Time Discretization of Structural Problems

William J. Turner
Department of Mathematics
Iowa State University

1 Introduction

Until recently, scientists investigating structural problems were primarily concerned with finding efficient methods for discretizing the spatial component of the structure. Once they constructed an ODE matrix system with a spatial semidiscretization, they simply used standard ODE methods to time-step the system so as to approximate the structural dynamics. Unfortunately these standard ODE methods were often inefficient but most researchers were not concerned with the inefficiencies, because modal bases were considered in many initial structural investigations [7, 8, 13] and the small systems involved often allowed such techniques to be successful. In contrast, the larger systems of fluid dynamics required scientists studying those problems to research efficient temporal solution techniques.

More recent investigations of structural problems have illustrated the difficulties associated with modal expansions for complex systems [2, 14] , leading these investigators to develop more sophisticated approximation techniques. The problem is magnified when structural models are coupled with fluid or acoustic equations to model fluid-structure or structural acoustic systems. Upon spatial discretization, such models can yield ODE systems with up to 1000 unknowns. For systems of this size, scientists must find efficient time-marching techniques to permit numerical simulation of system dynamics. The problem is exacerbated in parameter estimation studies which require multiple system evaluations and control implementation which ultimately requires real time system integration.

This project addresses the problem of efficient temporal solution of structural problems through a comprehensive study of several commercial ODE packages, as well as the programming and investigation of a method currently employed in fluid applications. The investigation was performed in the context of approximating the dynamics of a thin cylindrical shell. This structure was chosen since it commonly arises in applications and the model is complex enough to yield relatively large ODE systems.

To evaluate the various ODE routines, three criteria were considered: preservation of the accuracy of the spatial discretization, amount of required system time, and number of function evaluations.

The method must preserve the accuracy of the spatial discretization throughout the time interval of interest. To determine this, the investigation used a Fourier/cubic spline method to discretize the spatial components. As demonstrated by numerical results in [10, 11], this yielded an order $\mathcal{O}(h_x^4)$ spatial convergence rate for spatial gridlengths h_x. The first test of

the ODE methods was to determine whether this accuracy was maintained when integrating the systems through time intervals of $1-2$ seconds.

The required system time provides a measure of how quickly the method performs the integration; however, it can be influenced by the machine usage and hence provides only a qualitative measure of performance.

The number of function evaluations refers to the number of times the right hand side of the ODE system is constructed (this requires the multiplication of system matrices by the current states). This number is independent of machine usage and provides a quantitative measure of method performance.

This investigation considered three commercial packages and a programmed and tested modified trapezoid method. Sections 3 and 4 contain discussions and numerical results for the stiff and nonstiff VODE (Variable-coefficient Ordinary Differential Equation) solvers. The stiff VODE solver uses Backward Differentiation Formula (BDF) methods with modified Newton (Chord) Iterations, while the nonstiff solver uses implicit Adams methods with simple iteration. Sections 5, 6, and 7, respectively, contain discussions and results for an explicit $4^{th}-5^{th}$ order Runge-Kutta software package, the implicit 5^{th} order Runge-Kutta ODE routine *Radau IIA*, and the modified trapezoid rule discussed in Lambert [6].

2 Thin Shell Model and Test Problem

For the temporal solution method, we considered the problem of approximating forced shell dynamics. We consider a shell of length ℓ, thickness h, and radius R having mass density ρ, Young's modulus E, Poisson ratio ν, and Kelvin-Voigt damping coefficient c_D. As shown in Figure 1, the axial direction is along the x-axis. The displacements of the middle surface in the longitudinal, tangential, and transverse directions are denoted by u, v, and w, respectively.

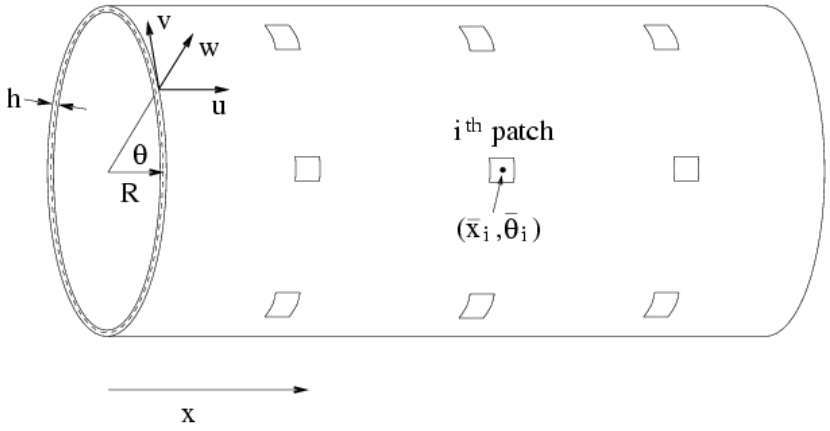


Figure 1: The thin cylindrical shell with patches.

We assume s pairs of piezoceramic patches are bonded to the shell. These can be used as

sensors or actuators in structural and structural acoustic applications [1, 5]. To simplify the exposition, the patches are all assumed to have thickness h_{pe}, Young's Modulus E_{pe}, Poisson ratio ν_{pe}, and Kelvin-Voigt damping coefficient $c_{D_{pe}}$. Furthermore, the glue bonding layer is assumed to provide negligible contributions to the structural dynamics.

We also assume that external inputs to the shell will be in the form of transverse, longitudinal, and tangential surface forces as well as line moments and forces generated by the patches. The surface forces can be used to model acoustic fields [1]. The patch moments and forces arise when the elements are used as actuators.

2.1 Donnell-Mushtari Shell Model

In this model, we assumed the shell has the clamped edge boundary condition

$$u = v = w = \frac{\partial w}{\partial x} = 0 \quad , \quad x = 0, \ell \ . \tag{2.1}$$

This is consistent with many experimental setups in which the shell is supported by heavy end caps, such as the experimental shell apparatus at NASA Langley Research Center [10].

If we consider a shell with a length relatively short compared to its radius, we can use the Donnell-Mushtari equations [2, 7, 10] to approximate its motion. The piezoceramic patches we are considering introduce discontinuities into the system. These discontinuities come from material changes introduced by the patches and the external patch contributions which are only defined over regions covered by the patches. Because of this, we must consider the weak form of the modeling equations.

The weak form of the model equations, as derived from the energy considerations in [2], is given by

$$\int_0^{2\pi}\int_0^{\ell}\left\{R\rho h\frac{\partial^2 u}{\partial t^2}\eta_1 + RN_x\frac{\partial \eta_1}{\partial x} + N_{\theta x}\frac{\partial \eta_1}{\partial \theta} - Rq_x\eta_1 - R\sum_{i=1}^{s}\left(N_x\right)_{pe_i}\frac{\partial \eta_1}{\partial x}\right\}dxd\theta = 0$$

$$\int_0^{2\pi}\int_0^{\ell}\left\{R\rho h\frac{\partial^2 v}{\partial t^2}\eta_2 + N_{\theta}\frac{\partial \eta_2}{\partial \theta} + RN_{x\theta}\frac{\partial \eta_2}{\partial x} - Rq_{\theta}\eta_2 - \sum_{i=1}^{s}\left(N_{\theta}\right)_{pe_i}\frac{\partial \eta_2}{\partial \theta}\right\}dxd\theta = 0 \tag{2.2}$$

$$\int_0^{2\pi}\int_0^{\ell}\left\{R\rho h\frac{\partial^2 w}{\partial t^2}\eta_3 + N_{\theta}\eta_3 - RM_x\frac{\partial^2 \eta_3}{\partial x^2} - \frac{1}{R}M_{\theta}\frac{\partial^2 \eta_3}{\partial \theta^2} - 2M_{x\theta}\frac{\partial^2 \eta_3}{\partial x \partial \theta}\right.$$
$$\left. -Rq_n\eta_3 + \sum_{i=1}^{s}\left[R\left(M_x\right)_{pe_i}\frac{\partial^2 \eta_3}{\partial x^2} + \frac{1}{R}\left(M_{\theta}\right)_{pe_i}\frac{\partial^2 \eta_3}{\partial \theta^2}\right]\right\}dxd\theta = 0$$

for all $\vec{\eta} = [\eta_1, \eta_2, \eta_3] \in V$ where $V = H_0^1(\Omega) \times H_0^1(\Omega) \times H_0^2(\Omega)$ denotes the space of test functions [10]. Here

$$H_0^1 = \{\eta \in H^1(\Omega) : \eta(0) = \eta(\ell) = 0\}$$
$$H_0^2 = \{\eta \in H^1(\Omega) : \eta(0) = \eta_x(0) = \eta(\ell) = \eta_x(\ell) = 0\} \ .$$

Assuming the stress is proportional to a linear combination of strain and strain rate, the internal moments and force resultants in regions of the shell *not* covered by patches are

$$\begin{aligned}
N_x &= \frac{Eh}{(1-\nu^2)}\left[\frac{\partial u}{\partial x}+\nu\left(\frac{1}{R}\frac{\partial v}{\partial \theta}+\frac{w}{R}\right)\right]+\frac{c_D h}{(1-\nu^2)}\frac{\partial}{\partial t}\left[\frac{\partial u}{\partial x}+\nu\left(\frac{1}{R}\frac{\partial v}{\partial \theta}+\frac{w}{R}\right)\right]\\
N_\theta &= \frac{Eh}{(1-\nu^2)}\left[\frac{1}{R}\frac{\partial v}{\partial \theta}+\frac{w}{R}+\nu\frac{\partial u}{\partial x}\right]+\frac{c_D h}{(1-\nu^2)}\frac{\partial}{\partial t}\left[\frac{1}{R}\frac{\partial v}{\partial \theta}+\frac{w}{R}+\nu\frac{\partial u}{\partial x}\right]\\
N_{x\theta} = N_{\theta x} &= \frac{Eh}{2(1+\nu)}\left[\frac{\partial v}{\partial x}+\frac{1}{R}\frac{\partial u}{\partial \theta}\right]+\frac{c_D h}{2(1+\nu)}\frac{\partial}{\partial t}\left[\frac{\partial v}{\partial x}+\frac{1}{R}\frac{\partial u}{\partial \theta}\right]\\
M_x &= \frac{-Eh^3}{12(1-\nu^2)}\left[\frac{\partial^2 w}{\partial x^2}+\frac{\nu}{R^2}\frac{\partial^2 w}{\partial \theta^2}\right]-\frac{c_D h^3}{12(1-\nu^2)}\frac{\partial}{\partial t}\left[\frac{\partial^2 w}{\partial x^2}+\frac{\nu}{R^2}\frac{\partial^2 w}{\partial \theta^2}\right]\\
M_\theta &= \frac{-Eh^3}{12(1-\nu^2)}\left[\frac{1}{R^2}\frac{\partial^2 w}{\partial \theta^2}+\nu\frac{\partial^2 w}{\partial x^2}\right]-\frac{c_D h^3}{12(1-\nu^2)}\frac{\partial}{\partial t}\left[\frac{1}{R^2}\frac{\partial^2 w}{\partial \theta^2}+\nu\frac{\partial^2 w}{\partial x^2}\right]\\
M_{x\theta} = M_{\theta x} &= \frac{-Eh^3}{12R(1+\nu)}\frac{\partial^2 w}{\partial x\partial\theta}-\frac{c_D h^3}{12R(1+\nu)}\frac{\partial}{\partial t}\left[\frac{\partial^2 w}{\partial x\partial\theta}\right]
\end{aligned} \tag{2.3}$$

(see [2]).

