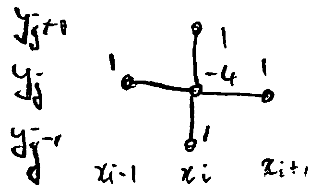
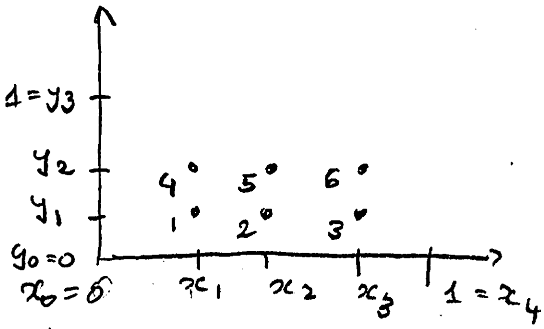
This is known as the 5-point stencil:



There are more accurate approximations to Laplacian that involve 9 points, etc... but we will not see them in class.

The nodes in the grid can be ordered <u>lexicographically</u> (i.e. as in a dictionary). Here is an example with $m=3$, $n=2$, so that we have 6 unknowns to order.



In Matlab if we store all unknowns as a matrix:

$$U \approx \begin{bmatrix} u(x_1,y_1) & u(x_1,y_2) \\ u(x_2,y_1) & u(x_2,y_2) \\ u(x_3,y_1) & u(x_3,y_2) \end{bmatrix}$$

Then the vector with ordering as in the figure is:

$$\boxed{u = U(:)}$$

This command stacks all columns of a matrix on top of each other to create a long vector with same number of elements as U.

With this ordering, the system matrix is of the form:

$$A = \frac{1}{h^2}\begin{bmatrix} T & I & & & \\ I & T & I & & \\ & I & T & I & \\ & & \ddots & \ddots & I \\ & & & I & T \end{bmatrix} \in \mathbb{R}^{n^2 \times n^2}$$

(Assuming $n = m$ for simplicity)

where

$$T = \begin{bmatrix} -4 & 1 & & & \\ 1 & -4 & 1 & & \\ & \ddots & \ddots & \ddots & \\ & & 1 & -4 & 1 \\ & & & 1 & -4 \end{bmatrix} \in \mathbb{R}^{n \times n}$$

and

$$I = \begin{bmatrix} 1 & & & \\ & 1 & & \\ & & \ddots & \\ & & & 1 \end{bmatrix} \in \mathbb{R}^{n \times n}$$

$A \equiv$ block tridiagonal matrix
$\equiv$ penta diagonal matrix (i.e. 5 diagonals)

Since A is sparse it is suitable for direct sparse methods or iterative methods.

<u>Note</u>: using a more accurate approx of Laplacian e.g.

$$\begin{array}{cccc} x_{i-1} & x_i & x_{i+1} & \\ 1 & 4 & 1 & y_{j+1} \\ 4 & -20 & 4 & y_j \\ 1 & 4 & 1 & y_{j-1} \end{array} \equiv \text{9 point stencil}$$

we obtain a system matrix with 9 diagonals.

→ less sparse than with 5-point stencil

→ more expensive to solve.

So better accuracy comes at a greater computational cost.

<u>Note</u>: Here we ordered only the unknowns, since the B.C. determine all the other nodes we did not number.

We can include the boundary nodes in the system, and if we put them, say all at the end, we get the system:

$$\begin{bmatrix} A & 0 \\ 0 & I \end{bmatrix} \begin{bmatrix} u_{\text{interior}} \\ u_{\text{bdry}} \end{bmatrix} = \begin{bmatrix} f \\ 0 \end{bmatrix}$$

where I = identity of size the number of boundary nodes.

This helps in dealing with other Dirichlet B.C. since we can impose u_{bdry} by adding extra equations specifying u_{bdry}.

Local truncation error (LTE)

Recall that:

(A) $$u(x+h) = u(x) + h u'(x) + \frac{h^2}{2} u''(x) + \frac{h^3}{3!} u^{(3)}(x) + \frac{h^4}{4!} u^{(4)}(x+)$$

(B) $$u(x-h) = u(x) - h u'(x) + \frac{h^2}{2} u''(x) - \frac{h^3}{3!} u^{(3)}(x) + \frac{h^4}{4!} u^{(4)}(x-)$$

(A)+(B)

$$\Rightarrow \frac{u(x+h) - 2u(x) + u(x-h)}{h^2} = u''(x) + \frac{h^2}{4!}\left(u^{(4)}(x+) + u^{(4)}(x-)\right)$$

The error term can be rewritten as:

$$\frac{h^4}{12} u(\xi), \quad \text{where } \xi \in [x-h, x+h]$$

using the IVT.

