HW#6 Solution: Getting Started with GIZMO

In this problem set you will get up to speed with the public, multi-physics parallel astrophysics code GIZMO. You can checkout the code here: `https://bitbucket.org/phopkins/gizmo-public/src/master/`). The documentation is here: `http://www.tapir.caltech.edu/~phopkins/Site/GIZMO_files/gizmo_documentation.html`.

1. **Hello GIZMO :)**

Familiarize yourself with the GIZMO documentation and relevant code method papers (e.g., Hopkins 2015, Hopkins & Raives 2016). Follow the documentation to download and compile GIZMO on Stampede2 (see §6 and §12 "General Super-Computing Questions" in the documentation. I have created a summary document for getting GIZMO running on Stampede2 (but see the TACC and GIZMO documentation for complete details.)

2. **Testing, testing, 1, 2, 3...**

The fastest way to get up to speed with a code is to run already existing problem setups, i.e., test problems. It is also generally a good idea to run test problems to make sure the code is behaving as expected. Run the following test problems and make plots of the results (see documentation, the initial condition files are located here: ***`http://www.tapir.caltech.edu/~phopkins/sims/`****). If the problem has a given analytic or expected numerical result, comment on how the code performs. Record how many processors you used for the test.*

For each test, make a plot of the gas density versus position at a few different times (see notes on reading and plotting snapshots below).

a) **Sod Shock Tube.** *See left panels in Fig 10 in Hopkins 2015. Compare against the expected analytic solution (the predicted solution is available as a text file). Discuss any features you see in the result.*

The following plot shows the Shock Tube test problem solution run with the MFM solver. It compares reasonably well with the analytic result, but has smoothing due to dissipation across all of the density jumps. This is due to the Riemann solver. It also has an overshoot (or 'ringing') at the edge of the rarefaction wave. The effective resolution across this range is about 100 cells, which is comparable to our previous shock problem setup. Note that I have shortened the domain to focus on *one* of the shock tube solutions. The periodic boundary conditions for the problem and the discontinuous nature of the initial conditions means that two shocks will form, which is why the test solution in the output looks more complex than expected.

b) **Optional: MFM vs. MFV.** *Run the Sod shock tube in both MFM and MFV modes and*

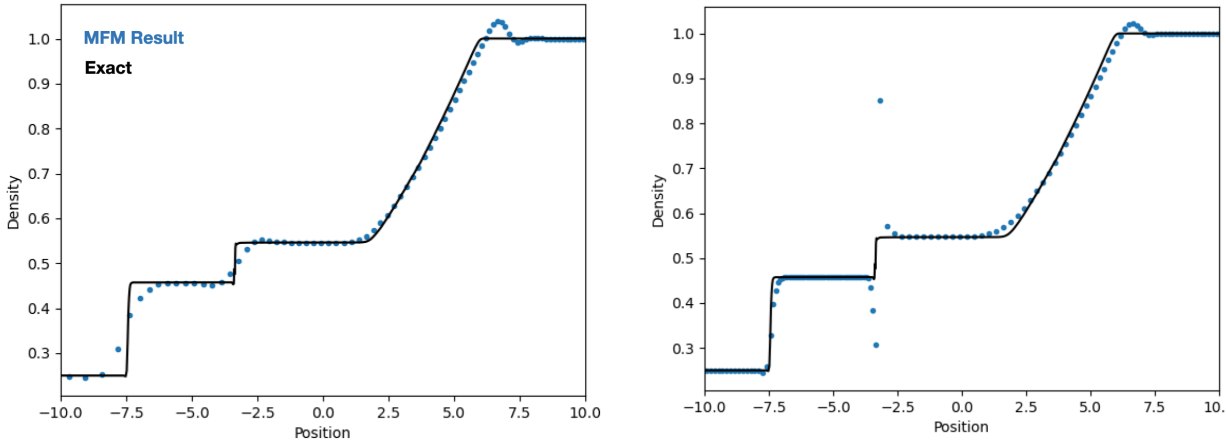

Figure 1: Left: Shock tube (blue points) with analytic solution (black solid line) for MFM solver. Right: Shock tube (blue points) with analytic solution (black solid line) for the MFV solver.

compare the two solutions. Which do you think is better and why, i.e., how do the artifacts in the solution differ?