Bonding the piezoceramic patches to the shell produces contributions due to both the internal and external moments and forces. The internal contributions arise from geometrical and material changes from the patches, and the external contributions come from the piezoelectric effect in the patches which generates strains in response to applied voltages.

As detailed in [2, 3], the internal force resultant N_x, which incorporates patch contributions, is given by

$$\begin{aligned}
N_x = &\frac{Eh}{(1-\nu^2)}\left[\frac{\partial u}{\partial x}+\nu\left(\frac{1}{R}\frac{\partial v}{\partial \theta}+\frac{w}{R}\right)\right]+\sum_{i=1}^{s}\frac{2E_{pe}h_{pe}}{1-\nu_{pe}^2}\left[\frac{\partial u}{\partial x}+\frac{\nu_p e}{R}\left(\frac{\partial v}{\partial \theta}+w\right)\right]\chi_{pe_i}(x,\theta)\\
&+\frac{c_D h}{(1-\nu^2)}\frac{\partial}{\partial t}\left[\frac{\partial u}{\partial x}+\nu\left(\frac{1}{R}\frac{\partial v}{\partial \theta}+\frac{w}{R}\right)\right]+\sum_{i=1}^{s}\frac{2c_{D_{pe}}h_{pe}}{1-\nu_{pe}^2}\left[\frac{\partial u}{\partial x}+\frac{\nu_p e}{R}\left(\frac{\partial v}{\partial \theta}+w\right)\right]\chi_{pe_i}(x,\theta)\ .
\end{aligned} \tag{2.4}$$

Here $\chi_{pe_i}(x,\theta)$ is the characteristic function for the i^{th} pair of patches.

$$\chi_{pe}(x,\theta)=\begin{cases}1 & , \quad x_1\le x\le x_2 \quad ,\theta_1\le\theta\le\theta_2\\ 0 & , \quad \text{otherwise}\end{cases}\ .$$

Analogous expressions are given in [2, 3] for the remaining resultants and the resultants when the patches have differing material characteristics.

To characterize the external contributions, we start with the assumption that the strains generated by a patch are proportional to the applied voltage across that patch [2]. We will use $V_{i1}(t)$ and $V_{i2}(t)$ to denote the voltages on the outer and inner patches in the i^{th} pair, respectively. In general, these voltages are not the same. Using the proportionality constant d_{31} to relate the generated strain to the input voltage, we get the total external moments and

forces generated by the patches to be

$$\begin{aligned}
(M_x)_{pe_i} &= \left[(M_x)_{pe_{i1}} + (M_x)_{pe_{i2}}\right] \chi_{pe_i}(x,\theta) \\
(M_\theta)_{pe_i} &= \left[(M_\theta)_{pe_{i1}} + (M_\theta)_{pe_{i2}}\right] \chi_{pe_i}(x,\theta) \\
(N_x)_{pe_i} &= \left[(N_x)_{pe_{i1}} + (N_x)_{pe_{i2}}\right] \chi_{pe_i}(x,\theta) S_{1,2}(x) \hat{S}_{1,2}(\theta) \\
(N_\theta)_{pe_i} &= \left[(N_\theta)_{pe_{i1}} + (N_\theta)_{pe_{i2}}\right] \chi_{pe_i}(x,\theta) S_{1,2}(x) \hat{S}_{1,2}(\theta) \ ,
\end{aligned} \tag{2.5}$$

where

$$\begin{aligned}
(M_x)_{pe_{i1}} &= \frac{-E_1}{1-\nu_1}\left[\frac{1}{8}\left(4\left(\frac{h}{2}+h_{pe}\right)^2 - h^2\right) + \frac{1}{R}\frac{1}{24}\left(8\left(\frac{h}{2}+h_{pe}\right)^3 - h^3\right)\right]\frac{d_{31}}{h_{pe}} V_{i1} \\
(M_x)_{pe_{i2}} &= \frac{E_2}{1-\nu_2}\left[\frac{1}{8}\left(4\left(\frac{h}{2}+h_{pe}\right)^2 - h^2\right) - \frac{1}{R}\frac{1}{24}\left(8\left(\frac{h}{2}+h_{pe}\right)^3 - h^3\right)\right]\frac{d_{31}}{h_{pe}} V_{i2} \\
(M_\theta)_{pe_{i1}} &= \frac{-E_1}{1-\nu_1}\left[\frac{1}{8}\left(4\left(\frac{h}{2}+h_{pe}\right)^2 - h^2\right)\right]\frac{d_{31}}{h_{pe}} V_{i1} \\
(M_\theta)_{pe_{i2}} &= \frac{E_2}{1-\nu_2}\left[\frac{1}{8}\left(4\left(\frac{h}{2}+h_{pe}\right)^2 - h^2\right)\right]\frac{d_{31}}{h_{pe}} V_{i2} \\
(N_x)_{pe_{i1}} &= \frac{-E_1}{1-\nu_1}\left[h_{pe} + \frac{1}{R}\frac{1}{8}\left(4\left(\frac{h}{2}+h_{pe}\right)^2 - h^2\right)\right]\frac{d_{31}}{h_{pe}} V_{i1} \\
(N_x)_{pe_{i2}} &= \frac{-E_2}{1-\nu_2}\left[h_{pe} - \frac{1}{R}\frac{1}{8}\left(4\left(\frac{h}{2}+h_{pe}\right)^2 - h^2\right)\right]\frac{d_{31}}{h_{pe}} V_{i2} \\
(N_\theta)_{pe_{i1}} &= \frac{-E_1}{1-\nu_1} h_{pe} \frac{d_{31}}{h_{pe}} V_{i1} \\
(N_\theta)_{pe_{i2}} &= \frac{-E_2}{1-\nu_2} h_{pe} \frac{d_{31}}{h_{pe}} V_{i2}
\end{aligned} \tag{2.6}$$

(see [2]).

2.2 Spatial Approximation

We used a Fourier-Galerkin approximation to discretize the spatial behavior of the shell model. The discretization method uses a uniform partition along the x-axis with grid points $x_n = nh$, where $h = \ell/N$ and $n = 1, 2, \ldots, N$, and a basis of cubic splines modified to meet the required boundary conditions. For $n = -1, 0, 1, \ldots, N+1$, the standard cubic splines are given by

$$b_n(x) = \frac{1}{h^3}\begin{cases}
(x - x_{n-2})^3 & , \quad x \in [x_{n-2}, x_{n-1}] \\
h^3 + 3h^2(x - x_{n-1}) + 3h(x - x_{n-1})^2 - 3(x - x_{n-1})^3 & , \quad x \in [x_{n-1}, x_n] \\
h^3 + 3h^2(x_{n+1} - x) + 3h(x_{n+1} - x)^2 - 3(x_{n+1} - x)^3 & , \quad x \in [x_n, x_{n+1}] \\
(x_{n+2} - x)^3 & , \quad x \in [x_{n+1}, x_{n+2}] \\
0 & , \quad \text{otherwise}
\end{cases} \tag{2.7}$$

(see [9]). To construct basis functions satisfying fixed displacements $u = v = 0$ at $x = 0, \ell$, the modified cubic splines are

$$\hat{b}_n(x) = \begin{cases} b_0(x) - 4b_{-1}(x) & , \quad n = 0 \\ b_1(x) - b_{-1}(x) & , \quad n = 1 \\ b_n(x) & , \quad n = 2, \cdots, N-2 \\ b_{N-1}(x) - b_{N+1} & , \quad n = N-1 \\ b_N(x) - 4b_{N+1}(x) & , \quad n = N \end{cases} , \tag{2.8}$$

for a total of $N+1$ basis functions. With this definition, $\hat{b}_n(0) = \hat{b}_n(\ell) = 0$. For the transverse displacements, in which both the displacements and the slopes are fixed, $w = \frac{\partial w}{\partial x} = 0$ at $x = 0, \ell$, the modified cubic splines are

$$\tilde{b}_n(x) = \begin{cases} b_0(x) - 2b_{-1}(x) - 2b_1(x) & , \quad n = 1 \\ b_n(x) & , \quad n = 2, \cdots, N-2 \\ b_N(x) - 2b_{N-1}(x) - 2b_{N+1}(x) & , \quad n = N-1 \end{cases} , \tag{2.9}$$

for a total of $N-1$ basis functions, with $\tilde{b}_n(0) = \tilde{b}'_n(0) = \tilde{b}_n(\ell) = \tilde{b}'_n(\ell) = 0$. The axial basis functions are given by $B_{u_n}(x) = B_{v_n}(x) = \hat{b}_n(x)$, and $B_{w_n}(x) = \tilde{b}_n(x)$.

