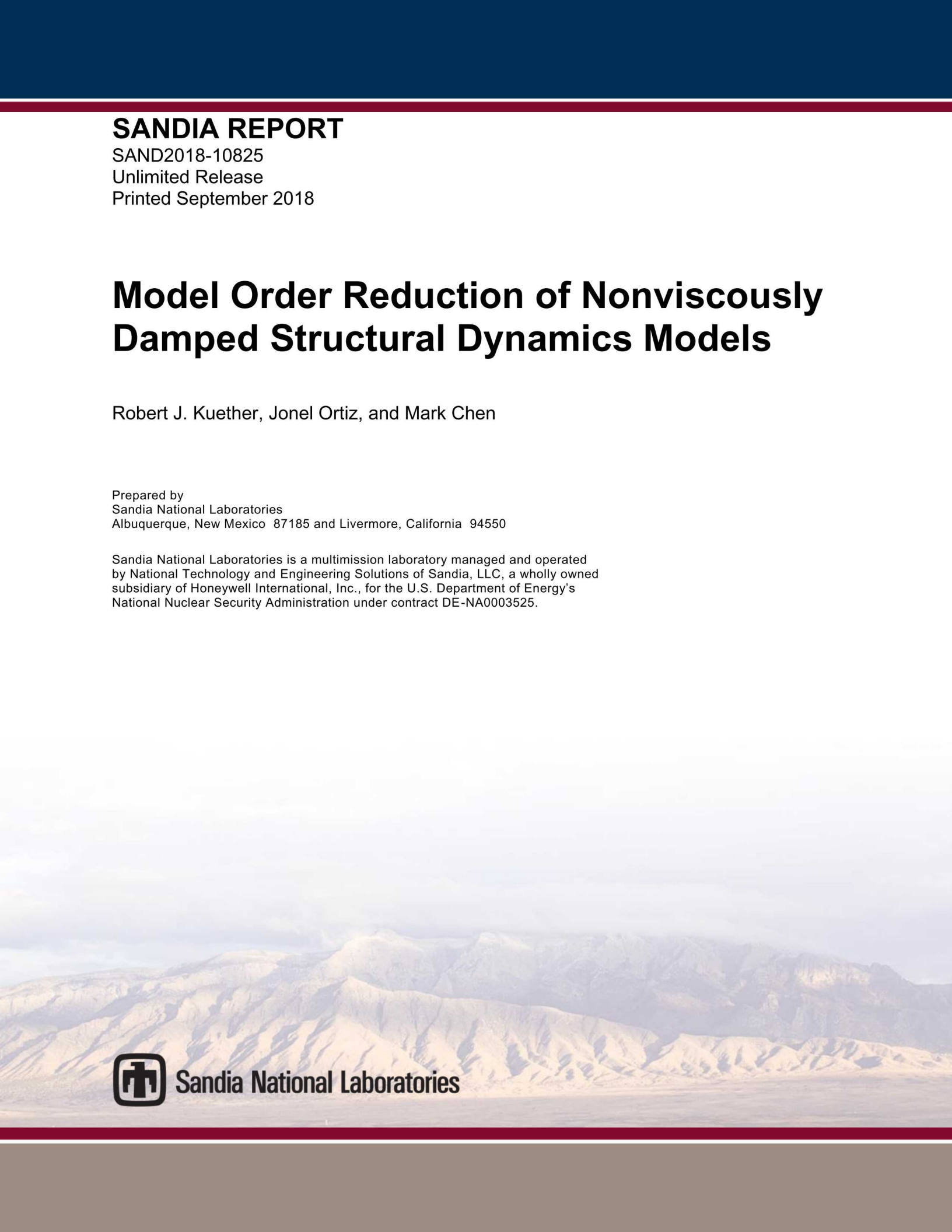
**SANDIA REPORT**
SAND2018-10825
Unlimited Release
Printed September 2018

# Model Order Reduction of Nonviscously Damped Structural Dynamics Models

Robert J. Kuether, Jonel Ortiz, and Mark Chen

Prepared by
Sandia National Laboratories
Albuquerque, New Mexico 87185 and Livermore, California 94550

Sandia National Laboratories is a multimission laboratory managed and operated by National Technology and Engineering Solutions of Sandia, LLC, a wholly owned subsidiary of Honeywell International, Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525.



Issued by Sandia National Laboratories, operated for the United States Department of Energy by National Technology and Engineering Solutions of Sandia, LLC.



Available to DOE and DOE contractors from

U.S. Department of Energy
Office of Scientific and Technical Information
P.O. Box 62
Oak Ridge, TN 37831

Telephone: (865) 576-8401
Facsimile: (865) 576-5728
E-Mail: reports@osti.gov
Online ordering: http://www.osti.gov/scitech

Available to the public from

U.S. Department of Commerce
National Technical Information Service
5301 Shawnee Rd
Alexandria, VA 22312

Telephone: (800) 553-6847
Facsimile: (703) 605-6900
E-Mail: orders@ntis.gov
Online order: http://www.ntis.gov/search

SAND2018-10825
Printed September 2018
Unlimited Release

# Model Order Reduction of Nonviscously Damped Structural Dynamics Models

Robert J. Kuether, Jonel Ortiz
Component Science and Mechanics (Org. 01556)
Mark Chen
Computational Solid Mechanics and Structural Dynamics (Org. 01542)
Sandia National Laboratories
P. O. Box 5800
Albuquerque, New Mexico 87185-0346

## Abstract

A reduced order modeling capability has been developed to reduce the computational burden associated with time-domain solutions of structural dynamic models with linear viscoelastic materials. The discretized equations-of-motion produce convolution integrals resulting in a linear system with nonviscous damping forces. The challenge associated with the reduction of nonviscously damped, linear systems is the selection and computation of the appropriate modal basis to perform modal projection. The system produces a nonlinear eigenvalue problem that is challenging to solve and requires use of specialized algorithms not readily available in commercial finite element packages. This SAND report summarizes the LDRD discoveries of a reduction scheme developed for monolithic finite element models and provides preliminary investigations to extensions of the method using component mode synthesis. In addition, this report provides a background overview of structural dynamic modeling of structures with linear viscoelastic materials, and provides an overview of a new code capability in Sierra Structural Dynamics to output the system level matrices computed on multiple processors.

## ACKNOWLEDGMENTS

Supported by the Laboratory Directed Research and Development program at Sandia National Laboratories, a multimission laboratory managed and operated by National Technology and Engineering Solutions of Sandia, LLC., a wholly owned subsidiary of Honeywell International, Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA-0003525.

In addition, the authors would like to acknowledge a number of people who helped contribute to this research. First, we would like to thank Kevin Troyer for his contributions during the proposal and early stages of the work. His insight into viscoelasticity was tremendously beneficial to the physical and numerical aspects of the model reduction approach. Next we would like to thank the Sierra/SD code development team for explaining aspects of the code implementation and fixing various bugs that were identified throughout the LDRD project term. In particular, we are grateful for the conversations and support provided by Garth Reese, Tim Walsh, and Nate Crane. We are thankful to the LDRD management team, Basil Hassan and Jim Redmond, for their support from the program office. Finally, we would like to acknowledge various staff members who have helped us identify specific areas in which this early-career LDRD could benefit engineering design and analysis. Special thanks to Jill Blecke and Adam Brink for the numerous conversations related to viscoelastic materials in computational mechanics and dynamics.

## TABLE OF CONTENTS



## FIGURES



## TABLES

## NOMENCLATURE

| Abbreviation | Definition |
|---|---|
| **DOF** | Degrees-of-freedom |
| **EOM** | Equations-of-motion |
| **FEA** | Finite Element Analysis |
| **GHM** | Golla-Hughes-McTavich |
| **GRE** | Global Relative Error |
| **HCB** | Hurty/Craig-Bampton |
| **HHT** | Hilber-Hughes-Taylor |
| **LDRD** | Lab Directed Research and Development |
| **Sierra/SD** | Sierra/Structural Dynamics |
| **SNL** | Sandia National Laboratories |
| **ROM** | Reduced Order Model |
| **WLF** | Williams-Landel-Ferry |

# 1. INTRODUCTION

## 1.1. Background

Linear viscoelastic materials are commonly used in mechanical, biomedical and aerospace engineering disciplines due to their unique properties. For example, the aerospace and automotive industries often incorporate viscoelastic materials into mechanical designs to passively reduce excessive vibration levels seen during operation [1]. Constrained layer damping treatments add a lightweight, viscoelastic core between two stiff panels or components to mitigate vibrational energy and reduce weight. Core materials such as polymers and foams quickly dissipate energy and are excellent choices for achieving the desired performance. Viscoelastic materials are also used to encapsulate critical structural components such as payloads or electronics in order to isolate them from the potentially damaging environments. Because of the importance of these materials throughout the engineering sciences, there is a need to develop accurate and efficient models that capture their time-dependent physical behavior in the time-domain. Finite element analysis (FEA) is a powerful technique to generate structural dynamic models of components with complicated geometries and constitutive laws describing viscoelastic behavior.

The challenge associated with solving structural dynamic models with viscoelasticity is the fact that classical modal decomposition approaches are no longer valid. Time-domain solutions of high-fidelity models with millions of degrees-of-freedom (DOF) require direct numerical integration of the full-order model. This approach demands enormous computational resources compared to traditional modal methods. Additionally, current reduced order modeling (ROM) techniques, such as the Hurty/Craig-Bampton (HCB) method [2, 3], are developed for *linear elastic* FEA models, but they cannot incorporate the time-dependent material behavior exhibited by foams, rubbers, or composites. This SAND report documents the results from a three-year, early-career LDRD project titled “Reduced Order Models of Structures Incorporating Complex Materials”. We proposed to develop a ROM capability that incorporates time-dependent material behavior to significantly improve the computational efficiency and accuracy of large, component-level simulations of dynamic loading events. This document provides a summary of the project outcomes and demonstrates how the team was able to deliver on its objectives.

