Legendre Spectral Finite Elements for Reissner-Mindlin Plates

Kazh Brito

A report submitted in partial fulfillment of the requirements for the degree of

Master of Science

University of California, 2010

2010

Program Authorized to Offer Degree: Applied Mathematics

University of California, Merced
Graduate Division

This is to certify that I have examined a copy of a technical report by

Kazh Brito

and found it satisfactory in all respects, and that any and all revisions required by the examining committee have been made.

Research Advisor:

Michael Sprague

Chair of Supervisoring Committee:

Arnold Kim

Reading Committee:

Boaz Ilan

Graduate Coordinator:

Boaz Ilan

Date:

University of California, Merced

Legendre Spectral Finite Elements for Reissner-Mindlin Plates

by Kazh Brito

Project Advisor

Michael Sprague, Ph.D.

Chair of Supervisory Committee

Professor Arnold D Kim
Applied Mathematics

Abstract

This is an exploration of Legendre spectral finite-element (LSFE) formulations for Reissner-Mindlin plates. The goal was to compare high-order LSFEs with standard low-order finite elements in terms of computational efficiency, and determine an optimal formulation for thin-walled elastic media. Simulations using various LSFE and standard FE formulations were carried out. Model performance is compared by examining the error as a function of both model size (DoF) and model efficiency (FLOPs) for the various formulations. Results showed that LSFEs using a mixed formulation consisting of nodal Gauss-Lobatto-Legendre quadrature for the bending matrix, and reduced Gauss-Legendre quadrature for the shear matrix were most computationally efficient of all elements tested.

1 Introduction

The current study is an exploration of Legendre spectral finite-element (LSFE) formulations for structural mechanics, in this case Reissner-Mindlin plates. The goal was to develop a high-order spectral method that accurately represents the response of thin-walled elastic structures, with more computational efficiency than current low-order finite-element (FE) methods. To that end, a variety of element formulations are tested on both static and dynamic response of Reissner-Mindlin plates, and compared in terms of their accuracy as a function of model size and efficiency. The hope is to show that the choice of quadrature scheme is crucial in determining computational accuracy and efficiency.

LSFEs provide a number of advantages that make them an attractive choice for the modeling of elastic media. In traditional finite-element (FE) schemes, the unknown displacement field is approximated as a linear combination of piecewise, low-order interpolating polynomials (e.g. linear or quadratic). Model error is decreased by increasing the resolution of the underlying mesh (increasing the number of elements), but keeping the polynomial order constant (an overview of traditional finite element analysis can be found in [5]). This is so-called "h-refinement." In spectral finite elements, model error is reduced by increasing the polynomial order of the approximation. Thus, the number of elements remains the same, but the number of interpolating polynomials, and thus nodes, per element is increased (so-called "p-refinement").

Another advantage LSFEs have over traditional FEs is the choice of element nodes. Nodes in the element coordinate system are positioned at the Gauss-Legendre-Lobatto (GLL) quadrature points. So, for an n^{th}-order polynomial approximation of the unknown function, the shape functions are interpolated through the $(n+1)$ GLL points. This choice of element nodes has two important advantages. First, the uneven spacing of the GLL points causes a "clustering" of element nodes at element interfaces and domain boundaries. This allows for greater resolution of edge effects and boundary layers [17], without additional treatment. Furthermore, numerical GLL quadrature may be used in calculation of the various matrix and vector quantities associated with the numerical approximation. Using coincident element and quadrature node locations in so-called "nodal quadrature" allows for efficient computation of matrix-vector products, and creates a diagonal mass matrix.

Additionally, advantages specific to thin-walled elastic structures (such as Reissner-Mindlin plates) have been observed. Specific advantages include the absence of "shear-locking," [17] a phenomenon where the displacements shrink in magnitude as element thickness is decreased; and the absence of "spurious energy modes," [17] the development of zero-energy modes, artifacts of the quadrature scheme used in the formulation. These two facts make LSFEs especially attractive for modeling of elastic media.

Early work using spectral elements was aimed at fluid dynamics [10] and heat transport [13]. Early work in spectral elements used shape functions interpolated through the Gauss-Chebyshev-Lobatto points [10], but focused quickly shifted to the GLL points [10], as they represent a more "natural" quadrature choice (the GLL points are used for quadrature with respect to a unity weighting functions, the GLC points are not), and for their more efficient computations for the relatively low-order approximations used [13]. It should be noted that some recent works still use spectral finite elements with the GLC points (see, e.g., [2]).

Initial use of LSFEs for elastic media began with the 1D wave equation [16] and later 2D Reissner-Mindlin plates [17]. Additional works focused on the use of LSFEs applied to propagation of elastic waves for seismological problems [2, 3, 15]. More current works focus on the use of LSFEs in the role of damage detection in 1D beams [6], 2D plates [7, 11], and 3D elastic media [11]. Sprague and Geers (2008) studied LSFEs for the dynamic Timoshenko beam (the 1-D analogue

to the current study's Reissner-Mindlin plates), and compared their accuracy to that of standard h-type FEs in the context of model degrees of freedom and computational efficiency [14].

The primary deficiencies of recent works are a lack of attention to convergence (i.e. the convergence rates of the methods), and inexact characterization of computational efficiency (excepting [14]). The current study hopes to fill these gaps by measuring more exactly error decay as it relates to mesh resolution (degrees of freedom) and computational efficiency (number of floating point operations per simulation). Additionally, the author has found no studies that compare different types of quadrature schemes for LSFEs. Most studies employ nodal GLL quadrature [3, 6, 7, 11, 15–17], without any comparison to Gauss-Legendre quadrature schemes used in traditional finite elements. The current study will follow in the footsteps of [14] and compare LSFEs not only to traditional, h-refined FEs, but also to different quadrature formulations of LSFEs. This includes a new "mixed" formulation with combines GLL and GL quadrature in the same element formulation. Performance of element types will be characterized in terms of both convergence rates (i.e., error versus degrees of freedom), and efficiency (i.e. error versus number of arithmetic operations).

2 Legendre Spectral Finite Elements

2.1 Finite Element Formulations

In a standard finite-element (FE) formulation (see, e.g., [5]), the problem starts with the variational (or "weak") formulation of a differential equation. For example, consider the following ordinary differential equation, on a one-dimensional domain $\Omega = [a, b]$:

$$\frac{d^2u}{dx^2} = -f(x), \quad x \in \Omega, \tag{2.1}$$

$$u(a) = u_{\mathrm{D}}, \tag{2.2}$$

$$u'(b) = u_{\mathrm{N}}, \tag{2.3}$$

for some forcing function $f(x)$ and boundary values $u_{\mathrm{D}}, u_{\mathrm{N}} \in \Re$. This has corresponding weak formulation

$$\int_\Omega \frac{dw}{dx}\frac{du}{dx}\,dx - u_{\mathrm{N}}w(b) = \int_\Omega w(x)f(x)\,dx. \tag{2.4}$$

Equation (2.4) must hold for all "weight functions" $w(x)$ in some admissible function space [5]. In addition, the weight functions must satisfy certain conditions on the boundary, in this case $w(a) = 0$, and $w(b) \neq 0$. The FE approximation is then obtained by approximating the unknown function $u(x)$ and the set of weight functions $w(x)$ as a linear combination of "shape functions," i.e.

$$u(x) = \sum_{i=1}^{n} d_i\phi_i(x), \tag{2.5}$$

$$w(x) = \sum_{i=1}^{n} c_i\phi_i(x), \tag{2.6}$$

for some coefficients $d_i, c_i \in \Re$, where $\phi_i(x)$ would typically be a set of piecewise polynomials, interpolated through a discrete selection of points, which would define the finite-element mesh. By substituting (2.5) and (2.6) into (2.4), we obtain $u(x)$:

$$\mathrm{K}\mathbf{d} = \mathbf{F}, \tag{2.7}$$

where

$$\mathrm{K}_{ij} = \int_\Omega \frac{d\phi_i}{dx}\frac{d\phi_j}{dx}\,dx, \tag{2.8}$$

$$F_i = \int_\Omega \phi_i(x) f(x)\,dx. \tag{2.9}$$

We can then solve (2.7) by any number of methods. K is called the stiffness matrix of the problem, due to the similarity of (2.7) to Hooke's law. In this case, the stiffness matrix is symmetric positive semi-definite [5].

2.2 Element Discretization and Meshes

The domain Ω is discretized into a discretized set of N subdomains Ω_i such that

$$\bigcup_{i=1}^{N} \Omega_i = \Omega. \tag{2.10}$$

The shape functions are then defined as compactly supported, piecewise polynomials:

$$\phi_i(x) = \begin{cases} p(x), & \text{for} \quad x \in \Omega_i, \\ 0, & \text{for} \ \ x \notin \Omega_i, \end{cases} \tag{2.11}$$

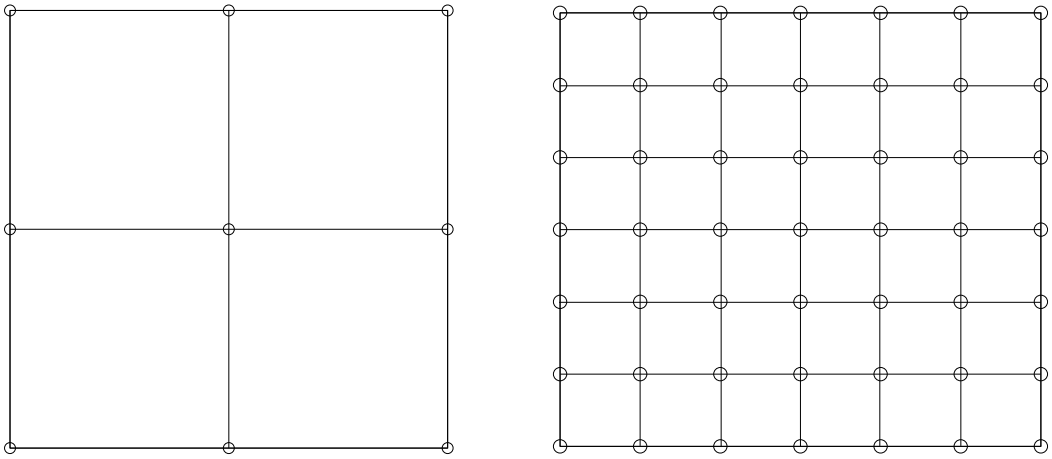
defined so as to be at least C^0 continuous across element boundaries. The functions $p(x)$ would typically be defined as linear (or other low-order) polynomials, interpolated through the nodes denoting the element domains. Thus, the integral inner products from (2.8) and (2.9) would have only non-zero values in their corresponding element. Calculation of the stiffness matrix and forcing vector are eased by uniformity of calculations from element to element. In addition, if the shape functions are Lagrangian interpolants through the mesh nodes, the displacement coefficients d_i found by solving 2.7 would be exactly the function values at the element nodes.

