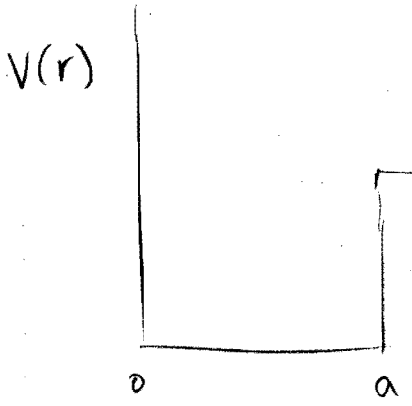
Lecture 14 Differential Equations

(Boundary Value Problems)



$$-\frac{\hbar^2}{2M}\nabla^2\psi(r) + V(r)\psi(r) = E\psi(r)$$

$\psi(0)$ $\psi(a)$

Standard form:
let $Y = \frac{d\psi}{dx}$, then

$$\begin{cases} \frac{dY}{dx} = -\frac{2m}{\hbar^2}(E - V(r))\psi \\ \frac{d\psi}{dx} = Y \end{cases}$$

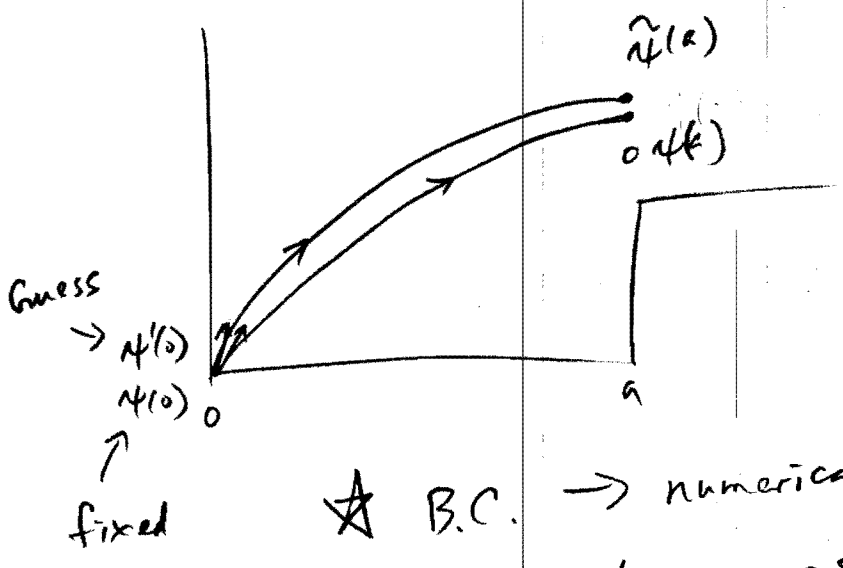
b.c. $\psi(0)$, $\psi(a)$ specified

☆ Not like I.V. problem, constraints are at $\psi(0)$, $\psi(a)$ and Not at one space pt $(\psi(0), \psi'(0))$

$\Rightarrow$ We can't propagate solution fm $r=0$ to other r !

Shooting Methods

- make a guess on $\psi'(0)$ $(Y(0))$ and use methods for I.V. to propagate to $\tilde{\psi}(a)$.
- Check $\tilde{\psi}(a)$ with $\psi(a)$ and refine initial guess.



☆ B.C. → numerically more intensive
- need to propogate soln to $\tilde{\psi}(a)$ for diff. choice of $\psi'(0)$!

Define the error by:

$$R(\beta) = \tilde{\psi}(a,\beta) - \psi(a) \qquad \beta = \psi'(0) \text{ (our initial Guess)}$$

☆ Then, shooting method translate to finding the zero of the function $R(\beta)$!

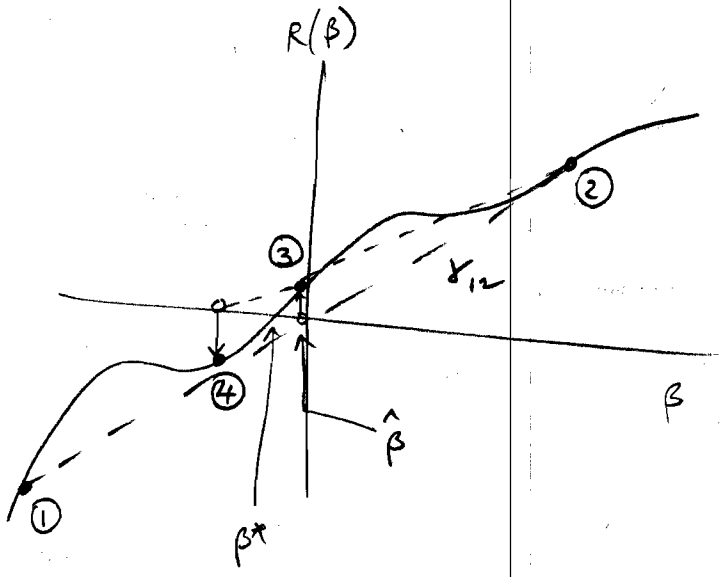
- If $\frac{dR}{d\beta}$ is known then Newton's $\beta_{n+1} = \beta_n - \frac{R(\beta_n)}{R'(\beta_n)}$ method can be used!

- If $\frac{dR}{d\beta}$ is not known (as in most cases), secant method can be used!

$$\beta_{n+1} = \beta_n - \frac{R(\beta_n)}{S} \quad ; \quad S = \frac{R(\beta_n) - R(\beta_{n-1})}{\beta_n - \beta_{n-1}}$$

Secant Method :

☆ assume $R(\beta)$ is smooth near β^* ($R(\beta^*) = 0$)



– Approximate β^* by $\emptyset$-crossing of γ_{12}

$$\gamma_{12}(\beta) = \left(\frac{R_2 - R_1}{\beta_2 - \beta_1}\right)\beta + \frac{(R_1\beta_2 - R_2\beta_1)}{\beta_2 - \beta_1} = 0$$

slope

↑ chosen so that $\gamma_{12}(\beta_1) = R_1$, $\gamma_{12}(\beta_2) = R_2$

$$\hat{\beta} = \frac{R_2\beta_1 - R_1\beta_2 \,(+R_2\beta_2 - R_2\beta_2)}{R_2 - R_1}$$

$$\hat{\beta} = \beta_2 - R_2\frac{(\beta_2 - \beta_1)}{R_2 - R_1}$$

This can be put into an iterative form:

$\beta_2 = \beta_i$

$\beta_1 = \beta_{i-1}$

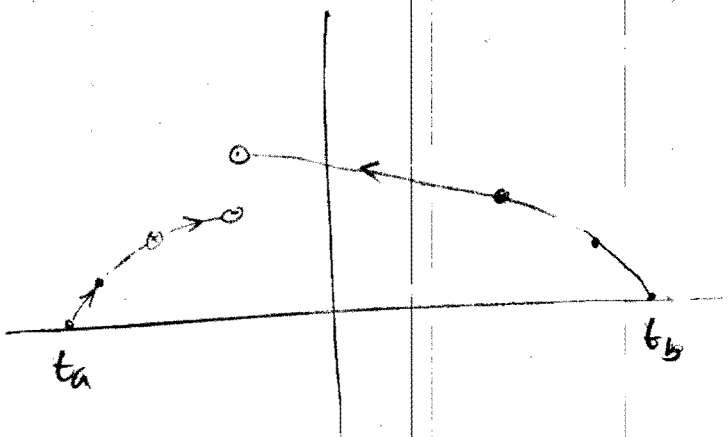
$\hat{\beta} = \beta_{i+1}$

Then, $$\beta_{i+1} = \beta_i - R(\beta_i)\frac{(\beta_i - \beta_{i-1})}{R(\beta_i) - R(\beta_{i-1})}$$

If $R(\beta)$ is smooth & close enough to β^*, then

$\beta_{i+1} \to \beta^*$!

Bi-directional Shooting



$\underset{\sim}{\dot{x}} = F(\underset{\sim}{x}, t)$ n dimensional

☆ need n b.c/I.C. to determine $\underset{\sim}{x}(t)$ uniquely!

Note:
- If all $\underset{\sim}{x}(t_a)$ are given → I.V. problem
- If $n/2$ $\underset{\sim}{x}(t_a)$ and $n/2$ $\underset{\sim}{x}(t_b)$ → b.c. problem
- In general, n_1 - $\underset{\sim}{x}(t_a)$ and n_2 - $\underset{\sim}{x}(t_b)$ with $n_1 + n_2 = n$.

Advantages of subdividing $[t_a, t_b]$:

- interval too large for us to propogate the solution all the way with good numerical accuracy!
- different region might require different integrator properties: e.g. step sizes
- one or both of the end pts are singular.

☆ – it is possible to use special techniques (eg. analytic continuation) to integrate slightly away from t_a then propogate

But, without knowing <u>in advance</u> where $\underset{\sim}{X}(t)$ will go, it is usually hard to propogate into a singular pt.

Bidirectional Shooting:

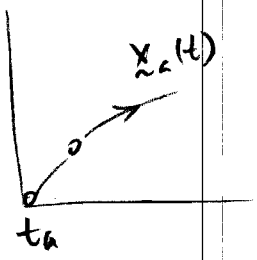
$\underset{\sim}{V}_a \rightarrow n_2$-vector, n_2 free parameters

$\underset{\sim}{V}_b \rightarrow n_1$-vector, n_1 free parameters

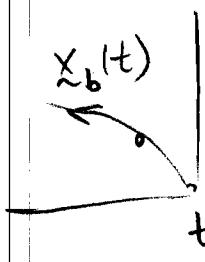
$\underset{\sim}{Y}_a \rightarrow n_1$-vector, n_1 b.c. at t_a

$\underset{\sim}{Y}_b \rightarrow n_2$-vector, n_2 b.c. at t_b.

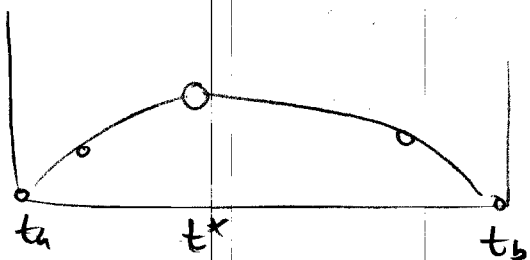
$\underset{\sim}{V}_a + \underset{\sim}{Y}_a \rightarrow n_1 + n_2 = n$ conditions to start $\dot{\underset{\sim}{x}} = F(\underset{\sim}{x}, t)$ from t_a



$\underset{\sim}{V}_b + \underset{\sim}{Y}_b \rightarrow n_2 + n_1 = n$ conditions to start $\dot{\underset{\sim}{x}} = F(\underset{\sim}{x}, t)$ from t_b



Adjust $\underset{\sim}{V}_a$, $\underset{\sim}{V}_b$ to match $\underset{\sim}{x}_a(t^*) = \underset{\sim}{x}_b(t^*)$ at t^*.



Similar to before, we have error fn.

$$R(\underset{\sim}{V}_a, \underset{\sim}{V}_b) = \underset{\sim}{x}_a(t^*) - \underset{\sim}{x}_b(t^*)$$

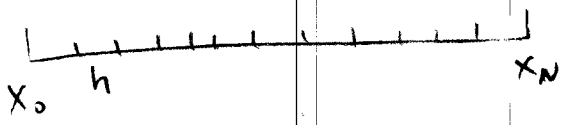
— Use Newton or Secante to find $\underset{\sim}{V}_a, \underset{\sim}{V}_b$!

Relaxation Methods

$$\frac{d^2\phi}{dx^2} - g(x)\phi(x) = 0 \qquad (*)$$

$$\phi(x_0) = \alpha_0 \qquad \phi(x_N) = \alpha_N$$

- Discretize $[x_0, x_N]$ into N steps with step size h



Convert diveratives to finite differences:

$$\left.\frac{d^2\phi}{dx^2}\right|_{x_k} = \frac{1}{h^2}\left(\phi_{k+1} - 2\phi_k + \phi_{k-1}\right) \quad \text{for constant step size.}$$

