

ABSTRACT

Title of dissertation: A Time Parallel Approach to Numerical Simulation of Asymptotically Stable Periodic Dynamical Systems with Application to CFD Models of Helicopter Rotors

Benjamin S. Silbaugh, Doctor of Philosophy, 2023

Dissertation directed by: Professor James D. Baeder
Department of Aerospace Engineering

Modern High Performance Computing (HPC) machines are distributed memory clusters, consisting of multi-core compute nodes. Engineering simulation and analysis tools must employ efficient parallel algorithms in order to fully utilize the compute capability of modern HPC machines. The trend in Computational Fluid Dynamics (CFD) has been to construct parallel solution algorithms based on some form of spatial domain decomposition. This approach has been shown to be a success for many practical applications. However, as one attempts to utilize more compute cores, limitations in strong scalability are inevitably reached due to a diminishing compute workload per compute core and either fixed or increasing communication cost. Furthermore, spatial domain decomposition approaches cannot be easily applied to mid-fidelity structural dynamics or rigid body dynamics models. A significant majority of industrial fluid and structural dynamic models utilize some form of time marching. Thus, if the domain decomposition strategy may be extended to include the temporal dimension, additional opportunity for increased parallelism

may be realized.

A new form of periodic multiple shooting is proposed that is matrix-free and may be applied to high-fidelity multiphysics models or other high dimensional systems. The proposed methodology is formulated entirely in the time domain. Therefore, existing time-domain simulation tools may utilize the proposed approach to achieve a high degree of distributed memory parallelism without requiring any reformulation. Furthermore, the proposed methodology may be combined with conventional space domain decomposition techniques and other forms of data parallelism to achieve maximal performance on modern HPC architectures. The proposed algorithm retains the iterative shoot-correct approach of conventional periodic shooting methods. However, the correction stage is formulated using a hierarchical evaluation strategy combined with an Arnoldi subspace approximation to eliminate the need for explicit formulation of Jacobian matrices. The local convergence of the proposed method is formally proven for the case of an asymptotically stable dynamical system. The proposed method is numerically tested for a 2D limit cycle problem, a rigid blade helicopter rotor model with quasi-steady aerodynamics and autopilot trim, and an OVERSET CFD model of a helicopter rotor with prescribed elastic blade motions. The method is observed to be convergent in all test cases and found to exhibit good scalability.

The proposed periodic multiple shooting method is a practical means of reducing time-to-solution for numerical simulations of asymptotically stable periodic systems on distributed memory parallel computers. Furthermore, the proposed method may be used to enhance the parallel scalability of OVERSET CFD models of heli-

copter rotors in steady periodic flight.

A Time Parallel Approach to Numerical Simulation of
Asymptotically Stable Periodic Dynamical Systems with Application
to CFD Models of Helicopter Rotors

by

Benjamin Silbaugh

Dissertation submitted to the Faculty of the Graduate School of the
University of Maryland, College Park in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
2023

Advisory Committee:
Professor James Baeder, Chair/Advisor
Professor Inderjit Chopra
Professor Anubhav Data
Professor Roberto Celi
Professor Amr Baz

© Copyright by
Benjamin S. Silbaugh
2023

Table of Contents

1	Introduction	1
1.1	Time Parallelism	3
1.2	Research Objectives	4
1.3	Organization of Dissertation	4
2	A Periodic Multiple Shooting Method for Asymptotically Stable Periodic Systems	6
2.1	Preliminaries	6
2.2	Theory	11
2.3	Algorithms and Data Structures	17
2.3.1	A Serial Algorithm	18
2.3.2	Parallel Algorithms	21
2.4	The Connection to Multiple Shooting with Incomplete Cyclic Reduction	23
2.5	Numerical Tests	31
2.5.1	2D Limit Cycle	31
2.5.1.1	Governing Equations	31
2.5.1.2	Numerical Model	32
2.5.1.3	Results	34
3	A Matrix-Free Periodic Multiple Shooting Method for Asymptotically Stable Periodic Systems	44
3.1	Theory	44
3.1.1	Approximating $DP_{i+1,i}$ Using Subspace Approximation	44
3.2	Numerical Tests	46
4	Periodic Multiple Shooting for Rotorcraft Trim	50
4.1	The Trim Problem	50
4.2	State-of-the-Art Helicopter Trim Theory and Practice	50
4.2.1	Mathematical Formulations of Helicopter Trim	51
4.2.1.1	System of ODE's with Integral Constraints	51
4.2.1.2	Explicit System of DAE's with Integral Constraints	53
4.2.1.3	Hamiltonian Formulations with Integral Constraints	53

4.2.1.4	Semi-Discrete Formulations with Integral Constraints	54
4.2.1.5	Systems with Hidden States	54
4.2.1.6	Under-determined systems	55
4.2.2	Existence, Uniqueness, and Solvability	55
4.2.3	Trim Methods and Algorithms	56
4.2.3.1	Closed Form Control Relations	56
4.2.3.2	Newton-like Methods with Time Marching	57
4.2.3.3	Periodic Shooting	61
4.2.3.4	Fast Floquet	65
4.2.3.5	Harmonic Balance	65
4.2.3.6	Finite Element in Time	71
4.2.3.7	Auto Pilot	71
4.2.3.8	Hybrid Schemes	77
4.2.3.9	Optimal Trim	79
4.2.4	Trim of Non-Periodic Systems	80
4.2.5	The Delta-Coupling Method	80
4.3	Theory	81
4.3.1	Equivalent Autonomous First Order System	82
4.3.1.1	Treatment of Periodic Forcing Terms	82
4.3.1.2	Treatment of Trim Constraints	86
4.3.2	Initial Solution Estimates	87
4.3.3	Exploiting Symmetries	87
4.4	Numerical Tests	88
4.4.1	Rigid Bladed Rotor with Flap DOF	88
4.4.2	Rotor System Model and Flight Condition	88
4.4.2.1	Mathematical Model	89
4.4.2.2	Numerical Model	92
4.4.2.3	Test Results	93
5	Application of Periodic Multiple Shooting Method to CFD Rotor Models	115
5.1	Theory	116
5.1.1	Reduction of PDE to ODE using Method of Lines	116
5.1.2	Methodology	116
5.1.3	Results	121
6	Summary and Conclusions	125
	Bibliography	127

Chapter 1: Introduction

Computers have seen widespread use as a tool to facilitate engineering analysis. The earliest simulations were carried out by special purpose mechanical devices. These mechanical computers were then replaced with analog electric circuits which could execute common mathematical operations which ranged from basic arithmetic to basic calculus. As the drive to realize more sophisticated simulation and analysis capabilities pushed analog computing methods to their limits, the capabilities of digital computers reached sufficient maturity to displace analog computing methods. Nowadays, the word “computing” has become synonymous with “digital computing”.

The transition from analog to digital computing was not entirely seamless, and drove a shift in mathematical methods and algorithm design. Since digital computers can only perform basic arithmetic (and logic operations), mathematical models based on differential or integral equations that could once be directly evaluated by an analog computer, now required reformulation into an approximate discrete system that could be evaluated using simple arithmetic. The rise of the digital computer drove advances in numerical methods.

A computer program is effectively a sequence of floating point operations in-

terleaved with memory load and store operations, and other required IO operations. The amount of time required for a program to complete is a function of how quickly a computer can execute these operations. For numerical simulations, the number of Floating Point Operations per Second (FLOPS) that a machine is capable of plays a significant role in determining a machine's performance. However, as the speed of Central Processing Units (CPUs) have progressively increased, the rate at which data can be retrieved from memory also plays a significant role in overall computer performance. The need for increasing effective memory bandwidth combined with thermal and power limitations on CPU speed has led to the emergence of parallel computer architectures. The wide spread adoption of parallel computers has spurred the development of parallel numerical algorithms. Just as the transition from analog to digital computing required one to rethink how they utilize computers to solve engineering problems, so has the transition from serial computers to parallel computers.

Modern High Performance Computing (HPC) machines are distributed memory clusters, consisting of multi-core compute nodes. Engineering simulation and analysis tools must employ efficient parallel algorithms in order to fully utilize the compute capability of modern HPC machines. The trend in Computational Fluid Dynamics (CFD) has been to construct parallel solution algorithms based on some form of spatial domain decomposition. This approach has been shown to be a success for many practical applications. However, as one attempts to utilize more compute cores, limitations in strong scalability are inevitably reached due to a diminishing compute workload per compute core and either fixed or increasing communication

cost. Furthermore, spatial domain decomposition approaches cannot be easily applied to mid-fidelity structural dynamics or rigid body dynamics models. A significant majority of industrial fluid and structural dynamic models utilize some form of time marching. Thus, if the domain decomposition strategy may be extended to include the temporal dimension, additional opportunity for increased parallelism may be realized.

1.1 Time Parallelism

The concept of time parallel computation is not completely new. Gopinath et. al. [1] proposed the use of time-spectral methods for time-periodic turbulent Navier-Stokes calculations past two and three dimensional bodies. A Fourier representation in time was used to exploit the periodic behavior and facilitate parallel computation in the frequency domain. Spectral convergence was demonstrated for a pitching airfoil and pitching wing. The spectral parallelism was not combined with spatial decomposition.

Choi et. al. [2] investigated the application of time spectral methods to helicopter rotor flows in steady flight. The accuracy of time-spectral methods for this application was assessed by comparing time-spectral predictions of vibratory loads against a “time-accurate” prediction. The study included fully coupled CFD/CSD model of a helicopter rotor. They identified that the primary source of error was frequency aliasing, and that it is possible to reduce this error by increasing the number of time instances. It was shown that it was possible to reduce computational wall

time by as much as one-fifth. However, accurately resolving sharp impluses (e.g. such as that resulting from unsteady shock movement or dynamic stall) required significantly increasing the number of time elements, which in turn degraded the cost savings.

1.2 Research Objectives

The objective of this work is to develop a new method for time-parallel numerical simulation of asymptotically stable dynamical systems. The new method ought to be applicable to high dimensional systems, including multi-physics simulations. It is desirable to formulate the methodology in the time-domain, as many state-of-the-art multi-physics simulation tools utilize time marching techniques. Finally, it ought to be possible for the methodology to be combined with established space domain decomposition techniques and other forms of data parallelism to maximize performance on modern HPC architectures.

1.3 Organization of Dissertation

Chapter 2 will introduce the proposed periodic multiple shooting method, and develop the supporting mathematical theory in the context of asymptotically stable periodic dynamical systems. Chapter 3 introduced an enhancement of the proposed periodic multiple shooting method that leverages a subspace approximation technique to eliminate the need to construct and evaluate matrix operators. Chapter 4 presents an adaptation of the multiple shooting method for a trimmed helicopter

rotor, and demonstrates its utility for a rigid blade rotor model with quasi-steady aerodynamics. An adaptation of the periodic multiple shooting method for overset URANS CFD models of helicopter rotors is presented in Chapter 5 and demonstrated for a high speed flight case with prescribed elastic deformations. Summary and conclusions are given in Chapter 6.

Chapter 2: A Periodic Multiple Shooting Method for Asymptotically Stable Periodic Systems

2.1 Preliminaries

The theoretical framework of the method of Periodic Shooting for asymptotically stable periodic systems is built upon the theory of Dynamical Systems. In the interest in being self contained, the essential concepts, definitions, and theorems from dynamical systems are reviewed here. For a more systematic development of these ideas the reader may wish to read Hirsch [3] and Wiggins [4]. Given that the ultimate goal is to develop a solution method for numerical models, it will suffice to consider dynamical systems on open subsets of Euclidean spaces ($\mathbb{R}^n$ with the Euclidean metric). For an introduction to the more general case of dynamical systems on manifolds, see Arnold [5].

Informally, a dynamical system may be thought of as a system that changes or evolves with time. A dynamical system embodies the concept of identity in the sense that it may change from one moment to the next and yet be considered the same object. The state of the system is a quantity (a vector of numbers) that encodes all of the necessary information required to determine the future and past history of

the system, together with a law that governs the advancement from one state to the next. The phase space of a dynamical system is the set of all possible states that the dynamical system can rationally posses. Mathematically, a dynamical system is defined as follows [4].

Definition 1. *Let $\mathcal{S}$ be an open subset of a Euclidean space. A "dynamical system" is a C^1 map $\phi : \mathbb{R} \times \mathcal{S} \rightarrow \mathcal{S}$ such that the maps of the form $\phi_t : \mathcal{S} \rightarrow \mathcal{S}$, each defined by the equation*

$$\phi_t(x) = \phi(t, x), \quad \forall x \in \mathcal{S},$$

satisfy the following:

1. $\phi_0 : \mathcal{S} \rightarrow \mathcal{S}$, is the identity map,
2. The composition $\phi_t \circ \phi_s = \phi_{t+s}$ for all $t, s \in \mathbb{R}$.

The set $\mathcal{S}$ is called the "phase space."

The map ϕ may be viewed as the law that governs the evolution of a particular dynamical system from one state to another over a time of t .

The following proposition from Hirsch [3] will prove useful.

Proposition 1. *The map $\phi_t : \mathcal{S} \rightarrow \mathcal{S}$ is C^1 for each t and has a C^1 inverse.*

The concept of a vector field provides a useful way to geometrically visualize and qualitatively reason about the evolution of a dynamical system. It also provides an important link to the classical notion of a differential equation.

Definition 2. *A "vector field" on $\mathcal{S}$ is a map $f : \mathcal{S} \rightarrow E$, where E is some vector space and $\mathcal{S}$ is an open subset of E .*

As shown by Hirsch [3], every dynamical system corresponds to a differential equation of the form

$$\dot{x} = f(x) \tag{2.1}$$

where $f \in C^k(U)$, $k \geq 1$, for some open subset U of $\mathbb{R}^n$. Furthermore, every differential equation of the form (2.1) corresponds to a dynamical system. Differential equations of the form (2.1) define a vector field on the phase space. The map ϕ_t of a dynamical system is a solution of (2.1). Wiggins [4] considers dynamical systems and differential equations of form (2.1) as one in the same. Hirsch [3], on the other hand, considers dynamical systems and differential equations to be distinct mathematical objects related by way of the map ϕ_t . If one only considers dynamical systems on open subsets of Euclidean spaces, then the two viewpoints are interchangeable.

Three geometric concepts are essential to any discussion on the behavior of a dynamical system ϕ : trajectory, integral curve, and orbit.

Definition 3. *The “trajectory of ϕ through point x_0 at time t_0 ” is the function defined by $(t, t_0, x_0) \mapsto \phi(t - t_0, x_0)$.*

Definition 4. *The “integral curve of ϕ passing through point x_0 at time t_0 ”, is the set $\{(x, t) \in \mathcal{S} \times \mathbb{R} : x = \phi(t - t_0, x_0), t \in I\}$, where I is the largest interval, containing t_0 , where $\phi(t - t_0, x_0)$ is defined.*

Definition 5. *The “orbit of ϕ through point x_0 ” is the set $\{x \in \mathcal{S} : x = \phi(t, x_0), t \in I\}$, where I is the largest interval, containing zero, where $\phi(t, x_0)$ is defined.*

Observe that the trajectory of a dynamical system is effectively the solution of the corresponding differential equation with initial condition x_0 at time t_0 . The

integral curve may be thought of as the graph of system states x as a function of time t . An orbit of ϕ is the set of points in the phase space that belong to the same history or evolution of the system represented by ϕ . Sometimes the orbit of ϕ through point x_0 is written $O(x_0)$. By definition $x_0 \in O(x_0)$.

The notion of stability needs to be defined. The most general notion of stability is defined first.

Definition 6. *A trajectory $\bar{x}$ is said to be “Liapunov stable” if, for any $\epsilon > 0$, there exists a $\delta > 0$ such that, for any other trajectory y satisfying $\|\bar{x}(t_0) - y(t_0)\| < \delta$, then $\|x(t) - y(t)\| < \epsilon$ for all $t > t_0$, $t_0 \in \mathbb{R}$.*

Note that δ appearing in the above definition may depend on ϵ . Observe that Liapunov stability does not require neighboring trajectories to converge towards a common trajectory. Rather, it suffices that a neighboring trajectory remains within a neighborhood of radius ϵ . As an example, consider the equilibrium position of a spring-mass system. The phase portrait of such a system is a continuous family of concentric ellipses centered around the equilibrium point $\bar{x}$. For any $\epsilon > 0$, one can identify a displacement δ from equilibrium that will result in a motion whose trajectory in the phase space does not exceed a distance of ϵ from $\bar{x}$.

Definition 7. *A trajectory $\bar{x}$ is said to be “asymptotically stable” if it is Liapunov stable, and for any other trajectory y , there exists a constant $b > 0$ such that, if $\|\bar{x}(t_0) - y(t_0)\| < b$, then $\lim_{t \rightarrow \infty} \|\bar{x}(t) - y(t)\| = 0$.*

Asymptotic stability requires that neighboring trajectories converge towards a common trajectory. An example of an asymptotically stable system is a spring-

mass-damper system.

Definition 8. A trajectory $\bar{x}$ through point x_0 is said to be “periodic of period T ” if there exists a $T > 0$ such that $\bar{x}(t, t_0) = \bar{x}(t + T, t_0)$ for all $t \in \mathbb{R}$.

Definition 9. An orbit of a point $x_0 \in \mathcal{S}$ is said to be “periodic of period $k > 0$ ” if there exists an integer $k > 0$ and a map $g : \mathcal{S} \rightarrow \mathcal{S}$ such that $g^k(x_0) = g(x_0)$.

Note that, given a periodic trajectory $\bar{x}$ of period T , the corresponding orbit in $\mathcal{S}$ traced by $\bar{x}$ is a periodic orbit. This follows by simply taking g in the foregoing definition to be ϕ_T and $k = 1$.

Definition 10. Suppose ϕ is a dynamical system on $\mathcal{S}$, $f \in C^r(\mathcal{S})$ is the vector field, and $O(x_0)$ is a periodic orbit. Let $\mathcal{P}$ be a $n - 1$ dimensional surface, where n is the dimension of $\mathcal{S}$. If $u \in \mathbb{R}^n$ is a vector normal to $\mathcal{P}$ at x_0 , and $f(x_0) \cdot u \neq 0$, then $\mathcal{P}$ is called a “cross section” to the vector field f .

The terms “Poincare section”, “Poincare surface”, or “Poincare plane” are alternative names for cross section.

By continuity of ϕ , one can be assured of the existence of an open set $U \subset \mathcal{P}$ containing x_0 , and an open interval $\mathcal{T} \subset \mathbb{R}$ containing T , such that $\phi_t(x) \in \mathcal{P}$ for any $x \in U$ and $t \in \mathcal{T}$. This justifies the following definition for a Poincare map.

Definition 11. A “Poincare map” is a map $P : U \rightarrow \mathcal{P}$, that satisfies $P(x) = \phi_t(x)$ for any $x \in U$ and $t \in \mathcal{T}$.

In other words, a Poincare map is a map that takes a point near a periodic orbit and on a cross section of the flow, to the point in the same cross section that

corresponds to the first return along the trajectory passing through the original point. In the special case where the original point coincides with the periodic orbit, the Poincare map returns the same point, $P(x_0) = x_0$; i.e. x_0 is a fixed point of P .

Proposition 2. *Suppose ϕ is a dynamical system and $x_0 \in \mathcal{S}$ is a point on a periodic orbit of ϕ . Let $\mathcal{P} \subset \mathcal{S}$ be a cross section of $O(x_0)$ containing x_0 , and $U \subset \mathcal{P}$ an open neighborhood of x_0 such that the Poincare map $P : U \rightarrow \mathcal{P}$ is defined. The orbit $O(x_0)$ is asymptotically stable if and only if the Eigenvalues of $DP(x_0)$ all have negative real components.*

Proof. See Wiggins [4]. □

2.2 Theory

Suppose $\phi : M \rightarrow M$ is a dynamical system on an open subset $M \subset \mathbb{R}^n$, and $x \in M$ is such that the orbit $O(x)$ is periodic and asymptotically stable.

Choose $m \in \mathbb{N}$ distinct cross sections of $O(x)$, and enumerate each section beginning with 0 and incrementing by 1 in the direction of the flow on $O(x)$. Denote the i^{th} cross section by S_i . Let $U_i \subset S_i$ be a small open set that contains $O(x) \cap S_i$; i.e. the intersection of the periodic orbit and the cross section S_i .

Let $P_{i,i+1} : U_i \subset S_i \rightarrow S_{i+1}$ be the map that takes any point y of S_i near the orbit of $O(x)$ to the point in S_{i+1} that is on the same trajectory as y ; informally, $P_{i,i+1}$, “sends” points from U_i to S_{i+1} along their respective trajectories. Denote the derivative of $P_{i,i+1}$ as $DP_{i,i+1}$, and the “derivative of $P_{i,i+1}$ at y ” by $DP_{i,i+1}(y)$. The action of the derivative $DP_{i,i+1}(y)$ on a vector in the tangent plane at y is denoted

by juxtaposition.

Lemma 1. *For any $u, v, y \in U_i$, there is a constant $0 < \alpha < 1$ such that $\|u - v\| = \alpha \|DP_{i,i+1}(y)\| \|u - v\|$.*

Proof. This directly follows from the asymptotic stability of $O(x)$, continuity of $DP_{i,i+1}$, and that U_i is a “sufficiently small” neighborhood. $\square$

Lemma 2. *There is a constant $L > 0$ such that $\|y_1 - y_2\| < L \|DP_{i,i+1}(y_1 - y_2)\|$ for any $y_1, y_2 \in U_i$; i.e. $DP_{i,i+1}$ is Lipschitz continuous.*

Proof. By hypothesis, $DP_{i,i+1}$ is continuous, thus it is also Lipschitz continuous. $\square$

Now, consider the following iterative method for finding the intersections of $O(x)$ with the chosen cross planes S_i $i = 0, \dots, m - 1$:

1. Choose an integer l such that $0 < l < m$.
2. Choose any initial set of points $y_i^0 \in U_i$, $i = 0, \dots, m - 1$.
3. Compute

$$\tilde{y}_i^{k+1} = P_{i-1,i}(y_{i-1}^k), i = 0, \dots, m - 1. \quad (2.2)$$

4. Compute

$$\tilde{r}_i^{k+1} = \tilde{y}_i^{k+1} - y_i^k, i = 0, \dots, m - 1 \quad (2.3)$$

5. Compute

$$r_i^{k+1} = \tilde{r}_i^{k+1} + \sum_{j=1}^l \left(\prod_{j'=1}^j DP_{i-j'+1,i-j'}(y_{i-j'}^k) \right) \tilde{r}_{i-j}^k. \quad (2.4)$$

6. Compute

$$y_i^{k+1} = y_i^k + r_i^k. \quad (2.5)$$

7. Return to step 3

Theorem 1. *Let $l = m - 1$, and m be even. Then for each cross section $S_i, i = 0, \dots, m - 1$, the sequence of points $y_i^0, y_i^1, \dots, y_i^k, \dots$ generated by equations (2.2) through (2.5) converges to the periodic attractor $O(x)$. Furthermore, $\|\tilde{r}^{k+1}\| \leq c_k \|\tilde{r}^k\|$, where $c_k \rightarrow \alpha^m$ as $k \rightarrow \infty$, for an $\alpha < 1$.*

Proof. Substituting equation (2.4) into (2.3),

$$\tilde{r}_i^{k+1} = \tilde{y}_i^{k+1} - y_i^k \quad (2.6)$$

$$= P_{i,i-1}(y_{i-1}^k) - y_i^k \quad (2.7)$$

$$= P_{i,i-1}(y_{i-1}^k) - P_{i,i-1}(y_{i-1}^{k-1}) - \sum_{j=1}^l \left(\prod_{j'=1}^j DP_{i-j'+1,i-j'}(y_{i-j'}^k) \right) \tilde{r}_{i-j}^k \quad (2.8)$$

From multivariable calculus, we have the identity

$$P_{i,i-1}(y_{i-1}^k) - P_{i,i-1}(y_{i-1}^{k-1}) = \int_0^1 DP_{i,i-1}(y_{i-1}^{k-1} + s(y_{i-1}^k - y_{i-1}^{k-1})) ds (y_{i-1}^k - y_{i-1}^{k-1}) \quad (2.9)$$

This allows equation (2.8) to be written as

$$\begin{aligned} \tilde{r}_i^{k+1} = & \int_0^1 DP_{i,i-1}(y_{i-1}^{k-1} + s(y_{i-1}^k - y_{i-1}^{k-1})) ds r_{i-1}^k \\ & - \sum_{j=1}^l \left(\prod_{j'=1}^j DP_{i-j'+1,i-j'}(y_{i-j'}^k) \right) \tilde{r}_{i-j}^k \end{aligned} \quad (2.10)$$

Adding and subtracting $DP_{i,i-1}(y_{i-1}^{k-1})(y_{i-1}^k - y_{i-1}^{k-1})$,

$$\begin{aligned}\tilde{r}_i^{k+1} = & \int_0^1 \{DP_{i,i-1}(y_{i-1}^{k-1} + s(y_{i-1}^k - y_{i-1}^{k-1})) - DP_{i,i-1}(y_{i-1}^k)\} r_{i-1}^k ds \\ & + DP_{i,i-1}(y_{i-1}^k) r_{i-1}^k \\ & - \sum_{j=1}^l \left(\prod_{j'=1}^j DP_{i-j'+1,i-j'}(y_{i-j'}^k) \right) \tilde{r}_{i-j}^k \quad (2.11)\end{aligned}$$

Substituting equation (2.4) into (2.11) an cancelling terms gives

$$\begin{aligned}\tilde{r}_i^{k+1} = & \int_0^1 \{DP_{i,i-1}(y_{i-1}^{k-1} + s(y_{i-1}^k - y_{i-1}^{k-1})) - DP_{i,i-1}(y_{i-1}^k)\} r_{i-1}^k ds \\ & + \left(\prod_{j'=1}^{l+1} DP_{i-j'+1,i-j'}(y_{i-j'}^k) \right) \tilde{r}_{i-l-1}^k \quad (2.12)\end{aligned}$$

Taking norms and applying the triangle inequality gives

$$\begin{aligned}\|\tilde{r}_i^{k+1}\| \leq & \left\| \int_0^1 \{DP_{i,i-1}(y_{i-1}^{k-1} + s(y_{i-1}^k - y_{i-1}^{k-1})) - DP_{i,i-1}(y_{i-1}^k)\} r_{i-1}^k ds \right\| \\ & + \left\| \left(\prod_{j'=1}^{l+1} DP_{i-j'+1,i-j'}(y_{i-j'}^k) \right) \tilde{r}_{i-l-1}^k \right\| \quad (2.13)\end{aligned}$$

Since DP is Lipschitz continuous, an upper bound on the integrated quantity is given by

$$\begin{aligned}\left\| \int_0^1 \{DP_{i,i-1}(y_{i-1}^{k-1} + s(y_{i-1}^k - y_{i-1}^{k-1})) - DP_{i,i-1}(y_{i-1}^k)\} r_{i-1}^k ds \right\| \\ \leq L \|r_{i-1}^k\|^2. \quad (2.14)\end{aligned}$$

Thus,

$$\|\tilde{r}_i^{k+1}\| \leq L \|r_{i-1}^k\|^2 + \alpha^{l+1} \|\tilde{r}_{i-l-1}^k\| \quad (2.15)$$

Substituting equation (2.12) once more, and applying the parallelogram identity

$$\|\tilde{r}_i^{k+1}\| \leq 2L\|\tilde{r}_{i-1}^k\|^2 + 2L\alpha^{l+1}\|\tilde{r}_i^k\|^2 + \alpha^{l+1}\|\tilde{r}_{i-l-1}^k\| \quad (2.16)$$

Since $l = m - 1$, and by periodicity $\tilde{r}_{i-m}^k = \tilde{r}_i^k$,

$$\|\tilde{r}_i^{k+1}\| \leq 2L\|\tilde{r}_{i-1}^k\|^2 + 2L\alpha^m\|\tilde{r}_i^k\|^2 + \alpha^m\|\tilde{r}_i^k\| \quad (2.17)$$

Let $\delta(k) = \max\{\|\tilde{r}_0^k, \dots, \tilde{r}_{m-1}^k\|\}$. Then the following inequality holds:

$$\|\tilde{r}_i^{k+1}\| \leq 2L\delta(k)\|\tilde{r}_{i-1}^k\| + (2L\delta(k) + 1)\alpha^m\delta(k)\|\tilde{r}_i^k\| \quad (2.18)$$

This implies the system of inequalities

$$\begin{bmatrix} \|\tilde{r}_0^{k+1}\| \\ \|\tilde{r}_1^{k+1}\| \\ \vdots \\ \|\tilde{r}_{m-1}^{k+1}\| \end{bmatrix} \leq \begin{bmatrix} a & & b \\ & b & a \\ & & \ddots & \ddots \\ & & & b & a \end{bmatrix} \begin{bmatrix} \|\tilde{r}_0^k\| \\ \|\tilde{r}_1^k\| \\ \vdots \\ \|\tilde{r}_{m-1}^k\| \end{bmatrix}, \quad (2.19)$$

where $a = (2L\delta + 1)\alpha^m$ and $2L\delta$.

