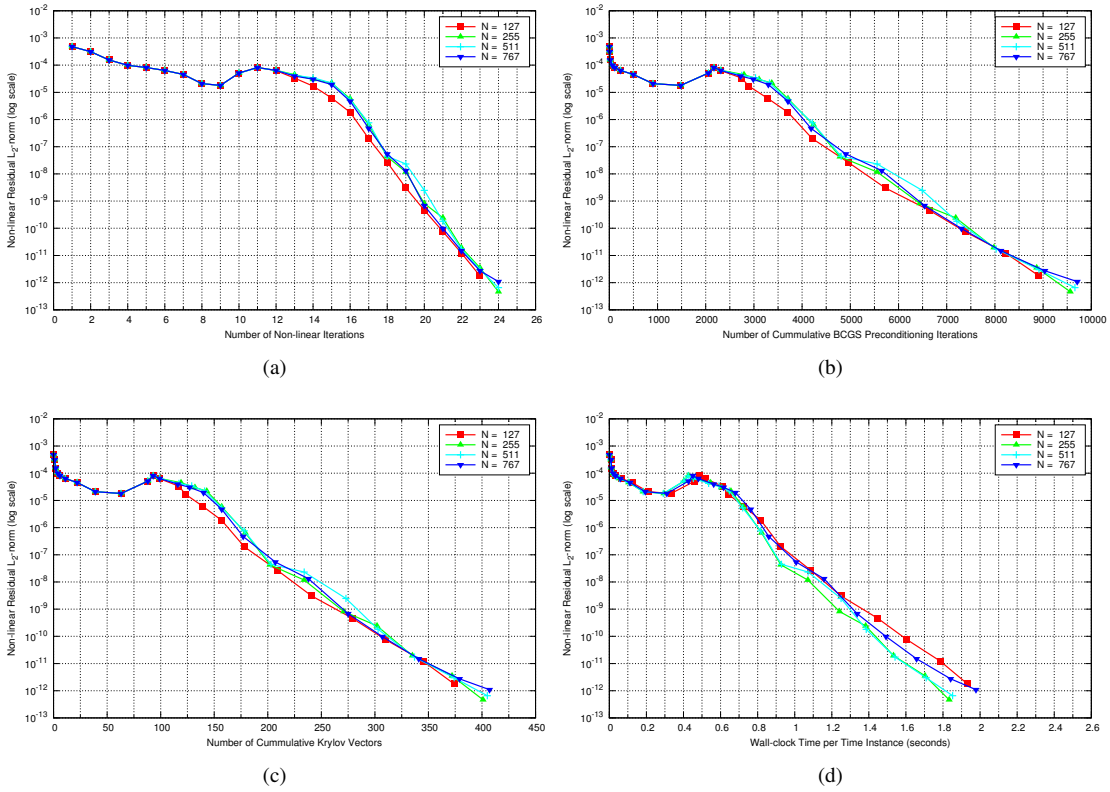


Figure 7. Comparison of convergence for the AGARD test case No. 5 with the indicated numbers of time instances using the GMRES-AF solver on the coarse grid with Non-linear Iterations (a), BCGS Iterations (b), and Krylov vectors (c), and wall-clock time per time instance (d) on the x-axis

Table 5. Convergence of the $2k_c$ Test Case using the GMRES-AF Solver on the Coarse Mesh

Num. Time Inst.	Non-linear Iter.	BCGS Iter.	Krylov Vectors	Wall-clock Time (s)	BCGS CFL
15	23	8,355	350	77.5	3.0
31	24	9,103	381	115	3.0
63	24	9,098	381	157	3.0
127	24	8,935	374	262	3.0
255	24	9,127	382	518	3.0
511	24	9,174	384	1,118	3.0
767	24	8,959	388	1,710	3.0
1,023	24	9,006	377	2,521	3.0
1,535	24	8,839	370	4,606	3.0
2,047	24	8,767	367	8,662	3.0

$O\left(N^3\right)$ (or $O\left(N^2\right)$ per computational core), Figure (10) plots the wall-clock time per the number of time instances squared versus the number of time instances. As can be seen, once a sufficient number of time instances are used, the spatial solution overhead is overcome, and the amount of wall-clock time per time instance squared is constant.

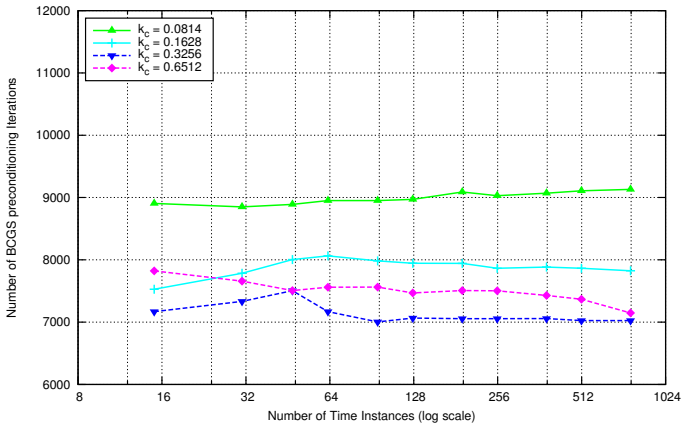


Figure 8. BCGS preconditioner iterations needed for converged solutions on the coarse grid versus number of time instances for four reduced frequencies – from the AGARD5 frequency to eight times the AGARD5 frequency, doubling the frequency each time– using GMRES-STI solver

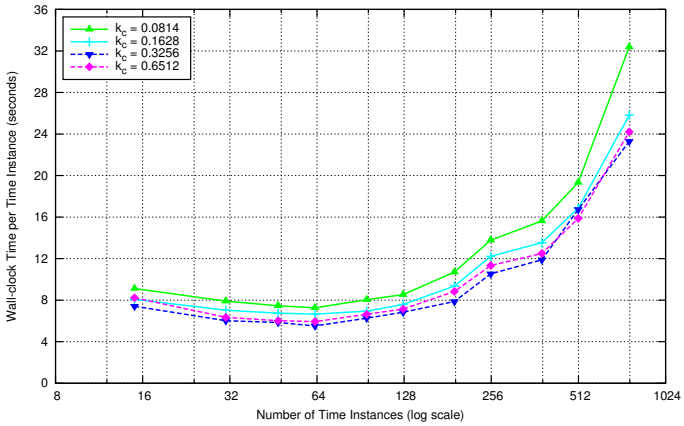


Figure 9. Wall-clock time per time instance needed for converged solutions on the coarse grid versus number of time instances for four reduced frequencies – from the AGARD5 frequency to eight times the AGARD5 frequency, doubling the frequency each time– using GMRES-STI solver

