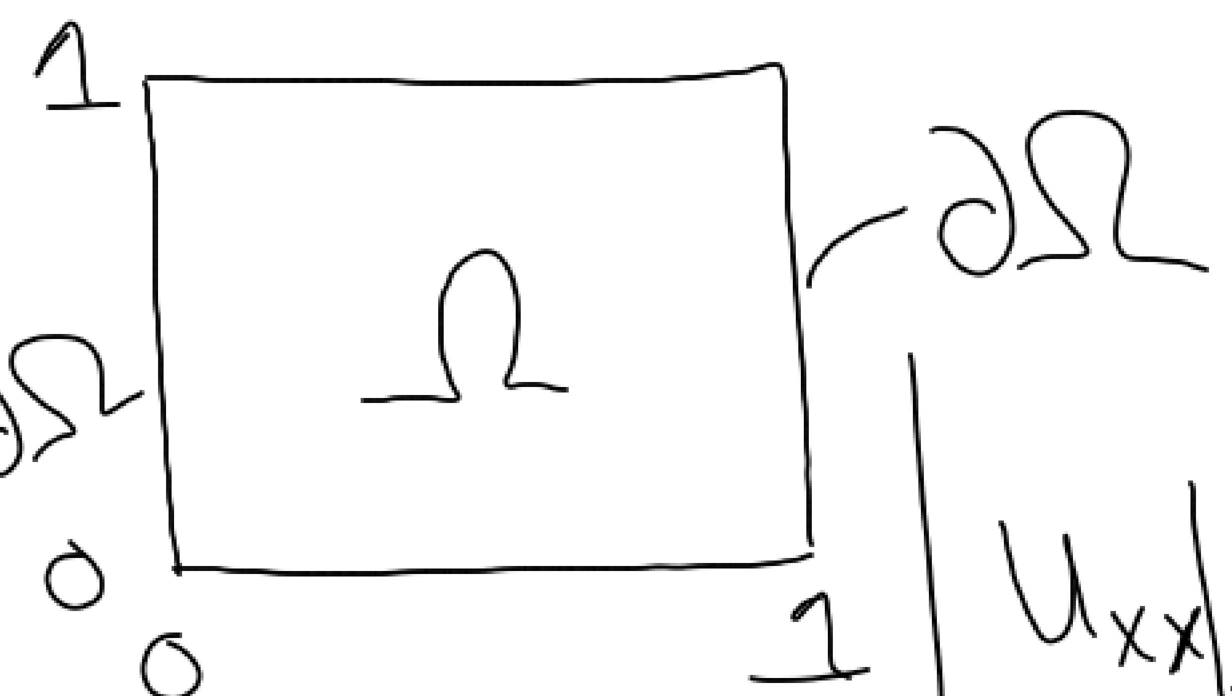
The Heat Equation in 2D

$$u_t = u_{xx} + u_{yy} \qquad u = u(x,y,t)$$

$$u(x,y,t=0) = \eta(x,y)$$

$$u(x,y,t) = g(x,y,t) \text{ for } (x,y) \in \partial\Omega$$



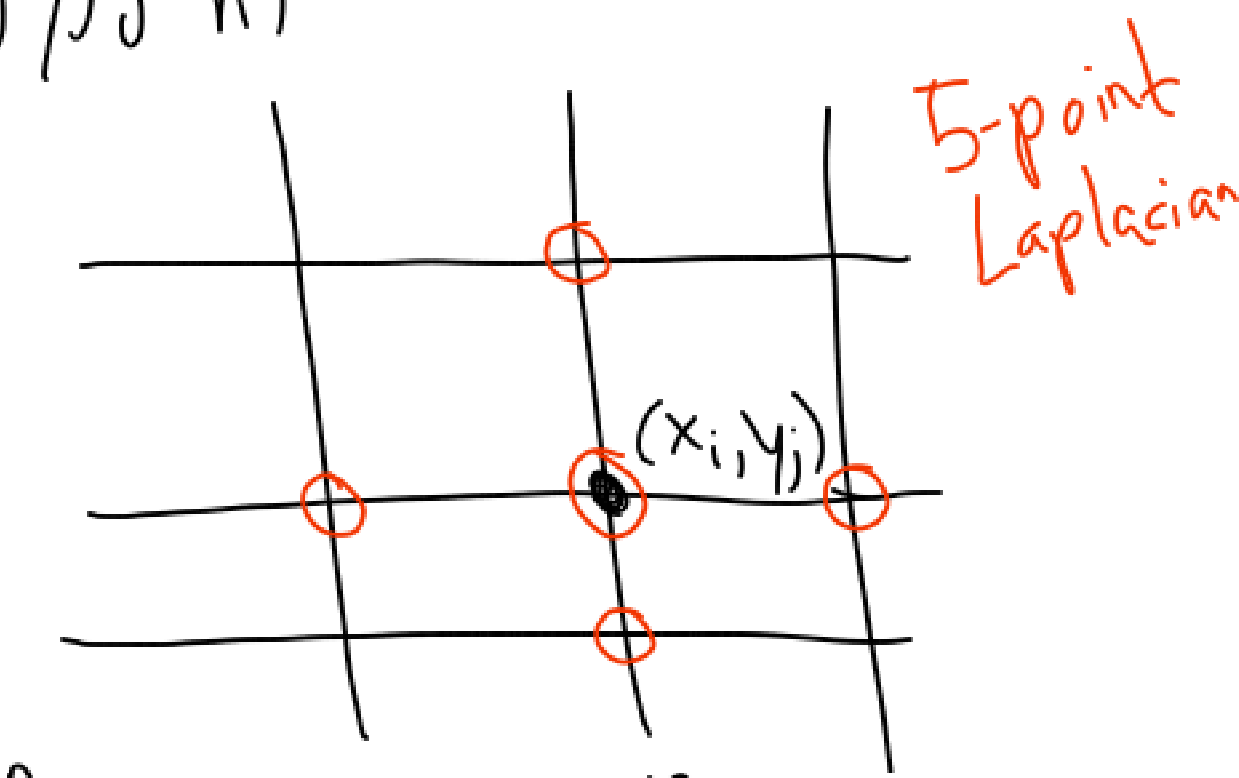
Discretize in space:

$$x_i = ih \qquad i,j = 0,1,\dots,m+1$$

$$y_j = jh \qquad h = \frac{1}{m+1}$$

$$U_{ij}(t) \approx u(x_i, y_j, t) \quad \text{Semi-discrete}$$

$$U_{ij}^n \approx u(x_i, y_j, t_n)$$



$$u_{xx}\big|_{x_i,y_j,t_n} \approx \frac{U_{i+1,j}^n - 2U_{ij}^n + U_{i-1,j}^n}{h^2} = (D_x^2 U)_{ij}$$

$$u_{yy}\big|_{x_i,y_j,t_n} \approx \frac{U_{i,j+1}^n - 2U_{ij}^n + U_{i,j-1}^n}{h^2} = (D_y^2 U)_{ij}$$

We need to choose an ordering of all m^2 grid points to write U^n as a vector. For example, row-wise ordering:

$$U = \begin{bmatrix} U_{11} \\ U_{21} \\ U_{31} \\ \vdots \\ U_{m1} \\ U_{12} \\ \vdots \\ U_{m2} \\ \vdots \\ \vdots \\ U_{mm} \end{bmatrix}$$

$$\left.U'(t) = D_x^2 U(t) + D_y^2 U(t)\right] \text{System of } m^2 \text{ ODEs}$$