We then approximate the longitudinal, tangential, and transverse displacements using Fourier expansions in the θ-direction (to take advantage of periodicity) and cubic splines in the x-direction. This yields approximate solutions of the form

$$\begin{aligned} u^N(t,\theta,x) &= \sum_{m=-M_u}^{M_u} \sum_{n=1}^{N_u} u_{mn}(t) e^{im\theta} B_{u_n}(x) \\ v^N(t,\theta,x) &= \sum_{m=-M_v}^{M_v} \sum_{n=1}^{N_v} v_{mn}(t) e^{im\theta} B_{v_n}(x) \\ w^N(t,\theta,x) &= \sum_{m=-M_w}^{M_w} \sum_{n=1}^{N_w} w_{mn}(t) e^{im\theta} B_{w_n}(x) \ . \end{aligned} \tag{2.10}$$

Here, the total number of basis functions is $\mathcal{N}_u = N_u \cdot (2M_u + 1)$, $\mathcal{N}_v = N_v \cdot (2M_v + 1)$, and $\mathcal{N}_w = N_w \cdot (2M_w + 1)$, where M_u, M_v, M_w are the Fourier limits and N_u, N_v, N_w are the number of splines used in each expansion.

To simplify the notation, we define the following bases

$$\begin{aligned}
\mathcal{B}_{u_k}(\theta,x) &= \left\{ \begin{array}{c} \sin(m\theta) \\ 1 \\ \cos(m\theta) \end{array} \right\} B_{u_n}(x), \\
\mathcal{B}_{v_k}(\theta,x) &= \left\{ \begin{array}{c} \sin(m\theta) \\ 1 \\ \cos(m\theta) \end{array} \right\} B_{v_n}(x), \\
\mathcal{B}_{v_k}(\theta,x) &= \left\{ \begin{array}{c} \sin(m\theta) \\ 1 \\ \cos(m\theta) \end{array} \right\} B_{w_n}(x) \ ,
\end{aligned} \tag{2.11}$$

letting n vary for fixed m. This can be thought of as movement in the θ-direction followed by movement in the x-direction. For example, for the the clamped edge boundary condition, the index k in $\hat{\mathcal{B}}_k$ would change from 1 to 85 for $N_u = 16$ and $M_u = 2$. Using this notation, the approximate displacements can be compactly expressed as

$$\begin{aligned}
u^N(t,\theta,x) &= \sum_{k=1}^{\mathcal{N}_u} u_k(t)\mathcal{B}_{u_k}(\theta,x), \quad \mathcal{N}_u = N_u \cdot (2M_u+1) \\
v^N(t,\theta,x) &= \sum_{k=1}^{\mathcal{N}_v} v_k(t)\mathcal{B}_{v_k}(\theta,x), \quad \mathcal{N}_v = N_v \cdot (2M_v+1) \\
w^N(t,\theta,x) &= \sum_{k=1}^{\mathcal{N}_w} w_k(t)\mathcal{B}_{w_k}(\theta,x), \quad \mathcal{N}_w = N_w \cdot (2M_w+1) \ .
\end{aligned} \tag{2.12}$$

Substituting the force and moment resultants (2.2) into the three equations of the weak form and consolidating gives us the following mass and stiffness submatrices [10]:

(i) $[U_M]_{l,k} = \displaystyle\int_0^{2\pi}\int_0^{\ell} \rho h R \mathcal{B}_{u_k}\mathcal{B}_{u_l} dx d\theta$

(ii) $[V_M]_{l,k} = \displaystyle\int_0^{2\pi}\int_0^{\ell} \rho h R \mathcal{B}_{v_k}\mathcal{B}_{v_l} dx d\theta$

(iii) $[W_M]_{l,k} = \displaystyle\int_0^{2\pi}\int_0^{\ell} \rho h R \mathcal{B}_{w_k}\mathcal{B}_{w_l} dx d\theta$

(iv) $[U_{11}]_{l,k} = \displaystyle\int_0^{2\pi}\int_0^{\ell} \frac{EhR}{1-\nu^2}\frac{\partial \mathcal{B}_{u_k}}{\partial x}\frac{\partial \mathcal{B}_{u_l}}{\partial x} dx d\theta$

(v) $[U_{12}]_{l,k} = \displaystyle\int_0^{2\pi}\int_0^{\ell} \frac{Eh}{2R(1+\nu)}\frac{\partial \mathcal{B}_{u_k}}{\partial \theta}\frac{\partial \mathcal{B}_{u_l}}{\partial \theta} dx d\theta$

(vi) $[U_{21}]_{l,k} = \displaystyle\int_0^{2\pi}\int_0^{\ell} \frac{Eh\nu}{1-\nu^2}\frac{\partial \mathcal{B}_{u_k}}{\partial x}\frac{\partial \mathcal{B}_{v_l}}{\partial \theta} dx d\theta$

(vii) $[U_{22}]_{l,k} = \int_0^{2\pi}\int_0^{\ell} \frac{Eh}{2(1+\nu)} \frac{\partial \mathcal{B}_{u_k}}{\partial \theta}\frac{\partial \mathcal{B}_{v_l}}{\partial x} dx d\theta$

(viii) $[U_{31}]_{l,k} = \int_0^{2\pi}\int_0^{\ell} \frac{Eh\nu}{1-\nu^2} \frac{\partial \mathcal{B}_{u_k}}{\partial x}\mathcal{B}_{w_l} dx d\theta$

(ix) $[V_{11}]_{l,k} = \int_0^{2\pi}\int_0^{\ell} \frac{Eh\nu}{1-\nu^2} \frac{\partial \mathcal{B}_{v_k}}{\partial \theta}\frac{\partial \mathcal{B}_{u_l}}{\partial x} dx d\theta$

(x) $[V_{12}]_{l,k} = \int_0^{2\pi}\int_0^{\ell} \frac{Eh}{2(1+\nu)} \frac{\partial \mathcal{B}_{v_k}}{\partial x}\frac{\partial \mathcal{B}_{u_l}}{\partial \theta} dx d\theta$

(xi) $[V_{21}]_{l,k} = \int_0^{2\pi}\int_0^{\ell} \frac{Eh}{R(1-\nu^2)} \frac{\partial \mathcal{B}_{v_k}}{\partial \theta}\frac{\partial \mathcal{B}_{v_l}}{\partial \theta} dx d\theta$

(xii) $[V_{22}]_{l,k} = \int_0^{2\pi}\int_0^{\ell} \frac{EhR}{2(1+\nu)} \frac{\partial \mathcal{B}_{v_k}}{\partial x}\frac{\partial \mathcal{B}_{v_l}}{\partial x} dx d\theta$

(xiii) $[V_{31}]_{l,k} = \int_0^{2\pi}\int_0^{\ell} \frac{Eh}{R(1-\nu^2)} \frac{\partial \mathcal{B}_{v_k}}{\partial \theta}\mathcal{B}_{w_l} dx d\theta$

(xiv) $[W_{11}]_{l,k} = \int_0^{2\pi}\int_0^{\ell} \frac{Eh\nu}{1-\nu^2} \mathcal{B}_{w_k}\frac{\partial \mathcal{B}_{u_l}}{\partial x} dx d\theta$

(xv) $[W_{21}]_{l,k} = \int_0^{2\pi}\int_0^{\ell} \frac{Eh}{R(1-\nu^2)} \mathcal{B}_{w_k}\frac{\partial \mathcal{B}_{v_l}}{\partial \theta} dx d\theta$

(xvi) $[W_{31}]_{l,k} = \int_0^{2\pi}\int_0^{\ell} \frac{Eh}{R(1-\nu^2)} \mathcal{B}_{w_k}\mathcal{B}_{w_l} dx d\theta$

(xvii) $[W_{32}]_{l,k} = \int_0^{2\pi}\int_0^{\ell} \frac{Eh^3R}{12(1-\nu^2)} \frac{\partial^2 \mathcal{B}_{w_k}}{\partial x^2}\frac{\partial^2 \mathcal{B}_{w_l}}{\partial x^2} dx d\theta$

(xviii) $[W_{33}]_{l,k} = \int_0^{2\pi}\int_0^{\ell} \frac{Eh^3\nu}{12R(1-\nu^2)} \frac{\partial^2 \mathcal{B}_{w_k}}{\partial \theta^2}\frac{\partial^2 \mathcal{B}_{w_l}}{\partial x^2} dx d\theta$

(xix) $[W_{34}]_{l,k} = \int_0^{2\pi}\int_0^{\ell} \frac{Eh^3}{12R^3(1-\nu^2)} \frac{\partial^2 \mathcal{B}_{w_k}}{\partial \theta^2}\frac{\partial^2 \mathcal{B}_{w_l}}{\partial \theta^2} dx d\theta$

(xx) $[W_{35}]_{l,k} = \int_0^{2\pi}\int_0^{\ell} \frac{Eh^3\nu}{12R(1-\nu^2)} \frac{\partial^2 \mathcal{B}_{w_k}}{\partial x^2}\frac{\partial^2 \mathcal{B}_{w_l}}{\partial \theta^2} dx d\theta$

(xxi) $[W_{36}]_{l,k} = \int_0^{2\pi}\int_0^{\ell} \frac{Eh^3}{6R(1+\nu)} \frac{\partial^2 \mathcal{B}_{w_k}}{\partial x \partial \theta}\frac{\partial^2 \mathcal{B}_{w_l}}{\partial x \partial \theta} dx d\theta$

(xxii) $[F_u]_l = \int_0^{2\pi}\int_0^{\ell} \left\{ Rq_x\mathcal{B}_{u_l} + R\hat{N}_x\frac{\partial \mathcal{B}_{u_l}}{\partial x}\right\} dx d\theta$

(xxiii) $[F_v]_l = \int_0^{2\pi}\int_0^{\ell} \left\{ Rq_\theta\mathcal{B}_{v_l} + \hat{N}_\theta\frac{\partial \mathcal{B}_{v_l}}{\partial \theta}\right\} dx d\theta$

(xxiv) $[F_w]_l = \int_0^{2\pi}\int_0^{\ell} \left\{ Rq_n\mathcal{B}_{w_l} - R\hat{M}_x\frac{\partial^2 \mathcal{B}_{w_l}}{\partial x^2} - \frac{1}{R}\hat{M}_\theta\frac{\partial^2 \mathcal{B}_{w_l}}{\partial \theta^2}\right\} dx d\theta$

In an analogous manner, the matrices $\tilde{U}_{11}$, $\tilde{U}_{12}$, $\tilde{U}_{21}$, $\tilde{U}_{22}$, $\tilde{U}_{31}$, $\tilde{V}_{11}$, $\tilde{V}_{12}$, $\tilde{V}_{21}$, $\tilde{V}_{22}$, $\tilde{V}_{31}$, $\tilde{W}_{11}$, $\tilde{W}_{21}$, $\tilde{W}_{31}$, $\tilde{W}_{32}$, $\tilde{W}_{33}$, $\tilde{W}_{34}$, $\tilde{W}_{35}$, and $\tilde{W}_{36}$ containing the internal damping contributions can be found by replacing E by c_D in the definitions **(iv)** through **(xx-iv)** above.

