LAWRENCE
LIVERMORE
NATIONAL
LABORATORY

LLNL-PROC-464197

Using the Schur Complement to Reduce Runtime in KULL's Magnetic Diffusion Package

T. A. Brunner, T. V. Kolev

December 17, 2010

Nuclear Explosives Code Developers' Conference 2010
Los Alamos, NM, United States
October 18, 2010 through October 22, 2010

Disclaimer

Using the Schur Complement to Reduce Run time in KULL's Magnetic Diffusion Package (U)

[1]Thomas A. Brunner and [2]Tzanio V. Kolev
[1]*Weapons and Complex Integration*
[2]*Center for Applied Scientific Computing*
Lawrence Livermore National Laboratory, Livermore, CA

Recently a Resistive Magnetohydrodynamics (MHD) package has been added to the KULL code. In order to be compatible with the underlying hydrodynamics algorithm, a new sub-zonal magnetics discretization was developed that supports arbitrary polygonal and polyhedral zones. This flexibility comes at the cost of many more unknowns per zone—approximately ten times more for a hexahedral mesh. We can eliminate some (or all, depending on the dimensionality) of the extra unknowns from the global matrix during assembly by using a Schur complement approach. This trades expensive global work for cache-friendly local work, while still allowing solution for the full system. Significant improvements in the solution time are observed for several test problems. (UNC)

Prepared for Proceedings of the NECDC 2010

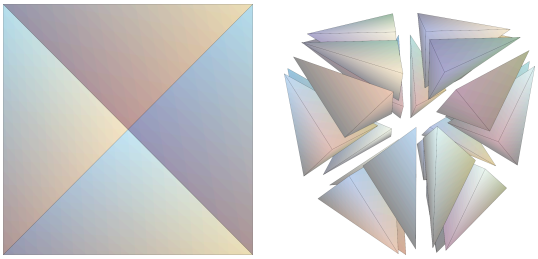


Figure 1: The sub-zonal mesh in 2D (left) and 3D (right). In 3D, 24 tetrahedral sub-elements are added to each hexahedron.

Introduction

A resistive magnetohydrodynamics (MHD) package [3] has recently been added to the KULL code [10]. KULL supports unstructured, arbitrary polygonal and polyhedral mesh zones. The MHD algorithm is one of the most tightly coupled physics packages to the underlying hydrodynamics discretization in the code. KULL employs a compatible algorithm [5, 2] that uses sub-zonal corner pressures. The extension of the hydro algorithm to support magnetic fields also needs sub-zonal magnetic stresses.

We have chosen to discretize our MHD equations on the sub-zonal elements called "sides" in KULL that consist of the triangles formed in 2D by two consecutive nodes and the zone center and of the tetrahedrons in 3D formed by two neighboring nodes, the face center, and the zone center, as show in Figure 1. We will use the term "side" to denote either a triangle or tetrahedron, depending on the dimensionality of the problem. Discretizing on the side sub-mesh allows us to support fully arbitrary polygonal and polyhedral zone shapes, as well as to match the sub-zonal characteristics of the hydrodynamics algorithm.

The evolution of the magnetic field due to diffusion is described by

$$\nabla \times \frac{\Delta t}{\mu_0} \nabla \times \mathbf{E} + \overline{\overline{\sigma}} \cdot \mathbf{E} = \nabla \times \frac{\mathbf{B}^n}{\mu_0} \quad \text{and} \quad (1)$$

$$\mathbf{B}^{n+1} = \mathbf{B}^n - \Delta t \nabla \times \mathbf{E}. \quad (2)$$

In order to ensure that $\nabla \cdot \mathbf{B} = 0$ always, we choose to solve for the electric field integrated along the edges in the mesh, or

$$E_e = \int_e \mathbf{E} \cdot d\mathbf{l}. \quad (3)$$

Instead of standard nodal finite elements, we use the lowest order Nédélec (edge) finite elements [9], where the vector edge basis function has an integral of one along its edge, and zero along any other edge in the mesh. This ensures that $\nabla \cdot \mathbf{B} = 0$ to machine precision in our simulation. Using the edge basis functions, we apply a standard Galerkin method to derive a matrix equation,

$$\mathbf{A}\mathbf{x} = \mathbf{y}. \quad (4)$$

The number of unknowns in the vector $\mathbf{x}$ is the number of edges in the sub-zonal side mesh, which can be considerably larger than the number of edges in the original zonal mesh.

