UNIVERSITY OF CAMBRIDGE

Ghost fluid methods (part 1)

Stephen Millmore

Laboratory for Scientific Computing

Outline

- Principles of the ghost fluid method
- The original ghost fluid method
- History of ghost fluid methods
- Implementation of a ghost fluid method

Outline

- Principles of the ghost fluid method
- The original ghost fluid method
- History of ghost fluid methods
- Implementation of a ghost fluid method

Ghost fluid methods

- Ghost fluid methods are a class of numerical technique for providing boundary conditions at interfaces between materials, at which the interface is:

    a) Sharp

    b) Implicitly represented

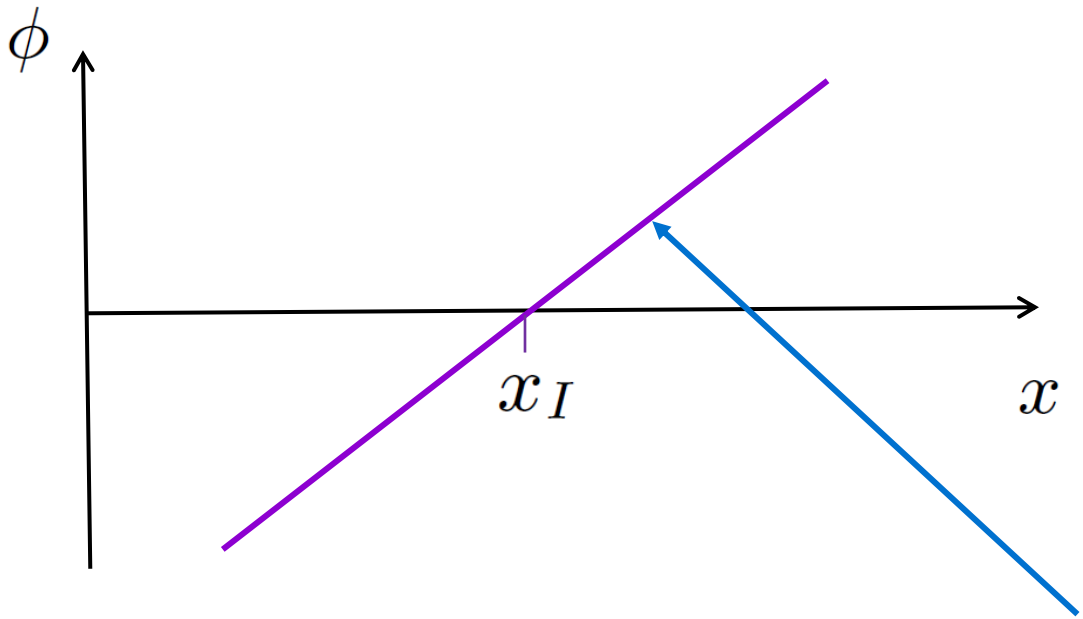
- Therefore, they typically use a level set function to define the interface, though volume-of-fluid techniques exist too
- Boundary conditions are needed because level set functions are strictly passive; they do not influence the flow at all
- As we saw before, naive assumptions as to how to model the interface can lead to non-physical features and oscillations

Dynamic boundary conditions

- Ghost fluid methods use the fact that level set methods tell us which cells are directly adjacent to a material boundary
- This means we can apply boundary conditions here, in much the same way as we do for cells at the edge of the computational domain
- Because both the boundary itself can move, and the conditions are dependent on the materials either side of the boundary, which constantly evolve, they are referred to as **dynamic boundary conditions**
- Because these boundary conditions do depend on two materials, the challenge was to find a **thermodynamically consistent** technique to achieve this

Ghost fluid method concepts

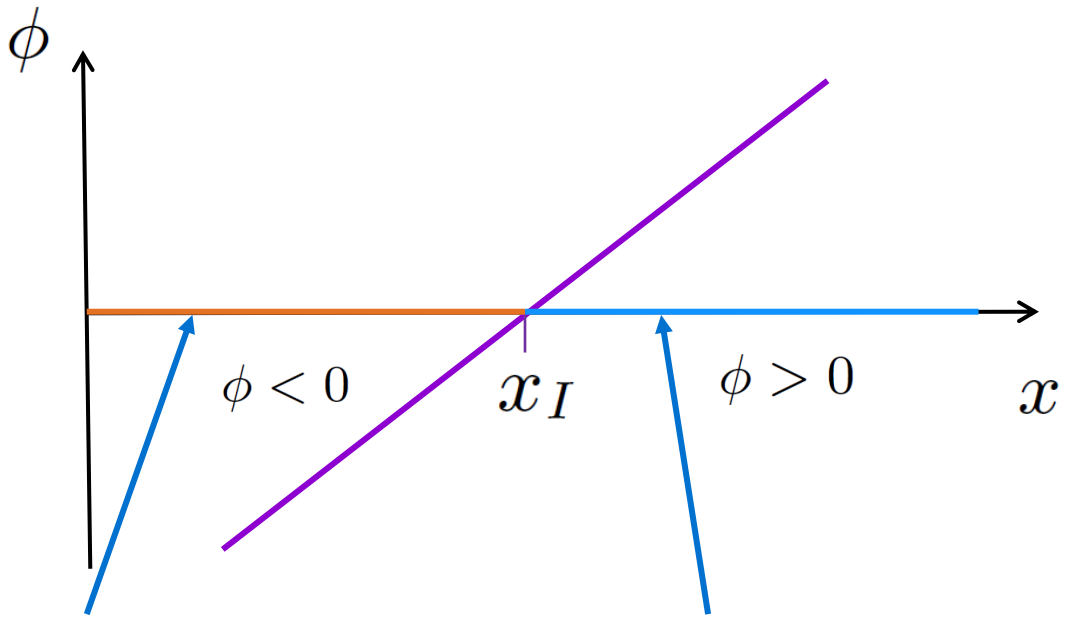
- We shall consider a set up a level set function divides the domain up into two regions



$$\frac{\partial \mathbf{u}_1}{\partial t} + \nabla \cdot \mathbf{f}(\mathbf{u}_1) = 0, \qquad \frac{\partial \mathbf{u}_2}{\partial t} + \nabla \cdot \mathbf{f}(\mathbf{u}_2) = 0 \qquad \frac{\partial \phi}{\partial t} + \mathbf{v} \cdot \nabla \phi = 0$$

Ghost fluid method concepts

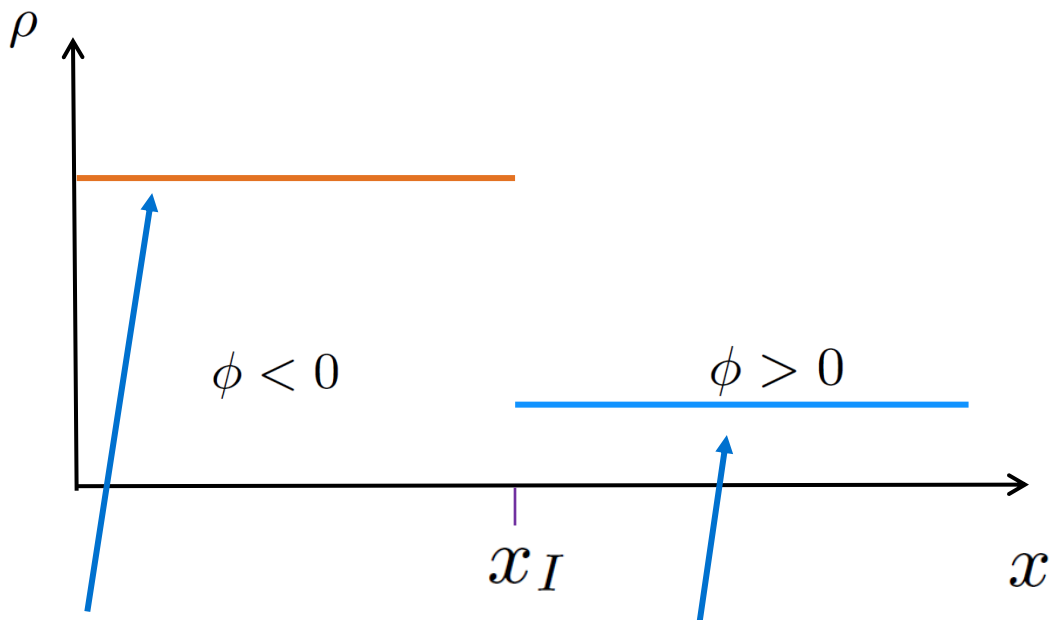
- Depending on the sign of the level set function, we know which material model we have



$$\frac{\partial \mathbf{u}_1}{\partial t} + \nabla \cdot \mathbf{f}(\mathbf{u}_1) = 0, \qquad \frac{\partial \mathbf{u}_2}{\partial t} + \nabla \cdot \mathbf{f}(\mathbf{u}_2) = 0 \qquad \frac{\partial \phi}{\partial t} + \mathbf{v} \cdot \nabla \phi = 0$$

Ghost fluid method concepts

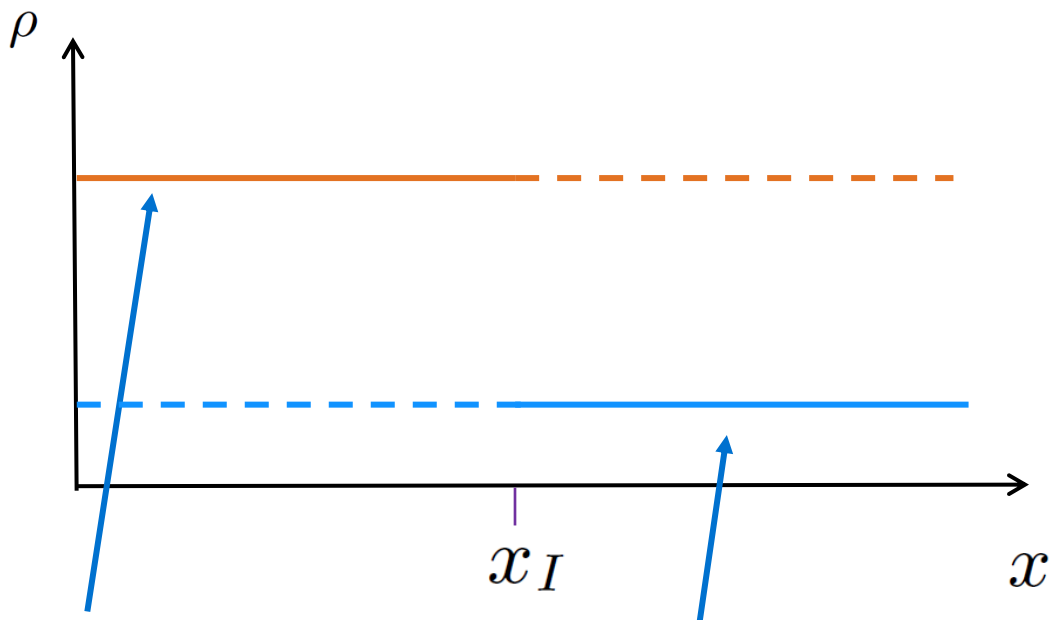
- The two materials then exist in the ‘physical model’ over these two regions



$$\frac{\partial \mathbf{u}_1}{\partial t} + \nabla \cdot \mathbf{f}(\mathbf{u}_1) = 0, \qquad \frac{\partial \mathbf{u}_2}{\partial t} + \nabla \cdot \mathbf{f}(\mathbf{u}_2) = 0 \qquad \frac{\partial \phi}{\partial t} + \mathbf{v} \cdot \nabla \phi = 0$$