The orthogonality properties of the generalized Fourier/cubic spline bases give rise to matrices which are block diagonal with the block matrices being composed of symmetric matrices.

With these submatrices, we now define the vectors $\mathcal{U}(t), \mathcal{V}(t)$ and $\mathcal{W}(t)$

$$\mathcal{U}^{\mathcal{N}_u}(t) = \begin{bmatrix} u_1(t) \\ \vdots \\ u_{\mathcal{N}_u}(t) \end{bmatrix}, \quad \mathcal{V}^{\mathcal{N}_v}(t) = \begin{bmatrix} v_1(t) \\ \vdots \\ v_{\mathcal{N}_v}(t) \end{bmatrix}, \quad \mathcal{W}^{\mathcal{N}_w}(t) = \begin{bmatrix} w_1(t) \\ \vdots \\ w_{\mathcal{N}_w}(t) \end{bmatrix}. \tag{2.13}$$

which contain the generalized Fourier coefficients of the expansions for the displacements (2.12). These are then stored in the vector $\vartheta^{\mathcal{N}}(t) = [\mathcal{U}(t)^{\mathcal{N}_u}, \mathcal{V}^{\mathcal{N}_v}(t), \mathcal{W}^{\mathcal{N}_w}(t)]^T$, where $\mathcal{N} = \mathcal{N}_u + \mathcal{N}_v + \mathcal{N}_w$.

We also form the mass, stiffness, and damping matrices, as well as the forcing vector, for the full system by

$$M^{\mathcal{N}} = \left[\begin{array}{c|c|c} U_M & & \\ \hline & V_M & \\ \hline & & W_M \end{array}\right],$$

$$K_E^{\mathcal{N}} = \left[\begin{array}{c|c|c} U_{11}+U_{12} & V_{11}+V_{12} & W_{11} \\ \hline U_{21}+U_{22} & V_{21}+V_{22} & W_{21} \\ \hline U_{31} & V_{31} & \sum_{k=1}^{6} W_{3k} \end{array}\right], \tag{2.14}$$

$$K_{c_D}^{\mathcal{N}} = \left[\begin{array}{c|c|c} \tilde{U}_{11}+\tilde{U}_{12} & \tilde{V}_{11}+\tilde{V}_{12} & \tilde{W}_{11} \\ \hline \tilde{U}_{21}+\tilde{U}_{22} & \tilde{V}_{21}+\tilde{V}_{22} & \tilde{W}_{21} \\ \hline \tilde{U}_{31} & \tilde{V}_{31} & \sum_{k=1}^{6} \tilde{W}_{3k} \end{array}\right]$$

and

$$\hat{F}^{\mathcal{N}}(t) = \begin{bmatrix} F_u \\ F_v \\ F_w \end{bmatrix}. \tag{2.15}$$

In first order form, this yields the system

$$\begin{bmatrix} K_E^{\mathcal{N}} & 0 \\ 0 & M^{\mathcal{N}} \end{bmatrix} \begin{bmatrix} \dot{\vartheta}^{\mathcal{N}}(t) \\ \ddot{\vartheta}^{\mathcal{N}}(t) \end{bmatrix} = \begin{bmatrix} 0 & K_E^{\mathcal{N}} \\ -K_E^{\mathcal{N}} & -K_{c_D}^{\mathcal{N}} \end{bmatrix} \begin{bmatrix} \vartheta^{\mathcal{N}}(t) \\ \dot{\vartheta}^{\mathcal{N}}(t) \end{bmatrix} + \begin{bmatrix} 0 \\ \hat{F}^{\mathcal{N}}(t) \end{bmatrix}$$
$$\begin{bmatrix} K_E^{\mathcal{N}} & 0 \\ 0 & M^{\mathcal{N}} \end{bmatrix} \begin{bmatrix} \vartheta^{\mathcal{N}}(0) \\ \dot{\vartheta}^{\mathcal{N}}(0) \end{bmatrix} = \begin{bmatrix} g_1^{\mathcal{N}} \\ g_2^{\mathcal{N}} \end{bmatrix}. \tag{2.16}$$

By multiplying by the inverted mass matrix, we obtain a Cauchy equation of the form

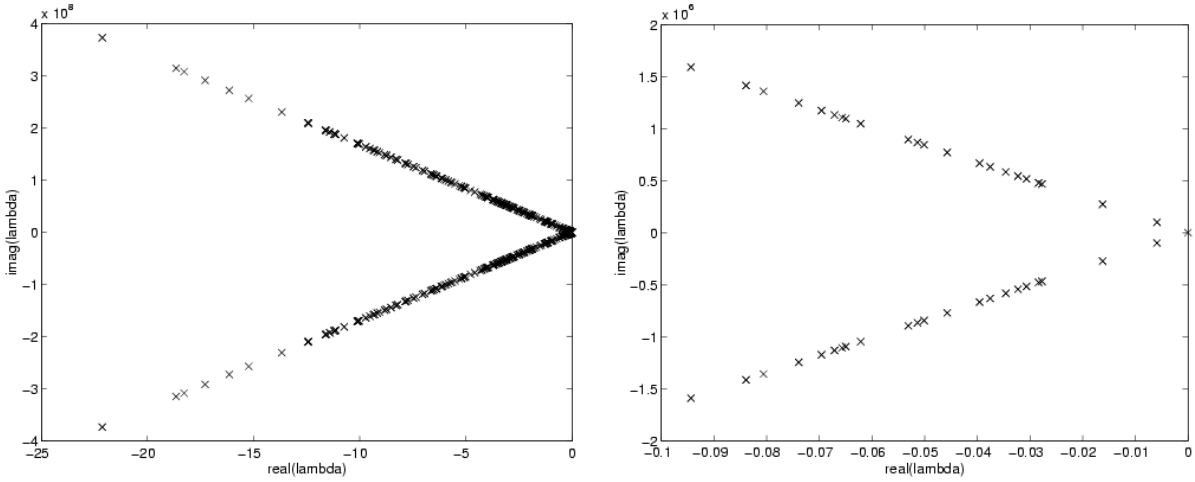
$$\begin{aligned} \dot{y}^N(t) &= A^N y^N(t) + g^N(t) \\ y^N(0) &= y_0^N \ , \end{aligned} \tag{2.17}$$

where $y^N \in R^{2\mathcal{N}} = \left[\vartheta^{\mathcal{N}}(t), \dot{\vartheta}^{\mathcal{N}}(t)\right]^T$. This form is suitable for simulations, parameter estimation, and control applications.

This system is stiff because all the real parts of the eigenvalues of the matrix A^N are negative and the ratios of the largest and smallest real parts of the eigenvalues are large, as illustrated in Figure 2 and Table 1 [10]. Stiffness implies that some components of the solution decay much more rapidly than others. This forces numerical methods with small regions of stability to use a steplength which is excessively small compared to the smoothness of the exact solution. Thus, stability requirements, rather than accuracy requirements, constrain the steplength.

Because of the stiffness of the system, we expect the solvers specialized for stiff problems, that is, ones having large regions of absolute stability, to perform better than the others. In particular, we expect the stiff VODE solver and the implicit Runge-Kutta solver to work much more efficiently than the nonstiff VODE solver and the explicit Runge-Kutta solver, respectively, which will be forced to use very small step sizes to keep the error from getting extremely large.

For the rest of this discussion, we will consider only equal number of grid points in each of the spatial directions, $N = N_u = N_v = N_w$.



(a) (b)

Figure 2: Eigenvalues for the matrix A^N. All the eigenvalues are plotted in (a), while only the first 44 nonzero eigenvalues are plotted in (b).

2.3 Test Problem

In the following examples, we consider a shell with the following parameters:

thickness $h = 0.01$ in,
length $\ell = 12$ in,
radius $R = 3$ in,
mass density $\rho = 0.283$ lb/in^3,
Young's Modulus $E = 3 \times 10^7$ psi,
Poisson ratio $\nu = 0.3$,
Kelvin-Voigt damping coefficient $c_D = 3.56$ lb s in^2.

To satisfy the boundary conditions (2.1), the true solution to the time dependent example is

$$\begin{aligned} u(x,\theta) &= t^2 \sin(2\pi x/\ell)\sin(\theta)\ , \\ v(x,\theta) &= t^2 \sin(4\pi x/\ell)\sin(2\theta)\ , \\ w(x,\theta) &= t^2 \left[\cos(2\pi x/\ell) - 1\right]\cos(3\theta)\ . \end{aligned}$$

To illustrate their form, the solutions are plotted for time $T = 1.2$ seconds in Figures 3, 4, and 5.