The focus of this work was to develop reduction methods for FEA models with Prony series representations of the time-dependent moduli of viscoelastic materials. A number of constitutive models for viscoelastic materials are available in the literature [4, 5], but the Prony series was chosen since it is i.) easily implemented into computational tractable numerical integration schemes, and ii.) is currently the only implementation within the Sierra Structural Dynamics (Sierra/SD) finite element code [6] at Sandia National Laboratories (SNL). One of the most common reduction techniques in structural dynamics is the projection-based method, where a subset of deflection shapes are projected onto the system matrices of the governing equations-of-motion (EOM). Model reduction techniques are typically evaluated based on their efficiency to reduce the online computational costs of the model, as well as their accuracy to preserve the model predictions in comparison to the full-order model. Rouleau et al. [7] recently published a paper reviewing projection-based methods as applied to frequency-domain solutions of linear viscoelastic FEA models.

As Rouleau et al. concludes, the greatest challenge associated with projection-based methods is the selection of the appropriate basis vectors. For linear structural dynamics models with light, *viscous* damping, the superposition of real-valued eigensolutions supplemented with static correction vectors [8, 9] serve as the optimal choice to reduce the EOM. These bases ignore damping forces in their calculation and are readily selected based on the frequency bandwidth of interest. Bilasse et al. [10] examined the use of four different mode shapes to create ROMs of FEA models of viscoelastic sandwich beams. They explored the use of real eigenmodes, improved real eigenmodes, approached complex eigenmodes and exact complex eigenmodes. The results suggest that the two bases involving complex eigenmodes work well, and the real modes present erroneous results but could be improved by linearizing the frequency dependent matrices about a non-zero frequency. Bilasse and Oguamanam [11] reduce the frequency-domain equations of sandwich plate structures using real eigenmodes and exact complex eigenmode basis. They found that complex modes work best for higher damping levels (albeit at a higher upfront computational cost), while real modes work sufficiently well for lower damping levels.

Much of the existing literature addresses techniques applied to frequency-domain solutions, and few researchers address ROMs that are solved in the time-domain with numerical integration. de Lima et al. [12] utilize the static residual basis [13] to reduce a time-domain FEA model and use Newmark integration schemes to solve the equations. The constitutive behavior was modeled with a four parameter fractional derivative model [14]. Other time-domain reduction techniques reduce the Golla-Hughes-McTavish (GHM) model [15], but the use of internal variables greatly increases the order of the equations to be solved. The objective of the early-career LDRD project was to develop a viscoelastic ROM framework that could be solved in the time-domain, provide accurate and efficient response predictions, maintain parametric dependence on temperature/frequency, and minimize the offline costs associated with computation of the reduction basis. The next subsection summarizes the project outcomes that meet each of these objectives to deliver a novel ROM capability that accounts for viscoelasticity.

## 1.2. Summary of LDRD Publications

The LDRD project was able to document the research discoveries made in various conference and journal papers throughout the three year term. The most notable contribution from this project is the journal article titled “Two-Tier Model Reduction of Viscoelastically Damped Finite Element Models” [16]. Two conference papers were also written prior to this to disseminate developments of the work to the greater research community. This served as a valuable tool to discuss ideas and methods with other researchers. The first conference paper published from the project was presented at the 2016 ISMA conference titled “Time Domain Model Reduction of Linear Viscoelastic Finite Element Models” [17]. One year later, another conference paper was published and presented at the 2017 IMAC XXXV conference titled “Substructuring of Viscoelastic Subcomponents with Interface Reduction” [18]. A brief summary of these works are provided below, but the interested reader should refer to the full publications for complete detail.

Analogous to model reduction theory for linear systems, the complex, frequency-dependent modes of the linear viscoelastic system generally serve as an ideal reduction basis. These mode shapes take into account the nonviscous damping forces in the governing equations and allow for frequency-based mode selection. As discovered by Bilasse et al. [10], the exact complex modes provide highly accurate results, particularly in the case when damping forces are large.

Unfortunately, one must solve a nonlinear eigenvalue problem that is challenging to compute numerically compared to its linear counterpart. These modes require specialized solvers not typically available in commercial finite element codes with parallel processing capabilities. Several researchers have developed advanced algorithms capable of solving such problems [19-21], but these algorithms remain computationally expensive when applied to large-scale problems. The LDRD research addresses this issue by developing a novel, two-tier reduction scheme [16] that efficiently obtains an approximation of the exact complex modes and reduces the full-order EOM using these ideal shapes.

The first reduction, termed the tier-one reduction step, uses the Multi-Model (MM) approach [22] developed by Balmès to compute several modal bases using real eigensolutions linearized about various frequency discretizations of the viscoelastic material. The mode calculations leverage real eigenmode solvers commonly found in commercial FEA codes. The tier-one reduction preserves the parametric temperature dependence since the model is sampled at multiple points on the material time-temperature scale. The tier-two reduction follows by fixing the operation temperature of the material and exactly solving the nonlinear eigenvalue problem using Newton's method [23] within the tier-one reduced space. This basis reduces the EOM using the real and imaginary parts of the complex eigenvectors and allows for mode truncation based on frequency content. The offline cost of solving the nonlinear eigenvalue problem in the tier-one reduced space is significantly lower in comparison to the solution of the full-order equations. This approach builds upon the original work presented in [17], which uses an iterative approach to approximate the exact modes from the full-order model. This motivated the theoretical developments in [16] to derive an efficient approach to compute the exact complex mode shapes without needing to solve nonlinear eigenvalue problems of the full-order model.

The model reduction technique was applied to a finite element model of a cantilevered sandwich plate to evaluate accuracy and efficiency. The inner, constrained layer served as the core material and was modeled as a viscoelastic solid with properties of 828 DEA. The outer two layers were structural components and were modeled as linear elastic with properties of Aluminum 6061-T6. The mesh is shown in Figure 1.

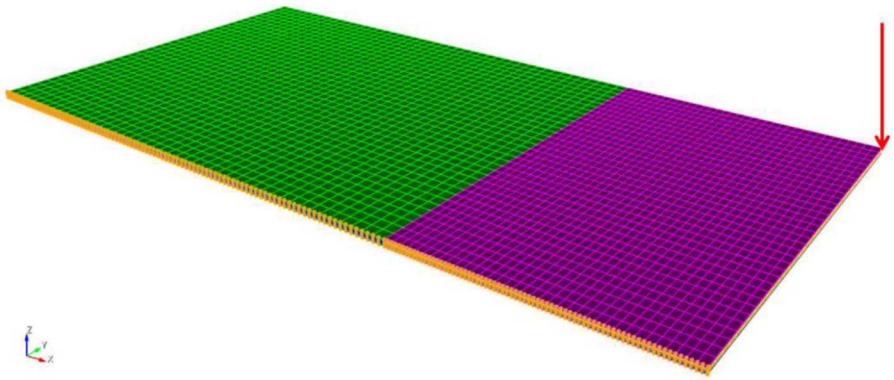


**Figure 1. Finite element mesh of cantilevered sandwich plate.**

The results from the tier-two ROMs were evaluated by predicting the transient response to a haversine impulse with a peak of 20 lbf and 0.004 s pulse duration. Each tier-two ROM was created by first choosing an operating temperature of the model, and then calculating the exact

complex mode solutions. Three different ROMs were evaluated as having cut-off frequencies of 750 Hz, 1000 Hz, and 1250 Hz. A direct comparison of the transient displacement predicted from the ROM and full-order model was computed using the Global Relative Error (GRE) metric proposed by Farhat et al. in [24]. The GRE for each tier-two ROM over the full temperature spectrum is plotted in Fig. 2.

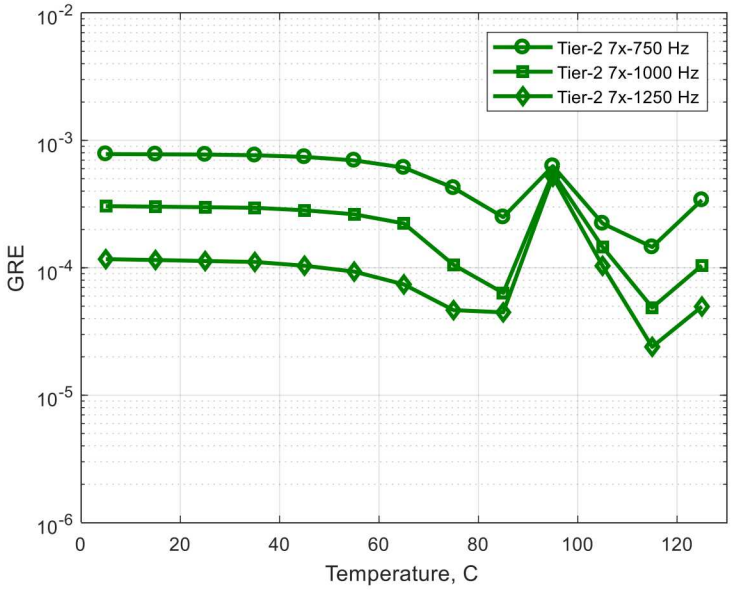



**Figure 2. GRE for transient response predicted by each tier-two ROM.**

The accuracy obtained with the tier-two models were extremely high as GRE's over the full temperature range were less than 0.1 %. As with all reduced order modeling techniques, there are offline costs associated with the computation of the modal reduction basis, and the online costs associated with numerical time integration. A summary of the computational costs of each tier-two ROM is provided in Table 1. The offline costs correspond to calculation of the tier-one and tier-two modal bases, which are compared alongside the online cost of the total simulation time required to numerically integrate the response to the haversine load. Each comparison is for the numerical simulation at an operating temperature of 95 °C with the same time step and period of $\Delta t = 1e-6$ s and 0.1 s, respectively, resulting in a total of 100,000 numerical solves. The tier-two ROMs are five to six orders of magnitude faster than the full-order model when computing the time integrated solution. This significant speedup shows how efficient the models are in terms of online cost.

