Shocktube Problem - Status Report 4
ICE Algorithm Description

Oren E. Livne *

April 26, 2005

Abstract

We describe the main ICE advection algorithm for scalar advection problems (advection and Burgers), and then for the compressible Euler system of equations (Sod's Shocktube problem), without boundary conditions. We explain each step in this original Kashiwa ICE algorithm [Kas00], and its relation of both the Davis scheme [Dav87] and the algorithm currently implemented in Uintah.

Key words. Advection, ICE, Shocktube problem, Euler equations, limiters, Davis scheme.

*SCI Institute, 50 South Central Campus Dr., Room 3490, University of Utah, Salt Lake City, UT 84112.Phone: +1-801-581-4772. Fax: +1-801-585-6513. Email address: livne@sci.utah.edu

Contents

1 ICE Algorithm for Scalar Advection

We describe and implement Bucky's ICE algorithm for scalar conservation laws, see [Kas00, pp. 27–29]. This scheme turns to be almost equivalent to Davis' scheme [Dav87], however, it produces better results than those reported in [Liv05] for the Davis scheme. In §1.1 we define the equation.

1.1 The Equations

We cosnider a general one dimensional scalar advection equation,

$$q_t + f(q)_x = q_t + (f'(q))q_x = 0, \tag{1}$$

defined over an infinite domain, or for $x \in [0,1]$ with periodic boundary conditions. The convective velocity is denoted by $u = u(x) = f(q(x))$.

1.2 Discretization

We use a finite volume discretization on a uniform grid with meshsize Δx. The discrete $\{q_j\}_j$ are defined at cell centers $x_j = j\Delta x$; the numerical fluxes $\{f_{j+\frac{1}{2}}\}_j$ are defined at face centers, $x_{j+\frac{1}{2}} = (j+\frac{1}{2})\Delta x$; the fluxing velocities $u^*_{j+\frac{1}{2}}$ are defined at face centers. See Fig. 1.

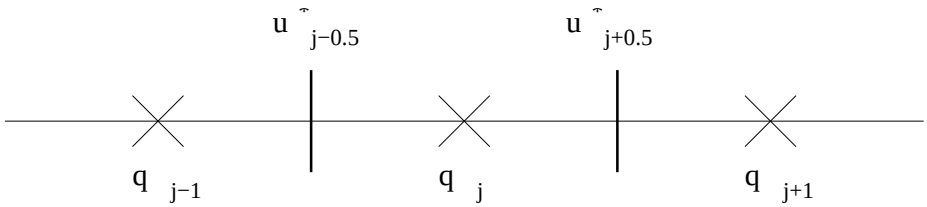

Figure 1: Cell-centered discretization of (1). The discrete state variables q_j are defined at cell centers j ("x"). Cell faces $j+\frac{1}{2}$ are marked by "|", where f and u are defined.

1.3 Advection Scheme

The ICE algorithm timestep for computing $q^{(n+1)}$ at time t_{n+1} at all cell centers j from $q = q^n$ at all cell centers consists of the following steps:

1. Compute the numerical flux: compute the fluxing velocity $u^*_{j+\frac{1}{2}}$ at all faces using an averaged Jacobian from neighboring cell centers Jacobians.
$$u^*_{j+\frac{1}{2}} = 0.5\left(f'\left(q_j\right) + f'(q_{j+1})\right) \qquad \forall j. \tag{2}$$

2. Compute the numerical flux: compute the gradient-limited fluxes at cell faces $j+\frac{1}{2}$:
$$\begin{aligned} r^+_j &= (q_{j+2} - q_{j+1})/(q_{j+1} - q_j) && (3)\\ r^-_j &= (q_j - q_{j-1})/(q_{j+1} - q_j) && (4)\\ \phi_j &= 2 - \max\left\{0, \min\left\{1, 2r^+_j\right\}\right\} - \max\left\{0, \min\left\{1, 2r^-_j\right\}\right\} && (5)\\ \Delta^* t_j &= 0.5\left((1-\phi_j)\Delta t + \phi_j \Delta x / u^*_{j+\frac{1}{2}}\right) && (6)\\ f_{j+\frac{1}{2}} &= u^*_{j+\frac{1}{2}}\left(0.5(q_j + q_{j+1}) - \left(u^*_{j+\frac{1}{2}}\Delta^* t_j/\Delta x\right)(q_{j+1} - q_j)\right) && (7) \end{aligned}$$

3. Advect and advance in time:
$$q^{n+1}_j = q^n_j - \frac{\Delta t}{\Delta x}\left(f_{j+\frac{1}{2}} - f_{j-\frac{1}{2}}\right) \qquad \forall j. \tag{8}$$

4. Compute Δt: the global Δt should satisfy $|u^*_{j+\frac{1}{2}}|\Delta t/\Delta x \le 1$ for all j [Kas00, p. 29], hence we choose
$$\Delta t = \min_j \left\{ \frac{\mathrm{CFL}\cdot \Delta x}{|u^*_{j+\frac{1}{2}}| + \varepsilon} \right\}, \tag{9}$$
where $\varepsilon = 10^{-30}$ is a small number, and $0 < \mathrm{CFL} < 1$ is a prescribed desired Courant number.

1.4 Numerical Results

We define two scalar model problems,

(A) *Advection:* $f(q) = qu$, where $u = 1$ is a constant speed propagation (wave propagates to the right).

(B) *Burgers:* $f(q) = q^2/2$. Here the speed propagation is $f'(q) = q$. A shock may develop; sonic points occur when $q = 0$.