$(*) \Rightarrow \quad \phi_{k+1} - 2\phi_k + \phi_{k-1} - h^2 g_k \phi_k = 0$

there are $(N-1)$ of these equations with $k = 1, \ldots, N-1$

B.C. gives $\begin{cases} \phi_0 = \alpha_0 \\ \phi_N = \alpha_N \end{cases}$

* Note: there are N intervals $(N+1)$ data pts

Combining with above, we have a matrix equation:

$$\underset{\sim}{A}\cdot\underset{\sim}{Y} = \underset{\sim}{b} \qquad (\#)$$

$$\begin{cases} \phi_0 = \alpha_0 \\ -\phi_{k-1} + (2+h^2 g_k)\phi_k - \phi_{k+1} = 0 \qquad k = 1, \dots, N-1 \\ \phi_N = \alpha_N \end{cases}$$

$$\underset{\sim}{A} = \begin{pmatrix} 1 & 0 & 0 & 0 & \cdots & 0 \\ -1 & \eta_1 & -1 & 0 & \cdots & 0 \\ 0 & -1 & \eta_2 & -1 & \cdots & 0 \\ & \vdots & & & \vdots & \\ 0 & \cdots & & \cdots & -1 & \eta_{N-1} & -1 \\ 0 & \cdots & & \cdots & 0 & 1 \end{pmatrix} \qquad \eta_k = 2+h^2 g_k$$

$$\underset{\sim}{Y} = \begin{pmatrix} \phi_0 \\ \vdots \\ \phi_N \end{pmatrix} \qquad \underset{\sim}{b} = \begin{pmatrix} \alpha_0 \\ 0 \\ \vdots \\ 0 \\ \alpha_N \end{pmatrix}$$

Note: N might be a large number!

$\underset{\sim}{A} \to (N+1)\times(N+1)$ might be huge

But, $\underset{\sim}{A}$ is sparse! ☆ Relaxation methods take advantage of this fact to solve for $\underset{\sim}{Y}$!

★ In Press, they describe two other methods to solve ⊛ in other special cases.

★ Using finite differences to transform Differential Eqs to matrix eqns can be done for PDEs also! ⇒ Finite differences in space and time.

Example:

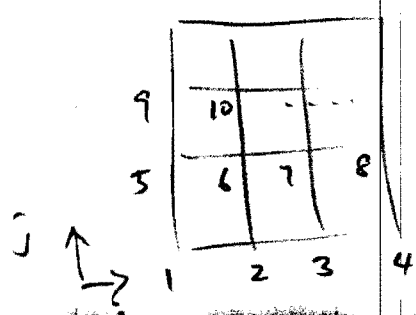
Poisson eqn

$$\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = p(x,y)$$

$$\Rightarrow \frac{u_{j+1,l} - 2u_{j,l} + u_{j-1,l}}{\Delta^2} + \frac{u_{j,l+1} - 2u_{j,l} + u_{j,l-1}}{\Delta^2} = p_{j,l}$$

$$\Rightarrow u_{j+1,l} + u_{j-1,l} + u_{j,l+1} + u_{j,l-1} - 4u_{j,l} = \Delta^2 p_{j,l}$$

- putting together boundary specifications of u and relabelling (j,l) to $n \in [1, \underset{M}{N\times N}]$



★ we again get $\underset{\sim}{A} \cdot \underset{\sim}{Y} = \underset{\sim}{b}$

$\underset{\sim}{A}$: $M \times M$ large but sparse matrix.

① "rapid" (Fourier) Methods

- special class — ⓐ Constant coefficients or more generally separable in chosen coordinates
  ⓑ boundaries coincide with coordinate lines (rect).

② Direct Method - depends strongly on the structure of the sparse matrix.
(basically solving the linear Eq directly!)

- If ⓐ and ⓑ are met, then we can Fourier transform our PDE to a algebraic equation!
- solve the algebraic equation
- Inverse FFT back to desired soln in real space

Relaxation Methods

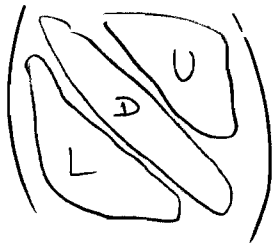
- an iterative approach to solve a system containing a large # of linear algebraic equations.

$$\underset{\sim}{A} \cdot \underset{\sim}{x} = \underset{\sim}{b}$$

For any given matrix $\underset{\sim}{A}$, we can split it into

$$\underset{\sim}{A} = \underset{\sim}{L} + \underset{\sim}{D} + \underset{\sim}{U}$$

Lower triangular — diagonal — upper triangular



Then, $\underset{\sim}{A} \cdot \underset{\sim}{x} = \underset{\sim}{b}$ becomes,

$$\underset{\sim}{D}\,\underset{\sim}{x} + (\underset{\sim}{L} + \underset{\sim}{U}) \cdot \underset{\sim}{x} = \underset{\sim}{b}$$

★ A convergent procedure to solve this when $\underset{\sim}{A}$ is "strictly diagonally dominant" is to do the following iteration:

$$\underset{\sim}{D} \cdot \underset{\sim}{x}^{n} = -(\underset{\sim}{L} + \underset{\sim}{U}) \cdot \underset{\sim}{x}^{n-1} + \underset{\sim}{b}$$

starting $\underset{\sim}{x}^{0}$ with some initial guess.

This is called the Jacobi method!

The convergence characteristic of this procedure is given by:

$$\underset{\sim}{X}^{n} = -\underset{\sim}{D}^{-1} \cdot (\underset{\sim}{L} + \underset{\sim}{U}) \cdot \underset{\sim}{x}^{n-1} + \underset{\sim}{D}^{-1} \cdot \underset{\sim}{b}$$

... the eigenvalues of $\underset{\sim}{D}^{-1} \cdot (\underset{\sim}{L} + \underset{\sim}{U})$!

✡ In the case when $\underset{\sim}{A}$ is "strictly diagonally dominated", i.e.

$\geq \rightarrow$ just diagonally dominant

$$|a_{ii}| > \sum_{i \neq j} |a_{ij}|$$

(For most finite difference methods, this usually is the case! $\rightarrow \frac{d^2\phi}{dx^2} \sim \phi_{k+1} - 2\phi_k + \phi_{k-1}$),

then it can be shown that

$$\left| \lambda \left(\underset{\sim}{D}^{-1} \cdot (\underset{\sim}{L} + \underset{\sim}{U}) \right) \right| < 1$$

$\Rightarrow$ the iterative procedure is convergent!

