Statistics and Numerical Methods, Tsinghua University

Partial Differential Equations

Xuening Bai (白雪宁)

Institute for Advanced Study (IASTU) & Department of Astronomy (DoA)

清華大學

Tsinghua University

Oct. 29, 2024

Outline

- General concepts
- Initial value problem: hyperbolic equations

  Finite difference method

  Finite volume method
- Initial value problem: parabolic equations
- Boundary value problem: elliptic equations

  Multigrid method

What are partial differential equations

Differential equations for a multivariable function u that involve more than one partial derivatives.

$$\mathcal{N}(u, \boldsymbol{x}) = 0 \ , \quad \boldsymbol{x} \in \Omega$$

N is some differential operator, generally non-linear.

Much more complex than ODEs. There are a wide variety of classes with substantially different properties.

No general theory to the mathematical properties of PDEs.

No general numerical algorithms that are good for all.

Here we only focus on simple types of PDEs that arise naturally from classical physics.

General concepts

We mainly consider linear PDEs:

The PDE can be written using linear combinations of u and its derivatives (but the coefficients can be nonlinear functions of its independent variables).

The order of a PDE is set by the order of its highest derivative.

Order can be reduced as in ODEs, but does not necessarily give much benefit.

Examples:

$$\frac{\partial^2 u}{\partial x^2} - y\frac{\partial^2 u}{\partial y^2} + x^2 u = xy^2$$

This is a 2nd order linear PDE.

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

This is a 1st order nonlinear PDE (Burger's equation)

Initial value problem vs boundary value problem

Just as in ODEs, proper initial/boundary conditions must be supplied to ensure a unique solution.

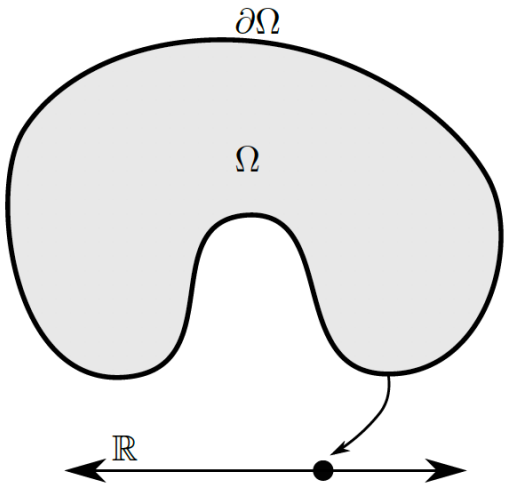
PDEs resulting from physical systems usually has time t as one independent variable, and we solve for time evolution.

- Solving such systems is usually considered as initial value problems.
- Similar to IVPs for ODEs, but with additional spatial derivatives.
- The spatial dimension must be supplemented by boundary conditions.

When seeking for stationary solutions but in multi-dimensions, the resulting PDEs are considered as boundary value problems.

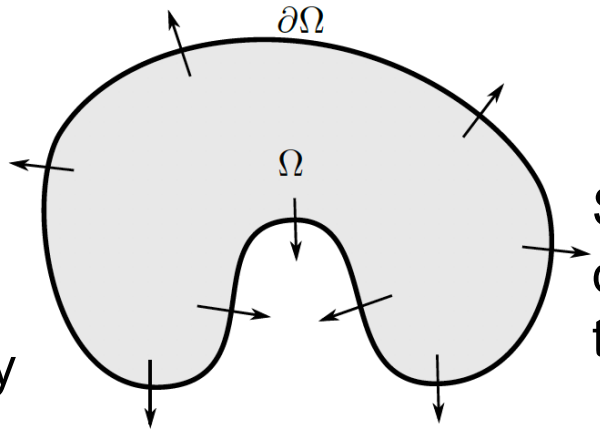
Types of boundary conditions

PDEs are solved within a domain (can be regular/irregular), supplemented by boundary conditions (BC).



Dirichlet

Directly specify the values of u over the boundary.



Neumann

Specify the value of u derivative orthogonal to the boundary.

There are also boundary conditions of mixed type.

There can also be periodic BC in regular domains.

Standard types of PDEs

Consider 2nd order linear PDEs, their general form reads:

$$a\partial_{xx}^2 u + b\partial_{xy}^2 u + c\partial_{yy}^2 u + d\partial_x u + e\partial_y u + fu = g$$

It can be categorized based on the discriminant:

$$b^2 - 4ac \begin{cases} < 0 & \rightarrow \quad \text{Elliptic} \\ = 0 & \rightarrow \quad \text{Parabolic} \\ > 0 & \rightarrow \quad \text{Hyperbolic} \end{cases}$$

When their coefficients are nonlinear, or when the PDE itself is nonlinear, the nature of the system may change between different regimes.

Standard types of PDEs

Prototype of elliptic PDE:

Poisson equation: $\nabla^2 u = f$ (self-gravity)

Prototype of parabolic PDE:

Diffusion equation: $\partial_t u = D\partial^2_{xx} u$ (thermal conduction)

Prototype of hyperbolic PDE:

Wave equation: $\partial^2_{tt} u - c^2 \partial^2_{xx} u = 0$

Solution strategies

- Finite difference

  Use finite difference to approximate the differentiation operators.

- Finite volume

  Developed for systems of (nonlinear) hyperbolic conservation laws.

- Finite element

  Most widely used in engineering, and is flexible with complex geometry.

- Spectral methods

  Solve equations in a function space spanned by a set of basis functions.

- Meshfree methods

  Usually involves Lagrangian particles that track fluid motion.

There are also hybrid methods, most notably Discontinuous Galerkin method.

Outline

- General concepts
- Initial value problem: hyperbolic equations

  Finite difference method

  Finite volume method
- Initial value problem: parabolic equations
- Boundary value problem: elliptic equations

  Multigrid method

Hyperbolic equations

Prototype of hyperbolic PDE:

Wave equation: $\partial_{tt}^2 u - c^2 \partial_{xx}^2 u = 0$

Many familiar equations can be reduced to this form.

Maxwell's equations in vacuum:

$$\frac{1}{c}\frac{\partial \boldsymbol{E}}{\partial t} = \nabla \times \boldsymbol{B} \ ,$$
$$\frac{1}{c}\frac{\partial \boldsymbol{B}}{\partial t} = -\nabla \times \boldsymbol{E} \ ,$$
$$\nabla \cdot \boldsymbol{E} = \nabla \cdot \boldsymbol{B} = 0 \ .$$

$$\frac{\partial^2 \boldsymbol{B}}{\partial t^2} + c^2 \nabla^2 \boldsymbol{B} = 0$$
$$\frac{\partial^2 \boldsymbol{E}}{\partial t^2} + c^2 \nabla^2 \boldsymbol{E} = 0$$

Hyperbolic equations

Prototype of hyperbolic PDE:

Wave equation: $\partial_{tt}^2 u - c^2 \partial_{xx}^2 u = 0$

By defining (assuming c is constant) $\xi \equiv x - ct \ , \ \eta \equiv x + ct$

$$\Rightarrow \quad \frac{\partial^2 u}{\partial \xi \partial \eta} = 0 \qquad \xrightarrow[\text{solution}]{\text{general}} \qquad u(x,t) = F_1(x - ct) + F_2(x + ct)$$

F_1, F_2 are solutions to the following linear advection equations:

$$\frac{\partial u}{\partial t} + c\frac{\partial u}{\partial x} = 0 \ , \quad \frac{\partial u}{\partial t} - c\frac{\partial u}{\partial x} = 0 \ .$$

These are simpler and more fundamental, which we will focus on next.

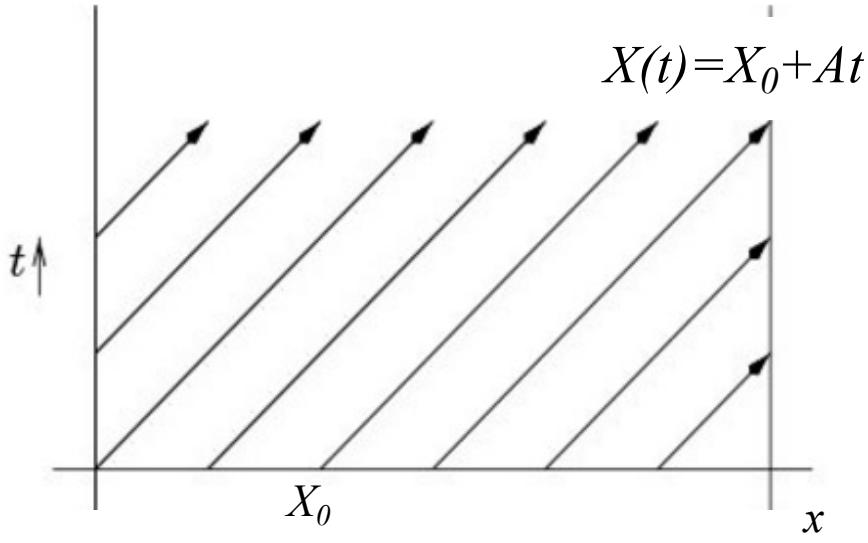
Linear advection equation (const coefficient)

Consider a linear advection eq with constant A: $\partial_t u + A\partial_x u = 0$

