UNIVERSITY OF CAMBRIDGE

Mesh generation for continuum modelling (Part 2)

Dr. Nandan Gokhale, 2023

MPhil in Scientific Computing
Centre for Scientific Computing
Cavendish Laboratory

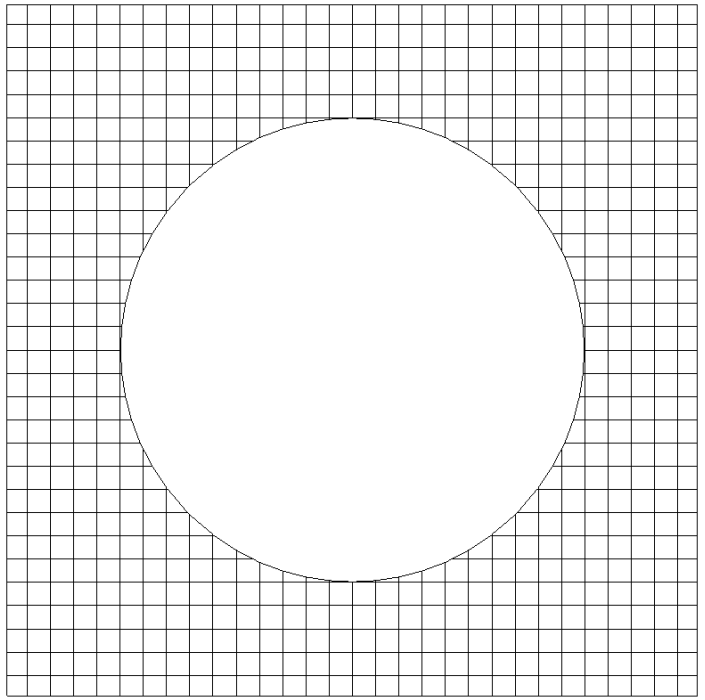
Structure of this lecture

- Introduction
- Solutions to the small cell problem
- The flux stabilisation cut cell method
- Summary and conclusions

UNIVERSITY OF CAMBRIDGE

Introduction

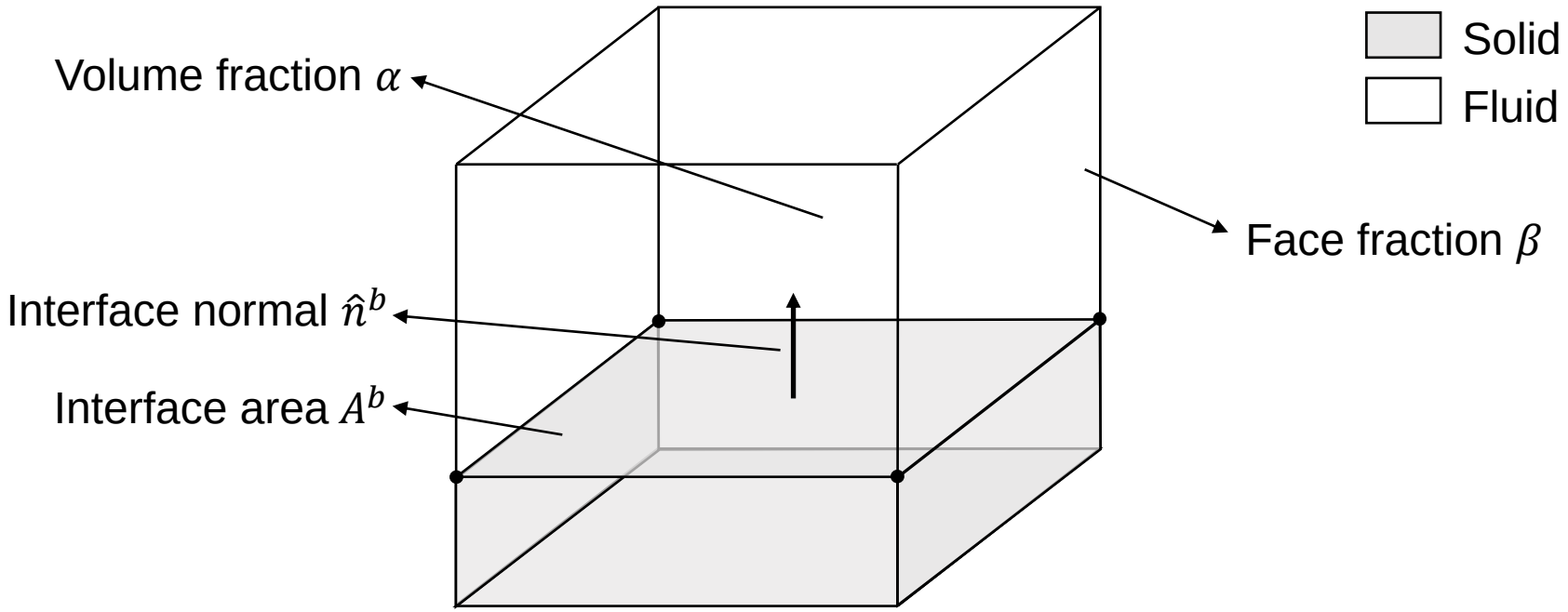
Recall



- Cut cell meshes are generated by ‘cutting out’ the geometry from a background Cartesian grid.
- If computing the solution involves solving a linear system, the equations should be discretised so that the matrix is well-conditioned.
- We will focus on cut cell methods for equations that are being integrated explicitly (hyperbolic or explicitly integrated parabolic parts of systems), where the cut cells constrain the explicit Δt (small cell problem).

UNIVERSITY OF CAMBRIDGE

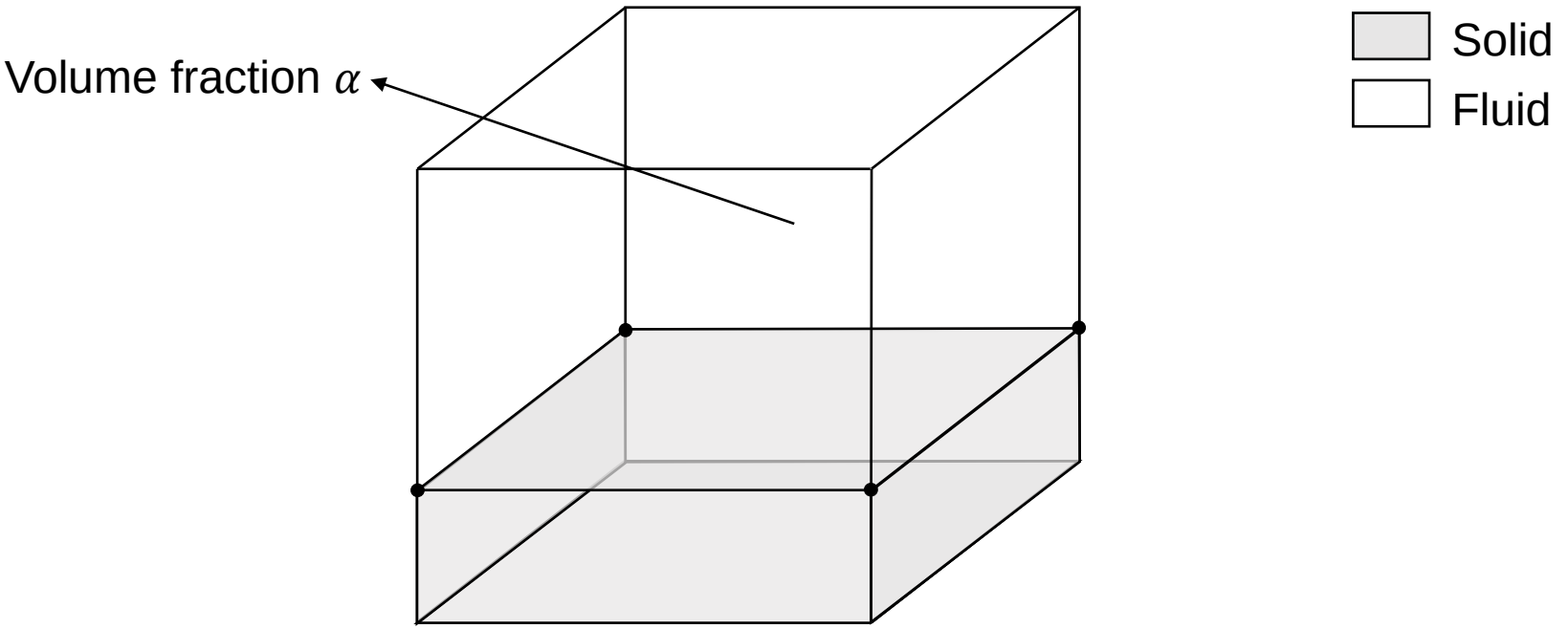
Geometric parameters describing a cut cell



- The materials on either side of the interface do not have to be a fluid and a solid, but we will assume this for convenience when defining the parameters.