**Table 1. Computational cost of mode calculations and time integration.**

| ROM | Total DOF | Tier-one mode calculation | Tier-two mode calculation | Offline cost (mode calculation) | Online cost (numerical simulation) |
|---|---|---|---|---|---|
| Full-order Model | 130,305 | - | - | 0 s | 4.51E+06 s |
| Tier-2 7x–750 Hz | 30 | 832.4 s | 13.8 s | 846.2 s | 7.2 s |
| Tier-2 7x–1000 Hz | 38 | 832.4 s | 16.4 s | 848.8 s | 12.8 s |
| Tier-2 7x–1250 Hz | 48 | 832.4 s | 20.6 s | 853.0 s | 14.5 s |

A component mode synthesis approach was developed [18] using an iterative approach to approximate the exact, complex fixed-interface modes from each subcomponent model. An interface reduction was also applied to the reduction basis to further reduce the boundary DOF associated with the connection points. The approach achieved similar levels of accuracy as the work presented in [17], but was never extended to the two-tier approach due to time constraints of the project. Future works could extend the two-tier approach in [16] to component mode synthesis techniques to provide an efficient modal basis that preserves parametric dependence of operating temperature. This would provide a powerful simulation tool that would enable analysts to specify different operating temperatures of the subcomponents within an assembly and perform efficient, parametric simulations to explore the global system response at various temperatures.

## 1.3. Outline of Report

The interested reader should refer to the external publications [16-18] for details related to the ROM approaches developed for monolithic and substructure models. The remainder of this report describes the relevant background related to structural dynamics modeling. Section 2 provides a complete background of the finite element modeling theory for structures modeled with viscoelastic materials described by Prony series representations. This overview is intended to prepare the reader with the appropriate background to understand the model reduction schemes developed in the external papers. In Section 3, an overview of a newly developed Sierra/SD code capability is described. The code development team created MATLAB scripts to properly reassemble system matrices and displacements output on multiple processors. This capability was needed to allow the ROM approach to scale to large-order models common to those developed at SNL. Conclusions from the report are provided in Section 4.

# 2. REVIEW OF LINEAR VISCOELASTIC FINITE ELEMENT MODELS

This section provides background information regarding spatial discretization of finite element models with linear viscoelastic materials modeled with a Prony series representation. The Sierra/SD finite element code at SNL utilizes this generalized formulation to solve time- and frequency-domain analyses. This sections provides an overview of the governing equations, projection-based model reduction, nonlinear eigenvalue problem, and frequency- and time-domain solution algorithms.

## 2.1. Governing Equations

Finite element modeling is a common approach to solve structural dynamic problems of components or systems with complicated geometries. The time-domain, space discretized equations-of-motion with linear viscoelastic constitutive models are obtained using standard finite element techniques [8],

$$\mathbf{M}\ddot{\mathbf{x}} + \mathbf{C}\dot{\mathbf{x}} + \mathbf{K}_K \int_0^t \zeta_K(t-\tau)\dot{\mathbf{x}}(\tau)d\tau + \mathbf{K}_G \int_0^t \zeta_G(t-\tau)\dot{\mathbf{x}}(\tau)d\tau + \mathbf{K}_\infty \mathbf{x} = \mathbf{f}(t) \tag{1}$$

The coupled integro-differential equations consist of real, symmetric $N \times N$ matrices for mass, $\mathbf{M}$, viscous damping, $\mathbf{C}$, viscoelastic bulk stiffness, $\mathbf{K}_K$, viscoelastic shear stiffness, $\mathbf{K}_G$, and elastic stiffness, $\mathbf{K}_\infty$. The physical displacements are the $N \times 1$ vector $\mathbf{x}$ where the overdot corresponds to the derivative with respect to time. The $N \times 1$ vector $\mathbf{f}(t)$ is the external time varying force applied to the model.

The discretization in Eq. (1) assumes an isotropic, linear viscoelastic solid whose three-dimensional constitutive stress equation is separated into dilatational and deviatoric components of the strain,

$$\sigma_{ij}(t) = \int_0^t 3K(t-\tau)\left(\frac{1}{3}\delta_{ij}\frac{d\varepsilon_{kk}}{d\tau}\right)d\tau + \int_0^t 2G(t-\tau)\left(\frac{d\varepsilon_{ij}}{d\tau} - \frac{1}{3}\delta_{ij}\frac{d\varepsilon_{kk}}{d\tau}\right)d\tau \tag{2}$$

The time dependent stress, $\sigma_{ij}(t)$, is written in Einstein summation form where $i,j$ = 1, 2 or 3. In Eq. (2), $\delta_{ij}$ is the Dirac delta function, $\varepsilon_{ij}$ is the strain, $t$ and $\tau$ are the time and integration time variables, respectively, and $\varepsilon_{kk} = \varepsilon_{11} + \varepsilon_{22} + \varepsilon_{33}$. The summation of two independent viscoelastic functions for the bulk relaxation modulus, $K(t)$, and shear relaxation modulus, $G(t)$, may have different time dependence. The advantage of using dilatational and deviatoric components in the constitutive equation is that the viscoelastic terms in Eq. (2) are separated into two viscoelastic functions that allow a constant matrix to be factored out when discretizing the equations. Other representations, such as those written in terms of Young's modulus and Poisson's ratio, do not produce a decoupled form and complicate model order reduction operations.

The bulk and shear relaxation moduli are written in a general form in Eq. (2). There are a variety of functional forms that describe the material behavior, such as the four-parameter fractional derivative model [14] or the Golla-Hughes-McTavish (GHM) model [25, 26]. Adhikari [4] and Lakes [5] review several functions to describe linear viscoelastic constitutive models. The shear and bulk relaxation used throughout this work are described by a series of exponential functions

of varying amplitudes and decay constants, referred to as the Prony series model. Other references refer to this as the Biot model [27]. Since both the bulk and shear relaxation moduli have the same series exponential form, only the bulk Prony series is developed for brevity. The bulk relaxation modulus from Eq. (2) has the form,

$$K(t) = K_\infty + \left(K_g - K_\infty\right)\zeta_K(t) \tag{3}$$

where

$$\zeta_K(t) = \sum_{i=1}^{N_K} \widehat{K}_i e^{\frac{-t}{\tau_{K,i}}} \tag{4}$$

and the coefficients are normalized to unity,

$$\sum_{i=1}^{N_K} \widehat{K}_i = 1 \tag{5}$$

The rubbery and glassy moduli, $K_\infty$ and $K_g$, respectively, are considered the long-term (soft) and short-term (hard) moduli of the viscoelastic material. The time-dependent kernel function, $\zeta_K(t)$, is a summation of $N_K$ exponential functions with an amplitude coefficient $\widehat{K}_i$ for a prescribed time constant $\tau_{K,i}$. The shear and bulk Prony terms are obtained by fitting experimental master curves for the material of interest. In general, $\zeta_K(t)$ is restricted to be continuous and monotonically decreasing [28], thus requiring that $\widehat{K}_i$ and $\tau_{K,i}$ be positive values.

For thermorheologically simple viscoelastic materials [29], the principle of time-temperature superposition holds and allows for the temperature dependent material properties to be described by shifting the effective time scale of the material. A change in temperature causes the properties to strictly move horizontally along either the time or frequency axis of the master curve. The shift factor, $a_T$, is an empirical value obtained by measuring material properties at different operating temperatures, $T$. When the temperature is above the glass transition temperature, the shift factors are fit to the WLF equation [30],

$$\log(a_T) = -\frac{C_1(T-T_{ref})}{C_2+(T-T_{ref})} \quad \forall\, T \geq T_{ref} \tag{6}$$

Below the glass transition temperature, the shift factor is fit to the following equation [6],

$$\log(a_T) = A_1\left(1 - e^{A_2(T-T_{ref})}\right) \quad \forall\, T < T_{ref} \tag{7}$$

The scalar constants $A_1$, $A_2$, $C_1$, and $C_2$ depend on the material of interest. To shift the material parameters to a operating temperature away from the reference temperature, $T_{ref}$, the Prony series is shifted by multiplying the shift factor by each time constant, i.e. $a_T\tau_{K,i}$. In Eq. (1), the viscoelastic damping matrices are independent of the kernel functions $\zeta_K(t)$ and $\zeta_G(t)$, and hence the matrices do not need to be reformulated for each temperature.

## 2.2. Model Order Reduction

A projection based reduction scheme can be utilized to reduce the physical equations in Eq. (1) onto a lower order subspace. The reduction basis must be formulated to be sufficiently small to reduce the size of the time-domain equations. For model reduction schemes, the main objective is to achieve computational speedups while maintaining acceptable accuracy in reference to the full order model solutions. In general, a so-called modal-based reduction approximates the physical DOF, **x**, in Eq. (1) with a reduced dimensional subspace, **q**, using a set of shapes,

$$\mathbf{x} \approx \mathbf{T}\mathbf{q} \tag{8}$$

The transformation matrix, $\mathbf{T}$, has dimension $N \times N_r$ whose columns are a collection of shape vectors used to define the kinematics of the reduction space. In general, these shapes may consist of vibration modes or static deflections that describe the dynamics in response to arbitrary time-varying loads, or other shapes computed from proper orthogonal decomposition. Since the reduction is applied in the time-domain, the matrix $\mathbf{T}$ is restricted to be real valued. The reduced, generalized coordinates, $\mathbf{q}$, are a $N_r \times 1$ vector of time-varying functions that described the amplitude of each shape. A Galerkin projection of Eq. (8) onto the finite element equations in Eq. (1) is written as,

$$\mathbf{T}^{\mathrm{T}}\mathbf{M}\mathbf{T}\ddot{\mathbf{q}} + \mathbf{T}^{\mathrm{T}}\mathbf{C}\mathbf{T}\dot{\mathbf{q}} + \mathbf{T}^{\mathrm{T}}\mathbf{K}_K\mathbf{T}\int_0^t \zeta_K(t-\tau)\dot{\mathbf{q}}(\tau)d\tau + \mathbf{T}^{\mathrm{T}}\mathbf{K}_G\mathbf{T}\int_0^t \zeta_G(t-\tau)\dot{\mathbf{q}}(\tau)d\tau + \mathbf{T}^{\mathrm{T}}\mathbf{K}_\infty\mathbf{T}\mathbf{q} = \mathbf{T}^{\mathrm{T}}\mathbf{f}(t) \tag{9}$$

where $(\cdot)^{\mathrm{T}}$ is the transpose operator. The reduced order equations in Eq. (9) are rewritten in condensed form,

$$\bar{\mathbf{M}}\ddot{\mathbf{q}} + \bar{\mathbf{C}}\dot{\mathbf{q}} + \bar{\mathbf{K}}_K\int_0^t \zeta_K(t-\tau)\dot{\mathbf{q}}(\tau)d\tau + \bar{\mathbf{K}}_G\int_0^t \zeta_G(t-\tau)\dot{\mathbf{q}}(\tau)d\tau + \bar{\mathbf{K}}_\infty\mathbf{q} = \bar{\mathbf{f}}(t) \tag{10}$$

The reduction is applied directly to the equations-of-motion by compressing the matrices offline. This operation is straightforward when using the separated formulation with shear and bulk viscoelastic terms since the matrices do not need to be reformulated for different operating temperatures. The reduced equations in Eq. (10) are then efficiently solved either in the time-domain using numerical time integration schemes, or in the frequency-domain by performing a Laplace transformation on the reduced equation. Once a solution of the generalized coordinates, $\mathbf{q}$, is obtained, the physical DOF responses are recovered through the transformation in Eq. (8).

