Scalar hyperbolic equations - stable solutions

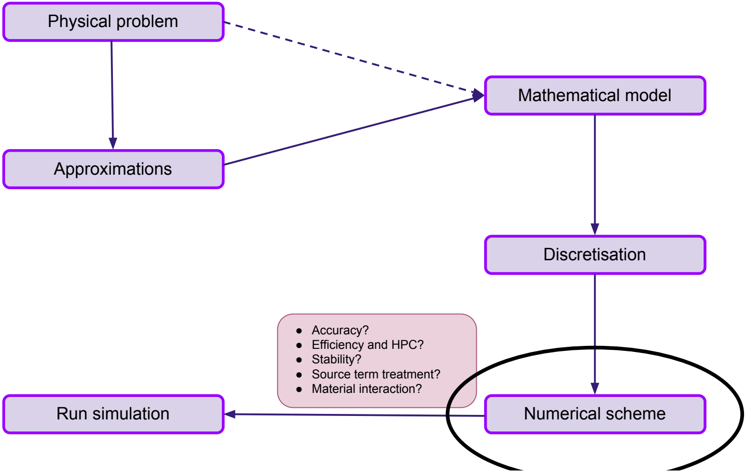
Stephen Millmore and Nikos Nikforakis

Laboratory for Scientific Computing, University of Cambridge

Outline

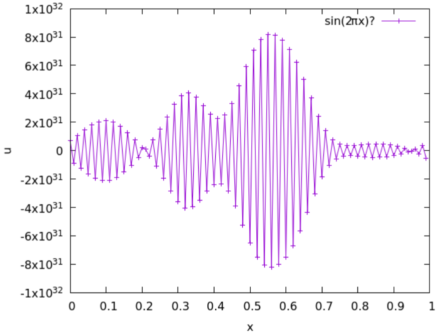
Outline

Stable numerical methods



Unstable simulations

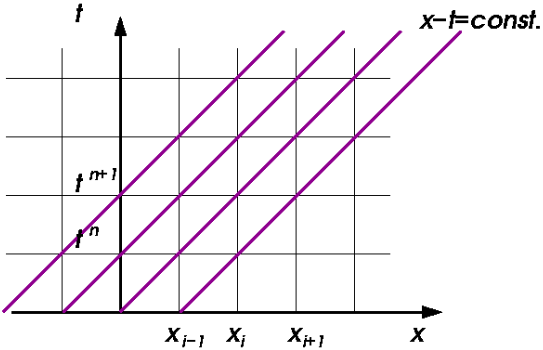
- It is clear that numerical methods are not simply a matter of discretising partial differential equations



- We need to know that the method chosen is **stable**
- This is a combination of the choice of discretisation and the choice of time step
- Full derivation of the stability of numerical methods is a course in its own right
- Here, we shall cover some of the basic considerations

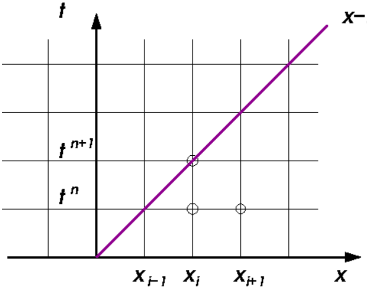
Domain of dependence

- One of the key concepts behind stability is the **domain of dependence** of the solution u_i^{n+1}
- To see this, we revisit the $x - t$ diagram, setting $a = 1$ and overlay the numerical grid

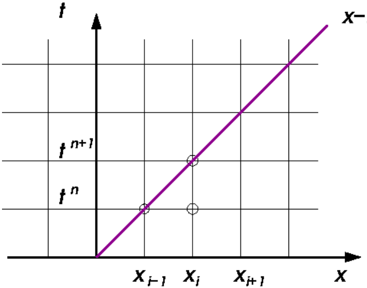


Domain of dependence - choice of discretisation

- For the advection equation case with $a = 1$ and $\Delta t = \Delta x$, it is clear that the exact solution implies $u_i^{n+1} = u_{i-1}^n$
- In this case, it is clear that the backward difference used the point u_{i-1}^n to compute its solution (in fact, the update scheme **gives** $u_i^{n+1} = u_{i-1}^n$)
- By contrast, the forward difference does not use the solution u_{i-1}^n at all



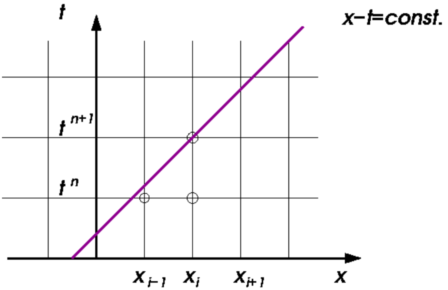
Forward

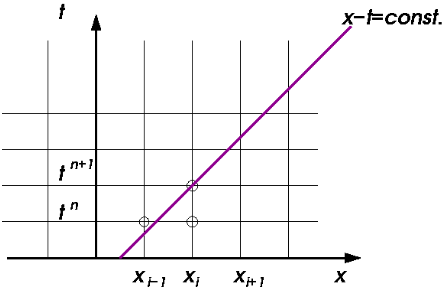


Backward

Domain of dependence - choice of Δt

- If we consider what happens if change Δt, keeping $a = 1$ (equivalent to changing a, in the third part of the practical)
- If Δt is too large, the location from which the solution at u_i^{n+1} comes from lies outside $x \in [x_{i-1}, x_i]$ - outside of the numerical stencil of the method
- For smaller Δt, the location from which the solution at u_i^{n+1} comes from lies inside the numerical stencil





Information travel

- The stability of a numerical method for a hyperbolic system of equations is directly related to the wave speed(s) within the system
- Recall from the method of characteristics for the advection equation,

$$\frac{\mathrm{d}x}{\mathrm{d}t} = a$$

- This is the wave speed within the advection equation - the speed with which information about u travels
- Broadly speaking, the information at u_i^{n+1} cannot come from outside the numerical stencil if we want stable solutions from a numerical method
- That is why forward differencing, and large time steps, did not work for $a = 1$
- But what about centred differencing?