Let

$$\tilde{r}^k = \begin{bmatrix} \|\tilde{r}_0^k\| \\ \vdots \\ \|\tilde{r}_{m-1}^k\| \end{bmatrix}, \quad (2.20)$$

and A be the matrix in equation (2.19).

By the properties of the l_2 operator norm,

$$\|\tilde{r}^{k+1}\|_2 \leq \sigma(A)\|\tilde{r}^k\|_2. \quad (2.21)$$

Note that $\|\tilde{r}^k\|_2$ may be interpreted as the standard l_2 -norm on an element of

$\mathbb{R}^{nm}$. This can be shown by direct calculation.

One can exploit the structure of A to directly compute the spectral radius. Doing so gives $\sigma(A) = 2L\delta(\alpha^m + 1) + \alpha^m$.

Now consider the first step. Provided that the initial residual error is sufficiently small, it follows that $\sigma(A) < 1$. Therefore,

$$\|\tilde{r}^1\|_2 \leq \|\tilde{r}^0\|_2.$$

By induction, it follows that $\delta(k+1) < \delta(k)$, and

$$\|\tilde{r}^{k+1}\|_2 \leq \|\tilde{r}^k\|_2.$$

That is, $\|\tilde{r}^k\| \rightarrow 0$ as $k \rightarrow \infty$. By continuity of the flow on M , this implies that for each $i = 0, \dots, m-1$, y_i^k tends towards the periodic orbit $O(x)$.

To characterize the convergence rate, recall equation (2.21). Let $c_k = 2L\delta(k)(\alpha^m + 1) + \alpha^m$. Since $\delta(k+1) < \delta(k)$ and $\delta(k) > 0$, $c_k \rightarrow \alpha^m$ as $k \rightarrow \infty$.

□

Some remarks are warranted. First, the convergence of the method is formally first order. The factor by which the continuity error is reduced asymptotically approaches α^m . This implies that increasing m increases the convergence factor. Convergence was formally shown for the special case of $l = m - 1$. As will be shown in the numerical experiments that follow, one can in practice use any $l \leq m - 1$; however, the convergence will be degraded. In a parallel computation setting, one might choose $l < m - 1$ when m is large as a means of mitigate parallel communication costs.

2.3 Algorithms and Data Structures

Serial and parallel algorithms for performing the computational steps of the proposed numerical method are presented.

For all of the algorithms presented, the only data structures required are arrays of vectors (column vectors) and arrays of matrices. In a language such as Fortran, it suffices to model these data structures using the intrinsic multidimensional arrays. In C++, one may want to consider combining an STL container that supports efficient random access with elements defined as vector or matrix classes provided by a dedicated C++ linear algebra package.

The algorithms appearing in the algorithm listings that follow are expressed in the form of scripts, with procedures and functions representing key logical units (or steps) of the algorithm defined first, followed by a series of statements that constitute the main driver program of the algorithm. Unless stated otherwise, all arguments passed to a procedure are passed by reference. This means that any modification of these variables by the procedure will be reflected in the corresponding variables of the calling procedure or program. All arguments passed to functions, on the other hand, are passed by value (or equivalently, passed as an immutable reference). Each function returns its results by explicitly invoking the **return** statement. Some algorithm listings may depend on externally defined data, procedures, or functions. All such dependencies are listed at the top of each algorithm listing, and denoted by the keyword **Require**. These external dependencies are introduced either to simplify the exposition or to abstract out algorithm elements whose implementation

is problem dependent. The intent of the algorithm listings is to communicate the essential elements of each algorithm as a concept, and how those elements come together to form a whole. In practice, one may need pass additional data to the procedures and functions shown in each algorithm listing, add statements to write or return intermediate solution data, add mechanisms for exception handling and error reporting, or repackaging the algorithm in a way that adheres to modern Procedural, Functional, or Object Oriented design practices.

Note that the provided algorithm listings in this section do not explicitly show how to compute S . There are many strategies for doing this, all of which depend on the size of the problem and the approximations being used (exact linearization or some other approximation).

2.3.1 A Serial Algorithm

A serial algorithm implementing the computational steps of the numerical method proposed in Section 2.2 is given in Listing 1.

The listed algorithm requires two parameters, one procedure, and two functions to be externally defined. The parameters may be thought of as global parameters whose values are set based on experience or some other heuristic. The implementation of these external procedures and functions are problem dependent and therefore will implicitly defined by their input-output behavior. The first parameter ϵ is the convergence tolerance. The algorithm terminates when the continuity error falls below this threshold. The second parameter is $kmax$. This is the maximum

number of partitions over which continuity error is propagated.

The first of these externally defined procedures is INITIALIZE. This procedure takes X as an argument, and initializes its values to the estimate location of the intersection between the periodic attractor A and the p Poincare sections.

The externally defined function TIMESTEP(.) advances each state vector from one time step to the next. The implementation of this function may be realized as the application of a conventional time marching algorithm to the governing equations of the system to be analyzed. The listing shows TIMESTEP(a)s a function of a single argument x ; however, if a multi-step time marching algorithm is desired, then additional state vectors would be passed as additional arguments.

The externally defined function ATPOINCARESECTION(.) is a predicate function that returns true when a state vector is contained in one of the p Poincare sections, and false otherwise.

The first procedure in the listing is SHOOT. This procedure computes the continuity error, and takes two arguments X and E . X is an array containing the current estimate for the intersection of the periodic attractor at each Poincare section. X is unmodified by the procedure. E is an array that accumulates the continuity error computed by the SHOOT procedure. The SHOOT procedure computes the continuity error by using a time marching algorithm to follow the orbit that passes through each point described in X from the one Poincare section to the next. Then the continuity error is computed as the difference between the integrated value and the current approximation. (Since SHOOT is free of any side-effects, one could also define SHOOT as a function that takes X as input and returns E .)

The second procedure listed is `CORRECT`. This procedure uses the continuity error E computed by `SHOOT` to compute a correction to X that brings X closer to the periodic attractor. The procedure begins by initializing a temporary array Δx to 0, which will subsequently be used to accumulate the corrections to be applied to X . This is followed by three nested loops. The first loop increments the “order” of the approximation used to correct X . The second loop increments the partition index from which the periodic error is propagated, and the third loop effectively propagates this periodic error through each partition by application of the S operator associated with each partition. The results of the second and third loop are accumulated in Δx . Finally, X is corrected by performing array addition (elementwise sum) of X and Δx . Note that algorithm as shown presumes that each $S_j^{l(j+1)}$ operator is known analytically. For “real-world” problems, this may need to be based on a finite difference approximation that is computed and during the time marching process in `SHOOT`, or some other approximation.

The main driver program of the algorithm begins at line 23. First X is initialized, then the main iteration loop is started. First `SHOOT` is called, which returns the continuity error E . Then the continuity error is checked against the convergence tolerance ϵ . If the error is sufficiently small, the program exits. Otherwise, X is corrected by calling `CORRECT`, and then another iteration cycle begins.

2.3.2 Parallel Algorithms

A parallel algorithm implementing the computational steps of the numerical method proposed in Section 2.2 is given in Listing 2. This algorithm exploits the natural parallelism that exists in the previously discussed algorithm (Listing 1). The parallel algorithm may be characterised as a data-parallel algorithm based on domain decomposition of the state space, with Poincare sections defining the domain boundaries. The algorithm in Listing 2 implicitly associates each partition of the state space to an computational process. In principle, one could associate multiple partitions to each process. However, there does not appear to be any utility in doing this. It is also assumed that this algorithm is executed on a machine that supports mechanisms for inter-process communication and synchronization. The parallel programming model assumed to be similar to that of the Message Passing Interface .

The algorithm in Listing 2 depends on the same two external parameters as previously discussed for the serial algorithm (Listing 1). Likewise, it depends on external procedures INITIALIZE, and functions TIMESTEP, ATPOINCARESECTION. These remain essentially unchanged, with the only difference being that INITIALIZE need only take X as a single vector, rather than an array of vectors. The procedure INITIALIZE sets X to the estimated location of the periodic attractor on the “left” or “upwind” Poincare boundary of the partition associated with the process executing the algorithm.

A new external procedure is introduced, SHIFTRIGHT. This procedure takes

a single argument, a vector, and utilizes the inter-process communication and synchronization mechanisms to send the value of its argument to the process on the right, overwriting it with the value received from the left. This procedure may be viewed as an analog of the MPI procedure `MPI_SENDRCV` combined with a ring process topology.

The `SHOOT` procedure computes the periodic error associated X . As with the serial algorithm (Listing 1), it does this by integrating the governing equations, with X as an initial condition, until the Poincare section forming the “downwind” boundary of the domain is reached. Then, the result of the integration process, initially stored in x , is sent to the process on the right and overwritten by the value received by the process on the left by calling `SHIFTRIGHT`. The periodic error is computed by evaluating $E = x - X$ as vectors. Note that the procedure arguments X and E are now vectors, whereas these arguments were arrays of vectors for the serial algorithm.

Procedure `CORRECT` uses the continuity error E to correct X such that it is closer to the periodic attractor. Unlike the serial version of this procedure, no array of vectors or array of matrices are needed. All variables are either scalars, vectors (or 1D-arrays), or matrices (or 2D-arrays). The procedure begins by initializing a work vector Δx to zero. This vector is used to accumulate the correction to be applied to X . The integer k is a counter representing the current order of approximation used to compute the correction, and is initialized to zero. Next, the vector r , is a temporary variable that accumulates the action of S on each processor, as it is translated across each domain with each incremental increase in the order of

approximation. The matrix S , is the linearized state advancement operator that takes each point on the left Poincare boundary to the Poincare boundary on the right

The main program, beginning on line 31, is identical to the main program of the serial algorithm.

2.4 The Connection to Multiple Shooting with Incomplete Cyclic Reduction

The method proposed in Section 2.2 may be viewed as a close cousin of a multiple shooting method combined with incomplete cyclic reduction. Furthermore, the two approaches yield equivalent algorithms under certain circumstances.

Multiple shooting is a domain decomposition approach to boundary value problems. It may be used as a foundation for realizing parallel marching algorithms, as well as a way to improve upon the numerical stability limits of classical shooting methods. In the context of periodic problems, multiple shooting may be viewed as a generalization of periodic shooting. Given a periodic system of known period T , one chooses a number of partitions p , and partitions the time interval $[0, T]$ into p partitions,

$$I_i = [t_i, t_{i+1}], \quad i = 0, \dots, p-1 \quad (2.22)$$

where $t_0 = 0$ and $t_p = T$. Let $x \in C^1(M)$ denote a periodic solution to (2.1). For

each i partition, define $a_i, b_i \in M$ by the equations

$$a_i = x(t_i) \tag{2.23}$$

$$b_i = x(t_{i+1}) \tag{2.24}$$

By definition x is a continuous periodic function. Therefore

$$b_i = a_j, \quad j = i + 1 \pmod{p}. \tag{2.25}$$

Conversely, if $x_i \in C^1 M : I_i \rightarrow M$ satisfy $x_i(t_i) = a_i$ and $x_i(t_{i+1}) = b_i$, then $x(t) = x_i(t)$. That is, each x_i is equal to the restriction of x to I_i . The multiple shooting method attempts to construct x by trying to guess the value of each a_i such that solving (2.1) with a_i as an initial condition yields an x_i whose end point b_i satisfies the continuity condition (2.25). In mathematical terms, let $\phi_{\Delta t_i} : M \rightarrow M$ be the map that takes each a_i to b_i . Then multiple shooting seeks a_i that satisfy

$$\begin{bmatrix} \phi_{\Delta t}(a_{p-1}) - a_0 \\ \phi_{\Delta t}(a_0) - a_1 \\ \vdots \\ \phi_{\Delta t}(a_{p-2}) - a_{p-1} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \tag{2.26}$$

A common approach to approximating a solution to (2.26) is to apply Newton's Method:

1. Guess values for $a_i^0, i = 0, \dots, p-1$.
2. Solve the linear system

$$J\Delta a^k = -r^k \tag{2.27}$$

where

$$J = \begin{bmatrix} -I & & & D\phi_{\Delta t}(a_{p-1}^k) \\ D\phi_{\Delta t}(a_0^k) & -I & & \\ & D\phi_{\Delta t}(a_1^k) & -I & \\ & & \ddots & \ddots \\ & & & D\phi_{\Delta t}(a_{p-2}^k) & -I \end{bmatrix}, \quad (2.28)$$

$$\Delta a^k = \begin{bmatrix} \Delta a_0^k \\ \Delta a_1^k \\ \Delta a_2^k \\ \vdots \\ \Delta a_{p-1}^k \end{bmatrix} \quad (2.29)$$

and

$$r^k = \begin{bmatrix} r_0^k \\ r_1^k \\ r_2^k \\ \vdots \\ r_{p-1}^k \end{bmatrix} = \begin{bmatrix} \phi_{\Delta t}(a_{p-1}^k) - a_0^k \\ \phi_{\Delta t}(a_0^k) - a_1^k \\ \phi_{\Delta t}(a_1^k) - a_2^k \\ \vdots \\ \phi_{\Delta t}(a_{p-2}^k) - a_{p-1}^k \end{bmatrix} \quad (2.30)$$

3. Then compute

$$a^{k+1} = a^k + \Delta a^k \quad (2.31)$$

4. Repeat steps 2 and 3 until satisfactory accuracy has been achieved.

Inspection of equation (2.28) shows that the Jacobian matrix is a banded sparse matrix. With the exception of the non-zero element in the upper right hand

corner, it has the structure of a lower block diagonal matrix. There are many strategies one could use to solve equation (2.27). Iterative methods have become a popular choice for solving sparse linear systems, especially when exact solutions are not required. Many iterative methods, however, require that the system be either diagonally dominant or Symmetric Positive Definit (SPD). Furthermore, preconditioning may be required to ensure efficient convergence. Direct methods are an alternative approach to iterative methods. As a general rule, direct methods tend to be too expensive (or too inefficient) for application to sparse systems. A noteworthy exception to this rule, however, is cyclic reduction. A general introduction to cyclic reduction methods is provided by Gander and Golub [6]. Cyclic reduction has also found utility as a conceptual tool for designing parallel algorithms for sparse linear systems .

Consider an arbitrary matrix M of the same form as J in equation (2.28),

$$M = \begin{bmatrix} -I & & & & A_1 \\ A_2 & -I & & & \\ & A_3 & -I & & \\ & & A_4 & -I & \\ & & & \ddots & \ddots \\ & & & & A_n & -I \end{bmatrix}, \quad (2.32)$$

where each $A_i, i = 1, \dots, n$ are submatrices. Furthermore, assume that n is an even number. Applying the principle strategy of cyclic reduction, let P be the

permutation matrix collects even and odd numbered elements,

$$P \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} x_1 \\ x_3 \\ \vdots \\ x_{n-1} \\ x_2 \\ x_4 \\ \vdots \\ x_n \end{bmatrix} \quad (2.33)$$

Then

$$PMP^{-1} = \left[\begin{array}{ccc|ccc} -I & & & 0 & & A_1 \\ & -I & & A_3 & 0 & \\ & & -I & & A_5 & 0 \\ & & & \ddots & & \ddots \\ & & & & -I & A_{n-1} & 0 \\ \hline A_2 & & & -I & & \\ & A_4 & & & -I & \\ & & A_6 & & & -I \\ & & & \ddots & & \\ & & & & A_n & -I \end{array} \right] \quad (2.34)$$

Thus, given a linear system

$$Mx = r \quad (2.35)$$

one can use P to transform this into

$$PMP^{-1}x = Pr \quad (2.36)$$

$$\Rightarrow \begin{bmatrix} -I & F \\ D & -I \end{bmatrix} \begin{bmatrix} x_o \\ x_e \end{bmatrix} = \begin{bmatrix} r_o \\ r_e \end{bmatrix} \quad (2.37)$$

where F is the submatrix corresponding to the partition in the upper right of (2.34), and D is the diagonal submatrix that correspond to the partition on the lower left of (2.34). By forming the Schur compliments of x_o and x_e , equation (2.37) may be convertex into a block diagonal system

$$\begin{bmatrix} FD - I & 0 \\ 0 & DF - I \end{bmatrix} \begin{bmatrix} x_o \\ x_e \end{bmatrix} = \begin{bmatrix} r_o + Fr_e \\ r_e + Dr_o \end{bmatrix} \quad (2.38)$$

Consider the matrix expression $FD - I$,

$$\begin{aligned} FD - I &= \begin{bmatrix} 0 & & & & A_1 \\ A_3 & 0 & & & \\ & A_5 & 0 & & \\ & & \ddots & \ddots & \\ & & & A_{n-1} & 0 \end{bmatrix} \begin{bmatrix} A_2 \\ & A_4 \\ & & A_6 \\ & & & \ddots \\ & & & & A_n \end{bmatrix} - I \\ &= \begin{bmatrix} -I & & & & A_1A_n \\ A_3A_2 & -I & & & \\ & A_5A_4 & -I & & \\ & & \ddots & \ddots & \\ & & & A_{n-1}A_{n-2} & -I \end{bmatrix} \end{aligned} \quad (2.39)$$

Likewise, consider $DF - I$,

$$\begin{aligned}
DF - I &= \begin{bmatrix} A_2 & & & & & \\ & A_4 & & & & \\ & & A_6 & & & \\ & & & \ddots & & \\ & & & & A_n & \\ & & & & & \end{bmatrix} \begin{bmatrix} 0 & & & & & A_1 \\ A_3 & 0 & & & & \\ & A_5 & 0 & & & \\ & & \ddots & \ddots & & \\ & & & A_{n-1} & 0 & \end{bmatrix} - I \\
&= \begin{bmatrix} -I & & & & & A_2 A_1 \\ A_4 A_3 & -I & & & & \\ & A_6 A_5 & -I & & & \\ & & \ddots & \ddots & & \\ & & & A_n A_{n-1} & -I & \end{bmatrix} \quad (2.40)
\end{aligned}$$

Observe that both $FD - I$ and $DF - I$ have the same form as the original matrix M . Thus, one may solve equation (2.38) by recursively applying the permutation strategy just outlined to each subsystem and forming schur compliments. Let $A_{-j} = A_{n-j}$ for $j = 0, 1, \dots, n-1$. Then, repeating this process m times, $0 \leq m \leq \log_2 n - 1$, and solving for x_i gives

$$x_i = \left(\prod_{j=1}^{2m} A_{i-j} \right) x_{i-2m} - \left[r_i + \sum_{j=l}^{i-1} \left(\prod_{l=j}^{i-1} A_l \right) r_j \right], \quad (2.41)$$

and for $m = \log_2 n$

$$x_i = - \left(I - \prod_{j=1}^{n-1} A_{i+n-j} \right)^{-1} \left[r_i + \sum_{j=l}^{i-1} \left(\prod_{l=j}^{i-1} A_l \right) r_j \right], \quad (2.42)$$

Observe that the diagonal elements in (2.41) involve products of A_i . Likewise, in the case of full application of the cyclic reduction process, represented by

equation (2.42), the inverse matrix is the inverse of identity minus products of A_i . It was identified by Hockney that there are conditions under which the diagonal elements diminish at a quadratic rate, and that under such conditions one may be able to terminate the reduction process early (i.e. use $m < \log_2 n$) and ignore off-diagonal elements. This is called “incomplete cyclic reduction”. It was shown by Heller, incomplete cyclic reduction may be applied to diagonally dominate systems. Application of incomplete cyclic reduction to (2.35) yeilds the solution to be

$$x_i = - \left[r_i + \sum_{j=i-m}^{i-1} \left(\prod_{l=j}^{i-1} A_l \right) r_j \right]. \quad (2.43)$$

Recall the multiple shooting problem. Application of incomplete cyclic reduction to equation (2.31), gives

$$\begin{aligned} a_i^{k+1} &= a_i^k + \Delta a_i^k \\ &= a_i^k + r_i + \sum_{j=i-m}^{i-1} \left(\prod_{l=j}^{i-1} D\phi_{\Delta t}(a_l^k) \right) r_j \\ &= \phi_{\Delta t}(a_{i-1}^k) + \sum_{j=i-m}^{i-1} \left(\prod_{l=j}^{i-1} D\phi_{\Delta t}(a_l^k) \right) r_j. \end{aligned} \quad (2.44)$$

Observe that this has the same form as equation (??) in Theorem ??. This principle difference is that equation (2.44) is based on partitioning of the time domain, whereas (??) is based on a partitioning of the state space at Poincare sections. This is a subtle but important difference. Partitioning at the Poincare boundaries ensures that the continuity error is always orthogonal to the flow, and does not require one to know a-prior the fundamental period of the system.

2.5 Numerical Tests

2.5.1 2D Limit Cycle

2.5.1.1 Governing Equations

Consider a one-dimensional system governed by the nonlinear second order differential equation

$$\ddot{x} + 2\zeta(\omega x, \dot{x})\omega\dot{x} + \omega^2 x = 0, \quad (2.45)$$

where the function $\zeta : \mathbb{R} \times \mathbb{R} \rightarrow \mathbb{R}$ is given by the equation of the form

$$\zeta(a, b) = c(a^2 + b^2 - 1), \quad (2.46)$$

for some constant $c \in \mathbb{R}$.

The foregoing second order differential equation (2.45) may be reduced to a first order system. Let $y = \dot{x}$. Then $\dot{y} = \ddot{x}$, and equation (2.45) may be written as

$$\begin{bmatrix} \dot{x} \\ \dot{y} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -\omega^2 & -2\zeta(\omega x, y)\omega \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}. \quad (2.47)$$

At any point $(x_0, y_0) \in \mathbb{R}^2$, the linearized form of (2.47) is given by

$$\begin{bmatrix} \dot{x} \\ \dot{y} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -\omega^2 - 4c\omega^2 x_0 y_0 & -4\omega c y_0^2 - 2\omega\zeta(x_0, y_0) \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}. \quad (2.48)$$

It can be shown that for any $c > 0$, the the long term dynamics of the foregoing system are governed by a single periodic attractor centered on the origin of $\mathbb{R}^2$. A plot of the flow of the dynamical system represented by (2.47) is shown in Figure 2.1.

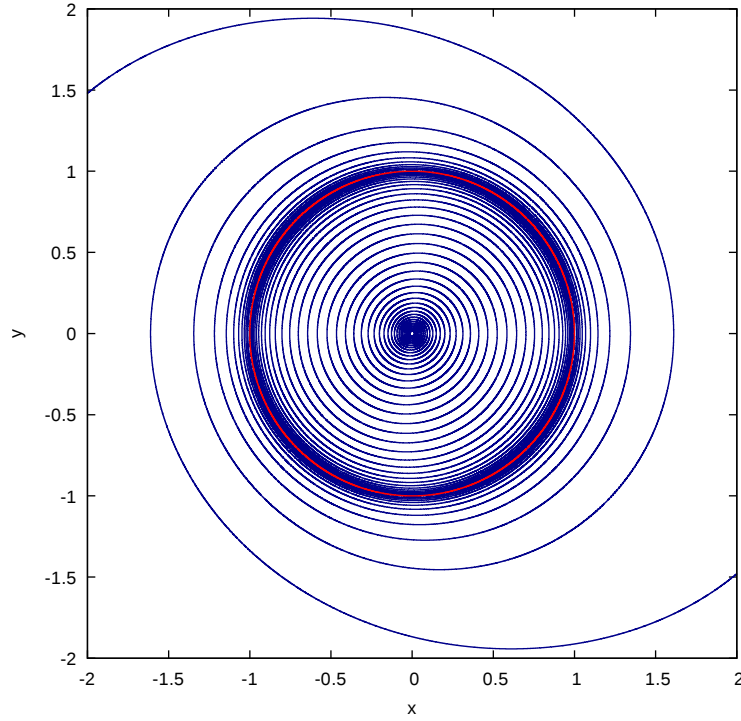


Figure 2.1: Flow of the dynamical system represented by equation (2.47). Transient behavior corresponds to blue curves. Periodic attractor is represented by the red curve.

2.5.1.2 Numerical Model

There are a number of time marching methods one could use to effect a solution to equation (2.45). Implicit time marching methods are a practical necessity for URANS CFD as a means of overcoming the otherwise strict time step size limits required for numerical stability. Implicit integrators have also gained popularity in structural mechanics solvers. Thus, the implicit Euler method will be used for this study. (Higher order implicit methods are also of interest, and will be investigated in subsequent work.)

Let $x_n \in \mathbb{R}^n$ denote a state vector of an autonomous differential system given by equation (2.1) at time t_n . Given a time step h , the state x_{n+1} at time $t_{n+1} = t_n + h$ is implicitly defined by the equation

$$0 = hf(x_{n+1}) - x_{n+1} + x_n. \quad (2.49)$$

Thus, given a solution x_0 , the states along the trajectory $x : \mathbb{R} \rightarrow \mathbb{R}^n$ satisfying equation (2.1) and $x(0) = x_0$ may be approximated by repeated application of (2.49).

The time step h was chosen such that the periodic solution of the numerical model was visibly indistinguishable from that of the analytical solution when plotted. During the exploration process of finding a suitable value for h , it was observed that larger values of h tended to produce a numerical solution whose periodic attractor was of smaller radius than the true solution. However, even for large values of h , the center of the attractor correctly remains centered at $(0, 0)$, and all states are all equidistant from the origin and equally distributed around the numerical attractor (the attractor of the discrete system remains circular).

Newton’s method was used to solve (2.49) at each time level. Since the test problem is rather simple, the analytical Jacobian was used. Also, given the low dimensionality of this test problem, dense matrix operations together with direct inversion of the Jacobian was used. For larger “real-world” problems, one would typically use an approximate Jacobian, and seek to use an iterative linear solution method that exploits the sparse matrix structure that often results from discretization of PDEs. A fixed convergence tolerance of 0.5×10^{-6} was used to terminate the

Newton iteration sequence at each time level, with the number of Newton iterations being allowed to vary as needed to meet this criteria.

The state advancement operator was formulated to be consistent with the implicit Euler time integration scheme. Consider a single time step, from t_n to t_{n+1} . Applying implicit differentiation with respect to x_n to equation (2.49) yields,

$$0 = h \frac{\partial f}{\partial x} \frac{\partial x_{n+1}}{\partial x_n} - \frac{\partial x_{n+1}}{\partial x_n} + I. \quad (2.50)$$

Let $A_{n+1} = \frac{\partial f}{\partial x}(x_{n+1})$. Furthermore, recognize that $\frac{\partial x_{n+1}}{\partial x_n}$ is the linear state advancement operator ϕ_{n+1} that takes x_n at time t_n to its linear approximation at t_{n+1} . Rewriting (2.50) using this notation and factoring gives

$$I = [I - hA_{n+1}] \phi_{n+1}. \quad (2.51)$$

Thus,

$$\phi_{n+1} = [I - hA_{n+1}]^{-1}. \quad (2.52)$$

It follows from the chain rule of Calculus that

$$\phi_{n+m,n} = \phi_{n+m} \cdots \phi_{n+1} \phi_n. \quad (2.53)$$

2.5.1.3 Results

First, the behavior of the multiple algorithm was explored for a range of time partition counts. All cases were executed using the serial algorithm. (The parallel algorithm is discussed later.)

The solution history for direct time marching and multiple shooting with 1, 2, 4, 8 and 16 time partitions are shown in Figure 2.2. Comparing Figures 2.2(a)

and 2.2(b), it can be seen that the two solutions are identical. This behavior is to be expected, as the correction to x_0 at each iteration is to simply assume the value of $x(T)$ from the previous iteration. Comparing Figures 2.2(b) through 2.2(f), it can be seen that increasing the number of time partitions reduces the initial continuity error, while initial error between the numerical solution and analytical solution remains essentially the same. Thus, caution is needed when using the continuity error as a measure of the numerical error. Likewise, if periodicity error is used as a stopping criterion for the multiple shooting method, one needs to be sure that a sufficiently small tolerance is used to avoid prematurely terminating the algorithm.

A comparison of convergence histories is shown in Figure 2.3. The “residual” on the y -axis of the plots is the l_2 -norm of the periodicity error. As previously observed, the initial continuity error of the solution is inversely proportional to the number of time partitions. It is also observed that the number of iterations required to reach the target convergence level is of the same order of magnitude; in this case, 10-15 iterations are required.

DOUBLE CHECK: Inspection of the solution files (not shown) revealed that the total number of time steps executed by the 1-partition case was the same as that required by the direct time marching algorithm.

The foregoing results have used $k = N_p - 1$. This means that the correction of the initial conditions for each time partition is based on continuity error information gathered at every partition boundary. It is natural to wonder, however, how sensitive is the convergence rate to this choice? Can one still realize an acceptable rate of convergence by using a smaller value of k ? The convergence histories for values of

k ranging from 1 to 15 with the number of time partitions fixed at 16 is shown in Figure 2.3. It can be observed that the effect of decreasing k from the maximum possible value of 15 to the minimum value of 1 causes corresponding increase in the number of iterations required to reach convergence. Thus, the benefits of using a reduced value of k , namely a reduction in the cost of computing the correction of initial conditions, must be weighted against the cost of additional iteration cycles. The optimal trade will depend on the particular techniques used to compute the state transition operators.