The increased unknown count relative to a standard zonal discretization may seem high, but the benefits outweigh the costs. Discretizing the magnetic diffusion equation on the sub-zonal mesh allows us to support an unstructured, arbitrary polygonal or polyhedral mesh. More importantly, it is necessary for stability when coupling with the underlying hydrodynamics algorithm. The rest of this paper is focused on how we significantly speed up the solution of Eq. 4 by trading extra local work during the matrix construction for expensive work during the global matrix solve.

Reduced Global Matrix

Instead of solving Eq. 4 directly for the full number of unknowns in the problem, we perform a two-step solve. As we assemble the matrix, we preform

some local dense linear algebra to temporarily eliminate some unknowns to form a smaller global matrix. After solving the global matrix, we then solve for the unknowns we eliminated, recovering the full solution as if we solved the full problem.

Reduced Matrix Assembly and Full Solution
A finite element discretization can be specified by a set of elements $\{g\}$, with corresponding degrees of freedom (dofs) $\{\mathbf{x}_g\}$, and local element matrices $\{\mathbf{A}_g\}$, which are assembled to form $\mathbf{A}$. With that definition in mind, we consider the following 6-step process for the solution of the full global problem, Eq. 4:

1. Gather sides into groups. Each original side is in exactly one group. How to choose these groups will be discussed in the next section.

2. Compute matrix and right hand side contributions to the full matrix for the sides in this group.

$$\mathbf{A}_g = \sum_s \mathbf{A}_s, \quad \mathbf{b}_g = \sum_s \mathbf{b}_s \tag{5}$$

3. Determine the interior and boundary degrees of freedom of the group.

$$\begin{pmatrix} \mathbf{A}_{g,ii} & \mathbf{A}_{g,ib} \\ \mathbf{A}_{g,bi} & \mathbf{A}_{g,bb} \end{pmatrix} \begin{pmatrix} \mathbf{x}_{g,i} \\ \mathbf{x}_{g,b} \end{pmatrix} = \begin{pmatrix} \mathbf{y}_{g,i} \\ \mathbf{y}_{g,b} \end{pmatrix}, \tag{6}$$

   where i stands for the "interior" unknowns that are eliminated, and b corresponds to the "boundary" unknowns.

4. The global matrix element contributions for the interior unknowns are completely contained in this group sub-matrix. We can then eliminate them from the group matrix, and calculate their influence on the boundary unknowns, which are coupled to other groups. The reduced group matrices and right hand sides are computed using a Schur complement, and summing them into the global matrix using

$$\mathbf{S}_g = \mathbf{A}_{g,bb} - \mathbf{A}_{g,bi}\mathbf{A}_{g,ii}^{-1}\mathbf{A}_{g,ib}, \tag{7}$$

$$\mathbf{y}_g = \mathbf{y}_{g,b} - \mathbf{A}_{g,bi}\mathbf{A}_{g,ii}^{-1}\mathbf{y}_{g,i}, \tag{8}$$

$$\mathbf{S} = \sum_g \mathbf{S}_g, \text{ and } \mathbf{y} = \sum_g \mathbf{y}_g. \tag{9}$$

   The inversion of $\mathbf{A}_{g,ii}$ is a direct inversion, but is computationally inexpensive since it is usually less than a 6×6 matrix, and dense linear algebra of this form is very efficient on modern compute cores.