$$= \nabla_h^2 U(t)$$

$$\nabla_h^2 \in \mathbb{R}^{m^2 \times m^2}$$

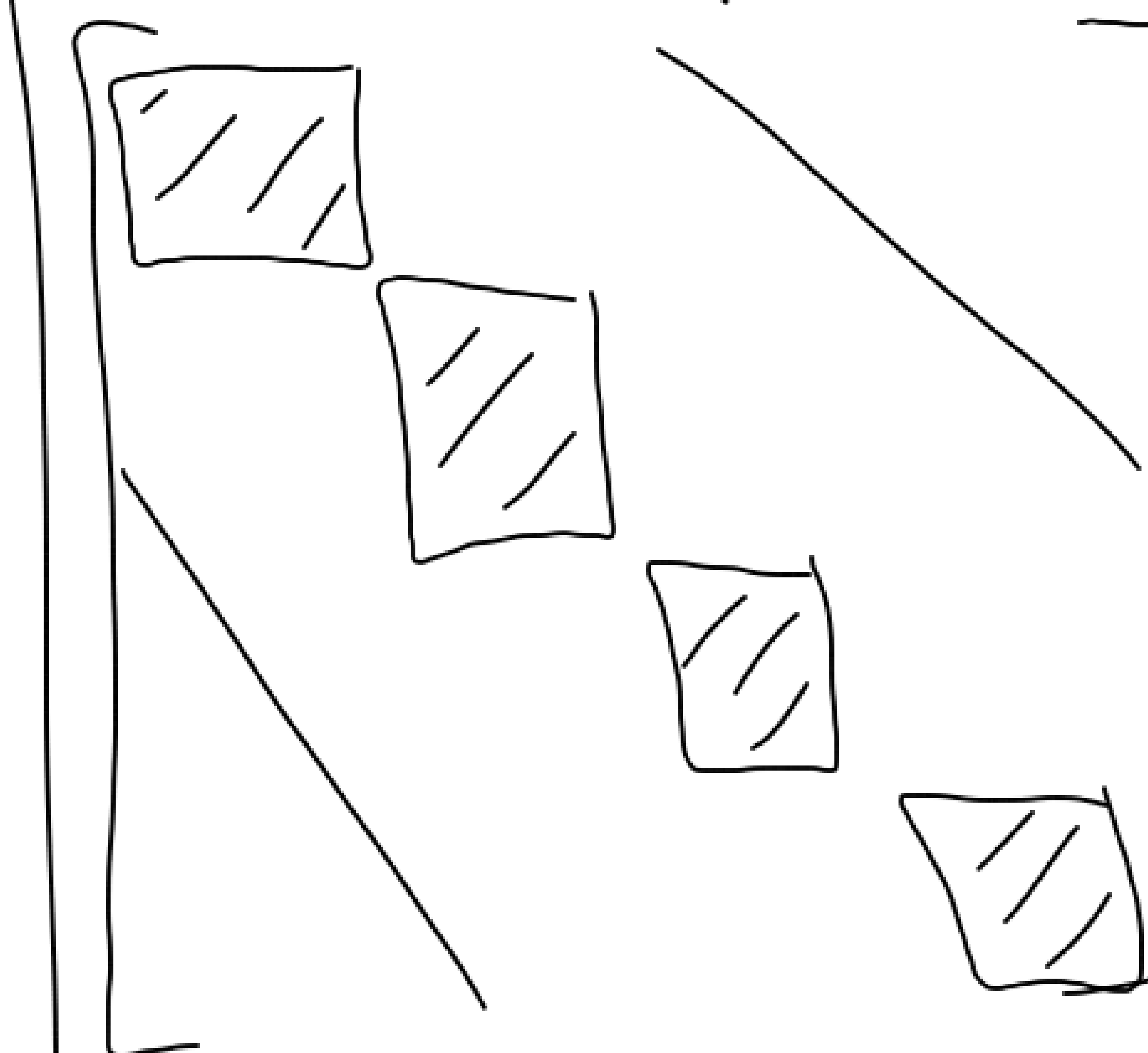
Implicit Trapezoidal method (Crank-Nicolson)

$$U^{n+1} = U^n + \frac{k}{2}\left[\nabla_h^2 U^n + \nabla_h^2 U^{n+1}\right]$$

$$\underbrace{\left[I - \frac{k}{2}\nabla_h^2\right]}_{A}\underbrace{U^{n+1}}_{x} = \underbrace{\left[I + \frac{k}{2}\nabla_h^2\right]U^n}_{b}$$

$A = I - \frac{k}{2}\nabla_h^2$

A is sparse



$\sim 5m^2$ non-zero entries

We have to solve a sparse $m^2 \times m^2$ linear system, $O(1/k)$ times.

Typically we would take

$k \sim h$

We could use LU factorization, and reuse it at each step.

Cost: $\mathcal{O}(m^4/k) = \mathcal{O}(m^5)$

Let's use an iterative method instead:

- Krylov subspace methods
- Multigrid methods

The efficiency of the methods depends on:

1. Initial guess
2. Condition # of A

$$K(A) = \|A\| \cdot \|A^{-1}\|$$

What is $K(A)$?

$$K(A) = \frac{\max|\lambda|}{\min|\lambda|} \quad \lambda \in \sigma(A)$$

Eigenvalues of A:

$$\lambda_{p,q} = 1 - \frac{k}{h^2}\left(\cos(p\pi h) + \cos(q\pi h) - 2\right)$$

$$p,q = 1, 2, \ldots, m$$

Smallest: $\lambda_{1,1} = 1 + 2\pi^2 k + \mathcal{O}(kh)$
$= 1 + \mathcal{O}(k)$

Largest: $\mathcal{O}\left(\frac{k}{h^2}\right) = \mathcal{O}\left(\frac{1}{h}\right) = \mathcal{O}(m)$

So $K(A) = \mathcal{O}(m)$

Total work: $\mathcal{O}(m \log m / k) = \mathcal{O}(m^2 \log m)$

Iterative methods will often converge in just 1-2 iterations.

