Introduction to CFD: numerical stability

The classic method for understanding stability is to consider the growth of a single Fourier mode in our discretisation.

$$a_i^n = A^n e^{Ii\theta}, \text{ where } I = \sqrt{-1}$$

θ represents a phase.

A method is stable if: $|A^{n+1}/A^n| \leq 1.$

Exercise 4.5

Using the above stability analysis, considering the amplitude of a single Fourier mode, show that the growth of a mode for the upwind method (Eq. 4.2) is:

$$\left|\frac{A^{n+1}}{A^n}\right|^2 = 1 - 2C(1-C)(1-\cos\theta) \tag{4.12}$$

and stability requires $2C(1-C) \geq 0$ *or* $0 \leq C \leq 1$.

Introduction to CFD: FTCS method

The upwind method is first-order accurate.

Ultimately we will want higher-order accurate methods. The most obvious change from our initial discretisation is to try a higher-order spatial derivative.

Exercise 4.4

You may think that using a centered-difference for the spatial derivative, $a_x \sim (a_{i+1} - a_{i-1})/(2\Delta x)$ would be more accurate. This method is called FTCS (forward-time, centered-space). Try this on the same test problems.

Using the above stability analysis, considering the amplitude of a single Fourier mode, study the growth of a mode for the FTCS method. What do you conclude?

You will find that no matter what value of C you choose with the FTCS method, the solution is unconditionally *unstable*. If you continue to evolve the equation with this method, you would find that the amplitude grows without bound.

Introduction to CFD: numerical stability

It is important to note that this stability analysis only works for linear equations, where the different Fourier modes are decoupled, nevertheless, we use its ideas for nonlinear advection problems as well.

Truncation analysis can also help us understand stability. The idea here is to keep the higher order terms in the Taylor series to understand how they modify the actual equation you are trying to solve.

Introduction to CFD: numerical stability

To get an alternate feel for stability, we can ask what the terms left out by truncation look like. That is, we can begin with the discretized equation:

$$a_i^{n+1} - a_i^n = -\frac{u\Delta t}{\Delta x}(a_i^n - a_{i-1}^n) \tag{4.13}$$

and replace a_i^{n+1} with a Taylor expansion in time, and replace a_{i-1}^n with a Taylor expansion in space, keeping terms to $O(\Delta t^3)$ and $O(\Delta x^3)$. Replacing $\partial a/\partial t$ with $-u\partial a/\partial x$ in the higher-order terms, show that our difference equation more closely corresponds to

$$a_t + u a_x = \frac{u\Delta x}{2}\left(1 - \frac{\Delta t u}{\Delta x}\right)\frac{\partial^2 a}{\partial x^2} \tag{4.14}$$

$$= \frac{u\Delta x}{2}(1 - \mathcal{C})\frac{\partial^2 a}{\partial x^2} \tag{4.15}$$

Introduction to CFD: numerical stability

Notice that the righthand side of Eq. 4.14 looks like a diffusion term, however, if $C > 1$, then the coefficient of the diffusion is negative—this is unphysical.

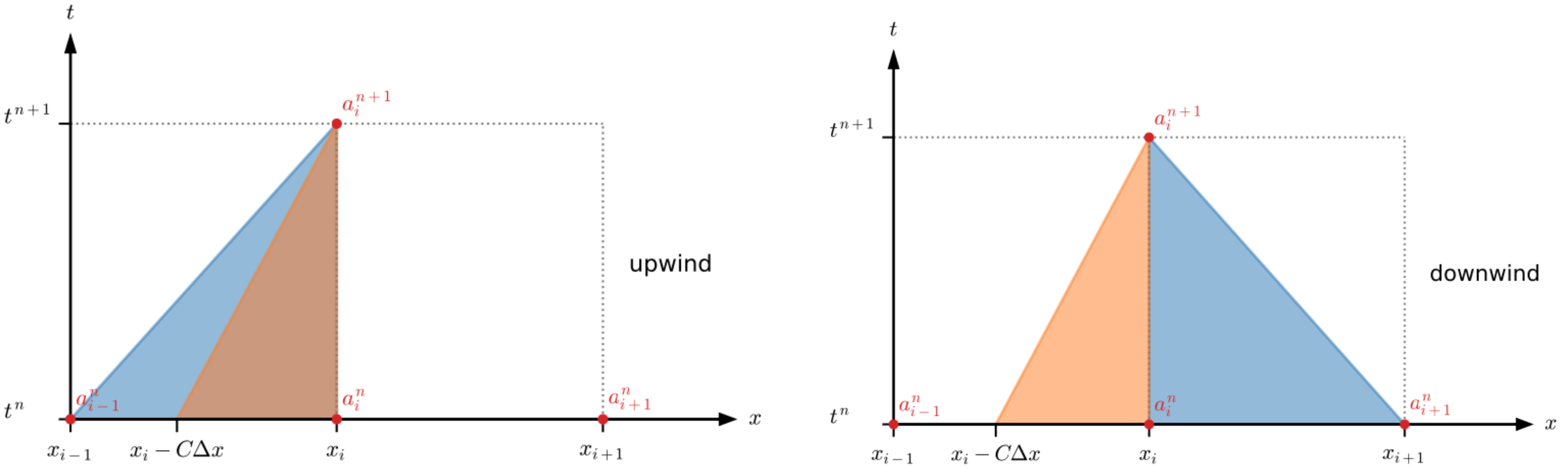
This means that the diffusion would act to take smooth features and make them more strongly peaked—the opposite of physical diffusion.

