A Scalable Time-Parallel Solution of Periodic Rotor Dynamics in X3D

Mrinalgouda Patil*
Graduate Research Assistant

Anubhav Datta
Associate Professor

University of Maryland, College Park, MD

A time-parallel algorithm is developed for large-scale three-dimensional rotor dynamic analysis. A modified harmonic balance method with a scalable skyline solver forms the kernel of this algorithm. The algorithm is equipped with a solution procedure suitable for large-scale structures that have lightly damped modes near resonance. The algorithm is integrated in X3D, implemented on a hybrid shared and distributed memory architecture, and demonstrated on a three-dimensional structural model of a UH-60A-like fully articulated rotor. Flight-test data from UH-60A Airloads Program transition flight C8513 are used for validation. The key conclusion is that the new solver converges to the time marching solution more than 50 times faster and achieves a performance greater than 1 teraFLOPS. The significance of this conclusion is that the principal barrier of computational time for trim solution using high-fidelity three-dimensional structures can be overcome with the scalable harmonic balance method demonstrated in this paper.

Nomenclature

A	square symmetric matrix
C	damping matrix
C_T	rotor thrust coefficient
C_a	artificial damping matrix
D	diagonal matrix
F	nodal aerodynamic and internal force vector
F_0	0^{th} harmonic component of F
F_{nc}	n^{th} harmonic cosine component of F
F_{ns}	n^{th} harmonic sine component of F
F_n	n^{th} harmonic complex Fourier coefficient of F
h_j	height of skyline matrix of j^{th} column
K	stiffness matrix
L	lower triangular matrix
M	mass matrix
N	number of harmonics
q	nodal solution vector
q_0	0^{th} harmonic component of q
q_{nc}	n^{th} harmonic cosine component of q
q_{ns}	n^{th} harmonic sine component of q
q_n	n^{th} harmonic complex Fourier coefficient of q
S	vector storing skyline entries of A
T	time period, s
α_s	shaft tilt angle, $^\circ$
ϵ_{11}	axial strain
ϵ_{13}	transverse shear strain
μ	rotor advance ratio
θ	blade pitch angle, $^\circ$
θ_0	collective, $^\circ$
θ_{1c}	lateral cyclic, $^\circ$
θ_{1s}	longitudinal cyclic, $^\circ$
σ	rotor solidity
Ω	rotor rotation speed, rad/s
ψ	blade azimuth angle, $^\circ$

Introduction

Rotorcraft comprehensive analyses today use beam-based structural dynamic models (Ref. 1). The extraction of periodic dynamics under trim condition is carried out routinely using finite elements in time or the harmonic balance method. This paper is on large-scale three-dimensional (3D) finite element based structures—a departure from 40 years of convention. For these large-scale structures, the routine methods of periodic solution break down.

The objective of this paper is to develop a time-parallel and scalable algorithm for the periodic solution of large-scale 3D structural dynamics. The aerodynamic forces on a rotor in trim condition occur periodically at integer multiples of the rotor speed. The rotor is in trim state for most of its flight, including the highest vibration regimes at transition and high speeds, hence obtaining an accurate periodic response is crucial for designing a rotorcraft. A large-scale 3D structural solver is a special-purpose high-fidelity tool envisioned for modeling new and advanced rotor blades with material and geometric discontinuities and predicting dynamic stresses and strains from first principles. The term large-scale means a large number of degrees of freedom, from tens of thousands to millions. The structural dynamics of a rotor is unique because of near (or at) resonance lightly damped modes, stiffness matrix of high

*Corresponding author; email: mpcsdspa@gmail.com.
Presented at the Vertical Flight Society 76th Annual Forum & Technology Display, October 6–8, 2020, Virtual. Manuscript received February 2021; accepted June 2021.

condition number, and nonlinear inertial couplings, all of which make direct extraction of periodicity on 3D structures difficult.

Motivation

Three-dimensional (3D) finite element structures unified with multibody dynamics is a recent development in helicopter rotors (Refs. 2–6). Its routine use is impeded by the high computational cost of trim solution in forward flight. There is a need for an approach that can solve the governing equations efficiently in a parallel and scalable manner both in space and time domains. In this paper, the focus is on the time domain.

The linearized governing equations obtained from a finite element model (FEM) of a structure has the well-known form,

$$M\ddot{q} + C\dot{q} + Kq = F \tag{1}$$

where M, C, and K are the mass, damping, and stiffness matrices, respectively. F is the nodal force vector from aerodynamic and internal forces (for nonlinear). The vector q is the nodal solution. Several unique conditions make it difficult to solve this equation to find the periodic solution q in response to a periodic forcing F. These are:

1) low (or zero) damping in many modes (especially lag) makes marching in time to reach periodic solution very inefficient;

2) the presence of near (or at) resonance modes (particularly for teetering/gimballed or articulated rotors) makes direct extraction of periodic solution to periodic forcing ill-conditioned;

3) numerical linearization of a large-scale problem into sensitivities typical of beam-based comprehensive codes,

$$F = A_0 + A_1 q + A_2 \dot{q} + A_3 \ddot{q} \tag{2}$$

to extract aerodynamic damping is impractical; and

4) numerically stiff problems having a large mass (M) and stiffness (K) matrix typically result in high condition numbers (ratio of highest and lowest frequencies), particularly in the presence of joints connecting the finite element structures.

The first two conditions are generic to all rotor models regardless of large– or small–scale, whereas the next two are unique to large-scale structures. Thus, trim solution in rotorcraft with a large-scale structural model is challenging and a fertile ground for innovative parallel and scalable algorithms in time.

Background

Periodic dynamics is a vast area with important applications in a variety of fields, from crystallography (Ref. 7) to circuit analysis (Ref. 8), to orbital mechanics (Ref. 9). Within mechanics, rotary-wing aeroelasticity has been a key area of application for many years. The commonly used approaches for solving periodic dynamics are broadly classified into: periodic shooting, finite elements in time, and time-spectral and harmonic balance methods.

Periodic shooting methods are a variation of time marching methods and simply involve iterations to find the right initial conditions. The main drawback of the shooting methods is stability. In addition, every iteration for initial condition will require many rotor revolutions to attain periodicity (due to condition 1). The initial conditions can also be found by the Floquet transition matrix (Ref. 10), a method more robust but also more expensive.

Direct extraction of periodicity without time marching can be carried out using finite elements in time, time-spectral methods, and harmonic balance methods. The first two are formulated in the time domain, and the latter is formulated in the frequency domain.

Finite elements in time (FET) is a widely used method. This method uses local time shape functions and is well suited for capturing local gradients. It can be traced back to the 1960s when Argyris and Scharpf (Ref. 11) introduced the method in nuclear engineering. Borri (Ref. 12), and Peters and Hou (Ref. 13) first applied it in rotorcraft to study a flapping rotor. The work was extended by Panda and Chopra (Ref. 14) and Dull and Chopra (Ref. 15) to flap, lag, and torsion. The main advantage of FET is that it can capture high-frequency response with a relatively few time elements. These formulations are typically used in conjunction with numerical linearization of forcing to extract aerodynamic damping, thereby overcoming condition 2. However, for large-scale structures, this linearization is impractical, as noted in condition 3.

The time-spectral method uses global time shape functions. It is unsuitable for capturing local gradients. It is also prone to aliasing errors. It is rarely used in structures, but has been used with computational fluid dynamics (CFD) for analysis of helicopter rotor flows (Ref. 16). These methods might be suitable for fluids but lead to matrices of overwhelming size in structures with no apparent benefit.

Harmonic balance is a frequency-domain method. It uses Fourier expansions as shape functions. It is a global method and also unsuitable for local gradients. But it has no aliasing errors, and for elliptic problems such as structures where the solution is guaranteed to be smooth, as many harmonics as needed can be introduced to resolve the solution. Several other frequency-domain methods have been explored in large-scale fluid flows from systems including variable time periods (Ref. 17) to occasional helicopter rotor flows (Ref. 18), but none of these methods are suitable for near resonance low damped structural dynamics characterized by the four unique conditions given earlier. In rotors, harmonic balance is used routinely since the early work of Peters and Ormiston in the 1970s (Ref. 19). It will be shown later that this standard harmonic balance is not scalable to large problems due to matrix structure.

Historically, most efforts to parallelize the numerical solution of partial differential equations focus on the spatial discretization of the problem. In dynamic problems, time adds a new dimension, and when periodic solution is desired, particularly in a low damped near resonance system like the rotor, time becomes a critical bottleneck to scalability. A class of algorithms called Parallel in Time Algorithms (PITA) have been studied since 2001 for dynamic problems (Ref. 20) and have found applications in various areas, from chemical kinetics (Ref. 21) to power systems (Ref. 22). However, they can fail with second-order systems of linear oscillators due to stability problems of near resonance modes (condition 2).

Time is fundamentally a serial concept. It has no boundaries, hence cannot be partitioned and parallelized in the same manner as space. For periodic systems, a transformation to frequency domain can be utilized to solve in parallel. But in addition to this transformation, other modifications are needed to account for the unique conditions 1–4 encountered in rotorcraft. These are the principal objectives of this work.

Organization of paper

Following this introduction, the paper begins by listing the test cases to be used for verification and validation throughout the paper. The next section explains why the two common approaches, finite elements in time and standard harmonic balance, fail for large-scale structures. The following section describes a Modified Harmonic Balance (MHB) method proposed in this paper. The skyline solver, the main building block of the MHB, is described next. The following two sections cover the implementation of the solver and validation with UH-60A flight-test data. The final section presents parallel performance and comparison with time-integration. The paper ends with conclusions.

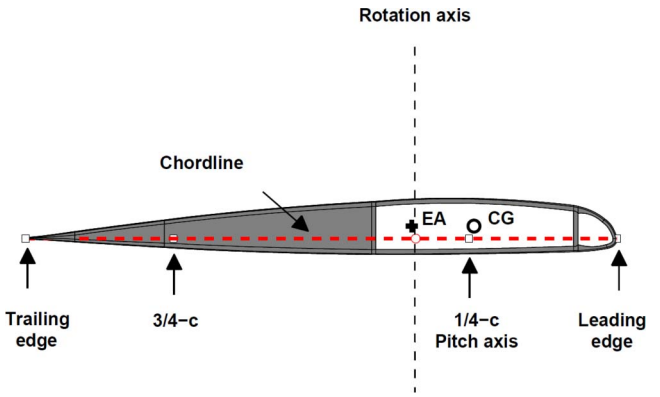


Fig. 1. Idealized internal structure of a UH-60A-like articulated rotor.

Test Cases

Throughout the paper, the following test cases will be used.

1) *An elementary case*. A 3D cantilevered beam with
   (a) constant tip force, and
   (b) a sinusoidal tip force.

2) *An analytical flap bending problem*. This problem given by Harris (Ref. 23) has an exact flexible flap bending solution in forward flight at an advance ratio $\mu = 0.287$ with a prescribed analytical aerodynamic forcing.

3) *The UH-60A-like rotor*. The internal structure is an idealization (see Fig. 1), reverse engineered to reproduce similar fan plot as the real rotor. The true internal structure is not available in public domain. The external geometry and aerodynamic description including airfoil decks are nearly exact and follows the Army-NASA Database. Only the trim tab is neglected.

The control inputs are provided through joint commands at the pitch bearing. The pitch bearing is coincident with the flap and lag hinge colocated at 4.66%R so a single joint is sufficient. The bearing also includes a linear lag damper. The actual damper properties and connection to the blade are complications deemed unnecessary for the purposes of this paper. Figure 2 shows the 3D model. There are 592 hexahedral bricks with 27 nodes each, a total of 17,523 degrees of freedom. Further details of the 3D model can be found in Ref. 3.

