

ALEGRA: An Arbitrary Lagrangian-Eulerian Multimaterial, Multiphysics Code

Allen C. Robinson*, Thomas A. Brunner*, Susan Carroll*, Richard Drake*,

Christopher J. Garasi*, Thomas Gardiner*, Thomas Hail*, Heath Hanshaw*, David Hensinger*,

Duane Labreche*, Raymond Lemke*, Edward Love*, Christopher Luchini*, Stewart Mosso*,

John Niederhaus*, Curtis C. Ober*, Sharon Petney*, William J. Rider†, Guglielmo Scovazzi*,

O. Erik Strack*, Randall Summers*, Timothy Trucano*, V. Greg Weirs*, Michael Wong*,

Thomas Voth*

ALEGRA is an arbitrary Lagrangian-Eulerian (multiphysics) computer code developed at Sandia National Laboratories since 1990. The code contains a variety of physics options including magnetics, radiation, and multimaterial flow. The code has been developed for nearly two decades, but recent work has dramatically improved the code's accuracy and robustness. These improvements include techniques applied to the basic Lagrangian differencing, artificial viscosity and the remap step of the method including an important improvement in the basic conservation of energy in the scheme. We will discuss the various algorithmic improvements and their impact on the results for important applications. Included in these applications are magnetic implosions, ceramic fracture modeling, and electromagnetic launch.

Nomenclature

\bar{p}	mean pressure
\mathbf{B}	magnetic induction
\mathbf{d}	incremental displacement field
\mathbf{n}	direction unit vector
\mathbf{x}_i	vector of nodal coordinates for component i
\mathbf{D}	symmetric part of the velocity gradient tensor
\mathbf{U}	total displacement field
\mathbf{E}	electric field in laboratory frame
\mathbf{E}'	electric field in co-moving frame
$\frac{d}{dt}$	$\frac{\partial}{\partial t} + \mathbf{u} \cdot \nabla$
\mathbf{F}_r	radiative flux
\mathbf{H}	magnetic field
\mathbf{J}	electric current density
$\mathbf{A}_{i,j}$	Jacobian matrix of element i , corner j
$\mathbf{W}_{i,j}$	target Jacobian matrix of element i , corner j

*Sandia National Laboratories, MS 0378, Albuquerque, NM 87185

†Corresponding Author, e-mail: [wjriders@sandia.gov](mailto:wjrider@sandia.gov)

\mathcal{H}	hourglass functions
\mathcal{F}	mesh quality objective function
\mathbf{P}_r	radiation pressure tensor
\mathbf{T}	stress tensor
\mathbf{T}^M	magnetic stress tensor
\mathbf{u}	fluid velocity
$\mathbf{q}_{i/e}$	ion/electron heat flux
$\mathbf{T}_{i/e}$	ion/electron fluid stress tensor
\mathbf{W}	anti-symmetric part of the velocity gradient tensor
a	coefficient for a parabolic interpolation at element edge
B	bulk modulus
B_ν	Planckian radiation emission distribution, $2h\nu^3/[c^2(\exp(h\nu/kT_e) - 1)]$
B_k	bulk modulus of the k th material
c	speed of light
c_1	linear viscous coefficient
c_2	quadratic viscous coefficient
c_k	sound speed of the k th material
c_s	sound speed
$C_{Vi/e}$	ion/electron specific heat
e	internal energy
e	material internal energy
e_k	internal energy of the k th material
e_r	radiation energy density
$e_{i/e}$	ion/electron fluid specific energy density
f_k	volume fraction of the k th material
h	Planck's constant
I	radiation specific intensity
j	determinant of incremental deformation gradient (incremental volume element)
K	kinetic energy $\frac{1}{2}\mathbf{u}^T\mathbf{u}$
k	Boltzmann's constant
m	element mass
q	artificial viscosity
q	heat conduction flux
r^2	linear fit residual
s	slope
S_e	energy source term
s_s	shock speed linear coefficient
t	time
$T_{i/e}$	ion/electron fluid temperature
u	one dimensional velocity
V	Volume
W	Lagrangian shock speed
x	one dimensional coordinate
$(\cdot)^{n+\alpha}$	quantity (\cdot) at time $t^{n+\alpha}$
\mathbf{b}	body force
\mathbf{b}_i	nodal 'strain-displacement' vector for component i
\mathbf{h}_j	hourglass vector for mode j
$\mathbf{N}(\boldsymbol{\xi})$	vector of element shape functions
\mathbf{r}_m^{hg}	hourglass rate for mode m
\mathbf{f}	incremental deformation gradient

Subscripts

j one dimensional mesh index

Symbols

α modern viscosity limiter

β	kinetic energy remap limiter
$\Delta_j x$	one dimensional coordinate undivided difference
$\Delta_j u$	one dimensional velocity undivided difference
Δt	time step size
η	artificial viscous coefficient
η	electric resistivity
γ	adiabatic coefficient
ϕ	incremental motion
κ	opacity
$\kappa_{i/e}$	ion/electron thermal conductivity
μ_0	permeability of free space
ν	frequency
ν	photon frequency
Ω	direction of photon
Ψ	integral of coordinate times variable
ψ	general remapped variable
Π	reconstructed local polynomial
ρ	density
σ_a	radiation absorption cross section
τ	$1/\rho$ fluid specific volume
τ_p	pressure relaxation time
τ_{ei}	collision time between electrons and ions
θ	local coordinate for polynomial reconstruction
Σ	artificial stress tensor
ξ	element natural coordinates
ξ	general coordinate, x , volume or mass
$\zeta_{i,j}$	quality metric for corner j of element i
<i>Superscripts</i>	
n	time index

I. Introduction

THE use of arbitrary Lagrangian-Eulerian (ALE) computer codes has been an enabling technology for many important defense-related applications. These computer codes are developed through combining modern algorithms for Lagrangian hydrodynamics, meshing technology and remap methods developed for high-resolution Eulerian methods. ALE methods were introduced in 1974 by Hirt, Amsden, and Cook,¹ but needed the development and wide-spread use of high-resolution method such as FCT,² or slope-limiters³ to become accurate enough for practical application. This was due to the overly dissipative remap associated with the use of upwind or donor-cell differencing. The upwind-type of differencing was necessary to produce physically bounded quantities in the remap essential for challenging problems.

In addition, ALE methods have benefited from the development of modern Lagrangian methods in the past 15 years. Two primary thrusts have shaped Lagrangian hydrodynamics during this period, one is the form of the discrete difference equations, and the second is the form of the artificial viscosity used to compute shocked solutions. The discrete difference equations have been improved radically through the use of mimetic principles, symmetry and conservation.⁴⁻⁹ The second development has utilized the technology used to develop high-resolution methods to limit the amount of dissipation in calculations.^{5,10,11} These methods combined to improve the quality of Lagrangian integrators substantially over the classical methods.¹²⁻¹⁴

Finally, the meshing methods have improved, albeit to a lesser extent than the integration method. Thus, meshing methods remain an active area of research with the promise of determining progress in achieving higher quality with ALE simulations.

In order to reap the rewards of the advances in ALE methodology, the ALEGRA project was initiated. The origins of the ALEGRA shock wave physics code project can be found in two internal Sandia memorandums that were written in the spring of 1988. These called for the planning of a code development project to develop a 3-D radiation hydrodynamics capability at Sandia. At this point, it is useful to describe some of the main assumptions that entered into the formal proposal.¹⁵

- The project was viewed as long term.
- The project was intended to be an advanced development project for computational shock wave physics as well as for the ICF effort.
- The project was intended to start with an existing code.
- The new code would need to be capable of modeling large changes in length scales and turbulence to as well as complicated coupling between the hydrodynamics and other physical phenomena important to ICF. Complex 3-D geometries were expected to be of importance in the code applications.
- The code development strategy, as well as the software implementation, needed to be flexible and open-ended. This ultimately was a major contributing factor to the decision to break new ground and use C++ as the major development language for the project by 1992.
- The code was to be developed for massively parallel computing platforms, although the machine architecture that this might include was not at all obvious in 1989.
- Pre- and post-processing tools were not to be developed under specific expenditures of this project, because of scarcity of resources, as well as the existence of additional projects intended to provide solutions in these areas.

Based on these assumptions, it was concluded that the best way to meet the prescribed objectives was to write an arbitrary connectivity multi-material arbitrary Lagrangian Eulerian (MMALE) code. The chosen development kernel with which to start was the PRONTO code,^{16,17} a Lagrangian arbitrary (but fixed) connectivity (finite element) 3-D code in existence at Sandia at the time of the planning culmination. The Lagrangian mode would accommodate changes in length scale with a conforming mesh, while the Eulerian mode (to be added) would accommodate the complex shearing and fluid distortion associated with turbulent flows.

The proposal also regarded the new code as a potential and logical next-generation extension of the CTH code,¹⁸ a production 3-D Eulerian code developed by Sandia and in wide use by the early 1990s. Though generally highly successful, CTH proved inadequate for certain classes of problems, such as capsule implosion simulations. Needless to say, a variety of new physics was required that was not implemented in either CTH or PRONTO: radiation transport, electron-ion two-temperature fluid approximations to plasma flows, electron thermal conduction, fusion burn physics, charged particle beam deposition, and potentially MHD physics associated with fast Z-pinches.

The code project which was born during the two year period of 1988 and 1989 finally started formally in March of 1990. An early 2D Fortran incarnation called RHALE was available later that same year.¹⁹ In the late fall of 1990, the decision was made to recast RHALE into an object-oriented structure using the C++ programming language. The advantages of this change included: enhanced data structures and memory management, object-oriented programming constructs, excellent debugging and code development environments, and improved ability to use massively parallel computing hardware.

The remainder of the paper is organized into eight sections. In the next section we discuss the underlying numerical scheme for the Lagrangian, remap and remesh steps. Next, we discuss the multimaterial aspects of ALEGRA, followed by the description of several important multi-physical processes: magneto-hydrodynamics, heat conduction, and radiation transport. This is followed by a discussion of the software infrastructure used to develop ALEGRA. Finally we will present the use of ALEGRA for several ALE applications spanning a broad physical range.

II. Fundamentals

II.A. Governing Equations

The continuum evolution equations for multi-material single fluid radiation-magnetohydrodynamics with thermal conduction and resistive magnetic diffusion in a two-temperature (ion-electron) approximation are:

1. Conservation of mass

$$\frac{\partial f_k \rho_k}{\partial t} = -\nabla \cdot (f_k \rho_k (\mathbf{u} - \mathbf{u}_g)). \quad (1)$$

2. Conservation of momentum

$$\frac{\partial \rho \mathbf{u}}{\partial t} = -\nabla \cdot (\rho (\mathbf{u} - \mathbf{u}_g) \mathbf{u} - \mathbf{T} - \mathbf{T}^M + \mathbf{P}_r) + \mathbf{b}. \quad (2)$$

3. Conservation of total energy

$$\begin{aligned} \frac{\partial \rho \left(e + e_r + \frac{1}{2} \mathbf{u}^T \mathbf{u} + \frac{1}{2\rho\mu_0} \mathbf{B}^T \mathbf{B} \right)}{\partial t} = & - \nabla \cdot \left(\rho (\mathbf{u} - \mathbf{u}_g) \left(e + e_r + \frac{1}{2} \mathbf{u}^T \mathbf{u} + \frac{1}{2\rho\mu_0} \mathbf{B}^T \mathbf{B} \right) \right) \\ & - \nabla \cdot (\mathbf{u} (-\mathbf{T} - \mathbf{T}^M + \mathbf{P}_r) - \mathbf{q}) + \mathbf{J} \cdot \mathbf{E}' + S_e. \end{aligned} \quad (3)$$

For ion-electron temperatures the non-conservative form of the energy equation becomes

$$\rho \frac{de_e}{dt} = \mathbf{T}_e : \nabla \mathbf{u} - \nabla \cdot \mathbf{q}_e + \mathbf{J} \cdot \mathbf{E}' + \rho C_{Ve} \frac{\theta_i - \theta_e}{\tau_{ei}} - \int_{4\pi} \int_0^\infty \sigma_a (B_\nu(T_e) - I) d\nu d\Omega, \quad (4)$$

$$\rho \frac{de_i}{dt} = \mathbf{T}_i : \nabla \mathbf{u} - \nabla \cdot \mathbf{q}_i + \rho C_{Vi} \frac{\theta_e - \theta_i}{\tau_{ei}}. \quad (5)$$

4. Faraday's Law in Moving Media

$$\frac{\partial \mathbf{B}}{\partial t} + \nabla \times (\mathbf{B} \times (\mathbf{u} - \mathbf{u}_g)) + (\mathbf{u} - \mathbf{u}_g) (\nabla \cdot \mathbf{B}) + \nabla \times \mathbf{E}' = 0. \quad (6)$$

5. Involution constraint on magnetic flux density

$$\nabla \cdot \mathbf{B} = 0. \quad (7)$$

6. Ampere's Law (neglecting displacement current)

$$\nabla \times \mathbf{H} = \mathbf{J}. \quad (8)$$

7. The Boltzmann transport equation is used to describe the thermal radiation transport with the specific intensity $I(\mathbf{r}, \Omega, \nu, t) = ch\nu f$ as the primary variable, where $f(\mathbf{x}, \Omega, \nu, t)$ is a phase space density, ν is the photon frequency, h is Planck's constant, and c is the speed of light. The Boltzmann equation for radiation transport is²⁰⁻²³

$$\frac{1}{c} \frac{\partial I}{\partial t} + \Omega \cdot \nabla I = \sigma_a (B_\nu(T_e) - I). \quad (9)$$

It is important to note that we do not include the radiation pressure or photon scattering in the evolution equations as it is assumed to be negligible for the circumstances modeled by ALEGRA .

We must also specify the closure equations

$$\begin{aligned} \mathbf{T} &= \mathbf{T}_e(\rho, \theta_e, \theta_i) + \mathbf{T}_i(\rho, \theta_i), \\ \mathbf{T}^M &= \nu_0 \left(\mathbf{B} \otimes \mathbf{B} - \frac{1}{2} \mathbf{B}^2 \mathbf{I} \right), \\ \mathbf{E}' &= \eta \mathbf{J} = \sigma^{-1} \mathbf{J} \\ \mathbf{H} &= \mathbf{B} / \mu_0 = \nu_0 \mathbf{B}, \\ \mathbf{q}_e &= -\kappa_e \nabla \theta_e, \\ \mathbf{q}_i &= -\kappa_i \nabla \theta_i, \\ e_r &= \frac{1}{c} \int_{4\pi} I_\nu d\Omega, \\ \mathbf{F}_r &= \int_{4\pi} \Omega I_\nu d\Omega, \\ \mathbf{P}_r &= \frac{1}{c} \int_{4\pi} \Omega \Omega I_\nu d\Omega. \end{aligned}$$

The equation for the material stress may not actually be given in functional form. We may also have a hypoelastic stress rate equations for the stress in the material configuration \mathbf{S} or $dev(\mathbf{S})$ given by formally by

$$\begin{aligned}\dot{\mathbf{S}} &= \mathbf{f}(\mathbf{c}, \mathbf{S}), \\ \mathbf{c} &= \mathbf{R}^T \mathbf{D} \mathbf{R}, \\ \mathbf{T} &= \mathbf{R} \mathbf{S} \mathbf{R}^T,\end{aligned}$$

where the stretch and rotation are computed via a rate equation

$$\begin{aligned}\mathbf{F} &= \mathbf{V} \mathbf{R} \\ \dot{\mathbf{V}} &= (\mathbf{D} + \mathbf{W}) \mathbf{V} - \mathbf{V} \Omega, \\ \dot{\mathbf{R}} &= \Omega, \\ \mathbf{Z} &= \mathbf{D} \mathbf{V} - \mathbf{V} \mathbf{D} \\ \boldsymbol{\omega} &= \mathbf{w} + (Tr(\mathbf{V}) \mathbf{I} - \mathbf{V})^{-1} \mathbf{z}.\end{aligned}$$

where \mathbf{D} and \mathbf{W} are the symmetric and anti-symmetric part of the velocity gradient tensor, \mathbf{L} and \mathbf{z} , \mathbf{w} , $\boldsymbol{\omega}$ are the axial vectors corresponding to \mathbf{Z} , \mathbf{W} and Ω .

II.B. Operator Splitting Methodology

The full set of ALEGRA equations are operator split in time. The three primary are the Lagrangian step in which the equations are solved on a moving mesh, the remesh step in which the relative mesh velocity is computed, and the remap step in which the degrees of freedom are computed on the new mesh. The spatial operators should be considered as acting in the *current* frame of reference \mathbf{x} . The involution constraint $\nabla \cdot \mathbf{B} = 0$ is required to hold at the beginning of the computation and each step is required to preserve this constraint.

1. Lagrangian Step

(a) Ideal magnetohydrodynamics (MHD) in a moving frame

The ideal magnetohydrodynamic Lagrangian step moves the mesh with the material velocity, \mathbf{u} , according to the thermodynamic and magnetic stresses. The step does not include any explicit physical dissipation mechanisms but does include artificial viscosity.

$$\dot{\mathbf{x}} = \mathbf{u} \quad (10)$$

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{u}) = 0 \quad (11)$$

$$\rho \dot{\mathbf{u}} = \nabla \cdot (\mathbf{T} + \mathbf{T}^M) + \mathbf{b} \quad (12)$$

$$\rho \dot{e} = \mathbf{T} \cdot \nabla \mathbf{u} + S_e \quad (13)$$

$$\frac{\partial \mathbf{B}}{\partial t} + \nabla \times (\mathbf{B} \times \mathbf{u}) + \mathbf{u} (\nabla \cdot \mathbf{B}) = 0 \quad (14)$$

It easier to understand the proper implementation of the above balance laws in the corresponding integral form

$$\frac{d}{dt} \int_{\Omega_t} \rho \, dv = 0 \quad (15)$$

$$\frac{d}{dt} \int_{\Omega_t} \rho \dot{\mathbf{u}} \, dv = \int_{\Omega_t} \nabla \cdot (\mathbf{T} + \mathbf{T}^M) \, dv \quad (16)$$

$$\frac{d}{dt} \int_{\Omega_t} \rho e \, dv = \int_{\Omega_t} \mathbf{T} \cdot \nabla \mathbf{u} \, dv \quad (17)$$

$$\frac{d}{dt} \int_{\Gamma_t} \mathbf{B} \cdot \mathbf{n} \, dA = 0. \quad (18)$$

where the integration regions are moving with the material.

