UNIVERSITY OF CAMBRIDGE

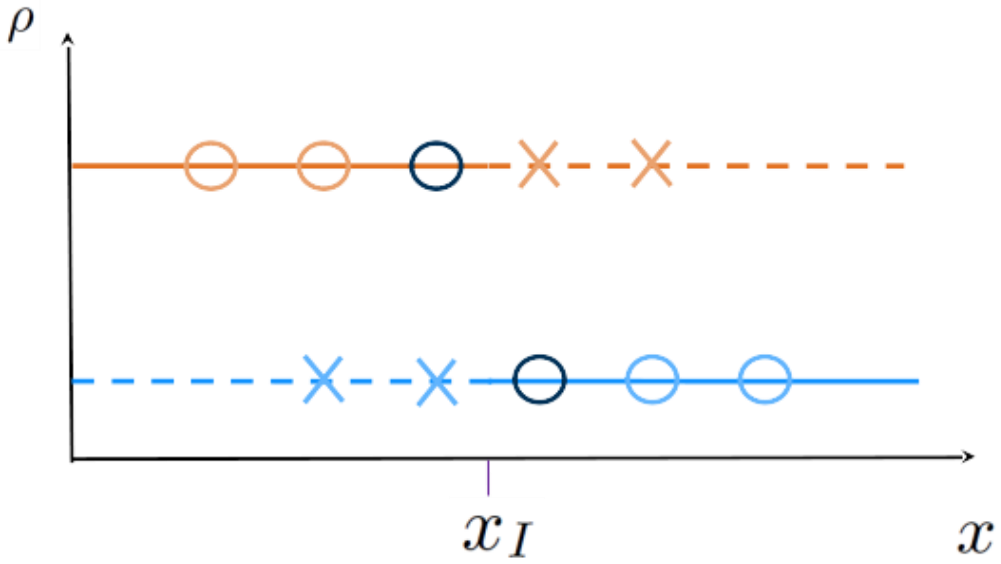
Ghost fluid methods – part 2

Stephen Millmore

Laboratory for Scientific Computing

Riemann problem-based ghost fluid methods

- When we last discussed ghost fluid methods, we stated that Riemann problem-based methods could be used to provide dynamic boundary conditions
- We shall return to the configuration where we have identified states adjacent to the interface, and wish to ensure that the ghost fluid region is set correctly for wave interactions to be reproduced



Outline

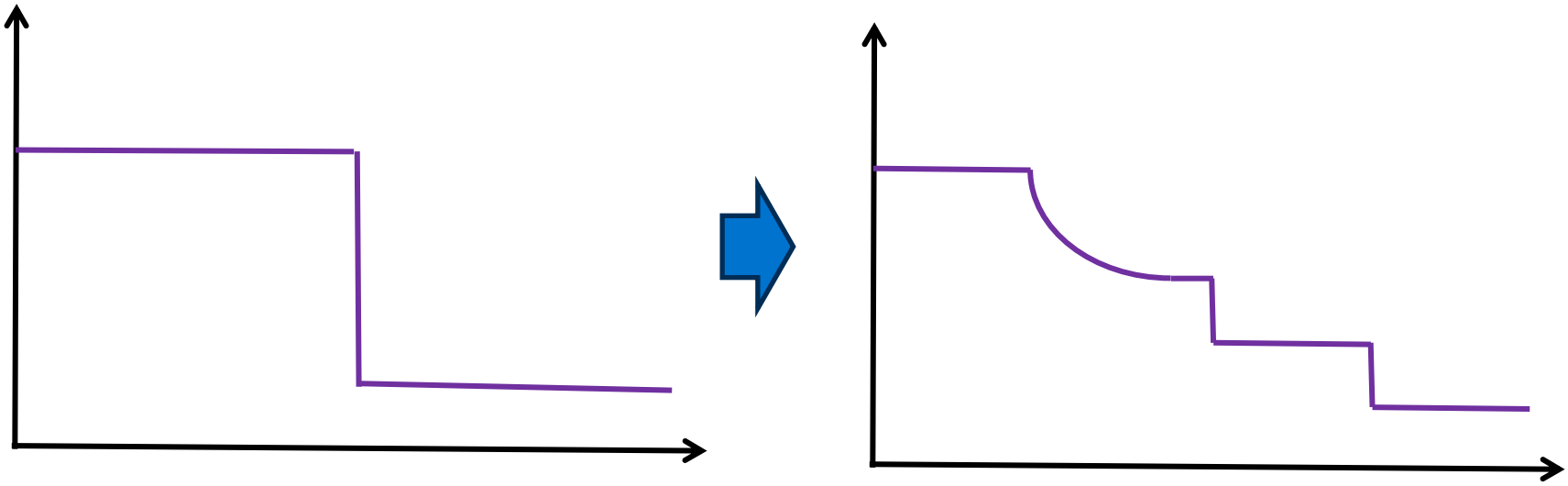
- Riemann problems and the ghost fluid method
- Mixed-material Riemann problems
- Ghost fluid methods in multiple dimensions

Outline

- Riemann problems and the ghost fluid method
- Mixed-material Riemann problems
- Ghost fluid methods in multiple dimensions

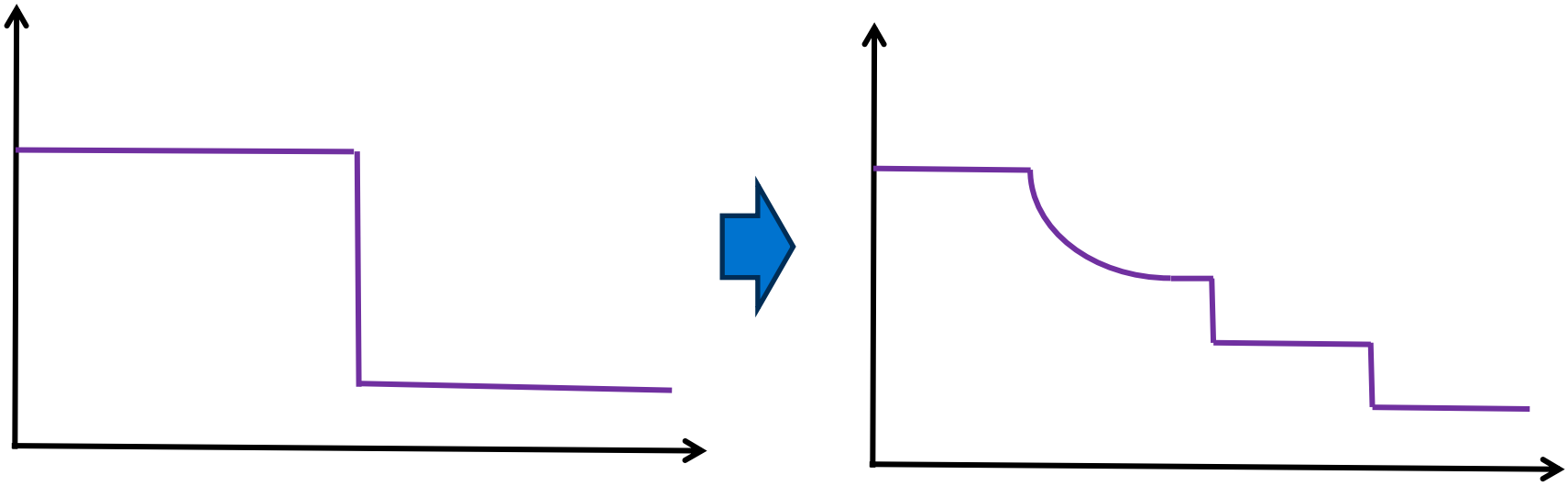
Generalising a Riemann problem to two materials

- Perhaps the first question we should address is why we can consider mixed-material Riemann problems in the first place
- We start with a single material, three-wave solution



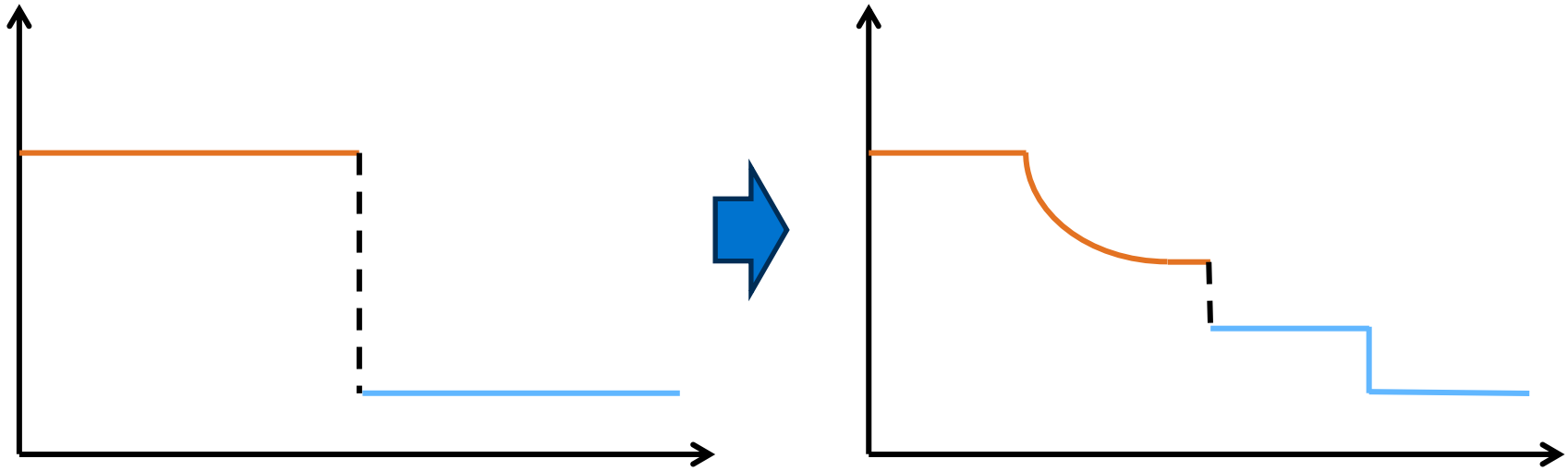
Generalising a Riemann problem to two materials

- When we originally derived the Riemann problem solution, we did so assuming each initial state obeyed the Euler equations
- We did not say that each initial state **had** to have the same equation of state though (we just assumed that they did)



Generalising a Riemann problem to two materials

- If the two materials had different equations of state, we already know that the interface moves with material velocity, i.e. behaves like a contact discontinuity
- Therefore we'd simply end up with one intermediate state coming from one material, and one from the other

