Linear System

August 24, 2016

Consider the Dirichlet boundary-value problem:

$$-\Delta u \equiv -u_{xx} - u_{yy} = 2\pi^2 \sin \pi x \sin \pi y, \text{ for } (x, y) \in \Omega, \tag{1}$$
$$u(x, y) = 0 \ \ (x, y) \in \partial\Omega, \tag{2}$$

for $\Omega := \{x, y | 0 < x, y < 1\} \subseteq \mathbb{R}^2$ with boundary $\partial\Omega$, which has the exact solution

$$u(x, y) = \sin \pi x \sin \pi y,$$

and is shown in Figure 1.

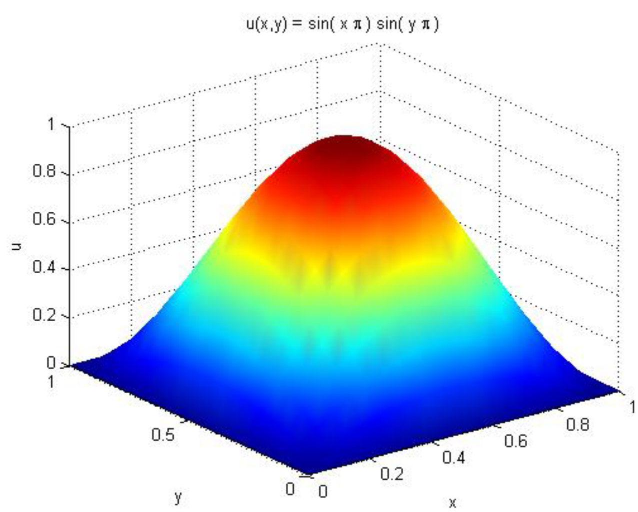

Figure 1: Exact solution.

Let $n \geq 1$ be an integer and $h = \frac{1}{n+1}$. Define

$$f_{ij} \equiv 2h^2\pi^2 \sin \pi x_i \sin \pi y_j$$

with $x_i = ih,\ y_j = jh,\ i, j = 0, 1, \ldots, n+1$. Then, applying the finite difference scheme to discrete Eq. (2), it results the linear system

$$Ax = f, \tag{3}$$

where

$$f = \begin{bmatrix} f_{:,1} \\ f_{:,2} \\ \vdots \\ f_{:,n} \end{bmatrix}, \quad A = \begin{bmatrix} A_1 & -I_n & & \\ -I_n & A_1 & \ddots & \\ & \ddots & \ddots & -I_n \\ & & -I_n & A_1 \end{bmatrix} \in \mathbb{R}^{n^2 \times n^2}$$

with

$$f_{:,j} = \begin{bmatrix} f_{1,j} \\ f_{2,j} \\ \vdots \\ f_{n,j} \end{bmatrix}, \quad \text{for } j = 1, \ldots, n,$$

$$A_1 = \begin{bmatrix} 4 & -1 & & \\ -1 & \ddots & \ddots & \\ & \ddots & \ddots & -1 \\ & & -1 & 4 \end{bmatrix} \in \mathbb{R}^{n \times n}.$$

Problem: How to solve the linear system $Ax = f$.

Requirements:

(a) Construct the coefficient matrix A with dense format and right hand side vector f. Solve this linear system by using $x = A\backslash f$.

(b) Use *function* command to construct the coefficient matrix A with sparse format and right hand side vector f. Solve this linear system by using $x = A\backslash f$.

(c) Compare the performance of (a) and (b). (Use "tic" and "toc" functions in MATLAB to estimate the CPU times.)

(d) Use conjugate gradients method to solve linear system (3).

1. Use MATLAB function `pcg` without any preconditioner:

   [x, flag, relres, iter] = pcg(A, b, tol, maxit)

2. Use MATLAB function `pcg` with a given preconditioner:

   [x, flag, relres, iter] = pcg(A, b, tol, maxit, M),
   [x, flag, relres, iter] = pcg(A, b, tol, maxit, M1, M2),
   [x, flag, relres, iter] = pcg(A, b, tol, maxit, [], M2),
   [x, flag, relres, iter] = pcg(A, b, tol, maxit, MFUN).

(i) **Jacobi method:** $A = D + (L + U), \quad M = D$

$$x_{k+1} = -D^{-1}(L + U)x_k + D^{-1}b,$$

where D is the diagonal part, L is the strictly lower triangular part, and U is the strictly upper triangular part, of A, respectively.

(ii) **Gauss-Seidel:** $A = (D + L) + U, \quad M = D + L$

$$x_{k+1} = -(D + L)^{-1}Ux_k + (D + L)^{-1}b.$$

(iii) **SSOR:** $A = D + L + L^T, \quad M = M(\omega)$

$$x^{(k)} = \left(M_\omega^{-T} N_\omega^T M_\omega^{-1} N_\omega\right) x^{(k-1)} + M(\omega)^{-1}b,$$

where

$$M(\omega) = \frac{1}{\omega(2-\omega)}(D + \omega L)D^{-1}\left(D + \omega L^T\right).$$

(iv) M may be a function handle MFUN returning $M^{-1}x$

[x, flag, relres, iter] = pcg(A, b, tol, maxit, ...
@(x)precSSOR(x,omega,mtxLower,mtxdiag)

(v) Use MATLAB functions "triu(A,1)" and "tril(A,-1)" to extract the strictly upper and lower triangular parts of A, respectively.