Mulard: A Multigroup Thermal Radiation Diffusion Mini-Application

Thomas A. Brunner
Lawrence Livermore National Laboratory, P. O. Box 808, Livermore, CA 94551

April 16-18, 2012

Abstract

In high energy density physics simulations, multigroup radiation diffusion can dominate the total run time and memory usage. Mulard, a new mini-application implements a finite-element discretization on an unstructured mesh for the coupled, implicit diffusion solves. The amount of matrix data needed to solve the tens to hundreds of coupled diffusion equations can be very large, but the related nature of the group matrices presents new opportunities for optimization not possible when optimizing a single implicit solve.

1 Introduction

Mulard is a sample application that solves the multi-group diffusion equations coupled to a material energy equation[2, 11] – a common approximation used in multi-physics hydrocodes. This example is designed to be used as a test bed for exploring different coding techniques for advanced computer architectures.

This mini-application is a little heavier weight than many of the mini-apps being produced today for this same purpose. The idea behind this is to preserve a little more of the complexity and limitations of a production code that aren't present in some of the smaller mini-apps.

In particular, several features are kept to increase the code complexity more representative of that required by the production codes; abstract material properties and problem setup and several test problems are supported. Multiple, related and coupled diffusion equations are solved together. The order of the discretization can be changed. The visualization files can be written. Many of the code abstractions come from the math, allowing more rapid code development. There are various run-time switches that allow changes to the discretization.

A secondary goal in creating this proxy application is to provide a useful vehicle for doing research on numerical methods used to solve this physics. Specifically, one new feature not seen in production codes is the ability to change the discretization order; this has the potential to increase the computational intensity needed per byte, which might be needed on future platforms.

Mulard is designed to explore the intra-node parallelism techniques that we expect to be needed in the future. This includes programming paradigms such as CUDA, OpenCL, threads, and transactional memory features. This explicitly excludes inter-node parallelism

currently performed using MPI for communications. We believe the inter-node parallelism is well understood using scalable preconditioners[6], and did not want to complicate the task of exploring intra-node parallelism by having an MPI layer.

2 The physical system

We want to solve the nonlinear, coupled, time dependent problem

$$\frac{\partial E_g}{\partial t} - \nabla \cdot \frac{1}{3\sigma_{t,g}} \nabla E_g = \sigma_{a,g} \left[B_g(T) - E_g \right] + S_g, \tag{1}$$

$$\frac{\partial u}{\partial t} = \sum_g \sigma_{a,g} \left[E_g - B_g(T) \right] + Q, \tag{2}$$

where E_g is the radiation energy density in group g, g is group number (there are G coupled diffusion equations), t is time, S_g is an arbitrary source of thermal radiation for group g, $\sigma_{a,g}$ is the absorption coefficient for group g, $\sigma_{t,g}$ is the total (absorption plus scattering) coefficient for group g, T is the material temperature, $u = \rho C_v T$ is the material energy density, C_v is the material heat capacity, ρ is the material density, Q is an arbitrary source of thermal energy, and $B_g(T)$ is the black body emission function[4] for group g. All the material properties, σ_g, C_v, etc., can depend on the material temperature and density.

Each of the group diffusion equations comes from discretizing the photon energy dependence. Each equation is simply an integral over an energy range of the photon energy-dependent diffusion equation. The one group equation, integrated over all photon frequencies is called the gray diffusion equation.

We will assume a unitless system where the speed of light $c = 1$, Planck's constant $h = 1$, and the Boltzmann constant $k_b = 1$. The radiation energy density E_g, the material energy density u and the temperature T all have units of energy per volume.

2.1 Boundary conditions

There are three different kinds of boundary conditions that we use.

The first is a Dirichlet boundary, where we specify the exact energy density on the boundary,

$$E_g(\vec{x}) = f(\vec{x}, g) \tag{3}$$

where $\vec{x}$ is the position on the boundary, and f is the specified function. This type is not physical, but is extremely useful for testing purposes.

The second type of boundary condition is a symmetry, or reflecting, boundary condition. The net flux of radiation crossing the boundary is zero, namely

$$\vec{F_g} = -\frac{1}{3\sigma_{t,g}} \nabla E_g = 0 \tag{4}$$

This type boundary is most useful for reducing the simulation size.

In the third type of boundary, the incoming flux of radiation is specified. This leads to a mixed boundary condition where a linear combination of the energy density E_g and the net flux $\vec{F}_g$ are specified.

$$F_{\text{in},g} = \frac{1}{4}E_g - \frac{1}{2}\vec{n}\cdot\vec{F}_g \tag{5}$$

$$= \frac{1}{4}E_g + \frac{1}{6\sigma_{t,g}}\vec{n}\cdot\nabla E_g \tag{6}$$

where $\vec{n}$ is the outward unit normal of the surface. This boundary condition is the most common physical boundary used in real problems, and includes vacuum boundary that specify no incoming flux or source boundaries that add energy to the system.