Ghost fluid method concepts

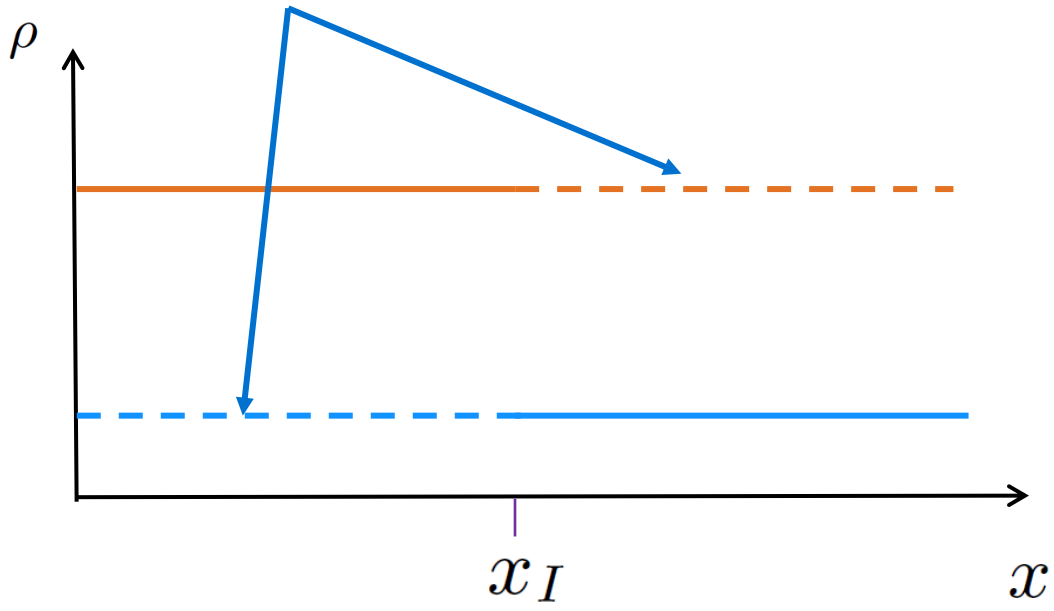
- However, because the materials are evolved completely separately, their data structure exists over the entire computational domain



$$\frac{\partial \mathbf{u}_1}{\partial t} + \nabla \cdot \mathbf{f}(\mathbf{u}_1) = 0, \qquad \frac{\partial \mathbf{u}_2}{\partial t} + \nabla \cdot \mathbf{f}(\mathbf{u}_2) = 0 \qquad \frac{\partial \phi}{\partial t} + \mathbf{v} \cdot \nabla \phi = 0$$

Ghost fluid method concepts

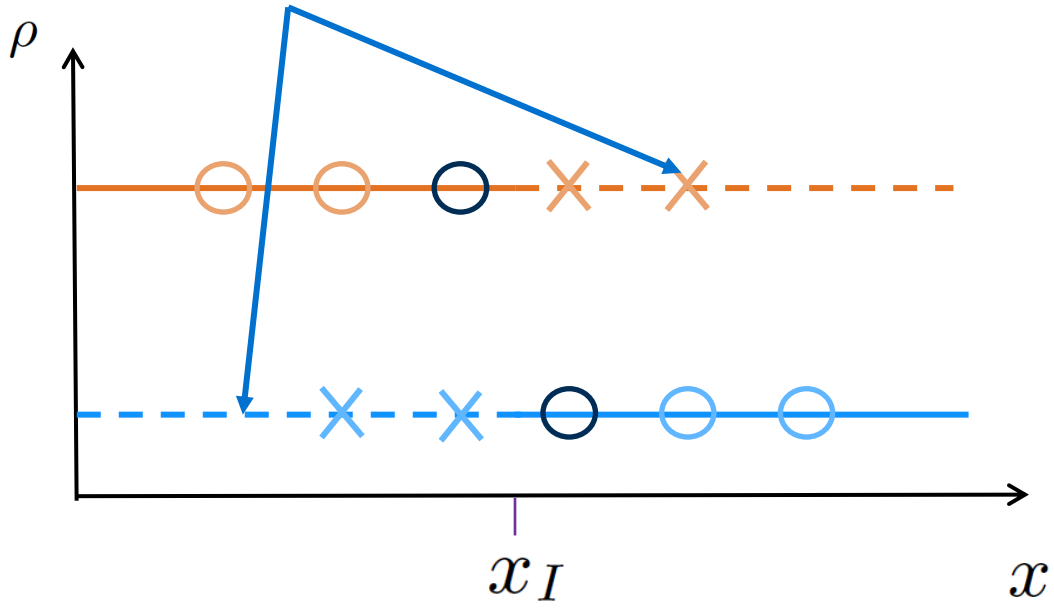
- The material that exists in the computational domain, but not the physical domain is the **ghost fluid**



$$\frac{\partial \mathbf{u}_1}{\partial t} + \nabla \cdot \mathbf{f}(\mathbf{u}_1) = 0, \qquad \frac{\partial \mathbf{u}_2}{\partial t} + \nabla \cdot \mathbf{f}(\mathbf{u}_2) = 0 \qquad \frac{\partial \phi}{\partial t} + \mathbf{v} \cdot \nabla \phi = 0$$

Ghost fluid method concepts

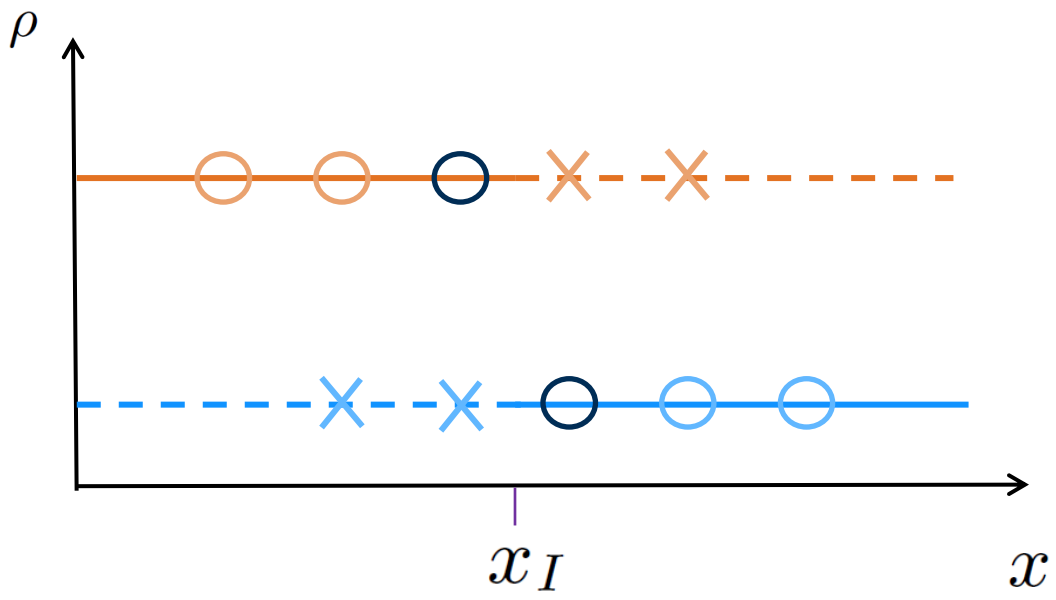
- Although the ghost fluid regions are not physical material, they will be used in material update due to the numerical stencil of the method used



$$\frac{\partial \mathbf{u}_1}{\partial t} + \nabla \cdot \mathbf{f}(\mathbf{u}_1) = 0, \qquad \frac{\partial \mathbf{u}_2}{\partial t} + \nabla \cdot \mathbf{f}(\mathbf{u}_2) = 0 \qquad \frac{\partial \phi}{\partial t} + \mathbf{v} \cdot \nabla \phi = 0$$

Ghost fluid method concepts

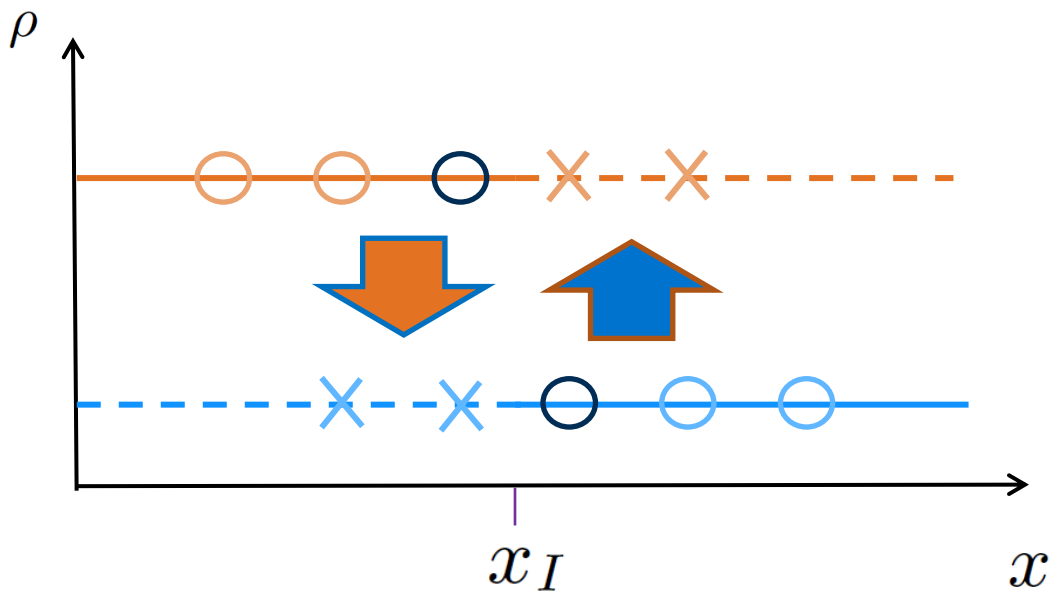
- The name ghost fluid comes from the concept of ghost zones – if the boundary is fixed, and not another material, we apply standard boundary conditions



$$\frac{\partial \mathbf{u}_1}{\partial t} + \nabla \cdot \mathbf{f}(\mathbf{u}_1) = 0, \qquad \frac{\partial \mathbf{u}_2}{\partial t} + \nabla \cdot \mathbf{f}(\mathbf{u}_2) = 0 \qquad \frac{\partial \phi}{\partial t} + \mathbf{v} \cdot \nabla \phi = 0$$

Ghost fluid method concepts

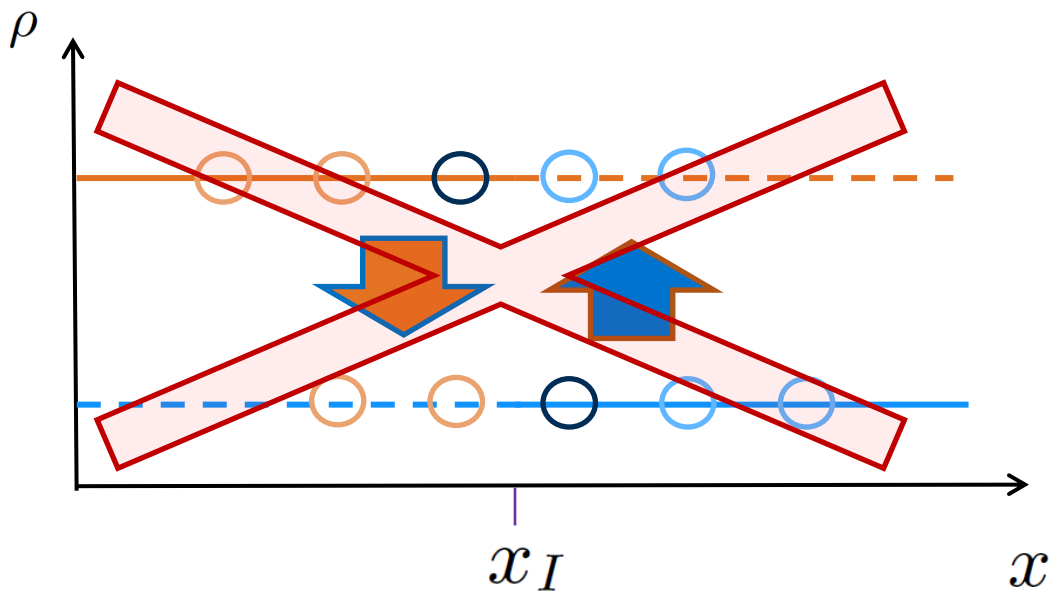
- The idea behind ghost fluid methods is that we apply boundary conditions, but these must be based on the **real material**



$$\frac{\partial \mathbf{u}_1}{\partial t} + \nabla \cdot \mathbf{f}(\mathbf{u}_1) = 0, \qquad \frac{\partial \mathbf{u}_2}{\partial t} + \nabla \cdot \mathbf{f}(\mathbf{u}_2) = 0 \qquad \frac{\partial \phi}{\partial t} + \mathbf{v} \cdot \nabla \phi = 0$$

Ghost fluid method concepts

- The challenge is that the two materials have different equations of state – we can't just use the state as it is



$$\frac{\partial \mathbf{u}_1}{\partial t} + \nabla \cdot \mathbf{f}(\mathbf{u}_1) = 0, \qquad \frac{\partial \mathbf{u}_2}{\partial t} + \nabla \cdot \mathbf{f}(\mathbf{u}_2) = 0 \qquad \frac{\partial \phi}{\partial t} + \mathbf{v} \cdot \nabla \phi = 0$$

Why can’t we use the state as it is?