## 2.3. Nonlinear Eigenvalue Problem

One natural choice of basis vectors to use would be the complex, frequency-dependent eigenvectors computed from the linear viscoelastic equations. Posing the eigenvalue problem from the autonomous form of Eq. (1) results in a nonlinear eigenvalue problem,

$$\left(\lambda_r^2\mathbf{M} + \lambda_r\mathbf{C} + \lambda_r\mathbf{K}_K\sum_{i=1}^{N_K}\frac{\hat{K}_i}{\lambda_r+1/\tau_{K,i}} + \lambda_r\mathbf{K}_G\sum_{i=1}^{N_G}\frac{\hat{G}_i}{\lambda_r+1/\tau_{G,i}} + \mathbf{K}_\infty\right)\boldsymbol{\phi}_r^* = \mathbf{0} \tag{11}$$

The $N \times 1$ vector $\boldsymbol{\phi}_r^*$ is the complex mode shape and $\lambda_r$ is the complex eigenvalue. The eigensolutions computed from Eq. (11) would form an ideal reduction basis in Eq. (8), analogous to real eigenvectors for undamped or lightly damped systems, since the shapes can be truncated based on the frequency content of interest. This nonlinear eigenvalue problem is challenging to solve numerically compared to its linear counterpart and its solvers are not readily accessible in commercially available finite element codes or in the Sierra finite element codes. Several researchers have developed advanced algorithms capable of solving such problems [19-21] but they remain computationally expensive especially for large-scale problems. Others have proposed solving the nonlinear eigenvalue problem by linearizing the eigenvalue term, $\lambda_r$, in the denominator of the Prony series about a chosen frequency and iteratively solving a linearized, quadratic eigenvalue problem [31-33]. Many finite element codes are capable of solving quadratic eigenvalue problems, however this approach requires several iterations to solve each complex mode solution and becomes prohibitively expensive for large-scale models. As mentioned in Section 1.2, a novel two-tier approach was developed in [16] to approximate the

exact complex modes without needing to run specialized, iterative algorithms on the full-order model.

## 2.4. Steady-state Response to Harmonic Excitation

The full-order model in Eq. (1) or the ROM in Eq. (10) can be solved in the frequency-domain by performing a Laplace transform on the equations-of-motion and assuming null initial conditions. The steady-state equations become,

$$\left(-\omega^2\mathbf{M} + \mathrm{i}\omega\mathbf{C} + \mathrm{i}\omega\mathbf{K}_K \sum_{i=1}^{N_K} \frac{\hat{K}_i}{\mathrm{i}\omega + 1/\tau_{K,i}} + \mathrm{i}\omega\mathbf{K}_G \sum_{i=1}^{N_G} \frac{\hat{G}_i}{\mathrm{i}\omega + 1/\tau_{G,i}} + \mathbf{K}_\infty\right)\mathbf{X} = \mathbf{F} \tag{12}$$

where the $N \times 1$ vectors $\mathbf{X}$ and $\mathbf{F}$ are the complex response and input force vector, respectively. The response is computed by inverting the left-hand side of Eq. (12) for a particular input frequency. For large-scale model, the steady-state response becomes expensive to solve numerically since the matrix needs to be updated for each frequency line prior to inversion. A reduced order modeling approach significantly decreases the computational burden by reducing the size of the matrix.

## 2.5. Transient Response with Numerical Time Integration

Direct numerical integration of the full-order model in Eq. (1) or the ROM in Eq. (10) requires evaluation of the convolution integrals for the shear and bulk viscoelastic forces at the current time step. Computing these integrals or storing the history of the viscoelastic force quickly becomes intractable for reasonably large problems integrated over long periods. It is preferred to store a single history variable that updates at each time step to track the loading history of the material. The following subsections present computationally efficient recursive algorithms that compute a viscoelastic history variable and continuously updates it at every time step without having to store state variables from all previous time steps [15]. In Section 2.5.1, this approach is combined with an implicit Newmark-Beta integration scheme [34] for second-order accurate time integration that is unconditionally stable. Section 2.5.2 presents the adaptation as applied to the explicit central difference method which is also second-order accurate but conditionally stable.

### *2.5.1. Implicit Newmark-Beta*

The Newmark-Beta method assumes that the displacement and velocity are approximated by the weighted average of the approximate acceleration,

$$\mathbf{q}_{n+1} = \mathbf{q}_n + \Delta t\dot{\mathbf{q}}_n + \frac{\Delta t^2}{2}[(1-2\beta)\ddot{\mathbf{q}}_n + 2\beta\ddot{\mathbf{q}}_{n+1}] \tag{13}$$

$$\dot{\mathbf{q}}_{n+1} = \dot{\mathbf{q}}_n + \Delta t[(1-\gamma)\ddot{\mathbf{q}}_n + \gamma\ddot{\mathbf{q}}_{n+1}] \tag{14}$$

The subscript $n$ denotes the current time step with the convention that $\mathbf{q}_n = \mathbf{q}(t_n)$, and step $n$+1 is the new time step to be solved. The algorithm assumes a constant time step throughout the integration period such that $\Delta t = t_{n+1} - t_n \ \forall n$. The numerical factors, $\gamma$ and $\beta$, control the numerical damping, stability and accuracy of the implicit scheme.

Since Newmark-Beta is an implicit method, the ROM equations in Eq. (10) are enforced at the $t_{n+1}$ time step,

$$\bar{\mathbf{M}}\ddot{\mathbf{q}}_{n+1} + \bar{\mathbf{C}}\dot{\mathbf{q}}_{n+1} + \bar{\mathbf{K}}_K \int_0^{t_{n+1}} \zeta_K(t_{n+1}-\tau)\dot{\mathbf{q}}(\tau)d\tau + \bar{\mathbf{K}}_G \int_0^{t_{n+1}} \zeta_G(t_{n+1}-\tau)\dot{\mathbf{q}}(\tau)d\tau + \bar{\mathbf{K}}_\infty \mathbf{q}_{n+1} = \bar{\mathbf{f}}(t_{n+1}) \tag{15}$$

The expressions in Eqns. (13) and (14) could be directly substituted into Eq. (15) to express the algebraic equations in terms of $\ddot{\mathbf{q}}_{n+1}$, however the kinematic approximations are rewritten in a way to depend on velocity, $\dot{\mathbf{q}}_{n+1}$, as,

$$\mathbf{q}_{n+1} = \mathbf{q}_n + \frac{\beta\Delta t}{\gamma}\dot{\mathbf{q}}_{n+1} + \Delta t\left(1-\frac{\beta}{\gamma}\right)\dot{\mathbf{q}}_n + \frac{\Delta t^2}{2}\left(1-\frac{2\beta}{\gamma}\right)\ddot{\mathbf{q}}_n \tag{16}$$

$$\ddot{\mathbf{q}}_{n+1} = \frac{1}{\gamma\Delta t}(\dot{\mathbf{q}}_{n+1} - \dot{\mathbf{q}}_n) - \left(\frac{1}{\gamma}-1\right)\ddot{\mathbf{q}}_n \tag{17}$$

The kinematics are written in this form to better accommodate the convolution integral which explicitly depends on velocity. Given the approximation of displacement and acceleration in Eqns. (16) and (17), these equations are substituted into Eq. (15) to discretize the equations-of-motion in time,

$$\begin{aligned}\left[\frac{1}{\gamma\Delta t}\bar{\mathbf{M}} + \bar{\mathbf{C}} + \frac{\beta\Delta t}{\gamma}\bar{\mathbf{K}}_\infty\right]\dot{\mathbf{q}}_{n+1} + \bar{\mathbf{K}}_K \int_0^{t_{n+1}} \zeta_K(t_{n+1}-\tau)\dot{\mathbf{q}}(\tau)d\tau + \bar{\mathbf{K}}_G \int_0^{t_{n+1}} \zeta_G(t_{n+1}-\tau)\dot{\mathbf{q}}(\tau)d\tau = \\ \bar{\mathbf{f}}(t_{n+1}) + \bar{\mathbf{M}}\left[\frac{1}{\gamma\Delta t}\dot{\mathbf{q}}_n + \left(\frac{1}{\gamma}-1\right)\ddot{\mathbf{q}}_n\right] + \bar{\mathbf{K}}_\infty\left[-\mathbf{q}_n + \Delta t\left(\frac{\beta}{\gamma}-1\right)\dot{\mathbf{q}}_n + \frac{\Delta t^2}{2}\left(\frac{2\beta}{\gamma}-1\right)\ddot{\mathbf{q}}_n\right]\end{aligned} \tag{18}$$

The convolution integral is addressed by assuming the material is unstressed for $t < 0$, such that the bulk viscoelastic damping forces in the time-domain have the general form,

$$\bar{\mathbf{K}}_K \int_0^{t_n} \zeta_K(t_n-\tau)\dot{\mathbf{q}}(\tau)d\tau \tag{19}$$

The derivation that follows is the same for the shear viscoelastic damping forces, and therefore is withheld for brevity. A history variable is defined at time $t_n$ for each exponential term in the Prony series,

$$\mathbf{h}_{K,i}(t_n) = \int_0^{t_n} \widehat{K}_i e^{-\frac{t_n-\tau}{\tau_{K,i}}}\dot{\mathbf{q}}(\tau)d\tau \tag{20}$$

such that Eq. (19) is defined as,

$$\bar{\mathbf{K}}_K \int_0^{t_n} \zeta_K(t_n-\tau)\dot{\mathbf{q}}(\tau)d\tau = \bar{\mathbf{K}}_K \sum_{i=1}^{N_K} \mathbf{h}_{K,i}(t_n) \tag{21}$$

The history variable in Eq. (20) accounts for a single term in the kernel function, so the cumulation of exponential functions is accounted for through the summation in Eq. (21). The history variables are defined here to allow for recursive updating of the viscoelastic damping forces with each time step, thus avoiding the need to integrate the convolution integrals over the entire bounds (i.e. 0 to $t_{n+1}$) at each time step.

