Parallel Mesh Relaxation and Advection Remap for High Order Curvilinear ALE Hydrodynamics

VLADIMIR TOMOV[3] WITH TZANIO KOLEV[1], VESELIN DOBREV[1], ROBERT RIEBEN[2], ROBERT ANDERSON[1]

[1]Center for Applied Scientific Computing, Lawrence Livermore National Laboratory
[2]Weapons and Complex Integration, Lawrence Livermore National Laboratory
[3]Department of Mathematics, Texas A&M University

Introduction

Pure Lagrange methods are often preffered for hydrodynamic calculations, because they provide a natural form of mesh adaptivity and easy coupling to other physics. The disadvantage is lack of robustness - extreme mesh distortion may cause tangling or too small CFL time steps. This motivates the use of ALE methods - they consist of a Lagrange phase, followed by mesh relaxation, solution remap and multi-material interface reconstruction. Our main goal is to extend the LLNL's Lagrangian High Order FE BLAST code to a full ALE code. We propose some methods and parallel implementations for the mesh relaxation and solution remap phases. We explore their accuracy and conservation of mass, kinetic energy and momentum.

Lagrange phase

The Lagrange method is based on taking the Euler equations of compressible hydrodynamics and making the computational mesh move with the material velocity. The formation of shocks is handled by adding artificial viscosity terms. The resulting equations are:

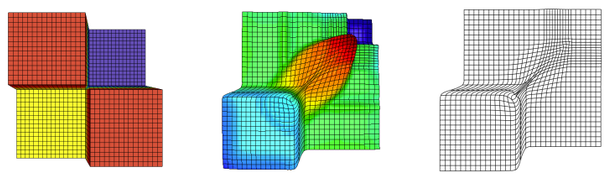
Equation of motion: $\frac{dx(x_0,t)}{dt} = v(x,t)$

Mass conservation: $\frac{d\rho(x,t)}{dt} = -\rho(x,t)\nabla \cdot v(x,t)$

Momentum conservation: $\rho(x,t)\frac{dv(x,t)}{dt} = \nabla \cdot \sigma(x,t)$

Energy conservation: $\rho(x,t)\frac{de(x,t)}{dt} = \sigma(x,t) : \nabla v(x,t)$

where $\sigma = -pI + \sigma_a$ is the total stress tensor that includes the artificial stress σ_a. Different finite element discretizations can be derived depending on the choice of shock-tracking method, viscosity terms and FE spaces. One alternative to the LLNL's Lagrangian High Order FE BLAST code is the Texas A&M's Entropy Viscosity FE Method (J.-L. Guermond, B. Popov, V. Tomov). That method tracks the shock positions by observations on the system's entropy production.
Example: A Riemann problem with 4 initial states immediately leading to strong compression in the middle of the mesh, motivating the need for some form of mesh relaxation:



Initial condition, final solution and mesh deformation at time 0.2 (Entropy Viscosity Method)

Mesh Relaxation

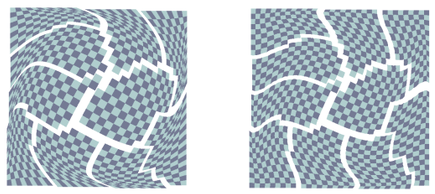
The goal of this step is to increase the CFL timestep and to avoid mesh tangling. We have developed a parallel method that implements the relaxation scheme:

$$x^{n+1} = x^n + M^{-1}\left(f - Lx^n\right)$$

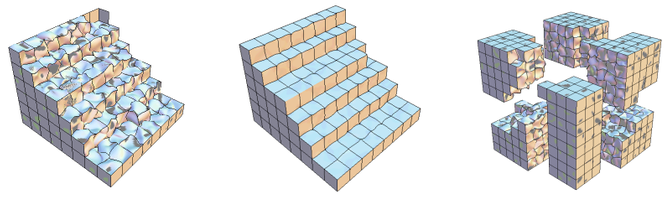
- x^n are the nodes of the high-order mesh after n relaxation steps.
- L is a topological operator, with $L_{ii} = 1$ and $\sum_j Lij = 1$. Furthermore, if J is the set of indices corresponding to the interior nodes and B is the set of indices for the boundary nodes, then we have $L_{BJ} = 0$ and $L_{BB} = I$ meaning our boundary nodes do not move.
- f is a vector containing $-L_{JB}x_B$ as J indices and x_B as B indices.
- M is a preconditioner for L.

