《工程硕士数学》第二次计算实习

> 软硕232 丁浩宸 2023213911

第一题

理论基础

- J迭代法及其收敛条件（$\rho(B_J) < 1$））
- SOR迭代法及其收敛条件（$0 < \omega < 2$）

算法描述

- J迭代法：$\begin{cases} A = D - L - U \\ B_J = D^{-1}(L + U) \\ f_J = D^{-1}b \\ x^{(k+1)} = B_J * x^{(k)} + f_J \end{cases}$
- SOR迭代法：$\begin{cases} A = D - L - U \\ L_\omega = (D - \omega L)^{-1}[(1 - \omega)D + \omega U] \text{，式中}\omega\text{为松弛因子} \\ x^{(k+1)} = L_\omega x^{(k)} + \omega(D - \omega L)^{-1} \end{cases}$

计算代码

```
format long;
n = 6;
% n - 8;
% n = 10;
x = ones(n, 1);
A = hilb(n);
b = A * x

D = diag(diag(A));
L = -tril(A, -1);
U = -triu(A, 1);
BJ = D \ (L + U);
fJ = D \ b;
BG = (D - L) \ U;
fG = (D - L) \ b;
Lw = (D - wb * L) \ ((1 - wb) * D + wb * U);

eigs(BJ)
% wb = 2 / (1 + sqrt(1 - max(eig(BJ) * max(eig(BJ)))))

x0 = zeros(n, 1);
for i = 1:1:100
    % x0 = Lw * x + wb * (D - wb * L) \ b;
    % x = BG * x + fG
    x0 = BJ * x0 + fJ;
end
x0

w = 1.5;
```

```
% w = 1;
% w = 1.25;
x0 = zeros(n, 1);
for i = 1:1:100
    x0 = Lw * x0 + w * (D - w * L) \ b;
    % x = BG * x + fG
    % x = BJ * x + fJ
end
x0
```

结果分析

	n=6	n=8	n=10
J迭代法	$x=10^{63}*\begin{bmatrix}-1.08523\\-2.34194\\-3.12160\\-3.66779\\-4.07562\\-4.39315\end{bmatrix}$	$x=10^{78}*\begin{bmatrix}-0.42102\\-0.95282\\-1.30662\\-1.56633\\-1.76716\\-1.92792\\-2.05990\\-2.17039\end{bmatrix}$	$x=10^{89}*\begin{bmatrix}-0.33334\\-0.77994\\-1.09177\\-1.32845\\-1.51626\\-1.66978\\-1.79802\\-1.90697\\-2.00082\\-2.08257\end{bmatrix}$
SOR迭代法（$\omega=1$）	$x=\begin{bmatrix}1.02048\\0.85372\\1.14931\\1.11449\\0.99123\\0.85673\end{bmatrix}$	$x=\begin{bmatrix}1.02743\\0.86083\\1.03437\\1.12641\\1.10632\\1.03316\\0.94155\\0.84805\end{bmatrix}$	$x=\begin{bmatrix}1.02107\\0.94341\\0.90452\\1.05580\\1.11823\\1.10923\\1.06077\\0.99386\\0.92051\\0.84713\end{bmatrix}$
SOR迭代法（$\omega=1.25$）	$x=\begin{bmatrix}0.65717\\0.51267\\0.79687\\0.68934\\0.63066\\0.54297\end{bmatrix}$	$x=\begin{bmatrix}0.65959\\0.53661\\0.68263\\0.71882\\0.70481\\0.65808\\0.60157\\0.54356\end{bmatrix}$	$x=\begin{bmatrix}0.65218\\0.61610\\0.55101\\0.69523\\0.71398\\0.71007\\0.67754\\0.63557\\0.58932\\0.54314\end{bmatrix}$
SOR迭代法（$\omega=1.5$）	$x=\begin{bmatrix}0.45920\\0.33192\\0.60601\\0.43156\\0.46663\\0.36360\end{bmatrix}$	$x=\begin{bmatrix}0.45873\\0.36699\\0.48716\\0.48787\\0.49351\\0.45336\\0.42020\\0.37799\end{bmatrix}$	$x=\begin{bmatrix}0.45149\\0.44056\\0.35223\\0.51332\\0.47726\\0.50380\\0.46408\\0.44471\\0.40771\\0.37867\end{bmatrix}$

分析如下：

- 在$n=6$时$\rho(B_J)=4.30853$，在$n=8$时$\rho(B_J)=6.04213$，在$n=10$时$\rho(B_J)=7.77982$，这说明J迭代法在以上各种情况下都无法收敛，计算结果也印证了这一点。
- 由于ω取了$(0,2)$之间的值，因此SOR迭代法总能收敛，计算结果也印证了这一点，即绝对误差低于100%。