If you also ran the MFV test setup with MFV solver you will have noticed that the solution looks a little worse, i.e., there is a second extra overshoot at the contact discontinuity (e.g., see Fig. 1b and Fig. 2 below). The MFV solver result also appears to have less numerical viscosity across the shock – it looks pretty sharp.

3. Sod it! Not again!

Using Make_IC.py in the GIZMO `scripts` *directory as a guide (or another of the setup options) generate initial conditions for the Sod shock tube problem we did in HW2. You may either use GIZMO's MFM solver – in which case make the cell mass uniform in your setup – or the MFV solver — make the cell volume uniform (constant dx spacing).*

Run the problem with GIZMO and compare with your result from HW2.[1]

The following plot shows the HW2 Sod Shock Tube test problem solution run with the MFV solver. I chose to set up the problem with cells of constant volume, i.e., initially equally spaced cells with a length of $dx = 2/200$ (exactly how we set up the cells for HW2).

We see the GIZMO result compares reasonably well with the HW2 result, but it has a bit more smoothing due to dissipation across all of the density jumps (e.g., the discontinuities are

[1] If your HW2 implementation was not fully working, you can use my code – see solutions in `https://github.com/soffner/ComputationalAstrophysics`. Note that because of the periodic boundary condition you may need to add 'filler' like the example shock tube, since your solution with a periodic boundary will have 2 shock fronts.

spread over more particles). Also notice that the HW2 solution doesn't have the noticeable overshoots at the rarefaction wave and contact discontinuity. The effective resolution across this range is comparable, so it is perhaps surprising that our simple HW2 solver does so well! Note that the periodic boundary conditions for the problem and the discontinuous nature of the initial conditions means that a second shock occurs at the boundary, which is why the GIZMO solution departs from $\rho = 1$ at $x = [0, 0.4]$. This could be fixed by extending the domain such that the GIZMO problem spanned more than [0,2], thereby moving the boundary away from the x range of our HW2 problem.

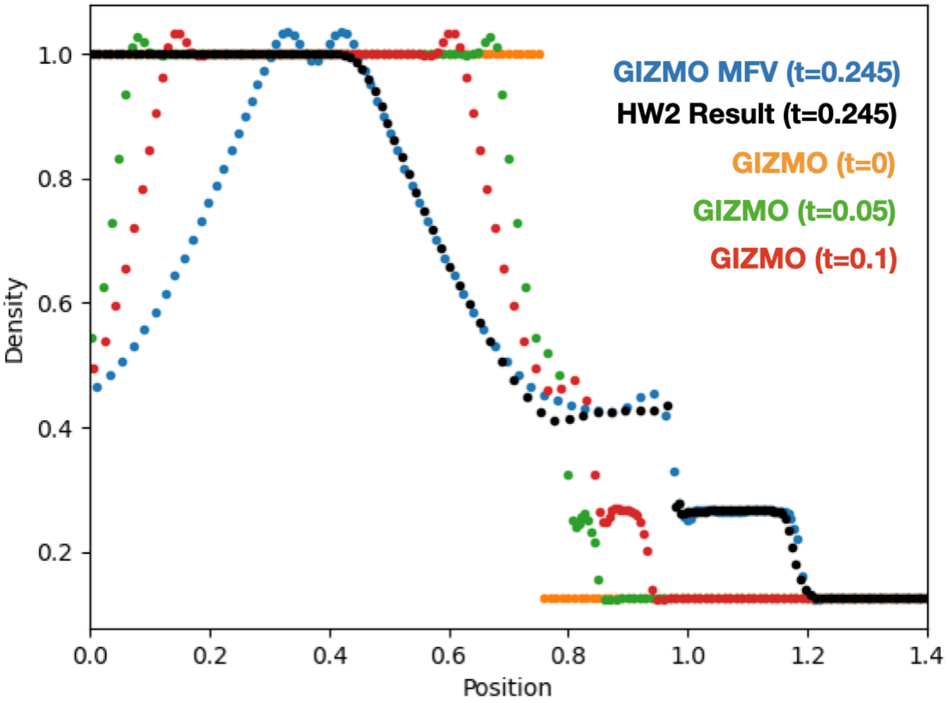

Figure 2: Shock tube from HW2 run in MFV mode. The colored points show different times for the GIZMO solution. The black dots show the HW2 solver result with 200 points.