Stability of a method

- It is clearly important to make sure that information is not required from outside of the domain of dependence
- However, it is also undesirable to use information which cannot affect the solution u_i^{n+1}
- If this information is included, it is possible that the errors this introduces grow with time - the solution goes **unstable**
- For difference schemes and linear systems, von Neumann stability analysis is used to identify (by decomposing errors as Fourier series) conditions for stability
- For an equation written as

$$u_i^{n+1} = u_i^n + r \times (\text{difference approximations})$$

bounds can be placed on r such that errors do not grow

Von Neumann Stability analysis (centred difference)

$$u_i^{n+1} = u_i^n - a\frac{\Delta t}{2\Delta x}\left(u_{i+1}^n - u_{i-1}^n\right)$$

- When we plug the right-hand side of this equation into a computer, we get a solution N_i^{n+1}, and some unknown error, $\mathcal{E}_i^{n+1}$, where

$$u_i^{n+1} = N_i^{n+1} + \mathcal{E}_i^{n+1}$$

- We **know** that u_i^{n+1} satisfies our forward difference method, by construction, and we **find** that N_i^{n+1} satisfies it
- This means that our error term itself also satisfies the equation, i.e.

$$\mathcal{E}_i^{n+1} = \mathcal{E}_i^n - a\frac{\Delta t}{2\Delta x}\left(\mathcal{E}_{i+1}^n - \mathcal{E}_{i-1}^n\right)$$

- The key to von Neumann stability analysis is that we can express the error as a Fourier series; i.e. a series of periodic errors

Von Neumann Stability analysis - the Fourier series

- Recall that a function, $g(x)$, with periodicity P, can be approximated as a Fourier series

$$g(x) \approx \sum_{m=-\infty}^{\infty} \left(c_m \exp\left(\frac{2\pi i m x}{P} \right) \right)$$

- In the practical, a periodic domain was considered, i.e. the methods considered had periodicity $P = 1$
- More generally, a domain $x \in [-L/2, L/2]$ has periodicity $P = L$
- The quantity m is related to the frequency of the oscillations
- In a discrete domain with $2M + 1$ points, there are a finite number of frequencies possible, $m \in [-M : M]$
- Discrete Fourier series exist,

$$\mathcal{E}(x,t) = \sum_{m=-M}^{M} \left(E_m(t) \exp\left(\frac{2\pi i m x}{L} \right) \right)$$

Von Neumann Stability analysis - the Fourier series

$$g(x) \approx \sum_{m=-\infty}^{\infty} \left(c_m \exp\left(\frac{2\pi i m x}{P}\right)\right) \quad \rightarrow \quad \mathcal{E}(x,t) = \sum_{m=-M}^{M} \left(E_m(t) \exp\left(\frac{2\pi i m x}{L}\right)\right)$$

- Here, we only consider the Fourier decomposition in space (time can be considered separately)
- Instead of constants, c_m, we have functions of time, $E_m(t)$, but these are still the amplitudes of the respective modes
- The quantity $2\pi m/L = k_m$ is also known as the wave number (the frequency of error mode m)
- Note, we have assumed periodic boundaries, but this is not necessary for stability analysis, the maths to this point changes slightly, but the result is the same

Von Neumann Stability analysis - the Fourier series

$$\mathcal{E}(x,t) = \sum_{m=-M}^{M} \left(E_m(t) \exp\left(\frac{2\pi i m x}{L} \right) \right)$$

- Since our centred difference is a linear equation, each mode, m, is independent
- In other words, each error mode, m, satisfies the equation

$$\mathcal{E}_{m,i}^{n+1} = \mathcal{E}_{m,i}^{n} - a\frac{\Delta t}{2\Delta x}\left(\mathcal{E}_{m,i+1}^{n} - \mathcal{E}_{m,i-1}^{n}\right)$$

- What we want to do is show that the error term does not increase with time, i.e. $\mathcal{E}_{m,i}^{n+1} < \mathcal{E}_{m,i}^{n}$ for all m
- The only term which can grow is the amplitude, $E_m(t)$
- Stability analysis now focuses on how this changes with time

Von Neumann Stability analysis - error amplitude

$$\mathcal{E}_i^{n+1} = \mathcal{E}_i^n - a\frac{\Delta t}{2\Delta x}\left(\mathcal{E}_{i+1}^n - \mathcal{E}_{i-1}^n\right)$$

- We rewrite our equation for a specific mode, m

$$E_m(t+\Delta t)e^{ik_m x} = E_m(t)e^{ik_m x} - a\frac{\Delta t}{2\Delta x}\left(E_m(t)e^{ik_m(x+\Delta x)} - E_m(t)e^{ik_m(x-\Delta x)}\right)$$

