An AMR Capable Finite Element Diffusion Solver for ALE Hydrocodes

A.C. Fisher[1], D.S. Bailey[1], T.B. Kaiser[1], D.C. Eder[1], B.T.N. Gunney[1], N.D. Masters[1], A.E. Koniges[2], R.W. Anderson[1]

1: Lawrence Livermore National Laboratory, P.O.Box 808, Livermore, CA 94551, USA
2: Lawrence Berkeley National Laboratory, 1 Cyclotron Rd. Berkeley, CA 94720, USA

E-mail: fisher47@llnl.gov

Abstract. We present a novel method for the solution of the diffusion equation on a composite AMR mesh. This approach is suitable for including diffusion based physics modules to hydrocodes that support ALE and AMR capabilities. To illustrate, we proffer our implementations of diffusion based radiation transport and heat conduction in a hydrocode called ALE-AMR. Numerical experiments conducted with the diffusion solver and associated physics packages yield 2*nd* order convergence in the L_2 norm.

Keywords: Hydrodynamic Simulation, Heat Conduction, Thermal Radiation, Adaptive Mesh Refinement, Finite Element Method
PACS: 07.05.Tp, 44.40.+a, 44.10.+i

1. Introduction

Modeling based on Arbitrary Lagrange Eulerian (ALE) hydrodynamics has a long track record of providing valuable insights through simulation for experimental programs. For instance hydrocodes have been used extensively at the National Ignition Facility (NIF) to model ignition target behavior during [1] and after the delivery of laser power [2][3]. Over the years, the complexity of these codes has grown as modeling of additional physics packages has been introduced. The computational cost of running these simulations has also grown significantly in recent years as users of these codes run more problems in 2D/3D with increased resolutions. Many hydrocode developers are introducing Adaptive Mesh Refinement (AMR) to their ALE codes [4][5][6][7]. This feature has a significantly beneficial impact on the computational cost of many simulations by enabling the user to put increased resolution where it really matters without refining the entire domain. However, AMR increases the complexity of the hydro implementation and complicates the introduction of physics packages. In particular most hydrocodes include physics packages modeling heat conduction and radiation transport with the diffusion equation. To use AMR for simulations that include these physics packages, a diffusion solver capable of supporting AMR is needed.

Researchers have studied a variety of different approaches to solving the diffusion equation in the context of an AMR mesh with both orthogonal [8] and non-orthogonal zones [9]. Each of these approaches has a distinct set of advantages and disadvantages that merit consideration for their suitability in particular simulations. For instance a finite volume approach has been proposed that has a discretization in line with what is used by many hydrocodes, but only yields 1*st* order accuracy [10]. A level based approach which solves the diffusion equations on each level and corrects the solution with

a sync solve has been studied as well, but only yields 1*st* order accuracy as well with "zig-zag" errors at the coarse fine boundary [11]. A support operator method has been proposed that shows 2*nd* order accuracy in convergence studies, but has significantly angularly dependent error oscillations which may pose problems for simulations that require high symmetry specifications [12].

In this paper we present an approach to adding new physical models using a finite element interface. To enable this approach, we solve various problems unique to working with an ALE and AMR capable code [13]. One such issue is that finite element methods need a global mesh with connectivity information to operate. Typical AMR capable ALE hydrocodes are built on a structured AMR approaches which represent all of the field variables on a level based hierarchy of data without global connectivity. Our method constructs a mapping between a level based representation and a flattened composite mesh representation in order to bridge this gap. All finite element matrix assembly operations can be performed on a virtual composite mesh through this mapping. Another issue unique to this application is the presence of arbitrary coarse-fine interfaces introduced by the AMR. Cells at these interfaces are treated with a transition element approach that maintains continuity across the hanging nodes, edges and faces [14].

To test these approaches we implement a nodal finite element based diffusion solver for ALE-AMR [15]. As the name suggests, ALE-AMR is an AMR capable ALE hydrocode built on SAMRAI (Structured AMR Application Infrastructure) [16]. Physical models for heat conduction and radiation transport are built upon this diffusion solver. These models introduce additional difficulties as temperature and energy in ALE hydrocodes are typically represented as piecewise constant values across the cell, and the quantities in our FEM solver are represented as piecewise nodal bi/trilinear values across the cell. However, good mappings between cell centered and node centered temperatures exist in the literature [17] and are used to resolve these difficulties.

The complexity of this approach necessitates rigorous verification and validation efforts to ensure accurate results. We apply a suite of unit tests to verify the correctness of the many finite element cases encountered at coarse fine boundaries. We also present an L_2 error analysis of the solver which displays 2*nd* order convergence. Additionally, we use the 2*D* dynamic Barenblatt [18] solution to validate the heat conduction module, and the Su-Olsen [19] solution to validate the radiation module again yielding 2*nd* order convergence in the L_2 norm.

2. An ALE-AMR Capable Finite Element Method

2.1. Transition Elements

In addition to the translation between field representations, a family of finite elements is required to account for all of the cases found in the composite mesh. Standard bilinear quads in 2*D* and trilinear hexes in 3*D* are used for elements that are not at a coarse-fine boundary. However, every possible permutation of face refinements at coarse-fine boundaries requires a special transition element. For these elements we use a construction approach similar to the work found in [14]. The extra nodes on the faces due to the transition have basis functions on that face with the value reaching 1 at that node and 0 at the other nodes on the face. In the dimension not on a transition face the basis function simply varies linearly. The corner basis functions in our transition elements are the standard linear functions with fractions of the new transition basis functions subtracted out in order to ensure that all basis functions are 0 at all the transition nodes. This method of construction yields a set of basis functions that satisfies the interpolation property and also enforces continuity across all the element faces. For example a 2*D* transition element with the top and right sides refined 3:1 would yield the following basis and basis gradients (see Figure 1).