Solution: $u(x,t) = u_0(x - At)$

The solution is constant along the ray (called the characteristic curve):

$$X(t) = X_0 + At$$



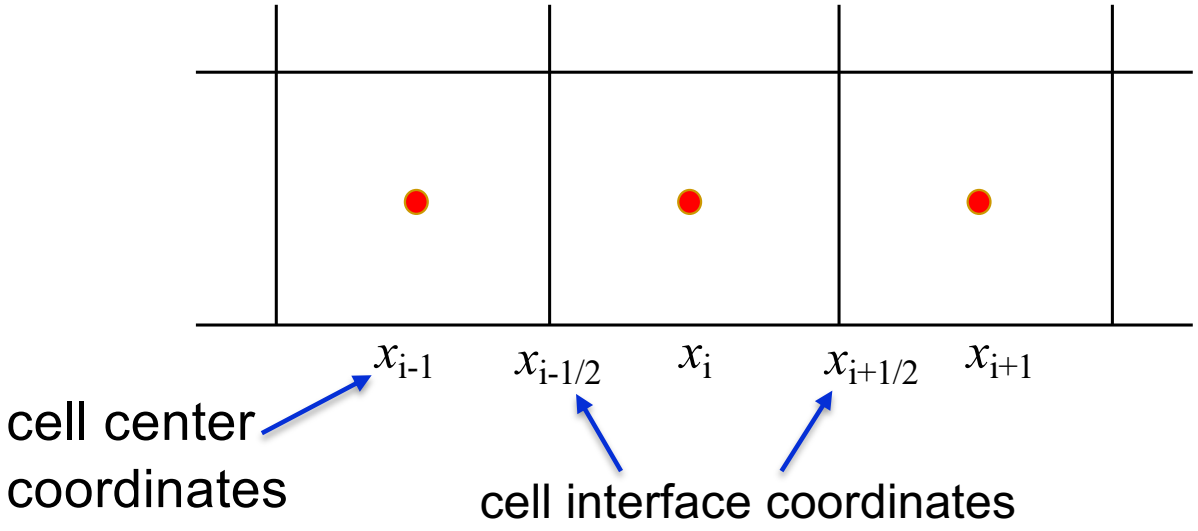
Proof:

$$\frac{d}{dt}u(X(t),t) = \partial_t u(X(t),t) + X'(t)\partial_x u(X(t),t) = \partial_t u(X,t) + A\partial_x u(X,t) = 0$$

Discretization

We consider grid-based methods, which inevitably involves discretization and use finite difference to approximate time/spatial derivatives.

For simplicity, we focus on Cartesian grid with uniform grid spacing Δx.



physical quantities are stored at cell centers

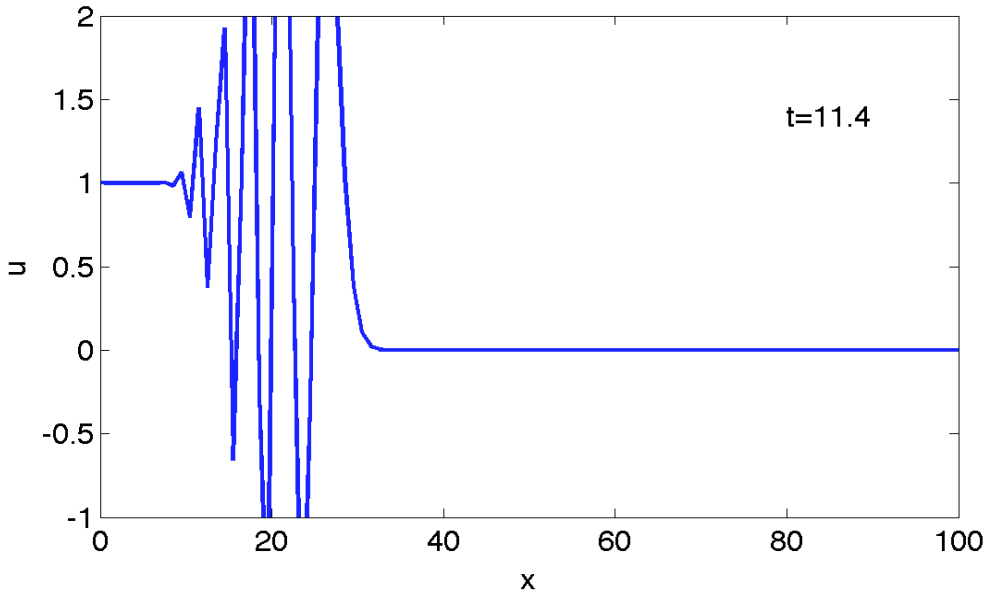
Forward-time central-space (FTCS)

$$\partial_t u + A\partial_x u = 0$$

Initial condition: u=1 (x<20), u=0 (x>20), A=1.

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

1st order in time
2nd order in space



The method is unconditionally unstable!

von Neumann stability analysis

Assume constant coefficient, constant time and space interval (Δt, Δx).

Assume the domain is infinite, i.e., no boundary conditions.

Pick up one Fourier mode as an eigenmode: $u_i^n = ae^{\mathrm{i}k(i\Delta x)}$

Apply the numerical scheme to obtain solution at next step:

$$u_i^{n+1} = C_k u_i^n$$

C_k: amplification factor (complex number)

Stability requires: $|C_k| \leq 1$ for any k.

Actual stability could be different when the problem is nonlinear / has non-periodic BC, etc., but this analysis is extremely useful in practice.

von Neumann stability analysis

For FTCS method: $$\frac{u_i^{n+1} - u_i^n}{\Delta t} = -A\left(\frac{u_{i+1}^n - u_{i-1}^n}{2\Delta x}\right)$$

It is straightforward to obtain:

$$u_i^{n+1} = u_i^n - \frac{A\Delta t}{\Delta x}\frac{e^{\mathrm{i}k\Delta x} - e^{-\mathrm{i}k\Delta x}}{2}u_i^n$$

Therefore, $C_k = 1 - \mathrm{i}\frac{A\Delta t}{\Delta x}\sin(k\Delta x)$

No matter what timestep we choose, $|C_k|>1$!

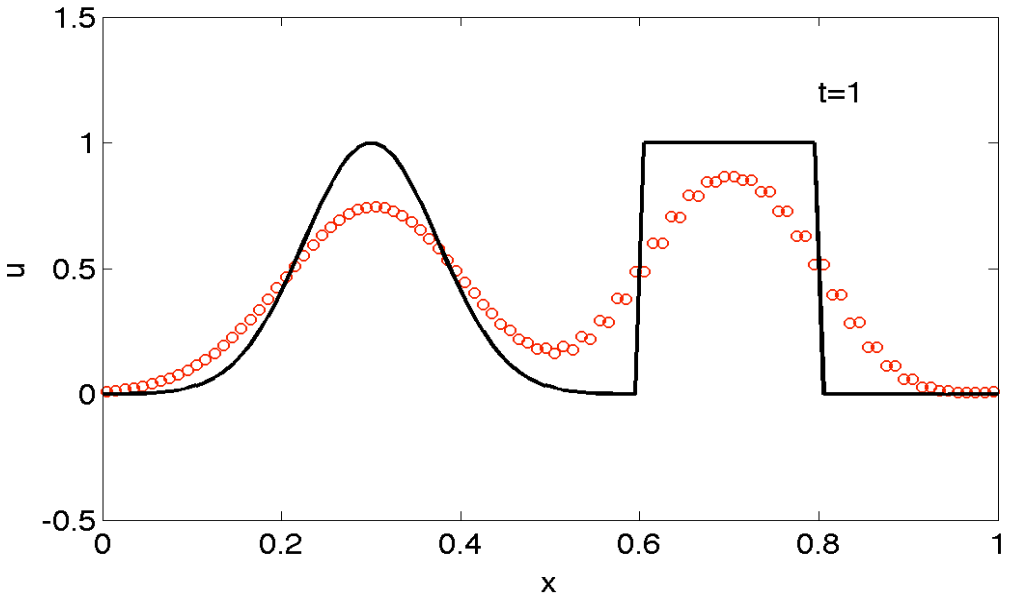
This means the solution is always exponentially amplified in time, i.e., it is unconditionally unstable!

Lax-Friedrichs (LF) method

$$\partial_t u + A\partial_x u = 0$$

IC: one Gaussian, one square waves, A=1, periodic BC.

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



Effectively, added extra numerical diffusion to stabilize FTCS.

The method is stable, but VERY diffusive!

von Neumann stability analysis

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

Take a Fourier mode of the form: $u_i^n = ae^{\mathrm{i}k(i\Delta x)}$

Solution at the next timestep is: $u_i^{n+1} = C_k u_i^n$

For the Lax-Friedrich method, it is straightforward to obtain:

$$C_k - \cos(k\Delta x) = -\mathrm{i}\frac{A\Delta t}{\Delta x}\sin(k\Delta x)$$

Clearly, the method is stable ($|C_k|<1$) if $\frac{A\Delta t}{\Delta x} < 1$, or $\boxed{\Delta t < \frac{\Delta x}{A}}$

Why is this method stable but diffusive?

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

Rewriting its formulation we obtain:

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

$$\sim \frac{\Delta x^2}{2\Delta t}\partial_{xx}u$$

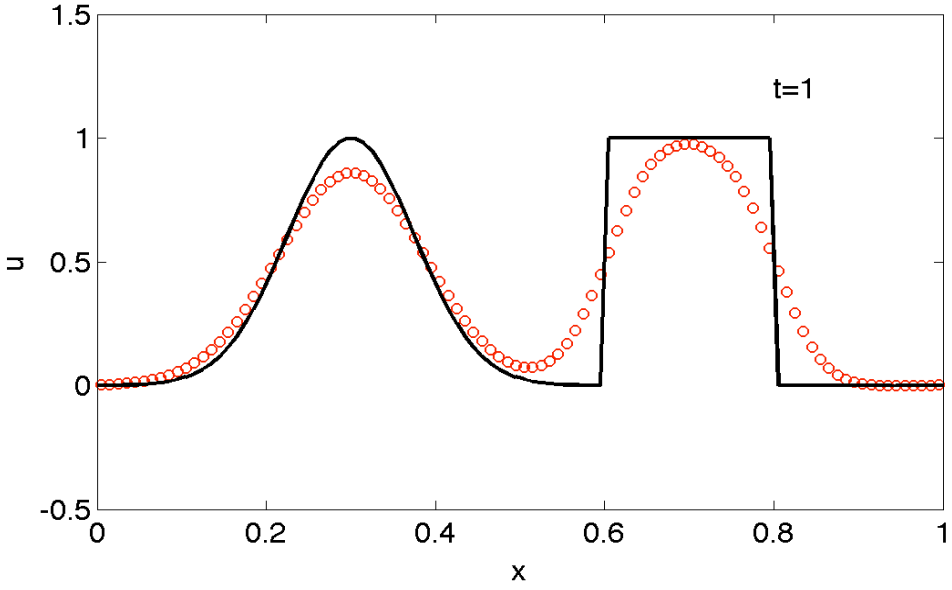
Essentially, this adds (a lot of) numerical diffusion.

