nmi | spring 2024

lecture 19 : PDEs

8 partial differential equations

PDE has multiple independent variables, limited here to two.

$$Au_{xx} + Bu_{xy} + Cu_{yy} + F(u_x, u_y, x, y) = 0$$

where x, y are the indendent variables for solution u. properties depending on the leading order terms

1. parabolic if $B^2 - 4AC = 0$
2. hyperbolic if $B^2 - 4AC > 0$
3. elliptic if $B^2 - 4AC < 0$.

ie, hyperbolic and parabolic have boundary conditions at one end of an open region; elliptic have boundary conditions on the entire boundary of a closed region.

8.1 parabolic

the **heat equation** $u_t = Du_{xx}$ represents heat along 1d homogenous rod with **diffusion coefficient** $D > 0$. for the PDE of how heat diffuses, initial and boundary conditions include initial temperature along the rod and measure of what is happening at the ends of the rod as time progresses. properly posed

$$\begin{cases} u_t = Du_{xx} & a \leq x \leq b, t \geq 0 \\ u(x, 0) = f(x) & a \leq x \leq b \\ u(a, t) = l(t) & t \geq 0 \\ u(b, t) = r(t) & t \geq 0 \end{cases}$$

where $f(x)$ gives initial distribution along length $[a, b]$ and $l(t), r(t)$ give temperatures at end points for $t \geq 0$.

8.1.1 forward difference method (FDM)

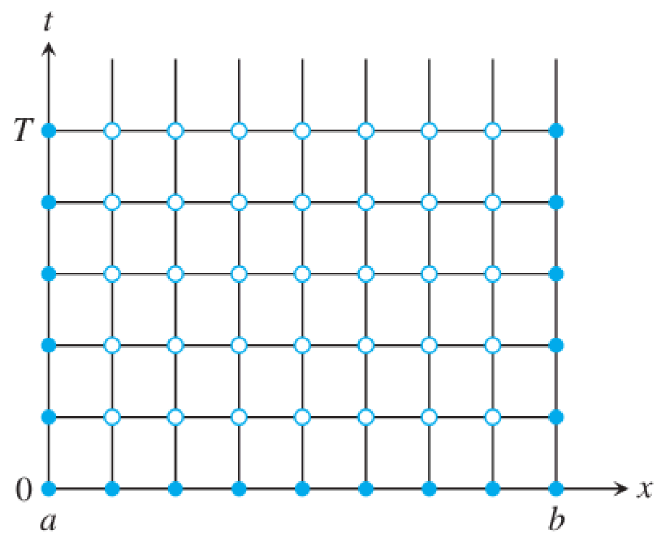


Figure 8.1 Mesh for the Finite Difference Method. The filled circles represent known initial and boundary conditions. The open circles represent unknown values that must be determined.

ie, an ODE may have independent variable t for time while the PDE for heat has independent variables t and x for length.

FDM uses a grid of independent variables and approximates by discretizing PDE. ie, a continuous problem is approximated by a finite number of euqations. a linear PDE can be solved by the methods for systems of equations like gaussian elimination.

to discretize the heat equation on time interval $[0, T]$, consider a mesh of time with distance x where number of steps M, N such that $h = \frac{b-a}{M}$ and $k = \frac{T}{N}$.

$$u_{xx}(x, t) \approx \frac{1}{h^2}(u(x+h, t) - 2u(x, t) + u(x-h, t))$$

with error $\frac{h^2 u_{xxxx}(c_1,t)}{12}$ and

$$u_t(x,t) \approx \frac{1}{k}(u(x,t+k) - u(x,t))$$

with error $\frac{k u_{tt}(x,c_2)}{2}$ where $x-h < c_1 < x+h$ and $t < c_2 < t+h$.

$$\Rightarrow \frac{D}{h^2}(w_{i+1} - 2w_{ij} + w_{i-1}) \approx \frac{1}{k}(w_{i,j+1} - w_{ij})$$

with local truncation error $\mathcal{O}(k) + \mathcal{O}(h^2)$.

given initial and boundary conditions w_{i0} for $i = 0, \ldots, M$ and w_{0j} for $j = 0, \ldots, N$

$$\begin{aligned} \Rightarrow \qquad w_{i,j+1} &= w_{ij} + \frac{Dk}{h^2}(w_{i+1,j} - 2wi,j + w_{i-1,j}) \\ &= \sigma w_{i+1,j} + (1-2\sigma)w_{ij} + \sigma w_{i-1,j}, \qquad \sigma = \frac{Dk}{h^2} \end{aligned}$$

FDM is **explicit** if previous values determine forward values and **implicit** if estimates from the future are involved.