Figures (11) and (12) show analogous trends for the GMRES-AF solver. As can be seen, like the GMRES-STI, the GMRES-AF solver exhibits wave-number independence since the number of preconditioning iterations is roughly constant regardless of the number of time instances or the reduced frequency. The wall-clock time per time instance is roughly constant between 95 and 511 time instances at all reduced frequencies meaning that the solution computational time scales as $O(N^2)$, or $O(N)$ per computational core. However, this trend breaks down for more than 511 time instances because there are essentially too many cores available to be used efficiently by the preconditioner given the size of the spatial grid. As noted earlier, for the GMRES-AF preconditioning step, the spatial grid is divided roughly evenly among the available cores, meaning that when 1023 time instances and cores are utilized, there are on average between eight and nine cells per core. Similarly, when 2047 cores are used, there are between four and five cells per cores. Obviously, this is too few and explains the increase in wall-clock time as the number of time instances is increased beyond 511. Despite this scaling, the GMRES-AF solver is over an order of magnitude faster than the GMRES-STI solver for large numbers of time instances regardless of the reduced frequency of the problem being solved.

The order $O(N^2)$ computational time scaling of the GMRES-AF solver mentioned previously is thought to be primarily a function of the communication time and this hypothesis is now examined further. Figure (13) plots the amount of wall-clock time per time instance used by the different parts of the code that are dependent on the number

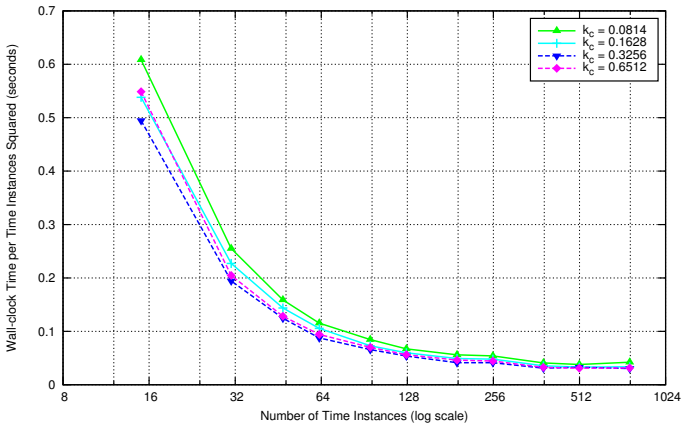


Figure 10. Wall-clock time per the square of the number of time instances needed for converged solutions on the coarse grid versus number of time instances for four reduced frequencies – from the AGARD5 frequency to eight times the AGARD5 frequency, doubling the frequency each time– using GMRES-STI solver

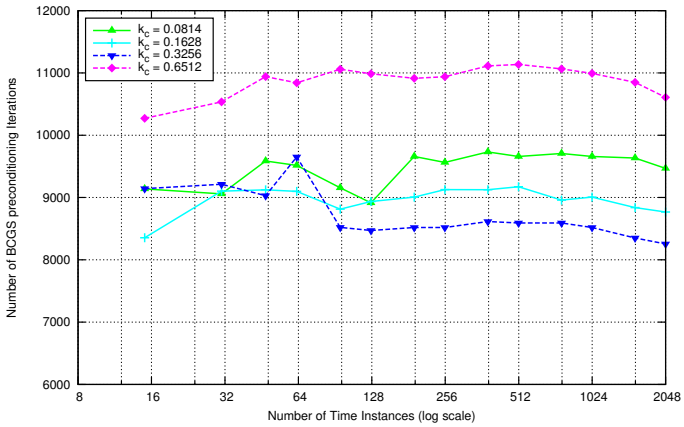


Figure 11. BCGS preconditioner iterations needed for converged solutions on the coarse grid versus number of time instances for four reduced frequencies – from the AGARD5 frequency to eight times the AGARD5 frequency, doubling the frequency each time– using GMRES-AF solver

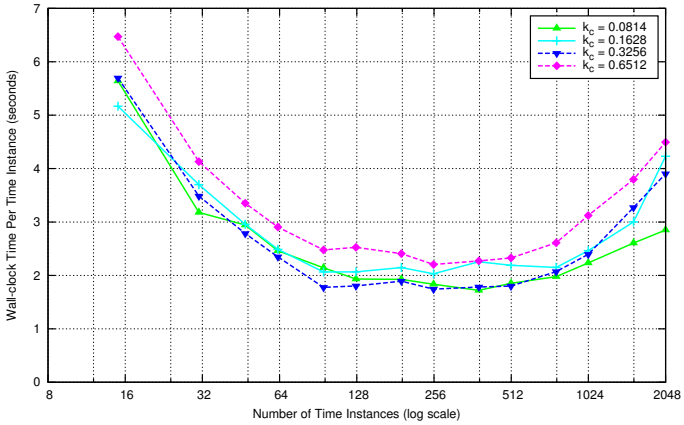


Figure 12. Wall-clock time per time instance needed for converged solutions on the coarse grid versus number of time instances for four reduced frequencies – from the AGARD5 frequency to eight times the AGARD5 frequency, doubling the frequency each time– using GMRES-AF solver

of time instances – the MPI communication time, the calculation of the time-spectral contribution to the residual, and the fast-Fourier transforms (FFTs) used in the preconditioner – for the AGARD5 test case. A fourth series is included, which corresponds to time-spectral independent parts of the calculation, i.e. the parts that provide the solution of the spatial discretization.