第三题

理论基础

J迭代法、SOR迭代法和CG法的速度比较

- 对于J法和SOR法，有$\begin{cases} R(B_J)=\frac{1}{2}\pi^2h^2+O(h^4) \\ R(L_{\omega_b})=2\pi h+O(h^3) \end{cases}$，即在$\omega\approx\omega_b$时SOR法远快于J法
- 对于CG法，课本上没有给出其收敛速度的确切表达式，但它最多n步即能得到方程组精确解，因此猜测其比J法和$\omega\approx\omega_b$时的SOR法都更快。

算法描述

- J法和SOR法的计算公式如上题“算法描述”所述
- CG法的计算公式为：$\begin{cases} \alpha_k=\frac{(r^{(k)},r^{(k)})}{(Ap^{(k)},p^{(k)})} \\ x^{(k+1)}=x^{(k)}+\alpha_k p^{(k)} \\ r^{(k+1)}=r^{(k)}-\alpha_k Ap^{(k)} \\ \beta_k=\frac{(r^{(k+1)},r^{(k+1)})}{(r^{(k)},r^{(k)})} \\ p^{(k+1)}=r^{(k+1)}+\beta_k p^{(k)} \end{cases}$，初始值$\begin{cases} r^{(0)}=b-Ax^{(0)} \\ p^{(0)}=r^{(0)} \end{cases}$

计算代码

```
format long;

v1 = [];
v2 = [];
v3 = [];
n = 40;
for i = 1:n
    v1 = [v1, 1];
    v2 = [v2, -8];
    v3 = [v3, 20];
end
v1 = v1';
v2 = v2';
v3 = v3';
diagonal = spdiags([v1 v2 v3 v2 v1], -2 : 2, n, n);
A = full(diagonal);
x = ones(n, 1);
b = zeros(n, 1);
D = diag(diag(A));
L = -tril(A, -1);
U = -triu(A, 1);
BJ = D \ (L + U);
fJ = D \ b;
BG = (D - L) \ U;
fG = (D - L) \ b;
```

```matlab
eigsBJ = eigs(BJ)

% J
i = 0;
while true
    if (norm(x, inf) <= 1e-6)
        i
        break
    end
    i = i + 1;
    % x0 = Lw * x + wb * (D - wb * L) \ b;
    % x = BG * x + fG
    x = BJ * x + fJ;
end
x

% SOR
for w = 1:0.2:1.8
    x = ones(n, 1);
    i = 0;
    Lw = (D - w * L) \ ((1 - w) * D + w * U);
    while true
        if (norm(x, inf) <= 1e-6)
            i
            break
        end
        i = i + 1;
        x = Lw * x + w * (D - w * L) \ b;
        % x = BG * x + fG
        % x = BJ * x + fJ
    end
    w
    x
end

% CG
x = ones(n, 1);
i = 0;
r = b - A * x;
p = r;
while true
    if (norm(x, inf) <= 1e-6)
        i
        break
    end
    i = i + 1;
    ak = (r' * r) / ((A * p)' * p);
    x = x + (ak * p);
    rnext = r - (ak * A * p);
    bk = (rnext' * rnext) / (r' * r);
    p = rnext + bk * p;
    r = rnext;
end
x
```

计算结果

收敛至$\|\|x^{(k)}\|\|_\infty \le 10^{-6}$所用的迭代次数	n=10	n=20	n=40
J迭代法	38	60	70
SOR迭代法（$\omega = 1$）	21	23	23
SOR迭代法（$\omega = 1.2$）	15	18	18
SOR迭代法（$\omega = 1.4$）	20	24	27
SOR迭代法（$\omega = 1.6$）	30	35	45
SOR迭代法（$\omega = 1.8$）	61	67	76
共轭梯度法	5	10	17

另经计算，J迭代法确实总是满足$\rho(B_J) < 1$的收敛条件。

分析：

- 课本中描述的“在$\omega \approx \omega_b$时SOR法远快于J法”确实得到印证，且在本题中ω_b的取值应在$\omega = 1.2$附近
- 共轭梯度法在$n = 10$和$n = 20$时显著快于SOR法和J法，并在$n = 40$时略快于SOR法、显著快于J法。
- 共轭梯度法“最多n步即能得到方程组精确解”（即不超过n步能达到要求的近似解）也确实成立。

注意本题中的收敛条件是无穷范数$||x^{(k)}||_\infty \le 10^{-6}$，需要使用 `norm(x, inf)` 函数而非 `norm(x)` ，否则收敛所需的迭代次数将有微小差异（约0-2次）。