- The simple answer is that this is **not thermodynamically consistent**
- If we have an interface between air and water, we would suddenly increase (or decrease) the density in the ghost fluid by a factor of 1000
- The real question might be what we mean by using the state ‘as it is’?
- If we copy across density, momentum and energy, then we will not copy across pressure, temperature or specific internal energy
- In other words, we can copy across the state ‘as it is’, provided we find the correct three variables to describe the state

Outline

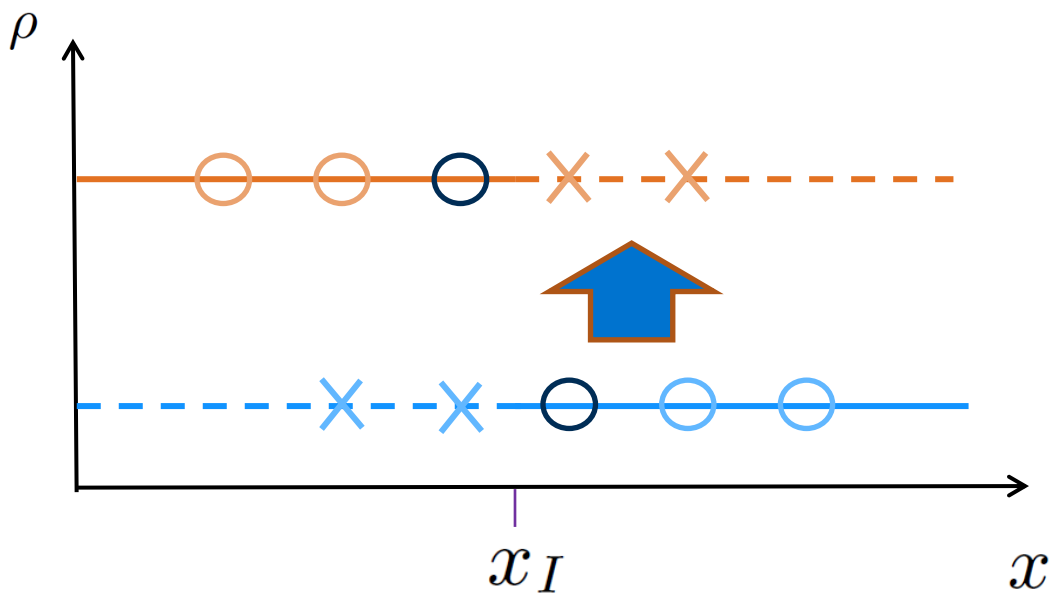
- Principles of the ghost fluid method
- The original ghost fluid method
- History of ghost fluid methods
- Implementation of a ghost fluid method

The original ghost fluid method

- Although technically not the first implementation of sharp interface multiphysics boundary conditions, the method referred to as the **original ghost fluid method** was the first which was able to work for more than a few specific materials and test cases
- Fedkiw, Aslam, Merriman and Osher (1999), “A Non-oscillatory Eulerian Approach to Interfaces in Multimaterial Flows (the Ghost Fluid Method)”
- Combined the authors’ interests in level set methods and shock thermodynamics
- Inspired by approaches which look to mitigate **shock heating** at reflective boundaries
- For these boundaries, some success was seen by forcing the boundary conditions to have **constant entropy**

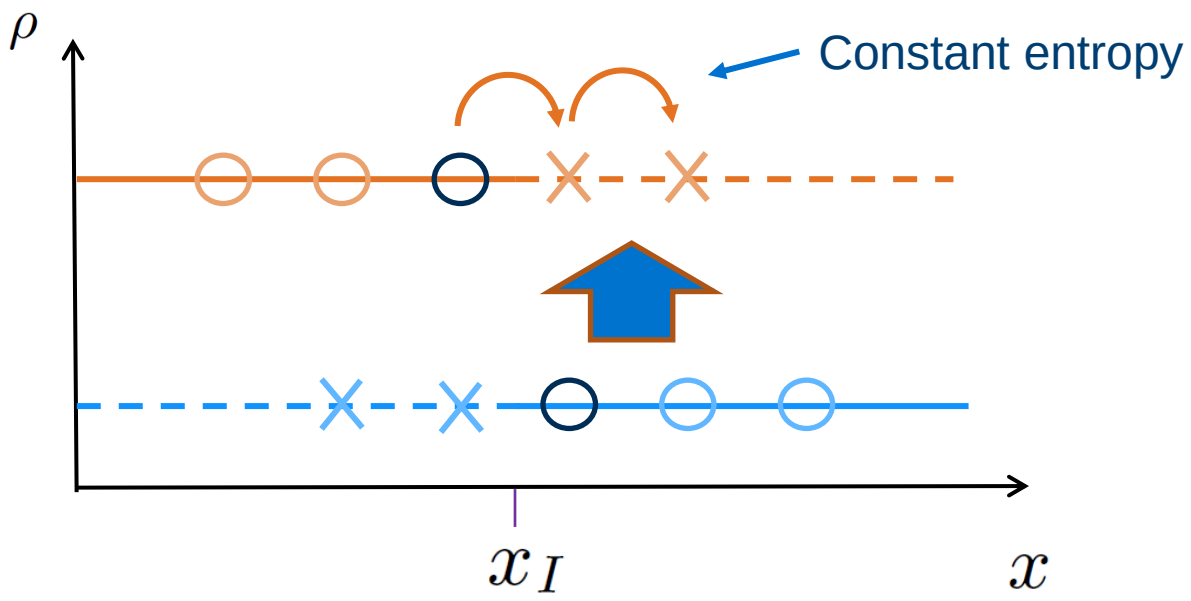
Choosing dynamic boundary conditions

- We have already discussed how the spontaneous generation of entropy leads to unphysical behaviour $T\mathrm{d}S = p\mathrm{d}V + \mathrm{d}U$
- Enforcing constant entropy does indeed stop the spontaneous generation (though perhaps the physical generation as well)



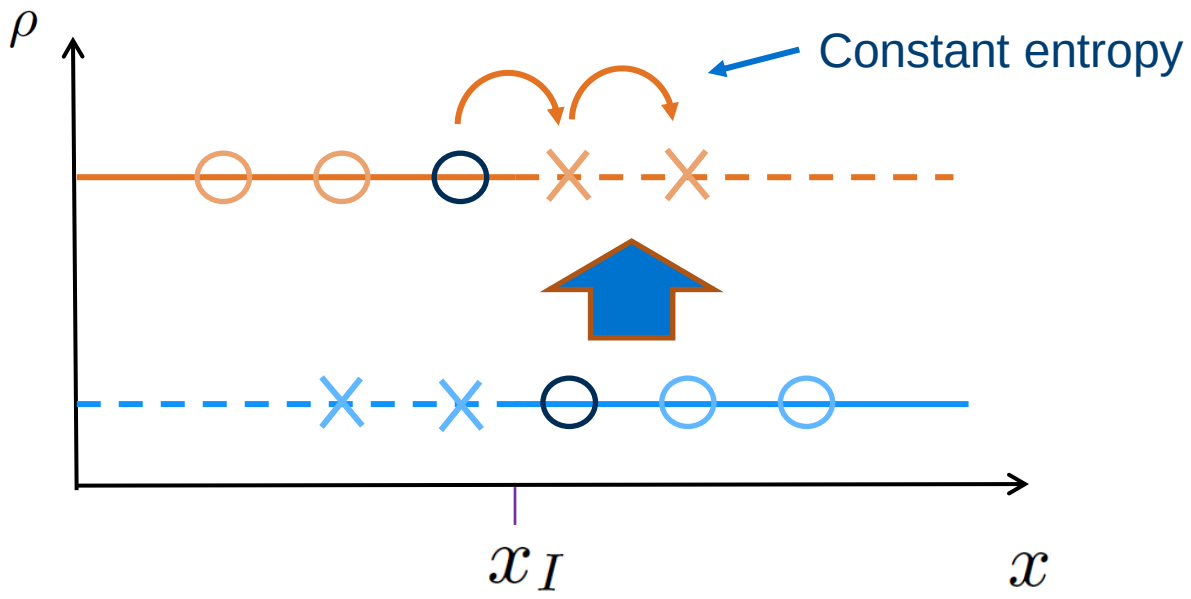
Choosing dynamic boundary conditions

- However, before the original ghost fluid method there were no good boundary conditions, so even an overly restrictive condition was an improvement
- This gives us one condition on our boundary condition variables – entropy remains constant in the ghost fluid



Choosing dynamic boundary conditions

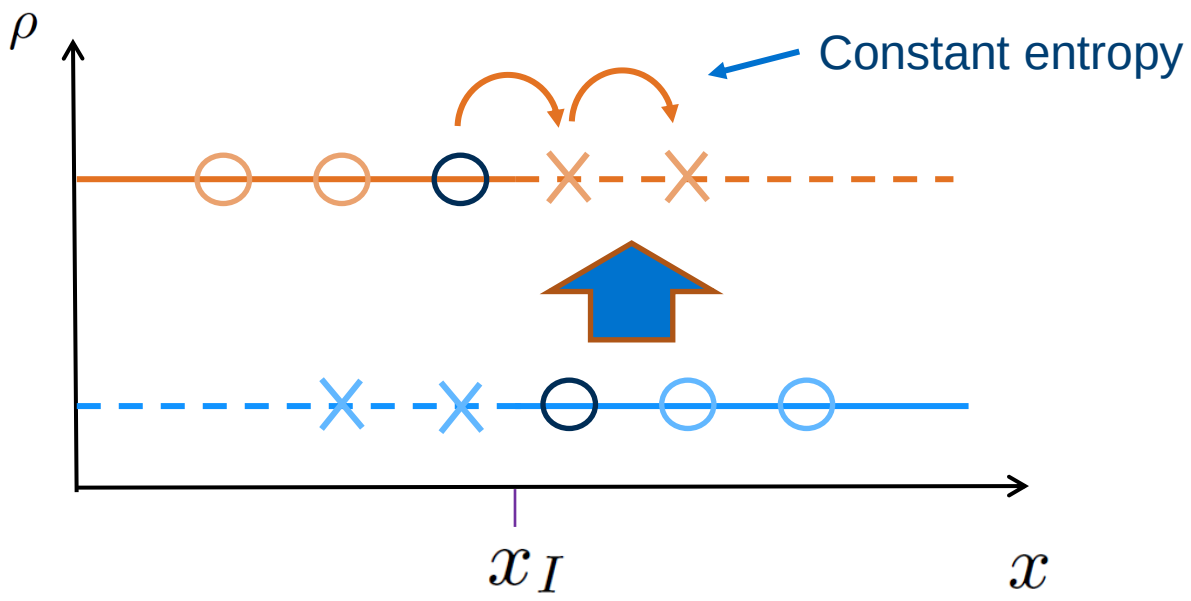
- The rest of the boundary conditions can now be filled in by considering the behaviour we expect for simple interface configurations
- For example, we expect to be able to have an interface at rest between two different materials (and different materials **should** have different densities)