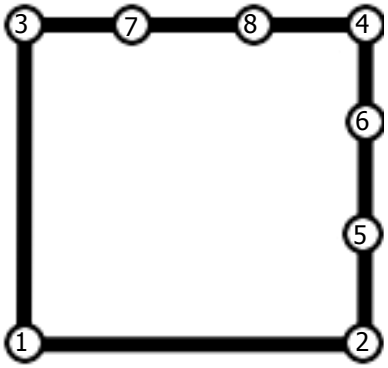

Figure 1. An example of an element with 3:1 refinement transitions on the top and right sides. The numbers identify the locations where the corresponding basis functions in the following basis take on a value of 1.0 and all other basis functions take on a value of 0.0

$$\phi_1(\xi,\eta) = (1-\xi)(1-\eta) \qquad \nabla\phi_1(\xi,\eta) = [\eta-1 \;\; \xi-1]^T \tag{1}$$

$$\phi_2(\xi,\eta) = \xi(1-\eta) - \tfrac{2}{3}\phi_5 - \tfrac{1}{3}\phi_6 \qquad \nabla\phi_2(\xi,\eta) = [1-\eta \;\; -\xi]^T - \tfrac{2}{3}\nabla\phi_5 - \tfrac{1}{3}\nabla\phi_6 \tag{2}$$

$$\phi_3(\xi,\eta) = (1-\xi)\eta - \tfrac{2}{3}\phi_7 - \tfrac{1}{3}\phi_8 \qquad \nabla\phi_3(\xi,\eta) = [\eta \;\; 1-\xi]^T - \tfrac{2}{3}\nabla\phi_7 - \tfrac{1}{3}\nabla\phi_8 \tag{3}$$

$$\phi_4(\xi,\eta) = \xi\eta - \tfrac{2}{3}\phi_6 - \tfrac{1}{3}\phi_5 - \tfrac{2}{3}\phi_8 - \tfrac{1}{3}\phi_7 \qquad \nabla\phi_4(\xi,\eta) = [\eta \;\; \xi]^T - \tfrac{2}{3}\nabla\phi_6 - \tfrac{1}{3}\nabla\phi_5 - \tfrac{2}{3}\nabla\phi_8 - \tfrac{1}{3}\nabla\phi_7 \tag{4}$$

$$\phi_5(\xi,\eta) = \xi \begin{Bmatrix} 3\eta & (0 \le \eta < \frac{1}{3}) \\ 3(\frac{2}{3}-\eta) & (\frac{1}{3} \le \eta < \frac{2}{3}) \\ 0 & (\frac{2}{3} \le \eta \le 1) \end{Bmatrix} \qquad \nabla\phi_5(\xi,\eta) = \begin{Bmatrix} [3\eta \;\; 3\xi]^T & (0 \le \eta < \frac{1}{3}) \\ [3(\frac{2}{3}-\eta) \;\; -3\xi]^T & (\frac{1}{3} \le \eta < \frac{2}{3}) \\ [0 \;\; 0]^T & (\frac{2}{3} \le \eta \le 1) \end{Bmatrix} \tag{5}$$

$$\phi_6(\xi,\eta) = \xi \begin{Bmatrix} 0 & (0 \le \eta < \frac{1}{3}) \\ 3(\eta-\frac{1}{3}) & (\frac{1}{3} \le \eta < \frac{2}{3}) \\ 3(1-\eta) & (\frac{2}{3} \le \eta \le 1) \end{Bmatrix} \qquad \nabla\phi_6(\xi,\eta) = \begin{Bmatrix} [0 \;\; 0]^T & (0 \le \eta < \frac{1}{3}) \\ [3(\eta-\frac{1}{3}) \;\; 3\xi]^T & (\frac{1}{3} \le \eta < \frac{2}{3}) \\ [3(1-\eta) \;\; -3\xi]^T & (\frac{2}{3} \le \eta \le 1) \end{Bmatrix} \tag{6}$$

$$\phi_7(\xi,\eta) = \begin{Bmatrix} 3\xi & (0 \le \xi < \frac{1}{3}) \\ 3(\frac{2}{3}-\xi) & (\frac{1}{3} \le \xi < \frac{2}{3}) \\ 0 & (\frac{2}{3} \le \xi \le 1) \end{Bmatrix} \eta \qquad \nabla\phi_7(\xi,\eta) = \begin{Bmatrix} [3\eta \;\; 3\xi]^T & (0 \le \xi < \frac{1}{3}) \\ [-3\eta \;\; 3(\frac{2}{3}-\xi)]^T & (\frac{1}{3} \le \xi < \frac{2}{3}) \\ [0 \;\; 0]^T & (\frac{2}{3} \le \xi \le 1) \end{Bmatrix} \tag{7}$$

$$\phi_8(\xi,\eta) = \begin{Bmatrix} 0 & (0 \le \xi < \frac{1}{3}) \\ 3(\xi-\frac{1}{3}) & (\frac{1}{3} \le \xi < \frac{2}{3}) \\ 3(1-\xi) & (\frac{2}{3} \le \xi \le 1) \end{Bmatrix} \eta \qquad \nabla\phi_8(\xi,\eta) = \begin{Bmatrix} [0 \;\; 0]^T & (0 \le \xi < \frac{1}{3}) \\ [3\eta \;\; 3(\xi-\frac{1}{3})]^T & (\frac{1}{3} \le \xi < \frac{2}{3}) \\ [-3\eta \;\; 3(1-\xi)]^T & (\frac{2}{3} \le \xi \le 1) \end{Bmatrix} \tag{8}$$

ALE-AMR typically employs a 3:1 refinement ratio for all AMR operations. Odd refinement ratios are chosen to make it possible to set up an n^d:1 correspondence between nodes in a fine and coarse representation respectively. This allows exact inversion of refinement by coarsening[4]. For a given refinement ratio there are 16 and 64 variations of transition elements in $2D$ and $3D$ respectively. Each of these transition elements is constructed in the manner above providing a representation for every possible combination of refined element faces.

2.2. Quadrature Rules

The FEM requires quadrature rules to approximate integrals of basis functions and their derivatives over the elements. There are many standard quadratures for linear quad and hex elements including Gauss-Legendre quadratures and mass lumping quadratures. However, such standard options are not readily available for the transition elements required for AMR support. The work on transition elements in [14] describes a compound Gauss-Legendre quadrature rule that divides the element into sub-elements and

applies the standard Gauss-Legendre quadrature to each sub-element. This provides the same integration accuracy for the transition elements as the usual Gauss-Legendre quadrature on a standard element.

