Division of Scientific Computing, Department of Information Technology, Uppsala University

Numerical Linear Algebra
Feb - March, 2021

Hands-on: Preconditioning for saddle point problems. Testing geometric and algebraic Multigrid methods

The necessary data files and the Matlab routines are in the shared `dropbox` directory `NLA_Course_2021`.
You are expected to send Matlab-created reports of the runs, possibly with your comments.

Exercise 1 (Indefinite matrices)
The files `Stokes_k.mat`, $k = 4, 5, 6, 7$ contain indefinite matrices, arising from discrete Stokes problem in 2D. The system to be solved is $\mathcal{A}\boldsymbol{w} = rhs$, where $\mathcal{A}$ has a special block structure:

$$\mathcal{A} = \begin{bmatrix} A & 0 & B_1 \\ 0 & A & B_2 \\ B_1^T & B_2^T & 0 \end{bmatrix} \quad \text{or} \quad \mathcal{A} = \begin{bmatrix} \boldsymbol{A} & \boldsymbol{B} \\ \boldsymbol{B}^T & 0 \end{bmatrix},$$

where $\boldsymbol{A} = \begin{bmatrix} A & 0 \\ 0 & A \end{bmatrix}$ and $\boldsymbol{B} = \begin{bmatrix} B_1 \\ B_2 \end{bmatrix}$. The standard method to solve indefinite systems is `MINRES`. It can be used in its unpreconditioned form or preconditioned by some matrix P, that can be given also in a factorized form $P = P1 * P2$:

```
[x,flag,relres,iter,resvec] = minres(A,rhs,tol,maxit)
[x,flag,relres,iter,resvec] = minres(A,rhs,tol,maxit,P)
[x,flag,relres,iter,resvec] = minres(A,rhs,tol,maxit,P1,P2)
```

- Is the matrix $\mathcal{A}$ nonsingular?
- Solve the systems with unpreconditioned `MINRES`. Monitor the iteration counts and the behaviour of the convergence when the system size grows.
- How to precondition the system? Note that the blocks A are Laplacians (spd). Further, the matrix M, included in the .mat files is the pressure mass matrix and it is shown that it is a very good approximation of the negative Schur complement S of $\mathcal{A}$, $S = \boldsymbol{B}^T\boldsymbol{A}^{-1}\boldsymbol{B}$. Note that S is spd.

  As `MINRES` requires to use a preconditioner to $\mathcal{A}$ which is positive definite, a good option is to use

$$P_D = \begin{bmatrix} \widetilde{\boldsymbol{A}} & 0 \\ 0 & \widetilde{M} \end{bmatrix},$$

where $\widetilde{\boldsymbol{A}}$ and $\widetilde{M}$ may be some approximations of $\boldsymbol{A}$ and M, respectively.

One very popular approximation of M is the so-called *lumped* mass matrix $\widehat{M}$, which is diagonal and the diagonal entries are the row sums of M.

How to handle the block $\boldsymbol{A}$? Will some incomplete factorization of K be a good idea? Or use an inner solver?

Tasks:

– Solve $\mathcal{A}$ by `MINRES` and use P_D as a preconditioner.

– Solve $\mathcal{A}$ by `GMRES` and use the block lower-triangular preconditioner

$$P_D = \begin{bmatrix} \widetilde{\boldsymbol{A}} & 0 \\ \boldsymbol{B}^T & -\widetilde{M} \end{bmatrix},$$

<u>Exercise</u> 2 (Testing Multigrid methods)
We consider as a model problem the two-dimensional convection-diffusion-reaction (CDR) equation,

$$-\frac{\partial}{\partial x}\left(\varepsilon_1 \frac{\partial u}{\partial x}\right) - \frac{\partial}{\partial y}\left(\varepsilon_2 \frac{\partial u}{\partial y}\right) + b_1 \frac{\partial u}{\partial x} + b_2 \frac{\partial u}{\partial y} + cu = f \tag{1}$$

with some suitable boundary conditions.
The domain of definition is the unit square $\Omega = [0,1]^2$, discretized by some triangular mesh. The problem is discretized using standard linear basis functions.

Problem (1) has three problem parameters: the diffusion coefficients $\varepsilon = [\varepsilon_1, \varepsilon_2]$, the convection vector field $\boldsymbol{b} = [b_1, b_2]$ and the reaction coefficient c.
By giving various values of the problem coefficients, we can simulate a series of classical problems of elliptic type:

(P1) Poisson's problem: $\varepsilon = [1,1]$, $\boldsymbol{b} = [0,0]$, $c = 0$.

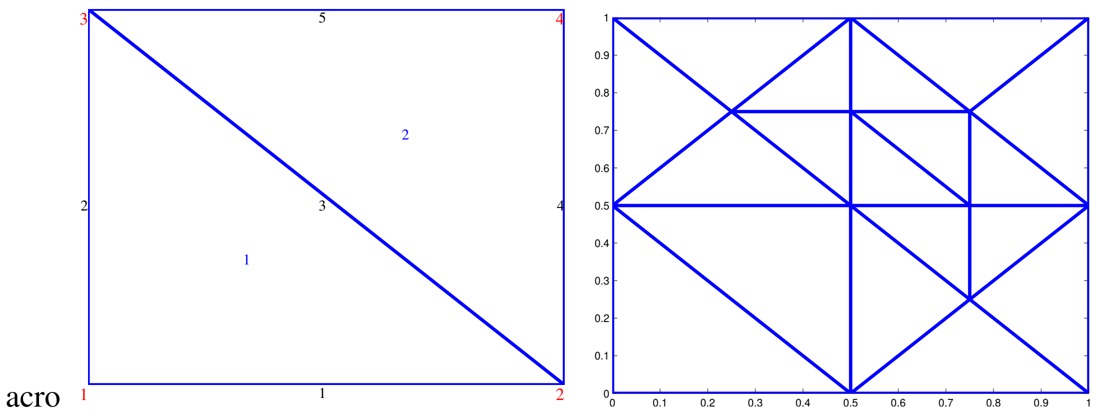
(P2) Anisotropic problem: $\varepsilon = [1, \varepsilon_2]$ or $\varepsilon = [\varepsilon_1, 1]$ with $\varepsilon_i \ll 1$, $\boldsymbol{b} = [0,0]$, $c = 0$.

