Prof. Dr. Peter Bastian, Linus Seelinger
IWR, Universität Heidelberg

Submission date: 6 June 2019

---

Note: Do not forget to update your dune installation as described in exercise sheet 2.

**Exercise 1** *Parallel overlapping Schwarz methods*

In this exercise we are going to experiment for the first time with an implementation of a parallel solver. We will implement and compare three variants of the overlapping Schwarz methods in total which work both in two and three dimensions.

The problem for the Schwarz methods is

$$\begin{aligned} -\Delta u(x) &= 0 && \text{in } \Omega = (0,1)^d, \\ u(x) &= \exp(-\|x\|_2^2) && \text{on } \partial\Omega. \end{aligned}$$

The file `additive_schwarz.cc` in the directory `uebungen/uebung04` provides a working implementation of the additive Schwarz method (ASM). Have a look at the functionality of the program, in particular

- `Dune::PDELab::OverlappingOperator`
- `Dune::PDELab::istl::ParallelHelper`
- `Dune::PDELab::OverlappingScalarProduct`
- `Dune::PDELab::SuperLUSubdomainSolver`

and do some test calculations. The sequential version of the program can be run with the command

```
./additive_schwarz
```

as you are used to. In order to use <p> processors of the computer, you simply have to add `mpirun -np <p>` before the run-command, i.e.

```
mpirun -np <p> ./additive_schwarz
```

In both cases the program creates an output file which can be visualized with ParaView. For the sequential case the file has the name `additive_schwarz.vtu`, for the parallel case the name has the structure `s00XX-additive_schwarz.pvtu` where `XX` is a placeholder for the number of processors used in the computation.

In the lecture you have also learned the multiplicative Schwarz method (MSM). In order to parallelize this algorithm, one has to color the subdomains. Figure 1 shows a collection of subdomains whose corrections can be calculated simultaneously. This method is called `coloring`. It is possible to extend this idea also to three dimensions.
This exercises only uses `Yasp-Grid` which numbers the subdomains in the parallel computation lexicographically. This ordering of the subdomains can be exploited in the coloring.

**Task 1** The implementation of the class *MultiplicativeSchwarzPreconditioner* can be found in the file `multiplicative_schwarz_preconditioner.hh`. Currently, the sequential version of the preconditioner involving `p` processors is implemented. Your task is to supplement the coloring in the preconditioner.

**Task 2** It is possible to modify this preconditioner to obtain a symmetric preconditioner. This symmetrization is referred to as the Symmetric Multiplicative Schwarz Method (SMSM).
Implement the symmetric preconditioner in the class *SymmetricMultiplicativeSchwarzPreconditioner*.

The main task of this exercise consists of comparing the different variants. Test the program for various mesh sizes, with overlap 1 or 2, in two and three dimensions with the number of processors $\in \{1, 4, 16, 64\}$.

**Task 3** Present the number of iterations for various combinations in form of a table.

Example for two dimensions, overlap 1, number of processors equal to 1:

| | ASM | Seq. MSM | Col. MSM | SMSM (col.) |
|---|---|---|---|---|
| mesh size | IT | IT | IT | IT |
| 1/32 | | | | |
| 1/64 | | | | |
| 1/128 | | | | |
| 1/256 | | | | |
| 1/512 | | | | |

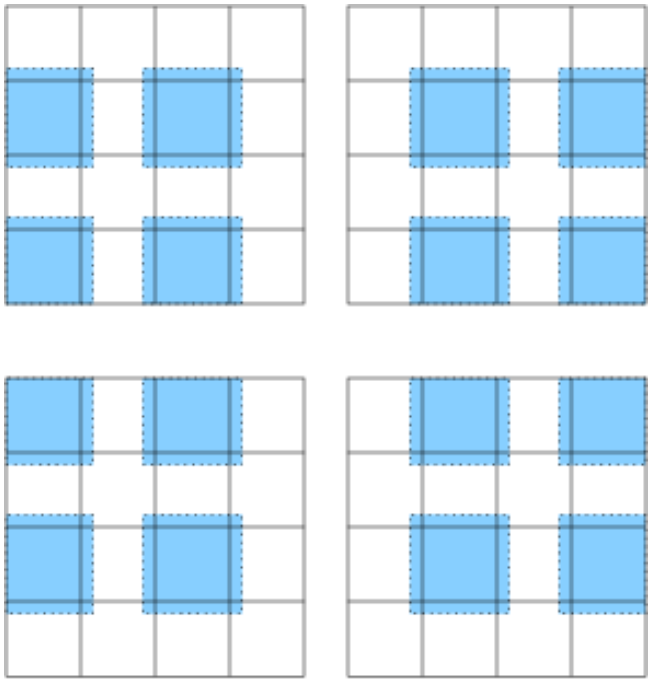

Figure 1: Simultaneously treated corrections in multiplicative Schwarz

**( 15 Points )**