MATH7003-00: Assignment #9

Sangman Jung
e-mail: sangmanjung@khu.ac.kr
Kyung Hee University – June 10, 2020

Problem.

#1. Prove the inequality (8.5.13) on slide 5 of week11 [1].

$$\| x - x^{(m+1)} \|_{\infty} \leq \frac{c}{1-c} \| x^{(m+1)} - x^{(m)} \|_{\infty}.$$

#2. Consider the example on slide 18 and write down m-files for Gauss-Seidel and SOR. Generate Table 8.9. Discuss about the result. [1].

Example. Solve

$$\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = 0 \qquad 0 \leq x, y \leq 1$$

$$u(0,y) = \cos(\pi y), \quad u(1,y) = e^{\pi}\cos(\pi y), \quad u(x,0) = e^{\pi x}, \quad u(x,1) = -e^{\pi x}.$$

The true solution is

$$u(x,y) = e^{\pi x}\cos(\pi y).$$

Example. Recall the previous example (8.8.11). This was solved with both the Gauss-Seidel method and the SOR method. The initial guess for the iteration was taken to be the "bilinear" interpolation formula for the boundary data f:

$$\begin{aligned} u_h^{(0)}(x,y) &= (1-x)f(0,y) + xf(1,y) + (1-y)f(x,0) + yf(x,1) \\ &\quad - [(1-y)(1-x)f(0,0) + (1-y)xf(1,0) + y(1-x)f(0,1) + xyf(1,1)] \end{aligned}$$

at all interior grid points. The error test to stop the iteration was

$$\max_{1 \leq j,k, \leq N-1} \left| u_h(x_j,y_k) - u_h^{(m)}(x_j,y_k) \right| \leq \epsilon$$

with $\epsilon > 0$ given and the right-hand size of (8.7.5) used to predict the error in the iterate. The numerical results for the necessary number of iterates are given in **Table 8.9**.

Solution. (problem #1)

For the initial guess $x^{(0)}$, we know the fact that $x^{(1)} - x^{(0)} = Cr^{(0)}$ where $r^{(0)} = b - Ax^{(0)}$, C is the approximation of A^{-1}. Then, in general, we can define this form recursively, and we obtain a recursion formula for the error as

$$\begin{aligned} \mathrm{C}x - x^{(m+1)} &= x - x^{(m)} - Cr^{(m)} = x - x^{(m)} - C\left[b - Ax^{(m)}\right] \\ &= x - x^{(m)} - C\left[Ax - Ax^{(m)}\right] \end{aligned}$$

Thus, $x - x^{(m+1)} = (I - CA)(x - x^{(m)})$. By induction, we also obtain $x - x^{(m)} = (I - CA)^m (x - x^{(0)})$ for $m \geq 0$. In this, if we assume $\| I - CA \| < 1$ for some matrix norm, then

$$\| x - x^{(m)} \| = \| I - CA \|^m \| x - x^{(0)} \|$$

and, this converges th zero as $m \to \infty$ for any choice of initial guess $x^{(0)}$. This implies that

$$(I - CA)^m \to 0 \text{ as } m \to \infty .$$

Now, note that the theorem and the definition.

Theorem 7.9 Let A be a square matrix of order n. Then A^m converges to the zero matrix as $m \to \infty$ if and only if $r_\sigma(A) < 1$.

Definition Let A be an arbitrary matrix. The **spectrum** of A is the set of all eigenvalues of A, and it is denoted by $\sigma(A)$. The **spectral radius** is the maximum size of these eigenvalues, and it is denoted by

$$r_\sigma(A) = \max_{\lambda \in \sigma(A)} |\lambda|$$

Then, by Theorem 7.9, this is equivalent to $r_\sigma(I - CA) < 1$. For the case of $I - AC$, we obtain

$$I - AC = A(I - CA)A^{-1}$$

and thus $I - AC$ and $I - CA$ are similar matrices and have the same eigenvalues, by the **diagonalization theorem**. (see Chapter 5 in [4]) So if $\| I - AC \| < 1$, then $r_\sigma(I - CA) < 1$ is true, even if $\| I - CA \| < 1$ is not true, and convergence will still occur.