[e.g. $\begin{pmatrix} a & \epsilon \\ \epsilon & b \end{pmatrix} \rightarrow \underset{\sim}{D}^{-1} \cdot (\underset{\sim}{L} + \underset{\sim}{U}) = \begin{pmatrix} 0 & \epsilon/a \\ \epsilon/b & 0 \end{pmatrix}$

$\begin{pmatrix} \frac{1}{a} & 0 \\ 0 & \frac{1}{b} \end{pmatrix} \begin{pmatrix} 0 & \epsilon \\ \epsilon & 0 \end{pmatrix}$

$\lambda = \pm \sqrt{\frac{\epsilon^2}{ab}}$

$|\lambda| < 1$

Proof of $\lambda(\underline{\underline{D}}^{-1}\cdot(\underline{\underline{L}}+\underline{\underline{U}})) < 1$:

- let $\underline{\underline{R}} = \underline{\underline{L}}+\underline{\underline{U}}$ be the nondiagonal part of $\underline{\underline{A}}$
- let λ be the eigenvalue of $\underline{\underline{D}}^{-1}\underline{\underline{R}}$ with eigenvector $\underline{v}$.
- choose $\underline{v}$ so that $\|\underline{v}\|_\infty = 1$, so that

$$v_m = 1 \quad \text{for some } 1 \le m \le n.$$

(Note: this can be done by dividing the $\underline{v}$ by its largest component. And, $\alpha\underline{v}$ is also an eigenvector of $\underline{\underline{D}}^{-1}\underline{\underline{R}}$ with the same eigenvalue λ.)

- Then, we have

$$\underline{\underline{D}}^{-1}\cdot\underline{\underline{R}}\cdot\underline{v} = \lambda\underline{v}$$

$$\underline{\underline{R}}\cdot\underline{v} = \lambda\underline{\underline{D}}\cdot\underline{v}$$

- Since $r_{mm} = 0$ ($\underline{\underline{R}}$ contains only non-diagonal terms), taking the absolute value of the mth component of this vector equation, we have

$$|r_{m,1}v_1 + r_{m,2}v_2 + \dots + r_{m,m-1}v_{m-1} + r_{m,m+1}v_{m+1} + \dots r_{mn}v_n|$$
$$= |\lambda d_{mm} v_m|$$
$$= |\lambda||d_{mm}| \quad (v_m = 1)$$

– Since $\|\underset{\sim}{v}\|_\infty = 1$, $|v_i| \leq 1$,

$$LHS \leq \sum_{j \neq m} |r_{mj}| < |d_{mm}|$$ strictly diagonally dominate

$$\Rightarrow \quad |\lambda||d_{mm}| < |d_{mm}|$$

$$\Rightarrow \quad |\lambda| < 1 \quad !$$

spectral radius ρ_s :

the eigenvalue with the largest modulus

- gives the rate of slowest decaying eigenmode.

In general, $\rho_s - 1 \sim \frac{1}{N^d}$ d – # of independent variables or dimension of grid.

For the specific Poisson example with Dirichlet b.c., (in 2D)

$$\rho_s \simeq 1 - \frac{\pi^2}{2N^2} \quad (*)$$

(Jacobi)

Question: What is the number of iterations required to reduce overall error by a factor of 10^{-p}?

$$\frac{\varepsilon'}{\varepsilon} = 10^{-p} \Rightarrow \rho_s^r = 10^{-p}$$

$$\Rightarrow r \simeq \frac{p \ln 10}{-\ln \rho_s}$$

($\ast$ $\ln x \simeq 1 - x$)

with $(*)$, $r \simeq \frac{2N^2}{\pi^2} p \ln 10 \simeq \frac{1}{2} p N^2$ ← too slow for practical use!

The Gauss-Seidel Method :

$$(\underset{\sim}{L} + \underset{\sim}{D})\,\underset{\sim}{x}^{n+1} = -\underset{\sim}{U}\,\underset{\sim}{x}^{n} + \underset{\sim}{b}$$

Jacobi $D x^{n+1} = -(L+U)x^n + b$

For N = 3, this looks like,

$$\begin{cases} x_1^{n+1} = \frac{1}{A_{11}}\left(b_1 - A_{12}x_2^n - A_{13}x_3^n\right) \\ x_2^{n+1} = \frac{1}{A_{22}}\left(b_2 - A_{21}x_1^{n+1} - A_{23}x_3^n\right) \\ x_3^{n+1} = \frac{1}{A_{33}}\left(b_2 - A_{31}x_1^{n+1} - A_{32}x_2^{n+1}\right) \end{cases}$$

$\underset{\sim}{D}\,\underset{\sim}{x}^{n+1} = -\underset{\sim}{L}\,\underset{\sim}{x}^{n+1} - \underset{\sim}{U}\,\underset{\sim}{x}^{n} + \underset{\sim}{b}$

Compare with Jacobi,

$$\begin{cases} x_1^{n+1} = \frac{1}{A_{11}}\left(b_1 - A_{12}x_2^n - A_{13}x_3^n\right) \\ x_2^{n+1} = \frac{1}{A_{22}}\left(b_2 - A_{21}x_1^n - A_{23}x_3^n\right) \\ x_3^{n+1} = \frac{1}{A_{33}}\left(b_3 - A_{31}x_1^n - A_{32}x_2^n\right) \end{cases}$$

★★ updating $\underset{\sim}{x}^n$ as soon as we set new components!

bottom line: GS is faster by a factor of two

but still not fast enough!

Successive Overrelaxation SOR

(Standard algorithm until 1970s)

↑

multigrid methods introduced by Brandt.

- We can increase the convergence rate (or reduce ρ_s) by making an overcorrection to $\underset{\sim}{x}^{n+1}$ at the nth stage of GS iteration

Recall in GS

$$\underset{\sim}{x}^{n+1} = (\underset{\sim}{L}+\underset{\sim}{D})^{-1}\cdot\left[-\underset{\sim}{U}\cdot\underset{\sim}{x}^{n} + \underset{\sim}{b}\right]$$

add & subtract $\underset{\sim}{x}^{n}$

$$= \underset{\sim}{x}^{n} - (\underset{\sim}{L}+\underset{\sim}{D})^{-1}\cdot\left[(\underset{\sim}{L}+\underset{\sim}{D}+\underset{\sim}{U})\,\underset{\sim}{x}^{n} - \underset{\sim}{b}\right]$$

↓

residual vector

$$\underset{\sim}{\xi}^{n} = \underset{\sim}{A}\,\underset{\sim}{x}^{n} - \underset{\sim}{b}$$

$$\Rightarrow \quad \underset{\sim}{x}^{n+1} = \underset{\sim}{x}^{n} - (\underset{\sim}{L}+\underset{\sim}{D})^{-1}\cdot\underset{\sim}{\xi}^{n}$$

think of this: x_{n+1} is corrected from x_n by subtracting the error term!

Now, we overcorrect:

$$\underset{\sim}{x}^{n+1} = \underset{\sim}{x}^{n} - \omega(L+D)^{-1}\cdot\xi^{n}$$

, where w is the overrelaxation parameters and this method is called successive overrelaxation (SOR).

