**17. Решение систем методом квадратного корня. Укажите область применения метода, погрешности вычисления.**

Метод квадратного корня (метод Холецкого) применяется для решения систем линейных уравнений СЛАУ.

Если матрица системы является симметричной и положительно определенной (для положительно определенной матрицы необходимы и достаточны неравенства Сильвестра всех её главных миноров:

), то для решения системы применяют метод квадратного корня. В основе метода лежит алгоритм специального -разложения матрицы , в результате чего она приводится к виду . Если разложение получено, то, как и в методе -разложения, решение системы сводится к последовательному решению двух систем с треугольными матрицами: и . Для нахождения коэффициентов матрицы неизвестные коэффициенты матрицы приравнивают соответствующим элементам матрицы . Затем последовательно находят требуемые коэффициенты по формулам:

……………………………………………………………………………………

Исходный код ниже:

|  |
| --- |
| **package** Selivan;  **import** java.util.Scanner;  **public** **class** S {  **public** **static** **void** main(String[] args) {  Scanner scan = **new** Scanner(System.***in***);  System.***out***.print("Введите размер матрицы: ");  **int** n = scan.nextInt();  **double**[][] a = **new** **double**[n][n];  **double**[] b = **new** **double**[n];  **for** (**int** i = 0; i < n; i++) {  **for** (**int** j = 0; j < n; j++) {  System.***out***.print("Введите a["+i+"]"+"["+j+"]: ");  a[i][j] = scan.nextDouble();  }  System.***out***.print("Введите b["+i+"]: ");  b[i] = scan.nextDouble();  }  **double**[] x = **new** **double**[n];  **double**[][] L = **new** **double**[n][n];  L[0][0] = Math.*sqrt*(Math.*abs*(a[0][0]));  **for** (**int** j = 1; j < n; j++) {  L[0][j] = a[0][j] / (L[0][0]);}  **for** (**int** i = 1; i < n; i++) {  **double** sum = 0;  **for** (**int** k = 0; k <= i - 1; k++) {  sum += L[k][i] \* L[k][i];}  L[i][i] = Math.*sqrt*(Math.*abs*(a[i][i] - sum));  **double** l = 1 / (L[i][i]);  **for** (**int** j = i+1 ; j < n; j++) {  **double** sum1 = 0;  **for** (**int** k = 0; k <= i - 1; k++) {  sum1 += L[k][i] \* L[k][j];}  L[i][j] = l \* (a[i][j] - sum1);}}  **double**[] y = **new** **double**[n];  y[0] = b[0] / (L[0][0]);  **for** (**int** i = 1; i < n; i++) {  **double** sum = 0;  **for** (**int** j = 0; j <= i - 1; j++) {  sum += L[j][i] \* y[j];}  y[i] = (b[i] - sum) / (L[i][i]);}  x[n-1] = y[n-1] / L[n-1][n-1];  **for** (**int** i = n - 1; i >= 0; i--) {  **double** sum = 0;  **for** (**int** k = i+1 ; k < n; k++) {  sum += L[i][k] \* x[k];}  x[i] = (y[i] - sum) / L[i][i];}  **for** (**int** i = 0; i < n; i++) {  System.***out***.printf("x%d = %f\n", i, x[i]);  }  }  } |