We test them for two initial conditions at $t = 0$:

(1) *Positive square wave:*

$$q(x) := \begin{cases} 0, & \text{if } 0 \le x < 0.3, \\ 1, & \text{if } 0.3 < x < 0.5, \\ 0, & \text{if } 0.5 < x \le 1. \end{cases} \tag{10}$$

(2) *Square wave:* identical to (1), except a shift by 0.5 so that q crosses zero at two points (initially at $x = 0.3, 0.5$).

$$q(x) := \begin{cases} -0.5, & \text{if } 0 \le x < 0.3, \\ 0.5, & \text{if } 0.3 < x < 0.5, \\ 0.5, & \text{if } 0.5 < x \le 1. \end{cases} \tag{11}$$

Sonic points occur only for the case (B2). The profile $q(x)$ after 100 timesteps for each of the four cases (A1),(A2),(B1),(B2) is shown in Fig. 2.

We discretize in space using $N = 100$ points over the interval $[0, 1]$, hence $\Delta x = 0.01$. We use a CFL number of 0.2.

The ICE algorithm performs well for cases (A1),(A2) and (B1). It exhibits poor results in (B2), due to the sonic points: in (2) with $f'(q_j)$, $f'(q_{j+1}) \approx 0$, $u^*_{j+\frac{1}{2}}$ may be close to zero, and when we divide by it in (6), we may get absurd values for $\Delta^* t$. A "sonic point fix" (adding artificial viscosity in these cases) is required, as indicated in [Dav87, pp. 8–9].

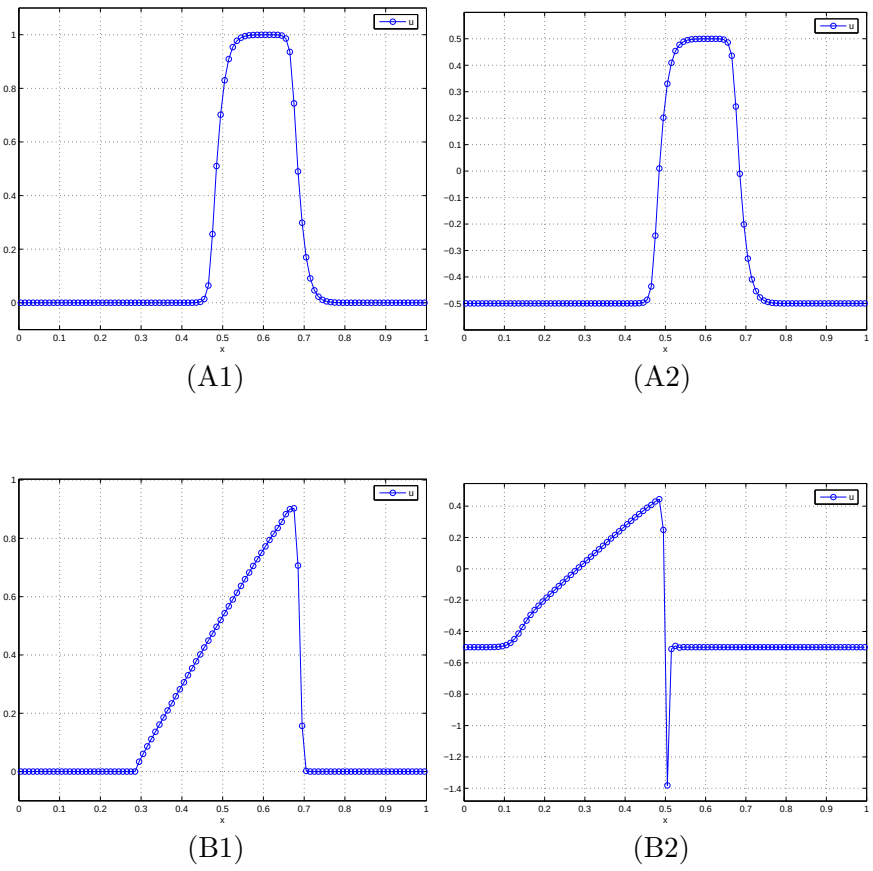

Figure 2: Results of Kashiwa ICE algorithm for scalar equations, (A1)–(B2), after 100 timesteps with 100 gridpoints and CFL of 0.2. Top line: constant advection; bottom line: Burger's equation. Left column: positive square wave initial data; left column: square wave initial data.

2 Uintah ICE Algorithm for Euler

We describe the ICE algorithm currently implemented in Uintah for the 1D compressible Euler equations.

2.1 The Equations

Because the Euler equations have many formulations, we specify the unknowns and the conservation equations they satisfy. The unknowns consist of a state vector $Q := (\rho, \rho u, \rho i)^T$ (ρ is the density, u is the velocity, i is the specific internal energy), and the pressure p. To set boundary conditions, we first translate these variables into (ρ, u, T) and p, set B.C. for those, and translates back to the state variables Q. This is a separate issue that will not be investigated here. The equations are

$$Q_t + C(Q)_x = -P(Q), Q := \begin{bmatrix} \rho \\ \rho u \\ \rho i \end{bmatrix}, C(Q) := \begin{bmatrix} u\rho \\ u\rho u \\ u\rho i \end{bmatrix}, S(Q) := \begin{bmatrix} 0 \\ p_x \\ pu_x \end{bmatrix}. \tag{12}$$

The Equation Of State (EOS) assumes an ideal gas model,

$$p = (\gamma - 1)\rho e \tag{13}$$

where $\gamma = 1.4$. In (12), C is the advection flux (convective term) and P represents sources due to pressure. Notice that P is not in flux form.

2.2 Discretization

We define all state vector at cell centers. Intermediate values inside the timestep are calculated at the faces for p and u. See Fig. 3.

We assume an infinite (or periodic) grid to avoid treating boundary conditions.