To find the LTE for 2D problem all we need to do is use LTE for finite differences in x and y directions.

$$T_{ij} = \frac{1}{h^2}\Big[u(x_{i-1}, y_j) + u(x_{i+1}, y_j) + u(x_i, y_{j-1}) + u(x_i, y_{j+1})$$

$$- 4u(x_i, y_j)\Big] - f(x_i, y_j)$$

= what remains when we plug solution u to Laplace eq. into numerical method.

$$= \frac{1}{12} h^2 (u_{xxxx} + u_{yyyy}) + O(h^4)$$

$\Rightarrow$ method is <u>second order accurate</u>.

Here is a nifty trick to construct the discretization matrix A in Matlab, using the function

kron.

kron(A,B) replicates matrix B with the "pattern" given by A.

For example, if A is 3×3:

$$A = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix}$$

then:

$$\text{kron}(A,B) = \begin{bmatrix} a_{11}B & a_{12}B & a_{12}B \\ a_{21}B & a_{22}B & a_{23}B \\ a_{31}B & a_{32}B & a_{32}B \end{bmatrix}$$

So: (again assuming $m=n$)

```
I = eye(m);
e = ones(m,1);
T = spdiags([e -4*e e], -1:1, m, m);
S = spdiags([e e], [-1,1], m, m);
A = (kron(I,T) + kron(S,I))/h^2;
```

$$\text{kron}(I,T) = \begin{bmatrix} T & & \\ & \ddots & \\ & & T \end{bmatrix} \qquad \text{kron}(S,I) = \begin{bmatrix} 0 & I & & \\ I & 0 & I & \\ & \ddots & \ddots & I \\ & & I & 0 \end{bmatrix}$$

~~When dealing with systems same analysis can be done and what matters are eigenvalues of A~~

$$\begin{cases} y'(t) = Ay + g \\ y(0) = b \end{cases}$$

~~If system is non-linear then similar concepts can be studied for linearization (wrt y) of $f(t, y)$.~~

Parabolic Problems

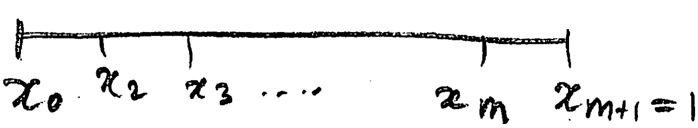
typical parabolic problem is heat equation (or diffusion)

$$\begin{cases} u_t = \kappa u_{xx} \\ u(0,t) = g_0(t) \\ u(1,t) = g_1(t) \\ u(x,0) = \eta(x) \end{cases} \quad \begin{matrix} \left.\begin{matrix} \\ \\ \end{matrix}\right\} \text{B.C. (Dirichlet)} \\ \text{I.C.} \end{matrix}$$

Idea: put together spatial + temporal discr.

$$x_i = ih, \quad i = 0, \dots, m+1, \quad h = \frac{1}{m+1} = \Delta x$$

$$t_n = nk, \quad k = 0, 1, 2, \dots, \quad k = \Delta t.$$
$= \text{time step}$



Notation:

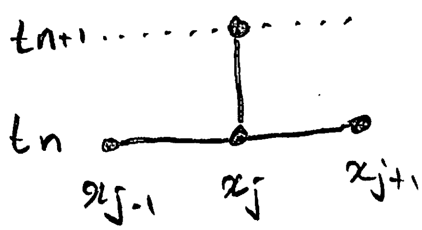
$$U_i^n \approx u(x_i, t_n)$$

One possible discr. is:

$$\boxed{\frac{U_i^{n+1} - U_i^n}{k} = \frac{1}{h^2}\left(U_{i-1}^n - 2U_i^n + U_{i+1}^n\right) \quad \text{(Euler)}}$$

This is explicit since:

$$U_i^{n+1} = U_i^n + \frac{k}{h^2}\left(U_{i-1}^n - 2U_i^n + U_{i+1}^n\right)$$



(stencil)