5. Solve the reduced global linear system for all the coupled boundary unknowns from the groups,

$$\mathbf{S}\mathbf{x}_b = \mathbf{y}. \tag{10}$$

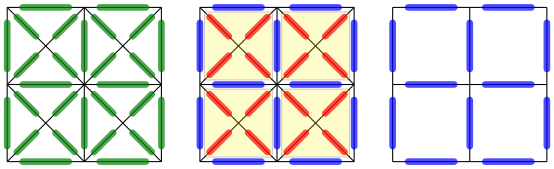
6. Locally compute the solution for the interior edges, finding the values of the unknowns eliminated from the global matrix, using

$$\mathbf{x}_{g,i} = \mathbf{A}_{g,ii}^{-1}\left(\mathbf{b}_{g,i} - \mathbf{A}_{g,ib}\mathbf{x}_{g,b}\right). \tag{11}$$

It is important to note that because $\mathbf{A}_{ii}$ is a block-diagonal sparse matrix, the final matrix $\mathbf{S}$ is also sparse, and easy to assemble in an zone by zone manner, much like if we assembled $\mathbf{A}$ instead.

The output of the algorithm is an induced finite element discretization on the reduced grid with elements that are the groups, degrees of freedom $\mathbf{x}_b$, and element matrices $\{\mathbf{S}_g\}$. We emphasize that this provides a consistent way of introducing a finite element discretization on any set of reduced elements, including geometries without a natural reference element. This is similar to the Piecewise Linear finite elements [11, 1], but they assume certain averaging rules for the eliminated unknowns, while in contrast we retain the unknowns eliminated from the global solve as independent variables.

We use the *hypre* library [7] to perform the linear solve in Eq. 10. Both the original matrix and



(a) Full triangulated mesh

(b) Reduced elements (yellow) and interior unknowns (red) identified

(c) Remaining (reduced) elements and degrees of freedom (blue)

Figure 2: In RZ the full triangular grid has edge degrees of freedom. The reduced elements are simply the quadrilateral elements. The final reduced mesh has the same number of degrees of freedom and sparsity pattern as the original quadrilateral grid.

the reduced matrix can have large null spaces (or near-null spaces) [4]. Specialized preconditioners, such as BoomerAMG [6] and the Auxiliary-space Maxwell Solver (AMS) [8], are needed to get good performance from the linear solvers.

Identifying Eliminated Edges

In 2D-XY coordinates, our edge unknowns are out of the plane of the mesh, and degenerate into nodes. We form groups of sides that correspond to the original mesh zones. The interior unknown is the node at the center of the zone. In axisymmetric RZ coordinates, we again identify the zone-interior sides as the ones to group and the zone-interior edges as the ones to eliminate, as show in Figure 2. This leads to a reduced matrix $\mathbf{S}$ that is the same size as the standard quad discretization and has the same sparsity pattern, but it has different matrix elements.

In three dimensions, choosing the sides that form a mesh-zone is not ideal. The number of nonzeros in the reduced matrix is approximately 2.5 times the number of nonzeros as the full matrix. It is better to group the sides that touch the faces of the mesh zones, and to eliminate the edges that correspond to the interior of the set of sides that touch that face,

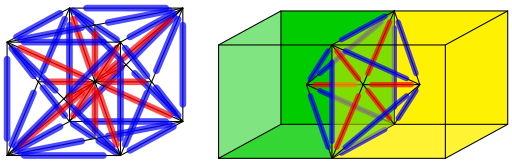


Figure 3: Hexahedral (left) and octahedral (right) reduced elements in 3D. The interior degrees of freedom, which will be eliminated, are highlighted in red. The remaining reduced degrees of freedom are colored in blue.

as shown in Figure 3.

If we consider an infinite quadrilateral or hexahedral mesh, we can estimate the size of the original matrix $\mathbf{A}$ and the reduced matrix for *XY*, *RZ*, and *XYZ* geometries as a function of the number of zones in Table I. Again, note that for the 3D case of eliminating the hexahedral zone interior edges the number of nonzeros in the reduced matrix is, in fact, larger than the original matrix. This increases the run time instead of reducing it.