(b) Magnetic diffusion in a fixed frame

$$\frac{\partial \mathbf{B}}{\partial t} + \nabla \times \mathbf{E}' = 0 \quad (19)$$

$$\nabla \times \mathbf{H} = \mathbf{J} \quad (20)$$

$$\rho \frac{\partial e}{\partial t} = \mathbf{J} \cdot \mathbf{E}'. \quad (21)$$

(c) Conduction in a fixed frame

$$\rho \frac{\partial e}{\partial t} = -\nabla \cdot \mathbf{q}, \quad (22)$$

or

$$\rho \frac{\partial e_e}{\partial t} = -\nabla \cdot \mathbf{q}_e + \rho C_{Ve} \frac{\theta_i - \theta_e}{\tau_{ei}} - \int_0^\infty (\kappa (4\pi B_\nu - cE_\nu)) d\nu,$$

$$\rho \frac{\partial e_i}{\partial t} = -\nabla \cdot \mathbf{q}_i + \rho C_{Ve} \frac{\theta_e - \theta_i}{\tau_{ei}}.$$

(d) Radiation energy is remapped in the Lagrangian step to place the degrees of freedom at the new Lagrangian mesh location.

2. Remesh Step

Compute a new desired mesh which may be the original mesh at the beginning of the time step or a smoothed version of the current mesh. This effectively defines a relative velocity \mathbf{u}_g for the remap.

3. Remap Step

A local remap operator associated with the pseudo relative velocity act on the degrees of freedom to place them on the new mesh.

$$\frac{d\mathbf{x}}{dt} = \mathbf{u}_g - \mathbf{u} \quad (23)$$

$$\frac{\partial f_k}{\partial t} = -(\mathbf{u} - \mathbf{u}_g) \cdot \nabla f_k \quad (24)$$

$$\frac{\partial_r f_k \rho_k}{\partial t} = -\nabla \cdot (f_k \rho_k (\mathbf{u} - \mathbf{u}_g)) \quad (25)$$

$$\frac{\partial_\rho \mathbf{u}}{\partial t} = -\nabla \cdot (\rho \mathbf{u} (\mathbf{u} - \mathbf{u}_g)) \quad (26)$$

$$\frac{\partial_\rho e}{\partial t} = -\nabla \cdot (\rho e (\mathbf{u} - \mathbf{u}_g)) \quad (27)$$

$$\frac{\partial_\rho K}{\partial t} = -\nabla \cdot (\rho K (\mathbf{u} - \mathbf{u}_g)) \quad (28)$$

and

$$\frac{\partial \mathbf{B}}{\partial t} = -\nabla \times (\mathbf{B} \times (\mathbf{u} - \mathbf{u}_g)) - (\mathbf{u} - \mathbf{u}_g) (\nabla \cdot \mathbf{B}). \quad (29)$$

4. Radiation Step (computed in a fixed frame of reference)

$$\frac{1}{c} \frac{\partial I}{\partial t} + \boldsymbol{\Omega} \cdot \nabla I = \sigma_a (B_\nu(T_e) - I).$$

$$\frac{\partial e_e}{\partial t} = - \int_{4\pi} \int_0^\infty \sigma_a (B_\nu(T_e) - I) d\nu d\boldsymbol{\Omega}.$$

II.C. Lagrangian Step

Presently, ALEGRA employs a classical time-staggered method. In this method the velocities are staggered in time as well as space with respect to the thermodynamic variables (density, pressure, and energy). Note that masses can be used to simplify the differencing, $m = \rho V$ or $\rho \Delta x$, which is a Lagrangian invariant. It is instructive to describe the method in the simplest setting of one-dimensional pure hydrodynamics before discussing the finite element implementation used in ALEGRA. Within this method, the concept of pushing variables forward at second-order accuracy is conceptually simple (in one-dimension),

$$\mathbf{u}_{j+1/2}^{n+1/2} = \mathbf{u}_{j+1/2}^{n-1/2} - \Delta t \frac{p_{j+1}^n - p_j^n + q_{j+1}^{n-1/2} - q_j^{n-1/2}}{m_j + m_{j+1}} \quad (30)$$

$$\mathbf{x}_{j+1/2}^{n+1} = \mathbf{x}_{j+1/2}^n + \Delta t \mathbf{u}^{n+1/2}. \quad (31)$$

the density is then given by $\rho^{n+1} = \rho^n V^n / V^{n+1}$ where the volume, V is computed using the coordinates, x . For example in one dimension $V_j = \Delta x = x_{j+1/2} - x_{j-1/2}$.

$$\mathbf{e}_j^{n+1} = \mathbf{e}_j^n - \Delta t \left(p^n + q_j^{n-1/2} \right) \frac{\mathbf{u}_{j+1/2}^n - \mathbf{u}_{j-1/2}^n}{m_j}, \quad (32)$$

where $\mathbf{u}_{j+1/2}^n = 1/2 \left(\mathbf{u}_{j+1/2}^{n-1/2} + \mathbf{u}_{j+1/2}^{n+1/2} \right)$ is the time averaged velocity to consistently difference the energy update in time. It is notable that the differencing of coordinates/mass and momentum is second-order in time and space, but Equation 32 is first-order accurate in time.

In reality these equations are discretized in space using the finite element method (described in Section II.E) using the above described time differencing. In two or three dimensions the velocities are defined at the corners of the elements and other quantities (not true for magnetics) at element centers.

We also note that the numerical properties of the staggered integration scheme may have beneficial properties for applications where material strength (and linear wave propagation) are especially important. The numerical properties of the staggered integration scheme are neutral dissipation and excellent dispersion properties. Conversely, for propagation of step functions, the staggered mesh integrators have extremely bad properties. As such the optimal viscosity formulation should be different from forward-in-time methods. Thus applications such as flyer plates and penetration may benefit greatly from this scheme.

II.D. Incremental kinematics

For solid mechanics the update of the material state is an essential detail. The goal of this subsection is to describe the time-discrete kinematics used in ALEGRA. Define the midpoint configuration of the body as the convex combination

$$\boldsymbol{\varphi}^{n+\alpha} = (1 - \alpha)\boldsymbol{\varphi}^n + \alpha\boldsymbol{\varphi}^{n+1} \quad \text{for } \alpha \in [0, 1]. \quad (33)$$

Consistent with this, the midpoint deformation gradient is the convex combination

$$\mathbf{F}^{n+\alpha} = (1 - \alpha)\mathbf{F}^n + \alpha\mathbf{F}^{n+1} \quad \text{for } \alpha \in [0, 1]. \quad (34)$$

The incremental motion relating the configurations Ω^n and Ω^{n+1} is

$$\boldsymbol{\phi} = \boldsymbol{\varphi}_{n+1} \circ (\boldsymbol{\varphi}^n)^{-1} = \mathbf{id} + \mathbf{d}^{n+1}, \quad (35)$$

where \mathbf{id} is the identity mapping. By this construction

$$\boldsymbol{\phi} : \Omega^n \longrightarrow \Omega^{n+1}. \quad (36)$$

The incremental displacement field is defined as

$$\mathbf{d}^{n+1} = (\boldsymbol{\varphi}^{n+1} - \boldsymbol{\varphi}^n) \circ (\boldsymbol{\varphi}^n)^{-1}. \quad (37)$$

The displacement field \mathbf{d}^{n+1} can be evaluated as a function of \mathbf{x}^n or as a function of $\mathbf{x}^{n+\alpha}$ using the operator compositions

$$\tilde{\mathbf{d}}^{n+1} := \mathbf{d}^{n+1} \circ \boldsymbol{\varphi}^n \circ (\boldsymbol{\varphi}^{-1})^{n+\alpha} = (\boldsymbol{\varphi}^{n+1} - \boldsymbol{\varphi}^n) \circ (\boldsymbol{\varphi}^{n+\alpha})^{-1}. \quad (38)$$

Figure 1 provides a visual representation of these incremental kinematics.

The incremental deformation gradient is defined as

$$\mathbf{f}^{n+\alpha} = \text{grad}_n[\varphi^{n+\alpha} \circ (\varphi^n)^{-1}] = \mathbf{F}^{n+\alpha}(\mathbf{F}^n)^{-1}, \quad (39)$$

with an incremental volume element

$$j^{n+\alpha} = \det[\mathbf{f}^{n+\alpha}]. \quad (40)$$

To compute this deformation from the incremental displacement field, notice that

$$\mathbf{f}^{n+\alpha} = (1-\alpha)\mathbf{I} + \alpha\mathbf{f}^{n+1} = \mathbf{I} + \alpha \text{grad}_n[\mathbf{d}^{n+1}]. \quad (41)$$

It is possible to accumulate the total displacement field, if desired. This field is simply the accumulated sum

$$\mathbf{U}^{n+1} := \varphi^{n+1} - \varphi_0 = \sum_{i=0}^n (\varphi^{i+1} - \varphi^i) = \sum_{i=0}^n \mathbf{d}^{i+1} \circ \varphi_i \quad (42)$$

There are at least two ways to compute the total deformation using the total displacement field:

1.

$$\mathbf{F}^{n+1} = \mathbf{I} + \nabla_{\mathbf{x}}[\mathbf{U}^{n+1}]$$

2.

$$(\mathbf{F}^{n+1})^{-1} = \mathbf{I} - \text{grad}_{n+1}[\mathbf{U}^{n+1} \circ (\varphi^{n+1})^{-1}]$$

3.

$$\mathbf{F}^{n+\alpha} = \nabla_{\mathbf{x}}[\varphi^{n+\alpha}] = (1-\alpha)\mathbf{F}^n + \alpha\mathbf{F}^{n+1}$$

II.E. Spatial Interpolations

In almost all regards ALEGRA uses a standard finite element spatial interpolations. Uniform strain isoparametric tensor product elements transformed from the parent domain $[-1, 1]^{N_{\text{dim}}}$ are used. These correspond to four-node quadrilateral elements in two dimensions and eight-node hexahedral elements in three dimensions. These are the simplest (non-locking) elements available for general purpose finite element analysis. Standard nodal shape functions are used to interpolate continuous fields such as position, velocity and acceleration. Shape function gradients are calculated by appropriate transformation of natural coordinate gradients. Any needed kinematic tensor fields are computed using the shape function gradients.

Let the domain of the body be denoted by $\Omega \subset \mathbb{R}^3$, with boundary Γ . All relevant (thermodynamic) fields, which include but are not limited to, deformation rate, stress, pressure, density and internal energy, are assumed spatially constant in each element. Thus position and velocity are $C^0(\Omega) \subset H^1(\Omega)$ fields defined by nodal values and thermodynamic quantities are $L^2(\Omega)$ fields defined by piecewise constant element values. This is often referred to as a *staggered grid* spatial interpolation.

For standard finite element analysis,

$$\mathbf{x} = \sum_{A=0}^{N_{\text{nodes}}-1} N^A \mathbf{x}^A \implies \text{grad}[\mathbf{x}] = \sum_{A=0}^{N_{\text{nodes}}-1} \mathbf{x}^A \otimes \text{grad}[N^A], \quad (43)$$

where \mathbf{x}^A is the value of the position \mathbf{x} at node A and the shape function for node A is denoted by N^A . The balance of linear momentum in matrix form is

$$\sum_{B=0}^{N_{\text{nodes}}-1} M^{AB} \mathbf{a}^B + \mathbf{F}^{A,\text{int}} - \mathbf{F}^{A,\text{ext}} = \mathbf{0}, \quad (44)$$

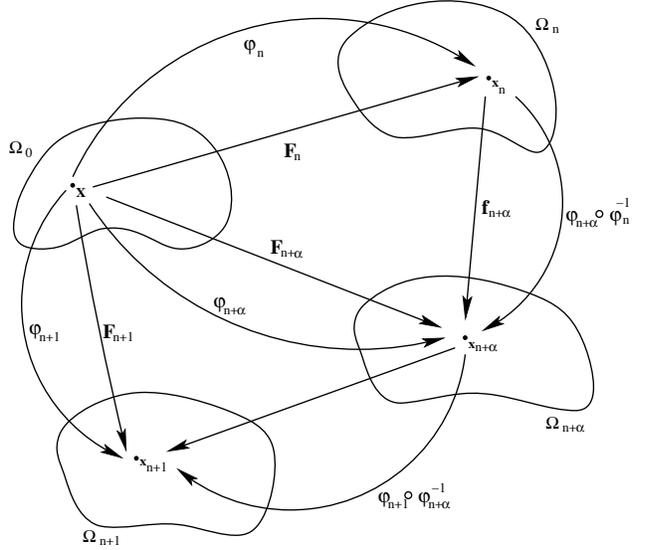


Figure 1. Incremental kinematics of motion of a deforming body.

where

$$M^{AB} = \int_{\Omega} \rho N^A N^B \, d\Omega, \quad (45)$$

is the mass matrix,

$$\mathbf{F}^{A,\text{int}} := \int_{\Omega} \mathbf{T} \, \text{grad}[N^A] \, d\Omega, \quad (46)$$

is the internal force vector at node A and

$$\mathbf{F}^{A,\text{ext}} := \int_{\Omega} N^A \rho \mathbf{b} \, d\Omega + \int_{\Gamma} N^A \bar{\mathbf{t}} \, d\Gamma, \quad (47)$$

is the external force vector at node A .

If the mass matrix is lumped then

$$\bar{M}^A := \sum_{B=0}^{N_{\text{nodes}}-1} \left[\int_{\Omega} \rho N^A N^B \, d\Omega \right] = \left[\int_{\Omega} \rho N^A \, d\Omega \right],$$

and $M^{AB} = \bar{M}^A \delta^{AB}$ (no sum on A). The balance of energy equation, in matrix form, for an element with domain Ω_e is

$$M_e \dot{\varepsilon} = \left[\sum_{A=0}^{N_{\text{nodes}}-1} \mathbf{F}^{A,\text{int}} \bullet \mathbf{u}^A \right], \quad (48)$$

where M_e is the mass of the element and \mathbf{u} is the velocity field.

II.F. Central-Difference Method

The central difference algorithm is implemented as follows:

1. Compute acceleration at time t^n

$$\sum_{B=0}^{N_{\text{nodes}}-1} M^{AB} (\mathbf{a}^n)^B + (\mathbf{F}^n)^{A,\text{int}} - (\mathbf{F}^n)^{A,\text{ext}} = \mathbf{0} \quad (49)$$

2. Compute velocity at time $t^{n+1/2}$:

$$\mathbf{u}^{n+1/2} = \mathbf{u}^{n-1/2} + (t^{n+1/2} - t^{n-1/2}) \mathbf{a}^n \quad (50)$$

3. Compute position at time t_{n+1} :

$$\mathbf{x}^{n+1} = \mathbf{x}^n + (t^{n+1} - t^n) \mathbf{u}^{n+1/2} \quad (51)$$

4. Compute velocity at time t_n :

$$\mathbf{u}^n = \frac{1}{2} \left[\mathbf{u}^{n-1/2} + \mathbf{u}^{n+1/2} \right] \quad (52)$$

5. Compute energy at time t^{n+1} :

$$M_e (\varepsilon^{n+1} - \varepsilon^{n+1}) = (t^{n+1/2} - t^{n-1/2}) \left[\sum_{A=0}^{N_{\text{nodes}}-1} \mathbf{F}_n^{A,\text{int}} \bullet (\mathbf{u}^n)^A \right] \quad (53)$$

6. Compute the stress at time t^{n+1} .

This procedure is equivalent to the simpler description for simple 1-D hydrodynamics given by Equations 30–32.

II.F.1. Artificial viscosity

In shock physics calculations, physical viscosity is truly negligible and has space and time scales much smaller than those resolved by the grid. The weak, or integral, Euler equations are inherently ill-posed (having an infinite number of solutions). Computational solutions involving shocks, without some form of stability control, usually have severe oscillations because of this. Mathematically, it is common practice to add a viscosity term to shock equations and consider the unique solution arising in the limit as the viscosity vanishes.^{24–26} This approach is known as viscosity solutions. Computationally, a similar effect can be achieved by use of an artificial viscosity to stabilize the computed solution about a shock.

Like physical viscosity, artificial viscosity appears in both the fluid momentum and energy equations through the stress tensor. In the momentum equation the artificial viscosity stress, Σ_{av} , should act to reduce momentum at the shock front, and hence also reduce the kinetic energy there. In the energy equation, the kinetic energy lost should be dissipated (and entropy produced) by raising the fluid internal energy ($\Sigma_{av} \cdot \nabla \mathbf{v} \geq 0$). Thus, like physical viscosity, artificial viscosity acts to convert kinetic energy into internal energy at the shock front.

The basic Von Neumann-Richtmyer artificial viscosity idea is to exchange the space and time scales from the physical viscosity for ones applicable to the grid and fluid velocity. They replaced the term $v_{th}\lambda$ in μ with the product of a term proportional to the spatial gradient of the velocity and a space-scale squared. The space-scale is selected to be a grid length, resulting in a compact smearing of the shock. This basic idea,

$$\Sigma_{visc,quad} = (\rho h^2 |\nabla \mathbf{u}|) |\nabla \mathbf{u}| \leftarrow (\rho v_{th} \lambda) |\nabla \mathbf{u}|, \quad (54)$$

results in a term quadratic in the velocity gradient. In multidimensional applications, the simplest approach is to use a scalar coefficient and to compute $|\nabla \mathbf{v}|$ as $\nabla \cdot \mathbf{v}$, independent of shock direction (since evaluating the shock direction robustly remains an open problem). This simplest application in multi-D is what ALEGRA uses for its quadratic term,

$$\Sigma_{visc,quad,ALEGRA} = Q_{quad} = (\rho l^2 \nabla \cdot \mathbf{u}) \nabla \cdot \mathbf{u}, \quad (55)$$

where l is the aspect ratio. The differences between the two types of artificial viscosity are evident in the solution to Noh's shock reflection²⁷ shown in Figure 2.