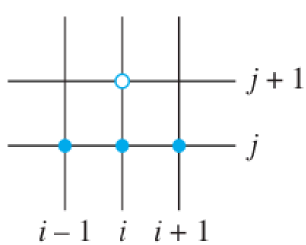


Figure 8.2 Stencil for Forward Difference Method. The open circle represents $w_{i,j+1}$, which can be determined from the values $w_{i-1,j}, w_{ij}$, and $w_{i+1,j}$ at the closed circles by (8.7).

ie, $w_{i,j+1}$ at time t_{j+1} from $w_{j+1} = Aw_j + s_j$

$$\Rightarrow \quad \begin{bmatrix} w_{1,j+1} \\ \vdots \\ w_{m,j+1} \end{bmatrix} = \begin{bmatrix} 1-2\sigma & \sigma & 0 & \dots & 0 \\ \sigma & 1-2\sigma & \sigma & \ddots & \vdots \\ 0 & \sigma & 1-2\sigma & \ddots & 0 \\ \vdots & \ddots & \ddots & \ddots & \sigma \\ 0 & \dots & 0 & \sigma & 1-2\sigma \end{bmatrix} \begin{bmatrix} w_{1,j} \\ \vdots \\ w_{m,j} \end{bmatrix} + \sigma \begin{bmatrix} w_{0,j} \\ \vdots \\ w_{m+1,j} \end{bmatrix}$$

A is $m \times m$ matrix where $m = M - 1$ and right-most vector s_j represents what happens at the ends of the rod.

consider $D = 1$ with initial condition $f(x) = sin^2 2\pi x$ and boundary conditions $u(0,t) = u(1,t) = 0$ for all t.

