

## Parallel Numerics

### Exercise 10: Solution of a two-dimensional Poisson equation

Modelling of numerous physical problems leads to a Poisson equation which can be written in two spatial dimensions as

$$\frac{\partial^2 p}{\partial x^2} + \frac{\partial^2 p}{\partial y^2} = d$$

on the domain  $\Omega = [0, 1] \times [0, 1]$ , where  $p(x, y)$  is the unknown function and  $d(x, y)$  is some function; the physical problem also requires that certain boundary conditions are satisfied; let these be  $p = 0$  at the boundaries in our case.

For a numerical solution the domain is subdivided into  $i_{max} \times j_{max}$  grid cells of equal size  $\Delta x \times \Delta y$ . For the formulation of boundary conditions one layer of grid cells is attached outside the domain in each direction (see figure 1). The unknown

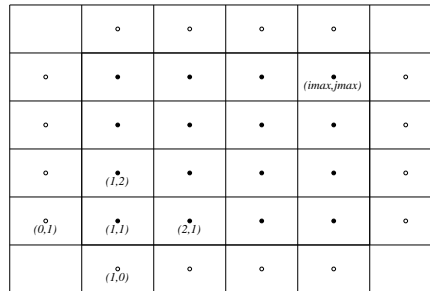


Figure 1: Spatial discretisation and numbering of nodes

function  $p(x, y)$  as well as  $d(x, y)$  are now computed resp. evaluated at discrete points in space which could be the centres of the grid cells. The discretisation of the Laplace operator can be done in an obvious way applying 5-point-stencils:

$$\frac{p_{i+1,j} - 2p_{i,j} + p_{i-1,j}}{\Delta x^2} + \frac{p_{i,j+1} - 2p_{i,j} + p_{i,j-1}}{\Delta y^2} = d_{i,j}$$

Together with the equations for the boundary conditions

$$p_{0,j} = -p_{1,j} \quad \text{and} \quad p_{i_{max}+1,j} = -p_{i_{max},j} \quad \text{for } j = 1, \dots, j_{max}$$

$$p_{i,0} = -p_{i,1} \quad \text{and} \quad p_{i,j_{max}+1} = -p_{i,j_{max}} \quad \text{for } i = 1, \dots, i_{max}$$

we get a system of linear algebraic equations where the number of unknowns is equal to the number of equations. The solution can be computed e.g. by any method of the class of the splitting methods (Jacobi, Gauss-Seidel, SOR). →

Write a serial program that calculates the approximate discrete solution of a poisson equation. For simplicity we restrict ourselves to a Laplace equation, i.e.  $d(x, y) = 0$  everywhere in the domain.

The program is now parallelised by decomposing the domain into equal subdomains. Each process is assigned to a subdomain and calculates now the solution for  $i_{max} \times j_{max}$  points on a grid, which consists of the corresponding grid cells of this subdomain and one additional cell layer in each direction as described for the serial algorithm (see figure 2). If the additional cell layer is part of the

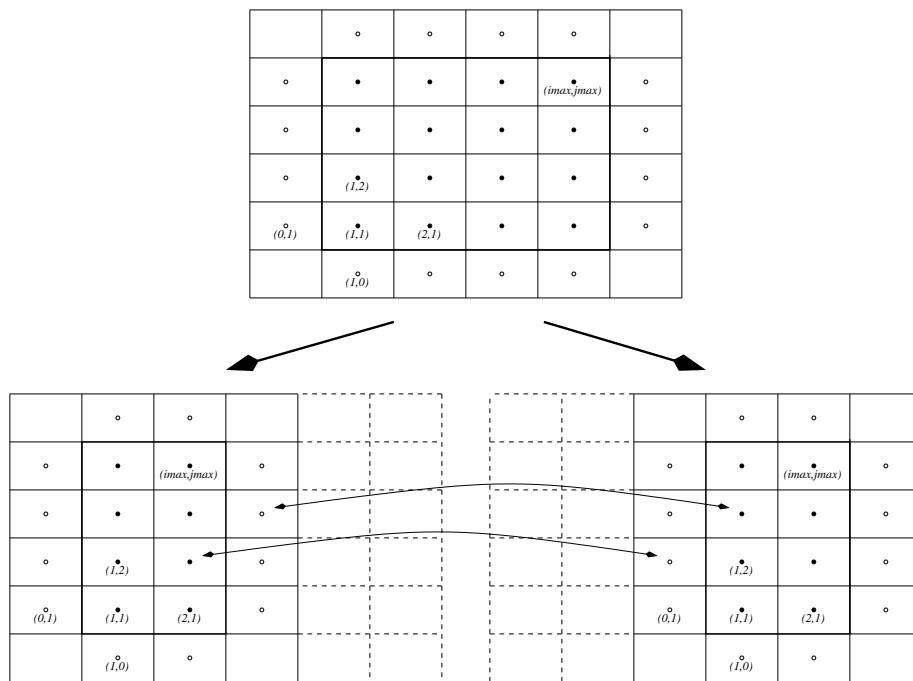


Figure 2: Scheme of the spatial decomposition with overlapping subdomains

(global) computational domain, it is used for the exchange of data between two adjacent subdomains. If it does not lie in the (global) domain, it is used for the formulation of the boundary conditions.

While iterative equation solvers are applied, the transfer of information between the subdomains is done before each iteration. It is achieved by copying the values at the outermost layer within each subdomain to the additional cell layer of the corresponding neighbour (see figure 2).

Implement the parallel algorithm in a way that there is no master process, i.e. the information of the local subdomains is only available to the corresponding process. After the calculation is done each process writes its results to the hard disc. It is convenient to use the MPI-functions related to virtual topologies here.