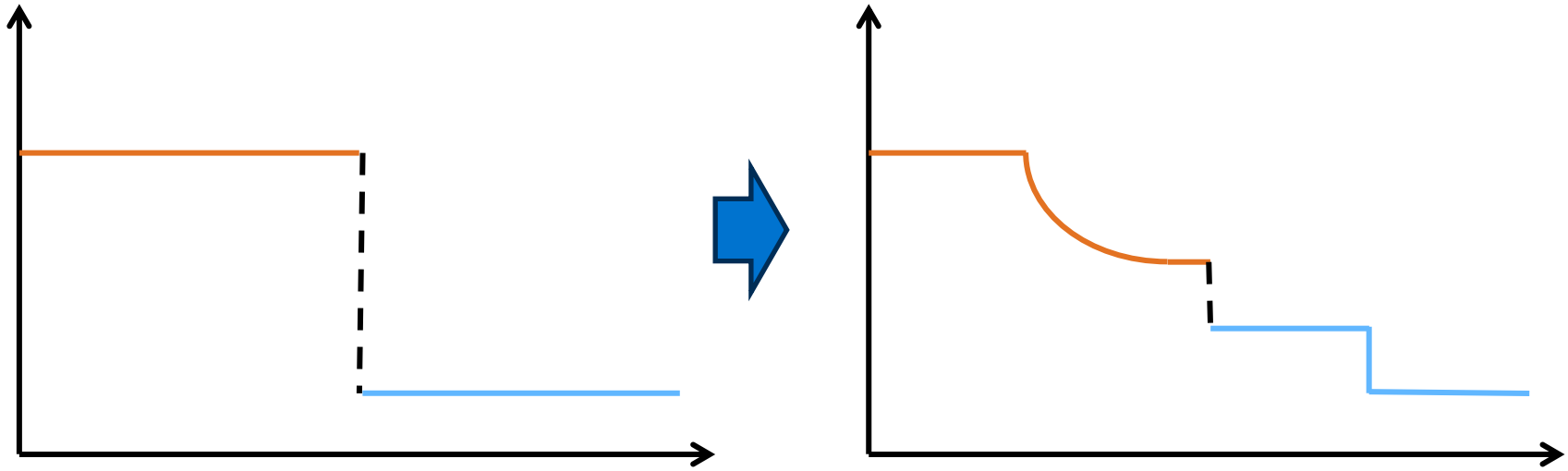


Generalising a Riemann problem to two materials

- Mathematically, this works out too

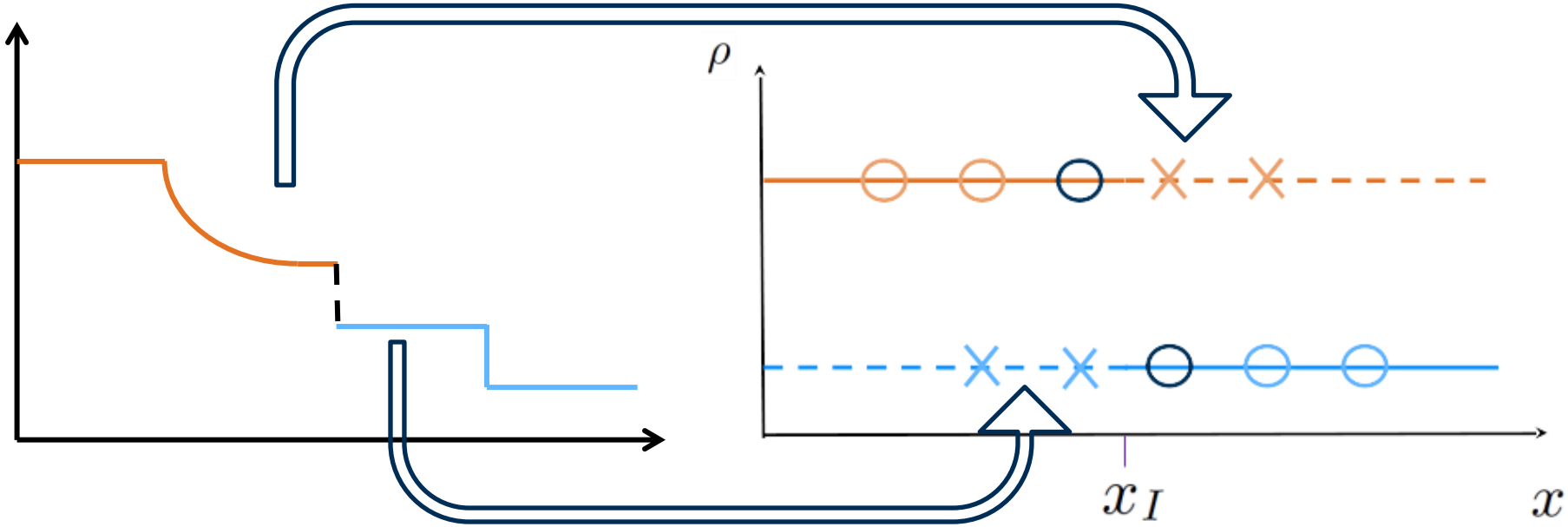
$$f(p^*, \mathbf{u}_L, \mathbf{u}_R) = g(p^*, \mathbf{u}_R) - g(p^*, \mathbf{u}_L) = f_R(p^*, \mathbf{u}_R) + f_L(p^*, \mathbf{u}_L) + \Delta v = 0,$$

- There are two functions that depend on either **only** on the left state or **only** on the right state



So what makes this a suitable boundary condition?

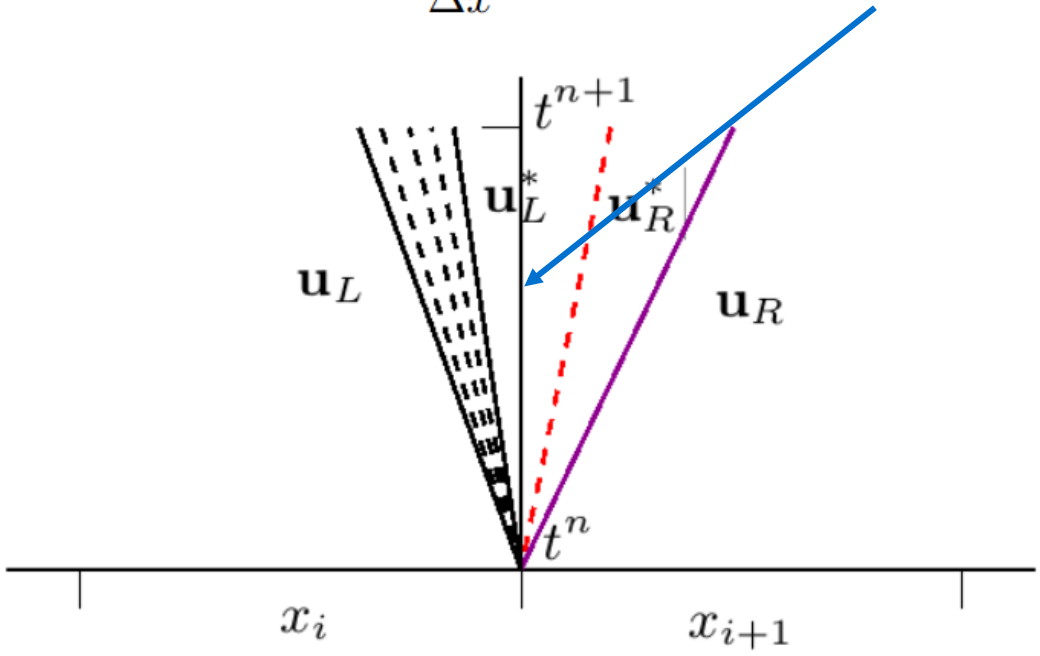
- It is clear that the solution to the Riemann problem is going to be a **physically valid state**
- But we want to consider why it is then a suitable ghost fluid state, which is slightly different, since the cell boundary doesn't move at all



What do we need from our ghost fluid method

- The aim of our boundary conditions is to ensure that we get fluxes for each material at the interface correct

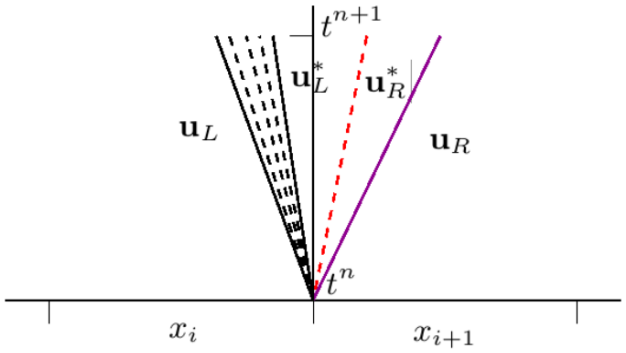
$$\mathbf{u}_i^{n+1} = \mathbf{u}_i^n + \frac{\Delta t}{\Delta x}\left[\mathbf{f}\left(\mathbf{u}_{i-1/2}^n\right) + \mathbf{f}\left(\mathbf{u}_{i+1/2}^n\right)\right]$$

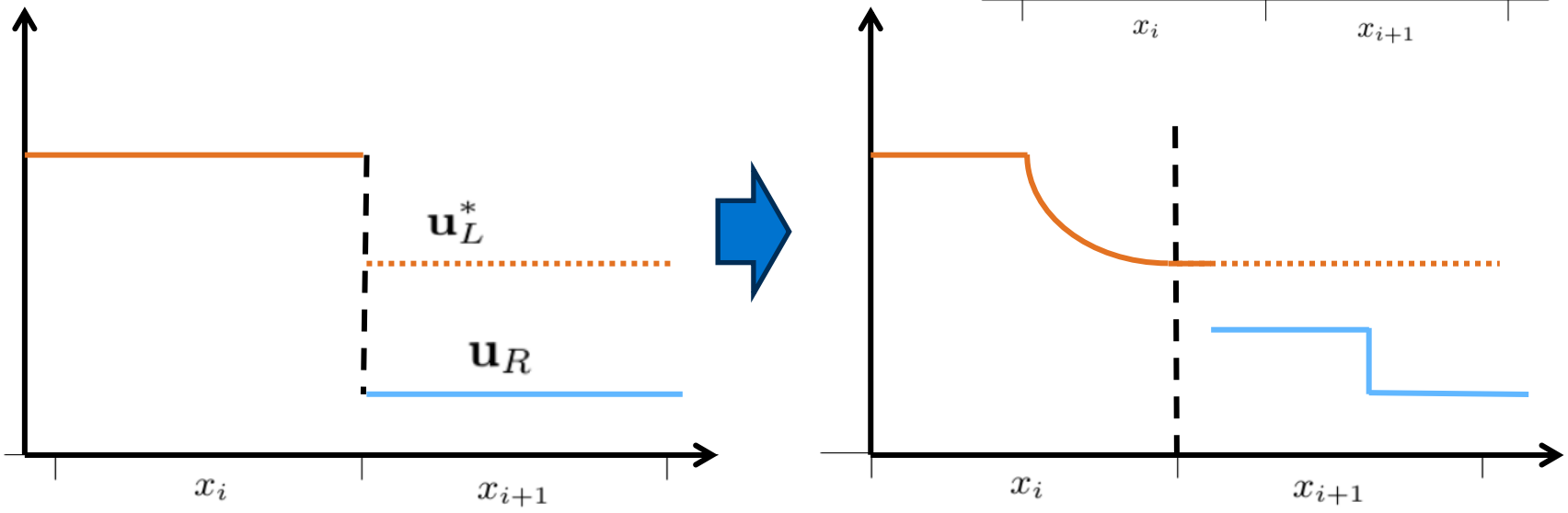


What do we need from our ghost fluid method

- Both right-hand states here produce the same left-moving wave, and the same behaviour at the cell boundary
- This ghost state is clear

$$\mathbf{u}_i^{n+1} = \mathbf{u}_i^n + \frac{\Delta t}{\Delta x}\left[\mathbf{f}\left(\mathbf{u}_{i-1/2}^n\right) + \mathbf{f}\left(\mathbf{u}_{i+1/2}^n\right)\right]$$