The UH-60A model is tested on three cases. Cases (a) and (b) are made-up test cases used for verifying certain features of the solver. Case (c) is a real flight-test case used for validating the solver.

(a) *Hover with arbitrary higher harmonic pitch inputs*. The pitch angle is given by Eq. (3), where Ω is the rotor speed and t is the time. In hover, there is no interharmonic coupling, a p/rev cyclic produces a p/rev response.

$$\theta = 10^\circ + 3^\circ\cos(\Omega t) - 5^\circ\sin(\Omega t) + 3^\circ\sin(2\Omega t) + 3^\circ\sin(4\Omega t) + 2^\circ\sin(6\Omega t) \quad (3)$$

(b) *Forward flight at advance ratio $\mu = 0.37$ with arbitrary 1/rev pitch inputs*. In forward flight, there is interharmonic coupling.

$$\theta = 10^\circ + 3^\circ\cos(\Omega t) - 5^\circ\sin(\Omega t) \quad (4)$$

(c) *Low-speed transition flight at advance ratio $\mu = 0.15$*. The pitch inputs are calculated by trim solution. This is flight-test counter 8513 from the UH-60A Airloads Program. Measured rotor thrust and hub moments are targeted with collective and cyclic controls. The aircraft pitch is fixed at the measured value. This is the same condition previously studied in the Black Hawk Loads workshop (Ref. 24).

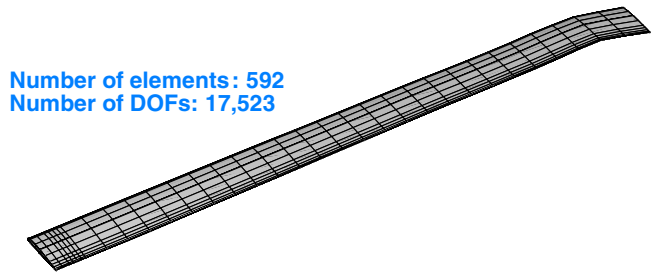
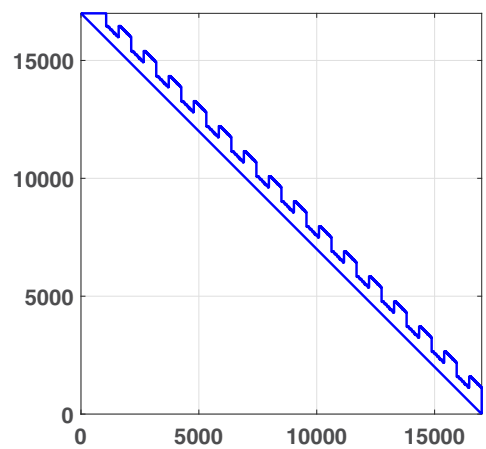
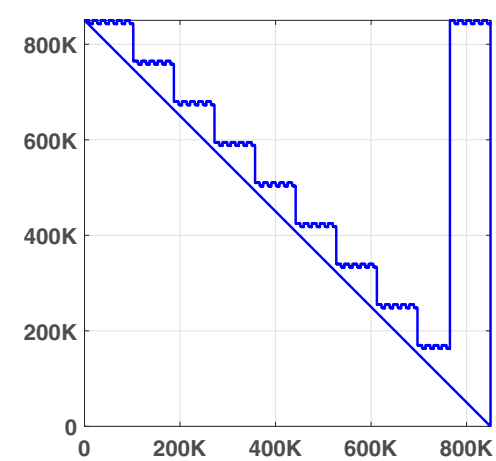


Fig. 2. Three-dimensional brick mesh of an UH-60A-like articulated rotor; lifting-line geometry, and aerodynamic definitions are exact (based on Ames database); the flap, lag, and torsion bearing are at 4.66% R.



(a) FEM skyline



(b) FET skyline

Fig. 3. Structure of FEM and FET matrices; (a) FEM skyline, and (b) FET skyline.

Finite Element in Time

Figure 3(a) shows the structure of the 3D finite element model (FEM) stiffness matrix of the UH-60A rotor. Figure 3(b) is the matrix obtained using Finite Element in Time (FET). The structure of both matrices resembles a skyline where only the entries below the stepped line are nonzero. Skyline representation is a form of sparse matrix storage widely

used in structural mechanics, where only the entries below the skyline are stored. More details on skyline representation and skyline solver are given later. The FEM matrix has a size of around 17,000, but it is tiny compared to the size of the FET matrix. The size of the FET matrix—with 10 time elements and fifth-order polynomial shape functions—is of the order of 850,000. This makes it impractical for storage. It also has a higher bandwidth due to the periodic boundary condition. This kills the efficiency of the solver. Due to these reasons, the FET is infeasible for a large-scale structural problem.

Harmonic Balance

In harmonic balance, the size of the matrices do not increase as drastically as in finite element in time. But there are other problems. In harmonic balance, the solution and the forcing are expanded as Fourier series. Consider the discrete linearized governing equations of the finite element model (FEM) given in Eq. (1). The solution and the forcing are expanded as Eqs. (5) and (6), respectively.

$$q = q_0 + \sum_{n=1}^{N} q_{nc}\cos(n\Omega t) + \sum_{n=1}^{N} q_{ns}\sin(n\Omega t) \tag{5}$$

$$F = F_0 + \sum_{n=1}^{N} F_{nc}\cos(n\Omega t) + \sum_{n=1}^{N} F_{ns}\sin(n\Omega t) \tag{6}$$

A total of N harmonics is assumed with a fundamental time period T. For a rotor, $T = 2\pi/\Omega$, where Ω is the rotor speed in rad/s. The assumption is M, C, K are time invariant, and the time varying quantities are all on the right-hand side of Eq. (1). For example, if $K = K_0 + \sum_{n=1}^{N} K_{nc}\cos(n\Omega t) + \sum_{n=1}^{N} K_{ns}\sin(n\Omega t)$, then only K_0 is kept on the left-hand side, the time-varying part is moved to the right. The right-hand side can then be iterated. In rotors, the dominant nonlinearity originates from rotation. Hence, the stiffness matrix (K_0) is calculated about a rotor solution in vacuum. Substituting the assumed solution (Eq. (5)) in the governing equation (Eq. (1)), and equating the coefficients from left and right-hand sides yields Eq. (7).

$$\begin{bmatrix} (K - n^2\Omega^2 M) & n\Omega C \\ -n\Omega C & (K - n^2\Omega^2 M) \end{bmatrix} \begin{pmatrix} q_{nc} \\ q_{ns} \end{pmatrix} = \begin{bmatrix} F_{nc} \\ F_{ns} \end{bmatrix} \tag{7}$$

This is the essence of the standard harmonic balance method; the amplitude of Fourier components is balanced frequency by frequency such that the governing equation is satisfied. The accuracy of the solution improves with the number of harmonics, N. The system of equations in Eq. (7) is a barrier for large-scale structures as the matrix skyline was broken, it is no longer symmetric, and the sine and cosine components are coupled.

Modified Harmonic Balance

In the modified harmonic balance method, the solution and the forcing are still expanded as Fourier series, but the solution procedure is modified. Consider Eq. (7) again. Two modifications are performed to retain the skyline, bring back a symmetric structure while keeping the sine and cosine components uncoupled.

Modification 1

The first modification recovers the symmetric form. This is performed by multiplying the transpose of the left-hand side matrix of Eq. (7) on both sides to produce Eq. (8).

$$\begin{bmatrix} (K - n^2\Omega^2 M)^2 + (n\Omega C)^2 & 0 \\ 0 & (K - n^2\Omega^2 M)^2 + (n\Omega C)^2 \end{bmatrix} \begin{pmatrix} q_{nc} \\ q_{ns} \end{pmatrix} = \begin{bmatrix} (K - n^2\Omega^2 M)F_{nc} - n\Omega C F_{ns} \\ (K - n^2\Omega^2 M)F_{ns} + n\Omega C F_{nc} \end{bmatrix} \tag{8}$$

The resulting system is symmetric and keeps the sine and cosine components uncoupled. The problem is that it breaks the original skyline.

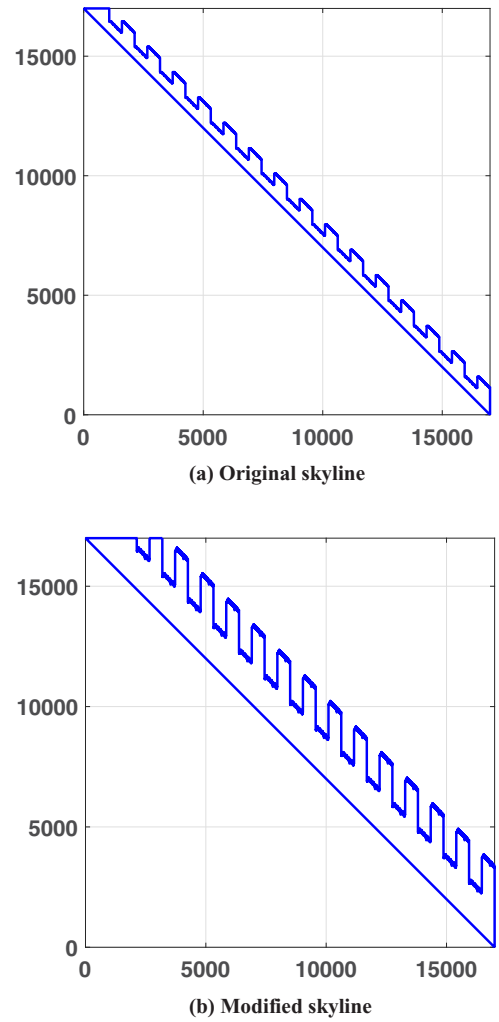


Fig. 4. Comparison of original and modified skyline; (a) original skyline, and (b) modified skyline; squaring of matrix alters the structure of the original skyline.