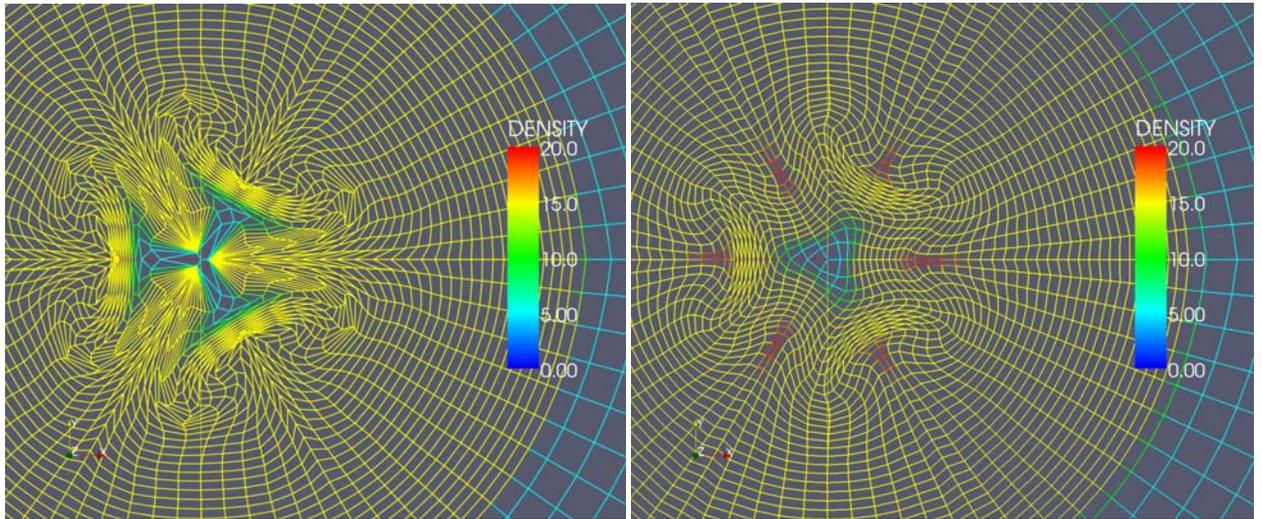


Figure 2. Note the difference in the quality of the mesh and solution for Noh's shock reflection in cylindrical coordinates between the original scalar (left) and tensor (right) artificial viscosity.

This term is quadratic in $\nabla \mathbf{v}$, whereas the physical viscous stress is only linear in $\nabla \mathbf{v}$. The second $\nabla \mathbf{v}$ in the quadratic term really serves only to help define the space and time scales over which the artificial viscosity acts. An alternative is to use the sound speed, c_s , to obtain those space/time scales. This results in the linear term used in ALEGRA ,

$$\Sigma_{visc,lin,ALEGRA} = Q_{lin} = (\rho l c_s) \nabla \cdot \mathbf{u}, \quad (56)$$

The standard viscosity formulation uses a linear plus quadratic term proportional to the velocity and is specified to only operate when a zone is in compression.

$$q_j = \rho \left(c_1 c_j |\Delta_j u| + c_2 (\Delta_j u)^2 \right) \quad (57)$$

where $\Delta_j u = u_{j+1/2} - u_{j-1/2}$. The coefficients are often described as being arbitrary positive values, although there is substantial reason to believe that their values should be clearly defined. The default values in ALEGRA are $c_1 = 0.15$, and $c_2 = 2.0$. Artificial viscosity is only applied if the fluid is in compression, and not on expansion where the fluid should behave adiabatically. Based on this reasoning, the artificial viscosity uses the following logic,

$$q_j = \begin{cases} 0 & \text{if } \Delta_j u \geq 0 \\ \rho \left(c_1 c_j |\Delta_j u| + c_2 (\Delta_j u)^2 \right) & \text{if } \Delta_j u < 0 \end{cases} \quad (58)$$

In inviscid shock physics, the only mechanism for generating dissipation in the flow dynamics is during the passage of a shock wave which requires the divergence of velocity to be negative (see more below).

II.F.2. Modern Artificial Viscosity

The starting point for this discussion comes from the analytical values of the linear and quadratic coefficients of viscosity. The starting point are the Rankine-Hugoniot conditions at a shock,

$$W [u] = [p], \quad (59)$$

where $[u] = u_1 - u_0$ is the jump across the shock here the convention is that u_1 is the post shock state and u_0 is the unshocked state. One can rearrange Equation 59 into the following useful form,

$$p_1 = p_0 + W (u_0 - u_1), \quad (60)$$

noting the reversed sign convention with the velocity difference. Now approximate the shock speed in the following fashion, $W = \rho_0 (c_s + s_s (u_0 - u_1))$. This form has been used to reduce flyer plate experimental data, and $(u_0 - u_1)$ is often referred to as the particle velocity (with u_0 often being stationary). The form for an ideal gas is more complex although it has very simple expressions for s in the limit where $(u_0 - u_1)/c_s \rightarrow 0$ with $s_s = 1/4(\gamma + 1)$ and where $(u_0 - u_1)/c_s \rightarrow \infty$ with $s_s = 1/2(\gamma + 1)$. The value of s_s is well-defined for many materials and is part of the basis of the Mie-Gruneisen equation of state. We can then see that analytically, $c_1 = 1$ and $c_2 = s_s$.

Christenson introduced a flux-limited artificial viscosity (as documented in the open literature by Benson¹¹) that incorporates detection of discontinuous or under-resolved flows developed for Eulerian hydrodynamics with artificial viscosity. Effectively, the method allows more dissipation to be applied at a shock, but less away from the shock. The mechanism to produce this effect was borrowed from TVD methods^{28,29} with the label flux limiters although the use is more consistent with slope-limiting applied to Godunov-type methods.^{3,30} At each cell face (or more properly centering point for the velocity), the first-term in a Taylor series is evaluated, and the velocity is extrapolated to the cell-centers. The difference in the extrapolated velocities takes the place of the velocity jumps used in the classical viscosity. The term in the Taylor series is “limited” so that it does not take unreasonable values when the velocities are discontinuous or under-resolved. The conditions that define an under-resolved flow are intimately related to the conditions that yield monotone (or high-resolution) advection methods.

We have developed a similar method especially adapted for use with our finite element framework. In a one-dimensional implementation the method is defined as follows:

1. Compute a second-order accurate value for the gradient of the velocity at each cell face corresponding to a linear least squares solution, $(\Delta_j x = x_{j+1/2} - x_{j-1/2}, \Delta_j u = u_{j+1/2} - u_{j-1/2})$,

$$\frac{\partial u}{\partial x_{j+1/2,C}} = \frac{(\Delta_{j+1}x)^3 \Delta_j u + (\Delta_j x)^3 \Delta_{j+1}u}{(\Delta_{j+1}x)^3 \Delta_j x + (\Delta_j x)^3 \Delta_{j+1}x}. \quad (61)$$

2. Determine the residual of the least square problem by computing the difference between the least square fit and the data,

$$r_{j+1/2}^2 = \left(u_{j-1/2} - u_{j+1/2} + \Delta_j - \frac{\partial u}{\partial x_{j+1/2,C}} \right)^2 + \left(u_{j+3/2} - u_{j+1/2} + \Delta_{j+1}x + \frac{\partial u}{\partial x_{j+1/2,C}} \right)^2, \quad (62)$$

3. then compute the coefficient that will modify the velocity differences used in the artificial viscosity,

$$\alpha_{j+1/2} = \sqrt{\frac{r^2}{(u_{j-1/2} - u_{j+1/2})^2 + (u_{j+3/2} - u_{j+1/2})^2}}. \quad (63)$$

4. use this coefficient to modify the magnitude of the velocity jump in an element,

$$\Delta_j u = 1/2 (\alpha_{j-1/2} + \alpha_{j+1/2}) (u_{j+1/2} - u_{j-1/2}). \quad (64)$$

The last item to consider is whether the viscosity should be turned off in the expansion. With a classical artificial viscosity experience has shown this to be unacceptable. With a flux-limited viscosity this may be worth considering because as the flow becomes resolved the viscosity shuts off, but is present when the flow is grossly under-resolved and can help numerical stability.

II.G. Hourglass control

The goal of this section is to review the hourglass control algorithms currently implemented in ALEGRA .

II.G.1. Shape Function Representations

In what follows, attention is primarily focused upon the eight(8)-node isoparametric tri-linear brick element for which $N_{\text{nodes}} = 8$. Following the presentation(s) in,³¹⁻³³ let $\mathbf{N}(\boldsymbol{\xi})$ denote the 8×1 vector of shape functions such that

$$\mathbf{N}(\boldsymbol{\xi}) = \{N^1, N^2, \dots, N^8\}^T \quad \text{with} \quad N^A := \frac{1}{8}(1 + \xi_1 \xi_1^A)(1 + \xi_2 \xi_2^A)(1 + \xi_3 \xi_3^A), \quad (65)$$

where $\boldsymbol{\xi}^A = \{\xi_1^A, \xi_2^A, \xi_3^A\}^T$ are the vertices of the bi-unit cube $[-1, 1]^3$. These shape functions may be written as

$$\mathbf{N}(\boldsymbol{\xi}) = \mathbf{b}_0 + \sum_{i=1}^3 x_i \mathbf{b}_i + \frac{1}{8} \sum_{j=1}^4 \mathcal{H}_j(\boldsymbol{\xi}) \boldsymbol{\gamma}_j, \quad (66)$$

where $\mathcal{H}_j(\boldsymbol{\xi})$ are the *hourglass functions* defined as

$$\mathcal{H}_1(\boldsymbol{\xi}) := \xi_2 \xi_3, \quad \mathcal{H}_2(\boldsymbol{\xi}) := \xi_3 \xi_1, \quad \mathcal{H}_3(\boldsymbol{\xi}) := \xi_1 \xi_2, \quad \mathcal{H}_4(\boldsymbol{\xi}) := \xi_1 \xi_2 \xi_3. \quad (67)$$

The twelve(12)-dimensional *hourglass deformation space* of the eight-node brick is defined as

$$H_{12} := \text{span}\{\mathcal{H}_1, \mathcal{H}_2, \mathcal{H}_3, \mathcal{H}_4\} \times R^3. \quad (68)$$

This space may be viewed as consisting of four(4) *hourglass modes*, where each mode can act in any one of three(3) coordinate directions.

The constant 8×1 vectors \mathbf{b}_0 , \mathbf{b}_i and $\boldsymbol{\gamma}_j$ can be determined in a relatively straight-forward manner. Evaluation of equation (66) at the nodes of the bi-unit cube and use of the Kronecker property $N^A(\boldsymbol{\xi}_B) = \delta_{AB}$ yields the 8×8 matrix expression

$$\mathbf{I}_8 = \mathbf{b}_0 \otimes \mathbf{1}_8 + \sum_{i=1}^3 \mathbf{b}_i \otimes \mathbf{x}_i + \frac{1}{8} \sum_{j=1}^4 \boldsymbol{\gamma}_j \otimes \mathbf{h}_j. \quad (69)$$

In this equation $\mathbf{1}_8$ is the 8×8 identity matrix, $\mathbf{x}_i = \{x_i^1, x_i^2, \dots, x_i^8\}^T$ is the 8×1 vector of nodal coordinates for component i and

$$\begin{aligned}\mathbf{1}_8 &:= \{1, 1, 1, 1, 1, 1, 1, 1\}^T \\ \mathbf{h}_1 &:= \{1, 1, -1, -1, -1, -1, 1, 1\}^T \\ \mathbf{h}_2 &:= \{1, -1, -1, 1, -1, 1, 1, -1\}^T \\ \mathbf{h}_3 &:= \{1, -1, 1, -1, 1, -1, 1, -1\}^T \\ \mathbf{h}_4 &:= \{-1, 1, -1, 1, 1, -1, 1, -1\}^T\end{aligned}\tag{70}$$

The reader may easily verify that $\mathbf{1}_8 \bullet \mathbf{h}_j = 0$ and $\mathbf{h}_j \bullet \mathbf{h}_k = 8\delta_{jk}$. The multiplication of equation (69) by the vector $\mathbf{1}_8$ yields the equation

$$\mathbf{b}_0 = \frac{1}{8} \left[\mathbf{1}_8 - \sum_{i=1}^3 (\mathbf{1}_8 \bullet \mathbf{x}_i) \mathbf{b}_i \right].\tag{71}$$

The multiplication of equation (69) by the vector \mathbf{h}_k yields the equation

$$\gamma_j = \left[\mathbf{h}_j - \sum_{i=1}^3 (\mathbf{h}_j \bullet \mathbf{x}_i) \mathbf{b}_i \right].\tag{72}$$

Taking the derivative of (66) with respect to x_k (or, alternatively, ξ_k) and evaluation at the element center $\boldsymbol{\xi} = \mathbf{0}$ yields

$$\mathbf{b}_i = \sum_{j=1}^3 (\mathbf{J}_0^{-T})_{ij} \frac{\partial \mathbf{N}}{\partial \xi_j}(\mathbf{0}) \quad \text{where} \quad \mathbf{J}_0 := \frac{\partial \mathbf{x}}{\partial \boldsymbol{\xi}}(\mathbf{0}).\tag{73}$$

Since \mathbf{b}_i are consistently computed 'strain-displacement' operators, one can show that (see,³⁴ chapter 3)

$$\mathbf{b}_i \bullet \mathbf{1}_8 = 0 \quad \text{and} \quad \mathbf{b}_i \bullet \mathbf{x}_j = \delta_{ij}.\tag{74}$$

Given these properties, it must also hold from (72) that

$$\gamma_i \bullet \mathbf{1}_8 = 0 \quad \text{and} \quad \gamma_i \bullet \mathbf{x}_j = 0,\tag{75}$$

and thus the vectors γ_i are *orthogonal to the homogeneous deformation modes* of the tri-linear brick element.

II.G.2. Hourglass Rates

Let $\mathbf{u}^h(A)$ for $A \in \{1, 2, \dots, N_{\text{nodes}}\}$ denote the nodal velocity field of a finite element. Consistent with the developments in^{35,31,36, 37} define the *hourglass rate* for mode m as

$$\mathbf{r}_m^{hg} := \sum_{A=1}^{N_{\text{nodes}}} \gamma_m(A) \mathbf{u}^h(A).\tag{76}$$

II.G.3. Hourglass Resistance

The goal here is to compute, based on the hourglass rates \mathbf{r}_m^{hg} , a reasonable value of the hourglass resistance \mathbf{f}_m^{hg} .

II.G.4. Hourglass Nodal Forces

Let $\mathbf{F}_m^{hg}(A)$ for $A \in \{1, 2, \dots, N_{\text{nodes}}\}$ denote the nodal hourglass force field for mode m of a tri-linear brick element. Once the hourglass resistance has been determined, the nodal forces for hourglass mode m are simply computed as

$$\mathbf{F}_m^{hg}(A) = \gamma_m(A) \cdot \mathbf{f}_m^{hg}.\tag{77}$$

II.H. Remesh

ALEGRA remesh/mesh-enhancement algorithms include Discrete Optimization (DO) schemes (via the MESQUITE library^{38,39}) and more standard equipotential methods (e.g. weighted and unweighted Tipton⁴⁰). These algorithms are implemented through an interface that supports development and testing of other remesh algorithms (e.g. Prescriptive Laplace-Beltrami methods⁴¹ are currently being examined via this interface.)

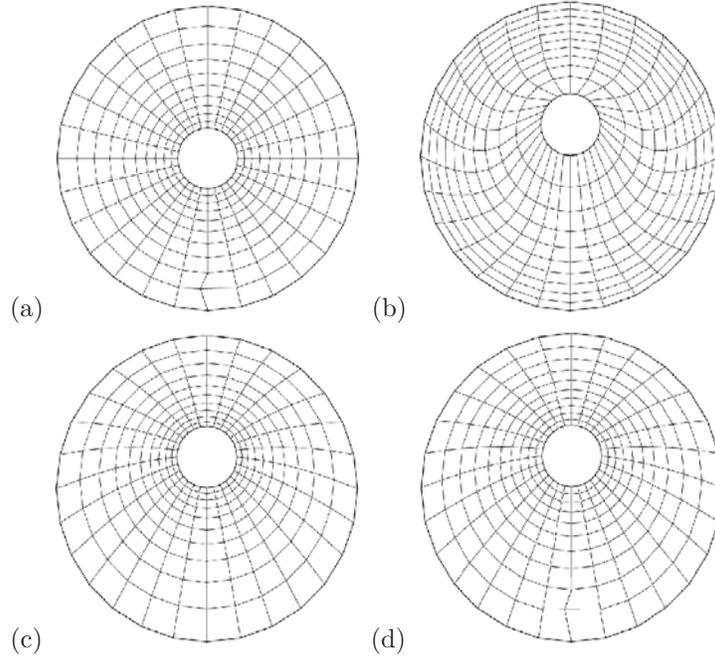


Figure 3. Application of ALEGRA's remeshing algorithms to the initial mesh shown in (a). ALEGRA's Tipton, Mesquite initial mesh, and ideal mesh results are shown ((b)—(d) respectively).

The equipotential methods, as available in ALEGRA, are well known and so will not be discussed here; rather the reader is directed to the substantial literature on the subject.^{40,41} However, DO techniques are relatively new for this application space, and so some generalities are provided below. Again, details are available elsewhere.^{38,39}

The MESQUITE-based mesh improvement strategies available in ALEGRA are defined by two main aspects: a scalar measure of mesh quality and the optimization method used to optimize the mesh (relative to that measure). The following subsections describe the DO approach taken in ALEGRA. Note that MESQUITE provides a much richer set of mesh-optimization tools than described here. The methodologies chosen for use in ALEGRA are those that best fit its application space.

A measure of mesh quality is defined by the quality metric, $\zeta_{i,j}$ for corner j of element i . The ζ are combined over appropriate elements and corners to produce an objective function,

$$\mathcal{F}(\mathbf{x}) = \sum_{i=1}^{N_{el}} \left(\frac{1}{k_i} \sum_{j=1}^{k_i} \hat{\zeta}_{i,j} \right) \quad (78)$$

where k_i is the number of corners in element i , N_{el} is the number of ALE elements in the mesh and \mathbf{x} is a vector of node coordinates. \mathcal{F} is minimized in the space of valid node locations and constrained by boundary nodes that are either fixed or constrained to a specific geometry. Currently ALEGRA supports fully unconstrained, fully constrained or planar surface constrained node movement.

