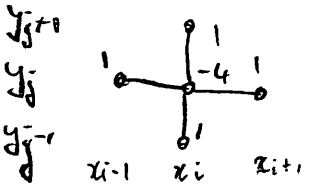
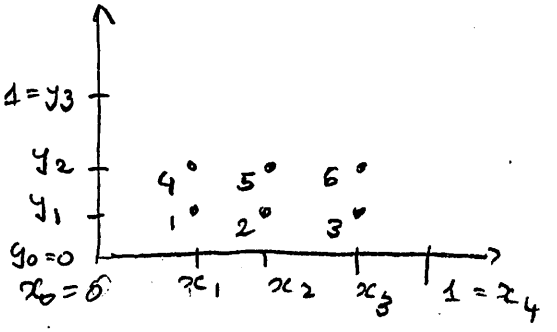
This is known as the 5-point stencil:



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:

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

(Assuming $n = m$ for simplicity)

where

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

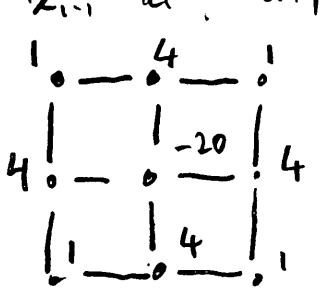
and

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

$A \equiv$ 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.

Note: using a more accurate approx of Laplacian e.g.



$\equiv$ 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.

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

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

(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.

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

$\Rightarrow$ 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:

$$\text{kron}(A,B) = \begin{bmatrix} a_{11}B & a_{12}B & a_{13}B \\ a_{21}B & a_{22}B & a_{23}B \\ a_{31}B & a_{32}B & a_{32}B \end{bmatrix}$$

