UCRL-JRNL-205643

A Generalized Mass Lumping Technique for Vector Finite Element Solutions of the Time Dependent Maxwell Equations

A. Fisher, R. Rieben, G. Rodrigue, D. White

July 30, 2004

IEEE Transactions on Antennas and Propagation

Disclaimer

This document was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor the University of California nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or the University of California. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or the University of California, and shall not be used for advertising or product endorsement purposes.

A Generalized Mass Lumping Technique for Vector Finite Element Solutions of the Time Dependent Maxwell Equations[1]

A. Fisher[2] R. Rieben[3] G. Rodrigue[4] and D. White[5]

Abstract

Time domain finite element solutions of Maxwell's equations require the solution of a sparse linear system involving the mass matrix at every time step. This process represents the bulk of the computational effort in time dependent simulations. As such, mass lumping techniques in which the mass matrix is reduced to a diagonal or block-diagonal matrix are very desirable. In this paper, we present a special set of high order 1-form (also known as curl-conforming) basis functions and reduced order integration rules that together, allow for a dramatic reduction in the number of non-zero entries in a vector finite element mass matrix. The method is derived from the Nedelec curl-conforming polynomial spaces and is valid for arbitrary order hexahedral basis functions for finite element solutions to the second order wave equation for the electric (or magnetic) field intensity. We present a numerical eigenvalue convergence analysis of the method and quantify its accuracy and performance via a series of computational experiments.

I. Introduction

Finite element solutions of the Maxwell vector wave equation using 1-form (also known as "edge" or curl-conforming) basis functions have been used with tremendous success over the past several years. However, one major drawback to this approach is that a sparse linear system involving the mass matrix needs to be solved at every time step (more than once per step for high order time integration methods [1]). This computational cost is not prohibitive, but it does represent the bulk of computational effort required to perform time domain simulations of wave propagation. For the special case of lowest order basis functions and an orthogonal Cartesian mesh, it is possible to invert the mass matrix excactly in $O(n)$ operations [2]. This algorithm does not extend to higher-order discretization, however. Mass lumping techniques, in which the mass matrix is reduced to a diagonal matrix, have been used in the computational mechanics and acoustics [3] communities for the standard first order Lagrangian (or nodal) elements with much success, and in fact this is the norm for time dependent computational mechanics [4]. In this approach, the mass matrix is reduced to a diagonal matrix where each entry is a measure of the volume (or Δh) of each element. This matrix is trivial to invert, resulting in a drastic speed-up for time dependent problems.

An analogue of this procedure is therefore highly desirable for 1-form finite element solutions of Maxwell's equations. Unstructured triangular and tetrahedral computational meshes are popular for electromagnetics because they allow for the accurate modeling of complicated geometries. In [5] the mass lumping idea is applied to Maxwell's equations in a relativly straightforward manner with good results for $1st$ order elements. In [6] and [7] orthogonal 1-form basis functions were developed for triangular and tetrahedral elements, respectively. These result in a diagonal mass matrix. One disadvantage of this approach is that it employs local "bubble" functions, that are required to produce orthogonality, but do not increase the accuracy of the computational solution. A second disadvantage is that it is not clear how to extend these orthogonal basis functions to arbitrary order.

Hexahedral computation meshes are difficult to generate for complex geometries, but for simple geometries they often lead to increased efficiency. It is well known that for the special case of lowest order 1-form basis functions on an orthogonal Cartesian mesh, the mass matrix can be made diagonal by using the trapezoidal rule for the required integrations. The result is equivalent to a FDTD method. In [8] an effective preconditioner is created in order to efficently invert mass matrices formed with $1st$ order elements. An approach for higher-order 1-form basis functions, involving the second family of Nedelec finite element spaces [9], was developed in [10], [11]. In this approach a three way tensor product of Gauss-Lobatto points is chosen for both the interpolation points for the Nedelec edge basis functions, and the integration points used to compute the mass and stiffness integrals. This leads to the formation of block diagonal mass matrix which can be inverted with little computational cost. However, the second family of Nedelec spaces introduces spurious modes to the solution which can lead to the degradation of accuracy.

Another approach developed in [10] avoids the problem of spurious modes by using the original Nedelec finite element spaces [12]. In this approach orthogonal elements are used so that the directions of the basis functions are either in $\hat{\mathbf{x}}$, $\hat{\mathbf{y}}$, or $\hat{\mathbf{z}}$. Since these directions are mutually orthogonal, mass integrals involving differing directions will all evaluate to zero leaving only 3 sets of

[1] This work was performed under the auspices of the U.S. Department of Energy by the University of California, Lawrence Livermore National Laboratory under contract No. W-7405-Eng-48 and supported in part under AFOSR grant F49620-01-1-0327

[2] University of California, Davis and Institute for Scientific Computing Research, Lawrence Livermore National Laboratory, fisher47@llnl.gov

[3] Defense Sciences Engineering Division, Lawrence Livermore National Laboratory, rieben1@llnl.gov

[4] University of California, Davis and Lawrence Livermore National Laboratory, ghrodrigue@ucdavis.edu

[5] Defense Sciences Engineering Division, Lawrence Livermore National Laboratory, white37@llnl.gov

mass integrals to consider. Separate quadrature rules are applied to each of the 3 sets of mass integrals and the off diagonal mass matrix elements are eliminated. While this approach does lead to a diagonal mass matrix, the restriction to orthogonal meshes limits its utility.

In our proposed Generalized Mass Lumping (GML) approach [13], the original Nedelec finite element spaces [12] are used along with multiple quadrature rules. By considering a total of 9 sets of mass integrals, we are able to generalize to non-Cartesian unstructured hexahedral meshes. We believe it is possible to extend this method to include tetrahedral meshes, but such an extension is non-trivial, and beyond the scope of this paper. On orthogonal meshes this method still produces diagonal mass matrices, and on non-orthogonal elements this method greatly increases the sparsity of mass matrices. In this paper we evaluate the accuracy and efficiency of the GML approach on both orthogonal and non-orthogonal meshes. An eigenvalue analysis is used to quantify the dispersive proprieties of the GML approach. The key result is that, on a distorted unstructured hexahedral mesh, the GML method reduces the computational cost by an order of magnitude, with negligible loss in accuracy.

II. Semi-Discrete Wave Equation

The second order wave equation for the time dependent electric field intensity is given by

$$\begin{array}{rcll} \varepsilon\frac{\partial^2}{\partial t^2}\mathbf{E}+\sigma\frac{\partial}{\partial t}\mathbf{E} & = & -\nabla\times(\mu^{-1}\nabla\times\mathbf{E})-\frac{\partial}{\partial t}\mathbf{J} & \text{in } \Omega \\ \nabla\cdot(\varepsilon\mathbf{E}) & = & 0 & \text{in } \Omega \\ \hat{\mathbf{n}}\times\mathbf{E} & = & \mathbf{E}_{bc} & \text{on } \partial\Omega \\ \mathbf{E}(t) & = & \mathbf{E}_{ic} & \text{at } t=t_0 \\ \frac{\partial}{\partial t}\mathbf{E}(t) & = & \frac{\partial}{\partial t}\mathbf{E}_{ic} & \text{at } t=t_0 \end{array} \tag{1}$$

For sake of simplicity, we consider here the case of time-independent material properties and zero conductivity. The semi-discrete version equation can be written as

$$M\frac{\partial^2}{\partial t^2}e = S\,e-\frac{\partial}{\partial t}j \tag{2}$$

where M and S are the mass and stiffness matrices. Typically a second-order central-difference "leapfrog" method is used to integrate (2), which requires the repeated solution of a linear system of the form $\mathbf{Mx}=\mathbf{b}$. For efficiency, iterative methods such as diagonally-scaled conjugate gradient, are used to solve this system. The computational cost depends upon both the sparsity and conditioning of the mass matrix $\mathbf{M}$. In section III below, special basis functions and quadrature rules are developed that result in increased sparsity and improved conditioning of the mass matrix $\mathbf{M}$, significantly increasing the efficiency of the overall method. As done in [14] and [15] the basis functions are constructed on the reference element (a cube) and then transformed to the actual element using the appropriate transformation laws.