For FTCS, a similar truncation analysis would show that the diffusion term is always negative.

Introduction to CFD: domain of dependence

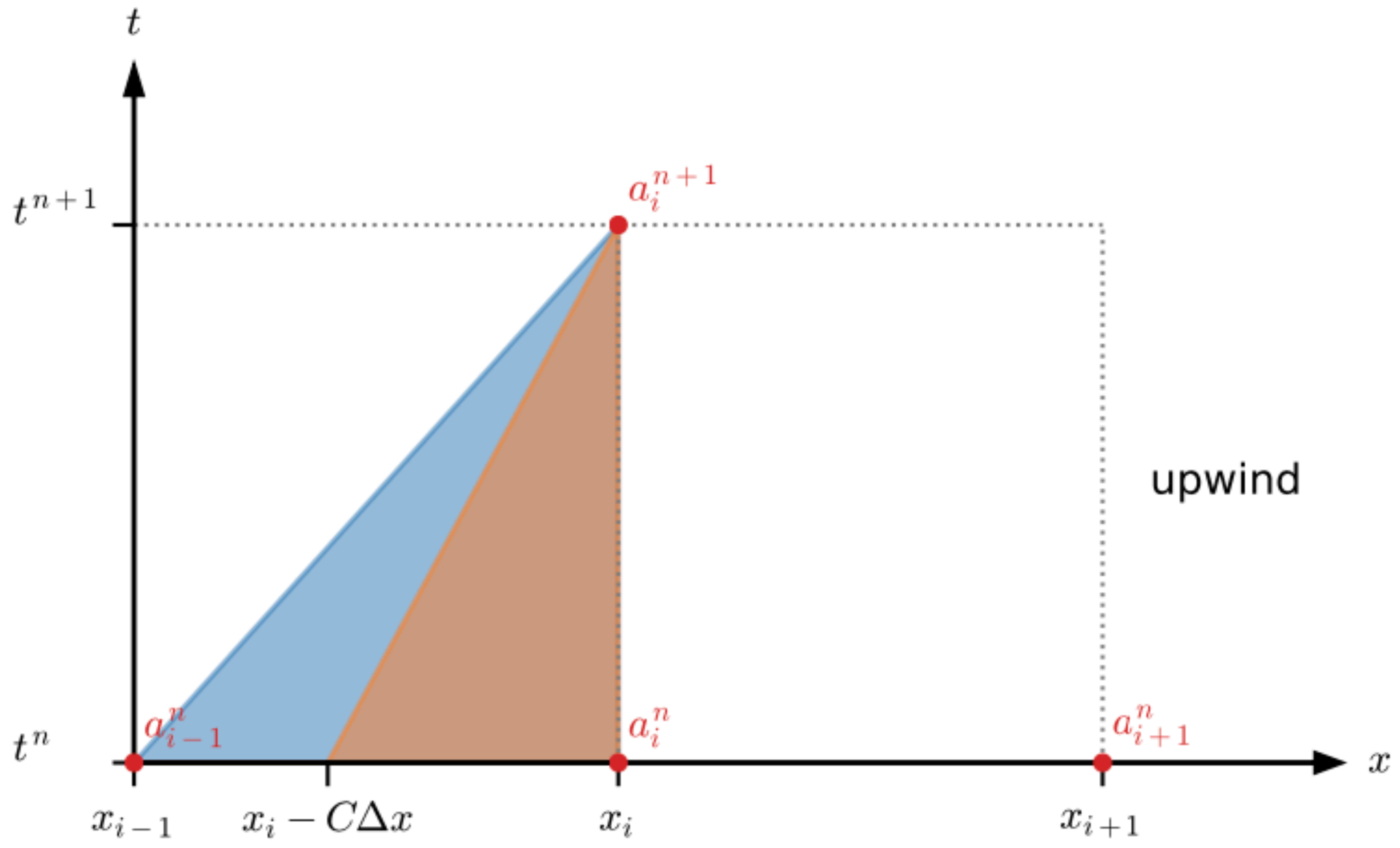
Another important view of our numerical difference scheme is to look at the domain of dependence. The figure below illustrates this for the updated point a^{n+1}.

The numerical domain of dependence shows the points that can influence the updated value of a_i using our finite difference methods.



Space-time diagrams showing the numerical domain of dependence (blue region) for three different difference methods.

Introduction to CFD: domain of dependence



For the upwind scheme, we see that this is a triangle that includes a_i^n and a_i^{n-1}. The **physical domain of dependence** is shown as the **orange triangle** —this is formed by tracing backwards in time from a_i^{n+1} along a characteristic, reaching out to the point $x_i - C\Delta x = x_i - u\Delta t$ over Δt.

Introduction to CFD: domain of dependence

Any stable numerical method must have a numerical domain of dependence that includes the physical domain of dependence.

If it does not, then the update to the solution simply does not see the points that contribute to the solution over the time-step.

Notice that this is a necessary, but not sufficient condition for stability.

e.g. the FTCS method has a domain of dependence that includes the physical domain of dependence, but it is not stable.