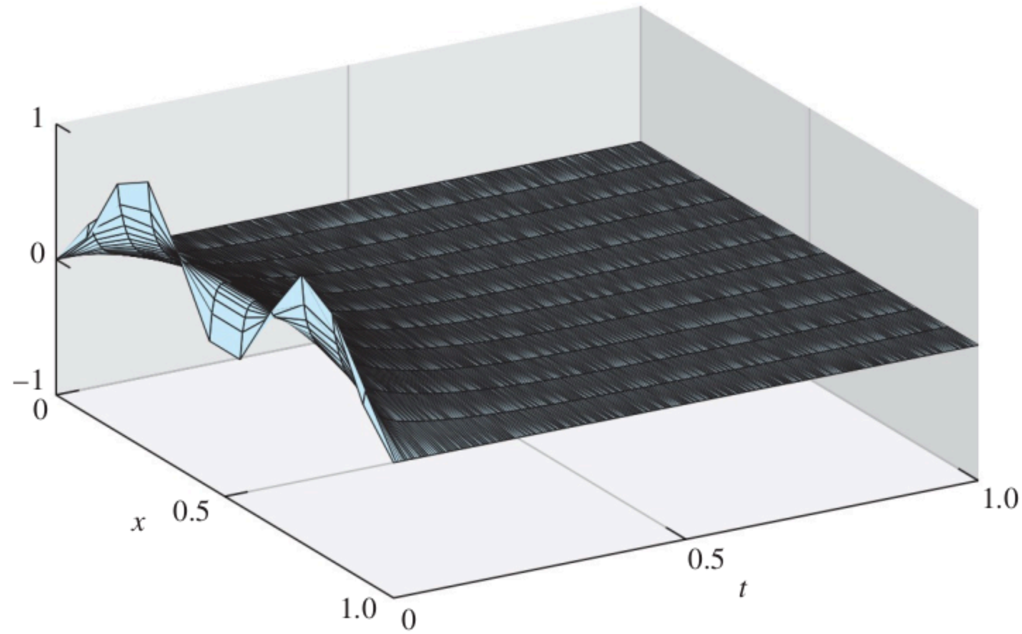
> code, matlab

↳ *1 cell hidden*

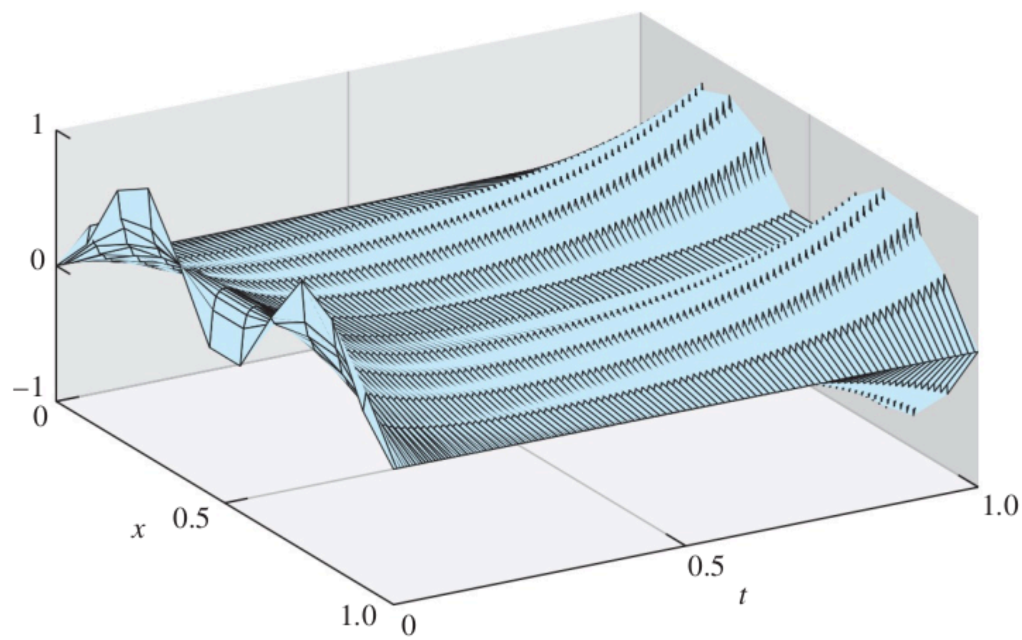
> code, python

[ ] ↳ *1 cell hidden*

⌄ usw



(a)



(b)

Figure 8.3 Heat Equation (8.2) approximation by Forward Finite Difference Method of Program 8.1. The diffusion parameter is $D = 1$, with initial condition $f(x) = \sin^2 2\pi x$. Space step size is $h = 0.1$. The Forward Difference Method is (a) stable for time step $k = 0.0040$, (b) unstable for $k > .005$.

initial temperature peaks flatten towards zero with time but error aggregates, so mind step size k.

8.1.2 FDM stability analysis

so theres error from discretization and theres error magnification. for the latter, use von neumann stability analysis addresses amplification. for a stable method, choose step size whose amplification factor is no larger than one.

let y_j be the exact solution for $y_{j+1} = Ay_j + s_j$, w_j be the computational approximation satisfying $w_{j+1} = Aw_j + s_j$. then

$$\begin{aligned} e_j = w_j - y_j &= Aw_{j-1} + s_{j-1} - (Ay_{j-1} + s_{j-1}) \\ &= A(w_{j-1} - y_{j-1}) \\ &= Ae_{j-1}. \end{aligned}$$

suporting theorem: if $n \times n$ matrix A has spectral radius $\rho(A) < 1$ and b is arbitrary, then for any vector x_0 the iteration $x_{k+1} = A\,x_k + b$ converges. in fact, there exists a unique x_* such that $\lim_{k\to\infty} x_k = x_*, \quad x_* = A\,x_* + b$.

ie, to minimize e_j, require spectral radius $\rho(A) < 1$.

this limits h, k and those limits need more information on the eigenvalues of the symmetric diagonal matrices.

theorem 01

the eigenvectors of T are vectors v_j for $j = 1, \ldots, m$ with corresponding eigenvalues $\lambda_j = 1 - 2 cos \pi j/(m+1)$.

