Title

Good morning, my name is Truman Ellis. I have spent the last two summers at LLNL studying high order finite elements for Lagrangian CFD.

My advisor is Dr. K and Drs. Robert Rieben and Tzanio Kolev directed our research group at LLNL.

Outline

… but before we get into the meat of the matter, let's consider a possible practical application for this research...

Frameworks

Most people familiar with CFD with tend to view it according to the Eulerian perspective. So let me explain what I mean by Lagrangian CFD. There are three primary perspectives or frameworks that you can use to describe the equations of fluid dynamics.

The Eulerian framework is the most common. You draw a mesh, initialize your fluid, then watch the fluid flow through your stationary mesh. This is especially appropriate when the problem **domain does not change**.

The Lagrangian framework is less common, but equally valid. Here you draw an initial mesh, initialize your fluid properties, then move your mesh nodes with fluid. This eliminates the need for any flux calculations because each zone represents a **small moving fluid volume**. This is especially appropriate when the problem **domain itself is changing**, as with many unsteady simulations. The area where Lagrangian methods particularly shine is with **multi-material** simulations and **interface tracking**. Another nice attribute of Lagrangian CFD is that you automatically get mesh **refinement** near shocks due to the tendency of shocks to compress surrounding fluid. Lagrangian simulations are also less **numerically diffusive**.

The figure below is a Lagrangian simulation of a triple-point shock. One downside to the Lagrangian description is that that for complicated or vortical flows, the mesh tends to get **tangled** which can cause a simulation to crash. Notice how the mesh is about to tangle.

This is where the ALE technique comes in handy. We run the Lagrangian simulation until mesh tangling is detected. We then pause the simulation for the mesh relaxation stage. We then move the mesh back to a less tangled state while advecting the stationary fluid through the moving mesh. Then the simulation is resumed. ALE attempts to combine the best aspects of both the Eulerian and Lagrangian frameworks.

Introduction

The figure on the right illustrates an ALE simulation of the triple point shock problem. This is a much higher resolution simulation, but notice all of the extra details of the flow structure that can be observed.

This thesis...

Glossary

Euler Equations in a Lagrangian Frame

If we take the full Navier-Stokes equations of fluid motion and assume that inertial forces are much more significant than viscous forces (i.e. high Reynolds number), we can negate the viscous terms and obtain the much simplified Euler equations which consist of conservation of mass, momentum, and energy. Finally, the fluids equation of state closes the system. These can be solved by a variety of methods: **finite difference**, **finite volume** – we will be using finite elements.

Domain Decomposition

Let us first derive a semi-discrete finite element approach to solving the Euler equations. This thesis is concerned with the spatial discretization of the problem. Any temporal discretization could be applied to the end result, such as a **predictor-corrector**, Runge-Kutta, etc. We will stick to a simple forward Euler-like time scheme.

Anyone who is familiar with CFD knows that the first step is to divide the original problem domain into a computational mesh.

Lagrangian Mesh Motion

As the simulation progresses, we will propagate the mesh nodes with the fluid and reconstruct each zone based on these moved particles or nodes. There is an inherent error in this reconstruction process. Suppose our fluid volume would have ideally deformed into a curved shape, very reasonable considering it is a fluid. With bi-linear elements, we can only represent straight edges. This would introduce more reconstruction error than bi-quadratic elements. This issue comes up when we consider the Sedov explosion problem. If we take an initially Cartesian grid and apply the exact solution of the Sedov problem, we arrive at the result shown on the right. Low order methods are simply unable to represent these curved zones.

Lagrangian Mesh Motion: Animation

Let's watch a video of a Lagrangian simulation in action. This is the solution of the Sedov problem with high order finite elements. Initially we have a perfectly Cartesian mesh. We introduce a large amount of **energy** into the bottom left zone and apply **symmetry** BC's to the bottom and left edges. This is basically a simplified explosion. A blast wave propagates **circularly** out from the origin. The computational mesh is **dragged** along with the expanding fluid.