Choosing dynamic boundary conditions

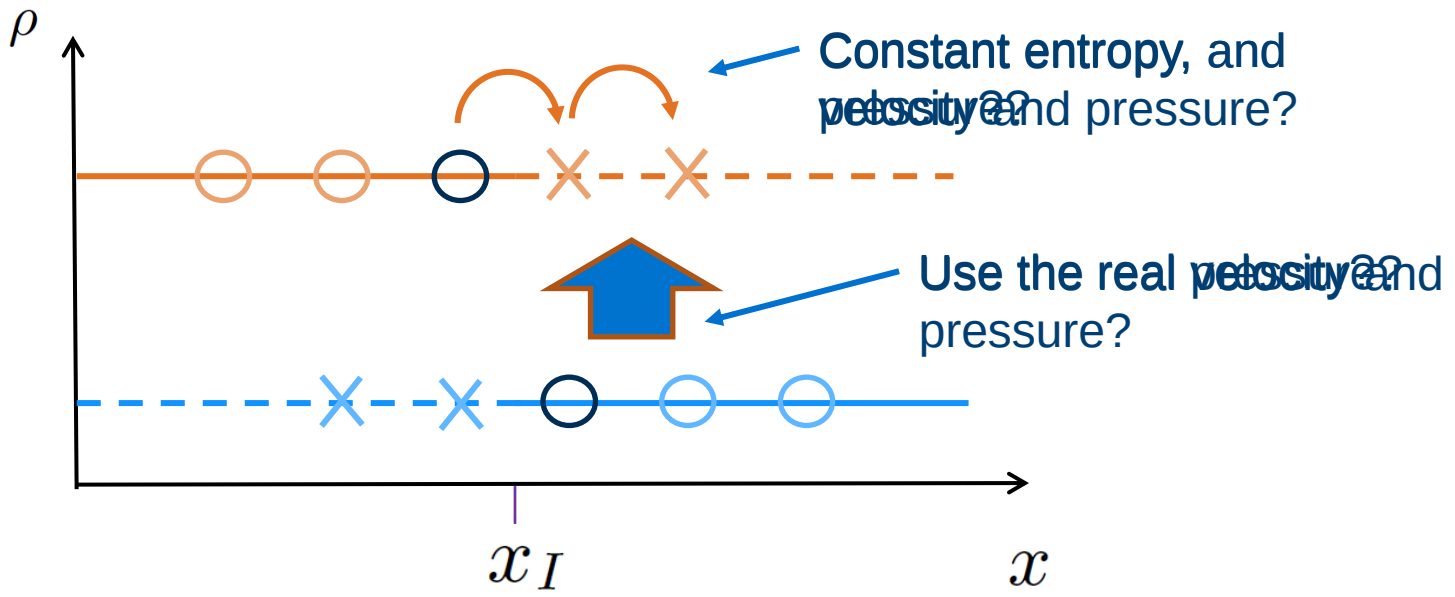
- For an interface at rest, the only way to stop evolution happening is pressure equilibrium (or normal stress equilibrium)

$$\frac{\partial \rho \mathbf{v}}{\partial t} + \nabla \cdot (\rho \mathbf{v} \otimes \mathbf{v} + \mathbf{I}p) = 0$$



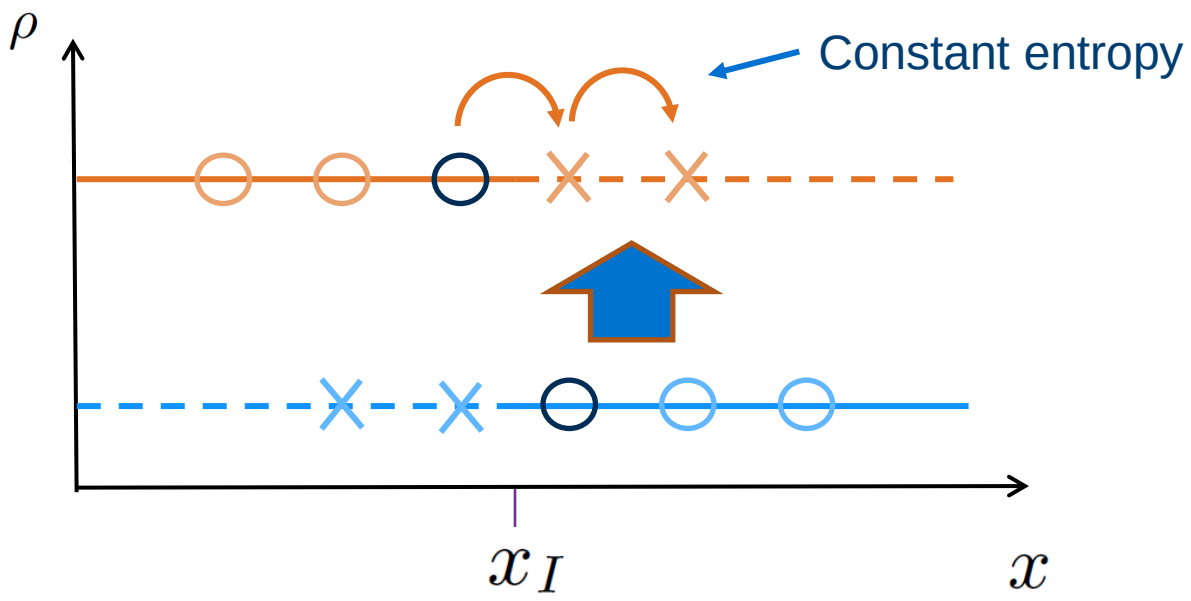
Choosing dynamic boundary conditions

- Similarly, it should be possible to have an interface moving with a constant velocity – there should be velocity equilibrium
- Unfortunately, this does not tell us much, since all possible starting points for pressure and velocity were constant anyway – but we know what variables to consider now!



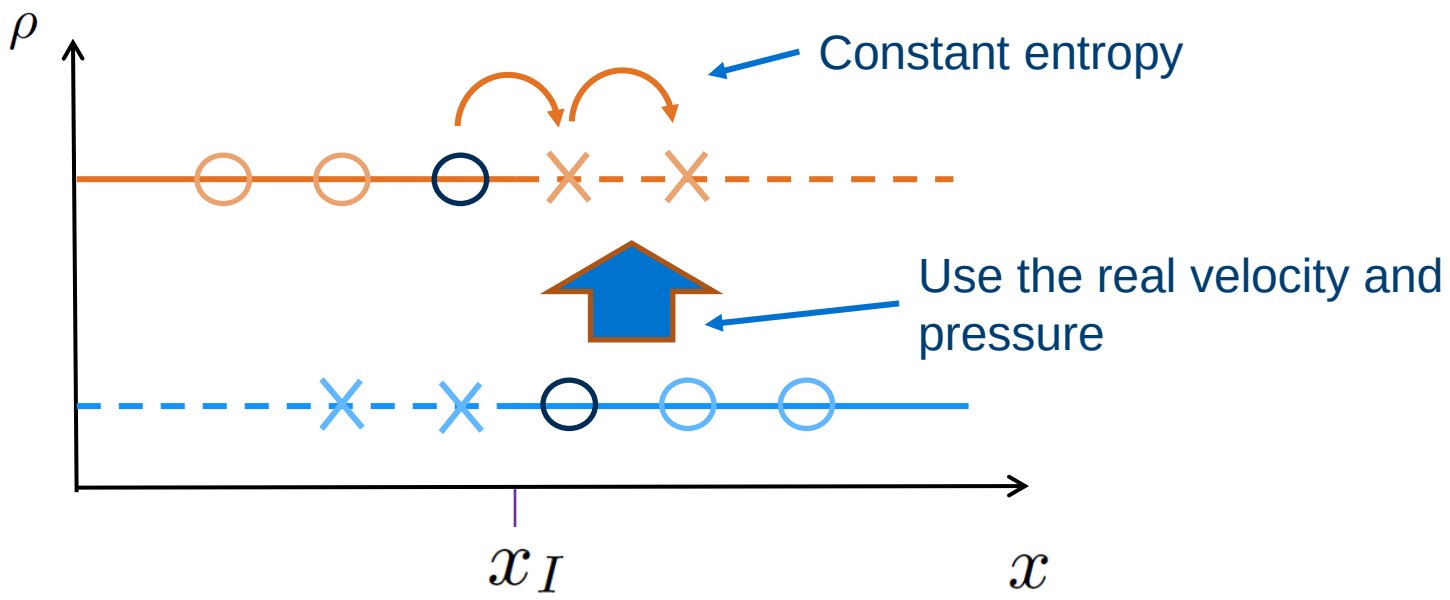
Choosing dynamic boundary conditions

- Instead we have to consider what happens when the interface is not at rest, and there is a wave moving across it
- Rarefactions are an ideal wave to consider, since we know that these obey constant entropy already



Choosing dynamic boundary conditions

- If we had an interface between two identical materials, ideally, this be indistinguishable to a single-material solution
- The only way to enforce this under our current set up is to ensure that variables are copied into the ghost fluid from the real fluid



Is this all we need?

- We now have three conditions for thermodynamically consistent boundary conditions:

  1. Constant entropy across the interface
  2. Velocity copied from the real material to the ghost fluid
  3. Pressure copied from the real material to the ghost fluid

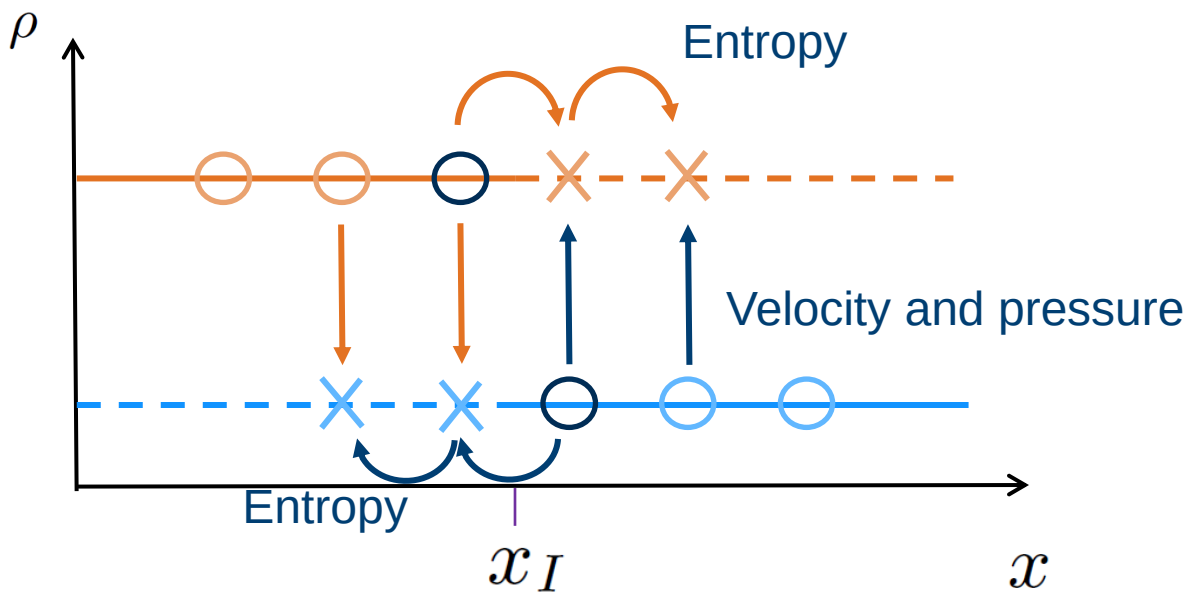
- We typically refer to pressure, but this would be a stress component for an elastoplastic solid (assuming you could get this method to work)
- Because $p = p(\rho, s)$ is a suitable set of variables for closing the equations, if we can specify entropy, we can compute density, and all other variables, in the ghost fluid

Shock waves?