- A little rearranging gives

$$E_m(t+\Delta t)e^{ik_m x} = E_m(t)e^{ik_m x}\left[1 - a\frac{\Delta t}{2\Delta x}\left(e^{ik_m\Delta x} - e^{-ik_m\Delta x}\right)\right]$$

- This allows us to define the change in error over a time step

$$\Delta E_m = \frac{E_m(t+\Delta t)}{E_m(t)} = 1 - ia\frac{\Delta t}{\Delta x}\sin(k_m\Delta x)$$

Von Neumann Stability analysis (centred difference)

$$\Delta E_m = \frac{E_m(t + \Delta t)}{E_m(t)} = 1 - ia\frac{\Delta t}{\Delta x}\sin(k_m \Delta x)$$

- The final step, to understand how this amplitude changes, is to consider the magnitude

$$|\Delta E_m| = \left(1 - ia\frac{\Delta t}{\Delta x}\sin(k_m \Delta x)\right)\left(1 + ia\frac{\Delta t}{\Delta x}\sin(k_m \Delta x)\right) = 1 + a^2\frac{\Delta t^2}{\Delta x^2}\sin^2(k_m \Delta x)$$

- No matter what values we choose for Δt, Δx or a, we have

$$|\Delta E_m| \geq 1 \quad \Rightarrow \quad E_m(t + \Delta t) \geq E_m(t)$$

- Therefore, the error for this method **always** grows; it is **unconditionally unstable**

Von Neumann Stability analysis (backward difference)

- We now consider a case where we know the method **can** be stable, jumping straight to the specific Fourier mode, we have

$$E_m(t+\Delta t)e^{ik_m x} = E_m(t)e^{ik_m x} - a\frac{\Delta t}{\Delta x}\left(E_m(t)e^{ik_m x} - E_m(t)e^{ik_m(x-\Delta x)}\right)$$

- Rearranging this to consider the change in error gives

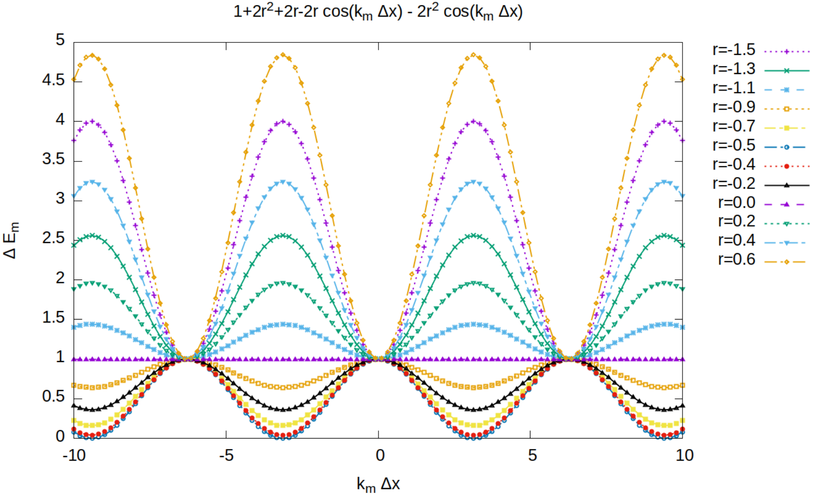
$$\Delta E_m = 1 - a\frac{\Delta t}{\Delta x}\left(1 - e^{-ik_m \Delta x}\right) = 1 - a\frac{\Delta t}{\Delta x}\left[1 - (\cos(k_m\Delta x) - i\sin(k_m\Delta x))\right]$$

- For simplicity, we define $r = -a\Delta t/\Delta x$
- Taking the magnitude and rearranging gives

$$|\Delta E_m| = 1 + 2r^2 + 2r - 2r\cos(k_m\Delta x) - 2r^2\cos(k_m\Delta x)$$

- Clearly it is not always immediately obvious what the restrictions on the time step will be, even for simple schemes
- As a first step, we can plot this error function for different values of r

Von Neumann Stability analysis (backward difference)



Von Neumann Stability analysis (backward difference)

$$|\Delta E_m| = 1 + 2r^2 + 2r - 2r\cos(k_m \Delta x) - 2r^2\cos(\Delta x)$$

- It is clear that there are definite values of r which ensure $\Delta E_m \leq 1$, i.e. that $E_m(t + \Delta t) \leq E_m(t)$
- With a bit of rearranging we can be more rigorous about the ranges

$$\Delta E_m = 1 + 2r\,(r+1)\,(1 - \cos(k_m \Delta x))$$

- In order to have $\Delta E_m \leq 1$, we now see that we need $r\,(r+1) \leq 0$
- This requires $-1 \leq r \leq 0$
- We can use this to get a constraint on the time step

$$-1 \leq -a\frac{\Delta t}{\Delta x} \leq 0 \quad \Rightarrow \quad \Delta t \leq \frac{\Delta x}{a}, \quad a \geq 0$$

Stability of the methods - overview

Forward difference:

$$u_i^{n+1} = u_i^n - a\frac{\Delta t}{\Delta x}\left(u_{i+1}^n - u_i^n\right), \qquad r = -a\frac{\Delta t}{\Delta x}$$

Stable if: $0 \leq |r| \leq 1, \qquad a < 0 \qquad$ (conditionally stable)

Backward difference:

$$u_i^{n+1} = u_i^n - a\frac{\Delta t}{\Delta x}\left(u_i^n - u_{i-1}^n\right), \qquad r = -a\frac{\Delta t}{\Delta x}$$