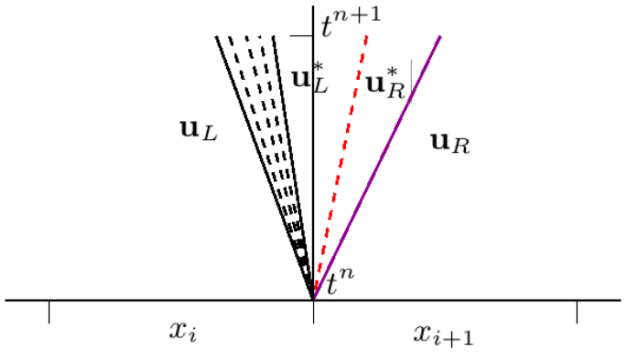


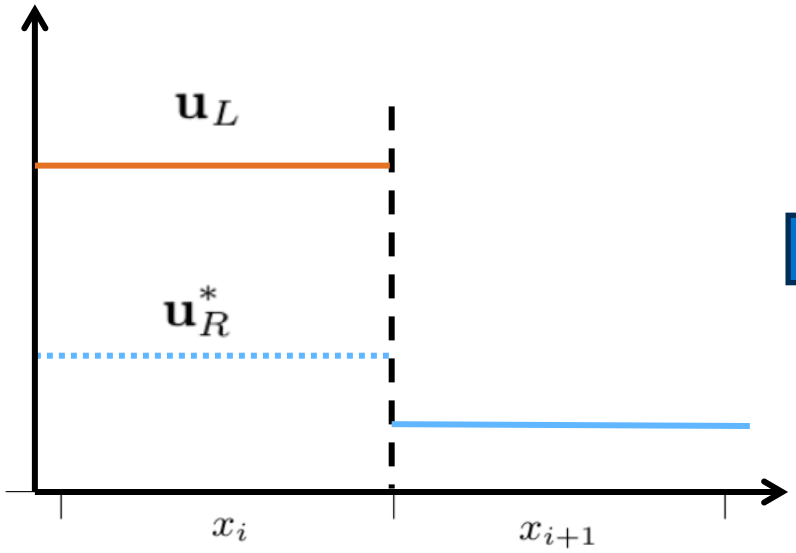


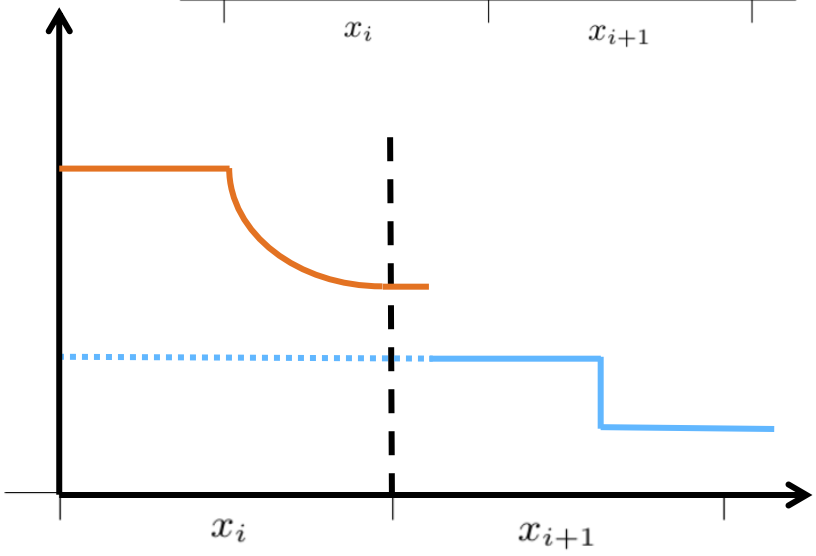
What do we need from our ghost fluid method

- Both left-hand states generate the same right-moving wave, though the state at the cell boundary is different
- Is this ok?

$$\mathbf{u}_i^{n+1} = \mathbf{u}_i^n + \frac{\Delta t}{\Delta x}\left[\mathbf{f}\left(\mathbf{u}_{i-1/2}^n\right) + \mathbf{f}\left(\mathbf{u}_{i+1/2}^n\right)\right]$$



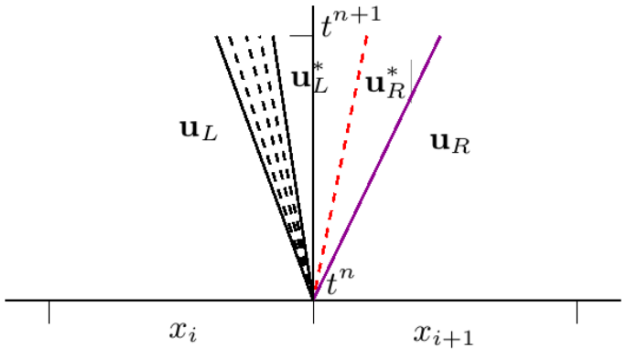


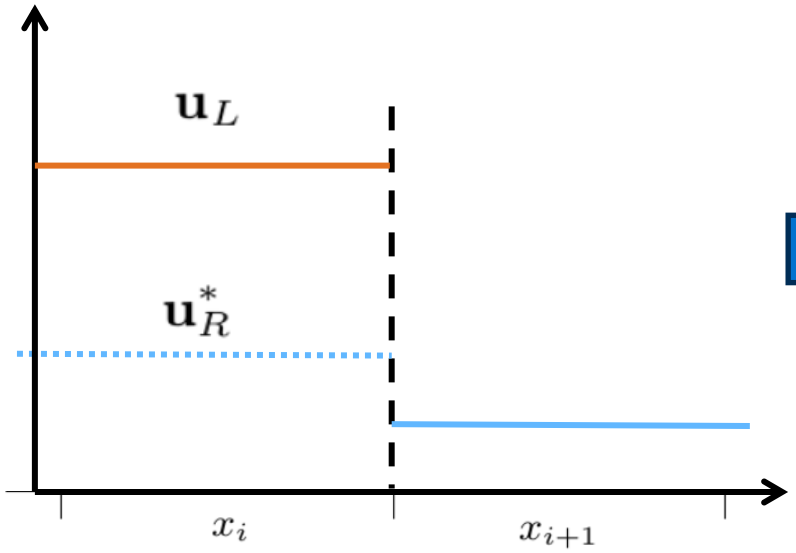


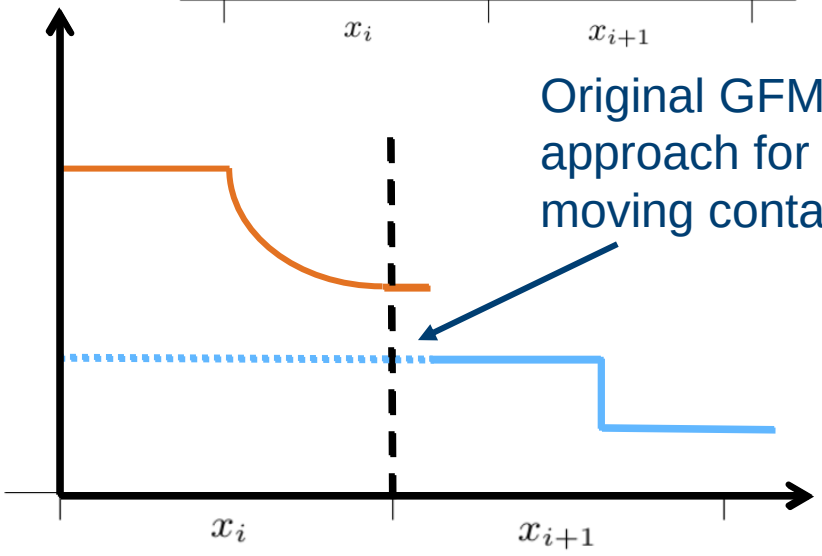
What do we need from our ghost fluid method

- Provided the left-moving wave doesn't cross the cell boundary, yes
- In this case, as well as the correct right-moving wave, we ensure the correct movement of the contact discontinuity

$$\mathbf{u}_i^{n+1} = \mathbf{u}_i^n + \frac{\Delta t}{\Delta x}\left[\mathbf{f}\left(\mathbf{u}_{i-1/2}^n\right) + \mathbf{f}\left(\mathbf{u}_{i+1/2}^n\right)\right]$$



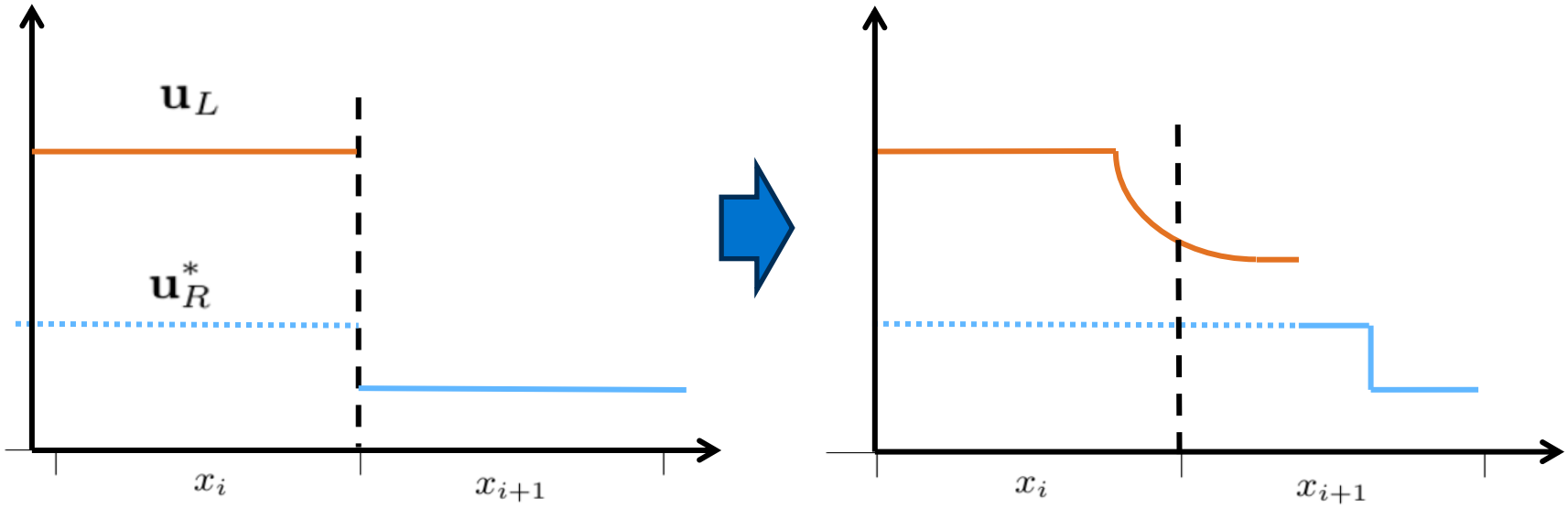




Original GFM approach for a moving contact

What do we need from our ghost fluid method

- If we have a situation where additional waves cross the boundary, we are really revealing one of the weaknesses of the single material-per-cell approach
- The practical GFM does make some attempts to rectify this, but this is still an underdetermined problem



Do we need to improve ghost fluid methods?

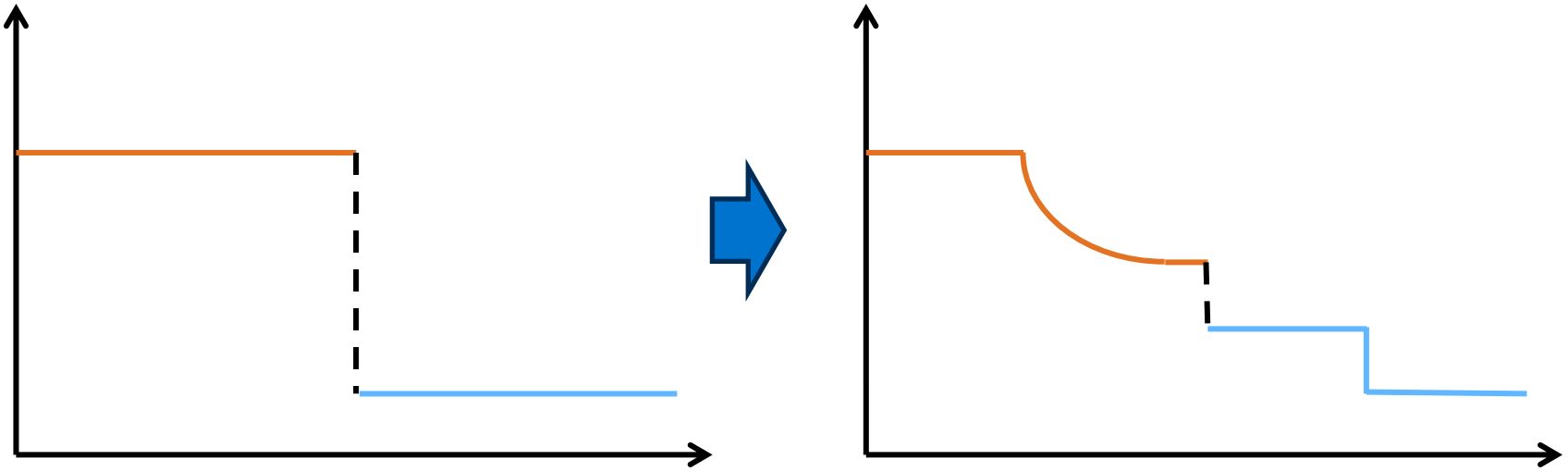
- Despite these weaknesses, Riemann problem-based ghost fluid methods have been used successfully for interfaces between solids, liquids, gases and plasmas
- And most of the issues do still converge away as resolution increases
- And the fact that the ghost fluid states are definitely thermodynamically reasonable is enough to ensure stable behaviour, just perhaps slightly changing the problem being solved
- The obvious improvement, whilst maintaining the sharp interface nature of the problem is multimaterial cut cell methods
- These are being actively developed, but are still very much research – how do you provide cut cell boundaries over a time step in which the boundary moves?