It should be noted that in many cases the "leapfrog" method is not the preferred choice for time discretization. For instance, the unconditionally stable Newmark-Beta method can be advantagous and is quite popular for the solution on such equations. The improvements of implicit methods like Newmark-Beta do not come without some cost, which in this case is the need to solve a more complicated linear system involving the sum of the mass and stiffness matrices. This calls into question the usefulness of the improved mass matrix sparsity that the GML method provides, since we will be stuck with the much less sparse stiffness matrix anyway. However, initial testing has shown that when the same GML method is applied to the stiffness matrix there is also an increase in sparsity. When considering the sum of the mass and stiffness matrices there is a roughly %25 reduction in the number of non-zero entries. While this is not the order of magnitude one sees when using the "leapfrog" method, it is still a welcome improvement.

III. Basis Functions

The Lagrange interpolatory polynomial of degree p is defined by a distinct set of $p+1$ real valued interpolation points denoted by the symbol X, such that $X=\{x_0,x_1,\ldots,x_p\}$. The polynomial is constructed in such a way that it has a value of unity at interpolation point i and a value of zero at every other interpolation point. A formula for the Lagrange interpolatory polynomial of degree p is given by

$$L_i^p(x;X)=\prod_{\substack{j=0\\ j\neq i}}^{p}\frac{(x-x_j)}{(x_i-x_j)} \tag{3}$$

By construction these polynomials satisfy the interpolation property given by

$$\begin{array}{c} L_i^p(x_j;X)=0 \\ \forall i\neq j \end{array} \tag{4}$$

Let $\hat{W}$ denote a 1-form basis on the reference element. These 1-form basis functions of polynomial degree p are given by

$$\begin{aligned} \hat{W}^{(x)} &= L_i^p(y;B)L_j^p(z;B)L_k^{p-1}(x;G)\,\hat{\mathbf{x}} \\ \hat{W}^{(y)} &= L_i^p(x;B)L_j^p(z;B)L_k^{p-1}(y;G)\,\hat{\mathbf{y}} \\ \hat{W}^{(z)} &= L_i^p(x;B)L_j^p(y;B)L_k^{p-1}(z;G)\,\hat{\mathbf{z}} \end{aligned} \tag{5}$$

$$i,j = 0,\ldots,p;\;\; k = 0,\ldots,p-1$$

where the superscripts (x), (y), and (z) denote the basis functions parallel to the $\hat{\mathbf{x}}$, $\hat{\mathbf{y}}$, and $\hat{\mathbf{z}}$ direction vectors. The basis function interpolation locations are found at the well known Gauss-Legendre integration points (G) and at the lesser known Gauss-Lobatto integration points (B) on $[0,1]$. The Gauss-Legendre points, along with their associated weights form a quadrature rule that maximizes the order of integration. The Gauss-Lobatto points and weights work in a similar fashion, however, they maximize the order of integration with two of the integration points constrained to the endpoints 0 and 1.

In order to satisfy the locality property, we can break this set of basis functions into three mutually disjoint subsets such that

$$\hat{W} = \hat{W}_e \cup \hat{W}_f \cup \hat{W}_v \tag{6}$$

where the subscripts e, f and v denote the edges, faces and volume of the reference element respectively. For 1-forms, locality implies that the edge basis functions should have non-vanishing tangential components along one and only one edge. The face basis functions will have non-vanishing tangential components along one and only one face with zero tangential components along any edges. Finally, the volume basis functions will have zero for tangential components along either edges or faces. The 1-form edge basis functions of polynomial degree p are given by

$$\hat{W}_e = \begin{cases} L_i^p(y;B)L_j^p(z;B)L_k^{p-1}(x;G)\,\hat{\mathbf{x}} \\ L_i^p(x;B)L_j^p(z;B)L_k^{p-1}(y;G)\,\hat{\mathbf{y}} \\ L_i^p(x;B)L_j^p(y;B)L_k^{p-1}(z;G)\,\hat{\mathbf{z}} \end{cases} \tag{7}$$

$$i,j = 0,p;\;\; k = 0,\ldots,p-1$$

This set of functions is grouped into three sub-sets, one for each of $\hat{\mathbf{x}}$, $\hat{\mathbf{y}}$, $\hat{\mathbf{z}}$. The indices i and j loop over the 4 edges that are tangent to these basis vectors. The index k loops over the p basis functions per edge for a total of $12p$. The 1-form face basis functions of polynomial degree p are given by

$$\hat{W}_f = \begin{cases} L_i^p(x;B)L_j^p(z;B)L_k^{p-1}(y;G)\,\hat{\mathbf{y}} \\ L_i^p(x;B)L_j^p(y;B)L_k^{p-1}(z;G)\,\hat{\mathbf{z}} \\ L_i^p(y;B)L_j^p(z;B)L_k^{p-1}(x;G)\,\hat{\mathbf{x}} \\ L_i^p(y;B)L_j^p(x;B)L_k^{p-1}(z;G)\,\hat{\mathbf{z}} \\ L_i^p(z;B)L_j^p(y;B)L_k^{p-1}(x;G)\,\hat{\mathbf{x}} \\ L_i^p(z;B)L_j^p(x;B)L_k^{p-1}(y;G)\,\hat{\mathbf{y}} \end{cases} \tag{8}$$

$$i = 0,p;\;\; j = 1,\ldots,p-1;\;\; k = 0,\ldots,p-1$$

This set of functions is grouped into six sub-sets, two for each face representing the contravariant basis vectors that are in the plane of that face. The index i loops over the 2 faces that are coplanar to these basis vectors. The indices j and k loop over the $2p(p-1)$ basis functions per face for a total of $12p(p-1)$. Finally, there will be a total of $3p(p-1)^2$ interpolatory basis functions that are internal to the reference element given by

$$\hat{W}_v = \begin{cases} L_i^p(y;B)L_j^p(z;B)L_k^{p-1}(x;G)\,\hat{\mathbf{x}} \\ L_i^p(x;B)L_j^p(z;B)L_k^{p-1}(y;G)\,\hat{\mathbf{y}} \\ L_i^p(x;B)L_j^p(y;B)L_k^{p-1}(z;G)\,\hat{\mathbf{z}} \end{cases} \tag{9}$$

$$i,j = 1,\ldots,p-1;\;\; k = 0,\ldots,p-1$$

IV. Bilinear Forms and Generalized Mass Lumping

In order to make integration computations as simple and efficient as possible, it is advantageous to begin by introducing a reference element and associated coordinate system. As shown, we chose the unit cube and the standard basis vectors for our reference element and coordinate system $\hat{\Omega}$.

$$\hat{\Omega} = \{(\hat{\mathbf{x}},\hat{\mathbf{y}},\hat{\mathbf{z}}); 0 \le (\hat{\mathbf{x}},\hat{\mathbf{y}},\hat{\mathbf{z}}) \le 1\} \tag{10}$$

For 1-forms on an arbitrary hexahedral element Ω, we have the following symmetric bilinear forms (integrated over the reference coordinate system for convenience)

$$M(\hat{\mathbf{w}}_i,\hat{\mathbf{w}}_j)=\int_{\hat{\Omega}}((\varepsilon\circ\Phi)\mathbf{J}^{-1}\hat{\mathbf{w}}_i)\cdot(\mathbf{J}^{-1}\hat{\mathbf{w}}_j)\;|\mathbf{J}|d\hat{\Omega} \tag{11}$$

$$S(\hat{\mathbf{w}}_i,\hat{\mathbf{w}}_j)=\int_{\hat{\Omega}}((\mu^{-1}\circ\Phi)\frac{1}{|\mathbf{J}|}\mathbf{J}^T(\nabla\times\hat{\mathbf{w}}_i))\cdot(\frac{1}{|\mathbf{J}|}\mathbf{J}^T(\nabla\times\hat{\mathbf{w}}_{\mathbf{j}}))\;|\mathbf{J}|d\hat{\Omega} \tag{12}$$

where $\hat{\mathbf{w}}_i\in\hat{W}$, Φ is the mapping from the coordinate system of the reference element to the coordinate system of the arbitrary element, and J is the Jacobian of that mapping. The mass and stiffness matrices created using these forms will scale as $\alpha\,m^1$ (meters) and $\gamma\,m^{-1}$ respectively.