$$T = \begin{bmatrix} 1 & -1 & 0 & \ldots & 0 \\ -1 & 1 & -1 & \ddots & \vdots \\ 0 & -1 & 1 & \ddots & 0 \\ \vdots & \ddots & \ddots & \ddots & -1 \\ 0 & \ldots & 0 & -1 & 1 \end{bmatrix}.$$

proof

from trig for integer i and real number x,

$$\begin{aligned}sin(i-1)x &= sinixcosx - cosixsinx\\ sin(i+1)x &= sinixcosx + consixsinx\end{aligned}$$

$$\Downarrow$$

$$sin(i-1)x + sin(i+1)x = 2sinixcosx$$

$$\Downarrow$$

$$-sin(i-1)x + sinix - sin(i+1)x = (1-2cosx)sinix$$

$$\Downarrow$$

$$v_j = \left[sin\frac{\pi j}{m+1}, sin\frac{2\pi j}{m+1}, \ldots, sin\frac{m\pi j}{m+1}\right].$$

$$\Downarrow$$

$$x = \frac{\pi j}{m+1} \quad \Rightarrow \quad Tv_j = \left(1 - 2cos\frac{\pi j}{m+1}\right)v_j, \quad j = 1, \ldots, m. \blacksquare$$

usw

theorem 01 can be used to find the eigenvalues of any symmetric tridiagonal matrix whose main diagonal and superdiagonal are constant.

eg, for the heat equation of this section, matrix $A = -\sigma T + (1-\sigma)I$ has eigenvalues

$$-\sigma(1-2cos\frac{\pi j}{m+1})+(1-\sigma)=\sigma(cos\frac{\pi j}{m+1}-1)+1, \quad j=1,\dots m.$$

apply theorem for spectral radius, convergence. bc $-2 < cosx-1 < 0$ for $x=\frac{\pi j}{m+1}$ where $1 \le j \le m$, the eigenvalues of $A \in [-4\sigma+1, 1]$. assume diffusion coefficient $D > 0$, then restrict $\sigma < \frac{1}{2}$ to ensure that $\rho(A) < 1$.

theorem 02 FDM stability

let h be the space step and k be the time step for FDM applied to heat equation with $D > 0$. if $\frac{Dk}{h^2} < \frac{1}{2}$, FDM is stable.

8.1.3 backward difference method

implicit, surprise. replace u_{xx} with the backward-difference formula (vs the centered-difference formula).

$$u_t = \frac{1}{k}(u(x,t)-u(x,t-k))+\frac{k}{2}u_{tt}(x,c_0), \quad t-k < c_0 < t$$

$$\Downarrow$$

$$\frac{1}{k}(w_{ij}-w_{i,j-1} = \frac{D}{h^2}(w_{i+1,j}-2w_{ij}+w_{i-1,j}, \quad e_j = \mathcal{O}(k)+\mathcal{O}(h^2)$$

$$\Downarrow$$

$$-\sigma w_{i+1,j}+(1+2\sigma)w_{ij} = w_{i,j-1}, \quad \sigma=\frac{Dk}{h^2}$$

$$\Downarrow$$

$$\begin{bmatrix} 1+2\sigma & -\sigma & 0 & \dots & 0 \\ -\sigma & 1+2\sigma & -\sigma & \ddots & \vdots \\ 0 & -\sigma & 1+2\sigma & \ddots & 0 \\ \vdots & \ddots & \ddots & \ddots & -\sigma \\ 0 & \dots & 0 & -\sigma & 1+2\sigma \end{bmatrix} \begin{bmatrix} w_{1,j} \\ \vdots \\ w_{m,j} \end{bmatrix} = \begin{bmatrix} w_{1,j-1} \\ \vdots \\ w_{m,j-1} \end{bmatrix} + \sigma \begin{bmatrix} w_{0,j} \\ 0 \\ \vdots \\ 0 \\ w_{m+1,j} \end{bmatrix}.$$

> code, matlab

↳ 1 cell hidden

> code, python

[ ] ↳ 1 cell hidden

example 01

apply BDM.