Additionally, it is desirable to have an analogue to the standard mass lumping quadrature for the transition elements. Mass lumping quadratures place the integration points coincident with the nodes of an element. This approach has a lower order of integration accuracy, however, the mass matrices computed are diagonal which is useful in many situations. We construct mass lumping quadrature rules for the family of transition elements by aligning the integration points with the element nodes and then constraining the weights to provide first order integration accuracy and maintain the same symmetry in the weights as the element itself. For example a $2D$ transition element with the left side refined would have 6 quadrature points coincident with the 6 nodes in the element. Enforcement of the following equation ensures first order accuracy

$$\begin{aligned} \Sigma_{i=1}^{6} w_i f(\mathbf{q}_i) &= \int_0^1 \int_0^1 f(\xi,\eta) d\xi d\eta \\ f(\xi,\eta) &= a\xi + b\eta + c\xi\eta + d \end{aligned} \tag{9}$$

where each w_i is a quadrature weight, each $\mathbf{q}_i$ is a quadrature point, and a through d are arbitrary constants. Additionally we ensure that the weights that are simply reflections of each other across the bottom top symmetry line are equal. These constraints yield 6 equations too, but this linear system is only rank 5 leaving 1 dimension of potential solutions to choose from. We choose a sensible result that simplifies the bookkeeping for generating these weights. Applying this process to the $2D$ family of 16 transition elements for a refinement ratio of $3:1$ results in the weights found in Figure 2.

These mass lumping quadrature rules are sufficient to compute mass matrices, however, they cannot be used to compute the stiffness matrices. This deficiency arises due to the undefined derivatives at the node locations on the transition face of an element. This problem can be overcome by averaging the results of the derivatives taken in the limit from all directions within the element. The piecewise definitions of the transition basis functions form distinct regions in the derivatives of those functions separated by cut lines where no derivative exists. This allows the average limit to be computed by averaging 2 derivative evaluations in $2D$ and up to 4 derivative evaluations in $3D$. Thus the quadrature points on the transition side can be split into a distinct number of quadrature points at the same location with the derivatives evaluated in the different distinct regions adjoining that point. These "blurred" quadrature rules require a little extra bookkeeping in order to decide which regions are adjoining and must be sampled, but are otherwise the same as to the usual transition element mass lumping quadrature rules (see Figure 3).

3. An AMR Capable FEM Diffusion Solver

Using the composite mesh mapping and family of transition elements outlined above it is possible to apply the FEM within the framework of ALE-AMR. We now turn our attention to the solution of the following diffusion equation.

$$\nabla \cdot \delta \nabla u + \sigma u = f \tag{10}$$

We employ the standard Galerkin approach and multiply by a test function v and integrate over the the domain Ω.

$$\int_\Omega (\nabla \cdot \delta \nabla u + \sigma u) v \, d\Omega = \int_\Omega f v \, d\Omega \tag{11}$$

Continuing this approach we apply integration by parts to transform the equation to the weak form and rearrange some terms to yield the following.

$$\int_\Omega (\sigma u v - \delta \nabla u \cdot \nabla v) \, d\Omega + \int_{\partial\Omega} \delta \nabla u \cdot \mathbf{n} v \, dS = \int_\Omega f v \, d\Omega \tag{12}$$

Now we can approximate u and v in the basis function constructed using standard 1st order nodal shape functions and the transition shape functions described in the preceding section. By assuming

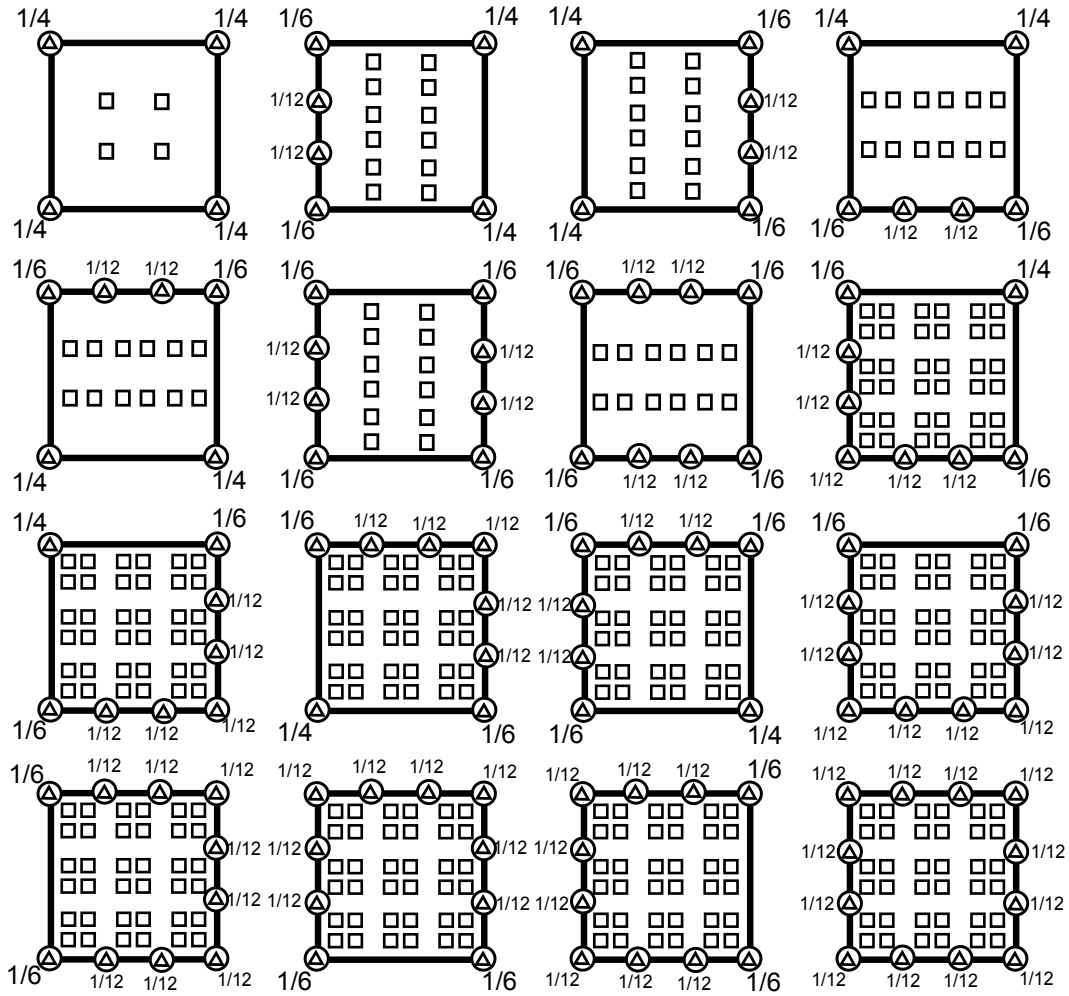

Figure 2. The family of 3 : 1 transition elements in $2D$. Circles are placed at the locations of the element nodes, squares are placed at the integration locations of the compound Gauss-Legendre quadrature rules, and triangles are placed at the integration locations of the transition element mass lumping rules. The fractions indicate the weights associated with the nearby quadrature points.