2.3 Advection Scheme

The ICE algorithm timestep for computing the quantities at time t_{n+1} (denoted $Q^{(n+1)}$) from quantities at time n (denoted Q, omitting the n-superscript) consists of the following steps:

1. Compute pressure: compute p_j from the EOS at all cell centers j,

$$p_j = (\gamma - 1)\rho_j i_j, \qquad \forall j. \tag{14}$$

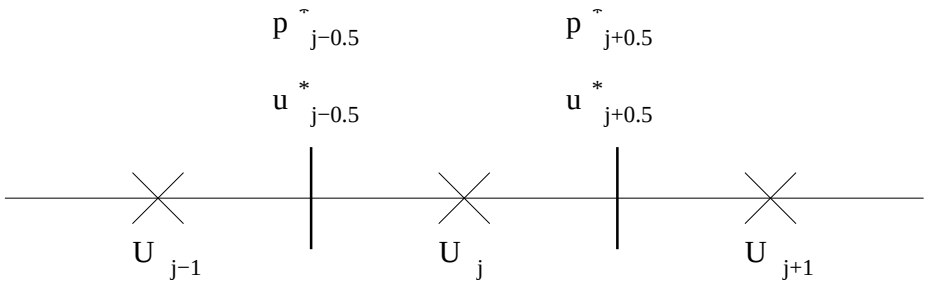

Figure 3: Cell-centered discretization of (12). The discrete state variables Q_j are defined at cell centers j ("x"). Cell faces $j+\frac{1}{2}$ are marked by "|".

Then compute the speed of sound,

$$c_j = \left(\sqrt{\frac{\partial p}{\partial \rho} + \frac{\partial p}{\partial i}\frac{p}{\rho^2}}\right)_j = \sqrt{(\gamma-1)\left(i_j + \frac{p_j}{\rho_j}\right)}, \qquad \forall j. \tag{15}$$

2. Compute the face-centered velocities: we denote them by $u^*_{j+\frac{1}{2}}$, at all faces $j+\frac{1}{2}$.

$$u^*_{j+\frac{1}{2}} = \frac{\rho_j u_j + \rho_{j+1}u_{j+1}}{\rho_j + \rho_{j+1}} - \Delta t \frac{1}{2}\left(\frac{1}{\rho_j} + \frac{1}{\rho_{j+1}}\right)\frac{p_{j+1}-p_j}{\Delta x}, \qquad \forall j. \tag{16}$$

3. Compute pressure correction and update pressure. Note that the correction is defined at cell centers.

$$(\Delta P)_j = -\Delta t c_j^2 \rho_j ADV(vol, u^*)_j, \qquad \forall j. \tag{17}$$

$$p_j = p_j + (\Delta P)_j, \qquad \forall j. \tag{18}$$

Here *vol* is the volume fraction of the material, defined at each cell center. In this setting, we have one material, so $vol \equiv 1$. The notation $ADV(a, u^*)$ is the discrete advection operator of the function $a = a(x)$ using the velocities u^* at the faces. It will be further explained later on.

4. Compute the face-centered pressure: denoted by $p^*_{j+\frac{1}{2}}$, and computed at all faces $j+\frac{1}{2}$ as averages of the cell-centered pressure.

$$p^*_{j+\frac{1}{2}} = \frac{\rho_{j+1}p_j + \rho_j p_{j+1}}{\rho_j + \rho_{j+1}}. \tag{19}$$

5. Compute Lagrangian quantities: these values seem to be the values of the cell-centered state variables, which are advected along the characteristics of the system.

$$\begin{aligned} \rho_j^L &= \rho_j && (20)\\ (\rho u)_j^L &= \rho_j u_j - \Delta t \frac{p^*_{j+\frac{1}{2}} - p^*_{j-\frac{1}{2}}}{\Delta x} && (21)\\ (\rho i)_j^L &= \rho_j i_j - \Delta t p_j ADV(vol, u^*)_j && (22) \end{aligned}$$

   for all cell centers j.

6. Advect and advance in time: we pass back from the Lagrangian coordinate system to the cell centers at time $n+1$.

$$\begin{aligned} \rho_j^{n+1} &= \rho_j^L - \Delta t ADV(\rho^L, u^*)_j && (23)\\ (\rho u)_j^{n+1} &= (\rho u)_j^L - \Delta t ADV((\rho u)^L, u^*)_j && (24)\\ (\rho i)_j^{n+1} &= (\rho i)_j^L - \Delta t ADV((\rho i)^L, u^*)_j. && (25)\\ & && (26) \end{aligned}$$

   for all cell centers j.

7. Compute Δt: the next timestep is computed similarly to (9). Based on a user-specified Courant number $0 < \mathrm{CFL} < 1$,

$$\Delta t = \min_j \left\{ \frac{\mathrm{CFL} \cdot \Delta x}{|u^*_{j+\frac{1}{2}} + c_j| + \varepsilon} \right\}, \tag{27}$$

   where $\varepsilon = 10^{-30}$ is a small number.

2.3.1 The ADV operator

This section is based on the Uintah code. It turns out that it contradicts the description in [Kas00]. As we will see, it is probable that the paper is wrong and the code is correct.

The advection operator $ADV(q, u^*)$ takes cell centered values $\{q_j\}_j$ of an advected quantity $q = q(x)$, and returns a correction defined at cell centers. When this correction is multiplied by Δt and subtracted, we obtain $\{q_j^{(n+1)}\}_j$. ADV requires the face-centered velocities $\{u^*_{j+\frac{1}{2}}\}_j$. $ADV(q, u^*)_j$ can be interpreted as discrete divergence operator centered at cell j.

- *Viewpoint 1: moving "slabs".* Geometrically, we can approximate the amount of volume advected in and out every cell during a Δt time interval; the volumes depend on Δx, Δt and the u^*'s, but *independent of* $\{q_j\}_j$. Then, we approximate the average of q on each slab to get the total in-flux and out-flux of q to/from cell j. The resulting formula can thus be applied to all state variables (substituted for q).