At time $t_{n+1}$, the viscoelastic damping force is expressed as,

$$\bar{\mathbf{K}}_K \int_0^{t_{n+1}} \zeta_K(t_{n+1}-\tau)\dot{\mathbf{q}}(\tau)d\tau = \bar{\mathbf{K}}_K \sum_{i=1}^{N_K} \mathbf{h}_{K,i}(t_{n+1}) \tag{22}$$

where the updated history variable is defined as,

$$\mathbf{h}_{K,i}(t_{n+1}) = \int_0^{t_{n+1}} \widehat{K}_i e^{-\frac{t_n+\Delta t-\tau}{\tau_{K,i}}}\dot{\mathbf{q}}(\tau)d\tau \tag{23}$$

The summation rule for definite integrals allows Eq. (23) to split into two parts,

$$\mathbf{h}_{K,i}(t_{n+1}) = \int_0^{t_n} \widehat{K}_i e^{-\frac{t_n+\Delta t-\tau}{\tau_{K,i}}} \dot{\mathbf{q}}(\tau)d\tau + \int_{t_n}^{t_{n+1}} \widehat{K}_i e^{-\frac{t_n+\Delta t-\tau}{\tau_{K,i}}} \dot{\mathbf{q}}(\tau)d\tau \tag{24}$$

From the first part of the split integral, the product law of exponentials allows the history variable at time step $t_n$ to be factored out as,

$$\int_0^{t_n} \widehat{K}_i e^{-\frac{\Delta t}{\tau_{K,i}}} e^{-\frac{t_n-\tau}{\tau_{K,i}}} \dot{\mathbf{q}}(\tau)d\tau = e^{-\frac{\Delta t}{\tau_{K,i}}} \mathbf{h}_{K,i}(t_n) \tag{25}$$

The second integral in Eq. (24) is evaluated by assuming a midpoint velocity representation for $\dot{\mathbf{q}}(\tau)$,

$$\dot{\mathbf{q}}(\tau) = \frac{\dot{\mathbf{q}}(t_n)+\dot{\mathbf{q}}(t_{n+1})}{2} \tag{26}$$

Substituting this approximation into the second integral in Eq. (24) results in,

$$\widehat{K}_i e^{-\frac{\Delta t}{\tau_{K,i}}} \int_{t_n}^{t_{n+1}} e^{-\frac{t_n-\tau}{\tau_{K,i}}} \dot{\mathbf{q}}(\tau)d\tau = \frac{1}{2}\widehat{K}_i e^{-\frac{\Delta t}{\tau_{K,i}}} \left(\dot{\mathbf{q}}(t_n) + \dot{\mathbf{q}}(t_{n+1})\right) \int_{t_n}^{t_{n+1}} e^{-\frac{t_n-\tau}{\tau_{K,i}}} d\tau \tag{27}$$

Evaluating the integral term produces,

$$\int_{t_n}^{t_{n+1}} e^{-\frac{t_n-\tau}{\tau_{K,i}}} d\tau = \tau_{K,i}\left[e^{\frac{\Delta t}{\tau_{K,i}}} - 1\right] \tag{28}$$

Substituting Eqns. (25), (27) and (28) into Eq. (24) produces the history variable at $t_{n+1}$,

$$\mathbf{h}_{K,i}(t_{n+1}) = e^{-\frac{\Delta t}{\tau_{K,i}}} \mathbf{h}_{K,i}(t_n) + \frac{1}{2}\widehat{K}_i\tau_{K,i}\left(1 - e^{-\frac{\Delta t}{\tau_{K,i}}}\right)\dot{\mathbf{q}}_n + \frac{1}{2}\widehat{K}_i\tau_{K,i}(1 - e^{-\frac{\Delta t}{\tau_{K,i}}})\dot{\mathbf{q}}_{n+1} \tag{29}$$

The form of the viscoelastic damping force used within the Newmark-Beta algorithm becomes,

$$\begin{gathered}\overline{\mathbf{K}}_K \int_0^{t_{n+1}} \zeta_K(t_{n+1}-\tau)\dot{\mathbf{q}}(\tau)d\tau = \\ \overline{\mathbf{K}}_K \sum_{i=1}^{N_K}\left[e^{-\frac{\Delta t}{\tau_{K,i}}}\mathbf{h}_{K,i}(t_n) + \frac{1}{2}\widehat{K}_i\tau_{K,i}\left(1 - e^{-\frac{\Delta t}{\tau_{K,i}}}\right)\dot{\mathbf{q}}_n + \frac{1}{2}\widehat{K}_i\tau_{K,i}\left(1 - e^{-\frac{\Delta t}{\tau_{K,i}}}\right)\dot{\mathbf{q}}_{n+1}\right]\end{gathered} \tag{30}$$

By utilizing the derivation of the bulk and shear viscoelastic terms presented in Eq. (30) and combining these with Eq. (18), the final time-discretized equations-of-motion become,

$$\begin{gathered}\left[\frac{1}{\gamma\Delta t}\overline{\mathbf{M}} + \overline{\mathbf{C}} + \frac{1}{2}\overline{\mathbf{K}}_K \sum_{i=1}^{N_K}\left[\widehat{K}_i\tau_{K,i}\left(1 - e^{-\frac{\Delta t}{\tau_{K,i}}}\right)\right] + \frac{1}{2}\overline{\mathbf{K}}_G \sum_{i=1}^{N_G}\left[\widehat{G}_i\tau_{G,i}\left(1 - e^{-\frac{\Delta t}{\tau_{G,i}}}\right)\right] + \frac{\beta\Delta t}{\gamma}\overline{\mathbf{K}}_\infty\right]\dot{\mathbf{q}}_{n+1} = \\ \bar{\mathbf{f}}(t_{n+1}) + \overline{\mathbf{M}}\left[\frac{1}{\gamma\Delta t}\dot{\mathbf{q}}_n + \left(\frac{1}{\gamma} - 1\right)\ddot{\mathbf{q}}_n\right] + \overline{\mathbf{K}}_\infty\left[-\mathbf{q}_n + \Delta t\left(\frac{\beta}{\gamma} - 1\right)\dot{\mathbf{q}}_n + \frac{\Delta t^2}{2}\left(\frac{2\beta}{\gamma} - 1\right)\ddot{\mathbf{q}}_n\right] - \\ \overline{\mathbf{K}}_K \sum_{i=1}^{N_K}\left[e^{-\frac{\Delta t}{\tau_{K,i}}}\mathbf{h}_{K,i}(t_n) + \frac{1}{2}\widehat{K}_i\tau_{K,i}\left(1 - e^{-\frac{\Delta t}{\tau_{K,i}}}\right)\dot{\mathbf{q}}_n\right] - \overline{\mathbf{K}}_G \sum_{i=1}^{N_G}\left[e^{-\frac{\Delta t}{\tau_{G,i}}}\mathbf{h}_{G,i}(t_n) + \frac{1}{2}\widehat{G}_i\tau_{G,i}\left(1 - e^{-\frac{\Delta t}{\tau_{G,i}}}\right)\dot{\mathbf{q}}_n\right]\end{gathered} \tag{31}$$

The integration scheme solves for the new velocity, $\dot{\mathbf{q}}_{n+1}$, by directly solving the algebraic equations in Eq. (31), and then updating the displacements, accelerations, and history variables using Eqns. (16), (17) and (29), respectively.

### *2.5.2. Explicit Central Difference*

The explicit integration procedure begins by dividing the time interval of interest $[0, T]$ into subintervals,

$$[0, T] = \bigcup_{n=0}^{N_t}[t_n, t_{n+1}] \tag{32}$$

and defining a constant time increment as $\Delta t = t_{n+1} - t_n \ \forall n$. Here, $T$ is the termination time of the analysis, $t_n$ is the time at the beginning of the step, $t_{n+1}$ is the time at the end of the step, and $N_t$ is the number of time steps.

The explicit central difference method is derived from the three-parameter Hilber-Hughes-Taylor (HHT) method [35] by setting the parameters $\alpha = 0$, $\beta = 0$, and $\gamma = {}^1/_2$. The method is second-order accurate and has conditional stability of the time step increments. The general form of the integration scheme adapted to Eq. (10) becomes,

$$\ddot{\mathbf{q}}_{n+1} = -\overline{\mathbf{M}}^{-1}[\overline{\mathbf{C}}\dot{\mathbf{q}}_{n+1} + \overline{\mathbf{K}}_K \int_0^{t_{n+1}} \zeta_K(t_{n+1} - \tau)\dot{\mathbf{q}}(\tau)d\tau + \cdots \\ \ldots \overline{\mathbf{K}}_G \int_0^{t_{n+1}} \zeta_G(t_{n+1} - \tau)\dot{\mathbf{q}}(\tau)d\tau + \overline{\mathbf{K}}_\infty \mathbf{q}_n - \bar{\mathbf{f}}(t)] \tag{33}$$

$$\mathbf{q}_{n+1} = \mathbf{q}_n + \Delta t\dot{\mathbf{q}}_n + \frac{\Delta t^2}{2}\ddot{\mathbf{q}}_n \tag{34}$$

$$\dot{\mathbf{q}}_{n+1} = \dot{\mathbf{q}}_n + \frac{\Delta t^2}{2}(\ddot{\mathbf{q}}_n + \ddot{\mathbf{q}}_{n+1}) \tag{35}$$

The integration scheme in Eqns. (33) - (35) is driven by displacement; that is, the displacement at $t_{n+1}$ is computed first from the displacement, velocity, and acceleration from the preceding time step $t_n$, as shown in Eq. (34).