an insulating boundary conditions the boundary term is identically 0 and we have the following

$$\begin{aligned} u &= \textstyle\sum_j u_j \phi_j \\ v &= \phi_i \\ \textstyle\int_\Omega (\sigma \sum_j u_j \phi_j \phi_i - \delta \nabla \sum_j u_j \phi_j \cdot \nabla \phi_i)\, d\Omega &= \textstyle\int_\Omega f \phi_i \, d\Omega \end{aligned} \tag{13}$$

where each u_j is a degree of freedom and each ϕ_j is a basis function that varies in space with local support. Finally, linearity in the integral and differential operators allows us to factor the $\sum_j u_j$ coefficients outside the integrals and transform to a matrix representation

$$\begin{aligned} A\mathbf{u} &= \mathbf{f} \\ A &= M_\sigma - K_\delta \\ (M_\alpha)_{ij} &= \textstyle\int_\Omega \alpha \phi_i \phi_j \, d\Omega \\ (K_\alpha)_{ij} &= \textstyle\int_\Omega \alpha \nabla \phi_i \cdot \nabla \phi_j \, d\Omega \end{aligned} \tag{14}$$

where M is the mass matrix, K is the stiffness matrix. A set of quadrature rules is needed to approximate the integrals and construct the matrices. For the generation of mass and stiffness matrices on standard

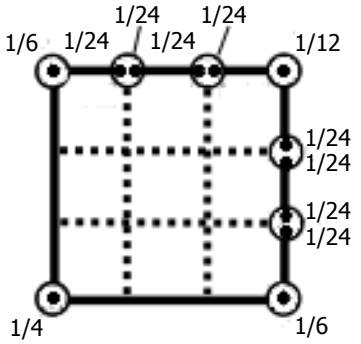

Figure 3. This is an example blurred quadrature rule for the 3:1 transition element with the top and right sides refined. The dotted lines are locations where the transition basis functions have undefined derivatives. Mass lumping quadrature would normally place points on the cut lines, so instead the blurred quadrature splits them into 2 points placed on either side of the cut.

quads and hexes we use basic mass lumping integration rules. These quadrature rules generate an elemental A matrices that are inverse positive. For the transition elements the transition mass lumping quadrature and the blurred transition mass lumping quadrature rules are used for the computation of M and K respectively also yielding inverse positive A matrices. Inverse positivity is an important property for physical models such as heat conduction since temperatures are expected to stay above absolute zero. The element mass and stiffness matrices are assembled into their global counterparts forming a linear system that approximates the diffusion equation. That linear system is solved using the HYPRE [20] GMRES solver with the Euclid [21] preconditioner.

4. Heat Conduction and Radiation Transport Modeling

Now that we have a diffusion equation solver, both heat conduction and radiation transport can be modeled with relative ease. Heat conduction can be modeled with the dynamic diffusion equation

$$C_v \frac{dT}{dt} = \nabla \cdot D(\nabla T) - \alpha T \tag{15}$$

where C_v is the specific heat, T is temperature, D is the heat conductivity, and α is the absorptivity of the medium. This equation is time evolved implicitly yielding

$$\begin{aligned} C_v \frac{T^{n+1}-T^n}{\Delta t} &= \nabla \cdot D^n \nabla T^{n+1} - \alpha T^{n+1} \\ \delta &= D^n \\ \sigma &= -\alpha - \frac{C_v}{\Delta t} \\ f &= -\frac{C_v}{\Delta t} T^n \end{aligned} \tag{16}$$

where δ, σ, and f are the static diffusion equation parameters from (10). This allows us to compute the solution to T^{n+1} from T^n on an AMR hexahedral mesh by applying diffusion solver constructed above to set up and solve a matrix equation.

Similarly radiation transport can be modeled in the diffusion approximation as follows

$$\begin{aligned} \frac{dE_R}{dt} &= \nabla \cdot \lambda(\tfrac{c}{\kappa_r}) \nabla E_R + \kappa_p c(B - E_R) \\ C_v \frac{dT}{dt} &= -\kappa_p c(B - E_R) \end{aligned} \tag{17}$$

where E_R is the radiation energy represented at the nodes, λ is a function used to impose flux limiting on the diffusion approximation, c is the speed of light, κ_r is the Rosseland opacity, κ_p is the Planck opacity,

and B is the blackbody intensity. These equations are implicitly time evolved yielding

$$\begin{array}{rcl} \frac{E_R^{n+1}-E_R^n}{\Delta t} & = & \nabla\cdot\lambda(\frac{c}{\kappa_r})\nabla E_R^{n+1}+\tilde{\kappa}_p c(B^n-E_R^{n+1}) \\ C_v\frac{T^{n+1}-T^n}{\Delta t} & = & -\tilde{\kappa}_p c(B^n-E_R^{n+1}) \\ \delta & = & \lambda(\frac{c}{\kappa_r}) \\ \sigma & = & -\tilde{\kappa}_p c-\frac{1}{\Delta t} \\ f & = & -\frac{E_R^n}{\Delta t}-\tilde{\kappa}_p cB^n \end{array} \tag{18}$$

where $\tilde{\kappa}_p$ is a modification to Planck opacity which is used to linearize the equation similar to that found in [22] (see Section 6). Again this allows us to time advance both E_R and T on an AMR mesh by using the FEM diffusion solve described above.

After E_R and T are evolved through the above equations, the material temperatures and energies must be updated to reflect the changes. However, in ALE-AMR like in many hydrocodes the material temperatures and energies are represented at the cell centers and not the nodal locations which are being updated by these heat conduction and radiation transport models. Thus, to couple these physics modules into ALE-AMR we need a method to map variables from nodes to cell centers and back. We utilize the method described by [17] in which changes in temperature are mapped between nodes and cells. This approach defines projection integrals that map cell centered fields to nodes and node centered fields to cells as follows

$$\begin{array}{rcl} U_i & = & F_{cell\rightarrow node}(U_{cells})=\sum_c U_c\int_{\Omega_c}\phi_i dV \\ U_c & = & F_{node\rightarrow cell}(U_{nodes})=\int_{\Omega_c}\sum_i U_i\phi_i dV\int_{\Omega_c}dV \end{array} \tag{19}$$

where i and c are the node and cell indices respectively for the generic field quantity U. At the end of the hydro step the difference in cell temperatures is computed.