Given the foregoing observations, it can be concluded that the proposed multiple shooting method does provide a reliable means of computing periodic solutions, however, it is of little practical value when used as a basis for a serial algorithm: the computational work is not reduced by adding more time partitions, nor is the stability of the method improved in a meaningful way by increasing the number of time partitions. The practical value of the multiple shooting method becomes much more apparent when considered as a parallel algorithm. This is illustrated in Figure 2.5, which shows the speed up realized by executing the proposed multiple shooting algorithm in parallel. The number of time partitions was taken to be equal to the number of MPI processes, and each MPI process was assigned to one time partition. The maximum allowable value of k was used in each test case. It can be seen that the parallel speedup of the proposed multiple shooting algorithm is nearly linear over the range of processors available on the test machine. Furthermore, the parallel efficiency of this algorithm for this test case remains above 90%. This is quite good, considering the low dimensionality of the problem being solved.

Algorithm 1 A serial algorithm for implementing the computational steps of the numerical method identified in Section 2.2.

Require: ϵ , $kmax$, INITIALIZE, TIMESTEP, ATPOINCARESECTION

```

1: procedure SHOOT( $X, E$ )
2:   for  $i = 0$  to  $p - 1$  do
3:      $x \leftarrow X[i]$ 
4:     repeat
5:        $x \leftarrow \text{TIMESTEP}(x)$ 
6:     until ATPOINCARESECTION( $x$ )
7:      $E[i] \leftarrow x - X[\text{MOD}(i + 1, p)]$ 
8:   end for
9: end procedure

10: procedure CORRECT( $X, E$ )
11:    $\Delta x \leftarrow 0$ 
12:   for  $k = 0$  to  $kmax$  do
13:     for  $i = 0$  to  $p - 1$  do
14:        $r \leftarrow E[i]$ 
15:       for  $j = i$  to  $i + k - 1$  do
16:          $r \leftarrow S_j^{l(j+1)} r$ 
17:       end for
18:        $\Delta x[\text{MOD}(i + k, p)] \leftarrow \Delta x[\text{MOD}(i + k, p)] + r$ 
19:     end for
20:   end for
21:    $X \leftarrow X + \Delta x$ 
22: end procedure

23: INITIALIZE( $X$ )
24: loop
25:   SHOOT( $X, E$ )
26:   if  $\max_{0 \leq i < p} \|E[i]\| < \epsilon$  then
27:     EXIT
28:   end if
29:   CORRECT( $X, E$ )
30: end loop

```

Algorithm 2 A parallel algorithm for implementing the computational steps of the numerical method identified in Section 2.2.

Require: ϵ , $kmax$, INITIALIZE, TIMESTEP, ATPOINCARESECTION, SHIFTRIGHT

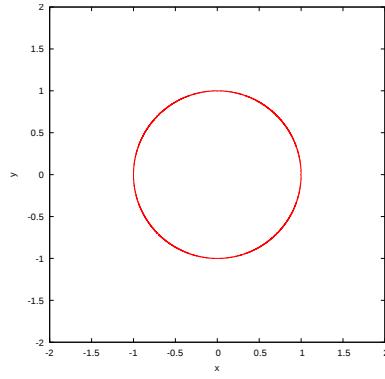
```

1: procedure SHOOT( $X, E$ )
2:    $x \leftarrow X$ 
3:   repeat
4:      $x \leftarrow \text{TIMESTEP}(x)$ 
5:   until ATPOINCARESECTION( $x$ )
6:   SHIFTRIGHT( $x$ )
7:    $E \leftarrow x - X$ 
8: end procedure

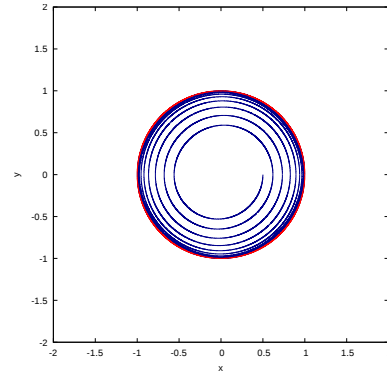
9: procedure CORRECT( $X, E$ )
10:   $\Delta x \leftarrow 0$ 
11:   $k \leftarrow 0$ 
12:   $r \leftarrow E$ 
13:  loop
14:    SHIFTRIGHT( $r$ )
15:     $\Delta x = \Delta x + r$ 
16:     $k = k + 1$ 
17:    if  $k > kmax$  then
18:      BREAK
19:    end if
20:     $r = Sr$ 
21:  end loop
22:   $X \leftarrow X + \Delta x$ 
23: end procedure

24: INITIALIZE( $X$ )
25: loop
26:  SHOOT( $X, E$ )
27:  if  $\max_{0 \leq i < p} \|E[i]\| < \epsilon$  then
28:    EXIT
29:  end if
30:  CORRECT( $X, E$ )
31: end loop

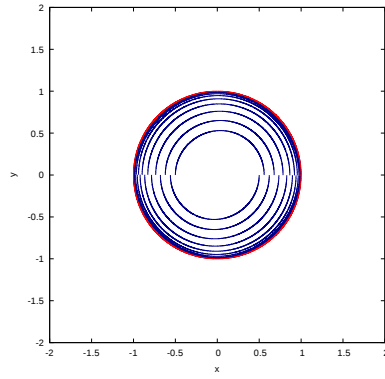
```



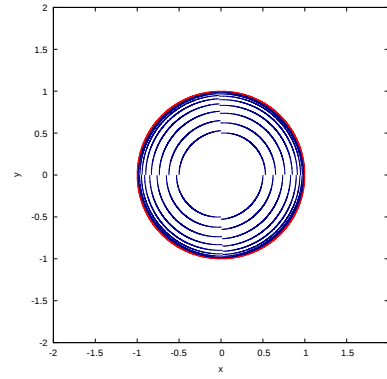
(a) Direct time marching.



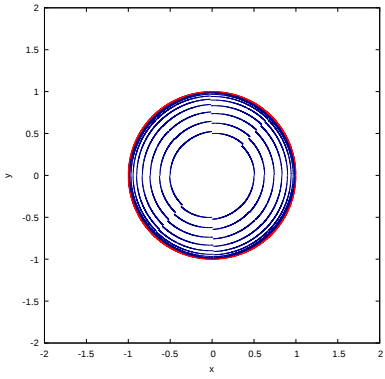
(b) 1 time partition.



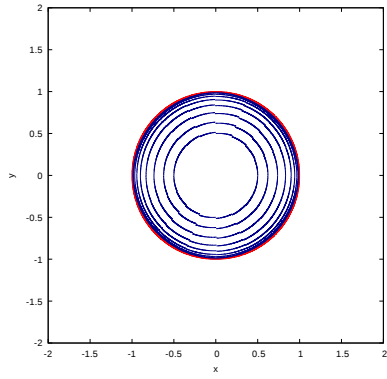
(c) 2 time partitions.



(d) 4 time partitions.



(e) 8 time partitions.



(f) 16 time partitions.

Figure 2.2: Solution histories for direct time marching and multiple shooting with 1, 2, 4, 8 and 16 time partitions. The exact solution is denoted by the red curve. The numerical solution history is shown by the blue curves.

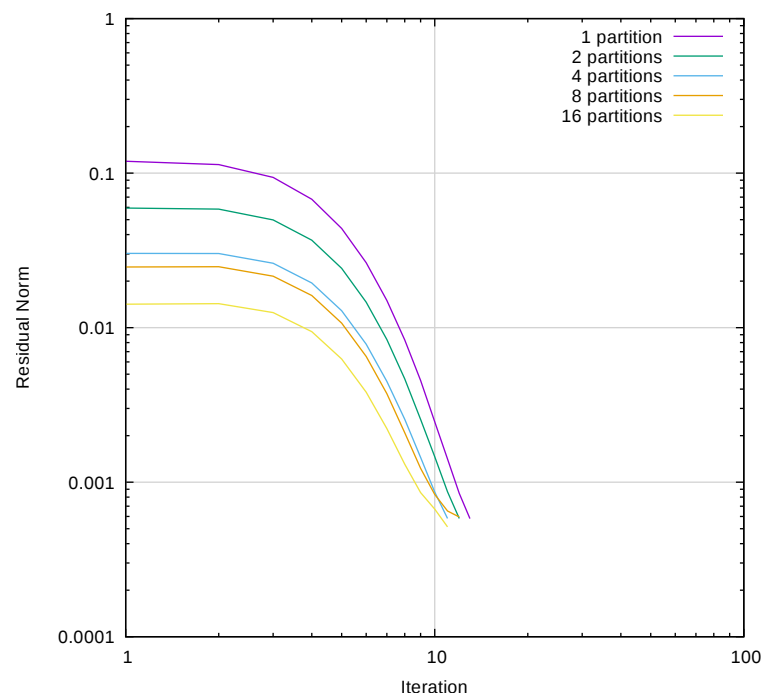


Figure 2.3: Convergence history for 1, 2, 4, 8, and 16 time partition test cases.

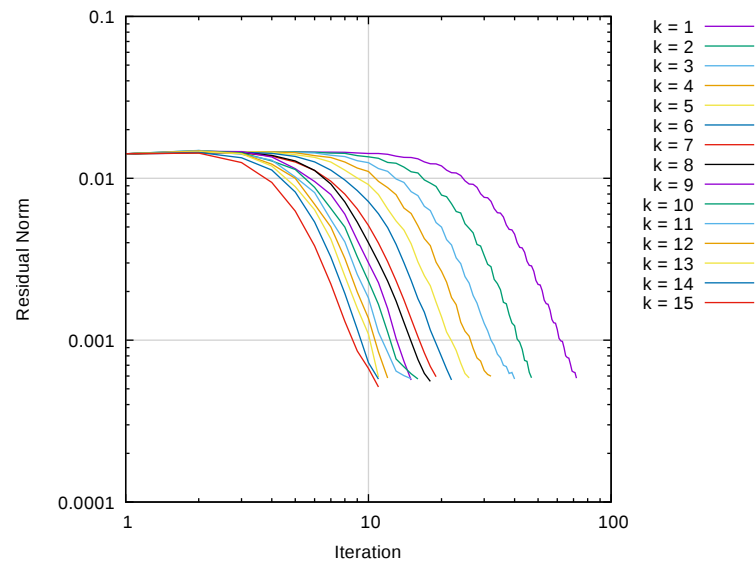


Figure 2.4: Convergence history as a function of k with number of time partitions fixed at 16.

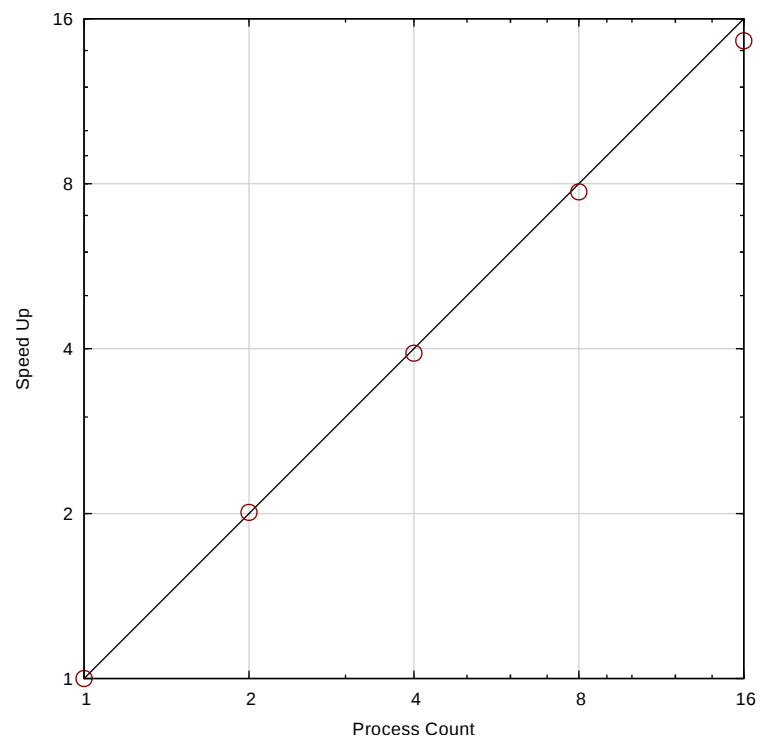


Figure 2.5: Parallel speedup.

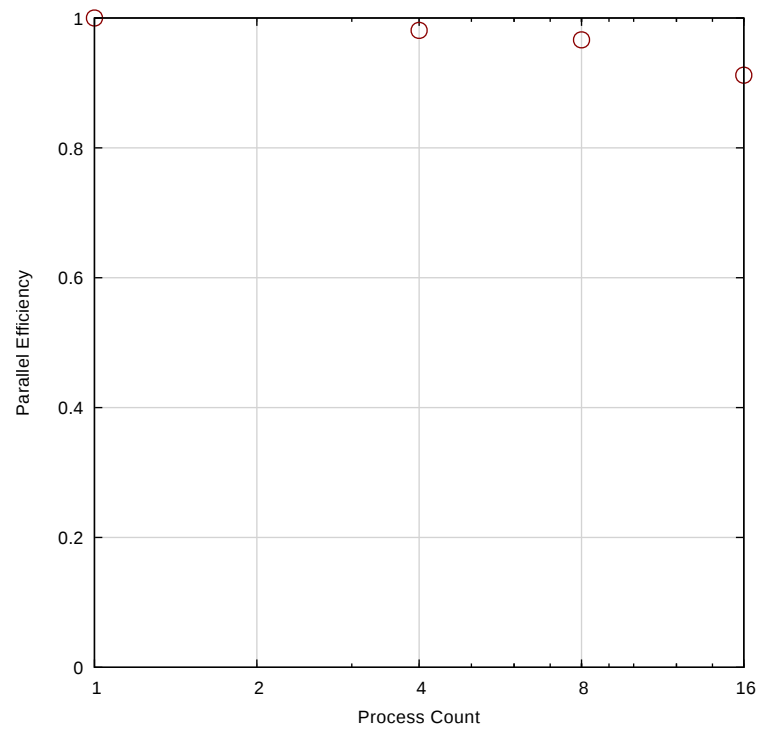


Figure 2.6: Parallel efficiency.

Chapter 3: A Matrix-Free Periodic Multiple Shooting Method for Asymptotically Stable Periodic Systems

The Periodic Multiple Shooting method requires one to evaluate expressions of the form $DP_{i+1,i}r_i^k$, where $DP_{i+1,i}$ is the derivative of section map $P_{i+1,i}$. In the case of high fidelity mathematical models, computing $DP_{i+1,i}$ can be tedious to formulate analytically, and expensive to approximate using conventional finite difference approaches. A practical approach for solving this problem is proposed. The proposed approach is draws its inspiration from Krylov subspace methods for solving linear systems.

3.1 Theory

3.1.1 Approximating $DP_{i+1,i}$ Using Subspace Approximation

The following theorem is based on the Arnoldi method for constructing subspace approximations to linear operators. It differs from Arnoldi in that it considers a linear operator that maps between two distinct linear spaces (two separate tangent spaces) rather than an operator that bijectively maps a vector space back onto itself.

Let $A : U \mapsto V$ be a linear operator between two n -dimensional vector spaces.

Furthermore, suppose $u_1, \dots, u_m$ are a sequence of linearly independent vectors in U and $v_1, \dots, v_m$ are the images of each u_i vector under A ; i.e. $v_i = Au_i$. One can apply Gram-Schmidt process to the vectors $u_1, \dots, u_m$ to yield an orthogonal basis $b_1, \dots, b_m$; i.e.

$$b_i = u_i - \sum_{j=1}^{i-1} \langle u_i, u_j \rangle u_j$$

Let $c_{i,j} = \langle u_i, u_j \rangle$. Then, the image of each basis vector b_i under A is

$$\begin{aligned} Ab_i &= Au_i - \sum_{j=1}^{i-1} c_{i,j} Au_j \\ \Rightarrow Ab_i &= v_i - \sum_{j=1}^{i-1} c_{i,j} v_j. \end{aligned}$$

If $m = n$, then any vector u in U can be expressed as

$$u = \sum_{i=1}^n d_i b_i$$

The image of u under A is then

$$Au = \sum_{i=1}^n d_i \left(v_i - \sum_{j=1}^{i-1} c_{i,j} v_j \right)$$

If $m < n$, then

$$u = \sum_{i=1}^m d_i b_i + e$$

where e is the component of u that is orthogonal to the subspace spanned by $b_1, \dots, b_m$. It can be shown that $|e|$ diminishes as $m \rightarrow n$.

Thus, as one executes the Periodic Shooting Method, each r_i^k vector that is generated upon completion of a shooting iteration can be saved and used to construct an approximation to $DP_{i+1,i}$.

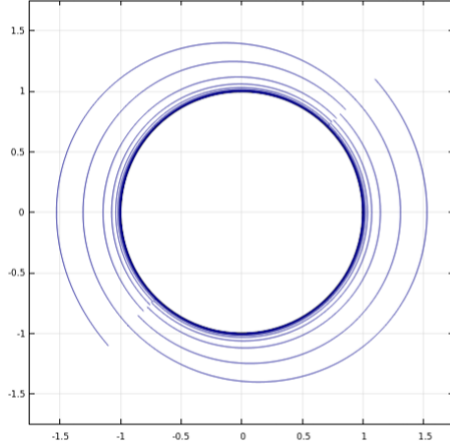


Figure 3.1: Limit cycle solution using 2 processors

3.2 Numerical Tests

The limic cycle test introduced in the preceding chapter were repeated using the proposed subspace approximation for $DP_{i+1,i}$. The dimension of the approximation subspace was kept at 1. The results are shown in Figure3.1 through 3.4 for process counts of 2, 4, 8, and 16, respectively. It can be seen that the method still converges to the correct solution. The parallel speed up is shown in Figure3.5, and parallel efficiency in Figure3.6. It can be seen that use of the matrix-free approach reduces the average parallel efficiency by about 10%.

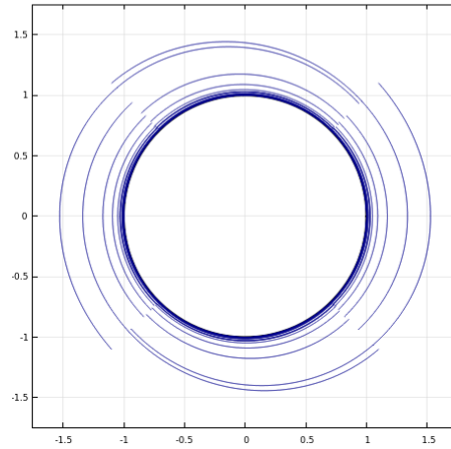


Figure 3.2: Limit cycle solution using 4 processors

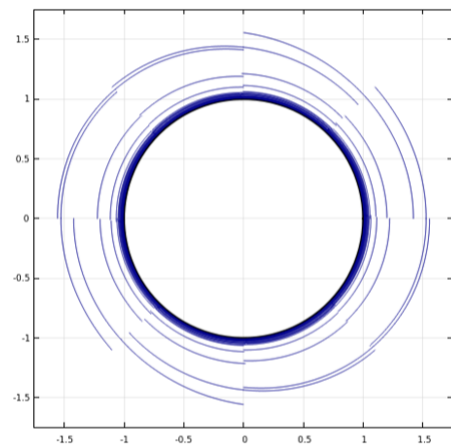


Figure 3.3: Limit cycle solution using 8 processors

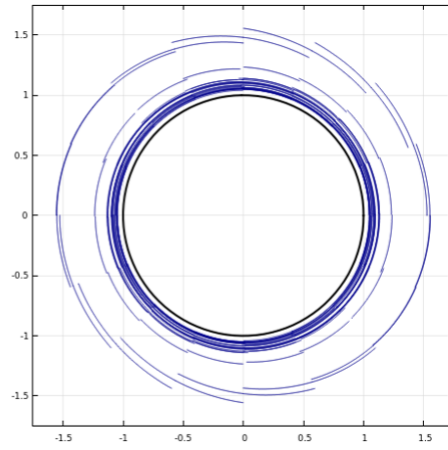


Figure 3.4: Limit cycle solution using 16 processors

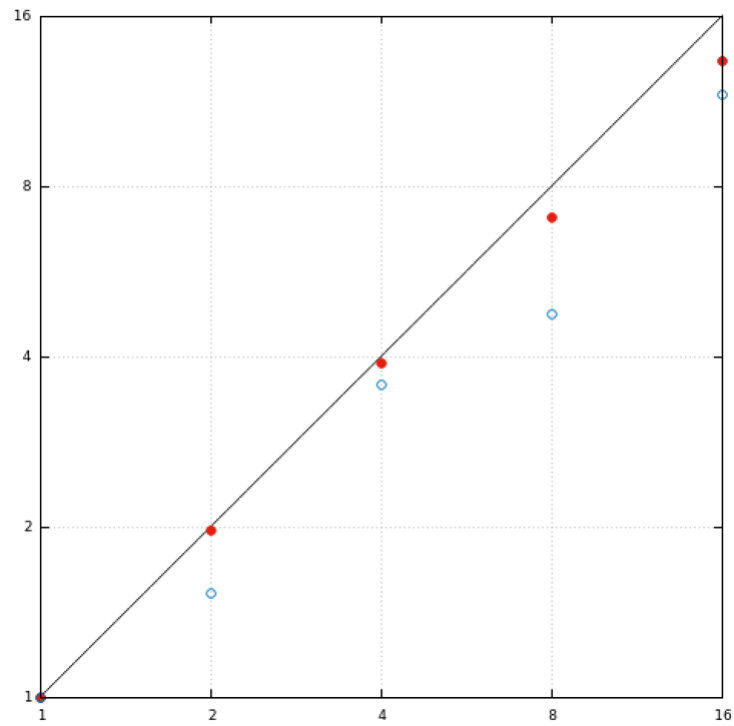


Figure 3.5: Parallel speedup for limit cycle test with matrix-free formulation.

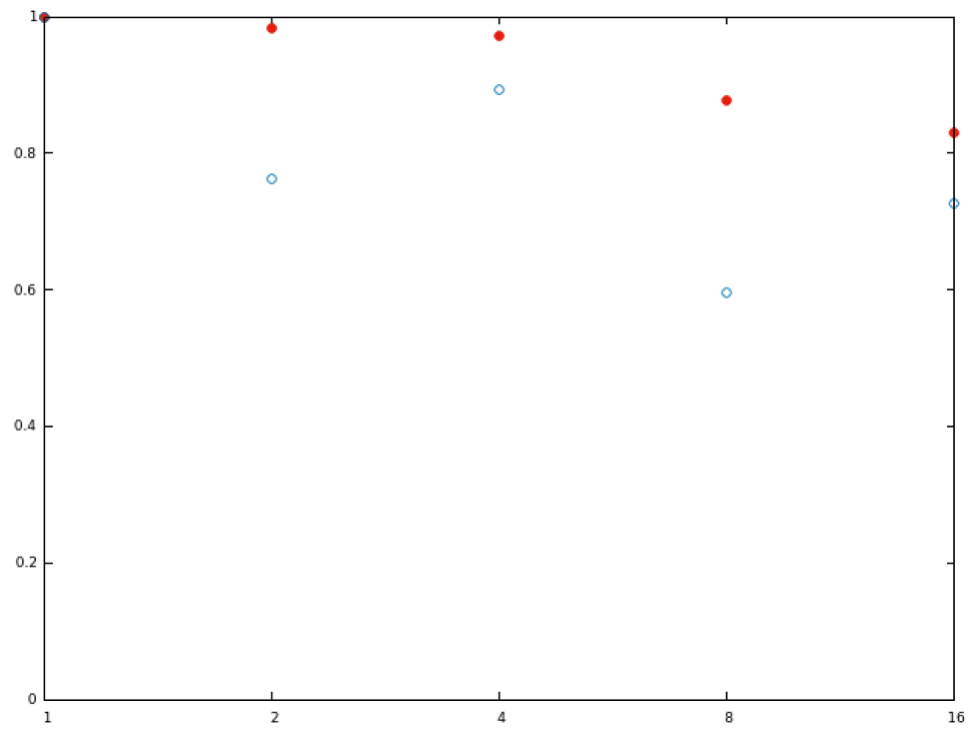


Figure 3.6: Parallel efficiency for limit cycle test with matrix-free formulation.

Chapter 4: Periodic Multiple Shooting for Rotorcraft Trim

4.1 The Trim Problem

A distinguishing characteristic of helicopter aeromechanics analysis from fixed-wing aeroelastic analysis, is that even the most basic of aerodynamic or structural dynamic analysis requires one to determine the fixed-stick control inputs that brings the rotor system into equilibrium for the flight condition under consideration. That is, one must “trim” the rotor. The task of finding the control parameters and rotor states that correspond to a given operating state is refereed to as a “trim problem”. One must also solve a trim problem as a prerequisite to performing a stability analysis of a rotor. Thus the theory and practice of solving the rotor trim problem stands as a corner stone of rotorcraft aeromechanics analysis.

4.2 State-of-the-Art Helicopter Trim Theory and Practice

For over the past 30 years, researchers have investigated the trim problem and various solution methods. Despite this, there are still many open questions regarding the mathematical theory of helicopter trim. Furthermore, the demand for new, more effective, solution methods are continually driven by the growing

mathematical complexity of rotor models and evolving computer architectures.

In regards to the theory of trim, there does not exist a universal theory or method [7]. Each type of trim problem possesses unique mathematical characteristics which demand different solution approaches.

A survey of the current theory and solution methods is as follows.

4.2.1 Mathematical Formulations of Helicopter Trim

The mathematical formulations of the trim problem as they have appeared in the literature are summarized. The key observation is that the trim problem corresponds to a unique kind of differential algebraic system of equations—one that is unlike the differential algebraic equations commonly found in mechanics. In particular, the constraints that express the desired trim conditions are functions of temporal averages rather than instantaneous values.

4.2.1.1 System of ODE's with Integral Constraints

In its simplest form, the trim problem may be mathematically expressed as follows: Given a known function $f : \mathbb{R}^n \mapsto \mathbb{R}^n$, $y : \mathbb{R}^n \times \mathbb{R}^m \mapsto \mathbb{R}^m$, known vector $\bar{y} \in \mathbb{R}^m$, and known period $T \in \mathbb{R}$, find $x : \mathbb{R} \mapsto \mathbb{R}^n$ and $c \in \mathbb{R}^m$ such that

$$\dot{x}(t) = f(x(t), t, c), \quad \forall t \in \mathbb{R}, \quad (4.1)$$

$$x(0) = x(T), \quad (4.2)$$

and

$$\bar{y} = \frac{1}{T} \int_0^T y(x(\tau), c) d\tau. \quad (4.3)$$

Equation (4.1) represents the system of differential equations that govern the dynamics of the rotor (vehicle), with x corresponding to the states of the rotor (vehicle) and c the vector of control parameters. Typically, f is periodic with respect to the time parameter t . If the vector c were known, one could hypothetically determine x using only equations (4.1) and (4.2). Equation (4.3) has been referred to as the “trim constraint” [7]. Physically, it usually either represents some kind of force and moment balance or a set of constraints on vehicle accelerations, with y corresponding to the instantaneous observation of forces and moments or vehicle accelerations, and $\bar{y}$ the corresponding mean value. The period T is typically taken to be period of the main rotor, or average period in the case of a system with multiple rotors operating at non-harmonic periods [7].

The dynamics of the rotor and vehicle typically involve second order differential equations. However, all such systems can be converted for first order. In fact, systems with higher order time derivatives can be converted to first order [3]. Thus, the above form subsumes all autonomous formulations with explicit states (assuming fixed controls).

Since the control parameters are taken to be constants, the foregoing system of equations could be interpreted to represent a family of autonomous systems that are parameterized by c , and the equation of constraint (4.3) representing a desired set of system properties.

4.2.1.2 Explicit System of DAE's with Integral Constraints

Detailed models of rotor and vehicle systems may result in the appearance of additional constraints. In recent years, multi-body dynamics has gained increasing popularity as a means of modeling complex rotor systems. Depending on the level of generality, a multi-body dynamics formulation may employ the use of Lagrange multipliers. This leads to a modification of the trim problem: Given a known functions $f : \mathbb{R}^n \mapsto \mathbb{R}^n$, $g : \mathbb{R}^n \mapsto \mathbb{R}^{n'}$, $y : \mathbb{R}^n \times \mathbb{R}^{n'} \times \mathbb{R}^m \mapsto \mathbb{R}^m$, known vector $\bar{y} \in \mathbb{R}^m$, and known period $T \in \mathbb{R}$, find $x : \mathbb{R} \mapsto \mathbb{R}^n$, $\lambda : \mathbb{R} \mapsto \mathbb{R}^{n'}$ and $c \in \mathbb{R}^m$ such that

$$\dot{x}(t) = f(x(t), \lambda(t), t, c), \quad \forall t \in \mathbb{R}, \quad (4.4)$$

$$0 = g(x), \quad (4.5)$$

$$x(0) = x(T), \quad (4.6)$$

and

$$\bar{y} = \frac{1}{T} \int_0^T y(x(\tau), \lambda(\tau), c) d\tau. \quad (4.7)$$

4.2.1.3 Hamiltonian Formulations with Integral Constraints

Hamiltonian formulations have appeared in the literature as a basis for formulating Finite Element models of helicopter rotors. In the context of trim, one augments the Hamiltonian with a “trim constraint” as is done for ODE's. Thus, the

Hamiltonian formalism is only applied to the dynamics of the rotor system—without regard to the concept of trim.

