TW776: Assignment 6 For Thursday, 12 May

Assignment Guidelines

- Write a short report to illustrate your work. Use full sentences and include code snippets where applicable.
- Reports are to be handed in during the lecture on the due date.
- Feel free to discuss the work with your classmates, but write your own computer codes, generate your own figures, devise your own proofs and examples, and write your own report.
- It is acceptable to consult the library or internet on any problem. But if you do, it is important that you rewrite the material in your own words and cite your source.
- Unless told otherwise, use built-in functions in MATLAB or Python as far as possible. Do not write your own. This is not a course in programming.

Conjugate Gradients

(a) Write your own implementation of conjugate gradients (CG), based on the algorithm on p. 196 in Ascher & Greif, or p. 49 in Shewchuck. Use your code to solve the $N^2 \times N^2$ model problem with a random right-hand side $\boldsymbol{b}$, for a smallish value of N, say $N = 4$. In parts (b)–(d) you should now check that your code works correctly.

(b) Compute the "exact" solution $\boldsymbol{x}$ using a direct method (built-in). Then compute the errors $\|\boldsymbol{x} - \boldsymbol{x}_k\|_2$ at each step of the iteration, and check that convergence takes place.

(c) On p. 200 of Ascher & Greif, and on p. 34 of Shewchuck, it is stated that if A has only m distinct eigenvalues, then CG converges in m steps. Figure out how many distinct eigenvalues the model matrix A has (recall formula p. 183 in Ascher & Greif). Check that your CG algorithm converges in the predicted number of steps.

(d) Both MATLAB and Python have built-in CG algorithms. (In MATLAB, use `pcg` with no preconditioner. In Python, use `scipy.sparse.linalg.cg`) Use the one of your choice, and check that it produces at each step of the iteration the same results as your own CG code.

(e) Once you are confident that your CG code works correctly, use it to reproduce a figure similar to Figure 7.5 (right), p. 198, Ascher & Greif. Also add to the figure the convergence curve of SOR with the optimal ω.[1] Which method converges faster?

Preconditioned Conjugate Gradients

Implement the preconditioned conjugate gradient method (PCG), as summarized in the algorithm on p. 201 in Ascher & Greif, or p. 51 in Shewchuck. As preconditioners, use (i) IC (incomplete Cholesky), and (ii) SSOR. (The latter stands for symmetric SOR, and you will have to do an internet or library search to see the details.) For both these preconditioners, plot convergence curves such as those in part (e) of the previous problem. Based on your empirical data, arrange the following four methods in order of speed of convergence:

(A) CG with no preconditioning; (B) SOR (optimal ω); (C) PCG (with IC); (D) PCG (with SSOR, $\omega = 1$)

Experiment also with larger values of N before you make a final judgment.

The convergence analysis of PCG shows that a good preconditioner M is one for which the eigenvalues of $M^{-1}A$ are clustered (see, e.g., pp. 34 and 40 in Shewchuk). For each of the preconditioners (A) (i.e., $M = I$), (C) and (D) plot the eigenvalues of $M^{-1}A$ on the real axis.[2] Discuss the clustering and how it relates to the speed of convergence.

[1] In my own experiments there was less of a gap between the two error curves, see if you agree.

[2] Ok to compute the inverse. This means, of course, that you will have to restrict your investigations to smaller values of N.

Operation Count for Solving the Model Problem

For all our iterative methods the operation count to solve a problem is (work per iteration) $\times$ (number of iterations). For all our methods the work per iteration is one matrix-vector multiplication, which is $O(N^2)$ work in case of the model problem. The methods differ only in the number of iterations for convergence. For Jacobi, GS and steepest descents we showed that the number of steps for convergence is $O(N^2 \log \frac{1}{\epsilon})$, and for CG and SOR (optimal ω) it is $O(N \log \frac{1}{\epsilon})$.

Assume that for PCG the number of steps is $O(N^p \log \frac{1}{\epsilon})$, and if the preconditioner is any good then $0 < p < 1$. By numerical experimentation and analyzing your data, estimate the value of p for the PCG method with the two preconditioners, IC and SSOR. What is the total operation count for solving the model problem with these two preconditioners?