$$\Delta T_c = T_c^{n+1}-F_{node\rightarrow cell}(T_i^n) \tag{20}$$

Using this temperature cell difference and specific heat capacity obtained from an EOS $C_{v,c}$ an energy difference and specific heat on the nodes is computed

$$\begin{array}{r} \Delta e_i = F_{cell\rightarrow node}(\rho C_{v,c}\Delta T_c) \\ C_{v,i} = F_{cell\rightarrow node}(\rho C_{v,c}) \end{array} \tag{21}$$

which are used to update the nodal temperature to the post hydro step time.

$$T_i^* = T_i^n+\Delta e_i/C_{v,i} \tag{22}$$

In some cases this process can create unphysically extreme temperatures that are filtered out using the minimum and maximum temperature values found in the surrounding pre-mapped nodes. This filtering procedure does not upset energy conservation because it only affects the pre-diffusion nodal temperatures and only the post-diffusion differences are captured for mapping to cells [17].

$$T_i^* = max[T_{i,min}, min(T_{i,max}, T_i^*)] \tag{23}$$

The heat conduction and radiation transport models are then applied to update T^* and E_R to the $n+1$ time. Finally, the changes in the nodal temperature must be mapped back to the cells and used to update the internal energy of the cells as follows.

$$\Delta T_c' = F_{node\rightarrow cell}(T_i^{n+1}-T_i^*)e_c^{n+1} = e_c^*+C_{v,c}\Delta T_c' \tag{24}$$

These difference mappings make it possible to transfer energy between nodes and cells without introducing large amounts of artificial diffusion. Applying the mappings to the transition elements at

coarse fine boundaries is straightforward. The extra nodes in the transition elements simply add extra values and basis functions to sum over in (19). The integrals are evaluated using the Gauss-Legendre quadratures and their compound extensions previously discussed.

It should be noted that these solutions are implicit in nature and have no convergence limits on time step. However, non-linearities introduced into the diffusion coefficient due to temperature dependence in the equations of state can cause the accuracy of the method to plummet if time steps are too large and cause large changes of temperature in a single step. We limit our time steps based on the maximal expected change in energy in a single time step. We only allow time steps large enough to change energy by some fraction, usually within $0.05 - 0.1$.

5. Verification and Validation

5.1. Unit Testing

There are many variations of transition elements and quadrature rules used in this method. This significantly complicates the development of the code to represent them and increases the chance of introducing errors. Given such issues, we believe it prudent to do verification work to provide confidence in this new code. The approach we take is to provide a suite of unit tests that ensure various properties known about our finite elements and their quadrature rules for every element type in the family of transition elements. For the transition elements we test the interpolation property and the values of the basis function gradients at the center of the cell. We also ensure that the quadrature rules have all positive weights and that the sum of the weights of each quadrature rule is 1. Also, we test the transition elements together with the quadrature rules by forming element mass matrices on random elements and ensuring that they are all diagonal. Finally, we test the inverse positivity of the dynamic diffusion operator $\alpha M + K$ by forming these element matrices on high aspect ratio zones and ensuring that the operator is an M-matrix. Executing these unit tests gives us confidence that the transition element code that we have constructed operates according to design. The unit tests are also very useful when changes are made to the code, as they catch errors that cause one or more of the tests to fail and provide clues about the problem.

5.2. Static Diffusion Error Convergence

The static diffusion solver introduced in Section 3 relies on some unique approaches with untested accuracy. As such, it is important to measure the convergence rate of the solver to make sure it is in line with the 2*nd* order convergence expected from a linear FEM solution of the diffusion equation. We measure the convergence rate of this solution using a standard L_2 error convergence test on the following Laplace equation.

$$\begin{aligned} \nabla^2 u &= 0 \qquad on\ \Omega = \{0 \le (x,y) \le 1\} \\ u(x,0) &= x \\ u(x,1) &= 1-x \\ u(0,y) &= y \\ u(1,y) &= 1-y \end{aligned} \tag{25}$$

A random mesh is generated with the right side refined using a ratio of 3:1 in order to test the transition elements (see Figure 4). The entire random mesh is then refined 3 more times yielding a total of 4 meshes. Each of these meshes is used to approximate the solution to a the simple Poisson problem defined above. These approximations are then compared to the analytical result to obtain the L_2 norm of the error. The slope of 2 in the error norms, shown in the results Figure 4, indicates that the method has 2*nd* order convergence. This is in line with the convergence rates of the other methods in ALE-AMR.

5.3. Dynamic Heat Conduction Results

The integration of the heat conduction into ALE-AMR relies on another new technique. The cell/node mapping approach mentioned above has been utilized before, but not in combination with the transition