4.2.1.4 Semi-Discrete Formulations with Integral Constraints

With the exception of Hamiltonian formulations of rotor systems with elastic blades, little work has been done with respect to trim solutions of systems governed by hyperbolic PDE's. In particular, aerodynamic modeling based numerical solutions to the Euler or Navier-Stokes Equations. In literature, the prevailing assumption is that one employs a semi-discrete (i.e. method of lines) approach to reduce the system of PDE's to a system of ODE's. Then one theorizes about the existence the solutions and/or selects a solution methodology as though one were solving a system of ODE's. This is not necessarily wrong. However, it remains as an open question if one can exploit certain characteristics of the underlying PDE's system to realize a more computationally efficient and robust method for effecting trim solutions. Furthermore, modern approaches to PDE may involve adaptive meshing, which implies a variable number of state variables. Thus, the assumption that a system governed by a PDE can be equated to a ODE has its limits.

4.2.1.5 Systems with Hidden States

As identified by Peters, some aerodynamic models may have “hidden states”. A common example of this is a dynamic stall model that contains time delays.

The significance of hidden states is that one cannot directly construct a linear

approximation to the model by way of computing derivative of model with respect to the state vector. This has implications both towards (asymptotic) stability analysis as well the application of iterative methods based on successive linear approximations (e.g. Newton's Method, Periodic Shooting).

4.2.1.6 Under-determined systems

Advanced rotorcraft design concepts may allow for more control parameters than there are trim targets. This results in an under-determined system of equations. An example of this is a helicopter with variable RPM.

One typically effects a solution to this class of problems by treating imposing the usual trim constraints (e.g. force moment balance) together with some kind of optimization criteria (e.g. power minimization).

4.2.2 Existence, Uniqueness, and Solvability

Very little has been done with respect to determining the conditions under which the trim problem has a solution, if the solution is unique, and if it is possible to determine a solution through a sequence of finite steps (or at least approximate it to arbitrary accuracy). In practice, one chooses a method that seems to have a probability of working, and tries it. If they fail to find a solution, the initial guess is adjusted or another method is tried. If they continue to fail to find a solution, one simply gives up and concludes that either there is not a solution or the problem cannot be solved using known methods.

Peters [7] has proposed conditions under which a DAE model of a rotor system is solvable. The solvability conditions proposed by Peters are based on the sufficiency conditions for Newton-like and Auto-Pilot algorithms to converge. Unfortunately, none of these conditions can be easily checked in practice.

4.2.3 Trim Methods and Algorithms

4.2.3.1 Closed Form Control Relations

One strategy for determining the trim state of a helicopter (or rotor) is to seek a closed form expression of the control parameters as a function of the trim targets. Once such an expression is found, then one can eliminate the unknown control parameters from the governing equations. Unfortunately, such expressions have only been found for very simple models. Thus, this approach may only be used to approximate the trim solution of more sophisticated models. When the control parameters are approximated in this way, the resulting accuracy has been reported to be generally inadequate for loads, vibration, and stability analysis. Furthermore, one still must decide how to solve the differential equations that govern the vehicle/rotor response. That said, one could consider using such relations to determine the “initial guess” of a more accurate iterative method.

An example of such an approach is that used by Wei for predicting control

settings of a trimmed rotor operating in the “stalled regime” (as cited in [8]).

$$\theta_0 = \left(6\bar{C}_T + \frac{9}{8}\phi\right) + \left(15\bar{C}_T + \frac{19}{16}\phi\right)\mu^2 \quad (4.8)$$

$$\theta_s = -\left(16\bar{C}_T + \frac{3}{2}\phi\right)\mu - \left(16\bar{C}_T - \frac{3}{4}\phi\right)\mu^3 \quad (4.9)$$

$$\theta_c = \frac{\gamma}{p^2} \left[\left(\bar{C}_T + \frac{1}{48}\phi\right)\mu - \left(\frac{5}{9}\bar{C}_T \frac{1}{16}\phi\right)\mu^3 \right] \quad (4.10)$$

where

$$\phi = \frac{4}{3} \left[\frac{1}{2} \left(\sqrt{C_T^2 + \mu^4} - \mu^2 \right) \right]^{\frac{1}{2}} \quad (4.11)$$

and

$$\bar{C}_T = \frac{C_T}{a\sigma}. \quad (4.12)$$

The formulas were reported by Peters [8] to be “inadequate for the stalled conditions.” It is interesting to observe that equations (4.8), (4.9), and (4.10) are independent of the rotor states.

4.2.3.2 Newton-like Methods with Time Marching

A common approach to the trim problem is to iteratively solve for the control vector and rotor (vehicle) response as follows:

1. Initialize approximate control vector c .
2. Compute the rotor (vehicle) response with c held constant by marching the system to “steady state”.
3. Evaluate the trim constraint.
4. If the trim constraint is satisfied then stop.

5. Update the estimate for the control vector $\tilde{c}$ using a Newton-like scheme.
6. Return to step 2.

Mathematically speaking, this is equivalent to defining the function $X : \mathbb{R} \times \mathbb{R}^m \mapsto \mathbb{R}^n$ by the equation

$$X(t, c) = x(0) + \int_0^t f(x(\tau), \tau, c) d\tau \quad (4.13)$$

Then define the function $F : \mathbb{R}^m \mapsto \mathbb{R}^m$ by the equation

$$F(c) = \bar{y} - \frac{1}{T} \int_0^T y(X(t, c), c) dt, \quad (4.14)$$

and iteratively compute an approximate solution for c by applying a Newton-like Algorithm to equation (4.14). The rotor (vehicle) response x is determined as a side effect of the iterative solution for c .

The “loop” implied by step 6 is sometimes called the “trim loop”, and the derivative $\frac{\partial F}{\partial c}$ is sometimes called the “trim Jacobian”.

In practice, one does not typically use a closed form expression for $\partial F / \partial c$ but rather approximates this with a finite difference approximation(). Furthermore, equations (4.13) and (4.14) are written assuming that $x(0)$ is a point in the periodic attractor. In practice, this is not the case. Rather, one chooses $x(0)$ based on a convenient guess (e.g. zero, or the last known state from the previous iteration) rather than knowledge of the periodic attractor’s location. This means that one needs to integrate equation (4.13) until the solution appears sufficiently periodic to approximate the periodic solution.

The principle advantage of this method is its ease of implementation, and modular architecture. That said, there are a number of detractors.

One detractor of this method may only be used to determine stable trim solutions. This is a direct consequence of determining the rotor (vehicle) response by solving an Initial Value Problem. If the periodic attractor is unstable, then unless $x(0)$ is in the attracting set, the solution will diverge. In practice, $x(0)$ is chosen based on a convenient guess—which rarely happens to coincide with the attracting set. Thus, if the penultimate goal is to support stability analysis, a different method will need to be used. Also, if one is interested in free-flight trim, the governing equations of the vehicle will need to be augmented with fictitious springs and dampers to stabilize the system.

Another detractor of this method is its computational cost. Depending on how $x(0)$ is chosen, the cost of evaluating equation (4.13) can be quite high. Furthermore, if finite differencing is used to compute $\partial F/\partial x$, then the number of evaluations of equation (4.13) increases. One may try to offset the cost of the finite differencing by reusing the computed derivative during subsequent iteration steps (this is technically a chord method). However, this strategy is appropriate when near the solution and it will reduce the convergence rate.

In general, Newton’s Method is not a globally convergent method. It has been empirically shown that this is indeed the case when Newton’s Method is used in this application. In fact, it has been shown that Newton’s Method will fail even when the initial guess is close to the solution. It has been speculated that $\partial F/\partial c$ can become singular or ill-conditioned when near the solution. This carries the

implication that care is needed in choosing the initial estimate for the control vector c , and/or one needs to employ some kind of a globalization scheme. A globalized form of Newton’s Method, known as “Damped Newton’s Method” has been shown to overcome convergence issues related to ill-conditioning of $\partial F/\partial c$.

Sekhar et. al. have hypothesized that the basin of attraction for Newton’s Method, when applied to the helicopter trim problem, is a fractal set [9]. This hypothesis has been extrapolated from the work of Epureanu [10], Haruta [11], and Jeong [12], which explored Newton’s method in a different context. Sekhar performed a series of numerical experiments using a comprehensive (aeroelastic) rotor model with free-wake to explore this hypothesis. The test results support the hypothesis. However, the test results support other hypothesis as well; for example, that the basis of Newton’s method is non-convex, or that it is not simply connected. Neither of these alternative hypothesis imply that the basis of attraction have a fractal structure. Using similar numerical models, Gouravaraju et. al. have attempted to estimate the Hausdorff-Besicovitch dimension of the hypothesized fractal basin [13]. However, their conclusions are drawn based on the presupposition that the basin is in fact fractal. Either way, the work of Sekhar does demonstrate that the basin of attraction for Newton’s Method is non-trivial, and supports the notion that one must look elsewhere if a robust method is desired.

4.2.3.3 Periodic Shooting

Periodic shooting is general purpose iterative method for finding periodic solutions to ordinary differential equations; i.e. finding a solution to equation (4.1) and (4.2). In its simplest form, periodic shooting seeks to find the initial condition x_0 as root of the residual function $r : \mathbb{R}^n \mapsto \mathbb{R}^n$ defined by the equation

$$r(x_0) = x(T) - x_0. \quad (4.15)$$

The solution to equation (4.15) is then iteratively computed by applying Newton's Method (or a variation thereof), with the evaluation of x_T by integrating the ODE from the current estimate for x_0 . This may be expressed as the following algorithm:

1. Choose an initial estimate for x_0 .
2. Compute x_T by integrating equation (4.1) from the initial condition x_0 .
3. Evaluate $r(x_0) = x_T - x_0$.
4. If $r(x_0) = 0$ (or $r(x_0) < \epsilon$, for some small number $\epsilon > 0$) stop.
5. Update the estimate for x_0 using Newton's Method:

$$x_0 \leftarrow x_0 - \left[\frac{\partial x_T}{\partial x_0} - I \right]^{-1} r(x_0)$$

6. Goto step 2.

The Periodic Shooting method has been applied to both autonomous ODE's and ODE's with periodic coefficients. For autonomous ODE's, periodic shooting

is ill-posed [14]. This is a consequence of the fact that any point on the periodic attractor is a solution. Reithmeier [15] has proposed strategies for overcoming this problem, as well as strategies for treating cases where the period T is unknown (e.g. finding limit cycles).

An alternative formulation of the Periodic Shooting method is based on the concept of a state transition matrix. Given a linear system with periodic coefficients

$$\dot{x}(t) = A(t)x(t), \quad (4.16)$$

there exists a matrix $\Phi(t, t_0)$ that maps $x(t_0)$ to $x(t)$, where x is a solution of (4.16). The matrix $\Phi(t, t_0)$ is called a state transition matrix. In the case of a nonlinear system, one can hope to locally approximate the nonlinear system by equation (4.16), and use the state transition matrix as a means to correct the initial condition $x(0)$ so as to satisfy the periodicity constraint (4.2). That is, given a means of determining $\Phi(T, 0)$, one can alternatively express step 5 as

$$x_0 \leftarrow x_0 - [\Phi(T, 0) - I]^{-1} r(x_0).$$

This approach has been investigated by Friedmann [16] for computing rotor aeroelastic response and stability analysis. As shown by Friedmann, one of the noteworthy aspects of this alternative formulation lies in the fact that one can determine the stability characteristics of a rotor based on the Eigenvalues of $\Phi(T, 0)$.

The periodic shooting method was first generalized as a method for finding solutions to the trim problem by Peters [17]. The principle modification is to augment

the periodic constraint (4.2) with the trim constraint (4.3):

$$r \left(\begin{bmatrix} x_0 \\ c \end{bmatrix} \right) = \begin{bmatrix} r_1(x_0, c) \\ r_2(x_0, c) \end{bmatrix} = \begin{bmatrix} x_T - x_0 \\ \bar{y} - \frac{1}{T} \int_0^T y(x(\tau), c) d\tau \end{bmatrix} \quad (4.17)$$

Then apply Newton's method as before to the augmented system:

1. Choose an initial estimate for x_0 and control parameters c .
2. Compute x_T by integrating equation (4.1) from the initial condition x_0 .
3. Evaluate residual (4.17)
4. If $r(x_0, c) = 0$ (or $r(x_0, c) < \epsilon$, for some small number $\epsilon > 0$) stop.
5. Update the estimate for x_0 and c using Newton's Method :

$$\begin{bmatrix} x_0 \\ c \end{bmatrix} \leftarrow \begin{bmatrix} x_0 \\ c \end{bmatrix} - \begin{bmatrix} \frac{\partial x_T}{\partial x_0} - I & \frac{\partial r_1}{\partial c} \\ \frac{\partial r_2}{\partial x_0} & \frac{\partial r_2}{\partial c} \end{bmatrix}^{-1} r \left(\begin{bmatrix} x_0 \\ c \end{bmatrix} \right)$$

6. Goto step 2.

In practice, the Jacobian matrix appearing in step 5 is computed using a finite difference approximation, rather than an exact formulation [17].

As highlighted by Peters, one of the appealing aspects of Periodic Shooting is that the method may be applied to unstable systems, and upper left partition of the Jacobian appearing in step 5 is an approximation of the state transition matrix, and therefore can be used to deduce the stability characteristics of the rotor. Thus, Periodic Shooting provides is very attractive for comprehensive rotor analysis, where one is interested in predicting rotor response, loads and stability.

Furthermore, Periodic Shooting allows one to reuse rotor models that have been formulated in the time domain. The disadvantage of Periodic Shooting is primarily its computational cost. Another potential issue is that Periodic Shooting cannot, in principle, be applied to models that possess hidden states (e.g. a dynamic stall model with pre-programmed time delays).

A modified version of the Periodic Shooting algorithm for distributed parallel architectures was proposed by Subramanian et. al. [18]. The parallel algorithm assumes that one is using a finite difference approximation of the Jacobian, and that the Jacobian is a dense matrix. Given these assumptions, the algorithm works by associating each element of the Jacobian matrix with a distinct process, and exploiting the fact that each of these processes can compute the necessary function integrations for each of their respective state vectors and state perturbations independently of other processes. It was shown that the resulting algorithm has a high degree of parallel efficiency. It should be noted, however, that this method assumes that the size of the rotor model is such that it can fit within the memory limits of each process. This may not be the case, for example, for a large CFD simulation.

It can be observed that the Periodic Shooting method assumes a system with a fixed number of state variables. For rotor aeromechanic models represented by PDE's, it may be desirable to incorporate solution adaptation in the numerical discretization. This would imply a variable number of state variables. A generalization of Periodic Shooting that can accommodate a variable number of states has yet to be developed.

4.2.3.4 Fast Floquet

The Fast Floquet method is a variation of Periodic Shooting that exploits the symmetry of rotor dynamics to reduce computational cost. Given a rotor with N_b identical blades, the response of each blade is identical to that of the blade that precedes it (or follows it). Therefore, one can apply a similar solution strategy as outlined for Periodic Shooting, but instead of computing a full rotor revolution at each iteration, one only needs to integrate $1/N_b$ revolutions of the rotor. Furthermore, one can compute $\Phi(T, 0)$ directly from $\Phi(T/N_b, 0)$ using this symmetry. The Fast Floquet method was originally identified by McVicar [19]. Peters identified that the Fast Floquet Method generally gives improved numerical accuracy of the rotor Eigenvalues than the conventional Periodic Shooting Method [20].

A Parallel version of the Fast Floquet algorithm was proposed by Subramanian et. al. [21]. The method is a direct extension of their parallel periodic shooting algorithm [18] that exploits the symmetry relationship of the Fast Floquet method.

4.2.3.5 Harmonic Balance

Like Periodic Shooting, Harmonic Balance is a general purpose family of methods for computing periodic solutions to differential equations. The principle strategy of Harmonic Balance is to approximate the solution as a partial sum of a Fourier Series,

$$x(t) = A_0 + \sum_{k=1}^N [A_k \cos(k\omega t) + B_k \sin(k\omega t)]. \quad (4.18)$$

Substitute equation (4.18) into the governing equation. After extensive application of trigonometric identities and asymptotic expansions of nonlinear terms, one then should be able to algebraically express the result in the form

$$\begin{aligned}
0 = & f_0(A_0, \dots, A_N, B_1, \dots, B_N) \\
& + \sum_{k=1}^N f_k(A_0, \dots, A_N, B_1, \dots, B_N) \sin(k\omega t) \\
& + \sum_{k=1}^N g_k(A_0, \dots, A_N, B_1, \dots, B_N) \cos(k\omega t).
\end{aligned} \tag{4.19}$$

Then one imposes the condition that each f_k and g_k are identically zero. This results a system of $2N + 1$ algebraic equations, plus an initial condition, that one tries to solve for $A_0, \dots, A_N$ and $B_1, \dots, B_N$, and ω . If ω is known, then one can drop the need for an initial condition.

Nayfeh and Mook [22] considered nonlinear single degree of freedom systems governed by an equation of the form

$$\ddot{x} + f(x + u_0) = 0, \tag{4.20}$$

Their solution procedure begins by using a Taylor series expansion to f to transform (4.20) into

$$\ddot{x} + \sum_{n=1}^N \alpha_n x^n = 0, \tag{4.21}$$

where each α_n is a Taylor series coefficient. Then a solution of the form

$$x(t) = \sum_{m=0}^M A_m \cos(\omega m t + m\beta_0) \tag{4.22}$$

is assumed, where β_0 and A_1 are determined by initial conditions, and the remaining harmonic coefficients $A_0, A_2, \dots, A_M$ and fundamental frequency ω are determined

by harmonic balance. They remark that predicting ω with this method can be very sensitive to the number of harmonics included in the approximation.

The harmonic balance method, as an perturbation method, has been used to predict limit cycle flutter of two-dimensional airfoils. Shahrzad and Mahzoon [23] compared flutter speeds as predicted by a first order harmonic balance method, the center manifold theory, and theory of normal forms. They found that the harmonic balance can predict flutter speed for airfoil models with nonlinear stiffness with good accuracy, as compared to direct numerical integration with fifth order Runge-Kutta. Liu and Dowell [24] have investigated the application of harmonic balance to the dynamic response of an airfoil with freeplay control surfaces, with the aerodynamics modeled using a state-space unsteady aerodynamic model.

Cheung and Lau [25] proposed the Incremental Harmonic Balance (IHB) method for strongly nonlinear problems in structural mechanics. The IHB method is based on the Incremental Hamiltonian method together with what the authors call a "time-space strip procedure." However, it appears as though one can arrive at the same outcome by applying Newton-Homotopy method to the standard Harmonic Balance equations, with the fundamental frequency ω and Fourier amplitudes of the generalized forces taken as homotopy parameters. (The IHB method presumes that one knows the fundamental frequency of the system.) The IHB method does not, in general, provide guidance on how to choose the size of the homotopy increments, nor how each homotopy parameter ought to be increased relative to one another. Cheung and Chen [26] later demonstrated the applicability of IHB to more general periodic systems with cubic non-linearities, and demonstrated the combined use of

IHB and Floquet theory to determine the stability and response of a periodic system. Raghothama and Nrayanan [27] demonstrated the application of IHB to the limit cycle analysis of a pitching-plunging airfoil with cubic stiffness.

Hall et al. [28] investigated the application of the harmonic balance method to spatially and temporally periodic flows modeled by the Euler equations and the 2D RANS equations with the Spalart Almaras turbulence model. The fundamental excitation frequency was assumed to be known a-priori. Their initial approach was based on a direct application of the harmonic balance method to the governing equations with periodic boundary conditions. They reported that the “method produced accurate unsteady flow solutions.” However, the cost of the analysis increased proportional to N^3 , where N is the number of harmonics included in the analysis. Thus, they proposed a modified harmonic balance method that allows one work directly with the flow variables known on a temporal grid with $2N + 1$ points, but with a spectral approximation to the time derivative used in place of conventional finite-differencing. Despite working directly with physical flow variables, the method does require the use of Discrete Fourier Transforms to be used during the iteration process to enforce boundary conditions. A pseudo time marching method was used to solve the discrete equations. They applied the method to the flow of a front stage transonic rotor of a “modern” high-pressure compressor, for both steady and unsteady flow. The unsteady flow was excited by prescribed blade oscillations at a single frequency. The authors found that 3 to 5 harmonics were required to accurately predict the zeroth and first harmonics of the solution. The authors also reported that the method failed to converge when 7 harmonics

were included. (The authors did not proceed to try including 8 or more harmonics.) Despite this, the method was shown to be capable of capturing the combined presence of steady shocks waves and flow separation, and to predict the nominal chord-wise pressure distribution along the upper and lower surfaces of the blades (2D airfoils in this case). The unsteady analysis was reported to be approximately 10 times the cost of a steady-state analysis. No discussion or demonstration of the application of this method to parallel computer architectures was given. Thomas et al. [29] investigated the application of this method to the prediction of Limit-Cycle Oscillations arising from 2D airfoil flutter. The wing was modeled as a rigid 2D body mounted on a spring damper system, with the NLR 7301 airfoil geometry. Three harmonics were used in the analysis. The predicted LCO parameters showed commensurate agreement with the cited predictions of others, but the predicted LCO pitch amplitude did not appear to correlate well with the measured amplitude from wind tunnel experiment.

Liu et al. [30] restated the foregoing harmonic balance method in more general terms and called the method the “High Dimensional Harmonic Balance” method or “HDHB” method, and compared the HDHB method to the classical Harmonic Balance (HB) method using the Duffing equation as a model problem. It was shown that the HB and HDHB methods are identical when applied to linear systems (or nonlinear systems whose amplitude of motion is small), but nonlinear terms in the governing equations can lead to significant differences in approximation for the same finite number of harmonics. Despite the computational benefits of the HDHB method, it was shown that the HDHB method can yield non-physical solutions.

Furthermore, it was concluded that, in order to obtain the accuracy of the HB method, one needs to use approximately twice as many harmonics in the HDHB method as used in the HB method. Liu et al. [31] arrived at these same conclusion when using the Van Der Pol oscillator as a model problem. Application of the HDHB method to rotorcraft aerodynamics was investigated by Ekici et al. [32]. The flow was modeled using the Euler equations (inviscid, compressible flow), and periodic boundary conditions in space and time were used. They showed that it is possible to obtain reasonable predictions of the chordwise pressure distributions of the Carodona-Tung rotor in hover and non-lifting forward flight when at least 5 harmonics are used. They also demonstrated the application of the HDHB method to lifting forward flight with prescribed blade motions and control inputs. Trim analysis was not performed. Application of the HDHB method to parallel computer architectures was not addressed.

Generalization of the Harmonic Balance method to the trim problem generally follows the same strategy as other generalizations of periodic solution techniques to trim: apply the standard discretization or approximation techniques as one would if one were directly solving for a periodic solution, augment the resulting system of algebraic equations with the trim constraint, then solve the augmented system using standard iterative methods for nonlinear equations. An example application of the harmonic balance to trim analysis is demonstrated by the work of Hsu and Peters [33]. Despite Harmonic Balance being frequently mentioned in the literature on helicopter trim, there does not appear to be any work published in the last 30 years that directly investigates the computational effectiveness of various generalizations

of the Harmonic Balance method to the trim problem for high fidelity rotor models and/or modern computer architectures. This may be due to the fact that Harmonic Balance does not scale well with respect to the number of harmonics, risk of spurious solutions, difficulties formulating the Harmonic Balance equations for complex models, or lack of a perceived path to realizing an efficient parallel algorithm.

4.2.3.6 Finite Element in Time

Application of the Finite Element in the time domain, often referred to as “Finite Element in Time” or simply “FET”, was introduced for rotorcraft applications by Borri [34]. Since then it has gained some popularity for directly computing periodic response of rotors. While one could simultaneously solve the FET discretized rotor equations and the trim constraints, a more common approach is to solve the trim constraints using a separate Newton-like iteration.

4.2.3.7 Auto Pilot

The Auto Pilot is a family of methods that approach the trim problem by modeling the rotor (vehicle) controls as time dependent functions governed by a feedback controller. The feedback controller is designed in such a way as to enforce the trim constraint in the limit of $t \rightarrow \infty$. The entire rotor (vehicle) system, together with the control laws, are then integrated from a user specified initial condition until the rotor (vehicle) states and control values reach a periodic state (or a reasonable approximation thereof). Mathematically speaking, one would solve an initial value

problem of the form

$$\dot{x}(t) = f(x(t), \theta(t), t) \quad (4.23)$$

$$\tau\ddot{\theta} + \dot{\theta} = ACM(y_* - y(x(t))) \quad (4.24)$$

$$x(0) = x_0 \quad (4.25)$$

$$\theta(0) = \theta_0 \quad (4.26)$$

where x are the system states and θ is the control vector, y_* is the desired trim targets, and y is the observed values of the trim targets. There have been a number of variations on this theme proposed throughout the literature.

The Auto Pilot method of helicopter trim was first proposed by Peters [35].

Peters proposed the following control relation

$$\tau\ddot{\theta} + \dot{\theta} = ACM\Delta, \quad (4.27)$$

where $\Delta \in \mathbb{R}^m$ is the observed difference between the trim metrics (e.g. instantaneous rotor thrust and hub moments) and their respective target values, $CM \in \mathbb{R}^{m \times m}$ is a “coupling matrix”, $A \in \mathbb{R}^{m \times m}$ is gain matrix, and $\tau \in \mathbb{R}^{m \times m}$ is a diagonal matrix of time constants. Peters stated that the coupling matrix CM is formulated based on an a-priori understanding of the system behavior, but it does not need to be exact; it only needs to cause θ to move in the correct direction. The term $ACM\Delta$ may be thought of as defining a nominal rate of change in θ as a function of Δ . The term $\tau\ddot{\theta}$ serves to filter changes in θ that would otherwise be driven by the vibratory components of Δ (i.e. one only wishes the time averaged value of Δ to become zero). The foregoing control scheme was tested for a

simple rotor model with a rigid blade with flap and torsion degrees of freedom (no lag), and a quasi-steady aerodynamic model with uniform inflow. Only one blade was modeled. The governing equations were integrated using a “predictor corrector method.” For purposes of numerical testing, the time constants and gains for θ_c and θ_s were taken to be equal, thereby reducing the number of control parameters to be empirically determined (i.e. ”tunned”). Peters then proceeded to graphical and formal numerical optimization methods to determine the optimum time constants and gains. When manually tuning an autopilot, one increases the gains to increase the speed at which the controller converges, while increasing the time constants as needed to ensure stability and eliminate control oscillations. Peters found that the optimum value for the time constants and gains were approximately equal to the period of one rotor revolution. However, these values do not in general ensure that the control parameters reach constant values; small oscillations in the controls may still be present. One can reduce control oscillations by increasing the time constants, but this comes at the price of increasing the time required to trim the rotor. Peters also compared the autopilot to periodic shooting. He concluded that the autopilot was computationally more efficient than periodic shooting when the rotor system is moderately to heavily damped. (In this case, the rotor did not have a lag degree of freedom, so this condition was satisfied.) However, this cost comparison does not include the cost of tuning the autopilot, and presumes that a sufficiently accurate CM matrix can be derived. Furthermore, the cost analysis of Periodic Shooting is based on the assumption that finite differencing and dense matrix operations are used. Peters noted that the autopilot is limited to stable rotor

systems, whereas periodic shooting may be used for unstable systems. Thus, if one is ultimately interested in performing rotor stability analysis, the autopilot method cannot be used.

Peters and Chouchane [8] investigated the effect of dynamic stall on trimmed rotor response using both the auto-pilot method and a set of closed form relations derived using the harmonic balance method. The Unified Lift model was used to model the aerodynamics of the rotor blades, including the effects of dynamic stall. The Unified Lift model is based on an ODE formulation for the unsteady lift of an airfoil. The aerodynamic model does not model the unsteady drag that often accompanies dynamic stall. The authors formulated an elastic model for the rotor blades, but only used rigid flap and lag modes in their numerical study. A “predictor corrector” method was used to integrate the system equations. The authors explored a range of thrust and advance ratios. The authors found that the approximate closed form relations that they had derived using the harmonic balance method were generally inadequate for trimming the rotor, and that the Auto Pilot method was comparatively more reliable.

Investigation of the effect of dynamic stall on trimmed rotor response was continued by Peters et. al. [36]. In this follow on study, they included torsional degree-of-freedom (in addition to lag and flap). As in the previous study, they considered a moment trim of the rotor. The auto-pilot was tuned using a combination of gradient based numerical optimization and graphical methods. The authors observed that the settling time of the controller can exhibit discontinuous behavior as a function of the time constants and gains. This made it difficult to apply gradient

based numerical optimization methods to tune the controller. The authors suggested that if one wished to manually tune an auto-pilot, a good strategy is to first increase the gains of the controller until stability limits are reached, then proceed to increase the time constants to eliminate control oscillations. It was also stated that one generally wants to keep the time constants of the controller small enough so that the natural period of the controller is about 3-4 rotor revolutions. The authors found that the optimal control parameters for hover differed significantly from those for forward flight. Thus, auto-pilot parameters depend on both the rotor design and flight condition. The authors also reported difficulties tuning the auto-pilot for rotors that were soft in torsion. They speculated that part of the difficulty may be related to some of the simplifying approximations used to formulate the CM matrix. However, they also speculated that an adaptive control scheme may ultimately be needed. Finally, the authors concluded that the auto-pilot method is not appropriate for effecting trim solutions near stability boundaries.