We obtained the forcing functions by substituting the true solutions into the Donnell-Mushtari equations and solving for the functions q_x, q_θ and q_n. This gives us the following:

$$\begin{aligned} q_x &= -\frac{Eh}{1-\nu^2}\left[-\left(\frac{2\pi}{\ell}\right)^2 \sin(2\pi x/\ell)\sin(\theta)\right. \\ &\qquad \left. + \frac{\nu 8\pi}{R\ell}\cos(4\pi x/\ell)\cos(2\theta) - \frac{\nu 2\pi}{R\ell}\sin(2\pi x/\ell)\cos 3\theta)\right] \\ &- \frac{Eh}{2(1+\nu)}\left[\frac{8\pi}{\ell}\cos(4\pi x/\ell)\cos(2\theta) - \frac{1}{R}\sin(2\pi x/\ell)\sin(\theta)\right]\ , \\ q_\theta &= -\frac{Eh}{R(1-\nu^2)}\left[-\frac{4}{R}\sin(4\pi x/\ell)\sin(2\theta)\right. \\ &- \left.\frac{3}{R\ell}[\cos(2\pi x/\ell) - 1]\sin(3\theta) + \frac{\nu 2\pi}{\ell}\cos(2\pi x/\ell)\cos\theta)\right] \\ &- \frac{Eh}{2(1+\nu)}\left[-\left(\frac{4\pi}{\ell}\right)^2 \sin(4\pi x/\ell)\sin(2\theta) + \frac{2\pi}{R\ell}\cos(2\pi x/\ell)\cos(\theta)\right] \end{aligned}$$

	$\frac{\max Re(-\lambda)}{\min Re(-\lambda)}$
$N = 4$	$9.1111 + 2$
$N = 8$	$9.5832 + 2$
$N = 16$	$3.7580 + 3$

Table 1: Stiffness Ratios

and

$$
\begin{aligned}
q_n &= \frac{Eh^3}{12(1-\nu^2)}\left[\left(\frac{2\pi}{\ell}\right)^4\cos(2\pi x/\ell)\cos(3\theta)+\frac{9\nu}{R^2}\cdot\left(\frac{2\pi}{\ell}\right)^2\cos(2\pi x/\ell)\cos(3\theta)\right] \\
&+ \frac{Eh^3}{12R^2(1-\nu^2)}\left[\frac{81}{R^2}[\cos(2\pi x/\ell)-1]\cos(3\theta)+9\nu\cdot\left(\frac{2\pi}{\ell}\right)^2\cos(2\pi x/\ell)\cos(3\theta)\right] \\
&+ \frac{Eh^3}{6R^2(1+\nu)}\left[9\left(\frac{2\pi}{\ell}\right)^2\cos(2\pi x/\ell)\cos(3\theta)\right] \\
&+ \frac{Eh}{R(1-\nu^2)}\left[\frac{2}{R}\sin(4\pi x/\ell)\cos(2\theta)+\frac{1}{R}[\cos(2\pi x/\ell)-1]\cos 3\theta\right) \\
&\qquad + \frac{2\pi\nu}{\ell}\cos(2\pi x/\ell)\sin(\theta)\Big] \ .
\end{aligned}
$$

Since the the forces are time independent, we obtain the coefficients

$$\vartheta^{\mathcal{N}} = [u_1,\ldots,u_{\mathcal{N}_u},v_1,\ldots,v_{\mathcal{N}_v},w_1,\ldots,w_{\mathcal{N}_w}]^T$$

for the approximate displacements $u^N(\theta,x)$, $v^N(\theta,x)$, and $w^N(\theta,x)$ of (2.10) by solving the matrix system

$$K_E^{\mathcal{N}}\vartheta^{\mathcal{N}} = \hat{F}^{\mathcal{N}} \ ,$$

where $K_E^{\mathcal{N}}$ and $\hat{F}^{\mathcal{N}}$ are defined in (2.14) and (2.15), respectively.

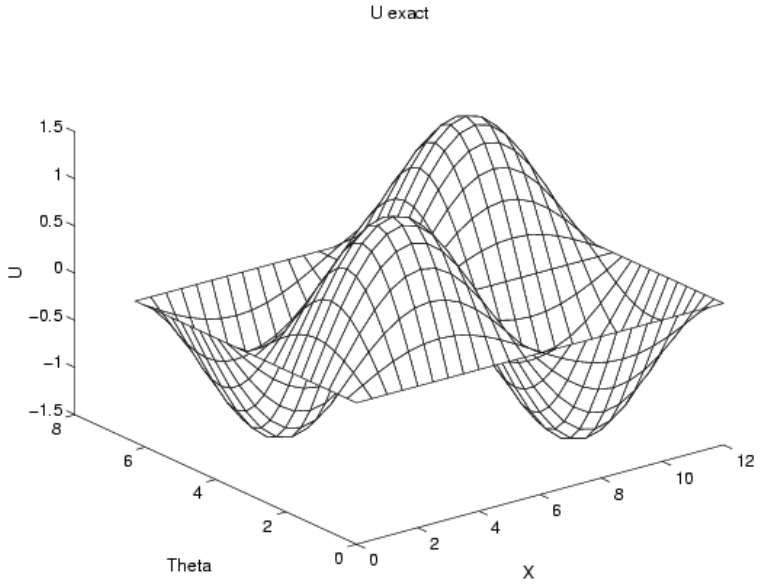


Figure 3: True longitudinal displacement u at $T = 1.2$ seconds.

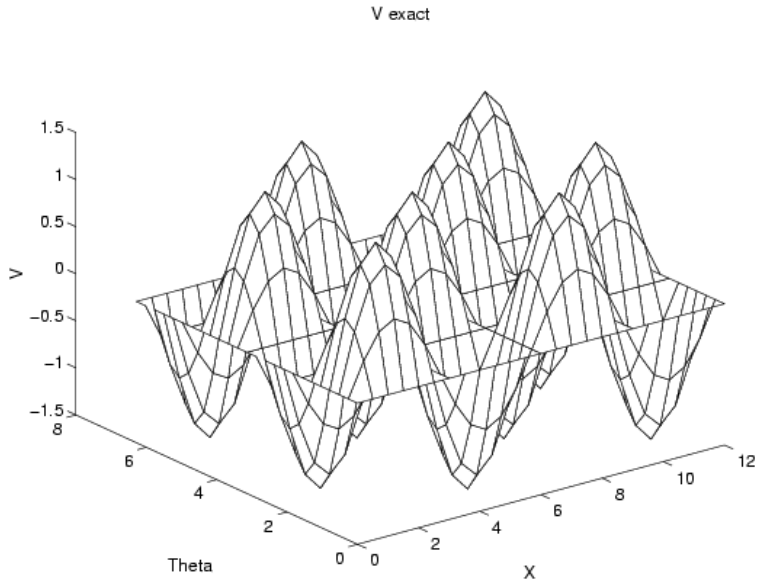


Figure 4: True tangential displacement v at $T = 1.2$ seconds.

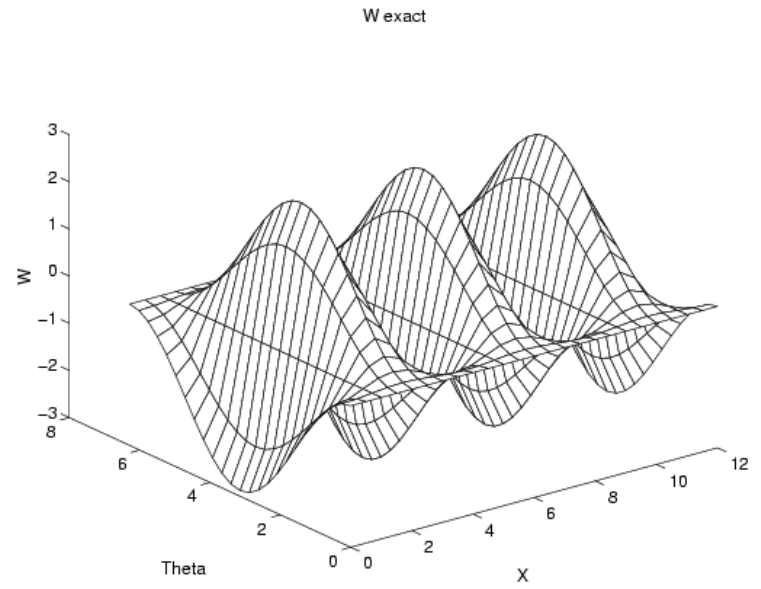


Figure 5: True transverse displacement w at $T = 1.2$ seconds.

3 The Stiff VODE Solver

The stiff VODE (Variable-coefficient Ordinary Differential Equation) [4] solver with fixed leading coefficient implementation was tested first. VODE is an initial value problem solver for systems of ordinary differential equations of the form $\frac{dy}{dt} = f(y,t)$. For stiff problems, VODE uses Backward Differentiation Formula (BDF) methods using a modified Newton (Chord) Iteration. This requires the local Jacobian, which can be provided by the user or generated by the code using difference quotients. The user also specifies whether the code is to use a banded or full Jacobian matrix. VODE uses only variable-step variable-order methods, and it overcomes the difficulties in changing step sizes in multistep methods by using a Nordsieck implementation.

BDF methods are good for stiff problems because of their large regions of absolute stability. As Figure 6 shows, BDF1 and BDF2 are A-stable. That is, their stability regions include all of the left half of the complex plane. BDF methods of higher order are all A(α)-stable. That is, their stability regions all include the sector of the complex plane $\{z = re^{i\theta} : r > 0, \pi - \alpha < \theta < \pi + \alpha\}$. However, for BDF methods of order higher than 4, this α becomes very small, and thus these methods are not generally used.

In implementing the stiff VODE solver, we chose to supply the full Jacobian for the code. Because this is a linear problem, the Jacobian is just the matrix A^N. This was the method used in [10] and initial tests were designed to corroborate these results.

Table 2 shows the absolute errors of the longitudinal, tangential, and transverse displacements. These differ slightly from those reported in [10] due to the differing value of the Kelvin-Voigt damping coefficient c_D and the final time $T = 1.2$ seconds as opposed to the $T = 1.5$ seconds used in [10]. (Note that results identical to those in [10] were obtained when c_D and T were matched.)

The numbers in Table 2 are in exponential form. For instance, $2.33951 - 1$ should be read as 2.3951×10^{-1}.

Table 3 shows the ratio of these errors when the number of basis functions are doubled from $N = 4$ to $N = 8$ and again to $N = 16$. Note the method does exhibit the expected $\mathcal{O}(h_x^4)$ convergence. If the spatial step size h_x is doubled, the accuracy of the approximation is expected to increase by a factor of $2^4 = 16$. So, if we compare the absolute errors for the $N = 4$ and $N = 8$ cases, and also for the $N = 8$ and $N = 16$ cases, we expect to see ratios of at least 16. The smallest ratio here is 24.084 and occurs in the tangential displacement v, when the number of basis functions doubles from $N = 8$ to $N = 16$. These ratios are exactly the same as those in [10].