Proved Theorems:

① SOR is convergent only for $0 < w < 2$.
If $0 < w < 1$, we speak of underrelaxation.
If $w = 1$, we get GS back
If, $1 < w < 2$, overrelaxation

② Only $1 < w < 2$, SOR is faster the GS.

③ If ρ_{Jacobi} is the spectra radius of Jacobi, the optimal choice for w is

$$w = \frac{2}{1 + \sqrt{1 - \rho_J^2}}$$

then $$\rho_{SOR} = \left(\frac{\rho_J}{1 + \sqrt{1 - \rho_J^2}} \right)^2$$

★ If $\rho_J \simeq 1 - \frac{\pi^2}{2N^2}$ $\longrightarrow$ $\rho_J^2 \simeq \left(1 - \frac{\pi^2}{2N^2}\right)^2$

$\rho_J^2 \simeq 1 - 2\frac{\pi^2}{2N^2}$

$\left(\frac{\pi}{N} \ll 1\right)$

then $w \simeq \frac{2}{1+\pi/N}$ $\longleftarrow$ $\sqrt{1-\rho_J^2} = \frac{\pi}{N}$

$\rho_{SOR} \simeq 1 - \frac{2\pi}{N}$ for N large

(see note ★)

$\Rightarrow r = \frac{R \ln 10}{-\ln \rho_{SOR}} \simeq \frac{R N \ln 10}{2\pi} \simeq \frac{1}{3} R N$ for SOR !

Thus, SOR is geometrically faster than Jacobi or GS.

Chebyshev acceleration

w_{opt} → over correction might overshoot at the beginning !

($w = 1$ might actually be better at the initial iterations)

★ Modification to deal with this :

$$\begin{cases} w^{(0)} = 1 \\ w^{(1)} = 1/(1 - \rho_J^2/2) \end{cases}$$

$w^n = 1/(1 - \rho_J^2 w^{n-1}/4)$ for $n > 1$

$w^\infty \to w_{opt}$

$$\rho_{SOR} = \left(\frac{\rho_J}{1+\sqrt{1-\rho_J^2}} \right)^2$$

$$\simeq \left(\frac{1 - \frac{\pi^2}{2N^2}}{1 + \frac{\pi}{N}} \right)^2$$

$$= \left(1 - \frac{\pi^2}{2N^2}\right)^2 \left(1 + \frac{\pi}{N}\right)^{-2}$$

$$\simeq \left(1 - \frac{\pi^2}{N^2}\right)\left(1 - \frac{2\pi}{N}\right)$$

$$\simeq 1 - \frac{2\pi}{N} \quad \left(\text{to order } \frac{1}{N}\right)$$

Introduction to Partial Differential Equations (PDEs)

Def: A differential equation involving more than one independent variable.

Theoretically, PDE can be classified into three classes: (General form - see 17')

① Hyperbolic ($B^2 - 4AC > 0$)

important physics example:

$$\frac{\partial^2}{\partial t^2} u(x,t) = c^2 \frac{\partial^2}{\partial x^2} u(x,t)$$

(wave equation)

with b.c. $u(0,t)$; $u(L,t)$

and I.c. $u(x,0)$; $\frac{\partial u}{\partial t}(x,0)$

☆ Finite Differences approximations to $\frac{\partial^2}{\partial t^2} u$ & $\frac{\partial^2}{\partial x^2} u$ are basis in solving PDEs numerically.

ex: central differences

$$\frac{\partial^2}{\partial t^2} u(x,t) \approx \frac{u_j^{n+1} - 2u_j^n + u_j^{n-1}}{\Delta t^2}$$, and

PDE's General Form

(2 independent variables = x, y)

$$A u_{xx} + B u_{xy} + C u_{yy} + F(u_x, u_y, u, x, y) = 0$$

$$\frac{\partial^2}{\partial x^2} u(x,t) \cong \frac{u_{j+1}^n - 2u_j^n + u_{j-1}^n}{\Delta x^2}$$

Here, $n \rightarrow$ time index, Δt – time discretization step

$j \rightarrow$ space index, Δx – space discretization step

So, the PDE becomes,

$$\frac{u_j^{n+1} - 2u_j^n + u_j^{n-1}}{\Delta t^2} = c^2 \frac{u_{j+1}^n - 2u_j^n + u_{j-1}^n}{\Delta x^2}$$

— Solving this equation for u_j^{n+1}, we arrive at the following iteration equation,

$$u_j^{n+1} = (2 - 2r^2) u_j^n + r^2 (u_{j+1}^n + u_{j-1}^n) - u_j^{n-1} \quad (*)$$

$r \equiv \frac{c\Delta t}{\Delta x}$

↑ next timestep

↑ previous time steps

So, if we start with $u(x,0)$ & $\frac{\partial}{\partial t} u(x,0)$ (initial conditions), the discretization of I.C. $\rightarrow u_j^1, u_j^0$ for all j in the first two time steps

$\Rightarrow$ subsequent time steps of u_j^n can be constructed from $(*)$.

✡✡ However, as we will discuss later, explicit scheme as in ⊛ has stability concerns !!

(For ⊛, $r = \frac{c\Delta t}{\Delta x} \leq 1$ is necessary for stability.)
↳ "conditional stability"

✡ As in regular ODE, there are implicit schems which are stable for all r ("unconditional stability").

---

② Parabolic $(B^2 - 4AC = 0)$

important physics example : $\frac{\partial}{\partial t} u(x,t) = c^2 \frac{\partial^2}{\partial x^2} u(x,t)$

(Heat Equation)

✡ As in the previous case, depending on the finite difference scheme used, the numerical method to solve this can be explicit or implicit and one need to be concerned with the stability.

example : Forward time Central space

$$\frac{\partial}{\partial t} u(x,t) \cong \frac{u_j^{n+1} - u_j^n}{\Delta t}$$

$$\frac{\partial^2}{\partial x^2} u(x,t) \cong \frac{u_{j+1}^n - 2u_j^n + u_{j-1}^n}{\Delta x^2}$$

Next: Solving tridiagonal simultaneous equations **Up:** FINITE DIFFERENCING IN (omega,x)-SPACE
Previous: The leapfrog method