$$\text{example 01} \quad \begin{cases} u_t = Du_{xx} & 0 \le x \le 1, t \ge 0, D = 1 \\ u(x,0) = sin^2 2\pi x & 0 \le x \le 1 \\ u(0,t) = 0 & t \ge 0 \\ u(1,t) = 0 & t \ge 0 \end{cases}$$

use $h = k = 1$. which is the same as the program 2 code but with flipped dependencies for flexible h, k.

usw

why is N so much smaller? as with the von neumann stability analysis of FDM, the relevant quantites are the eigenvalues of A^{-1}. matrix $A = \sigma T + (1+\sigma)I$ has eigenvalues

$$\sigma(1 - 2cos\frac{\pi j}{m+1}) + (1+\sigma) = 1 + 2\sigma(1 - cos\frac{\pi j}{m+1}), \quad j = 1, \dots m.$$

so for $\rho(A^{-1}) < 1$,

$$|1 + 2\sigma(1 - cosx)| > 1$$

which is true for all σ bc $1 - cosx > 0$ and $\sigma = \frac{Dk}{h^2} > 0$. therefore step size depends on local truncation error.

theorem 03 BDM stability

let h be space step and k be time step for BDM applied to heat equation with $D > 0$. then for any h, k BDM is stable.

example 02

apply BDM.

$$\text{example 02} \quad \begin{cases} u_t = Du_{xx} & 0 \leq x \leq 1, t \geq 0, D = 4 \\ u(x,0) = e^{-\frac{x}{2}} & 0 \leq x \leq 1 \\ u(0,t) = e^t & t \geq 0 \\ u(1,t) = e^{t-\frac{1}{2}} & t \geq 0 \end{cases}$$

use $h = k = 0.1$. which is the same as the program 2 code but with flipped dependencies for flexible h, k.

code, matlab

↳ *1 cell hidden*

code, python

[ ] ↳ *1 cell hidden*

usw

recall local truncation error, $e_j = \mathcal{O}(k) + \mathcal{O}(h^2)$. for small $h \approx k$, time step size k will dominate space step size h. ie, $\mathcal{O}(k) + \mathcal{O}(h^2) \approx \mathcal{O}(k)$.

types of boundary conditions. if, for the heat equation, temperatures are given, that would be a **dirichlet** boundary condition. if the boundary is insulated, then a **neumann** boundary condition gives the value of a derivative at the boundary. eg, requiring $u_x(a,t) = u_x(b,t) = 0$ for all t corresponds to an insulated boundary. in general, boundaries set to zero are **homogeneous** boundary conditions.

example 03

apply BDM to heat equation with homogeneous neumann boundary conditions.

$$\text{example 03} \begin{cases} u_t = Du_{xx} & 0 \le x \le 1, t \ge 0, D = 1 \\ u(x,0) = sin^2 2\pi x & 0 \le x \le 1 \\ u(0,t) = 0 & t \ge 0 \\ u(1,t) = 0 & t \ge 0 \end{cases}$$

second-order approximations from ch 5 (bc function values from both sides of x are not available),

$$f'(x) = \frac{-3f(x) + 4f(x+h) - f(x+2h)}{2h} + \mathcal{O}(h^2).$$

$$\Downarrow$$

$$\begin{aligned} u_x(0,t) &\approx \frac{-3u(0,t) + 4u(0+h,t) - u(0+2h,t)}{2h} \\ u_x(1,t) &\approx \frac{-u(1-2h,t) + 4u(1-h,t) - u(1,t)}{2h} \end{aligned}$$

$$\Downarrow$$

$$\begin{aligned} &-3w_0 + 4w_1 - w_2 = 0 \\ &-w_{M-2} + 4w_{M-1} - 3w_M = 0 \end{aligned}$$

with neumann, $A_{m\times m} \to A_{M+1\times M+1}$ with the first and last euqations replaced by neumann conditions w_0, w_M.

> code, matlab

↳ 1 cell hidden

> code, python

[ ] ↳ 1 cell hidden

usw

note: with neumann conditions, the boundary values are not fixed at zero but the solution floats to meet the value of the initial data.