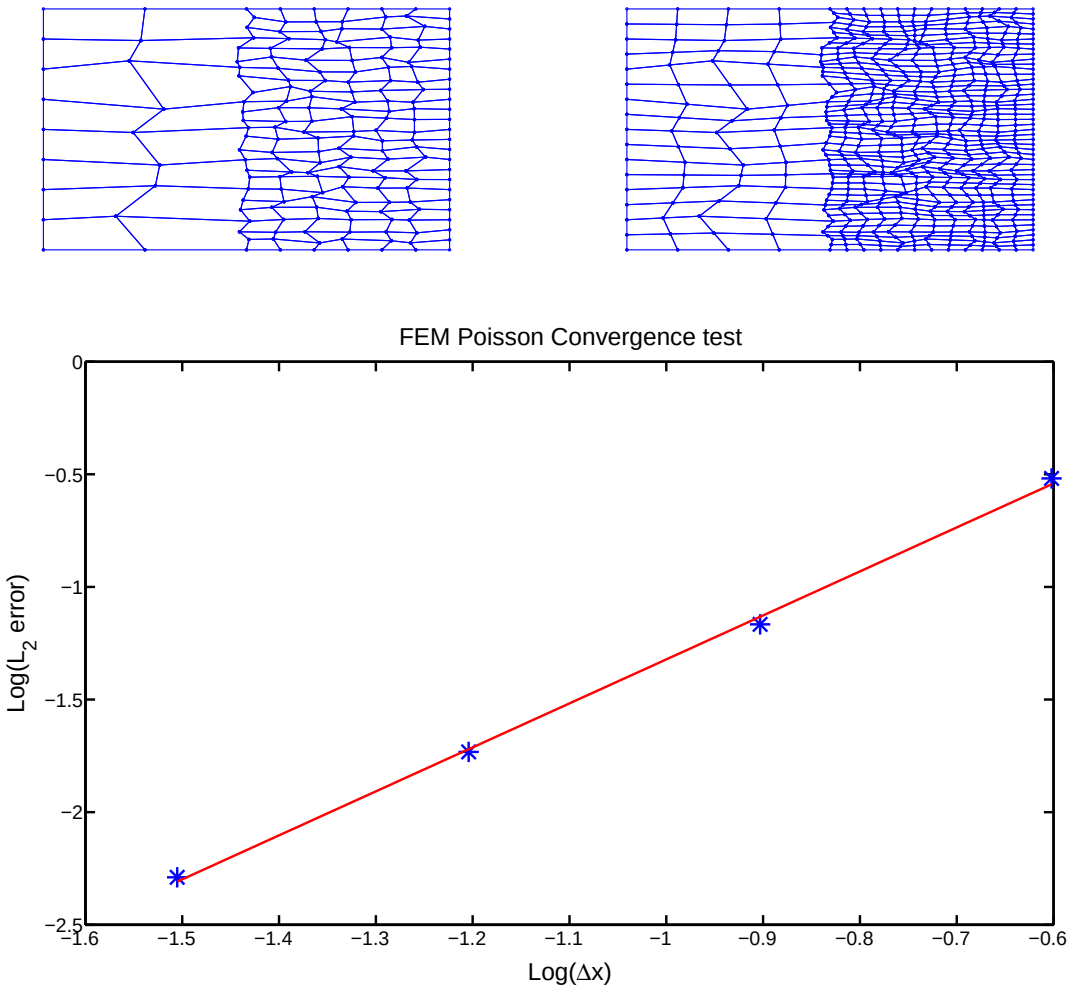

Figure 4. Convergence test of a Poisson problem on a randomized mesh with transition elements. The meshes displayed are the first two meshes used in the convergence test. The asterisks on the plot indicate the L_2 errors of the Poisson solution on each of the 4 successive meshes. The line fit has a slope of 1.95 indicating 2nd order convergence to the exact solution.

elements and quadrature rules outlined previously in this paper. It is also useful to examine the behavior of the solver with non-linearities introduced into the diffusion coefficient. For these reasons we present validation work with the Barenblatt problem [18] in $2D$. This problem begins with a non-zero energy inserted into a single point in a background of zero-energy. The material is an ideal gas with constant specific heat and a conductivity model of the following form

$$D = d_0 \rho^a T^b \tag{26}$$

where d_0, a, and b are parameters of the model. In the simulation we insert the non-zero energy in to the bottom left cell of the domain and avoid reflections off the top and right boundaries by using a large domain. In particular we use an $81x81$ uniform mesh to model a $3cmx3cm$ domain. The boundary conditions are all Neumann type which introduces quarter plane symmetry at the lower left corner. Aside from the aforementioned hot spot, the field begins with a uniform temperature of $0K$ and a density of $1g/cm^3$. The energy in the hot spot is allowed to diffuse through heat conduction for $1\mu s$ and the results

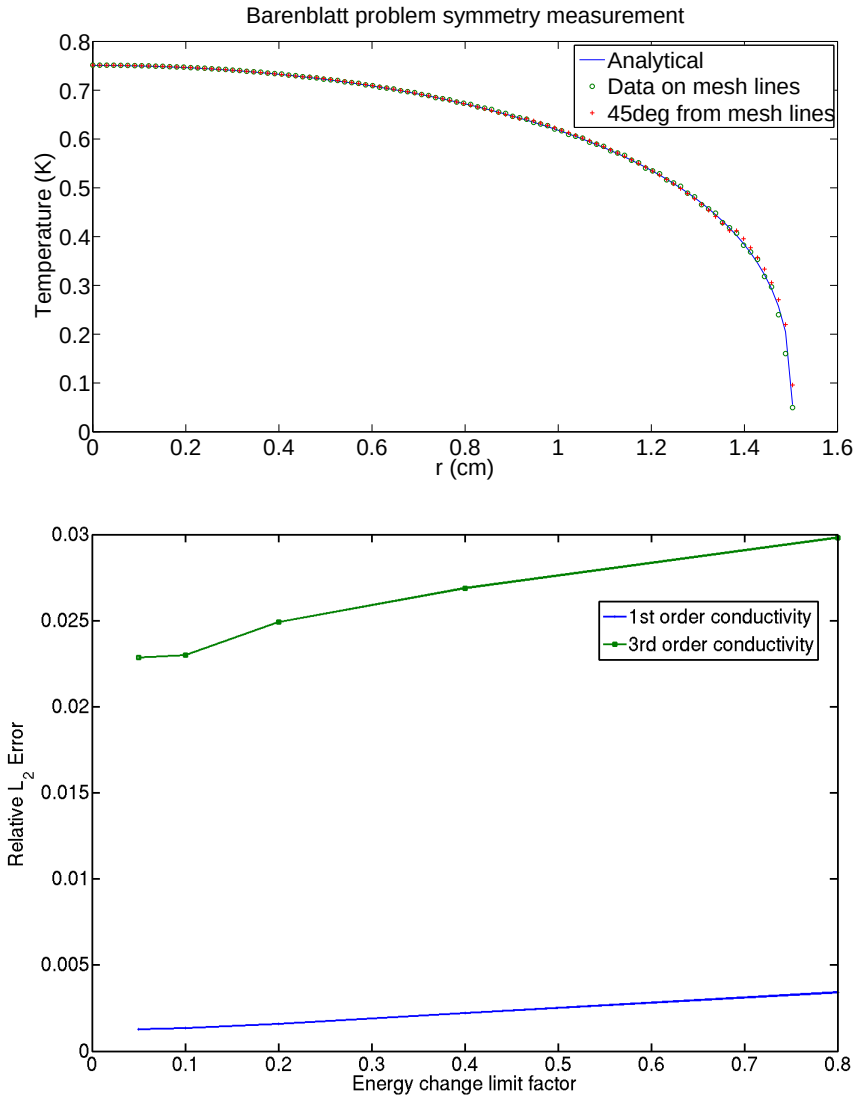

Figure 5. On the top are the results of the ALE-AMR simulation of the Barenblatt problem with $b = 3$ and an energy change limit of 0.8. The simulated results are quite well converged with the exact solution. The L_2 errors for the results on the mesh lines and diagonally across the zones are 0.0078 and 0.0084 respectively. On the bottom are the relative L_2 error measurements for the $b = 1$ and $b = 3$ case as a function of the fractional energy change limit for Δt. Even with a large energy change limit of 0.8 the relative error is less than 3%

of such simulations are compared to the exact results found in [18]. For these simulations we set the conductivity parameters to the following constants: $d_0 = 1.0$, $a = 1$, and $b = 1$ *or* 3. We limit the time step by a maximum energy change fraction in any cell which we vary from 0.05 to 0.8 in numerous simulations. The results of these simulations are found in Figure 5.