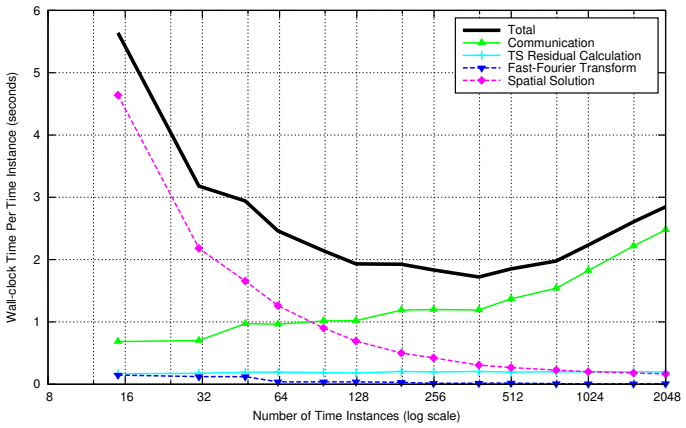


Figure 13. Wall-clock time per time instance needed for converged solutions on the coarse grid versus number of time instances using GMRES-AF solver with curves for the total time, communication time, TS residual calculation time, FFT time, and non-TS operations time.

As can be seen, for all numbers of time instances, either the communication time or the spatial computational time dominates the total wall-clock time needed for convergence. For small numbers of time instances, the spatial calculation time dominates, but as the number of time instances increases, the spatial time becomes smaller while the communication time increases. The other two parts that depend on the number of time instances – the TS residual calculation time and the TS-FFT time per time instance remain roughly constant and small compared to the other parts of the calculation. Perhaps a better way to consider the effects of the different computations on the overall wall-clock time used is as a percent of that total. Figure (14) plots the percent of the total wall-clock time used by MPI communication, TS residual calculations, preconditioner FFT calculations, and the spatial solution.

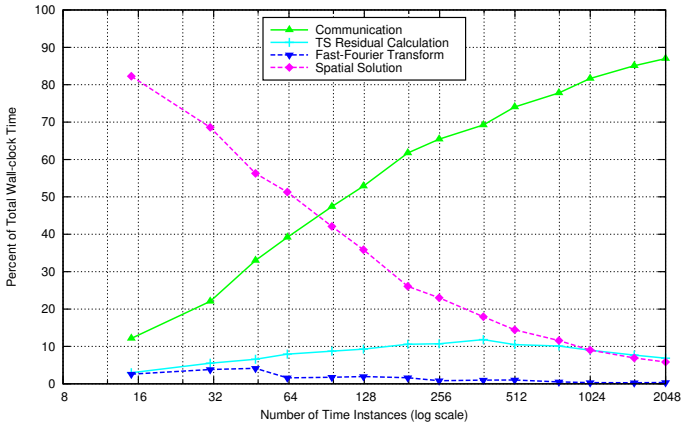


Figure 14. Percentage of the total wall-clock time needed for converged solutions on the coarse grid versus number of time instances using GMRES-AF solver with curves for the communication time, TS residual calculation time, FFT time, and non-TS operations time.

As can be seen, the communication takes about 12% of the time for 15 time instances and close to 88% for 2047 time instances while the spatial calculation takes around 82% of the time for 15 time instances and less than 7% for 2047 time instances. The TS residual calculation trend does not increase or decrease monotonically, but rather varies between about 3% at 15 time instances and 12% at 383 time instances, while the FFT time is less than 5% for all numbers of time instances considered. It appears, at least for flow alone results on the coarsest mesh, that as the number of time instances increases, the communication time tends to dwarf all other parts of the code in terms of the wall-clock time consumed.

2. *Medium-mesh efficiency comparison*

To confirm that the above results are not only true on the coarse mesh, the same series of AGARD5 based test cases – at four reduced frequencies from the AGARD5 frequency on the low side to eight times that frequency on the high side – is rerun on the medium resolution mesh. Figure (15) plots the wall-clock time per time instance for converged solutions on this medium grid. As can be seen, first, the medium-grid solutions take about ten times the wall-clock time as the same solutions on the coarse grid for solutions using between 15 and 767 time instances, while this multiplier is not as high for cases using more time instances, falling to between five times to seven times, depending on the reduced frequency, for 2047 time instances. The relative decrease in the multiplier compared to the coarse grid for higher numbers of time instances is because, for a grid with more cells, there are more cells per processor at these higher numbers of time instances. Specifically, since the medium mesh is between seven and eight times the size of the coarse mesh, there are seven to eight times as many cells per computational core.

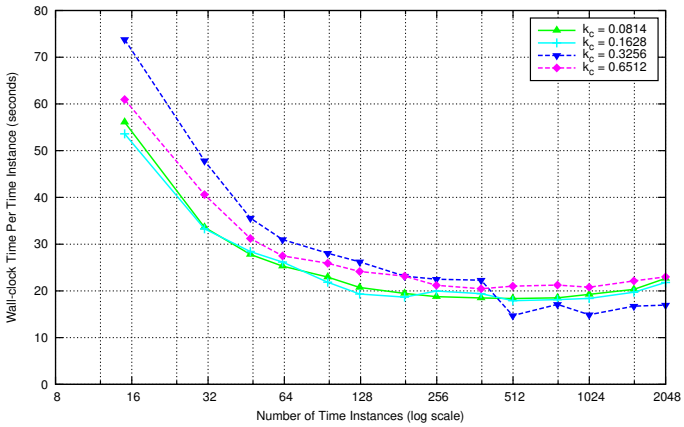


Figure 15. Wall-clock time per time instance needed for converged solutions on the medium grid versus number of time instances for four reduced frequencies – from the AGARD5 frequency to eight times the AGARD5 frequency, doubling the frequency each time– using GMRES-AF solver

Figures (16) and (17) plot the wall-clock time per time instance and the percentage of the total wall-clock time used by the different parts of the code. The same general trends are observed as for the coarse grid, but with a few notable, specific differences. Again, the spatial contributions are the most time-intensive contribution for low numbers of time instances and communication is most time-intensive for large numbers of time instances. However, the TS residual calculation uses a much larger share of the time than previously at about 9% for 15 time instances up to a little more than 30% for 767 time instances. This result makes sense as the TS residual calculation time is a function of the number of cells in the mesh multiplied by the number of time instances. Overall, the results for the medium

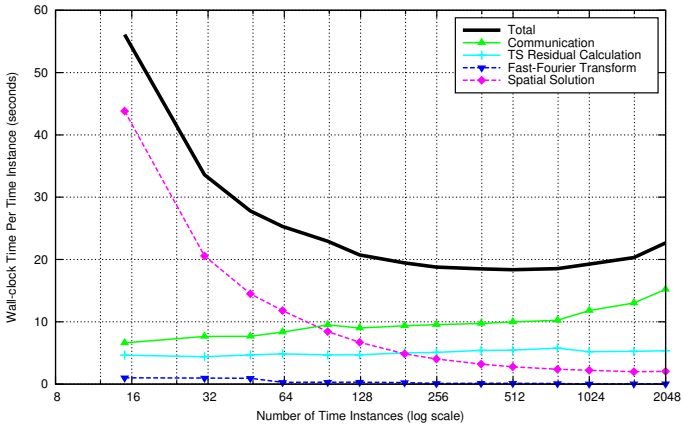


Figure 16. Wall-clock time per time instance needed for converged solutions on the medium grid versus number of time instances using GMRES-AF solver with curves for the total time, communication time, TS residual calculation time, FFT time, and non-TS operations time.