Table I: Asymptotic estimates for the number of rows, n_{rows}, and non-zeros, n_{nz}, in the original and reduced matrices in each geometry, where $\mathcal{N}_z$ is the number of zones. In 3D, two different eliminations are shown for the hexahedral-zone interior, $\mathbf{S}_H$, and the face-based octahedron, $\mathbf{S}_O$.

matrix	n_{rows} (reduction)	n_{nz} (reduction)
$\mathbf{A}_{XY}$	$2\,\mathcal{N}_z$	$14\,\mathcal{N}_z$
$\mathbf{S}_{XY}$	$\mathcal{N}_z$ ($\times 2$)	$9\,\mathcal{N}_z$ ($\times 1.6$)
$\mathbf{A}_{RZ}$	$6\,\mathcal{N}_z$	$30\,\mathcal{N}_z$
$\mathbf{S}_{RZ}$	$2\,\mathcal{N}_z$ ($\times 3$)	$14\,\mathcal{N}_z$ ($\times 2.1$)
$\mathbf{A}$	$29\,\mathcal{N}_z$	$461\,\mathcal{N}_z$
$\mathbf{S}_H$	$15\,\mathcal{N}_z$ ($\times 1.9$)	$1107\,\mathcal{N}_z$ ($\times 0.4$)
$\mathbf{S}_O$	$11\,\mathcal{N}_z$ ($\times 2.6$)	$335\,\mathcal{N}_z$ ($\times 1.4$)

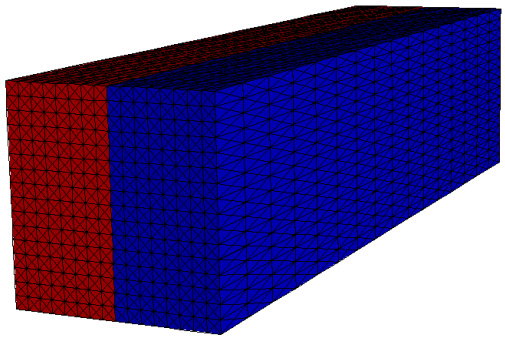


Figure 4: The Box problem is a simple test problem that we use to test the effects of varying solver parameters, material properties, and aspect ratios. The mesh, with sub-zones, is shown with a high conductivity region shown in red ($\sigma_c = 1$) and a varying low conductivity shown in blue ($0 \leq \sigma_{nc} \leq 1$).

Results

We now turn our attention to testing the speed and robustness of the proposed algorithm on two test problems. In the following experiments we use the Conjugate Gradient (CG) Krylov solver with the BoomerAMG and AMS preconditioners from the *hypre* library applied in the *XY* and *RZ/3D* cases, respectively. The tests were run on the multi-core cluster Hera at Lawrence Livermore National Laboratory (LLNL).

The Box Problem

We start with a simple diffusion problem posed on a structured box in *3D*, see Figure 4, with the *XY* and *RZ* cases corresponding to the front and the top sides, respectively. The box is split in two parts, with the conductivity σ varying between $\sigma_c = 1$ in the material half of the domain and $0 \leq \sigma_{nc} \leq 1$ in the non-conducting (void) half. The mesh is initially uniform, but we stretch it to test the dependence on the aspect ratio. For this problem, we take $\Delta t/\mu = 10^{-3}$ and use convergence tolerance of 10^{-10} in CG.

We explore the run-time behavior in *XY* with respect to increasing aspect ratio $1/\varepsilon$ in Table II. We set $\sigma_{nc}/\sigma_c = 0$ and report the number of CG iterations (n_{it}) as well as the combined time spent in the solver setup and solution phases (t_{solver}). The remaining time, including the matrix assembly as well as the elimination and the recovery of the internal degrees of freedom, is denoted by (t_{assemble}). For all of these quantities we present the data for the full and reduced global matrix solve in the format "original/reduced". Finally, we compute and report the total run time speedup of the computational cycle due to the reduction.