  Thus, ADV is given by

$$ADV(q,u^*)_j := -\frac{1}{\Delta x \Delta t}\left(q_{in}V_{in} - q_{out}V_{out}\right), \tag{28}$$

  where V_{in} is the slab volume advected into cell j, q_{in} is an average of q over this in-slab, V_{out} is the slab volume advected outside cell j, and q_{out} is the average of q over the out-slab.

  Fig. 4 illustrates the case where $u^*_{j-\frac{1}{2}} > 0$, $u^*_{j+\frac{1}{2}} > 0$. The in-slab is a volume of cell $j-1$ entering cell j, and the out-slab is a volume of cell j leaving cell j and entering cell $j+1$. Assuming that $u^*_{j+\frac{1}{2}}$ is the average velocity at the face $j+\frac{1}{2}$ over the time period $[t_n, t_{n+1}]$, $V_{in,j} = \Delta t u^*_{j-\frac{1}{2}}$ and $V_{ou,jt} = \Delta t u^*_{j-\frac{1}{2}}$; if $u^*_{j+\frac{1}{2}}$ is a velocity value at the face at $t_n + \frac{1}{2}\Delta t$, the slab volumes are exact to $O(\Delta t^2)$. The average of q over V_{in} is approximated to $O(\Delta x^2)$ by the value q^S_j of q in the middle of the slab.

  In general, the in-flux is a summation of the in-flux through face $j-\frac{1}{2}$ and $j+\frac{1}{2}$. If we define

$$u^+_{j+\frac{1}{2}} := \max\left\{u^*_{j+\frac{1}{2}}, 0\right\}, \qquad u^-_{j+\frac{1}{2}} := \min\left\{u^*_{j+\frac{1}{2}}, 0\right\} \tag{29}$$

  then

$$V_{in,j} = q^S_{j-1}\Delta t u^+_{j-\frac{1}{2}} - q^S_{j+1}\Delta t u^-_{j+\frac{1}{2}} \tag{30}$$

$$V_{out,j} = -q^S_j \Delta t u^-_{j-\frac{1}{2}} + q^S_j \Delta t u^+_{j+\frac{1}{2}}. \tag{31}$$

$$\tag{32}$$

  Substituting into (28) gives

$$ADV(q,u^*)_j = -\frac{1}{\Delta x}\left(q^S_{j-1}u^+_{j-\frac{1}{2}} - q^S_{j+1}u^-_{j+\frac{1}{2}} + q^S_j u^-_{j-\frac{1}{2}} - q^S_j u^+_{j+\frac{1}{2}}\right). \tag{33}$$

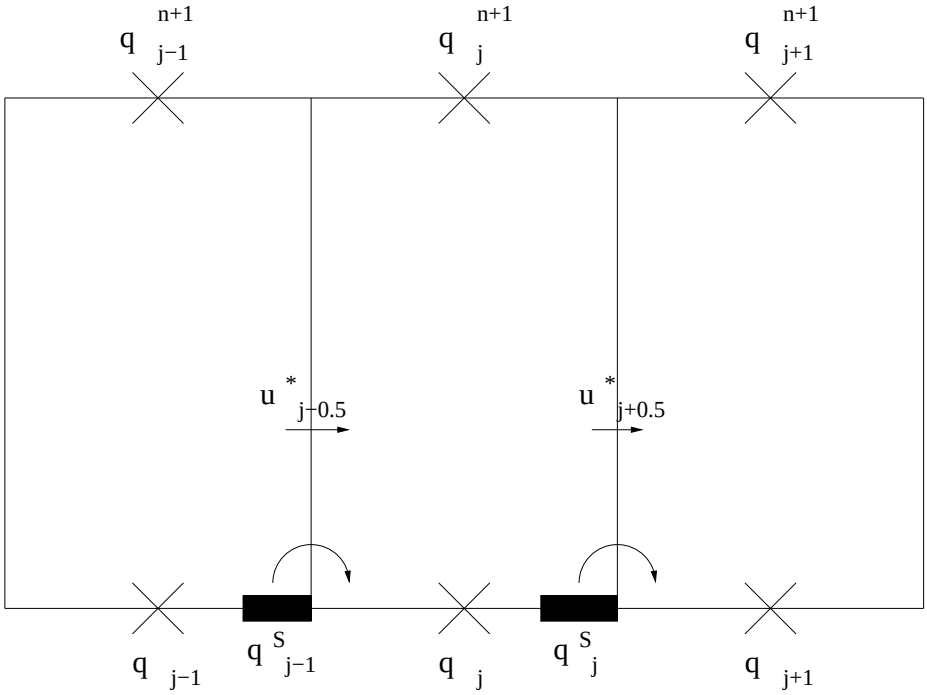

Figure 4: The in-slab and out-slab of cell j for the case of $u^*_{j-\frac{1}{2}} > 0$, $u^*_{j+\frac{1}{2}} > 0$.

- *Viewpoint 2: conservation form.* As in (8), we can rearrange (33) in terms of face $j-\frac{1}{2}$, $j+\frac{1}{2}$ contributions rather than "in" and "out" contributions. Namely, if

$$ADV(q,u^*) = \frac{1}{\Delta x}\left(C^*_{j+\frac{1}{2}} - C^*_{j-\frac{1}{2}}\right), \tag{34}$$

  where the numerical flux is

$$C^*_{j+\frac{1}{2}} := q^S_j u^+_{j+\frac{1}{2}} + q^S_{j+1} u^-_{j+\frac{1}{2}}. \tag{35}$$

Note that when $q \equiv 1$, $C^*_{j+\frac{1}{2}} = u^+_{j+\frac{1}{2}} + u^-_{j+\frac{1}{2}} = u^*_{j+\frac{1}{2}}$, and ADV becomes the discrete divergence operator.

The advantage of viewpoint 1 is that the same ADV operator (except for the limiter) applies to all state variables in the Euler equations, as they are all advected with the same velocity u^*. The only quantities that need to be computed for each new variable are the slab averages $\{q^S_j\}_j$.