Enns [37] demonstrated the feasibility of training a neural network for trimming a model of an Apache helicopter. Their principle aim was to investigate the application of a neuro-control scheme as part of a flight control system for a physical helicopter. One might consider directly using this strategy as a means of overcoming the limitations of auto-pilot method proposed by Peters. However, the cost of training the neural network is quite high. It was reported that the neural network required 1000 training epochs to effect a trim solution, where the time duration for each training epoch needs to be long enough to observe the steady response of the rotor for a given control input. That said, the control architecture and training

methods used by Enns is just one of many possibilities.

Riviello [38] proposed an neuro-autopilot scheme that is a composition of a low cost analytical reference model together with a neural network that is trained online to learn the error between the reference model and the detailed helicopter model to be trimmed. In principle, if the reference model is properly formulated, the amount of input-output data samples required to train the neural network ought to be small. This implies that the total time required to effect a trim solution ought to be small. The control scheme proposed by Riviello is a form of Nonlinear Model Predictive Control (NMPC). The rotor model to be trimmed is presumed to be a high-fidelity flexible-multibody dynamics model with the aerodynamics modeled using unsteady lifting models or CFD. The principle strategy is to determine an “optimal” control schedule that maneuvers the rotor (vehicle) from an initial given state into the desired trim state by solving an optimal control problem using a computationally inexpensive model of the rotor (vehicle) input-output behavior. The model used by the control optimizer is formulated as a composition of a simple reduced order model and a Neural Network. The purpose of the Neural Network is to correct the error between the simple reference model and the high fidelity rotor (vehicle) model. Training of the Neural Network is performed online using the standard back-propagation training algorithm. The purpose of the analytically formulated reference model is to reduce the amount of input-output samples that need to be collected to train the Neural Network. The Neural Network is updated at the end of each prediction horizon, during which the error between predicted plan input-output behavior and the actual system are monitored and used in subsequent

Neural Network training updates. After each Neural Network update, a new control schedule is recomputed, and the simulation resumes using the new control schedule. Riviello demonstrated the NMPC scheme for a flexible multi-body dynamics model of the UH60 rotor. The rotor controls were modeled using direct actuation of the pitch bearing; the details of the swash-plate and pitch-link control system were not modeled. The aerodynamics were modeled using simple strip theory. The numerical tests demonstrated that the NMPC method can effect a trim solution of the foregoing UH60 rotor model at advanced ratios of 0 and 0.35. It was also demonstrated that it is possible to tune a classical auto-pilot to yield the same performance as the NMPC controller. However, this requires a great deal of user trial-and-error. Furthermore, optimal parameter selection for a classical auto-pilot is typically sensitive to flight condition. The NMPC scheme, on the other hand, is self "tuning" and is easier to use in practice. These conclusions were later shown to also hold for a flexible multi-body dynamic model coupled with a more detailed lifting line analysis and dynamic inflow model by Bottasso and Riviello [39].

4.2.3.8 Hybrid Schemes

A principle strategy used in engineering design is to identify primitive solutions to a problem, catalog their respective strengths and weaknesses, then seek to create a new solution as a composition of these primitive solutions in a way that retains their combined strengths while eliminating their weaknesses. This design strategy has been applied to a trim methods; however, research along these lines has not

been exhaustive.

Kim and Celi [40] explored a hybridization of the harmonic balance and periodic shooting method. The method proceeds by using harmonic balance to compute an approximate trim solution. Then, using the approximation solution computed using harmonic balance, the periodic shooting method is used to refine the estimate. This composition serves to mitigate the computational cost of periodic shooting and the inherent accuracy limitations of the harmonic balance method. This presumes that the cost of Periodic Shooting is dominated by the number of iterations, and that Since neither harmonic balance nor periodic shooting are limited to stable systems, the composite method is in principle capable of determine a solution to an unstable system. They successfully demonstrated their composite method for a full helicopter model in forward flight. The model was consisted of rigid body blade and fuselage models with blade-element theory and dynamic inflow aerodynamics. It was observed that relatively few iterations of periodic shooting were required to realize an a solution of acceptable accuracy; however, they did not provide a direct comparison between the cost of the hybrid method vs the cost of exclusively using either Periodic Shooting or Harmonic Balance.

Peters and Li [41] proposed a hybrid algorithm based on periodic-shooting and the auto-pilot method. The method composes Periodic-Shooting and the Auto-Pilot methods by way of a decision function that adaptively toggles between the two methods based on measured computational time and solution convergence. Periodic Shooting is used by default for the first iteration. The CM matrix of the Auto-Pilot in subsequent iterations is taken from the Jacobian computed for the most recent

application of Periodic Shooting. The time constants and gains of the Auto-Pilot must still be chosen by the analyst. The hybrid method was demonstrated for a “rigid blade flap-lag model of a rotor with pitch and hub degrees of freedom.” By varying the hub stiffness, it was possible to simulate (approximately) the effects of free-flight and fuselage elastic degrees of freedom on rotor trim. (The hub did not have translational degrees of freedom.) The hybrid method was successfully effect a trim solution at an advance ratio of 0.3 and a thrust coefficient of 0.08 for the fixed-hub and simulated elastic-hub test cases. The hybrid method was found to yield a factor of 3 speed up for the test cases considered.

4.2.3.9 Optimal Trim

Advanced rotorcraft configurations such as compound helicopters or variable RPM helicopters often provide additional control parameters in excess of the number of trim constraints. This implies the existence of multiple trim solutions at each flight condition. In the case of variable RPM rotors with swashplate control there are infinitely many solutions.

Shank [\[42\]](#) proposed using a numerical optimization approach to solve this class of problems. It was shown that one can find trim solutions that minimize power. It was not shown (nor claimed), however, that this approach necessarily yields a unique optimal solution.

4.2.4 Trim of Non-Periodic Systems

Very little work has been done in the area of trim methods for non-periodic systems. It has been well established that multi-rotor systems, where each rotor operates at a non-integer frequency multiple of the other rotors, will exhibit a non-periodic response. In these cases, it has become common practice to simply treat the problem as having a periodic solution with a fundamental frequency equal to the average fundamental frequency of the rotors [7].

4.2.5 The Delta-Coupling Method

Application of Computational Fluid Dynamics (CFD) to problems in helicopter aeromechanics has exclusively been by way of the delta-coupling method. The delta-coupling method is not a trim method in of itself. It is merely an iterative means of correcting the airload models used by legacy comprehensive rotorcraft analysis tools based on CFD airload predictions. The comprehensive analysis tools solves the trim problem using one of the methods previously discussed.

The delta-coupling method was first introduced by Tung et. al. [43]. They sought a means of improving unsteady lift predictions on rotors in high speed forward flight, where unsteady transonic flow is significant. The delta-coupling method provided a convenient way to use Finite Difference approximations to the Transonic Small Disturbance equations to predict the lift on the advancing blade, and incorporate this information in CAMRAD, an existing comprehensive rotorcraft analysis tool. Potsdam et. al. [44] successfully demonstrated the application of the

delta-coupling method to prediction of UH-60 airloads and structural loads for high-speed, low-speed, and high-thrust flight conditions. In this case, the CFD analysis was OVERFLOW-D (an overset URANS solver) and the comprehensive analysis was CAMRAD II. The CAMRAD II aerodynamic model used a lifting line analysis method combined with free-wake, airfoil tables, and a dynamic stall model. A detailed review of the delta-coupling method and the beneficial impact of CFD analysis on rotorcraft aeromechanic is provided by Datta et. al. [45]. The delta-coupling method is currently used in the Helios program [46].

4.3 Theory

An adaptation of the forgoing general purpose methodology is presented for the special class of problems commonly referred to as the “rotor trim problem”. The principle strategy is to convert the governing equations of the rotor system to an equivalent autonomous ODE. When properly converted, one can directly apply the proposed method, Periodic Multiple Shooting (PMS), to determine the states and control values of the rotor system. Furthermore, one can realize a non-trivial cost reduction by modifying the method slightly to exploit symmetries within the state-space of the converted system. Such modifications can realize a cost reduction on the order of the number of blades in the rotor.

4.3.1 Equivalent Autonomous First Order System

The governing equations for a rotor system generally consist of a second order differential equation with periodic forcing. In addition, there is a trim constraint, which relates control parameters (e.g. collective and cyclic control angles) and system states to a set of desired, time averaged, system observations (e.g. rotor thrust and hub moments).

Reduction of high order differential equations to an equivalent first order system is a straightforward process that has been well documented in textbooks (e.g. the book by Hirsch et al. [3]). The treatment of periodic forcing terms has received some attention in the literature (e.g. for example Chapter 7 and 10 of Wiggins [4]). However, there is no canonical approach, and each approach introduces its own modeling tradeoffs. The treatment of integral constraints does not appear to have received any attention in the literature. Thus, the subsequent two sections will address each of these later two issues in turn.

4.3.1.1 Treatment of Periodic Forcing Terms

Consider a non-homogeneous first order system governed by the equation

$$\dot{x} = f(x, \cos(\omega t), \sin(\omega t)). \quad (4.28)$$

The objective is to identify an autonomous system

$$\dot{y} = g(y) \quad (4.29)$$

together with a map χ such that $\chi(y)$ is a solution to (4.28) (i.e. $\chi(y) = x$), and y is both periodic and continuous on $\mathbb{R}$. Furthermore, if equation (4.28) possesses a single periodic attractor, then so should (4.29).

A strategy used by Wiggins [4] is based on the observation that the function $\theta : \mathbb{R} \rightarrow \mathbb{R}$, given by the equation

$$\theta(t) = \omega t \mod 2\pi, \quad \forall t \in \mathbb{R} \quad (4.30)$$

is the solution to the differential equation

$$\dot{\theta} = \omega. \quad (4.31)$$

Thus, one may rewrite (4.28) as

$$\dot{x} = f(x, \cos(\theta), \sin(\theta)), \quad (4.32)$$

$$\dot{\theta} = \omega. \quad (4.33)$$

The above system may be rewritten in the same concise form as (4.29), by taking $y = (x, \theta)$, and χ as the relation $\chi((x, \theta)) = x$ for all $(x, \theta) \in \mathbb{R}^n \times \mathbb{R}$. However, y is not continuous on $\mathbb{R}$, because θ is not continuous on $\mathbb{R}$. If one attempts to avoid the discontinuity by taking $\theta(t) = \omega t$, then θ fails to be periodic. Thus, one cannot use this strategy as a means of enabling the application of PMS.

Another strategy is to look for dynamical system representations of sine and cosine.

Proposition 3. *Suppose $g, h : \mathbb{R} \rightarrow [-1 : 1]$ are functions given by the equations*

$$g(t) = \cos(\omega t) \quad (4.34)$$

and

$$h(t) = \sin(\omega t), \quad (4.35)$$

for $\omega \neq 0$. Then, for all $t \in \mathbb{R}$, f and g may also be expressed as

$$g(t) = s_1(t), \quad (4.36)$$

$$h(t) = -s_2(t)/\omega, \quad (4.37)$$

where s_1 and s_2 are the coordinate functions of the trajectory, passing through the point $(1, 0)$ at $t = 0$, for the dynamical system given by

$$\begin{bmatrix} \dot{s}_1 \\ \dot{s}_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -\omega^2 & 0 \end{bmatrix} \begin{bmatrix} s_1 \\ s_2 \end{bmatrix}. \quad (4.38)$$

Proof. Substitute equation (4.34) into (4.36), (4.35) into (4.37), and then solve for s_1 and s_2 to yield

$$s_1 = \cos(\omega t), \quad (4.39)$$

$$s_2 = -\omega \sin(\omega t). \quad (4.40)$$

Setting $t = 0$ in the above equations shows that $(s_1, s_2) = (1, 0)$. Taking derivatives with respect to time gives

$$\dot{s}_1 = -\omega \sin(\omega t)$$

$$= s_2$$

$$\dot{s}_2 = -\omega^2 \cos(\omega t)$$

$$= -\omega^2 s_1.$$

Writing the above equations in matrix form yields (4.38).

Conversly, it can be shown that equations (4.39) and (4.40) constitute a unique solution to (4.38) for initial conditions $(s_1, s_2) = 0$ [3]. $\square$

By Proposition 3, one may then rewrite (4.28) as the autonomous system of differential equations

$$\dot{x} = f(x, s_1, -s_2/\omega)$$

$$\dot{s}_1 = s_2$$

$$\dot{s}_2 = -\omega^2 s_1$$

The above system is periodic and continuous on $\mathbb{R}$. The only issue is that the above system does not posses a unique periodic attractor. In fact, there is a continuum of periodic solutions. Given a small perturbation in the initial condition, and solution will move to a higher (lower) orbit.

The proposed solution is to use a limit cycle model, to ensure that the periodic attractor of (4.28) (if it exists) remains unique.

Proposition 4. *Suppose $g, h : \mathbb{R} \rightarrow [-1 : 1]$ are functions given by the equations*

$$g(t) = \cos(\omega t) \tag{4.41}$$

and

$$h(t) = \sin(\omega t), \tag{4.42}$$

for $\omega \neq 0$. Then, for all $t \in \mathbb{R}$, f and g may also be expressed as

$$g(t) = s_1(t), \tag{4.43}$$

$$h(t) = -s_2(t)/\omega, \tag{4.44}$$

where s_1 and s_2 are the coordinate functions of the trajectory, passing through the point $(1, 0)$ at $t = 0$, for the (nonlinear) dynamical system given by

$$\begin{bmatrix} \dot{s}_1 \\ \dot{s}_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -\omega^2 & 0 \end{bmatrix} \begin{bmatrix} s_1 \\ s_2 \end{bmatrix} - \begin{bmatrix} 0 \\ 2\omega\zeta(s_1, s_2) \end{bmatrix}, \quad (4.45)$$

$$\zeta(s_1, s_2) = ((s_1)^2 + (s_2/\omega)^2 - 1) c. \quad (4.46)$$

Furthermore, the foregoing dynamical system has only one periodic attractor.

Proof. Equations (4.41), (4.42), (4.43), and (4.44) may be combined and solved for s_1 and s_2 to give

$$s_1 = \cos(\omega t),$$

$$s_2 = -\omega \sin(\omega t).$$

Substituting the above results into (4.46) gives

$$\zeta(s_1(t), s_2(t)) = 0$$

for all $t \in \mathbb{R}$. Thus, equation (4.45) reduces to (4.38). As was already shown for Proposition 3, the above expressions for s_1 and s_2 satisfy (4.38). Furthermore, setting $t = 0$ gives $(s_1, s_2) = (1, 0)$.

Uniqueness of the periodic attractor is now shown.

□

4.3.1.2 Treatment of Trim Constraints

One of the challenges of the trim problem is the presence of algebraic constraints that relate rotor thrust and moments to control parameters. A convenient

solution is to use the autopilot method described in the foregoing section. Replacing trim constraints with an autopilot model and combining it with the above governing equations in autonomous form, results in a single autonomous system that may be solved using Periodic Multiple Shooting.

4.3.2 Initial Solution Estimates

As an iterative algorithm, Periodic Multiple Shooting requires the specification of an initial solution. This raises the question of how to best construct an initial solution. In principle, the initial solution should be chosen to be as close to the true solution as possible. Ultimately, the best strategy for constructing an initial solution is going to be problem dependent. Such strategies range from utilizing low cost models to estimate the solution to simply taking an educated guess.

4.3.3 Exploiting Symmetries

It is common practice to design helicopter rotors with identical, equally spaced, blades. This design practice serves to mitigate aircraft vibration. A beneficial side-effect of this design practice, is that one can exploit the symmetry of the system to simplify the analysis. A classical example of this strategy put to practice is the Fast Floquet Method [19, 20], which was discussed in Chapter 1. Such an approach can be directly applied to the Periodic Multiple Shooting.

4.4 Numerical Tests

4.4.1 Rigid Bladed Rotor with Flap DOF

As a first step to understanding the application of PMS to a rotor model and assessing its utility, a test using a simple model of a rotor in forward flight has been performed. Such a model is not adequate for “real-world” engineering analysis of rotor loads or vibration. Key physical behaviors, such as flap-torsion coupling, and unsteady aerodynamic effects, are not represented. Nevertheless, use of simple models does help to facilitate an understanding of how a method succeeds (or fails) on a fundamental level. Furthermore, well-documented examples, which range in complexity, provide useful pedagogical waypoints for those who wish to assimilate these ideas and subsequently improve upon them or put them to practice.

Two methods were applied to this rotor model: a conventional time marching solution strategy with an autopilot, and the PMS method (with autopilot). The results of the two methods are reported below. It will be seen that the PMS method yields an identical solution as the time marching method, with the added benefit of enabling parallel computation.

4.4.2 Rotor System Model and Flight Condition

The rotor system is an articulated rotor with 4 identical blades, equally spaced around the azimuth. The rotor radius was $R = 26.83$ ft. The rotational speed of the rotor was a fixed rate of $\omega = 27.0$ rad/sec.

Each blade had a single flap-wise degree-of-freedom. The offset of the flap hinge was $e = 2.683$ ft. The mass distribution of the rotor is a uniform value of $m = 1$ slug/ft². The platform was rectangular with a chord length of $c = 2.0$ ft. A linear lift coefficient was assumed with $c_l(\alpha) = 2\pi\alpha$, where α is the local angle of attack.

The aerodynamic loads acting on each rotor blade was modeled using quasi-steady strip theory. A uniform inflow model was used to model the influence of the rotor wake on the local angle of attack at each blade section.

The rotor system was “flown” at an advance ratio of $\mu = 0.2$ at standard sea level conditions. The shaft angle of attack was fixed at 6.0° (tilted forward).

4.4.2.1 Mathematical Model

The only degree-of-freedom in this model is the flap angle of each blade, β_i , $i \in \{1, \dots, N_b\}$. Each β_i is assumed to belong to the interval $(-b, b)$, where $b \ll 1$ (rad). Thus, $\cos(\beta_i) \approx 1$ and $\sin(\beta_i) \approx \beta_i$.

Invoking the principle of angular momentum balance about the flap hinge yeilds the following second order differential equation for the flap response:

$$\ddot{\beta}_i + \Omega^2 \nu_\beta \beta_i = \frac{1}{I_\beta} M_\beta(\beta_i, \dot{\beta}_i, \theta_i, \cos(\psi_i), \sin(\psi_i)) \quad (4.47)$$

where $I_\beta \in \mathbb{R}$ is the (second) moment of inertia of the blade about the flap hinge, $S_\beta \in \mathbb{R}$ is the first moment of inertia about the flap hinge, $\Omega \in \mathbb{R}$ is the rotational speed of the rotor, $\nu_\beta \in \mathbb{R}$ is the non-dimensional flap frequency

$$\nu_\beta = 1 + \frac{eS_\beta}{I_\beta}, \quad (4.48)$$

and $M_\beta : \mathbb{R}^5 \rightarrow \mathbb{R}$ is the function that maps local flap angle β_i , local flap rate $\dot{\beta}_i$, local pitch angle $\theta_i \in \mathbb{R}$, and local azimuth position, represented by $\cos(\psi_i)$ and $\sin(\psi_i)$, to the instantaneous aerodynamic moment about the flap hinge. (The function M_β is mathematically defined over $\mathbb{R}^5$, but is only physically meaningful over small subset of $\mathbb{R}^5$.)

Assuming a strip theory model of aerodynamic forces, the aerodynamic moment is given by

$$M_\beta(\beta_i, \dot{\beta}_i, \theta_i, \cos(\psi_i), \sin(\psi_i)) = \frac{1}{2} \rho c c_{l\alpha} \Omega^2 R^4 \left(A_\theta \theta_i + A_\lambda \lambda + A_{\dot{\beta}} \dot{\beta}_i + A_\beta \beta_i \right), \quad (4.49)$$

where

$$A_\theta = \left(\frac{1}{4} - \frac{1}{3} \frac{e}{R} \right) (1 + 2\mu \sin(\psi)) + \left(\frac{1}{2} - \frac{e}{R} \right) (\mu \sin(\psi_i))^2, \quad (4.50)$$

$$A_\lambda = -\frac{1}{3} + \frac{1}{2} \frac{e}{R} - \mu \sin(\psi_i) \left(\frac{1}{2} - \frac{e}{R} \right), \quad (4.51)$$

$$A_{\dot{\beta}} = -\frac{1}{\Omega} \left(\frac{1}{4} - \frac{1}{3} \frac{e}{R} + \mu \sin(\psi_i) \left(\frac{1}{3} - \frac{1}{2} \frac{e}{R} \right) \right), \quad (4.52)$$

$$A_\beta = -(\mu \cos(\psi_i)) \left(\frac{1}{3} - \frac{1}{2} \frac{e}{R} \right) - (\mu^2 \sin(\psi_i) \cos(\psi_i)) \left(\frac{1}{2} - \frac{e}{R} \right). \quad (4.53)$$

The instantaneous out of plain force $F_{z,i}$ contributed by blade i is given by

$$F_{z,i}(\beta_i, \dot{\beta}_i, \theta_i, \cos(\psi_i), \sin(\psi_i)) = \frac{1}{2} \Omega^2 R^3 \rho c c_{l\alpha} \left(B_\theta \theta_i + B_\lambda \lambda + B_{\dot{\beta}} \dot{\beta}_i + B_\beta \beta_i \right) \quad (4.54)$$

where

$$B_\theta = \frac{1}{3} \left(1 - \frac{e^3}{R} \right) + \left(1 + \frac{e^2}{R} \mu \sin(\psi_i) \right) + \left(1 - \frac{e}{R} \right) (\mu \sin(\psi_i))^2, \quad (4.55)$$

$$B_\lambda = -\frac{1}{2} \left(1 - \frac{e^2}{R} \right) + \left(1 - \frac{e}{R} \right) \mu \sin(\psi_i), \quad (4.56)$$

$$B_{\dot{\beta}} = -\frac{1}{\Omega} \left(\frac{1}{3} \left(1 - \frac{e^3}{R} \right) + \frac{1}{2} \left(1 - \frac{e^2}{R} \right) \mu \sin(\psi_i) \right), \quad (4.57)$$

$$B_\beta = -\frac{1}{2} \left(1 - \frac{e}{R} \right) \mu \cos(\psi_i) - \left(1 - \frac{e}{R} \right) \mu^2 \sin(\psi_i) \cos(\psi_i). \quad (4.58)$$

The total instantaneous thrust is then

$$T = \sum_{i=1}^{N_b} F_{z,i} \quad (4.59)$$

Since the rotor is articulated (and there is no hinge spring or damper), the instantaneous hub moment is given by

$$M_x = \sum_{i=1}^{N_b} e T_i \sin(\psi_i), \quad (4.60)$$

$$M_y = -\sum_{i=1}^{N_b} e T_i \cos(\psi_i). \quad (4.61)$$

A simple swash-plate control model was used. For $i \in \{1, \dots, N_b\}$, the local pitch angle θ_i is given by the equation

$$\theta_i = \theta_0 + \theta_{1c} \cos(\psi_i) + \theta_{1s} \sin(\psi_i). \quad (4.62)$$

The autopilot model is based on the original linear model proposed by Peters [36]. Denote the target thrust as T^* and target hub moments about the x and y axes as M_x and M_y , respectively. The feedback control law is given by

$$\tau \begin{bmatrix} \ddot{\theta}_0 \\ \ddot{\theta}_{1c} \\ \ddot{\theta}_{1s} \end{bmatrix} + \begin{bmatrix} \dot{\theta}_0 \\ \dot{\theta}_{1c} \\ \dot{\theta}_{1s} \end{bmatrix} = \begin{bmatrix} k_0 & 0 & 0 \\ 0 & k_c & 0 \\ 0 & 0 & k_c \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & -b \\ 0 & -b & 1 \end{bmatrix} \begin{bmatrix} T^* - T \\ M_x^* - M_x \\ M_y^* - M_y \end{bmatrix}. \quad (4.63)$$

where

$$b = \frac{8(\nu_\beta - 1)}{\gamma}. \quad (4.64)$$

The control parameter τ is the time constant, and the control parameters k_0 and k_c are the gains corresponding to the collective and cyclic control parameters, respectively.

To facilitate the application of PMS, equations (4.47) and (4.63) were converted to equivalent first order differential equations, and $\cos(\psi_i)$ and $\sin(\psi_i)$ was replaced with equivalent limit cycle model.

4.4.2.2 Numerical Model

Trapezoidal integration was used for this test, as it is energy conserving. Use of a dissipative method, such as Implicit Euler, requires one to use an excessively small time step size to yield a solution with acceptable accuracy.

The linear state advancement model was based on a numerically consistent formulation.

The time step size was fixed at $\Delta t = 2\pi/(\Omega n_s)$, where $n_s = 1024$. This time step corresponds to approximately 0.351° change in azimuth. Use of 1440 steps per cycle (a step size of 0.25° of azimuth), is not uncommon in rotorcraft analysis. The choice of 1024 steps per cycle was preferred over 1440 since it is a power of 2. Use of a power of 2 facilitates parallel scalability studies.

4.4.2.3 Test Results

Direct integration of the governing equations was performed over a time interval equal to 100 rotor revolutions. Use of this method for computing rotor response in steady flight is well understood and commonly used in practice. Thus, the results of direct time integration may be used as a reference solution for assessing the relative accuracy and computational efficiency of subsequent Periodic Multiple Shooting analysis.

The full time history of the flap response for all four blades is shown in Figure 4.1. It can be seen that exhibit qualitatively the same transient behavior, and that a periodic flap response is realized after 80 revolutions.

It can also be seen in Figure 4.1 that the local amplitude of the flap response passes through a minima near the 10th revolution. Looking at the corresponding control time history in Figure 4.2, one can see that autopilot initially moves the lateral cyclic in the wrong direction, and then begins to correct near rev 5. At rev 10, the lateral cyclic is passing through zero. During this time the nominal lateral tilt of the rotor is moving from left (port) to right (starboard). Thus the minima in local flap amplitude is due to the autopilot tilting the rotor laterally from left to right.

The autopilot requires about 80 rotor revolutions to determine the trimmed control angles. Figure 4.2 shows autopilot determining the lateral cyclic by revolution 30 (despite initially moving the lateral cyclic in the wrong direction), the collective by revolution 60, and longitudinal cyclic by revolution 80.

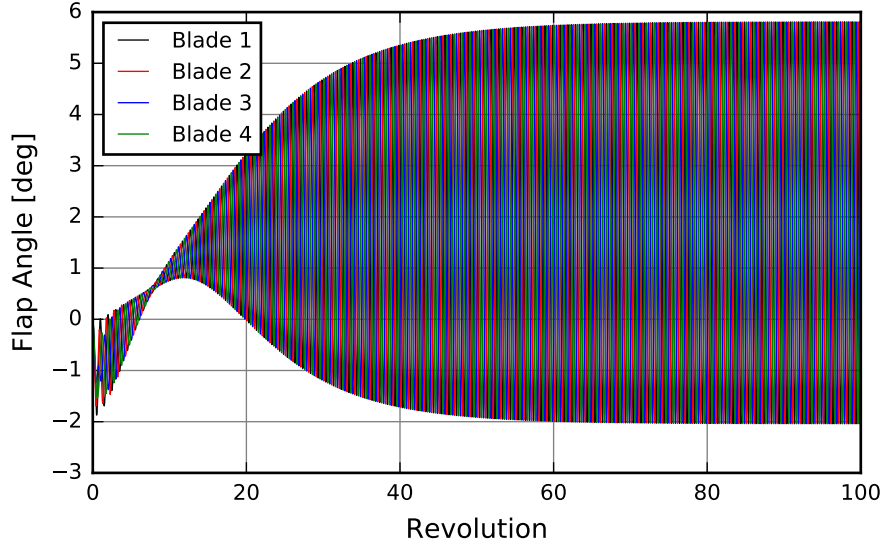


Figure 4.1: Flap response time history for all 4 blades.

The correctness of the autopilot's decision can be verified by inspecting the time histories for the thrust, hub moment about the x -axis, and hub moment about the y -axis, in Figures 4.3, 4.4, and 4.5, respectively.

The flap response over the final revolution is shown in Figure 4.6. Consistent with theory, the flap response of each blade has the same waveform as all other blades, and differs only in phase. The phase shift of each blade is 90° relative to the blade that precedes it. This is the value that one would expect for a 4-bladed rotor. The autopilot does converge and yield constant control values, as shown in Figure 4.7. Figures 4.8, 4.9, and 4.10 confirm that the mean thrust, and hub moments were driven by the autopilot to the specified target values. Furthermore, it can be seen that the thrust and hub moments contain a $4p$ frequency, which is consistent with theory for a 4-bladed rotor.

The results of applying Periodic Multiple Shooting (PMS) are now considered.

