Matlab Programming Session for the DD25 Summer School

Martin J. Gander Laurence Halpern Felix Kwok Ian May

July 22, 2018

Instructions

We use as a model problem the heating of a room which was also shown in one of the lectures. For simplicity, we use a square room, as shown in Figure 1, and an adimensionalized temperature which equals a step function g on the left edge Γ, which assumes values $g = 0.3$ if $0 \leq x \leq 0.5$ or $0.9 \leq x \leq 1$, modeling a warm wall, and $g = 1$ if $0.5 < x < 0.9$, modeling an even warmer door. On the rest of the boundary the temperature is fixed to zero, modeling cold walls. Inside the room there is a radiator whose temperature is equal to 50. The radiator is modeled by a source term $f(x, y) = 50$ if $(x, y) \in [0.4, 0.6] \times [0.4, 0.6]$ and zero otherwise. Our goal is to find the temperature distribution inside the room, described by the steady state heat equation, i.e. the Poisson equation,

$$\begin{aligned} -\Delta u &= f, \quad \text{in } \Omega, \\ u &= g, \quad \text{on } \Gamma, \\ u &= 0, \quad \text{on } \partial\Omega \setminus \Gamma. \end{aligned}$$

We will use Schwarz, Dirichlet-Neumann and Neumann Neumann methods to compute this temperature, and compare the result to the result of a direct solver.

To help you get started, we provide you with Matlab scripts which you can first run, and then modify to experiment with the different parameters of the methods. The Matlab codes are available at

`http://www.unige.ch/~gander/MatlabFiles.zip`

Problem 1: The Parallel Schwarz Method

1. ☐ Run the script `problem1.m` and see first the global solution obtained by the direct solver, followed by the approximations of the parallel Schwarz method as they converge to the global solution. Simply press any key to go from one iteration to the next.

2. ☐ Run the script with different sizes of the overlap `d` and positions of the interface `a`. What is the influence of the overlap on the convergence? You can disable the plots by commenting the two commands `pause` in the script `problem1.m`.

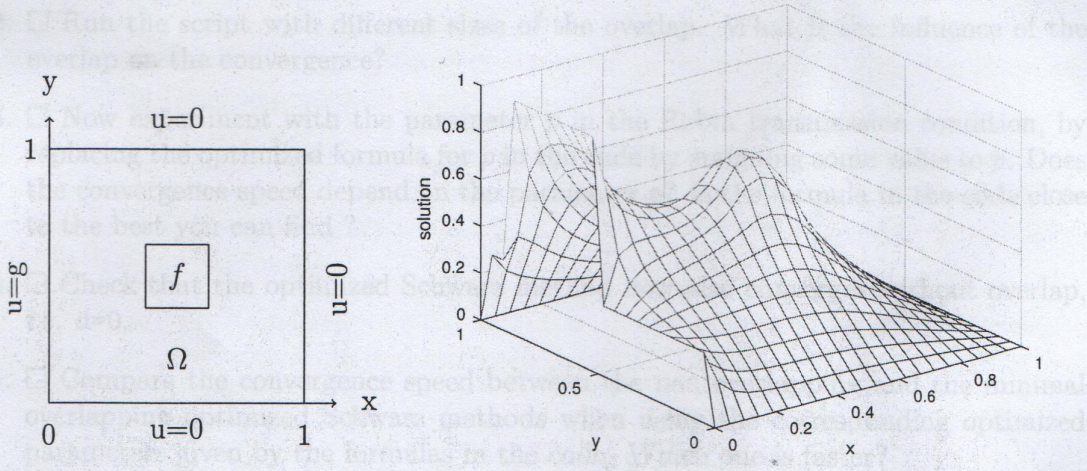

Figure 1: Geometric configuration of the room with the heater placed in the center and a source modeled by a warm wall and even warmer door on the left, and corresponding solution.

3. ☐ Change the script `problem1.m` to obtain the alternating Schwarz method. How are the iterates of the alternating Schwarz method related to the iterates of the parallel Schwarz method ?

4. ☐ Nowadays new buildings have underfloor heating systems due to their increased efficiency compared to classical radiators. To simulate an underfloor heating systems we enlarge the support of the source term f. In `RoomData.m` modify the line

```
f([yi>0.4 & yi<0.6],[xi>0.4 & xi<0.6])=50;
```

to

```
f([yi>0.1 & yi<0.9],[xi>0.1 & xi<0.9])=const,
```

where `const` is some number. Find the approximate value of `const` such that the room temperature is similar to the previous case. Do we need hotter or colder water if we use an underfloor heating system?

5. ☐ (longer) Experiment also with different room dimensions and wall temperatures, and by adding more doors or windows to the room. Can you model approximately the room you are living in ?