Modification 2

The matrix $(K - n^2\Omega^2 M)^2 + (n\Omega C)^2$ in Eq. (8) has a new and deeper skyline. Figure 4(a) shows the skyline of $K - n^2\Omega^2 M$ for the UH-60A-like rotor, whereas Fig. 4(b) shows the skyline of $(K - n^2\Omega^2 M)^2$. This problem is solved by expressing the squared matrix as a product of two complex matrices (Eq. (9)).

$$\begin{aligned} &[(K - n^2\Omega^2 M)^2 + (n\Omega C)^2]q_{nc} \\ &\quad = [(K - n^2\Omega^2 M) + i(n\Omega C)] \times [(K - n^2\Omega^2 M) - i(n\Omega C)]q_{nc} \\ &\quad = [(K - n^2\Omega^2 M) + i(n\Omega C)]\hat{q}_{nc} \end{aligned} \tag{9}$$

Equation (9) assumes C to be a diagonal matrix where i is the imaginary unit, $i = \sqrt{-1}$. Now two linear solves are performed for each harmonic as shown in Eqs. (10) and (11).

$$[(K - n^2\Omega^2 M) + i(n\Omega C)]\ \hat{q}_{nc} = (K - n^2\Omega^2 M)F_{nc} - n\Omega C F_{ns} \quad (10)$$

$$[(K - n^2\Omega^2 M) - in\Omega C] \quad q_{nc} = \hat{q}_{nc} \quad (11)$$

First, an intermediate solution $\hat{q}_{nc}$ is obtained from Eq. (10). Then, the final solution q_{nc} is obtained from Eq. (11). The price to pay is two solves instead of one. The above modifications were reported in Ref. 25. An alternative approach is to express the solution and forcing in the complex-exponential form directly using Eqs. (12) and (13), respectively. q_n and F_n are the n^{th} harmonic Fourier coefficients of nodal solution and forcing expressed in complex form, respectively.

$$q = \sum_{n=-N}^{N} q_n\ e^{in\Omega t} \quad (12)$$

$$F = \sum_{n=-N}^{N} F_n\ e^{in\Omega t} \quad (13)$$

Substitution in the governing equation yields Eq. (14). This complex-exponential representation avoids the formation of nonsymmetric system in the first place. Moreover, the equation corresponding to each harmonic retains the original skyline. However, it comes at the cost of storage, a complex number requires twice the storage of a real number.

$$[(K - n^2\Omega^2 M) + i(n\Omega C)] \quad q_n = F_n \quad (14)$$

Expressing the solution in complex domain requires necessary transformations between complex and real domain. Equation (15) performs the conversion from complex to real, and Eq. (16) real to complex.

$$q_{nc} = 2\ \text{Re}\{q_n\}; \quad q_{ns} = -2\ \text{Im}\{q_n\} \quad (15)$$

$$q_n = \tfrac{1}{2}(q_{nc} - i q_{ns}) \quad (16)$$

The real coefficients of the solution (q_{nc}, q_{ns}) are the deflections, which are then used to calculate airloads. The real coefficients of airloads $(F_{nc},\ F_{ns})$ are then converted to the complex coefficient F_n, which is used to solve Eq. (14). Thus, the harmonic solution q_n is obtained in the complex domain.

Of course, both the formulations produce the same solution. The second approach in complex domain uses one skyline solve per harmonic, making it faster than the first approach when implemented in serial. When implemented in parallel, both approaches yield the solution at the same time as the cosine and sine harmonics are uncoupled. Thus, solving Eqs. (10) and (11) (first approach) or Eq. (14) (second approach) now lies at the heart of the algorithm. This is described in the next section.

The Skyline Solver

Finite element matrices have a structure that resembles a skyline. To reduce the storage requirements for large-scale structures, only entries below the skyline are stored. A skyline solver uses direct factorization to solve a system of algebraic equations:

$$Ax = b \implies (LDL^T)x = b, \quad (17)$$

where A is a square symmetric matrix, and L and D are lower triangular and diagonal matrices, respectively. The solution x can then be found by triangular solves (forward and backward substitutions). The skyline storage is preserved, which means all the entries that might change from zero to a nonzero value during the factorization are guaranteed to fall within the skyline; so no entry outside the skyline is ever used or populated. The LDL^T factorization is the intricate and expensive stage.

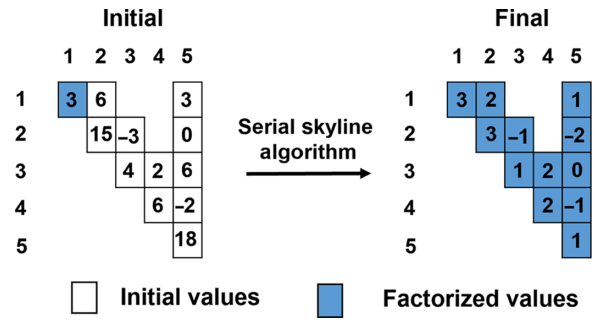


Fig. 5. Skyline of a matrix in its initial and final states.

Serial skyline solver

Skyline solvers are routine in finite elements, and its parallelization is what is novel in this paper. But in order to understand the parallel algorithm, it is useful to review the serial algorithm first.

Suppose there is a square symmetric matrix A of size (5×5) with a skyline structure as shown in Fig. 5. It is symmetric, so only the diagonal and upper triangular part is shown. The matrix A is stored as a vector S containing only the skyline entries. Another vector of row pointers r contains the indices of the diagonal positions of each column. In Fig. 5, S and r are,

$$S = (3, 6, 15, -3, 4, 2, 6, 3, 0, 6, -2, 18)$$
$$r = (1, 3, 5, 7, 12) \quad (18)$$

The skyline format stores the entries of the matrix column by column, starting from the first to the last column. Within each column, the entries are stored from the first nonzero row to the diagonal element. The size of the vector r is equal to size of the matrix. The height (h_j) of each column j is,

$$h_j = r_j - r_{j-1} \quad \text{for} \quad j > 1 \quad (19)$$

An entry of the original matrix can be found from the skyline vector as

$$A_{ij} = S(r_j - j + i) \quad \text{for} \quad j \geq i \quad (20)$$

In Fig. 5, the white entries are the initial matrix values, while the shaded entries are the final factorized values. The starting entry will remain the same, hence it is already shaded. After the factorization is completed, S will automatically contain the solution L^T at the nondiagonal positions, and D at the diagonal positions. For our simple example,

$$\text{Factorized } S = (3, 2, 3, -1, 1, 2, 2, 1, -2, 0, -1, 1) \quad (21)$$

The factorization of any nondiagonal element at row i and column j is given by Eq. (22), and that of any diagonal element in column j is given by Eq. (23), where A^{fact} is the final factorized entry and A^{orig} is the original matrix entry. In Eq. (22), m_i and m_j correspond to the row number of first element in column i and column j, respectively.

$$A_{ij}^{\text{fact}} = \frac{1}{A_{ii}} \left(A_{ij}^{\text{orig}} - \sum_{l=MAX(m_i,m_j)}^{i-1} A_{li}\ A_{lj} \right) \quad (22)$$

$$A_{jj}^{\text{fact}} = A_{jj}^{\text{orig}} - \sum_{i=m_j}^{j-1} A_{ij}^2\ A_{ii} \quad (23)$$

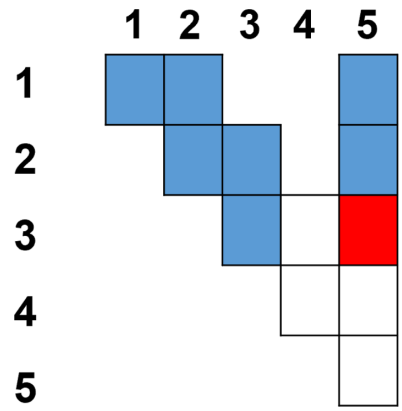


Fig. 6. Skyline solver algorithm — to compute entry (3,5), all the shaded entries are needed.

The factorization is performed each element at a time, in the ascending order of the columns. To compute an element at row i and column j, the following are required:

(1) all elements above (i, j) in column j, and

(2) all elements up to column i.

Figure 6 depicts the operation pictorially. For example, to compute the entry (3,5), information from all the shaded elements is needed. This means all the shaded elements must be factorized before the entry (3,5). Observe, the elements from column 4 are not needed. The parallel algorithm is built to take advantage of this observation.

Parallel skyline solver

The parallel algorithm is built for speed, hence involves a large number of communications. Shared memory architectures are best suited for its implementation because of guaranteed data locality. For purposes of illustration, consider 4 processors, namely $\mathbf{P_0}$, $\mathbf{P_1}$, $\mathbf{P_2}$ and $\mathbf{P_3}$. Each processor is assigned a column as shown in Fig. 7.

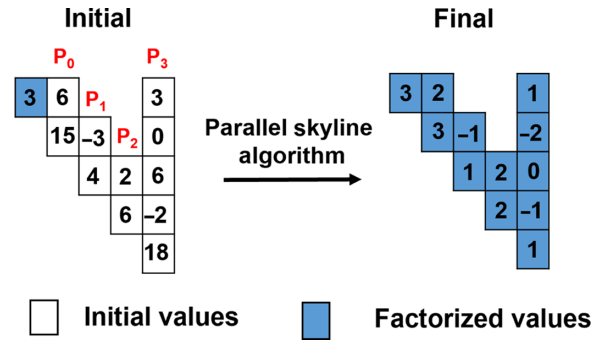


Fig. 7. Assignment of processors to different columns of skyline.

Each processor is instructed to complete the computations in its column and then proceed to the next available column. In the first step, the processors $\mathbf{P_0}$ and $\mathbf{P_3}$ can compute the first elements of their respective columns. This is because all the elements needed are available. In the next step, only $\mathbf{P_0}$ can compute the next element. Once $\mathbf{P_0}$ has completed the column it was assigned, it can move on to the next available column. Note that, at each step, only the processors which have the required entries available can compute, others have to wait. Figure 8 shows a step-by-step depiction of the parallel execution. The fact that it is stored within the skyline makes it complicated for implementation, but possible, as shown here.

Pseudo-code of parallel skyline algorithm

The pseudo-code for the parallel algorithm of skyline solver is given in Algorithm 1. The algorithm is implemented using OpenMP, which is an API that supports shared-memory programming. The algorithm begins by assigning shared variables: the skyline vector S and row pointer vector r. Next, private variables are assigned to each processor: namely, the processor number id, the column number j, the position of an element

Algorithm 1 Parallel Skyline Solver

```
1:  OMP SHARED (....S, r,....)                                   ▷ Assign shared variables
2:  OMP PRIVATE (....id, j, k, jp, kp,....)                      ▷ Assign private variables
3:  while {All columns are factorized} do
4:      Assign next available column j to processor id
5:      jp = Most recent factorized column number                ▷ Start with jp = 1
6:      kp = Most recent factorized diagonal position            ▷ Start with kp = 1
7:      while {Each element k of column j is factorized} do
8:          OMP FLUSH (jp, kp)   ▷ Updates jp, kp on that processor to the most recent written value
9:          if {Column (j − 1) is factorized} then               ▷ Complete factorization of the column
10:             Factorize all elements of column j
11:         else                                                 ▷ Partial factorization of the column
12:             if {Conditions 1 and 2 for each k in column j} then
13:                 Factorize each k inside column j that satisfies both conditions
14:             else
15:                 Wait till the conditions are satisfied
16:             end if
17:         end if
18:         if {Column j is factorized} then
19:             OMP CRITICAL                                     ▷ Performed by one processor at a time
20:             Update jp = j
21:             Update kp = r_j
22:         end if
23:     end while
24: end while
```

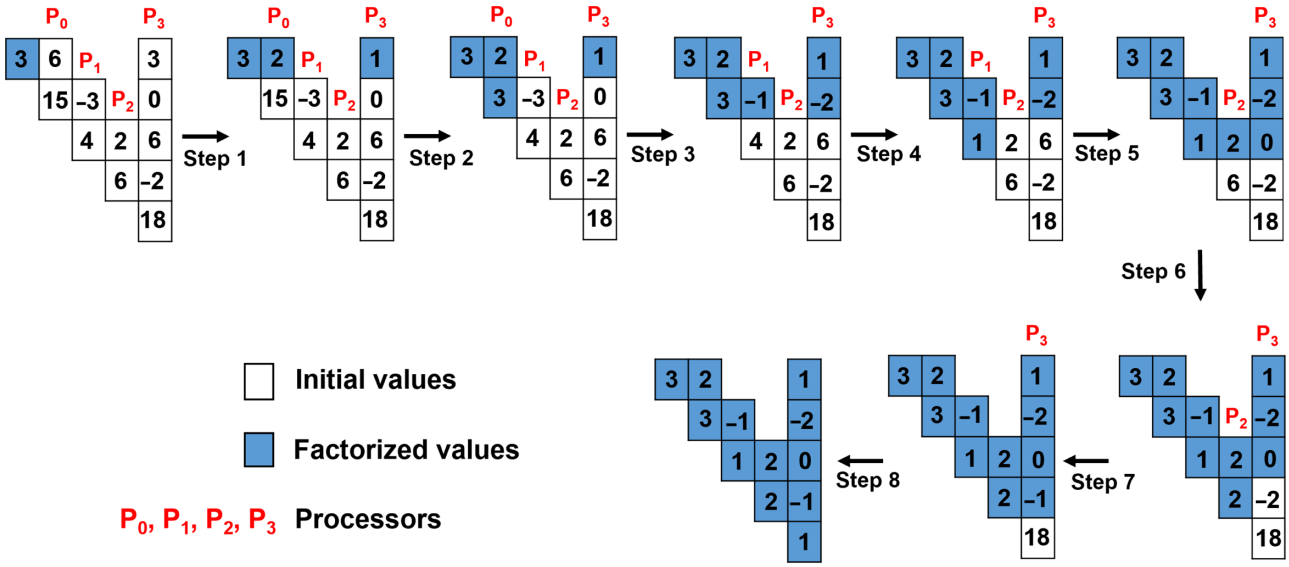


Fig. 8. Step-by-step execution of the parallel skyline algorithm.