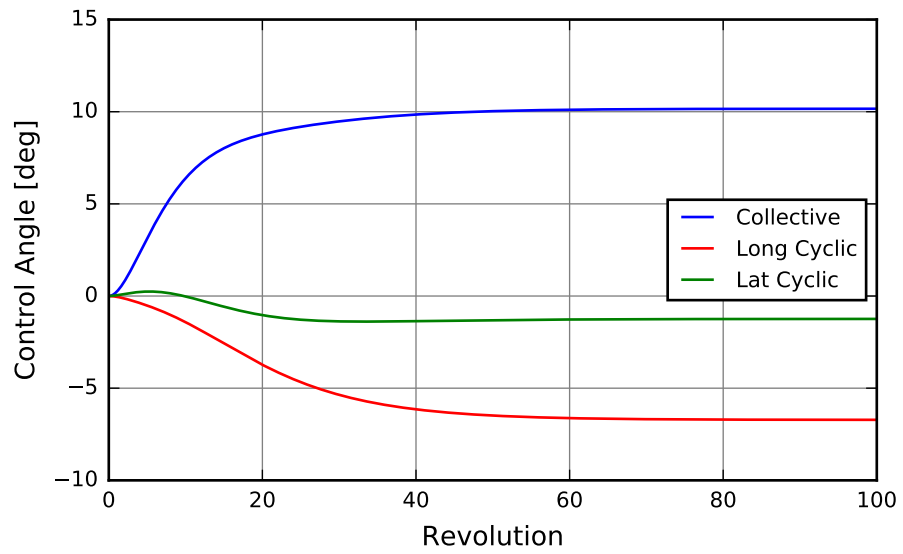


Figure 4.2: Control angle time histories.

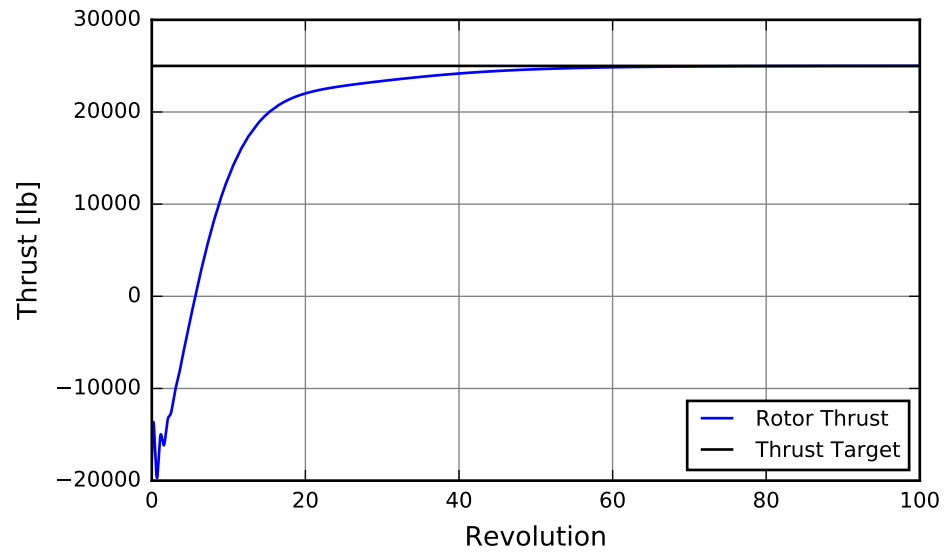


Figure 4.3: Thrust time history.

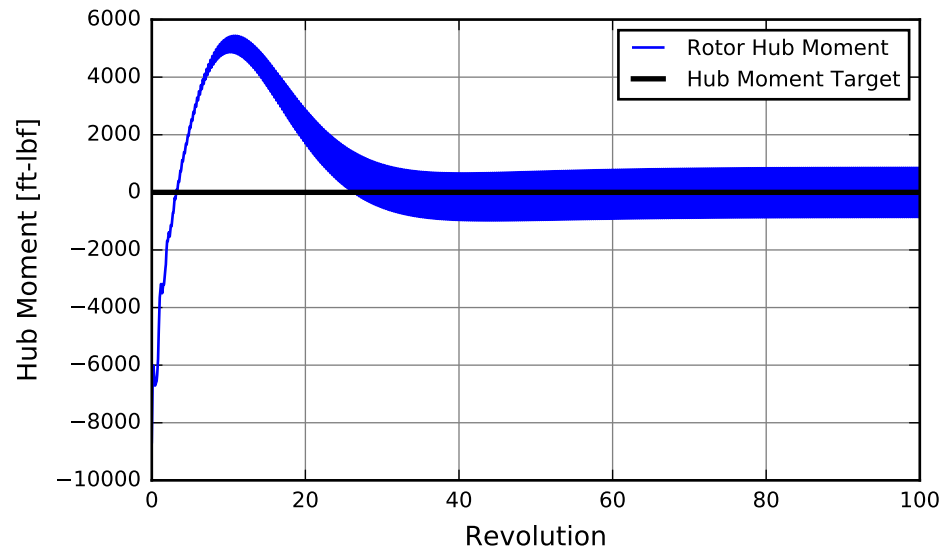


Figure 4.4: Hub moment time history about the x -axis.

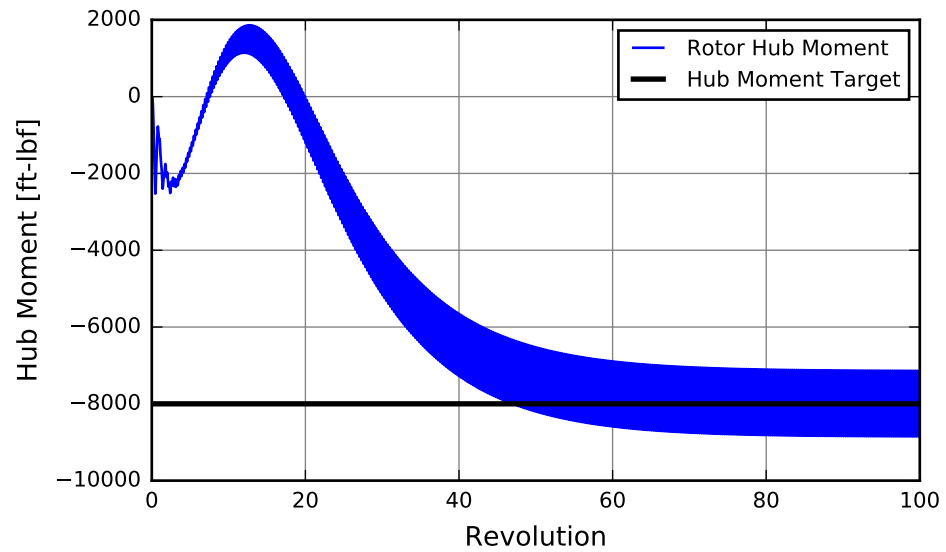


Figure 4.5: Hub moment time history about the y -axis.

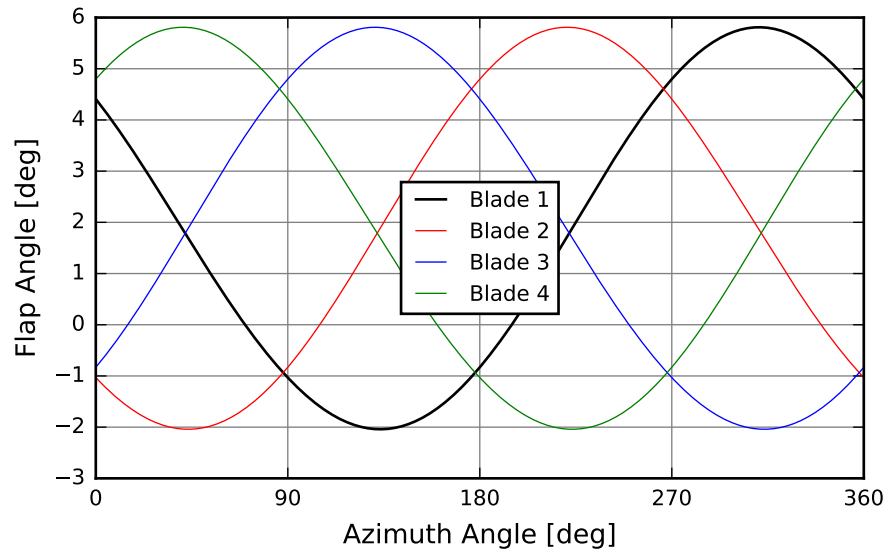


Figure 4.6: Converged flap response.

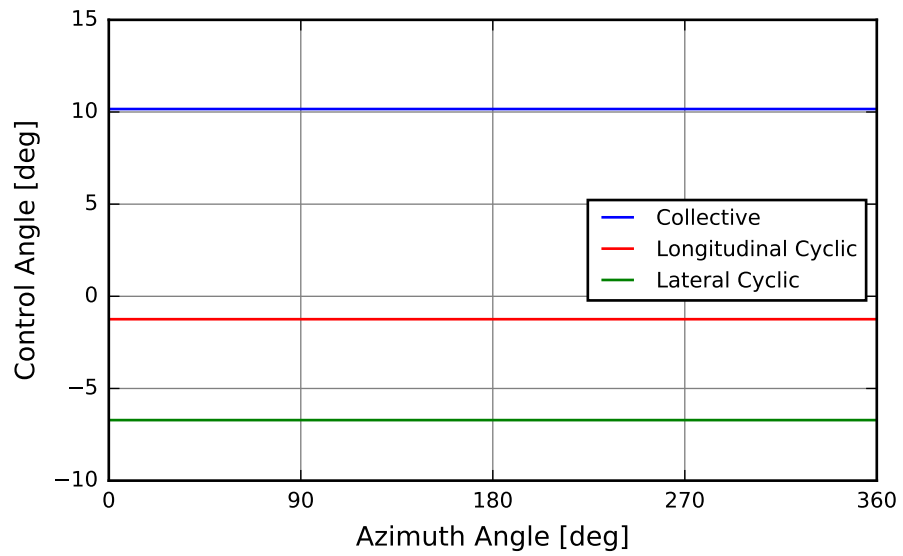


Figure 4.7: Converged control angles.

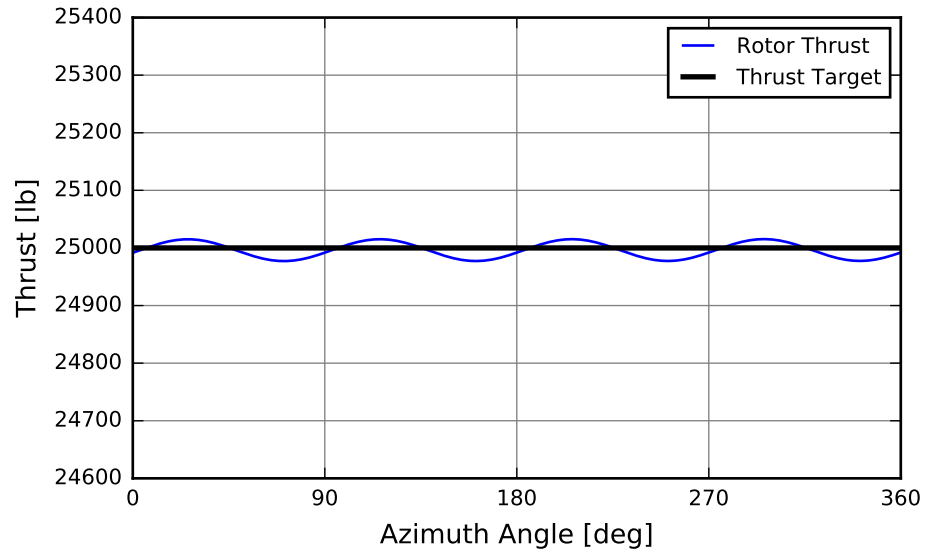


Figure 4.8: Converged thrust.

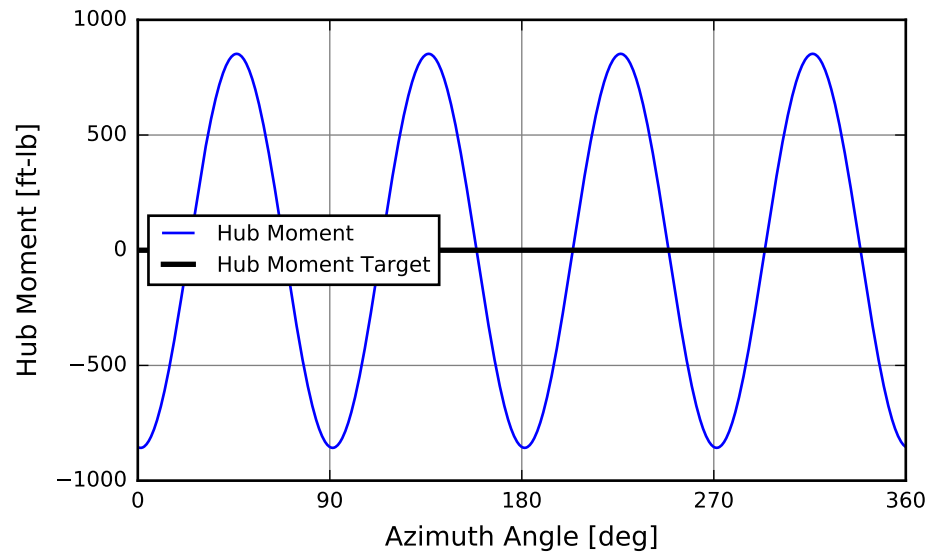


Figure 4.9: Converged hub moment about x -axis.

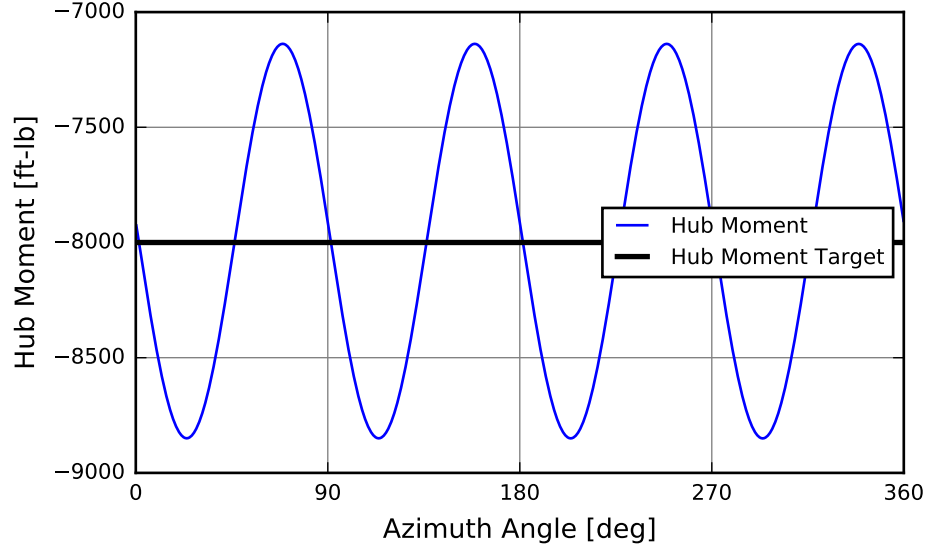


Figure 4.10: Converged hub moment about y -axis.

Four state-space partitions were used, and each partition was associated with a distinct computational process. Figure 4.11 shows the azimuth angle. Figures 4.12, 4.13, 4.14, and 4.15 show the flap response for blades 1, 2, 3, and 4, respectively. The control angles θ_0 , θ_{1c} and θ_{1s} are shown in Figures 4.16, 4.17, and 4.18, respectively. The thrust is shown in Figure 4.19, and hub moment about the x -axis in Figure 4.20 and y -axis in 4.21. Each connected curve in the foregoing figures represents an intermediate solution of the PMS method on a partition of the state-space. There are a total of 100 iterations shown. The curves are colored by partition (process) number.

Recall that the azimuth angle is not prescribed as a direct function of time, but rather as a function of two state variables whose governing equation corresponds to a dynamical system with a single periodic attractor. The governing equation was designed with the intent of yielding an azimuth angle that is equivalent to the

function $\psi(t) = \psi_0 + \Omega t \bmod 2\pi$, where $\psi_0 \in [0, 2\pi]$ is the value of ψ at $t = 0$. Figure 4.11 confirms that the azimuth angle of the rotor model does behave as intended. It can be seen that at the beginning of each shooting phase, ψ begins at the azimuth angle associated with the “left” boundary of each respective partition (process), and increases linearly until the azimuth angle of the “right” boundary is reached. Furthermore, the starting azimuth angle returns to the correct value after each correction step, despite the fact that the other state variables are changing (as will become subsequently evident). This suggests that once a state variable reaches its converged value, it remains at that value for subsequent iterations of PMS, even if the other state variables have not yet converged.

Now, consider the flap response of blade 1 (Figure 4.12). Each partition begins with the initial guess of zero, and the “shooting” step results in a decreasing flap angle. In fact, the results of the first shooting step are nearly identical on all partitions for this case. The initial prediction of a downward trend is correct on partition 0, but incorrect on all other partitions. By the second iteration, however, the incorrect prediction of a downward trend is corrected, and by the 10th iteration, the solution begins to take on a qualitatively correct shape. By the 20th iteration, the intermediate solutions begin to monotonically approach the final converged solution. The rate at which the solver converges appears to be approximately the same on each partition (process). These same trends can also be seen in the flap response for blades 2, 3, and 4 (Figures 4.13, 4.14, 4.15, respectively). The solution for all four blades appears to progress at approximately the same rate.

Throughout the iteration process, the collective can be seen to monotonically

trend towards the converged solution (Figure 4.16), likewise for the Logitudinal cyclic (4.18). The Lateral cyclic, on the other hand, shows an initial trend that progresses in the wrong direction, but is corrected by the 5th iteration (4.17). This behavior of the lateral cyclic mimicks that seen for the direct time marching solution. Likewise, the collective appears to converge the fastest, followed by the lateral cyclic, then the longitudinal cyclic. This convergence trend mimicks the behavior seen for the direct time marching solution. As was seen for the flap response, the rate at which a control parameter converges appears to be indepenent of the partition it is associated with.

The thrust can be seen to steadily converge towards the target value of 25,000 lb (Figure 4.19). The hub moment about the x -axis (Figure 4.20) and y -axis (Figure 4.21) show an initial trend that progresses in the wrong direction, but this trend corrects itself by about the 10th iteration. From that point on, one can see a steady convergence trends towards the mean target values of zero and -8000, respectively.

A comparision between the converged solution obtained using PMS and that obtained using direct time marching is shown in Figures 4.22 through 4.31. In each of these plots, the “reference solution” is the last revolution of the time marching solution (revolution 100). The reference solution is show by a thick black line in all plots. The converged PMS solution is shown by the thin colored curves, where each color corresponds to a state-space partition (process). In all figures, it can be seen that the PMS solution directly overlays the reference solution. This suggest that PMS converges to the same solution as direct time marching, and therefore does not impact the temporal accuracy of the solution.

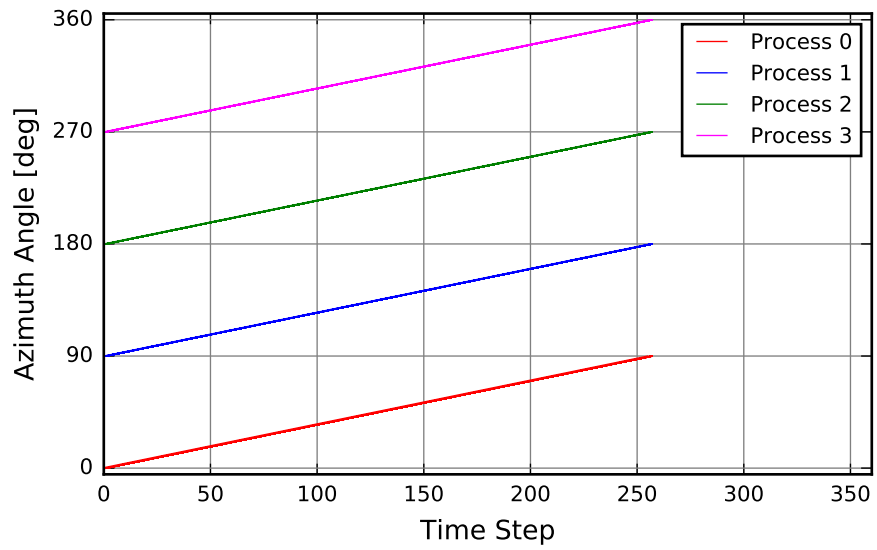


Figure 4.11: Azimuth angle.

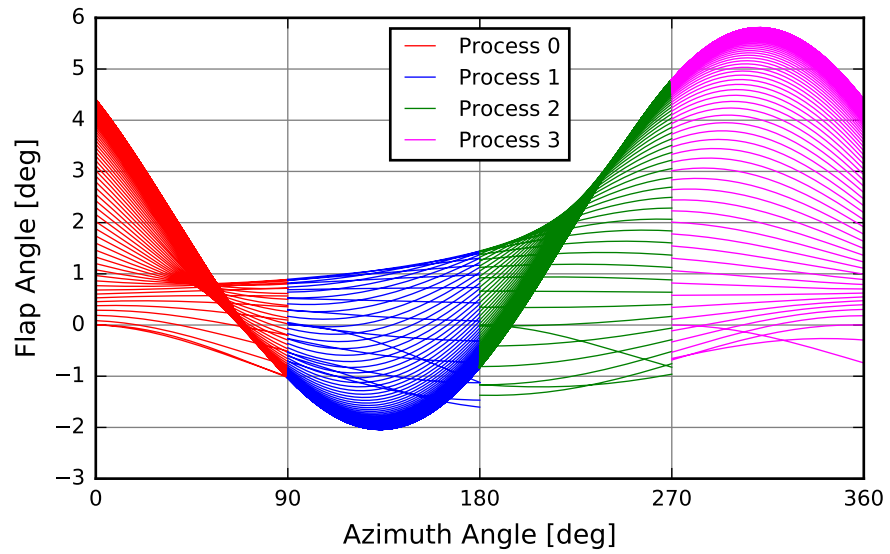


Figure 4.12: Blade 1 flap angle.

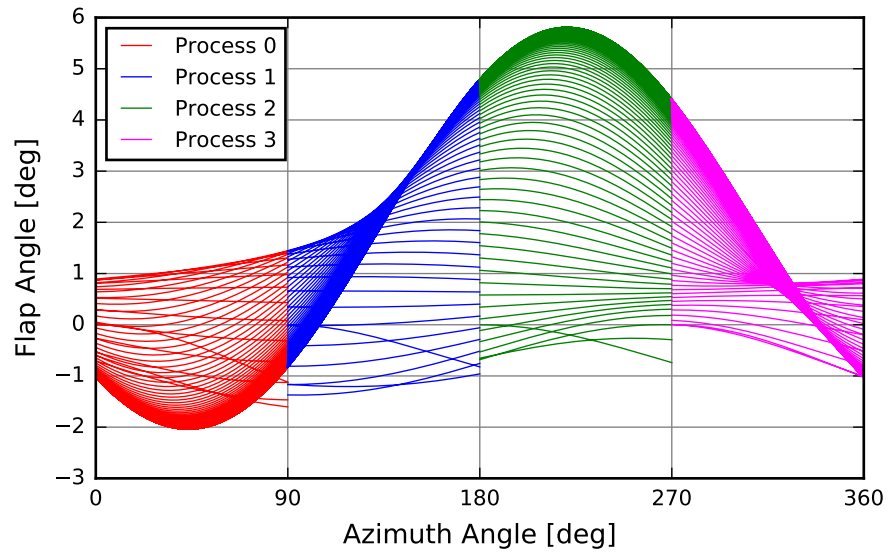


Figure 4.13: Blade 2 flap angle.

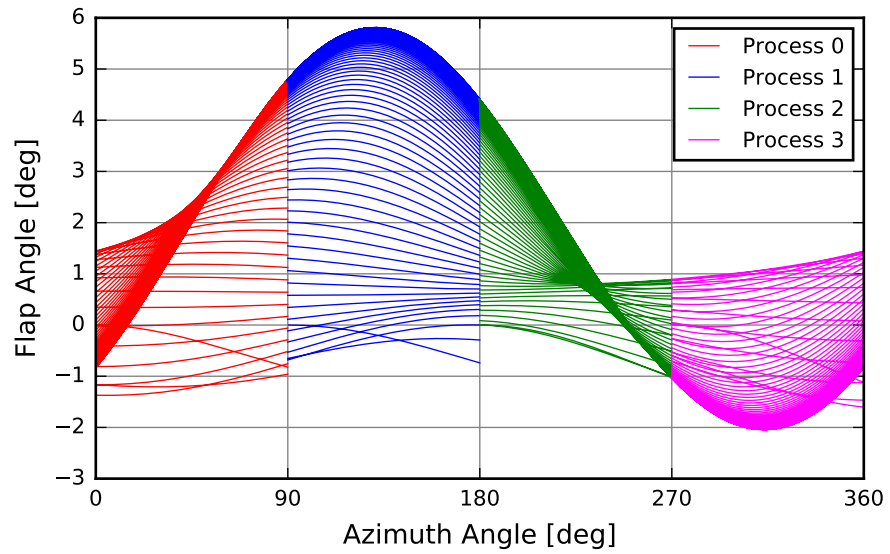


Figure 4.14: Blade 3 flap angle.

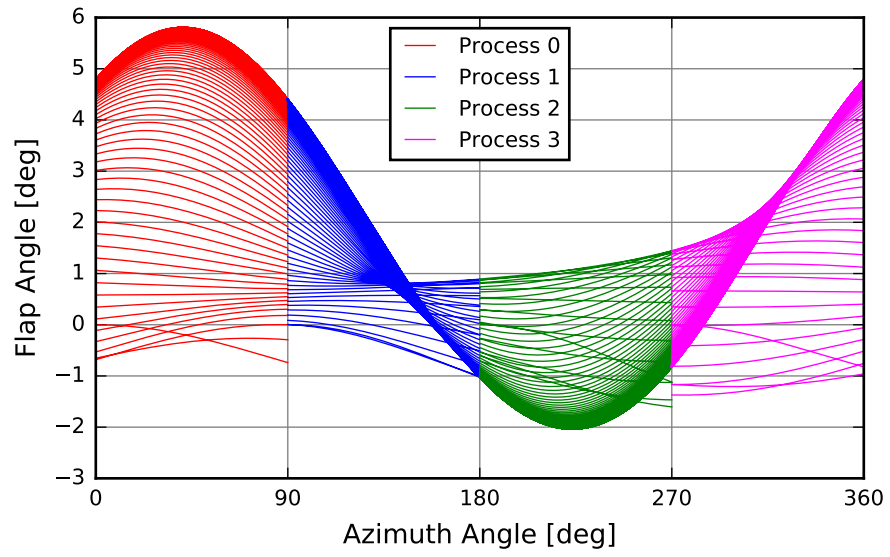


Figure 4.15: Blade 4 flap angle.

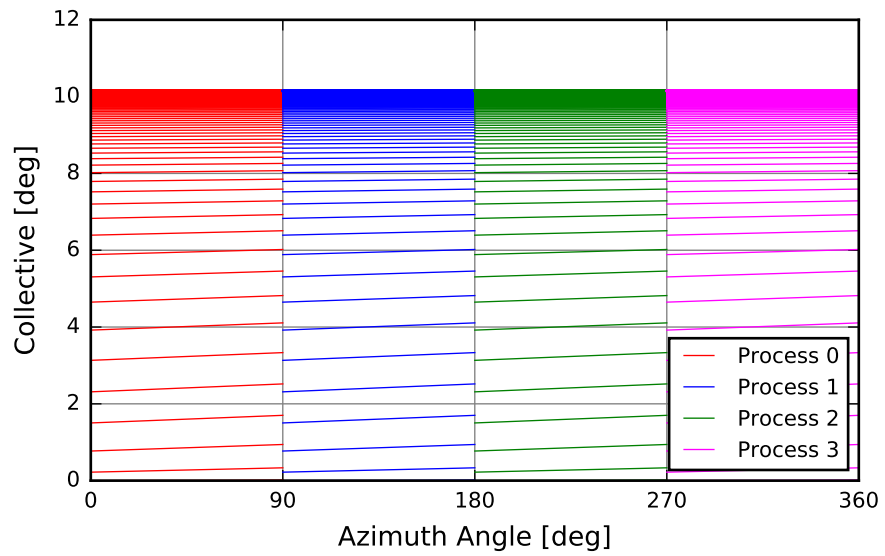


Figure 4.16: Collective.

