WPROWADZENIE

W ramach tego projektu należy zrealizować i przetestować implementację procedury numerycznej rozwiązującej układ równań liniowych (1) *metodą Gaussa-Seidla* oraz *metodą SOR* (ang. *Successive Over-Relaxation*), a następnie otrzymać optymalne wartości parametru relaksacyjnego ω w metodzie SOR.

Zaczynamy od układu równań liniowych

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n = b_1 \\ \vdots \\ a_{n1}x_1 + a_{i2}x_2 + \ldots + a_{nn}x_n = b_n \end{cases} \tag{1}$$

Macierz $A = [a_{ij}]$ powyższego układu powinna spełniać *warunek dominującej przekątnej*, aby iteracje były zbieżne do rozwiązania. Warunek ten oznacza, że wartość bezwzględna każdego elementu na przekątnej, $|a_{ii}|$, jest większa od sumy wartości bezwzględnych pozostałych elementów w wierszu:

$$|a_{ii}| > \sum_{j=1,\, j\neq i}^{n} |a_{ij}| \quad \text{dla} \quad i = 1, 2, \ldots, n. \tag{2}$$

Metody iteracyjne często analizowane są i opisywane w języku macierzy poprzez rozbicie macierzy A na sumę trójkątnej dolnej L, diagonalnej D, oraz trójkątnej górnej U. Zapiszmy zatem macierz A w postaci sumy:

$$A = L + D + U$$
$$= \begin{bmatrix} 0 & 0 & \cdots & 0 & 0 \\ a_{21} & 0 & & \vdots & \vdots \\ a_{31} & a_{32} & \ddots & \vdots & \vdots \\ \vdots & \vdots & \ddots & 0 & 0 \\ a_{n1} & a_{n2} & \cdots & a_{n,n-1} & 0 \end{bmatrix} + \begin{bmatrix} a_{11} & 0 & \cdots & 0 & 0 \\ 0 & a_{22} & & \vdots & \vdots \\ 0 & 0 & \ddots & \vdots & \vdots \\ \vdots & \vdots & \ddots & a_{n-1n-1} & 0 \\ 0 & 0 & \cdots & 0 & a_{nn} \end{bmatrix} + \begin{bmatrix} 0 & a_{12} & \cdots & & a_{1n} \\ 0 & 0 & a_{23} & \vdots & a_{2n} \\ 0 & 0 & \ddots & \vdots & \vdots \\ \vdots & \vdots & \ddots & 0 & a_{n-1n} \\ 0 & 0 & \cdots & 0 & 0 \end{bmatrix} \tag{3}$$

i teraz metodę Jacobiego można macierzowo wyrazić następująco

$$\boldsymbol{x}^{(k)} = \begin{bmatrix} x_1^{(k)} \\ \vdots \\ x_n^{(k)} \end{bmatrix} \rightarrow \boldsymbol{x}^{(k+1)} = \begin{bmatrix} x_1^{(k+1)} \\ \vdots \\ x_n^{(k+1)} \end{bmatrix}, \quad \text{gdzie} \quad \boldsymbol{x}^{(k+1)} = D^{-1}(b - (L+U)\boldsymbol{x}^{(k)}. \tag{4}$$

Oznacza to, że na poszczególne składowe wektora (kolumny) $\boldsymbol{x}^{(k+1)}$ otrzymujemy następujące wzory dla metody Jacobiego:

$$x_i^{(k+1)} = \left(b_i - \sum_{j\neq i} a_{ij} x_j^{(k)} \right) / a_{ii} \quad \text{dla} \quad i = 1, \ldots, n \tag{5}$$

METODA GAUSSA–SEIDLA. W metodzie Gaussa–Seidla stosujemy modyfikację metody Jacobiego. W przypadku metody Jacobiego do obliczenia nowego wektora przybliżającego rozwiązanie *wykorzystywany był cały wektor z poprzedniego kroku*. Modyfikacja polega na tym, że po prawej

stronnie wzory (5) dla $i>1$ wykorzystujemy już nowe wartości $x_j^{(k+1)}$ dla indeksów $1\le j<i$. Zatem iteracje w metodzie Gaussa–Seidla wyglądają następująco:

$$x_i^{(k+1)}=\left(b_i-\sum_{j<i}a_{ij}x_j^{(k+1)}-\sum_{j>i}a_{ij}x_j^{(k)}\right)/a_{ii}\ \text{ dla }\ i=1,\ldots,n. \tag{6}$$