The Crank-Nicolson method

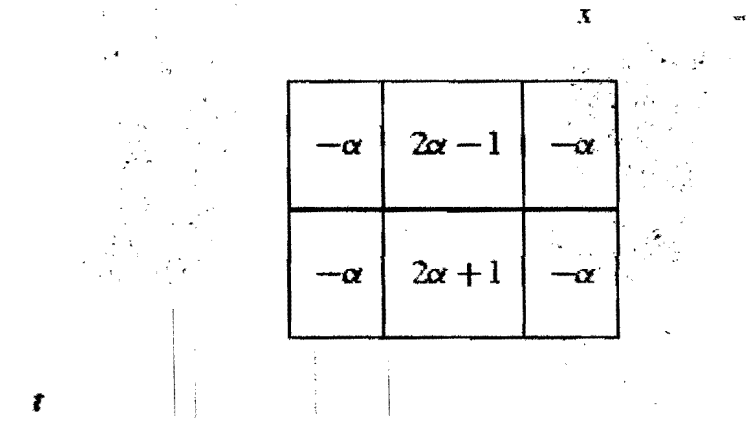
The **Crank-Nicolson method** solves both the accuracy and the stability problem. Recall the difference representation of the **heat-flow equation** (27).

$$q_{t+1}^{x} - q_{t}^{x} \quad = \quad a \left(q_{t}^{x+1} - 2q_{t}^{x} + q_{t}^{x-1}\right) \tag{29}$$

Now, instead of expressing the right-hand side entirely at time t, it will be averaged at t and t+1, giving

$$q_{t+1}^{x} - q_{t}^{x} \quad = \quad \frac{a}{2}\left[\left(q_{t}^{x+1} - 2q_{t}^{x} + q_{t}^{x-1}\right) + \left(q_{t+1}^{x+1} - 2q_{t+1}^{x} + q_{t+1}^{x-1}\right)\right] \tag{30}$$

This is called the **Crank-Nicolson method**. Defining a new parameter $\alpha = a/2$, the difference star is



$-\alpha$	$2\alpha - 1$	$-\alpha$
$-\alpha$	$2\alpha + 1$	$-\alpha$

(31)

When placing this star over the data table, note that, typically, three elements at a time cover unknowns. To say the same thing with equations, move all the t+1 terms in (30) to the left and the t terms to the right, obtaining

$$-\alpha q_{t+1}^{x+1} + (1+2\alpha) q_{t+1}^{x} - \alpha q_{t+1}^{x-1} \quad = \quad \alpha q_{t}^{x+1} + (1-2\alpha) q_{t}^{x} + \alpha q_{t}^{x-1} \tag{32}$$

Now think of the left side of equation (32) as containing all the ***unknown*** quantities and the right side as containing all ***known*** quantities. Everything on the right can be combined into a single known quantity, say, d_t^x. Now we can rewrite equation (32) as a set of simultaneous equations. For definiteness, take the x-axis to be limited to five points. Then these equations are:

$$\begin{bmatrix} c_{\text{left}} & -\alpha & 0 & 0 & 0 \\ -\alpha & 1+2\alpha & -\alpha & 0 & 0 \\ 0 & -\alpha & 1+2\alpha & -\alpha & 0 \\ 0 & 0 & -\alpha & 1+2\alpha & -\alpha \\ 0 & 0 & 0 & -\alpha & c_{\text{right}} \end{bmatrix} \begin{bmatrix} q_{t+1}^{1} \\ q_{t+1}^{2} \\ q_{t+1}^{3} \\ q_{t+1}^{4} \\ q_{t+1}^{5} \end{bmatrix} = \begin{bmatrix} d_t^1 \\ d_t^2 \\ d_t^3 \\ d_t^4 \\ d_t^5 \end{bmatrix} \tag{33}$$

Equation (32) does not give us each q_{t+1}^{x} ***explicitly***, but equation (33) gives them ***implicitly*** by the solution of simultaneous equations.

The values c_{left} and c_{right} are adjustable and have to do with the side **boundary conditions**. The important thing to notice is that the matrix is **tridiagonal**, that is, except for three central diagonals all the elements of the matrix in (33) are zero. The solution to such a set of simultaneous equations may be economically obtained. It turns out that the cost is only about twice that of the **explicit method** given by (27). In fact, this **implicit method** turns out to be cheaper, since the increased accuracy of (32) over (27) allows the use of a much larger numerical choice of Δt.A program that demonstrates the stability of the method, even for large Δt, is given next.

A **tridiagonal** simultaneous equation solving subroutine `rtris()` explained in the next section. The *results are stable, as you can see.*

---

next up prev print clean

Next: Solving tridiagonal simultaneous equations **Up:** FINITE DIFFERENCING IN (omega,x)-SPACE
Previous: The leapfrog method
Stanford Exploration Project
12/26/2000

$\longrightarrow$ ...

$$u_j^{n+1} = (1-2r)u_j^n + r(u_{j-1}^n + u_{j+1}^n)$$

explicit scheme

$$r = \frac{c^2 \Delta t}{\Delta x^2}$$

again, stable only if $\underline{0 \le r \le \frac{1}{2}}$.

— Here is a stable implicit scheme for the heat equation:

Crank-Nicolson Method

- approximate $\frac{\partial u}{\partial t}(x, t+\frac{\Delta t}{2})$ by central difference:

$$\left(\frac{\partial}{\partial t} u(x, t+\tfrac{\Delta t}{2}) = \frac{u_j^{n+1} - u_j^n}{\Delta t} \right)$$

u_j^{n+1} ∘ $t+\Delta t$

∘ $t+\frac{\Delta t}{2}$

u_j^n ∘ t

- approximate $\frac{\partial^2 u}{\partial x^2}(x, t+\frac{\Delta t}{2})$ by the average of $\frac{\partial^2}{\partial x^2} u(x, t+\Delta t)$ & $\frac{\partial^2}{\partial x^2} u(x,t)$:

$$\left(\frac{\partial^2}{\partial x^2} u(x, t+\tfrac{\Delta t}{2}) = \frac{(u_{j+1}^{n+1} - 2u_j^{n+1} + u_{j-1}^{n+1}) + (u_{j+1}^n - 2u_j^n + u_{j-1}^n)}{2\Delta x^2} \right)$$

↑ the implicit part

$\longrightarrow$ ...

implicit scheme

$$\left[-r u_{j-1}^{n+1} + (2+2r) u_j^{n+1} - r u_{j+1}^{n+1} = (2-2r) u_j^n + r(u_{j-1}^n + u_{j+1}^n) \right]$$

$$r = \frac{c^2 \Delta t}{\Delta x^2}$$

↑ a system of coupled equations with $(u_1^{n+1}, \ldots, u_N^{n+1})^T$ to be solved from matrix eq.