Therefore, $x - x^{(m+1)} = (I - CA)(x - x^{(m)})$ shows that the rate of convergence of $x^{(m)}$ to x is linear:

$$\| x - x^{(m+1)} \| \leq c \| x - x^{(m)} \| \text{ for } m \geq 0, \text{ with } c < 1 \text{ unknown.}$$

And, c is estimated computationally with

$$c = \mathrm{Max} \frac{\| x^{(m+2)} - x^{(m+1)} \|}{\| x^{(m+1)} - x^{(m)} \|} .$$

Then, we now easily can produce an error bound that

$$\begin{aligned} \| x^{(m+1)} - x^{(m)} \| &\geq \| \left[x - x^{(m)}\right] - \left[x - x^{(m+1)}\right] \| \quad \cdots \text{(I)} \\ &\geq \| x - x^{(m)} \| - \| x - x^{(m+1)} \| \quad \cdots \text{(II)} \\ &\geq \| x - x^{(m)} \| - c \| x - x^{(m)} \| \quad \cdots \text{(III)} \end{aligned}$$

(I) is $x^{(m+1)} - x^{(m)} = x^{(m+1)} + (x - x) - x^{(m)} = \left[x^{(m+1)} - x\right] + \left[x - x^{(m)}\right] = \left[x - x^{(m)}\right] - \left[x - x^{(m+1)}\right]$, (II) is by triangle inequality, and (III) is $\| x - x^{(m+1)} \| \leq c \| x - x^{(m)} \|$.

Thus, we obtain

$$\begin{aligned} \| x^{(m+1)} - x^{(m)} \| &\geq \| x - x^{(m)} \| - c \| x - x^{(m)} \| \\ &= (1 - c) \| x - x^{(m)} \| \end{aligned}$$

and, finally we have

$$\| x - x^{(m)} \| \leq \frac{1}{1-c} \| x^{(m+1)} - x^{(m)} \| \quad \Rightarrow \quad \| x - x^{(m+1)} \| \leq \frac{c}{1-c} \| x^{(m+1)} - x^{(m)} \|$$

since $\| x - x^{(m+1)} \| \leq c \| x - x^{(m)} \| \Rightarrow c \| x - x^{(m)} \| \leq c \left(\frac{1}{1-c} \| x^{(m+1)} - x^{(m)} \| \right)$.

■

Solution. (problem #2)

For $j = 1,2,...,N-1$ and $k = 1,2,...,N-1$, the '**matrix-free**' form of the **Gauss-Seidel (G-S) method** of the given partial differential equation is as follows.

$$u_h^{(m+1)}(x_j,y_k) = \frac{1}{4}\left[u_h^{(m)}(x_{j+1},y_k) + u_h^{(m)}(x_j,y_{k+1}) + u_h^{(m+1)}(x_{j-1},y_k) + u_h^{(m+1)}(x_j,y_{k-1})\right].$$

For the boundary points, use

$$u_h^{(m)}(x_j,y_k) = f(x_j,y_k) \text{ for all } m \geq 0.$$

Since the equation of this example is the homogeneous equation, $g(x_j,y_k) = 0$. (refer to (8.8.12) in [1]) Also, the **successive overrelaxation (SOR) method** is as follows.

$$v_h^{(m+1)}(x_j,y_k) = \frac{1}{4}\left[u_h^{(m)}(x_{j+1},y_k) + u_h^{(m)}(x_j,y_{k+1}) + u_h^{(m+1)}(x_{j-1},y_k) + u_h^{(m+1)}(x_j,y_{k-1})\right],$$

$$u_h^{(m+1)}(x_j,y_k) = \omega v_h^{(m+1)}(x_j,y_k) + (1-\omega)u_h^{(m)}(x_j,y_k) \text{ for } j = 1,2,...,N-1,\ k = 1,2,...,N-1.$$

Note that the **optimal acceleration parameter** is

$$\omega^* = \frac{2}{1+\sqrt{1-\xi^2}} \text{ where } \xi = 1 - 2\sin^2\left(\frac{\pi}{2N}\right).$$

If $\omega^* = 1$, then the formula of SOR is directly equals to the fomular of Gauss-Seidel. i.e., we can easily know that the SOR method is the generalized version of Gauss-Seidel method.