The ζ_i used in ALEGRA are based on the Target Matrix paradigm.⁴² In this paradigm a $N_{dim} \times N_{dim}$ Jacobian matrix, $\mathbf{A}_{i,j}$, is calculated at the corners of each ALE element in mesh (N_{dim} is the dimension of the element). Also defined at each corner is a target (or reference) matrix, $\mathbf{W}_{i,j}$, providing a measure of the "ideal" element corner. Finally, a measure of the distance between $\mathbf{W}_{i,j}$ and $\mathbf{A}_{i,j}$ is provided by $\mathbf{T} = \mathbf{A}\mathbf{W}^{-1}$ (essentially a mapping from a target corner to the actual corner; element/corner subscripts have been dropped).

A key to the success of the DO method for ALEGRA applications is appropriate choice of \mathbf{T} for the problem at hand. Through ALEGRA, two classes of target matrices are available. The first method defines the element corner \mathbf{T} to be the $N_{dim} \times N_{dim}$ identity matrix. The effect of this target matrix is to force the mesh to tend towards elements with orthogonal corners. The second method employs the initial mesh to calculate the \mathbf{T} . Specifically, this method evaluates the Jacobian matrices at the element corners for the original mesh, and uses those matrices as the targets throughout the simulation.

When using the initial mesh method, the initial mesh minimizes the objective function because it was used to calculate the target matrices. Therefore, the optimization will not modify the initial configuration until mesh nodes have been moved by the physics. As the simulation progresses, the original mesh may no longer be attainable due to changes in internal or external domain boundaries (e.g. internal nodes may be marked as Lagrangian to define an interface between dissimilar materials). In this case, the optimization will try to make the elements close to the original mesh in the sense of the element quality metric (Eq. (78)). In contrast, when using the ideal weight \mathbf{T} , the scheme will modify the initial mesh prior to the on-set of physical motion (e.g. material velocities are still zero).

To this point we have not provided the functional form of ζ . Although a range of metrics are available in MESQUITE the the Inverse Mean Ratio metric,

$$\zeta = \frac{1}{N_{dim}} \frac{\|\mathbf{T}\|^2}{\det(\mathbf{T})^{N_{dim}}} \quad (79)$$

provides rotational and size invariance as well as a barrier to prevent element tangling (cf.³⁹). This quality metric is minimized with a value of 1 when there is no distortion of shape between the target corner and the actual corner.

To find the solution to the minimization problem, Jacobi iteration is used. This algorithm proceeds by looping over each node in the mesh, calculating an improved location with other nodes assumed fixed. Once improved locations are calculated for all nodes associated with ALE elements, the positions are updated, and the algorithm begins another iteration or terminates depending on whether the termination criteria have been met. To find the improved location locally for each node, a Feasible Newton solve is used.⁴³

While the Jacobi iteration is not as efficient as some optimization algorithms available in MESQUITE it was chosen because it met several important ALEGRA requirements. Specifically, ALEGRA requires the algorithm to,

1. operate in parallel,
2. be invariant to the number of processors, and
3. maintain symmetries that exist in the mesh, when possible.

While the Jacobi-style iteration is inferior to the Gauss-Seidel-style iteration in several ways (specifically convergence rate and resistance to tangling) unless converged, Gauss-Seidel iterations typically violate the second and third requirements.

Finally, Fig. 3 illustrates application of ALEGRA's Tipton implementation (without weights), MESQUITE with ideal mesh \mathbf{T} and initial mesh \mathbf{T} smoothers (Figs. (b)-(d) respectively). Fig. 3 (a) shows the initial mesh/domain where an inner cylinder, which moves upward with a given velocity, is enclosed by a fixed outer cylinder. The domain between the two cylinders is filled with a air. Clearly, the MESQUITE initial mesh option preserves mesh features (c.f. the kink on bottom centerline of the domain in (a)) while the ALEGRA's Tipton implementation modifies mesh grading near the inner cylinder while equilibrating element volumes.

II.I. The Remap Step

The remap step is implemented in an operator split fashion dimension-by-dimension so that effective one-dimensional differencing methods can be utilized. The remap is implemented in both volume and mass based coordinates defined below. The volume based coordinate is used for mass, but all other conserved variables are remapped in mass coordinates making use of the volume remap result. Staggered mesh variables (such as velocity) are remapped using Benson's half-index shift (HIS) algorithm⁴⁴

II.I.1. Accurate Differencing on Uneven Meshes

A rigorous approach to deriving the difference formula on uneven meshes uses a procedure known as the primitive function. The primitive function produces a polynomial interpolant that naturally conserves the quantity being remapped, thus achieving high quality in the sense of a simple, but essential constraint for advection methods. The primitive function is the conservative integral across a local span of mesh cells that is simple and straightforward due to the basic nature of control volume (conservation form) differencing.

One determines the primitive function by first noting the desired domain of dependence for the desired stencil, for example three cells, $j - 1$, j and $j + 1$. The function begins at the leftmost boundary with an arbitrary values (zero), and simply forms values at each cells right interface as

$$\Psi(x_k + 1/2) = \sum_{k=j-1}^{k=j+1} \Delta\xi_k \psi_k. \quad (80)$$

Here ξ is either the physical coordinate, the volume coordinate or the mass coordinate (density times volume).

One then computes a Lagrange interpolant of this function to use for deriving the approximations used for advection. The desired approximations can be made by taking the derivative of the interpolant producing the original function with the first derivative and the slopes with the second derivative.

Below we will describe the two basic methods that use the finite volume formalism and polynomial reconstruction. Each method can produce high-resolution results and we outline how to improve the accuracy. Other accuracy improvements can be made through relaxing the monotonicity constraints used to produce results without oscillations.

II.I.2. Useful limiter functions

We use a median function to provide appropriately bounded values satisfying the conditions for monotonicity. The median function can be defined using a minmod function, where

$$\text{minmod}(a, b) = 1/2 (\text{sign}(a) + \text{sign}(b)) \min(|a|, |b|),$$

and

$$\text{median}(a, b, c) = a + \text{minmod}(b - a, c - a).$$

An important property is that the median function's result is independent of the order of the arguments, $\text{median}(a, b, c) = \text{median}(c, a, b) = \text{median}(b, c, a)$. The median function also produces order-preserving choices through two of the arguments being at least a certain order, say m th, than the result of the application of the median function will be m th order accurate.

II.I.3. Piece-wise Linear Godunov Method (PLM)

Begin Algorithm II.I.3: Piece-wise Linear Reconstruction

1. The polynomial is piece-wise linear, $\psi(\xi)_j = \Pi_j(\theta)$; $\Pi(\theta) = \Pi_0 + \Pi_1\theta$; $\Pi_0 = \psi_j$; $\Pi_1 = s_j$.
2. Choose the initial value for the slope. The slope can be found by determining the Lagrange interpolant on the three points at taking the first derivative and evaluating at ξ_j ,

$$s_j = \frac{-\Delta\xi_{j+1}^2 \psi_{j-1} + (\Delta\xi_{j+1}^2 - \Delta\xi_{j-1}^2) \psi_j + \Delta\xi_{j-1}^2 \psi_{j+1}}{\Delta\xi_{j-1} \Delta\xi_{j+1} (\Delta\xi_{j-1} - \Delta\xi_{j+1})} \quad (81)$$

3. Choose a method to insure nonlinear stability: a monotonicity **Algorithm II.I.3**.
4. The piece-wise linear polynomial provides a time-centered value of $\psi_{j\pm 1/2, \mp}^{n+1/2} = \Pi_0 + \Pi_1(\pm 1/2 - 1/2\nu)$ where $\nu = \Delta t u / \Delta x$ is the Courant number for the remap. The Courant number can also be defined in terms of the volume or mass coordinate $\nu = \Delta t u A / \mathcal{V}$ or $\nu = \Delta t \rho u A / m$ respectively.

End Algorithm II.I.3: Piece-wise Linear Reconstruction

Begin Algorithm II.I.3: Piece-wise Linear Monotonicity

1. Define the base differences at first-order,

$$s_- = \psi_j - \psi_{j-1}, s_+ = \psi_{j+1} - \psi_j.$$

2. Monotonicity is defined by

$$Q^* = 2 \text{ median } (0, s_-, s_+)$$

and

$$s_j := \text{median } (0, s_j \Delta \xi_j, Q^*) / \Delta \xi_j.$$

End Algorithm II.I.3: Piece-wise Linear Monotonicity

II.I.4. Piece-wise Parabolic Godunov Method (PPM)⁴⁵

The same ideas can be used to construct a third-order piece-wise parabolic method. The parabola is determined by three values: ψ_j , the integral average of the variable ψ in cell j , and the edge values, $\psi_{j\pm 1/2}$, which are approximated by a linear combination of neighboring zone data. Like the PLM method, the parabolic reconstruction can be described in a brief sequence of steps in which one can vary the order of accuracy for the edge values and the method used for nonlinear stability.

Begin Algorithm II.I.4: Piece-wise Parabolic Reconstruction

1. The polynomial is piece-wise linear, $\psi(\xi)_j = \Pi_j(\theta)$; $(\theta) = \Pi_0 + \Pi_1\theta + \Pi_2\theta^2$; $\Pi_0 = 3/2\psi_j - 1/4(\psi_{j-1/2} + \psi_{j+1/2})$; $\Pi_1 = \psi_{j+1/2} - \psi_{j-1/2}$. $\Pi_2 = 3(\psi_{j+1/2} - 2\psi_j + \psi_{j-1/2})$
2. Choose the initial value for the edge values, $\psi_{j\pm 1/2}$. The third-order values for the edge variables are defined as being upwind to each edge with respect to the cell-center.

In order to define the edge values we will use a different nomenclature. This is to denote the use of conservation form to define the methods and the width of each mesh cell. The mesh cell is defined by its width $\Delta \xi_j$ for the j th zone (element). The third-order edge is defined simply as

$$\psi_{j+1/2} = a_{j-1}\psi_{j-1} + a_j\psi_j + a_{j+1}\psi_{j+1}, \quad (82)$$

where

$$a_{j-1} = -\frac{\Delta \xi_j \Delta \xi_{j+1}}{(\Delta \xi_{j-1} + \Delta \xi_j)(\Delta \xi_{j-1} + \Delta \xi_j + \Delta \xi_{j+1})},$$

$$a_j = \frac{\Delta \xi_{j+1} [3\Delta \xi_j + \Delta \xi_{j-1}(\Delta \xi_{j-1} + \Delta \xi_{j+1}) + \Delta \xi_j(3\Delta \xi_{j-1} + 2\Delta \xi_{j+1})]}{(\Delta \xi_{j-1} + \Delta \xi_j)(\Delta \xi_j + \Delta \xi_{j+1})(\Delta \xi_{j-1} + \Delta \xi_j + \Delta \xi_{j+1})},$$

$$a_{j+1} = \frac{\Delta \xi_j (\Delta \xi_j + \Delta \xi_{j-1})}{(\Delta \xi_j + \Delta \xi_{j+1})(\Delta \xi_{j-1} + \Delta \xi_j + \Delta \xi_{j+1})}.$$

3. Choose a method to insure nonlinear stability: a monotonicity **Algorithm II.J**.
4. The piece-wise linear polynomial provides a time-centered value of $\psi_{j\pm 1/2, \mp}^{n+1/2} = \Pi_0 + \Pi_1(\pm 1/2 - 1/2\nu) + \Pi_2(\frac{1}{4} \mp 1/2\nu + \frac{1}{3}\nu^2)$ where $\nu = \Delta t u \Delta x$ is the Courant number for the characteristic being reconstructed.

End Algorithm II.I.4 Piece-wise Parabolic Reconstruction

II.J. PPM Limiter⁴⁶

The algorithm used to define the monotonicity of a piece-wise parabolic reconstruction can be written compactly. This algorithm is algebraically equivalent to the original PPM monotonicity conditions.⁴⁵ For PPM, monotonicity is assured if

$$\psi_{j\pm 1/2} \in [\psi_j, \psi_{j\pm 1}]$$

and

$$\psi_{j\pm 1/2} \in [\psi_j, 3\psi_j - 2\psi_{j\mp 1/2}].$$

Begin Algorithm II.J: Piece-wise Parabolic Monotonicity

1. First, compute the following relations to bound edge values by adjacent data,

$$\psi_{j\pm 1/2} := \text{median}(\psi_j, \psi_{j\pm 1/2}, \psi_{j\pm 1}).$$

2. Finish the algorithm through bounding the edge values accounting for transport effects,

$$\psi_{j\pm 1/2} := \text{median}(\psi_j, \psi_{j\pm 1/2}, 3\psi_j - 2\psi_{j\mp 1/2}).$$

End Algorithm II.J: Piece-wise Parabolic Monotonicity

This algorithm has the cost of four applications of the median function per zone.

II.K. Remapping Kinetic Energy

One considerable difficulty with ALE methods is the treatment of kinetic energy by the method. With the Lagrangian method, the invariance of the Lagrangian mass tends to allow a non-conservative method to produce very good results despite having no theoretical expectation to do so. Once the masses are allowed to flow through a remap process solutions often begin to drift away from the good analytical results achieved with the purely Lagrangian solver. One telltale sign of this drift is the failure to conserve energy which can be explained through the problems with remapping kinetic energy, moreover the difference between the linear momentum (or velocities) and their quadratic derived quantity, kinetic energy. In other words, remapping the linear momentum and kinetic energy leads to a mismatch in the values on a grid. Of course the problem exists because the code is using internal energy rather than total energy for a variable.

Fortunately, there is a way to fix this problem known as the DeBar fix.⁴⁷ A more clear explanation of this algorithm is given by Anderson and Pember.⁴⁸ The method works as follows:

1. Compute the remap of density, nodal velocities and internal energy, e^{n+1} .
2. Also compute a remap of cell-centered definition of kinetic energy,

$$K_{i,j,k} = \frac{1}{\text{nodes of } i, j, k} \sum_{\text{nodes of } i, j, k} 1/2 (u^2 + v^2 + w^2). \quad (83)$$

3. Compute a correction of the internal energy to account for the difference between the cell-centered and nodal kinetic energies,

$$\Delta e_{i,j,k}^{\text{remapped}} = \left(K_{i,j,k}^{\text{remapped}} - \frac{1}{\text{nodes of } i, j, k} \sum_{\text{nodes of } i, j, k} 1/2 (u^2 + v^2 + w^2)^{\text{remapped}} \right). \quad (84)$$

4. If the quantity $q_j/p_j < 0.0001$ set $\Delta e_{i,j,k}^{\text{remapped}} = 0$.
5. If the magnitude of $\Delta e_{i,j,k}^{\text{remapped}} < -\beta e_{i,j,k}^{\text{remapped}}$, $\Delta e_{i,j,k}^{\text{remapped}} = -\beta e_{i,j,k}^{\text{remapped}}$.
6. Modify the remapped energy as

$$e_{i,j,k}^{\text{remapped}} := e_{i,j,k}^{\text{remapped}} + \Delta e_{i,j,k}^{\text{remapped}}. \quad (85)$$

It is important to emphasize the fact that when $C < 1$, the overall method will cease conserving total energy (internal plus kinetic energy).

III. Multimaterial Methods

A key aspect of the computational method for many application is the treatment of the dynamics of multiple materials. In particular there are two key components to the treatment of multiple materials are the numerical treatment of the transport of material interfaces, and their dynamics in the context of general hydrodynamics. Below we describe these algorithms first the interface reconstruction method for transporting material interfaces without inducing numerical diffusion. The second aspect is the dynamics of multiple materials described through thermodynamics described in the following subsection.

In defining the evolution of interfaces volume of fluid methods⁴⁹ are popular due to their conservation properties and high resolution approximation of the interface. When the interface is not sufficiently resolved, the method produces results that breakdown in a particular fashion. The interface becomes disjoint and tends to create isolated blobs that appear to be associated with an intrinsic surface tension. This occurs whenever the interface resolution drops below three zones in width. This character arises directly from the conservative nature of the approximation. If the interface should not break apart, but remain continuous this nature of the interface topology is limiting. Fortunately, there are alternative methods available to transport interfaces without inducing too much numerical diffusion at the interface.

III.A. Interface Reconstruction



Figure 4. A comparison of the reconstruction of a planar interface before (left) and after (right) the smoothing step of the algorithm.

In multimaterial simulation codes the majority of elements contain a single material. Fluxing material from these clean elements is solely concerned with accurately determining the distribution of fluxed quantities in each donor element. For the elements that contain two or more materials, the fluxing must determine how much of each material is fluxed in addition to their fluxed quantities. ALEGRA represents the interface between the distinct materials in mixed elements with a second-order planar interface that is computed by the Patterned Interface Reconstruction (PIR) algorithm.⁵⁰ The algorithm is derived from the three-dimensional Youngs reconstruction algorithm.⁵¹ PIR is adapted to the unstructured meshes in ALEGRA and can function on elements bounded by an arbitrary number of planar facets.

PIR starts with a Youngs like reconstruction step. The interface normal is approximated by the local gradient of the material volume fraction. The interface is then positioned within each mixed element to conservatively reproduce the material volumes. For a planar interface there are many orientations that are exactly reproduced by the Youngs algorithm; however, not all are reproduced. Curved interfaces also are approximated to visual satisfaction but not to second-order accuracy. Interface reconstruction accuracy is defined as the rate of convergence of the volume between the reconstructed interface and the exact input interface as the element size is reduced. To improve the accuracy of the reconstruction, a position for the interface is determined and the local shape of the interface is determined. The position that is chosen is called the stability point and is the areal centroid of the interface. The development of the stability point concept is discussed in.⁵² For each mixed element a series of patterns are fitted to the neighboring stability



Figure 5. A comparison of the reconstruction of a mixed element before (left) and after (right) the smoothing step of the algorithm.

points. The patterns currently consist of a planar and a curved shape such as a sphere. The volume between the neighboring interface and the extrapolated pattern interface determines which pattern produces the more accurate approximation. The normal of the more accurate pattern is then used to update the normal for the mixed element. This fitting is performed over all the mixed elements for a material in the mesh and the process is repeated to convergence.

To demonstrate the visual accuracy of the PIR, a planar interface is shown before and after the smoothing step in Figure 4. In Figure 5, a set of planar, T shaped interfaces is presented before and after smoothing. In Figure 6, the before and after interfaces for a Cassini oval is shown.