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;
```

$$\begin{bmatrix} T & & \\ & \ddots & \\ & & T \end{bmatrix} \qquad \begin{bmatrix} 0 & I & & \\ I & 0 & I & \\ & \ddots & \ddots & I \\ & & I & 0 \end{bmatrix}$$

Before we continue with Parabolic problems, we need to review A-stability and the absolute stability region.

Here are some preliminaries.

Difference equation fundamentals

$x = (x_1, x_2, x_3, \cdots)$

$y = (y_1, y_2, y_3, \cdots)$ = sequences

A difference equation can be written in terms of the shift operator:

$$Ex = (x_2, x_3, x_4, \cdots), \text{ where } x = (x_1, x_2, \cdots)$$

It is easy to show that:

$$(Ex)_n = x_{n+1}$$

$$(E^k x)_n = x_{n+k}$$

$$E^0 x = x$$

Linear difference operator is:

$$L = \sum_{i=0}^{m} a_i E^i$$

A linear difference eq. is $Lx = 0$

So we want to find all x in null (L).

Example:

$$x_{n+2} - 3x_{n+1} + 2x_n = 0 \qquad (DE)$$

$$\Leftrightarrow \left(E^2 - 3E + 2E^0\right) x = 0$$

$$\Leftrightarrow \quad p(E)\, x = 0$$

where $p(\lambda) = \lambda^2 - 3\lambda + 2$ = polynomial of degree 2.

The solutions to a linear difference equation can be obtained from <u>roots of $p(\lambda)$</u>:

<u>Theorem (simple roots)</u>: If p is a polynomial and λ a root of $p(\lambda)$, then a solution to the linear difference eq.

$$p(E)\, x = 0$$

is

$$x = (\lambda, \lambda^2, \lambda^3, \lambda^4, \dots) \qquad (*)$$

Moreover if all roots of p are simple (multiplicity 1) and $\neq 0$ all solutions to $p(E) = 0$ are in the span of all solutions of form $(*)$ where λ = root of $p(\lambda)$.

<u>check (with example)</u> $\quad p(\lambda) = \lambda^2 - 3\lambda + 2 = (\lambda - 1)(\lambda - 2)$

$x = (1, 1, 1, 1, \dots)$ satisfies (DE)

$x = (2, 2^2, 2^3, \dots)$ also satisfies (DE) since:

$$p(E)\, x = \left(2\,p(2),\ 2^2 p(2),\ 2^3 p(2), \dots\right) = (0, 0, 0, \dots)$$

In general it is not hard to verify:

$$E(\lambda, \lambda^2, \lambda^3, \dots) = \lambda(\lambda, \lambda^2, \lambda^3, \dots)$$
$$E^k(\lambda, \lambda^2, \lambda^3, \dots) = \lambda^k(\lambda, \lambda^2, \lambda^3, \dots)$$

Thus:

$$p(E)(\lambda, \lambda^2, \lambda^3, \dots) = p(\lambda)(\lambda, \lambda^2, \lambda^3, \dots)$$
$$= 0$$

when λ is a root of p.

The general case where roots are allowed to be multiple is summarized in following theorem:

Theorem (multiple roots):

Let p be a poly with $p(0) \neq 0$. Then a basis for nullspace $(p(E))$ is:

with each root λ of p with multiplicity k, associate k solutions:

$$x(\lambda), x'(\lambda), x''(\lambda), \dots, x^{(k-1)}(\lambda)$$

where

$$x(\lambda) = (\lambda, \lambda^2, \lambda^3, \dots)$$
$$x'(\lambda) = (1, 2\lambda, 3\lambda^2, \dots)$$
$$x''(\lambda) = (0, 2, 6\lambda, \dots)$$
$$\vdots \text{ etc.}$$

A-Stability and Absolute stability region

Consider the simple problem

$$(*)\begin{cases} y' = \lambda y \\ y(0) = 1 \end{cases} \quad \text{with solution } y(t) = e^{\lambda t}.$$

If we allow $\lambda = a + ib \in \mathbb{C}$, then

$$y(t) = \underbrace{e^{at}}_{\text{exp. decay a growth}} \underbrace{e^{ibt}}_{\hookrightarrow \text{ oscillatory since } e^{ibt} = \cos(bt) + i\sin(bt).}$$

Thus

$$y(t) \to 0 \text{ as } t \to \infty \iff \mathrm{Re}(\lambda) < 0.$$

We want our numerical method to preserve this decay.

Example Euler's method applied to $(*)$

$$y_{n+1} = y_n + h f(t_n, y_n)$$

$$\begin{aligned} y_0 &= 1 \\ y_1 &= y_0 + h\lambda y_0 = (1 + h\lambda) y_0 \\ y_2 &= y_1 + h\lambda y_1 = (1 + h\lambda)^2 y_0 \\ &\vdots \\ y_n &= (1 + h\lambda)^n y_0 \end{aligned}$$

The only way we can have

$$y_n \to 0 \quad \text{as} \quad n \to \infty$$

to mimic behavior of true sol. $y(t) = e^{\lambda t}$, when $\text{Re}(\lambda) < 0$; is when:

$$|1 + h\lambda| < 1$$

This restricts time step! Let $\lambda < 0$, be a real number then inequality becomes:

$$-1 < 1 + h\lambda < 1$$

$$-2 < h\lambda < 0$$

$$\boxed{h < -\frac{2}{\lambda}}$$

→ time step needs to be small enough to capture "time scale" of decay λ.

→ if $-\lambda$ is large then h needs to be even smaller! Makes sense: we need finer resolution in time to capture such transient behaviors.

Basically same analysis can be carried on systems.

$$\begin{cases} \underline{y}'(t) = A\underline{y} \\ \underline{y}(0) = \underline{b} \end{cases}$$

Solution is: $\underline{y}(t) = e^{tA}\underline{b}$

where if A is symmetric w/ eigenvalue decomposition:

$$A = Q\Lambda Q^T, \quad \Lambda = \text{diag}(\lambda_1, \dots, \lambda_n), \quad Q^TQ = I$$

$$e^{tA} = Q e^{t\Lambda} Q^T$$

and $$e^{t\Lambda} = \begin{bmatrix} e^{t\lambda_1} & & & \\ & e^{t\lambda_2} & & \\ & & \ddots & \\ & & & e^{t\lambda_n} \end{bmatrix}$$

So systems have many time scales and if we want to mimic overall decay of solution when $\text{Re}(\lambda_i) < 0$, we need in Euler's method:

$$|1 + k\lambda_i| < 1 \quad \text{for } \underline{\text{all}} \text{ eigenvalues } \lambda_i \text{ of } A.$$

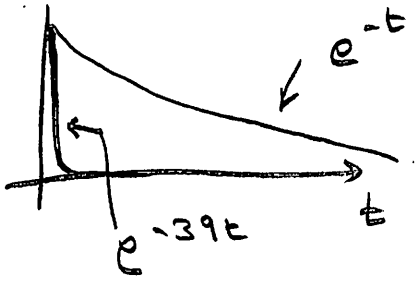
Here is an example:

$$\underline{y}' = \begin{bmatrix} -20 & -19 \\ -19 & -20 \end{bmatrix} \underline{y}$$

$$\underline{y}(0) = \begin{pmatrix} 2 \\ 0 \end{pmatrix}$$

$$\underline{y}(t) = \begin{bmatrix} e^{-39t} + e^{-t} \\ e^{-39t} - e^{-t} \end{bmatrix}$$

So we do have $\underline{y}(t) \to 0$ as $t \to \infty$, but at drastically different timescales.



Here time step needs to be

$$h < \frac{-2}{-1} \quad \text{and} \quad h < \frac{-2}{-39},$$

much smaller.

We now look at general case:

Linear multistep methods

(M) $a_k y_n + a_{k-1} y_{n-1} + \cdots + a_0 y_{n-k} = h[b_k f_n + \cdots + b_0 f_{n-k}]$

Here a_k, b_k are scalar coefficients

$y_n \approx y(t_n)$

$f_n \equiv f(t_n, y_n)$

Applying (M) to (*) we get.

$$a_k y_n + \cdots + a_0 y_{n-k} = h\lambda[b_k y_n + \cdots + b_0 y_{n-k}]$$

$\Leftrightarrow$

$$(a_k - h\lambda b_k) y_n + (a_{k-1} - h\lambda b_{k-1}) y_{n-1} + \cdots + (a_0 - h\lambda b_0) y_{n-k} = 0$$

This is a <u>difference equation</u>, its characteristic polynomial can be written as:

$$\phi(z) = (a_k - h\lambda b_k) z^k + (a_{k-1} - h\lambda b_{k-1}) z^{k-1} + \cdots + (a_0 - h\lambda b_0) z^0$$
$$= p(z) - h\lambda q(z)$$

where:

$$p(z) = \sum_{i=0}^{k} a_i z^i$$

$$q(z) = \sum_{i=0}^{k} b_i z^i$$

We can now see that when $\mathrm{Re}(\lambda) < 0$ if we want method (M) to mimic decay of true solution, we need: **all roots of ϕ inside $\{\zeta \mid |\zeta| < 1\}$**

Definition (A-stability) When $h > 0$ and $\mathrm{Re}(\lambda) < 0$, an A-stable method of the form (M) is s.t. all roots of

$$\phi(\zeta) = p(\zeta) - h\lambda\, q(\zeta)$$

have magnitude < 1.

Example: Implicit trapezoidal method

$$y_n - y_{n-1} = \frac{1}{2}h\left[f_n - f_{n-1}\right]$$

$$p(\zeta) = \zeta - 1$$

$$q(\zeta) = \frac{1}{2}(\zeta + 1)$$

$$\phi(\zeta) = \zeta - 1 - \frac{\lambda h}{2}(\zeta + 1)$$

root: $$\boxed{\zeta_* = \frac{2 + \lambda h}{2 - \lambda h}}$$

$$|2 + \lambda h|^2 = 4 + 4\,\mathrm{Re}(\lambda h) + |\lambda h|^2$$
$$|2 - \lambda h|^2 = 4 - 4\,\mathrm{Re}(\lambda h) + |\lambda h|^2$$

Therefore when $\mathrm{Re}(\lambda h) < 0$:

$$|2+\lambda h|^2 < |2-\lambda h|^2 \Rightarrow |\xi_+| < 1.$$

Note: A result by Dahlquist states that all A-stable linear multistep methods must be implicit and with max. order 2.

⇝ implicit trapz. rule is often used as it achieves maximum accuracy.

However A-stability is too restrictive.

Definition (Region of absolute stability)