- When we worked out our boundary conditions, we mentioned rarefactions and interfaces moving at constant velocity (contact discontinuities), but did not mention shock waves
- One of the features of a shock wave is that there is a jump in entropy across the wave
- This is one of the key weaknesses of the original ghost fluid method, the boundary conditions do not hold
- However, in practice, the original GFM was a very successful method
- In part, this is because shock waves are often numerically smeared, hence there is a continuous change in entropy which is captured reasonably well

Applying the boundary conditions

- Copying velocity and pressure is straightforward, since these are easy to compute
- Extrapolating entropy is used to give us a condition on the density in the ghost fluid



Extrapolating entropy

- Fortunately, we do not have to compute entropy to make a constant extrapolation, but we do need to know how it is defined
- For example, for an ideal gas, we have

$$s = c_v \ln p - c_p \ln \rho + \text{const}$$

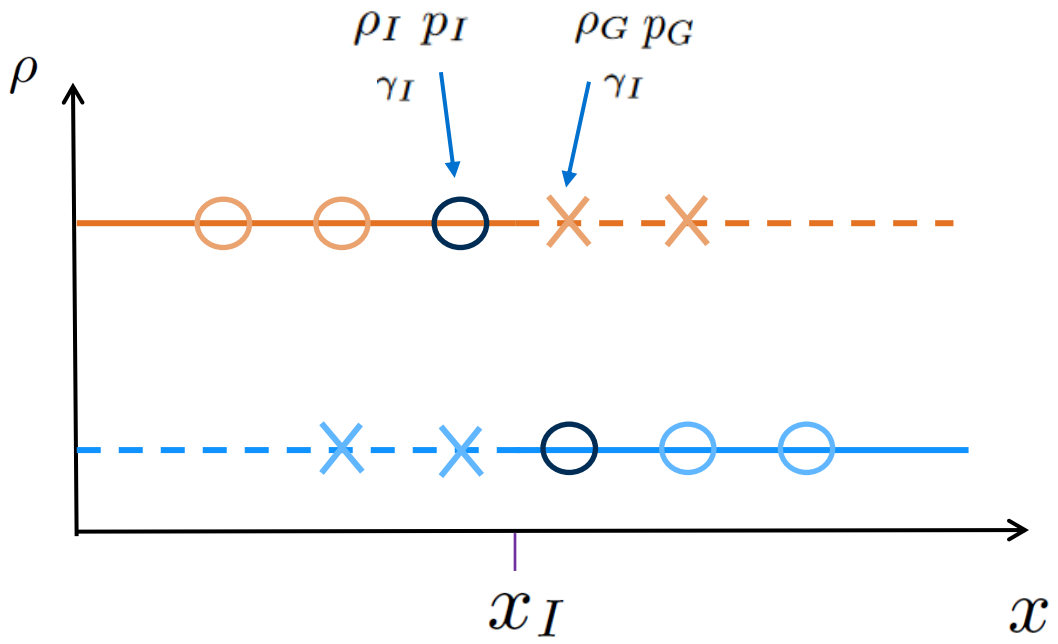
- Rearranging this, with constant entropy, we have

$$\frac{p}{\rho^\gamma} = \text{const}$$

- We have enough information to compute the density in the ghost fluid region from this

Extrapolating entropy

- We consider two sets of values, the **real** values, or **interface** values, which are the fluid values at the last cell before the interface
- We also have the ghost fluid values of the same quantities

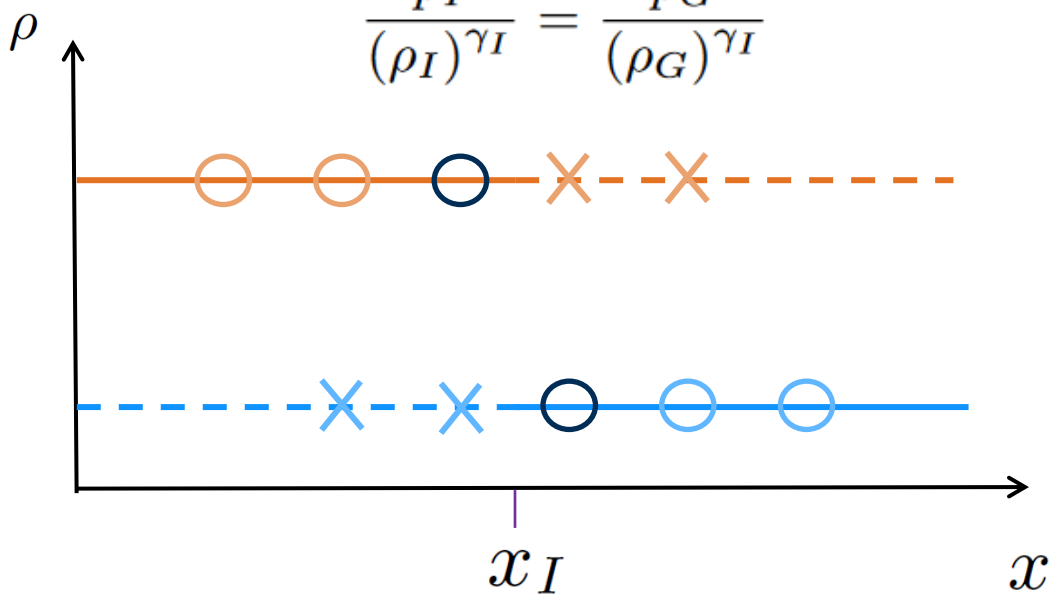


Note that both real and ghost cells are part of the **same material**, so have the same adiabatic index

Extrapolating entropy

- Since we know all interface values (as these just come from the simulation), and we know the pressure in the ghost region (this comes from the other material), we can work out density in the ghost region
- The same process works for all ghost cells in your numerical stencil

$$\frac{p_I}{(\rho_I)^{\gamma_I}} = \frac{p_G}{(\rho_G)^{\gamma_I}}$$



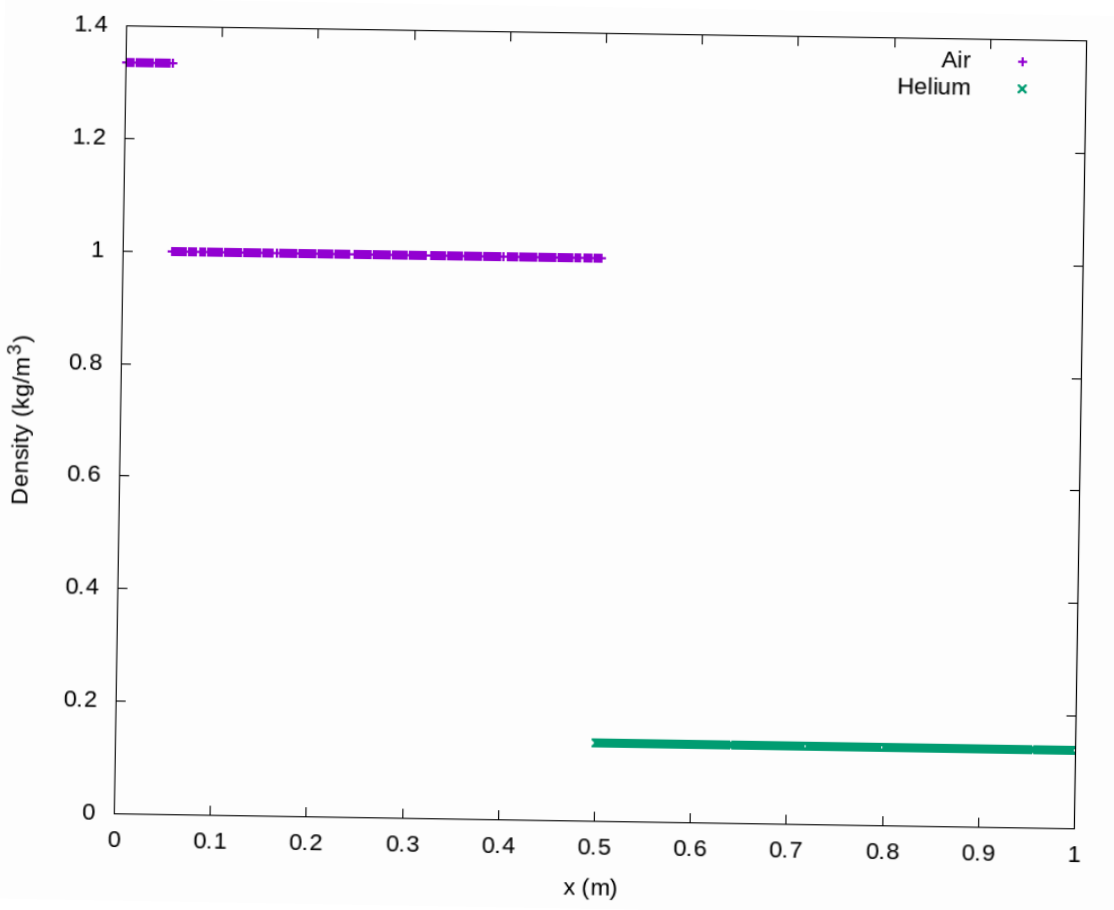
Example using the ghost fluid method

- Shock tube tests are a standard method for validating a ghost fluid method implementation
- In the case of a shock wave hitting a material interface, we expect a pattern of:

a) transmitted shock wave,

b) reflected rarefaction

c) advection of the interface



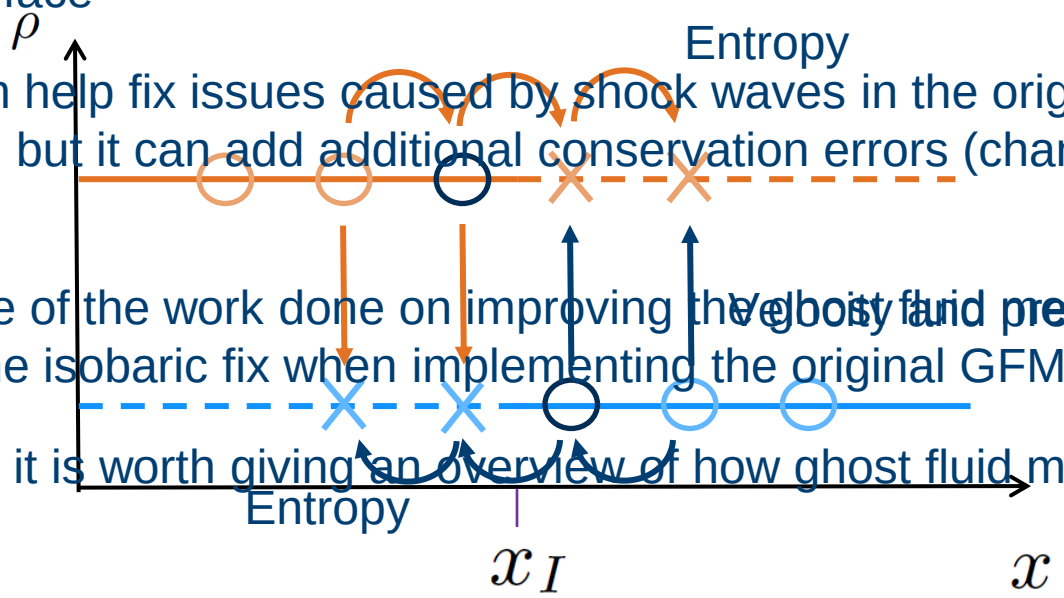
Assumptions of the original ghost fluid method