We use the stop criterion in **problem #1** as

$$\| x - x^{(m+1)} \|_\infty \leq \frac{c}{1-c} \| x^{(m+1)} - x^{(m)} \|_\infty, \quad \text{where} \quad c \doteq \frac{\| x^{(m+1)} - x^{(m)} \|_\infty}{\| x^{(m)} - x^{(m-1)} \|_\infty},$$

in order for our implementation. Now, the results of **problem #2** is attached below. (see **Figure 1.**)

Table 8.9 Number of iterates necessary to solve (8.8.5)

N	ϵ	Gauss–Seidel	SOR
8	.01	25	12
8	.001	40	16
16	.001	142	32
32	.001	495	65
8	.0001	54	18
16	.0001	201	35
32	.0001	733	71

명령 창

Table 8.9 Number of iterates necessary to solve (8.8.5)

N	epsilon	Gauss-Seidel	SOR
8	0.01	25	12
8	0.001	40	16
16	0.001	142	31
32	0.001	495	65
8	0.0001	54	16
16	0.0001	201	31
32	0.0001	733	65

fx >>

Figure 1. The results of the problem 2 using MATLAB. The left-side is in the textbook.

Discussion. (problem #2)

Our problem has the equivalent step size $h = 1/N$ for x and y. Hence the finite mesh grid depends on the step parameter N. This implies that if N is larger, then the iteration number is basically larger.

Moreover, if the stop criterion $\epsilon > 0$ is smaller, we need to iterate more than previous, in order to meet more rigorous criterion ϵ.

In Table 8.9., we suggest the Gauss-Seidel and SOR results for $N = 8, 16, 32$ and $\epsilon = 0.01, 0.001, 0.0001$, and we can see that if N and ϵ is larger, the iteration of both methods is larger.

Especially, we already learned the fact that the SOR method is the acceleration of Gauss-Seidel method, so we intuitively can see that the iteration number of SOR results are better than Gauss-Seidel results in Table 8.9. So we conclude that the use of SOR greatly reduces the resulting work, although it still is large when N is large.

With either method, note that doubling N will increase the number of equations to be solved by a factor of 4, and thus the work per iteration will increase by the same amount.