(ch.10, "Numerical Methods using Matlab," Mathews)

③ Elliptic ($B^2 - 4AC < 0$):

important physics examples $= \nabla^2 u = \frac{\partial^2}{\partial x^2} u(x,y) + \frac{\partial^2}{\partial y^2} u(x,y)$

$$= \begin{cases} 0 & \text{Laplace's} \\ \rho(x,y) & \text{Poisson's} \\ (V(x,y) - E)u(x,y) & \text{Helmholtz's} \end{cases}$$

– there are various finite difference schemes –

★★ One important difference between this class and the previous two is that

③ → purely a boundary value problem

①&② → are initial value problems

– basically, start with $u(x,0)$ and/or $\frac{\partial u}{\partial t}(x,0)$

then, move soln forward!

Initial Value PDEs

Flux - Conservative Equations

$$\frac{\partial \underset{\sim}{u}}{\partial t} = -\frac{\partial \underset{\sim}{F}(\underset{\sim}{u})}{\partial x} \qquad \text{(general form)}$$

$\underset{\sim}{F}(\underset{\sim}{u})$ is called the conserved flux.

- Most initial-value PDEs can be put into this general form:

e.g.: $\dfrac{\partial^2 u}{\partial t^2} = c^2 \dfrac{\partial u}{\partial x^2}$ with <u>constant</u> c

let $r \equiv c\dfrac{\partial u}{\partial x}$ & $s \equiv \dfrac{\partial u}{\partial t}$

then evaluate $\dfrac{\partial r}{\partial t} = c\dfrac{\partial}{\partial t}\left(\dfrac{\partial u}{\partial x}\right) = c\dfrac{\partial^2 u}{\partial x \partial t} = c\dfrac{\partial s}{\partial x}$

$$\frac{\partial s}{\partial t} = \frac{\partial^2 u}{\partial t^2} = c^2\frac{\partial^2 u}{\partial x^2} = c\frac{\partial}{\partial x}\left(c\frac{\partial u}{\partial x}\right)$$

(↑ using PDE)

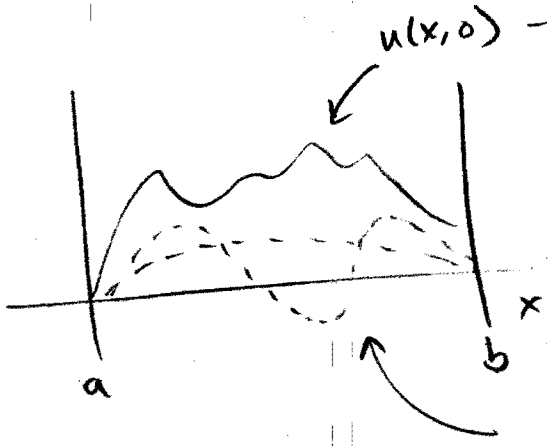
$$= c\frac{\partial r}{\partial x}$$

$$\rightarrow \begin{cases} \dfrac{\partial r}{\partial t} = c\dfrac{\partial s}{\partial x} \\ \dfrac{\partial s}{\partial t} = c\dfrac{\partial r}{\partial x} \end{cases}$$

Well, as we will see next, this scheme won't work! Similar to ODE's with explicit Euler's schemes, this is unstable!

Von Neumann Stability Analysis

- Initial wave can be decomposed into eigenmodes:



e^{ikx} → eigenmode with wave number k.

- assume that the coefficients of the finite difference equation are slowly varying to be considered constant in space and time (which is trivially true in our simple example),

- Then, the time evolution of the wave within this interval can be approximated by

then, let $\underset{\sim}{u} = \begin{pmatrix} r \\ s \end{pmatrix}$

$$\underset{\sim}{F}(\underset{\sim}{u}) = \begin{pmatrix} 0 & -c \\ -c & 0 \end{pmatrix} \cdot \underset{\sim}{u}$$

Then (*) is in $\frac{\partial \underset{\sim}{u}}{\partial t} = -\frac{\partial \underset{\sim}{F}(\underset{\sim}{u})}{\partial x}$!

Stability Concerns for PDEs

— A simple illustrative example

$$\frac{\partial u}{\partial t} = -v \frac{\partial u}{\partial x}$$

★ analytic soln: $\longrightarrow u = f(x - vt)$ where f is an arbitrary fn defined by b.c. & I.C.. (progressing wave)

Finite Differences:

$$\left.\frac{\partial u}{\partial t}\right|_{j,n} = \frac{u_j^{n+1} - u_j^n}{\Delta t} + O(\Delta t) \quad , \quad u_j^n = u(x_j, t_n)$$

(forward Euler)

$$\left.\frac{\partial u}{\partial x}\right|_{j,n} = \frac{u_{j+1}^n - u_{j-1}^n}{2\Delta x} + O(\Delta x^2)$$

(central difference)

explicit scheme $\Rightarrow$

$$\frac{u_j^{n+1} - u_j^n}{\Delta t} = -v\left(\frac{u_{j+1}^n - u_{j-1}^n}{2\Delta x}\right)$$

FTCS (Forward Time Central Space)

Consider a particular mode with wavevector k,

$x = j\Delta x$

$$u_j^n = \xi^n e^{ikj\Delta x}$$

↑ time evolution by a multiplicative factor

What $\xi = \xi(k)$ is a complex function of k only.

☆☆ Note: the time evolution of each eigenmode k is being multiplied by the factor $\xi(k)$ at each time step.

Thus, if $|\xi(k)| > 1$, unstable for mode k!

To see this in more details:

let $u(x,t) = \xi(k,t) e^{ikx}$

then $u_j^n = \xi^n e^{ikj\Delta x}$

putting this into the finite diff. equation:

$$\frac{u_j^{n+1} - u_j^n}{\Delta t} = -v \frac{u_{j+1}^n - u_{j-1}^n}{2\Delta x}$$

$$\xi^{n+1} e^{ikj\Delta x} - \xi^n e^{ikj\Delta x} = -\frac{v\Delta t}{2\Delta x}\left(\xi^n e^{ik(j+1)\Delta x} - \xi^n e^{ik(j-1)\Delta x}\right)$$

Soln advanced in one time step → $\xi^{n+1} = \xi^n\left(1 - \frac{v\Delta t}{2\Delta x}\left(e^{ik\Delta x} - e^{-ik\Delta x}\right)\right)$

$$\xi(k) = 1 - \frac{v\Delta t}{\Delta x} i \left(\frac{e^{ik\Delta x} - e^{-ik\Delta x}}{2i} \right)$$

$$\xi(k) = 1 - i \frac{v\Delta t}{\Delta x} \sin(k\Delta x)$$

So, $|\xi(k)| > 1$ for all $k \neq 0$!