Table 5 indicates the system time, in seconds, required to run the program. The table includes three time categories: real, user, and system. The first two, real time and user time, both depend very strongly on what else the computer is running. This is difficult to control on the public machines serving multiple users, so they should be disregarded for the most part. The system time, on the other hand, gives a much better indication of the efficiency of the solver, although it, too, depends somewhat on what else is running. For this reason, it should only be used as a qualitative measure of the method's efficiency.

Table 4 shows the number of times the stiff VODE solver evaluated the function. Here, evaluating the function means computing the right hand side of the matrix equation (2.17) describing the system. This involves multiplying the state matrix A^N by the current state

vector $y_N(t)$ and adding the forcing vector $g^N(t)$. Because this is very costly in computer time for large systems, keeping the number of function evaluations to a minimum while still maintaining the same accuracy will give us a more efficient method.

Notice the significantly fewer function evaluations and shorter run time for the $N = 8$ case compared to either $N = 4$ or $N = 16$. We do not have an explanation of what is causing this, but we hypothesize that the $N = 8$ case is starting with a temporal step size which does not require much adjusting as the solver progresses through time.

Figures 7, 8, and 9 illustrate the errors of the longitudinal, tangential, and transverse displacements, respectively. Note, however, only the $N = 4$ case is shown because the errors for $N = 8$ and $N = 16$ are sufficiently small that they cannot be distinguished from the $x\theta$-plane.

	$N = 4$	$N = 8$	$N = 16$
u	$2.3951 - 1$	$2.8839 - 3$	$5.7074 - 5$
v	$8.1893 - 1$	$2.1608 - 2$	$8.9718 - 4$
w	$4.5161 - 1$	$5.8681 - 3$	$7.9759 - 5$

Table 2: Absolute errors of the longitudinal u, tangential v, and transverse displacements w for the stiff VODE solver.

	$\frac{N = 4}{N = 8}$	$\frac{N = 8}{N = 16}$
u	83.052	50.529
v	37.899	24.084
w	76.961	73.573

Table 3: Ratios of the absolute errors for the stiff VODE solver.

$N = 4$	$N = 8$	$N = 16$
$41,864$	83	$94,882$

Table 4: Function evaluations for the stiff VODE solver.

	$N = 4$	$N = 8$	$N = 16$
real	626.2	23.6	$35,047.7$
user	602.3	19.3	$34,781.5$
system	4.8	2.8	65.1

Table 5: Run time for the stiff VODE solver in seconds.

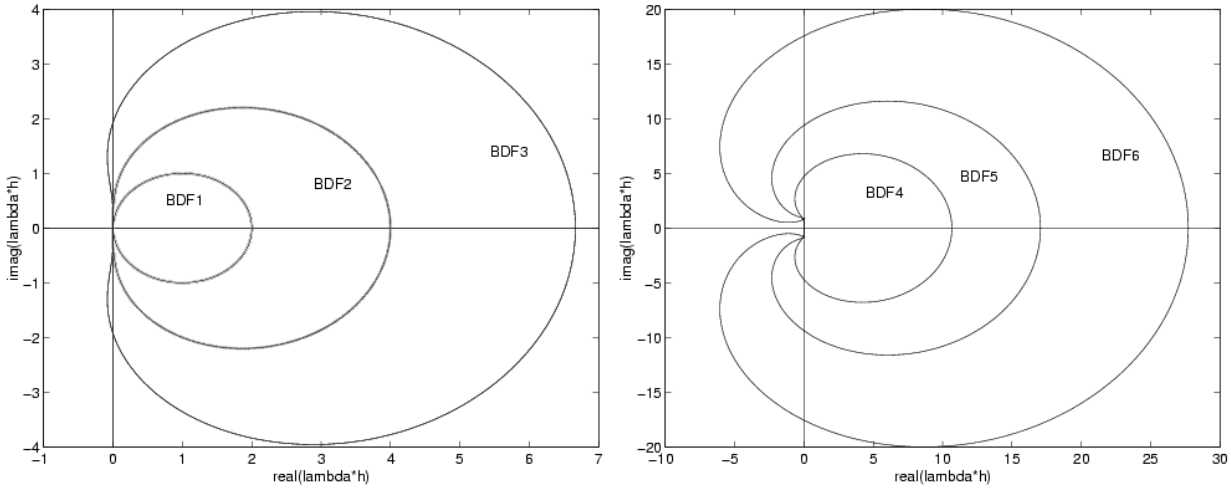


Figure 6: Stability regions for 1^{st} (BDF1) through 6^{th} (BDF6) order BDF methods. Note that the stability region lies *outside* the circle-like plots.

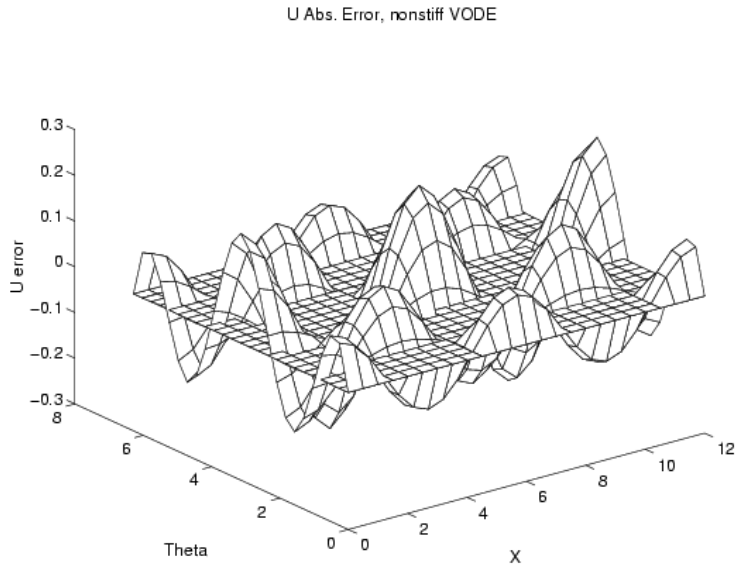


Figure 7: Errors of longitudinal displacement u at $T = 1.2$ seconds with $N = 4$ for the stiff VODE solver.

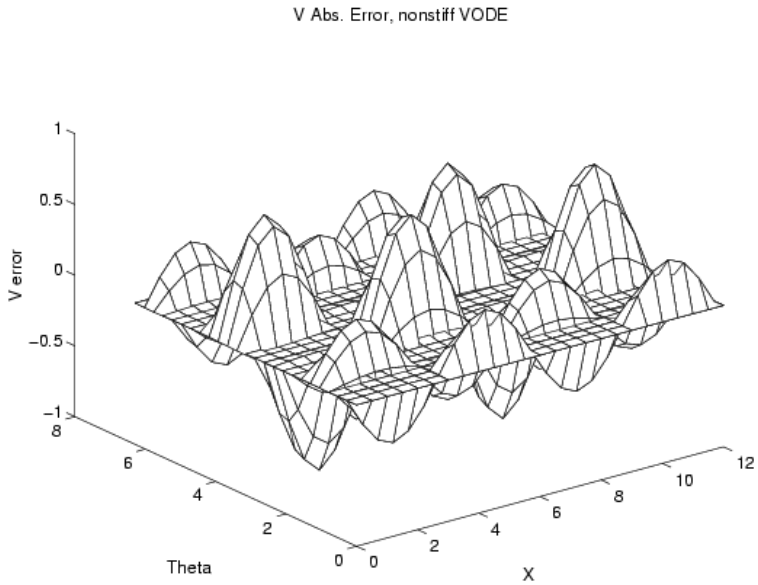


Figure 8: Errors of tangential displacement v at $T = 1.2$ seconds with $N = 4$ for the stiff VODE solver.

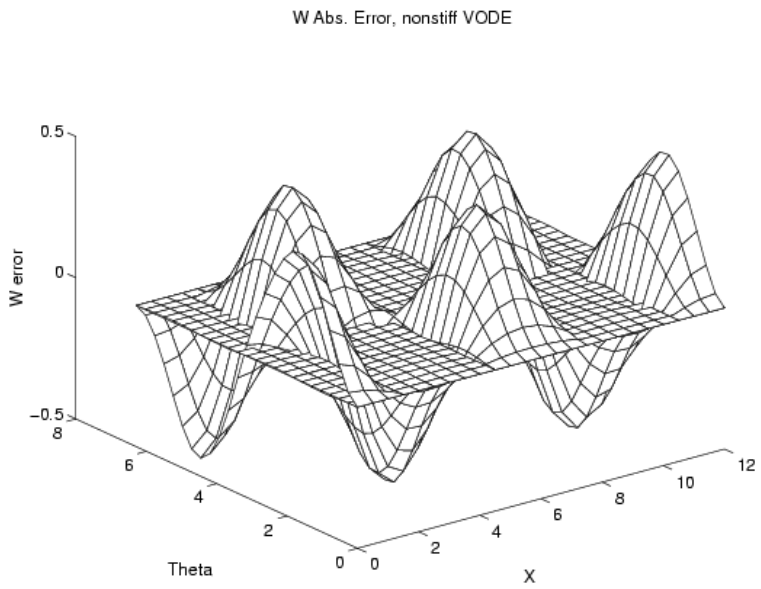


Figure 9: Errors of transverse displacement w at $T = 1.2$ seconds with $N = 4$ for the stiff VODE solver.

4 The Nonstiff VODE Solver

Next, the program was changed to incorporate a nonstiff VODE solver. For nonstiff problems, VODE uses implicit Adams methods with simple iteration. Adams methods are multistep methods, using previously computed solution values. This means they must save the data for later use. It also makes changing step sizes difficult, although VODE overcomes these difficulties by using a Nordsieck implementation.

Adams methods have stability regions as shown in Figure 10. Due to the stiff nature of the problem, it was expected that this method would take significantly longer and evaluate the function significantly more times than it did with the stiff VODE solver, while still giving the same results.

Here, Tables 6 and 7 show the function evaluations and run times, respectively, for the nonstiff VODE solver. The number of function evaluations for the nonstiff VODE solver are significantly larger than those obtained with the stiff option for all three cases, as are the run times for all but the $N = 4$ case. (Compare with Tables 4 and 5.)