Upwind method: more physically motivated

$$\partial_t u + A\partial_x u = 0$$

IC: one Gaussian, one square waves, A=1, periodic BC.

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



The method is stable and less diffusive, though still only first order accurate.

Can be improved using higher-order spatial interpolation schemes.

von Neumann stability analysis

$$\frac{u_i^{n+1} - u_i^n}{\Delta t} = -A(u_i^n - u_{i-1}^n)/\Delta x \quad (A \geq 0)$$

This can be rewritten as: $u_i^{n+1} = (1-\mu)u_i^n + \mu u_{i-1}^n$ where $\mu \equiv \frac{A\Delta t}{\Delta x}$

Setting $u_i^n = ae^{\mathrm{i}k(i\Delta x)}$ and $u_i^{n+1} = C_k u_i^n$, we find

$$C_k = (1-\mu) + \mu e^{-\mathrm{i}k\Delta x}$$

and hence $|C_k|^2 = (1-\mu+\mu e^{-\mathrm{i}k\Delta x})(1-\mu+\mu e^{\mathrm{i}k\Delta x})$

$$= 1 - 2\mu(1-\mu)(1-\cos k\Delta x)$$

Stability requires $\mu(1-\mu) > 0$, or again $\boxed{\Delta t < \frac{\Delta x}{A}}$

Towards higher order of accuracy

$$\partial_t u + A\partial_x u = 0$$

We would like to construct a scheme that minimize numerical diffusion.

For a smooth solution $u(x,t)$, we can Taylor expand it as

$$\begin{aligned} u(x,t+\Delta t) &= u(x,t) + \Delta t \partial_t u(x,t) + \frac{1}{2}\Delta t^2 \partial_{tt} u(x,t) + \ldots \\ &= u(x,t) - A\Delta t \partial_x u(x,t) + \frac{1}{2} A^2 \Delta t^2 \partial_{xx} u(x,t) + \ldots \end{aligned}$$

This motivates us to consider the following scheme (Lax-Wendroff):

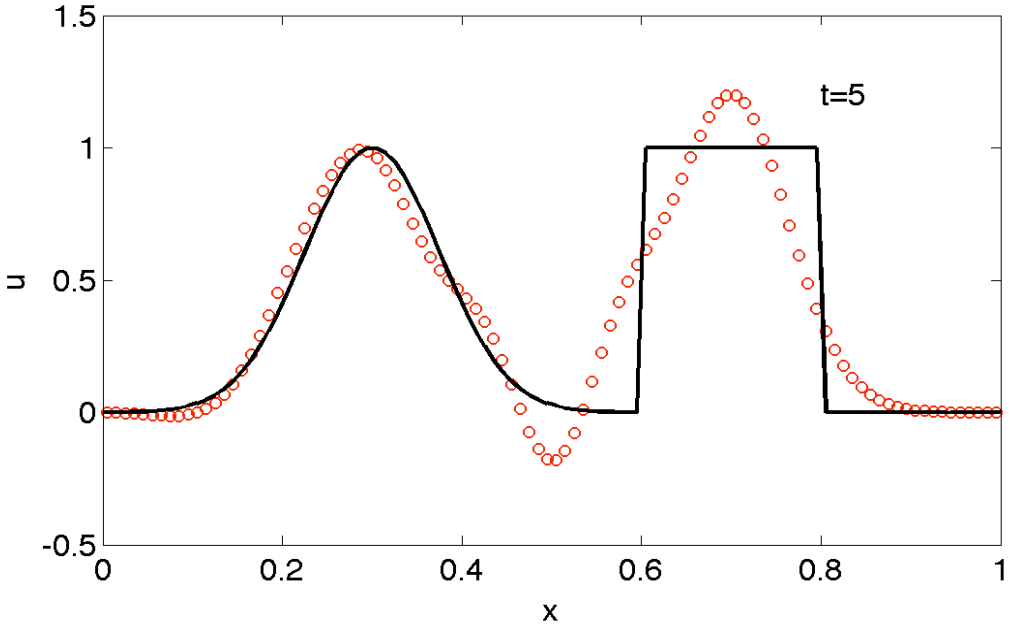
$$u_i^{n+1} = u_i^n - \frac{A\Delta t}{2\Delta x}(u_{i+1}^n - u_{i-1}^n) + \frac{A^2\Delta t^2}{2\Delta x^2}(u_{i+1}^n - 2u_i^n + u_{i-1}^n)$$

Lax-Wendroff method

$$\partial_t u + A\partial_x u = 0$$

IC: one Gaussian, one square waves, A=1, periodic BC.

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



Method is stable but:

1). Oscillatory solution at discontinuities.

2). Phase shift in the smooth region.

More later.

Outline

- General concepts
- Initial value problem: hyperbolic equations

  Finite difference method

  Finite volume method
- Initial value problem: parabolic equations
- Boundary value problem: elliptic equations

  Multigrid method

Linear advection equation: another look

Constant A can be taken into the derivative: $\frac{\partial u}{\partial t} + \frac{\partial (Au)}{\partial x} = 0$

conserved quantity

Flux of u

More generally, we have $\frac{\partial u}{\partial t} + \nabla \cdot F(u) = 0$

Conservation laws are ubiquitous in physics.

We can express conservation laws in integral form:

$$\int dV \frac{\partial u}{\partial t} + \oint_S \boldsymbol{F} \cdot d\boldsymbol{S} = 0$$

Finite volume method is designed to evolve this form of the equation such that conservation laws are preserved to machine precision.

See “Finite Volume Methods for Hyperbolic problems” by R. LeVeque.

Example: hydrodynamics

Describes the dynamics of a fluid as a continuous media:

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \boldsymbol{v}) = 0$$ mass conservation

$$\frac{\partial \rho \boldsymbol{v}}{\partial t} + \nabla \cdot (\rho \boldsymbol{v}\boldsymbol{v} + P\mathbf{I}) = \rho \boldsymbol{g}$$ momentum conservation

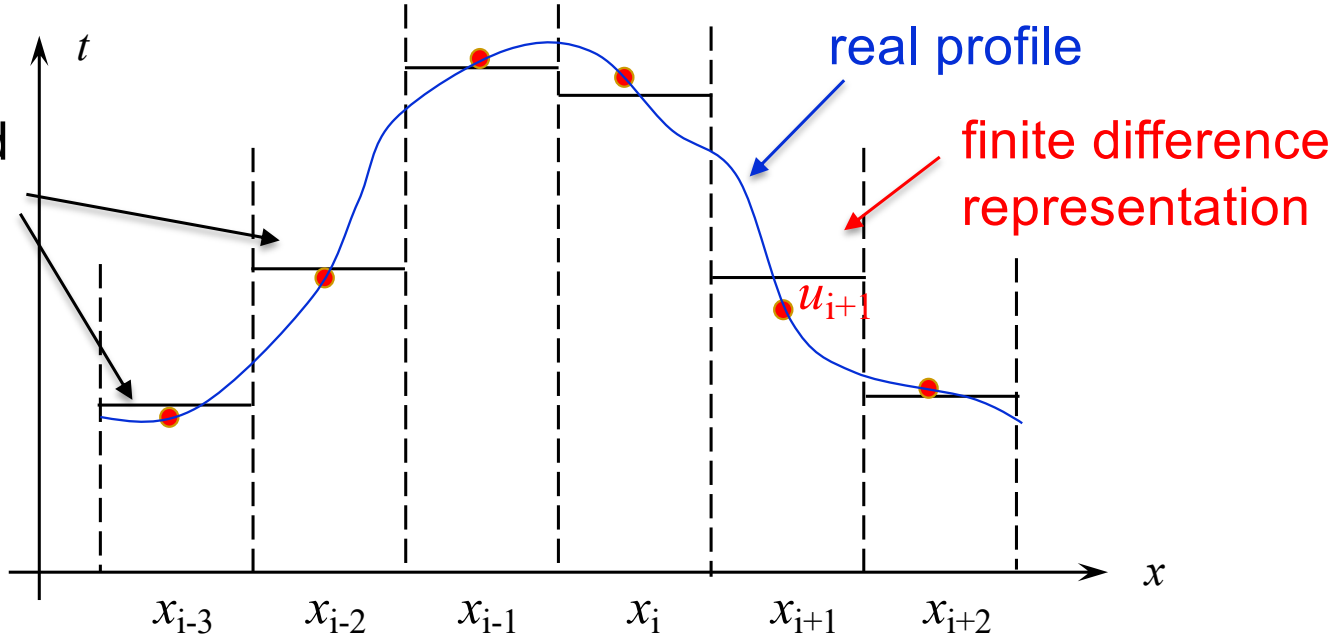
$$\frac{\partial}{\partial t}\left[\rho\left(\frac{1}{2}v^2 + \Phi + \epsilon\right)\right] + \nabla \cdot \left[\rho \boldsymbol{v}\left(\frac{1}{2}v^2 + \Phi + \epsilon\right) + P\boldsymbol{v}\right] = 0$$ energy conservation

It forms a system of 5 non-linear equations of hyperbolic conservation laws.