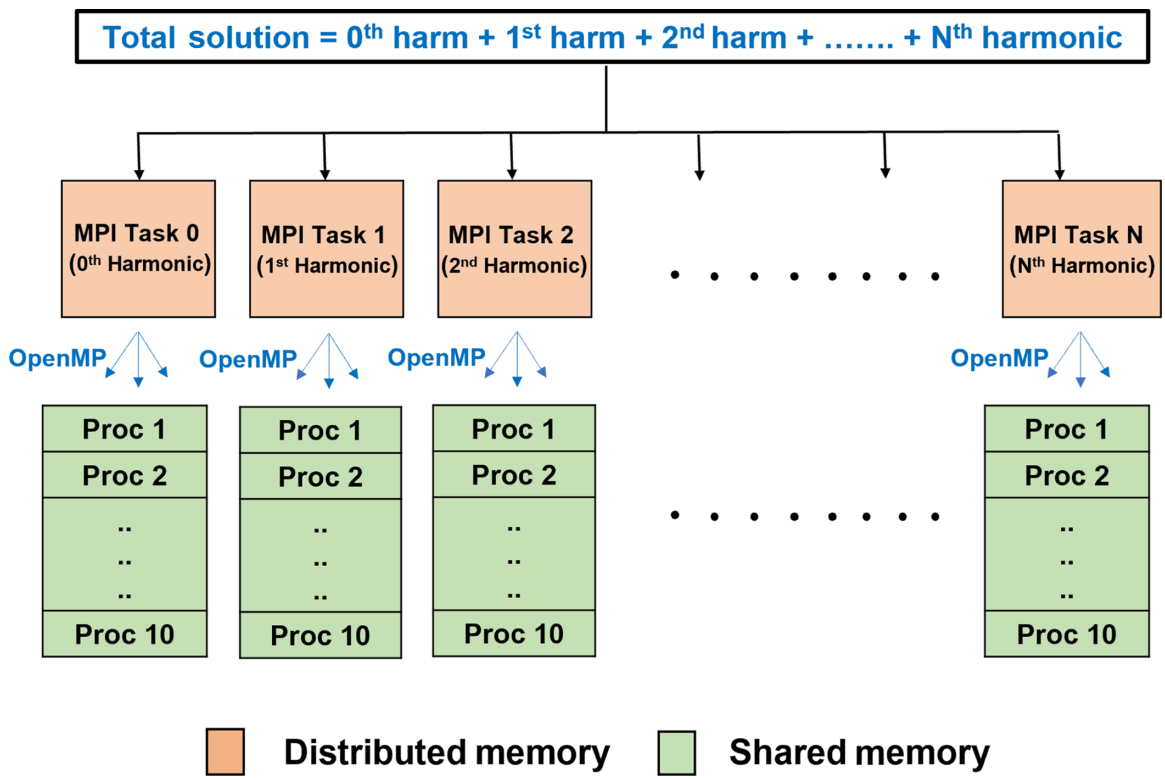


Fig. 9. Implementation of modified harmonic balance using shared and distributed memory architecture.

inside the column k, the most recent factorized column number jp and the most recent factorized diagonal position kp.

The outer loop executes until all the columns are factorized (While loop in line 3). The first step is to assign each processor a column j. Both jp and kp are set to 1 because the first column is already factorized. So to speak each processor cycles over each element k in its column until all are factorized. This is the inner loop (While loop in line 7).

Within the inner loop, line 8 of the algorithm keeps the values of jp and kp current using the OpenMP command FLUSH. The FLUSH command ensures that each processor knows the current values of jp and kp of other processors. The processor assigned to the second column ($j = 2$) can factorize the entire column as the first column ($j = 1$) is already factorized. In general, all the elements of column j can be factorized if column $j - 1$ is available in factorized form (If condition in line 9). If, however, column $j - 1$ is not available, then column j can only be factorized partially (Else condition in line 11) only for entries for which both the conditions mentioned in Fig. 6 are met (If condition in line 12). For the example shown in Fig. 8, in step 1 processor $\mathbf{P_3}$ could factorize the first element of its column whereas processors $\mathbf{P_1}$, $\mathbf{P_2}$ must wait until $\mathbf{P_0}$ completes its column (Wait condition in line 15).

Each processor updates jp and kp once the column assigned to it is factorized entirely (lines 20 and 21). This step is performed in serial using

the OpenMP command CRITICAL, to avoid date races across processors. For the example shown in Fig. 8, at the end of second step, processor $\mathbf{P_0}$ updates jp to 2 and kp to 3. The updated jp and kp are flushed across other processors. With the completion of its column, a processor is assigned to the next available column, and the process repeats.

The wait times encountered by the processors during the execution of the algorithm do not affect the solver's performance too adversely. When a processor is assigned a new column, it immediately begins to factorize what it can. During this time, the conditions necessary for complete factorization are often fulfilled. So the process proceeds to completion. Hence for a problem with many degrees of freedom, it is found that the wait times become negligible as the factorization proceeds.

Rotor Solution Procedure

The solution procedure for helicopter rotors call for an additional modification. This is generic to all models, large or small scale. But for large scale, the typical linearization to provide aerodynamic damping is no longer practical. Yet, damping is even more crucial in large scale due to the richer frequency content.

$$M\ddot{q}_k + C_a\dot{q}_k + C\dot{q}_k + Kq_k = F + C_a\dot{q}_{k-1} \tag{24}$$

The solution is simple: an artificial diagonal damping C_a is added to the left and removed from the right through iterations. Thus, even at flapping frequency 1/rev, the problem remains well posed. The original Eq. (1) is modified to Eq. (24). The subscript k refers to the iteration number. A relaxation factor can be used for faster convergence.

The modified harmonic balance is designed for implementation on a large-scale computer architecture consisting of both shared and distributed memory. As stated earlier, the solution of the governing equation is expanded as the sum of harmonics (Eq. (5)). Since these harmonics can be solved independent of one another, they are computed in parallel using distributed memory architecture (MPI). A total of $N + 1$ MPI tasks are assigned for obtaining solution with N harmonics $(0, 1, 2, \ldots, N)$. Each MPI task is assigned with 10 shared memory processors (OpenMP programming). Overall, hybrid MPI-OpenMP parallelism involving a total of $10 \times (N + 1)$ processors is used. Note that the shared memory processors are only spawned when necessary. Figure 9 shows the schematic representation of the employment of shared and distributed memory processors in implementing the modified harmonic balance. This hybrid parallelism improves performance dramatically while reducing memory requirements for each processor.

A pseudo-code is shown in Algorithm 2. The algorithm begins with an assumed solution in frequency domain (q_0, q_{nc}, q_{ns}) and assigns each harmonic to a separate MPI task. The harmonics are solved iteratively till the solution is converged (line 3). The first step in the iteration is to obtain the mass (M), stiffness (K), and damping (C) matrices. This is followed by computing the nodal solution (q) and velocities $(\dot{q})$ in the time domain using Eq. 5 (line 5). These are used to calculate the aerodynamic forcing at all azimuths. The aerodynamic forcing obtained is now assembled as a force vector F using the finite element model at all azimuths (line 7). The next step adds the artificial damping to F and computes F_n (lines 8 and 9). The final step solves the system of equations in Eq. 14 for each harmonic (line 10) in complex domain (q_n) which is in turn converted to the real domain (q_0, q_{nc}, q_{ns}) for starting the next iteration. The skyline solves of each harmonic are performed in parallel by each MPI task. Within each MPI task, the parallel skyline solver is employed across shared memory processors.

For large-scale finite element structures with multibody joints, the construction of matrices and forcing vectors at all azimuths significantly contributes to the total solution time. This workload of construction is distributed across MPI tasks and are internally parallelized using OpenMP.

X3D

X3D is a US Army / University of Maryland aeroelastic solver which utilizes 3D finite element analysis to model rotors (Ref. 4). Kinematic couplings are simulated by multibody joints. 3D stresses and strains fall out of the solution directly. In addition to structural dynamics, X3D has a built-in lifting-line model and an interface to couple with CFD.

The modified harmonic balance was implemented in X3D to build a new, refined, parallel version. In addition to the solution procedure, all other parts of X3D were parallelized using the shared memory architecture. Only the inputs and outputs were left serial. The results shown in the later sections were obtained using this new version of X3D.

The results were obtained on Deepthought2, University of Maryland's High-Performance Computing cluster. It consists of 480 nodes with a dual socket on each node. Each socket has 10 Intel Ivy Bridge E5-2680v2 processors running at 2.8 GHz. So a total of 20 processors are available on each node. Each of these processors has a separate L1 cache (64 KB), a separate L2 cache (256 KB), a shared L3 cache (25 MB), and a total shared memory of 128 GB.

Performance of Skyline Solver

The first test case is 1a: 3D cantilevered beam with a constant tip force (Fig. 10). This elementary problem is used to record the speedup of the parallel skyline solver. Figure 11 shows the speedup on a node for various mesh sizes ranging from 10 K to 250 K degrees of freedom. Regardless of size, a speedup of up to 17 is obtained over 19 processors (90% scalability). Of 20 processors on each node, one is always dedicated to system management. Thus, only 19 were used for computation.

Algorithm 2 Periodic Rotor Response

1: Start with assumed solution in frequency domain (q_0, q_{nc}, q_{ns})
2: Assign each harmonic to a MPI task
3: **while** {solution q is converged} **do**
4: | Obtain the mass (M), stiffness (K), and damping (C) matrices ▷ Performed in parallel using OpenMP
5: | Obtain the nodal solution, velocities in time using Eq. 5 $(q, \dot{q}, \ddot{q})$
6: | Obtain airloads at all azimuths
7: | Assemble airloads and internal forces to get F ▷ Performed in parallel by each MPI task internally using OpenMP
8: | Artificial Damping: $F = F + C_a\dot{q}$
9: | Fourier decomposition of F in complex domain (F_n) ▷ Performed in parallel using OpenMP
10: | Skyline solve of each harmonic (Eq. 14) ▷ Performed in parallel by each MPI task internally using OpenMP
11: | Convert complex solution q_n to real (q_0, q_{nc}, q_{ns})
12: **end while**

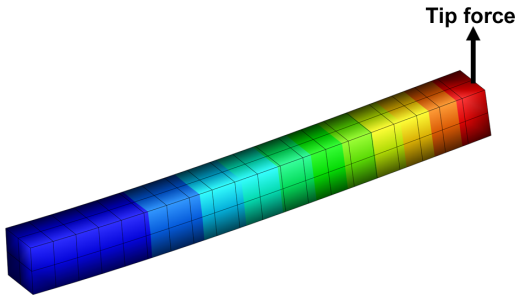


Fig. 10. Deformation of a 3D cantilevered beam with a tip force; test case used for performance study of skyline solver.

