Division of Scientific Computing
Department of Information Technology
Uppsala University

NGSSC: Numerical Methods in Scientific Computing
January 2012

Project - Solving linear systems with complex symmetric matrices

1 Aim of the project

The project deals with large scale linear systems with complex symmetric matrices and their solution using iterative methods. Consider the system

$$A\mathbf{x} = \mathbf{b}, \tag{1}$$

where $\mathbf{x} \in C^n, \mathbf{x} \in C^n$ and $A \in C^{n\times n}$.
The matrix is called *complex symmetric* if $A = A^T$. We assume also that the real part of A is symmetric positive definite. (The theory is valid also in the case when the symmetric part of the imaginary part of A, i.e., the matrix $1/2(imag(A) + (imag(A))^T)$, is positive semidefinite.)

2 Solution approaches

Method 1:

Solve the system as it is given (i.e., as a complex system) using a suitable iterative solution method.

Method 2:

Since there is much more experience in solving real systems of equations with iterative methods, rewrite the system (1) as a twice larger real system of equations and solve it, again, via some iterative method.
We describe the idea in some more details.
The matrix A and the vector $\mathbf{b}$ have complex entries. The solution $\mathbf{x}$ is a complex vector itself.
We rewrite (1) and separate the equations for the real and the complex part of the solution vector $\mathbf{x}$. Let $\mathbf{x} = \mathbf{x}_r + i\,\mathbf{x}_c$ and similarly, $A = A_r + i\,A_c$ and $\mathbf{b} = \mathbf{b}_r + i\,\mathbf{b}_c$. We substitute the latter

in (1) and obtain the following real linear system of equations which, as already mentioned, is twice larger:

$$\begin{bmatrix} A_r & -A_c \\ A_c & A_r \end{bmatrix} \begin{bmatrix} \mathbf{x}_r \\ \mathbf{x}_c \end{bmatrix} = \begin{bmatrix} \mathbf{b}_r \\ \mathbf{b}_c \end{bmatrix} \tag{2}$$

The matrix $\widehat{A} = \begin{bmatrix} A_r & -A_c \\ A_c & A_r \end{bmatrix}$ is skew-symmetric, i.e., $\widehat{A}^T = -\widehat{A}$. (It has purely imaginary eigenvalues.)
The system in (2) is now real and can be solved using standard iterative solution methods, such as `GMRES` and others.
For systems of the form

$$\begin{bmatrix} B & -C \\ C & B \end{bmatrix},$$

where B is symmetric and positive definite, the following preconditioner is known to be efficient:

$$\widehat{P} \equiv \begin{bmatrix} B & -C \\ C & B+2C \end{bmatrix} = \begin{bmatrix} B & 0 \\ C & B+C \end{bmatrix} \begin{bmatrix} I & -B^{-1}C \\ 0 & B^{-1}(B+C) \end{bmatrix} \tag{3}$$

It is shown that all eigenvalues of the preconditioned system $\widehat{P}^{-1}\widehat{A}$ belong to the interval $[0.5, 1]$. Thus, when solving the system (2) iteratively, using $\widehat{P}$ as a preconditioner, the number of iterations is be bounded, independently of the size of the system. You are encouraged to check the eigenvalue estimates numerically.
For details on the preconditioner, see [2].

3 Tasks

1. You have at your disposal two generators of complex symmetric matrices (see below). Using both of them, create consecutively matrices of increasing size: 50, 100, 500, 5000 $\cdots$. You could be curious and
(i) check the sparsity of the above matrices (`spy`) and their structure,
(ii) for some smaller-sized matrices compute the complete spectra by using the `MATLAB` function `eig` and see how the spectrum changes with size.

2. The *Quasi-Minimal Residual (QMR)* method is one of the iterative solution methods, recommended for solving complex linear systems. Read some theory about QMR and describe it briefly. What are the main features of this method? Is it computationally cheaper than GMRES, for instance?

3. Solve Problem (1) using unpreconditioned `QMR`. Plot the convergence. (Please use `semilogy` and not `plot`.)

   For $n = 50$ you see that it takes exactly 50 iterations for QMR to converge. This is an illustration of the *final termination property of the method*. What does the property mean?

4. Solve Problem (2) using a preconditioned `GMRES`.

   Implement the action of the preconditioner as follows:

   Use the preconditioner $\widehat{P}$ as defined in (3). The preconditioner should be implemented in its factored form. You have to write a Matlab routine, called, say, `blkprec.m`, which solves a system with the factorized preconditioner, using the particular block form.