As in equation (5) the basis functions can be divided into 3 distinct sets based on the directions they are pointing. Using these three sets of basis functions, a total of 9 sub mass matrices can be defined by

$$\begin{gathered} M^{(ab)}=M(\hat{\mathbf{w}}_i{}^{(a)},\hat{\mathbf{w}}_j{}^{(b)}) \\ a,b\in\{x,y,z\} \end{gathered} \tag{13}$$

again superscripts (x), (y), and (z) denote the directions the basis functions are pointing. These sub mass matrices can be grouped together to form the original mass matrix by

$$M_\alpha=\begin{pmatrix} M^{(xx)} & M^{(xy)} & M^{(xz)} \\ M^{(yx)} & M^{(yy)} & M^{(yz)} \\ M^{(zx)} & M^{(zy)} & M^{(zz)} \end{pmatrix} \tag{14}$$

Now, as long as the order of accuracy remains consistent, each of these 9 sub-mass matrices can be formed using different quadrature rules to approximate their mass integrals. The 3 sub-mass matrices on the diagonal each consist of basis function interactions from a single set of bases. Since each set of basis functions interpolates on a simple set of points defined by a tensor product, the following choices arise for their quadrature points

$$\begin{gathered} \xi^{(xx)}_{lmn}=(g_l,b_m,b_n) \\ \xi^{(yy)}_{lmn}=(b_l,g_m,b_n) \\ \xi^{(zz)}_{lmn}=(b_l,b_m,g_n) \\ g_i\in G,\quad b_i\in B \end{gathered} \tag{15}$$

where the G and B are again the Gauss-Legendre and Gauss-Lobatto quadrature points respectively, and l, m, and n are the indices into G, and B. The choice of these quadrature points with their associated weights defines 3 quadrature rules with a single high order of accuracy $(2p-1)$ in x,y, and z. While there are many other possibilities using the same number of quadrature points, they either have lower orders of accuracy or they have differing orders of accuracy in the different directions and the 3 different integration rules. To maintain accurate results, it is important to have consistent high orders of accuracy in the quadrature rules. Also, by choosing the quadrature points and basis interpolation points that coincide, use of the interpolation property (4) is made in all 3 dimensions to obtain the following property

$$\begin{gathered} \hat{\mathbf{w}}^{(x)}_{(ijk)}(\xi^{(xx)}_{lmn})=0 \\ \forall(ijk)\neq lmn \end{gathered} \tag{16}$$

where the (ijk) corresponds the the i, j, and k found in the definition of the basis functions (5). Now, if the mass integral is approximated with a quadrature rule using the $\xi^{(xx)}_{lmn}$ points and the associated quadrature weights $a^{(xx)}_{lmn}$ the following result is obtained.

$$\begin{gathered} M^{(xx)}_{(ijk)(ijk)'}\approx\sum_{lmn}a^{(xx)}_{lmn}((\alpha\circ\Phi)\mathbf{J}^{-1}\hat{\mathbf{w}}^{(x)}_{(ijk)}(\xi^{(xx)}_{lmn}))\cdot(\mathbf{J}^{-1}\hat{\mathbf{w}}^{(x)}_{(ijk)'}(\xi^{(xx)}_{lmn}))|\mathbf{J}|=0 \\ \forall(ijk)\neq(ijk)' \\ since\;\hat{\mathbf{w}}^{(x)}_{(ijk)}(\xi^{(xx)}_{lmn})=0\;\;or\;\;\hat{\mathbf{w}}^{(x)}_{(ijk)'}(\xi^{(xx)}_{lmn})=0\;\;\forall lmn \end{gathered} \tag{17}$$

This forms a diagonal $M^{(xx)}$ sub-mass matrix by the mechanism outlined in Figure 1. Similar results exist to form diagonal $M^{(yy)}$ and $M^{(zz)}$ sub-mass matrices using the (xx) and (yy) quadrature points defined in (15).

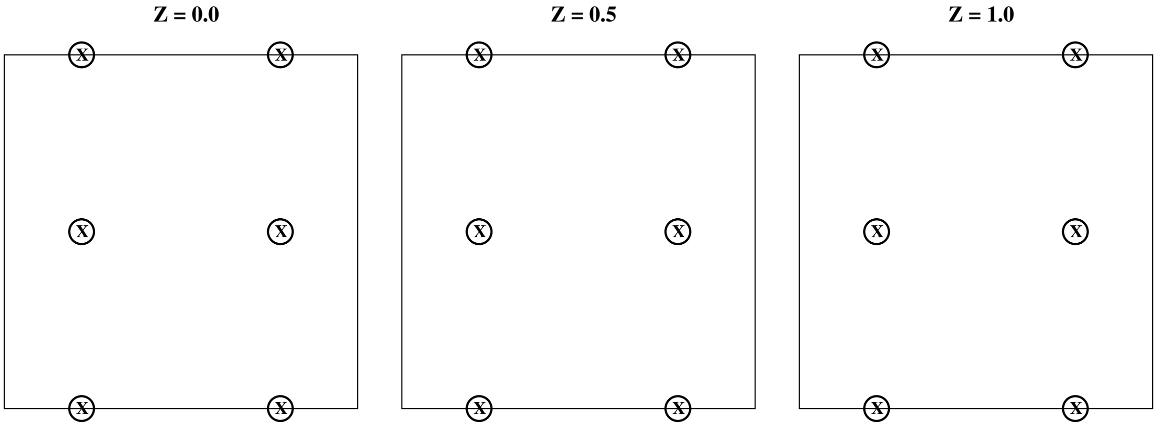


Fig. 1. Illustration of the (xx) quadrature rule being applied to an (xx) mass integral for 2nd order basis functions on the unit cube. The 3 different planes are slices for different z values, with the x's marking the interpolation points for the (x) basis functions, and the circles marking the coinciding quadrature points. Each of the 18 basis functions evaluates to 1 at one of the interpolation points, and evaluates to 0 at all of the others. This is also true for at quadrature points since they coincide with the interpolation points. If an (xx) mass integral includes two different (x) basis functions, then the approximate integral evaluates to 0, since one of the two basis function will evaluate to 0 at all of the quadrature points.

On an orthogonal element the other 6 sub-mass matrices are all zero due to the dot products in their mass integrals and the orthogonality of the basis function vectors. However, on an arbitrary element these conditions cannot be guaranteed and the off-diagonal sub-mass matrices must be constructed. Also, whether an element is orthogonal or not, a quadrature rule is needed in order to evaluate the stiffness integrals. In order to maintain consistency a quadrature rule with a similar order of accuracy $(2p-1)$ to the other 3 quadrature rules must be used. One such quadrature rule lies on the points defined by

$$\xi^{(*)}_{lmn} = (b_l, b_m, b_n) \tag{18}$$

This particular set of quadrature points has the advantage of using the interpolation property (4) to introduce more zeros into the $M^{(xy)}$ matrix with the following result

$$\begin{gathered} M^{(xy)}_{(ijk)(ijk)'} \approx \sum_{lmn} a^{(*)}_{lmn}((\alpha\circ\Phi)\mathbf{J}^{-1}\hat{\mathbf{w}}^{(x)}_{(ijk)}(\xi^{(*)}_{lmn}))\cdot(\mathbf{J}^{-1}\hat{\mathbf{w}}^{(y)}_{(ijk)'}(\xi^{(*)}_{lmn}))|\mathbf{J}| = 0 \\ \{(ijk),(ijk)' \mid k \neq k'\} \end{gathered} \tag{19}$$

This result does not lead to the formation of diagonal sub-mass matrices, however, a simple count reveals that it does lead to the elimination of $p/(p+1)$ of the elements in the sub-mass matrix (see Figure 2). Similar results apply to the other 5 sub-mass matrices to eliminate $p/(p+1)$ of their entries.