The choice of high-order mesh Laplacian L determines the smoothing properties. Our parallel implementation works with the following options:

- L_1 - the matrix is formed by the FE space sparsity pattern, equal weights. Example: Q_2 mesh relaxed by L_1, 5 steps, 10 mpi tasks:



- L_2 - the matrix is formed by the high-order stiffness matrix on the reference element. Example: 3D Q_3 mesh relaxed by L_2, 3 steps, 6 mpi tasks:



Future work:
- local mesh optimization - smoothing in specific parts of the domain.
- solution-dependent mesh smoothing - algorithm making use of information about shock positions and material boundaries.
- parallel versions of other mesh smoothers.

Advection Remap

We're discretizing the advection equation:

$$\frac{d\rho(x,t)}{d\tau} = u(x,\tau) \cdot \nabla\rho(x,\tau)$$

where τ is the "pseudo-time" in which the old (perturbed) mesh transitions to the new (relaxed) mesh, u is the mesh's pseudo-velocity, x corresponds to the mesh's configuration at time τ and ρ is a scalar function we want to remap. We choose a continuous FE space $V \subset H^1$ and its moving basis $\{\psi_i\}$ and define:

$$M_{ij} = \int_{\Omega(\tau)} \psi_j\psi_j, \quad A_{ij} = \int_{\Omega(\tau)} \psi_j u \cdot \nabla\psi_i, \quad \rho(x,t) = \boldsymbol{\rho}^T\boldsymbol{\psi}(x,\tau), \quad \boldsymbol{m} = M\boldsymbol{\rho}$$

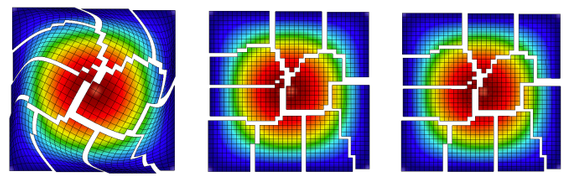
We compare two alternative semi-discrete matrix-vector forms:

- function - based formulation: $\frac{\partial\boldsymbol{\rho}}{\partial\tau} = -M^{-1}A^T\boldsymbol{\rho}$

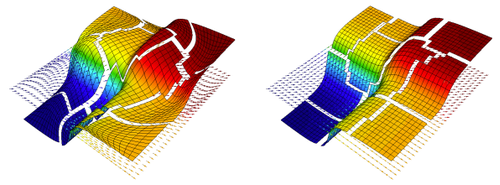
- moment - based formulation: $\frac{\partial\boldsymbol{m}}{\partial\tau} = AM^{-1}\boldsymbol{m}$

The velocity field is remapped by a function - based momentum remap method:

$$\frac{\partial\boldsymbol{v}}{\partial\tau} = -M_\rho^{-1}A_\rho^T\boldsymbol{v}$$



Density on initial mesh, function and moment remaps for $\rho = \sin(\pi x)\sin(\pi y)$



Remap of velocity field $v = (\pi/2 + \arctan(20(x-0.5)), \pi/2 + \arctan(20(y-0.5)))$

Errors between original and remapped fields:

h, # steps	$L^\infty(\rho)$	$L^1(\rho)$	$L^2(\rho)$	mass	kin. energy	momentum (x, y)
1/8, 5	$7.7e^{-3}$	$1.5e^{-3}$	$2.0e^{-3}$	$2.2e^{-7}$	$2.5e^{-6}$	$(3.49e^{-7}, 3.49e^{-7})$
1/16, 10	$2.2e^{-3}$	$3.3e^{-4}$	$4.6e^{-4}$	$1.4e^{-8}$	$8.7e^{-7}$	$(2.21e^{-8}, 2.21e^{-8})$
1/32, 20	$6.5e^{-4}$	$7.9e^{-5}$	$1.1e^{-4}$	$8.9e^{-10}$	$7.8e^{-8}$	$(1.40e^{-9}, 1.40e^{-9})$
1/64, 40	$1.8e^{-4}$	$2.1e^{-5}$	$3.2e^{-5}$	$5.1e^{-11}$	$6.9e^{-9}$	$(7.9e^{-11}, 8.0e^{-11})$

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

LLNL-POST-570159