With the above formulation, an increase in accuracy of the approximation could be obtained by increasing the number of subdomains ("h-refinement"). A typical two-dimensional mesh is displayed in Fig. 1, as well as a corresponding refined mesh with an increased number of elements. The approximation of using this approach is governed "algebraic" convergence, i.e. error is proportional to $(1/N)^{p+1}$, where N is the number of elements in the mesh, and p is the polynomial order of the shape functions.

2.3 p-refinement

In Fig. 1, each element consists of 2×2 nodes, corresponding to linear shape functions. The polynomial order of the shape functions will determine the number of nodes per element, with N_e^{th}-order shape functions corresponding to $(N_e + 1) \times (N_e + 1)$ elements per node. In a p-refined FE formulation, the mesh would not be refined by increasing the number of elements, as shown above. Instead, each step of refinement would introduce an increase in the polynomial order, and thus the number of elements remains fixed, and the number of nodes per element is increased.

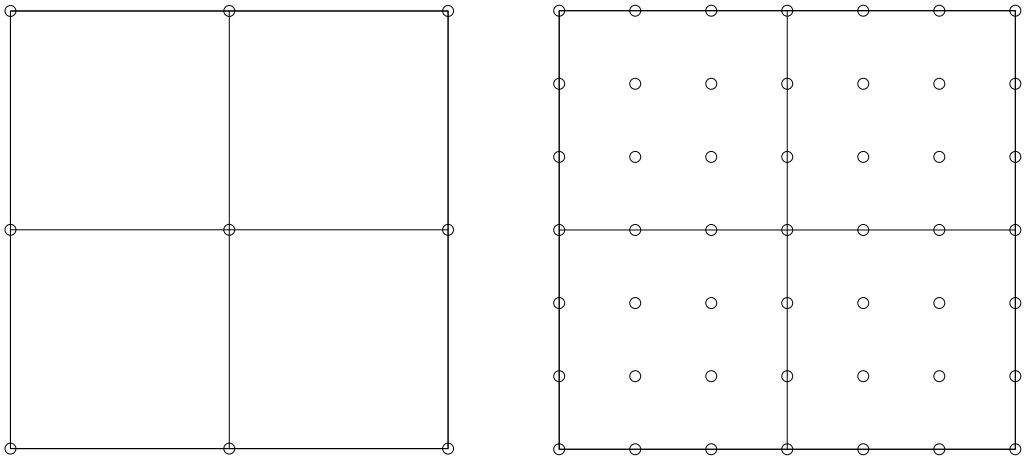
Notice that the refined meshes in Figs. 1 and 2 have the same number of nodes for the global mesh, but the problem formulations are fundamentally different: one has 36 elements, with linear shape functions, while the other has only 4 elements, with cubic shape functions.



(a) 2×2 element mesh

(b) 6×6 element mesh

Figure 1: Example meshes for two-dimensional FE discretizations with bilinear elements.



(a) 2×2 element mesh, with 1^{st}-order shape functions.

(b) 6×6 element mesh, with 3^{rd}-order shape functions.

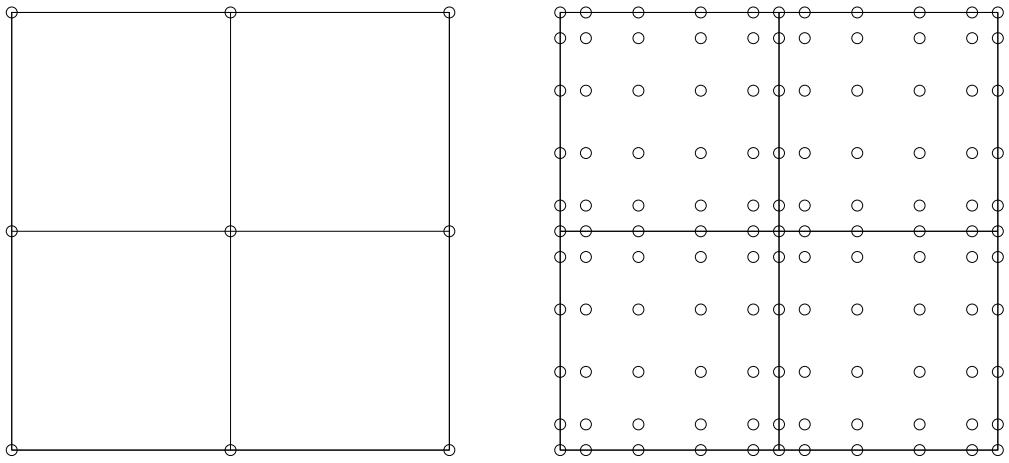
Figure 2: p-refinement for two-dimensional finite element discretizations.

2.4 Legendre-Spectral Finite Elements

In the above sections, two different mesh refinements were discussed: h-refinement, and p-refinement. In Legendre Spectral FEs (LSFEs), the nodes are spaced, in the element coordinate system, at the Gauss-Legendre-Lobatto points, quadrature points defined to the $(n+1)$ solutions to the equation

$$0 = (1 - x^2)L'_n(x), \tag{2.12}$$

where $L_n(x)$ is the n^{th}-order Legendre polynomial. This leads to an irregular nodal spacing with a "clustering" of nodes near element boundaries. It is these types of elements which will be investigated and compared to traditional, h-refined finite elements with equally-spaced nodes.



(a) 2×2 element mesh, with 1^{st}-order shape functions.

(b) 2×2 element mesh, with 5^{th}-order shape functions. Nodes are spaced at the GLL points.

Figure 3: Spectral LSFE refinement for two-dimensional finite-element discretizations.

2.5 Numerical Quadrature

To solve the linear system (2.7), the stiffness matrix K and forcing vector $\mathbf{F}$ must be computed from the inner-products in (2.8) and (2.9), which is typically done with Gaussian quadrature. Since the shape functions will usually be polynomials, proper choice of quadrature can make for exact computation of the stiffness matrix (for rectangular elements). For example, in (2.8), the integral is a product of two derivatives. If the original shape functions were quadratic, then the resulting integral would be a the product of two linear functions, i.e. a quadratic itself. n-point Gauss-Legendre (GL) quadrature is exact for integration of polynomials of order $2n-1$ or less, so 2-point GL quadrature would compute that integral exactly. The quadrature rules investigated here are the standard Gauss-Legendre (GL), and Gauss-Legendre-Lobatto (GLL). n-point GLL quadrature is exact for integrals of polynomials of order $2n-3$ or less.

2.6 Element-Level Calculations

As mentioned, one advantage of finite elements is the uniformity of calculations across elements. To calculate the linear system in, for example, (2.7), the inner products are generally converted to "element-level" coordinates, where they can be calculated in the same manner, and converted back to the global coordinate system. Thus the only difference between calculations in each element is the Jacobian of the transformation. The global stiffness matrix and forcing vector are then assembled based on element connectivity. The resultant linear system is then solved. For most elements, and particularly those used here, the element coordinate domain is set, in two-dimensions, as $\xi, \eta \in [-1,1] \times [-1,1]$, for element coordinates ξ and η, to utilize the aforementioned Gaussian quadrature schemes. Details for the particular problem mentioned below can be found in the appendices.

3 Reissner-Mindlin Plates

One popular model for thin, elastic plates, is the Reissner-Mindlin plate model. A detailed overview of the assumptions and formulation can be found in [5]. The solution obtained (for $(x, y) \in \Omega \subset \Re^2$) is of the form $\mathbf{u}(x, y) = (w(x, y), \theta_x(x, y), \theta_y(x, y))^{\mathrm{T}}$, where w is the transverse displacement, θ_x and θ_y are rotations about the y- and x-axes, respectively.

3.1 Static Formulation

The weak formulation for $\mathbf{u}(x, y)$ is given by

$$\int_\Omega \left[\bar{\theta}_{i,j} c_{ijkl} \theta_{k,l} + \bar{\gamma}_i c_{ij} \gamma_j\right] \, d\Omega + \int_\Omega \left[\bar{\theta}_i C_i - \bar{w} F\right] \, d\Omega + \int_{\partial\Omega_n} \left[\bar{\theta}_i M_i - \bar{w} Q\right] \, ds = 0, \tag{3.1}$$

where bars denote weight functions from the admissible function space, subscript indices (e.g. i, j) are 1 or 2, and define components of $\theta_i = (\theta_x, \theta_y)^T$, and $\gamma_i = (-\theta_x + w_{,x}, -\theta_y + w_{,y})$. Furthermore C_i is rotational forcing, F is transverse forcing, M_i are Neumann-type, rotational boundary conditions, and Q are transverse boundary conditions. c_ijkl and c_ij are material property tensors. To simplify the c's, an isotropic formulation is assumed, i.e.

$$c_{ijkl}\theta_{k,l} = \frac{a^3}{12}\left[\bar{\lambda}\delta_{ij}\theta_{k,k} + 2\mu\theta_{i,j}\right], \tag{3.2}$$

$$c_{ij} = a\mu\delta_{ij}, \tag{3.3}$$

where λ and μ are the Lamé parameters, and a is the thickness of the plate. Note that in practice c_{ij} is often replaced with κc_{ij}, where κ is a shear correction factor, usually $\kappa = \frac{5}{6}$. This is to ensure consistency with classical bending models. Furthermore, the Lamé parameters are often rewritten in terms of E and ν, Young's modulus, and Poisson's ratio. Transformations between these two sets of parameters can be found in [5]

When the above is discretized, an equation similar to (2.7) is obtained. The difference being that the stiffness matrix is written as a sum of a matrix consisting of contributions from bending stiffness ($\mathrm{K^B}$), and one consisting of contributions from shear stiffness ($\mathrm{K^S}$)

$$\mathrm{K}^{RM} = \mathrm{K^B} + \mathrm{K^S}, \tag{3.4}$$

where $\mathrm{K}^B \propto a^3$ and $\mathrm{K}^S \propto a$. A more detailed derivation of this equation can be found in the appendices.

3.1.1 Shear Locking and Choice of Quadrature

To solve a linear system coming from the discretization of (3.1), one may think to compute the inner products exactly. Not only may this be computationally inefficient, but the makeup of the linear system leads to a phenomenon known as "shear locking," which causes displacements to shrink (numerically) in the thin plate limit (when, for a given forcing, the opposite should be the case). A detailed overview of shear locking can be found in [12]. In practice, shear-locking can be avoided with careful choice of quadrature rule. Reducing the quadrature, simply using an inexact quadrature rule, can alleviate and, in some cases, eliminate locking completely, by introducing a singularity in the shear matrix, while maintaining global invertibility. This is done, in general, by applying reduced (i.e. inexact) quadrature to the components of the shear matrix, while maintaining full

(i.e. exact) quadrature for the components of the bending matrix, a formulation known as "selective reduced quadrature" or "selective shear quadrature."