This displacement-driven algorithm is ideal for numerical implementation of time-domain viscoelastic integration schemes. Following the derivation used in Section 2.5.1 for the implicit Newmark-Beta method, the structure is assumed to be unstressed for $t < 0$. The viscoelastic forces in the time domain can similarly be cast in the same form as Eq. (19), and the same derivation follows up through Eq. (25). Instead of utilizing a midpoint velocity representation for $\dot{\mathbf{q}}(\tau)$ for the central difference scheme, the second mean-value theorem for integrals is used to evaluate the second term in Eq. (24) as,

$$\int_{t_n}^{t_{n+1}} \widehat{K}_i e^{-\frac{t_n+\Delta t-\tau}{\tau_{K,i}}} \dot{\mathbf{q}}(\tau)d\tau = \frac{d\mathbf{q}(k)}{d\tau}\int_{t_n}^{t_{n+1}} \widehat{K}_i e^{-\frac{t_n+\Delta t-\tau}{\tau_{K,i}}} d\tau \tag{36}$$

where $k \in [t_n, t_{n+1}]$. Using the central difference rule (which satisfies the bound restriction on $k$), Eq. (36) is recast as,

$$\frac{d\mathbf{q}(k)}{d\tau}\int_{t_n}^{t_{n+1}} \widehat{K}_i e^{-\frac{t_n+\Delta t-\tau}{\tau_{K,i}}} d\tau \approx \frac{\tau_{K,i}\widehat{K}_i\left(1-e^{-\frac{\Delta t}{\tau_{K,i}}}\right)}{\Delta t}(\mathbf{q}_{n+1} - \mathbf{q}_n) \tag{37}$$

Combining Eqns. (25), (36), and (37), Eq. (24) may be expressed in a form that is explicitly dependent upon displacements rather than velocities,

$$\mathbf{h}_{K,i}(t_{n+1}) = e^{-\frac{\Delta t}{\tau_{K,i}}}\mathbf{h}_{K,i}(t_n) + \frac{\tau_{K,i}\widehat{K}_i\left(1-e^{-\frac{\Delta t}{\tau_{K,i}}}\right)}{\Delta t}(\mathbf{q}_{n+1} - \mathbf{q}_n) \tag{38}$$

The final, displacement-driven viscoelastic force used within Eq. (33) becomes,

$$\bar{\mathbf{K}}_K \int_0^{t_{n+1}} \zeta_K(t_{n+1}-\tau)\dot{\mathbf{q}}(\tau)d\tau = \bar{\mathbf{K}}_K \sum_{i=1}^{N_k} \left[ e^{-\frac{\Delta t}{\tau_{K,i}}} \mathbf{h}_{K,i}(t_n) + \frac{\tau_{K,i}\widehat{K}_i\left(1-e^{-\frac{\Delta t}{\tau_{K,i}}}\right)}{\Delta t}(\mathbf{q}_{n+1}-\mathbf{q}_n) \right] \tag{39}$$

It is important to note that Eq. (39) only requires storage of $\mathbf{h}_{K,i}(t_n)$ and $\mathbf{q}_n$ from the preceding time step to compute the updated force at $t_{n+1}$. The same derivation is used to evaluate the viscoelastic shear forces.

# 3. EXTRACTING SYSTEM MATRICES FROM SIERRA/SD

This section explains how to extract the system matrices assembled within the Sierra/SD finite element code. These matrices are required to perform the model order reduction within a numerical computation package, such as MATLAB. The code enhancements discussed here are generic and can be used to support other research or analysis activities that necessitate the extraction of system matrices, mode shapes, etc.. on parallel processors. Section 3.1 provides specifics about extracting the matrices of a linear viscoelastic finite element model. Section 3.2 details the extraction of matrices or displacements when computing on multiple processors.

## 3.1. Viscoelastic Matrices from Elastic Matrix Assembly

In Sierra/SD [6], the constitutive model for an isotropic linear viscoelastic material uses a normalized Prony series to describe the time-dependent decay from the glassy moduli to the rubbery moduli. Following the theoretical development of the finite element formulation in the theory manual, the element stiffness matrices are cast as,

$$\mathbf{k}_K = \left(K_g - K_\infty\right) \int \mathbf{B}^{\mathrm{T}} \mathbf{D}_K \mathbf{B} dV \tag{40}$$

$$\mathbf{k}_G = \left(G_g - G_\infty\right) \int \mathbf{B}^{\mathrm{T}} \mathbf{D}_G \mathbf{B} dV \tag{41}$$

$$\mathbf{k}_e = K_\infty \int \mathbf{B}^{\mathrm{T}} \mathbf{D}_K \mathbf{B} dV + G_\infty \int \mathbf{B}^{\mathrm{T}} \mathbf{D}_G \mathbf{B} dV \tag{42}$$

The matrix **B** is the strain-displacement matrix that depends on the element shape function, while the scalar parameters $K_\infty$, $K_g$, $G_\infty$ and $G_g$ represent the rubbery (subscript $\infty$) and glassy (subscript $g$) bulk and shear moduli. Both $\mathbf{D}_K$ and $\mathbf{D}_G$ are the constitutive matrices for the bulk and shear terms, respectively. These element stiffness matrices (along with the element mass matrix) are then assembled using standard finite element techniques, resulting in Eq. (1) in Section 2.1.

The system level matrices (**M**, $\mathbf{K}_K$, $\mathbf{K}_G$, and $\mathbf{K}_\infty$) can be directly assembled and output from Sierra/SD by writing out the matrices of an *isotropic linear elastic* FEA model. The mass and stiffness matrices for the a-set DOF are written to .m files when using the **dump** solution type in the input deck. The mass matrix extraction is straightforward since it only depends on the density; however, extracting the individual stiffness matrices is more complicated. A method for extracting the system-level stiffness matrices using the **dump** solution type is given in Table 2. The table lists what values to set for the elastic bulk and shear moduli when assembling the particular stiffness matrix. Note that for the viscoelastic matrices, all other material blocks should be set to zero such that the matrix only accounts for the contribution from the viscoelastic material block of interest.

**Table 2. Linear elastic material parameters to output system-level stiffness matrices using the** dump **solution type.**

| Output Matrix in Eq. (1) | Input Bulk Moduli | Input Shear Moduli |
|---|---|---|
| $\mathbf{K}_\infty$ | $K_\infty$ | $G_\infty$ |
| $\mathbf{K}_K$ | $K_g - K_\infty$ | 0 |
| $\mathbf{K}_G$ | 0 | $G_g - G_\infty$ |

An example of the input is shown below to extract the $\mathbf{K}_K$ stiffness matrix.

```
SOLUTION
case 'dump matrices'
   dump
END

FILE
  geometry_file 'plate_9by9inch.exo'
END

ECHO
  mass
END

BLOCK 1
  hex20
  material 1
END

//K_g = 9.8039e6
//K_inf = 7.0e6
//G_g = 3.7594e6
//G_inf = 2.5e6

MATERIAL 1
  Isotropic
  G= 1e-4 // essentially zero
  K= 2.8039e6 // = K_g – K_inf
  density=0.00024739
END
```

**Figure 3. Example Sierra/SD input deck to dump assembled matrices.**

### 3.2. Parallel Matrix Extraction from Sierra/SD

As a massively parallel finite elements code, Sierra/SD provides a robust way of solving large problems in a scalable way. Its workflow specifically involves dividing a mesh before dispatching the processors to simultaneously obtain results on a sub-portion of the mesh. Final results and quantities of interests are then recovered by reassembling each sub-processor results. In this way, Sierra/SD produces accurate results in a significantly reduced amount of time. However, this parallel process masks much of the internal and intermediary steps.

To solve this problem in parallel, Sierra/SD divides each component of the system into *n*-sub-components that *n*-processors will solve. This involves creating *n*-stiffness submatrices, displacement vectors, and forcing vectors. At the end of the simulation, while results such as displacement and forces can easily be comprehended, the submatrices of the stiffness matrix are not. These matrices may be prohibitively large, making the storage and use of it in further analysis intractable. As such and if requested, Sierra/SD will only output parallel results and

intermediary files for additional post-processing in MATLAB. A capability was implemented to i.) reassemble the mass and stiffness matrices for comprehensibility and ii.) provide analysts with ways to parallelize their own operations. This allows users to extract key system information (e.g. stiffness matrix) in addition to potentially further their own analysis by allowing processor-level matrix-vector operations. The following scripts are developed to support this effort, as it demonstrates the ability to reconstruct parallelized matrices and produces the mapping required for other assembly.

### 3.2.1. How to Run Code

***What does this code do? What scripts are there?***

The developed MATLAB scripts are to be used as a post-processing step and run only after Sierra/SD has completed its analysis. The Sierra/SD input deck should request intermediary files by turning on the mfile option; this produces a large number of MATLAB-supported files required by the scripts for a reassembly process. The scripts reassemble the processor submatrices first reading in processor mappings (global ids and Fetimap) and specific FE results such as the stiffness matrix, mass matrix, and displacements. The scripts then provide a mapping from the processor level to a global level; this can be thought of converting all of the parallel outputs to that of a serial one. Finally, reassembly occurs by applying the mapping to the processor-specific components. The scripts demonstrate the reassembly process by forming the global stiffness and mass matrices from a parallel run, and they are verified by comparison to the results of a serial Sierra/SD run and by checking that the reassembled results satisfy the eigenvalue problem. When using the scripts, the following functions should be loaded in the MATLAB workspace,

**Table 3. List of MATLAB scripts.**

| Function Name | Description |
|---|---|
| assembleMatrices.m | Main script (to be edited and run by user) |
| loadStiff.m | Sub-function to load processor stiffness sub-matrices |
| loadMass.m | Sub-function to load processor mass sub-matrices |
| loadDisp.m | Sub-function to load processor eigen vectors |
| loadGids.m | Sub-function to load processor global IDs |
| loadMaps.m | Sub-function to load processor FetiMap |
| parallelmap.m | Supporting function to determine processor (parallel) mapping to global (serial) mapping |
| triplet.m | Supporting function to connect FetiMap to global IDs |
| getDofPerNode.m | Supporting function to find max number of dofs per node; currently returns 8 and only supports models with 8 or less dofs per node |
| isVolumetric.m | Supporting function to check FetiMap and globalID consistency |
| getLastGlobalRow.m | Supporting function to find maximum size of global matrices |

Note that the user should only be editing assembleMatrices.m. This is the main function that calls the other functions. The remaining functions can be divided into two groups: sub-functions that load results from the Sierra/SD run into MATLAB, and supporting functions which provides the logic and algorithm needed to reassemble the matrices. The current version of the MATLAB scripts require that the Sierra/SD input deck have the "mfile" output option and the "mfile_format 3column" parameter option, which is discussed later. Any users of this capability are asked to please report any and all bugs to the authors.