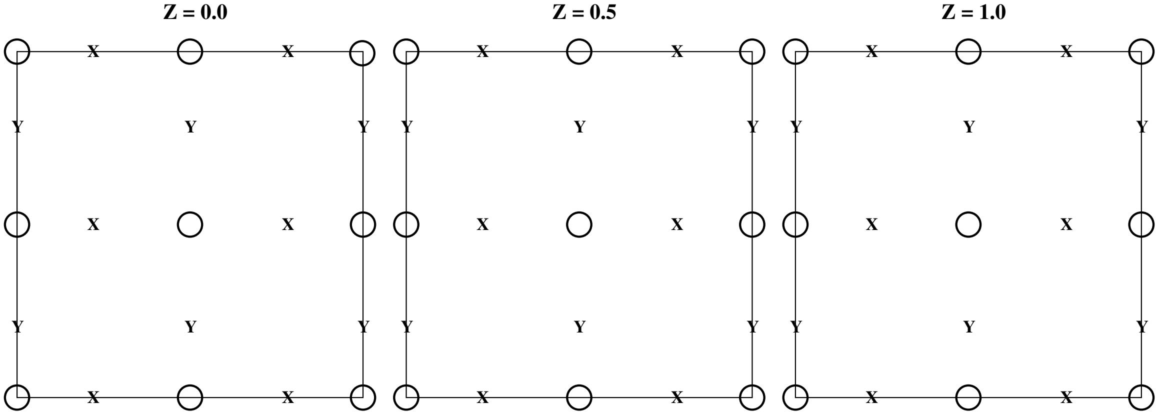
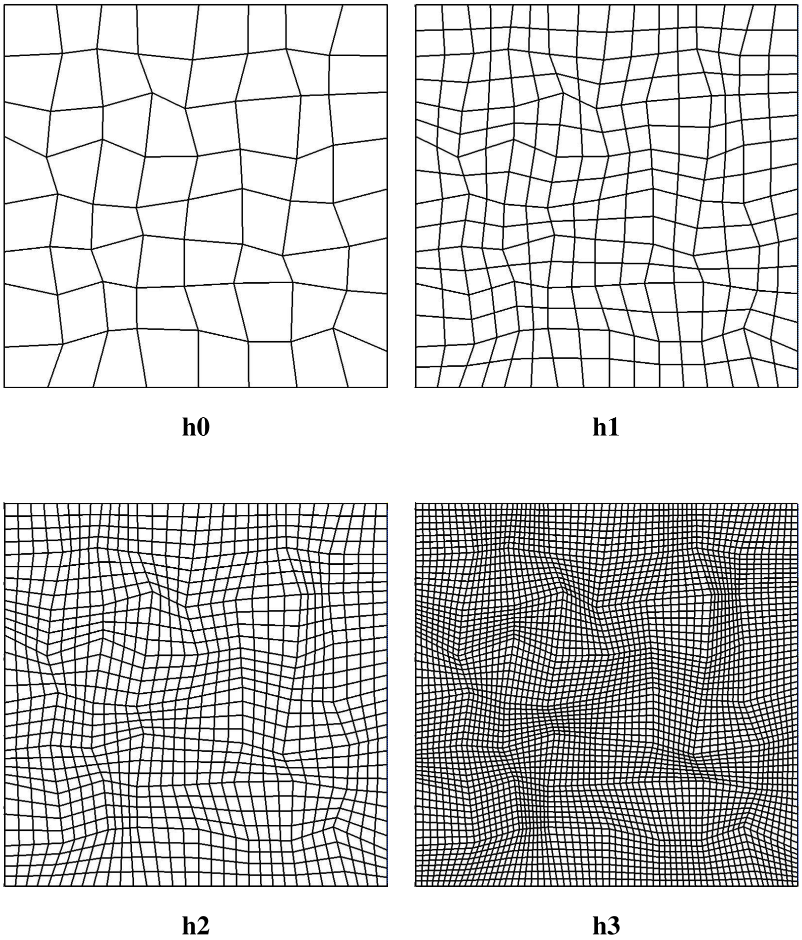


Fig. 2. Illustration of the $(*)$ quadrature rule being applied to an (xy) mass integral for 2nd order basis functions on the unit cube. The 3 different planes are slices for different z values, with the x's and y's marking the interpolation points for the (x) and (y) basis functions, and the circles marking the quadrature points. Each of the 36 basis functions evaluates to 1 at one of the x or y interpolation points, and evaluates to 0 at the two z-planes that do not contain that interpolation point. If an (xy) mass integral includes an (x) basis function from one z-plane and a (y) basis function from a different z-plane, then the approximate integral will evaluate to 0, since either the (x) or (y) basis function will evaluate to 0 at all of the quadrature points.

V. Numerical Experiments

A. Scalability Analysis

In this experiment methods are employed that are similar to those described in [16] to determine the scalability of preconditioned conjugate gradient inversion of the lumped mass matrix. In order to carry out the experiment a simple 3D, non-uniform mesh is created and mass matrices are constructed for various $h-$ and $p-$refinements of the mesh. The $h-$refinements are designed to preserve the fundamental structure of the original mesh (see Figure 3). Diagonally scaled condition numbers are computed for each of the matrices and can be found in tables Table I and Table II. The condition numbers of the mass matrices formed using full integration rules are included for comparison.



h0

h1

h2

h3

Fig. 3. Mesh refinements maintaining the original mesh structure

Refinement	Full Integration	GML Integration
h0	11.69	2.10
h1	12.15	2.18
h2	12.65	2.55
h3	13.17	3.07

TABLE I

Condition numbers for h-refinements

Basis Order	Full Integration	GML Integration
p1	11.69	2.10
p2	34.40	2.54
p3	39.53	2.97

TABLE II

CONDITION NUMBERS FOR P-REFINEMENTS

Also, the conjugate gradient method with diagonal preconditioning is applied and the number of iterations required for convergence on a solution to a tolerance of 10^{-9} can be found in Table III and Table IV. Again, the numbers for fully integrated matrices are included for comparison.

Refinement	Full Integration	GML Integration
h0	29	12
h1	31	13
h2	31	14
h3	31	16

TABLE III

# OF PCG ITERATIONS FOR H-REFINEMENTS

Basis Order	Full Integration	GML Integration
p1	29	12
p2	48	14
p3	43	15

TABLE IV

# OF PCG ITERATIONS FOR P-REFINEMENTS

This data illuminates more of the computational strengths of the GML method. For the case of $h-$refinement the diagonally scaled condition numbers appear to be nearly constant for both the fully integrated and GML matrices indicating scalability for both. However, the lumped mass matrices have condition numbers roughly $5\times$ better than the fully integrated mass matrices. This leads to a roughly $2\times$ improvement in computational cost due to the reduced number of PCG iterations required to converge to a solution.

In the case of $p-$refinement the effects are even more dramatic. The condition numbers for the fully integrated mass matrices do not appear to be constant while the GML matrices seem nearly so, indicating that only the GML matrices maintain scalability. At higher order the gap in condition numbers jumps to roughly $10\times$. This leads to a roughly $3\times$ improvement in computational cost due to the reduced number of PCG iterations required to converge on a solution.

