POLITECNICO
MILANO 1863

Course of Numerical Methods for Engineering
Lab 8

Luca Bonaventura, Tommaso Benacchio

MOX - Politecnico di Milano

PHYS-ENG, A.Y. 2020-21

2-3/11/2020

Topic of this session:

- **Iterative methods for linear systems**

Iterative methods

Definition

An iterative method for the solution of problem $Ax = b$ consists in a sequence

$$x(k+1) = Bx(k) + f,\ k \geq 0$$

computed starting from an initial estimate of the solution $x^{(0)}$, that must be determined independently, and applying repeatedly an iteration matrix B that depends on the particular method used.

The vector $r(k) = b - Ax^{(k)}$ is called the residual of the method at the k-th iteration.

Definition

An iterative method is said to be convergent to the solution x of the problem $Ax = b$ in the vector norm $\|\cdot\|$ if, at least for some choice of $x^{(0)}$,

$$\lim_{k\to\infty} \|x^{(k)} - x\| = 0.$$

Preconditioned conjugate gradient and GMRES methods

The preconditioned conjugate gradient method (for symmetric positive definite matrices) and GMRES method (for general matrices) are two examples of Krylov methods, which look for solutions in Krylov spaces.

Definition

The $m-$dimensional Krylov space associated to the $n \times n$ matrix A and to the vector v is the linear space $\mathcal{K}_m(v)$ generated by $\mathrm{v}, \mathrm{Av}, \ldots, \mathrm{A}^{m-1}\mathrm{v}$

Octave/MATLAB provides the intrinsic functions `pcg` and `gmres`. For the preconditioned case, use is as follows:

```
[x,flag,relres,iter] = pcg(A,b,tol,maxit,M)
[x,flag,relres,iter,resvec] = gmres(A,b,restart_val,tol,maxit,M)
```

Exercise 1

Given the matrix $A = I_N + H_N$, where I_N and H_N denote the $N \times N$ identity and Hilbert matrices, consider the linear system $Ax = b$ with a right hand side such that its exact solution is given by the vector $x_{ex} = [-1, -1, \ldots, -1, -1]^T$.

(a) **Solve the system in the case $N = 5$ with the conjugate gradient method (without preconditioning) using the function `pcg` with an error tolerance of 10^{-13} and a maximum number of 100 allowed iterations. Check the number of iterations that were actually carried out. Compute the relative errors and the normalized residual in the Euclidean norm, and estimate the computation time.**

(b) **Repeat the computation using the preconditioned conjugate gradient method, using as preconditioner the diagonal matrix with the same diagonal as the matrix A. Explain the differences in the results on the basis of the theory.**

(c) **Repeat the tests with $N = 25$.**

Exercise 2

Build the 800×800 **sparse matrix** A **whose main diagonal is given by the vector** $[12, 12, \ldots, 12, 12]^T$ **and whose 10-th,20-th and 80-th sub- and superdiagonals contain entries equal to -1. Build** $b = Ax_{ex}$ **where** $x_{ex} = [1, 1, \ldots, 1, 1]^T$.

(a) **Solve the system using the conjugate gradient method (without preconditioning) using the function** `pcg` **with an error tolerance of** 10^{-9} **and a maximum number of 300 allowed iterations. Check the number of iterations that were actually carried out. Compute the relative errors and the normalized residual in the Euclidean norm.**

(b) **Solve the system using the preconditioned version of the method, using as preconditioner the diagonal matrix with the same diagonal as the matrix** A. **Explain the differences in the results on the basis of the theory.**

(c) **Solve the system with the Cholesky factorization, then compare the execution times of the three approaches.**

(d) **Repeat the previous points for the matrix that has on the main diagonal the vector** $[12, 24, \ldots, 12 \times 800]^T$.

(e) **Repeat the previous points for** 8000×8000 **sparse matrices with the same structure, which also have the 2000-th subdiagonal and superdiagonal with entries equal to -1.**

Exercise 3

Repeat exercise 3 using the same matrices, additionally with 1 on the 400-th superdiagonal (4000-th superdiagonal for point (e)), using the GMRES method instead of the preconditioned conjugate gradient method, and LU factorization instead of Cholesky factorization.

Exercise 4

Build a block-diagonal matrix A **of dimension** 1000×1000 **using blocks of dimension** 5×5 **around the main diagonal equal to**

$$B = \begin{pmatrix} 10 & 3 & 4 & 1 & 1 \\ 3 & 30 & 2 & 2 & 2 \\ 4 & 2 & 50 & 3 & 4 \\ 1 & 2 & 3 & 30 & 4 \\ 1 & 2 & 4 & 4 & 10 \end{pmatrix}$$

Compute the condition number of the matrix A **in the Euclidean and infinity norm. Compute the vector** $b = Ax_{ex}$, **where** $x_{ex} = [2, -3, \ldots, 2, -3]^T$, $x_{ex} \in \mathbb{R}^{1000}$. **After representing** A **as sparse matrix, solve the system** $Ax = b$ **using:**

(a) **the command** $\backslash$ **(backslash);**

(b) **the Cholesky factorization**

(c) **the conjugate gradient method (with and without preconditioning with the diagonal part of** A**) using the function** `pcg` **with an error tolerance of** 10^{-12} **and a maximum number of 300 iterations.**

Estimate the time required for the solution by each method and compute the absolute and relative error of the computed solution with respect to x_{ex} **in the** l^1, l^2 **and** l^∞ **norm. Compute the** *a priori* **and** *a posteriori* **error bounds and check that the corresponding error estimates are satisfied.**

Exercise 5

Repeat the previous exercise building the matrix A **of dimension** 25000×25000 **from the blocks**

$$\mathrm{B} = \begin{pmatrix} 10 & 3 & 4 & 1 & 1 \\ 3 & 300 & 2 & 2 & 2 \\ 4 & 2 & 500 & 3 & 4 \\ 1 & 2 & 3 & 3000 & 5 \\ 1 & 2 & 4 & 4 & 10^7 \end{pmatrix}.$$

The matrix A **also has elements equal to 1 on the 2000-th superdiagonal and -1/40 on the 3000-th subdiagonal. Use, instead of the Cholesky factorization and conjugate gradient method, the LU factorization and the GMRES method (with and without preconditioning with the diagonal of** A**) with a restart value of 32, an error tolerance of** 10^{-12} **and a maximum number of 300 outer iterations.**

Exercise 6

Implement your own version of the preconditioned conjugate gradient method and check its performance against the intrinsic Octave/MATLAB function `pcg` on solving the systems with symmetric matrices in the previous exercises.