Additionally, we are interested in the AMR performance of the heat conduction module. We run simulations of the Barenblatt problem similar to those above in order to measure the AMR performance. These simulations use the same $3cm$ square domain, and at the finest level the mesh has $243x243$ elements. In one case we add a coarser level and allow the AMR to coarsen the mesh in places where

the second differences of the cell energies are low, and in the other case we maintain the fine zones throughout the mesh. This process occurs at every time step so the mesh is coarsened and refined based on our energy criterion. We set the conductivity parameters to $d_0 = 1$, $a = 1$, and $b = 3$, and we use the large energy change limit of 0.8. The relative L_2 error is computed for each simulation and the wall clock simulation time is recorded. We report the results of this procedure in the following table. The recorded

num. levels	rel. L_2 err.	wall clock time (s)
1	0.008	1840
2	0.009	755

Table 1. AMR performance for the Barenblatt problem. Enabling a second AMR level in the problem reduces the wall clock time by a factor of $2.5x$ while maintaining similar error levels.

values show that enabling 2 levels with AMR yields a $2.5x$ performance improvement in this case with small loss in solution accuracy. The results of all the Barenblatt simulations indicate that the diffusion solver has reasonable levels of accuracy for large time steps and good AMR performance. As such, this heat conduction implementation is a good choice for use in ALE-AMR and other ALE hydrocodes like it.

5.4. Dynamic Radiation Results

We are also interested in modeling radiation transport with the diffusion equation. As is the case with the heat conduction modeling, the radiation diffusion implementation also requires cell/node mapping of temperatures. Additionally, this implementation introduces radiation energy as a new nodal field variable. Since this radiation energy is a per volume quantity, special care must be taken to update the nodal values during the Lagrange and remap ALE steps. Time steps are set to limit the maximum fractional energy change, but they also must be limited in order to avoid overstepping the radiation/thermal temperature equilibrium that the equations (17) represent. Finally, many radiation diffusion problems require mixed boundary conditions which are treated with special boundary elements in our diffusion solver.

We test the implementations of these unique attributes with the classic Su-Olsen solution [19] to the Marshak diffusion problem. A pseudo-1D simulation is enabled by using a 2D domain and applying Neumann boundaries at the top and bottom. In particular we use a $5cm$ domain with uniformly sized elements ranging from $0.025cm$ to $0.1cm$ in 4 different simulations. The simulations are run with the retardation parameter $\varepsilon = 1$ and terminated at the dimensionless time $\tau = 1$. The results of these simulations are compared with the benchmark results listed in tables found in [19], and a refinement study is included in the following Figure 6. The slope of the line in the refinement study indicates $2nd$ order convergence as expected.

6. Conclusions and Future Work

We have presented and implemented an approach to adding heat conduction and radiation transport physics packages to an AMR capable ALE hydrocode. These capabilities are built on an AMR enabled FEM diffusion solver that we designed for this purpose. This diffusion solver was shown to have $2nd$ order convergence. Also, we ran test problems with the heat conduction and radiation transport modules. The performance in these test problems indicated $2nd$ order convergence to analytic solutions. Finally our implementation showed significantly improved performance with AMR enabled.

There are two avenues of future research that we believe would be particularly fruitful. The first, and more straightforward, of the avenues would be to use the existing transition element and composite mesh FEM framework to solve other equations that represent interesting physics. For instance building a biharmonic equation solver on this framework would be a reasonable approach for modeling of surface tension effects in an ALE code with AMR. The second, more fundamental, avenue of research would be

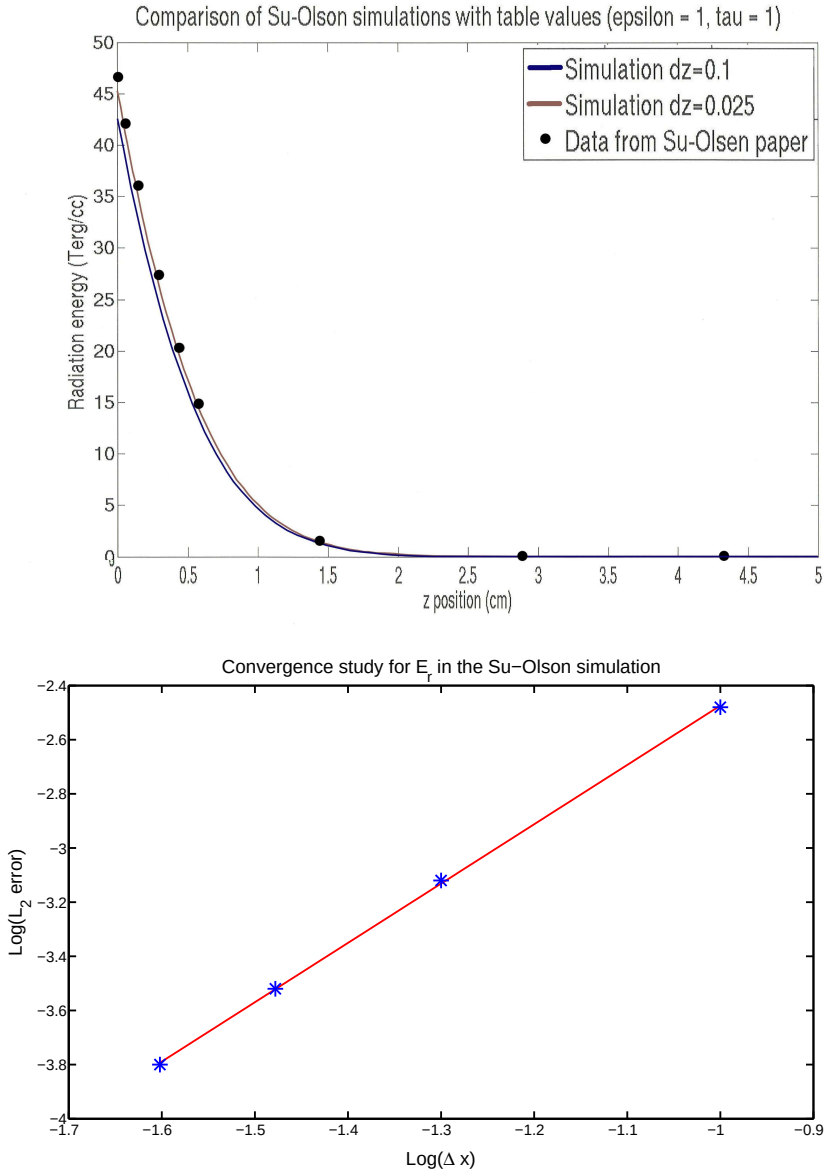

Figure 6. On the top are the results of ALE-AMR simulations of the Su-Olsen problem with $\varepsilon = 1$, $\tau = 1$, and 2 different resolutions. The circles on the plot represent the analytical values obtained from the Su-Olsen paper. On the bottom is the L_2 error as a function of resolution. The asterisks represent the L_2 error values obtained with simulations on successively refined meshes, and the line represents the best fit to those points. The line fit slope of 2.19 indicates 2nd order convergence.