B. Eigenmode Convergence

A significant issue encountered when modeling Maxwell's equations with the Nedelec finite element approach is the introduction of numerical dispersion error. This error causes an inaccuracy in the phase velocity of simulated waves. In simulations where the phase information of the waves is important, the dispersion error limits the effective length of time a simulation can be evolved before losing accuracy. Since an analytical dispersion analysis is intractable on unstructured computational meshes, we employ an eigenvalue analysis to quantify the dispersion.

If a time harmonic solution is assumed for a source free region in equation (1) in the absence of conductivity, we have the vector Helmholtz equation

$$\nabla \times (\mu^{-1} \nabla \times \mathbf{E}) - \omega^2 \varepsilon \mathbf{E} = 0 \tag{20}$$

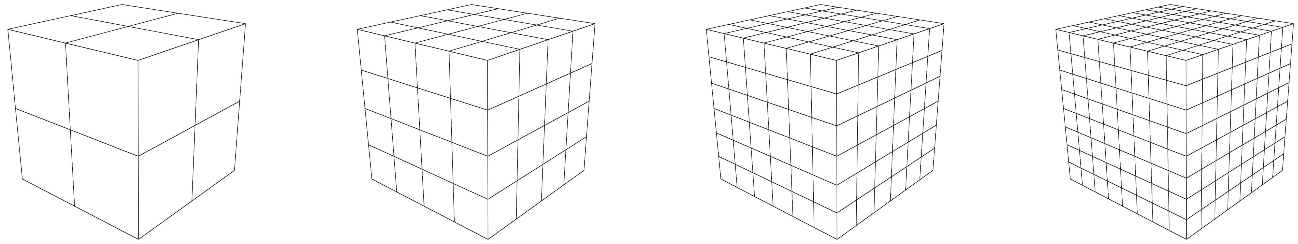


Fig. 4. Unit cube refinements used in eigenvalue convergence analysis.

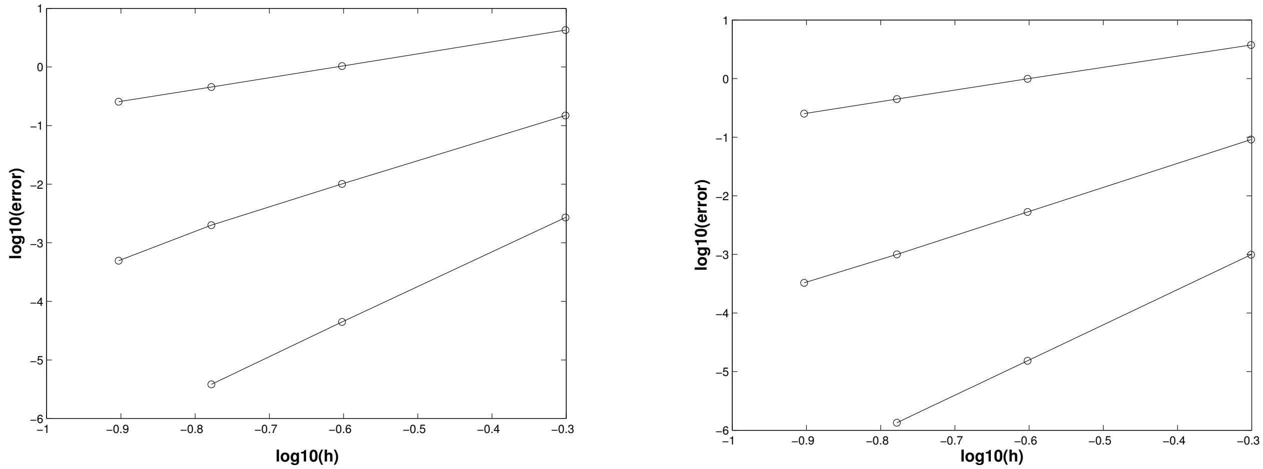


Fig. 5. Eigenvalue error for $p = 1$, $p = 2$, and $p = 3$ (lower errors correspond to higher order) with respect to mesh element size h. The left plot shows the eigenvalue convergence for mass and stiffness matrices formed using full integration, while the right plot shows the same for matrices formed using GML integration.

This equation can be cast into a generalized eigenvalue problem where the domain of the problem is a resonant cavity and the ω^2 values are the eigenmodes of that cavity. Similarly, a discrete version of this eigenvalue problem can be formed in the following manner

$$(S-\tilde{\omega}^2 M)e = \hat{0} \tag{21}$$

where the $\tilde{\omega}^2$ values approximate the eigenmodes of the cavity. Using these two eigenvalue problems the eigenmodes can be computed both analytically and computationally. Exact and approximate dispersion relations can also be introduced and combined in the following manner

$$\begin{aligned} \omega^2 &= k^2c^2 \\ \tilde{\omega}^2 &= k^2v^2 \\ \omega^2-\tilde{\omega}^2 &= k^2(c^2-v^2) \end{aligned} \tag{22}$$

where k is the wave number, c is the exact wave speed, and v is the approximate wave speed. This gives a simple relationship between eigenvalue approximation error and the dispersive phase velocity error.

In order to study this issue, the dominant mode error for the unit cube was computed using the 4 different refinements found in Figure 4 for $p = 1$, $p = 2$, and $p = 3$. Mass and stiffness matrices using both full integration and GML integration we used to compute the errors for comparison, and all of the results can be found in Figure 5.

As the data in Figure 5 shows, the levels of eigenmode error are similar for matrices created using full integration, and GML integration. This indicates that the dispersion errors are also similar for the two different integration techniques. It is interesting to note that using exact numerical integration results in over-estimating the eigenvalues (phase velocity higher than c), while the GML integration results in under-estimating the eigenvalues (phase velocity lower than c). This is consistent with results for the standard second-order elliptic eigenvalue problem [17], [18]

Finally, by measuring the slopes of the trends found in Figure 5, the order of accuracy can be found for both eigenmodes and dispersion. The slopes of the trends were found by obtaining a linear least squares fit, and the results can be found in Table V.

Basis order	Full integration order	GML integration order
p1	2.0320	1.8679
p2	4.2865	4.0724
p3	5.9608	6.0090

TABLE V

THE SLOPES OF THE TRENDS FOUND IN FIGURE 5. THESE TRENDS INDICATE ORDER $2p$ ACCURACY IN EIGENMODES AND DISPERSION WHEN USING EITHER INTEGRATION TECHNIQUE.

As expected from the results in [19], the data in Table V shows that when using the pth order Nedelec elements with full integration, a $2p$ order of accuracy is obtained. The data also shows that an accuracy of order $2p$ is a obtained when GML integration is used as well. This indicates that the GML integration technique does not significantly effect the dispersion error of the Nedelec finite element approach.

C. Eigenmodes Using a Non-Cartesian Mesh

Early finite element methods for modeling Maxwell's equations did not use the proper spaces of approximation, and therefore introduced spurious modes into their solutions. With some methods these spurious modes coincide with the proper eigenmodes when a Cartesian mesh is used, but they begin to diverge on more general meshes. Numerous analyses have shown that the use of 1-form basis functions eliminate spurious modes. However these analysis assume exact integration, and it is therefore important to investigate the effect of using approximate integration in the GML method. As spurious modes are most clearly seen in eigenvalue computations, a simple resonant cavity experiment was performed in the following manner.

As in the previous section, a resonant cavity can be approximated with the discrete eigenmode equation (21). So, the exact and approximate eigenmodes can be computed and compared to find the error. Also, it is well known that spurious modes may arise in the computational method if it does not accurately model the following grad-curl relationship

$$\nabla \times \nabla f = \hat{0} \quad \forall f \tag{23}$$

In that case the null space dimension of the generalized eigenvalue matrix $M^{-1}S$ will be too small and there will be fewer zero eigenmodes in the computation. In this case, since the gradients of functions are represented with order p 0-form (nodal) elements, the null space should have dimension equal to the number of order p 0-form degrees of freedom, save 1 for constant functions.