Table_8_9_HW9.m

```
%% MATH7003-00: Assignment #9, 2019310290 Sangman Jung.
clear,clc

N = [8 16 32]; % step N
epsilon = [0.01 0.001 0.0001]; % error criterion
f = @(x,y) exp(pi*x).*cos(pi*y); % exact solution of our problem

ksi = 1-2.*((sin(pi./(2*N))).^2);
omega = 2./(1+sqrt(1-(ksi).^2)); % optimal acceleration parameter

uval{length(N),length(epsilon),2} = []; % pre-allocation of the numerical solution

iteration = zeros(length(N),length(epsilon)); % pre-allocation of G-S iteration
iteration_sor = zeros(length(N),length(epsilon)); % pre-allocation of SOR iteration

% Full Loop
for w_iter = 1:2 % w_iter = 1 is G-S, w_iter = 2 is SOR
    for N_iter = 1:length(N) % iterate per step N
        h = 1/N(N_iter); % step size
        x = 0:h:1; y = x; % spatial variables (grid)
        nx = length(x); ny = length(y); % the size of the grid
        u_ini = zeros(nx,ny); % initial guess
        u_exact = zeros(nx,ny); % pre-allocation of exact solution
        u_new = zeros(nx,ny); % pre-allocation of numerical solution
        for eps_iter = 1:length(epsilon) % iteration of the criterion
            m_iter = 1; % initialize the iteration number
            for k = 1:ny
                for j = 1:nx
                    u_ini(j,k)=(1-x(j))*f(0,y(k))+x(j)*f(1,y(k))+(1-y(k))*f(x(j),0)+...
                        y(k)*f(x(j),1)-((1-y(k))*(1-x(j))*f(0,0)+(1-y(k))*x(j)*f(1,0)+...
                        y(k)*(1-x(j))*f(0,1)+x(j)*y(k)*f(1,1)); % compute the initial guess
                    u_exact(j,k)=f(x(j),y(k)); % allocate the exact solution
                end
            end
            u_old = u_ini; % update the m-iteration

            % boundary values of the problem
            u_new(1,:) = u_exact(1,:); % left vertical
            u_new(:,1) = u_exact(:,1); % bottom
            u_new(end,:) = u_exact(end,:); % right vertical
            u_new(:,end) = u_exact(:,end); % top

            error = 1; % initialize the error
            while 1 % Main Loop
                for k = 2:ny-1
                    for j = 2:nx-1
                        % Gauss-Seidel (G-S) method
                        u_new(j,k) = (u_old(j+1,k)+u_old(j,k+1)+u_new(j-1,k)+u_new(j,k-1))/4;
                        if w_iter > 1
                            % Successive OverRelaxation (SOR) method
                            u_new(j,k) = omega(N_iter)*u_new(j,k) + (1-omega(N_iter))*u_old(j,k);
                        end
                    end
                end
                if m_iter > 1 % compute the error (8.7.5) in Chapter 8.7
                    c = max(max(abs(u_new-u_old)))/max(max(abs(u_old-u_2old)));
                    error = c/(1-c)*(max(max(abs(u_new-u_old))));
                end
                if error <= epsilon(eps_iter) % error criterion
                    break;
                end
                m_iter = m_iter + 1; % update the error after passing the criterion
                u_2old = u_old; % uptate from m-1 to m
                u_old = u_new; % update from m to m+1
            end
            if w_iter == 1 % G-S iteration
                iteration(N_iter,eps_iter) = m_iter;
            else % SOR iteration
                iteration_sor(N_iter,eps_iter) = m_iter;
            end
            uval(eps_iter,N_iter,w_iter) = {u_new}; % save the numerical solution
        end
    end
end

% summarize our results in order to obtain the table
N_table = [N(1) N N]';
eps_table = [epsilon(1) repmat(epsilon(2),1,3) repmat(epsilon(3),1,3)]';
iteration = reshape(iteration,[9 1]);
iteration = iteration([1 4:9]);
iteration_sor = reshape(iteration_sor,[9 1]);
iteration_sor = iteration_sor([1 4:9]);

% print Table 8.9
fprintf('\nTable 8.9  Number of iterates necessary to solve (8.8.5)\n')
fprintf('-------------------------------------------------------\n')
fprintf('    N       epsilon       Gauss-Seidel       SOR    \n')
fprintf('-------------------------------------------------------\n')
for t = 1:length(N_table)
    fprintf('   %3.0f       %6.4g           %3.0f           %3.0f\n',...
        [N_table(t) eps_table(t) iteration(t) iteration_sor(t)]);
end
fprintf('-------------------------------------------------------\n')
```

References.

[1] Atkinson, K. E. (2008). An introduction to numerical analysis. John wiley & sons.

[2] Atkinson, K., Han, W., & Stewart, D. E. (2011). Numerical solution of ordinary differential equations (Vol. 108). John Wiley & Sons.

[3] Atkinson, K. E., & Han, W. (1985). Elementary numerical analysis (p. 17). New York et al.: Wiley.

[4] Lay, D. C. (2012). Linear algebra and its applications. Addison-Wesley.