The advantage of viewpoint 2 is that it allows a direct relation to schemes based on Riemann solvers. (35) is a special case of a flux-vector splitting [Lev02, p. 83, (4.56)].

We now describe the choice of q^S_j.

- *First order.* Here

$$q^S_j := q_j. \tag{36}$$

  Thus (35) is an upwind flux [Lev02, p. 75, (4.33)]. More generally, it can be viewed as a Godunov method, where q^S_j represents the value solution $\hat{q}$ of the Riemann problem with piecewise constant initial data q_{j-1}, q_j on the left and the right of the face $j+\frac{1}{2}$, evaluated at mid-time (i.e., $q^S_j = \hat{q}((j+\frac{1}{2})\Delta x, (n+\frac{1}{2})\Delta t)$).

- *Second order.* Here we use

$$q^S_j := q_j + r_j \frac{q_{j+1} - q_{j-1}}{2\Delta x}, \tag{37}$$

  where $r_j := |\Delta x/2 - u^*_{j+\frac{1}{2}}\Delta t/2|$ is the "centroid" vector, that is, the vector pointing from the cell center to the slab center. Following the derivation of [KV98] and [Lev02, §6.4], we see that (35) is equivalent to a Godunov method based on piecewise linear reconstruction of q. (37) is a second-order approximation to the value of q at the center of the slab.

- *Limited.* This combines the first and second order methods using

$$q_j^S := q_j + r_j \phi_j \frac{q_{j+1} - q_{j-1}}{2\Delta x}, \tag{38}$$

ϕ_j is either a van-Leer gradient limiter, when advecting volume fraction and density, or a compatible flux limiter [KV98], for all other advected quantities.

2.4 Scheme Analysis

We reformulate ICE in terms of the state variables Q only. The target is to find the operator transforming Q^n into Q^{n+1}. The ICE timestep can be rewritten as follows.

1. Advance pressure to time n using the EOS (p^n).
2. Compute half-space, [locally] time advanced quantities u^*, p^*. Let

$$A(q)_{j+\frac{1}{2}} := \frac{1}{2}(q_{j+1} + q_j) \tag{39}$$

$$D(q)_{j+\frac{1}{2}} := \frac{1}{2}(q_{j+1} - q_j); \tag{40}$$

for all j; then in vector form,

$$u^* = \frac{A(\rho u)}{A(\rho)} - \Delta t A\left(\frac{1}{\rho}\right) D(p^n) \tag{41}$$

$$\Delta p = -\Delta t c^2 \rho \left[D\left(\frac{A(\rho u)}{A(\rho)}\right) - \Delta t D\left(A\left(\frac{1}{\rho}\right) D(p^n)\right)\right] \tag{42}$$

$$p^{n+1} = p^n - \Delta t c^2 \rho D\left(\frac{A(\rho u)}{A(\rho)}\right) + \Delta t^2 c^2 \rho D\left(A\left(\frac{1}{\rho}\right) D(p^n)\right) \tag{43}$$

$$p^* = \frac{1}{A\left(\frac{1}{\rho}\right)}\left[A\left(\frac{p^n}{\rho}\right) - \Delta t c^2 D\left(\frac{A(\rho u)}{A(\rho)}\right) + \Delta t^2 c^2 D\left(A\left(\frac{1}{\rho}\right) D(p^n)\right)\right] \tag{44}$$

3. *Lagrangian phase:* Advance in time based on the sources,

$$Q^L = Q^n - \Delta t LAG(Q^n, u^*, p^*) \tag{45}$$

4. *Eulerian phase:* advect forward in time. The advection operator is based on the already-computed Lagrangian values.

$$Q^{n+1} = Q^L - \Delta t ADV(Q^L, u^*) \tag{46}$$

The Lagrangian operator advances Q using the equation $Q_t = -S(Q)$ (without advection). Discretizing S (see (12)) at cell centers (using differencing of the face-centered u^*, p^*, we define

$$LAG(Q)_j := \begin{bmatrix} 0 \\ \frac{p^*_{j+\frac{1}{2}} - p^*_{j-\frac{1}{2}}}{\Delta x} \\ p^{n+1}_j \left(u^*_{j+\frac{1}{2}} - u^*_{j-\frac{1}{2}} \right)_j \end{bmatrix}. \tag{47}$$

Note that LAG contains two pressures: face-centered p^* and cell-centered p^{n+1}. Both are time-advanced. In ICE we are not really conforming to Bucky's page-29 scheme for p^*, rather, we average time-$n+1$ pressure values (equivalent to $\Delta^*_p t = \Delta t$ and taking the average over the entire right-hand-side for the p^* equation on page 29).

2.5 Numerical Results

Our initial data is the shock tube piecewise constant data depicted in Fig. 5.

We tested the ICE algorithm using CFL $= 0.45$ and an initial $\Delta t = 10^{-6}$ for the first timestep (subsequent timesteps are computed using (27)). We discretize in space using $N = 100$ points over the interval $[0, 1]$, hence $\Delta x = 0.01$. The state variable profiles after 100 timesteps are depicted in Fig. 6. We tested the same algorithm with first order advection (36). The results are depicted in Fig. 7.

Note that

- All state variables have oscillations near the shocks, for first order advection. I cannot explain this result - shouldn't the first order advection operator ADV be monotone (even though it's a system of equations, not a scalar conservation law)?
- First order advection is much more diffusive, which is to be expected.
- The oscillation appear also for the second order limited case, thus we might be led to thinking that the limiter is not working properly. But even if it would reduce the advection order to first near shocks, as we already saw, oscillations still occur. So the limiter might not be the problem, although we do not know for sure before we resolve the first order oscillations issue.