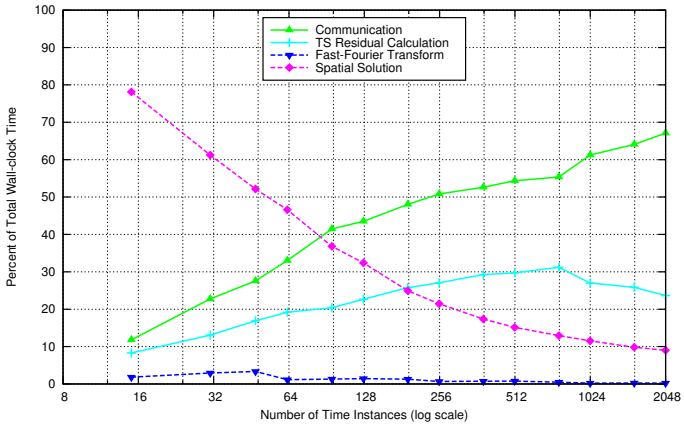


Figure 17. Percentage of the total wall-clock time needed for converged solutions on the medium grid versus number of time instances using GMRES-AF solver with curves for the communication time, TS residual calculation time, FFT time, and non-TS operations time.

mesh reinforce the trends observed for the coarse mesh: the two main sources of inefficiency are the spatial BCGS preconditioner and the MPI communication.

3. Fine-mesh results

Finally, for flow alone, the four reduced frequency cases are again rerun on the fine grid shown previously. Figure (18) plots the wall-clock time per time instance for solutions on the fine grid. As can be seen, the same trends observed

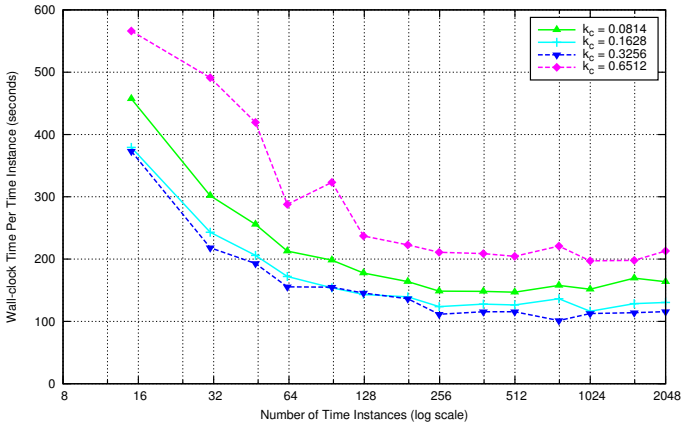


Figure 18. Wall-clock time per time instance needed for converged solutions on the fine grid versus number of time instances for four reduced frequencies – from the AGARD5 frequency to eight times the AGARD5 frequency, doubling the frequency each time– using GMRES-AF solver

previously hold, where, overall the computational time asymptotically scales as $O(N^2)$, the total wall-clock time scales as $O(N)$, and the wall-clock time per time instance scales as $O(1)$. It can be observed that the $k_c = 0.6512$ or eight times the base reduced frequency case uses about 25% more wall-clock time than the next closest case. This is because this particular case on this particular mesh proved especially difficult to converge so a much less aggressive CFL growth strategy had to be utilized to guarantee convergence. Otherwise, as noted, the trends are as expected.

B. Aeroelastic results

To examine the convergence of the aeroelastic solver, the AGARD5 test case presented previously is modified so that it responds aeroelastically. The same NACA-0012 unstructured, computational meshes are used. The free stream Mach number of 0.755 and mean incidence α_0 of 0.016 degrees are retained. The airfoil again undergoes a forced pitching oscillation of $\alpha_A = 2.51^o$, however, this oscillation occurs about the airfoil elastic axis, instead of the quarter-chord as

was true for the AGARD5 test case. In addition to the forced pitching, the airfoil is allowed to respond aeroelastically in both pitch and plunge such that the true pitch angle of the airfoil at a given time is the combination of the prescribed and aeroelastic contributions to pitch. The reduced frequency of the forced pitching is $k_c = 0.0814$ just as in the AGARD5 test case. The structural parameters for this and all other aeroelastic cases considered are as follows:

$$V_f = 0.50, \quad x_\alpha = 1.8, \quad r_\alpha^2 = 3.48, \quad \frac{\omega_h}{\omega_\alpha} = 1.0, \quad \mu = 64.0, \quad a = -2.0.$$

The quantity a is the non-dimensional elastic axis location along the chord of the airfoil measured from the mid-chord of the airfoil when it is in the neutral position. Since it is non-dimensionalized by the semi-chord of the airfoil, the elastic axis is located half a chord length ahead of the leading edge of the airfoil in this particular case. This position is also the axis about which the forced pitching occurs.

Figure (19) compares the lift, moment, and pressure drag coefficients of a single period of motion of this test case for different numbers of time-spectral time instances to a reference time-implicit solution with 4,096 time steps per period. It appears that all the time-spectral solutions show good agreement in lift and moment, while only the 63 time instance solution shows visually exact agreement in pressure drag. It is observed that the time-spectral solution converges to the reference solution as the number of time instances increases for the aeroelastic case as well. For all aeroelastic results, only the GMRES-AF solver is utilized.

1. *Coarse-mesh efficiency comparison*

Table (6) and Figure (20) show the convergence of the AGARD5 aeroelastic test case on the coarse grid using the GMRES-AF solver. No comparison is made to the GMRES-STI solver as it is clearly much less efficient.