$$R = \{\omega \in \mathbb{C} \mid \text{all roots } r \text{ of } p - \omega q \text{ are s.t. } |r| < 1\}$$

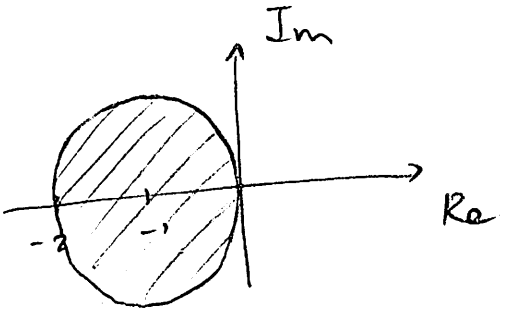
- A method will mimic decay of sol of (*) when $\mathrm{Re}(\lambda) < 0$ if $\lambda h \in R$ = region of absolute stability.
- A method (M) is A-stable iff region of abs. stab.
  $= \{\xi \in \mathbb{C} \mid \mathrm{Re}\,\xi < 0\}$
  = left half plane.

Example: Region of Absolute stability for Euler's method.

$$y_n - y_{n-1} = h f_{n-1}$$

$$\Rightarrow \rho(\zeta) = \zeta - 1$$
$$\sigma(\zeta) = 1$$
$$\phi(\zeta) = \zeta - 1 - \omega, \text{ root: } \zeta^* = 1 + \omega$$

$\Rightarrow$ need $|1+\omega| < 1$



- When a method that is not A-stable is used, it is hoped that $\omega = \lambda h \in$ abs. stab. region.
- If we fix h, and let $\lambda h \in$ negative real axis (pure exp. decay) then there is only a certain range of "time scales" that Euler's method will be able to deal with without blowing up. Go too far in negative real axis and solution will blow up.
- Now take $\lambda h \in$ negative real axis and abs. stab. region for Euler's method. Then if we perturb λ with a purely imaginary ia: $(\lambda + ia)h$, there is only a certain range of "frequencies" for which Euler will not blow up. If freq is too high (a large) then Euler will not keep up.

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

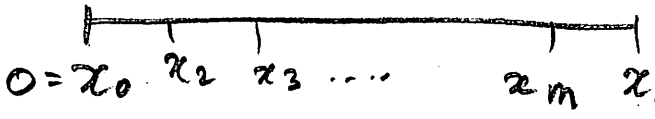
typical parabolic problem is heat equation (or diffusion)

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

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:

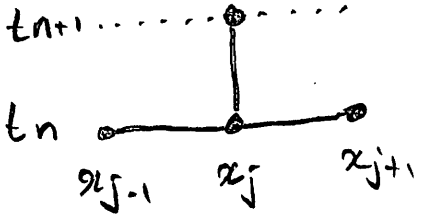
$$U_i^n \approx u(x_i, t_n)$$

One possible discr. is:

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

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 trapez. 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

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

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

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

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

thus:

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

$\Rightarrow$
- 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.

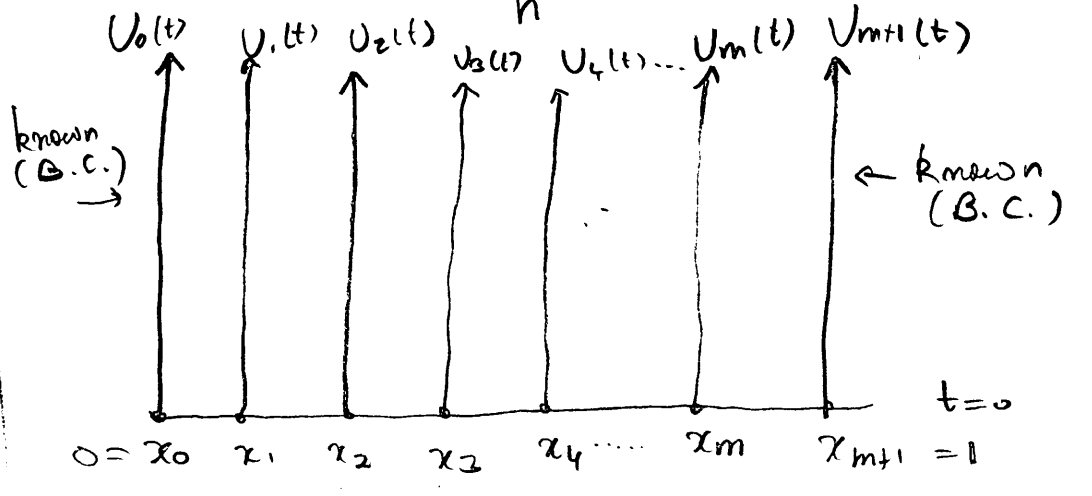
We can get a theoretical insight by relating parabolic problem to a system of ODEs:

Method of lines discretization

Idea: discretize in space first → 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 $i = 1, \ldots, m$



In matrix form this system of ODEs becomes:

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

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

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

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

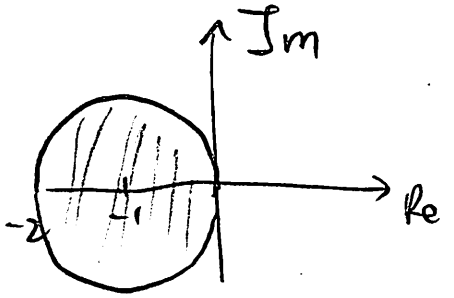
$$U^{n+1} = U^n + k\, f(U^n) \quad (\text{where } f(U) = AU + g)$$

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 Eulers method:



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

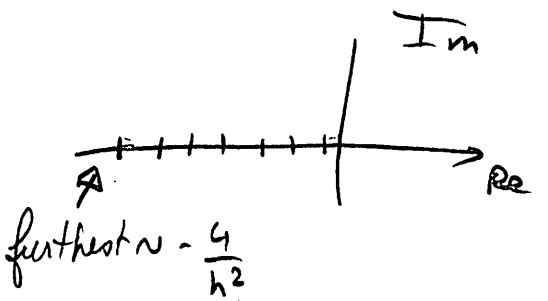
we need all products $\lambda_p(A)\, h \in R$.

In this case A has closed form eigenvalues

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

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

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



Thus requiring that all eigenvalues lie inside stability region for Eulers method:

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

very restrictive

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

Example: Trapezoidal rule (Crank-Nicholson)

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

$\Rightarrow$ C.N. is <u>stable for any time-step</u>

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