Outline

- Riemann problems and the ghost fluid method
- Mixed-material Riemann problems
- Ghost fluid methods in multiple dimensions

Mixed-material Riemann problems

$$f(p^*, \mathbf{u}_L, \mathbf{u}_R) = g(p^*, \mathbf{u}_R) - g(p^*, \mathbf{u}_L) = f_R(p^*, \mathbf{u}_R) + f_L(p^*, \mathbf{u}_L) + \Delta v = 0,$$



- We have already hinted at the fact that mixed-material Riemann problems are mathematically supported by everything we have covered so far
- And, fortunately, they are not actually any more complex to solve

Ideal gas interface REORDER ANIMATION

- The 'simplest' interface is between two ideal gases, and, for this, we can still compute the exact solution

Now have material dependence

$$f\left(p^*, \mathbf{q}_L, \right) + f\left(p^*, \mathbf{q}_R\right) + v_R - v_L = 0,$$

$$f\left(p^*, \mathbf{q}_K\right) = \begin{cases} \left(p^* - p_K\right)\left(\dfrac{A_K}{p^* + B_K}\right)^{1/2} & p^* > p_K \\ \dfrac{2c_{s,K}}{\gamma_K - 1}\left[\left(\dfrac{p^*}{p_K}\right)^{\frac{\gamma_K - 1}{2\gamma_K}} - 1\right] & p^* \leq p_K \end{cases}$$

These terms also contain γ

- The only difference between this and the method introduced in Numerical Methods for Compressible Fluid Dynamics is that the left and right states each use their own material's γ
- Additionally, the two-rarefaction solution can no longer be directly analytically obtained, an iterative solution must always be used

Stiffened gas interfaces

- Ideal gas interfaces are relatively uncommon in short-time scale sharp interface problems
- Gas-liquid or liquid-liquid interfaces are much more common, as these will often remain sharp
- It is possible to perform the derivation of the exact Riemann problem solver for the Euler equations with a stiffened gas EoS

$$p = (\gamma - 1)\,\rho\varepsilon - \gamma p_\infty$$

- Doing this only slightly changes the eventual expression

Stiffened gas interfaces

$$f\left(p^*, \mathbf{u}_K\right) = \begin{cases} \left(p^* - p_K\right)\left(\frac{A_K}{p^* + B_K}\right)^{1/2} & p^* > p_K \\ \frac{2c_{s,K}}{\gamma_K - 1}\left[\left(\frac{p^* + p_{\infty,K}}{p_K + p_{\infty,K}}\right)^{\frac{\gamma_K - 1}{2\gamma_K}} - 1\right] & p^* \leq p_K \end{cases}$$

$$A_K = \frac{2}{\left(\gamma_K + 1\right)\rho_K}, \qquad B_K = \frac{\gamma_K - 1}{\gamma_K + 1} p_K + \frac{2\gamma_K p_{\infty,K}}{\gamma_K + 1}.$$

- This is no more challenging to solve than the ideal gas version; even for a Newton-Raphson method, the derivatives are still straightforward
- In the case of a gas-liquid liquid interface, one of the materials simply needs $p_\infty = 0$

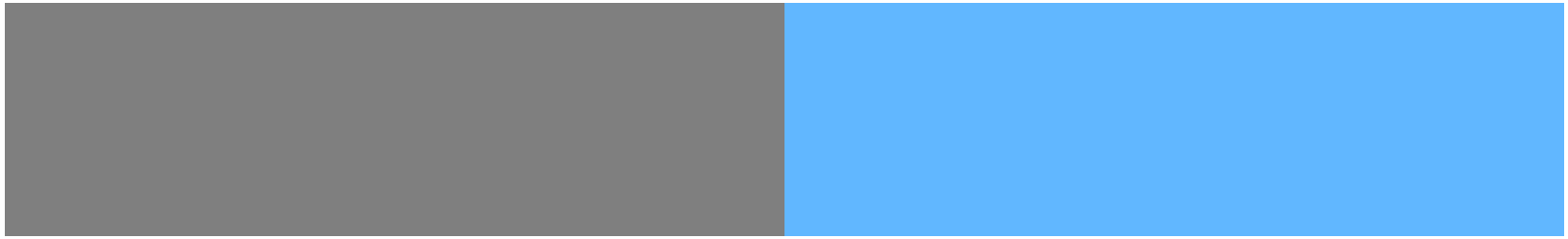
General interfaces

- As we move to more complicated materials, and systems of equations, we do not always have explicit forms relating the initial states to the star states,
- And when we do, these require solution techniques which are neither fast nor reliable for computing the star state (e.g. exact MHD solvers)
- Approximate Riemann solvers can be constructed, provided they do resolve the contact discontinuity (HLL is not a good idea)
- But, because the intermediate state is chosen only for consistency of the underlying equations, it does not guarantee the correct mixed-material behaviour
- Oscillations and larger errors are therefore often seen if attempting to use an approximate solver (based only on obeying the consistency condition)

General interfaces

- Interfaces involving solid materials are even more difficult - ‘pressure’ in a solid is defined through a stress tensor, not a scalar variable
- However, behaviour at contact discontinuities, i.e. across the interface, is known, and, providing we have a means of specifying intermediate states, we can determine boundary conditions
- An alternative approach to using exact solutions is to solve a **linearised Riemann problem**
- This uses the ideas some of the early GFM improvements, and, although technically an approximation, it attempts to approximate the actual solution, rather than the finite volume behaviour of the solution
- And once an initial guess has been made, iterative methods exist to improve the solution guess

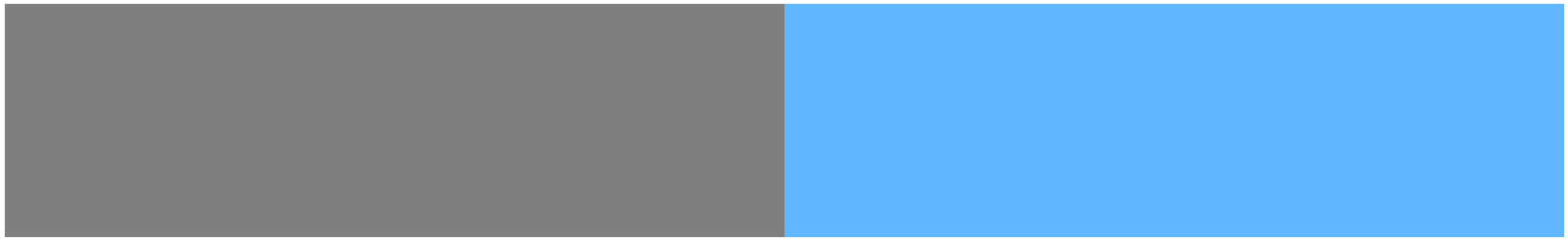
A complex-interface example



- In order to look at linearised techniques for finding mixed-material Riemann problem solutions, we shall look at a complex example
- We shall consider all the things which could make this a challenge

1. Different numbers of equations for each material
2. "Different" conserved variables for each material
3. Considering stress and pressure at the interface
4. Allowing for diffuse interface formulations at the interface

A complex-interface example



- Our first material will be an elastoplastic solid

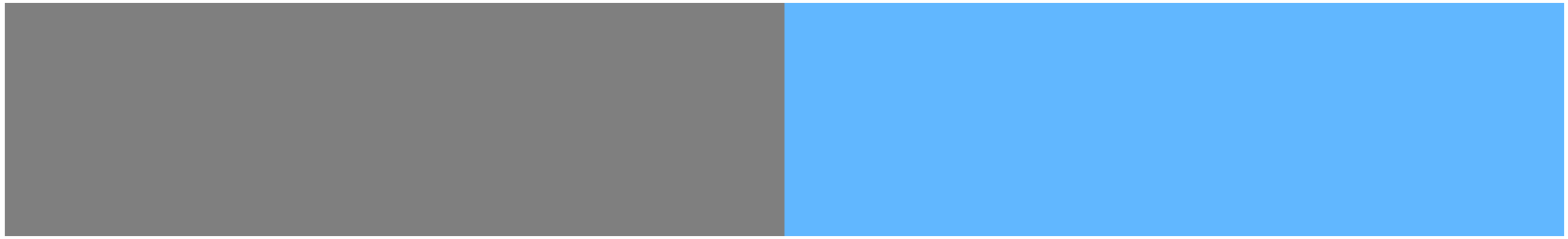
$$\frac{\partial \rho v_i}{\partial t} + \nabla_j \left(\rho v_i v_j - \sigma_{ij} \right) = 0$$

$$\frac{\partial E}{\partial t} + \nabla_j \left[v_j E + v_k \sigma_{kj} \right] = 0$$

$$\frac{\partial}{\partial t} \left(\rho F^e_{ij} \right) + \nabla_k \left(v_k \rho F^e_{ij} - v_i \rho F^e_{kj} \right) = -v_i \nabla_k \left(\rho F_{kj} \right) - \rho F^e_{ik} L^p_{kj}$$

$$L^p_{kj} = F^e_{kl} \frac{\mathrm{D} F^p_{lm}}{\mathrm{D} t} \left(F^p_{mn} \right)^{-1} \left(F^e_{nj} \right)^{-1}$$

A complex-interface example



- Our second material will be a diffuse interface between two liquids/gases, where one of them may be reactive
- Source terms are solved independently, we don't consider them

$$\frac{\partial \alpha_1 \rho_1}{\partial t} + \frac{\partial}{\partial x_k}\left(\alpha_1 \rho_1 v_k\right) = 0$$

$$\frac{\partial \alpha_2 \rho_2}{\partial t} + \frac{\partial}{\partial x_k}\left(\alpha_2 \rho_2 v_k\right) = 0$$

$$\frac{\partial \rho v_i}{\partial t} + \frac{\partial}{\partial x_k}\left(\rho v_i v_k + \delta_{ik} p\right) = 0$$

$$\frac{\partial E}{\partial t} + \frac{\partial}{\partial x_k}\left[\left(E + p\right) v_k\right] = 0$$

$$\frac{\partial \alpha_1}{\partial t} + v_k \frac{\partial \alpha_1}{\partial x_k} = 0$$

$$\frac{\partial \alpha_2 \rho_2 \lambda}{\partial t} + \frac{\partial}{\partial x_k}\left(\alpha_2 \rho_2 v_k \lambda\right) = 0$$

Intermediate state behaviour

- Our linearised solutions will not change the fact that we have a material interface
- This means it will always move with a single velocity, and the **normal velocity is constant** across the interface
- When dealing with stress tensors, we can always work out a component of the stress tensor which is normal to the interface
- This must also remain constant across the interface
- This still holds for solid-liquid or solid-gas interfaces, where the pressure is the only component of the stress tensor
- This is then sufficient for computing linearised states for any combination of materials

Linearising the solution

- Linearised solutions start from the **characteristic form** of the equations

$$\frac{\partial \boldsymbol{\mathcal{V}}}{\partial t} + \Lambda(\boldsymbol{\mathcal{V}}) \frac{\partial \boldsymbol{\mathcal{V}}}{\partial x} = 0$$

- Previously, we have said that this is not an easy form to work with for complex, nonlinear systems of equations
- However, linearising this equation means we assume $\Lambda(\boldsymbol{\mathcal{V}})$ is constant
- Clearly this is a big assumption, though in the limits of small space and time steps, it is reasonable
- Unless, of course, we have nonlinear behaviour...