UNIVERSITY OF CAMBRIDGE

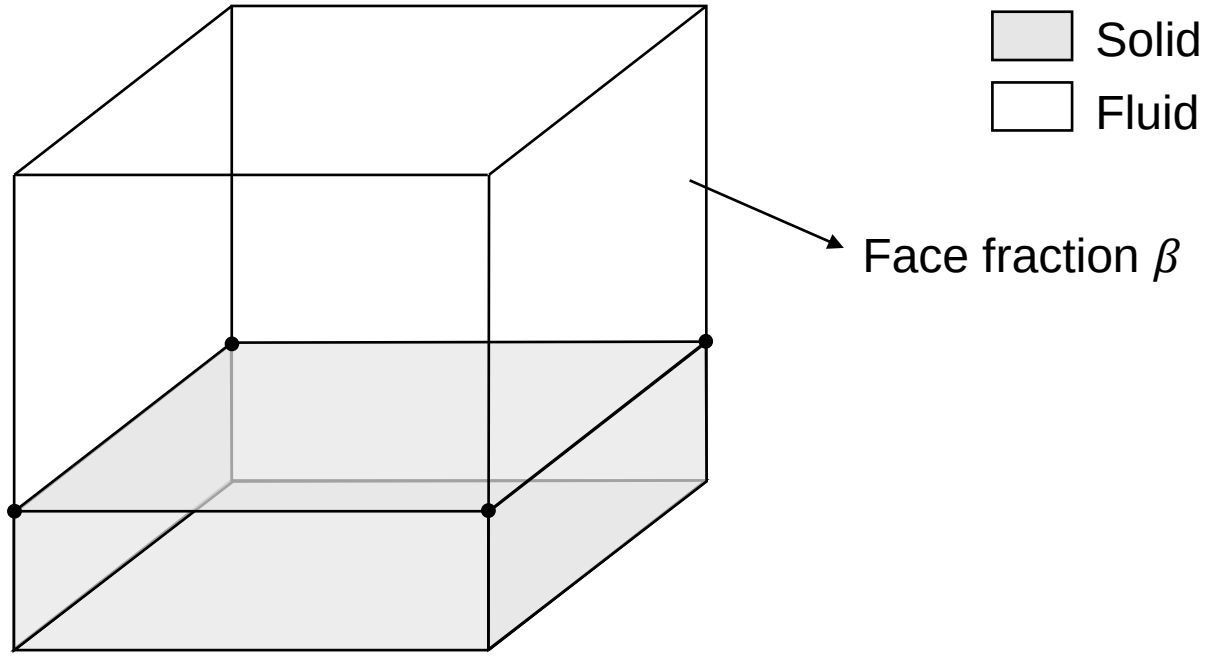
Geometric parameters describing a cut cell, α



- The volume fraction, $\alpha \in [0,1]$, of the cell is the fluid volume of the cell non-dimensionalised by the total cell volume.

UNIVERSITY OF CAMBRIDGE

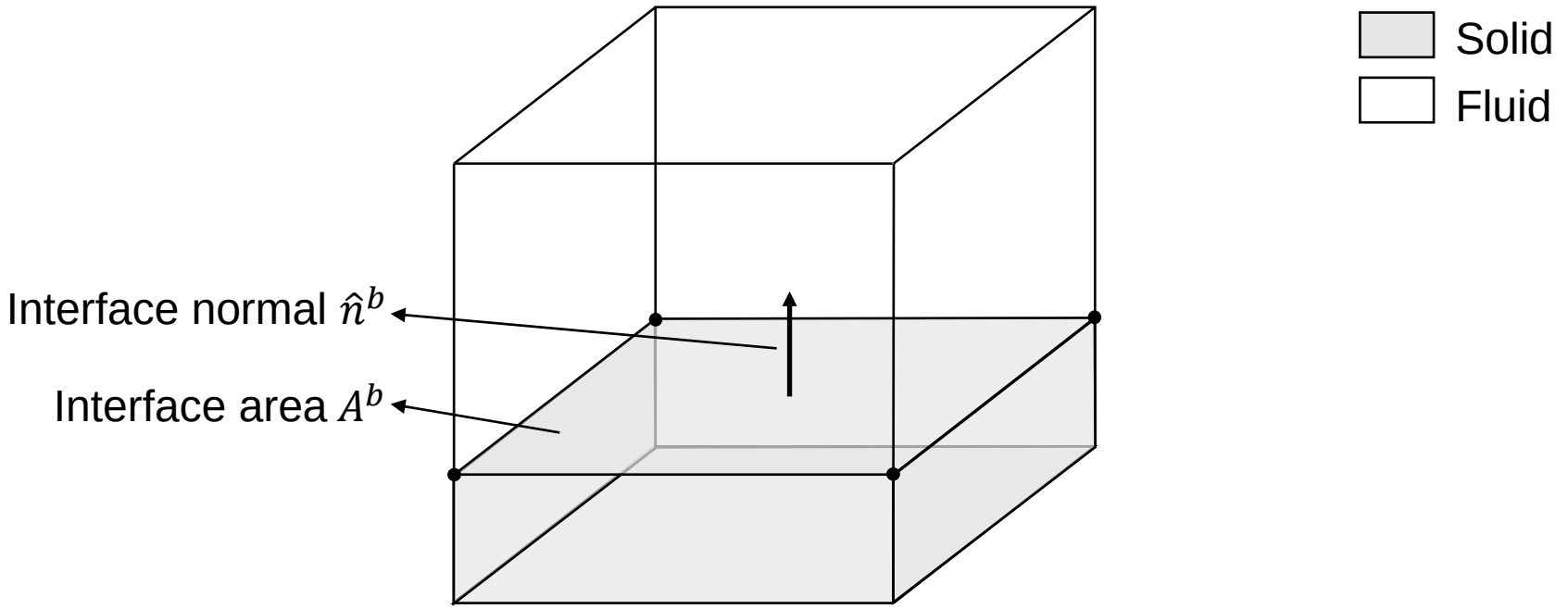
Geometric parameters describing a cut cell, β



- The face fraction, $\beta \in [0,1]$, of a cell face represents the fluid area of the face non-dimensionalised by total cell face area.
- Note: It may be convenient to split β into different parts (all of which will sum up to β). We will use this idea extensively when studying the properties of a cut cell method later in the lecture.

UNIVERSITY OF CAMBRIDGE

Geometric parameters describing a cut cell, A^b and $\hat{n}^b$



- A^b is the area of the reconstructed interface in the cell and $\hat{n}^b$ is the the interface unit normal.

UNIVERSITY OF CAMBRIDGE

Other geometric parameters describing a cut cell

- Various other geometric parameters (e.g. cut cell volumetric centroid, centroid of the reconstructed interface) may also be required, depending on the system being solved, and the solvers that are used.

- All of these parameters can be calculated from the signed distance function in the way we discussed last time.

Solutions to the small cell problem

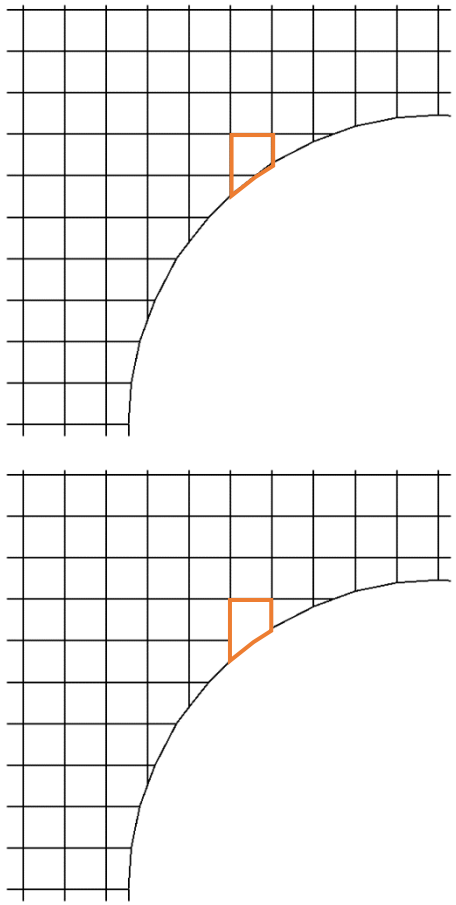
Small cell problem solutions overview

- Cut cell methods have been an active area of research since the early 1980s, and there are a lot of different methods that have been developed.

- We will take a brief look at the most popular approaches, and then a more detailed look at one recent method.

- Most cut cell methods cause a reduced order of accuracy at the boundary. This can be alleviated using mesh refinement of the interface (AMR can be very useful here).

Cell merging



- Cell merging solves the small cell problem by combining cut cells with their neighbours to produce larger volumes with a combined α of at least 0.5, typically.
- Earliest mentions: Clarke et al., 1986 [1] and Quirk, 1992 [2].
- Although conceptually simple, cell merging modifies the otherwise simple Cartesian flux/state data structures, and is algorithmically difficult to implement, particularly in 3D.
- Merging naturally reduces the accuracy of the solution at the boundary.

UNIVERSITY OF CAMBRIDGE

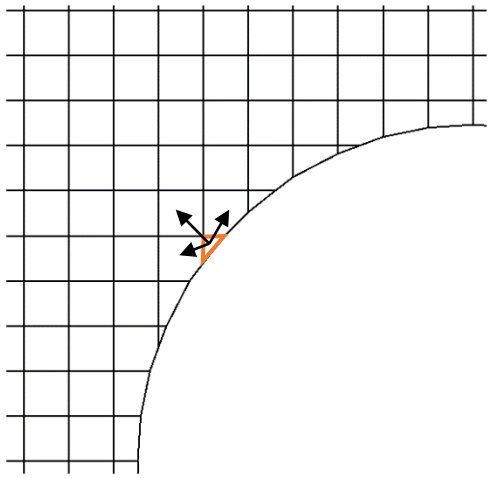
[1] Clarke, D.K. et al., Euler calculations for multielement airfoils using Cartesian grids. AIAA, 1986.
[2] Quirk, J. J., An alternative to unstructured grids for computing gas dynamic flows around arbitrarily complex two-dimensional bodies, Computers & Fluids, 1992.

Cell merging variants

- Variants have been developed to alleviate some of the algorithmic challenges.
- With cell linking (Kirkpatrick et al., 2003 [1]), the cells are kept distinct in the data structures, but algorithmically treated as merged in the time integration.
- With state mixing (Hu et al., 2006 [2]), cells are kept distinct in the data structures, and cell merged states are attained using post-update mixing terms.
- State redistribution (Berger, 2021 [3]) is similar in spirit to state mixing, but it allows for overlapping merging neighbourhoods, i.e. a cut cell can be merged with more than one neighbour. This method is implemented in various AMReX-based codes.

[1] Kirkpatrick, M. P. et al., A representation of curved boundaries for the solution of the Navier–Stokes equations on a staggered three-dimensional Cartesian grid, J. Comput. Phys, 2003.
[2] Hu, X. Y. et al., A conservative interface method for compressible flows, J. Comput. Phys, 2006.
[3] Berger, M. and Giuliani, A., A state redistribution algorithm for finite volume schemes on cut cell meshes, J. Comput. Phys, 2021.