Table 6. Convergence of the AGARD 5 Aeroelastic Test Case using the GMRES-AF Solver

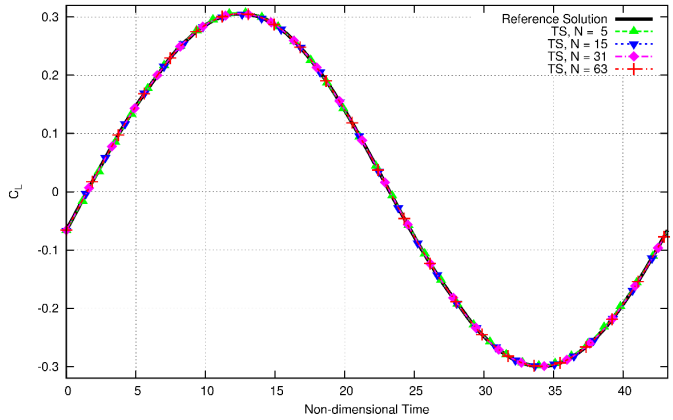
Num. Time Inst.	Non-linear Iter.	BCGS Iter.	Krylov Vectors	Wall-clock Time (s)	BCGS CFL
15	20	9,852	466	115	3.0
31	21	11,582	549	157	3.0
63	21	12,202	577	229	3.0
127	21	11,921	562	346	3.0
255	21	11,974	564	615	3.0
511	21	12,416	563	1,338	3.0
767	21	12,403	574	2,153	3.0
1,023	21	12,316	573	3,277	3.0
1,535	21	11,898	552	5,478	3.0
2,047	21	11,793	548	7,995	3.0

As can be seen, all the trends are the same as for the flow-alone case except the aeroelastic case uses about 20% more preconditioner iterations, Krylov vectors, and wall-clock time. This increase is expected as the aeroelastic problem should be more difficult to solve. Figures (21) and (22) plot the number of BCGS iterations and the wall-clock time for converged solutions on the coarse grid for the four reduced frequencies considered previously. As can be seen, the number of BCGS iterations ranges between 9,852 and 15,007 for these cases with the twice-reduced-frequency case proving more difficult to converge, in terms of BCGS iterations, than usual. Otherwise, the convergence trends follow as expected.

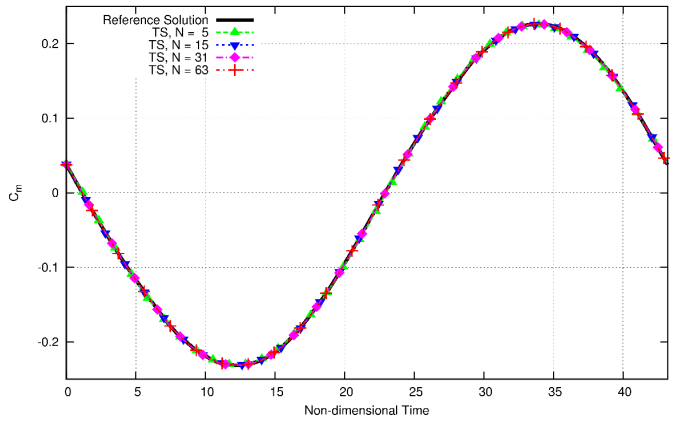
As previously, Figures (23) and (24) break down the contributions of the different calculations to the total time needed for solution at the AGARD5 reduced frequency only. As can be seen, results conform to the patterns previously observed for the flow alone

2. *Medium-mesh efficiency comparison*

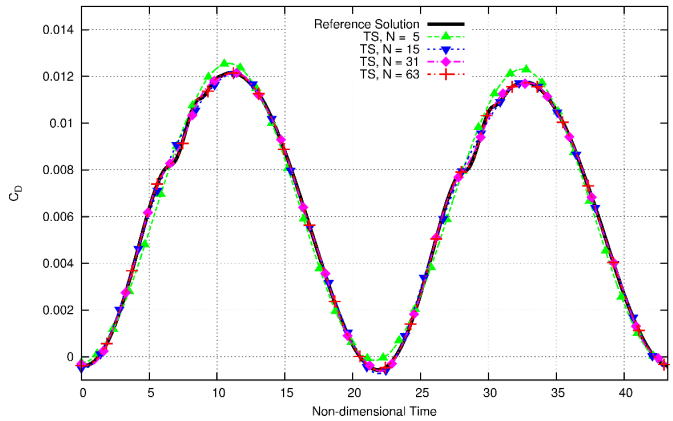
This section mirrors Section III.A.2 for the flow alone case with the same sequence of figures given by Figures (25)-(27). There is not much more that needs to be said here as the trends shown follow expectations except for the 47 time instance case at the base reduced frequency, which proved particularly difficult to converge. Otherwise, the medium



(a)



(b)



(c)

Figure 19. Comparison of computed lift coefficient (a), moment coefficient (b), and pressure drag coefficient (c) using the time-spectral method to a reference, time-implicit solution for aeroelastic AGARD5 test case