III.B. Multimaterial treatment

III.B.1. Equal Volume Treatment

First, consider the effective differential equations for multiple materials with the material index k in the Lagrangian frame if an equal volume partitioning is made,

(volume fraction)

$$\frac{df_k}{dt} = 0, \quad (86)$$

(mass)

$$\frac{df_k \rho_k}{dt} = -f_k \rho_k \frac{\partial u}{\partial x}, \quad (87)$$

(momentum)

$$\rho \frac{du}{dt} = -\frac{\partial \bar{p}}{\partial x}, \quad (88)$$

(energy)

$$f_k \rho_k \frac{de_k}{dt} = -f_k \bar{p} \frac{\partial u}{\partial x}. \quad (89)$$

Here \bar{p} is the mean pressure typically computed as $\bar{p} = \sum_k f_k p_k$ with $p_k = P(\rho_k, e_k)$.

There are a set of consistency conditions associated with these equations that are essential to satisfy. One key relation is the volume filling requirement, $\sum_k f_k = 1$, which is trivially satisfied by Equation 86. The

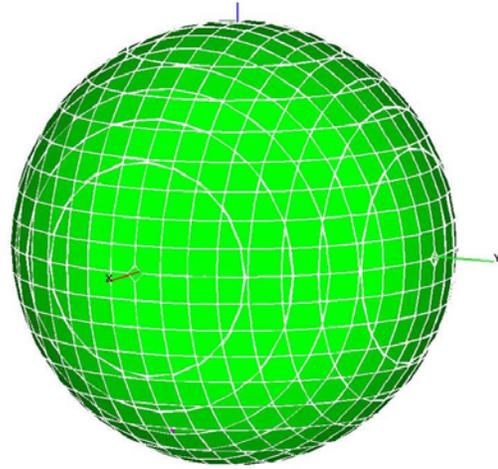


Figure 6. The reconstruction of a Cassini oval using the algorithm described here.

work in the energy equation should equal the work done on the total energy of the system, again trivially satisfied by Equation 89.

As noted in many publications, this approximation can yield unphysically realistic results when the materials have greatly different responses to compressible motion.

III.B.2. Bulk Modulus Weighted

When the volume changes are partitioned on the basis of the properties of the materials,
(volume fraction)

$$\frac{df_k}{dt} = f_k \left(\frac{\bar{B}}{B_k} - 1 \right) \frac{\partial u}{\partial x}, \quad (90)$$

(energy)

$$f_k \rho_k \frac{de_k}{dt} = -f_k \frac{\bar{B}}{B_k} \bar{p} \frac{\partial u}{\partial x}. \quad (91)$$

Equation 90 can be derived from the mass equations,

$$\frac{df_k \rho_k}{dt} = -\rho_k f_k \frac{\partial u}{\partial x},$$

with the following steps. Expand this equation using the chain rule to,

$$\rho_k \frac{df_k}{dt} + f_k \frac{d\rho_k}{dt} = -\rho_k f_k \frac{\partial u}{\partial x}.$$

Combine and simplify terms to yield

$$\frac{df_k}{dt} = -\frac{f_k}{\rho_k} \frac{d\rho_k}{dt} - f_k \frac{\partial u}{\partial x}.$$

Replace the time-derivative of density with pressure (the mean pressure!),

$$\frac{df_k}{dt} = -\frac{f_k}{\rho_k} \frac{\partial \rho_k}{\partial \bar{p}} \frac{d\bar{p}}{dt} - f_k \frac{\partial u}{\partial x}.$$

Realizing that the pressure equation on an adiabat evolves with the following equation,

$$\frac{d\bar{p}}{dt} = -\rho \bar{c}^2 \frac{\partial u}{\partial x},$$

and $\frac{\partial \bar{p}}{\partial \rho_k} = c_k^2$. Substitute and rearranging this result provides the last step,

$$\frac{df_k}{dt} = f_k \frac{\rho c^2}{\rho_k c_k^2} \frac{\partial u}{\partial x} - f_k \frac{\partial u}{\partial x},$$

and Equation 90 can be recovered by the definition $B = \rho c^2$ and $B_k = \rho_k c_k^2$.

The compatibility conditions can be met by choosing the definitions for \bar{B} and \bar{p} . First, define \bar{B} as

$$\bar{B} = \left(\sum_k \frac{f_k}{B_k} \right)^{-1}, \quad (92)$$

then the mean pressure can be combined,

$$\bar{p} = \bar{B} \left(\sum_k \frac{f_k \rho_k}{B_k} \right). \quad (93)$$

These relationships can be derived in several ways.

These basic relations can be derived from fundamental thermodynamic relations. If the right hand side of Equation 90 sums over the material to zero as required for volume conservation,

$$\sum_k f_k \left(\frac{\bar{B}}{B_k} - 1 \right) \frac{\partial u}{\partial x} = 0, \quad (94)$$

this can be rearranged to give

$$\sum_k f_k \left(\frac{\bar{B} - B_k}{B_k} \right) = 0$$

and expanding gives the necessary relation for \bar{B} ,

$$\sum_k \frac{f_k \bar{B} - f_k B_k}{B_k} = \bar{B} \sum_k \frac{f_k}{B_k} - \sum_k f_k = 0.$$

The equality is satisfied by $\bar{B} = (\sum_k f_k/B_k)^{-1}$. The derivation of \bar{p} is associated with the pressure relaxation step discussed below in Section III.B.3. One can also derive this relation from the thermodynamic definition of the bulk modulus (in this presentation V is the material specific volume, $1/\rho$),

$$B = \gamma p = -V \left. \frac{\partial p}{\partial V} \right|_S \approx -V \frac{\Delta p}{\Delta V}, \quad (95)$$

and the following discrete representation,

$$B_k = \frac{V_k (p_k - \bar{p})}{\Delta V_k},$$

using the identity $\Delta V_k/V_k = \Delta f_k/f_k$ and summing over the available materials,

$$\sum_k \Delta f_k = \sum_k \frac{f_k p_k - f_k \bar{p}}{B_k} = 0$$

and finding the final relation

$$\bar{p} \sum_k \frac{f_k}{B_k} = \sum_k \frac{f_k p_k}{B_k},$$

through realizing that $\bar{B} = (\sum_k f_k/B_k)^{-1}$.

This entire approach was pioneered by James LeBlanc, and has been more recently applied by Bob Tipton⁵³ as well as documented in the open literature by Miller and Puckett.⁵⁴

III.B.3. Pressure relaxation^{54, 55}

If the desired state for the fluid is to be consistent with a fluid in pressure equilibrium, the state of the fluid should be modified. Pressure differences in the fluid should be removed by changing the volume fractions. This evolution can be written as a differential equation,

$$\frac{df_k}{dt} = \frac{f_k}{\tau_p} \left(\frac{p_k - \bar{p}}{B_k} \right), \quad (96)$$

where τ_p is a relaxation time scale. The pressure relaxes due to the action of acoustic waves, so one might surmise that $\tau_p = \ell/c$ where ℓ is a characteristic length scale which may be on the order of the mesh scale Δx . We will return to this model at the end of this section. These changes should also be applied to the energy of the fluids by partitioning the work associated with the volume changes. The details of the algorithm are slightly different depending on whether B or γ is used to weight the changes. Start with a set of volume fractions, f_k^0 where the pressures are out of equilibrium. For the bulk modulus the changes can be defined as follows:

$$\Delta f_k = f_k \left(\frac{p_k - \bar{p}}{B_k} \right), \quad (97)$$

$$f_k^1 = f_k^0 + \Delta f_k; \rho_k^1 = \frac{f_k^0 \rho_k^0}{f_k^1} \quad (98)$$

$$f_k^1 \rho_k^1 e_k^1 = f_k^0 \rho_k^0 e_k^0 - \bar{p} \Delta f_k. \quad (99)$$

When using the adiabatic coefficient to weight the changes the equation for Δf_k changes to,

$$\Delta f_k = f_k \left(\frac{p_k - \bar{p}}{\gamma_k \bar{p}} \right). \quad (100)$$

This procedure is designed to quickly drive the materials into pressure equilibrium. This can often be an unphysically fast process to occur in a single time step. In addition, the changes in the equation of state may be so large and nonlinear that the linearization used to derive these relations is invalid. Taking the relaxation equation Equation 96 we can provide a reasonable way to limit the size of the volume changes. If we simply assert that $\tau_p = \Delta x/c$ and then solve the differential equation with the right hand side held constant as $Df_k/Dt = \lambda f_k$ we find that the change in volume fraction can be expressed simply as

$$\Delta f_k = \exp \left[\frac{c \Delta t}{\Delta x} \left(\frac{p_k - \bar{p}}{B_k} \right) \right], \quad (101)$$

and similarly for the case of the γ based method. Unfortunately this method is not normalized and does not lead to a consistent results, $\sum_k \Delta f_k = 0$. The issue revolves around how the averaging for \bar{p} is done, which must be constrained to assure that consistency conditions are met.

The same issue is present with an implicit integration of the equations where the explicit averaging defined earlier will not insure consistency. This can only be done through an iterative procedure and the sort of renormalization discussed in the next section.

III.B.4. Limiters and Renormalization

While we have tried to present these methods in a rigorous fashion for a robust implementation some more ad hoc techniques are usually necessary to avoid changes that are unphysically large, introduce negative volume fractions, or leave the sum of volume fractions not equal to one.

This renormalization should probably always be completed to keep the volume fraction sum from deviating from one simply due to roundoff error. This step is quite simple

$$f_k := \frac{f_k}{\sum_k f_k}. \quad (102)$$

This renormalization should also be applied to modify the density and energy to be consistent with the new volume fractions (just as the pressure relaxation algorithm). Actually Equation 102 should be computed in two steps to facilitate the other updates carried out in the following order,

$$\Delta f_k = f_k \left(\left(\sum_k f_k \right)^{-1} - 1 \right),$$

$$\rho_k := \frac{f_k \rho_k}{f_k + \Delta f_k}$$

or by conveniently defining the mass of the k th material as $m_k = f_k \rho_k$,

$$\rho_k = \frac{m_k}{f_k + \Delta f_k}$$

$$e_k := e_k - \frac{\bar{p} \Delta f_k}{m_k},$$

and

$$f_k := f_k + \Delta f_k$$

One may also want to limit changes so that volume fractions remain bounded between zero and one,

$$f_k := \max [0, \min (f_k, 1)]. \quad (103)$$

III.B.5. Void Treatment

we will try to suggest a structured manner to handle void. This will assume to some degree that this process is decoupled from the process that introduces void (presumably either in initial conditions or when a material is expanded below its zero pressure density).

If void is present, either $\bar{\gamma} = 0$ or $\bar{B} = 0$, and $\bar{p} = 0$. The void itself will have these same properties, $\gamma_{\text{void}} = 0$ or $B_{\text{void}} = 0$ and $p_{\text{void}} = 0$. Inserting these arrangements gives a clear view of what will happen, in either case,

$$\frac{df_{\text{void}}}{dt} = \text{indeterminate}, \quad (104)$$

and for all $k \neq \text{void}$

$$\frac{df_k}{dt} = -f_k \frac{\partial u}{\partial x}. \quad (105)$$

Thus, if the flow is compressive, the volume fraction of the non-void materials will rise, and if the flow is expansive the volume fraction of the non-void materials will drop. This is precisely what should happen physically. For expansion the “missing” volume fraction should be added to the void and is not included in the equation, so by the use of the consistency relations, Equation 104 becomes

$$\frac{df_{\text{void}}}{dt} = \frac{\partial u}{\partial x} \sum_{k \neq \text{void}} f_k. \quad (106)$$

For expansions this provides no difficulties, but for compression it is possible for the void to completely disappear. we will describe how to treat this case next.

We can compute the state of the cell when the void is completely compressed out as follows,

$$f_k^* = \frac{f_k}{\sum_{k \neq \text{void}} f_k}. \quad (107)$$

The data needed at this point is how much additional compression is necessary to be applied to the cell, again this can be easily defined, if the total compression of the cell is $\delta f = -\frac{\partial u}{\partial x} \Delta t$, then the remaining compression is simply $\delta f^* = \delta f - f_{\text{void}}$. This compression should be partitioned based on the properties of the materials in the cell and their volume fractions without void, f_k^* as

$$\delta f_k^* = \frac{\bar{B}^* - B_k}{B_k} \delta f^*, \quad (108)$$

where $\bar{B}^* = (\sum_k f_k^*/B_k)^{-1}$. At this point pressure relaxation, renormalization and limiting may be applied.

IV. Multiphysics: Magnetohydrodynamics (MHD)

Magnetohydrodynamic in ALEGRA-MHD implies a model which combines hydrodynamics and Maxwell’s equations in moving media with displacement currents neglected. Most cases of interest to users do not fall into the ideal magnetohydrodynamic regime. Generally, very significant magnetic diffusion across a wide range of resistivities is expected. Therefore, depending on the time scales and material properties of interest the flow could range from a near ideal MHD regime to a highly diffusive regime. Non-conducting regions are treated as being highly resistive. MHD environments may require a wide range of physics phenomena to be modeled in an environment which includes magneto-solid dynamic motion transitioning into melted and vaporized material interacting with magnetic fields. Thermal transport modeling may be required and is one means for smoothing the localized energy sources which can develop due to Joule heating. A simple radiation emission model may also be used for purposes of removing excess energy. Applications which require true radiation transport coupled to magnetics utilize the ALEGRA-HEDP code. Magnetic material modeling is not supported at this time.

The 2D version of ALEGRA-MHD supports both Cartesian (XY) and cylindrical (RZ) geometries. 2D (XY or RZ) MHD modeling^{56,57} the magnetic field $\mathbf{B} = (B_x, B_y)$ may be in the plane of the mesh and the current density J_z is orthogonal to the plane, or else the magnetic field B_z may be orthogonal to the plane of the mesh and the current density $\mathbf{J} = (J_x, J_y)$ is in the plane. The 2D code supports unstructured quadrilateral

grids. When the magnetic field is in the plane a nodal magnetic vector potential formulation is implemented and when the field is out of the plane a nodal magnetic flux density representation is implemented. This historical choice imposes different requirements on the overall algorithms as well as different essential and natural boundary conditions depending on the representation. Coupling for both in-plane and out-of-plane fields at the same time is not supported.

The 3D version of ALEGRA-MHD supports 3D Cartesian (XYZ) MHD modeling on unstructured grids. Structurally the algorithm implements electric fields on edges and magnetic flux density on faces and thus is algorithmically similar to the 2D in-plane field and out-of-plane vector potential formulation. The 3D code implements a magnetic diffusion solution based on edge and face elements.⁵⁸ A sophisticated $H(curl)$ magnetic field solver is required for robust parallel solution of the associated discrete matrix. A discrete, divergence free property is maintained both in the magnetics solve and during a constrained transport algorithm in the remap phase. The magnetic flux density is represented in terms of face elements with the element magnetic flux on faces as degrees of freedom. The 3D code supports unstructured hexahedral grids.

Magnetohydrodynamic coupling can be thought of as occurring in two primary direct processes: First, exchange of energy between the magnetic field and kinetic energy of the material through magnetic forces and second deposition of thermal energy in the material due to Joule heating associated with magnetic diffusion. The two processes will be described in some detail in the next two sections. Of course, changes to material internal energy translate into thermal pressure which then translates into kinetic energy and thus works against the magnetic field to cause a change in the magnetic energy but we shall consider these as secondary exchange pathways since they do not enter directly into the operator split methodology for the MHD Lagrangian step.

IV.A. Magnetic forces and the Ideal MHD step

In the context of ALEGRA the best way to compute the Lorentz force, $\mathbf{J} \times \mathbf{B}$, is to compute a Maxwell stress tensor, \mathbf{T}^M , such that

$$\mathbf{T}^M = \frac{1}{\mu_0} \left(\mathbf{B} \otimes \mathbf{B} - \frac{1}{2} \mathbf{B}^2 \mathbf{I} \right), \quad (109)$$

or in component form,

$$\mathbf{T}_{ij}^M = \frac{1}{\mu_0} \left(B_i B_j - \frac{1}{2} \delta_{ij} B_k B_k \right). \quad (110)$$

One can show that

$$\mathbf{J} \times \mathbf{B} = \nabla \cdot \mathbf{T}^M - \mathbf{B}(\nabla \cdot \mathbf{B})/\mu_0 = \nabla \cdot \mathbf{T}^M \quad (111)$$

since $\nabla \cdot \mathbf{B} = 0$. In ALEGRA the magnetic stress tensor is approximated by evaluating \mathbf{T}^M at element centers. This stress can then be fed to the mean stress finite element divergence operator discussed in the hydrodynamic section to assemble the nodal forces. There is no intrinsic exact numerical correspondence with the methodology between the magnetic forces and the magnetic energy. Recent efforts have investigated the consequences of computing the magnetic forces discretization directly from variation of the discrete magnetic energy metric with respect to nodal coordinates.⁵⁹ This is more expensive but in some circumstances may give a better energy conservation property and it is unclear at present what circumstances this more exact alternative discrete force description may be preferable.

