look at iterative methods for solving Poisson's/Laplace's eq:

Iterative methods start with an initial guess and refine the guess at each step (by iteration), converging to the solution.

We look at 3 methods:

① Jacobi method
② Gauss-Seidel method
③ successive over-relaxation method.

Recall: general solution for Poisson's eq:

$$-4u_{i,j} + u_{i+1,j} + u_{i-1,j} + u_{i,j+1} + u_{i,j-1} - h^2 f_{i,j} = 0$$

where $i = 1, \ldots, M-1$
$j = 1, \ldots, N-1$

Rearrange: $$u_{i,j} = \frac{1}{4}\Big(\underbrace{u_{i+1,j}}_{\text{right}} + \underbrace{u_{i-1,j}}_{\text{left}} + \underbrace{u_{i,j+1}}_{\text{up}} + \underbrace{u_{i,j-1}}_{\text{bottom}}\Big) - \frac{h^2}{4} f_{i,j}$$

Jacobi Method:

iteration step

Iterative scheme: $$u_{i,j}^{(n+1)} = \frac{1}{4}\left(u_{i+1,j}^{(n)} + u_{i-1,j}^{(n)} + u_{i,j+1}^{(n)} + u_{i,j-1}^{(n)}\right) - \frac{h^2}{4} f_{i,j}$$

You iterate this scheme above. Eventually, $u_{i,j}$ stops changing, meaning that scheme has converged to solution (exact value).

Note:
Jacobi method always converges. (think about e-values of matrix form
↓
are < 1.

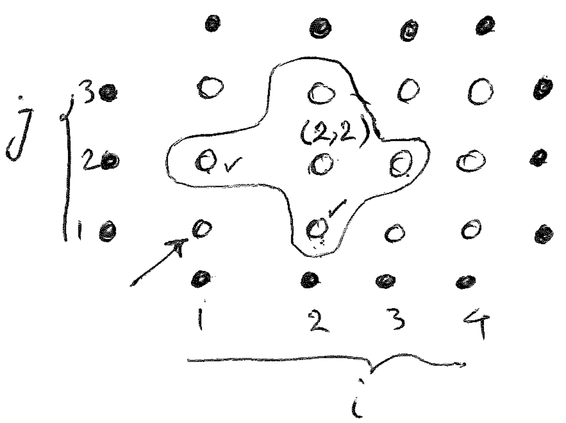
But, the convergence is slow.

Pseudo code for Jacobi Method:

```
for k = 1, ..., n_total
    for i = 1, ..., M-1
        for j = 1, ..., N-1
            update rule
        end
    end
    Copy new version of u_{i,j}^{(n+1)} into old version u_{i,j}^{(n)}
end
```

} loops over gird points (open circles)

Gauss - Seidel method: (G.S)



$$u_{2,2}^{(n+1)} = \frac{1}{4}\left(u_{3,2}^{(n)} + u_{1,2}^{(n+1)} + u_{2,3}^{(n)} + u_{2,1}^{(n+1)} \right) - \frac{h^2}{4} f_{2,2}$$

This method can double the rate Convergence, Compared to Jacobi method.

Successive over-relaxation (SOR) method.

The idea is to use the weighted-average of old value and new value for solution.

For example:

For example: (SOR):

$$u_{2,2}^{(n+1)} = (1-\omega)\, u_{2,2}^{(n)} + \frac{\omega}{4}\left(u_{3,2}^{(n)} + u_{1,2}^{(n+1)} + u_{2,3}^{(n)} + u_{2,1}^{(n+1)} - h^2 f_{2,2}\right)$$

For
- $\omega < 1$: under-relaxation ← slows down the convergence. X
- $\omega = 1$: G.S method.
- $\omega > 1$: over-relaxation ← speeds up the convergence. ✓

For simple problems we can find an optimal ω; e.g., $\omega = 1.8$.
You can experiment to find and optimal ω.

Note: For more details see Book Sauer.

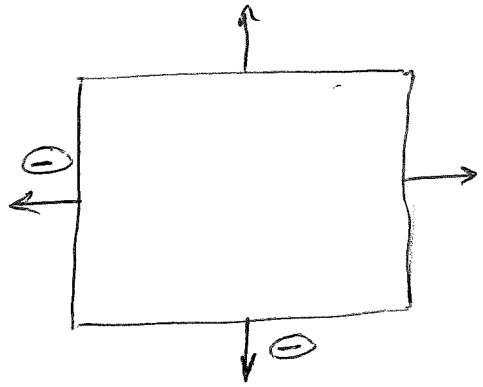
Final remark on BCs for Poisson equation:

If we were to solve $\nabla^2 u = f$ on the following grid, where Robin BCs are specified: as partial derivatives with respect to outward normal direction vector $\underline{n}$.

$$\beta_3(x) = \frac{\partial u}{\partial y}$$

$\beta_1(y) =$

$$\nabla u \cdot \underline{n} = \left(\frac{\partial u}{\partial x}, \frac{\partial u}{\partial y}\right)(-1, 0) = -\frac{\partial u}{\partial x}$$



$\beta_2(y) = \nabla u \cdot \underline{n} =$

$$= \left(\frac{\partial u}{\partial x}, \frac{\partial u}{\partial y}\right)(1, 0) = \frac{\partial u}{\partial x}$$

$$\beta_4(x) = \nabla u \cdot \underline{n} = \left(\frac{\partial u}{\partial x}, \frac{\partial u}{\partial y}\right)(0, -1) = -\frac{\partial u}{\partial y}$$

~~BC~~ BCs are directional derivatives vertical to the boundries of the grid, showing instantaneous rate of change of solution u along the boundaries.

You take first derivative using three-point approximation and apply to BCs:

Three-point approximation $f'(x) \approx \frac{-3f(x) + 4f(x+h) - f(x+2h)}{2h} + O(h^2)$

$$\beta_4(x) = \frac{-3u_{i,0} + 4u_{i,1} - u_{i,2}}{2\Delta y}$$

$\vdots$

$\beta_1(x)$

See Book for obtaining BCs.