***Obtaining the right version of Sierra/SD***

When using these MATLAB scripts for post-processing, it is important to note that this only supports Sierra/SD 4.49.6+. While earlier versions of Sierra/SD may produce output .m files when requested, the previous formats were often slower and not optimized for loading into MATLAB. As of 4.49.6, an enhancement was made to the "mfile_format" of the parameters section to turn the .m file outputs to a .csv output as a 3 column sparse format. This significantly speeds up run time, and as such makes the updated version of Sierra/SD a requirement. The syntax for the "mfile" and "mfile_format 3column" options are shown below.

```
OUTPUT
        Mfile
END
PARAMETERS
        Mfile_format 3column
END
```

**Figure 4. Notable part of Sierra/SD input deck.**

***Setting up the input parameters to run the scripts***

The start of the main script (`assembleMatrices.m`) sets up the problem parameters. Specifically, the parameter descriptions can be found in Table 4 below. Note that running this script requires that all the relevant Sierra/SD runs are completed and all results stored in the correct directories.

**Table 4. List of AssembleMatrices script parameters.**

| Variable name | Argument Type | Example | Description |
|---|---|---|---|
| Input_name | String | capacitor_model | .inp file name |
| Model_name | String | Capacitor_with_ns | .exo file name |
| Num_proc | Int | 60 | Number of processors used |
| Num_modes | Int | 10 | Number of modes requested in .inp |
| Serial_files_dir | String | Np1/ | Relative file path to serial files (if serial/parallel results are to be compared) |
| Parallel_files_dir | String | Np60/ | Relative file path to parallel files |
| checkParallelWithSerial | Bool (0 or 1) | 0 | Boolean to check parallel results against serial results |
| path_to_matlab_funcs | String | ../../ | Relative path to supporting MATLAB functions |

For validation purposes, the "`checkParallelWithSerial`" flag may be turned on if both serial and parallel results are obtained. This option will output the 2-norm of the differences between the assembled and serial matrices. The MATLAB equation is as follows,

$$\text{Error} = \text{norm}\ (\ K\_\text{serial} - K\_\text{assembled}, 2) \tag{43}$$

***Setting up folder paths***

While the script allows for control over where the files are located, the following structure is a suggested workflow. The main script (`assembleMatrices.m`) and exodus-matlab file (exodus_filename.mat) are placed in a working directory. A subdirectory is then created to store all the results of the parallel and/or serial Sierra/SD runs. Another directory (either subdirectory or another directory located elsewhere) is formed to hold all of the supporting MATLAB sub-functions. A picture representation of a possible file storage scheme can be seen in Fig. 5. Note that the sub-functions are stored in an external directory for this case.

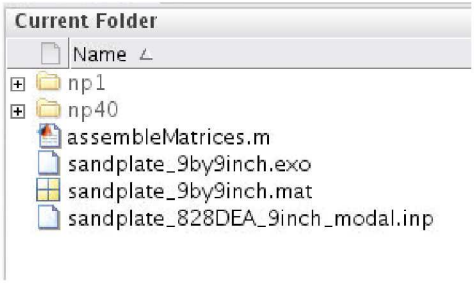



**Figure 5. Suggestion of file organization.**

***What files do I need from the Sierra/SD run?***

After turning on the "mfile" output option, there may be a variety of .m and .csv files produced, depending on the particular solution case. The following table lists the files that should always be present as they are required to reassemble the matrices.

**Table 5. List of Sierra/SD results files.**

| File | Description |
|---|---|
| Input_Deck_Name_gid_X.csv | Global ID Mapping (on processor X)* |
| FetiMap_a_X.csv | FetiMap for local node to dof mapping (on processor X)* |
| Stiff_X.m | Processor Stiffness Matrix |
| Mass_X.m | Processor Mass Matrix |
| Kssr_X.m | Processor Reduced Stiffness Matrix |
| Mssr_X.m | Processor Reduced Mass Matrix |

*these files are required for the logic behind reassembling the matrices

For an eigenanalysis, a possible subset of the outputted files is the following:

- Input_Deck_Name_ad_M_X.csv
- Input_Deck_Name_DispM_X.csv
- Input_Deck_Name_Disp_aM_X.csv

where M is the mode number and X is the processor number. It is important to note that only the reduced set (or the active set) is used in the reassembly process of the eigenvectors. This is because the files FetiMap_a and *_gid only include information on the active sets. There is no current capability to solve for the inactive sets. As such, from the list above, only Input_Deck_name_Disp_**a**M_X.csv is used in the reassembly.

***Interpreting results and verification***

The results of the script have been verified in two ways: i.) by comparing the serial results with the assembled results and ii.) verifying the mass normalized mode orthogonalize the mass matrix to unity (i.e. $\mathbf{\Phi}^{\mathrm{T}}\mathbf{M}\mathbf{\Phi} = \mathbf{I}$). A sample results output is shown below. Note how i.) the norms of the differences between the assembled and serial matrices are close to 0 and ii.) the identity matrix is a product $\mathbf{\Phi}^{\mathrm{T}}\mathbf{M}\mathbf{\Phi}$.

```
======================== RESULTS: sandplate_828DEA_9inch_modal.inp ========================
Num Proc Used:   40,      Number of Modes:    10
Total Run Time: 282.894834s

========== Stiffness Matrix
Serial K Norm:   2.3635e+08,    Assembled K Norm:   2.3635e+08,    Diff Norm: 9.7578e-15
Time for loading parallel K: 92.756184s

========== Mass Matrix
Serial M Norm:   1.3676e-05,    Assembled M Norm:   1.3676e-05,    Diff Norm: 7.5056e-15
Time for loading parallel M: 31.131017s

========== Phi Matrix
Serial Phi Norm: 8.9692e+02,    Assembled Phi Norm  8.9696e+02,    Diff Norm: 1.9998e+00
Time for loading parallel Phi: 40.381065s

========== Misc Timings
Time for loading serial data: 107.328888s
Time for total assembly process: 0.666877s
Time for calculating resulting values: 1.361956s

========== Matrix Operation Check
Multiplying Phi^T M Phi... (using assembled matrices):
    1.0000   -0.0000   -0.0000   -0.0000   -0.0000    0.0000   -0.0000   -0.0000    0.0000    0.0000
   -0.0000    1.0000    0.0000   -0.0000    0.0000    0.0000    0.0000   -0.0000   -0.0000    0.0000
    0.0000    0.0000    1.0000    0.0000   -0.0000    0.0000   -0.0000   -0.0000    0.0000    0.0000
   -0.0000   -0.0000    0.0000    1.0000    0.0000   -0.0000   -0.0000   -0.0000   -0.0000   -0.0000
   -0.0000    0.0000   -0.0000    0.0000    1.0000    0.0000   -0.0000   -0.0000    0.0000   -0.0000
    0.0000    0.0000    0.0000   -0.0000    0.0000    1.0000   -0.0000    0.0000    0.0000   -0.0000
   -0.0000    0.0000   -0.0000   -0.0000   -0.0000   -0.0000    1.0000    0.0000   -0.0000   -0.0000
   -0.0000   -0.0000   -0.0000   -0.0000   -0.0000    0.0000    0.0000    1.0000   -0.0000   -0.0000
    0.0000   -0.0000    0.0000    0.0000    0.0000    0.0000    0.0000   -0.0000    1.0000   -0.0000
    0.0000    0.0000    0.0000   -0.0000   -0.0000   -0.0000   -0.0000   -0.0000   -0.0000    1.0000



======================== END OF PROGRAM ========================
```

**Figure 6. Example results output from matrix reassembly verification.**

***How to use this script for other assembly processes***

The main workhorse of the script occurs in `parallelmap.m`. This sub-function produces "`globalRow`," which is how one can map the processor outputs to a single global output. For other potential parallel matrix operations, those results should be able to be assembled by using `globalRow`. The variable `globalRow` is an n-cell variable, where n is the number of processors and each cell is a vector of global DOF numbers. For example, `globalRow{2}` will produce the following vector,

$$\texttt{globalRow}\{2\} = [37\ 38\ 39\ 40\ 41\ 42\ \ldots\ 84] \tag{44}$$

That vector is the ordering for the submatrix from processor 2 (e.g. Kssr_1.m) to the global Kssr matrix. Row 1 of Kssr_1.m is assembled into row 37 of the global Kssr matrix, and row 2 of Kssr_1.m is put into row 38 of the global Kssr matrix, and so on.

### *3.2.2. Assembly Process*

The process of reassembling the matrices revolves around using the global ID maps (gids) and the FetiMap (FetiMap_a). These two files are generated per processor, and they indicate the nodes and node ordering on each specific processor. Additionally, this process requires the

(global) node numbering map. This map is generated with the complete exodus file, and it can be found by running exo2mat on the input .exo file and examining the resulting .mat file in MATLAB. Note that this works only for the active set DOF because Sierra/SD makes only that node numbering available (e.g. fetimap_a). While other full results may be obtained (e.g. Stiff_X, Mass_X), the assembly script only works on the reduced set (Kssr_X, Mssr_X).

The bulk of the logic in recovering the matrices comes from the `parallelmap` function. This returns the `localMap` and `globalRow`. The `localMap` variable supports the logic by forming a processor-specific mapping of "processor global dof number", "node number", and "processor local dof number." The `localMap` is then used in conjunction with the (global) `node_num_map` to identify the (global) active DOF as well as generate a vector of numbers corresponding to row indices of the global assembled matrix (`nodeBegVec`). These data (`node_num_map`, active DOF, and `nodeBegVec`) can then be used to map from the processor nodes to the corresponding global matrix row order (`globalRow`). The variable `globalRow` is the (per-processor) ordering of the parallel outputs.