IV.B. Resistive Diffusion, Transient Magnetics or Eddy Current Modeling

Energy is transferred from the magnetic field to internal energy through resistive magnetic diffusion also labels transient magnetics or eddy current approximation to Maxwell's equations. In the operator split approximation we have discussed we solve the transient magnetics equations coupled to a external magnetic energy sources through boundary conditions on a fixed grid. We describe here only the basic 3D methodology. ALEGRA utilizes a compatible finite element technology to solve for transient magnetic or eddy current diffusion equations in 3D.^{60,61} Briefly, The transient equations are follows: First we have the reduced form of Ampere's Law and Faraday's Law.

$$\nabla \times \mathbf{H} = \mathbf{J} \quad (112)$$

$$\frac{\partial \mathbf{B}}{\partial t} + \nabla \times \mathbf{E} = 0 \quad (113)$$

and the closure relations

$$\mathbf{H} = \nu_0 \mathbf{B} \quad (114)$$

$$\mathbf{J} = \sigma \mathbf{E} \quad (115)$$

where ν_0 is the constant reluctivity and σ is the conductivity which relate the relating the magnetic field to the flux density and the current density to the electric field respectively. We utilize a deRham complex finite element sequence utilized low order edge and face elements. Ampere's Law is imposed in weak form and Faraday's law is imposed using a backward Euler time integrator and the exact finite element sequence in space. Thus

$$\begin{aligned} \int_{\Omega} \sigma \mathbf{E}^{n+1} \cdot \hat{\mathbf{E}} - \nu_0 \mathbf{B}^{n+1} \cdot \nabla \times \hat{\mathbf{E}} d\Omega &= \int_{\Gamma} \mathbf{n} \times \mathbf{H}^{n+1} \cdot \hat{\mathbf{E}} d\Gamma, \\ \mathbf{B}^{n+1} &= \mathbf{B}^n - \Delta t \nabla \times \mathbf{E}^{n+1} \end{aligned} \quad (116)$$

which by substitution leads to

$$\begin{aligned} \int_{\Omega} \sigma \mathbf{E}^{n+1} \cdot \hat{\mathbf{E}} + \Delta t \nu_0 \nabla \times \mathbf{E}^{n+1} \cdot \nabla \times \hat{\mathbf{E}} d\Omega &= \int_{\Omega} \nu_0 \mathbf{B}^n \cdot \nabla \times \hat{\mathbf{E}} d\Omega \\ &+ \int_{\Gamma} \mathbf{n} \times \mathbf{H}^{n+1} \cdot \hat{\mathbf{E}} d\Gamma. \end{aligned} \quad (117)$$

This is what is called an $H(curl)$ system in the multigrid literature. The reason it is important to utilize the edge and face element sequence is so that the stiffness matrix must contain a large exact null space corresponding to the continuum operator. This discrete representation is crucial to avoid spurious transients which can destroy the accuracy of the solution. In real problems the conductivity required can vary by many orders of magnitude from highly resistive to highly conductive. Most real application problems will have some sort of "void" regions where the conductivity must be very small and this is added to a stiffness matrix which is singular. The end result is that this combined matrix is a significant challenge to solve. After the solution of \mathbf{E}^{n+1} then we can compute the flux density from Equation 113. Due to the relationship between the degrees of freedom for the deRham sequence elements this operators amounts to adding up edge circulation values associated with the edge elements utilized the proper sign convention to compute the flux degrees of freedom associated with the face element representation of the flux density. The flux density is allows exactly divergence free by construction. After the solution of \mathbf{E}^{n+1} then we can compute the flux density from Equation 113. Due to the relationship between the degrees of freedom for the deRham sequence elements this operators amounts to adding up edge circulation values associated with the edge elements utilized the proper sign convention to compute the flux degrees of freedom associated with the face element representation of the flux density. The flux density is exactly divergence free by construction. In 2D nodal finite element discretizations have been developed to solve the magnetic diffusion matrices associated with each supported representation.

ALEGRA has historically utilized the Aztec iterative solver library⁶² to solve the discrete partial differential equation that governs its physics. Aztec is an iterative solver library with many options for solving linear systems of equations. Aztec includes a number of Krylov iterative methods such as conjugate gradient (CG), generalized minimum residual (GMRES), and stabilized biconjugate gradient (BiCGSATB) as well as associated algebraic multigrid (AMG) capabilities. Sophisticated algebraic multigrid options⁶³ for the edge element diffusion formulation in 3D and node-centered 2D formulations are available. In addition, even parallel direct solvers can be implemented using the EPETRA coding interface in the Trilinos software environment.

Critical to obtaining predictive answers in high energy MHD regimes is proper modeling of the high temperature state transitions and in particular how the conductivity varies with density and temperature. Significant progress has been made at Sandia with respect to conductivity modeling in the solid-liquid-vapor transitions regions. The Lee-More Desjarlais (LMD) models provide the crucial first principles material modeling input to be able to obtain predictive results.^{64, 65}

If we choose $\hat{\mathbf{E}} = \mathbf{E}^{n+1}$ in equation 117 we get an equation which gives a precise description of the energy

partitioning associated with the finite element weak form with the backward Euler time discretization

$$\begin{aligned} \int_{\Omega} \sigma \mathbf{E}^{n+1} \cdot \mathbf{E}^{n+1} + \nu_0 \left(\frac{(\mathbf{B}^{n+1})^2 - (\mathbf{B}^n)^2}{2\Delta t} \right) + \nu_0 \left(\frac{(\mathbf{B}^{n+1} - \mathbf{B}^n)^2}{2\Delta t} \right) d\Omega \\ = \int_{\Gamma} \left(\mathbf{n} \times \mathbf{H}_b^{n+1} \right) \cdot \mathbf{E}^{n+1} d\Gamma \end{aligned} \quad (118)$$

The first term is the Joule heating, the second term is the change in magnetic energy, the third term is a time discretization energy error which can be tallied and the boundary term is the Poynting flux across the boundary. The fundamental relations such as the above arising directly from the finite element weak form allow for the development of a circuit equation coupling methodology. In mixed cells we define $\sigma = \sum_m \sigma_m \phi_m$ and the material Joule heating in the first term in Equation 118 is partitioned accordingly. This gives the correct Joule heating in mixed cells when the electric field is aligned with material boundary. This approach has not always been sufficient however and an additional limiter on the heating in mixed cells has been necessary. A highly robust and accurate methodology for the mixed cell diffusion problem for the full suite of deRham complex elements is lacking.

In many practical cases there is significant feedback from the electrical system external to the ALEGRA mesh which typically only includes the primary “load” where most of the significant heating and material motion is occurring. Self-consistent transfer of energy to and from external lumped element circuit equations should be modeled and can be critical to a proper understanding of the physical mechanisms at work. Users may connect the transient magnetic or MHD simulations to an external circuit set defined by a set of differential-algebraic equations. This technology is critical for flyer plate modeling.⁶⁶ The methodology used in ALEGRA is based on generating an explicit lumped element circuit representation for the mesh response by computing particular and homogeneous solutions associated with boundary and conditions and then passing this representation to the circuit solver to compute the expected state at the end of the time step. This approach has the advantage of an explicit separation of the circuit solve from the finite element mesh solution and a fixed number of required linear solves. lumped element circuit equations can be critical to a proper understanding of the physical mechanisms at work. Users may connect the transient magnetic or MHD simulations to an external circuit set defined by a set of differential-algebraic equations. This technology is critical for flyer plate modeling.⁶⁶ The methodology used in ALEGRA is based on generating an explicit lumped element circuit representation for the mesh response by computing particular and homogeneous solutions associated with boundary and conditions and then passing this representation to the circuit solver to compute the expected state at the end of the time step. This approach has the advantage of an explicit separation of the circuit solve from the finite element mesh solution and a fixed number of required linear solves.

Currently, there are aspects of some of the operator split methodology in 2D that blur a sharp distinction between the ideal MHD step and the transient magnetics step resulting in velocity dependent terms in the transient magnetics solve. This is unnecessary and a clean separation in 2D is planned. The transient magnetic diffusion step currently occurs after the hydrodynamic motion. This has the advantage of ensuring that the magnetic field is smoothed after the Lagrangian motion and is the current location of this algorithm based on experience in the early days of development. The operator split order is fairly arbitrary and should be the subject of continued investigation.

IV.C. Remap

After the Lagrangian ideal MHD step and the magnetic diffusion step, the mesh may be smoothed or moved back to the original location and operators are required to remap the fundamental magnetic field quantities to the new mesh. For the 2D B_z representation which degrees of freedom on nodes, standard ALEGRA nodal remap operators are utilized. For 2D vector potential representations and in 3D this is not sufficient. What is required is a remap operator that is properly upwinded and limits on the flux density variable in some way rather than a vector potential variable. This is because we desire monotonicity preservation in the gradients of the flux density, i.e. the current. Furthermore we require $\nabla \cdot \mathbf{B}$ be enforced exactly so we must either store a vector potential on edges or store a flux density that has been defined and is always updated as a signed sum of edge centered circulations.

The basic idea of a remap methodology which satisfies all these constraints was first introduced by Evans and Hawley in the context of a structured meshes.⁶⁷ Constrained transport provides a mechanism for

advection of magnetic flux density which also preserves a discrete $\nabla \cdot \mathbf{B} = 0$ property by always computing an accurate update increment on edges which can then be used to update the face centered fluxes. The $\nabla \cdot \mathbf{B} = 0$ property is satisfied exactly by utilizing a staggered grid in which updates are always given as circulations on edges. Constrained transport must intrinsically be implemented as a spatially unsplit method. The flux increment through the face is then easily computed as an oriented sum of the edge circulations and a discrete divergence free property is thus exactly maintained. The easiest way to visualize the algorithm for an unstructured grid is to consider the set of edge and associated faces of any grid structured or unstructured. The new grid is considered as an extrusion of the old grid to the new location. (No topological modification are allowed.) This extrusion sweeps out small volumes which are bounded by the surfaces created by the edge extrusion and the old and new faces. Apply the divergence theorem to each of these small volumes, V . We obtain

$$\int_V \mathbf{B} \cdot \mathbf{n} \, dA = \int_{\delta V_{old}} \mathbf{B} \cdot \mathbf{n} \, dA + \sum_{e=1}^{N_e} \int_{\delta V_e} \mathbf{B} \cdot \mathbf{n} \, dA + \int_{\delta V_{new}} \mathbf{B} \cdot \mathbf{n} \, dA = 0 \quad (119)$$

The first integral is the known flux, the last integral is the desired new flux quantity and the sum is a set of integrals over the swept edges. The key is then to compute accurate values of these edge centered fluxes so that the new flux can be computed. The edge centered flux values are computed once for each edge. The end result after taking proper account of signed normals is that the new flux values must be discretely divergence free by construction. Note that the updates are properly upwinded by construction but update values are needed that do not introduce spurious oscillations and have minimal energy dissipation. For this purpose we need a high order representation that that given by the low order face element. The methodology utilized in ALEGRA is based on reconstruction of the low order face element representation to what is essentially a so-called "BDM" element which provides cross face flux variations in addition to the average flux. The cross face flux degrees of freedom are limited to avoid oscillations and the high order reconstructed element values are utilized to compute a numerical quadrature of the swept edge fluxes.⁶⁸ Note that an alternative methodology has been proposed by Rieben, et. al. in which the high order edge updates themselves are limited instead of the underlying magnetic field representation.⁶⁹

V. Multiphysics: Thermal Conduction

ALEGRA supports an implicit thermal conduction modeling in the operators split methodology. Thermal conduction may become important when sharp thermal gradients are developed for example due to Joule heating in rapid magnetic transients. Problems which run for longer time scales may also introduce important effects due to thermal conduction. The HEDP version of the code also supports separate ion and electron conduction. ALEGRA implements two conduction numerical methods which both utilize an essentially similar approach. In all cases the methodology solves for flux updates on faces and these fluxes are summed to compute the change in the internal energy in the cell. The first methodology is based on the support operator method⁷⁰ and the second method is a finite element approach using deRham complex sequence. The methods result in an $H(div)$ type of matrix which must be solved. The thermal conduction packages utilize the Aztec library to solve the resulting linear system. No $H(div)$ multigrid capability is available at present but in general this is not a major issue since these matrices are often mass matrix dominated. The thermal conduction packages utilize the Aztec library to solve the resulting linear sytem. No $H(div)$ multigrid capability is available at present but in general this is not a major issue since these matrices are often mass matrix dominated.

We describe the discretization approach in the language of mixed finite elements. In the operator split formulation we need to solve on a fixed mesh the equations

$$\rho c_v \frac{\partial \theta}{\partial t} + \nabla \cdot \mathbf{q} = 0 \quad (120)$$

$$\mathbf{q} = -\kappa \nabla \theta \quad (121)$$

where c_v is the heat capacity at constant volume. Since internal energy in ALEGRA is element centered it makes sense to consider energies and temperatures which lie at element centers and energy fluxes on faces using a mixed finite element approach. We represent the temperature and the energy flux using basis functions in L^2 and $H(div)$ from a low order deRham complex space.⁷¹ The first equation is satisfy exactly in the deRham complex using a backward Euler time discretization and and the second equation is satisfied

weakly. We have the exact discrete relationship

$$\rho c_v \frac{\theta^{n+1} - \theta^n}{\Delta t} + \nabla \cdot \mathbf{q}^{n+1} = 0 \quad (122)$$

and the weak form for Equation 121 is

$$\int_{\Omega} \hat{\mathbf{q}} \cdot \kappa^{-1} \mathbf{q}^{n+1} d\Omega - \int_{\Omega} \theta^{n+1} \nabla \cdot \hat{\mathbf{q}} d\Omega = - \int_{\Gamma} \theta_b^{n+1} \hat{\mathbf{q}} \cdot \mathbf{n} d\Gamma \quad (123)$$

which via substitution becomes

$$\int_{\Omega} \hat{\mathbf{q}} \cdot \kappa^{-1} \mathbf{q}^{n+1} + \frac{\Delta t}{\rho c_v} (\nabla \cdot \mathbf{q}^{n+1}) (\nabla \cdot \hat{\mathbf{q}}) d\Omega = - \int_{\Gamma} \theta_b^{n+1} \hat{\mathbf{q}} \cdot \mathbf{n} d\Gamma + \int_{\Omega} \theta^n \operatorname{div} \hat{\mathbf{q}} d\Omega \quad (124)$$

This is an $H(\operatorname{div})$ matrix equation for the flux \mathbf{q}^{n+1} . It is clear that in this formulation normal flux boundary conditions are essential conditions and temperature boundary conditions are natural conditions. Given \mathbf{q}^{n+1} the change in energy in the cell is computed from Equation 122. This update amounts to a simple signed addition of the flux degrees of freedom associated with the face element representation.

VI. Multiphysics: Radiation Transport and 3-T Physics

ALEGRA has two different approximations to the Boltzmann Equation 9 for the thermal radiation transport. The first, and most accurate, is a multigroup Implicit Monte Carlo (IMC)⁷² package shared with the KULL code.^{73–75} In this method, many pseudo-photons are simulated through physical events to build up a statistical average of the photon behavior. This can be very accurate, but is also very time consuming.

A faster, but less accurate approximation supported in ALEGRA The second approximation for the radiation transport is a flux limited diffusion package. The multigroup diffusion approximation to Equation 9 can be written as

$$\frac{1}{c} \frac{\partial e_r^g}{\partial t} - \nabla \cdot \mathbf{F}_r^g = \sigma_a^g \left(\frac{1}{c} \int_{\nu_g}^{\nu_{g+1}} \int_{4\pi} B_\nu(T_e) d\Omega d\nu - e_r^g \right), \quad (125)$$

where

$$e_r^g = \frac{1}{c} \int_{\nu_g}^{\nu_{g+1}} \int_{4\pi} I d\Omega d\nu \quad (126)$$

Equation 4 is modified to use e_r^g

$$\frac{\partial e_e}{\partial t} = - \sum_{g=1}^G \sigma_a^g \left(\frac{1}{c} \int_{\nu_g}^{\nu_{g+1}} \int_{4\pi} B_\nu(T_e) d\Omega d\nu - e_r^g \right). \quad (127)$$

In the flux limited diffusion approximation, the radiative flux is approximated as

$$\mathbf{F}_r^g \approx - \frac{\chi(e_r^g)}{3\sigma_a^g} \nabla e_r^g, \quad (128)$$

where $\chi(e_r^g)$ is a nonlinear function of the energy density chosen to maintain certain physical limits. The solution of these equations is similar to the method presented by Morel.[?]

There are numerous applications, both in industrial production and in applied research, which use ionized gases, or plasmas, to obtain a desired result. In particular, the Z-pinch program at Sandia works with high energy, high density plasmas encompassing a wide range of materials from aluminum to tungsten to study radiation source technology, weapons physics, and ICF-related issues. Under many conditions, the electron and ion populations of such plasmas may not be in equilibrium; in other words, the electrons and ions are not characterized by the same Maxwellian temperature. This can occur when external stimuli, such as radiation, are introduced into the plasma, or when magnetic fields or spatial inhomogeneities reside in the plasma, interacting differently with each species, or when the plasma undergoes rapid compression or expansion. When the time scale of the processes which drive the non-equilibrium are shorter than the electron-ion equilibration time, the plasma will not be in equilibrium. Since the electron-ion equilibration time is proportional to the electron temperature to the 3/2 power and inversely proportional to the mass

density, hot, rarefied plasmas tend to have separate electron and ion distributions. However, rapidly-shocked high density plasmas also tend to exhibit this property over the dynamic time scale of the propagating shock front. This is typical of z-pinch implosions.

To account for the possibility of separate electron and ion temperatures in a fluid code, the single fluid energy equation is traditionally split into an electron energy equation and ion energy equation^a. Each equation contains the physics associated with that species, and can be derived from the Boltzmann equation. For many situations the electron-ion equilibration time might be short compared to other physical processes of interest, and the 3T physics equations remain unimportant. In addition, it is a goal of ALEGRA to be able to accurately simulate the all phases of Z pinch experiments starting with solid density, room temperature wire arrays and ending with plasma stagnation on axis and x-ray radiation production.

VII. Software Infrastructure

The NEVADA project was spun off of the ALEGRA finite element, coupled physics application around the year 2000. The idea was to separate the physics specific parts of the code from the rest, which would contain supporting services, such as the mesh topology, data storage, input deck parsing, mesh input and output, h-type spatial adaptivity, MPI utilities, and others. At that time, the concept of a “framework”, in which multiple applications plug into a core set of services, became popular both politically and in funding streams as a way to increase the speed of developing application capabilities. The ALEGRA code was written in the C++ language because it was viewed as a superior mechanism to manage the complexity of large code projects by using polymorphism and object oriented programming.

As is typical in software history, the early hype of frameworks using the C++ language may have fallen short in practice. However, there is much to be said for both software reuse and the C++ language in an integrated application development environment. Currently, most applications using NEVADA are integrated in both source code and in testing, although a few are using NEVADA in more of a third party library manner.

The classic reason for applications to join an integrated framework is to leverage existing code to do routine but difficult or tedious tasks. Less obvious and measurable, is the synergy obtained from the ability of NEVADA developers to immediately verify code modifications against integrated application tests, and of the applications to have their code updated to the latest NEVADA features, bug fixes, and platform ports. Additionally, all applications can leverage the nightly build and testing activities of NEVADA, the build and porting of third party libraries, and the tools for executing the applications on a variety of platforms.