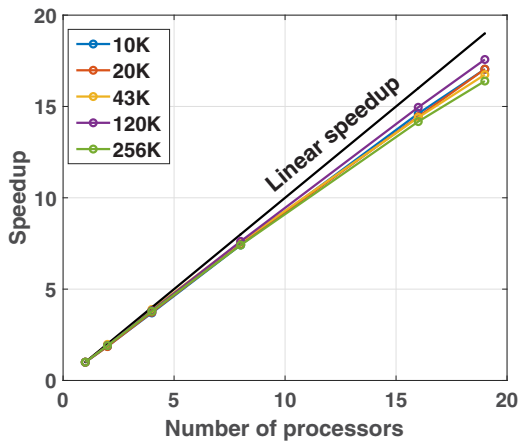


Fig. 11. Speedup of the parallel skyline solver on a single node of 20 processors.

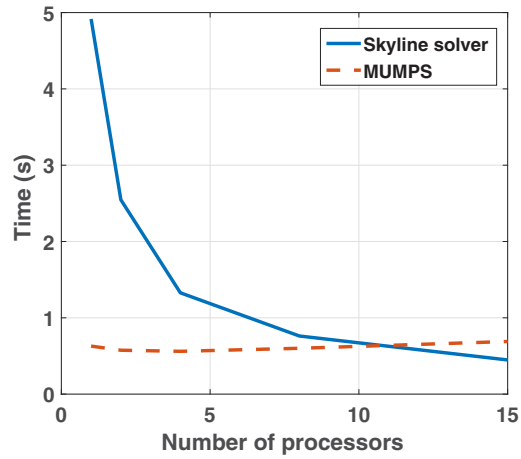
There are several open source sparse direct solvers available to solve large systems of finite element equations. A widely used solver is the MUMPS (Multifrontal Massively Parallel sparse direct Solver) (Refs. 26, 27). It implements the multifrontal method in parallel using MPI-OpenMP and uses BLAS and ScaLAPACK kernels for dense matrix computations. The performance of the solver developed here is compared against MUMPS on a local desktop with a maximum of 16 processors and similar specifications as Deepthought2. The deflection of the UH-60A rotor with a constant tip force is used as the benchmark problem.

Figure 12(a) shows the variation of solution time versus the number of shared-memory processors. As expected, the solution time reduces for skyline solver with the number of processors, whereas it is constant for MUMPS. Thus, MUMPS is significantly faster in serial mode or when using a low number of processors. But as the number of processors increase, the present solver appears to surpass it.

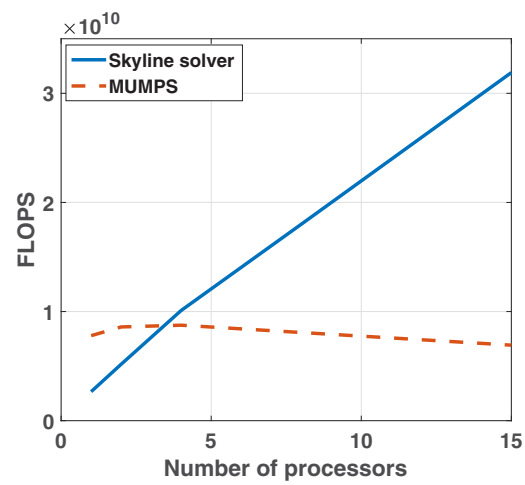
Direct solvers allow for the computation of the number of floating point operations. The number of floating point operations involved in the LDL^T factorization of the skyline solver is $\sum_{j=1}^{N} 1 + h_j(h_j - 1)$ where h_j is the height of column j in the skyline given by Eq. (19). MUMPS also reports the number of operations performed using its multifrontal method. Figure 12(b) shows that the maximum floating point operations per second (FLOPS) is 3 times higher for the present solver when using 15 processors.

Harris Flap Bending Problem

The modified harmonic balance was first studied on an analytical test case given by Harris in Ref. 23. The governing equations are solved for an analytical aerodynamic forcing on a beam. A uniform 3D beam



(a) Time vs number of processors



(b) FLOPS vs number of processors

Fig. 12. Performance comparison of skyline solver and MUMPS for UH-60A problem; (a) solution time versus number of processors, and (b) FLOPS versus number of processors.

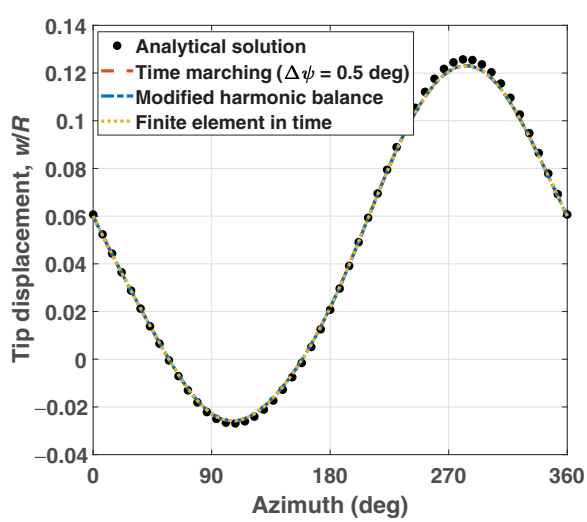
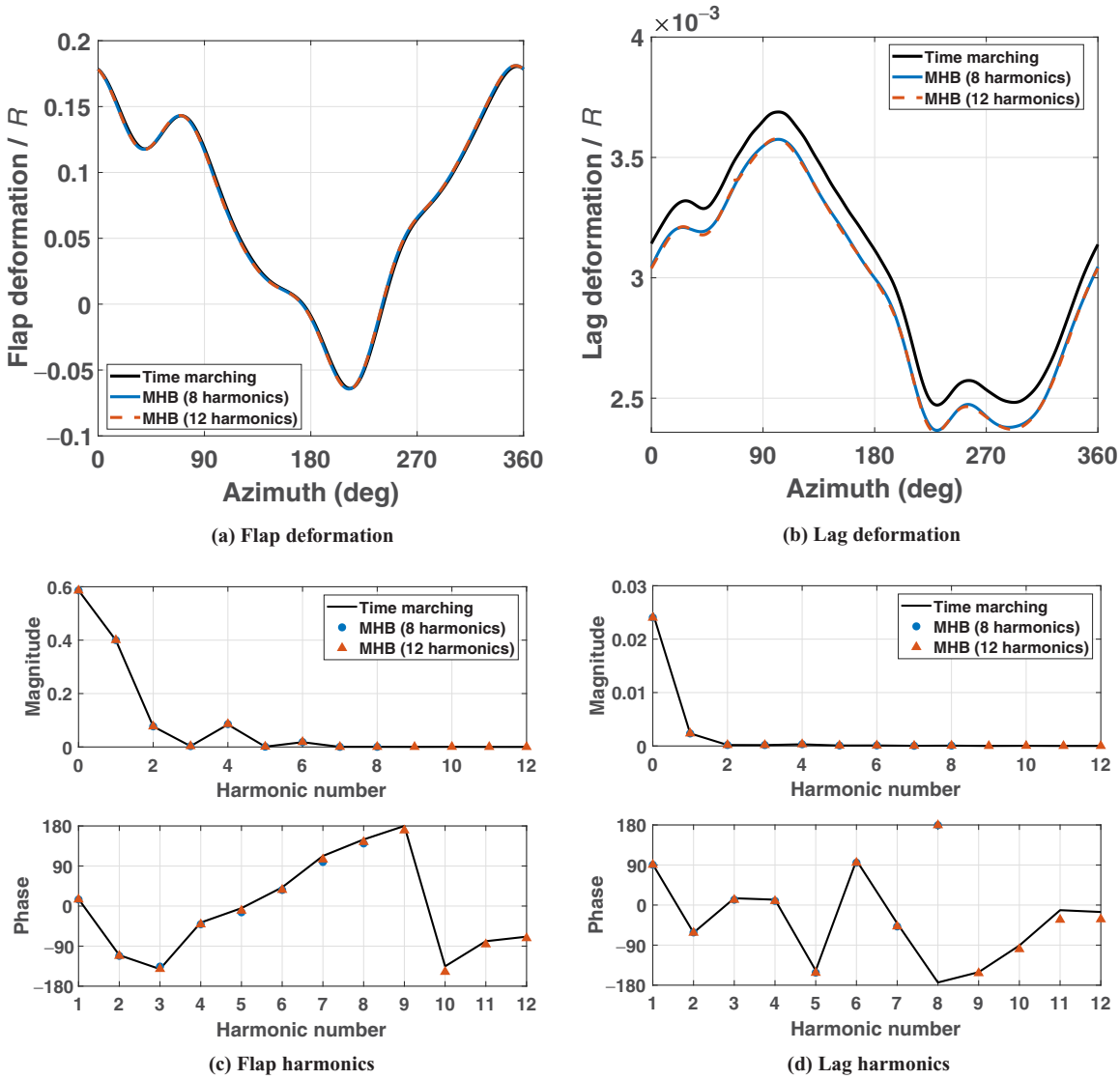


Fig. 13. Tip displacement obtained using different solution procedures for the analytical rotor flap bending case.



(a) Flap deformation

(b) Lag deformation

(c) Flap harmonics

(d) Lag harmonics

Fig. 14. Comparison of flap and lag deformations at the tip for hover with higher harmonic inputs; (a) flap deformation; (b) lag deformation; (c) flap harmonics; and (d) lag harmonics.

with a total of 567 degrees of freedom is used. Figure 13 shows the tip displacement calculated using various methods and verified with analytical solution.

The number of degrees of freedom was purposefully kept small so that finite element in time could also be employed. Figure 13 verifies all three numerical methods provide the same solution and all match the analytical solution. The time marching solution was obtained for different azimuth steps ranging from 10° to 1°. The solution achieves 5% accuracy with $\Delta\psi = 10°$, 2% with $\Delta\psi = 5°$, 0.5% with $\Delta\psi = 1°$, and 0.1% with $\Delta\psi = 0.5°$. The solution for $\Delta\psi = 0.5°$ is shown in Fig. 13.

The UH-60A-like rotor

Hover with harmonics

First, the test case 3a: ideal hover with higher harmonic inputs is studied. There is no interharmonic coupling in hover, only harmonics of excitation are expected in the solution. The solution was obtained with 8 and 12 harmonics. Figures 14(a) and 14(b) compare the flap and lag deformations (at tip 1/4 chord) with a converged time marching solution obtained with Newmark with $\Delta\psi = 1°$. The harmonic breakdown in Figs. 14(c) and 14(d) show that as expected, only steady, 1, 2, 4, and 6 /rev harmonics are present in the solution. The steady error in lag is not understood, but likely a numerical artifact from the time marching solution. The lag is in general too small in this case. The phase error for harmonic 8 is merely a 360° wrap around.