Despite the difficulty from nonlinearity, great insight is gained again through the analysis of linear advection equation.

Finite volume vs finite difference methods

FVM interprets grid discretization very differently.



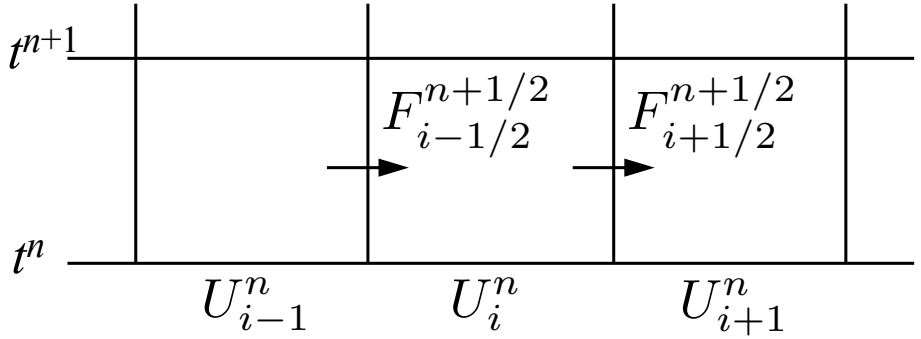
- FVM works with the integral form of the conservation laws.

Conserved variables are volume-averaged:

$$U_i^n = \frac{1}{\Delta x} \int_{x_{i-1/2}}^{x_{i+1/2}} u(x, t_n) dx$$

Finite volume method

- FVM works with the integral form of the conservation laws.



Conserved variables are volume-averaged:

$$U_i^n = \frac{1}{\Delta x}\int_{x_{i-1/2}}^{x_{i+1/2}} u(x,t_n)dx$$

Interface fluxes are time-averaged:

$$F_{i-1/2}^{n+1/2} = \frac{1}{\Delta t}\int_{t_n}^{t_{n+1}} f(u(x_{i-1/2},t))dt$$

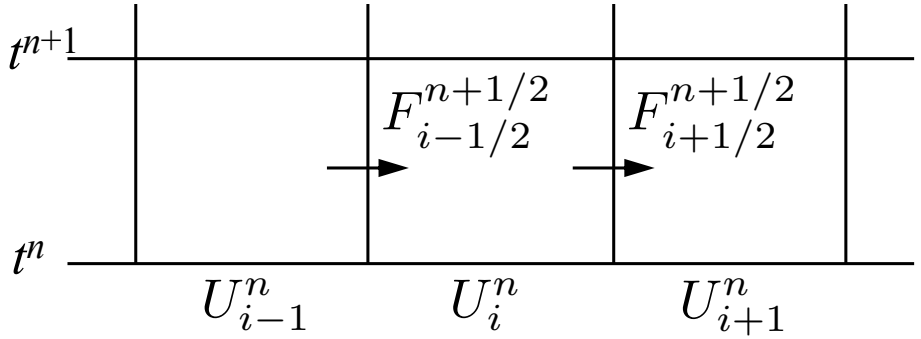
Finite volume update:

$$U_i^{n+1} = U_i^n - \frac{\Delta t}{\Delta x}(F_{i+1/2}^{n+1/2} - F_{i-1/2}^{n+1/2})$$

Conserved variables are conserved to machine accuracy.

How to compute the fluxes?

Consider a general scalar conservation law: $\partial_t u + \partial_x f(u) = 0$



We only know the volume-averaged values U.

To get the flux at cell interfaces, we essentially need to know the value of u at $x_{i+1/2}$ through some sort of averaging/interpolation.

Alternatively, need to find ways to approximate interface fluxes directly.

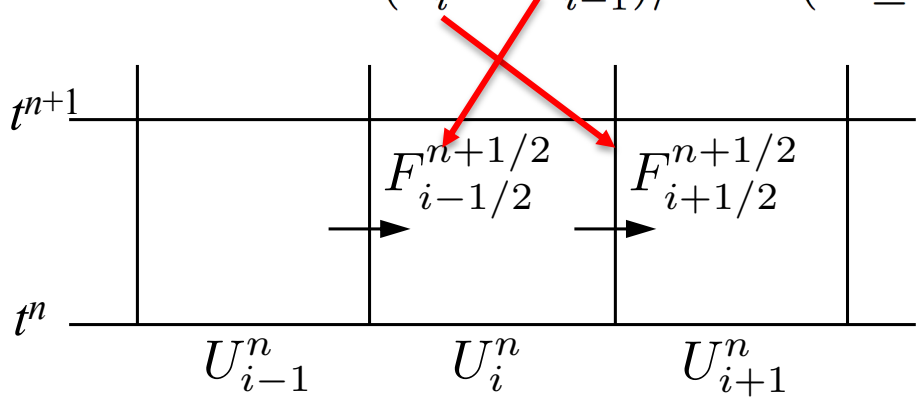
This is the key to finite volume methods.

Linear advection equation

For the linear advection equation, the aforementioned finite-difference methods can be given finite-volume interpretations:

For the upwind method:

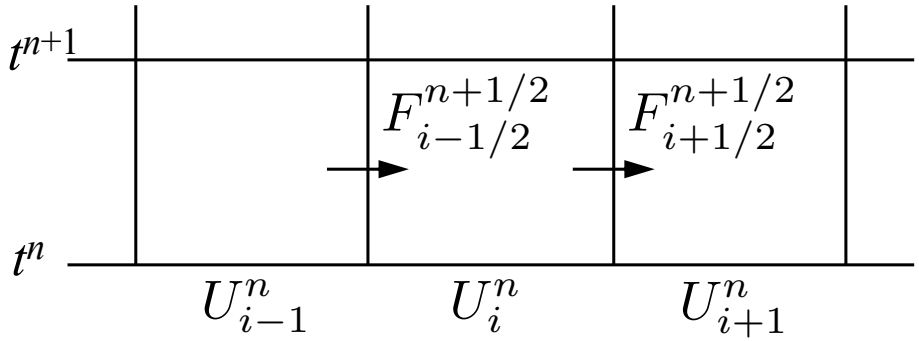
$$\frac{U_i^{n+1} - U_i^n}{\Delta t} = \begin{cases} -A(U_{i+1}^n - U_i^n)/\Delta x & (A < 0) \\ -A(U_i^n - U_{i-1}^n)/\Delta x & (A \geq 0) \end{cases}$$



Upwind flux: $F_{i-1/2} = \begin{cases} AU_{i-1} & (A \geq 0) \\ AU_i & (A < 0) \end{cases}$

Linear advection equation

Similarly, Lax-Friedrichs and Lax-Wendroff methods can also be interpreted in the finite volume framework:



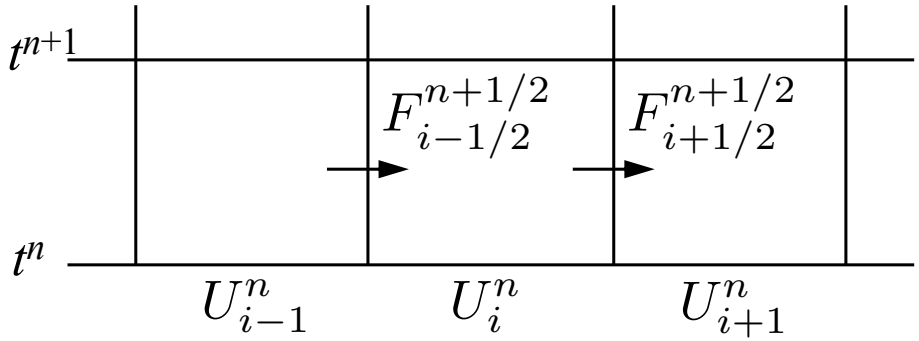
Lax-Friedrichs:

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

$$F_{i-1/2} = \frac{1}{2}A(U_{i-1}+U_i) - \frac{\Delta x}{2\Delta t}(U_i - U_{i-1})$$

Linear advection equation

Similarly, Lax-Friedrichs and Lax-Wendroff methods can also be interpreted in the finite volume framework:



Lax-Wendroff:

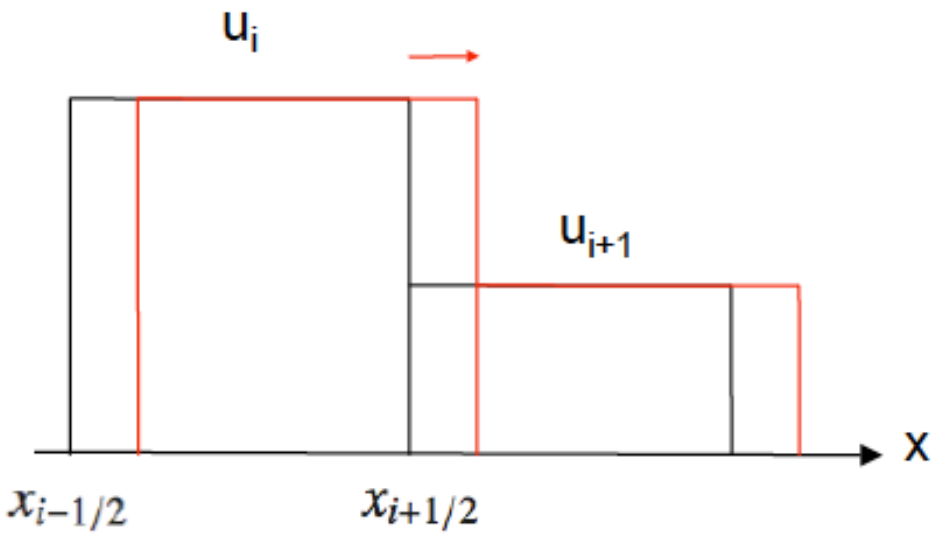
$$\frac{u_i^{n+1}-u_i^n}{\Delta t} = -A\left(\frac{u_{i+1}^n - u_{i-1}^n}{2\Delta x}\right) + \frac{A^2\Delta t}{2}\left(\frac{u_{i+1}^n - 2u_i^n + u_{i-1}^n}{\Delta x^2}\right)$$

$$F_{i-1/2} = \frac{1}{2}A(U_{i-1}+U_i) - \frac{\Delta t}{2\Delta x}A^2(U_i - U_{i-1})$$

Godunov method (Basic idea)

A finite volume method originally proposed by Godunov (1959) for solving (non-linear) equations of gas dynamics.