For example, for a 3-processor parallel Sierra/SD run producing submatrices Stiff_0, Stiff_1, and Stiff_2, the reassembled matrix (K_global) can be found by the following operations:

$$\begin{aligned} &\text{K\_global(globalRow\{1\}(, globalRow\{1\}) += Stiff\_0;} \\ &\text{K\_global(globalRow\{2\}, globalRow\{2\}) += Stiff\_1;} \\ &\text{K\_global(globalRow\{3\}, globalRow\{3\}) += Stiff\_2;} \end{aligned} \tag{45}$$

# 4. CONCLUSION

This SAND report summarizes the project outcomes of a three year, early-career LDRD project tasked with developing model order reduction approaches to efficiently and accurately solve structural dynamics models with linear viscoelastic material behavior. Three external publications have been produced [16-18] to provide the technical details of the approaches for both monolithic and substructure finite element models. The sections of this report provide an overview of the structural dynamics modeling in Sierra/SD that was utilized for the development of reduced order models. The Prony series representation of the constitutive model provides a tractable model for numerical implementation that is flexible enough to fit to experimental master curves. A new code capability was added to Sierra/SD to allow for the mass and stiffness matrices output to multiple processors to be reassembled for use in numerical computing codes such as MATLAB. This enhancement allows for the reduced order modeling approach to be scaled to large order models that can benefit most from model reduction schemes.

Future research in this area could extend the two-tier reduction approach to substructuring techniques. In addition, further development is needed to populate a database of Prony series terms of common viscoelastic materials used in engineering sciences. It is important that proper testing be performed to obtain the time-dependent material behavior. An exemplar problem demonstrating the ability of a ROM to accurately predict damping of a component model would help motivate the adoption of this work as a predictive analysis tool to predict structural damping from material constitutive behavior.

## REFERENCES


[1] M. D. Rao, "Recent applications of viscoelastic damping for noise control in automobiles and commercial airplanes," *Journal of Sound and Vibration,* vol. 262, no. 3, pp. 457-474, 2003.

[2] R. R. J. Craig and M. C. C. Bampton, "Coupling of Substructures for Dynamic Analysis," *AIAA Journal,* vol. 6, no. 7, pp. 1313-1319, 1968.

[3] W. C. Hurty, "Vibrations of structural systems by component mode synthesis," *Journal of the Engineering Mechanics Division,* vol. 86, no. 4, pp. 51-70, 1960.

[4] S. Adhikari, *Structural Dynamic Analysis with Generalized Damping Models: Analysis*. Hoboken, NJ: John Wiley & Sons, 2014.

[5] R. Lakes, *Viscoelastic Materials*. New York, NY: Cambridge University Press, 2009.

[6] Sierra Structural Dynamics Development Team, "Sierra Structural Dynamics-User's Notes," SAND2016-3046 O, Sandia National Laboratories, Albuquerque, NM, April 2016.

[7] L. Rouleau, J.-F. Deü, and A. Legay, "A comparison of model reduction techniques based on modal projection for structures with frequency-dependent damping," *Mechanical Systems and Signal Processing,* vol. 90, pp. 110-125, 2017.

[8] R. D. Cook, D. S. Malkus, M. E. Plesha, and R. J. Witt, *Concepts and Applications of Finite Element Analysis*, Fourth ed. John Wiley and Sons, 2002.

[9] R. R. J. Craig and A. J. Kurdila, *Fundamentals of Structural Dynamics*, 2nd ed. New York: John Wiley and Sons, 2006.

[10] M. Bilasse, E. M. Daya, and L. Azrar, "Linear and nonlinear vibrations analysis of viscoelastic sandwich beams," *Journal of Sound and Vibration,* vol. 329, no. 23, pp. 4950-4969, 2010.

[11] M. Bilasse and D. C. D. Oguamanam, "Forced harmonic response of sandwich plates with viscoelastic core using reduced-order model," *Composite Structures,* vol. 105, pp. 311-318, 2013.

[12] A. M. G. d. Lima, N. Bouhaddi, D. A. Rade, and M. Belonsi, "A time-domain finite element model reduction method for viscoelastic linear and nonlinear systems," *Latin American Journal of Solids and Structures,* vol. 12, pp. 1182-1201, 2015.

[13] A. M. G. de Lima, A. R. da Silva, D. A. Rade, and N. Bouhaddi, "Component mode synthesis combining robust enriched Ritz approach for viscoelastically damped structures," *Engineering Structures,* vol. 32, no. 5, pp. 1479-1488, 2010.

[14] R. L. Bagley and P. J. Torvik, "A generalized derivative model for an elastomer damper," *Shock and Vibration Bulletin,* vol. 49, pp. 135-143, 1979.

[15] M. I. Friswell and D. J. Inman, "Reduced-Order Models of Structures with Viscoelastic Components," *AIAA Journal,* vol. 37, no. 10, pp. 1318-1325, 1999.

[16] R. J. Kuether, "Two-Tier Model Reduction of Viscoelastically Damped Finite Element Models," *Computers & Structures (in internal peer review),* 2018.

[17] R. J. Kuether, K. L. Troyer, and M. R. W. Brake, "Time Domain Model Reduction of Linear Viscoelastic Finite Element Models," presented at the ISMA2016 - International Conference on Noise and Vibration Engineering, Leuven, Belgium, Sept. 19-21, 2016.

[18] R. J. Kuether and K. L. Troyer, "Substructuring of Viscoelastic Subcomponents with Interface Reduction," Cham, 2017, pp. 17-27: Springer International Publishing.

[19] M. Bilasse, I. Charpentier, and Y. Koutsawa, "A generic approach for the solution of nonlinear residual equations. Part II: Homotopy and complex nonlinear eigenvalue

method," *Computer Methods in Applied Mechanics and Engineering,* vol. 198, no. 49, pp. 3999-4004, 2009.

[20] E. M. Daya and M. Potier-Ferry, "A numerical method for nonlinear eigenvalue problems application to vibrations of viscoelastic structures," *Computers & Structures,* vol. 79, no. 5, pp. 533-541, 2001.

[21] Y. Koutsawa, I. Charpentier, and M. Cherkaoui, "A generic approach for the solution of nonlinear residual equations. Part I: The Diamant toolbox," *Computer Methods in Applied Mechanics and Engineering,* vol. 198, no. 3, pp. 572-577, 2008.

[22] E. Balmès, "PARAMETRIC FAMILIES OF REDUCED FINITE ELEMENT MODELS. THEORY AND APPLICATIONS," *Mechanical Systems and Signal Processing,* vol. 10, no. 4, pp. 381-394, 1996.

[23] A. Ruhe, "Algorithms for the Nonlinear Eigenvalue Problem," *SIAM Journal on Numerical Analysis,* vol. 10, no. 4, pp. 674-689, 1973.

[24] C. Farhat, P. Avery, T. Chapman, and J. Cortial, "Dimensional reduction of nonlinear finite element dynamic models with finite rotations and energy-based mesh sampling and weighting for computational efficiency," *International Journal for Numerical Methods in Engineering,* vol. 98, no. 9, pp. 625-662, 2014.

[25] D. F. Golla and P. C. Hughes, "Dynamics of Viscoelastic Structures—A Time-Domain, Finite Element Formulation," *Journal of Applied Mechanics,* vol. 52, no. 4, pp. 897-906, 1985.

[26] D. J. McTavish and P. C. Hughes, "Modeling of Linear Viscoelastic Space Structures," *Journal of Vibration and Acoustics,* vol. 115, no. 1, pp. 103-110, 1993.

[27] M. A. Biot, "Variational Principles in Irreversible Thermodynamics with Application to Viscoelasticity," *Physical Review,* vol. 97, no. 6, pp. 1463-1469, 1955.

[28] R. Christensen, "Restrictions upon viscoelastic relaxation functions and complex moduli," *Transactions of the Society of Rheology,* vol. 16, no. 4, pp. 603-614, 1972.

[29] F. Schwarzl and A. Staverman, "Time-temperature dependence of linear viscoelastic behavior," *Journal of Applied Physics,* vol. 23, no. 8, pp. 838-843, 1952.

[30] J. D. Ferry, *Viscoelastic Properties of Polymers*, Second ed. New York, NY: John Wiley, 1970.

[31] R. Lin and M. Lim, "Complex eigensensitivity-based characterization of structures with viscoelastic damping," *The Journal of the Acoustical Society of America,* vol. 100, no. 5, pp. 3182-3191, 1996.

[32] M. A. Trindade, A. Benjeddou, and R. Ohayon, "Modeling of Frequency-Dependent Viscoelastic Materials for Active-Passive Vibration Damping," *Journal of Vibration and Acoustics,* vol. 122, no. 2, pp. 169-174, 1999.

[33] C. Vasques, R. Moreira, and J. D. Rodrigues, "Viscoelastic Damping Technologies-Part I: Modeling and Finite Element Implementation," *Journal of advanced research in Mechanical Engineering,* vol. 1, no. 2, 2010.

[34] N. M. Newmark, "A method of computation for structural dynamics," *Journal of the engineering mechanics division,* vol. 85, no. 3, pp. 67-94, 1959.

[35] H. M. Hilber, T. J. R. Hughes, and R. L. Taylor, "Improved numerical dissipation for time integration algorithms in structural dynamics," *Earthquake Engineering & Structural Dynamics,* vol. 5, no. 3, pp. 283-292, 1977.

## DISTRIBUTION

| | | | |
|---|---|---|---|
| 1 | MS0346 | Adam Brink | 1556 |
| 1 | MS0346 | Lili Heitman | 1553 |
| 1 | MS0346 | Robert Kuether | 1556 |
| 1 | MS0346 | Jonel Ortiz | 1556 |
| 1 | MS0359 | Donna Chavez | 1171 |
| 1 | MS0828 | Basil Hassan | 1510 |
| 1 | MS0840 | James Redmond | 1550 |
| 1 | MS0845 | Mark Chen | 1542 |
| 1 | MS0845 | Nate Crane | 1542 |
| 1 | MS0899 | Technical Library | 9536 (electronic copy) |

Sandia National Laboratories