The main drawback of full integrated development is a loss of flexibility by the applications. Software quality practices may not be entirely compatible between application and framework, there are limitations of the build, test, and execution tools, and maintaining successful tests on multiple platforms can be time consuming. Alternative to full integration is to treat NEVADA as a third party library. However, NEVADA’s interface and implementation is still fast moving, which implies a high frequency of release and patch, and increases the support burden for bugs and installation.

The direction and focus of NEVADA code development continually evolves as new application capabilities are needed in an evolving strategy to optimize customer success and respond to application code team and corporate pressures for increased levels of rigor in code project management. Customers and stakeholders will continue to be the primary source of direction and focus for NEVADA development.

At a high level, the NEVADA project offers third party software management, software components for use in applications, and the tools and processes for application software development. As such, high level goals for NEVADA are as follows.

1. Provide software components that are hard to beat in performance and have the flexibility needed by diverse applications and algorithms.
2. Provide an application development environment in which the overhead associated with necessary activities is minor.
3. Provide third party software management tools that application developers would choose to use even if they were not using the toolkit.

^aWhen the material temperature and electron temperature are close, the physics is referred to as “2T”, for the one material and one radiation temperature. When the electrons and ions are out of equilibrium, this is referred to as “3T”, since there is an electron, ion, and radiation temperature.

4. Provide packaging tools that will enable applications to deliver a positive out-of-box-experience for their customers.

High performing software components are critical to supporting the needs of most applications. Flexibility of the components is important for code reuse and in order to mitigate problems that arise from the difficulty of predicting future application needs and use cases.

The current software development environment must provide for a moderate sized team (10 to 25 people, local and remote), a medium level of rigor applied to software quality practices, many third party libraries, multiple applications, a large number of platforms and compilers, internal and external customers, and extensive test suites. In such an environment, the tools of software development have a huge impact on the overhead of writing correct code and delivering it to application customers.

The packaging, porting, building, and installation of third party libraries and programs is a core capability of NEVADA and could potentially be of great value to applications as a standalone capability.

Finally, from an application user's perspective, the degree of success depends significantly on the ability to obtain, build, and execute the application with little effort. This aspect can be addressed by providing tools to the developer which enable the application to be packaged in a way that is convenient to the end user.

VII.A. Meshing and Communication Services

As an example of key services provided by NEVADA to ALEGRA we highlight the meshing and communication infrastructure utilities.

Early on in the development of ALEGRA all meshes were generated externally, spatially decomposed and then read in parallel. This paradigm works well for small problems but starts to break down as very large problems are desired. A relatively recent thrust, which has gained enthusiastic support from users, is the concept of inline parallel mesh generation. In this approach the mesh is defined in the user input along with a requested decomposition strategy and the mesh is generated and partitioned automatically when the parallel run is launched. Simple meshed objects can currently be generated in parallel. Long term the optimal strategy is seen to scalably connect more general geometry descriptions to automatic parallel meshing strategies.

NEVADA provides a single layer of "ghost" elements in the parallel infrastructure in order to facilitate development of parallel algorithms based on the nevada mesh object. Parallel communications services for updating data residing on the nodes, edges and faces residing on processor boundaries as well as data residing in the "ghost" region are available. This gives the code developer very precise and transparent control over the parallel algorithm implementation. Although distributed memory MPI calls exist "under the hood" in NEVADA, applications developers have no need to interact directly with the lower MPI layers.

One key feature in NEVADA is support for translational and rotational periodic boundaries. The support is built into the communication infrastructure in such a way that unique degrees of freedom and proper rotational and translation updates can be effected within an algorithmic philosophy which views periodic boundary conditions as a variant of a spatial parallel decomposition.⁷⁶ Periodic boundary condition support is useful for verification testing but more particularly can be extremely important from an application perspective to reduce the constraints on a modeled system. For example, a periodic wedge can be used to model a full 360 degree system while allowing for angular motions which are less constraining than a fixed wedge which is constrained to have no normal displacement on the wedge faces.

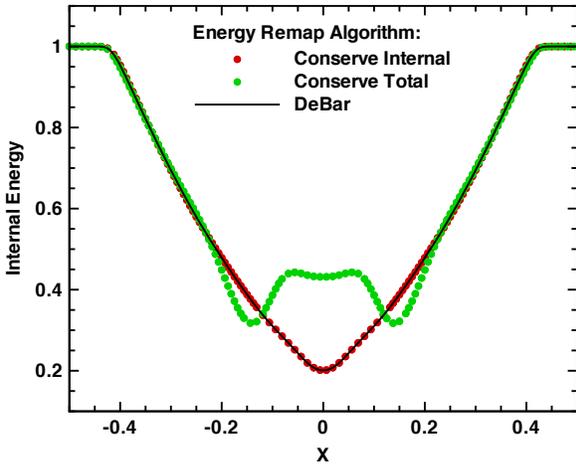


Figure 7. Internal energy profiles for the Einfeldt problem, from simulations with different energy remap algorithms. Straight-forward conservation of total energy often leads to poor prediction of the internal energy in regimes where kinetic energy dominates. The DeBar algorithm performs well in this regime.

VIII. Applications

Below we will show several important applications of ALEGRA. First, we show the performance of ALEGRA on verification problems. Next, the ability of ALEGRA to compute the dynamics of a wire array implosion at Sandia’s Z-machine. Finally, we will demonstrate the performance of ALEGRA on difficult ceramic impact simulations with an advanced fracture model designed to provide mesh-independent results.

VIII.A. Verification

Of the range of different techniques to assess various types of error in numerical simulations, code verification is the foundation on which they all rely. By code verification, we refer to quantitative assessment of ALEGRA algorithms and their implementations through test problems with known reference solutions. As ALEGRA is under continuous development, code verification is an ongoing activity. Previously developed test problems and their associated analyses are maintained and rerun regularly, and this raises the effectiveness of our efforts.

The goals of ALEGRA code verification efforts are to find coding mistakes (bugs) and to identify algorithmic weaknesses. This second goal makes our view of code verification broader and more pragmatic than that in the AIAA V&V guide.⁷⁷ Algorithmic weaknesses are defined in the context of particular applications, and as such, our choices for test problems are weighted towards exact solutions from the literature, rather than manufactured solutions. With that said, manufactured solutions may be the only options for rigorous testing in multiphysics regimes.

While improving our algorithm for remapping the energy, we developed two verification test problems to measure our progress. These two problems illustrate that quantitative analyses are done when possible but understanding how algorithms behave for important applications is also valuable.

A problem which highlights the treatment of the energy is the Einfeldt problem.⁷⁸ This problem consists

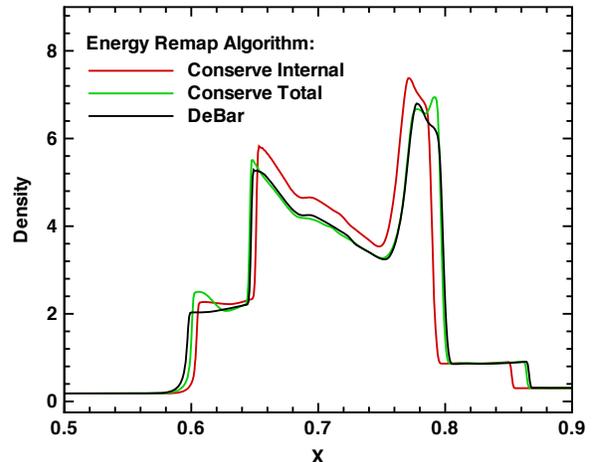


Figure 8. Density profiles for the Woodward-Colella blast waves problem, from simulations with different energy remap algorithms. The DeBar algorithm predicts the same shock locations as the method conserving total energy, but the profile is more similar to the method with conserves internal energy.

of two expansion waves traveling away from each other, leaving a near-vacuum state between. As a Riemann problem the exact solution can be computed and the error is unambiguous. The internal energy is shown for three remap approaches in Fig. 7. In the first approach, the internal energy and the momenta are remapped; this does not conserve the total energy. In the second, the total energy and the momenta are remapped. The internal energy is computed by subtracting the kinetic energy from the total energy, which are nearly the same; any errors in these variables dominate the true value of the internal energy, as seen in Fig. 7. This behavior is typical of methods which conserve total energy in a naive way, and in applications often leads to a lack of robustness. The third approach is the DeBar fix described earlier.

The Woodward-Colella blast waves problem⁷⁹ does not have a rigorously defined solution, but it is very well known in the shock-capturing community. It consists of two strong shocks which reflect off opposite ends of a shock tube. These shocks interact with a number of other waves before colliding. Figure 8 shows typical ALEGRA results for the same three approaches to the remap. Since total energy is not conserved in the first approach, the resulting shock speeds are erroneous. In the second approach the total energy is conserved, but spurious features are visible in the solution behind the strong shocks near $x = 0.6$, $x = 0.65$, and $x = 0.8$. The third approach is the DeBar fix described earlier; the shock locations are the same as when total energy is conserved, but the wave structure matches the profiles computed by conserving internal energy.

VIII.B. Wire-array Implosion

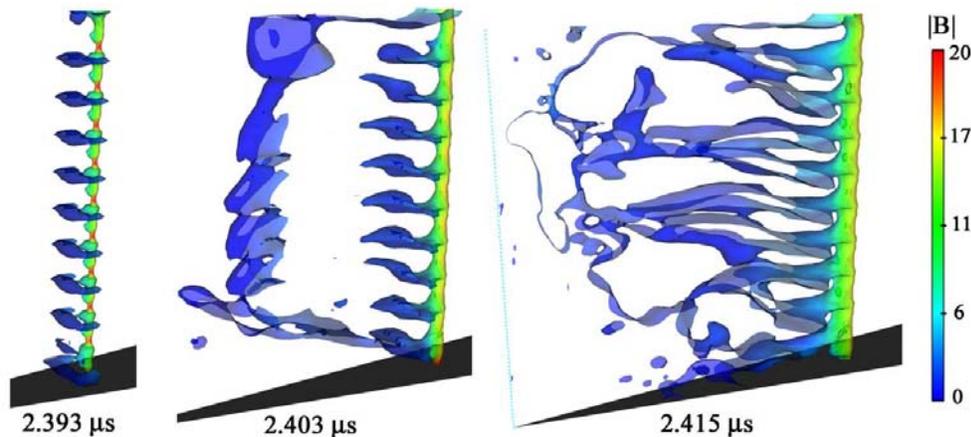


Figure 10. An alegra calculation of the late stages of a wire array implosion. The wires ablate and drive material toward the axis of symmetry, but leaves a substantial amount of mass behind.

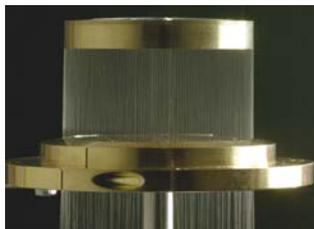


Figure 9. A typical wire array used in Sandia's Z-machine experiments. The wires are made of tungsten and drive an implosion on the axis of the array.

One of the most challenging simulations to conduct with ALEGRA is the implosion of a wire array driven by Sandia's Z-machine. A typical wire array is shown in Figure 9. A powerful electric current is applied to the wires that comprise the wire array. This causes the wires to transition into a plasma and ablate ultimately driving the resultant plasma towards the axis of symmetry of the wire array. The late stages of this process are depicted in Figure 10 showing the complex flow. The implosion creates a strongly radiating

plasma on top of the magnetohydrodynamic flow. In addition substantial mass trails the implosion further complicating the situation by “shorting” the circuit defined by the plasma. The sum total of the physical processes and their intricate interplay represent an enormous challenge to model.

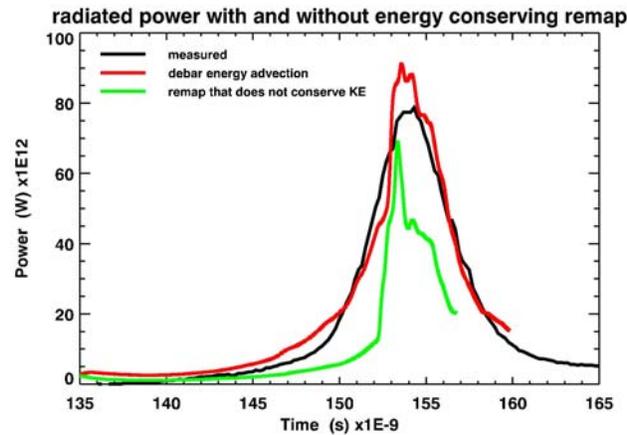


Figure 11. A comparison of the radiated power from a wire array implosion measured experimentally, computed with alegra using the DeBar kinetic energy correction, and without it.

A key product of the wire array implosion is an intense radiation pulse. This radiation can be used in a variety of manners, and the production of this radiation is optimized using results from ALEGRA. The inclusion of the DeBar kinetic energy advection has had a profound impact on the quality of the simulations of wire array implosions. The improvement in the simulation is shown in Figure 11. Prior to the DeBar kinetic energy advection, ALEGRA could not quantitatively or qualitatively reproduce the radiated energy signature. After this algorithmic improvement, ALEGRA could predict this quantity to within the experimental measurement error in most cases.

VIII.C. Advanced Material Modeling

The ALEGRA ceramic model was developed to address mesh dependency issues (which prevent models from being predictive) that ARL analysts were having with existing models for ceramic armor. The new capability is based on the notion that micro-flaws in the ceramic govern the failure processes and strength of ceramics. These micro-flaws are incorporated in the model in a statistical fashion, at the finite element level. An example of these micro-flaws is shown in Figure 12. The ALEGRA ceramic model was developed to address mesh dependency issues (which prevent models from being predictive) with conventional continuum damage models for ceramics. The new capability is based on the notion that micro-flaws in the ceramic govern the failure processes and strength of ceramics. These micro-flaws are incorporated in the model in a statistical fashion, at the finite element level. An example of a statistically seeded mesh is shown in Figure 12.

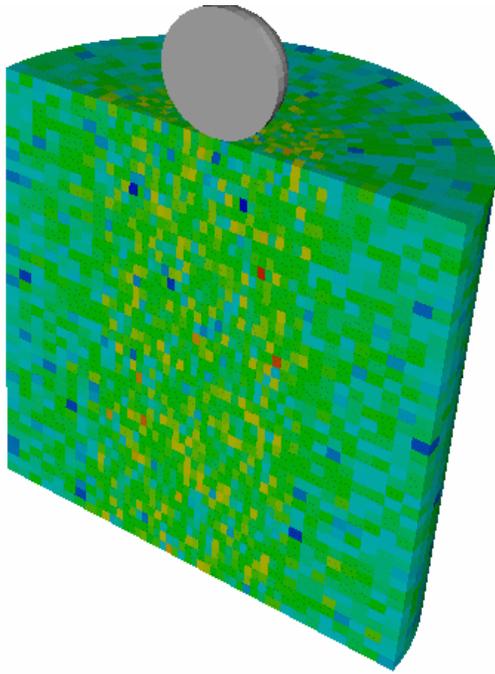
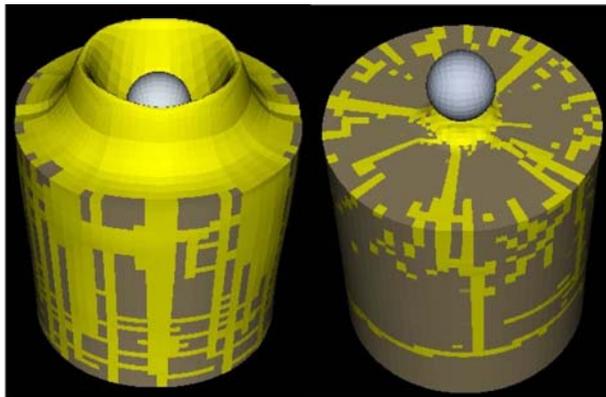


Figure 12. A depiction of the impact of a sphere onto a ceramic. The color coding in the ceramic target denotes the distribution of strengths (representing micro-flaws) in the ceramic. These flaws are critical to the failure process.



Without Variability With Variability

Figure 13. Here we show the difference between the old model (left) for fracture and the new model (right) is shown for the impact problem. The old model produces an unphysical fracture and high mesh dependence while the new model is physically meaningful and much more mesh independent.

Each finite element is assigned a strength that is part of the size-dependent Weibull distribution of strengths observed in experiments. Sandia Brazilian experiments have shown that, on average, small samples are stronger than large samples. The interpretation is that larger samples are more likely to contain critical flaws. Strength experiments have always measured the weakest flaws in a sample, and the ALEGRA ceramic model takes this into account. This is as opposed to assigning the strength measured in the laboratory to the entire mesh, which is done conventionally and results in the bulk of the material being too weak. Some results comparing old and new models is shown in Figure 13 which illustrates the dramatic improvement gained by statistically seeding the mesh.

Recent ALEGRA simulations of have shown a dramatic reduction in mesh size dependency, which is a necessary first step in enabling the model to be predictive.

VIII.D. Magnetic Flyer Plate Launch

Magnetic fields can be used to quasi-isentropically launch flyer plates to extreme velocities for equation of state purposes on the Sandia Z machine. This application has been a significant success for ALEGRA modeling because it drove development of an overall capability that can now be used for predictive design purposes in this extreme environment.⁸⁰

IX. Conclusion

The ALEGRA multiphysics hydrodynamics code has been developed by Sandia for more than 15 years. During this development, substantial capability has been embedded in the code allowing a wide variety of simulations to be conducted. A number of examples are provided above. These utilize the broad spectrum of physical processes available in the code.