6. ☐ (longer) Try to add more subdomains in the x-direction. What do you observe when iterating with more and more subdomains ? (Hint: start by adding first just a third subdomain)

Problem 2: The Optimized Schwarz Method

1. ☐ Run the script `problem2.m` and observe the behavior of the iterations of the optimized Schwarz method as they converge to the global solution. What do you observe comparing with the classical Schwarz method ?

2. ☐ Run the script with different sizes of the overlap. What is the influence of the overlap on the convergence?

3. ☐ Now experiment with the parameter p in the Robin transmission condition, by replacing the optimized formula for p in the code by assigning some value to p. Does the convergence speed depend on the parameter p? Is the formula in the code close to the best you can find ?

4. ☐ Check that the optimized Schwarz method now also converges without overlap, i.e. `d=0`.

5. ☐ Compare the convergence speed between the non overlapping and the minimal overlapping optimized Schwarz methods when using the corresponding optimized parameters given by the formulas in the code. Which one is faster?

6. ☐ (longer) Try to add more subdomains in the x-direction. What do you observe when iterating with more and more subdomains ? (Hint: start by adding first just a third subdomain)

Problem 3: The Dirichlet-Neumann Method

1. ☐ Run the script `problem3.m` and observe the behavior of the iterations of the Dirichet-Neumann method as they converge to the global solution.

2. ☐ Experiment with the position `a` of the interface. Can you make the method diverge ?

3. ☐ Vary the relaxation parameter θ and find heuristically an optimal value. How does this compare to the optimal choice given in the lectures ? Can you make the Dirichlet-Neumann method always converge ?

4. ☐ Can you make the method into a direct solver using that you learned in the lectures this is possible for a perfectly symmetric situation with relaxation parameter $\theta = \frac{1}{2}$? Hint: this needs some thinking for the mesh you are currently using.

5. ☐ (longer) Modify the script `problem3.m` in order to simulate an insulated wall on the right edge of the domain. Recall that an insulated wall can be modeled through a homogeneous Neumann condition such as $\partial_n u = 0$. Compare graphically the solution with the one obtained by fixing the temperature to zero on the right edge.

6. ☐ (longer) Try to add a third and a fourth subdomain. What possibilities do you have in choosing the Dirichlet and Neumann conditions for each subdomain ?

Problem 4: The Neumann-Neumann Method

1. ☐ Run the script `problem4.m` and observe the behavior of the iterations of the Neumann-Neumann method as they converge to the global solution.

2. ☐ Experiment with the position `a` of the interface. Can you make the method diverge ?

3. ☐ Vary the relaxation parameter θ and find heuristically an optimal value. How does this compare to the optimal choice given in the lectures ? Can you make the Neumann-Neumann method always converge ?

4. ☐ Can you make the method into a direct solver using that you learned in the lectures this is possible for a perfectly symmetric situation with relaxation parameter $\theta = \frac{1}{4}$? As for the Dirichlet-Neumann case, this needs some thinking for the mesh you are currently using.

5. ☐ (longer) Try to add more subdomains in the x-direction. What do you observe when iterating with more and more subdomains ? (Hint: start by adding first just a third subdomain)

Problem 5: Krylov Acceleration

1. ☐ Run the script `problem5.m` and compare the convergence behavior of the parallel Schwarz method with its Krylov accelerated version.

2. ☐ Experiment with different values of the overlap. Is the Krylov accelerated version always faster ? Is it possible to make them close ?

3. ☐ (longer) Implement in an analogous fashion also the Dirichlet-Neumann and Neumann-Neumann methods in order to accelerate them using a Krylov method. Experiment with different relaxation parameters. What do you observe ?

Problem 6: Coarse Space Correction

1. ☐ Run the script `problem6.m`, which solves a one dimensional Laplace problem on the interval (a, b) with $a = 0$ and $b = 1$ and Dirichlet boundary conditions (so the solution is a straight line), using a parallel Schwarz method on `I=4` subdomains, first without coarse grid, and then with a multigrid type coarse grid, and finally an optimal coarse grid. Watch how the iterates converge to the solution.

2. ☐ Increase the number of subdomains to `I=8` then `I=16` without changing the mesh size. What do you observe ? Hint: use the figure command to open a new figure for each case, so you can compare the results on the screen. Which method is scalable ?

3. ☐ Now refine the mesh size proportionally to the increase in the number of subdomains by increasing the number of mesh points `J`. What do you observe ? What happened to the overlap in this experiment, compared to the previous one ? Which method is scalable ?

4. ☐ Now modify the parameter $\eta = 0$ to $\eta = 5$, so that the method now solves the problem $(\eta - \Delta)u = 0$. Why is the present optimal coarse space in the code not optimal any more ?

5. ☐ (longer) What happens if instead of keeping the domain (a, b) fixed, you let it grow proportionally when you add subdomains ? Experiment with both $\eta = 0$ and $\eta = 5$. Do you observe scalability even for the one level method now ?