to broaden the family of transition elements to include edge-based and face-based basis function. This would open up a path to AMR for finite element based models that use edge-based and face-based basis functions such as those found in computational electromagnetics.

Appendix

Linearization of Blackbody Intensity

The equations representing diffusion based radiation transport have a nonlinearity in the matter-radiation energy coupling terms that require special attention. This is easily seen with the 4th order dependence in the following temperature update equation.

$$\begin{aligned} C_v(T^{n+1}-T^n) &= -\Delta t c\kappa_p(B^{n+1}-E_R^{n+1}) \\ B^{n+1} &= \frac{4\sigma_{sb}}{c}(T^{n+1})^4 \end{aligned} \tag{27}$$

The equations in (18) include the following linearization of the blackbody intensity which eliminates the need for nonlinear iterations. The fundamental approximation in this procedure is to hold the derivative of the blackbody intensity with respect to temperature constant over the course of the time step, thus:

$$\begin{aligned} \frac{dB^{n+1}}{dT} &\approx \frac{dB^n}{dT} = \frac{16\sigma_{SB}}{c}(T^n)^3 \\ B^{n+1} &\approx \frac{1}{4}\frac{dB^n}{dT}T^{n+1} \end{aligned} \tag{28}$$

By applying this approximation to (27) and solving for T^{n+1} the following results.

$$\begin{aligned} T^{n+1} &\approx \frac{CvT^n+\Delta t c\kappa_p E_R^{n+1}}{C_v+\frac{1}{4}\frac{dB^n}{dT}\Delta t c\kappa_p} \\ B^{n+1} &\approx \frac{4C_vB^n+\frac{\delta B^n}{\delta T}\Delta t c\kappa_p E_R^{n+1}}{4Cv+\frac{dB^n}{dT}\Delta t c\kappa_p} \end{aligned} \tag{29}$$

Finally, we can use this approximation for blackbody intensity to write the linearized form of the radiation-matter coupling term.

$$\begin{aligned} \Delta t c\kappa_p(B^{n+1}-E_R^{n+1}) &\approx \Delta t c\tilde{\kappa}_p(B^n-E_R^{n+1}) \\ \tilde{\kappa}_p &= \frac{4C_v\kappa_p}{4C_v+\frac{dB^n}{dT}\Delta t c\kappa_p} \end{aligned} \tag{30}$$

By applying this procedure we have removed the 4th order dependence on the current temperature and eliminated the need for a computationally expensive nonlinear iteration. It is also worth noting that by using this approach in the limit of large Δt, the matter and radiation energies approach equilibrium with $B^{n+1} = E^{n+1}$.

Acknowledgments

This work performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under Contract DE-AC52-07NA27344. Work by LBNL under DE-AC02-05CH11231 was supported by the Director, Office of Science of the U.S. Department of Energy and the Petascale Initiative in Computational Science and Engineering. We acknowledge the National Energy Research Scientific Computing Center, supported by the Office of Science, U.S. Department of Energy under Contract No. DE-AC02-05CH11231.

References

[1] Marinak M, Kerbel G, Gentile N, et al. 2001, Physics of Plasmas, 8: 2275
[2] Eder D, Koniges A, Landen O, et al. 2008, Journal of Physics: Conference Series, 112: 1742
[3] Koniges A, Masters N, Fisher A, et al. 2014, Plasma Science and Technology, (in review)
[4] Anderson R, Elliott N, Pember R. 2004, Journal of Computational Physics, 199: 598

[5] Morrell J, Sweby P, Barlow A. 2007, Journal of Computational Physics, 226: 1152
[6] Koniges A, Masters N, Fisher A, et al. 2010, Journal of Physics: Conference Series, 244: 1742
[7] Morrell J. 2011, Computers & Fluids, 46: 375
[8] Pernice M, Philip B. 2006, SIAM Journal of Scientific Computing, 27: 1709
[9] Breil J, Maire P. 2007, Journal of Computational Physics, 224: 785
[10] Coudire Y, Villedieu P. 2000, SAIM: Mathematical Modeling and Numerical Analysis, 34: 1123
[11] Howell L, Greenough J. 2003, Journal of Computational Physics, 184: 53
[12] Lipnikov K, Morel J, Shashkov M. 2004, Journal of Computational Physics, 199: 589
[13] Fisher A, Bailey D, Kaiser T, et al. 2010, Journal of Physics: Conference Series, 244: 022075
[14] Gupta A. 1978, International Journal for Numerical Methods in Engineering, 12: 35
[15] Anderson R, Elliott N, Pember R. 2004, Journal of Computational Physics, 199: 598
[16] Wissink A, Hornung R, Kohn S, et al. 2001, Large Scale Structured AMR Calculations Using the SAMRAI Framework, SuperComputing 2001, Denver CO
[17] Shestakov A, Milovich J, Prasad M. 2001, Journal of Computational Physics, 170: 81
[18] Barenblatt G. 1979, "Similarity, Self-Similarity, and Intermediate Asymptotics", Consultants Bureau: New York, USA
[19] Su B, Olsen G. 1996, Journal of Quantitative Spectroscopy and Radiative Transfer, 56: 337
[20] Chow E, Cleary A, Falgout R. 1998, "Design of the hypre Preconditioner Library", SIAM Workshop on Object Oriented Methods for Inter-operable Scientific and Engineering Computing, Yorktown Heights NY
[21] Hysom D, Pothen A. 2001, SIAM Journal of Scientific Computing, 22: 2194
[22] Shestakov A, Harte J, Kershaw D. 1988, Journal of Computational Physics, 76: 385