8.1.4 crank-nicolson

so for parabolic PDEs, explicit is sometimes stable and implicit is always stable and both have error $\mathcal{O}(k+h^2)$ when stable and both need small time step k for accuracy. CN uses the backwards-forward difference formula, surprise, is unconditionally stable and has error $\mathcal{O}(h^2)+\mathcal{O}(k^2)$.

eg, for the heat equation, replace u_t with the backward difference formula

$$\frac{1}{k}(w_{ij} - w_{i,j-1})$$

and u_{xx} with the mixed difference

$$\frac{1}{2}\left(\frac{w_{i+1,j} - 2w_{ij} + w_{i-1,j}}{h^2}\right) + \frac{1}{2}\left(\frac{w_{i+1,j-1} - 2w_{i,j-1} + w_{i-1,j-1}}{h^2}\right).$$

with $\sigma = \frac{Dk}{h^2}$

$$2w_{ij} - 2w_{i,j-1} = \sigma[w_{i+1,j} - 2w_{ij} + w_{i-1,j} + w_{i+1,j-1} - 2w_{i,j-1} + w_{i-}$$

$$\Downarrow$$

$$-\sigma w_{i-1,j} + (2+2\sigma)w_{ij} - \sigma w_{i+1,j} = \sigma w_{i-1,j-1} + (2-2\sigma)w_{i,j-1} + \sigma w_{i+1,j-1}.$$

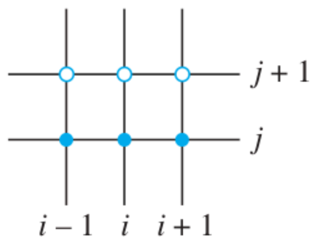


Figure 8.7 Mesh points for Crank–Nicolson Method. At each time step, the open circles are the unknowns and the filled circles are known from the previous step.

in matrix form, $Aw_j = Bw_{j-1} + \sigma(s_{j-1} + s_j), \quad w_j = [w_{1j}, \ldots, w_{mj}]^T$ where

$$A = \begin{bmatrix} 2+2\sigma & -\sigma & 0 & \cdots & 0 \\ -\sigma & 2+2\sigma & -\sigma & \ddots & \vdots \\ 0 & -\sigma & 2+2\sigma & \ddots & 0 \\ \vdots & \ddots & \ddots & \ddots & -\sigma \\ 0 & \cdots & 0 & -\sigma & 2+2\sigma \end{bmatrix}, \quad B = \begin{bmatrix} 2-2\sigma & \sigma & 0 & \cdot\cdot \\ \sigma & 2-2\sigma & \sigma & \ddots \\ 0 & \sigma & 2-2\sigma & \ddots \\ \vdots & \ddots & \ddots & \ddots \\ 0 & \cdots & 0 & \sigma \end{bmatrix}$$

and $s_j = [w_{0j}, 0, \ldots, 0, w_{m+1,j}]^T$.

> code, matlab

↳ 1 cell hidden

> code, python

[ ] ↳ 1 cell hidden

usw

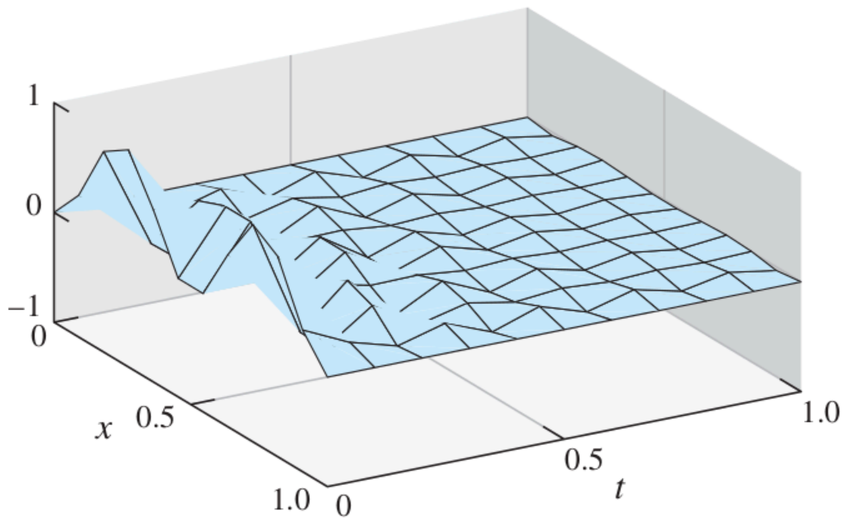


Figure 8.8 Approximate solution of Heat Equation (8.2) computed by Crank–Nicolson Method. Step sizes $h = 0.1, k = 0.1$.