Recall in Fig. 3 the spacing of the nodes at the GLL points. Besides the clumping at element boundaries/interfaces, the GLL points have the advantage of being set at points corresponding to a Gaussian quadrature scheme. Recall that the shape functions are formulated as Lagrangian interpolants. In other words

$$\phi_i(\xi_j) = \delta_{ij}, \tag{3.5}$$

where $\phi_i(\xi)$ is an element-level shape function, and ξ_j is an element node (again, in element coordinates), and δ_{ij} is the Kronecker delta. This leads to savings in the number of operations needed to compute the stiffness matrix, which can be seen more specifically in the appendices. The disadvantage, as noted above, is that, for a given integral, GLL quadrature is less exact that GL quadrature. The question therefore becomes whether or not the loss in quadrature error is "worth" the gain in computational efficiency (a question the current study seeks to answer).

Table 3.1.1 lists the various types of elements and quadrature schemes compared in the following sections. LFE-SRQ and QFE-SRQ are h-refined elements, linear and quadratic, respectively, with equally-spaced nodes in both the global and element coordinate systems, and calculated with the selective shear GL quadrature mentioned above. LSFE-SRQ, LSFE-NRQ, and LSFE-MRQ all correspond to Legendre spectral finite elements, i.e. p-refined elements with nodes in the element coordinate system spaced at the GLL points. LSFE-SRQ uses the same type of selective shear GL quadrature as the h-refined elements. LSFE-NRQ uses nodel GLL quadrature for both the bending and shear matrices. LSFE-MRQ uses nodal GLL quadrature for the bending matrix, and reduced GL quadrature for the shear matrix.

Element Type	# Nodes per Element	Bending Matrix Quadrature	Shear Matrix Quadrature
LFE-SRQ	2×2	2×2 GL	1×1 GL
QFE-SRQ	3×3	3×3 GL	2×2 GL
LSFE-SRQ	$(N+1) \times (N+1)$	$(N+1) \times (N+1)$ GL	$N \times N$
LSFE-NRQ	$(N+1) \times (N+1)$	$(N+1) \times (N+1)$ GLL	$(N+1) \times (N+1)$ GLL
LSFE-MRQ	$(N+1) \times (N+1)$	$(N+1) \times (N+1)$ GLL	$N \times N$ GL

Table 1: Types of elements and associated quadrature schemes used to model Reissner-Mindlin plates.

3.2 Dynamic Formulation

Transitioning from a static to a dynamic formulation, with time dependendent displacement and forcing, merely introduces a mass term related to Newton's law:

$$\mathrm{K}\mathbf{d} + \mathrm{M}\ddot{\mathbf{d}} = \mathbf{F}, \tag{3.6}$$

where $\mathbf{d}$ is a vector of nodal approximations to the function values $\mathbf{u}(x_i, y_j)$ for node locations (x_i, y_j), and K is the same stiffness matrix used in the static formulation. The matrix M is the "mass matrix," and its integral form for this problem can be found in [17]. In addition to quadrature and spatial discretization, the above formulation must be discretized in time as well. For the results below, an explicit centered difference approximation of the second derivative in time is used, i.e.

$$\ddot{\mathbf{d}} \approx \frac{\mathbf{d}^{k+1} - 2\mathbf{d}^k + \mathbf{d}^{k-1}}{\Delta t^2}, \tag{3.7}$$

where $\mathbf{d}^k$ is the global displacement vector at time step k, and Δt is a to be determined time step (which will be discussed in detail below). If (3.7) is substituted into (3.6), it yields the following update process in time:

$$\mathbf{d}^{i+1} = \Delta t^2 \left[\mathrm{M}^{-1}\mathbf{F}^k - (\mathrm{M}^{-1}\mathrm{K})\mathbf{d}^k \right] + \left[2\mathbf{d}^k - \mathbf{d}^{k-1} \right]. \tag{3.8}$$

The update scheme (3.8) requires that a matrix inversion of M be done at each time step. In practice, however, the mass matrix is often "lumped" into a diagonal form. Several techniques for doing so exist (see e.g [5]), but the one used in the current formulation is to simply compute the mass matrix using nodal GLL quadrature. This form has been shown to diagonalize the matrix without any loss of convergence rate [4].

3.2.1 Critical Time Step

For the explicit scheme outlined above, selection of a stable time step is critical. This means calculating Δt_{crit}, the maximum time step that will allow for a stable solution. The exact form of the critical time step can be found in [9] and is found by first solving the eigenvalue problem

$$\left[\mathrm{K} - \omega^2 \mathrm{M} \right] \mathbf{u} = \left[\mathrm{K} - \lambda \mathrm{M} \right] \mathbf{u} = 0, \tag{3.9}$$

which has eigenvalues corresponding to the eigenvalues of $\mathrm{M}^{-1}\mathrm{K}$. Then the maximum stable time step can be found with

$$\Delta_t \leq \Delta t_{\text{crit}} = \frac{2}{\omega_{\max}} = \frac{2}{\sqrt{\lambda_{\max}}}, \tag{3.10}$$

where $\omega_{\max}$ is the maximum frequency that can be represented by the global system, and $\lambda_{\max}$ is the maximum eigenvalue of the global matrix $\mathrm{M}^{-1}(\mathrm{K})$. Thus, the maximum eigenvalue of that matrix must be computed or bounded to get any kind of bound on the time step. The current formulation uses the power method to approximate directly the value of $\lambda_{\max}$.

4 Numerical Results

Below are results from numerical simulations run on Reissner-Mindlin plate models with a variety of element types. Two types of comparions are made to assess the accuracy and efficiency of the various element types. First, accuracy is compared to model complexity. Accuracy is measured, in this case, by comparing the solution obtained from FE analysis to a benchmark solution and taking some measure of error (details in subsequent sections). Model complexity is measured by "degrees of freedom," i.e. the number of free variables being solved for in the linear system. At each node, the unknown function $\mathbf{u}(x, y)$ has three components: one displacement and two rotations. The number of degrees of freedom (DoF) is the total number nodal function values in the system (3×# of nodes), minus those displacements and rotations fixed by boundary conditions. Secondly, accuracy (computed in the same way as above) is compared to computational efficiency. Efficiency is measured by measuring the total number of floating point operations needed to solve the linear system or carry out the explicit time update scheme. Certain pre- and post-processing steps are ignored, and the counts are done primarily on the matrix-vector products and other linear algebra operations required to complete the simulations.

4.1 Static Analyses

Static simulations were peformed on a square, simply supported Reissner-Mindlin plate undergoing uniform forcing. Model symmetry was exploited and only a quarter of the plate was modeled, with symmetry conditions enforced at non-boundary edges.

The mathematical formulation of the symmetry and simply-supported conditions is as follows:

$$w = \theta_y = 0 \quad \text{on} \quad x = L, \tag{4.1}$$

$$w = \theta_x = 0 \quad \text{on} \quad y = L, \tag{4.2}$$

$$\theta_x = 0 \quad \text{on} \quad x = \frac{L}{2}, \tag{4.3}$$

$$\theta_y = 0 \quad \text{on} \quad y = \frac{L}{2}, \tag{4.4}$$

where L is the global side length of the plate. The first two conditions represent simply-supported boundary conditions, and the second two symmetry conditions.

Material properties for the plate were chosen to mimic those of steel, and were $E = 2.1 \times 10^{11}\ N/m^2$, and $\nu = 0.3$. Plate dimensions were chosen to give a typical length to width ratio of $L/a = 40$ (e.g., [17]): $L = 10m$ and $a = 0.25m$. A uniform transverse forcing of $1.0 \times 10^3\ N/m^2$ was applied. The linear system was solved using preconditioned conjugate-gradient with a Jacobi pre-conditioner (i.e., the pre-conditioning matrix P = diag(K)), as in [1].

For the static simulations, accuracy was computed by first calculating the difference between the function values at the nodal points given by the FE solution and those given by a benchmark solution found in [8]. This difference function is then used to calculate a normalized integral L_2 norm using GLL quadrature which is used as the error value for that simulation, i.e.

$$\begin{aligned} \text{diff} &= \sum_{i=1}^{3} \int_\Omega (u^i_{\text{FE}}(x,y) - u^i_{\text{B}}(x,y))^2 \, dA, \\ \text{norm} &= \sum_{i=1}^{3} \int_\Omega (u^i_{\text{B}})^2 \, dA, \\ \text{err} &= \frac{\text{diff}}{\text{norm}}, \end{aligned} \tag{4.5}$$

where the superscript i's represent the 3 components of the solution vector.

4.1.1 Accuracy versus Model Complexity

Figure 4 shows model accuracy versus model complexity, i.e. L_2 error versus # DoF. From Fig. 4, it is clear that the error of the p-refined elements reduces more quickly (in a DoF sense) than the h-refined. Additionally, the MRQ and SRQ formulations have the lowest global error for any number of DoF.

One of the more important results that this plot shows is the comparison between the selective-shear GL LSFEs (LSFE-SRQ) and the mixed formulation (LSFE-MRQ). The difference between them is that the mixed formulation uses inexact, nodal quadrature for the bending matrix, while the selective-shear uses exact GL quadrature (the shear quadratures are identical). With the exception of the linear elements, the errors are the same, and thus the solutions would be nearly indistinguishable. This seems to imply that the quadrature error gained from inexactly calculating the bending component of the stiffness matrix is, especially for higher-order elements, negligible.

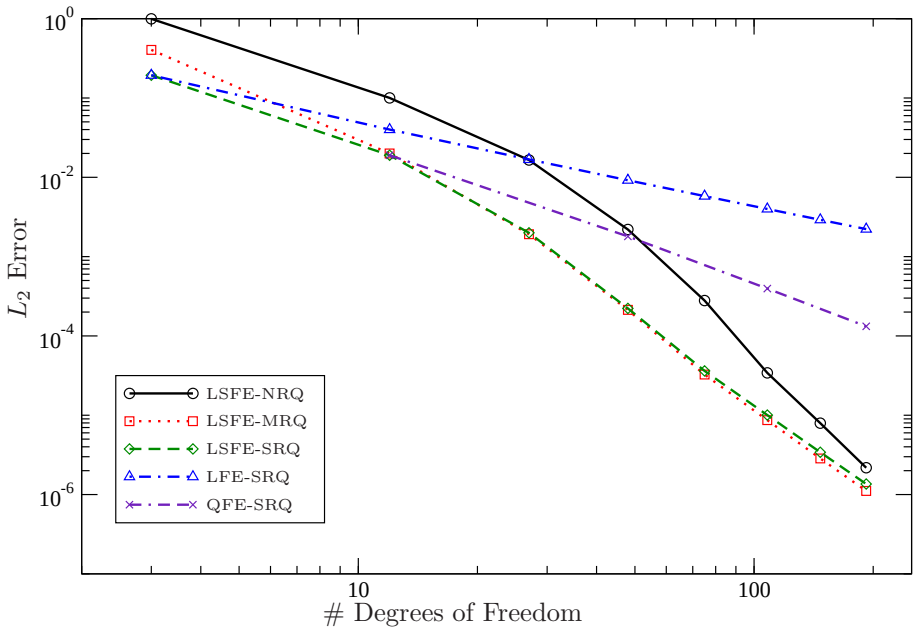


Figure 4: L_2 error vs. DoF for static-response calculations.