- We have already mentioned that the original ghost fluid method does not provide the correct behaviour at shock waves
- This has always been known, and was discussed in the original paper
- However, because the method does work well in practice, and it is easy to implement, it has seen a lot of use
- But there are some cases in which the original ghost fluid method does not work at all

1. At very strong shock waves – it can predict a rarefaction when a shock wave should occur
2. Across very large density gradients, e.g. at air-water interfaces, though careful choice of equation of state can help
3. Across very different materials, especially if it is not easy to obtain density from entropy

The isobaric fix

- In the original ghost fluid paper, you will see mention of the isobaric fix
- This extends the extrapolation of entropy, taking the value from once cell further from the interface
- As a result, the isobaric fix also changes the density in the real cell adjacent to the interface
- This can help fix issues caused by shock waves in the original ghost fluid method, but it can add additional conservation errors (changing the mass within a cell)
- Because of the work done on improving the ghost fluid method, do not worry about the isobaric fix when implementing the original GFM
- Instead, it is worth giving an overview of how ghost fluid methods have changed



Outline

- Principles of the ghost fluid method
- The original ghost fluid method
- History of ghost fluid methods
- Implementation of a ghost fluid method

Improving ghost fluid methods

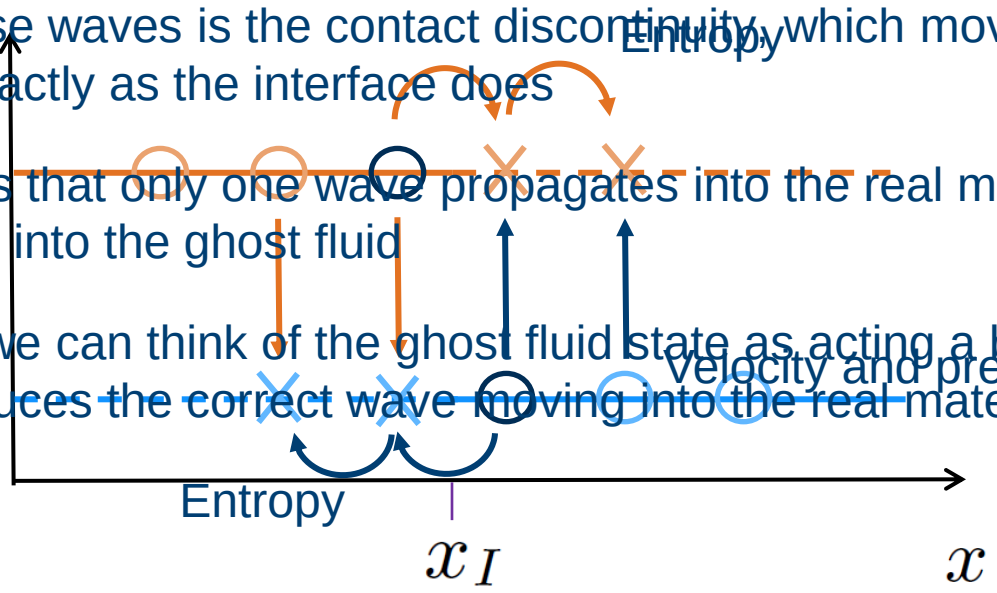
- Due to the shortcomings of the ghost fluid method, various improved versions have been developed
- Most of these occurred within about 10 years of the first publication, though there is still active development
- These developments allow ghost fluid methods to provide a stable multimaterial technique even for strong shock waves or different material properties across the interface
- Though these various methods all differ, the underlying techniques do not change; the specification of a ghost fluid region to describe the material interaction at the interface
- We summarise some of the notable implementations

The modified ghost fluid method

- This was the first major development of the ghost fluid method by Liu, Khoo and Yeo (2002)
- The aim of this work was **specifically** to address the problems suffered when a strong shock wave impacts an interface
- Their approach focuses on when the original GFM cannot capture the correct behaviour of the waves (it may predict a rarefaction when physically a shock wave occurs)
- Their modification to the method is to predict the wave behaviour at the interface, and thus the star states that would arise from a Riemann problem across the interface, such that the correct wave structure can be obtained
- What is the connection between ghost fluid methods and Riemann problems?

Riemann problems and Ghost Fluid Methods

- When we set Riemann problem boundary conditions, we are setting a state across the interface which is meant to reproduce the correct wave interaction
- This will have three waves (for the Euler equations)
- One of these waves is the contact discontinuity, which moves with the material velocity, exactly as the interface does
- That means that only one wave propagates into the real material, and one propagates into the ghost fluid
- Therefore we can think of the ghost fluid state as acting a bit like a star state, which produces the correct wave moving into the real material



Strong shock waves

- Because our ghost fluid is acting like a star state, it should reproduce the correct wave behaviour
- Recall that for a shock wave to be generated,

$$f_K\left(p^*, \mathbf{u}_K\right) = \begin{cases} \left(p - p_K\right)\sqrt{\frac{A_K}{p^* + B_K}}, & p* > p_K \\ \frac{2c_{s,K}}{\gamma - 1}\left[\left(\frac{p^*}{p_K}\right)^{\frac{\gamma-1}{2\gamma}} - 1\right], & p^* < p_K \end{cases} \qquad A_K = \frac{2}{(\gamma+1)\rho_K} \qquad B_K = \frac{\gamma-1}{\gamma+1}$$

- The original ghost fluid method would always use the real pressure at the interface, in other words, it assumes $p^* = p_K$ for one of the left or right states
- So if the ghost fluid pressure is less than the real pressure, there will always be a rarefaction
- But there are many Riemann problems exist where a low pressure, but high density and/or high velocity can generate a shock wave

The modified ghost fluid method

- When we first looked at Riemann problems, we introduced characteristics, e.g.

$$\mathrm{d}p \pm \rho c_s \mathrm{d}u = 0$$

- These were used to calculate variables inside the intermediate state, but it was quite a complex process
- However, if we **linearise**, which, in this case, assumes all jumps are a line of constant gradient connecting a known state to an interface state, solving for the ghost fluid variables becomes easier

$$\mathrm{d}p = p_I - p_{i-1} \qquad \mathrm{d}p = p_{i+2} - p_I$$

- These expressions are for an interface which lies between cells $i-1$ and i

The modified ghost fluid method

- Expressions for the sound speed are

$$c_{s,L}^2 = \frac{\partial p}{\partial \rho} \approx \frac{p_I - p_{i-1}}{\rho_I^L - \rho_{i-1}} \qquad c_{s,R}^2 = \frac{\partial p}{\partial \rho} \approx \frac{p_I - p_{i+2}}{\rho_I^R - \rho_{i+2}}$$

- Quantities which don't jump are instead averaged $\rho \approx \sqrt{\rho_I^L \rho_{i-1}}$
- The system is closed with an assumption that two shock waves are generated, hence intermediate pressure is also given by the **two-shock** estimate for a Riemann problem (see Toro's book)
- This gives enough information to uniquely define two approximate Riemann problem states

The interface ghost fluid method

- This is another modification of the ghost fluid method which uses the characteristic jumps

$$\mathrm{d}p \pm \rho c_s \mathrm{d}u = 0$$

- Developed by Hu and Khoo (2004), it attempts to use more information about constant entropy in the ghost fluid region to provide boundary conditions
- This again builds on the idea that the ghost fluid is representing the star state of a Riemann problem
- Because we assume constant entropy in the ghost fluid region, we are assuming constant entropy across the wave moving into the real material
- We know how to derive expressions for intermediate states across acoustic waves with constant entropy

The interface ghost fluid method

$$u_I = u_L - \frac{2c_{s,L}}{\gamma_L - 1}\left[\left(\frac{p_I}{p_L}\right)^{\frac{\gamma_L - 1}{2\gamma_L}} - 1\right] \qquad u_I = u_R + \frac{2c_{s,R}}{\gamma_R - 1}\left[\left(\frac{p_I}{p_R}\right)^{\frac{\gamma_R - 1}{2\gamma_R}} - 1\right]$$

- These are the expressions for left- and right-moving rarefactions, but for two ideal gasses with different adiabatic index
- These are coupled with the same expression for density in the ghost fluid region as the original ghost fluid method
- In other words, we have four equations for four unknown variables
- But these equations do need to be solved iteratively, much like the exact solver for a standard Riemann problem

The explicit simplified interface method

- All ghost fluid methods presented so far have assumed there is a constant entropy across the interface
- This is not always true, if there is a temperature gradient in a material, then there is an entropy gradient too
- In this case, the constant entropy assumption can cause large error
- Lombard and Donat (2005) devised the Explicit Simplified Interface Method (ESIM) to use a continuous (linear) entropy assumption
- The idea here is that not only do we know how velocity and pressure jump across the interface (they don't), we can use Taylor expansions to compute how their derivatives jump

$$\left[\frac{1}{\rho}\frac{\partial p}{\partial x}\right] = 0 \qquad \left[\rho c_s^2 \frac{\partial v}{\partial x}\right] = 0$$

The explicit simplified interface method

$$\left[\frac{1}{\rho}\frac{\partial p}{\partial x}\right] = 0 \qquad \left[\rho c_s^2 \frac{\partial v}{\partial x}\right] = 0$$

- Note, we are not computing discontinuous derivatives, just comparing how the derivative either side of the interface, **from the interface into the real material,** is related in order for entropy to be linearly extrapolated
- This then gives us four equations for our four unknown variables in the ghost fluid states
- However, it gives us values at the interface itself, not finite volume quantities
- So an extrapolation back to cell centres must be made

$$\mathbf{u}_{i+1} = \mathbf{u}_I^L + (x_{i+1} - x_I)\,\dot{\partial}_x \mathbf{u}_I^L$$

The real ghost fluid method

- Almost all modifications to the GFM have started by identifying the similarity between the ghost fluid state and a Riemann problem star state
- And as they improved, they started to solve iterative equations to work out this state, **but**, under certain assumptions about the wave behaviour
- The real ghost fluid method of Wang, Liu and Khoo (2006) used the fact that the standard equation for a Riemann problem is actually a general equation for multiple materials

$$f_L\left(p_I, \mathbf{u}_L\right) + f_R\left(p_I, \mathbf{u}_R\right) + \Delta v = 0$$

- There is nothing in this equation which say the left and the right states have to have the same equation of state
- If we are going to iteratively solve an approximate Riemann problem, why not iteratively solve the **actual** Riemann problem

The real ghost fluid method

$$f_L\left(p_I, \mathbf{u}_L\right) + f_R\left(p_I, \mathbf{u}_R\right) + \Delta v = 0$$

- The real ghost fluid method is now the basis of most modern implementations of ghost fluid methods
- As a result, we shall look at this technique in much more detail later
- But under this approach, no assumptions need to be made that materials are ideal gases, or follow other simple equations of state
- As a result, it is ideal for sharp multiphysics interface between any states of matter

The Riemann ghost fluid method

- The Riemann ghost fluid method of Sambasivan and UdayKumar (2009) is a modification of the real ghost fluid method
- The mixed-material Riemann problem is still used to obtain the intermediate states, however the means of picking the states left and right of the interface changes
- This is only an issue in more than one-dimension, where more than one cell can be considered adjacent to the interface (in 1D, there is no practical difference between the Riemann and real GFMs)
- For example; an interface passing diagonally through a cell could have adjacent cells to the top and right - which one provides the initial condition for the interface?
- We shall return to this later, when considering more than one dimension

The practical ghost fluid method

- Once solving Riemann problems became standard, there were no substantial changes to ghost fluid methods
- However, Xu, Feng and Liu (2016) introduced a technique which uses the solution to the mixed-material Riemann problem to define the ghost fluid states, but not through direct application
- The Riemann problem solution is an evolved state of the problem ($t > t^n$), so conceptually, using this as the ghost fluid state at time t^n is incorrect (though in the limits of resolution, this can be considered correct)
- The practical ghost fluid method attempts to resolve this, by finding a state for the ghost fluid region. e.g. $\mathbf{u}_{G,L}$, such that the interaction between $\mathbf{u}_{G,L}$ and $\mathbf{u}_L$ results in $\mathbf{u}_L^*$
- However, this state is not unique, so the practical GFM attempts to use a ‘mirrored structure’ to identify a ghost fluid state