Table 8 shows the absolute errors for the nonstiff VODE solver, and Table 9 contains the ratio of the errors. As expected, the errors are on the order of those found using the stiff VODE solver. The method, again, maintains the $\mathcal{O}(h_x^4)$ convergence, and the smallest ratio is 24.069, which occurs in the same location as it did in the stiff solver.

The errors of the longitudinal, tangential, and transverse displacements are qualitatively the same as with stiff VODE solver, and thus are not plotted again here.

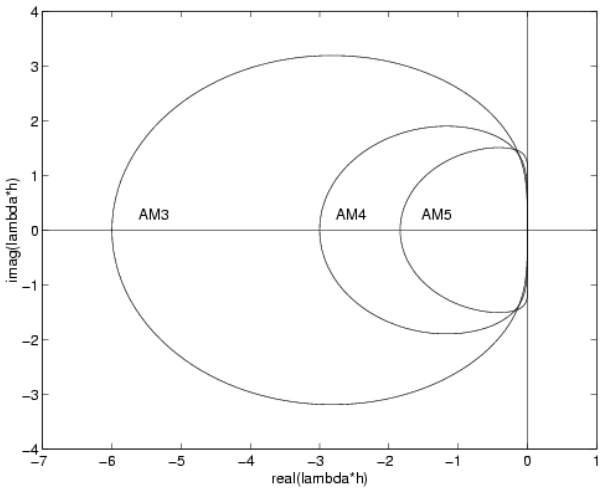


Figure 10: Stability regions for 3^{rd} (AM3) through $5th$ (AM5) order Adams methods. Note that the stability region lies *inside* the circle-like plots.

$N = 4$	$N = 8$	$N = 16$
$86,946$	$149,678$	$267,483$

Table 6: Function evaluations for the nonstiff VODE solver.

	$N = 4$	$N = 8$	$N = 16$
real	395.4	2824.0	$79,314.9$
user	389.0	2796.2	$19,608.4$
system	2.8	7.3	90.9

Table 7: Run time for the nonstiff VODE solver in seconds.

	$N = 4$	$N = 8$	$N = 16$
u	$2.3951 - 1$	$2.8839 - 3$	$5.6671 - 5$
v	$8.1893 - 1$	$2.1608 - 2$	$8.9777 - 4$
w	$4.5161 - 1$	$5.8681 - 3$	$7.9951 - 5$

Table 8: Absolute errors of the longitudinal u, tangential v, and transverse displacements w for the nonstiff VODE solver.

	$\frac{N = 4}{N = 8}$	$\frac{N = 8}{N = 16}$
u	83.052	50.888
v	37.899	24.069
w	76.961	73.396

Table 9: Ratios of the absolute errors for the nonstiff VODE solver.

5 The Explicit Runge-Kutta Solver

The third solver that was considered was the explicit $4^{th} - 5^{th}$ order Runge-Kutta solver from *RKSUITE*. Runge-Kutta methods are one-step methods, as opposed to the multistep BDF and Adams methods used by VODE. They arise from underlying quadrature formulas that use only data from the interval $[x_j, x_{j+1}]$.

As Figure 11 shows, the stability region for a 4^{th} order Runge-Kutta method is limited to roughly $|\lambda h| < 3$ on the left half of the complex plane [12].

Tables 10 and 11 show the absolute errors and the ratios of the errors, respectively, for this explicit Runge-Kutta solver. As in the previous cases, the $\mathcal{O}(h_x^4)$ convergence is observed. In this case the smallest ratio is once again 24.069 and it occurs in the tangential displacement for the ratio of the $N = 8$ and $N = 16$ cases.

The longitudinal, tangential, and transverse displacements are graphically indistinguishable from those from the stiff and nonstiff VODE solvers; therefore, they are not plotted here.

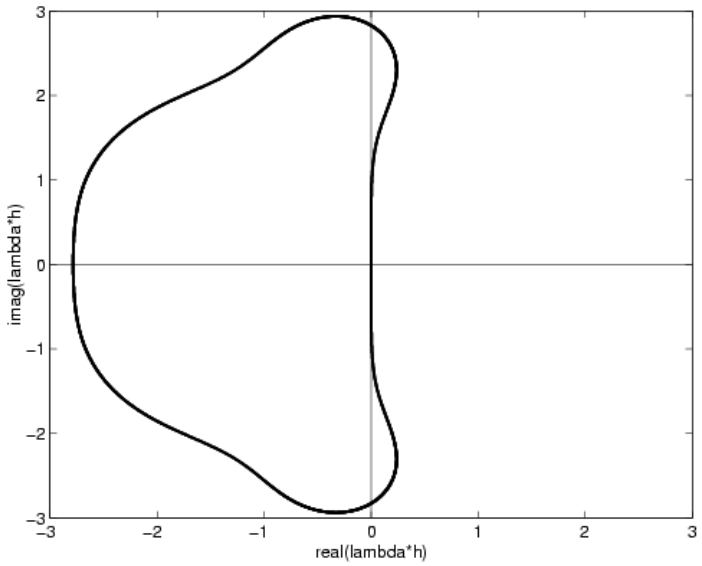


Figure 11: Stability region for a fourth order Runge-Kutta method. Note that the stability region lies *inside* the semicircle-like plot.

Because of the nature of the Runge-Kutta code, and the stiffness of the problem, it is expected to evaluate the function many more times than either of the VODE codes did, and thus take more system time than either. The reason we expect it to make more function evaluations than the Adams method is because it is not using data from previous points, as in the Adams method, but instead evaluating the function at several points inside the interval.

As Tables 12 and 13 show, the explicit Runge-Kutta method does take more system time and evaluate the function significantly more times than either VODE routine. Notice the solver uses 127.0 seconds of system time and evaluates the function 711,461 times for the $N = 16$ case, compared to 65.1 and 90.9 seconds of system time for the stiff and nonstiff VODE codes,

respectively, and 94,882 and 267,483 function evaluations. The numbers are closer together for the $N = 4$ and $N = 8$ cases, with the explicit Runge-Kutta solver actually taking less system time than the stiff VODE code. However, the number of function evaluations is still a factor of six times the number for the stiff VODE.

	$N = 4$	$N = 8$	$N = 16$
u	$2.3951 - 1$	$2.8839 - 3$	$5.6671 - 5$
v	$8.1893 - 1$	$2.1608 - 2$	$8.9777 - 4$
w	$4.5161 - 1$	$5.8681 - 3$	$7.9953 - 5$

Table 10: Absolute errors of the longitudinal u, tangential v, and transverse displacements w for the explicit Runge-Kutta solver.

	$\frac{N = 4}{N = 8}$	$\frac{N = 8}{N = 16}$
u	83.052	50.888
v	37.899	24.069
w	76.961	73.395

Table 11: Ratios of the absolute errors for the explicit Runge-Kutta solver.

	$N = 4$	$N = 8$	$N = 16$
real	1018.5	6166.6	$196,291.5$
user	1007.2	6137.6	$49,162.6$
system	4.0	9.7	127.0

Table 12: Run time for the explicit Runge-Kutta solver in seconds.

$N = 4$	$N = 8$	$N = 16$
$260,117$	$361,916$	$711,461$

Table 13: Function evaluations for the explicit Runge-Kutta solver.

6 The Implicit Runge-Kutta Solver

The final packaged solver we considered was the *Radau IIA* implicit 5^{th} order Runge-Kutta code. This code was downloaded from the *Universite de Geneve* through an anonymous FTP connection. Unlike the explicit Runge-Kutta code, it is based on a formula that uses the value of the function at the new point to implicitly solve for that point. It is different than BDF methods in that it is a one-step method. It is also A-stable, which means than its stability region includes the entire left half of the complex plane.

Table 14 shows the number of times the code evaluated the function. The number of function evaluations is by far the lowest of any of the methods. This solver evaluates the function only 35 times for the $N = 16$ case, compared with the next lowest method, which is the 94,882 times the stiff VODE method evaluates the function. In fact, the $N = 8$ case is the only time any of the other methods are close in the number of function evaluations. In that case, the stiff VODE method evaluates the function 83 times, while the implicit Runge-Kutta solver evaluates it 51 times.

Notice the solver evaluated the function fewer times for the $N = 8$ case than the $N = 4$ case, and still fewer for the $N = 16$ case. As with the $N = 8$ case for the stiff VODE solver, the reason for this is not clear.

The code run time, as shown in Table 15, is also much lower than the run time of any other method. Here the longest system time used is 6.2 seconds in the $N = 16$ case. The other methods take at least ten times as much system time for the same number of basis functions.

Tables 16 and 17 show the absolute errors and the ratios of the absolute errors. Once more, the method exhibits $\mathcal{O}(h_x^4)$ convergence, with the smallest ratio of the absolute errors being 24.069, occurring in the tangential displacement v when the number of basis functions doubles from $N = 8$ to $N = 16$.

The errors in the displacements are graphically the same as those for the first three solvers, so they are not plotted here.

$N = 4$	$N = 8$	$N = 16$
252	51	35

Table 14: Function evaluations for the implicit Runge-Kutta solver.

	$N = 4$	$N = 8$	$N = 16$
real	12.8	29.8	144.2
user	10.7	26.0	129.7
system	2.0	2.5	6.2

Table 15: Run time for the implicit Runge-Kutta solver.

	$N = 4$	$N = 8$	$N = 16$
u	$2.3951 - 1$	$2.8839 - 3$	$5.6671 - 5$
v	$8.1893 - 1$	$2.1608 - 2$	$8.9777 - 4$
w	$4.5161 - 1$	$5.8681 - 3$	$7.9953 - 5$

Table 16: Absolute errors of the longitudinal u, tangential v, and transverse displacements w for the implicit Runge-Kutta solver.

	$\frac{N = 4}{N = 8}$	$\frac{N = 8}{N = 16}$
u	83.052	50.888
v	37.899	24.069
w	76.961	73.395

Table 17: Ratios of the absolute errors for the implicit Runge-Kutta solver.