Stable if: $0 \leq |r| \leq 1, \qquad a > 0 \qquad$ (conditionally stable)

Centred difference:

$$u_i^{n+1} = u_i^n - a\frac{\Delta t}{2\Delta x}\left(u_{i+1}^n - u_{i-1}^n\right), \qquad r = -a\frac{\Delta t}{2\Delta x}$$

Stable if: $r = 0 \qquad$ (unconditionally unstable)

The CFL number

- In practice, the criteria for a stable numerical method is used to define the time step for a simulation
- This is often written as

$$\Delta t = C\frac{\Delta x}{|a|}$$

- C is the Courant-Friedrichs-Lewy (CFL) number, which is directly related to r, and is a number defined by the stability properties of a numerical method
- The relationship above is then the **CFL condition**
- **Generally**, a good numerical method for a hyperbolic PDE is stable for $C \leq 1$ (for some systems, it is possible to go beyond this), and for some methods, additional constraints may be required on a
- The CFL condition can be thought of as a ratio between the speed with which the physical medium moves and the speed with which the numerical method moves information

$$C = \frac{a}{(\Delta x/\Delta t)}$$

Choosing a time step from $\Delta t = C\Delta x / |a|$

- The stability analysis gives us a range of stable time steps dependent on the properties of the equation being solved, and your choice of spatial discretisation
- Alternatively, you could choose a time step, and use this to pick Δx, however for complex systems of equations, a is **not constant**
- This would then require continually adjusting Δx, which is much more complex to implement,
- You'd need to interpolate to your new grid spacing, and this would introduce additional error
- This does assume your problem has initial data as a function of space at a given time, $f(x, 0)$

Choosing a time step from $\Delta t = C\Delta x / |a|$

- We will assume that you will always select Δx yourself, and use your scheme to work out Δt
- The next aim is then to keep the error in your simulation as **low as possible**
- Going back to our Fourier modes, for a given mode, m, the total error from the simulation will be

$$E_m^{\text{tot}} = \sum_{n=1}^{T} \frac{E_m(t^n)}{E_m(t^{n-1})}$$

- We can minimise this error by ensuring that T is as small as possible, i.e. we take the **fewest** time steps possible
- This means we want to select C to be as **large as possible**

Outline

Towards a complete numerical method

- So far we have specified some simple numerical schemes which can be used in certain cases
- It is far more useful to have a single method which **works in all cases**
- It is also desirable to have methods which make use of second order approximations to derivatives (provided we make sure they are stable)
- Many such schemes exist, we will consider a few commonly encountered ones

First-order upwind method

- This is the name given if we wrap our forwards and backwards difference schemes in an if statement:

$$u_i^{n+1} = \begin{cases} u_i^n - a\frac{\Delta t}{\Delta x}\left(u_i^n - u_{i-1}^n\right) & a \geq 0 \\ u_i^n - a\frac{\Delta t}{\Delta x}\left(u_{i+1}^n - u_i^n\right) & a < 0 \end{cases}$$

- The name derives from the comparison between the advection speed a and a wind speed - things that are downwind from you give you no idea what is approaching
- This is probably the easiest stable numerical method
- It is common when writing this (and other) methods to use the CFL number directly

$$C = a\frac{\Delta t}{\Delta x}$$

Lax-Friedrichs method

- This method is a technique to stabilise the centred difference update

$$u_i^{n+1} = u_i^n - \frac{1}{2} C \left(u_{i+1}^n - u_{i-1}^n \right)$$

- It does so by replacing the u_i^n term with an average $\frac{1}{2} \left(u_{i+1}^n + u_{i-1}^n \right)$
- This gives a scheme which can be written

$$u_i^{n+1} = \frac{1}{2} \left(1 + C \right) u_{i-1}^n + \frac{1}{2} \left(1 - C \right) u_{i+1}^n$$

- Despite using second order differencing, the average used to replace u_i^n is first order accurate, hence so is the method
- However, it is stable for $0 \leq |C| \leq 1$

Lax-Wendroff method

- This method again uses averages to allow for a method which uses u_{i-1}, u_i and u_{i+1}
- In this case, it is the x-derivative which is averaged, but with a weighted average based on C

$$\frac{\partial u}{\partial x} = \beta_1 \frac{u_i^n - u_{i-1}^n}{\Delta x} + \beta_2 \frac{u_{i+1}^n - u_i^n}{\Delta x}$$

  where

$$\beta_1 = \frac{1}{2}(1+C), \qquad \beta_2 = \frac{1}{2}(1-C),$$

- This gives an update scheme which can be written

$$u_i^{n+1} = \frac{1}{2}C(1+C)\,u_{i-1}^n + \left(1-C^2\right)u_i^n - \frac{1}{2}C(1-C)\,u_{i+1}^n$$

- Despite being constructed from first-order approximations, error terms cancel - this is a second-order method
- It is also stable for $0 \leq |C| \leq 1$

