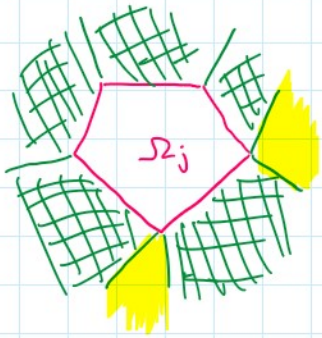
Over the j-th cell Ω_j, we consider the weak form

$$M^j \partial_t U^j - (S^j)^T F^j = - \Phi^j F^* \Big|_{\partial\Omega_j}$$

The flux F^* depends on U^i for any neighboring cell Ω_i

Generally, in dimension D, the numerical flux is the sum of numerical fluxes associated with the $D-1$ dimensional faces of the cell Ω_j.



This is still a semi-discrete formulation.

What does the time-discretization look like?

We generate sequences $U_0^j, U_1^j, U_2^j, \ldots, U_n^j, \ldots$ indexed by the time step n.

The time difference between each time step is $k > 0$, as before.

1) What is the accuracy, that is, the local discretization error?
2) Stability: does the discretization introduce oscillations?
3) Do conserved quantities stay conserved?

Example: **Explicit Euler** (Forward time)

$$M^j \frac{U^j_{n+1} - U^j_n}{k} - (S^j)^T F^j_n = - \Phi^j F^*_n \Big|_{\partial\Omega_j}$$

Isolate U^j_{n+1}:

$$U^j_{n+1} = U^j_n + k (M^j)^{-1} \left[(S^j)^T F^j_n - \Phi^j F^*_n \Big|_{\partial\Omega_j} \right]$$

We need to invert a local matrix at each step.

The local truncation error is $O(k^2)$ but this method suffers from numerical instability for large time steps.

Example: **Implicit Euler**

$$M^j \frac{U^j_{n+1} - U^j_n}{k} - (S^j)^T F^j_{n+1} = - \Phi^j F^*_{n+1} \Big|_{\partial\Omega_j}$$

Fixpoint formulation:

$$U^j_{n+1} = U^j_n + k (M^j)^{-1} \left[(S^j)^T F^j_{n+1} - \Phi^j F^*_{n+1} \Big|_{\partial\Omega_j} \right]$$

At each time step, we need to solve the global system above, which is a fixpoint equation.

The local truncation error is $O(k^2)$. It is stable even for large time steps. Main disadvantage: computational costs.

Example: **Crank-Nicolson method**

$$M^j \frac{U^j_{n+1} - U^j_n}{k} - \frac{1}{2}(S^j)^T F^j_{n+1} - \frac{1}{2}(S^j)^T F^j_n$$

$$= -\frac{1}{2}\Phi^j F^*_{n+1} - \frac{1}{2}\Phi^j F^*_n \Big|_{\partial\Omega_j}$$

Similar properties as implicit Euler but often with energy conservation.

Example: **θ - method** $\quad 0 \le \theta \le 1$

$$M^j \frac{U^j_{n+1} - U^j_n}{k} - \theta (S^j)^T F^j_{n+1} - (1-\theta)(S^j)^T F^j_n$$

$$= -\theta \Phi^j F^*_{n+1} - (1-\theta)\Phi^j F^*_n \Big|_{\partial\Omega_j}$$

Special case includes Crank-Nicolson $\theta = \frac{1}{2}$

A few words about parallel computing.

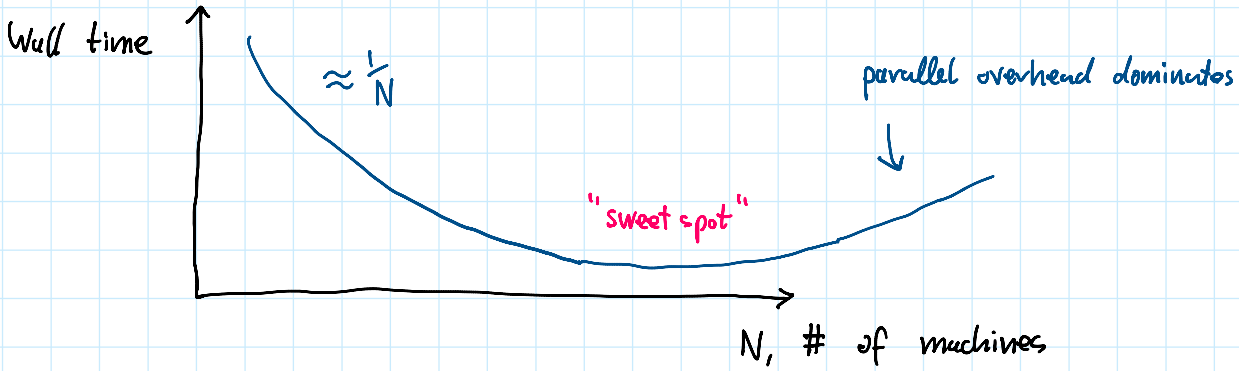
In a parallel computing environment, we have N computers that can send messages to each other

Typically sending messages requires more wall-clock time than the local computations on each machine. Hence, in a computational task that is distributed over several machines:

1) we want to keep the number of recipients small
2) " " " " the number of messages small
3) " " " " the messages short

If that can be ensured, we often have a speed up in comparison to a serial implementation.

Ideally, the computing is anti-proportional to the number of machines.



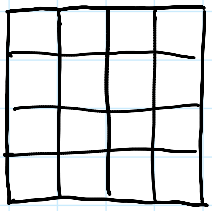
Application in numerical PDE:

We distribute the mesh over several computers.

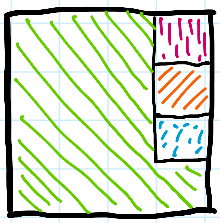
At each time step, the evolution of a local cell requires information from neighboring cells, which might be managed by other computers

We want to keep the parallel overhead low
and the local computation load balanced

Example: 4×4 grid, 4 machines



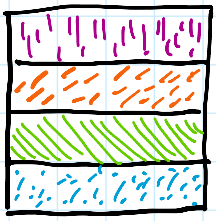
How to distribute these 16 cells to 4 machines? (4 regions)



Bad choice: very unbalanced load.

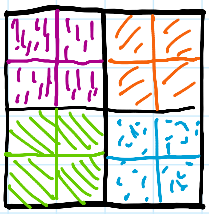
(The 4 regions should have the same volume)

So so choice:



So-so choice:

balanced load
but 12 interfaces between regions



Good choice:

balanced load,
8 interfaces between regions

Typically, the distribution is heuristically optimized as a preprocessing step before the actual computation begins.

Example heuristic: minimize interface surface while keeping the volumes comparable.

If local mesh refinement is performed, then the load may become imbalanced over time, and re-distribution becomes necessary.