the von neumann stability analysis of CN, the relevant quantites are the eigenvalues of $A^{-1}B$. matrix $A = \sigma T + (2+\sigma)I$, matrix $B = -\sigma T + (2-\sigma)I$ for jth eigenvector v_j of T,

$$\begin{aligned} A^{-1}Bv_j &= (\sigma T + (2+\sigma)I)^{-1} + (-\sigma\lambda_j v_j + (2-\sigma)v_j) \\ &= \frac{1}{\sigma\lambda_j + 2 + \sigma}(-\sigma\lambda_j + 2 - \sigma)v_j \end{aligned}$$

where λ_j is the eigenvalue of T associated with v_j. the eigenvalues of $A^{-1}Bare$

$$\frac{-\sigma\lambda_j + 2 - \sigma}{\sigma\lambda_j + 2 + \sigma} = \frac{4 - (\sigma(\lambda_j + 1) + 2}{\sigma(\lambda_j + 1) + 2} = \frac{4}{L} - 1,$$

where $L = \sigma(\lambda_j + 1) + 2 > 2$ since $\lambda_j > -1$. therefore the eignevalues are between -1 and 1. ie, CN is unconditionally stable like implicit FDM.

however it is not straightforward to derive u_t bc for the wave equation - and poissons equation - only second order derivatives appear.

theorem 04

CN applied to the heat equation with $D > 0$ is stable for any step sizes $h, k > 0$.

CN truncation error, $\mathcal{O}(h^2) + \mathcal{O}(k^2)$, derivation

assume the existence of higher and partial derivatives of u as needed. and previously,

1) $u_t(x,t)$, backward-difference

$$u_t(x,t) = \frac{u(x,t) - u(x,t-k)}{k} + \frac{k}{2!}u_{tt}(x,t) - \frac{k^2}{3!}u_{ttt}(x,t_1), \quad t-k < t_1 < t;$$

2,3) $u_{xx}(x,t)$ and $u_{xx}(x,t-k)$, centered-difference

$$u_{xx}(x,t-k) = u_{xx}(x,t) - ku_{xxt}(x,t) + \frac{k^2}{2}u_{xxtt}(x,t_2), \quad t-k < t_2 < t$$
$$\Rightarrow u_{xx}(x,t) = u_{xx}(x,t-k) + ku_{xxt}(x,t) - \frac{k^2}{2}u_{xxtt}(x,t_2)$$

$$\Downarrow$$

$$u_{xx}(x,t) = \frac{u(x+h,t) - 2u(x,t) + u(x-h,t)}{h^2} + \frac{h^2}{2*3!}u_{xxxx}(x_1,t), \quad x < x_1 < x$$
$$u_{xx}(x,t-k) = \frac{u(x+h,t-k) - 2u(x,t-k) + u(x-h,t-k)}{h^2} + \frac{h^2}{2*3!}u_{xxxx}(x_1,t-k$$

which slot into this incarnation of

4) the heat equation

$$u_t = D\left(\frac{1}{2}u_{xx} + \frac{1}{2}u_{xx}\right).$$

$$\Downarrow$$

$$\frac{u(x,t)-u(x,t-k)}{k}+\frac{k}{2!}u_{tt}(x,t)-\frac{k^2}{3!}u_{ttt}(x,t_1)=$$

$$\frac{D}{2}\left[\frac{u(x+h,t)-2u(x,t)+u(x-h,t)}{h^2}+\frac{h^2}{2*3!}u_{xxxx}(x_1,t)\right]$$

$$+\frac{D}{2}\left[\frac{u(x+h,t-k)-2u(x,t-k)+u(x-h,t-k)}{h^2}+\frac{h^2}{2*3!}u_{xxxx}(x_1,t-k)\right.$$