Curved Geometries

But how do we handle curved zones mathematically? The solution is actually pretty straightforward. We define each finite element basis function on a **unit reference** element and use the Jacobian of transformation to map to physical space. The Jacobian is simple to define for any order mapping: bi-linear, bi-quadratic, bi-cubic, etc. In two dimensions, it is always just a two by two matrix that is a function of the physical nodes.

Basis Functions

Each continuum field, whether velocity, density, pressure, or energy is represented by a basis function expansion. In general, each of these expansions may be chosen independently, but stability considerations dictates that certain choices are not independent. For example, **bi-quadratic** velocities do not work with piece-wise **constant** pressures. There is just not enough pressure information to move all of the higher order DOFs.

Semi-Discrete Momentum Conservation

The first step in any finite element solution is to redefine the governing equations in a variational formulation.

Let's take our conservation of momentum equation. First we multiply by a vector-valued test function and integrate over the computational mesh. Integrating by parts, we arrive at a surface integral and a volume integral. Now we replace each state variable with our basis function expansion. Taking a clue from Galerkin's method, we can assume that our test functions are in the same space as the velocity basis. Writing this in matrix form...

Computing Mass and Derivative Matrices

Ok, but how do we compute these matrices for arbitrarily distorted zones. The Jacobian of transformation makes this easy. We can perform the integration in reference space, a unit square, and transform to physical space. We use Gauss-Legendre quadrature to evaluate these integrals using a set of sample points and weights.

Semi-Discrete Mass Conservation

What about mass conservation? We define mass moments for each zone to be the integral over that zone of the density times each density basis function.

The Lagrangian description requires that the mass moments within each zone are constant.

If we insert our basis function expansion for density, we can get a matrix form of mass conservation for each zone, where M_z^rho is the local density mass matrix.

Additionally, we could impose a stronger mass moment conservation condition. This gives us the strong mass conservation principle that the density times the determinant of the Jacobian for any point is constant in time. With this assumption, we can avoid recomputing the global mass matrix at every time step.

Energy Conservation, EOS, and Artificial Viscosity

Now let's consider some miscellaneous loose ends.

We handle energy conservation locally. In this research we only consider a piece-wise constant energy discretization. Basically, the time rate of change of energy in a zone is the forces associated with that zone dotted with the velocity of that zone, i.e. power. The change in internal energy is the work done in that time step.

We could consider the equation of state in a variational sense, but for our purposes it works to just evaluate it point-wise to obtain the pressure update.

Ideally we would like to represent shocks with perfect discontinuities, but practically we require some artificial viscosity to prevent shocks from crushing zones to zero volume. This tensor artificial viscosity force is added to the corner forces of each zone and is define as the divergence of mu times the gradient of velocity. In a variational form, this becomes -S_z V_z where S is finite element stiffness matrix. The coefficient, mu contains adjustable linear and quadratic terms to control oscillations and shock smoothing.

Mixed Finite Elements

Now that we have derived a general finite element framework with which to solve the Euler equations, we can try a few choices for kinematic and thermodynamic basis functions. Here is a sampling of various finite element pairs that we have considered...

High Order Methods Using the Q2 Basis

The Q2-Q1d method showed the most promise in our investigation, and from here on out, we will mostly be comparing Q2-Q1d with the traditional Q1-Q0 pair.

Here are some examples of mapping from reference space to physical space using the Q2 Jacobian.

High Order Methods Using the Q2 Basis

At each time step we need to compute the following matrices using Gauss-Legendre quadrature.

Each local mass matrix is a symmetric 9x9 positive definite matrix.

The derivative matrix is a 4x9 rectangular matrix used in the momentum solve.

The stiffness matrix is a 9x9 symmetric matrix which is a discrete version of the Div Grad operator, used in the artificial viscosity.

In the following simulations we will use mass lumping and apply the strong mass conservation principle to avoid recomputing the mass matrix at each time step.