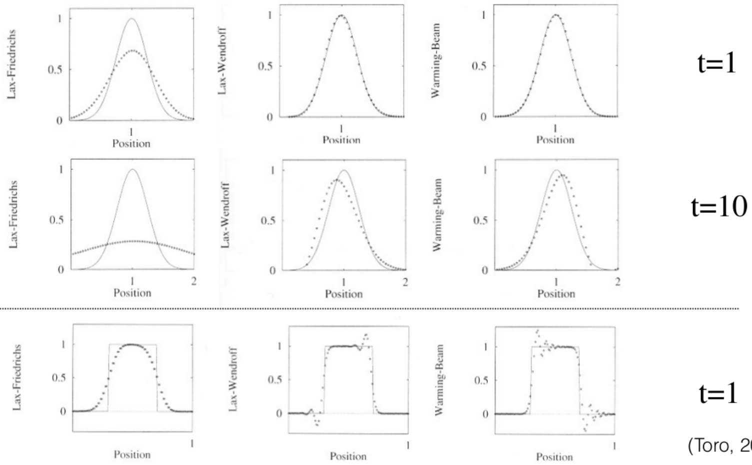
Warming-Beam method

- This method is more restrictive, like the upwind method, it must be defined differently depending on the sign of a
- For $a > 0$ we have

$$u_i^{n+1} = -\frac{1}{2}C\left(1 - C\right)u_{i-2}^n + C\left(2 - C\right)u_{i-1}^n + \frac{1}{2}\left(C - 1\right)\left(C - 2\right)u_i^n$$

- As might be suggested from the size of the stencil, this is a second order method
- However, it uses information from cell x_{i-2}, which means it is stable for $0 \le |C| \le 2$

Comparison of the different methods



(Toro, 2009)

The modified equation

- When we discretise the derivatives in the advection equation, we are actually writing (and solving) our equation as:

$$\frac{\partial u}{\partial t} + a\frac{\partial u}{\partial x} = A_1\frac{\partial^2 u}{\partial x^2} + A_2\frac{\partial^3 u}{\partial x^3} + \ldots$$

- The constant terms A_m have dependence on $(\Delta x)^m$ and vanish in the limit $\Delta x \to 0$
- For example, the first constant term for the upwind scheme is

$$A_1 = \frac{1}{2}\Delta x a\left(1 - |C|\right)$$

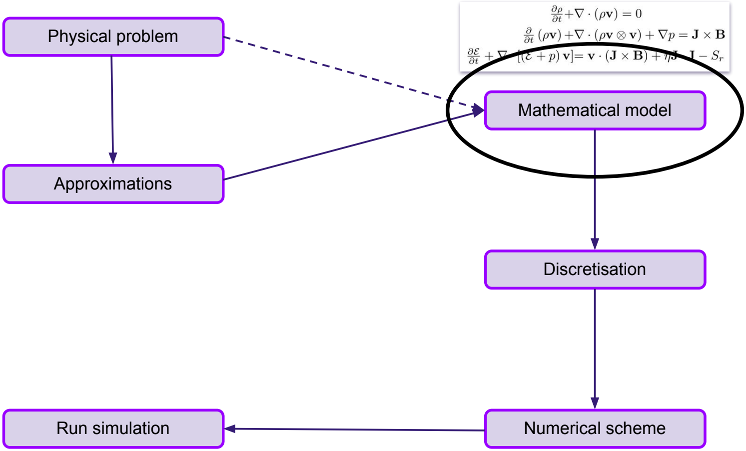
  whilst for second-order methods $A_1 = 0$
- The second derivatives are **diffusion** terms (similar to viscosity) and the third order terms are **dispersion**
- This causes smearing of solutions for numerical methods, due to the higher-order errors - commonly called **numerical viscosity** due to the similar diffusion-like processes

What about time derivatives?

- So far, when focusing on errors and cut-off terms, we have considered those relating to Δx
- Similarly, we have classed numerical methods which use second-order spatial derivatives, but first-order time derivatives, as ‘second-order methods’
- This is a deliberate choice, resulting from the application of these methods to **non-linear problems**
- The constant terms for the higher-order spatial derivatives are dependent on Δx and the wave speed (a)
- The constant terms for the higher-order temporal derivatives are dependent on Δt
- Non-linear behaviour is typically associated with **high wave speeds**, and therefore errors are dominated by those in the spatial derivatives
- When considering numerical methods for non-linear equations and systems, the order of accuracy is therefore (usually) governed by the approximations used for spatial derivatives

Outline

Burgers' equation



Moving to non-linear equations

- Being able to solve the advection equation numerically is good, but since we can solve it exactly, it is not very exciting...
- We now apply the techniques we have considered to non-linear equations
- For a non-linear hyperbolic PDE, discontinuous solutions form, even from smooth initial data
- It is a lot harder to determine what will happen at a given point in time for this sort of equation - numerical solutions are actually useful!

Burgers' equation

$$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} = 0$$

- Probably the simplest non-linear PDE, and an obvious extension of the advection equation, where the flow velocity is now the variable u
- Recall this equation came from constant-pressure inviscid assumption in the Navier Stokes equations
- Named after Burgers, though first identified by Bateman, so sometimes called the Bateman-Burgers equation
- Can be written in conservation form

$$\frac{\partial u}{\partial t} + \frac{\partial}{\partial x}\left(\frac{1}{2}u^2\right) = 0$$

Burgers' equation - hyperbolicity

$$\frac{\partial u}{\partial t} + \frac{\partial f}{\partial u}\frac{\partial u}{\partial x} = 0 \qquad \rightarrow \qquad \frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} = 0$$

- As with the advection equation, it is trivial to see that Burgers' equation is hyperbolic
- The Jacobian has the single eigenvalue, u

Burgers' equation - characteristic form

$$\frac{\partial \mathcal{V}}{\partial t} + \Lambda \frac{\partial \mathcal{V}}{\partial x} = 0 \qquad \to \qquad \frac{\partial u}{\partial t} + u \frac{\partial u}{\partial x} = 0$$