3 The Discretization

We discretize Eqs 1-2 using a semi-implicit, linearized time discretization and a continuous nodal finite element in space approximation. We use MFEM[8], a finite element discretization library to manage loading the unstructured mesh, managing the finite element integrations, and solving the linear systems. MFEM is our only external dependency.

3.1 Time Discretization

In order to discretize the equations, we use the backward Euler method for the time derivatives. The method implemented here is a simplification of what is described in the papers by Morel[10, 9]. The equations are still nonlinearly coupled, so we lag all material properties to their values at the end of the last time step. We also linearize the material emission around an estimate for the change in temperature. This leads to the following form of the equations:

$$\kappa = \left[\Delta t \sum_{g'} \sigma_{a,g'} B'_{g'}(T^n) + \rho C_v\right]^{-1} \tag{7}$$

$$\Delta T_{\text{est}} = \kappa\Delta t\left(Q + \sum_{g'}\sigma_{a,g'}\left(E^n_{g'} - B_{g'}(T^n)\right)\right) \tag{8}$$

$$B_{g,\text{est}} = B_g T^n + \Delta T_{\text{est}} B'_g(T^n) \tag{9}$$

$$-\nabla\cdot\frac{1}{3\sigma_{t,g}}\nabla E^{n+1}_g + \left(\sigma_{a,g}+\tau\right)E^{n+1}_g = \tau E^n_g + S_g + \sigma_{a,g}B_{g,\text{est}} \tag{10}$$

$$u^{n+1} = u^n + \Delta t\sum_g \sigma_{a,g}\left[E^{n+1}_g - B_{g,\text{est}}\right] + \Delta t Q \tag{11}$$

where $\tau = 1/\Delta t$, $B'_g = \frac{\partial B_g}{\partial T}$, and all quantities are evaluated at the beginning of the time step, unless otherwise noted.

This set of equations is often wrapped in an outer iteration to resolve the nonlinearities. We do not take that approach here, but instead perform just one step. (We are essentially doing one Newton iteration with just the nonlinearities in the material coupling resolved.)

In coupled problems, Because the old energy density $E_{g'}^n$ is used in the estimation of the temperature change, ΔT_{est}, we have a stability constraint (left to the interested reader as an exercise).

3.2 Space Discretization

We discretize the linearized equations using the Galerkin finite element method[7, 1]. The key variables are expanded in terms of finite element basis functions,

$$E_g \approx \sum_n E_{g,n} w_n, \tag{12}$$

$$u \approx \sum_z u_z v_z, \tag{13}$$

where w_n is a continuous, L^2 integrable, nodal basis function for nodal-unknown n, and v_z is a piece-wise continuous, L^2 integrable, zone basis function for zone-unknown z. For the lowest order finite element method, there is exactly one unknown per node or zone, but for the higher order methods there are extra degrees of freedom beyond the number of nodes and zones.

We insert these expansions into the equations, multiply by a basis function, and integrate over all space. This leads to the following system:

$$\begin{aligned} -\left(w, \frac{1}{3\sigma_{t,g}}\nabla w\right)_S E_g^{n+1} + \left(\nabla w, \frac{1}{3\sigma_{t,g}}\nabla w\right) E_g^{n+1} + \left(w, \left(\sigma_{a,g} + \tau\right) w\right) E_g^{n+1} \\ = \left(w, \tau w\right) E_g^n + \left(w, S_g\right) + \left(w, \sigma_{a,g} B_{g,\text{est}}\right) \end{aligned} \tag{14}$$

$$\left(v, v\right) u^{n+1} = \left(v, v\right) u^n + \left(v, \Delta t Q\right) + \sum_g \left(v, \Delta t \sigma_{a,g} w\right) E_g^{n+1} - \sum_g \left(v, \Delta t \sigma_{a,g} B_{g,\text{est}}\right) \tag{15}$$

To ensure conservation of energy between the two equations, we will integrate both equations at the exact same integration points; the opacity and material emission terms must be exactly equal in both equations.

We also have the option to lump the mass matrix term in either equation. In problems with a very strong variation in the solution, lumping is critical for a stable solution.

3.3 Boundary Conditions

In the finite element integrals above, we have a surface integral term. This represents the net flux of radiation across the surface. If we simply set this term to zero, we are specifying zero net flux, or the reflecting (or symmetry) boundary conditions above. For this system, these are the natural boundary conditions, and nothing additional is needed.

For Dirichlet boundaries, we also ignore this term, but specify E_g at the support points of the discretization. (Typically the nodes of the mesh, but also the higher order control points on the surface of the zone.)