Our experiment was carried out on a resonant cavity in the shape of the unit cube with a chevron pattern (see Figure 6), which has a set of well known eigenmodes. The mass and stiffness matrices were formed using a full integration method and the GML method for $p = 1$, $p = 2$, $p = 3$, and $p = 4$. The dimension of the $M^{-1}S$ null space was computed in each case by counting the number of zero eigenmodes and can be found in Table VI. The errors in the first few eigenmodes were computed in each situation and can be found in Table VII and Table VIII.

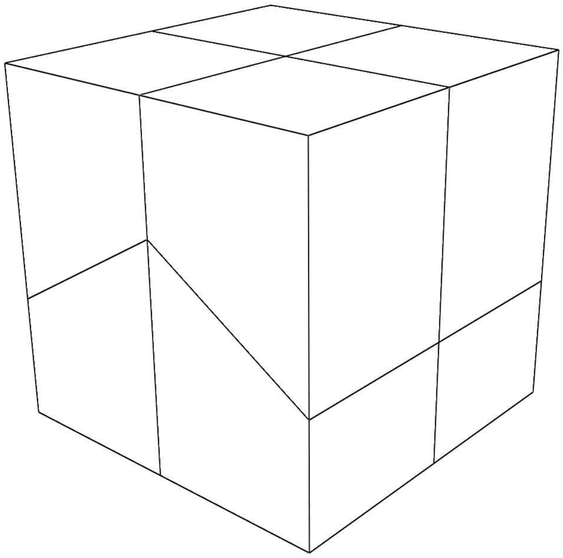


Fig. 6. The unit cube mesh w/chevron used for eigenmode analysis.

Basis order	Expected Dim.	Full Integration Dim.	GML Integration Dim.
p1	26	26	26
p2	124	124	124
p3	342	342	342
p4	728	728	728

TABLE VI

$M^{-1}S$ NULL SPACE DIMENSIONS FOR FULL AND GML INTEGRATION USING P1-P4 BASIS FUNCTIONS. A MATCH WITH THE EXPECTED VALUE INDICATES THE ABSENCE OF SPURIOUS MODES WHEN USING BOTH THE FULL INTEGRATION AND THE GML INTEGRATION SCHEME.

Distinct mode #	Exact ω^2	p1 Error	p2 Error	p3 Error	p4 Error
1	$2\pi^2$	5.3469	0.3626	0.0132	0.0003
2	$3\pi^2$	7.8166	0.4369	0.0145	0.0004
3	$5\pi^2$	17.871	9.9582	0.4296	0.0461
4	$6\pi^2$	21.539	11.572	0.4310	0.0478
5	$8\pi^2$	24.262	19.295	0.8968	0.1386

TABLE VII

EIGENMODE ERRORS FOR THE FULL INTEGRATION METHOD

Distinct mode #	Exact ω^2	p1 Error	p2 Error	p3 Error	p4 Error
1	$2\pi^2$	3.9058	0.2853	0.0185	0.0005
2	$3\pi^2$	5.6088	0.3390	0.0243	0.0006
3	$5\pi^2$	25.348	9.1023	0.7229	0.0437
4	$6\pi^2$	29.239	9.3155	0.8060	0.0447
5	$8\pi^2$	44.494	7.2889	1.4447	0.1047

TABLE VIII
EIGENMODE ERRORS FOR THE GML INTEGRATION METHOD

As Table VII and Table VIII show, the error grows with the value of the eigenmode. This is to be expected since higher eigenmodes correspond to higher frequencies which are more difficult to approximate with polynomials. The very high error values come from the low order basis functions attempting to approximate frequencies beyond their Nyquist sampling limit. However, the error can be alleviated by increasing the order of the approximation.

Also, the data shows that the matrices created using the GML integration scheme have similar levels of error in their eigenmodes to the matrices created using the full integration method. More importantly, in both cases the dimensions of the $M^{-1}S$ null spaces match their expected values. This indicates that the matrices created with the GML integration scheme can accurately model electromagnetic problems, without corrupting the solutions with spurious modes.

D. *Time Domain Coaxial Cable*

In order to get a sense of the computational savings that the GML scheme provides, a simple coaxial cable signal propagation simulation was performed. Such a cable simulation has numerous desirable qualities that make it a good diagnostic tool. As can be seen in Figure 7, the mesh is non-orthogonal, so it will test the GML method in the general case. There is a well known analytical solution (we assume a pure TEM mode) to the coaxial cable problem, so L_2 error norms can be computed and compared.

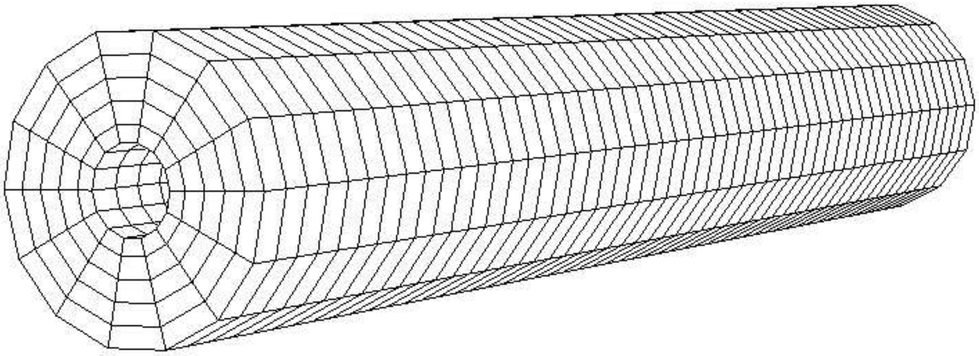


Fig. 7. View of the mesh representing the coaxial cable. In the light speed scaled coordinates, the inner radius is 2.0 light seconds, the outer radius is 6.69 light seconds, and the length of the cable is 80 light seconds. Finally, waves of 10 light seconds wavelength were fed into the cable.

In this experiment the mesh in Figure 7 was used with 2nd order basis functions utilizing both the full integration method, and the GML method to compute the matrices. For purposes of simplicity, both μ and ε were set to 1, which scales all of the coordinates by the speed of light. Sparsity plots of the mass matrices can be found in Figure 8. Perfect Electrical Conductor (PEC) boundary conditions were applied to the inner and outer surfaces, an Absorbing Boundary Condition (ABC) that is exact for the TEM mode was applied to one of the ends, and the analytical solution was used to provide the proper voltage source for the other end. In both cases a timestep of $0.1s$ was used. It is worth noting that a stability analysis similar to that found in [20] reveals that a maximum timestep of $0.19s$ could have been used in the full integration case, while a maximum timestep of $0.27s$ could have been used in the GML integration case. The ability to take larger timesteps in this case adds to the significant computational savings of the GML scheme. However, in this case it is advantageous to use the same timestep for accurate comparisons of the results. This experiment led to the results found in Table IX.