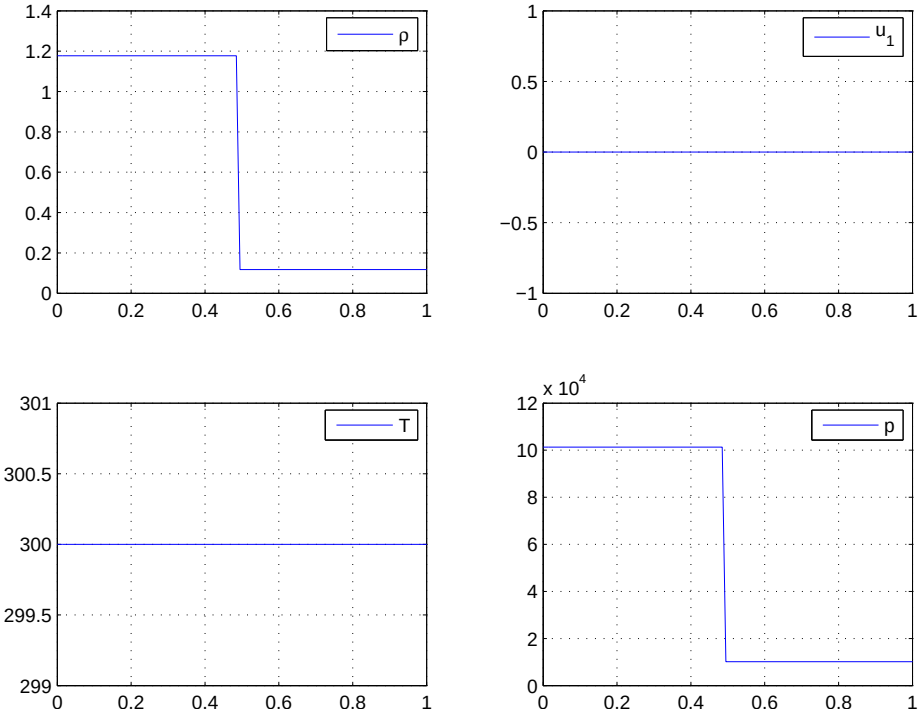

Figure 5: Initial data for the compressible Euler equations (shock tube problem).

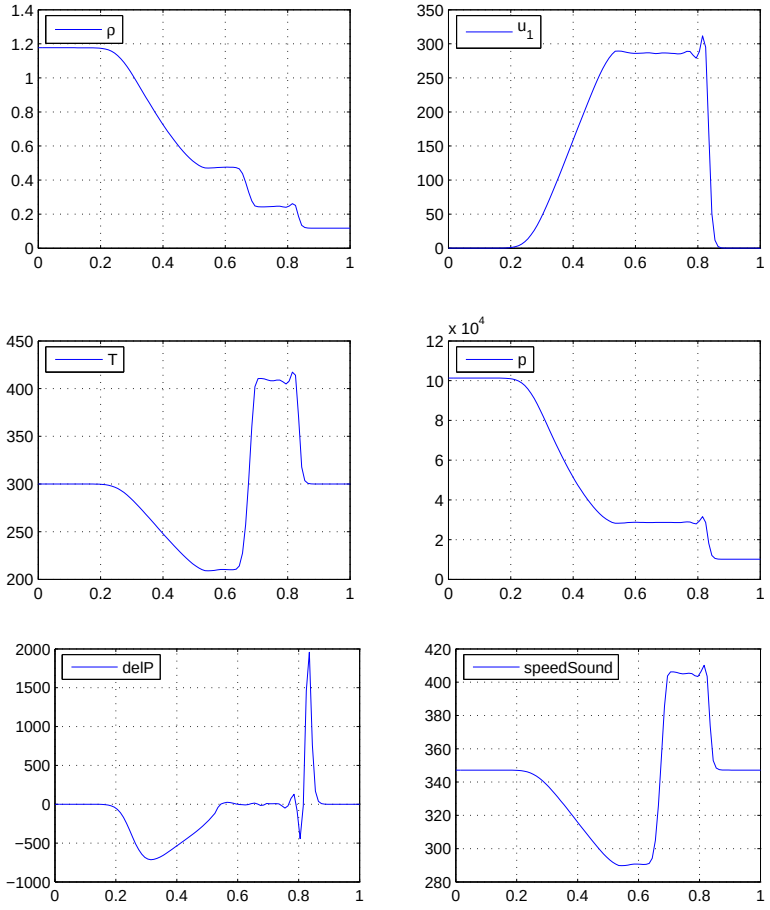

Figure 6: Results of Kashiwa ICE algorithm for the shock tube problem with a limiter, after 100 timesteps with 100 gridpoints and CFL of 0.45. The top two lines show $\rho, u_1 \equiv u, T$ and p vs. x. The last line shows the internal variables ΔP and the speed of sound c vs. x.