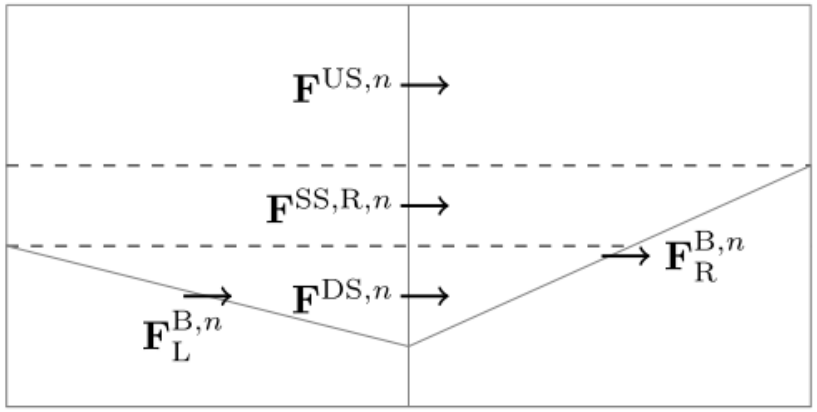
Flux redistribution



- Since the cut cell stable $\Delta t \propto \alpha$, flux redistribution solves the small cell problem by applying only α times the explicit flux update to the cut cell. To maintain conservation, the remainder of the flux is redistributed to its neighbours.
- See Chern and Colella, 1987 [1], Colella et al., 2006 [2].
- Can be shown to be first order at the boundary.
- Very widely used, and implemented in the AMReX and Chombo codes.

[1] Chern, I.L. and Colella, P., A conservative front tracking method for hyperbolic conservation laws, Lawrence Livermore National Laboratory Technical Report, 1987.
[2] P. Colella et al., A Cartesian grid embedded boundary method for hyperbolic conservation laws, J. Comput. Phys, 2006.

Flux stabilisation

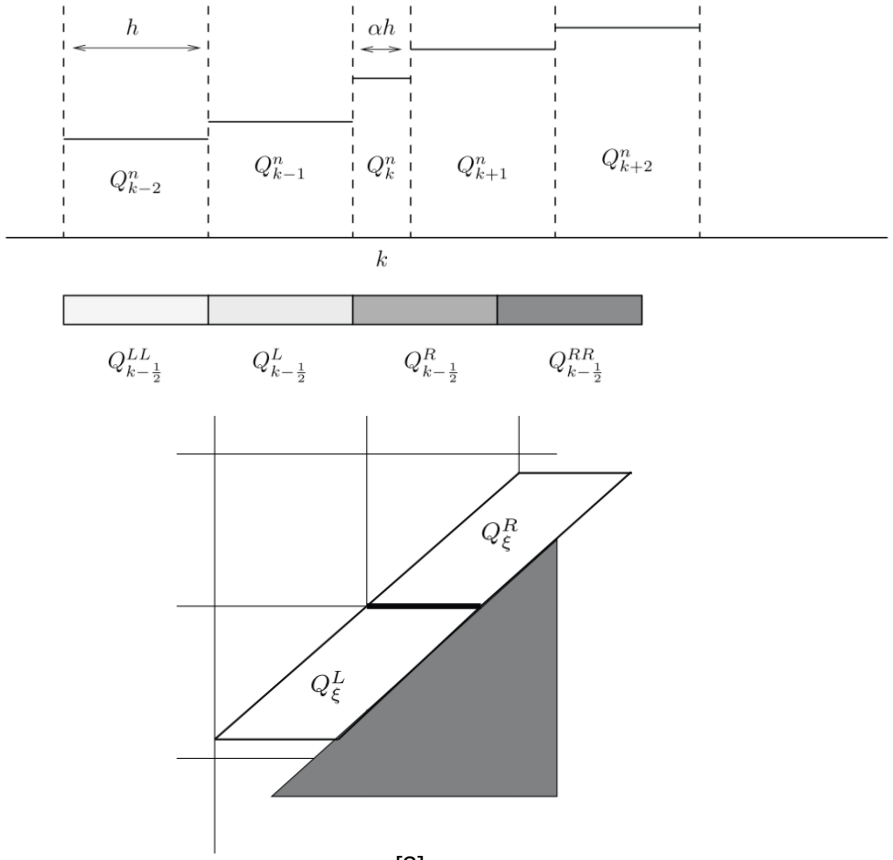


[2]

- This method solves the small cell problem by modifying the flux at the cut cell in a special way to produce a stable update for the cut cell [1,2,3].
- It is dimensionally split and simple to implement. It is one of the few methods demonstrated on complex 3D problems.
- Can be shown to be first order at the boundary.
- We will look into this method in more detail later.

[1] Klein, R. et al., Well-balanced compressible cut-cell simulation of atmospheric flow, Philos. Trans. R. Soc. A, 2009.
[2] Gokhale, N. et al., A dimensionally split Cartesian cut cell method for hyperbolic conservation laws, J. Comput. Phys, 2018.
[3] Bennett, W.P. et al., A moving boundary flux stabilization method for Cartesian cut-cell grids using directional operator splitting , J. Comput. Phys, 2018.

The h-box method



[2]

- This method solves the small cell problem by approximating cut cell fluxes based on initial values specified over regions of length 'h' (i.e. Δx).
- The method has been developed by M. Berger, R. LeVeque, C. Helzel and others. See [1,2], for example. It is comprehensively analysed, with a provably second order accurate L_∞ norm.
- Not yet implemented in 3D because of its complexity. (Berger, M., Cut Cells: Meshes and Solvers, Handbook of Numerical Analysis, Elsevier, 2017).

UNIVERSITY OF CAMBRIDGE

[1] Berger, M. J., Helzel, C., and LeVeque, R. J., H-Box Methods for the Approximation of Hyperbolic Conservation Laws on Irregular Grids, SIAM J. Numer. Anal., 2003.
[2] Berger, M. and Helzel, C., A Simplified h-box Method for Embedded Boundary Grids, SIAM J. Sci. Comput., 2012.

Order of convergence of a cut cell method (1/4)

- To measure how a simulation converges with increasing resolution, we first need a measure of the simulation error.
- Simulation error is usually quantified using the L^p norm:

$$L^p(E) = \left(\frac{1}{V}\int_\Omega E^p \, dV\right)^{1/p} = \left(\frac{1}{\sum_i \alpha_i}\sum_i \alpha_i E_i^p\right)^{1/p},$$

Continuous version Discrete version

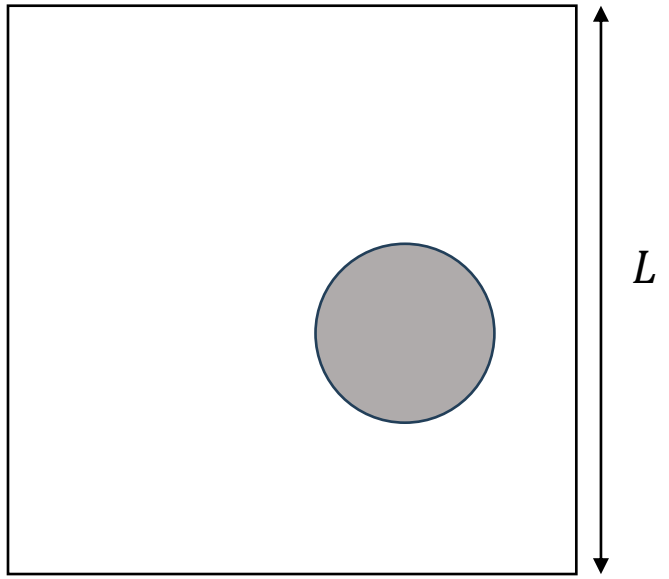
where E_i is the error in cell i (compared to a reference solution). Note that if we didn't have any cut cells, the formula simplifies to:

$$L^p(E) = \left(\frac{1}{N}\sum_i E_i^p\right)^{1/p},$$

where N is the total number of cells in the domain.

Order of convergence of a cut cell method (2/4)

- For a typical cut cell method that is $O(\Delta x)$ at the cut cells, and $O(\Delta x^2)$ elsewhere in the domain, how does the error as measured by the L^p norm converge?
- Let D be the number of dimensions, and n represent the number of cells in one co-ordinate direction. Then, the total number of cells $N \sim O(n^D)$, and the number of cut cells $N_{cc} \sim O(n^{D-1})$, since the cut cells only exist on the boundary.
- Furthermore, $n \sim O(\Delta x^{-1})$.
- For the example 2D simulation domain with a circular geometry shown on the right:
  - D = 2,
  - $n = \frac{L}{\Delta x}$ (i.e., $n \sim O(\Delta x^{-1})$),
  - $N = n^2$(i.e. $N \sim O(n^D)$),
  - $N_{cc} \sim O(n)$ (i.e., $N_{cc} \sim O(n^{D-1})$). Intuitively speaking, the boundary of the circle with known origin and radius can be parameterized with just one parameter (θ), i.e. it exists in 1D.



UNIVERSITY OF CAMBRIDGE

Order of convergence of a cut cell method (3/4)

- For a typical cut cell method that is $O(\Delta x)$ at the cut cells, and $O(\Delta x^2)$ elsewhere in the domain, how does the error as measured by the L^p norm converge?
- Let D be the number of dimensions, and n represent the number of cells in one co-ordinate direction. Then, the total number of cells $N \sim O(n^D)$, and the number of cut cells $N_{cc} \sim O(n^{D-1})$, since the cut cells only exist on the boundary.
- Furthermore, $n \sim O(\Delta x^{-1})$.
- Substituting these estimates into the L^p norm formula gives:

$$L^p(E) = \left(\frac{1}{N}\sum_i E_i^p\right)^{1/p},$$

$$\begin{aligned} L^p(E) &= \left(\frac{\mathcal{O}(\Delta x^p n^{D-1}) + \mathcal{O}(\Delta x^{2p})[\mathcal{O}(n^D) - \mathcal{O}(n^{D-1})]}{\mathcal{O}(n^D)}\right)^{\frac{1}{p}} \\ &= \left(\mathcal{O}(\Delta x^{p+1}) + \mathcal{O}(\Delta x^{2p}) - \mathcal{O}(\Delta x^{2p+1})\right)^{\frac{1}{p}} \\ &= \mathcal{O}(\Delta x^{\frac{p+1}{p}}). \end{aligned}$$

UNIVERSITY OF CAMBRIDGE

Order of convergence of a cut cell method (4/4)

- Hence, the L^p norm converges as $O\left(\Delta x^{\frac{p+1}{p}}\right)$.