The incoming flux boundary requires that both the matrix and right hand side be modified. If we multiply the boundary condition by a basis function, and integrate over the surface, we

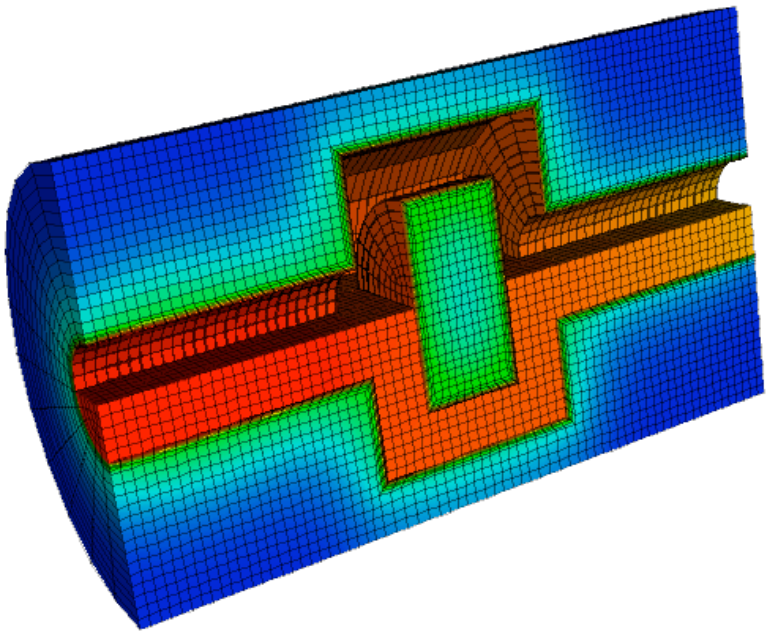

Figure 1: Results from the crooked pipe test problem computed using Mulard. The top half shows the material temperature, and the bottom half shows the radiation temperature. Radiation enters on the left side and flows down the duct, heating the material.

get

$$-\left(w, \frac{1}{3\sigma_{t,g}}\nabla E_g\right)_S = \left(w, \frac{1}{2}E_g\right)_S - \left(w, 2F_{\text{in},g}\right)_S \tag{16}$$

For multigroup problems, a temperature source can be specified on the boundary. The incoming flux is then given by

$$F_{\text{in},g} = \frac{1}{4}B_g(T). \tag{17}$$

4 Test Problems

Mulard supports running about ten different test problems that stress different aspects of the code. These support both two and three dimensional meshes of arbitrary connectivity. The mesh can be refined, and the order of the discretization can be increased. Some have analytic solutions, so the correctness of the code can be verified[3]. Others, like the crooked pipe test problem[5] have complicated geometries in three dimensions. Fig. 1 shows the results for this test problem on an unstructured mesh. We typically use unstructured meshes to conformally fit material interfaces and have mesh resolution increased only in one dimension. This greatly reduces our zone counts compared to a structured mesh for a given accuracy. The reduction more than offsets the overhead of dealing with the unstructured nature of the problem.

5 Conclusions

The Mulard code is ready to start exploring intra-node parallelization. It is a relatively full-featured multigroup radiation diffusion code that is small and flexible enough to easily refactor quickly.

Acknowledgment

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

References

[1] Teresa S. Bailey, Marvin L. Adams, Brian Yang, and Michael R. Zika. A piecewise linear finite element discretization of the diffusion equation for arbitrary polyhedral grids. *Journal of Computational Physics*, 227(8):3738 – 3757, 2008.

[2] Thomas A. Brunner. Forms of approximate radiation transport. Technical Report SAND2002-1778, Sandia National Laboratories, July 2002.

[3] Thomas A. Brunner. Development of a grey nonlinear thermal radiation diffusion verification problem, invited. In *Transactions of the American Nuclear Society*, volume 97, pages 876–878. American Nuclear Society, November 2006.

[4] B. A. Clark. Computing multigroup radiation integrals using polylogarithm-based methods. *Journal of Computational Physics*, 70(2):311 – 29, JUN 1987.

[5] N. A. Gentile. Implicit monte carlo diffusion: An acceleration method for monte carlo time-dependent radiative transfer simulations. *Journal of Computational Physics*, 172(2):543 – 571, 2001.

[6] V. E. Henson and U. M. Yang. BoomerAMG: a parallel algebraic multigrid solver and preconditioner. *Applied Numerical Mathematics*, 41(5):155–177, 2002. Also available as LLNL technical report UCRL-JC-141495.

[7] David S. Kershaw. Differencing of the diffusion equation in lagrangian hydrodynamic codes. *Journal of Computational Physics*, 39(2):375 – 395, 1981.

[8] Tzanio Kolev and Veslin Dobrev. Mfem: Finite element discretization library. `http://code.google.com/p/mfem/`.

[9] J.E. Morel, E.W. Larsen, and M.K. Matzen. A synthetic acceleration scheme for radiative diffusion calculations. *Journal of Quantitative Spectroscopy and Radiative Transfer*, 34(3):243 – 261, 1985.

[10] Jim E. Morel, T.-Y. Brian Yang, and James S. Warsa. Linear multifrequency-grey acceleration recast for preconditioned krylov iterations. *Journal of Computational Physics*, 227(1):244 – 263, 2007.

[11] Gordon L. Olson, Lawrence H. Auer, and Michael L. Hall. Diffusion, P1, and other approximate forms of radiation transport. *Journal of Quantitative Spectroscopy and Radiative Transfer*, 64(6):619–634, March 2000.