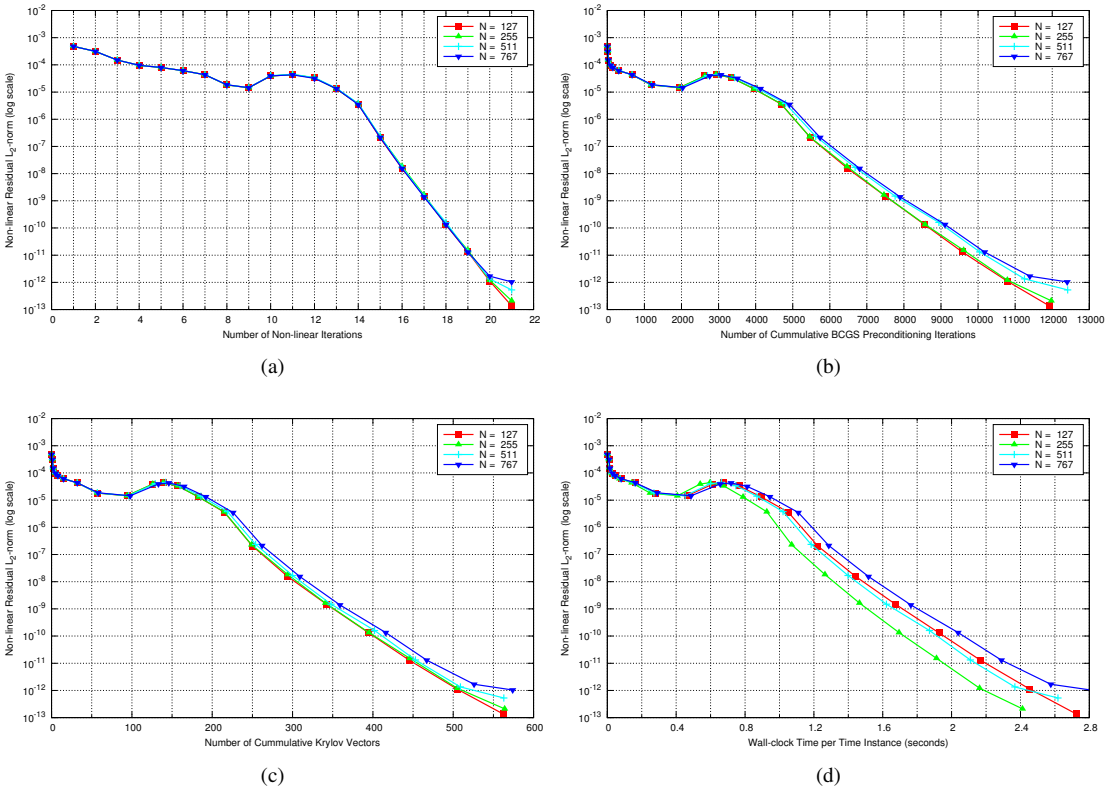


Figure 20. Comparison of convergence for the aeroelastic AGARD test case No. 5 with the indicated numbers of time instances using the GMRES-AF solver with Non-linear Iterations (a), BCGS Iterations (b), and Krylov vectors (c), and wall-clock time per time instance (d) on the x-axis

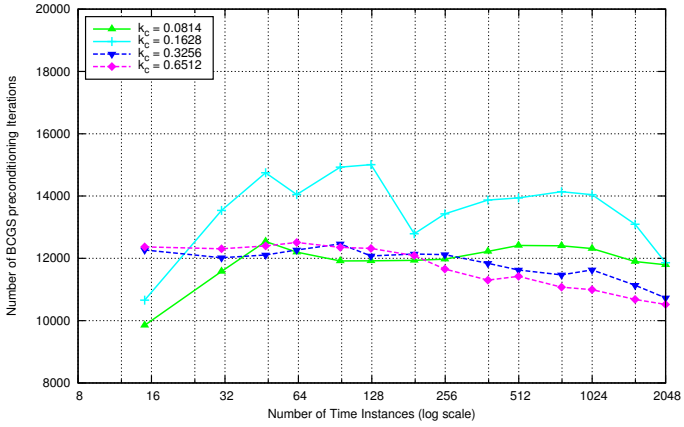


Figure 21. BCGS preconditioner iterations needed for converged aeroelastic solutions versus number of time instances for four reduced frequencies – from the AGARD5 frequency to eight times the AGARD5 frequency, doubling the frequency each time– using GMRES-AF solver

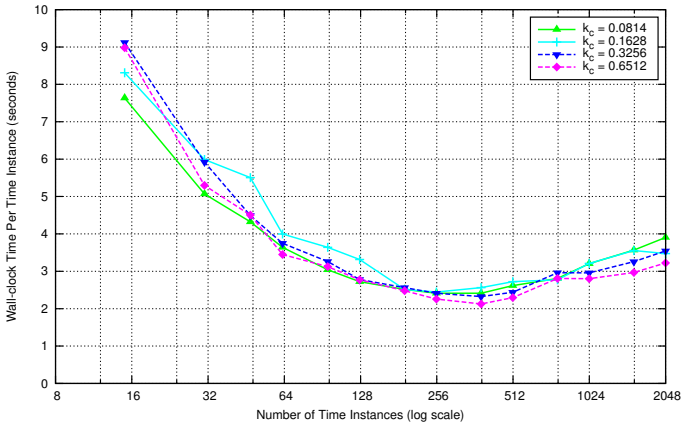


Figure 22. Wall-clock time per time instance needed for converged aeroelastic solutions versus number of time instances for four reduced frequencies – from the AGARD5 frequency to eight times the AGARD5 frequency, doubling the frequency each time– using GMRES-AF solver

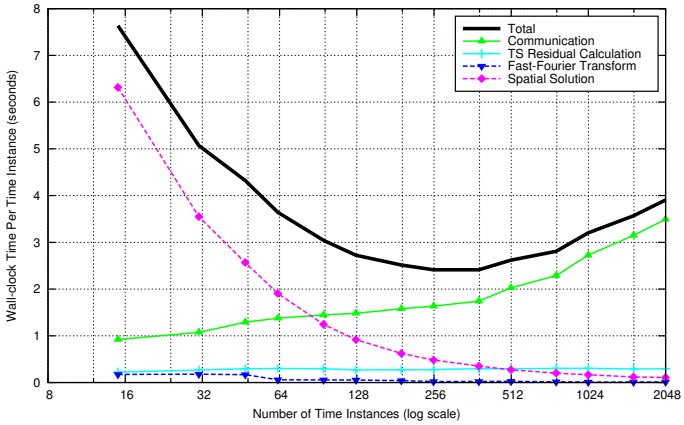


Figure 23. Wall-clock time per time instance needed for converged aeroelastic solutions on the coarse grid versus number of time instances using GMRES-AF solver with curves for the total time, communication time, TS residual calculation time, FFT time, and non-TS operations time.