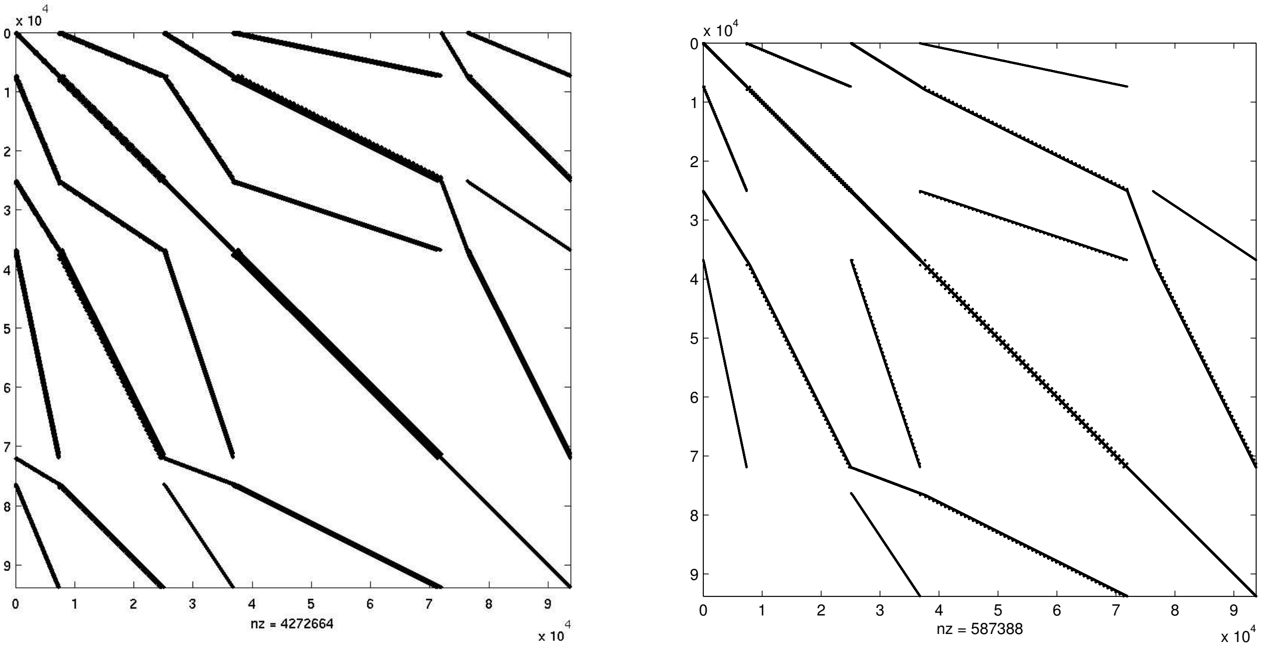


Fig. 8. Sparsity plots of the coaxial cable mass matrices using a full integration scheme and the GML integration scheme

	Full Integration	**GML Integration**
No. Unknowns	93,772	93,772
No. Timesteps	1,000	1,000
Avg. No. Iterations / Timestep	43	10
Avg. CPU Time / Timestep	3.36 s	0.94 s

TABLE IX

Comparison of computational cost for coax simulation using $p = 2$ basis function with a full integration method and the GML integration scheme.

The data in Table IX shows a roughly $3.5\times$ improvement in overall computational cost when using the GML method. This includes all of the overhead of applying the boundary conditions and constructing the rhs of the matrix equation at each timestep. If the extra computation savings of using the larger timestep is figured in, a total improvement of $5\times$ is expected.

In order to compare accuracy between methods, data on the L_2 error in the simulations was also collected and displayed in Figure 9.

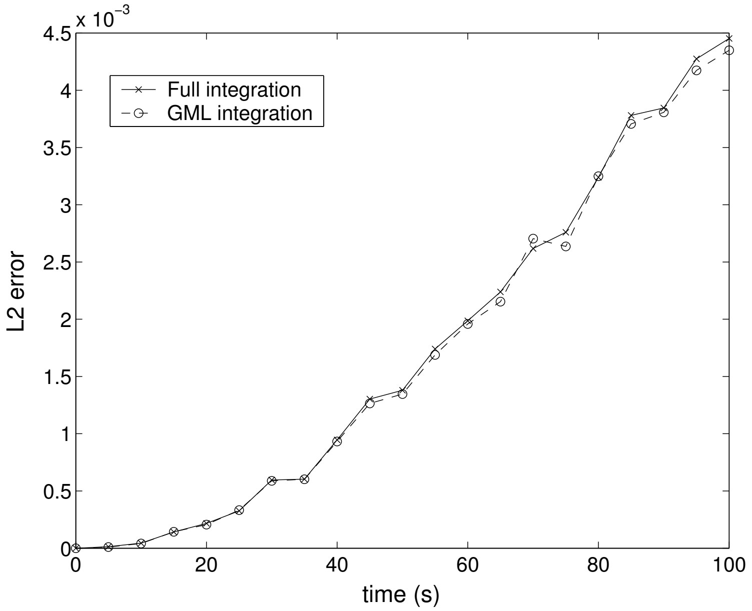


Fig. 9. L2 error in the coaxial cable simulation.

As the data in Figure 9 shows, the L_2 error when error using the GML method is nearly identical to the L_2 error when using the full integration method. The L_2 error in both cases is of course rising due to the phase velocity error discussed in a previous section.

E. RF Signal Propagation in a Random Rough Surface Environment

In this example, we simulate the propagation of a $200MHz$ RF signal in a random rough surface environment designed to represent a tunnel in earth. Our goal here is to demonstrate the performance of the GML method in a large scale, parallel computational environment. Modern cell-phones and 2-way radios used by the military, border patrol, first responders, etc. were not designed to operate in a cave or tunnel. If the electromagnetic properties of the tunnel could be characterized (dissipation, dispersion, fading, etc.) a more robust communication system could be designed that could operate in that environment, hence full wave EM simulations of propagation in this type of environment are very useful.

In this example simulation we choose a tunnel with an average diameter of $2m$ and a total length of $50m$. The random rough surface was generated as follows. Step 1: Generate a cylindrical surface of appropriate radius and length, Step 2: Add a random perturbation of specified standard deviation to the surface, Step 3: Smooth the random surface (low pass filter) to introduce a surface correlation of a given length, Step 4: Generate a 3D Cartesian mesh, where the electrical conductivity of each element depends upon whether the element is inside the random surface (air) or outside the random surface (earth). For mesh elements that straddle the random surface, a volume-fraction is used to determine the conductivity. The computational mesh consists of $388,800$ hexahedral elements and is shown in Figure 10. Note that the portion of the mesh representing the air has been removed to illustrate the random rough surface. Because the mesh is structured and orthogonal (i.e. Cartesian), the GML method will yield a diagonal mass matrix regardless of the polynomial degree of the basis functions, and will therefore lead to drastic improvement in computational performance for time dependent calculations.

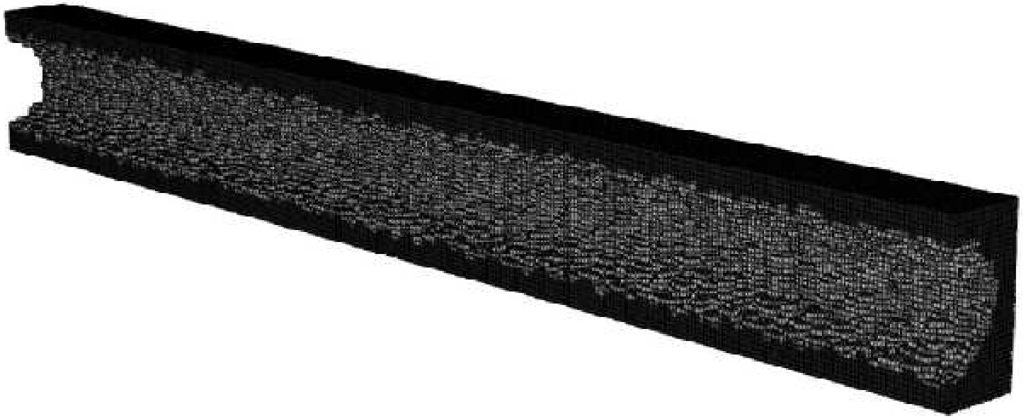


Fig. 10. Sliced view of Cartesian mesh representing a tunnel in earth.