Reconstruct ->
Evolve ->
Average



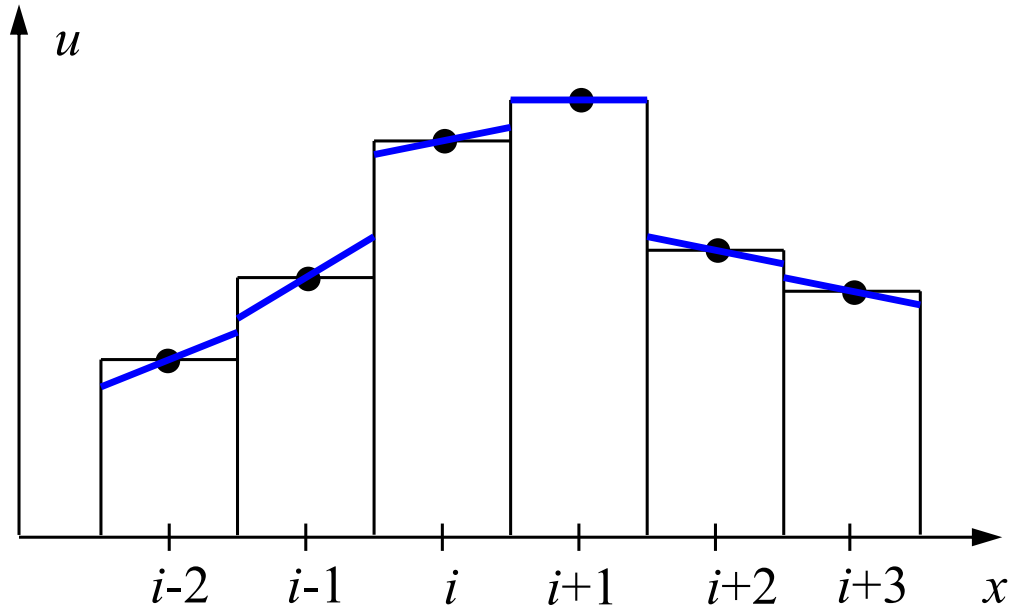
For linear advection equations, Godunov method with piecewise constant reconstruction = upwind method.

Key property: flux is properly upwinded to avoid spurious oscillations.

Toward higher order accuracy

Piecewise linear reconstruction:

$$u(x) = u_i + \sigma_i(x - x_i) \quad \text{for } (x_{i-1/2} \le x < x_{i+1/2})$$

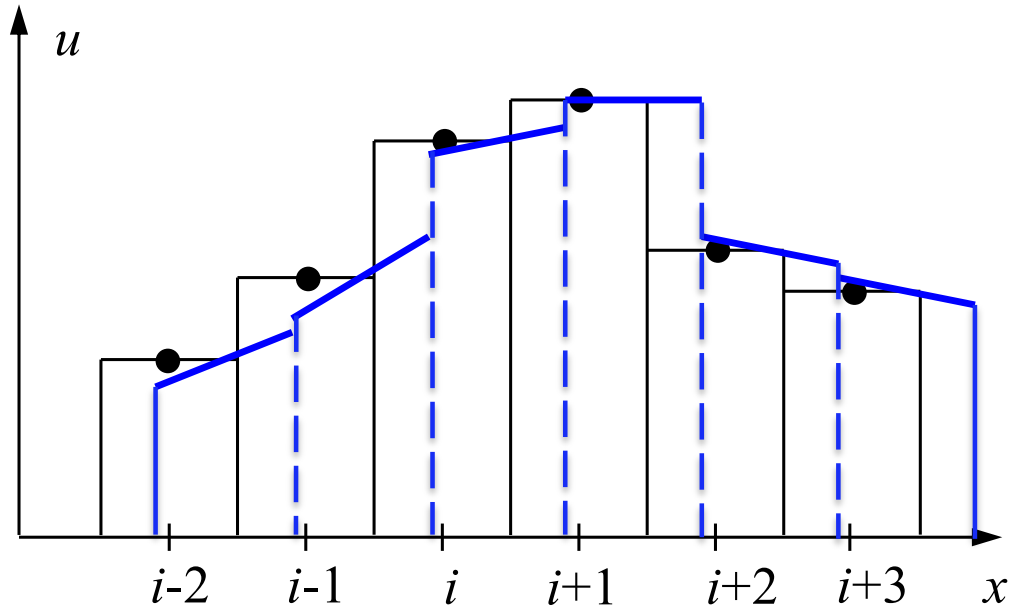


Toward higher order accuracy

Piecewise linear reconstruction:

$$u(x) = u_i + \sigma_i(x - x_i) \quad \text{for } (x_{i-1/2} \le x < x_{i+1/2})$$

Evolve reconstructed profile according to the (linear advection) equation.



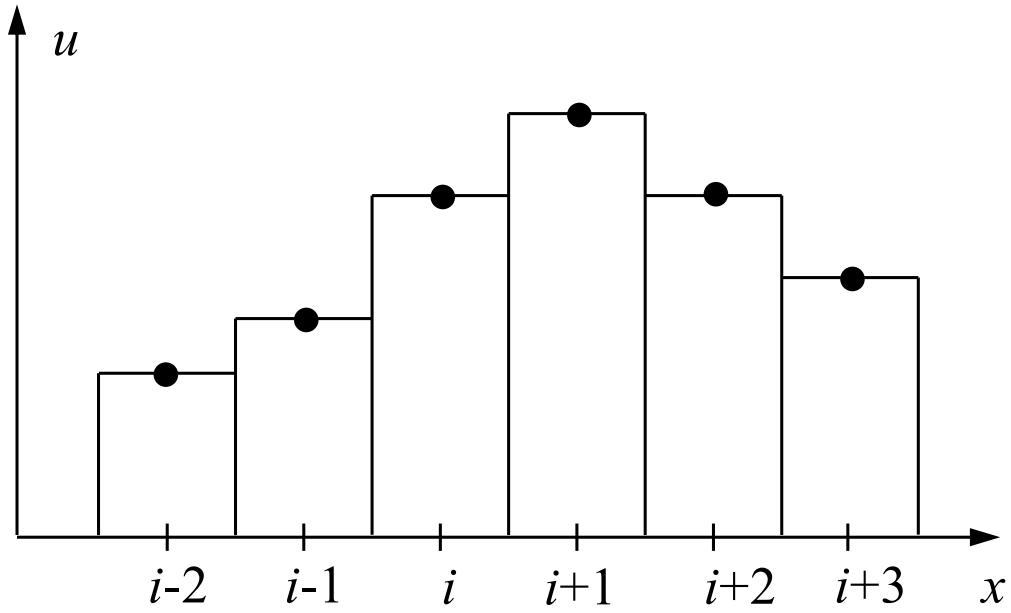
Toward higher order accuracy

Piecewise linear reconstruction:

$$u(x) = u_i + \sigma_i(x - x_i) \quad \text{for } (x_{i-1/2} \leq x < x_{i+1/2})$$

Evolve reconstructed profile according to the (linear advection) equation.

Volume average the evolved profile to the grid structure.



Can also be done at 3rd order: Piecewise-parabolic method
(Colella & Woodward, 1984)

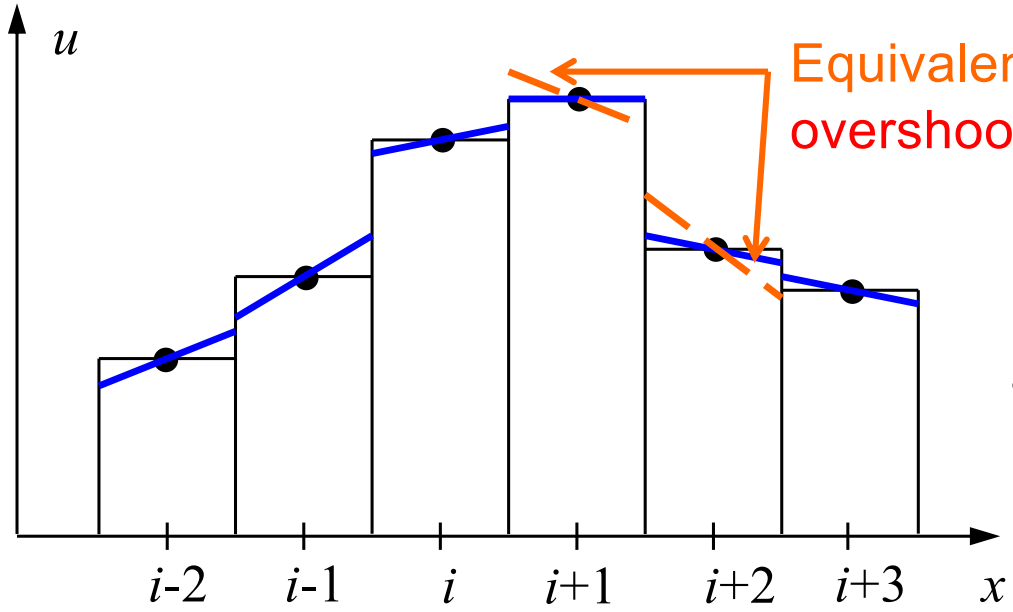
Toward higher order accuracy

How to choose the slopes?

$$u(x) = u_i + \sigma_i(x - x_i) \quad \text{for } (x_{i-1/2} \le x < x_{i+1/2})$$

Simplest choice: $\sigma_i = (u_{i+1} - u_{i-1})/2\Delta x$

This is in fact equivalent to the Lax-Wendroff method.