Characteristic form of the mixture equations

- Recall that for obtaining the characteristic form, what we actually need is the primitive variable form of the equations, and some eigenvectors

$$\mathrm{d}\boldsymbol{\mathcal{V}} = C^{-1}\mathrm{d}\mathbf{w}$$

- Whilst we have seen each of these equations in primitive form previously, this is not the form we shall use here
- Instead, it is more convenient (it turns out) to use

$$\mathbf{w} = \left[\rho, Y, \mathbf{v}, p, \alpha_1, \lambda\right]^T$$

$$Y = Y_1 = \frac{\rho_1 \alpha_1}{\rho}$$

$$\frac{\partial \alpha_1 \rho_1}{\partial t} + \frac{\partial}{\partial x_k}\left(\alpha_1 \rho_1 v_k\right) = 0$$

$$\frac{\partial \alpha_2 \rho_2}{\partial t} + \frac{\partial}{\partial x_k}\left(\alpha_2 \rho_2 v_k\right) = 0$$

$$\frac{\partial \rho v_i}{\partial t} + \frac{\partial}{\partial x_k}\left(\rho v_i v_k + \delta_{ik} p\right) = 0$$

$$\frac{\partial E}{\partial t} + \frac{\partial}{\partial x_k}\left[\left(E + p\right) v_k\right] = 0$$

$$\frac{\partial \alpha_1}{\partial t} + v_k \frac{\partial \alpha_1}{\partial x_k} = 0$$

$$\frac{\partial \alpha_2 \rho_2 \lambda}{\partial t} + \frac{\partial}{\partial x_k}\left(\alpha_2 \rho_2 v_k \lambda\right) = 0$$

Primitive variable form

$$\frac{\partial}{\partial t}\begin{pmatrix}\rho\\ Y\\ v_x\\ v_y\\ v_z\\ p\\ \alpha_1\\ \lambda\end{pmatrix}+\begin{pmatrix}v_x & 0 & \rho & 0 & 0 & 0 & 0 & 0\\ 0 & v_x & 0 & 0 & 0 & 0 & 0 & 0\\ 0 & 0 & v_x & 0 & 0 & 1/\rho & 0 & 0\\ 0 & 0 & 0 & v_x & 0 & 0 & 0 & 0\\ 0 & 0 & 0 & 0 & v_x & 0 & 0 & 0\\ 0 & 0 & \rho c_s^2 & 0 & 0 & v_x & 0 & 0\\ 0 & 0 & 0 & 0 & 0 & 0 & v_x & 0\\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & v_x\end{pmatrix}\frac{\partial}{\partial x}\begin{pmatrix}\rho\\ Y\\ v_x\\ v_y\\ v_z\\ p\\ \alpha_1\\ \lambda\end{pmatrix}=0$$

- We now need the eigenvalues and eigenvectors of this system
- The eigenvalues are, as expected,

$$\lambda_1=\lambda_2=\lambda_3=\lambda_4=\lambda_5=\lambda_6=v_x,\quad \lambda_7=v_x-c_s\quad \lambda_8=v_x+c_s$$

Eigenvectors

- The eigenvectors for this system are also very familiar, the only three 'interesting' ones are no different to those for the Euler equations

$$\mathbf{l}_3 = \left(-c^2, 0, 0, 0, 0, 1, 0, 0\right)$$
$$\mathbf{l}_7 = (0, 0, -\rho c, 0, 0, 1, 0, 0)$$
$$\mathbf{l}_8 = (0, 0, \rho c, 0, 0, 1, 0, 0)$$

- And from these, we have characteristic variables

$$\mathbf{l}_3\colon \quad \mathrm{d}p - c_s^2 \mathrm{d}\rho = 0 \qquad \text{from} \quad \lambda_3 = v_x$$

$$\mathbf{l}_7\colon \quad \mathrm{d}p - \rho c_s \mathrm{d}v_x = 0 \qquad \text{from} \quad \lambda_7 = v_x - c_s$$
$$\mathbf{l}_8\colon \quad \mathrm{d}p + \rho c_s \mathrm{d}v_x = 0 \qquad \text{from} \quad \lambda_7 = v_x + c_s$$

- All other characteristic variables are trivial, e.g. $\mathrm{d}Y = 0$ along $\lambda_2 = v_x$

Characteristic variables

- When we first introduced these characteristic variables, we could not easily simplify things

$$\mathbf{l}_3: \quad \mathrm{d}p - \boxed{c_s^2}\mathrm{d}\rho = 0 \qquad \text{from} \quad \lambda_3 = v_x$$

$$\mathbf{l}_7: \quad \mathrm{d}p - \boxed{\rho c_s}\mathrm{d}v_x = 0 \qquad \text{from} \quad \lambda_7 = v_x - c_s$$

$$\mathbf{l}_8: \quad \mathrm{d}p + \boxed{\rho c_s}\mathrm{d}v_x = 0 \qquad \text{from} \quad \lambda_7 = v_x + c_s$$

- Under the linearised assumption, however, these terms are all constant
- As a result, the integration step now becomes trivial
- We can use this to approximate an intermediate state based on the left and right states

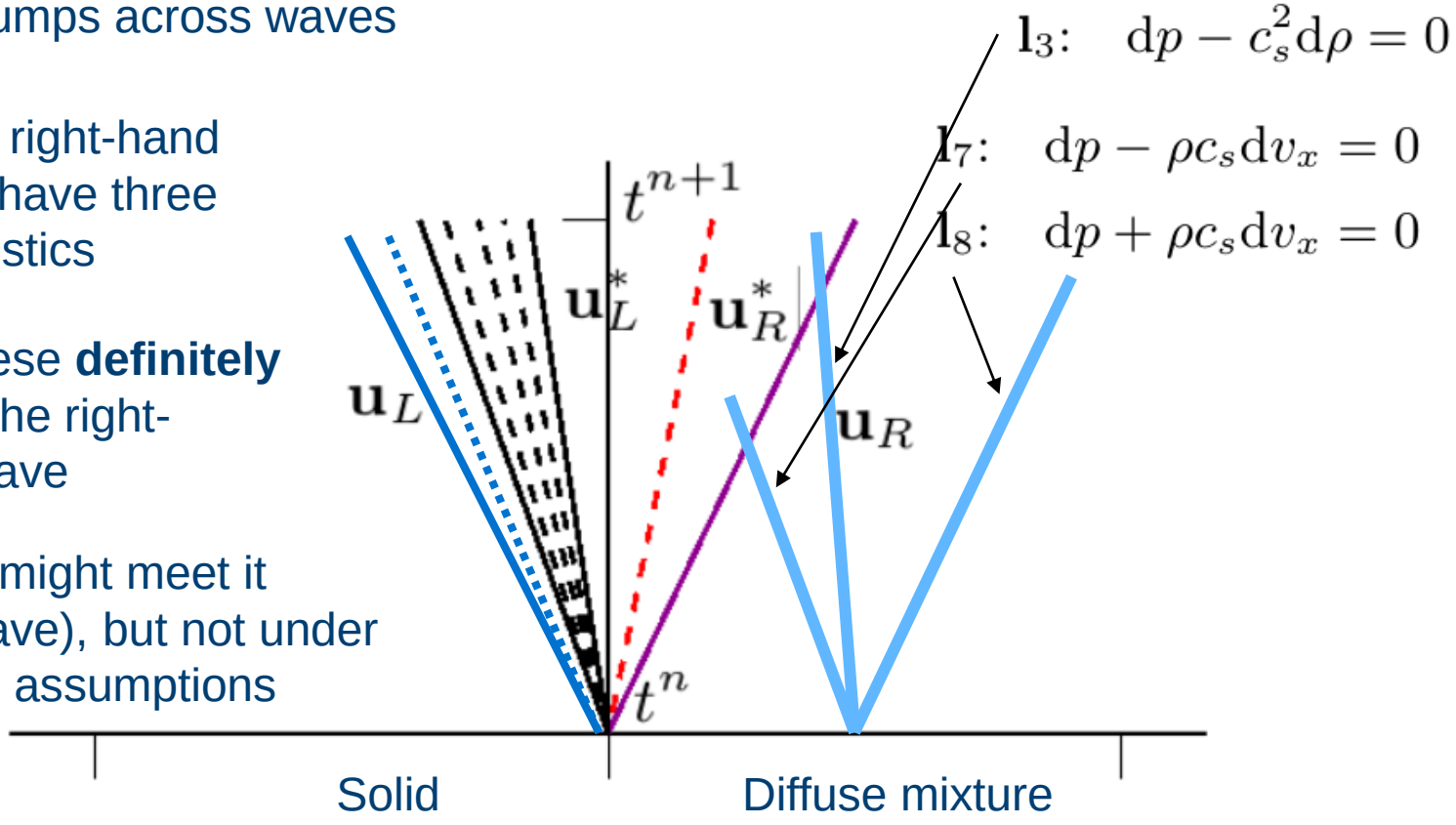
Characteristics

- Much like when deriving the solution to the exact Riemann problem, we can consider jumps across waves

- From the right-hand state, we have three characteristics

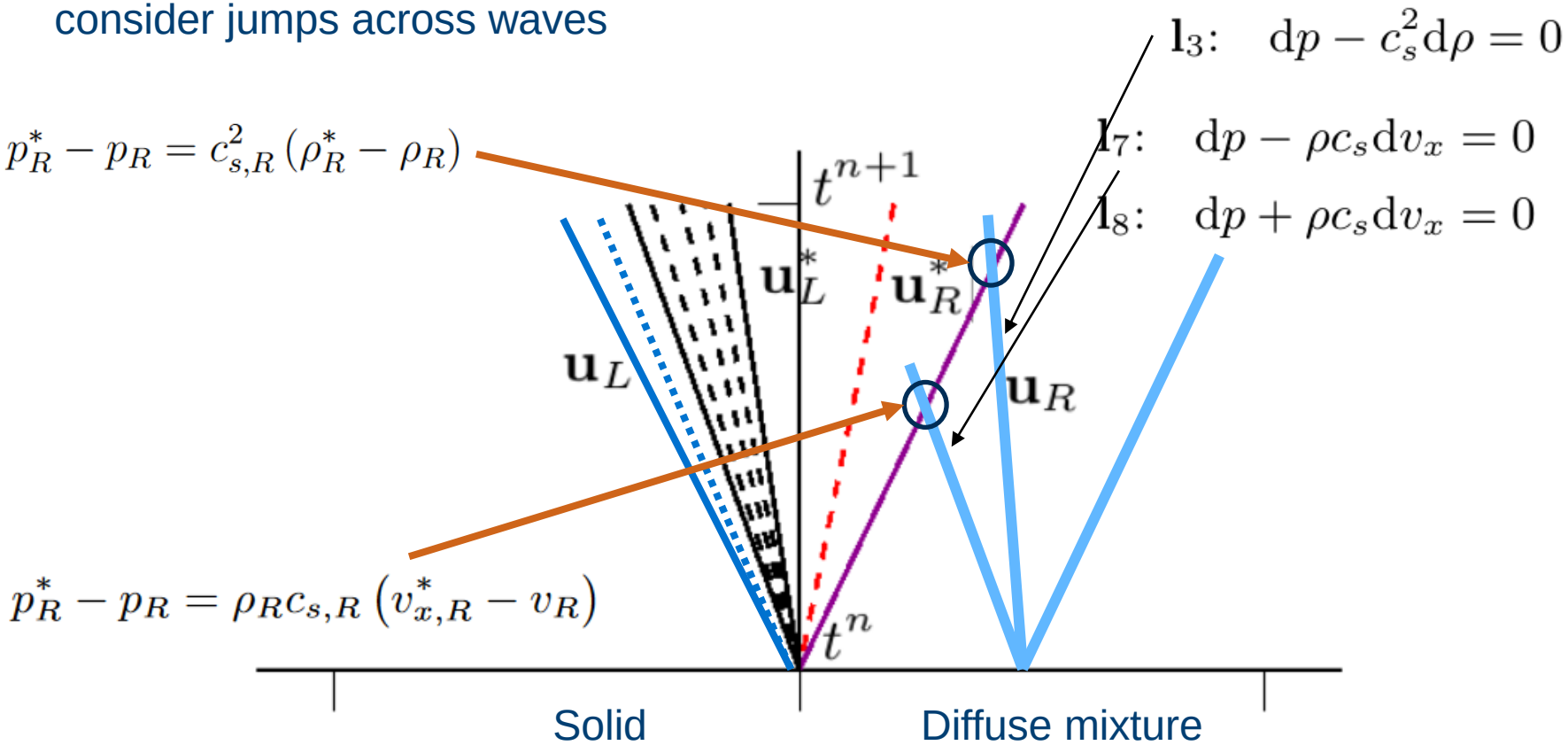
- Two of these **definitely** intersect the right-moving wave