Another possible discr in time is implicit trap. rule which is also known as Crank-Nicholson:

$$\frac{U_i^{n+1} - U_i^n}{k} = \frac{1}{2}\left(D^2 U_i^{n+1} + D^2 U_i^n\right)$$
$$= \frac{1}{2h^2}\left(U_{i+1}^{n+1} - 2U_i^{n+1} + U_{i-1}^{n+1} + U_{i+1}^n - 2U_i^n + U_{i-1}^n\right)$$

(Implicit)

where we used notation:

$$D^2 U_i^n = \frac{1}{h^2}\left(U_{i+1}^n - 2U_i^n + U_{i-1}^n\right)$$

Putting all new values U_i^{n+1} on one side:

$$-rU_{i-1}^{n+1} + (1+2r)U_i^{n+1} - rU_{i+1}^{n+1} = rU_{i-1}^n + (1-2r)U_i^n + rU_{i+1}^n$$

where $r = \dfrac{k}{2h^2}$.

Since we need to solve a system to find U_i^{n+1}, this is an <u>implicit method</u>. The system is:

$$\begin{bmatrix} 1+2r & -r & & & \\ -r & 1+2r & -r & & \\ & \ddots & \ddots & \ddots & \\ & & -r & 1+2r & -r \\ & & & -r & 1+2r \end{bmatrix} \begin{bmatrix} U_1^{n+1} \\ U_2^{n+1} \\ \vdots \\ U_m^{n+1} \end{bmatrix} = \begin{bmatrix} r(g_0(t_n)+g_0(t_{n+1})) + (1-2r)U_1^n + rU_2^n \\ rU_1^n + (1-2r)U_2^n + rU_3^n \\ \vdots \\ rU_{m-2}^n + (1-2r)U_{m-1}^n + rU_m^n \\ rU_{m-1}^n + (1-2r)U_m^n + r(g_1(t_n)+g_1(t_{n+1})) \end{bmatrix}$$

where we used B.C. $u(0,t) = g_0(t) \equiv U_0^n$
$u(1,t) = g_1(t) \equiv U_{m+1}^n$

<u>Local truncation error</u> (LTE) This is defined in the familiar way: plug in true solution into numerical method and see what is error. For Euler's method:

let $\tau_i^n \equiv \tau(x_i, t_n)$ where

$$\tau(x,t) = \frac{u(x,t+k) - u(x,t)}{k} - \frac{1}{h^2}\left(u(x-h,t) - 2u(x,t) + u(x+h,t)\right)$$

$$= \left(u_t + \frac{1}{2}k u_{tt} + \frac{1}{6}k^2 u_{ttt} + \dots\right) - \left(u_{xx} + \frac{1}{12}h^2 u_{xxxx} + \dots\right)$$

Since $u_t = u_{xx}$ (u solves PDE!), first term cancels

Moreover: $u_{tt} = (u_{xx})_t = u_{xxxx}$

thus:

$$\boxed{\tau(x,t) = \left(\tfrac{1}{2}k - \tfrac{1}{12}h^2\right)u_{xxxx} + O(k^2 + h^4)}$$

$\Rightarrow$
- second order accurate in space
- first order " " time

(since $\tau(x,t) = O(k + h^2)$)

For Crank-Nicholson, it is possible to show that:

$$\tau(x,t) = O(k^2 + h^2)$$

= second order accurate in time & space.