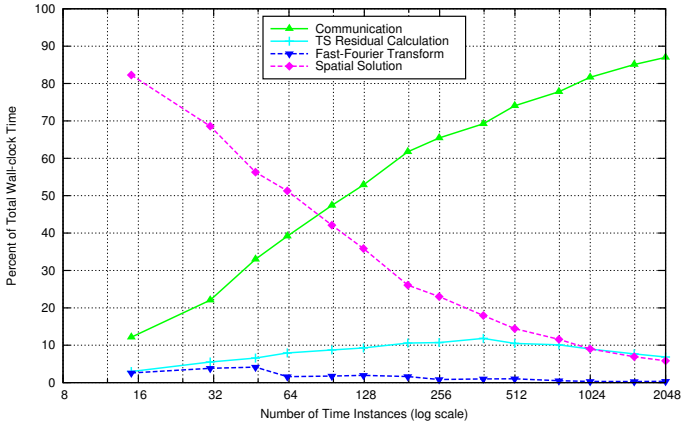


Figure 24. Percentage of the total wall-clock time needed for converged aeroelastic solutions on the coarse grid versus number of time instances using GMRES-AF solver with curves for the communication time, TS residual calculation time, FFT time, and non-TS operations time.

grid aeroelastic case serves to confirm the dominant use of computational time by the spatial calculations for small numbers of time instances and by the communication for large number of time instances. The aeroelastic test cases are not rerun on the fine grid because it is clear that such runs would offer no further insight.

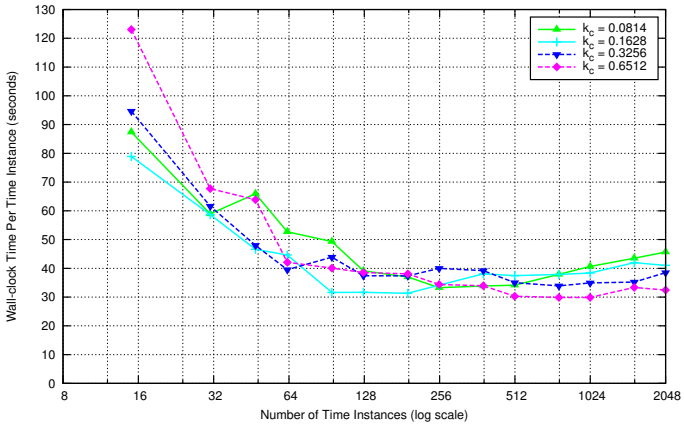


Figure 25. Wall-clock time per time instance needed for converged aeroelastic solutions on the medium grid versus number of time instances for four reduced frequencies – from the AGARD5 frequency to eight times the AGARD5 frequency, doubling the frequency each time– using GMRES-AF solver

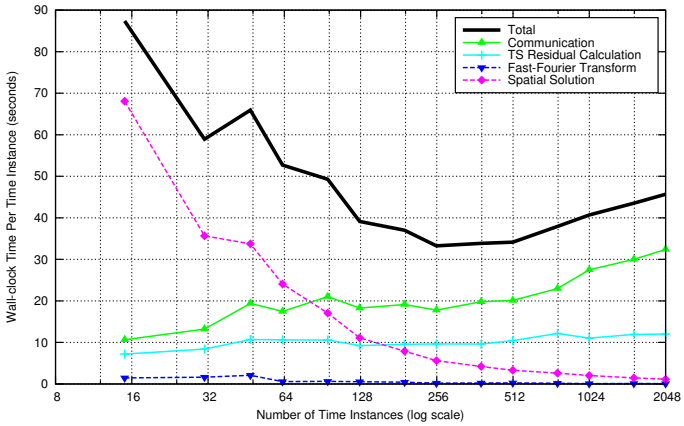


Figure 26. Wall-clock time per time instance needed for converged aeroelastic solutions on the medium grid versus number of time instances using GMRES-AF solver with curves for the total time, communication time, TS residual calculation time, FFT time, and non-TS operations time.

IV. Conclusions and Future Work

In the present work, improvements to the robust, wave-number independent time-spectral solver have been developed. The flexible variant of the Krylov subspace, generalized minimal residual method is used to solve the fully-coupled linear system of time-spectral Euler equations for each non-linear iteration. By using approximate factorization to invert the full spatial-temporal Jacobian approximately, wave-number independence of time-spectral solution convergence is retained while the wall-clock time to convergence is greatly improved. As compared to previous variants of the time-spectral solver,[11,12,17] the GMRES-AF solver reduces the solution time for all reduced frequencies and numbers of time instances considered. This time reduction is as little as a 36% of the GMRES-STI time for $N = 15$ time instances and the AGARD5 reduced frequency and as high as 93% of the GMRES-STI time for $N = 767$ time instances at the AGARD5 reduced frequency, i.e. the GMRES-STI solver can take over ten times longer than the GMRES-AF solver for the highest number of time instances compared. It should be noted, however, that cases with as many as $N = 2047$ time instances and at eight times the AGARD5 reduced frequency are also presented in this work. It should also be noted that the present work solves time-spectral problems on two-dimensional unstructured

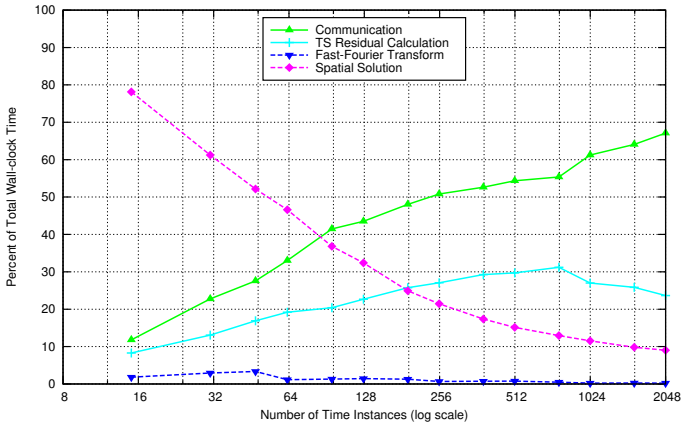


Figure 27. Percentage of the total wall-clock time needed for converged aeroelastic solutions on the medium grid versus number of time instances using GMRES-AF solver with curves for the communication time, TS residual calculation time, FFT time, and non-TS operations time.

grids using more time instances at higher reduced frequencies than has been found in the literature up to this point. That being said, no case yet examined has needed more than 255 time instances for an accurate solution, so, perhaps, attempts to make the solver more efficient for greater than 511 time instances are unwarranted. Further, although only two dimensions have been utilized, it is expected that all of the methods contained herein will readily extend to three dimensions.

