WPROWADZENIE

W ramach tego projektu należy zrealizować i przetestować implementację procedury numerycznej rozwiązującej układ równań liniowych

$$\begin{bmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{n1} & \cdots & a_{nn} \end{bmatrix} \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} b_1 \\ \vdots \\ b_n \end{bmatrix}, \tag{1}$$

iteracyjną *metodą Jacobiego*. 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:

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

Równanie numer *i* z układu liniowego (1) ma postać

$$\sum_{j=1}^{n} a_{ij} x_j = b_i,$$

skąd $a_{ii} x_i + \sum_{j\neq i} a_{ij} x_j = b_i$. Metoda Jacobiego powstaje przez wyliczenie z tego równania niewiadomej x_i:

$$x_i = \left(b_i - \sum_{j\neq i} a_{ij} x_j \right) / a_{ii}, \tag{3}$$

oraz przyjęcie, że wartości po prawej stronie powyższej równości pochodzą z poprzedniej iteracji (k-tej), a po lewej są nowymi wartościami. Prowadzi to do iteracji $\boldsymbol{x}^{(k)} \to \boldsymbol{x}^{(k+1)}$:

$$x_i^{(k+1)} = \left(b_i - \sum_{j\neq i} a_{ij} x_j^{(k)} \right) / a_{ii} \quad \text{dla} \quad i = 1, \ldots, n. \tag{4}$$

W ten sposób otrzymujemy ciąg wektorów $\boldsymbol{x}^{(0)}, \boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}, \boldsymbol{x}^{(3)}, \ldots$

$$\boldsymbol{x}^{(k)} = \begin{bmatrix} x_1^{(k)} \\ \vdots \\ x_1^{(k)} \end{bmatrix}, \quad k = 0, 1, 2, 3, \ldots \tag{5}$$

który chcemy, aby zmierzał do rozwiązania układu (1). Aby jednak rozpocząć iteracje należy wybrać wektor początkowy $\boldsymbol{x}^{(0)} = [x_1^{(0)}, \ldots, x_n^{(0)}]^T$. Okazuje się jednak, że jeżeli macierz jest dominująca przekątniowo – warunek (2), to ciąg przybliżeń $(\boldsymbol{x}^{(k)})_{k=0}^{\infty}$ jest zawsze zbieżny do rozwiązania układu (1) **niezależnie od wyboru wektora startowego**. Dlatego często wybiera się po prostu wektor początkowy jako wektor zerowy

$$\boldsymbol{x}^{(0)} = \begin{bmatrix} x_1^{(0)} \\ \vdots \\ x_n^{(0)} \end{bmatrix} = \begin{bmatrix} 0 \\ \vdots \\ 0 \end{bmatrix}. \tag{6}$$

ZADANIA DO WYKONANIA W RAMACH PROJEKTU

1) Należy napisać program w języku C/C++ realizujący procedurę (4). 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 – ale nie wszystkich, tylko na przykład co dziesiątej.

(iv) Liczbę iteracji zadajemy jako parametr. Na ogół liczba ta przekracza 100.

2) Przetestować program na przygotowanych przez siebie dwóch przykładach układów równań o rozmiarach $n = 5$ i $n = 10$ dla których znamy rozwiązanie.

3) Sprawdzić jak iteracje zależą od wyboru wektora początkowego $\boldsymbol{x}^{(0)}$. W tym celu rozwiązać przypadek $n = 5$ wybierając trzy różne wektory początkowe i porównać wyniki dla tej samej liczby iteracji.

4) Znaleźć macierz odwrotną do macierzy

$$A = \begin{bmatrix} 4 & 2/7 & 1 & 1/3 & 1/6 & 1/5 \\ 1/3 & -5 & 1/5 & 3/2 & 2/7 & 1/8 \\ 1/9 & 1/6 & -3 & -1 & 1/6 & 2/7 \\ 1/5 & 1/2 & -1 & 8 & 2 & 1/3 \\ 1/8 & 1/3 & 3/10 & 1/8 & 3 & 1/4 \\ 2/7 & 2 & 1/6 & 1/5 & -1 & 6 \end{bmatrix}. \tag{7}$$

W tym celu rozwiązać układy równań $Ax = e_i$ dla $i = 1, \ldots, n,$ metoda Jacobiego, gdzie prawa strona wynosi

$$e_i = \begin{bmatrix} 0 \\ \vdots \\ 0 \\ 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}, \tag{8}$$

tzn. jedynka jest na *i*-tej pozycji, a na pozostałych pozycjach są zera. Rozwiązania dla $i = 1, 2, \ldots, n$ będą kolejne kolumny macierzy odwrotnej A^{-1}. Tak uzyskaną macierz pomnożyć przez macierz wyjściową A i wynik zapisać do pliku. Powinna wyjść macierz „zbliżona" do macierzy jednostkowej. Macierz odwrotna do macierzy (7) (elementy z dokładnością do czterech miejsc po przecinku):

$$A^{-1} = \begin{bmatrix} 0,2484 & 0,0107 & 0,0812 & -0,0016 & -0,0221 & -0,0114 \\ 0,0150 & -0,1955 & -0,0193 & 0,0336 & -0,0026 & 0,0027 \\ 0,0084 & -0,0047 & -0,3119 & -0,0396 & 0,0486 & 0,0148 \\ -0,0025 & 0,0046 & -0,0482 & -0,0776 & -0,0776 & -0,0011 \\ -0,0112 & 0,0159 & 0,0305 & 0.3281 & 0,03281 & -0,0149 \\ -0,0188 & 0,0673 & 0,0179 & 0,0578 & 0,0578 & 0,1634 \end{bmatrix} \quad (9)$$