Table II: Comparison of overall solution performance for the Box problem in *XY* while varying the aspect ratio $1/\varepsilon$. The results are reported in the original/reduced format.

$1/\varepsilon$	n_{it}	t_{assemble}	t_{solver}	speedup
1	10/ 10	0.29/0.23	0.46/0.25	$\times$1.6
4	12/ 8	0.28/0.21	0.38/0.14	$\times$1.8
16	12/ 8	0.28/0.22	0.36/0.15	$\times$1.8
64	11/ 7	0.28/0.22	0.34/0.14	$\times$1.7
256	11/ 6	0.27/0.22	0.33/0.12	$\times$1.8
1024	11/ 7	0.28/0.23	0.33/0.15	$\times$1.6
4096	11/ 7	0.28/0.23	0.33/0.15	$\times$1.6

From Table II we see that the AMG-CG solver performs better on the reduced problem, both in terms of number of iterations and time. Even when the number of iterations is the same ($\varepsilon = 1$) there is still a factor of 1.6 speedup (1.8 in the solver). The speedup factor is nearly constant for all aspect ratios. Note also the interesting fact that, even with the extra work of inverting the local $\mathbf{A}_{ii}$ and the recovery of $\mathbf{x}_i$, the assemble time in Table II is always less for the reduced problem. This is a trend in all of our results. We suspect this is due to the cost of moving data from main memory to the processors. Each element in the global matrix is only used

once per solver iteration, while the matrix elements contained in $\mathbf{A}_{ii}$ are used many times in the local inversion process from the processor's cache.

Next, we consider similar tests in the *RZ* case, where $\sigma_{nc}/\sigma_c = 0$ and we use the the pure void solution procedure from [4]. In other words, *we apply AMS directly to the Schur complement of a singular matrix with a large kernel*.

The timing results, presented in Table III, show that the convergence deteriorates due to vanishing coefficients close to the axis of rotation, but overall the reduced AMS solver significantly outperforms the solver applied directly to $\mathbf{A}$. In particular, for the problem with the worse aspect ratio, we get more than a factor of 45 speedup in the reduced solver leading to more than 39 times total simulation speedup.

Table III: Comparison of overall solution performance for the Box problem in *RZ* and varying aspect ratio ε. The results are reported in the original/reduced format.

$1/\varepsilon$	n_{it}	t_{assemble}	t_{solver}	speedup
1	10/10	1.48/0.84	13.5/3.91	$\times$3.2
4	10/ 8	1.77/0.75	13.7/3.00	$\times$4.1
16	28/ 7	1.80/0.80	32.4/2.89	$\times$9.2
64	84/ 7	1.49/0.75	69.5/2.51	$\times$21.7
256	216/14	1.57/0.76	194./4.43	$\times$37.6
1024	594/22	1.74/0.74	451./6.25	$\times$64.7
4096	694/21	0.94/0.75	252./5.61	$\times$39.8

Finally, we consider the Box problem tests in 3D. In Table IV we investigate both regular AMS for conductivity jump of four orders of magnitude, as well as the robust AMS version for the pure void case. We note that there is a little difference between these cases in terms of solver performance (except that the pure void solver is a bit slower). This trend is typical for all the experiments we have run. Looking at the iteration counts in Table IV, we see that the convergence deteriorates significantly on stretched grids. There is a significant improvement due to the reduction, with speedup factors between 2 and 4.

Table IV: Comparison of overall solution times for the Box problem in *3D* with varying aspect ratio ε. The results for two different conductivity ratios are reported in the original/reduced format.