Equivalence of the Lax-Wendroff:
overshoots => oscillations

Need slope limiters so as to be "monotonicity preserving"

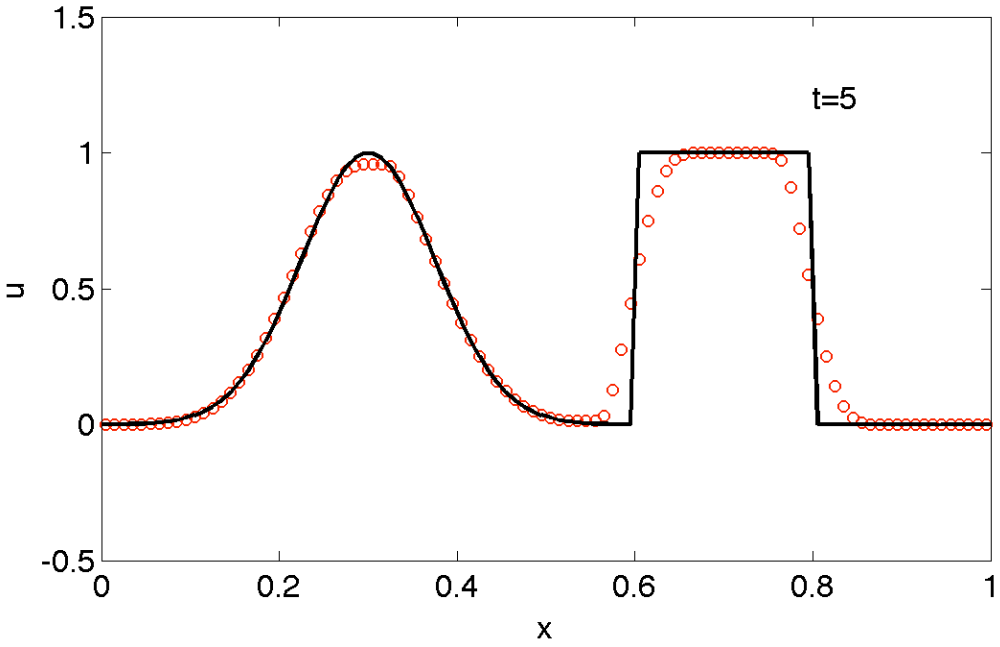
Solve the linear advection equation

Higher-order Godunov method with piecewise linear reconstruction

+the MC slope limiter.

$$\partial_t u + A\partial_x u = 0$$

Initial condition: one Gaussian, one square waves, A=1, periodic BC.



The method is stable and much more accurate:

2nd order accurate for smooth flow 1st order accurate at discontinuities.

Linear hyperbolic systems

A linear system of the form

$$\partial_t \boldsymbol{u} + \mathbf{A} \cdot \partial_x \boldsymbol{u} = 0$$

is hyperbolic if matrix **A** is diagonalizable with real eigenvalues.

Let us denote the eigenvalues by $\lambda^1 \leq \lambda^2 \leq ... \leq \lambda^m$

The matrix is diagonalizable if there is a complete set of eigenvectors such that: $A\boldsymbol{r}^p = \lambda^p \boldsymbol{r}^p$

The right-eigenvectors jointly form a matrix: $R \equiv (\boldsymbol{r}^1, \boldsymbol{r}^2, ..., \boldsymbol{r}^m)$

so that $AR = \Lambda R$ where $\Lambda \equiv \mathrm{diag}(\lambda^1, \lambda^2, ..., \lambda^m)$.

In this way, the matrix A is diagonalized as: $\boxed{R^{-1}AR = \Lambda}$

Linear hyperbolic systems

A linear system of the form

$$\partial_t \boldsymbol{u} + \mathbf{A} \cdot \partial_x \boldsymbol{u} = 0$$

is hyperbolic if matrix **A** is diagonalizable with real eigenvalues.

For any vector $\boldsymbol{u}$, we can rewrite the original equation into:

$$\partial_t (R^{-1}\boldsymbol{u}) + (R^{-1}AR)\partial_x (R^{-1}\boldsymbol{u}) = 0$$

By defining characteristic variables as $\boldsymbol{w} = R^{-1}\boldsymbol{u}$, the linear system becomes

$$\partial_t \boldsymbol{w} + \Lambda \partial_x \boldsymbol{w} = 0$$

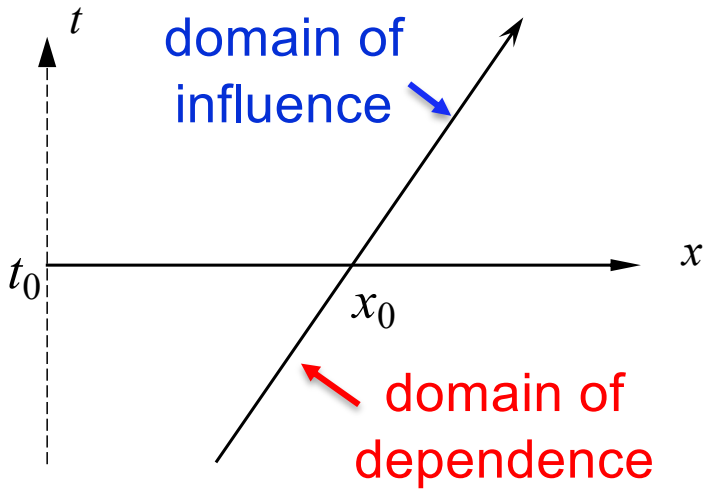
Or $\partial_t w^p + \lambda^p \partial_x w^p = 0$ (p=1,2,…,m)

This is a set of decoupled linear advection equations, with λ^p being wave speeds.

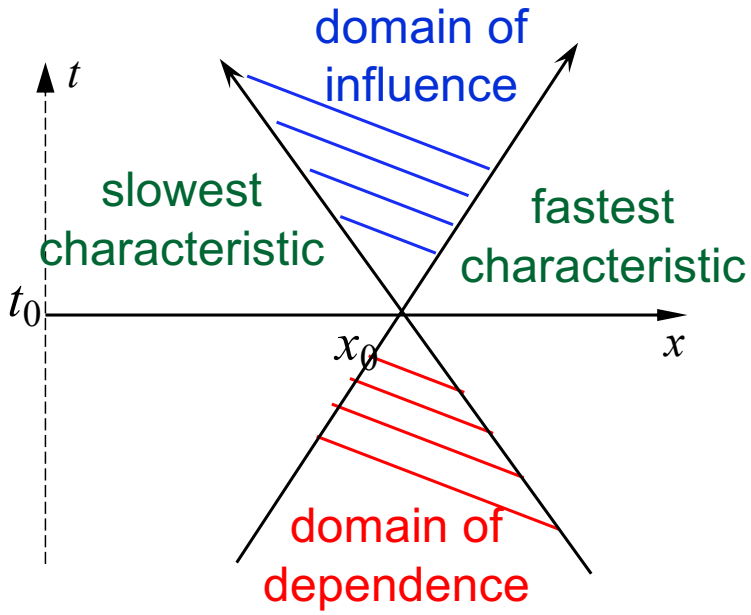
Domain of dependence/influence

The stable timestepping of numerical methods is closely related to the concept of domain of dependence and domain of influence.

For a scalar advection Eq.:



For a system of hyperbolic Eqs.:



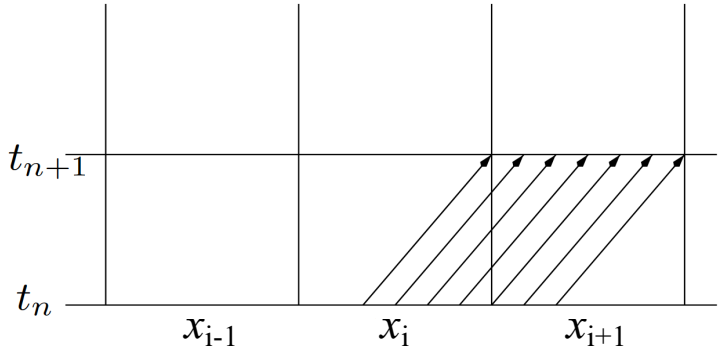
The Courant-Friedrichs-Lewy (CFL) condition

- A numerical method is stable only if its domain of dependence contains the true domain of dependence of the PDE.
- For explicit methods, the timestep Δt must be sufficiently small so that information propagates no more than one grid point per timestep.