Forward flight: Fixed controls

The solver was next tested in forward flight ($\mu = 0.37$), still with fixed controls (Test case 3b), but now interharmonic couplings are introduced naturally due to asymmetry in the flow. The high-speed case was chosen for maximum asymmetry. The solution was again obtained with 8 and 12 harmonics. Figures 15(a) and 15(b) compare the flap and lag deformations (at tip 1/4 chord) with a converged time marching solution. The harmonic breakdown in Figs. 15(c) and 15(d) show minor differences in the phase of higher harmonics. Overall, the results verify the accuracy of the modified harmonic balance.

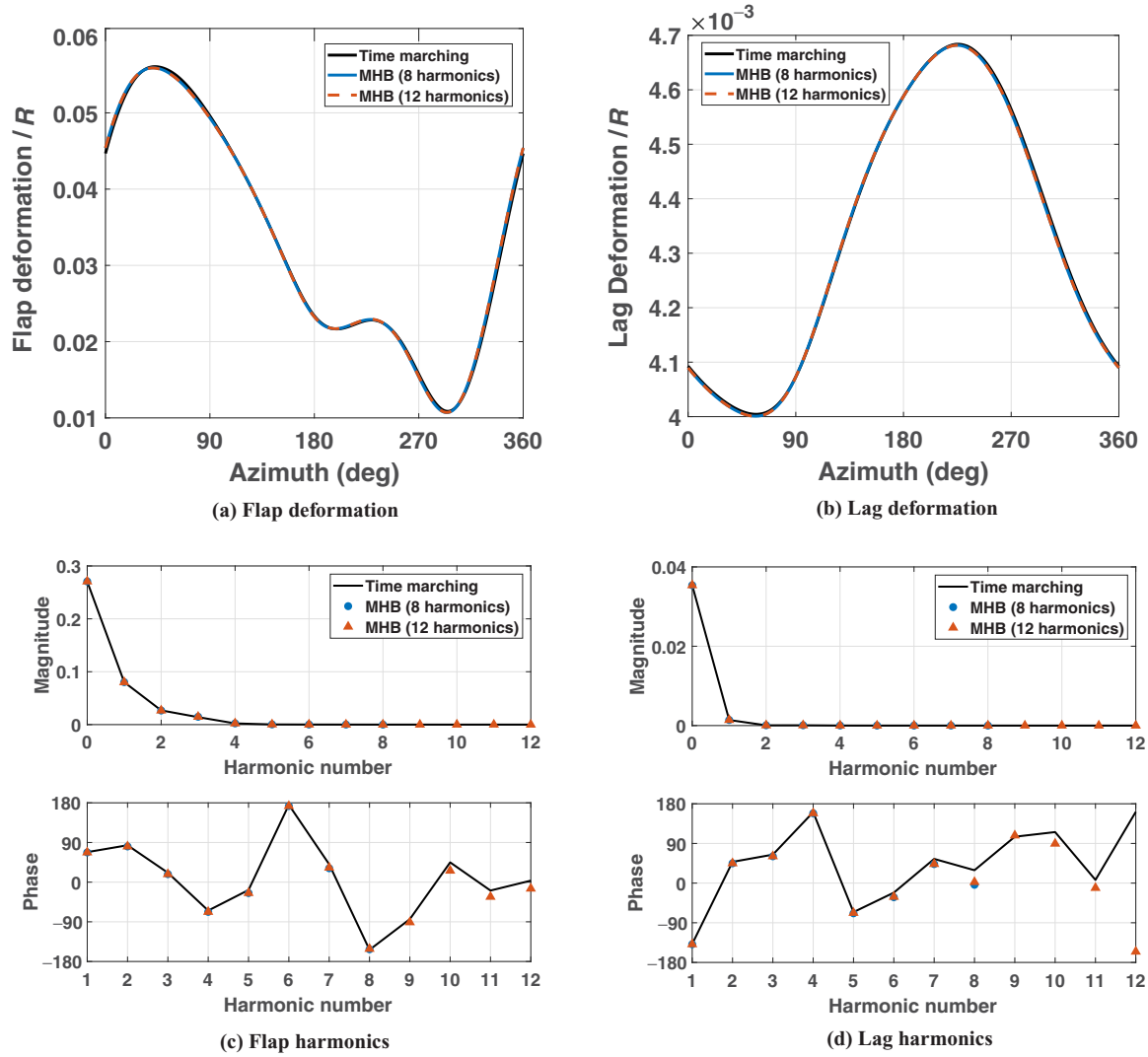


Fig. 15. Comparison of flap and lag deformations at the tip in forward flight with fixed control inputs; (a) flap deformation; (b) lag deformation; (c) flap harmonics; and (d) lag harmonics.

Forward flight: Trim solution

The final test case is a low-speed transition flight: counter 8513 of the UH-60A Black Hawk Airloads Program. The conditions are $C_T/\sigma = 0.076$, $\mu = 0.15$, $\alpha_s = -3.75^\circ$. The wake roll up and intertwining effects dominate the airloads at this condition and cause high vibratory loads. The rich harmonic content in airloads makes this an ideal case for validation. The free-wake model is a fully rolled-up single-tip vortex model with no inboard wake.

Figure 16 shows the measured and predicted normal forces at different radial stations along the azimuth. Figure 17 shows the vibratory harmonics (3,4,5 /rev) versus span. This level of accuracy is generally the state of the art with lifting-line free wake models. However, search for accuracy is not the objective here. The key conclusion is that the modified harmonic balance produces the same results as time marching. A converged time marching solution with $\Delta\psi = 5^\circ$ is used for the assessment of convergence. The small differences at the tip are an artifact of numerical deviations in time marching, not from the scheme, but really the airfoil tables.

The distribution of axial and transverse shear strain in vertical direction obtained at the azimuth, $\psi = 245^\circ$ is shown in Figs. 18 and 19. The harmonic balance solver appears to capture all the localized strain patterns of the internal structure with a similar accuracy as time marching, except in the foam behind the spar.

Performance Study of Modified Harmonic Balance

The time histories of flap, lag, and torsional deformations at the tip in forward flight with fixed control inputs (starting from zero) with $\Delta\psi = 1^\circ$ are shown in Fig. 20. As expected, time marching requires many rotor revolutions before periodicity is attained particularly in lag, which has little to no aerodynamic damping. The flap and torsion modes converge in 10 revolutions whereas lag reaches 5% accuracy in 10, 0.5% accuracy in 20 revolutions. The same level of accuracy in modified harmonic balance requires 25–30 iterations (k in Eq. (24)). Figure 21 shows the convergence of harmonics with the iteration number. A typical solution reaches 1% accuracy in 10 iterations, 0.01% in 30, and 0.0001% in 40 iterations. What time marching achieves with small azimuth step and a large number of revolutions, the modified harmonic balance achieves directly, albeit with iterations.

The solution times and speedup for fixed controls and trim solution cases are summarized in the Table 1. The times for the time marching

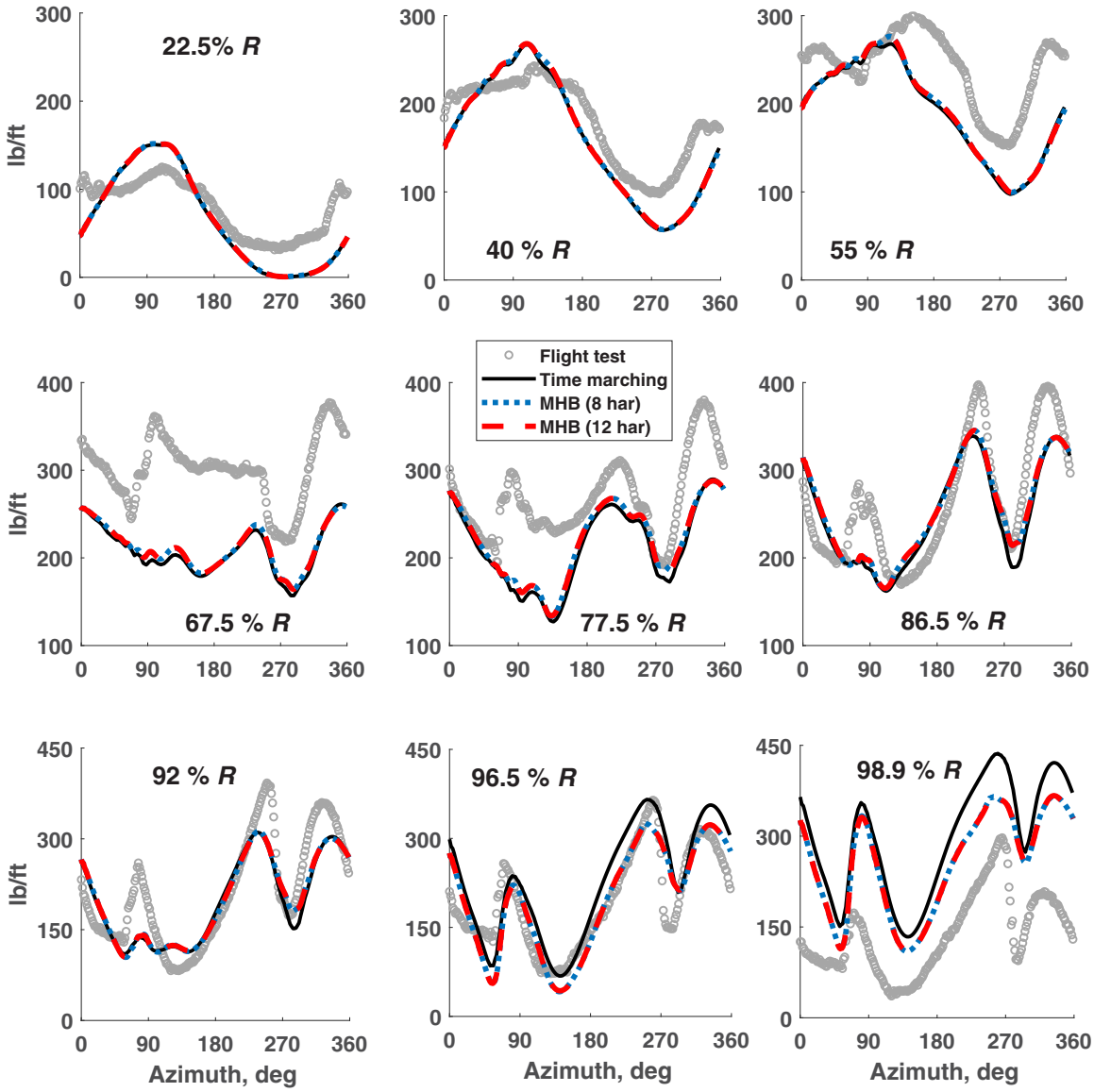


Fig. 16. Normal force from modified harmonic balance method using 8 and 12 harmonics; predictions compared with time marching solution and measured UH-60A airloads.

Table 1. Solution times for serial and parallel time marching and modified harmonic balance for UH-60A-like rotor test case; parallel execution on combined shared and distributed memory architecture.

Solution Procedure	Mode	Fixed Controls	Trim Solution	Speedup
Time marching	Serial	12 min	100 min	1
	Parallel	2 min	16 min	6
Modified harmonic balance	Serial	2.5 min	10 min	10
	Parallel	20 s	2 min	50

solution was obtained with $\Delta\psi = 5^\circ$. The time marching solution on a single processor for a trim solution takes 100 min. The parallel skyline, but still with time marching, reduces it to 16 min, a speedup of around 6. Replacing time marching with the modified harmonic balance on a single processor with no parallelization produces a speedup of 10. When implemented in parallel (9 MPI tasks, 0–8 harmonics) with each using 10 shared-memory processors internally, the time drops to 2 min, achieving a total speedup of 50. This massive speedup is a result of the combined effect of the parallel skyline and inherent parallel nature of the modified harmonic balance.