4.1.2 Accuracy versus Model Efficiency

Figure 5 shows model accuracy versus model efficiency, i.e. L_2 error versus # of floating point operations per simulation. From Fig. 5, it is again clear the the LSFEs are superior, in that they, in general, are more accurate for the same amount of computational effort (excepting the LSFE-NRQ). In other words, to obtain the same amount of error using any of the 5 element types above (i.e. a horizontal line on the above figure), the LSFEs would take significantly less operations, and thus less computational time. The most interesting result is that the LSFE-MRQ outperformed all other formulation types for every level of refinement.

4.2 Dynamic Analysis

Dynamic simulations were performed on a plate of the same physical parameteres and boundary conditions as above. In addition, the density of the plate was considered and set to $\rho = 8.0 \times 10^6\ g/m^3$. Initial conditions were set to a plate fully at rest at $t = 0$, and forcing was again uniform in space, but also in time, and set to $10^3\ N/m^2$. Simulations started at time $t = 0s$ and were carried out until $t = 5s$.

Error was calculated using the center displacement over time. For each simulation the center displacement was calculated at a series of time steps. The vector of center displacements is then compared to those of a benchmark solution, and a vector 2-norm is taken of the difference, and normalized by the 2-norm of the benchmark, to give a normalized error value. The benchmark itself was calculated numerically, using an 8×8 mesh of 16^{th}-order elements, with 49152 spatial DoF, i.e.refined enough to make the error well below machine roundoff, and thus, for all intents and purposes, the "real" solution. Figure 6 shows the dynamic benchmark solution.

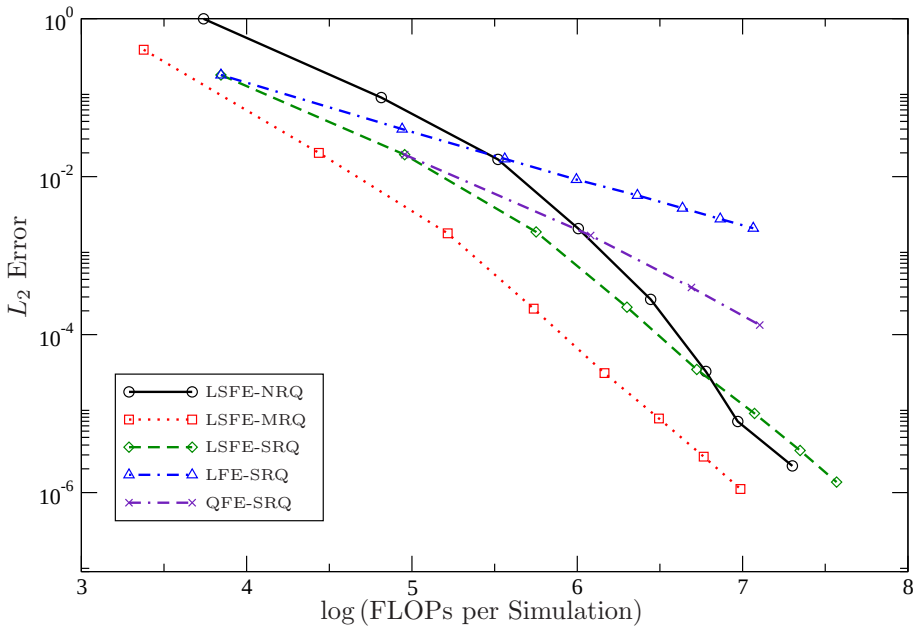


Figure 5: L_2 error vs. model efficiency for static response calculations.

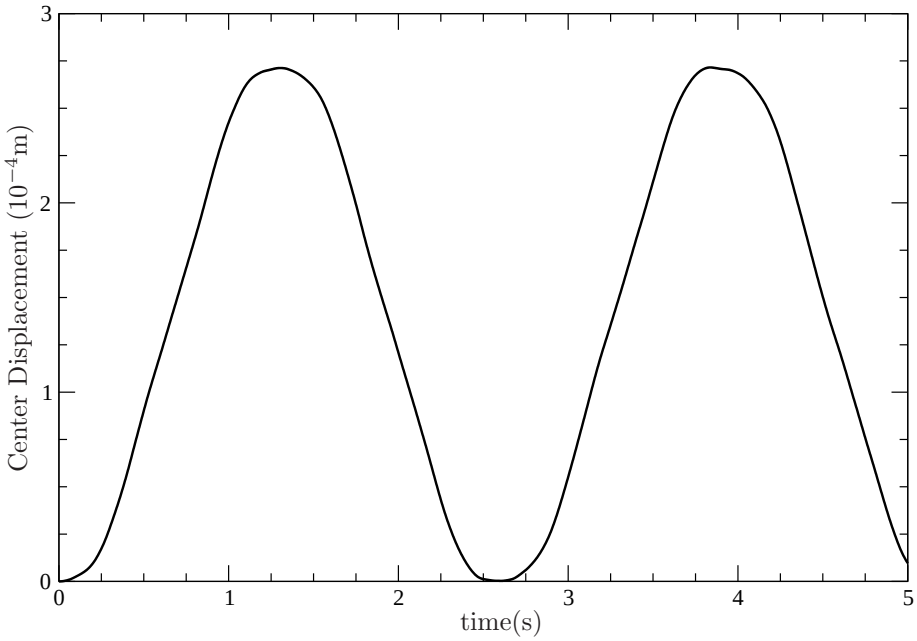


Figure 6: Dynamic benchmark solution for the center displacement of a Reissner-Mindlin plate.

4.2.1 Critical Time Steps

For the explicit scheme (3.8) the time step must be smaller than the critical time step defined in (3.10). Prior to each dynamic simulation, the critical time step was computed, and the actual time step used was $\Delta t = 0.9\Delta t_{\text{crit}}$.

Figure 7 shows the critical time step for the various element types as a function of DoF. It can

be seen above that the critical time step of h-type decays much more slowly with respect to model complexity. This is in contrast to the LSFE formulations, which have a quickly decaying time step. This implies that the number of time steps required to run a simulation over the same amount of time will be significantly higher for the LSFEs than for the standard FEs.

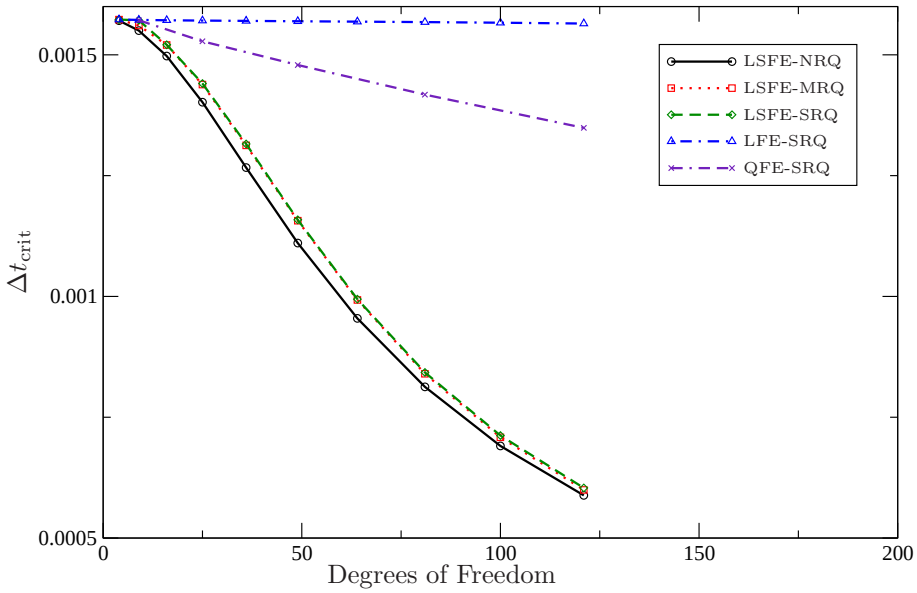


Figure 7: Critical Time Step vs. DoF for varying element and quadrature formulations.

4.2.2 Accuracy versus Model Complexity

Figure 8 shows the L_2 error of the center displacement of the plate as a function of DoF. As before, the error of the LSFEs reduces more quickly than that of the LFEs and QFEs. Specifically, the mixed (LSFE-MRQ) and the selective (LSFE-SRQ) have the lowest error per number of degrees of freedom. The nodal formulation starts out with the highest error, but as polynomial order is increased, begins to "catch up" to the other formulations. As expected, the h-type models' error decays in a more or less algebraic fashion, and they are eventually passed by the spectrally refined models.

4.2.3 Accuracy versus Model Efficiency

Figure 9 shows the L_2 error of the center displacement of the plate as a function of floating point operations. The most interesting result on Fig. 9 is the same as that on Fig. 5: the mixed formulation has the lowest number of operations for any given error value (and vice-versa). In addition, unlike before, the nodal quadrature formulation (LSFE-NRQ) passes beneath the other formulations after a smaller polynomial value than before (each point representing another level of spectral refinement for the LSFE models).

4.3 Discussion of Results

Going into this project, it was the author's belief that the nodal formulation was going to be the the most computationally efficient formulation. However, as shown in Figures 5 and 9, the most

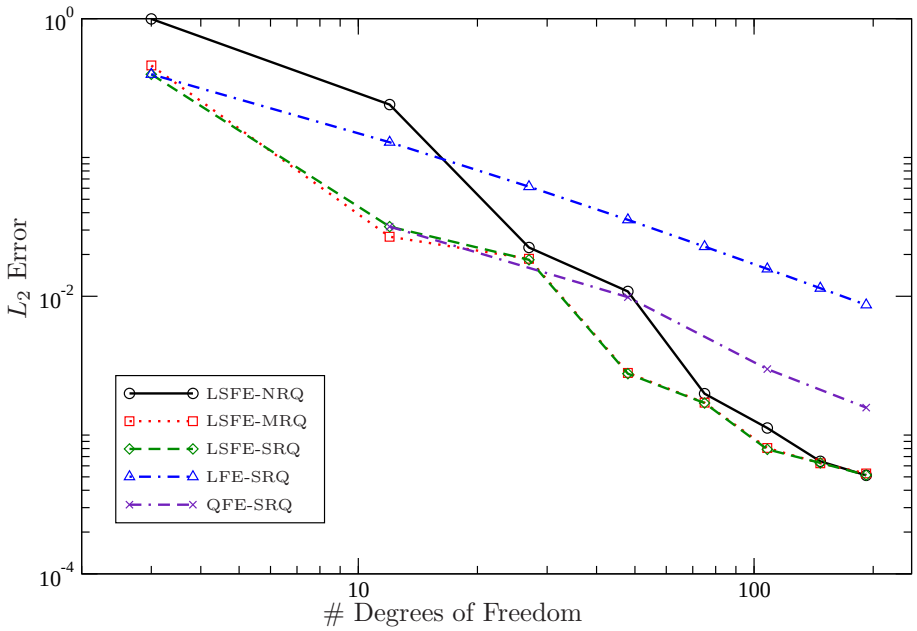


Figure 8: Center-displacement L_2 error vs. model complexity for dynamic-response calculations.