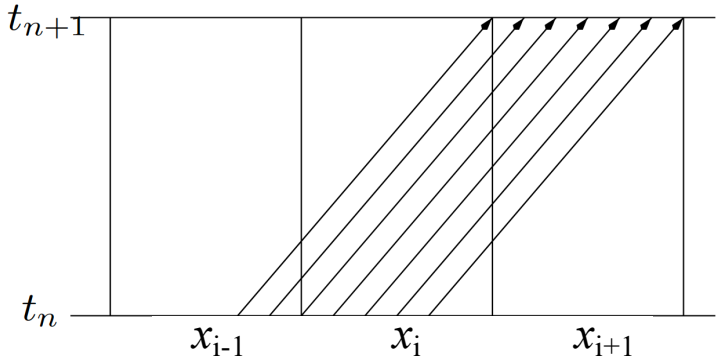
For the linear advection problem $\partial_t u + A\partial_x u = 0$,

$\Delta t \equiv C\dfrac{\Delta x}{A}$, where the **CFL number** $C \leq 1$ for stability in upwind method.

When time step is appropriate (C<1)



When time step is too large (C>1)



Outline

- General concepts
- Initial value problem: hyperbolic equations

  Finite difference method

  Finite volume method
- Initial value problem: parabolic equations
- Boundary value problem: elliptic equations

  Multigrid method

Parabolic equations

Prototype: $\partial_t u = D\partial_{xx}^2 u$

Physics context: random walk, diffusion and mixing, thermal (heat) conduction, viscosity, etc., which describes irreversible dissipation processes.

Starting from a δ function, and D being constant, the solution reads

$$u = \frac{u_0}{\sqrt{4\pi Dt}} e^{-x^2/4Dt} \quad \Rightarrow \quad \langle x^2 \rangle = 2Dt$$

It smears out non-smoothness of the system over a diffusion timescale:

$$t_{\text{diff}} \sim \frac{L^2}{D}$$

L: size of system; scale of interest, etc.

Truncation errors in most numerical algorithms can often be understood as having extra numerical diffusion, which usually plays a stabilizing role.

Solving the diffusion equation

Simple approach: FTCS

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

von Neumann stability analysis:

Setting $u_i^n = a e^{ik(i\Delta x)}$ and $u_i^{n+1} = C_k u_i^n$, we find

$$C_k = 1 + \frac{D\Delta t}{\Delta x^2}[2\cos(k\Delta x) - 2] = 1 - \frac{4D\Delta t}{\Delta x^2}\sin^2\left(\frac{k\Delta x}{2}\right)$$

$$\Rightarrow \quad \Delta t \le \frac{\Delta x^2}{2D} \quad \text{for stability.}$$

Can be generalized to higher order in time like in RK methods.

Solving the diffusion equation

In multi-D, von Neumann stability analysis:

$$C_k = 1 - \frac{4D\Delta t}{\Delta x^2}\sin^2\left(\frac{k_x\Delta x}{2}\right) - \frac{4D\Delta t}{\Delta y^2}\sin^2\left(\frac{k_y\Delta y}{2}\right) - \frac{4D\Delta t}{\Delta z^2}\sin^2\left(\frac{k_z\Delta z}{2}\right)$$

$$\Rightarrow \quad \Delta t \leq \frac{\Delta x_{\min}^2}{2ND} \qquad \text{for stability (N: \# of dimensions).}$$

This is the CFL condition for solving the diffusion equation explicitly:

Timestep can be no longer than the diffusion timescale across one cell.

The fact that $\Delta t \propto \Delta x^2$ places severe constraints on the timestepping.

Implementation example: thermal conduction

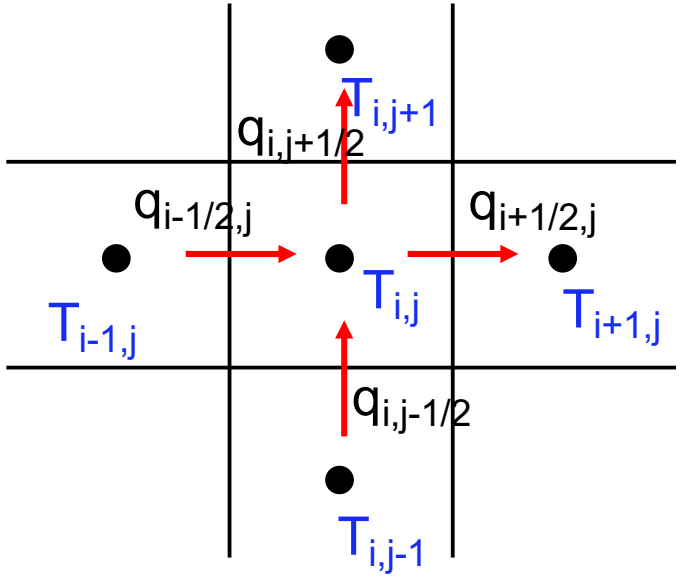
Diffusion equation also reflects conservation laws.

Consider evolution of internal energy under thermal conduction :

$$\frac{\partial \epsilon}{\partial t} = -\nabla \cdot \boldsymbol{q} = \nabla \cdot (\kappa \nabla T) \quad \text{where} \quad \epsilon = \rho c_V T$$

Thermal conduction should be implemented in the framework of energy conservation.

- Define temperature at cell centers.
- Define heat flux at cell interfaces.
- Update the energy by directly differencing the heat flux.



Lifting the timestep constraints

1. Do nothing.

   This is physics, you can not avoid.

2. Super timestepping (Alexiades et al. 1996, Meyer et al. 2012).

   Use pre-determined series of substeps with decreasing lengths, but much larger than the CFL constraints on average, which is shown to be stable.

3. Use implicit method.

   Fully eliminates the timestep constraint, at the cost of inverting a large sparse matrix.

Implicit schemes

In 1D, the implicit scheme gives: $$\frac{u_i^{n+1} - u_i^n}{\Delta t} = D\left(\frac{u_{i+1}^{n+1} - 2u_i^{n+1} + u_{i-1}^{n+1}}{\Delta x^2}\right)$$

This leads to a tridiagonal matrix:

$$\begin{pmatrix} 1+2\delta & -\delta & 0 & \cdots & 0 & 0 \\ -\delta & 1+2\delta & -\delta & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & -\delta & 1+2\delta \end{pmatrix} \begin{pmatrix} u_1^{n+1} \\ u_2^{n+1} \\ \vdots \\ u_n^{n+1} \end{pmatrix} = \begin{pmatrix} u_1^n + \delta u_L \\ u_2^n \\ \vdots \\ u_n^n + \delta u_R \end{pmatrix}$$

boundary values

where $\delta \equiv \frac{D\Delta t}{\Delta x^2}$

This can be solved efficiently on O(N).

However, matrix is no longer so simple in multi-D.

Alternating-Direction Implicit (ADI)

Consider solving: $\frac{\partial u}{\partial t} = \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}$

Directionally split the operators: $\mathcal{L}_x u = u_{i+1,j} - 2u_{i,j} + u_{i-1,j}$

$$\mathcal{L}_y u = u_{i,j+1} - 2u_{i,j} + u_{i,j-1}$$

Solve the equations that are implicit in one direction, while explicit in others:

$$\frac{u_i^{n+1/2} - u_i^n}{\Delta t/2} = -\frac{D}{\Delta x^2}(\mathcal{L}_x u^{n+1/2} + \mathcal{L}_y u^n)$$

$$\frac{u_i^{n+1} - u_i^{n+1/2}}{\Delta t/2} = -\frac{D}{\Delta x^2}(\mathcal{L}_x u^{n+1/2} + \mathcal{L}_y u^{n+1})$$

Each of the above leads to a tridiagonal matrix, and usually, the overall scheme is stable.

Combining different terms

Example: the convection-diffusion equation.

$$\frac{\partial u}{\partial t} + c(x,u)\frac{\partial u}{\partial x} = \frac{\partial}{\partial x}\left[D(x,u)\frac{\partial u}{\partial x}\right] + r(x,u)$$

"convection" diffusion source

Strategy 1: solve everything together where each term is integrated using the aforementioned methods, and embed the procedures into a time integrator as in ODEs (e.g., RK2).

Higher-order integrators usually require explicit schemes.

Generally, timestep is determined by the minimum Δt required by individual terms.

Can achieve better accuracy in time, but it also demands better accuracy in space to match that in time.

Combining different terms

Example: the convection-diffusion equation.

$$\frac{\partial u}{\partial t} + c(x,u)\frac{\partial u}{\partial x} = \frac{\partial}{\partial x}\left[D(x,u)\frac{\partial u}{\partial x}\right] + r(x,u)$$

"convection" diffusion source

Strategy 2, use operator split and solve three separate equations sequentially:

$$\frac{\partial u}{\partial t} + c(x,u)\frac{\partial u}{\partial x} = 0 \qquad (1)$$

$$\frac{\partial u}{\partial t} = \frac{\partial}{\partial x}\left[D(x,u)\frac{\partial u}{\partial x}\right] \qquad (2)$$

$$\frac{\partial u}{\partial t} = r(x,u) \qquad (3)$$

First solve (1) for Δt from u^n to u'.

Then solve (2) for Δt from u' to u''.

Then solve (3) for Δt from u'' to u^{n+1}.

More flexibility in integrating individual terms, e.g., can use implicit schemes, or do sub-cycling / super timestepping in step 2 to alleviate timestep constraints.

The overall scheme is 1st order accurate in time.