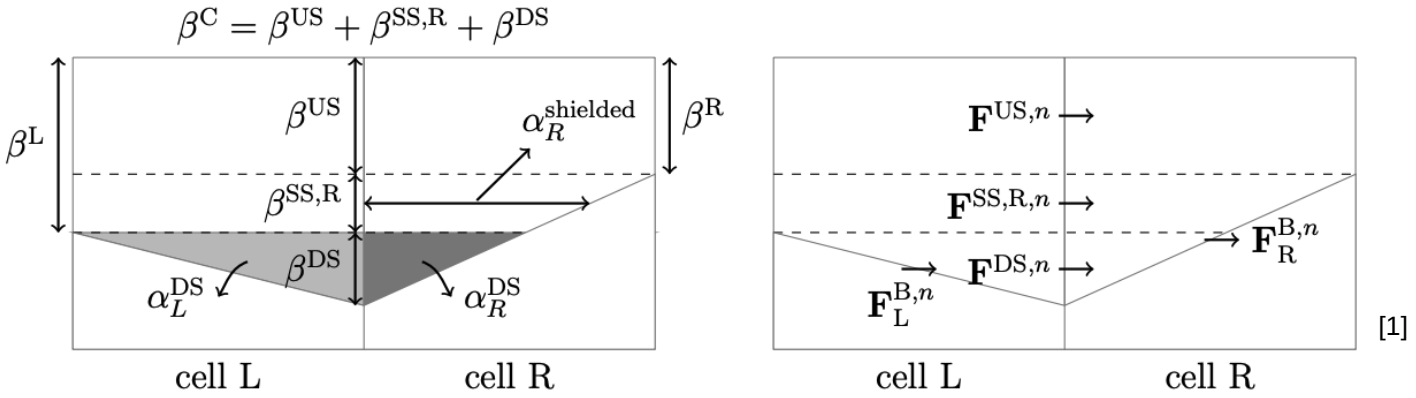
p	Convergence order
1	2
2	1.5
∞ (max error)	1

- The order of convergence depends on which norm is used. For example, the L^1 error will converge as $O(\Delta x^2)$ (second order), whereas the the L^∞ error would be expected to converge as the error at the cut cells, $O(\Delta x)$ (first order).

The flux stabilisation cut cell method

UNIVERSITY OF CAMBRIDGE

Interface decomposition and modified flux

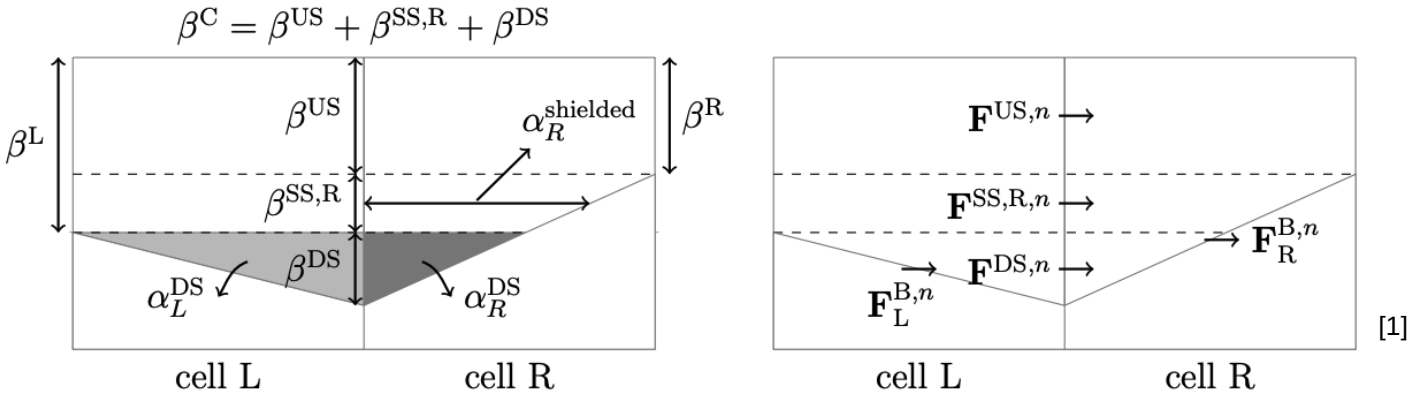


[1]

- Consider the configuration above, which shows two neighbouring cells (cell ‘L’ and ‘R’) in the current dimensional sweep direction (i.e. horizontal direction). In this case, they are both cut cells, but the method we are deriving does not require this to be the case.
- Recall from earlier that we said that it may be convenient to split-up the face fraction β. In the configuration above, the face area fraction between the cells, β^C, is split-up into β^{US}, $\beta^{SS,R}$ and β^{DS}, as will be described on the next slide.
- Note that $\beta^C = \beta^{US} + \beta^{SS,R} + \beta^{DS}$.

[1] Gokhale, N. et al., A dimensionally split Cartesian cut cell method for hyperbolic conservation laws, J. Comput. Phys, 2018.

Interface decomposition and modified flux



[1]

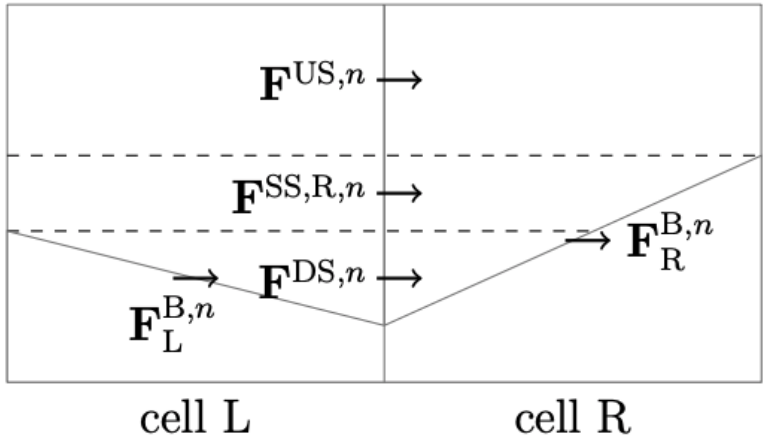
- We begin by decomposing the interface between neighbouring cells into 3 types:
  - 'Unshielded' (US) regions that do not 'face' the boundary in the current dimensional sweep direction.
  - 'Singly-shielded' (SS) regions that are 'covered' by the boundary from the left or right in the current sweep direction.
  - 'Doubly-shielded' (DS) regions that face the boundary from the left and right sides.
- If we can calculate stable fluxes for all these regions, then the stable flux at the cell face at time level 'n' is given by the area-weighted average of the constituent fluxes:

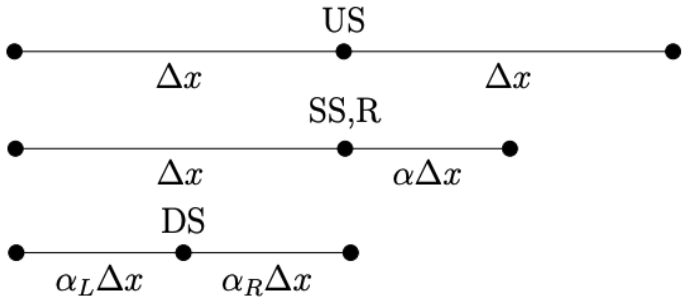
$$\mathbf{F}_{\mathrm{C}}^{\mathrm{modified},n} = \frac{1}{\beta^{\mathrm{C}}}\left[\beta^{\mathrm{US}}\mathbf{F}^{\mathrm{US},n} + \beta^{\mathrm{SS,L}}\mathbf{F}^{\mathrm{SS,L},n} + \beta^{\mathrm{SS,R}}\mathbf{F}^{\mathrm{SS,R},n} + \beta^{\mathrm{DS}}\mathbf{F}^{\mathrm{DS},n}\right]$$


UNIVERSITY OF CAMBRIDGE

[1] Gokhale, N. et al., A dimensionally split Cartesian cut cell method for hyperbolic conservation laws, J. Comput. Phys, 2018.

Equivalent 1D problems



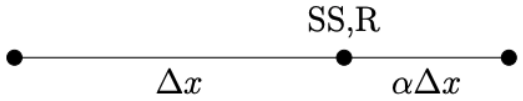


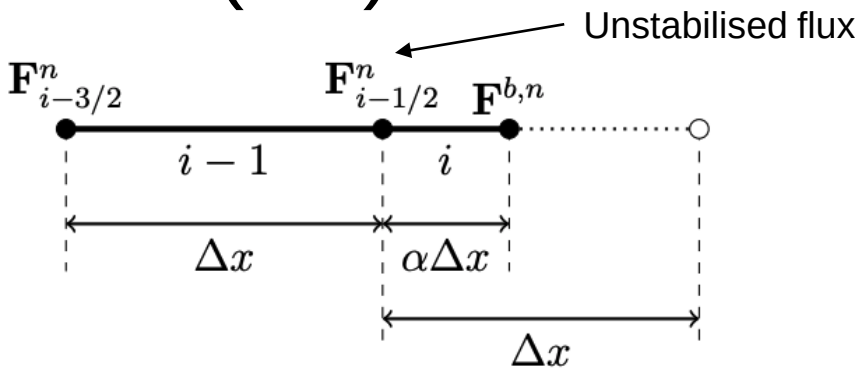
[1]

- The task of computing the stable flux has now been reduced to a series of 1D problems:
  - For the unshielded region, there is no stability restriction; the unshielded flux is therefore just the regular explicit flux.
  - For the singly-shielded region, the flux needs stabilising, since it sees the boundary to the right. **This is the configuration we will focus on.** Note that the 1D volume fractions $\alpha, \alpha_L, \alpha_R \in [0,1]$**.**
  - The doubly-shielded region is restricted from both sides. Dealing with that configuration is beyond the scope of this lecture. One solution is to define a conservative mixing procedure for that region [2].