7 The Trapezoid Method

In addition to the four software packages, a subroutine using a modified trapezoid method was programmed and tested. Based on the theta method as outlined in [6], the method is described by the equation

$$y_{n+1} - y_n = h((1-\theta)f_{n+1} + \theta f_n) \ . \tag{7.1}$$

Notice when $\theta = \frac{1}{2}$ this reduces to the familiar trapezoid formula

$$y_{n+1} - y_n = \frac{h}{2}(f_n + f_{n+1}) \ .$$

This method is A-stable, like the implicit Runge-Kutta method. However, unlike the packaged solvers, which all had higher order convergence, it only has $\mathcal{O}(h_t^2)$ convergence. This means it must use a much smaller temporal step size to maintain the same accuracy. Thus, we expect the solver to evaluate the function a significantly larger number of times than any of the packaged solvers did.

The method, as implemented, uses a uniform step size in time and, unlike the packaged software, does not automatically adjust the step size. Because parameter estimation requires knowing the solution at many times, this uniform step size has a distinct advantage in estimating the parameters of a system. When using any of the packaged systems with automatic step size finding routines, parameter estimation requires interpolating the data or stopping the solver at many different times T. The uniform step size allows us to find the solution at many evenly spaced points without forcing the solver to ignore the optimal step size it would otherwise use.

This solver uses a predictor-corrector implementation of the trapezoid method. First, Euler's method predicts a value for y_{n+1}. Then the solver uses this value to obtain f_{n+1} for the theta method (7.1) to find a corrected value of y_{n+1}. Because of this implementation, the solver evaluates the function twice for each step in time. This means the uniform step size, which is set beforehand, dictates exactly how many times the solver must evaluate the function.

One major disadvantage of not having an automatic step size-finding routine built into the solver is that the user must decide beforehand what step size to use. In this investigation, the step size was determined by trying several step sizes and finding the largest step size that yielded the necessary accuracy. The final step sizes are $h_t = 10^{-5}$ for the $N = 4$ case, $h_t = 5 \cdot 10^{-6}$ for the $N = 8$ case, and $h_t = 2.5 \cdot 10^{-6}$ for the $N = 16$ case. These gave absolute errors for the longitudinal, tangential, and transverse displacements, as well as the ratios between the errors of the different cases, which are comparable with those of the packaged solvers (Tables 18 and 19).

The errors of the longitudinal, tangential, and transverse displacements are qualitatively the same as with the packaged solvers, and thus they are not plotted here.

Finally, Tables 20 and 21 show the number of function evaluations and the system run time, respectively. Notice the trapezoid method evaluates the function many more times than any other method considered and takes far more system to run. This was to be expected because of the lower order of convergence for the method.

	$N = 4$	$N = 8$	$N = 16$
u	$2.3952 - 1$	$2.8839 - 3$	$6.2672 - 5$
v	$8.1892 - 1$	$2.1608 - 2$	$9.0377 - 4$
w	$4.5165 - 1$	$5.8681 - 3$	$8.6737 - 5$

Table 18: Absolute errors of the longitudinal u, tangential v, and transverse displacements w for the trapezoid solver.

	$\frac{N = 4}{N = 8}$	$\frac{N = 8}{N = 16}$
u	83.053	46.016
v	37.899	23.909
w	76.966	67.654

Table 19: Ratios of the absolute errors for the trapezoid solver.

$N = 4$	$N = 8$	$N = 16$
$240,002$	$480,000$	$960,002$

Table 20: Function evaluations for the trapezoid solver.

	$N = 4$	$N = 8$	$N = 16$
real	940.2	8272.9	$63,307.7$
user	763.2	7492.9	$62,430.5$
system	8.8	49.9	244.5

Table 21: Run time for the trapezoid solver in seconds.

8 Conclusion

Efficient temporal solution of structural problems is an important problem, especially in the large ODE systems resulting from coupling structural models with fluid or acoustic equations to model fluid-structure or structural acoustic systems. Efficiency becomes even more crucial in parameter estimation studies, in which the solver must evaluate the system multiple times, and control implementation, where the goal is for the solver to work at the same speed as the system evolves in real life.

In this work, the problem of efficient temporal solutions of structural problems was addressed through a comprehensive study of three commercial ODE packages as well as programming and investigation of a method currently being used in applications in fluid dynamics. This investigation was performed in the context of approximating the dynamics of a thin cylindrical shell, a structure which appears commonly in applications and whose model is sufficiently complex that relatively large ODE systems arise.

The three commercial ODE packages used variable-order methods or methods with 4^{th} or 5^{th} order convergence and automatic step size finding routines. The trapezoid method, on the other hand, only had order $\mathcal{O}(h_t^2)$ convergence, and a uniform step size which must be determined before the solver can approximate the system. However, the uniform step size of the trapezoid method gives it a distinct advantage in parameter estimation studies in which multiple system evaluations are required. For all the other methods, data must be interpolated, or the system must be stopped at many different points.

Each method was evaluated on three criteria: preservation of the accuracy of the spatial discretization, amount of required system time, and number of function evaluations.

The method had to preserve the accuracy of the spatial discretization throughout the interval in which we were interested. A Fourier/cubic spline method was used to discretize the spatial components, yielding an order $\mathcal{O}(h_x^4)$ spatial convergence rate for spatial gridlengths h_x. The various methods were all required to maintain this accuracy when integrating the systems through time to $T = 1.2$ seconds.

To measure this convergence rate, we looked at the ratios of the absolute errors for the cases $N = 4$ and $N = 8$ and also for $N = 8$ and $N = 16$. Because we doubled the number of basis functions in each of these, we expect the ratios to all be at least as large as $2^4 = 16$. In fact, the smallest ratio for each of the methods was approximately 24 and occurred in the tangential displacement v when the number of basis functions doubles from $N = 8$ to $N = 16$. Thus, all five methods exhibited the expected order $\mathcal{O}(h_x^4)$ convergence rate.

The other two criteria considered dealt with the efficiency of the solvers. One criterion was the amount of system time required by each method. As summarized in Table 22, the implicit Runge-Kutta solver used far less system time than any of the other methods, using less than one tenth the amount of system time of the other methods for the $N = 16$ case. The trapezoid method, on the other hand, took far more time to run than the other methods, using over forty times the amount of system time used by the implicit Runge-Kutta for $N = 16$. This, however, was to be expected since it has order $\mathcal{O}(h_t^2)$ convergence, while the other methods all have order $\mathcal{O}(h_t^4)$ convergence.

Because the system time can vary depending on what else is running on the machine, it can only give a qualitative measure of a method's efficiency. A much better measure quantitatively is the number of times a solver must evaluate the function to determine the

approximate derivative of the state vector. Once again, the implicit Runge-Kutta takes the least amount of system time to maintain the needed accuracy, and the Trapezoid method takes the most (Table 23).

Of the five methods, the implicit Runge-Kutta appears to be the most efficient. It takes the least system time and number of function evaluations to maintain the accuracy of the spatial discretization.

	$N = 4$	$N = 8$	$N = 16$
Stiff VODE	4.8	2.8	65.1
Nonstiff VODE	2.8	7.3	90.9
Explicit Runge-Kutta	4.0	9.7	127.0
Implicit Runge-Kutta	2.0	2.5	6.2
Trapezoid	8.8	49.9	244.5

Table 22: Summary of system time used (seconds).

	$N = 4$	$N = 8$	$N = 16$
Stiff VODE	41,864	83	94,882
Nonstiff VODE	86,946	149,678	267,483
Explicit Runge-Kutta	260,117	361,916	711,461
Implicit Runge-Kutta	252	51	35
Trapezoid	240,002	480,000	960,002

Table 23: Summary of the function evaluations.

References

[1] H.T. Banks and R.C. Smith, "Well-posedness of a Model for Structural Acoustic Coupling in a Cavity Enclosed by a Thin Cylindrical Shell," *Journal of Mathematical Analysis and Applications*, 191, pp. 1-25, 1995.

[2] H.T. Banks, R.C. Smith, and Y. Wang, "The Modeling of Piezoceramic Patch Interactions with Shells, Plates, and Beams," *Quarterly of Applied Mathematics*, 53(2), pp. 353-381, 1995.

[3] H.T. Banks, R.C. Smith, and Y. Wang, *Smart Material Structures: Modeling, Estimation and Control*, to be published by Masson/Wiley in the collection "Recherches en Mathématiques Appliquées."

[4] P.N. Brown, G.D. Byrne, and A.C. Hindmarsh, "VODE: A Variable Coefficient ODE solver," *SIAM J. Sci. Stat. Comput.*, 10 (1989), pp. 1038-1051. Also, LLNL Report UCRL-98412, June 1988.

[5] C.R. Fuller, S.D. Snyder, C.H. Hansen, and R.J. Silicox, "Active Control of Interior Noise in Model Aircraft Fuselages Using Piezoceramic Actuators," Paper 90-3922, AIAA 13th Aeroacoustics Conference, Tallahassee, FL, October 1990.

[6] J.D. Lambert, *Numerical Methods for Ordinary Differential Systems: The Initial Value Problem*, Wiley, New York, 1991.

[7] A.W. Leissa, *Vibration of Shells,* NASA SP-288, 1973; Reprinted by the Acoustical Society of America through the American Institute of Physics, 1993.

[8] S. Markuš, *The Mechanics of Vibrations of Cylindrical Shells*, Elsevier, New York, 1988.

[9] P.M. Prenter, *Splines and Variational Methods*, Wiley Classics Edition, New York, 1989.

[10] R.C.H. del Rosario and R.C. Smith, "Spline Approximation of Thin Shell Dynamics - Numerical Examples," Center for Research in Scientific Computation Technical Report CRSC-TR96-5.

[11] R.C.H. del Rosario and R.C. Smith, "Spline Approximation of Thin Shell Dynamics," ICASE Report 96-26, *International Journal for Numerical Methods in Engineering*, submitted.

[12] L.F. Shampine, *Numerical Solution of Ordinary Differential Equations*, Chapman & Hall, New York, NY, 1994.

[13] W. Soedel, *Vibrations of Shells and Plates*, 2nd ed. Marcel Dekker, New York, 1993.

[14] W. Soedel and M. Dhar, "Difficulties in Finding Modes Experimentally when Several Contribute to a Resonance," *Journal of Sound and Vibration*, 58(1), pp. 27-28, 1978.