- As with the advection equation, the characteristic form of Burgers' equation is trivial to identify
- Unlike the advection equation, we cannot use this to write down an exact solution of the form $f(x - at)$
- However, we can still use the method of characteristics to identify behaviour about the solution

Behaviour along characteristics

- We consider a general equation in characteristic form,

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

- Recall that when identifying behaviour of the characteristic variable along characteristics, this gave us three relationships

$$\frac{\mathrm{d}\mathcal{V}}{\mathrm{d}s} = 0, \qquad \frac{\mathrm{d}t}{\mathrm{d}s} = 1, \qquad \frac{\mathrm{d}x}{\mathrm{d}s} = \Lambda(\mathcal{V}),$$

- The first equation states that along the characteristic, $\mathcal{V}$ is constant, and the second that $s = t$
- However, this implies that along the characteristic, $\Lambda(\mathcal{V})$ must also be constant
- Therefore

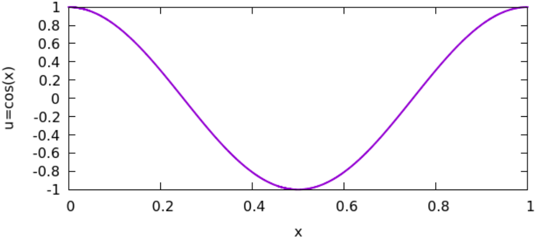
$$\frac{\mathrm{d}x}{\mathrm{d}s} = \frac{\mathrm{d}x}{\mathrm{d}t} = \mathrm{const}$$

i.e. characteristics are constant lines in the $x - t$ plane, with slope $\Lambda(u(x(0), 0)) = \Lambda(u_0)$

Characteristics for Burgers' equation

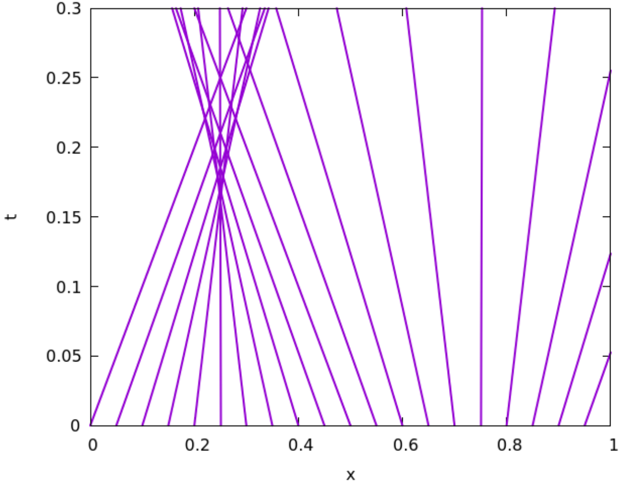
- We consider an initial value problem for Burgers' equation

$$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} = 0, \qquad u(x,0) = u_0(x) = \cos(2\pi x)$$



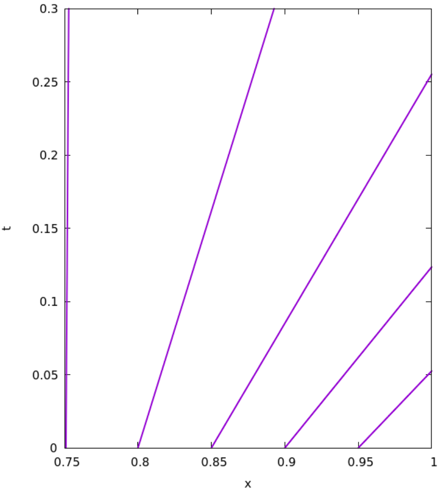
- We hope that by viewing the characteristics of this PDE, we will understand what the solution does
- What will the plot look like?

Characteristics for Burgers' equation



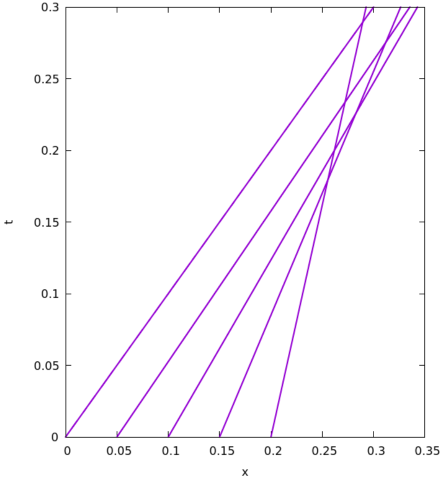
What happened?

- Consider only the characteristics that start at $x > 0.75$
- They become spaced out - the solutions is undergoing **expansion**
- Applicability of Burgers' equation to traffic flow is partly visible
- In a line of cars, if each car is faster than the one behind it, the line spreads out (the traffic becomes less dense)



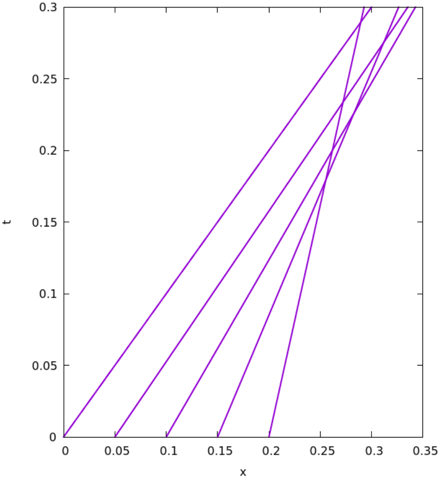
What happened?