[1] Gokhale, N., A dimensionally split Cartesian cut cell method for the Computational Fluid Dynamics, PhD Thesis, 2018.
[2] Gokhale, N. et al., A dimensionally split Cartesian cut cell method for hyperbolic conservation laws, J. Comput. Phys, 2018.

The KBN stabilised flux (1/2)





[1]

- One way to stabilise the cut cell flux is to use the method of Klein, Bates and Nikiforakis [2].
- We start by estimating the time-updated cut cell state by extending the 'influence' of the cut cell to the regular cell length Δx. Note that Δt is the stable time-step for the regular cells.

$$\bar{\mathbf{U}}_i^{n+1} = \mathbf{U}_i^n + \frac{\Delta t}{\Delta x}(\mathbf{F}^n_{i-i/2} - \mathbf{F}^{b,n}) \tag{1}$$

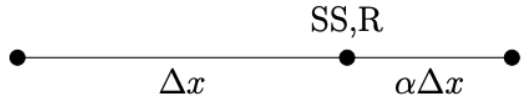
- The update of Eq. 1 is stable but non-conservative. The desired conservative update is given by:

Stabilised flux

$$\hat{\mathbf{U}}_i^{n+1} = \mathbf{U}_i^n + \frac{\Delta t}{\alpha\Delta x}(\mathbf{F}^{\mathrm{KBN},n}_{i-1/2} - \mathbf{F}^{b,n}) \tag{2}$$

[1] Gokhale, N. et al., A dimensionally split Cartesian cut cell method for hyperbolic conservation laws, J. Comput. Phys, 2018.
[2] Klein, R. et al., Well-balanced compressible cut-cell simulation of atmospheric flow, Philos. Trans. R. Soc. A, 2009.

The KBN stabilised flux (2/2)

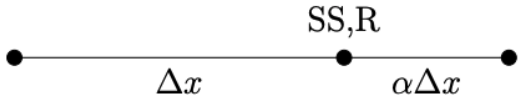


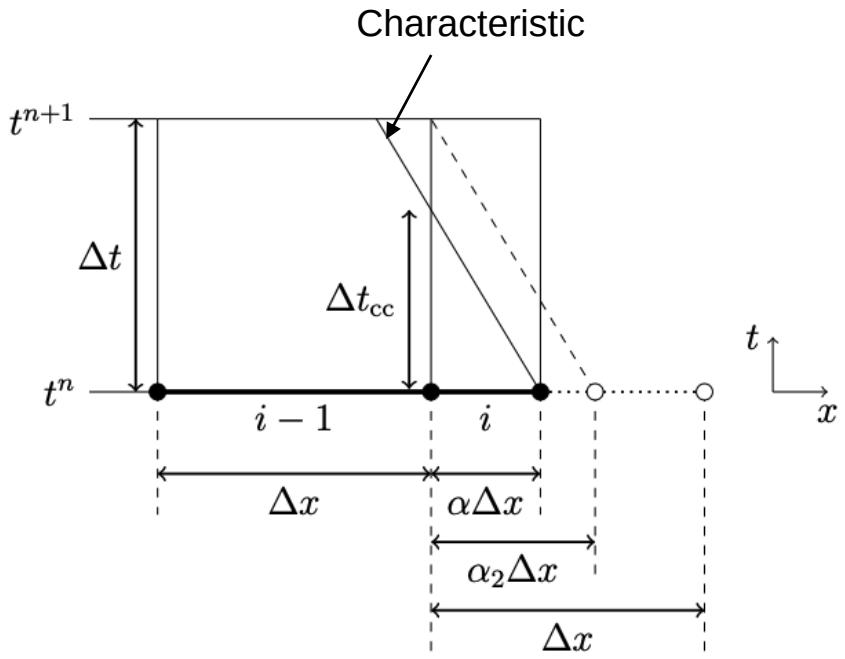
- Eq. 1 gives us a stable but non-conservative update, while Eq. 2 gives us a conservative update, but with an unknown stabilised flux. By equating the right hand sides of Eq. 1 and Eq. 2, we can solve for the unknown stabilised flux:

$$\mathbf{F}^{\mathrm{KBN},n}_{i-1/2} = \mathbf{F}^{b,n} + \alpha(\mathbf{F}^{n}_{i-i/2} - \mathbf{F}^{b,n})$$

- Note that the stabilised KBN flux is consistent with the natural limits of the grid, so that as $\alpha \to 1$, $\mathbf{F}^{\mathrm{KBN},n}_{i-1/2} \to \mathbf{F}^{n}_{i-i/2}$, and as $\alpha \to 0$, $\mathbf{F}^{\mathrm{KBN},n}_{i-1/2} \to \mathbf{F}^{b,n}$.
- This flux may be shown to be first order accurate. However, note that only the geometric parameter α is used when determining the stabilized flux.
- A more accurate flux can be determined by using also information about the wave speeds of the system.

The LPFS flux (1/2)





[1]

- Recall that for stability, we require that the numerical domain of dependence should contain the true domain of dependence.
- In the figure on the left, this implies that the cut cell depresses the stable time step to $\Delta t_{cc} < \Delta t$.
- For stability, the 'influence' (i.e. numerical domain of dependence) of the cut cell doesn't therefore need to be increased all the way to the regular cell length Δx, as in the original KBN method.
- By increasing the 'influence' by just the right amount ($\alpha_2 \Delta x$ in the figure), we can derive a new 'modified' KBN flux using the same approach as before:

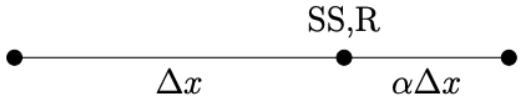
$$\mathbf{F}_{i-1/2}^{\mathrm{KBN,mod},n} = \mathbf{F}^{b,n} + \frac{\alpha}{\alpha_2}(\mathbf{F}_{i-i/2}^{n} - \mathbf{F}^{b,n})$$

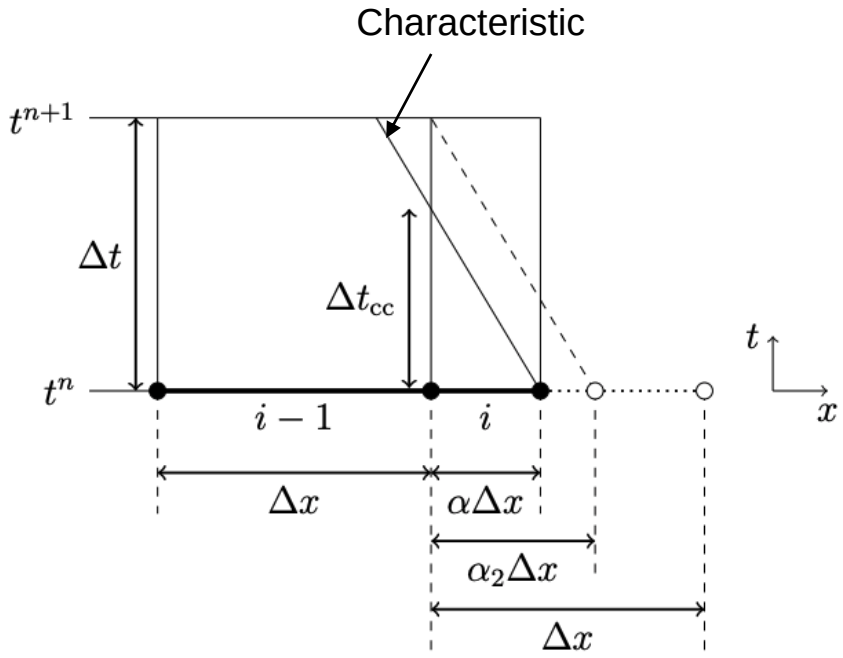
- Note that $\frac{\alpha}{\alpha_2} = \frac{\Delta t_{cc}}{\Delta t}$ can be calculated from an exact or approximate estimate of the characteristic speed.

UNIVERSITY OF CAMBRIDGE

[1] Gokhale, N. et al., A dimensionally split Cartesian cut cell method for hyperbolic conservation laws, J. Comput. Phys, 2018.

The LPFS flux (2/2)





[1]

- Furthermore, we note that for the fraction $\frac{\Delta t_{cc}}{\Delta t}$ of the time step, no flux stabilisation is necessary, since the numerical domain of dependence of the cut cell is already large enough.
- Using these two insights, we can devise the new LPFS flux, which is a linear combination of the unstabilised and stabilised fluxes [1]:

$$\mathbf{F}_{i-1/2}^{\mathrm{LPFS},n} = \frac{\Delta t_{\mathrm{cc}}}{\Delta t}\mathbf{F}_{i-1/2}^{n} + \left(1 - \frac{\Delta t_{\mathrm{cc}}}{\Delta t}\right)\mathbf{F}_{i-1/2}^{\mathrm{KBN,mod},n}$$

- We will see shortly that this method can also be shown to be first order accurate.
- However, the use of geometric and wave speed information leads to more accurate solutions than those calculated with the basic KBN flux.

UNIVERSITY OF CAMBRIDGE

[1] Gokhale, N. et al., A dimensionally split Cartesian cut cell method for hyperbolic conservation laws, J. Comput. Phys, 2018.