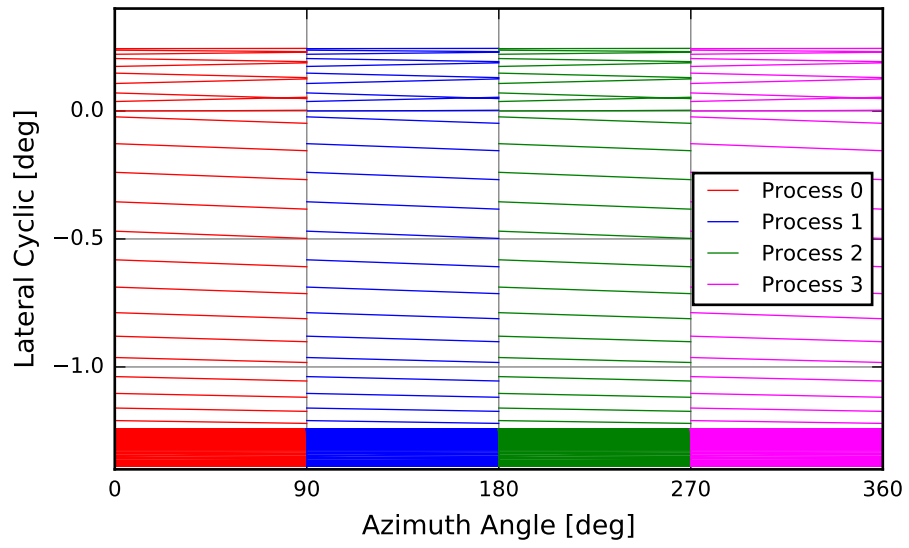


Figure 4.17: Lateral Cyclic.

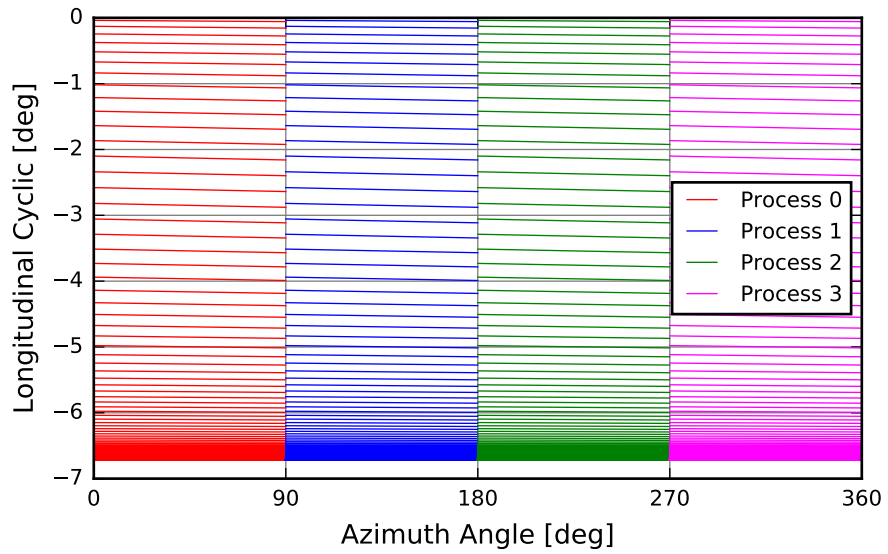


Figure 4.18: Longitudinal Cyclic.

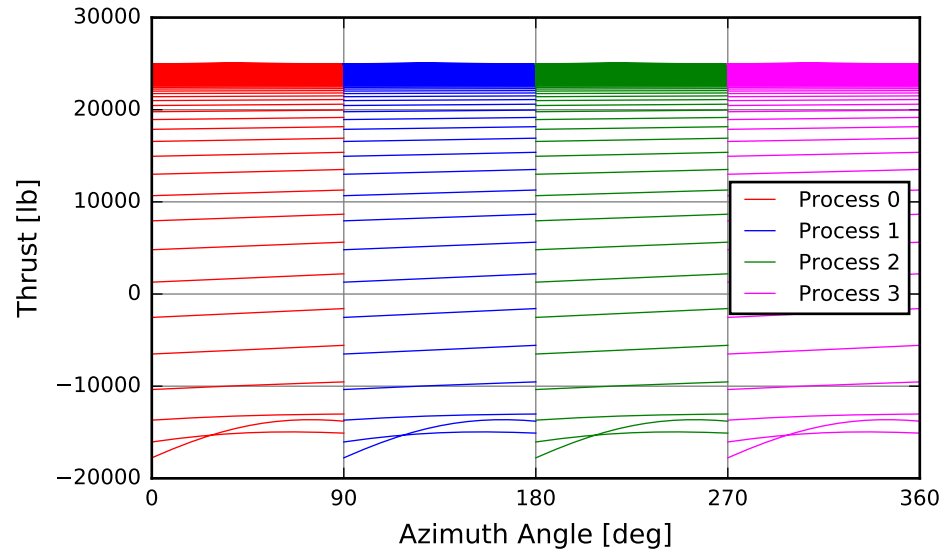


Figure 4.19: Rotor thrust.

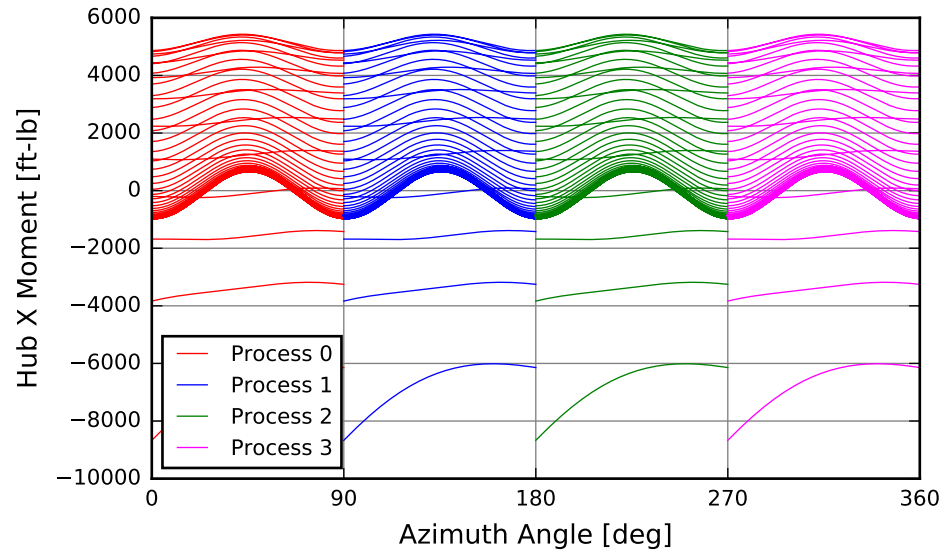


Figure 4.20: Hub moment about x -axis.

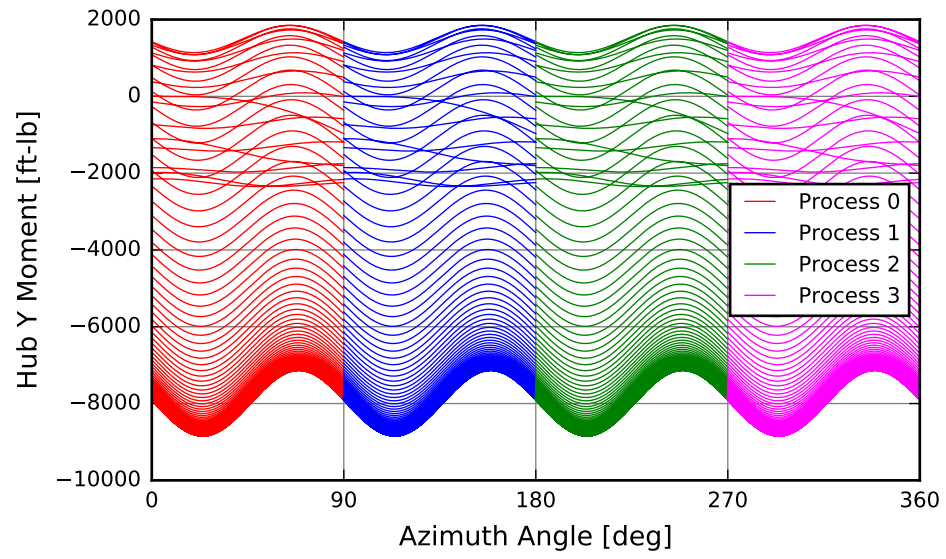


Figure 4.21: Hub moment about y -axis.

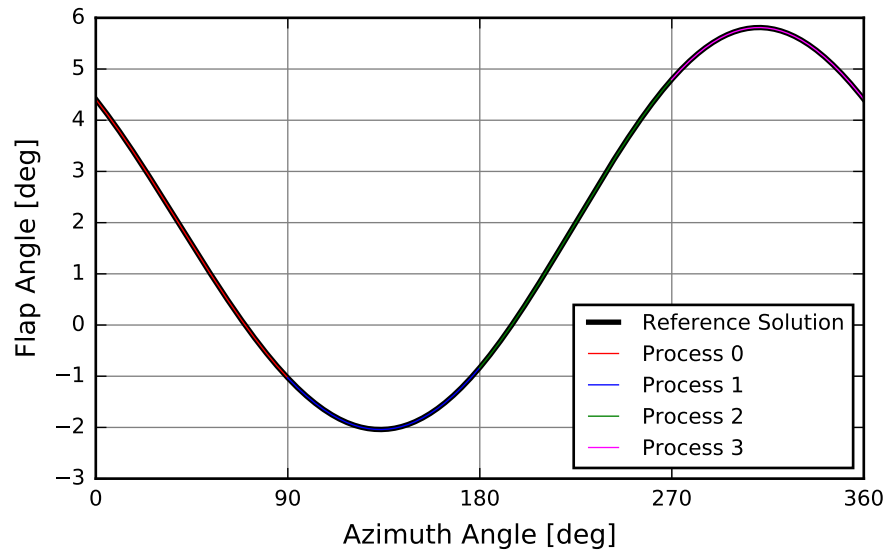


Figure 4.22: Blade 1 flap angle.

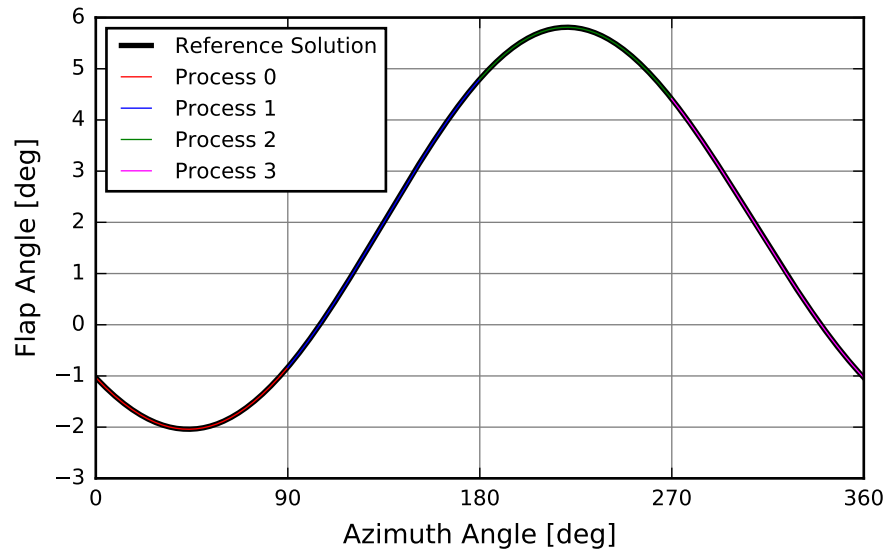


Figure 4.23: Blade 2 flap angle.

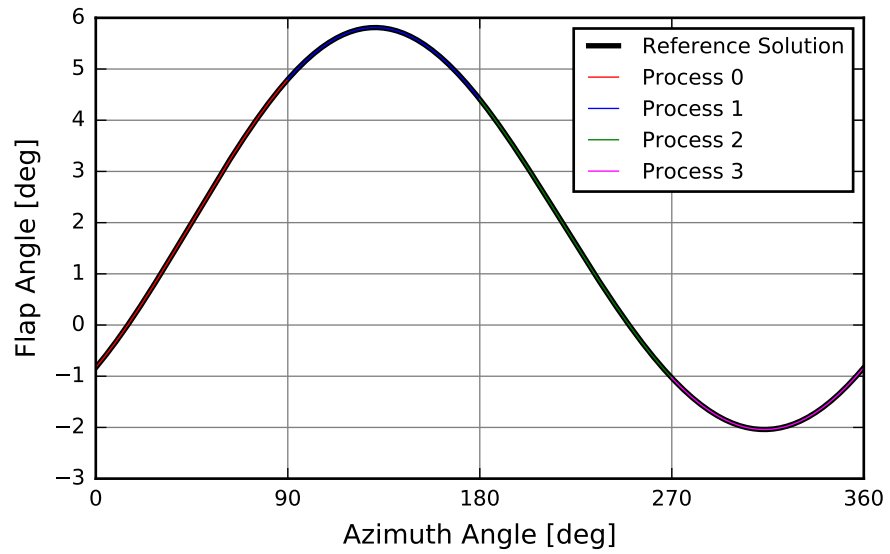


Figure 4.24: Blade 3 flap angle.

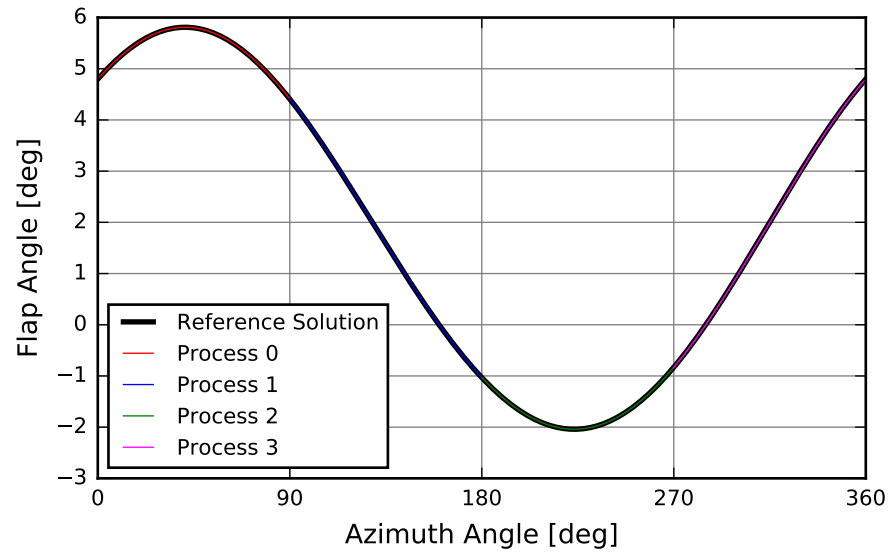


Figure 4.25: Blade 4 flap angle.

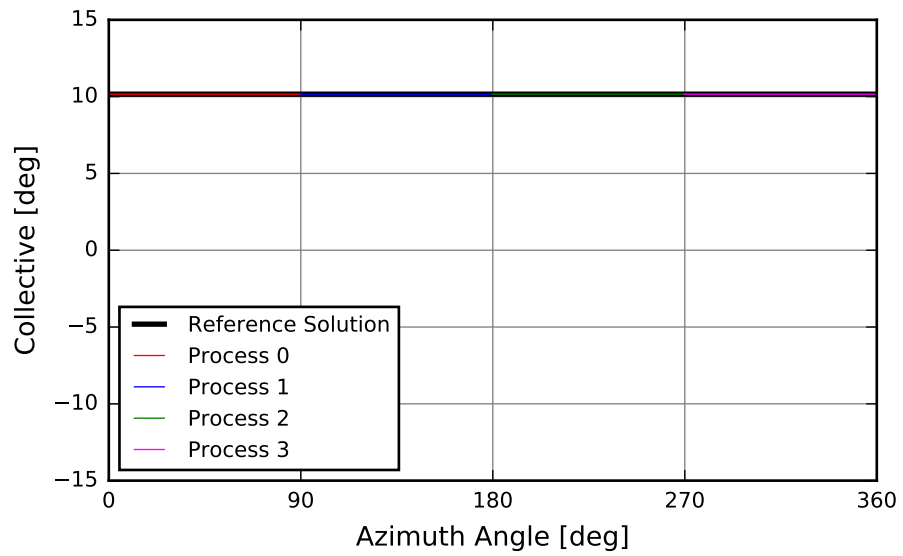


Figure 4.26: Collective.

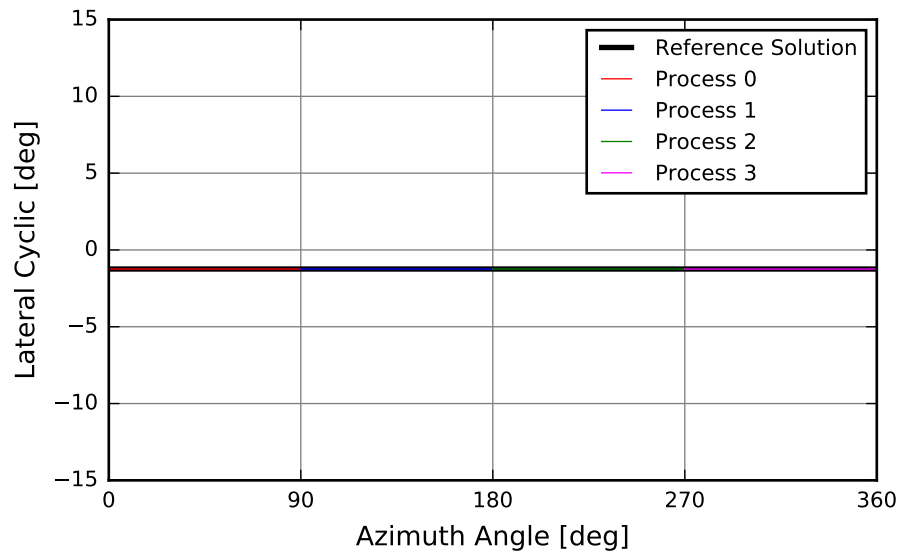


Figure 4.27: Lateral Cyclic.

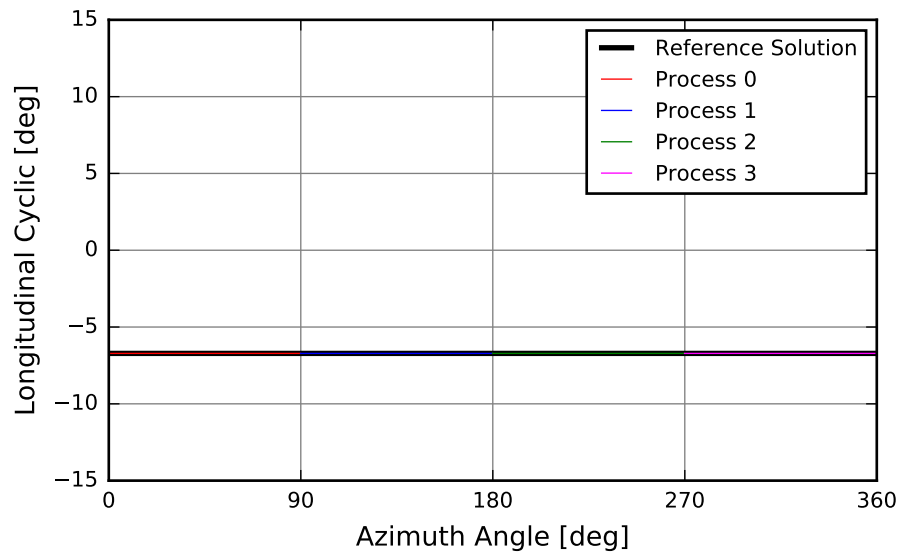


Figure 4.28: Longitudinal Cyclic.

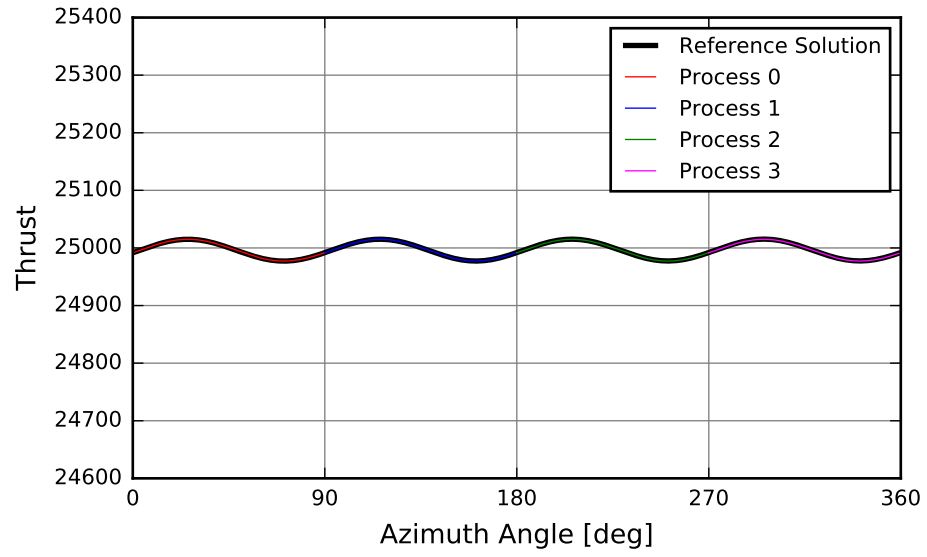


Figure 4.29: Rotor thrust.

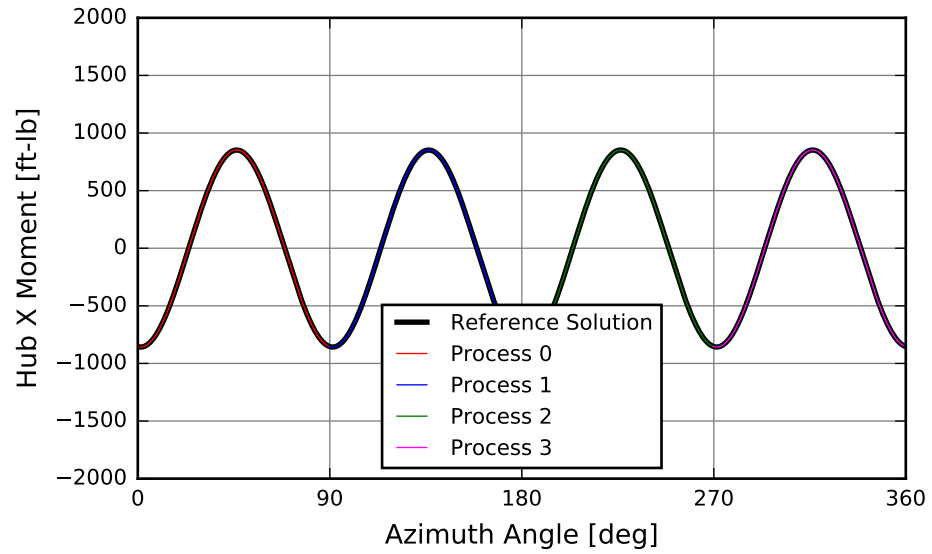


Figure 4.30: Hub moment about x -axis.

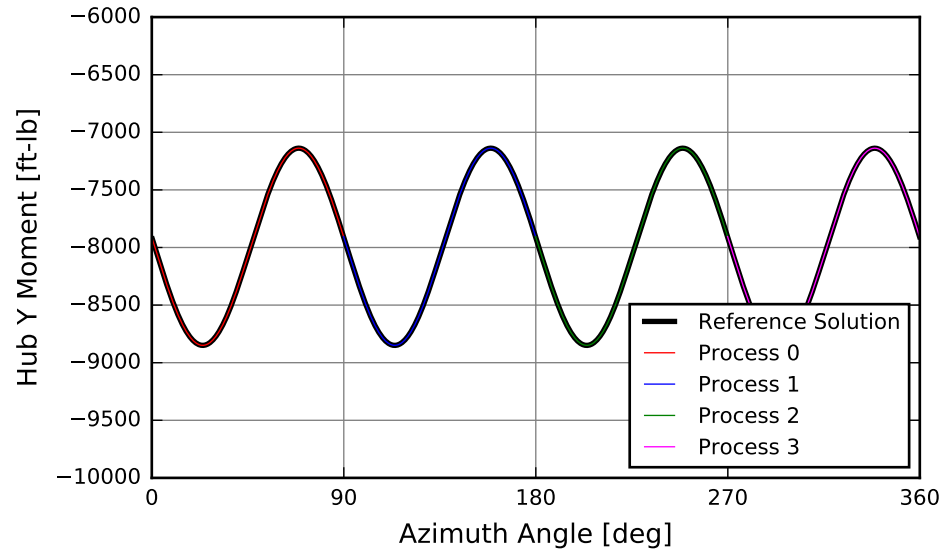


Figure 4.31: Hub moment about y -axis.

Figure 4.32 shows the l2-norm of the difference between each periodic solution estimate generated by PMS as a function of iteration. It can be seen that the change in the solution remains nearly constant for the 20 iterations, and then begins to diminish with each iteration.

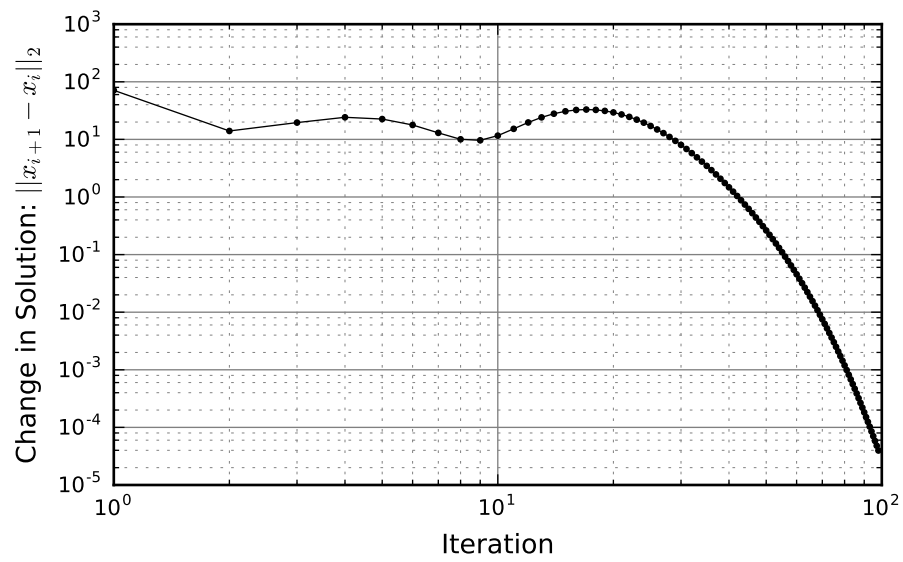


Figure 4.32: Change in solution as a function of iteration.

Algorithm 3 Parallel Symmetric Periodic Multiple Shooting

Require: ϵ , $kmax$, N_b , N_c , INITIALIZE, TIMESTEP, ATPOINCARESECTION, SHIFTRIGHT

```
1: procedure SYMBC( $X$ )
2:    $n \leftarrow (size(X) - N_c)/N_b$ 
3:   CYCLE( $X[0 : nN_b - 1], n$ )
4: end procedure

5: procedure SHOOT( $X, E$ )
6:    $x \leftarrow X$ 
7:   repeat
8:      $x \leftarrow \text{TIMESTEP}(x)$ 
9:   until ATPOINCARESECTION( $x$ )
10:  SHIFTRIGHT( $x$ )
11:  if  $rank = 0$  then
12:    SYMBC( $X$ )
13:  end if
14:   $E \leftarrow x - X$ 
15: end procedure

16: procedure CORRECT( $X, E$ )
17:    $\Delta x \leftarrow 0$ 
18:    $k \leftarrow 0$ 
19:    $r \leftarrow E$ 
20:   loop
21:     SHIFTRIGHT( $r$ )
22:      $\Delta x = \Delta x + r$ 
23:      $k = k + 1$ 
24:     if  $k > kmax$  then
25:       BREAK
26:     end if
27:      $r = Sr$ 
28:   end loop
29:    $X \leftarrow X + \Delta x$ 
30: end procedure

31: INITIALIZE( $X$ )
32: loop
33:   SHOOT( $X, E$ )
34:   if  $\max_{0 \leq i < p} \|E[i]\| < \epsilon$  then
35:     EXIT
36:   end if
37:   CORRECT( $X, E$ )
38: end loop
```

Chapter 5: Application of Periodic Multiple Shooting Method to CFD Rotor Models

Adaptation of Periodic Multiple Shooting to high-fidelity CFD models requires two considerations: how to efficiently construct and evaluate the linearized operators $DP_{i,i-1}$, and how to generalize the methodology to systems governed by Partial Differential Equations (PDEs). The former is addressed by introducing a subspace projection method that draws inspiration from the Arnoldi Method (see Chapter 3). The later is addressed by exploiting the Method of Lines discretization of PDEs, which converts the PDE to a system of ODEs.

Contemporary CFD approaches to problems involving moving bodies utilize overset mesh systems. The overlapping domains of the CFD meshes introduces some additional complications regarding efficient parallel construction and evaluation of $DP_{i,i-1}$ operators. A heuristic solution to this is proposed and subsequently validated using numerical tests.

The proposed approach is verified using an overset URANS CFD model of a UH60-like helicopter rotor model.