- The third might meet it (shock wave), but not under linearised assumptions



Characteristic jumps

- Much like when deriving the solution to the exact Riemann problem, we can consider jumps across waves



Using the characteristic jumps

- Exactly as we had in the exact Riemann problem solution, we now have information about the intermediate states based on the initial states

$$p_R^* - p_R = \rho_R c_{s,R} \left(v_{x,R}^* - v_R \right) \qquad p_R^* - p_R = c_{s,R}^2 \left(\rho_R^* - \rho_R \right)$$

- And, as we would expect, across the interface, we still have

$$p_L^* = p_R^* = p^*, \; v_{x,L}^* = v_{x,R}^* = v_x^*$$

- All we need to do is find the equivalent relationships for the waves moving across the solid (and to get to the intermediate state closest to the interface)
- Clearly this could be a lot more work – worth trying an approximate Riemann solver (under caveats about physical intermediate states)?

Characteristic form for an elastoplastic solid

- We have seen that the elastoplastic solid equations can be written in primitive variable form – one reduced way of writing this is

$$\frac{\partial}{\partial t}\begin{pmatrix} \mathbf{v} \\ {F^e}^T \mathbf{e}_1 \\ {F^e}^T \mathbf{e}_2 \\ {F^e}^T \mathbf{e}_3 \\ s \end{pmatrix} + \begin{pmatrix} v_x I & -A^{\alpha 1} & -A^{\alpha 2} & -A^{\alpha 3} & -B^{\alpha} \\ -{F^e}^T E_{11} & \mathbf{v}_\alpha I & 0 & 0 & 0 \\ -{F^e}^T E_{12} & 0 & \mathbf{v}_\alpha I & 0 & 0 \\ -{F^e}^T E_{13} & 0 & 0 & \mathbf{v}_\alpha I & 0 \\ s & 0 & 0 & 0 & \mathbf{v}_\alpha I \end{pmatrix} \frac{\partial}{\partial x}\begin{pmatrix} \mathbf{v} \\ {F^e}^T \mathbf{e}_1 \\ {F^e}^T \mathbf{e}_2 \\ {F^e}^T \mathbf{e}_3 \\ s \end{pmatrix} = 0$$

$$A_{ij}^{\alpha\beta} = \frac{1}{\rho}\frac{\partial \sigma_{\alpha i}}{\partial F^e_{\beta j}}, \qquad B_i^{\alpha} = \frac{1}{\rho}\frac{\partial \sigma_{\alpha i}}{\partial s}$$

- A lot of terms within the matrix have been left in ‘matrix’ or ‘vector’ form (see Matter under Extreme Conditions for the full matrix)
- To get characteristic form, all we need to do is compute the eigenvectors…

The eigenvectors…

$$r_1 = \frac{1}{2}((Q^{-1}D^{-1})_{11}, (Q^{-1}D^{-1})_{21}, (Q^{-1}D^{-1})_{31}, F_{11}(Q^{-1}D^{-2})_{11}, F_{12}(Q^{-1}D^{-2})_{11}, F_{13}(Q^{-1}D^{-2})_{11}, F_{11}(Q^{-1}D^{-2})_{21},$$
$$F_{12}(Q^{-1}D^{-2})_{21}, F_{13}(Q^{-1}D^{-2})_{21}, F_{11}(Q^{-1}D^{-2})_{31}, F_{12}(Q^{-1}D^{-2})_{31}, F_{13}(Q^{-1}D^{-2})_{31}, 0)^T,$$

$$r_2 = \frac{1}{2}((Q^{-1}D^{-1})_{12}, (Q^{-1}D^{-1})_{22}, (Q^{-1}D^{-1})_{32}, F_{11}(Q^{-1}D^{-2})_{12}, F_{12}(Q^{-1}D^{-2})_{12}, F_{13}(Q^{-1}D^{-2})_{12}, F_{11}(Q^{-1}D^{-2})_{22},$$
$$F_{12}(Q^{-1}D^{-2})_{22}, F_{13}(Q^{-1}D^{-2})_{22}, F_{11}(Q^{-1}D^{-2})_{32}, F_{12}(Q^{-1}D^{-2})_{32}, F_{13}(Q^{-1}D^{-2})_{32}, 0)^T,$$

$$r_3 = \frac{1}{2}\left((Q^{-1}D^{-1})_{13}, (Q^{-1}D^{-1})_{23}, (Q^{-1}D^{-1})_{33}, F_{11}(Q^{-1}D^{-2})_{13}, F_{12}(Q^{-1}D^{-2})_{13}, F_{13}(Q^{-1}D^{-2})_{13}, F_{11}(Q^{-1}D^{-2})_{23},\right.$$
$$\left.F_{12}(Q^{-1}D^{-2})_{23}, F_{13}(Q^{-1}D^{-2})_{23}, F_{11}(Q^{-1}D^{-2})_{33}, F_{12}(Q^{-1}D^{-2})_{33}, F_{13}(Q^{-1}D^{-2})_{33}, 0\right)^T,$$

$$r_4 = \left(0,0,0,F_{11}\Omega_{1i}^{-1}A_{i2}^{11}, F_{12}\Omega_{1i}^{-1}A_{i2}^{11} - 1, F_{13}\Omega_{1i}^{-1}A_{i2}^{11}, F_{11}\Omega_{2i}^{-1}A_{i2}^{11}F_{12}\Omega_{2i}^{-1}A_{i2}^{11}, F_{13}\Omega_{2i}^{-1}A_{i2}^{11}, F_{11}\Omega_{3i}^{-1}A_{i2}^{11}, F_{12}\Omega_{3i}^{-1}A_{i2}^{11}, F_{13}\Omega_{3i}^{-1}A_{i2}^{11}, 0\right)^T,$$

$$r_5 = \left(0,0,0,F_{11}\Omega_{1i}^{-1}A_{i3}^{11}, F_{12}\Omega_{1i}^{-1}A_{i3}^{11}, F_{13}\Omega_{1i}^{-1}A_{i3}^{11} - 1, F_{11}\Omega_{2i}^{-1}A_{i3}^{11}F_{12}\Omega_{2i}^{-1}A_{i3}^{11}, F_{13}\Omega_{2i}^{-1}A_{i3}^{11}, F_{11}\Omega_{3i}^{-1}A_{i3}^{11}, F_{12}\Omega_{3i}^{-1}A_{i3}^{11}, F_{13}\Omega_{3i}^{-1}A_{i3}^{11}, 0\right)^T,$$

$$r_6 = \left(0,0,0,F_{11}\Omega_{1i}^{-1}A_{i2}^{12}, F_{12}\Omega_{1i}^{-1}A_{i2}^{12}, F_{13}\Omega_{1i}^{-1}A_{i2}^{12}, F_{11}\Omega_{2i}^{-1}A_{i2}^{12}F_{12}\Omega_{2i}^{-1}A_{i2}^{12} - 1, F_{13}\Omega_{2i}^{-1}A_{i2}^{12}, F_{11}\Omega_{3i}^{-1}A_{i2}^{12}, F_{12}\Omega_{3i}^{-1}A_{i2}^{12}, F_{13}\Omega_{3i}^{-1}A_{i2}^{12}, 0\right)^T,$$

$$r_7 = \left(0,0,0,F_{11}\Omega_{1i}^{-1}A_{i3}^{12}, F_{12}\Omega_{1i}^{-1}A_{i3}^{12}, F_{13}\Omega_{1i}^{-1}A_{i3}^{12}, F_{11}\Omega_{2i}^{-1}A_{i3}^{12}F_{12}\Omega_{2i}^{-1}A_{i3}^{12}, F_{13}\Omega_{2i}^{-1}A_{i3}^{12} - 1, F_{11}\Omega_{3i}^{-1}A_{i3}^{12}, F_{12}\Omega_{3i}^{-1}A_{i3}^{12}, F_{13}\Omega_{3i}^{-1}A_{i3}^{12}, 0\right)^T,$$

$$r_8 = \left(0,0,0,F_{11}\Omega_{1i}^{-1}A_{i2}^{13}, F_{12}\Omega_{1i}^{-1}A_{i2}^{13}, F_{13}\Omega_{1i}^{-1}A_{i2}^{13}, F_{11}\Omega_{2i}^{-1}A_{i2}^{13}F_{12}\Omega_{2i}^{-1}A_{i2}^{13}, F_{13}\Omega_{2i}^{-1}A_{i2}^{13}, F_{11}\Omega_{3i}^{-1}A_{i2}^{13}, F_{12}\Omega_{3i}^{-1}A_{i2}^{13} - 1, F_{13}\Omega_{3i}^{-1}A_{i2}^{13}, 0\right)^T,$$

$$r_9 = \left(0,0,0,F_{11}\Omega_{1i}^{-1}A_{i3}^{13}, F_{12}\Omega_{1i}^{-1}A_{i3}^{13}, F_{13}\Omega_{1i}^{-1}A_{i3}^{13}, F_{11}\Omega_{2i}^{-1}A_{i3}^{13}F_{12}\Omega_{2i}^{-1}A_{i3}^{13}, F_{13}\Omega_{2i}^{-1}A_{i3}^{13}, F_{11}\Omega_{3i}^{-1}A_{i3}^{13}, F_{12}\Omega_{3i}^{-1}A_{i3}^{13}, F_{13}\Omega_{3i}^{-1}A_{i3}^{13}, 0\right)^T,$$

$$r_{10} = -\left(0,0,0,F_{11}\Omega_{1i}^{-1}B_i^1, F_{12}\Omega_{1i}^{-1}B_i^1, F_{13}\Omega_{1i}^{-1}B_i^1, F_{11}\Omega_{2i}^{-1}B_i^1F_{12}\Omega_{2i}^{-1}B_i^1, F_{13}\Omega_{2i}^{-1}B_i^1, F_{11}\Omega_{3i}^{-1}B_i^1, F_{12}\Omega_{3i}^{-1}B_i^1, F_{13}\Omega_{3i}^{-1}B_i^1, -1\right)^T,$$

$$r_{11} = \frac{1}{2}\left((Q^{-1}D^{-1})_{\cdots}, (Q^{-1}D^{-1})_{\cdots}, (Q^{-1}D^{-1})_{\cdots}, F_{11}(Q^{-1}D^{-2})_{\cdots}, F_{12}(Q^{-1}D^{-2})_{\cdots}, F_{13}(Q^{-1}D^{-2})_{\cdots}, F_{11}(Q^{-1}D^{-2})_{\cdots}\right.$$