   In details, we want to solve

$$\begin{bmatrix} B & 0 \\ C & B+C \end{bmatrix} \begin{bmatrix} I & -B^{-1}C \\ 0 & B^{-1}(B+C) \end{bmatrix} \begin{bmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \end{bmatrix} = \begin{bmatrix} \mathbf{b}_1 \\ \mathbf{b}_2 \end{bmatrix}$$

   We first solve

$$\begin{bmatrix} B & 0 \\ C & B+C \end{bmatrix} \begin{bmatrix} \mathbf{y}_1 \\ \mathbf{y}_2 \end{bmatrix} = \begin{bmatrix} \mathbf{b}_1 \\ \mathbf{b}_2 \end{bmatrix}$$

   and then

$$\begin{bmatrix} I & -B^{-1}C \\ 0 & B^{-1}(B+C) \end{bmatrix} \begin{bmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \end{bmatrix} = \begin{bmatrix} \mathbf{y}_1 \\ \mathbf{y}_2 \end{bmatrix}$$

   The two steps consist of the following computational tasks:

   (I) Forward step:
   Solve $B\mathbf{y}_1 = \mathbf{b}_1$
   Solve $(B+C)\mathbf{y}_2 = \mathbf{b}_2 - C\mathbf{y}_1$

   (II) Backward step:
   Solve $(B+C)\mathbf{x}_2 = \mathbf{y}_2$
   Compute $\mathbf{x}_1 = \mathbf{y}_1 - \mathbf{y}_2 + \mathbf{x}_2$ (Why?)

   The way to use the preconditioner in MATLAB is as follows

```
[x,flag,relres,iter,resvec] = gmres(A,rhs,restart,tol,maxit,...
                                     @blkprec,[],[],A_r,A_c)
```

   The MATLAB function

```
 function w = blkprec(v,A_r,A_c)
```

   should implement the solution of the system $\widehat{P}\mathbf{w} = \mathbf{v}$ as sketched above. Use MATLAB backslash operator to solve systems with A_r and $A_r + A_c$.

   (OBS! It is not acceptable to assemble the triangular factors and then solve systems with those.)

   Remark: Of course, one can use OMR, preconditioned by $\widehat{P}$. However, QMR requires the action of the transposed of the preconditioner and some special attention should be paid to that issue when implementing the preconditioner in `blkprec.m`.

5. An additional task to bring 0.5 points extra: In [2], an algorithm is described, where one can reorganize the computations in (I) and (II) so that the solution with B can be avoided. If you implement that algorithm in `blkprec.m`, an extra 0.5 pt will be added to your final course points.

6. Derive the computational complexity of solving a system with $\widehat{P}$. Assume that you solve the systems with $B + C$ using algebraic Multigrid method.

4 Test matrices

You have at your disposal two Matlab routines, which generate complex symmetric test-matrices.

(1) `Pade_parabolic_matrix.m`
The so-called R_{22}*–Padé approximation* systems

$$A = \Big(I + \Big(1 + i\Big)\ L\Big) \tag{4}$$

which arise in Padé type integration schemes for parabolic problems.

(2) `Shift_omega_matrix.m`
The so-called *shifted-omega* systems

$$A = L + i\omega L \tag{5}$$

For more details on both types of test matrices, see [1] and the references therein.

5 Writing a report on the results

The report has to have the following issues covered:

1. Brief description of the problem and the methods used and the computational complexity of the preconditioner $\widehat{P}$.

2. Numerical experiments

   Describe the experiments (iteration counts, plots of the residual history) for representative cases. How does the number of iterations grow with the size? Which method is to recommended for the given test problems and why? Please note, that iteration counts should be repoted preferably in a table and not as a graph.

3. Conclusions.

A printout of the Matlab code must be attached to the report.
Success!

References

[1] O. Axelsson, A. Kucherov. Real valued iterative methods for solving complex symmetric linear systems. *Numerical Linear Algebra with Applications*, 7 (2000), no. 4, 197-218.

[2] O. Axelsson, P. Boyanova, M. Kronbichler, M. Neytcheva, X. Wu Numerical and Computational Efficiency of Solvers for Two-Phase Problems. Uppsala University, Institute for Information Technology. Technical report 2012-002, http://www.it.uu.se/research/publications/reports/2012-002/