(P3) Poisson's problem with variable (jumping) coefficients: let $\varepsilon_i, i = 1,2$ be functions of space or constants with different values in different parts of Ω, $\boldsymbol{b} = [0,0]$, $c = 0$.

(P4) Singularly perturbed convection-diffusion problem: $\varepsilon_1 = \varepsilon_2 \ll 1$, $\boldsymbol{b}$ nonzero.

(P5) Parabolic-type problems: $c > 0$. The coefficient c can be related to the inverse of the time-step when discretizing a parabolic problem with FEM in space and with some implicit time-stepping method.

(P_∞) Any other choice of problem parameters could also be of interest, however we limit ourselves to the cases (P3) with constant jump, P(2) and (P4).



(a) (b)

Figure 1:

The resulting system matrix is of the form $A = L(\varepsilon) + C(\boldsymbol{b}) + cM$, where L, C and M are the discrete divergence, convection and mass matrices, correspondingly.

Software tools and adjustments:

1. Download the `.mat` data files.
2. Download `MGDemmel.tar` and `hsl_mi20.tar` from `dropbox`. When uncompressed, they create subdirectories.
3. Download the file `Main_CDR_load.m` and correct the paths to your MG and mi20 directories.

Note: If you want to have a very efficient AMG solver, visit `http://agmg.eu/` and apply for a personal academic license for AGMG, in order to download the software. The particular AMG implementation provides Matlab interface.
The main routine to use is `Main_CDR_load.m`. You can edit it and load matrices for the different cases and problem sizes. The matrices for Laplace*, Anisot*, ConvDiff* correspond to a predefined mesh, shown in Figure 1(a). There are some extra matrices in the files Discont* correspond to a predefined mesh, shown in Figure 1(b).

The major task is to see the robustness and scalability of multilevel methods of Multigrid type. Robustness is understood as (near) independence of problem parameters and problem size. These methods are shown to have computational cost per iteration, linear in the number of degrees of freedom and (nearly) optimal convergence rate, i.e., the number of iterations stay (nearly) constant with the problem size (for a given set of problem parameters). The word 'near' reflects that when one uses a V-cycle, as in the provided solvers, a slight growth in the iteration counts might be observed when the problem size is increased.
The above study can be made by performing the following subtasks.

(T1) (This task must be done.)

For Problem (P1), compare the performance of a geometric Multigrid method (MG), provided by James Demmel and AMG method from the HSL library and the direct solver in `Matlab` ('backslash').

To this end, run `Main_CDR_load.m` loading the files `Laplace*.mat` for several problem sizes.

Have a look at the MG code (`makemgdemo_time,testfmgv_time`). As you can see, there is no matrix generation. The matrix corresponds to central differences and is 'hard-coded' in the code.

The FEM matrix differs from the finite difference one, but has the same condition number. The MG code uses a discontinuous solution and computes the corresponding right-hand-side by multiplying it with the matrix. For the FEM case, a very similar solution is generated, to make the comparisons fair. Due to the fact that the FEM code uses general triangulations and rather arbitrary ordering of the nodes, one-to-one comparisons are not straightforward. The solution can be plotted as follows:
`plot3(Node(1,:),Node(2,:),x,'*').`

(T2) Check the suitability of `mi20` to solve strongly anisotropic problems. Load the matrices `Anisot*.mat`. These correspond to $\varepsilon = [1, 10^{-3}]$, $\boldsymbol{b} = [0, 0]$, $c = 0$. How do the iteration counts behave? Check the spectrum of the original matrix and you will see that there is a whole cluster of very small eigenvalues, and the unpreconditioned CG will have a slow convergence rate.

(T3) Check the suitability of `agmg` for solving singularly perturbed convection-diffusion problems.

Load the matrices `ConvDiff*.mat`. These correspond to $\varepsilon_1 = \varepsilon_2 = 10^{-3}$, `flag=1`, $c = 0$. The variable `flag` determines which vector field to be used. The one suggested is of the form, shown in Figure 2.

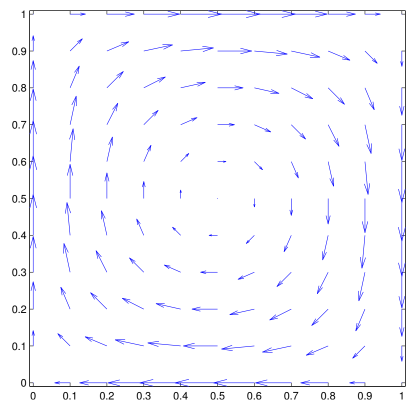


Figure 2:

Remark: Testing solvers for singularly perturbed convection-diffusion problems, where typically layers occur in the solution, is a difficult task as it requires suitable discretizations and adaptive meshes to relate h with ε in order to resolve those layers. Here, the boundary conditions are Dirichlet all over and, therefore, no layers are present.

Disclaimer:
1. I have not checked the convergence of mi20 for all the cases.
2. The program `Main_CDR_load.m` is ready only to use the Laplace matrices and one has to amend it to run for the other cases.
Maya Neytcheva