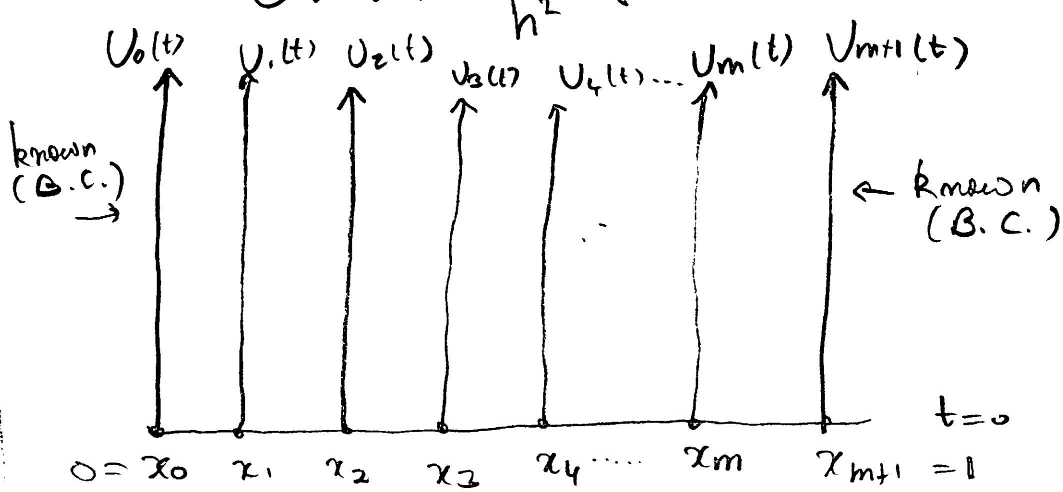
We can get a theoretical insight by relating parabolic problem to a system of ODEs:

Method of lines discretization

Idea: discretize in space first $\to$ obtain system of ODEs where each component corresponds to sol of PDE at a grid point. i.e.:

$$U_i'(t) = \frac{1}{h^2}\left(U_{i-1}(t) - 2U_i(t) + U_{i+1}(t)\right)$$

for $i = 1, \dots, m$



In matrix form this system of ODEs becomes:

$$(*) \qquad \underline{U}'(t) = A\,\underline{U}(t) + \underline{g}(t)$$

$$A = \frac{1}{h^2}\begin{bmatrix} -2 & 1 & & & \\ 1 & -2 & 1 & & \\ & \ddots & \ddots & \ddots & \\ & & 1 & -2 & 1 \\ & & & 1 & -2 \end{bmatrix}$$

$$\underline{g}(t) = \frac{1}{h^2}\begin{bmatrix} g_0(t) \\ 0 \\ \vdots \\ 0 \\ g_1(t) \end{bmatrix} \qquad \text{(takes care of B.C.)}$$

Can use ODE software to discr. (*) in time, say RK4, however we can learn more by revisiting Euler and Crank-Nicholson and stability analysis will follow from that for systems of ODEs:

Euler's method

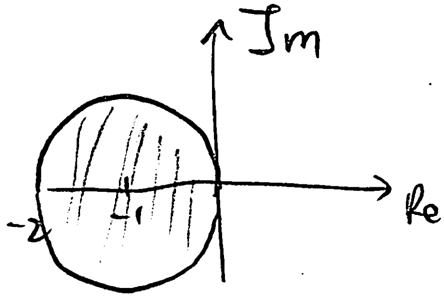
$$U^{n+1} = U^n + k\,f(U^n) \qquad (\text{where } f(U) = AU + g)$$

Trapezoidal rule (C.N.)

$$\frac{U^{n+1} - U^n}{k} = \frac{1}{2}\left(f(U^{n+1}) + f(U^n)\right)$$

Stability.

For Euler's method:



$$R = \{\omega \mid |1+\omega| < 1\}.$$

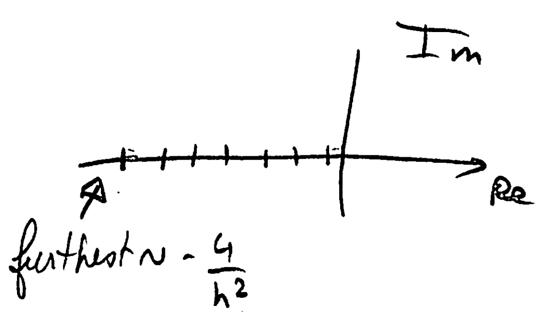
we need all products $\lambda_p(A) h \in R$.

In this case A has closed form eigenvalues

$$\lambda_p(A) = \frac{2}{h^2}\left(\cos(p\pi h) - 1\right), \quad p = 1, \dots, m$$

$$\left(h = \frac{1}{m+1}\right)$$

$$\Rightarrow \quad \frac{-4}{h^2} \le \lambda_p(A) \le 0$$



Thus requiring that all eigenvalues lie inside stability region for Euler's method:

$$\left|1 - \frac{4k}{h^2}\right| \le 1$$

very restrictive

$$-2 \le \frac{-4k}{h^2} \le 0 \quad \Rightarrow \quad \boxed{\frac{k}{h^2} \le \frac{1}{2}}$$

Example: Trapezoidal rule (Crank-Nicholson)

Trapz. rule is A-stable (ie. abs. stab region is left hand plane)

$\Rightarrow$ C.N. is stable for any time-step

⚠ this doesn't mean method is accurate for any time step. Large time steps will probably give inaccurate results!