Convergence and stability of the flux stabilisation

The Lax equivalence Theorem: *A consistent numerical method for a well-posed linear problem with smooth solutions is convergent iff it is linearly stable.*

- We will perform our analysis on the linear advection equation (w.l.o.g. assume $a > 0$):

$$u_t + a u_x = 0$$

using the linear first order upwind scheme:

$$U_i^{n+1} = U_i^n + c(U_{i-1}^n - U_i^n), \text{ where } c = \frac{a\Delta t}{\Delta x} \text{ (Courant number).}$$

- "Consistency" means that the local truncation error of the method vanishes as $\Delta t, \Delta x \to 0$.
- "Convergence" means that the numerical solution converges to the true solution of the PDE as $\Delta t, \Delta x \to 0$.
- "Stability" means that errors do not experience unbounded growth as $t \to \infty$.
- Conclusions from linear analysis are not guaranteed to apply to non-linear problems, but they are still useful. In practice, the flux stabilisation does work on complex, non-linear problems.

UNIVERSITY OF CAMBRIDGE

Small u and capital U notation

- U_i^n is the numerical, finite volume, discrete approximation of the solution in cell i at time n.
- When performing the convergence analysis, however, we want to be taking Taylor expansions of the exact solution $u(x,t)$ at different points in space and time.
- We therefore introduce u_i^n, which is called the 'grid function' of the exact solution $u(x,t)$ in cell i at time n. In other words, the 'small' u is the exact value we are hoping to approximate in the cell, and which we can take Taylor expansions of.
- Note, however, that $U_i^n = \frac{1}{\Delta x}\int_{x-1/2}^{x+1/2} u(x,t^n)dx = u_i^n + O(\Delta x^2)$, from the mid-point rule of numerical integration. In other words, $u_i^n = U_i^n$, to second order accuracy.
- In other words, for the analysis of methods which are at most second order, the pointwise values of the exact solution at the cell centre can be used interchangeably with the volume-averaged values.
- In our analysis, we can therefore use u_i^n and U_i^n interchangeably.

Step-by-step approach for assessing the consistency of a numerical method

- Recall that in order to show that a method is consistent, we need to prove that the local truncation error of the method vanishes as $\Delta t, \Delta x \to 0$.
- The local truncation error, Lu, can be calculated using the following steps:

1. Move all terms to the left hand side of the equation (i.e. $\frac{U_i^{n+1} - U_i^n}{\Delta t} - a\frac{U_{i-1}^n - U_i^n}{\Delta x} = 0$).

2. Using the terms on the LHS, replace the capital U_i^n with small u_i^n for all i and n.

3. Taylor expand the small u_i^n.

4. Simplify the resulting expression, remembering to make use of the original PDE (i.e. $u_t + au_x = 0$) when possible.

5. If the remaining terms are first (or higher) order in Δt and Δx, we can conclude that **the method is consistent**, since consistency is concerned with looking at what happens as $\Delta t, \Delta x \to 0$.

UNIVERSITY OF CAMBRIDGE

Consistency in the absence of cut cells

- The truncation error of the first order upwind scheme is given by:

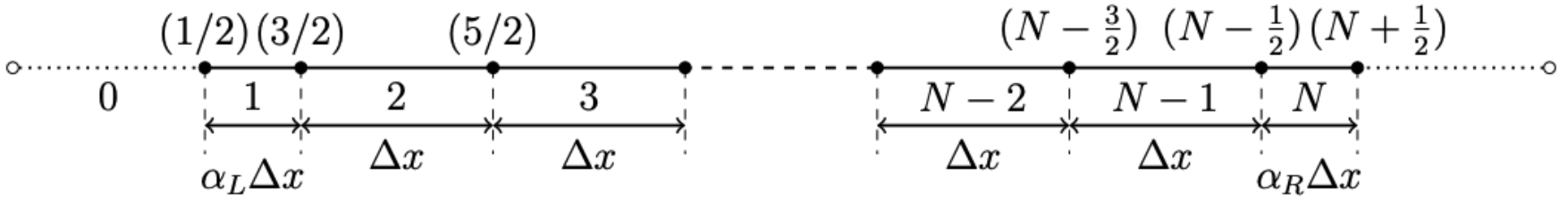
$$Lu = \frac{u_i^{n+1} - u_i^n}{\Delta t} - a\frac{u_{i-1}^n - u_i^n}{\Delta x}$$

$$= \frac{u(x_i, t^n) + \Delta t u_t(x_i, t^n) + O(\Delta t^2) - u(x_i, t^n)}{\Delta t} - a\frac{u(x_i, t^n) - \Delta x u_x(x_i, t^n) + O(\Delta x^2) - u(x_i, t^n)}{\Delta x}$$

$$= u_t(x_i, t^n) + a u_x(x_i, t^n) + O(\Delta t, \Delta x) = O(\Delta t, \Delta x)$$

- The numerical method is consistent, since the truncation error vanishes as $\Delta t, \Delta x \to 0$.
- If we could prove the linear stability of the scheme, we could then use the Lax equivalence theorem to conclude that it is convergent.

Consistency in the presence of cut cells (1/3)



[1]

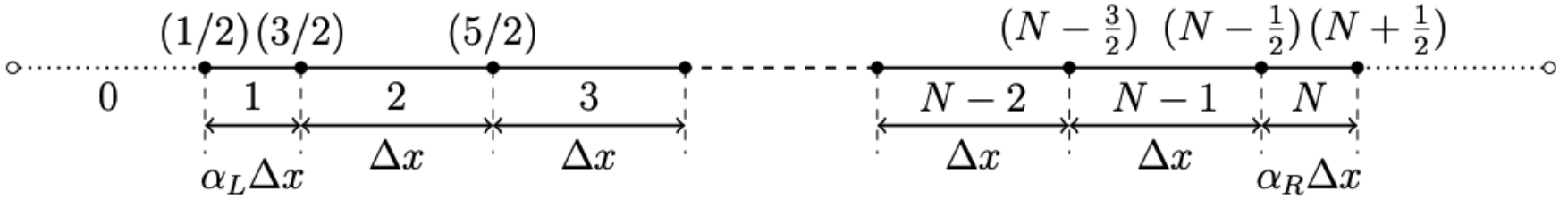
- Consider a 1D grid with cut cells on the left and right ends (volume fractions α_L and α_R respectively).

- Stabilising the numerical flux using the LPFS procedure gives the following update formulas for the cells that are affected by the flux stabilisation:

$$
\begin{aligned}
U_1^{n+1} &= U_1^n + c(2-\alpha_L)(U_0^n - U_1^n),\\
U_2^{n+1} &= U_2^n + c[(\alpha_L-1)^2 U_0^n + \alpha_L(2-\alpha_L)U_1^n - U_2^n],\\
U_{N-1}^{n+1} &= U_{N-1}^n + c[U_{N-2}^n - \alpha_R(2-\alpha_R)U_{N-1}^n - (\alpha_R-1)^2 U_N^n],\\
U_N^{n+1} &= U_N^n + c(2-\alpha_R)(U_{N-1}^n - U_N^n),
\end{aligned}
$$

[1] Gokhale, N. et al., A dimensionally split Cartesian cut cell method for hyperbolic conservation laws, J. Comput. Phys, 2018.

Consistency in the presence of cut cells (2/3)



[1]

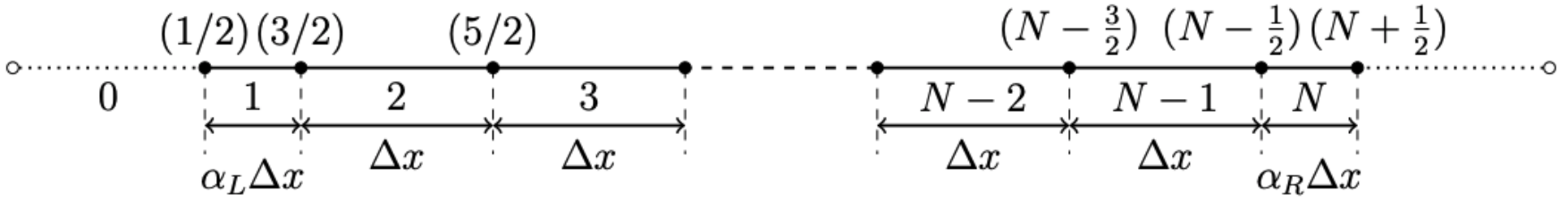
- Consider the truncation error of the scheme at cell N:

$$
\begin{aligned}
Lu_N &= \frac{u_N^{n+1}-u_N^n}{\Delta t} - (2-\alpha_R)a\frac{u_{N-1}^n-u_N^n}{\Delta x} \\
&= u_t(x_N,t^n) - (2-\alpha_R)a\frac{u_N^n - \frac{(1+\alpha_R)}{2}\Delta x u_x(x_N,t^n) - u_N^n}{\Delta x} + \mathcal{O}(\Delta t,\Delta x) \\
&= -au_x(x_N,t^n) + (2-\alpha_R)a\frac{(1+\alpha_R)}{2}u_x(x_N,t^n) + \mathcal{O}(\Delta t,\Delta x) \\
&= \frac{1}{2}\alpha_R(\alpha_R-1)au_x(x_N,t^n) + \mathcal{O}(\Delta t,\Delta x),
\end{aligned}
$$

- The truncation error doesn't vanish as $\Delta t, \Delta x \to 0$, so the scheme is not consistent.

[1] Gokhale, N. et al., A dimensionally split Cartesian cut cell method for hyperbolic conservation laws, J. Comput. Phys, 2018.

Consistency in the presence of cut cells (3/3)



[1]

- The scheme is not consistent in any of the cells affected by the flux stabilisation.
- However, numerical tests confirm the method does converge with first order.
- To prove this so-called *supraconvergence* property of the flux stabilisation, we make use of an idea developed by Wendroff and White [1], and applied in the context of cut cells by Berger, Helzel and LeVeque [2].

[1] Gokhale, N. et al., A dimensionally split Cartesian cut cell method for hyperbolic conservation laws, J. Comput. Phys, 2018.
[2] Wendroff, B. and White, A.B., A supraconvergent scheme for nonlinear hyperbolic systems, Comput. Math. Appl., 1989.
[3] Berger, M. J., Helzel, C., and LeVeque, R. J., H-Box Methods for the Approximation of Hyperbolic Conservation Laws on Irregular Grids, SIAM J. Numer. Anal., 2003.