- Consider only the characteristics that start at $x < 0.25$
- They become bunched together - the solutions is undergoing **compression**
- However, at some point, the characteristics cross
- Since u is constant along characteristics, this would imply that at the point they cross, the solution has two values
- By itself, this would be an **ill-posed** problem, to understand what is really happening, we return to the traffic flow problem



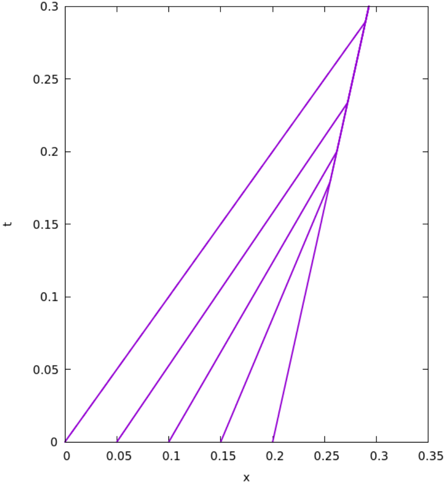
What happened?

- Considering the traffic flow application, in this case, each car is going slower than the one behind
- It is assumed that cars cannot overtake or pass through each other
- Once a car reaches the slower one in front, it now has to match the speed
- The traffic speed suffers an abrupt change caused by this slowest car - there is a **discontinuity**
- At this point, the underlying PDE is no longer valid - the gradient does not exist



What happened?

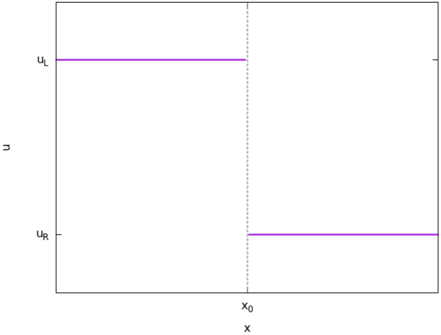
- Characteristic lines must be modified to deal with the formation of a discontinuity
- The solution we show here is **not** a solution to Burgers' equation, but is a (sort of) realistic solution to a traffic flow problem with five cars
- The assumption we make here is that the leading car will **always** maintain a constant speed - based on driver behaviour, not on underlying mathematics
- Mathematics tells us how discontinuities form in Burgers' equation, and reality tells us that traffic flow modelling may require a more detailed model



Wave types for Burgers' equation

- This initial investigation of Burgers' equation has shown us the two different types of behaviour that can arise in solutions
- Each type of behaviour is referred to as a **wave**
- Expansion generates **rarefaction waves**, also referred to as expansion waves, or an expansion fan (due to the pattern of the characteristics in the wave)
- Compression generates **shock waves**
- Our example with $u_0(x) = \cos(x)$ did demonstrate all these waves, but their interaction was complex
- In order to better understand what happens over the two wave types, we will return to the Riemann problem

The Riemann problem for Burgers' equation



- We have all the necessary components for a Riemann problem:
- A conservation equation:

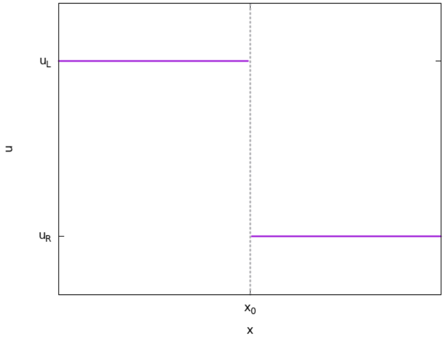
$$\frac{\partial u}{\partial t} + \frac{\partial}{\partial x}\left(\frac{1}{2}u^2\right) = 0$$

- and piece-wise continuous initial data:

$$u(x,0) = u_0(x) = \begin{cases} u_L & x < x_0 \\ u_R & x > x_0 \end{cases}$$

- The advantage of using the Riemann problem is it will generate a single wave, **either** a rarefaction or a shock, dependent on the initial data

Shock waves



- If $u_L > u_R$ then the characteristic lines will meet; a shock wave will be formed
- From the characteristics, we know that to the left of the shock, we have $u = u_L$, to the right of the shock, $u = u_R$,
- To fully describe the problem, all we need is a characteristic associated with the shock wave

- For this, we can use **jump conditions**, recall

$$\mathbf{f}(\mathbf{u}_R) - \mathbf{f}(\mathbf{u}_L) = S\left(\mathbf{u}_R - \mathbf{u}_L\right)$$

- What is S for Burgers' equation?