A dipole antenna is placed at one end of the tunnel, modeled by applying a time dependent current source to one of the mesh edges, while a simple absorbing boundary condition is placed at the other. A PEC boundary condition is applied to the remaining walls of the cave to fully define the problem. The simulation is performed using both standard lowest order basis functions and higher order $p = 2$ basis functions. In Table X we compare the computational performance statistics for the $p = 1$ case with full integration and GML integration. Note that without using the GML technique, the mass matrix requires on average 24 PCG iterations to achieve a residual error tolerance of 10^{-9}. When using the GML technique on Cartesian grids, the mass matrix is made diagonal and inversion is therefore trivial. The average CPU time used per simulation time step is an order of magnitude less when using the GML method for a total speed up of roughly $10\times$. Note that the total CPU time per step in this measurement includes more than just the linear solve time, it also includes the CPU time required for the application of boundary conditions, current sources, and the time integration. In Table XI we compare the computational performance statistics using high order $p = 2$ basis functions on the same Cartesian mesh. The total number of electric field unknowns has increased by an order of magnitude and to satisfy the Courant stability condition, the discrete time step must be cut in half. Again, note that without using the GML technique, the mass matrix now requires on average 31 PCG iterations to achieve a residual error tolerance of 10^{-9}. By using the GML technique, the mass matrix is once diagonal resulting in a speed up of roughly $10\times$. In Figure 11 we show plots of the base 10 log of the computed electric field vector magnitude at the final simulation time step along a "line-out" portion down the center of the cave for the $p = 2$ high order case. Note that the results with full integration and with GML integration are essentially identical.

	p1 Full Integration	**p1 GML Integration**
No. Unknowns	1,212,564	1,212,564
No. Parallel CPUs	12	12
No. Timesteps	2,100	2,100
Avg. No. Iterations / Timestep	24	1
Avg. CPU Time / Timestep	2.240 s	.210 s

TABLE X

COMPARISON OF COMPUTATIONAL COST FOR CAVE SIMULATION USING THE $p = 1$ BASIS FUNCTIONS WITH GML INTEGRATION AND FULL INTEGRATION.

	p2 Full Integration	p2 GML Integration
No. Unknowns	9,515,112	9,515,112
No. Parallel CPUs	128	128
No. Timesteps	4,200	4,200
Avg. No. Iterations / Timestep	30	1
Avg. CPU Time / Timestep	4.700 s	0.410 s

TABLE XI

COMPARISON OF COMPUTATIONAL COST FOR CAVE SIMULATION USING THE $p = 2$ BASIS FUNCTIONS WITH GML INTEGRATION AND FULL INTEGRATION.

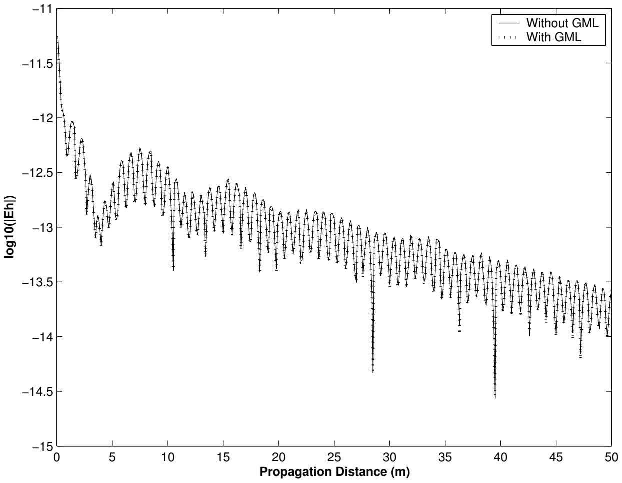


Fig. 11. Comparison of cave simulation results with full integration and GML integration. Note that the results are essentially identical.

VI. CONCLUSIONS

We have presented a generalized mass lumping scheme to be used in the vector finite element solution of Maxwell's equations. The methods uses the now standard Nedelec 1-form polynomial basis functions, with specially chosen interpolation points and specially chosen inexact quadrature rules. The method is designed to yield diagonal mass matrices for orthogonal meshes, and highly-sparse and well-conditioned matrices for non-orthogonal unstructured meshes. This GML method recuces the computation cost of solutions by an order of magnitude and maintains scalability for both $h-$ and $p-$ refinements. The scheme also has levels of dispersion error and L_2 error similar to the standard full integration schemes, making it suitable for use in computational electromagnetics.

REFERENCES

[1] R. Rieben, D. White, and G. Rodrigue. High order symplectic integration methods for finite element solutions to time dependent maxwell equations. *IEEE Trans. Ant. Prop.*, 52(8):2190–2195, August 2004.

[2] D. A. White. Solution of capacitance systems using incomplete Cholesky fixed point iteration. *Comm. Numer. meth. Engng*, 15:373–380, 1999.

[3] M. S. Jensen. High convergence order finite elements with lumped mass matrix. *Internat. J. Numer. Methods Engrg.*, 39(2):1879–1888, 1996.

[4] O. C. Zienkiewicz and R. L. Taylor. *The Finite Element Method.* McGraw-Hill, London, UK, fourth edition edition, 1994.

[5] S. Benhassine, W.P. Jr. Carpes, and L. Pichon. Comparison of mass lumping techniques for solving the 3d maxwell's equations in the time domain. *IEEE Trans. Mag.*, 36(4):1548–1552, 2000.

[6] D. A. White. Orthogonal basis functions for time domain finite element solutoion of the vector wave equation. *Commun. Numer. Meth. Engng.*, 35(3):1458–1461, 1999.

[7] D. Jiao and J. Jin. Three-dimensional orthogonal vector basis functions for time-domain finite element solution of the vector wave equation. *IEEE Trans. Ant. Prop.*, 51(1):59–66, 2003.

[8] D. A. White. Solution of capacitance systems using incomplete cholesky fixed point iteration. 15(5):373–380, 1999.

[9] J. C. Nédélec. A new family of mixed finite elements in R3. *Numer. Math.*, 50:57–81, 1986.

[10] G. Cohen and P. Monk. Gauss point mass lumping schemes for Maxwell's equations. *Num. Meth. Part. Diff. Eqs.*, 14(1):63–88, 1998.

[11] G. C. Cohen. *Higher Order Numerical Methods for Transient Wave Equations*. Springer-Verlag, 2001.

[12] J. C. Nédélec. Mixed finite elements in R3. *Numer. Math.*, 35:315–341, 1980.

[13] A. Fisher, G. Rodrigue, and D. White. A generalized mass lumping scheme for Maxwell's wave equation. In *Proceedings of the 2004 IEEE International Antennas and Propagation Symposium*, volume 2, pages 1507–1510, Monterey, CA, June 2004.

[14] R. Rieben, D. White, and G. Rodrigue. Improved conditioning of finite element matrices using new high order interpolatory bases. *IEEE Trans. Ant. Prop.*, December 2004. in press.

[15] P. Castillo, J. Koning, R. Rieben, and D. White. A discrete differential forms framework for computational electromagnetics. *Comp. Model. Engen. Sci.*, 5(4):331–346, 2004.

[16] G. Rodrigue, D. White, and J. Koning. Scalable preconditioned conjugate gradient inversion of vector finite element mass matrices. *J. Comput. Appl. Math.*, 123:307–321, 2000.

[17] M. G. Armentano and R. G. Duran. Mass lumping or not mass lumping for eigenvalue problems. *Numer. meth. PDE's*, 19(5):653–664, 2003.

[18] U. Banerjee and J. Osborn. Estimation of the effect of numerical integration in finite element eigenvalue approximation. 56:735–762, 1990.

[19] M. Ainsworth. Dispersive properties of high-order Nedelec/edge element approximation of the time-harmonic Maxwell equations. *Philisophical Transactions of the Royal Society of London*, 362(1816):471–491, 2004.

[20] G. Rodrigue and D. White. A vector finite element time-domain method for solving maxwell's equations on unstructured hexahedral grids. *SIAM J. Sci. Comp.*, 23(3):683–706, 2001.