METODA SOR. Metoda ta może być traktowana jako uogólnienie metody Gaussa–Seidla poprzez wprowadzenie pewnego dodatniego parametru $\omega>0$. Wychodzimy od równania (1), czyli w zapisie macierzowym $Ax=b$, mnożymy przez parametr ω i wykorzystujemy rozkład $A=L+D+U$, a następnie stosujemy kilka prostych przekształceń:

$$Ax=b\ \Rightarrow\ \omega(L+D+U)x=\omega b\ \Rightarrow\ (\omega L+\omega D+\omega U)x=\omega b\ \Rightarrow (D+\omega L+(\omega-1)D+\omega U)x=\omega b$$
$$\Rightarrow\ (D+\omega L)x=\omega b-\big((\omega-1)D+\omega U\big)x.$$

Ostatnia równość jest podstawą do metody iteracyjnej

$$(D+\omega L)\boldsymbol{x}^{(k+1)}=\omega b-\big((\omega-1)D+\omega U\big)\boldsymbol{x}^{(k)}. \tag{7}$$

Korzystając z tego, że po lewej stronie macierz $D+\omega L$ jest trójkątna dolna, po prawej $(\omega-1)D+\omega U$ jest trójkątna górna, możemy rozpisać tę równość na kolejne składowe dla $i=1,\ldots,n$:

$$\omega\sum_{j<i}a_{ij}x_j^{(k+1)}+a_{ii}x_i^{(k+1)}=\omega b_i-(\omega-1)a_{ii}x_i^{(k)}-\omega\sum_{j>i}a_{ij}x_j^{(k)},$$

czyli po wyliczeniu $x_i^{(k+1)}$ uzyskamy

$$x_i^{(k+1)}=(1-\omega)x_i^{(k)}+\frac{\omega}{a_{ii}}\left(b_i-\sum_{j<i}a_{ij}x_j^{(k+1)}-\sum_{j>i}a_{ij}x_j^{(k)}\right)\ \text{ dla }\ i=1,\ldots,n,\ k=0,1,2,\ldots \tag{8}$$

Wybór optymalnego parametru $\omega>0$ nie jest automatyczny i zależy od właściwości macierzy $A=[a_{ij}]$.

PRZERYWANIE ITERACJI. W przypadku metod iteracyjnych musimy w którymś momencie przerwać iteracje. Zazwyczaj jest to w przypadku przekroczenia pewnej zadanej maksymalnej liczby iteracji lub przy osiągnięciu zadowalającej dokładności określonej poprzez zadany błąd względny $\varepsilon>0$ zdefiniowany następująco

$$\frac{\|x^{(k+1)}-x^{(k)}\|}{\|x^{(k)}\|}\le\varepsilon,$$

czyli

$$\|x^{(k+1)}-x^{(k)}\|\le\ \varepsilon\,\|x^{(k)}\|. \tag{9}$$

Dla wektorów $x=[x_1,\ldots,x_n]^T$ stosujemy najczęściej jedną z trzech równoważnych norm

$$\|x\|_1=|x_1|+\ldots+|x_n|,\quad \|x\|_2=\sqrt{x_1^2+\ldots+x_n^2},\quad \|x\|_\infty=\max\{|x_1|,\ldots,|x_n|\}. \tag{10}$$

ZADANIA DO WYKONANIA W RAMACH PROJEKTU

1) Należy napisać program w języku C/C++ realizujący **metodę Gaussa–Seidla** oraz **metodę SOR**. Powinien on zawierać następujące elementy.

(i) Wczytanie z pliku macierzy kwadratowej $A=[a_{ij}]\in\mathbb{R}^{n\times n}$ oraz wektora prawych stron $b=[b_i]\in\mathbb{R}^{n\times 1}$ do tablic $n\times n$ oraz n. Mogą to być zwykła tablice języka C/C++ lub struktury `matrix` i `vector` biblioteki GSL – ale funkcje biblioteki nie będą wykorzystywana w tym projekcie.