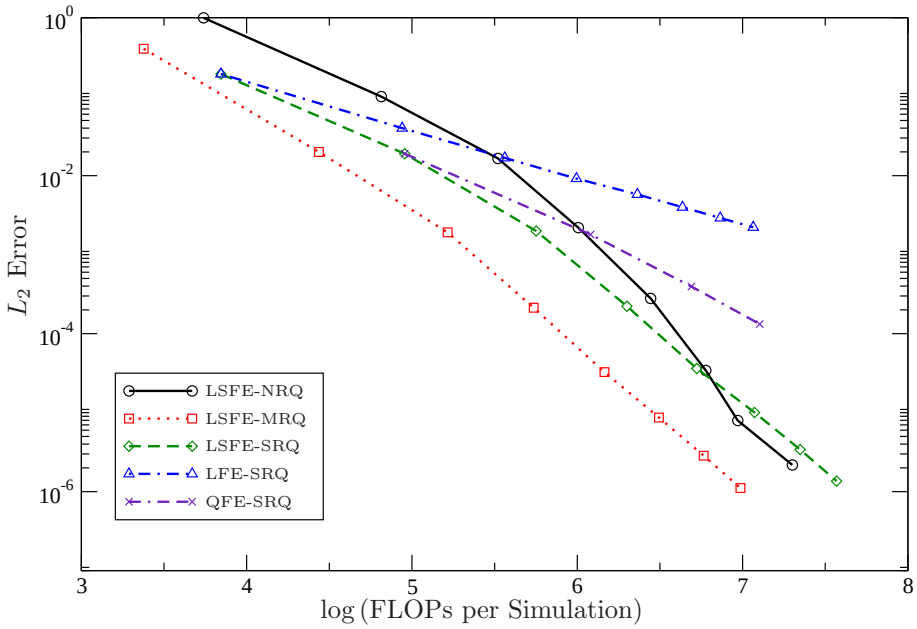


Figure 9: Center-displacement L_2 error vs. model efficiency for dynamic-response calculations.

efficient formulation is the mixed formulation (efficient in the sense that it takes the least amount of operations to obtain a given error). This is explained in part by the data in Figures 4 and 8. In both, the LSFE-MRQ and LSFE-SRQ curves are virtually indistinguishable. The implication being that the error caused by reducing the quadrature on the bending matrix is almost negligible. This is to be expected, of course, as the bending matrix is $\propto a^3$ while the shear is $\propto a$.

Another interesting thing to note is the comparison between Figures 5 and 9. In Figure 5, the nodal formulation is one of the least efficient, except for very high polynomial orders. But in Figure 9, the nodal formulation comes much surpasses the other formulations (excepting the mixed) at a lower polynomial order. This would appear to be because the total number of operations required to perform the dynamic simulations is much higher. In a typical static simulation, a single linear system must be solved. Using conjugate gradient, this requires one matrix-vector product per iteration (the matrix in this case being the global stiffness matrix K). Even for the high refinement, the number of iterations for the static simulations did not exceed the 100s. For the dynamics, as can be seen in Figure 7, the critical time step reaches well below 10^{-3}. Thus, the amount of steps, and the amount of matrix-vector products, often ran in the 1000s for the LSFE simulations. The implication seems to be that, the more matrix-vector products required, the more the operations savings from nodal quadrature start to add up. This is why the nodal quadrature LSFE curve shrinks so much faster in the dynamic case, and the selective GL LSFE curve flattens out so much. The larger the simulation, the more advantageous nodal quadrature becomes.

5 Conclusions

For both the static and dynamic simulations, error versus degree of freedom results were as expected. The p-type LSFE elements clearly outperfom the h-type elements with respect to convergence rates. In addition, the nodal quadrature LSFEs were outperformed by both other LSFEs, which is no surprise as the use of GLL quadrature represents inexact calculation of both bending and shear components of the stiffness matrix. One surprise was the comparative performance of the mixed GL/GLL LSFEs and the selective shear GL LSFEs. Despite the inexact GLL quadrature used for the bending matrix in the mixed formulation, the loss of accuracy is almost negligible.

In terms of performance of the various finite-element schemes, the error versus floating point operations results were most telling. For both static and dynamic simulations, the mixed formulation was superior, taking the least number of floating point operations for a given amount of error. Additionally, results for the other LSFE types revealed an interesting phenomenon. For low-order, smaller simulations, the selective shear GL formulation (LSFE-SRQ) out performed the nodal GLL formulation (LSFE-NRQ) (i.e. lower operation count for the same amount of error). For higher-order simulations, however, the nodal quadrature scheme is superior to the GL scheme. This seems to imply that the savings from using nodal GLL quadrature are much larger for more complex simulations. The answer to the question asked earlier about whether nodal quadrature was "worth it" seems to depend largely on how complex the simulation is.

The primary result of the current study is the superiority, both in convergence rate and computational efficiency, of the mixed quadrature LSFE elements to all others tested. While they are slightly less accurate than the traditional, selective shear quadrature elements (in terms of DoF), the speed gained by using nodal quadrature on the bending matrix more than makes up for any gain in error from inexact quadrature. And though they may be slightly slower for a given number of degrees of freedom than comparable h-types or fully nodal quadrature LSFEs, the gain in error makes them more computationally efficient. Based on their performance for static and dynamic Reissner-Mindlin plates, further analysis of these new element types is clearly warranted.

Acknowledgements

The author wishes to thank the financial supported granted by the Eugene Cota Robles Fellowship.

Appendices

A Finite Element Formulation

The following is the full finite element discretization of the Reissner-Mindlin plate equation presented in equation (3.1). By substituting the isotropic material constants from (3.2) and (3.3) into (3.1), to obtain

$$\begin{aligned}
&\frac{a^3}{12}\int_\Omega \bar{\theta}_{x,x}\left[(2\mu+\bar{\lambda})\theta_{x,x}+\bar{\lambda}\theta_{y,y}\right]+\mu\bar{\theta}_{x,y}\left[\theta_{x,y}+\theta_{y,x}\right]\,d\Omega+a\mu\int_\Omega \bar{\theta}_x\left[w_{,x}-\theta_x\right]\,d\Omega+\\
&\frac{a^3}{12}\int_\Omega \bar{\theta}_{y,y}\left[(2\mu+\bar{\lambda})\theta_{y,y}+\bar{\lambda}\theta_{x,x}\right]+\mu\bar{\theta}_{y,x}\left[\theta_{x,y}+\theta_{y,x}\right]\,d\Omega+a\mu\int_\Omega \bar{\theta}_y\left[w_{,y}-\theta_y\right]\,d\Omega+\\
&a\mu\int_\Omega \bar{w}_{,y}\left(w_{,y}-\theta_y\right)+\bar{w}_{,x}\left(w_{,x}-\theta_x\right)\,d\Omega\\
&\quad=F_{\theta_x}+F_{\theta_y}+F_w,
\end{aligned}\tag{A.1}$$

where

$$F_{\theta_x} = -\int_\Omega \bar{\theta}_x C_x\,d\Omega-\int_{\partial\Omega_n}\bar{\theta}_x M_x\,ds,\tag{A.2}$$

$$F_{\theta_y} = -\int_\Omega \bar{\theta}_y C_y\,d\Omega-\int_{\partial\Omega_n}\bar{\theta}_y M_y\,ds,\tag{A.3}$$

$$F_w = \int_\Omega \bar{w}F\,d\Omega+\int_{\partial\Omega_n}\bar{w}Q\,ds.\tag{A.4}$$

To formulate a finite element approximation to the above equation, we start by approximating the unknown function $u_k(x,y)=(w(x,y),\theta_x(x,y),\theta_y(x,y))^T$ and the weight function $\bar{u}_k(x,y)=(\bar{w}(x,y),\bar{\theta}_x(x,y),\bar{\theta}_y(x,y))^T$ with a finite number of shape functions:

$$\begin{aligned}
u_k(x,y)&\approx\sum_{\hat{\imath}=0}^{N}\Phi_{\hat{\imath}}(x,y)d_{\hat{\imath}}^k,\\
\bar{u}_k(x,y)&\approx\sum_{\hat{\imath}=0}^{N}\Phi_{\hat{\imath}}(x,y)c_{\hat{\imath}}^k,
\end{aligned}\tag{A.5}$$

for constants $d_{\hat{\imath}}^k$ and $c_{\hat{\imath}}^k$. Note the so-called Bubnov-Galerkin formulation: the shape functions are the same for the unknown function $\mathbf{u}$ and the trial functions $\bar{\mathbf{u}}$. These can be substituted into equation (A.1) to obtain:

$$\frac{a^3}{12}\int_\Omega \left(\sum_{\hat{\imath}=0}^{N}\frac{\partial\Phi_{\hat{\imath}}}{\partial x}c_{\hat{\imath}}^2\right)\left[(2\mu+\bar{\lambda})\left(\sum_{\hat{\jmath}=0}^{N}\frac{\partial\Phi_{\hat{\jmath}}}{\partial x}d_{\hat{\jmath}}^2\right)\right]+\mu\left(\sum_{\hat{\imath}=0}^{N}\frac{\partial\Phi_{\hat{\imath}}}{\partial y}c_{\hat{\imath}}^2\right)\left[\left(\sum_{\hat{\jmath}=0}^{N}\frac{\partial\Phi_{\hat{\jmath}}}{\partial y}d_{\hat{\jmath}}^2\right)+\left(\sum_{\hat{\jmath}=0}^{N}\frac{\partial\Phi_{\hat{\jmath}}}{\partial x}d_{\hat{\jmath}}^3\right)\right]\ d\Omega+$$

$$a\mu\int_\Omega \left(\sum_{\hat{\imath}=0}^{N}\Phi_{\hat{\imath}}(x,y)c_{\hat{\imath}}^2\right)\left[\left(\sum_{\hat{\jmath}=0}^{N}\frac{\partial\Phi_{\hat{\jmath}}}{\partial x}d_{\hat{\jmath}}^1\right)-\left(\sum_{\hat{\jmath}=0}^{N}\Phi_{\hat{\jmath}}(x,y)d_{\hat{\imath}}^2\right)\right]\ d\Omega+$$