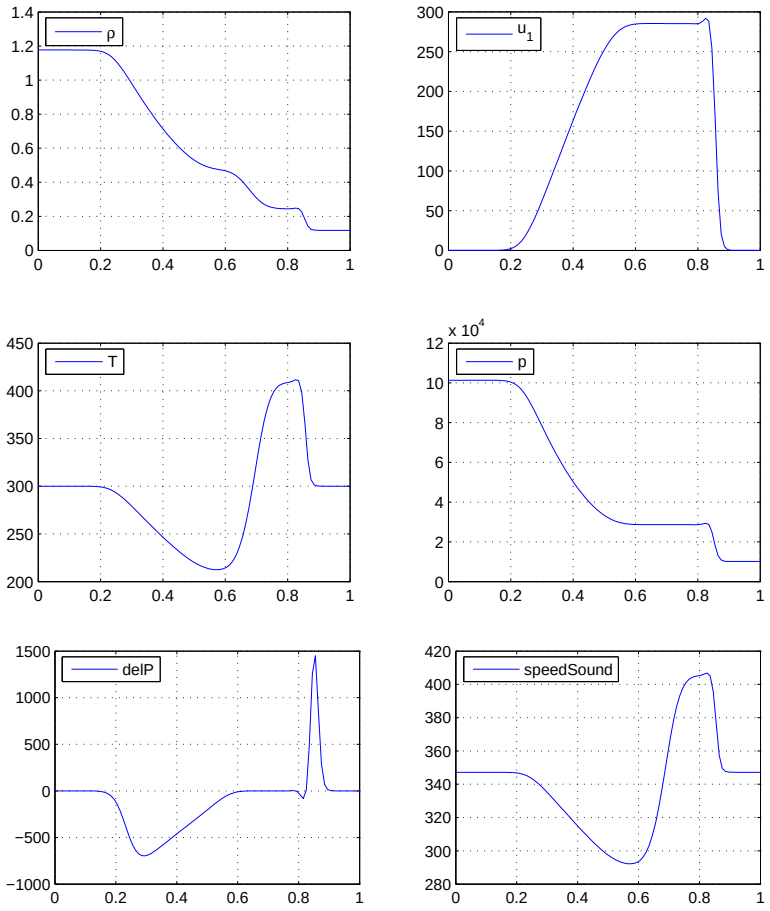

Figure 7: The same as Fig. 6, for first order advection in ADV.

3 Conservative ICE Algorithm for Euler

3.1 The Equations

The difference between the algorithm in this section and the ICE algorithm in §2 is the formulation of the energy equation. We use the conservative form of the Euler system, replacing i (specific internal energy) by e (specific energy), where $e = i + \frac{1}{2}u^2$. We do not deal here with the implications of this change on the boundary conditions. Working in this form allows us to compare ICE to the very similar Zha-Bilgen flux vector splitting described in [ZB93].

The equations are

$$Q_t + F(Q)_x = 0, \qquad Q := \begin{bmatrix} \rho \\ \rho u \\ \rho e \end{bmatrix}, \qquad F(Q) := \begin{bmatrix} u\rho \\ \rho u^2 + p \\ (\rho e + p)u \end{bmatrix}, \tag{48}$$

The Equation Of State (EOS) assumes an ideal gas model,

$$p = (\gamma - 1)\rho\left(e - \frac{1}{2}u^2\right) \tag{49}$$

where $\gamma = 1.4$. The eigenvalues of the Jacobian $\partial F/\partial Q$ has three real eigenvalues, $u, u + c, u - c$, where $c = \sqrt{\gamma p/\rho}$ is the speed of sound. We split F as in [ZB93, eq. (7)],

$$F = C + P, \qquad C := u \begin{bmatrix} \rho \\ \rho u \\ \rho e \end{bmatrix}, \qquad P := \begin{bmatrix} 0 \\ p \\ pu \end{bmatrix}. \tag{50}$$

In (50), F is total flux, C is the convective flux and P is the pressure term, now in flux form (compare with (12)). The eigenvalues of C are u, u, u and P's eigenvalues are $0, c, -c$.

3.2 Discretization

Q and p are defined at cell centers. We will again use u^* and p^* at the face centers. We assume an infinite (or periodic) grid to avoid treating boundary conditions.

3.3 Advection Scheme

In this case we can write ICE in discrete conservative form. ICE's operator splitting consists of the stages

$$\begin{aligned} p^n &= p(Q)) \quad \text{(Evaluate EOS)} && (51)\\ &\text{Compute } u^*, p^{n+1}, p^* && (52)\\ Q^L &= Q^n - \Delta t LAG(Q^n, u^*, p^*) \quad \text{(Lagrangian phase)} && (53)\\ Q^{n+1} &= Q^L - \Delta t ADV(Q^L, u^*) \quad \text{(Eulerian phase)} && (54) \end{aligned}$$

u^*, p^{n+1}, p^* are computed as in the Uintah ICE algorithm. The Lagrangian phase is now cast in terms of $(\rho, \rho u, \rho e)$ rather than $(\rho, \rho u, \rho i)$. Instead of (21)–(22) we have

$$\begin{aligned} \rho_j^L &= \rho_j && (55)\\ (\rho u)_j^L &= \rho_j u_j - \Delta t \frac{p^*_{j+\frac{1}{2}} - p^*_{j-\frac{1}{2}}}{\Delta x} && (56)\\ (\rho e)_j^L &= \rho_j e_j - \Delta t ADV(p^{(n+1)}, u^*)_j. && (57) \end{aligned}$$

Thus, the Lagrangian phase now has the numerical flux form

$$\begin{aligned} LAG(Q, u^*) &= \frac{1}{\Delta x}\left(P^*_{j+\frac{1}{2}} - P^*_{j-\frac{1}{2}}\right), && (58)\\ P^*_{j+\frac{1}{2}} &:= \begin{bmatrix} 0 \\ p^*_{j+\frac{1}{2}} \\ u^+_{j+\frac{1}{2}} p^{n+1,S} + u^-_{j+\frac{1}{2}} p^{n+1,S} \end{bmatrix}. && (59) \end{aligned}$$

The convective numerical flux is given as before (see (35)) by

$$\begin{aligned} ADV(Q, u^*) &= \frac{1}{\Delta x}\left(C^*_{j+\frac{1}{2}} - C^*_{j-\frac{1}{2}}\right), && (60)\\ C^*_{j+\frac{1}{2}} &:= u^+_{j+\frac{1}{2}} Q^S_j + u^-_{j+\frac{1}{2}} Q^S_{j+1}. && (61) \end{aligned}$$