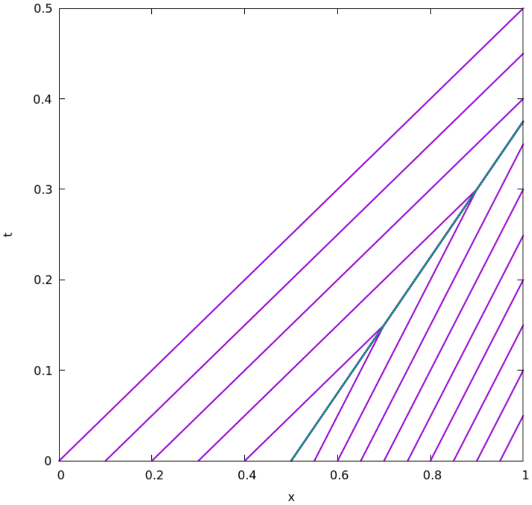
Shock wave characteristics

- $S = \frac{1}{2}(u_L + u_R)$
- Here, we have initial data

$$u_0(x) = \begin{cases} 2 & x < 0.5 \\ 1 & x > 0.5 \end{cases}$$

- This gives a shock speed of $S = 1.5$
- In the $x - t$ diagram, the shock wave is a characteristic

$$t = \frac{x - x_0}{S}$$

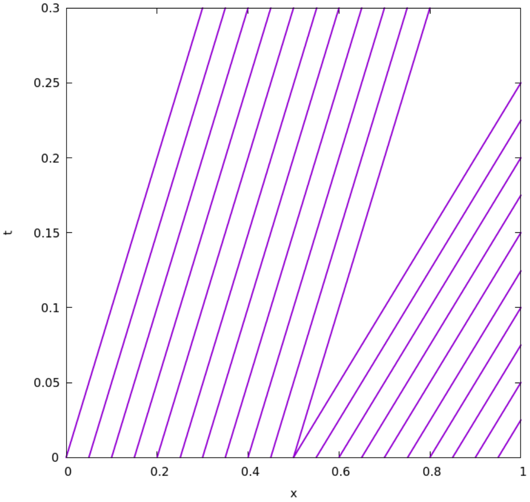


Rarefaction wave characteristics

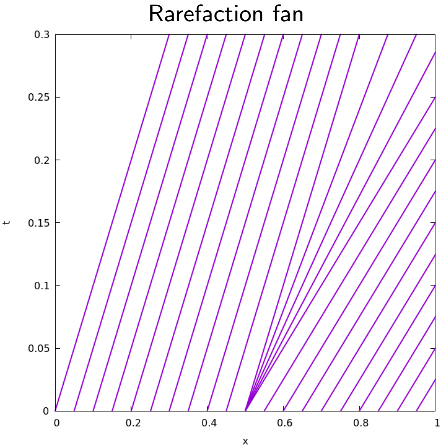
- We now consider $u_L < u_R$
- Here, we have initial data

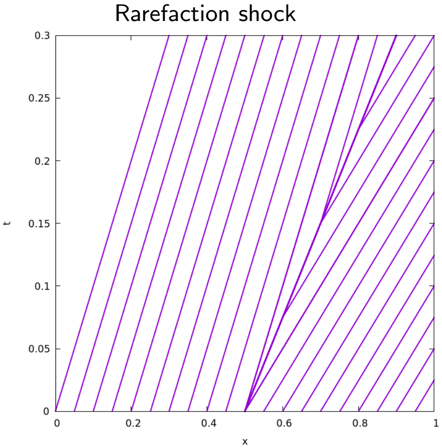
$$u_0(x) = \begin{cases} 1 & x < 0.5 \\ 2 & x > 0.5 \end{cases}$$

- The behaviour at almost all points is clear, but what happens to characteristics at the point $x = 0.5$?
- In other words - how do we fill the gap between the diverging characteristics?
- Mathematically, more than one solution exists...



Characteristic within a rarefaction wave





- Note - both these solutions **conserve** u

Weak solutions

- The fact that we have two possible solutions is a sign of an ill-posed problem - these are two **weak solutions** of the problem
- We need to impose some criterion to define which of the two solutions is correct - this originates in the physics of the problem, in this case, solution stability
- Consider the initial data for $u_L < u_R$ and small δ

$$u_0(x) = \begin{cases} u_L & x < 0.5 - \delta \\ (u_L + u_R)/2 & 0.5 - \delta < x < 0.5 + \delta \\ u_R & x > 0.5 + \delta \end{cases}$$

- This can be considered a **perturbation** of the Riemann problem initial data
- If we assume the solution is a rarefaction fan, then the characteristic diagram is unchanged - the solution is stable under perturbation (this can be proven mathematically)
- What about the rarefaction shock?

The entropy condition

- If the solution between two states with $u_L < u_R$ is a rarefaction shock, then for the initial data shown previously, we have two of these

$$S_1 = \frac{3u_L + u_R}{4}, \qquad S_2 = \frac{u_L + 3u_R}{4}$$

- We have a very different solution given this slight change - the solution is unstable under perturbation (again, this can be proven mathematically)
- Our additional criterion for our solutions is that the solutions must be stable under perturbation (a physically reasonable assumption)
- This can often be phrased as a condition on shock waves, rather that rarefaction, a shock wave can only exist if

$$\mathbf{f}\left(\mathbf{u}_L\right) > S > \mathbf{f}\left(\mathbf{u}_R\right)$$

- This is known as the **entropy condition** - in systems for which entropy has a physical meaning, an entropy condition-violating shock leads to a spontaneous increase in entropy (not allowed thermodynamically)

Techniques for solving Burgers' equation

- We now know how to describe the solution to Burgers' equation - it is straightforward for a Riemann problem, and trickier for other initial data, such as the cosine
- We will now move on to techniques to solve Burgers' equation numerically
- This is a non-linear equation, and, as a result, we need to be careful with how we choose to solve it
- This goes beyond our choice of stencil and discrete representation - we will see that the **conservation form** of the equation is needed to capture non-linear behaviour
- In moving to conservation form, we shall move from the finite difference form we've seen so far to **finite volume methods**