Static Momentum Equation Solve Convergence Study

Before we implement any method, we need to make sure it can solve the momentum equation. Here we project a pressure field onto a series of meshes and solve the static momentum equation. We then calculate the L2 error norm and graph the convergence rates.

The plots on the right make it clear that high order methods converge much more quickly to the exact solution.

1D Sod Shock Tube: Velocity and Density

Now let's consider a one-dimensional simulation. We initialize a tube of length one with a high density gas on the left and a low density gas on the right. When the diaphragm bursts, the high density gas expands into the low density and a shock forms. At the same time an expansion wave forms in the high density gas. This problem is useful because it is one-dimensional, it contains the essential aspects of compressible flow, it has an exact solution, and we can determine whether high order methods show any promise for compressible Lagrangian simulations.

We consider three cases: Q1-Q0 on a fine mesh, and Q2-Q1d on both a coarse and fine mesh. The coarse Q2-Q1d has the same number of DOFs as the fine Q1-Q0.

1D Sod Shock Tube: Velocity and Density (Zoomed View)

We can see some more details when we zoom in on the shock.

First of all, the velocity plot shows us that we need twice as many low order zones to achieve comparable accuracy to the coarse high order method. This becomes important later when we consider computational efficiency. In general you need 2^nD times more Q1-Q0 elements than Q2-Q1d elements to achieve comparable accuracy. Therefore in two dimensions you need 4 times more elements and in 3D you would need 8 Q1-Q0 elements for every 1 Q2-Q1d element.

Also notice...

The Acoustic Wave Problem

Now let's consider a two-dimensional low-speed problem. The acoustic wave problem lets us test for the presence or absence of spurious velocity and pressure modes. The surest way of exciting any latent spurious modes is with a delta function disturbance. Therefore we

wiggle the center node and observe an acoustic wave propagate outward. The superposition of any noise identifies the presence of spurious modes.

Acoustic Wave Problem: Q1-Q0

When we solve this problem with Q1-Q0 elements, we observe the well-known hourglass / checkerboard modes.

Acoustic Wave Problem: Q2-Q1d

Hourglass / Checkerboard Modes

Vesselin Dobrev and Tzanio Kolev proposed that hourglass modes are any continuous velocity mode that has non-zero divergence but zero discrete divergence. We can observe this in the two patches of Q1-Q0 and Q2-Q1d elements. Q2-Q1d appears to have the same hourglass structure as Q1-Q0, but on a finer level. This is somewhat disappointing...

The Noh Implosion Problem

The Saltzman Piston Problem

Saltzman Piston Problem, 50 by 10 Mesh: Q1-Q0

Shock front is not straight.
Waves in the horizontal mesh lines.
Significant shock-mesh interference.

Saltzman Piston Problem, 50 by 10 Mesh: Q2-Q1d

Straighter shock front.
Straighter horizontal mesh lines.
Reduced shock-mesh interference.

The Sedov Explosion Problem

Mesh Comparison

Now let's take a look at both solutions side by side.
Q2-Q1d shock front is much more circular.
Zones are much more realistically curved.
Shock is resolved much more sharply.

Full Mesh Comparison

Now let's try something a little different. Instead of splitting the explosive energy over four

zones, let's concentrate it all in one element. The original Q1-Q0 element blows up, but it lack the flexibility to naturally expand with the force of the blast. The solution eventually crashes. The Q2-Q1d element, on the other hand, is flexible enough to expand naturally with the force of the explosion, and produces a very respectable solution.

Other Elements Considered: Q1-Q1d

In my thesis I actually considered two more elements in depth. If we take the traditional Q1-Q0 and enrich the pressure space, we arrive at the Q1-Q1d element.

Other Elements Considered: Q2-Q2d

Analogously, if we further enrich the pressure space of Q2-Q1d, we arrive at Q2-Q2d...

Computational Efficiency

Now let's take a look at the relative computational efficiency of each method. We learned that in general it takes 2^nD low order elements to achieve comparable accuracy as a bi-quadratic method...

Conclusions