A total of $N + 1$ MPI tasks are launched for obtaining solution with N harmonics $(0, 1, 2, \ldots, N)$. With each MPI task assigned with 10 shared memory processors, a total of $10 \times (N + 1)$ processors (both MPI and OpenMP) are employed. The solution time for trim solution remains almost constant with an increase in number of harmonics. Thus, a finer harmonic resolution is possible with no increase in solution time

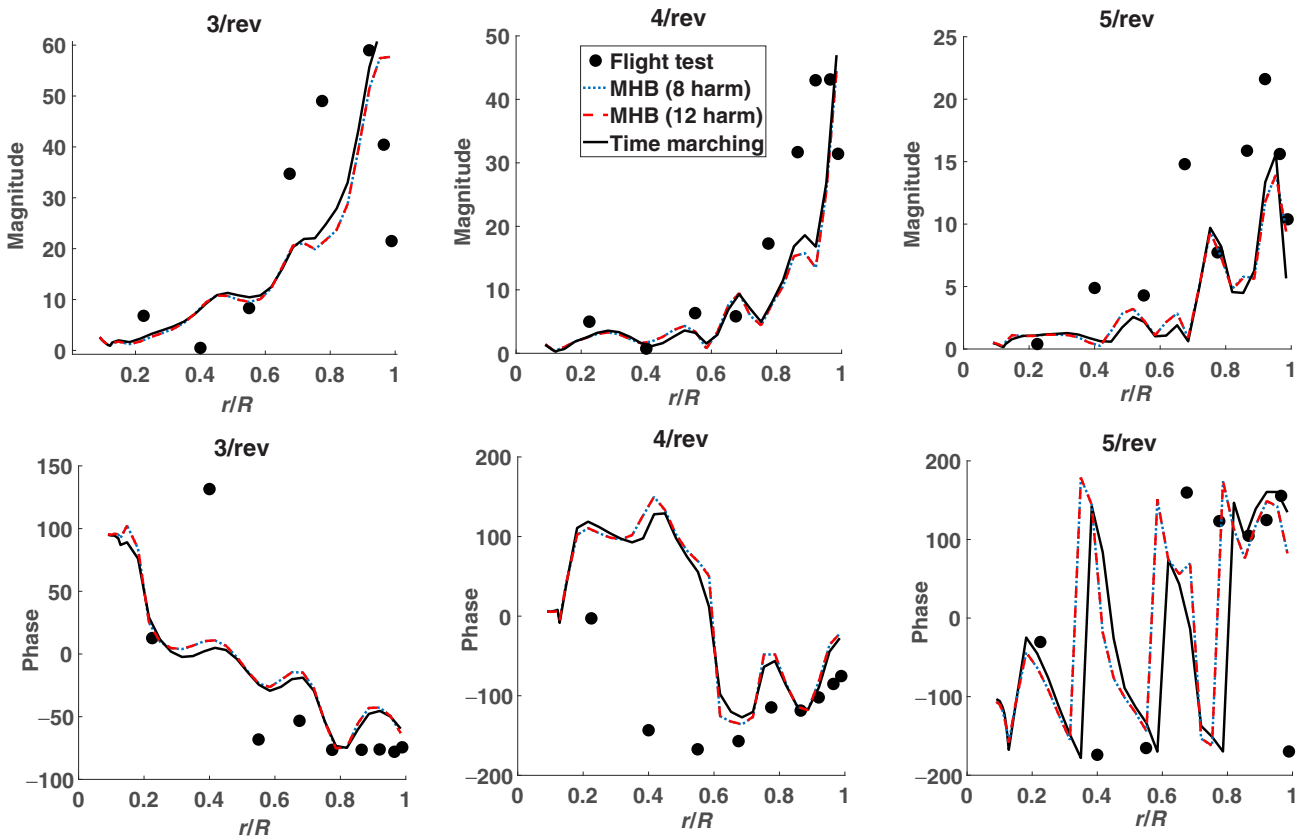


Fig. 17. Normal force vibratory harmonics from modified harmonic balance using 8 and 12 harmonics; predictions compared with UH-60A airloads for qualitative comparison.

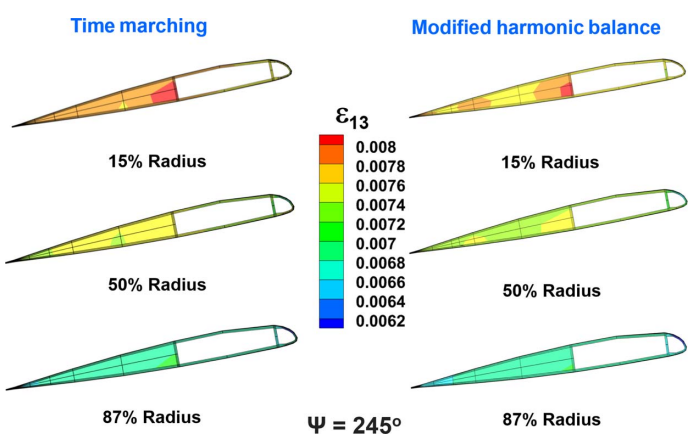


Fig. 18. Distribution of axial strain obtained using time marching and modified harmonic balance methods for different radial stations at $\psi = 245°$.

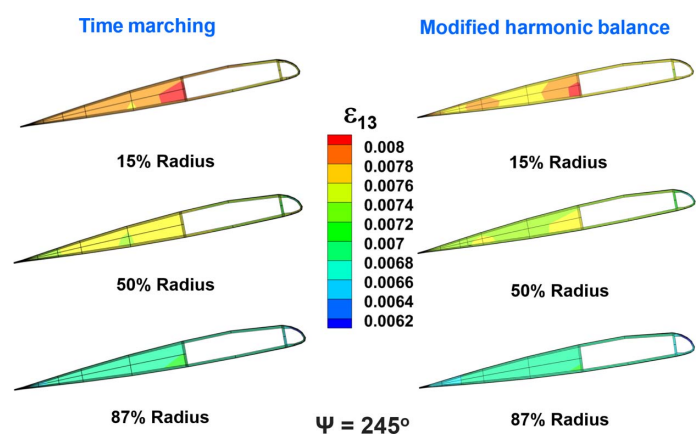


Fig. 19. Distribution of transverse shear strain in vertical direction obtained using time marching and modified harmonic balance methods for different radial stations at $\psi = 245°$.

if more processors are available. Figure 22 shows the trim solution time versus number of harmonics and processors. For example, a total of 90 processors are used to solve for 0, 1, 2, . . . , 8 harmonics. Figures 22 and 11 prove, respectively, the weak and strong scalability of the algorithm.

Figure 23 shows the variation of FLOPS with the number of degrees of freedom for time marching and modified harmonic balance. The 3D cantilevered beam with sinusoidal varying tip force is used as a test case for obtaining the FLOPS performance (Test case 1b). A maximum of only 10 gigaFLOPS can be achieved using time marching. On the other hand, the modified harmonic balance reaches higher than 1 teraFLOPS. The key conclusion is that the modified harmonic balance will produce a gain of at least two orders of magnitude in FLOPS compared to time marching.

Conclusions

A Modified Harmonic Balance (MHB) method was developed for large-scale structures tailored to the unique requirements of rotary-wing

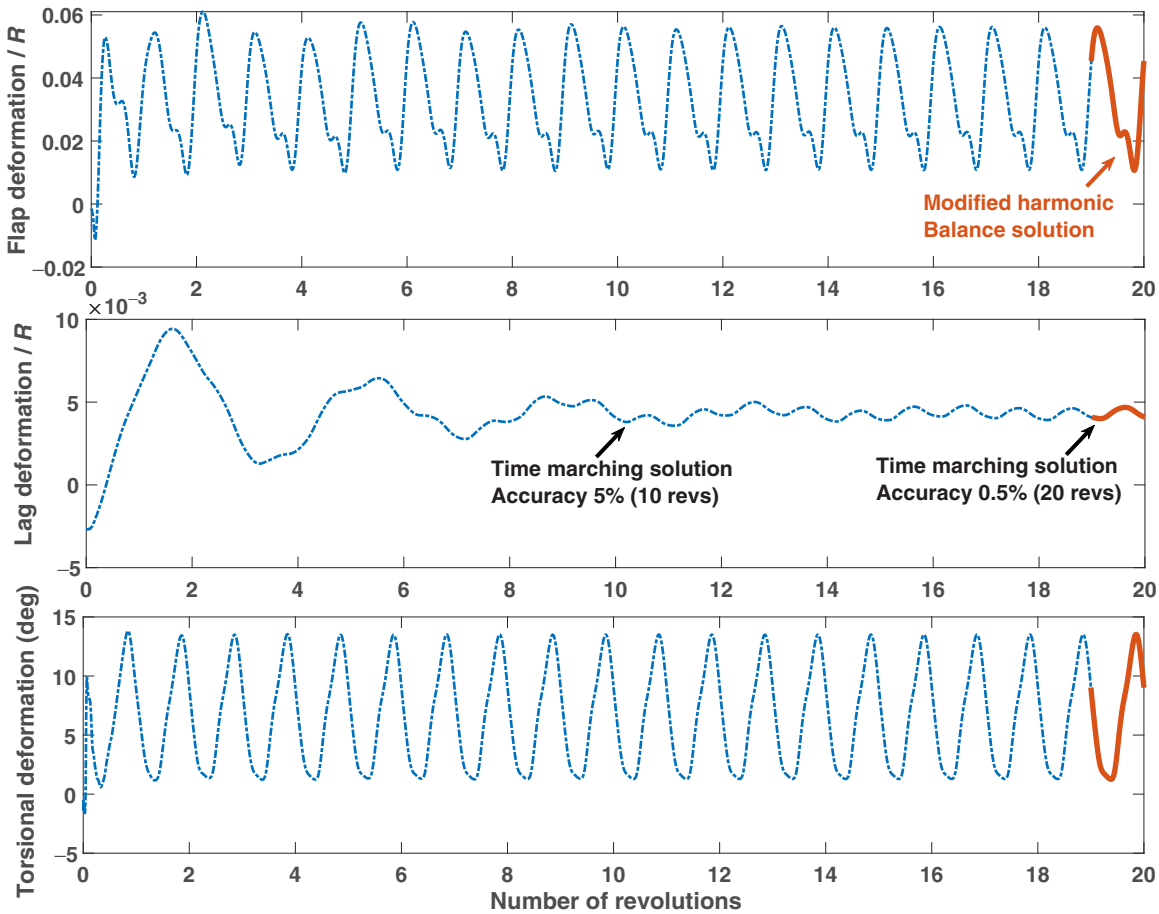


Fig. 20. Time history of flap, lag, and torsional deformation at tip of UH-60A-like rotor in forward flight.

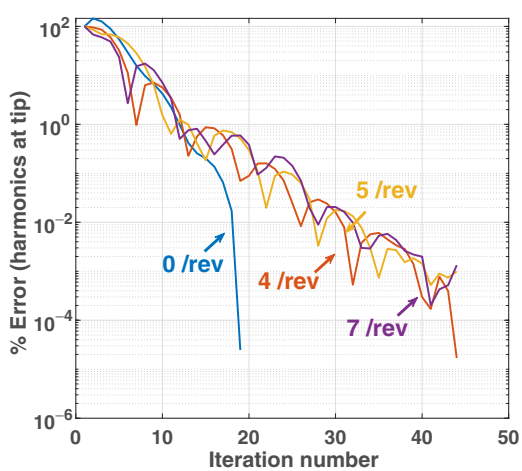


Fig. 21. Convergence of harmonics in the solution at tip with iterations.