Barton, Drikakis, Romenski and Titarev (2009)

Obtaining intermediate values

- At this point, we are extremely fortunate someone has done the work for us
- Jumps in the primitive variables across the waves are given by (see Barton and Drikakis (2010))

$$\mathbf{w}_L^* - \mathbf{w}_L = \frac{1}{\rho_L} \sum_{i=1}^{3} \hat{\mathbf{r}}_{10+i} \left(\sigma_{L,1i}^* - \sigma_{L,1i} \right)$$

Three of the thirteen eigenvectors

Stress components (assuming the interface is aligned with the x-axis)

- The jump in all normal stress components needs to be specified to be able to compute boundary condition states
- Only non-trivial complicated for solid-solid interactions, otherwise

$$\sigma_{L,11}^* = \sigma_{R,11}^*, \quad \sigma_{L,12}^* = \sigma_{R,12}^* = \sigma_{L,13}^* = \sigma_{R,13}^* = 0$$

Intermediate pressure

- We can, eventually, get to a single intermediate pressure (see Michael and Nikiforakis (2018) for more details)

$$p^* = \frac{v_L - v_R + \frac{1}{\rho_L}\Omega_{L,11}\sigma_{L,11} + \frac{p_L}{\rho_L c_{s,L}}}{\frac{1}{\rho_L}\Omega_{L,11}\sigma_{L,11} - \frac{1}{\rho_L c_{s,L}}}$$

- Here, we have the elastic acoustic tensor

$$\Omega_{ij} = \frac{1}{\rho}\frac{\partial \sigma_{1i}}{\partial F^e_{jk}} F^e_{1k}$$

- And for the solid

$$\sigma^*_{R,11} = -p^*$$

- So, although the equations are messy for this combination, the linearised method does provide an explicit calculation of intermediate values

Using linearised solvers

- This technique allows us to derive boundary conditions for arbitrary interfaces (for any new system, “just” compute the eigenvectors and values)
- Clearly, there is a large assumption behind this approach – the primitive variable matrix is constant
- For ‘mild’ problems, this assumption is suitably accurate to give good results – therefore useful for testing the implementation of a new system
- For ‘extreme’ problems, the linearised solver may fail; the estimation of intermediate states around shock waves could be entirely wrong, and oscillations or unphysical states occur
- In this case, we want to relax the assumption that $A(\mathbf{w}) = \text{const}$

Using linearised solvers

$$p_R^* - p_R = \rho_R c_{s,R} \left(v_{x,R}^* - v_R \right) \qquad p_R^* - p_R = c_{s,R}^2 \left(\rho_R^* - \rho_R \right)$$

- The simplest extension of the linearised approach is to slightly restrict the assumption that the primitive variable matrix itself is constant
- Instead it is some form of average of initial and intermediate states; for these expressions we could have

$$p_R^* - p_R = \overline{\rho_R c_{s,R}} \left(v_x^* - v_R \right) \qquad p_R^* - p_R = \overline{c_{s,R}^2} \left(\rho_R^* - \rho_R \right)$$

$$\overline{\rho_R c_{s,R}} = \frac{1}{2} \left(\rho_R c_{s,R} + \rho_R^* c_{s,R}^* \right), \qquad \overline{c_{s,R}^2} = \frac{1}{2} \left(c_{s,R}^2 + (c_{s,R}^*)^2 \right)$$

- Unfortunately, we don't have a closed expression any more, and we need a (relatively straightforward) iterative approach, see Schoch and Nikiforakis (2013) for details

Outline

- Riemann problems and the ghost fluid method
- Mixed-material Riemann problems
- Ghost fluid methods in multiple dimensions

Considerations in multiple dimensions

- So far, we have only considered one-dimensional interfaces
- In more than one dimension, we cannot expect our interface to be aligned with the numerical grid
- In this case, we have additional considerations:

  1. Only normal velocity (and stress) is continuous across an interface – how do we enforce this?
  2. If we are using a Riemann problem-based method, for a given left state, how do we determine what the adjacent right state is?
  3. How do we propagate information into the ghost fluid region?

Obtaining the normal components

- In order to obtain the velocity normal to the interface, we need to know what direction the normal points in, which, fortunately, is straightforward

$$\hat{\mathbf{n}} = \frac{\nabla \phi}{|\nabla \phi|}$$

- The normal component of the velocity is then

$$v_n = \hat{\mathbf{n}} \cdot \mathbf{v}$$

- The normal component of a vector is a scalar, and similarly, the normal component of a rank-two tensor is a vector
- For example, we may also need the normal component of stress at an interface

$$\sigma_i^n = \sigma_{ij} n_j$$

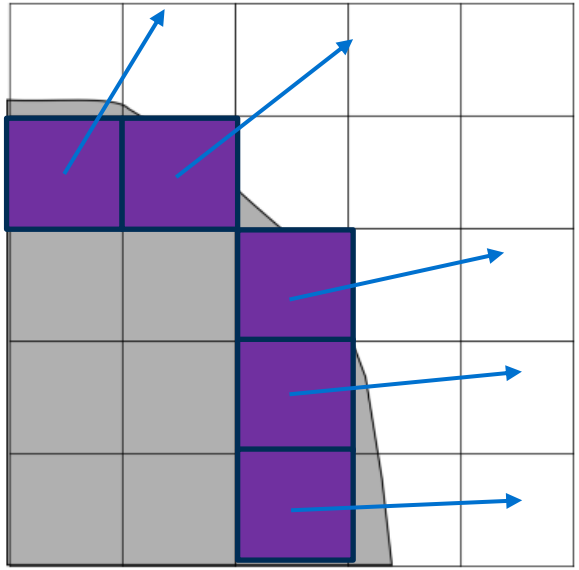
Obtaining the tangential components

- Once we move to multiple dimensions, we need to provide boundary conditions for tangential components at the interface, not just normal components
- As we saw when considering the solutions to Riemann problems in more than 1D, the intermediate states for tangential components were trivial, since they only jump across the contact discontinuity
- This doesn't change, now they only jump across the interface
- We **could** use our normal vector to decompose our full velocity (or stress) field into a coordinate system where one component is normal to the interface
- This would be a lot of work, though, and is not necessary; instead we maintain our regular Cartesian condition, and simply subtract normal contributions

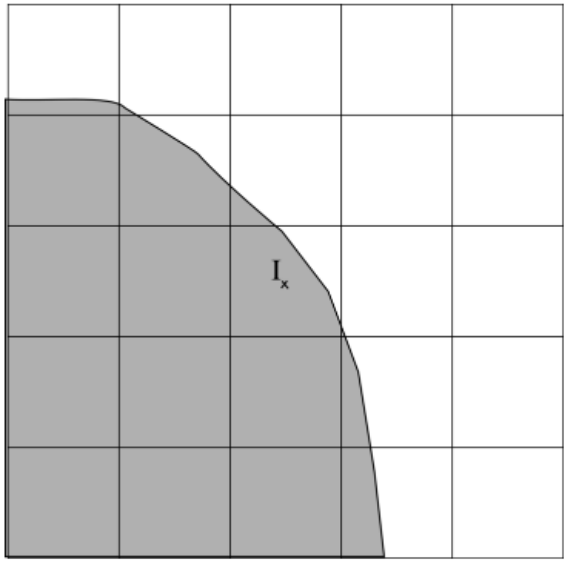
$$\mathbf{v}_t = \mathbf{v} - v_n \hat{\mathbf{n}} \qquad \sigma_{ij}^t = (\sigma_{ik} n_k) n_j$$

Basic ghost fluid in multiple dimensions (part 1)

- The basic ghost fluid is relatively easy to extend to multiple dimensions
- The copying of the pressure is entirely unchanged from 1D, and the copying of velocity becomes the copying of **normal** velocity
- The local normal direction can be used for all velocity copying – close to the interface, this is sufficiently accurate (and far away it doesn't matter)
- Entropy must now be extrapolated using some extrapolation algorithm (we shall come to this)
- The full tangential velocity vector must also be extrapolated in the same way



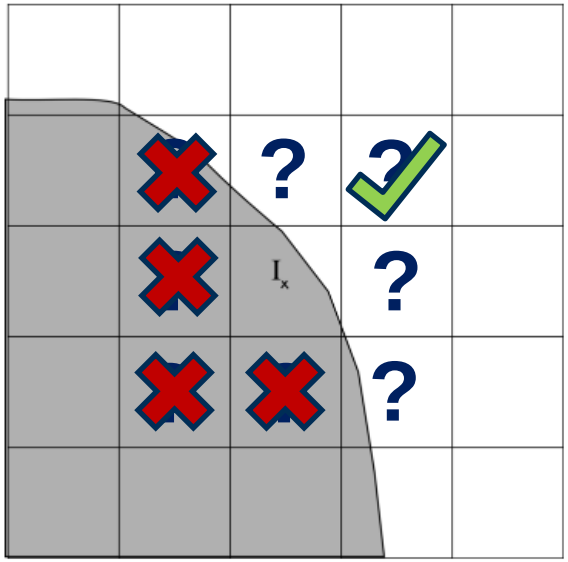
Riemann problem-based ghost fluid methods



- We need to be able to set up a Riemann problem between two states either side of the interface
- First we identify a cell adjacent to the interface
- As with reinitialisation techniques, this is done by comparing the sign of the level set function in adjacent cells

- In this example, we have found a cell with two adjacent cells in the other material
- Is either of these a good choice for the Riemann problem state?

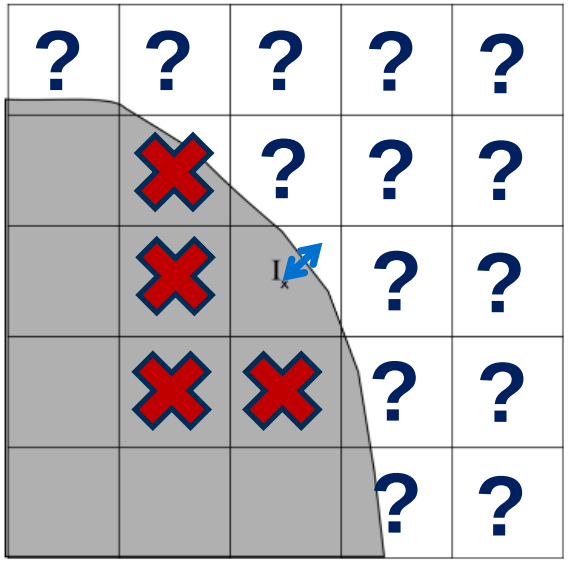
The real ghost fluid method



- This is where we see some differences in the Riemann problem-based ghost fluid methods
- The real GFM will attempt to pick the single, ‘most suitable’ cell from all adjacent cells
- In this 2D example, we have eight possible choices

- Clearly, any cell which is the same material can be ignored
- The cell which then aligns most closely with the normal vector is chosen
- Note that this cell is not actually adjacent to the interface itself – **every cell must be considered individually**

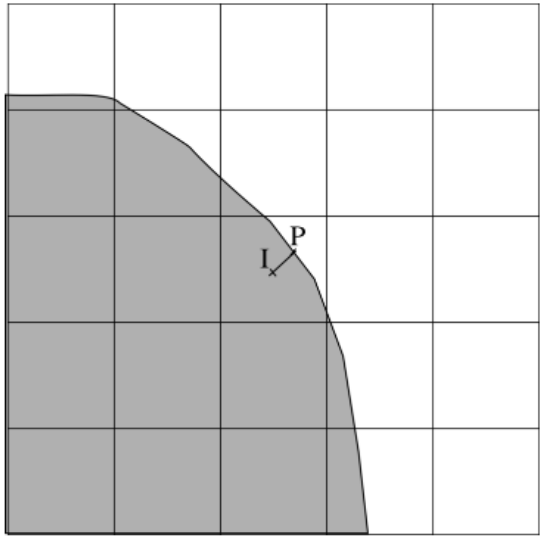
The Riemann ghost fluid method



- One potential issue with the real GFM is the two states for the Riemann problem are a different distance from the interface
- The Riemann GFM attempts to correct for this, but this requires using more than one cell
- Therefore information could come from any cell suitably close to the interface (within two cells – everything in this example!)