Outline

- General concepts
- Initial value problem: hyperbolic equations

  Finite difference method

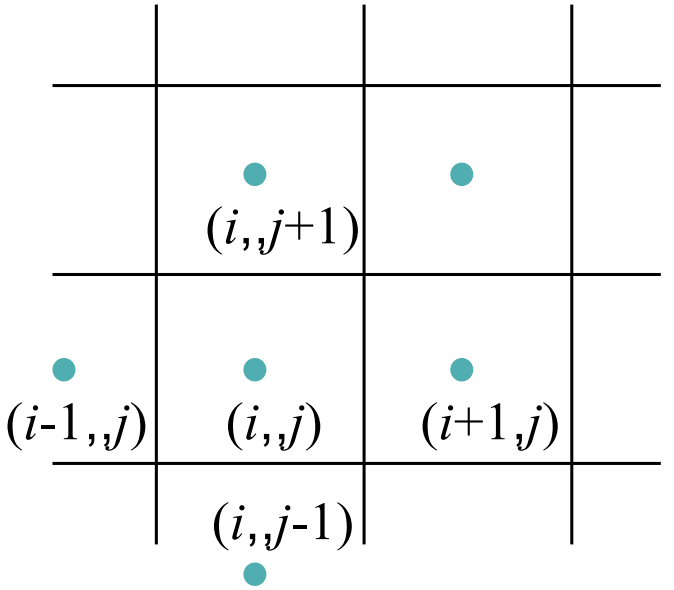
  Finite volume method
- Initial value problem: parabolic equations
- Boundary value problem: elliptic equations

  Multigrid method

Elliptic equations

Prototype: the Poisson equation

$$\nabla^2 \Phi = 4\pi G\rho \ .$$



Solving the Pooisson equation usually leads to a large sparse matrix system

$$\frac{\Phi_{i+1,j} + \Phi_{i-1,j} + \Phi_{i,j+1} + \Phi_{i,j-1} - 4\Phi_{i,j}}{\Delta x^2} = 4\pi G\rho_{i,j}$$

We have seen that this can be solved by applying iterative methods of linear sparse matrix solvers, particularly the CG method.

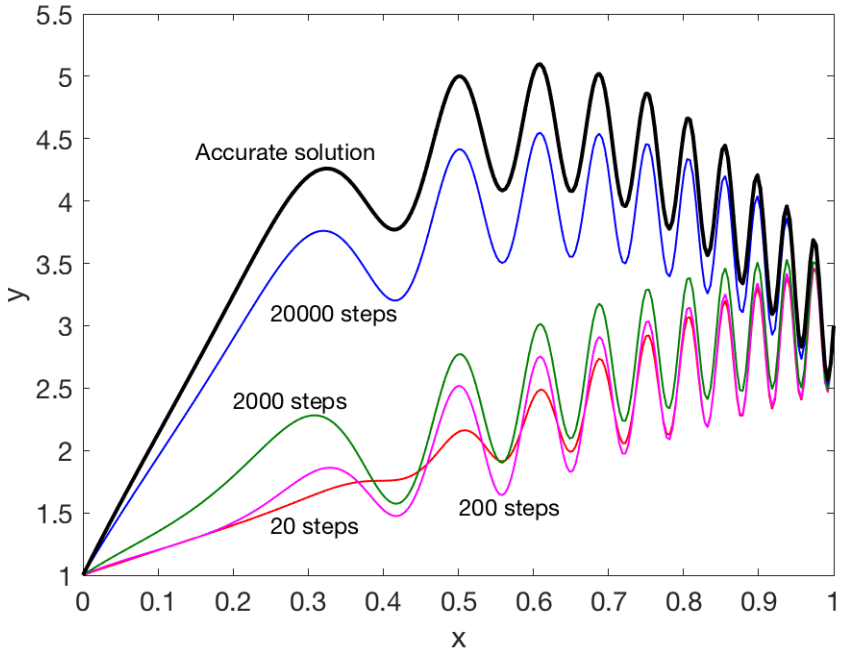
With periodic (or open) BC and uniform grid spacing, can also employ Fourier methods (later this course).

Why are classical iteration methods slow?

Consider an example: $y'' = f(x) \equiv -20 + 60\pi x \cos(20\pi x^3) - 1800\pi^2 x^4 \sin(20\pi x^3)\ ,\ y(0) = 1, y(1) = 3\ .$

Analytic solution: $y(x) = 1 + 12x - 10x^2 + \sin(20\pi x^3)/2$

Initial trial: $y^{(0)}(x) = 1 + 2x$



Solve with Jacobi iteration:

$$y_i^{n+1} = \frac{1}{2}[y_{i+1}^n + y_{i-1}^n - f(x_i)h^2]$$

where h=1/n, n=256.

It converges fast at large k, but extremely slowly at small k.

Multigrid method

More powerful is the multigrid method, which is a highly efficient and very general method for solving boundary value problems.

We aim to solve $\mathcal{L}u = f$ where $\mathcal{L}$ is the differentiation operator.

With grid spacing h, the finite difference form is given by $\mathcal{L}_h u_h = f_h$

Strategy: go through cycles of refinement (doubling resolution) and de-refinement (reduce resolution by half) to enable fast convergence at all scales.

When de-refine the grid, do "restriction": $u_{2h}^{(n)} = \mathcal{R} u_h^{(n)}$

When refine the grid, do "prolongation": $u_h^{(n)} = \mathcal{P} u_{2h}^{(n)}$

Commonly, multigrid is considered as a framework that aims to accelerate other iterative methods towards convergence. But it can also be used on its own.

Multigrid method: typical procedure

Take, say 10, iterations on the original grid (spacing h) to solve $\mathcal{L}_h u_h = f_h$

Coarsen the mesh with half the resolution (double grid spacing to 2h).

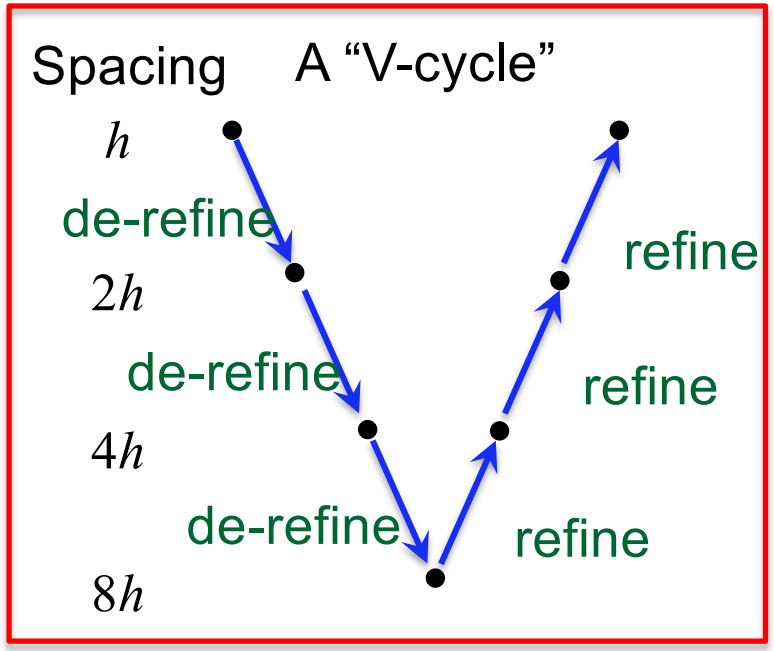
Solve $\mathcal{L}_{2h} u_{2h} = f_{2h}$ on coarser grid for same number of iterations.

Continue the above to coarsest level.

Refine the mesh with double resolution.

Solve the equation at refined grid with the same # of iterations.

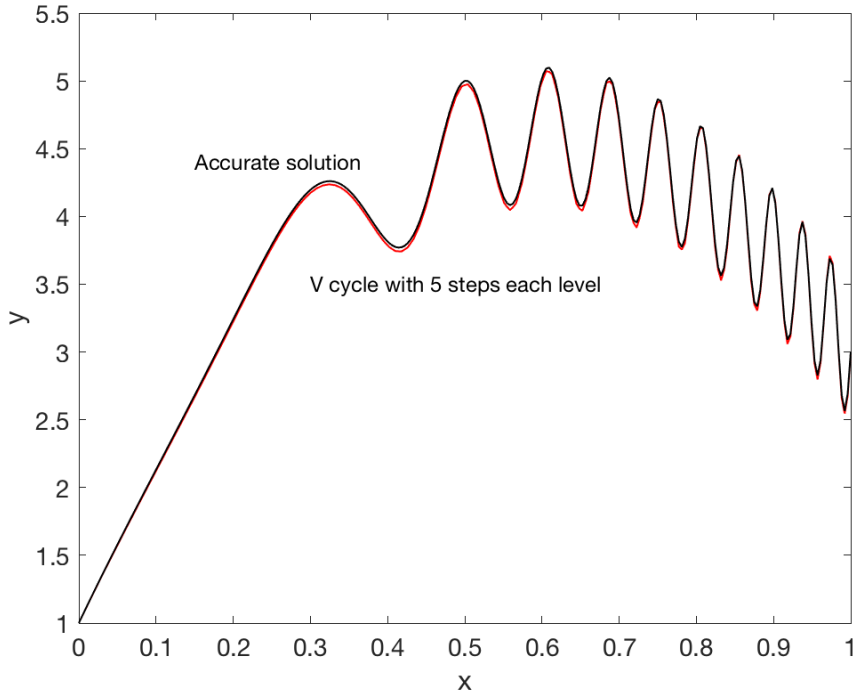
Continue the above to finest (original) level.



This describes a typical "V-cycle". Also common is the "W-cycle".

Multi-grid method: example

Same example as before, but with multigrid:



One V-cycle, total of 15 stages,
5 iterations in each stage.

Computational cost:
~20 iterations in full grid.

There is rich literature and much deeper math. See book “Multigrid” by Trottenberg et al. for more information.

Summary

- Initial value problem: hyperbolic equations
  - von Neumann stability analysis
  - Finite difference: FTCS unstable; need upwinding
  - Finite volume method: preserves conservation laws
  - Integration timestep Δt set by the CFL condition
- Initial value problem: parabolic equations
  - FTCS works but with more stringent constraint on Δt.
  - Can go implicit, particularly ADI, or use super timestepping.
- Boundary value problem: elliptic equations
  - Usually reduce to solving a large sparse linear system.
  - Multigrid method as a powerful framework for fast convergence.