Can just use U^n as initial guess.

Dimensional Splitting (LOD)

$$u_t = u_{xx} + u_{yy}$$

Instead, solve

$$u_t = u_{xx}$$

$$u_t = u_{yy}$$

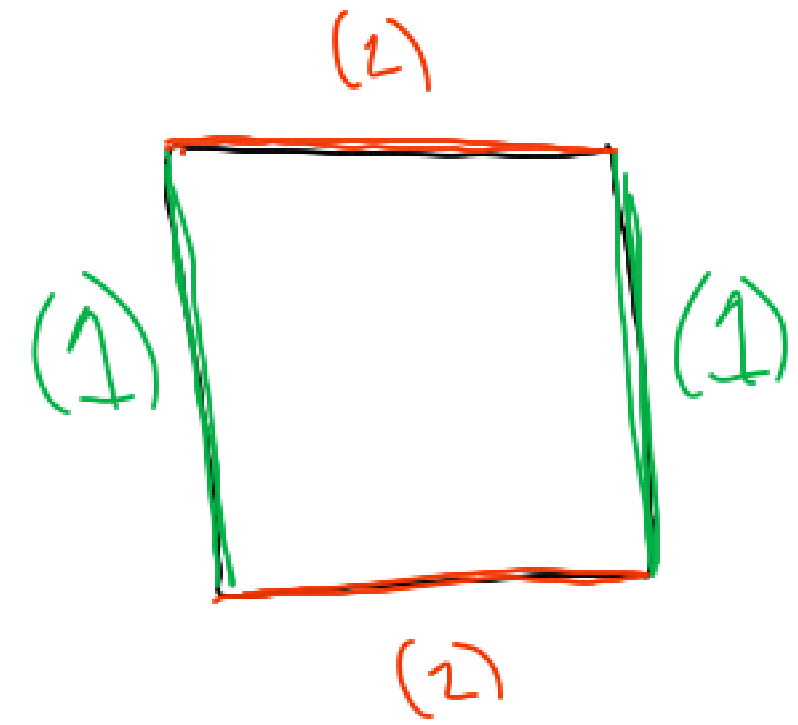
in alternation.

i.e.

$$\left(I - \frac{k}{2}D_x^2\right)U^* = \left(I + \frac{k}{2}D_x^2\right)U^n \qquad (1)$$

$$\left(I - \frac{k}{2}D_y^2\right)U^{n+1} = \left(I + \frac{k}{2}D_y^2\right)U^* \qquad (2)$$

Solve (1) for U^*, then solve (2) for U^{n+1}.



Cost Comparison:

Unsplit method: Solve one $m^2 \times m^2$ system per step

Total: $\mathcal{O}(m^5)$ or $\mathcal{O}(m^2 \log m)$

Split method: solve $2m$ systems of size $m \times m$ per step

Total cost: $\mathcal{O}(m^4)$ (direct)

or $\mathcal{O}(m^2 \log m)$

We need boundary conditions for U^* in (1) and (2).

In (2) we need BCs for U^* at top and bottom. We get the by evaluating $g(x, y=0, t=t_n)$ and $g(x, y=1, t=t_n)$ and solving (1) for those rows as well.

To get BCs for U^* in (1) at left and right, evaluate $g(x=0, y, t=t_{n+1})$, $g(x=1, y, t=t_{n+1})$ and solve (2) for U^* on those 2 columns.

Alternating-Direction-Implicit (ADI) method

$$U^* = U^n + \frac{k}{2}\left(D_y^2 U^n + D_x^2 U^*\right)$$

$$U^{n+1} = U^n + \frac{k}{2}\left(D_x^2 U^* + D_y^2 U^{n+1}\right)$$

Here $U^* \approx u(x, y, t_n + \frac{k}{2})$

So we can just evaluate the BCs at $t_n + \frac{k}{2}$.

ImEx Runge-Kutta:

$$u'(t) = f(u) + g(u)$$

$f(u)$: explicit RK method

$g(u)$: implicit RK method