$$\frac{a^3}{12}\int_\Omega \left(\sum_{\hat{\imath}=0}^{N}\frac{\partial\Phi_{\hat{\imath}}}{\partial y}c_{\hat{\imath}}^3\right)\left[(2\mu+\bar{\lambda})\left(\sum_{\hat{\jmath}=0}^{N}\frac{\partial\Phi_{\hat{\jmath}}}{\partial y}d_{\hat{\jmath}}^3\right)\right]+\mu\left(\sum_{\hat{\imath}=0}^{N}\frac{\partial\Phi_{\hat{\imath}}}{\partial x}c_{\hat{\imath}}^3\right)\left[\left(\sum_{\hat{\jmath}=0}^{N}\frac{\partial\Phi_{\hat{\jmath}}}{\partial y}d_{\hat{\jmath}}^2\right)+\left(\sum_{\hat{\jmath}=0}^{N}\frac{\partial\Phi_{\hat{\jmath}}}{\partial x}d_{\hat{\jmath}}^3\right)\right]\ d\Omega+$$

$$a\mu\int_\Omega \left(\sum_{\hat{\imath}=0}^{N}\Phi_{\hat{\imath}}(x,y)c_{\hat{\imath}}^3\right)\left[\left(\sum_{\hat{\jmath}=0}^{N}\frac{\partial\Phi_{\hat{\jmath}}}{\partial y}d_{\hat{\jmath}}^1\right)-\left(\sum_{\hat{\jmath}=0}^{N}\Phi_{\hat{\jmath}}(x,y)d_{\hat{\imath}}^3\right)\right]\ d\Omega+$$

$$a\mu\int_\Omega \left(\sum_{\hat{\imath}=0}^{N}\frac{\partial\Phi_{\hat{\imath}}}{\partial x}c_{\hat{\imath}}^1\right)\left[\left(\sum_{\hat{\jmath}=0}^{N}\frac{\partial\Phi_{\hat{\jmath}}}{\partial x}d_{\hat{\jmath}}^1\right)-\left(\sum_{\hat{\jmath}=0}^{N}\Phi_{\hat{\jmath}}(x,y)d_{\hat{\jmath}}^2\right)\right]\ d\Omega+$$

$$a\mu\int_\Omega \left(\sum_{\hat{\imath}=0}^{N}\frac{\partial\Phi_{\hat{\imath}}}{\partial y}c_{\hat{\imath}}^1\right)\left[\left(\sum_{\hat{\jmath}=0}^{N}\frac{\partial\Phi_{\hat{\jmath}}}{\partial y}d_{\hat{\jmath}}^1\right)-\left(\sum_{\hat{\jmath}=0}^{N}\Phi_{\hat{\jmath}}(x,y)d_{\hat{\jmath}}^3\right)\right]\ d\Omega$$

$$=$$

$$-\int_\Omega \left(\sum_{\hat{\imath}=0}^{N}\Phi_{\hat{\imath}}(x,y)c_{\hat{\imath}}^2\right)C_x\ d\Omega-\int_{\partial\Omega_n}\left(\sum_{\hat{\imath}=0}^{N}\Phi_i(x,y)c_{\hat{\imath}}^2\right)M_x\ \ ds$$

$$-\int_\Omega \left(\sum_{\hat{\imath}=0}^{N}\Phi_{\hat{\imath}}(x,y)c_{\hat{\imath}}^3\right)C_x\ d\Omega-\int_{\partial\Omega_n}\left(\sum_{\hat{\imath}=0}^{N}\Phi_{\hat{\imath}}(x,y)c_{\hat{\imath}}^3\right)M_y\ \ ds$$

$$\int_\Omega \left(\sum_{\hat{\imath}=0}^{N}\Phi_{\hat{\imath}}(x,y)c_{\hat{\imath}}^1\right)F\ \ d\Omega\ +\int_{\partial\Omega_n}\left(\sum_{\hat{\imath}=0}^{N}\Phi_{\hat{\imath}}(x,y)c_{\hat{\imath}}^1\right)Q\ \ ds, \tag{A.6}$$

which must hold for all test/trial functions $\bar{u}_k(x,y)$ in the admissible function space. Rearranging, and pulling the sums outside of the integrals yields the following:

$$\sum_{\hat{\imath}=0}^{N} c_{\hat{\imath}}^1 \left[\sum_{\hat{\jmath}=0}^{N} a\mu \left(\int_\Omega \frac{\partial \Phi_{\hat{\imath}}}{\partial x}\frac{\partial \Phi_{\hat{\jmath}}}{\partial x} d_{\hat{\jmath}}^1 - \frac{\partial \Phi_{\hat{\imath}}}{\partial x}\Phi_{\hat{\jmath}} d_{\hat{\jmath}}^2 - \frac{\partial \Phi_{\hat{\imath}}}{\partial y}\Phi_{\hat{\jmath}} d_{\hat{\jmath}}^3 + \frac{\partial \Phi_{\hat{\imath}}}{\partial y}\frac{\partial \Phi_{\hat{\jmath}}}{\partial y} d_{\hat{\jmath}}^1 \, d\Omega \right) \right.$$
$$\left. - \int_\Omega \Phi_{\hat{\imath}} F \, d\Omega - \int_{\partial\Omega_n} \Phi_{\hat{\imath}} Q \, ds \right] +$$
$$\sum_{\hat{\imath}=0}^{N} c_{\hat{\imath}}^2 \left[\sum_{\hat{\jmath}=0}^{N} \frac{a^3}{12} \left(\int_\Omega (2\mu+\bar{\lambda}) \frac{\partial \Phi_{\hat{\imath}}}{\partial x}\frac{\partial \Phi_{\hat{\jmath}}}{\partial x} d_{\hat{\jmath}}^2 + \mu \frac{\partial \Phi_{\hat{\imath}}}{\partial y}\frac{\partial \Phi_{\hat{\jmath}}}{\partial y} d_{\hat{\jmath}}^2 + \mu \frac{\partial \Phi_{\hat{\imath}}}{\partial y}\frac{\partial \Phi_{\hat{\jmath}}}{\partial x} d_{\hat{\jmath}}^3 \, d\Omega \right. \right.$$
$$\left. \left. + a\mu \int_\Omega \Phi_{\hat{\imath}} \frac{\partial \Phi_{\hat{\jmath}}}{\partial x} d_{\hat{\jmath}}^1 - \Phi_{\hat{\imath}}\Phi_{\hat{\jmath}} d_{\hat{\jmath}}^2 \, d\Omega \right) + \int_\Omega \Phi_{\hat{\imath}} C_x \, ds + \int_{\partial\Omega_n} \Phi_{\hat{\imath}} M_x \, d\Omega \right] +$$
$$\sum_{\hat{\imath}=0}^{N} c_{\hat{\imath}}^3 \left[\sum_{\hat{\jmath}=0}^{N} \frac{a^3}{12} \left(\int_\Omega (2\mu+\bar{\lambda}) \frac{\partial \Phi_{\hat{\imath}}}{\partial y}\frac{\partial \Phi_{\hat{\jmath}}}{\partial y} d_{\hat{\jmath}}^3 + \mu \frac{\partial \Phi_{\hat{\imath}}}{\partial x}\frac{\partial \Phi_{\hat{\jmath}}}{\partial y} d_{\hat{\jmath}}^2 + \mu \frac{\partial \Phi_{\hat{\imath}}}{\partial x}\frac{\partial \Phi_{\hat{\jmath}}}{\partial x} d_{\hat{\jmath}}^3 \, d\Omega \right. \right.$$
$$\left. \left. + a\mu \int_\Omega \Phi_{\hat{\imath}} \frac{\partial \Phi_{\hat{\jmath}}}{\partial y} d_{\hat{\jmath}}^1 - \Phi_{\hat{\imath}}\Phi_{\hat{\jmath}} d_{\hat{\jmath}}^3 \, d\Omega \right) + \int_\Omega \Phi_{\hat{\imath}} C_y \, ds + \int_{\partial\Omega_n} \Phi_{\hat{\imath}} M_y \, d\Omega \right]$$
$$= 0, \tag{A.7}$$

which, again, must hold for all test/trial functions. This implies that the above must hold for all $c_{\hat{\imath}} \in \Re$. Thus the 3 bracketed terms in equation (A.7) above must individually equal 0:

$$\sum_{\hat{\jmath}=0}^{N} a\mu \left(\int_\Omega \frac{\partial \Phi_{\hat{\imath}}}{\partial x}\frac{\partial \Phi_{\hat{\jmath}}}{\partial x} d_{\hat{\jmath}}^1 - \frac{\partial \Phi_{\hat{\imath}}}{\partial x}\Phi_{\hat{\jmath}} d_{\hat{\jmath}}^2 - \frac{\partial \Phi_{\hat{\imath}}}{\partial y}\Phi_{\hat{\jmath}} d_{\hat{\jmath}}^3 + \frac{\partial \Phi_{\hat{\imath}}}{\partial y}\frac{\partial \Phi_{\hat{\jmath}}}{\partial y} d_{\hat{\jmath}}^1 \, d\Omega \right) = \int_\Omega \Phi_{\hat{\imath}} F \, d\Omega + \int_{\partial\Omega_n} \Phi_{\hat{\imath}} Q \, ds, \tag{A.8}$$

$$\sum_{\hat{\jmath}=0}^{N} \left(\frac{a^3}{12} \int_\Omega (2\mu+\bar{\lambda}) \frac{\partial \Phi_{\hat{\imath}}}{\partial x}\frac{\partial \Phi_{\hat{\jmath}}}{\partial x} d_{\hat{\jmath}}^2 + \mu \frac{\partial \Phi_{\hat{\imath}}}{\partial y}\frac{\partial \Phi_{\hat{\jmath}}}{\partial y} d_{\hat{\jmath}}^2 + \mu \frac{\partial \Phi_{\hat{\imath}}}{\partial y}\frac{\partial \Phi_{\hat{\jmath}}}{\partial x} d_{\hat{\jmath}}^3 \, d\Omega \right.$$
$$\left. + a\mu \int_\Omega \Phi_{\hat{\imath}} \frac{\partial \Phi_{\hat{\jmath}}}{\partial x} d_{\hat{\jmath}}^1 - \Phi_{\hat{\imath}}\Phi_{\hat{\jmath}} d_{\hat{\jmath}}^2 \, d\Omega \right) = - \int_\Omega \Phi_{\hat{\imath}} C_x \, ds - \int_{\partial\Omega_n} \Phi_{\hat{\imath}} M_x \, d\Omega, \tag{A.9}$$

$$\sum_{\hat{\jmath}=0}^{N} \left(\frac{a^3}{12} \int_\Omega (2\mu+\bar{\lambda}) \frac{\partial \Phi_{\hat{\imath}}}{\partial y}\frac{\partial \Phi_{\hat{\jmath}}}{\partial y} d_{\hat{\jmath}}^3 + \mu \frac{\partial \Phi_{\hat{\imath}}}{\partial x}\frac{\partial \Phi_{\hat{\jmath}}}{\partial y} d_{\hat{\jmath}}^2 + \mu \frac{\partial \Phi_{\hat{\imath}}}{\partial x}\frac{\partial \Phi_{\hat{\jmath}}}{\partial x} d_{\hat{\jmath}}^3 \, d\Omega \right.$$
$$\left. + a\mu \int_\Omega \Phi_{\hat{\imath}} \frac{\partial \Phi_{\hat{\jmath}}}{\partial y} d_{\hat{\jmath}}^1 - \Phi_{\hat{\imath}}\Phi_{\hat{\jmath}} d_{\hat{\jmath}}^3 \, d\Omega \right) = - \int_\Omega \Phi_{\hat{\imath}} C_y \, ds - \int_{\partial\Omega_n} \Phi_{\hat{\imath}} M_y \, d\Omega, \tag{A.10}$$

which must hold $\forall \hat{\imath} = 0, 1, ..., N$. The above equations represent $3N$ equations in $3N$ unknowns, which, when solved, give the coefficients for the approximation for $u^k(x, y)$. The above is the matrix-vector representation of the problem presented in (2.7), where $\mathrm{K}\mathbf{d}$ is represented by the lefthand sides of equations (A.8)-(A.10) (where K^{B} are the terms proportional to $\frac{a^3}{12}$ and K^{S} are proportional to a), and $\mathbf{F}$ is represented by the righthand sides.