Convergence methods we've seen are of the form:

$$(**) \quad U^{n+1} = \underbrace{B(k)}_{\in \mathbb{R}^{m\times m}} U^n + \underbrace{b^n(k)}_{\in \mathbb{R}^m}, \quad h = \frac{1}{m+1}$$

we will assume $h = R(k)$, i.e. that we have fixed spatial discretization with time step according to some rule. e.g by analysis we did before we can use $k = 0.4 h^2$ which satisfies $k/h^2 \leq \frac{1}{2}$ automatically

For Euler's method:

$$B(k) = I + kA$$

For Crank-Nicholson:

$$B(k) = \left(I - \frac{k}{2}A\right)^{-1}\left(I + \frac{k}{2}A\right)$$

To show convergence we need consistency (i.e. local truncation error $\to 0$ as $k \to 0$ and $h \to 0$)

and some kind of stability:

Def (Lax-Richtmyer) A linear method of the form $(**)$ is said to be Lax-Richtmyer stable if for all time T there is a constant $C_T > 0$ s.t.

$$\|B(k)^n\| \leq C_T$$

for all $k > 0$ and integers n s.t. $\underbrace{kn}_{\text{discrete time}} \leq T$

Theorem (Lax Equivalence Theorem) A consistent method $(**)$ is convergent iff it is Lax-Richtmyer stable

The idea is the same as stability for ODE:
apply numerical method to exact solution $u(x,t)$:

$$u^{n+1} = Bu^n + b^n + k\underbrace{\tau^n}_{\text{local trunc. err.}}$$

where $u^n = \begin{bmatrix} u(x_1,t_n) \\ u(x_2,t_n) \\ \vdots \\ u(x_m,t_n) \end{bmatrix}$

$\tau^n = \begin{bmatrix} \tau(x_1,t_n) \\ \tau(x_2,t_n) \\ \vdots \\ \tau(x_m,t_n) \end{bmatrix}$ = local truncation error

subtracting the difference eq (our method):

$$U^{n+1} = BU^n + b^n$$

we get $E^{n+1} = BE^n - k\tau^n$, where $E^n = U^n - u^n$

hence after N time steps:

$$E^N = B^N E^0 - k\sum_{n=1}^{N} B^{N-n}\tau^{n-1}$$

$$\Rightarrow \quad \|E^N\| \le \|B^N\|\,\|E^0\| + k\sum_{n=1}^{N}\|B^{N-n}\|\,\|\tau^{n-1}\|$$

if method is L-R. stable then: for $Nk \le T$:

$$\|E^N\| \le C_T\|E^0\| + TC_T \max_{1\le n\le N}\|\tau^{n-1}\|$$

$\to 0$ as $k\to 0$ for $Nk \le T$.

since method is consistent we do have:

$\|\tau^n\| \to 0$ and we need good I.C. s.t. $\|E^0\| \to 0$ as $k\to 0$

Example : for heat eq :

$$B(k) = I + kA = \text{symm matrix.}$$

The eigenvalues of $B(k)$ are :

$$\lambda_p(B(k)) = 1 + k\,\lambda_p(A(k)) = 1 + \frac{2k}{h^2}\underbrace{(\cos(p\pi h) - 1)}_{-2 \leq \cdot \leq 0}$$

But we assumed that $\frac{k}{h^2} \leq \frac{1}{2}$:

$\underbrace{\qquad\qquad}_{-2 < \cdot \leq 0}$

$$\Rightarrow \quad |\lambda_p(B(k))| \leq 1$$

$$\Rightarrow \quad \|B(k)\| \leq 1$$

for Crank-Nicholson.

$$B(k) = (I - \tfrac{k}{2}A)^{-1}(I + \tfrac{k}{2}A)$$

$$\lambda_p(B(k)) = \frac{1 + k\lambda_p/2}{1 - k\lambda_p/2} \leq 1$$

So C-N is stable for all n.