‘Supraconvergence’ in the presence of cut cells (1/2)

- The basic idea is the following. Our aim was to show consistency, i.e., that:

$$|u_i - U_i| = O(\Delta t, \Delta x)\ , \forall i, \tag{1}$$

but we could not show this directly. Suppose, however, that we had a new grid function:

$$w = u + O(\Delta x), \tag{2}$$

and that for this grid function, we could show consistency, i.e.:

$$|w_i - U_i| = O(\Delta t, \Delta x)\ , \forall i. \tag{3}$$

- What would this show?

‘Supraconvergence’ in the presence of cut cells (2/2)

- What would this show?
  - It would show that although our numerical scheme that gives us the capital U_i is not consistent with the grid function of our real solution u_i, it ‘does’ consistently solve for another function w_i.
  - This other function w_i is not any old function. It is a function that differs from the real solution u_i by *precisely* $\mathrm{O}(\Delta \mathrm{x})$.
  - Since we are concerned about the behaviour of the numerical method as $\Delta x \to 0$, it means that for all intents and purposes, our numerical method *is* effectively consistent in the limit of $\Delta x \to 0$.
  - Assuming the scheme is stable (we still need to show this), this would prove the first order convergence (or ‘supraconvergence’ of the method).

The grid function w

- Proving that the appropriate grid function $w = u + O(\Delta x)$ exists is a cumbersome exercise, and deriving it is beyond the scope of this course.
- The grid function that we want does exist, and for the case of the linear advection equation on the cut cell grid that we are considering, it takes the following form:

$$w_i^n = \begin{cases} u_i^n + \frac{\alpha_L(\alpha_L - 1)}{2(2-\alpha_L)} \Delta x u_x(x_i, t^n) & \text{if } i = 1, \\ u_i^n + \left(\frac{1+\alpha_R(\alpha_R-2)}{\alpha_R - 2} \right) \Delta x u_x(x_i, t^n) & \text{if } i = N-1, \\ u_i^n + \frac{1}{2}(\alpha_R - 1)\Delta x u_x(x_i, t^n) & \text{if } i = N, \\ u_i^n & \text{otherwise,} \end{cases}$$

- Note also that $w = u + O(\Delta x)$ as desired.
- We will use the above grid function to test that our numerical method is consistent with w_i, as we want it to be.

UNIVERSITY OF CAMBRIDGE

Step-by-step approach for assessing the consistency of the numerical method with *w*

1. Move all terms to the left hand side of the equation.

2. Replace all capital U_i^n with small w_i^n for all i and n.

3. Replace the w_i^n with the u_i^n for all i using the derived expression for the grid function on the previous slide.

4. Taylor expand the small u_i^n.

5. Simplify the resulting expression, remembering to make use of the original PDE (i.e. $u_t + au_x = 0$) when possible.

6. If the remaining terms are first (or higher) order in Δt and Δx, we can conclude that **the method is consistent when solving for *w***, since consistency is concerned with looking at what happens as $\Delta t, \Delta x \to 0$.

UNIVERSITY OF CAMBRIDGE

Truncation error using the grid function w

$$w_i^n = \begin{cases} u_i^n + \frac{\alpha_L(\alpha_L-1)}{2(2-\alpha_L)}\Delta x u_x(x_i,t^n) & \text{if } i=1, \\ u_i^n + \left(\frac{1+\alpha_R(\alpha_R-2)}{\alpha_R-2}\right)\Delta x u_x(x_i,t^n) & \text{if } i=N-1, \\ u_i^n + \frac{1}{2}(\alpha_R-1)\Delta x u_x(x_i,t^n) & \text{if } i=N, \\ u_i^n & \text{otherwise,} \end{cases}$$

- Consider again the truncation error of the scheme at cell N:

$$\begin{aligned} Lw_N &= \frac{w_N^{n+1}-w_N^n}{\Delta t} - (2-\alpha_R)a\frac{w_{N-1}^n - w_N^n}{\Delta x} \\ &= \frac{u_N^{n+1} + \frac{1}{2}(\alpha_R-1)\Delta x u_x(x_N,t^{n+1}) - u_N^n - \frac{1}{2}(\alpha_R-1)\Delta x u_x(x_N,t^n)}{\Delta t} \\ &- (2-\alpha_R)a\frac{u_{N-1}^n + \left(\frac{1+\alpha_R(\alpha_R-2)}{\alpha_R-2}\right)\Delta x u_x(x_{N-1},t^n) - u_N^n - \frac{1}{2}(\alpha_R-1)\Delta x u_x(x_N,t^n)}{\Delta x} \\ &= \frac{u_N^{n+1} + \frac{1}{2}(\alpha_R-1)\Delta x u_x(x_N,t^n) - u_N^n - \frac{1}{2}(\alpha_R-1)\Delta x u_x(x_N,t^n)}{\Delta t} \\ &- (2-\alpha_R)a\frac{u_N^n - \frac{(1+\alpha_R)}{2}\Delta x u_x(x_N,t^n) + \left(\frac{1+\alpha_R(\alpha_R-2)}{\alpha_R-2}\right)\Delta x u_x(x_N,t^n)}{\Delta x} \\ &- (2-\alpha_R)a\frac{-u_N^n - \frac{1}{2}(\alpha_R-1)\Delta x u_x(x_N,t^n)}{\Delta x} + \mathcal{O}(\Delta t,\Delta x) \\ &= u_t(x_N,t^n) - (2-\alpha_R)a\frac{-\frac{\Delta x u_x(x_N,t^n)}{(2-\alpha_R)}}{\Delta x} + \mathcal{O}(\Delta t,\Delta x) \\ &= u_t(x_N,t^n) + a u_x(x_N,t^n) + \mathcal{O}(\Delta t,\Delta x) = \mathcal{O}(\Delta t,\Delta x). \end{aligned}$$

- Now, the truncation error vanishes as $\Delta t, \Delta x \to 0$, as required.
- The same can be shown for all the cells affected by the flux stablisation.
- Note that this also gives us an 'update formula' for w_N^n:

$$w_N^{n+1} = w_N^n + c(2-\alpha_R)(w_{N-1}^n - w_N^n) + \mathcal{O}(\Delta t, \Delta x)$$

- The above expression will come in handy when proving the stability of the method.

UNIVERSITY OF CAMBRIDGE

Linear stability of the flux stabilisation (1/3)

$$U_N^{n+1} = U_N^n + c(2-\alpha_R)(U_{N-1}^n - U_N^n)$$
$$w_N^{n+1} = w_N^n + c(2-\alpha_R)(w_{N-1}^n - w_N^n) + \mathcal{O}(\Delta t, \Delta x)$$

- What remains is to prove that the the computed U_i^n stably approximate the w_i^n.
- Consider the error function $v_i^n = U_i^n - w_i^n$. As per our earlier definition, stability requires that the error remains bounded as $n \to \infty$.
- We already have the LPFS-derived update formulas for U_i^n, and the truncation analysis has given us update formulas for w_i^n . Subtracting one from the other gives us expressions for the evolution of the error function in the boundary cells ($i = 1, 2, N-1, N$). These can be presented conveniently using matrix-vector notation:

$$\underbrace{\begin{bmatrix} v_1^{n+1} \\ v_2^{n+1} \\ v_{N-1}^{n+1} \\ v_N^{n+1} \end{bmatrix}}_{\mathbf{x}^{n+1}} = \underbrace{\begin{bmatrix} 1-c(2-\alpha_L) & 0 & 0 & 0 \\ c\alpha_L(2-\alpha_L) & 1-c & 0 & 0 \\ 0 & 0 & 1-c\alpha_R(2-\alpha_R) & -c(\alpha_R-1)^2 \\ 0 & 0 & c(2-\alpha_R) & 1-c(2-\alpha_R) \end{bmatrix}}_{A} \underbrace{\begin{bmatrix} v_1^n \\ v_2^n \\ v_{N-1}^n \\ v_N^n \end{bmatrix}}_{\mathbf{x}^n} + \underbrace{\begin{bmatrix} \mathcal{O}(\Delta t, \Delta x) \\ \mathcal{O}(\Delta t, \Delta x) \\ \mathcal{O}(\Delta t, \Delta x) \\ \mathcal{O}(\Delta t, \Delta x) \end{bmatrix}}_{\mathbf{b}^n}$$

Linear stability of the flux stabilisation (2/5)

- We therefore have a linear inhomogeneous recurrence relation for the evolution of the error function in the cells affected by the flux stabilisation.

$$\begin{aligned}\mathbf{x}^{n+1} &= A\mathbf{x}^n + \mathbf{b}^n \\ &= A(A\mathbf{x}^{n-1} + \mathbf{b}^{n-1}) + \mathbf{b}^n = A^2\mathbf{x}^{n-1} + A\mathbf{b}^{n-1} + \mathbf{b}^n \\ &= A^3\mathbf{x}^{n-2} + A^2\mathbf{b}^{n-2} + A\mathbf{b}^{n-1} + \mathbf{b}^n \\ &= A^{n+1}\mathbf{x}^0 + \sum_{\nu=0}^{n} A^\nu \mathbf{b}^{n-\nu},\end{aligned}$$

where we note that the $\mathbf{x}^0$ and $\mathbf{b}$ vectors are made up of $O(\Delta t, \Delta x)$ components. Using the triangle inequality*, we can deduce the following:

$$|\mathbf{x}^{n+1}| \leq \|A\|^{n+1}|\mathbf{x}^0| + \max_{k=0,\ldots,n}\{|\mathbf{b}^k|\}\sum_{\nu=0}^{n}\|A\|^\nu.$$

UNIVERSITY OF CAMBRIDGE

*$\|\mathbf{x}+\mathbf{y}\| \leq \|\mathbf{x}\| + \|\mathbf{y}\|$

Linear stability of the flux stabilisation (3/5)

$$|\mathbf{x}^{n+1}| \leq \|A\|^{n+1}|\mathbf{x}^0| + \max_{k=0,\ldots,n}\{|\mathbf{b}^k|\}\sum_{\nu=0}^{n}\|A\|^{\nu}.$$

- For the error to remain a bounded $O(\Delta t, \Delta x)$ term, we require that $\|A\| < 1$.
  - It is clear that this condition keeps the first time on the right hand side bounded.
  - It is also required to bound the second term because the summation term is essentially an infinite geometric series in $\|A\|$.