$1/\varepsilon$	n_{it}	t_{assemble}	t_{solver}	speedup
$\sigma_{nc}/\sigma_c = 10^{-4}$				
1	9/ 8	6.58/5.04	40.3/17.3	$\times$2.1
2	9/ 8	7.34/5.14	47.6/16.1	$\times$2.6
4	16/ 9	7.10/5.07	67.6/16.5	$\times$3.5
8	29/ 15	7.71/5.15	111./23.8	$\times$4.1
16	49/ 26	7.40/5.15	178./37.1	$\times$4.4
32	79/ 42	8.15/5.11	262./55.1	$\times$4.5
64	121/ 66	7.83/4.95	372./85.1	$\times$4.2
128	180/107	6.66/5.23	546./138.	$\times$3.8
$\sigma_{nc}/\sigma_c = 0$				
1	9/ 8	6.30/5.39	53.8/25.1	$\times$2.0
2	9/ 8	5.88/5.32	51.8/23.4	$\times$2.0
4	16/ 9	6.15/5.26	72.6/24.8	$\times$2.6
8	29/ 15	5.95/5.23	117./32.5	$\times$3.3
16	50/ 26	6.41/5.30	190./54.3	$\times$3.3
32	79/ 42	6.32/5.31	283./79.2	$\times$3.4
64	122/ 66	6.02/5.29	440./122.	$\times$3.5
128	177/103	6.60/5.30	657./187.	$\times$3.4

Coaxial Conductors

Next we applied the matrix reduction to a problem that mocks up the conductivity variation seen in Z-pinch simulations. The domain is a quarter of four concentric cylinders with different conductivi-

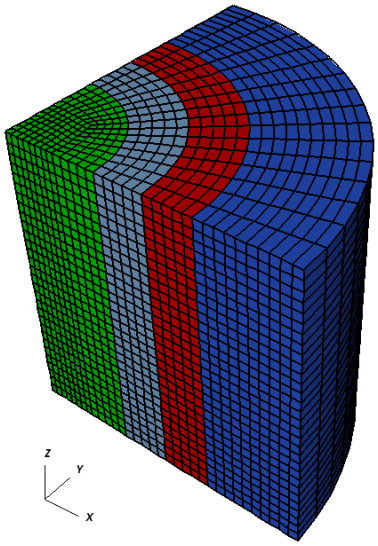


Figure 5: An idealized test problem mocks up the conductivity jumps seen in Z-pinch simulations with four regions of varying conductivity.

ties $\sigma = \{10^{-2}, 10^{-8}, 10^{-2}, 0\}$ from the inside out. The mesh and an approximate solution are shown in Figure 5. The *XY* and *RZ* cases correspond to the top and front sides of the 3D domain. Note that the jumps in σ and the pure void outer region make this problem's (near-)null space very challenging, and AMS is required for its robust solution.

In this test, we perform a weak scalability test, increasing the mesh refinement by a factor of two each step, and increasing the number of processors proportional to the total number of elements in the problem. Our goal is not to show the full scalability of the methods but rather to demonstrate their relative performance on large problems.

We report the number of processors used (n_{p}), the assembly time t_{assemble}, the AMS setup time t_{setup}, and the AMS-CG solve time t_{solve} as well as the total simulation speedup. The results for all three geometries (*XY*, *RZ*, and *3D*) are shown in Table V. Note that not only are the total solve times reduced, but the matrix assembly time is also reduced. More

Table V: The matrix sizes, iteration counts, and timings for the Coaxial problem in all three geometries (*XY*, *RZ*, and *3D*) show that the benefits of the matrix reduction extend to more realistic problems. The results are reported in the original/reduced format.