References

- ¹HIRT, C. W., AMSDEN, A. A., and COOK, J. L., "An Arbitrary Lagrangian-Eulerian computing method for all flow speeds." *Journal of Computational Physics*, Vol. 14, No. 3, 1974, pp. 227 – 53.
- ²Boris, J. P. and Book, D. L., "Flux-Corrected Transport I. SHASTA, A Fluid Transport Algorithm that Works," *Journal of Computational Physics*, Vol. 11, 1973, pp. 38–69.
- ³van Leer, B., "Towards the Ultimate Conservative Difference Scheme. IV. A New Approach to Numerical Convection," *Journal of Computational Physics*, Vol. 23, 1977, pp. 276–299.
- ⁴Caramana, E. J., Burton, D. E., Shashkov, M. J., and Whalen, P. P., "The Construction of Compatible Hydrodynamics Algorithms Utilizing Conservation of Total Energy," *Journal of Computational Physics*, Vol. 146, 1998, pp. 227–262.
- ⁵Caramana, E. J., Shashkov, M. J., and Whalen, P. P., "Formulations of Artificial Viscosity for Multi-Dimensional Shock Wave Computations," *Journal of Computational Physics*, Vol. 144, 1998, pp. 70–97.
- ⁶Caramana, E. J., Rousculp, C. L., and Burton, D. E., "A compatible, energy and symmetry preserving Lagrangian hydrodynamics algorithm in three-dimensional cartesian geometry." *Journal of Computational Physics*, Vol. 157, No. 1, 2000, pp. 89–119.
- ⁷Caramana, E. J. and Whalen, P. P., "Numerical preservation of symmetry properties of continuum problems." *Journal of Computational Physics*, Vol. 141, No. 2, 1998, pp. 174–98.
- ⁸Loubere, R. and Caramana, E. J., "The force/work differencing of exceptional points in the discrete, compatible formulation of Lagrangian hydrodynamics." *Journal of Computational Physics*, Vol. 216, No. 1, 2006, pp. 1–18.
- ⁹Bauer, A. L., Burton, D. E., Caramana, E. J., Loubere, R., Shashkov, M. J., and Whalen, P. P., "The internal consistency, stability, and accuracy of the discrete, compatible formulation of Lagrangian hydrodynamics." *Journal of Computational Physics*, Vol. 218, No. 2, 2006, pp. 572–93.
- ¹⁰Benson, D. J., "A new two-dimensional flux-limited shock viscosity for impact calculations," *Computer Methods in Applied Mechanics and Engineering*, Vol. 93, No. 1, December 1991, pp. 1991.
- ¹¹Benson, D. J., "Computational Methods in Lagrangian and Eulerian Hydrocodes," *Computer Methods in Applied Mechanics and Engineering*, Vol. 99, 1992, pp. 235–394.
- ¹²Schulz, W. D., "Tensor artificial viscosity for numerical hydrodynamics," *Journal of Mathematical Physics*, Vol. 5, No. 1, January 1964, pp. 133–138.
- ¹³Wilkins, M. L., "Use of Artificial Viscosity in Multidimensional Fluid Dynamic Calculations," *Journal of Computational Physics*, Vol. 36, 1980, pp. 281–303.
- ¹⁴Noh, W. F., "Errors for calculations of strong shocks using an artificial viscosity and an artificial heat flux," *Journal of Computational Physics*, Vol. 72, No. 1, September 1987, pp. 78–120.
- ¹⁵McGlaun, J. M. and Trucano, T. G., "Proposal for Development of a Rad/Hydro Code," Sandia National Laboratories unpublished report.
- ¹⁶Taylor, L. M. and Flanagan, D. P., "PRONTO-2D: A Two-Dimensional Transient Solid Dynamics Program," Technical report SAND86-0594, Sandia National Laboratories, Albuquerque, NM, March 1987.
- ¹⁷Taylor, L. M. and Flanagan, D. P., "PRONTO3D: A Three-Dimensional Transient Solid Dynamics Program," Technical report SAND87-1912, Sandia National Laboratories, Albuquerque, NM, March 1989.
- ¹⁸Bell, R. L. et al., "CTH User's Manual and Input Instructions, Version 4.00," Technical report, Sandia National Laboratories, Albuquerque, NM, 87185, April 1999.
- ¹⁹Budge, K. G. and Peery, J. S., "RHALE-2D: An ALE Shock Code," unpublished report.
- ²⁰Mihalas, D. and Weibel-Mihalas, B., *Foundations of Radiation Hydrodynamics*, Dover, 1999.
- ²¹Pomraning, G. C., *The Equations of Radiation Hydrodynamics*, Pergamon Press, 1973.
- ²²LeVeque, R. J., Mihalas, D., Dorfi, E. A., and Müller, E., *Computational Methods for Astrophysical Fluid Flow*, Springer, 1998.
- ²³Bell, G. I. and Glasstone, S., *Nuclear Reactor Theory*, Krieger, 1970.
- ²⁴Lax, P. D., *Hyperbolic Systems of Conservation Laws and the Mathematical Theory of Shock Waves*, SIAM, Philadelphia, 1972.
- ²⁵Richtmyer, R. D., "Proposed Numerical Method for Calculation of Shocks," Tech. Rep. LA-671, Los Alamos Scientific Laboratory, 1948.
- ²⁶VonNeumann, J. and Richtmyer, R. D., "A Method for the Numerical Calculation of Hydrodynamic Shocks," *Journal of Applied Physics*, Vol. 21, No. 3, March 1950, pp. 232–237.
- ²⁷Noh, W. F., "Errors for Calculations of Strong Shocks Using an Artificial Viscosity and an Artificial Heat Flux," *Journal of Computational Physics*, Vol. 72, 1987, pp. 78–120.
- ²⁸Harten, A., "High Resolution Schemes for Hyperbolic Conservation Laws," *Journal of Computational Physics*, Vol. 49, 1983, pp. 357–393, Reprinted in Volume 135 Number 2, pp. 260–278, August 1997.
- ²⁹Sweby, P. K., "High-Resolution Schemes using Flux Limiters for Hyperbolic Conservation Laws," *SIAM Journal of Numerical Analysis*, Vol. 21, 1984, pp. 995–1011.

- ³⁰van Leer, B., "Towards the Ultimate Conservative Difference Scheme. V. A Second-Order Sequel to Godunov's Method," *Journal of Computational Physics*, Vol. 32, 1979, pp. 101–136.
- ³¹Belytschko, T., Ong, J. S.-J., Liu, W. K., and Kennedy, J. M., "Hourglass control in linear and nonlinear problems," *Computer Methods in Applied Mechanics and Engineering*, Vol. 43, No. 3, May 1984, pp. 251–276.
- ³²Puso, M. A., "A highly efficient enhanced assumed strain physically stabilized hexahedral element," *International Journal for Numerical Methods in Engineering*, Vol. 49, No. 8, November 2000, pp. 1029–1064.
- ³³Simo, J. C., Armero, F., and Taylor, R. L., "Improved versions of assumed enhanced strain tri-linear elements for 3D finite deformation problems," *Computer Methods in Applied Mechanics and Engineering*, Vol. 110, No. 3-4, December 1993, pp. 359–386.
- ³⁴Hughes, T. J. R., *The Finite Element Method: Linear Static and Dynamic Finite Element Analysis*, Dover Publications, 2000.
- ³⁵Belytschko, T., Liu, W. K., and Moran, B., *Nonlinear finite elements for continua and structures*, John Wiley and Sons, New York, 2000.
- ³⁶Flanagan, D. P. and Belytschko, T., "A uniform strain hexahedron and quadrilateral with orthogonal hourglass control," *International Journal for Numerical Methods in Engineering*, Vol. 17, No. 5, 1981, pp. 679–706.
- ³⁷Ween, F. V. and Belytschko, T., "Correction of article by D. P. Flanagan and T. Belytschko," *International Journal for Numerical Methods in Engineering*, Vol. 19, No. 3, 1983, pp. 467–468.
- ³⁸Freitag, L., Knupp, P., Leurent, T., and Melander, D., "MESQUITE Design: Issues in the Development of a Mesh Quality Improvement Toolkit," *Proceedings of the Eighth International Conference on Grid Generation Methods for Numerical Field Simulations*, Honolulu, HI, 2002, pp. 159–168.
- ³⁹Brewer, M., Diachin, L. F., Knupp, P., Leurent, T., and Melander, D., "The Mesquite Mesh Quality Improvement Toolkit," *Proceedings of the 12th International Meshing Roundtable*, Sandia National Laboratories, Santa Fe, NM, USA, 2003, pp. 239–250.
- ⁴⁰Tipton, R., "Grid optimization by equipotential relaxation," Technical report UNPUBLISHED, Lawrence Livermore National Laboratories, Livermore, CA, 1992.
- ⁴¹Hansen, G. A., Douglass, R. W., and Zardecki, A., *Mesh enhancement: Selected elliptic methods, foundations and applications*, Imperial College Press, London, 2005.
- ⁴²Knupp, P., "Formulation of a Target-Matrix Paradigm for Mesh Optimization," Technical report SAND2006-2730J, Sandia National Laboratories, Albuquerque, NM 87185, 2006.
- ⁴³Freitag, L., Knupp, P., Munson, T., and Shontz, S., "A Comparison of optimization software for mesh shape-quality improvement problems," *Proceedings of the 11th International Meshing Roundtable*, Sandia National Laboratories, Ithaca, NY, 2002, pp. 29–40.
- ⁴⁴Benson, D. J., "Momentum Advection on a Staggered Mesh," *Journal of Computational Physics*, Vol. 100, 1992, pp. 143–162.
- ⁴⁵Colella, P. and Woodward, P., "The Piecewise Parabolic Method (PPM) for Gas-Dynamical Simulations," *Journal of Computational Physics*, Vol. 54, 1984, pp. 174–201.
- ⁴⁶Rider, W. J., Greenough, J. A., and Kamm, J. R., "Accurate Monotonicity- and Extrema-Preserving Methods through Adaptive Nonlinear Hybridizations," *Journal of Computational Physics*, Vol. In Press, 2007.
- ⁴⁷DeBar, R., "Fundamentals of the KRAKEN Code," Tech. Rep. UCIR-760, LLNL, 1974.
- ⁴⁸Pember, R. and Anderson, R., "A Comparison of Staggered-Mesh Lagrange Plus Remap and Cell-Centered Direct Eulerian Godunov Schemes for Eulerian Shock Hydrodynamics," Nuclear Explosives Code Developers Collaborations (NECDC) 2000 UCRL-JC-139820, Lawrence Livermore National Laboratory, November 2000.
- ⁴⁹Rider, W. J. and Kothe, D. B., "Reconstructing Volume Tracking," *Journal of Computational Physics*, Vol. 141, No. 1, 1996, pp. 112–152.
- ⁵⁰Mosso, S., Garasi, C., and Drake, R., "A smoothed two- and three-dimensional interface reconstruction method," *Computing and Visualization in Science*, Accepted for Publication 2008.
- ⁵¹Youngs, D. L., "Time-Dependent Multi-Material Flow with Large Fluid Distortion," *Numerical Methods for Fluid Dynamics*, edited by K. W. Morton and M. J. Baines, 1982, pp. 273–285.
- ⁵²Mosso, S., Swartz, B., and Kothe, D., "A parallel, volume-tracking algorithm for unstructured meshes," *Parallel Computational Fluid Mechanics*, edited by P. Shiano, A. Ecer, J. Periaux, and N. Satofuka, Elsevier, 1997, pp. 368–375.
- ⁵³Shashkov, M., "Notes on Tipton's Method," .
- ⁵⁴Miller, G. H. and Puckett, E. G., "A High-Order Godunov Method for Multiple Condensed Phases," *Journal of Computational Physics*, Vol. 128, 1996, pp. 134–164.
- ⁵⁵Colella, P., Glaz, H. M., and Ferguson, R. E., Multifluid Algorithms for Eulerian Finite Difference Methods, In preparation.
- ⁵⁶Hail, T. A. and Robinson, A. C., "Two-Dimensional Finite-Element Magneto-hydrodynamic Equations in ALEGRA, Part I: Magnetic Field Formulation," draft SAND report, Sandia National Laboratories, Albuquerque, NM 87185, 2005, to be published.
- ⁵⁷Hail, T. A. and Robinson, A. C., "Two-Dimensional Finite-Element Magneto-hydrodynamic Equations in ALEGRA, Part II: Vector Potential Formulation," draft SAND report, Sandia National Laboratories, Albuquerque, NM 87185, 2005, to be published.
- ⁵⁸Bochev, P. B., Hu, J. J., Robinson, A. C., and Tuminaro, R. S., "Towards robust 3D Z-pinch simulations: discretization and fast solvers for magnetic diffusion in heterogeneous conductors," *Electronic Transactions on Numerical Analysis (ETNA)*, Vol. 15, 2003, pp. 186–210, <http://etna.mcs.kent.edu>.
- ⁵⁹Schiemenz, A. R. and Robinson, A. C., *CSRI Summer Proceedings 2007 - The Computer Science Research Institute at Sandia National Laboratories*, edited by M. L. Parks and S. S. Collis,

<http://www.cs.sandia.gov/CSRI/Proceedings/CSRI2007.pdf>, SAND2007-7977, 2007, pp. 231–241, Energy Based Magnetic Forces Computation using Automatic Differentiation.

⁶⁰Robinson, A. C. and Garasi, C. J., “Three-dimensional z-pinch wire array modeling with ALEGRA-HEDP,” *Computer Physics Communications*, Vol. 164, No. 1-3, December 2004, pp. 408–413.

⁶¹Bochev, P. B. and Robinson, A. C., *Collected Lectures on the Preservation of Stability under Discretization*, chap. 8: Matching Algorithms with physics: exact sequences of finite element spaces, SIAM, 2002.

⁶²Tuminaro, R. S. et al., “Official Aztec Users’s Guide - Version 2.1,” Technical report SAND99-8801J, Sandia National Laboratories, Albuquerque, NM 87185, Nov. 1999.

⁶³Tong, C. and Tuminaro, R. S., “ML 2.0 Smoothed Aggregation User’s Guide,” Technical report SAND2001-8028, Sandia National Laboratories, Albuquerque, NM 87185, Dec. 2000.

⁶⁴Desjarlais, M. P., “Practical improvements to the Lee-More conductivity near the metal-insulator transition,” *Contributions to Plasma Physics*, Vol. 41, No. 2-3, March 2001, pp. 267–270.

⁶⁵Desjarlais, M. P., Kress, J. D., and Collins, L. A., “Electrical conductivity for warm, dense aluminum plasmas and liquids,” *Physical Review E*, Vol. 66, No. 2, Aug. 2002, pp. 025401(R).

⁶⁶Lemke, R. W., Knudson, M. D., Robinson, A. C., Hail, T. A., Struve, K. W., Asay, J. R., and Mehlhorn, T. A., “Self-consistent, two-dimensional, magnetohydrodynamic simulations of magnetically driven flyer plates,” *Physics of Plasmas*, Vol. 10, No. 5, May 2003, pp. 1867–1874.

⁶⁷Evans, C. R. and Hawley, J. F., “Simulation of magnetohydrodynamic flows: a constrained transport method,” *The Astrophysical Journal*, Vol. 332, Sept. 1988, pp. 659–677.

⁶⁸Robinson, A. C., Bochev, P., and Rambo, P., “Constrained Transport Remap on Unstructured Quadrilateral and Hexahedral Grids,” Presented SIAM National Meeting, San Diego, California, July 2001.

⁶⁹Rieben, R. N., White, D. A., Wallin, B. K., and Solberg, J. M., “An arbitrary Lagrangian-Eulerian discretization of MHD on 3D unstructured grids,” *Journal of Computational Physics*, Vol. 226, No. 1, September 2007, pp. 534–570.

⁷⁰Shaskhov, M. and Steinberg, S., “Solving Diffusion Equations with Rough Coefficients in Rough Grids,” *Journal of Computational Physics*, Vol. 129, 1996, pp. 383–405.

⁷¹Bochev, P. B. and Robinson, A. C., “Matching algorithms with physics: exact sequences of finite element spaces,” *Collected Lectures on the Preservation of Stability under Discretization*, edited by D. Estep and S. Tavener, chap. 8, SIAM, 2002.

⁷²Fleck, Jr., J. A. and Cummings, J. D., “An Implicit Monte Carlo Scheme for Calculating Time and Frequency Dependent Nonlinear Radiation Transport,” *Journal of Computational Physics*, Vol. 8, 1971, pp. 313–342.

⁷³Brunner, T. A., Urbatsch, T. J., Evans, T. M., and Gentile, N. A., “Comparison of four parallel algorithms for domain decomposed implicit Monte Carlo,” *Journal of Computational Physics*, Vol. 212, No. 2, March 2006, pp. 527–539.

⁷⁴Gentile, N. A., Kalos, M., and Brunner, T. A., “Obtaining Identical Results on Varying Numbers of Processors in Domain Decomposed Particle Monte Carlo Simulations,” *Computational Methods in Transport: Granlibakken 2004*, edited by F. Graziani, Vol. 48 of *Lectures Notes in Computational Science and Engineering*, Springer, 2006, Also UCRL-PROC-210823.

⁷⁵Gentile, N. A., Keen, N., and Rathkopf, J., “The KULL IMC Package,” Tech. Rep. UCRL-JC-132743, Lawrence Livermore National Laboratory, Livermore, CA, 1998.

⁷⁶Robinson, A. C., Weatherby, J. R., and Aidun, J. B., “Periodic Boundary Conditions in the ALEGRA Finite Element Code,” Technical report SAND99-2698, Sandia National Laboratories, Albuquerque, NM, Nov. 1999.

⁷⁷AIAA, “Guide for the verification and validation of computational fluid dynamics simulations,” Tech. Rep. AIAA-G-077-1998, American Institute of Aeronautics and Astronautics, Reston VA, USA, 1998.

⁷⁸Einfeldt, B., Munz, C. D., Roe, P. L., and Sjögreen, B., “On Godunov-Type Methods near Low Densities,” *Journal of Computational Physics*, Vol. 92, 1991, pp. 273–295.

⁷⁹Woodward, P. and Colella, P., “The Numerical Simulation of Two-Dimensional Fluid Flow with Strong Shocks,” *Journal of Computational Physics*, Vol. 54, 1984, pp. 115–173.

⁸⁰Mehlhorn, T., Brunner, T., Desjarlais, M., Garasi, C., Hail, T., Hanshaw, H., Lemke, R., Mattsson, T., Matzen, M., Oliver, B., Robinson, A., Slutz, S., T.G.Trucano, Yu, E., Vesey, R., Cuneo, M., Jones, B., Knudson, M. D., and Sinars, D., “Towards a predictive MHD simulation capability for designing hypervelocity magnetically-driven flyer plates and PW-class z-pinch x-ray sources on Z and ZR,” 2006.