Thus, the total numerical flux of the scheme is

$$\begin{aligned} F^*_{j+\frac{1}{2}} &= C^*_{j+\frac{1}{2}} + P^*_{j+\frac{1}{2}} \\ &= u^+_{j+\frac{1}{2}} \begin{bmatrix} \rho^S_j \\ (\rho u)^S_j \\ (\rho e)^S_j \end{bmatrix} + u^-_{j+\frac{1}{2}} \begin{bmatrix} \rho^S_{j+1} \\ (\rho u)^S_{j+1} \\ (\rho e)^S_{j+1} \end{bmatrix} \\ &\quad + \begin{bmatrix} 0 \\ p^*_{j+\frac{1}{2}} \\ u^+_{j+\frac{1}{2}} p^{n+1,S} + u^-_{j+\frac{1}{2}} p^{n+1,S} \end{bmatrix}. && (62) \end{aligned}$$

3.4 Numerical Results

Our initial data is the shock tube piecewise constant data depicted in Fig. 5. We tested the ICE algorithm using CFL $= 0.45$ and an initial $\Delta t = 10^{-6}$ for the first timestep (subsequent timesteps are computed using (27)). We discretize in space using $N = 100$ points over the interval $[0, 1]$, hence $\Delta x = 0.01$. The state variable profiles after 100 timesteps are depicted in Fig. 6. We tested first order advection only. The results are depicted in Fig. 8. These are initial results only. The same wiggles occur in here as in Uintah ICE. However, the velocity profile seems to be wrong, unlike the Uintah results.

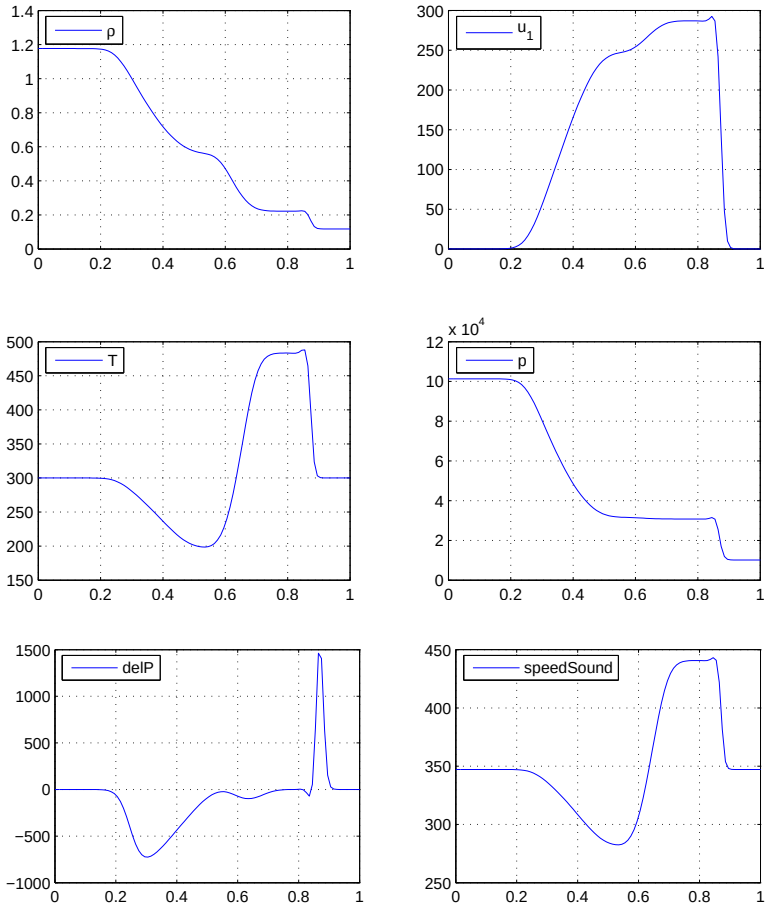

Figure 8: The same as Fig. 6, for first order advection in *ADV*.

4 Concluding Remarks and Questions for Bucky

- The scalar ICE algorithm works, except near sonic points, where we have to add more diffusion. This action item can be treated separately of the rest of the "machinery".
- First order advection in ICE produces oscillations. Can we show that it must be monotone? If not, what is the point of using a limiter to reduce the advection order from second to first in non-smooth regions, if we cannot anyway obtain a monotone profile using first order?
- The precise role of u^* and p^* from [Kas00, p. 29] in the ICE advection scheme is still not part of the description of this report and should be added as soon as possible as it might be the key to the questions above.
- Should p^* be based on old pressures (time n) or new pressure averages (time $n+1$)? Bucky has an old pressure average when $\Delta^* p_t = 0$ pn page 29, and again uses $p^L = p$ for explicit timestepping on pages $41-42$, which are averaged to give p^*.
- Can the entire ICE algorithm be written in conservation form (i.e., a grand numerical flux combining both the Lagrangian and Eulerian phases)?

References

[Dav87] S. F. Davis. A simplified TVD finite difference scheme via artificial viscosity. *SIAM J. Sci. Stat. Comput.*, 8:1–18, 1987.

[Kas00] B. A. Kashiwa. A multifield model and method for fluid-structure interaction dynamics. Technical report, Los Alamos National Laboratory, Los Alamos, NM, August 2000. Submitted to J. Comp. Phys.

[KV98] B. A. Kashiwa and W. B. VanderHeyden. Compatible fluxes for van leer advection. *J. Comp. Phys.*, 146:1–28, 1998.

[Lev02] R. J. Leveque. *Finite volume methods for hyperbolic problems*. Cambridge texts in applied mathematics. Cambridge university press, Cambridge, 2002.

[Liv05] O. Livne. Ice algorithm for the shocktube problem status report 3: Davis advection scheme. Technical report, University of Utah, Salt Lake City, UT, February 23, 2005.

[ZB93] G. C. Zha and E. Bilgen. Numerical solutions of Euler equations by using a new flux vector splitting scheme. *Int. J. Num. Methods in Fluids*, 17:115–144, 1993.