$$\left.+ku_{xxt}(x,t)-\frac{k^2}{2}u_{xxtt}(x,t_2)\right].$$

$$\Downarrow \quad u_t=u_{xx}$$

$$e_j=-\frac{k}{2!}u_{tt}(x,t)+\frac{k^2}{3!}u_{ttt}(x,t_1)+\frac{Dh^2}{2*2*3!}[u_{xxxx}(x_1,t)+u_{xxxx}(x_2,t-k)]+\frac{Dk}{2}u_{xx}$$

$$\Downarrow \quad Du_{xxt}=(Du_{xx})_t=u_{tt}$$

$$=\frac{k^2}{3!}u_{ttt}(x,t_1)+\frac{Dh^2}{2*2*3!}[u_{xxxx}(x_1,t)+u_{xxxx}(x_2,t-k)]-\frac{Dk^2}{2*2}u_{xxtt}(x,t_2)$$
$$=\frac{k^2}{3!}u_{ttt}(x,t_1)+\frac{h^2}{24D}[u_{tt}(x_1,t)+u_{tt}(x_2,t-k)]-\frac{k^2}{4}u_{ttt}(x,t_2). \quad \checkmark$$

example 04

apply CN.

$$\text{example 04}\quad\begin{cases} u_t=Du_{xx} & 0\le x\le 1, t\ge 0, D=4\\ u(x,0)=e^{-\frac{x}{2}} & 0\le x\le 1\\ u(0,t)=e^t & t\ge 0\\ u(1,t)=e^{t-\frac{1}{2}} & t\ge 0\end{cases}$$

use $h=k=0.1$. which is the same as the program 2 code but with flipped dependencies for flexible h,k.

> code, python

[ ] ↳ *1 cell hidden*

usw

error for FDM.

h	k	$u(0.5, 1)$	$w(0.5, 1)$	error
0.10	0.10	2.11700	2.12015	0.00315
0.10	0.05	2.11700	2.11861	0.00161
0.10	0.01	2.11700	2.11733	0.00033

error for CN.

h	k	$u(0.5, 1)$	$w(0.5, 1)$	error
0.10	0.10	2.11700002	2.11706765	0.00006763
0.05	0.05	2.11700002	2.11701689	0.00001687
0.01	0.01	2.11700002	2.11700069	0.00000067

example 05

apply CN to some population density. lets say prairie dogs in the badlands of the dakotas. bc these methods are about more than heat tho you cant tell by looking at the problem statement in numbers.

$$
\text{example 05} \quad \begin{cases} u_t = Du_{xx} + Cu & 0 \le x \le 1, t \ge 0, D = 4 \\ u(x,0) = sin^2(\frac{\pi}{L}x) & 0 \le x \le L \\ u(0,t) = e^t & t \ge 0 \\ u(L,t) = e^{t-\frac{1}{2}} & t \ge 0 \end{cases}
$$

to make it more incarcerated, note that the dirichlet boundary conditions assume that the population cannot live outside their physical territory of $0 \le x \le L$ this setup is a **reaction-diffusion** equation and the diffusion term Du_{xx} causes the population to spread along x while Cu contributes to the population growth. in reaction-diffusion equations, there is competition between the smoothing tendency of diffusion and the growth contribution of the reaction. so survival here depends on D, C, L.

for CN, apply to u_t,

$$
LHS = \frac{1}{k}(w_{ij} - w_{i,j-1})
$$
$$
RHS = \frac{1}{2}\left(D\frac{w_{i+1,j} - 2w_{ij} + w_{i-1,j}}{h^2} + Cw_{ij}\right) + \frac{1}{2}\left(D\frac{w_{i+1,j-1} - 2w_{i,j-1} + w_{i-1,j-1}}{h^2} +\right.
$$

$$
\Downarrow \quad \sigma = \frac{Dk}{h^2}
$$

$$
-\sigma w_{i+1,j-1} + (2 + 2\sigma - kC)w_{ij} - \sigma w_{i+1,j} = \sigma w_{i-1,j-1} + (2 - 2\sigma + kC)w_{i,j-1} + c
$$

ie, the main diagonal of A needs to subtract kC and the offset needs to add the same. thats two lines of code to change.

$D = 1$ and $C = 9.5$ (extinction!), 10 (survival) where $C > \frac{\pi^2 D}{L^2}$.

> code, matlab

↳ 1 cell hidden

> code, python

[ ] ↳ 1 cell hidden

usw