(ii) Sprawdzenie czy macierz A jest dominująca przekątniowo. Jeżeli ten warunek nie będzie spełniony, to program wypisuje stosowny komunikat i kończy działanie.

(iii) Wypisywanie na ekranie danych z iteracji metody Gaussa–Seidla i metody SOR – ale nie wszystkich, tylko na przykład co piątej.

(iv) Ustaloną maksymalną liczbę iteracji obliczających $\boldsymbol{x}^{(k)}\to\boldsymbol{x}^{(k+1)}$ zadawaną jako parametr oraz względną tolerancję błędu, *eps*.

(v) Iteracje powinny zatrzymać się w jednym z dwóch przypadków: osiągnięcie zadanej dokładności lub przekroczenie zadanej maksymalnej liczby iteracji:

$$\|\boldsymbol{x}^{(k+1)}-\boldsymbol{x}^{(k)}\|\le eps\cdot\|\boldsymbol{x}^{(k)}\| \quad \text{lub} \quad k>kmax, \tag{11}$$

gdzie *eps* (ang. ***reltolerr*** tolerancja względnego błędu) oraz ***kmax*** są zadanymi parametrami, a norma $\|\cdot\|$ jest jedną z trzech typowych norm – patrz (10). W programie użyć normy $\|\cdot\|_2$. Sprawdzić wartości tolerancji względnego błędu równe *eps* = 0.1, 0.01, 0.001, 0.0001 dla macierzy poniższego układu przy maksymalnej liczby iteracji co najmniej 200.

$$\begin{bmatrix} 6 & 1 & 3.1 & 0.5 \\ 2 & -5 & 0 & -1 \\ -0.7 & 1.5 & 3 & 0.1 \\ 0 & -2 & 1.2 & 4 \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix}=\begin{bmatrix} -1 \\ 2 \\ 4 \\ 3 \end{bmatrix}.$$

Rozwiązanie wynosi $x=[-0.8186,-0.7157,1.5021,-0.0585]^T$.

2) Przetestować program na przygotowanych przez siebie dwóch przykładach układów równań o rozmiarach $n=10$ i $n=20$ dla których znamy rozwiązanie. Przykładowe rozwiązanie można przygotować w następujący sposób. Zadajemy macierz $A=[a_{ij}]\in\mathbb{R}^{10\times 10}$, dbając tylko o to, aby był spełniony warunek dominującej przekątnej (2). Teraz ustalamy jakiś wektor rozwiązań, na przykład

$$\boldsymbol{x}=[x_1,\ldots,x_{10}]^T=[1,2,3,4,5,6,7,8,9,10]^T,$$

a następnie wyliczamy jaki powinien być wektor prawych stron układu (1), wykonując mnożenie i definiując $b:=Ax$. Tak otrzymany wektor jest prawą stroną, a rozwiązanie znamy, więc można łatwo weryfikować metodę.

3) Iteracje rozpoczynać od zerowego wektora początkowego $\boldsymbol{x}^{(0)}=[0,\ldots,0]^T$.

4) Dla przykładowych macierzy z p. 2) znaleźć metodą *brute force* optymalną wartość parametru relaksacji $\omega > 0$ dla dwóch tolerancji błędu, 0.01 i 0.0001 w następujący sposób. Przyjąć stosunkowo dużą maksymalną liczbę iteracji, np. ***kmax*** = 500 (tak aby iteracje kończyły się przed osiągnięciem tego

maksimum), a następnie dla każdego $\omega = 0.10, 0.15, 0,20, 0.25, 0.30, \ldots, 1.80, 1.85, 1.90.$ obliczyć przy jakiej liczbie iteracji przerwały się obliczenia (osiągnięto przyjęty poziom błędu ***eps***).

5) W sprawozdaniu (pdf) sporządzić wykres liczby iteracji przy których osiągnięto zadaną tolerancję błędu od ω dla każdej macierzy z p. 2) oraz dla dwóch tolerancji błędu z p. 4). Wykres może być w Excelu lub w programie *gnuplot*.