B Numerical Quadrature and Tensor-Product Factorization

Since the method used to solve the linear system outlined in (A.8)-(A.10) will be solved using Conjugate-Gradient, the stiffness matrix need not be explicitly formed, all that is needed is a form for the vector K**u** formed by multiplying the matrix by any arbitrary vector **u**. The lefthand sides of the above equations are a form for exactly that, so the following formulation will be a form of K**u** using numerical quadrature.

Furthermore, finite elements are designed such that all work can be done on the element level, in the element coordinate system, and assembled globally. So, again, all of the following will apply to element level stiffness matrices, based on integrals done over the domain $[-1,1]\times[-1,1]$. Thus, all integrals will be transformed into the element domain and calculated with numercal quadrature, and will include Jacobian terms for the transformation. To accomplish this, the shape functions will be decomposed in the following manner

$$\Phi_i(x(\xi,\eta),y(\xi,\eta)) = \phi_i(\xi)\phi_j(\eta), \tag{B.1}$$

where ξ and η are the element level coordinate variables. The transformation from global to element level is done in an isoparametric fashion, meaning that the representation of the unknown function **u** is also the representation of the geometric variables, i.e.

$$x(\xi,\eta) = \sum_{i=0}^{n}\sum_{j=0}^{n}\phi_i(\xi)\phi_j(\eta)x_{ij}, \tag{B.2}$$

$$y(\xi,\eta) = \sum_{i=0}^{n}\sum_{j=0}^{n}\phi_i(\xi)\phi_j(\eta)y_{ij}, \tag{B.3}$$

where x_{ij} and y_{ij} are the x and y values of the meshed domain in the element coordinates (i.e. $x_1 1$ would be the x coordinate at the (1,1) position using the element node ordering). Notice in the above that there are $(n+1)\times(n+1)$ nodes per element, i.e. the formulation used n^{th} order polynomial shape functions. Thus, the terms of the Jacobian of the transformation can be easily computed, for example

$$\frac{\partial x}{\partial \xi} = \sum_{i=0}^{n}\sum_{j=0}^{n}\phi_i'(\xi)\phi_j(\eta)x_{ij}, \tag{B.4}$$

with other partial derivatives calculated in a similar fashion. Numerical quadrature can now be applied, and a full-discretized matrix-vector product can be calculated.

B.1 Arbitrary GL Quadrature

Using arbitrary Gauss-Legendre quadrature, $n_b\times n_b$ quadrature points for the bending matrix components, and $n_s\times n_s$ quadrature points for the shear matrix components, for n_p^{th}-order polynomial shape functions, the following discretized formulation for K**d**, which holds for any arbitrary vector **d**:

$$
\begin{aligned}
Ku^1_{ij} = a\mu\kappa \sum_{m=1}^{n_s}\sum_{n=1}^{n_s} \tilde{D}^{n_s}_{im} P^{n_s}_{jn} \Big[& \tilde{\phi}^{1,n_s}_{mn} \left(\Gamma^{n_s}_{2121mn} + \Gamma^{n_s}_{1111mn}\right) + \tilde{\psi}^{1,n_s}_{mn} \left(\Gamma^{n_s}_{2122mn} + \Gamma^{n_s}_{1112mn}\right) + \\
& \gamma^{n_s}_{21mn} \tilde{\theta}^3_{mn} + \gamma^{n_s}_{11mn} \tilde{\theta}^2_{mn} \Big] + \\
P^{n_s}_{im} \tilde{D}^{n_s}_{jn} \Big[& \tilde{\phi}^{1,n_s}_{mn} \left(\Gamma^{n_s}_{2122mn} + \Gamma^{n_s}_{1112mn}\right) + \tilde{\psi}^{1,n_s}_{mn} \left(\Gamma^{n_s}_{2222mn} + \Gamma^{n_s}_{1212mn}\right) \Big] + \\
& \gamma^{n_s}_{22mn} \tilde{\theta}^3_{mn} + \gamma^{n_s}_{12mn} \tilde{\theta}^2_{mn} \Big], \qquad \text{(B.5)}
\end{aligned}
$$

$$
\begin{aligned}
Ku^2_{ij} = \frac{a^3}{12} \sum_{m=1}^{n_b}\sum_{n=1}^{n_b} (2\mu + \bar{\lambda}) \Big[& \tilde{D}^{n_b}_{im} P^{n_b}_{jn} \left(\Gamma^{n_b}_{1111mn} \tilde{\phi}^{2,n_b}_{mn} + \Gamma^{n_b}_{1112mn} \tilde{\psi}^{2,n_b}_{mn}\right) + \\
& P^{n_b}_{im} \tilde{D}^{n_b}_{jn} \left(\Gamma^{n_b}_{1112mn} \tilde{\phi}^{2,n_b}_{mn} + \Gamma^{n_b}_{1212mn} \tilde{\psi}^{2,n_b}_{mn}\right) \Big] + \\
\bar{\lambda} \Big[& \tilde{D}^{n_b}_{im} P^{n_b}_{jn} \left(\Gamma^{n_b}_{1121mn} \tilde{\phi}^{3,n_b}_{mn} + \Gamma^{n_b}_{1122mn} \tilde{\psi}^{3,n_b}_{mn}\right) + \\
& P^{n_b}_{im} \tilde{D}^{n_b}_{jn} \left(\Gamma^{n_b}_{1221mn} \tilde{\phi}^{3,n_b}_{mn} + \Gamma^{n_b}_{1222mn} \tilde{\psi}^{3,n_b}_{mn}\right) \Big] + \\
\mu \Big[& \tilde{D}^{n_b}_{im} P^{n_b}_{jn} \left(\Gamma^{n_b}_{2121mn} \tilde{\phi}^{2,n_b}_{mn} + \Gamma^{n_b}_{2122mn} \tilde{\psi}^{2,n_b}_{mn} + \Gamma^{n_b}_{1121mn} \tilde{\phi}^{3,n_b}_{mn} + \Gamma^{n_b}_{1221mn} \tilde{\psi}^{3,n_b}_{mn}\right) + \\
& P^{n_b}_{im} \tilde{D}^{n_b}_{jn} \left(\Gamma^{n_b}_{2122mn} \tilde{\phi}^{2,n_b}_{mn} + \Gamma^{n_b}_{2222mn} \tilde{\psi}^{2,n_b}_{mn} + \Gamma^{n_b}_{1122mn} \tilde{\phi}^{3,n_b}_{mn} + \Gamma^{n_b}_{1222mn} \tilde{\psi}^{3,n_b}_{mn}\right) \Big] + \\
a\mu\kappa \sum_{m=1}^{n_s}\sum_{n=1}^{n_s} & P^{n_s}_{im} P^{n_s}_{jn} \left(J^{n_s}_{mn} \tilde{\theta}^{2,n_s}_{mn} - \gamma^{n_s}_{11mn} \tilde{\phi}^{1,n_s}_{mn} - \gamma^{n_s}_{12mn} \tilde{\psi}^{1,n_s}_{mn}\right), \qquad \text{(B.6)}
\end{aligned}
$$

$$
\begin{aligned}
Ku^3_{ij} = \frac{a^3}{12} \sum_{m=1}^{n_b}\sum_{n=1}^{n_b} (2\mu + \bar{\lambda}) \Big[& \tilde{D}^{n_b}_{im} P^{n_b}_{jn} \left(\Gamma^{n_b}_{2121mn} \tilde{\phi}^{3,n_b}_{mn} + \Gamma^{n_b}_{2122mn} \tilde{\psi}^{3,n_b}_{mn}\right) + \\
& P^{n_b}_{im} \tilde{D}^{n_b}_{jn} \left(\Gamma^{n_b}_{2122mn} \tilde{\phi}^{3,n_b}_{mn} + \Gamma^{n_b}_{2222mn} \tilde{\psi}^{3,n_b}_{mn}\right) \Big] + \\
\bar{\lambda} \Big[& \tilde{D}^{n_b}_{im} P^{n_b}_{jn} \left(\Gamma^{n_b}_{1121mn} \tilde{\phi}^{2,n_b}_{mn} + \Gamma^{n_b}_{1221mn} \tilde{\psi}^{2,n_b}_{mn}\right) + \\
& P^{n_b}_{im} \tilde{D}^{n_b}_{jn} \left(\Gamma^{n_b}_{1122mn} \tilde{\phi}^{2,n_b}_{mn} + \Gamma^{n_b}_{1222mn} \tilde{\psi}^{2,n_b}_{mn}\right) \Big] + \\
\mu \Big[& \tilde{D}^{n_b}_{im} P^{n_b}_{jn} \left(\Gamma^{n_b}_{1121mn} \tilde{\phi}^{2,n_b}_{mn} + \Gamma^{n_b}_{1122mn} \tilde{\psi}^{2,n_b}_{mn} + \Gamma^{n_b}_{1111mn} \tilde{\phi}^{3,n_b}_{mn} + \Gamma^{n_b}_{1112mn} \tilde{\psi}^{3,n_b}_{mn}\right) + \\
& P^{n_b}_{im} \tilde{D}^{n_b}_{jn} \left(\Gamma^{n_b}_{1221mn} \tilde{\phi}^{2,n_b}_{mn} + \Gamma^{n_b}_{1222mn} \tilde{\psi}^{2,n_b}_{mn} + \Gamma^{n_b}_{1112mn} \tilde{\phi}^{3,n_b}_{mn} + \Gamma^{n_b}_{1212mn} \tilde{\psi}^{3,n_b}_{mn}\right) \Big] + \\
a\mu\kappa \sum_{m=1}^{n_s}\sum_{n=1}^{n_s} & P^{n_s}_{im} P^{n_s}_{jn} \left(J^{n_s}_{mn} \tilde{\theta}^{3,n_s}_{mn} - \gamma^{n_s}_{21mn} \tilde{\phi}^{1,n_s}_{mn} - \gamma^{n_s}_{22mn} \tilde{\psi}^{1,n_s}_{mn}\right), \qquad \text{(B.7)}
\end{aligned}
$$