- We could simply use information about the normal direction to find a position about which to interpolate
- However, in this example, the interpolation should use four surrounding cells – here there are only three in the correct material

Interpolating neighbouring cells



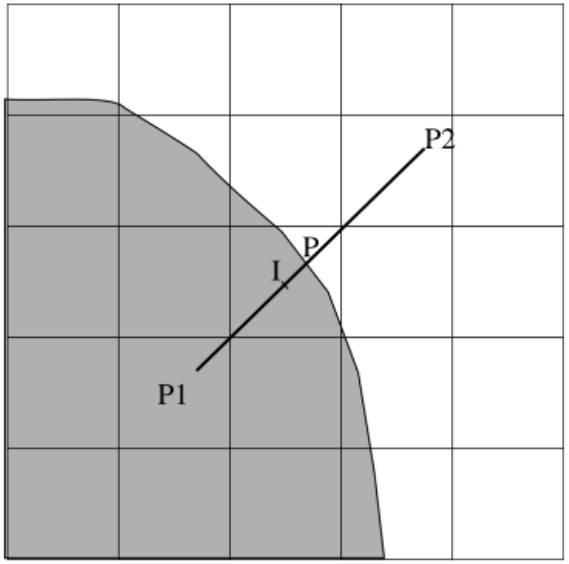
- We now aim to find two interpolated states, equidistant from the interface
- Therefore we first need to find a point on the interface, and the closest point to our cell
- Fortunately, the use of signed distance functions is once again an advantage – this tells us both the distance to the interface, and we can compute the direction (the normal)

- Therefore we simply have a location

$$\mathbf{x}_P = \mathbf{x}_I - \phi\hat{\mathbf{n}}$$

- Note – this has made an assumption that the normal vector is pointing **into** the grey region

Interpolating neighbouring cells

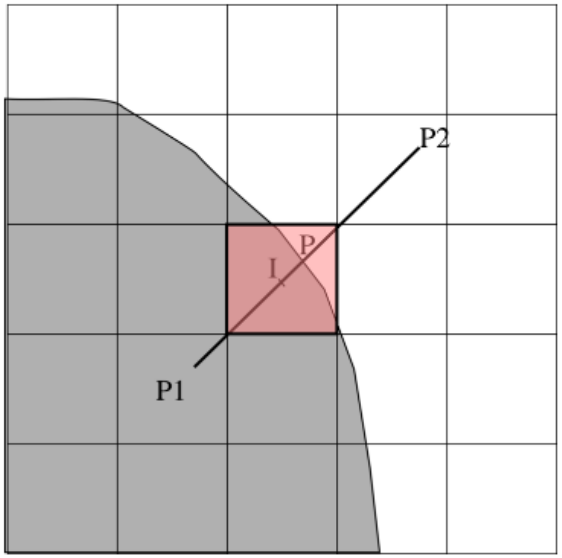


- Once we've identified our interface position, we now use this to compute two positions equidistant from the interface **along the same normal vector**
- This means that for the Riemann GFM, both initial states must be interpolated
- We now have two positions, each a distance δ from the interface

- Ideally, this distance should be the shortest such that both materials have an interpolated state from four valid cells (whilst not being too far away)
- In practice, it is common to use $\delta = 1.5\Delta x$, as this usually gives suitable points

$$\mathbf{x}_{P1,P2} = \mathbf{x}_P \pm 1.5\Delta x \hat{\mathbf{n}}$$

Obtaining the Riemann problem solution



- In order to interpolate the states, a bilinear interpolation is sufficient (higher order puts you too far from the interface)
- Once we have these states, the normal vector can be used to rotate them normal to the interface
- And then we solve the mixed-material Riemann problem

- Here, we are finding a ghost fluid state for the white material (from P2)
- In practice, we always set this material as the **left material**, and then use the left star state (the same applies to the real GFM)
- This is slightly different from how the original Riemann GFM was implemented, but a more natural extension to multiple materials

Solving the Riemann problem

- In order to obtain our Riemann problem solution, we could rotate all vectors and tensors to be normal to the interface, e.g.

$$\left(v_x, v_y, v_z\right)_L, \left(v_x, v_y, v_z\right)_R \rightarrow \left(v_n, v_{t1}, v_{t2}\right)_L, \left(v_n, v_{t1}, v_{t2}\right)_R$$

- The Riemann problem solution would then include velocities

$$\left(v^*, v_{t1,L}, v_{t2,L}\right), \left(v^*, v_{t1,R}, v_{t2,R}\right)$$

- However, as mentioned earlier, we can do this whilst avoiding identifying what the tangential vectors are

$$\left(v_x, v_y, v_z\right)_L, \left(v_x, v_y, v_z\right)_R \rightarrow \left(v_n, \mathbf{v} - v_n\mathbf{n}\right)_L, \left(v_n, \mathbf{v} - v_n\mathbf{n}\right)_R$$

- And then we get star-state velocities

$$\left(v^*, \left(\mathbf{v} - v_n\mathbf{n}\right)_L\right), \left(v^*, \left(\mathbf{v} - v_n\mathbf{n}\right)_R\right)$$

- This may require very slight changes to existing Riemann solvers

Populating the ghost cells

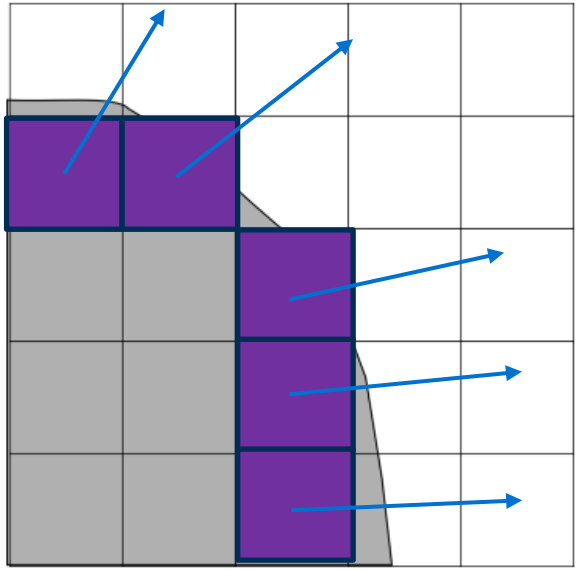
- All the multidimensional techniques so far have considered how we set values in cells **adjacent** to the interface
- Now we need to consider extrapolating these values
- For the basic ghost fluid method, we compute interface quantities which we use to compute ghost fluid values

$$\frac{p_I}{(\rho_I)^{\gamma_I}} = \frac{p_G}{(\rho_G)^{\gamma_I}} \qquad \mathbf{v}_G = v_{n,G}\mathbf{n} + \mathbf{v}_{t,I}$$

These come from the real material

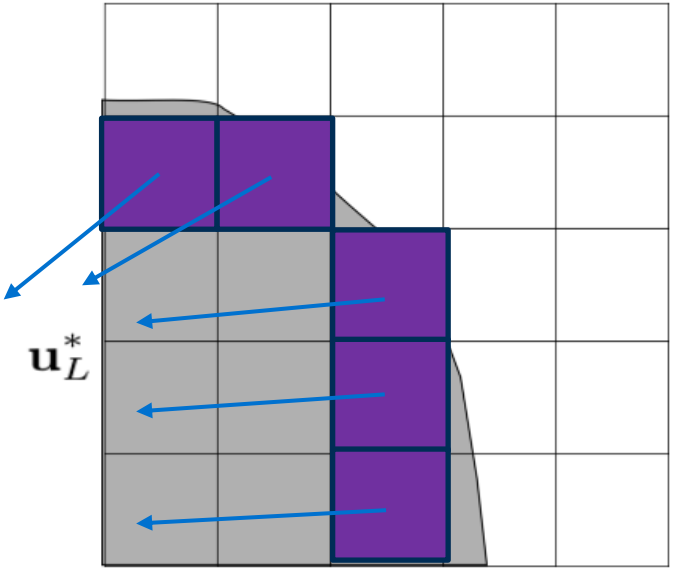
Extrapolate these quantities

And use them to compute these



Populating the ghost cells

- All the multidimensional techniques so far have considered how we set values in cells **adjacent** to the interface
- Now we need to consider extrapolating these values
- For Riemann problem-based ghost fluid methods, we have the entire intermediate state for the first line of ghost cells
- We don’t need to compute anything further, just extrapolate



Populating the ghost cells

- In order to fill ghost fluid boundary states, we need to perform a **constant extrapolation**
- We consider an arbitrary variable, Q, which needs extrapolating – this could be any of the quantities from either type of ghost fluid method
- Constant extrapolation is the solution to a boundary-value problem

$$\hat{\mathbf{n}} \cdot \nabla Q = 0$$

- There are similarities here to the Eikonal equation, which could be written

$$\hat{\mathbf{n}} \cdot \nabla \phi = 1$$

- As a result, the techniques for solving this equation are equally similar, and require very little modification from a reinitialisation procedure

Populating the ghost cells

- Although the techniques are similar, we are not solving an Eikonal equation

$$\hat{\mathbf{n}} \cdot \nabla Q \neq |\nabla Q|$$

- Here, the information about the normal vector comes from the level set function, which we already know
- However, this only makes things easier, and we can still use iterative, fast marching or fast sweeping methods
- To briefly mention fast marching methods, before considering the other two approaches:
- If we already know the level set function, we already know the order to which we march through the domain, and all derivatives are derivatives of Q

Populating the ghost cells – iterative approach

$$\hat{\mathbf{n}} \cdot \nabla \phi = 1$$

$$\frac{\partial \phi}{\partial \tau} + \mathrm{sgn}\,(\phi)\,(|\nabla \phi| - 1) = 0$$

$$\hat{\mathbf{n}} \cdot \nabla Q = 0$$

$$\frac{\partial Q}{\partial t} + \mathrm{sgn}(\phi)\hat{\mathbf{n}} \cdot \nabla Q = 0$$

- As before, all derivatives are first order upwinded calculations
- However, here, upwind means in the direction of the interface, so is still based on the level set function, not Q
- Again, only ‘enough’ iterations need to be performed, Fedkiw et al. say about 20 iterations is sufficient for boundary conditions

Populating the ghost cells – fast sweeping

$$\hat{\mathbf{n}} \cdot \nabla \phi = 1$$

$$\left(\frac{\phi_{i,j,k} - \phi_x}{\Delta x} \right)^2 + \left(\frac{\phi_{i,j,k} - \phi_y}{\Delta y} \right)^2 + \left(\frac{\phi_{i,j,k} - \phi_z}{\Delta z} \right)^2 = 1$$

$$\hat{\mathbf{n}} \cdot \nabla Q = 0$$

$$n_x \left(\frac{Q_{i,j,k} - Q_x}{\Delta x} \right) + n_y \left(\frac{Q_{i,j,k} - Q_y}{\Delta y} \right) + n_z \left(\frac{Q_{i,j,k} - Q_z}{\Delta z} \right) = 0$$

- Again, the level set is used to define the upwinded coefficients, e.g. Q_x
- And in this case, rather than solving a quadratic, this is a simple linear equation

Going beyond two materials

- Everything we have considered so far has been a multiphysics method for two materials (though we have combined some of these)
- This is not a restriction of these methods, just a convenience
- By including additional equations for volume fractions, densities (and maybe momenta and energies), diffuse interface methods can be trivially extended
- Mixture rules may become trickier, and identifying primitive variables can become a challenge which may require iterative root finding
- Sharp interface methods simply need an extra level set equation and material system of equations for each material
- Care must be taken to ensure each cell in the domain belongs to **a unique material** – level set fix-up algorithms exist to ensure this