Computer Exercise 8.4.3

The following program will produce approximate solutions to the following $\mathbf{Ax} = \mathbf{b}$ system

$$\begin{bmatrix} 7 & 3 & -1 & 2 \\ 3 & 8 & 1 & -4 \\ -1 & 1 & 4 & -1 \\ 2 & -4 & -1 & 6 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} -1 \\ 0 \\ -3 \\ 1 \end{bmatrix}$$

using the Jacobi method, the Gauss-Seidel method and the SOR method (with $\omega = 1.4$). In each case, a maximum amount of iterations $k_{\max}$ (selected based on testing each method) and an error tolerance of 10^{-4} (four decimal places of accuracy); the error is determined by $\text{error} = \max_{1 \le i \le n} |\mathbf{x}_{exact}^{(i)} - \mathbf{x}_{approx}^{(i)}|, i \in \{1, \ldots, n\}$ (here, the "i" indicates the component i^{th} of the corresponding vectors) where $\mathbf{x}_{exact} = [-1, 1, -1, 1]^T$ and in this case, $n = 4$. Each algorithm corresponding to each method is designed so that the program will exit when $\text{error} \le 10^{-4}$ or when $k \ge k_{\max}$ (however, the value for $k_{\max}$ is picked such that only the former condition is met).The error and iteration number evaluated upon exit of the program will be displayed along with $\mathbf{x}_{approx}$.

```
A = [7, 3, -1, 2; 3, 8, 1, -4; -1, 1, 4, -1; 2, -4, -1, 6];
b = [-1, 0, -3, 1]';
x0 = [0, 0, 0, 0]';
x_exact = [-1, 1, -1, 1]';


err_tol = (10^(-4));
kmax = 80;
xapprox = jacobi(A, b, x0, kmax, x_exact, err_tol)
```

```
error tolerance satisfied
exited at k = 73, error: 8.79977e-05
xapprox = 4×1
   -0.9999
    0.9999
   -1.0000
    0.9999
```

```
err_tol = (10^(-4));
kmax = 40;
xapprox = gaussseidel(A, b, x0, kmax, x_exact, err_tol)
```

```
error tolerance satisfied
exited at k = 37, error: 9.98589e-05
xapprox = 4×1
   -0.9999
    0.9999
   -1.0000
    0.9999
```

```
err_tol = (10^(-4));
kmax = 20;
w = 1.4;
```

```
xapprox = sor(A, b, x0, kmax, x_exact, err_tol, w)
```

```
error tolerance satisfied
exited at k = 12, error: 7.88288e-05
xapprox = 4×1
   -1.0001
    1.0001
   -1.0000
    1.0001
```

The SOR method provides the fastest convergence where it exits at $k = 12$; the Jacobi method and Gauss-Seidel method exit at $k = 73$ and $k = 37$ respectively.

```
function x = jacobi(A, b, x0, kmax, x_exact, err_tol)
    xkm1 = x0;
    n = length(b);
    k = 1;
    error = ones(n, 1);
    while (k < kmax) && (max(error) > err_tol)
        xk = zeros(n, 1);
        for i = 1:n
            sum = 0;
            for j = 1:n
                if j ~= i
                    sum = sum + ((A(i,j)/A(i,i))*xkm1(j));
                end
            end
            xk(i) = (b(i)/A(i, i)) - sum;
        end
        xkm1 = xk;
        k = k + 1;
        error = abs(xk - x_exact);
        if k >= kmax
            disp('max iterations reached')
            fprintf('exited at k = %d, error: %5.5e \n', k, max(error))
        elseif max(error) <= err_tol
            disp('error tolerance satisfied')
            fprintf('exited at k = %d, error: %5.5e \n', k, max(error))
        end
    end
    x = xk;
end

function x = gaussseidel(A, b, x0, kmax, x_exact, err_tol)
    xkm1 = x0;
    n = length(b);
    k = 1;
    error = ones(n, 1);
    while (k < kmax) && (max(error) > err_tol)
        xk = zeros(n, 1);
        for i = 1:n
            sum = 0;
            for j = 1:n
                if j < i
```

```
                    sum = sum + ((A(i,j)/A(i,i))*xk(j));
                elseif j > i
                    sum = sum + ((A(i,j)/A(i,i))*xkm1(j));
                end
            end
            xk(i) = (b(i)/A(i, i)) - sum;
        end
        xkm1 = xk;
        k = k + 1;
        error = abs(xk - x_exact);
        if k >= kmax
            disp('max iterations reached')
            fprintf('exited at k = %d, error: %5.5e \n', k, max(error))
        elseif max(error) <= err_tol
            disp('error tolerance satisfied')
            fprintf('exited at k = %d, error: %5.5e \n', k, max(error))
        end
    end
    x = xk;
end

function x = sor(A, b, x0, kmax, x_exact, err_tol, w)
    xkm1 = x0;
    n = length(b);
    k = 1;
    error = ones(n, 1);
    while (k < kmax) && (max(error) > err_tol)
        xk = zeros(n, 1);
        for i = 1:n
            sum = 0;
            for j = 1:n
                if j < i
                    sum = sum + ((A(i,j)/A(i,i))*xk(j));
                elseif j > i
                    sum = sum + ((A(i,j)/A(i,i))*xkm1(j));
                end
            end
            xk(i) = w*((b(i)/A(i, i)) - sum) + (1-w)*xkm1(i);
        end
        xkm1 = xk;
        k = k + 1;
        error = abs(xk - x_exact);
        if k >= kmax
            disp('max iterations reached')
            fprintf('exited at k = %d, error: %5.5e \n', k, max(error))
        elseif max(error) <= err_tol
            disp('error tolerance satisfied')
            fprintf('exited at k = %d, error: %5.5e \n', k, max(error))
        end
    end
    x = xk;
end
```