5.1 Theory

5.1.1 Reduction of PDE to ODE using Method of Lines

The unsteady Navier-Stokes equations have the form

$$\frac{\partial Q}{\partial t} + F\left(Q, \frac{\partial Q}{\partial x}, \frac{\partial Q}{\partial y}, \frac{\partial Q}{\partial z}, \frac{\partial^2 Q}{\partial x^2}, \frac{\partial^2 Q}{\partial y^2}, \frac{\partial^2 Q}{\partial z^2}\right) = 0$$

By approximating the spatial derivatives with appropriate numerical discrete operators, one arrives at the semi-discrete equations

$$\frac{\partial Q}{\partial t} + \tilde{F}(Q) = 0$$

which is a coupled system of ODEs. Once in this form, the Periodic Multiple Shooting method may be directly applied.

5.1.2 Methodology

The proposed Periodic Multiple Shooting method with subspace approximation of $DP_{i+1,i}$ operators was tested using the overset URANS CFD solver OVERTURNS. The required modifications to OVERTURNS included the addition of subroutines to execute the shoot and correct steps of the Periodic Multiple Shooting method, and modification of the IO subroutines to write solution files for each time partition to separately named files and directories. No modification of the core solver was needed, nor was any modification to the oversetting algorithms required.

OVERTURNS was executed within the TIFAS (Topology Independent Framework Aeromechanics Simulation) framework. TIFAS managed the coupling of pre-

scribed blade motions to OVERTURNS. The advantage of using TIFAS was that it allows executing OVERTURNS using either prescribed structural deformation or a coupled structural mechanics model without having to make any changes to the CFD solver.

The test case was UH60-like rotor in high-speed level flight. The high speed flight case demonstrates strong, unsteady shock wave formation on the advancing blades. Since the flow is locally, highly transonic, this flight condition demonstrates one of the cases where CFD is essential to modeling helicopter rotor aerodynamics.

All 4 rotor blades were modeled. The overset CFD mesh system consisted of 4 cylindrical background grids (Figure 5.2, and 4 body fitted grids (Figure 5.1). Each blade mesh had 129 points in the wrap-around direction, 129 points in the spanwise direction, and 65 points in the normal direction. Each cylindrical quadrant had 50 points in the azimuthal direction, 50 points in the radial direction, and 129 points in the axial direction. The overset mesh methodology was used to transfer information between the blade meshes and quadrant meshes. The background quadrant meshes shared overlapping points and cells; thus no interpolation was needed. Time integration was performed using Implicit Euler. The implicit system was resolved using Newton iteration with LU-SGS factorizations at each time step.

The blade deformations used in this test were obtained from a fully coupled time-marching simulation of this same rotor and flight condition using OVERTURNS and the structural dynamics modeling library RODYMOL. The UH60-like rotor system was modeled in RODYMOL using geometrically exact beams for each rotor blade. The pitch horn and lag damper pickups were modeled using rigid,

massless, links. The lag damper was an approximate linear model. The elastic stiffness of the pitchlink was modeled as an equivalent linear spring. The swashplate was modeled as a pitch actuator with vertical displacement for collective input, and angular displacement for cyclic input. See Figure 5.3.

OVERTURNS and RODYMOL were coupled using TIFAS. TIFAS supports partitioned fluid-structure modeling, with information exchanged between CFD and CSD according to the Conventional Serial Staggered method. The coupled simulation was initialized using free-stream initial solution for CFD and zero elastic deformations for the CSD model. Then the coupled simulation was time-marched until the structural dynamic response and aerodynamic loads became periodic. Time step size corresponded to 0.125° of azimuth. Approximately 20 rotor revolutions were required to achieve a nearly periodic solution. The elastic twist, flapwise displacement, lagwise displacement of each rotor blade was saved to file. The blade motions for the last revolution were then passed through a Fourier filter to remove any small transient errors that remained.

Using the periodic elastic blade motions from the time-marching OVERTURNS/RODYMO simulation, the flow solution was recomputed using OVERTURNS with Periodic Multiple Shooting. The flow solution was initialized to free-stream conditions. Iterations of the Periodic Multiple Shooting method were executed until the continuity errors fell below 0.5×10^{-3} .

Tests were executed on Zaratan, a high performance compute cluster at the University of Maryland. Each compute node consisted of a single 2.45 GHz AMD Epyc CPU with 128 cores per node and 512GB of memory. Care was taken when

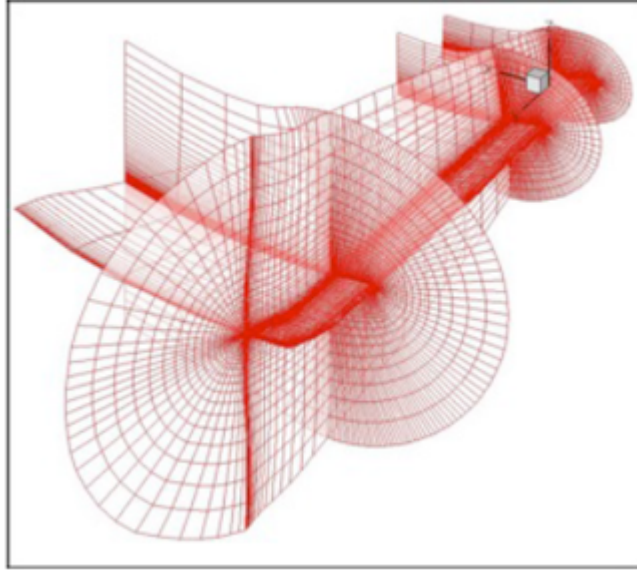


Figure 5.1: Body fitted blade mesh.

launching jobs to ensure that each test run had exclusive access to its assigned compute nodes (i.e. no other jobs were being executed on those nodes). Exclusive use of compute nodes helps to mitigate performance degradation due to memory and IO contention between processes.

OVERTURNS with Periodic Multiple Shooting was executed for 1, 2, 5, and 10 time partitions. The number of space partitions was held fixed at 12 for each test case. Thus, the total number of processors used ranged was 12, 24, 60, and 120, respectively. Parallel communication was done using MPI. Separate communicators were constructed to simplify communication between meshes that shared spatial boundaries and those that shared temporal boundaries.

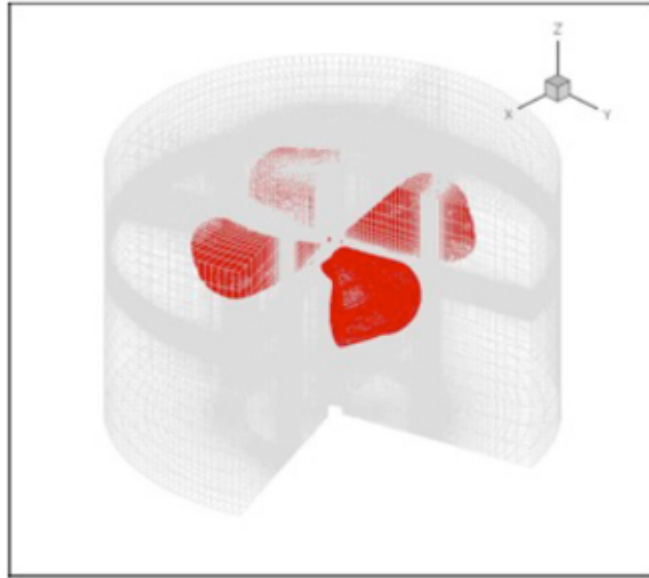


Figure 5.2: Cylindrical background mesh system.

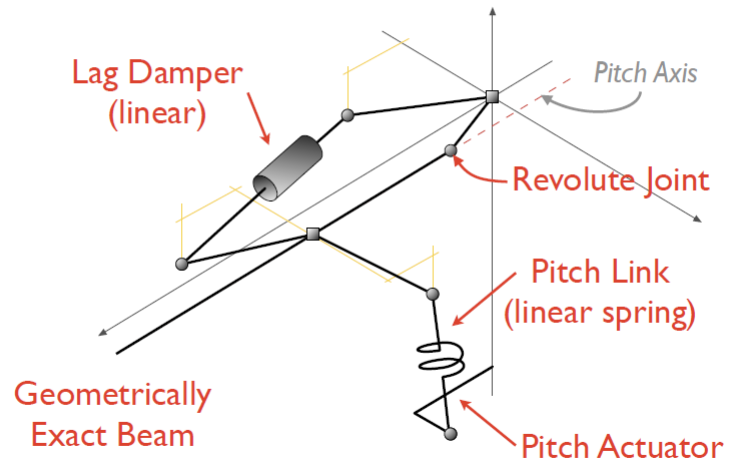


Figure 5.3: Schematic model of rotor system.

5.1.3 Results

A plot of the chordwise normal force at blade midspan is shown in Figures 5.4 through 5.7 for a maximum time partition of 2, 4, 5, and 10, respectively. All test cases converge towards the same solution as the original time marching result. It can also be seen that increasing the number of time partitions appears to coincide with a slight degradation in the convergence rate. This apparent degradation in the convergence rate is at least partially due to each time partition starting from free-stream conditions. When initial conditions are naively selected for each time partition, increasing the number of time partitions increases the initial error. Thus, additional iterations are required.

The parallel speedup obtained from this test is shown in Figure 5.8, and the corresponding parallel efficiency in Figure 5.9. Can be seen that, despite the converge being slightly degraded as more time partitions are added, the parallel efficiency still remains above 80% for the range of processes considered. This translates to approximately a factor of 10 speedup over using conventional spatial domain decomposition only.

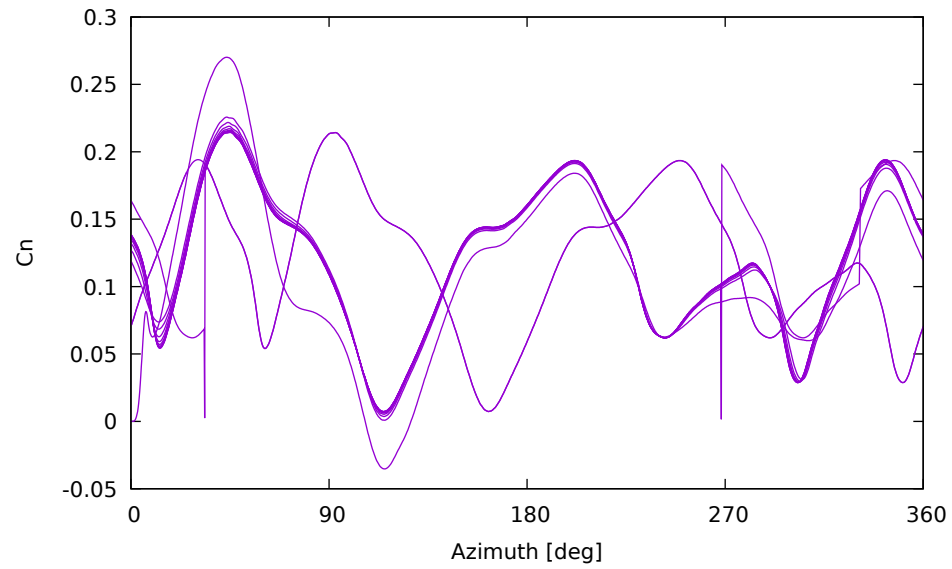


Figure 5.4: Normal force at midspan of rotor blade using one time partition.

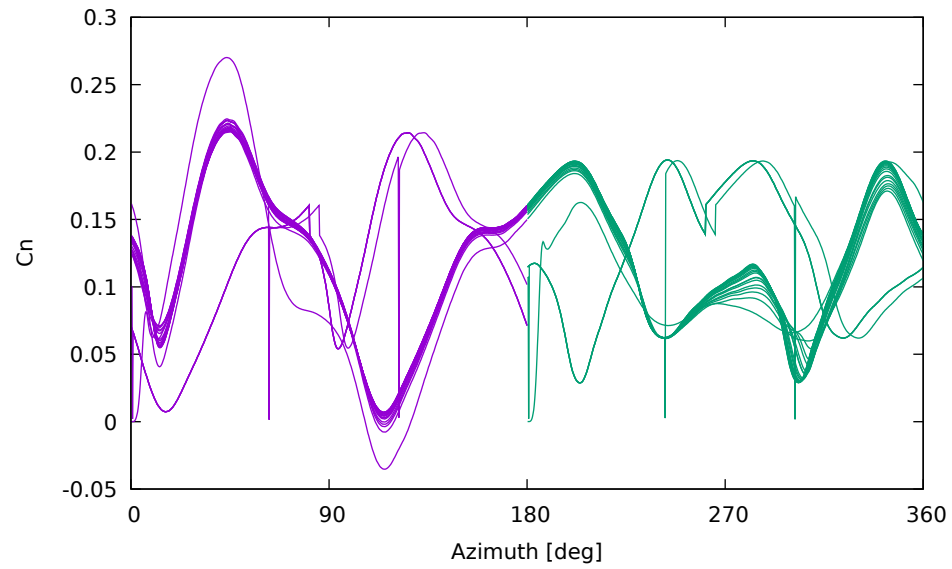


Figure 5.5: Normal force at midspan of rotor blade using two time partitions.

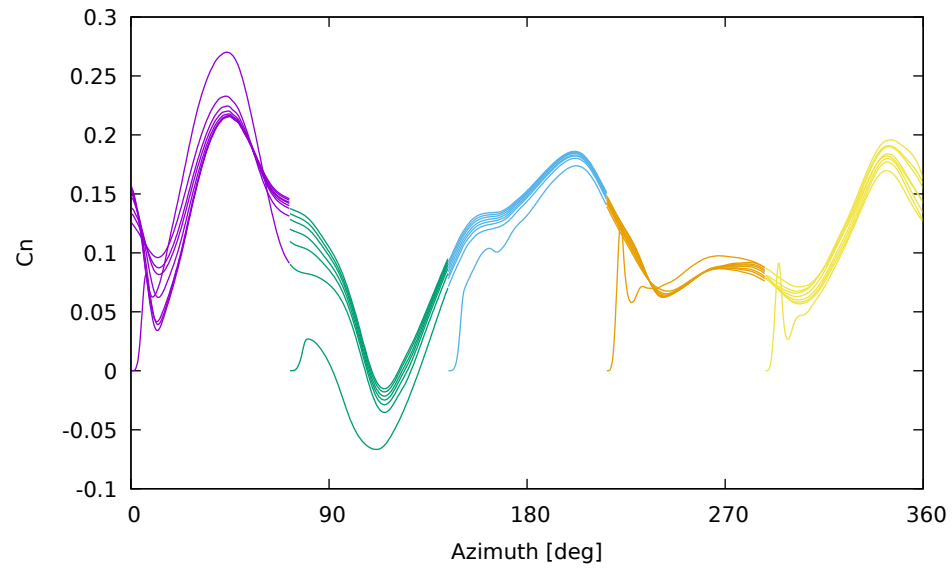


Figure 5.6: Normal force at midspan of rotor blade using 5 time partitions.

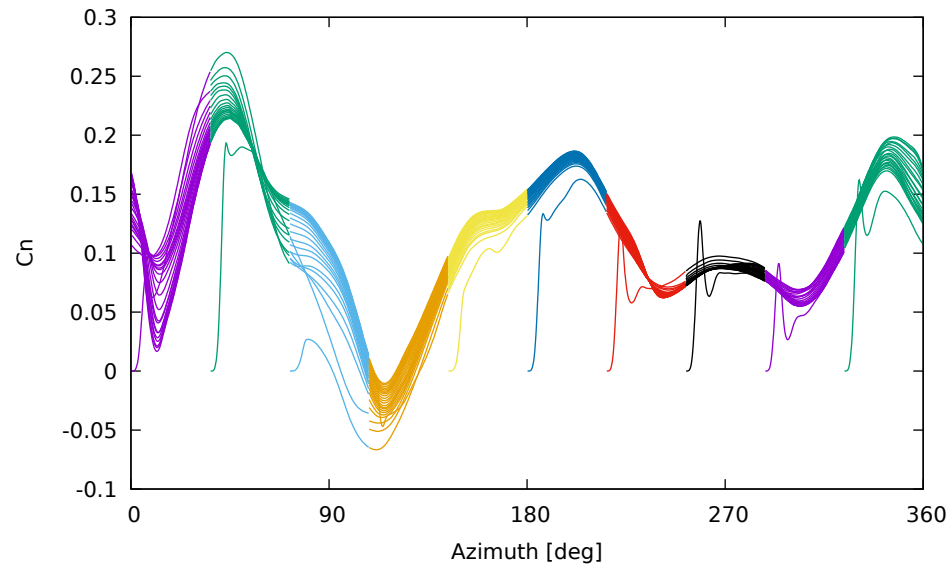


Figure 5.7: Normal force at midspan of rotor blade using 10 time partitions.

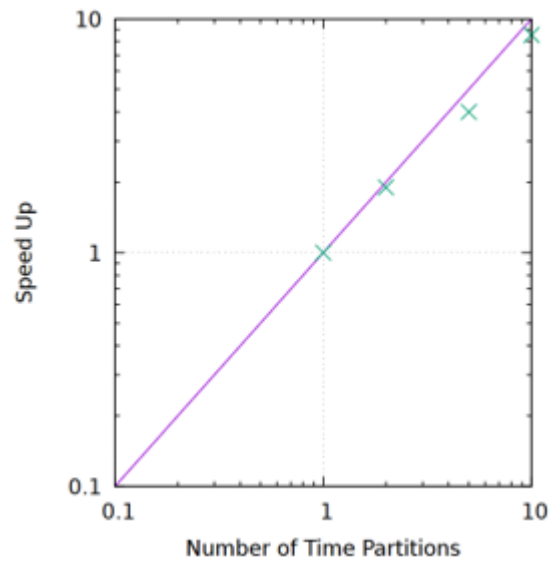


Figure 5.8: Parallel speedup.

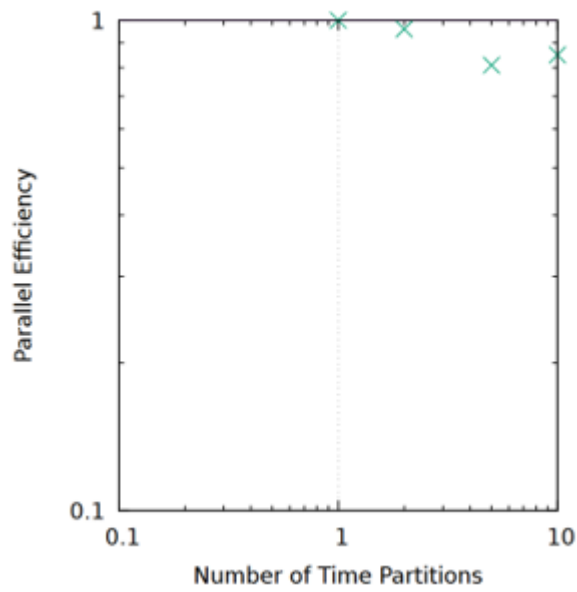


Figure 5.9: Parallel speedup.

Chapter 6: Summary and Conclusions

A new form of periodic multiple shooting was proposed that is matrix-free and may be applied to high-fidelity multiphysics models or other high dimensional systems. The proposed methodology was formulated entirely in the time domain. Therefore, existing time-domain simulation tools may utilize the proposed approach to achieve a high degree of distributed memory parallelism without requiring any reformulation. Furthermore, the proposed methodology may be combined with conventional space domain decomposition techniques and other forms of data parallelism to achieve maximal performance on modern HPC architectures. The proposed algorithm retains the iterative shoot-correct approach of conventional periodic shooting methods. However, the correction stage was formulated using a hierarchical evaluation strategy combined with an Arnoldi subspace approximation to eliminate the need for explicit formulation of Jacobian matrices. The local convergence of the proposed method was formally proven for the case of an asymptotically stable dynamical system. The proposed method was numerically tested for a 2D limit cycle problem, a rigid blade helicopter rotor model with quasi-steady aerodynamics and autopilot trim, and an OVERSET CFD model of a helicopter rotor with prescribed elastic blade motions. The method was observed to be convergent in all

test cases and found to exhibit good scalability.

The proposed periodic multiple shooting method is a practical means of reducing time-to-solution for numerical simulations of asymptotically stable periodic systems on distributed memory parallel computers. Furthermore, the proposed method may be used to enhance the parallel scalability of OVERSET CFD models of helicopter rotors in steady periodic flight.

Bibliography

- [1] Arathi K. Gopinath and Antony Jameson. Time spectral method for periodic unsteady computations over two and three dimensional bodies. In *43rd AIAA Aerospace Sciences Meeting and Exhibit*, January 2005.
- [2] Seongim Choi and Anubhav Datta. Cfd prediction of rotor loads using time-spectral method and exact fluid-structure interface. In *26th AIAA Applied Aerodynamics Conference*, August 2008.
- [3] Morris W. Hirsch and Stephen Smale. *Differential Equations, Dynamical Systems, and Linear Algebra*. Pure and Applied Mathematics. Academic Press, Inc., San Diego, 1974.
- [4] Stephen Wiggins. *Introduction to Applied Nonlinear Dynamical Systems and Chaos*, volume 2 of *Texts in Applied Mathematics*. Springer-Verlag, New York, second edition, 2003.
- [5] V.I. Arnold. *Ordinary Differential Equations*. MIT Press, Cambridge, 1973.
- [6] Walter Gander and Gene H Golub. Cyclic reduction—history and applications. *Scientific computing (Hong Kong, 1997)*, pages 73–85, 1997.
- [7] David A Peters and Dinesh Barwey. A general theory of rotorcraft trim. *Mathematical Problems in Engineering*, 2(1):1–34, 1996.
- [8] DA Peters and M Chouchane. Effect of dynamic stall on helicopter trim and flap-lag response. *Journal of Fluids and Structures*, 1(3):299–318, 1987.
- [9] D Chandra Sekhar and R Ganguli. Fractal boundaries of basin of attraction of newton–raphson method in helicopter trim. *Computers & Mathematics with Applications*, 60(10):2834–2858, 2010.
- [10] Bogdan I Epureanu and Henry S Greenside. Fractal basins of attraction associated with a damped newton’s method. *SIAM review*, pages 102–109, 1998.

- [11] Mako Haruta. Newton's method on the complex exponential function. *Transactions of the American Mathematical Society*, 351(6):2499–2513, 1999.
- [12] Moonja Jeong, Gi Ok Kim, and Seong-A Kim. Dynamics of newton's method for solving some equations. *Computers and Graphics*, 26(2):271 – 279, 2002.
- [13] Saipraneeth Gouravaraju and Ranjan Ganguli. Estimating the hausdorff-besicovitch dimension of boundary of basin of attraction in helicopter trim. *Applied Mathematics and Computation*, 218(21):10435–10442, 2012.
- [14] Peter Kunkel. *Differential-algebraic equations: analysis and numerical solution*. European Mathematical Society, 2006.
- [15] Eduard Reithmeier. *Periodic Solutions of Nonlinear Dynamical Systems*. Number 1483 in Lecture Notes in Mathematics. Springer-Verlag, Berlin, 1991.
- [16] Peretz P. Friedmann. Numerical methods for determining the stability and response of periodic systems with applications to helicopter rotor dynamics and aeroelasticity. *Computers and Mathematics with Applications*, 12(1):131 – 148, 1986.
- [17] David A Peters and Amir P Izadpanah. Helicopter trim by periodic shooting with newton-raphson iteration. In *American Helicopter Society, Annual Forum, 37 th, New Orleans, LA, Proceedings*, 1981.
- [18] S Subramanian, GH Gaonkar, J Nagabhushanam, and RM Nakadi. Parallel computing concepts and methods for floquet analysis of helicopter trim and stability. *Journal of the American Helicopter Society*, 41(4):370–382, 1996.
- [19] James SG McVicar and Roy Bradley. Robust and efficient trimming algorithm for application to advanced mathematical models of rotorcraft. *Journal of aircraft*, 32(2):439–442, 1995.
- [20] David A Peters. Fast floquet theory and trim for multi-bladed rotorcraft. *Journal of the American Helicopter Society*, 39(4):82–89, 1994.
- [21] Shyam Subramanian, S Venkataratnam, and GH Gaonkar. Parallel fast-floquet analysis of trim and stability for large helicopter models. *Mathematical and computer modelling*, 33(10):1155–1176, 2001.
- [22] Ali H. Nayfeh and Dean T. Mook. *Nonlinear Oscillations*. Wiley-VCH, 2004.
- [23] P. Shahrzad and M. Mahzoon. Limit cycle flutter of airfoils in steady and unsteady flows. *Journal of Sound and Vibration*, 256(2):213 – 225, 2002.
- [24] Liping Liu and Earl H. Dowell. Harmonic balance approach for an airfoil with a freeplay control surface. *AIAA Journal*, 43(4), 2005.

- [25] Y. K. Cheung and S. L. Lau. Incremental time-space finite strip method for non-linear structural vibrations. *Earthquake Engineering and Structural Dynamics*, 10:239 – 253, 1982.
- [26] Y. K. Cheung and S. H. Chen. Application of the incremental harmonic balance method to cubic non-linearity systems. *Journal of Sound and Vibration*, 140(2):273 – 286, 1990.
- [27] A. Raghothama and S. Narayanan. Non-linear dynamics of a two-dimensional airfoil by incremental harmonic balance method. *Journal of Sound and Vibration*, 226(3):493–517, 1999.
- [28] Kenneth C. Hall, Jeffery P. Thomas, and W.S. Clark. Computation of unsteady nonlinear flows in cascades using harmonic balance technique. *AIAA Journal*, 40(5), 2002.
- [29] Jeffrey P. Thomas, Earl H. Dowell, and Kenneth C. Hall. Modeling viscous transonic limit-cycle oscillation behavior using a harmonic balance approach. *Journal of Aircraft*, 41(6), 2004.
- [30] L. Liu, J.P. Thomas, E.H. Dowell, P. Attar, and K.C. Hall. A comparison of classical and high dimensional harmonic balance approaches for a duffing oscillator. *Journal of Computational Physics*, 215:298 – 320, 2006.
- [31] Liping Liu, Earl H. Dowell, and Kenneth C. Hall. A novel harmonic balance analysis for the van der pol oscillator. *International Journal of Non-Linear Mechanics*, 42:2–12, 2007.
- [32] Kivanc Ekici, Kenneth C. Hall, and Earl H. Dowell. Computationally fast harmonic balance methods for unsteady aerodynamic predictions of helicopter rotors. *Journal of Computational Physics*, 227:6206 – 6225, 2008.
- [33] T-K Hsu and David A Peters. Coupled rotor/airframe vibration analysis by a combined harmonic-balance, impedance-matching method. *Journal of the American Helicopter Society*, 27(1):25–34, 1982.
- [34] M Borri. Helicopter rotor dynamics by finite element time approximation. *Computers & mathematics with applications*, 12(1):149–160, 1986.
- [35] David A Peters, BS Kim, and H-S Chen. Calculation of trim settings for a helicopter rotor by an optimized automatic controller. *Journal of Guidance, Control, and Dynamics*, 7(1):85–91, 1984.
- [36] David A Peters, Mnaouar ChouChane, and Mark Fulton. Helicopter trim with flap-lag-torsion and stall by an optimized controller. *Journal of Guidance, Control, and Dynamics*, 13(5):824–834, 1990.

- [37] Russell Enns and Jennie Si. Helicopter trimming and tracking control using direct neural dynamic programming. *IEEE Transactions on Neural Networks*, 14(4):929–939, 2003.
- [38] Luca Riviello. *Rotorcraft trim by a neural model-predictive auto-pilot*. PhD thesis, Georgia Institute of Technology, 2005.
- [39] Carlo L Bottasso and Luca Riviello. Trim of rotorcraft multibody models using a neural-augmented model-predictive auto-pilot. *Multibody System Dynamics*, 18(3):299–321, 2007.
- [40] Frederick D Kim, Roberto Celi, and Mark B Tischler. Forward flight trim and frequency response validation of a helicopter simulation model. *Journal of Aircraft*, 30(6):854–863, 1993.
- [41] David A Peters and Si-Hao Li. A combined periodic-shooting, auto-pilot technique for rotorcraft analysis. In *Computational Mechanics’ 95*, pages 654–659. Springer, 1995.
- [42] Troy Shank. *Optimal aeroelastic trim for rotorcraft with constrained, non-unique trim solutions*. PhD thesis, Georgia Institute of Technology, 2008.
- [43] Chee Tung, Francis X. Caradonna, and Wayne R. Johnson. The prediction of transonic flows on an advancing rotor. *Journal of American Helicopter Society*, 31(3):4 – 9(6), 1986.
- [44] Mark Potsdam, Hyeonsoo Yeo, and Wayne Johnson. Rotor airloads prediction using loose aerodynamic/structural coupling. *Journal of Aircraft*, 43(3), May-June 2006.
- [45] Anubhav Datta, Mark Nixon, and Inderjit Chopra. Review of rotor loads prediction with the emergence of rotorcraft cfd. *Journal of the American Helicopter Society*, 52(4):287–317, 2007.
- [46] Jayanarayanan Sitaraman, Mark Potsdam, Buvaneswari Jayaraman, Anubhav Datta, Wissink Andrew, Dimitri Mavriplis, and Hossein Saberi. Rotor loads prediction using helios: a multi-solver framework for rotorcraft cfd/csd analysis. In *49th AIAA Aerospace Sciences Meeting including the New Horizons Forum and Aerospace Exposition*, page 1123, 2011.