n_{p}	t_{assemble}	t_{setup}	t_{solve}	speedup
XY results				
1	0.13/0.11	0.07/0.03	0.21/0.08	×1.8
4	0.14/0.13	0.09/0.05	0.23/0.09	×1.6
16	0.15/0.12	0.12/0.08	0.36/0.16	×1.7
64	0.21/0.14	0.35/0.19	0.79/0.29	×2.1
RZ results				
1	0.20/0.12	0.21/0.08	0.57/0.27	×2.1
4	0.20/0.11	0.34/0.15	0.84/0.42	×2.0
16	0.22/0.13	0.52/0.26	1.42/0.62	×2.1
64	0.23/0.14	1.05/0.63	2.13/1.17	×2.0
3D results				
1	3.66/2.59	10.0/3.20	23.1/6.83	×2.9
8	4.08/2.80	32.4/6.95	53.2/10.7	×4.4
64	4.37/3.00	73.1/16.9	89.7/20.6	×4.1
512	4.53/3.22	122./41.8	149./66.5	×2.5

work can be performed locally with in-cache data before filling in the global-matrix data. Speedups were between 1.6 and 4.4, which is a considerable savings.

Conclusions

To reduce the run time of a new magnetic diffusion package in KULL, we have developed a two-step solution process that effectively trades more local work for less global work. This procedure always speeds up the solution, usually by a factor of two, but sometimes by as much as a factor of nearly 40. This speedup comes from both a savings on the global matrix size and reducing iteration counts because the matrix properties have also improved.

Acknowledgments:

This work performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under Contract DE-AC52-07NA27344, LLNL-PROC-464197.

References

[1] T. S. Bailey, M. L. Adams, B. Yang, and M. R. Zika. "A piecewise linear finite element discretization of the diffusion equation for arbitrary polyhedral grids." *J. Comp. Physics* **227** (8), 3738 – 3757 (2008). doi: DOI: 10.1016/j.jcp.2007.11.026.

[2] P. Bochev and A. Robinson. "Matching algorithms with physics: exact sequences of finite element spaces." In D. Estep and S. Tavener, editors, "Preservation of stability under discretization," Colorado State University (SIAM, Philadelphia, 2001), pp. 145–165.

[3] T. Brunner and T. Gardiner. "Status of the magnetic physics in KULL." (2009). Presentation given at Sandia National Laboratory.

[4] T. Brunner and T. Kolev. "Algebraic Multigrid for Linear Systems Obtained by Explicit Element Reduction." *SIAM J. Sci. Comp.* (under review).

[5] E. Caramana, D. Burton, M. Shashkov, and P. Whalen. "The Construction of Compatible Hydrodynamics Algorithms Utilizing Conservation of Total Energy." *J. Comp. Physics* **146**, 227–262 (1998).

[6] V. Henson and U. Yang. "BoomerAMG: A Parallel Algebraic Multigrid Solver and Preconditioner." *Appl. Numer. Math.* **41**, 155–177 (2002).

[7] "*hypre*: High Performance Preconditioners." Http://www.llnl.gov/CASC/hypre/.

[8] T. Kolev and P. Vassilevski. "Parallel Auxiliary Space AMG for H(curl) Problems." *J. Comput. Math.* **27**, 604–623 (2009). Special issue on Adaptive and Multilevel Methods in Electromagnetics.

[9] J.-C. Nédélec. "Mixed finite elements in $\mathbb{R}^3$." *Numer. Math.* **35**, 315–341 (1980).

[10] J. Rathkopf, D. Miller, J. Owen, L. Stuart, M. Zika, P. Eltgroth, N. Madsen, K. McCandless, P. Nowak, M. Nemanic, N. Gentile, N. Keen, and T. Palmer. "KULL: LLNL's ASCI Inertial Confinement Fusion Simulation Code." In "PHYSOR 2000, ANS International Topical Meeting on the Advances in Reactor Physics and Mathematics and Computation into the Next Millennium," (American Nuclear Society, Pittsburgh, PA, 2000).

[11] H. G. Stone and M. L. Adams. "A piecewise linear finite element basis with application to particle transport." In "Nuclear Mathematical and Computational Sciences: A Century in Review, a Century Anew," (American Nuclear Society, Gatlinburg, TN, 2003).