- This condition $\|A\| < 1$ is satisfied if the spectral radius* of the matrix $\rho(A) < 1$.

- Since A is a block diagonal matrix, its eigenvalues will be the union of the eigenvalues of the block matrices positioned along the diagonal:

$$A_L = \begin{bmatrix} 1 - c(2-\alpha_L) & 0 \\ c\alpha_L(2-\alpha_L) & 1-c \end{bmatrix}, \; A_R = \begin{bmatrix} 1 - c\alpha_R(2-\alpha_R) & -c(\alpha_R - 1)^2 \\ c(2-\alpha_R) & 1 - c(2-\alpha_R) \end{bmatrix},$$

UNIVERSITY OF CAMBRIDGE

*maximum of absolute values of eigenvalues

Linear stability of the flux stabilisation (4/5)

$$A_L = \begin{bmatrix} 1 - c(2 - \alpha_L) & 0 \\ c\alpha_L(2 - \alpha_L) & 1 - c \end{bmatrix} \qquad \begin{aligned} \lambda_L^1 &= 1 - c, \\ \lambda_L^2 &= 1 - c(2 - \alpha_L) \end{aligned}$$

Eigenvalues

- It can be verified that:
  - λ_L^1 and λ_L^2 are real,
  - $\lambda_L^1 \in [0,1)$ for $c \in (0,1]$, and
  - $|\lambda_L^2| < 1$ for $c \in (0,1]$ and $\alpha_L \in (0,1]$.

- Hence, the eigenvalues of A_L have magnitude less than 1 over the full range of c and α as desired.

Linear stability of the flux stabilisation (5/5)

$$A_R = \begin{bmatrix} 1 - c\alpha_R(2-\alpha_R) & -c(\alpha_R - 1)^2 \\ c(2-\alpha_R) & 1 - c(2-\alpha_R) \end{bmatrix},$$

$$\lambda_R^1 = \frac{1}{2}\left[2 - 2c - \alpha_R c + \alpha_R^2 c - (\alpha_R - 1)\left(\sqrt{-4 + \alpha_R^2}\right)c\right]$$
$$\lambda_R^2 = \frac{1}{2}\left[2 - 2c - \alpha_R c + \alpha_R^2 c + (\alpha_R - 1)\left(\sqrt{-4 + \alpha_R^2}\right)c\right]$$

Eigenvalues

- Since $\alpha_R \in (0,1]$, the eigenvalues are complex conjugates with the same magnitude $|\lambda_R|$, and:

$$|\lambda_R|^2 = \lambda_R^1 \lambda_R^2 = \det(A_R)^* > 0.$$

- To confirm that $|\lambda_R| < 1$, we therefore need to confirm that $\det(A_R) < 1$.

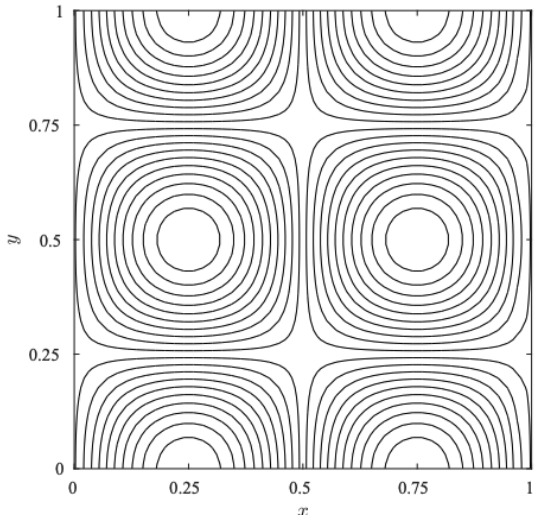
$$\det(A_R) = 1 - [2c(1-c) + \alpha_R c(1 - \alpha_R + c)]$$

- Since $2c(1-c)$ and $\alpha_R c(1 - \alpha_R + c)$ are greater than 0 for $c, \alpha_R \in (0,1]$, we note that $\det(A_R) < 1$ as required.
- This concludes the proof.

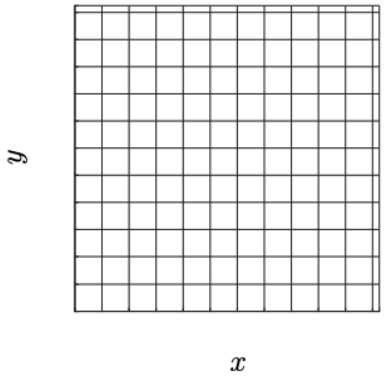
UNIVERSITY OF CAMBRIDGE

* The determinant of an $n \times n$ matrix is given by the product of its eigenvalues.

Two-dimensional diagonal advection validation



$$u(x, y) = \sin(2\pi x)\cos(2\pi y)$$



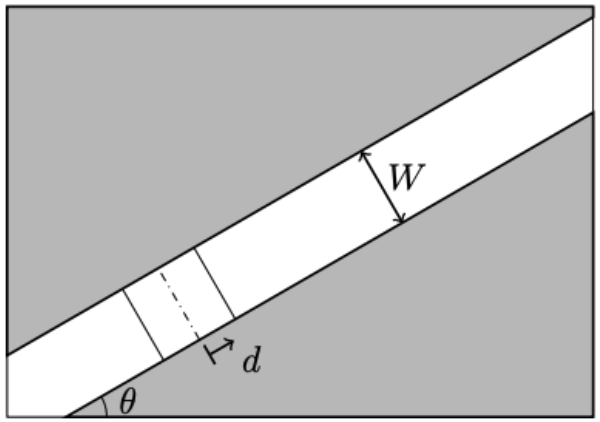
Close-up of cut cells at the top-right part of the mesh

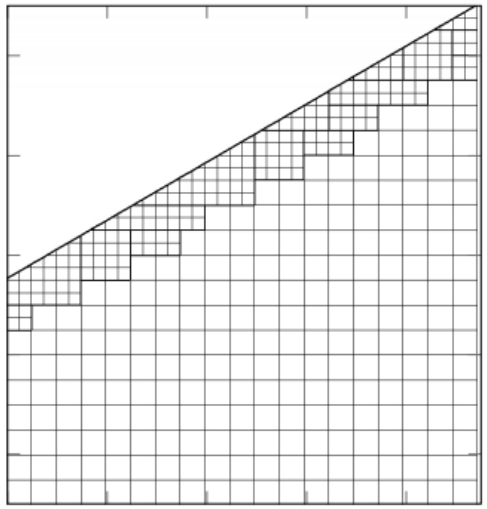
[1]

Resolution	L_1 norm	L_1 order	L_2 norm	L_2 order	L_∞ norm	L_∞ order
50×50	6.44×10^{-3}	-	8.75×10^{-3}	-	3.60×10^{-2}	-
100×100	1.56×10^{-3}	2.04	2.51×10^{-3}	1.80	1.85×10^{-2}	0.96
200×200	3.92×10^{-4}	1.99	8.21×10^{-4}	1.61	1.00×10^{-2}	0.88
400×400	9.88×10^{-5}	1.99	2.81×10^{-4}	1.55	5.29×10^{-3}	0.92

[1] Gokhale, N. et al., A dimensionally split Cartesian cut cell method for hyperbolic conservation laws, J. Comput. Phys, 2018.

Effect of AMR refinement on the cut cell error





[1]

Resolution	L_1 norm	L_2 norm	L_∞ norm
100×70 (no AMR)	6.19×10^{-3}	1.69×10^{-2}	1.32×10^{-1}
200×140 (second order, expected)	1.55×10^{-3}	4.23×10^{-3}	3.31×10^{-2}
200×140 (no AMR)	1.72×10^{-3}	5.75×10^{-3}	6.09×10^{-2}
200×140 (with cut cells refinement)	1.31×10^{-3}	3.62×10^{-3}	2.93×10^{-2}

- For this diagonal advection problem, the use of AMR at the boundary reduces the simulation error to below that which would be obtained by a method that was second order accurate everywhere.

[1] Gokhale, N. et al., A dimensionally split Cartesian cut cell method for hyperbolic conservation laws, J. Comput. Phys, 2018.

Things we haven't covered

Although we have looked into cut cell methods in quite a lot of detail, there are certain aspects we haven't touched upon:

- Cut cell meshes can be a good choice for moving boundary problems, since the grid re-generation (computing of new cuts) can be done locally and efficiently. However, other challenges (freshly created/covered cells, etc.) must be overcome.
  - References [1,2,3] are good sources to understand the challenges and how they may be overcome.

- When calculating viscous flows, careful consideration needs to be given to the equation discretisation at the boundary if the solution is not to be affected by the mesh irregularity at the cut cells.
  - References [4,5,6] are good sources to understand the challenges and how they may be alleviated.

[1] Bennett et al., A moving boundary flux stabilization method for Cartesian cut-cell grids using directional operator splitting, J. Comput. Phys, 2018.
[2] L. Schneiders et al., An efficient conservative cut-cell method for rigid bodies interacting with viscous compressible flows, J. Comput. Phys, 2016.
[3] M. Meinke et al., A cut-cell method for sharp moving boundaries in Cartesian grids, Computers & Fluids, 2013.
[4] M. Berger, M. J. Aftosmis, and S. R. Allmaras, Progress Towards a Cartesian Cut-Cell Method for Viscous Compressible Flow, AIAA 2012-1301, 2012.
[5] M. J. Berger and M. J. Aftosmis, An ODE-based Wall Model for Turbulent Flow Simulations, AIAA Journal, 2017.
[6] Gokhale et al., A dimensionally split Cartesian cut cell method for the compressible Navier-Stokes equations, J. Comput. Phys, 2018.

Summary

This lecture covered the following topics:

- An overview of the most popular cut cell methods that address the ‘small cell problem’.
- An in-depth look at the flux stabilisation cut cell method.