$\Rightarrow$ FTCS is unconditionally unstable !!

Lax's Fix

FTCS can be modified to be stable !

– replace u_j^n in the time derivative by its spatial average across diff. points :

$$u_j^n \rightarrow \frac{u_{j+1}^n + u_{j-1}^n}{2}$$

So, PDE becomes: $u_j^{n+1} = \frac{1}{2}(u_{j+1}^n + u_{j-1}^n) - \frac{v\Delta t}{2\Delta x}(u_{j+1}^n - u_{j-1}^n)$

Now, substitute $u_j^n = \xi^n e^{ikj\Delta x}$ into above,

$$\xi^{n+1} e^{ikj\Delta x} = \frac{1}{2}\left(\xi^n e^{ik(j+1)\Delta x} + \xi^n e^{ik(j-1)\Delta x}\right) - \frac{v\Delta t}{\Delta x}\left(\xi^n e^{ik(j+1)\Delta x} - \xi^n e^{ik(j-1)\Delta x}\right)$$

$$\xi = \frac{1}{2}\left(e^{ik\Delta x} + e^{-ik\Delta x}\right) - \frac{v\Delta t}{\Delta x}\frac{1}{2}\left(e^{ik\Delta x} - e^{-ik\Delta x}\right)$$

$$\xi(k) = \cos k\Delta x - i\frac{v\Delta t}{\Delta x}\sin k\Delta x$$

So, $|\xi(k)| \le 1$, if $\boxed{\left|\frac{v\Delta t}{\Delta x}\right| \le 1}$

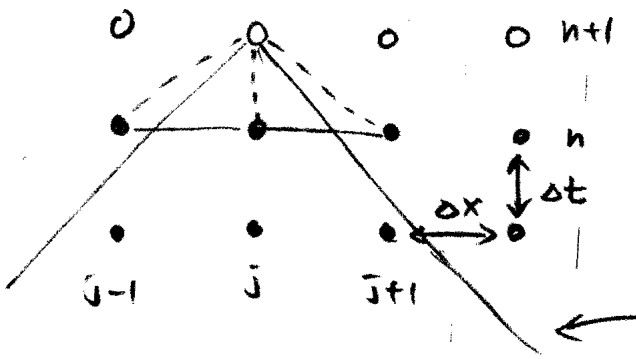
(stable)

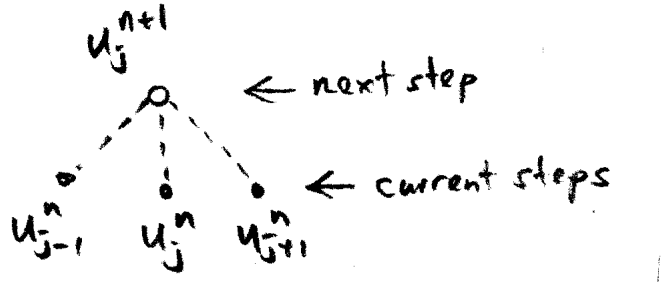
Courant Condition

→ need to choose Δt, Δx so that this condition is satisfied!

Intuitive Understanding of the Stability Condition

$\frac{|v|\Delta t}{\Delta x} \leq 1$ (stable)





slope $= \frac{1}{|v|}$ $(\Delta t/\Delta x)$

★ Note: Since $\frac{|v|\Delta t}{\Delta x} \leq 1$, slope of dotted line (numerical grid) $\leq$ slope of solid line ($\frac{1}{|v|}$)

Conceptual understanding of stable scheme

★★ → Information propagates at speed $\frac{\Delta x}{\Delta t}$ in the grid.

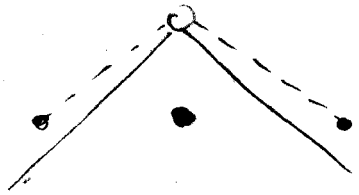
★ Causality requires that past info can only affect future grid points that can be reached by speed $\frac{\Delta x}{\Delta t}$!

⇒

only previous states • in the shaded area can influence o!

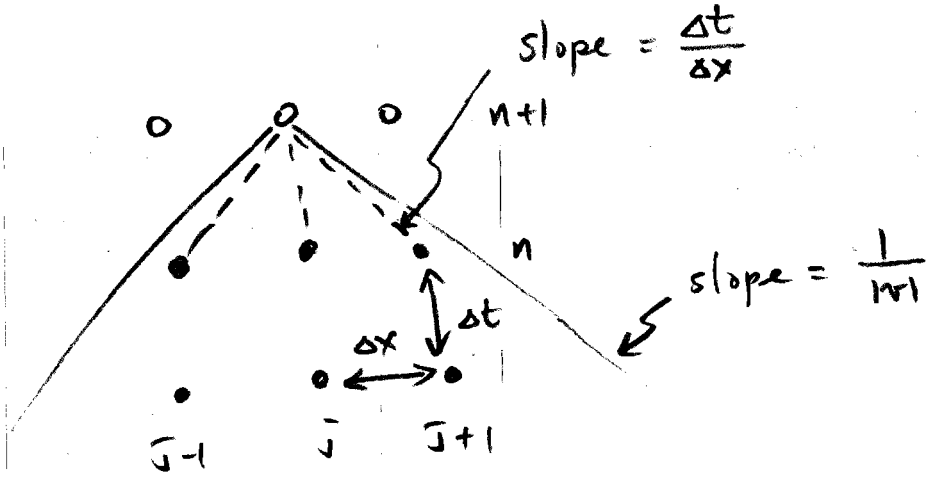
- In the case when $\frac{|v|\Delta t}{\Delta x} \leq 1$ or $\frac{1}{\Delta x/\Delta t} \leq \frac{1}{v}$

Since, the information in the PDE travels at a slower speed v, $\left(v \leq \frac{\Delta x}{\Delta t}\right)$



this choice of $\Delta x, \Delta t$ is stable!

$\frac{|v|\Delta t}{\Delta x} > 1$ (unstable)



$\rightarrow \frac{1}{\Delta x/\Delta t} > \frac{1}{v}$, $\left(v > \frac{\Delta x}{\Delta t}\right)$

- information in PDE travels faster than the grid can support $\rightarrow$ unstable!