where those interim tensors are defined as :

$$
\tilde{\phi}^{\alpha,n_q}_{ij} = \sum_{k=1}^{n_p+1}\sum_{l=1}^{n_p+1} \tilde{D}^{n_q}_{ki} P^{n_q}_{lj} d^{\alpha}_{kl}, \quad \tilde{\psi}^{\alpha,n_q}_{ij} = \sum_{k=1}^{n_p+1}\sum_{l=1}^{n_p+1} P^{n_q}_{ki} \tilde{D}^{n_q}_{lj} d^{\alpha}_{kl}, \quad \tilde{\theta}^{\alpha,n_q}_{ij} = \sum_{k=1}^{n_p+1}\sum_{l=1}^{n_p+1} P^{n_q}_{ki} P^{n_q}_{lj} d^{\alpha}_{kl}, \qquad \text{(B.8)}
$$

where $i, j = 1, ..., n_q$, n_q being the number of quadrature points (so either $n_q = n_b$ or n_s). In addition,

$$
\begin{aligned}
P^{n_q}_{ij} &= \phi_i(\xi_j), \qquad \text{(B.9)} \\
\tilde{D}^{n_q}_{ij} &= \phi'_i(\xi_j), \qquad \text{(B.10)}
\end{aligned}
$$

where the ξ's are the quadrature points. Finally,

$$J_{ij}^{n_q} = w_i w_j |\mathrm{J}|(\xi_i, \eta_j), \tag{B.11}$$

$$\Gamma_{abcdij}^{n_q} = w_i w_j \tilde{\gamma}_{ab}(\xi_i, \eta_j)\tilde{\gamma}_{cd}(\xi_i, \eta_j) J_{ij}^{n_q}, \tag{B.12}$$

$$\gamma_{abij}^{n_q} = w_i w_j \tilde{\gamma}_{ab}(\xi_i, \eta_j), \tag{B.13}$$

where $|\mathrm{J}|(\xi, \eta)$ is the determinant of the Jacobian, $\tilde{\gamma}_{ab}(\xi, \eta)$ is the a, b^{th} component of J^{-1} (so $a, b = 1$ or 2), and w_i is the i^{th} quadrature weight associated with the n_q point quadrature rule used. In practice, the above may be rearranged, and constants combined, for additional efficiency, but the above equation still holds.

B.2 Nodal GLL Quadrature

The formulation in equations (B.5)-(B.7) can be formulated for nodal Gauss-Legendre-Lobatto quadrature, and is given by:

$$\begin{aligned} Ku_{ij}^1 = a\mu\kappa \sum_{m=1}^{n_p+1} D_{im} \big[&(\Gamma_{1111mj} + \Gamma_{2121mj}) \phi_{mj}^1 + \\ &(\Gamma_{1112mj} + \Gamma_{2122mj}) \psi_{jm}^1 - \gamma_{11mj} d_{mj}^2 - \gamma_{21mj} d_{mj}^3 \big] + \\ D_{jm} \big[&(\Gamma_{1112im} + \Gamma_{2122im}) \phi_{im}^1 + \\ &(\Gamma_{1212im} + \Gamma_{2222im}) \psi_{mi}^1 - \gamma_{12im} d_{im}^2 - \gamma_{22im} d_{im}^3 \big], \end{aligned} \tag{B.14}$$

$$\begin{aligned} Ku_{ij}^2 = \frac{a^3}{12} \sum_{m=1}^{n_p+1} &(2\mu + \bar{\lambda}) \left[D_{im} \left(\Gamma_{1111mj} \phi_{mj}^2 + \Gamma_{1112mj} \psi_{jm}^2 \right) + D_{jm} \left(\Gamma_{1112im} \phi_{im}^2 + \Gamma_{1212im} \psi_{mi}^2 \right) \right] + \\ &\bar{\lambda} \left[D_{im} \left(\Gamma_{1121mj} \phi_{mj}^3 + \Gamma_{1122mj} \psi_{jm}^3 \right) + D_{jm} \left(\Gamma_{1221im} \phi_{im}^3 + \Gamma_{1222im} \psi_{mi}^3 \right) \right] + \\ &\mu \left[D_{im} \left(\Gamma_{2121mj} \phi_{mj}^2 + \Gamma_{2122mj} \psi_{jm}^2 + \Gamma_{1121mj} \phi_{mj}^3 + \Gamma_{1221mj} \psi_{jm}^3 \right) \right. \\ &\left. D_{jm} \left(\Gamma_{2122im} \phi_{im}^2 + \Gamma_{2222im} \psi_{mi}^2 + \Gamma_{1122im} \phi_{im}^3 + \Gamma_{1222im} \psi_{mi}^3 \right) \right] + \\ &a\mu\kappa \left(J_{ij} d_{ij}^2 - \gamma_{11ij} \phi_{ij}^1 - \gamma_{12ij} \psi_{ji}^1 \right), \end{aligned} \tag{B.15}$$

$$\begin{aligned} Ku_{ij}^3 = \frac{a^3}{12} \sum_{m=1}^{n_p+1} &(2\mu + \bar{\lambda}) \left[D_{im} \left(\Gamma_{2121mj} \phi_{mj}^3 + \Gamma_{2122mj} \psi_{jm}^3 \right) + D_{jm} \left(\Gamma_{2122im} \phi_{im}^3 + \Gamma_{2222im} \psi_{mi}^3 \right) \right] + \\ &\bar{\lambda} \left[D_{im} \left(\Gamma_{1121mj} \phi_{mj}^2 + \Gamma_{1221mj} \psi_{jm}^2 \right) + D_{jm} \left(\Gamma_{1122im} \phi_{im}^2 + \Gamma_{1222im} \psi_{mi}^2 \right) \right] + \\ &\mu \left[D_{im} \left(\Gamma_{1121mj} \phi_{mj}^2 + \Gamma_{1122mj} \psi_{jm}^2 + \Gamma_{1111mj} \phi_{mj}^3 + \Gamma_{1112mj} \psi_{jm}^3 \right) \right. \\ &\left. D_{jm} \left(\Gamma_{1221im} \phi_{im}^2 + \Gamma_{1222im} \psi_{mi}^2 + \Gamma_{1112im} \phi_{im}^3 + \Gamma_{1212im} \psi_{mi}^3 \right) \right] + \\ &a\mu\kappa \left(J_{ij} d_{ij}^3 - \gamma_{21ij} \phi_{ij}^1 - \gamma_{22ij} \psi_{ji}^1 \right), \end{aligned} \tag{B.16}$$

where the intermediate tensors are defined as

$$\phi_{ij}^\alpha = \sum_{m=1}^{n_p+1} D_{mi} d_{mj}^\alpha \quad \psi_{ij}^\alpha = \sum_{m=1}^{n_p+1} D_{mi} d_{jm}^\alpha, \tag{B.17}$$

and the other various tensors are defined in the same way as for the arbitrary GL quadrature, only using GLL points and GLL weights.

The formulation for the mixed is a combination of the two formulations above, with the GL form used for terms proportional to a, and the GLL used for the terms proportional to a^3. All intermediate and other tensors remain the same.

References

[1] K. E. Atkinson. *An Introduction to Numerical Analysis*. John Wiley & Sons, 2nd edition, 2004.

[2] W. Dauksher and A.F. Emery. The solution of elastostatic and elastodynamic poblems with Chebyshev spectral finite elements. *Computer Methods in Applied Mechanics and Engineering*, 188:217–233, 2000.

[3] E. Faccioli, F. Maggio, R. Paolucci, and A. Quarteroni. 2D and 3D elastic wave propagation by a pseudo-spectral domain decomposition method. *Journal of Seismology*, 1:237–251, 1997.

[4] I. Fried and D. S. Malkus. Finite element mass matrix lumping by numerical integration with no convergence rate loss. *International Journal of Solids and Structures*, 11:461–466, 1974.

[5] T. J.R. Hughes. *The Finite Element Method: Linear Static and Dynamic Analysis*. Dover Publications, Inc., 2000.

[6] P. Kudela, M. Krawczuk, and W. Ostahowicz. Wave propagation modelling in 1D structures using spectral finite elements. *Journal of Sound and Vibration*, 300:88–100, 2007.

[7] P. Kudela, A. Żak, M. Krawczuk, and W. Ostrachowicz. Modelling of wave propagation in composite plates using the time domain spectral element method. *Journal of Sound and Vibration*, 302:728–745, 2007.

[8] M. Levinson and D. W. Cooke. Thick rectangular plates I: The generalized Navier solution. *International Journal of Mechanical Sciences*, 25:199–203, 1983.

[9] X. Ling and H.P. Cherukuri. Stability analysis of an explicit finite element scheme for plane wave motions in elastic solids. *Computational Mechanics*, 29:430–440, 2002.

[10] A. T. Patera. A spectral element method for fluid dynamics: Laminar flow in a channel expansion. *Journal of Computational Physics*, 29:468–488, 1984.

[11] H. Peng, G. Meng, and F. Li. Modeling of wave propagation in plate structures using three-dimensional spectral element method for damage detection. *Journal of Sound and Vibration*, 320:942–954, 2009.

[12] E.D.L. Pugh, E. Hinton, and O.C. Zienkiewicz. A study of quadrilateral plate bending elements with 'reduced' integration. *International Journal for Numerical Methods in Engineering*, 12:1059–1079, 1978.

[13] E.M. Rønquist and A. Patera. A Legendre spectral finite element method for the Stefan problem. *International Journal for Numerical Methods in Engineering*, 24:2273–2299, 1987.

[14] M.A. Sprague and T.L. Geers. Legendre spectral finite elements for structural dynamics analysis. *Communications in Numerical Methods in Engineering*, 24:1953–1965, 2008.

[15] E. Zampieri and A. Tagliani. Numerical approximation of elastic waves equations by implicit spectral methods. *Computer Methods in Applied Mechanics and Engineering*, 144:33–50, 1997.

[16] U. Zrahia and P. Bar-Yoseph. Space-time spectral element method for solution of second-order hyperbolic equations. *Computational Methods in Applied Mechanics and Engineering*, 116:135–146, 1994.

[17] U. Zrahia and P. Bar-Yoseph. Plate spectral elements based upon Reissner-Mindlin theory. *International Journal for Numerical Methods in Engineering*, 38:1341–1360, 1995.