The GMRES-AF solver is much more efficient than the GMRES-STI solver, but it is still far from optimal. For an optimal solver, the wall-clock time to convergence should remain constant for time-spectral solutions with any number of time instances using one CPU core per time instance. All this being said, there are no obvious paths forward to improve the efficiency of the time-spectral solver by focusing on the temporal aspects of the solver, unless somehow communication can be reduced as it is the dominant user of time for large numbers of time instances. Rather, further gains in efficiency appear to involve the more efficient solution of the spatial system, for instance through a multigrid approach, which can reduce the communication time by reducing the total number of Krylov vectors and preconditioning iterations required to converge solutions. Overall, the wave-number independence of the preconditioner used in the GMRES linear solver has proven to be a critical advance in the efficient solution of the time-spectral method. Coupled with the ability to treat the temporal and spatial discretizations independently (if approximately) by using approximate factorization, a truly efficient solver/preconditioner combination has been developed. All of the aforementioned efficiencies have been shown readily to extend to both aeroelastic problems and finer grid resolutions.

Future work will be focused on three key paths. First, attempts will be made to discover methods that reduce the amount of communication, perhaps by grouping several time instances on the same computational core or utilizing a more sparse time-spectral matrix in the inversion step by keeping only a limited number of diagonals, thereby reducing the number of solutions that need to be communicated. An alternative method that would completely eliminate the TS communication problem would be to keep all time instances on the same cluster node, exploiting shared memory parallelism in time, while using distributed memory parallelism only in space. The second focus will be the investigation of more efficient spatial solvers with hope that such solvers can improve temporal efficiency by reducing the total number of preconditioner iterations needed and thereby communication time. Finally, this work and everything leading up to it will be extended to three-dimensions.

References

[1]Hall, K. C., Thomas, J. P., and Clark, W. S., "Computation of Unsteady Nonlinear Flows in Cascades Using a Harmonic Balance Technique," *AIAA Journal*, Vol. 40, No. 5, 2002, pp. 879–886.

[2]McMullen, M., Jameson, A., and Alonso, J. J., "Acceleration of Convergence to a Periodic Steady State in Turbomachineary Flows," AIAA Paper 2001-0152, Jan. 2001.

[3]McMullen, M., Jameson, A., and Alonso, J. J., "Application of a Non-Linear Frequency Domain Solver to the Euler and Navier-Stokes Equations," AIAA Paper 2002-0120, Jan. 2002.

[4]Gopinath, A. K. and Jameson, A., "Time Spectral Method for Periodic Unsteady Computations over Two- and Three- Dimensional Bodies," AIAA Paper 2005-1220, Jan. 2005.

[5]van der Weide, E., Gopinath, A. K., and Jameson, A., "Turbomachineary Applications with the Time Spectral Method," AIAA Paper 2005-4905, June 2005.

[6]Lee, K.-H., Alonso, J. J., and van der Weide, E., "Mesh Adaptation Criteria for Unsteady Periodic Flows Using a Discrete Adjoint Time-Spectral Formulation," AIAA Paper 2006-0692, Jan. 2006.

[7]Sankaran, S., Gopinath, A., Weide, E. V. D., Tomlin, C., and Jameson, A., "Aerodynamics and Flight Control of Flapping Wing Flight Vehicles: A Preliminary Computational Study," AIAA 2005-0841, Jan. 2005.

[8]Choi, S., Potsdam, M., Lee, K., Iaccarino, G., and Alonso, J. J., "Helicopter Rotor Design Using a Time-Spectral and Adjoint-Based Method," AIAA 2008-5810, Sep. 2008.

[9]Choi, S. and Datta, A., "CFD Prediction of Rotor Loads using Time-Spectral Method and Exact Fluid-Structure Interface," AIAA 2008-7325, Aug. 2008.

[10]Saad, Y., *Iterative Methods for Sparse Linear Systems*, PWS Publishing Company, 1996.

[11]Mundis, N. L. and Mavriplis, D. J., "GMRES applied to the Time Spectral and Quasi-periodic Time Spectral Methods," AIAA Paper 2013-3084, June 2013.

[12]Mundis, N. L. and Mavriplis, D. J., "An Efficient Flexible GMRES Solver for the Fully-coupled Time-spectral Aeroelastic System," AIAA Paper 2014-1427, Jan. 2014.

[13]Thomas, J. P., Custer, C., Dowell, E. H., Hall, K. C., and Corre, C., "Compact implementation strategy for a harmonic balance method within implicit flow solvers," *AIAA Journal*, Vol. 51, No. 6, 2013, pp. 1374–1381.

[14]Leffell, J. I., Murman, S. M., and Pulliam, T. H., "An Extension of the Time-Spectral Method to Overset Solvers," AIAA Paper 2013-0637, Jan. 2013.

[15]Leffell, J. I., Murman, S. M., and Pulliam, T. H., "Time-Spectral Rotorcraft Simulations on Overset Grids," AIAA Paper 2014-3258, June 2014.

[16]Leffell, J. I., Sitaraman, J., Lakshminarayan, V. K., Wissink, A. M., Schroder, J., Kolev, T., and Falgout, R., "Evaliuation od Parrallel-in-Time Methods for Unsteady Computational Fluid Dynamics," AIAA Paper 2016-Pending, Jan. 2016.

[17]Mundis, N. L. and Mavriplis, D. J., "Wave-number independent preconditioning for GMRES time-spectral solvers," AIAA Paper 2015-0568, Jan. 2015.

[18]Canuto, C., Hussaini, M. Y., Quarteroni, A., and Zang, T. A., *Spectral Methods in Fluid Dynamics*, Springer, 1987.

[19]Hesthaven, J., Gottlieb, S., and Gottlieb, D., *Spectral Methods for Time-Dependent Problems*, Cambridge Monographs on Applied and Computational Mathematics, 2007.

[20]Sicot, F., Puigt, G., and Montagnac, M., "Block-Jacobi Implicit Algorithm for the Time Spectral Method," *AIAA Journal*, Vol. 46, No. 12, 2008, pp. 3080–3089.

[21]Mundis, N. L. and Mavriplis, D. J., "Quasi-Periodic Time Spectral Method for Aeroelastic Flutter Analysis," AIAA Paper 2013-0638, Jan. 2013.

[22]Mundis, N. L., Mavriplis, D. J., and Sitaraman, J., "Quasi-Periodic Time-spectral Methods for Flutter and Gust Response," 69th Forum of the American Helicopter Society, AHS International, Alexandria, VA, May 2013.

[23]AGARD, "Compendium of Unsteady Aerodynamic Measurements," Tech. Rep. No. 702, 1982.