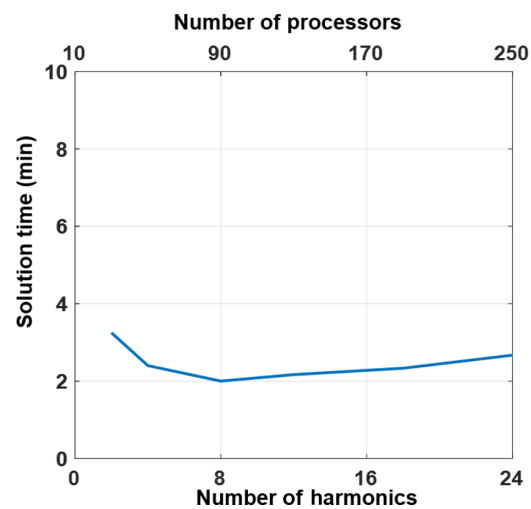


Fig. 22. Trim solution time versus number of harmonics and processors; a total of 10 × $(N + 1)$ processors are used to solve for 0, 1, 2, . . . , N harmonics; solution time remains almost constant an with increase in number of harmonics and processors.

dynamics. For large-scale structures, current methods were either found infeasible due to overwhelming matrix size (finite elements in time) or unscalable due to breakdown of matrix structure (harmonic balance). The modified harmonic balance is formulated in the complex domain built upon a scalable skyline solver and implemented on hybrid shared and distributed memory computer architecture. The algorithm is implemented in X3D. The accuracy was verified by comparison with the Frank Harris problem and to converged time marching and finite element in time solutions. Accuracy was validated with UH-60A flight-test Counter 8513 airloads data. Scalability with the number of processors

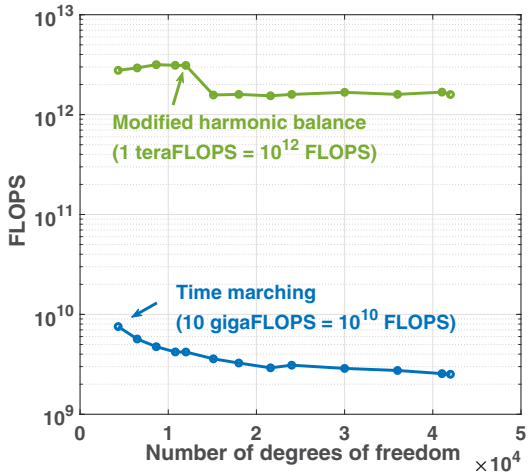


Fig. 23. Variation of FLOPS versus the number of degrees of freedom for time marching and modified harmonic balance.

and a peak performance of 1 teraFLOPS were observed. Based on this work, the following key conclusions are drawn:

1) Parallel and scalable extraction of periodic dynamics is possible for large-scale rotor dynamic models. But current algorithms will not do. Of the two widely used algorithms, the finite element in time is infeasible due to its overwhelming matrix size and harmonic balance is unscalable due to skyline breakdown of matrix. However, modifications to the harmonic balance are possible that circumvent these problems.

2) A Modified Harmonic Balance (MHB) formulated in the complex domain can retain the original size and skyline. The kernel of the algorithm then becomes a parallel solver for the skyline. This can be devised on a shared memory architecture deliberately designed for speed. The parallel skyline is then unleashed on each harmonic, and the harmonics are solved independently on distributed memory.

3) The modified harmonic balance is verified to converge to the same time marching solution. The error drops to 1% with 10 iterations, 0.01% with 20, and 0.0001% with 40 iterations. This level of accuracy and convergence is demonstrated on both idealized and practical rotor problems.

4) The solver was validated on an UH-60A-like rotor in a wake-dominated low-speed transition flight and showed consistent levels of accuracy with time marching for airloads, sectional deformations, and 3D blade stresses.

5) The performance results show a 50 times speedup compared to time marching in serial and at least 8 times speedup compared to time marching in parallel. The speedup progression for the trim solution is as follows: between time marching in serial to time marching in parallel, there is a speedup of 6. Between time marching in serial to modified harmonic balance in serial, there is a speedup of 10. Thus, the modified harmonic balance in parallel provides a net speedup of 50 from time marching in serial.

6) The method exhibits scalability with processors as well as with size. A peak performance of 1 teraFLOPS was recorded for a model problem with structures alone.

In summary, it appears that the principal barrier of computational time for trim solution using high-fidelity 3D structures on realistic rotor problems can be overcome with the modified harmonic balance solver presented here. The exceptional performance achieved from the solver can have a significant impact on the next-generation rotorcraft analysis, allowing for routine use of 3D structures for advanced rotorcraft design.

Acknowledgments

This work was carried out at the Alfred Gessow Rotorcraft Center, University of Maryland at College Park, under the Army/Navy/NASA Vertical Lift Research Center of Excellence (VLRCOE) grant (number W911W61120012), with technical monitoring from Dr. Mahendra Bhagwat. We wish to thank the technical points of contact: Dr. Wayne Johnson (NASA), Dr. Andrew Wissink (Army), and Dr. Brahmananda Panda (Boeing) for their encouragement and advise.

References

[1]Johnson, W., "A History of Rotorcraft Comprehensive Analyses," NASA/TP–2012–216012, April 2012.

[2]Datta, A., and Johnson, W., "Three-Dimensional Finite Element Formulation and Scalable Domain Decomposition for High Fidelity Rotor Dynamic Analysis," *Journal of the American Helicopter Society*, Vol. 56, (2), April 2011, pp. 1–14.

[3]Datta, A., and Johnson, W., "Integrated Aeromechanics with Three-Dimensional Solid-Multibody Structures," Proceedings of the 70th Annual Forum of the Vertical Flight Society, Montreal, Quebec, May 20–22, 2014.

[4]Datta, A., "X3D – A 3D Solid Finite Element Multibody Dynamic Analysis for Rotorcraft," Proceedings of the American Helicopter Society Technical Meeting on Aeromechanics Design for Vertical Lift, San Francisco, CA, January 20–22, 2016.

[5]Staruk, W., Datta, A., Chopra, I., and Jayaraman, B., "An Integrated Three-Dimensional Aeromechanics Analysis of the NASA Tilt Rotor Aeroacoustic Model," *Journal of the American Helicopter Society*, Vol. 63, (3), July 2018, pp. 1–12.

[6]Ward, E., Chopra, I., and Datta, A., "Rotation Frequency-Driven Extension-Torsion Coupled Self-Twisting Rotor Blades," *Journal of Aircraft*, Vol. 55, (5), September 2018, pp. 1–13.

[7]Blatov, V. A., and Proserpio, D. M., "Periodic-Graph Approaches in Crystal Structure Prediction," *Modern Methods of Crystal Structure Prediction*, Chap. 1, 2011, pp. 1–28.

[8]Aprille, T., and Trick, T., "Steady-state Analysis of Nonlinear Circuits with Periodic Inputs," *Proceedings of the IEEE*, Vol. 60, (1), January 1972, pp. 108–114.

[9]Meyer, K. R., "Periodic Solutions of the N-body Problem," *Journal of Differential Equations*, Vol. 39, (1), January 1981, pp. 2–38.

[10]Panda, B., and Chopra, I., "Flap-Lag-Torsion Stability in Forward Flight," *Journal of the American Helicopter Society*, Vol. 30, (4), October 1985, pp. 30–39.

[11]Argyris, J., and Scharpf, D., "Finite Elements in Time and Space," *Nuclear Engineering and Design*, Vol. 10, (4), December 1969, pp. 456–464.

[12]Borri, M., "Helicopter Rotor Dynamics by Finite Element Time Approximation," *Computers and Mathematics with Applications*, Vol. 12, (1), January 1986, pp. 149–160.

[13]Peters, D., and Hou, L. J., "hp-version Finite Elements for the Space-Time Domain," *Computational Mechanics*, Vol. 3, (2), March 1988, pp. 73–88.

[14]Panda, B., and Chopra, I., "Dynamics of Composite Rotor Blades in Forward Flight," *Vertica*, Vol. 11, (1), November–December 1987, pp. 187–210.

[15]Dull, A. L., and Chopra, I., "Aeroelastic Stability of Bearingless Rotors in Forward Flight," *Journal of the American Helicopter Society*, Vol. 33, (4), October 1988, pp. 38–46.

[16]Choi, S., Datta, A., and Alonso, J. J., "Prediction of Helicopter Rotor Loads Using Time-Spectral Computational Fluid Dynamics and an Exact Fluid-Structure Interface," *Journal of the American Helicopter Society*, Vol. 56, (4), October 2011, pp. 1–15.

[17]Mcmullen, M., Jameson, A., and Alonso, J. J., "Demonstration of Non-linear Frequency Domain Methods," *Journal of Aircraft*, Vol. 44, (7), July 2006, pp. 1428–1435.

[18]Ekici, K., Hall, K. C., and Dowell, E. H., "Computationally Fast Harmonic Balance Methods for Unsteady Aerodynamic Predictions of Helicopter Rotors," *Journal of Computational Physics*, Vol. 227, (12), June 2008, pp. 6206–6225.

[19]Peters, D. A., and Ormiston, R. A., "Flapping Response Characteristics of Hingeless Rotor Blades by a Generalized Harmonic Balance Method," NASA TN D-7856, February 1975.

[20]Lions, J., Maday, Y., and Turinici, G., "A Parareal in Time Discretization of PDEs," *Comptes Rendus de l'Académie des Sciences, Série I.*, Vol. 332, (7), April 2001, pp. 661–668.

[21]Adel, B., Boudin, L., and Kaber, S., "Parallel in Time Algorithms with Reduction Methods for Solving Chemical Kinetics," *Communications in Applied Mathematics and Computational Science*, Vol. 5, (2), January 2011, pp. 241–263.

[22]Gurrala, G., Dimitrovski, A., Sreekanth, P., Simunovic, S., and Starke, M., "Parareal in Time for Dynamics Simulations of Power Systems," *IEEE Transactions on Power Systems*, Vol. 31, (3), May 2016, pp. 1820–1830.

[23]Harris, F. D., "Technical Notes: The Rotor Blade Flap Bending Problem - An Analytical Test Case," *Journal of the American Helicopter Society*, Vol. 37, (4), October 1992, pp. 64–67.

[24]Datta, A., Nixon, M., and Chopra, I., "Review of Rotor Loads Prediction with the Emergence of Rotorcraft CFD," *Journal of the American Helicopter Society*, Vol. 52, (4), October 2007, pp. 287–317.

[25]Patil, M., and Datta, A., "A Scalable Time-Parallel Solution of Periodic Rotor Dynamics in X3D," Proceedings of the 76th Annual Forum of the Vertical Flight Society, October 6–8, 2020, Virtual.

[26]Amestoy, P. R., Duff, I. S., Koster, J., and L'Excellent, J.-Y., "A Fully Asynchronous Multifrontal Solver Using Distributed Dynamic Scheduling," *SIAM Journal on Matrix Analysis and Applications*, Vol. 27, (1), 2001, pp. 15–41.

[27]Amestoy, P. R., Buttari, A., L'Excellent, J.-Y., and Mary, T., "Performance and Scalability of the Block Low-rank Multifrontal Factorization on Multicore Architectures," *ACM Transactions on Mathematical Software*, Vol. 45, (1), 2019, pp. 1–26.