…and more

- There are more variants of the GFM than described here, though none that have seen widespread use
- All GFMs presented are not conservative; since each cell contains a single material, as the interface passes over a cell centre, the material suddenly ‘switches’, and a mass gain/loss occurs
- Typically, the conservation errors are small, and deemed an ‘acceptable’ price for the accuracy of a sharp interface
- Attempts have been made to make a conservative ghost fluid method (Nguyen et al. (2002) and Liu et al. (2011)) but typically these suffer from other accuracy issues at the interface

Outline

- Principles of the ghost fluid method
- The original ghost fluid method
- History of ghost fluid methods
- Implementation of a ghost fluid method

Coding a sharp interface method

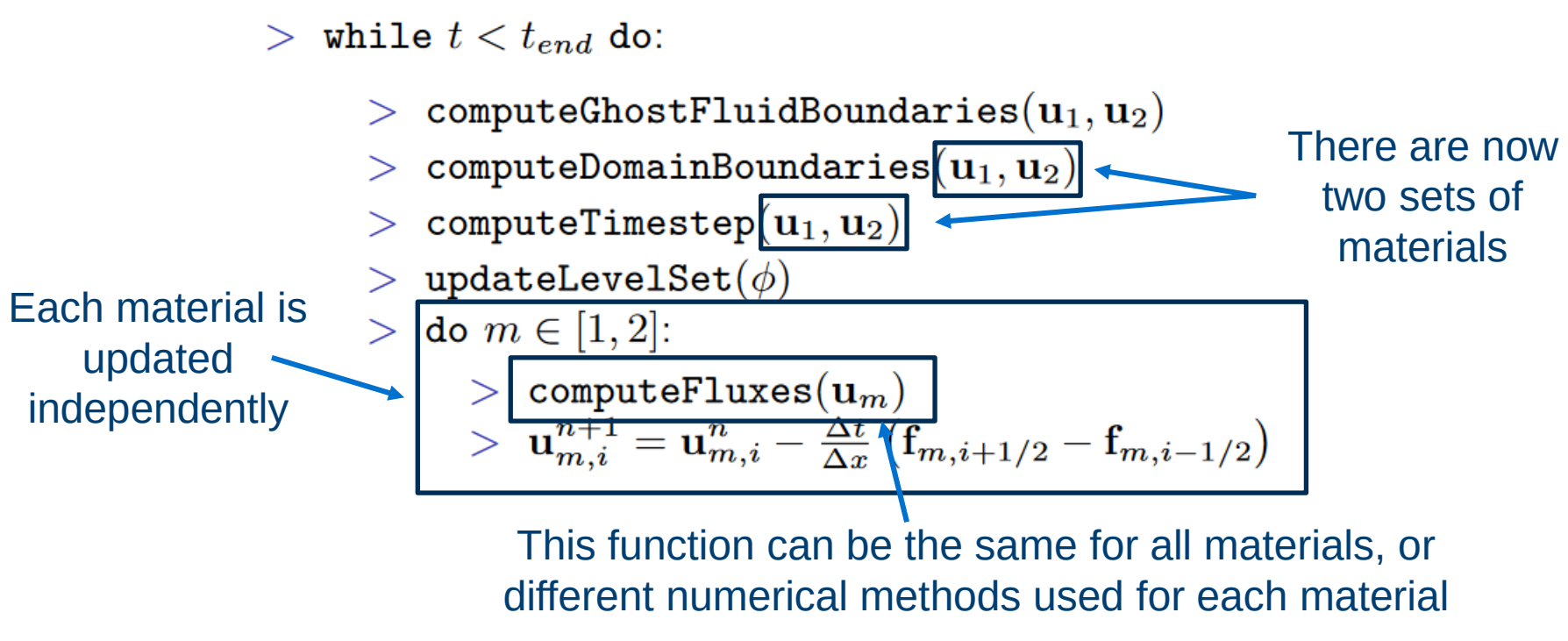
- One of the main advantages behind ghost fluid methods is that you can build them on top of existing code

> while $t < t_{end}$ do:
>
> > computeDomainBoundaries($\mathbf{u}$)
> > computeTimestep($\mathbf{u}$)
> > computeFluxes($\mathbf{u}$)
> > $\mathbf{u}_i^{n+1} = \mathbf{u}_i^n - \frac{\Delta t}{\Delta x}\left(\mathbf{f}_{i+1/2} - \mathbf{f}_{i-1/2}\right)$

- Most of the additional work is in creating new functions for the level set equation and the dynamic boundary conditions
- Of course, a few changes to make sure you have more than one equation of state and set of variables are necessary

Coding a sharp interface method

- The structure of a two-material ghost fluid multiphysics code would then be



Coding a sharp interface method

- The structure of a two-material ghost fluid multiphysics code would then be

```
> while t < t_end do:
    > computeGhostFluidBoundaries(u_1, u_2)
    > computeDomainBoundaries(u_1, u_2)
    > computeTimestep(u_1, u_2)
    > updateLevelSet(φ)
    > do m ∈ [1, 2]:
        > computeFluxes(u_m)
        > u_{m,i}^{n+1} = u_{m,i}^n − (Δt/Δx) (f_{m,i+1/2} − f_{m,i−1/2})
```

Usually each material has the same boundary condition, but they don't have to

The wave speed can be computed for each material and the overall maximum used

Sometimes care is needed to make sure material inside the ghost fluid region doesn't dominate the time step

Coding a sharp interface method

- The structure of a two-material ghost fluid multiphysics code would then be

```
> while t < t_end do:
    > computeGhostFluidBoundaries(u_1, u_2)
    > computeDomainBoundaries(u_1, u_2)
    > computeTimestep(u_1, u_2)
    > updateLevelSet(φ)
    > do m ∈ [1, 2]:
        > computeFluxes(u_m)
        > u_{m,i}^{n+1} = u_{m,i}^{n} − (Δt/Δx) (f_{m,i+1/2} − f_{m,i−1/2})
```

Ghost fluid boundaries are filled first, as these may then meet the physical boundary

Also, ghost fluid within the numerical stencil of a real cell can contribute to time step

And because the level set position is needed at the current time step, it must happen before the level set update

Coding a sharp interface method

- The structure of a two-material ghost fluid multiphysics code would then be

> while $t < t_{end}$ do:
>
> > computeGhostFluidBoundaries$(\mathbf{u}_1, \mathbf{u}_2)$
> > computeDomainBoundaries$(\mathbf{u}_1, \mathbf{u}_2)$
> > computeTimestep$(\mathbf{u}_1, \mathbf{u}_2)$
> > updateLevelSet(ϕ)
> > do $m \in [1, 2]$:
> >
> > > computeFluxes$(\mathbf{u}_m)$
> > > $\mathbf{u}_{m,i}^{n+1} = \mathbf{u}_{m,i}^{n} - \frac{\Delta t}{\Delta x}\left(\mathbf{f}_{m,i+1/2} - \mathbf{f}_{m,i-1/2}\right)$

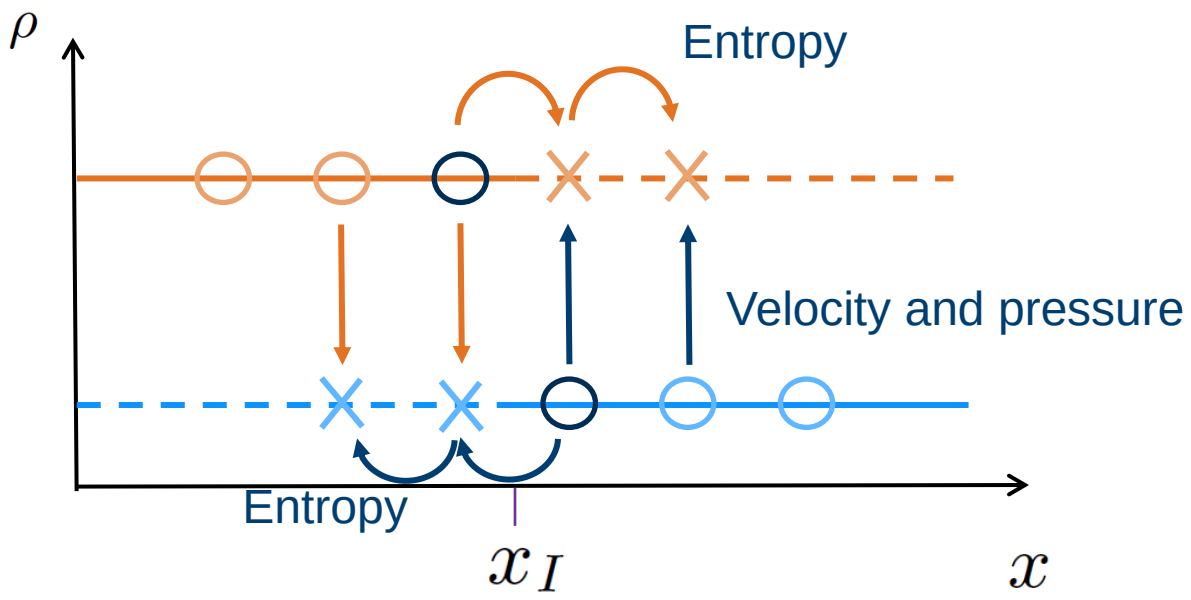
The level set update depends on the current velocity, so must happen before the material update

Technically, it does also need consideration in the stable time step (but it evolves with velocity only)

If you are reinitialising the level set, this is best done after the update

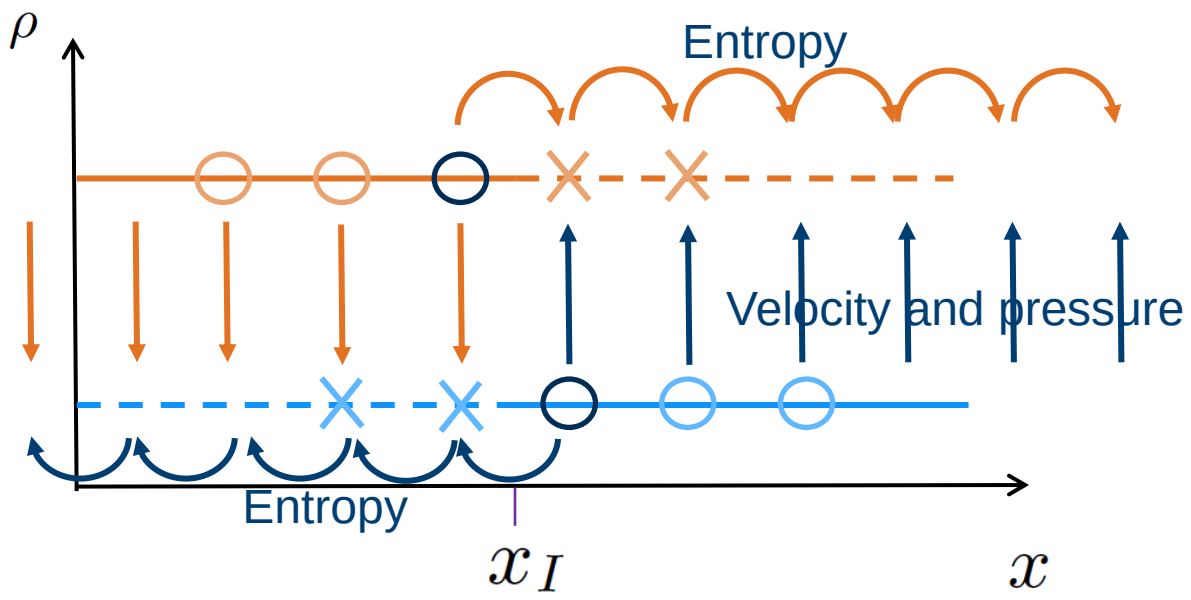
How many ghost cells need setting?

- When first describing our ghost fluid method, we illustrated two cells in the ghost fluid region being set (assuming that was our numerical stencil)
- But is the enough cells?



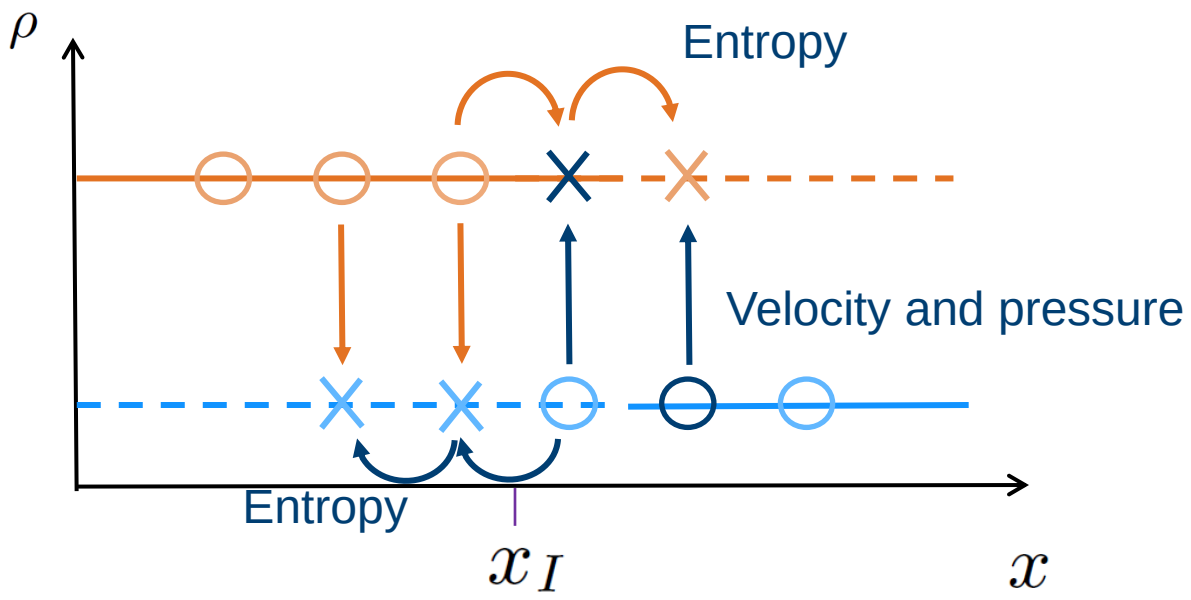
How many ghost cells need setting?

- There is nothing wrong with applying boundary conditions to **all** ghost fluid cells
- It is slightly inefficient, but is **certain to work**, so should be the first thing you try
- But what happens if the interface moves during a time step?



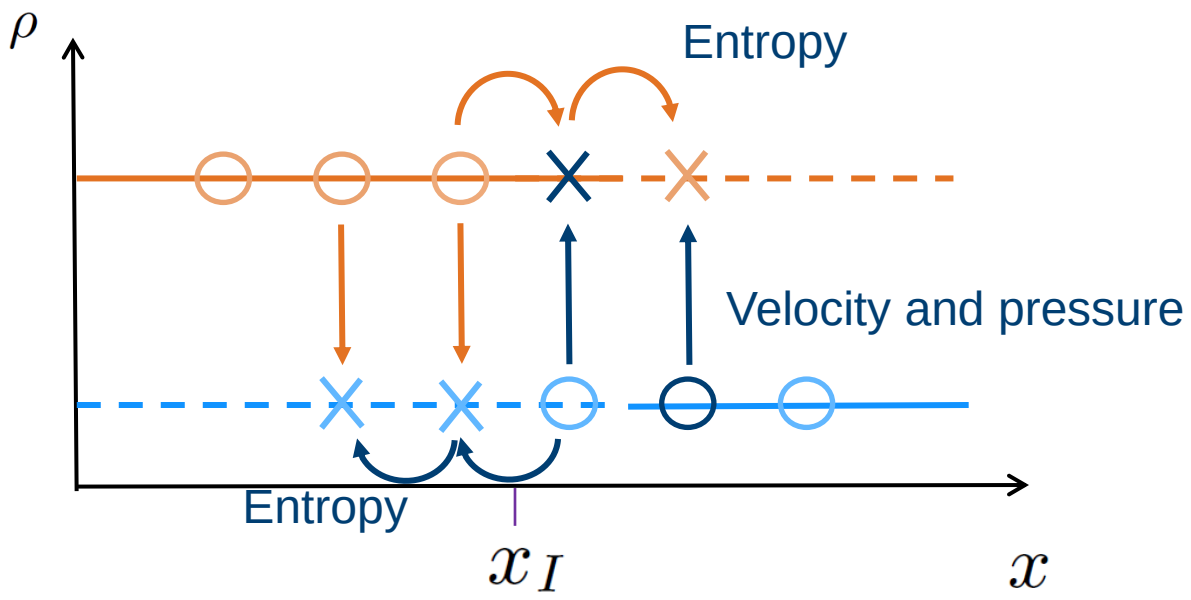
How many ghost cells need setting?

- Sometimes a real cell will become a ghost cell – this is not a problem, it simply gets boundary conditions applied next time step
- At the same time, a ghost cell becomes a real cell – how do we know if this has a sensible value?



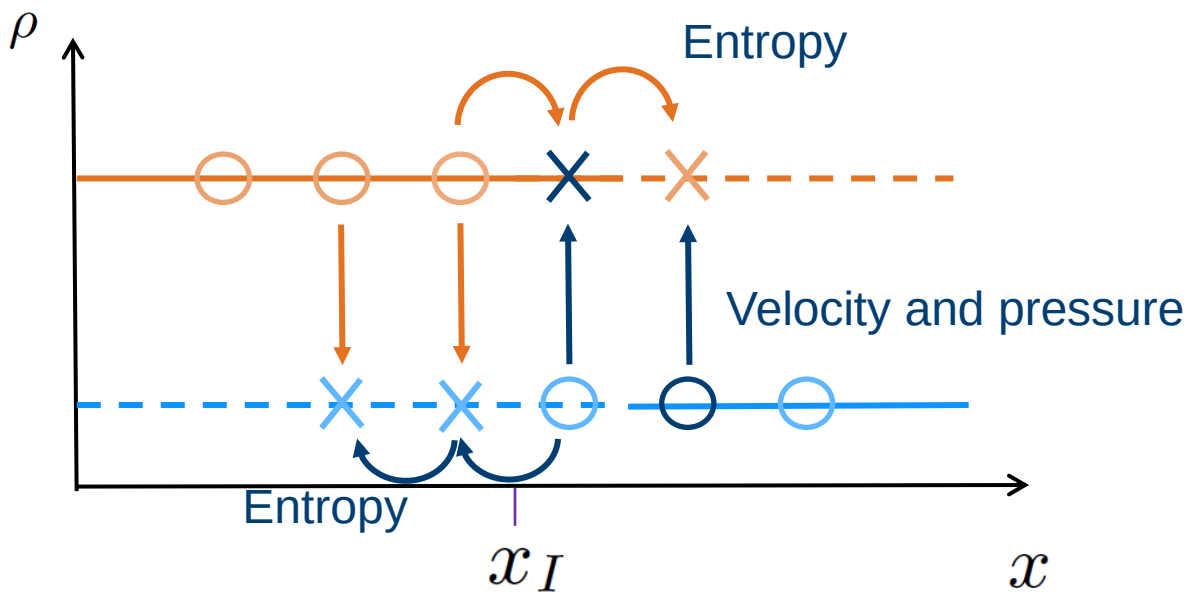
How many ghost cells need setting?

- We **could** attempt to **extrapolate into this newly revealed cell** – but that has issues (such as extrapolation over shocks)
- Or we could **leave it with whatever value it had**, and hope for the best (which is easy, and actually works well)



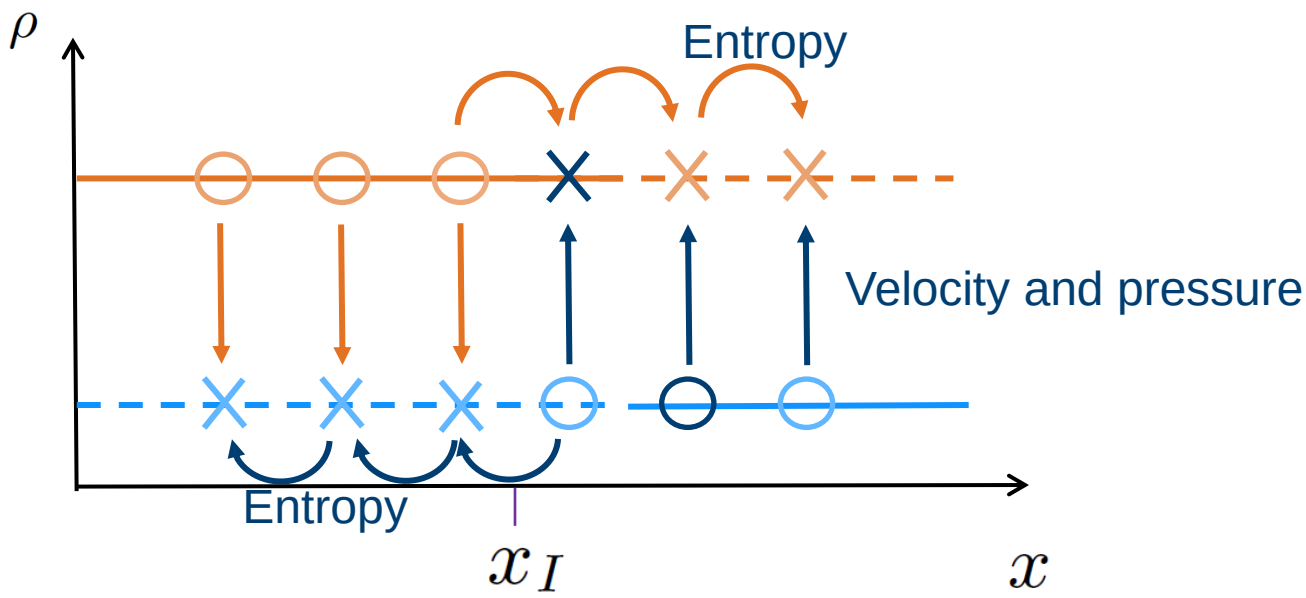
How many ghost cells need setting?

- Although it may seem a bit dodgy, because the ghost cells were, in some ways, analogous to Riemann problem intermediate states, they are a reasonable assumption as to what the behaviour is in this newly revealed cell
- But only if the update step hasn't messed everything up



How many ghost cells need setting?

- As a result, we need to use one extra ghost fluid cell than the numerical stencil requires
- Note that these efficiency concerns **are only really an issue in more than 1D**, where constant extrapolation becomes computationally intensive



References

- Fedkiw, Aslam, Merriman and Osher (1999), “A Non-oscillatory Eulerian Approach to Interfaces in Multimaterial Flows (the Ghost Fluid Method)”
- Liu, Khoo, Yeo (2003), “Ghost fluid method for strong shock impacting on material interface”
- Hu and Khoo (2004), “An interface interaction method for compressible multifluids”
- Lombard and Donat (2005), “The explicit simplified interface method for compressible multicomponent flows”
- Wang, Liu and Khoo (2006), “A real ghost fluid method for the simulation of multimedium compressible flow”
- Sambasivan and Udaykumar (2009), “Ghost Fluid Method for Strong Shock Interactions Part 1: Fluid Fluid Interfaces”
- Xu, Feng and Liu (2016) “Practical Techniques in Ghost Fluid Method for Compressible Multi-Medium Flows”