**12. Решение систем уравнений методом Зейделя. Погрешность вычисления.**

Метод Зейделя - это численный метод решения системы линейных уравнений вида с заданной точностью . Данный метод представляет собой некоторую модификацию метода простой итерации. В методе простой итерации на -ой итерации значения , вычисляются подстановкой в правую часть вычисленных на предыдущей итерации значений*.* В методе Зейделя при вычислении используются значения ,, , уже найденные на - ой итерации, а не , , …, , как в методе простой итерации, т.е. - е приближение строится следующим образом:

(1)

Эти формулы являются **расчетными формулами метода Зейделя.**

Введем нижнюю и верхнюю треугольные матрицы:

,

Матричная запись расчетных формул (1) имеет вид:

Так как , точное решение исходной системы удовлетворяет равенству:.

**Сходимость метода Зейделя:** Достаточным условием сходимости метода Зейделя является выполнение неравенства: *.* (2)

Неравенство (2) означает, что для сходимости метода Зейделя достаточно, чтобы максимальный по модулю элемент матрицы **B** был меньше единицы. Если выполнено условие (2), то справедлива следующая апостериорная оценка погрешности:

(3) где *-* максимальный элемент матрицы , 2- максимальный элемент матриц .

Правую часть оценки (3) легко вычислить после нахождения очередного приближения.

Исходный код ниже:

|  |
| --- |
| **package** Selivan;  **import** java.util.Scanner;  **public** **class** z {  **public** **final** **static** **double** ***eps*** = 0.001;  **public** **static** **double**[][] *A*;  **public** **static** **double**[] *B*;  **public** **static** **int** *n*;  **public** **static** **void** main(String[] args) {    *n* = 3;  *A* = **new** **double**[][] {{17,6, 1},{-4,-8,-1},{-2,-3,16}};  *B* = **new** **double**[] {15,22,44};    *Normalization*();    **for**(**int** i = 0; i < *n*; i++) {  **for**(**int** j = 0; j < *n*; j++) {  System.***out***.print(*A*[i][j]+" ");  }  System.***out***.println(*B*[i]);  }    **double**[] nextB = **new** **double**[*n*];  **double**[] lastB = **new** **double**[*n*];  nextB = *B*;  lastB = *B*;    **if** (*CheckNorm*()) {    **do** {  lastB = nextB;  nextB = *Step*(lastB);  } **while**(*Measure*(lastB,nextB)>=***eps***) ;    **for**(**int** i = 0; i < *n*; i++) {  System.***out***.printf("x%d = %f" + System.*lineSeparator*(),i+1,nextB[i]);  }  }  }    **private** **static** **void** Normalization() {  **for**(**int** i = 0; i<*n*; i++) {  **if**(*A*[i][i]!=0) {  **double** aii = *A*[i][i];  **for**(**int** j = 0; j<*n*; j++) {  *A*[i][j] = *A*[i][j] / aii;  }  *B*[i] = *B*[i]/aii;  *A*[i][i] = 0;  }**else** {  System.***out***.println("На главной диагонали не могуть быть нули.");  **break**;  }  }  }    **private** **static** **boolean** CheckNorm() {  **for**(**int** i = 0; i<*n*; i++) {  **double** sum = 0;  **for**(**int** j = 0; j<*n*; j++) {  sum += *A*[i][j];  }  **if**(sum>1) {  System.***out***.println("Достаточное условие сходимости метода простых итераций НЕ выполняется");  System.***out***.println(sum+" > 1");  **return** **false**;  }  }  **return** **true**;  }  **public** **static** **double**[] Step(**double**[] X) {  **double**[] nextX = **new** **double**[*n*];  **for**(**int** i = 0; i < *n*; i++) {  **for**(**int** j = 0; j < *n*; j++) {  nextX[i]+=*A*[i][j]\* (j<=i ? nextX[j]:X[j]);  }  nextX[i] =*B*[i] - nextX[i];  }  **return** nextX;  }    **private** **static** **double** Measure(**double**[] lastX, **double**[] X) {  **double** sum = 0;  **for**(**int** i = 0; i < *n*; i++) {  sum+= Math.*abs*(X[i]-lastX[i]);  }  **return** sum;  }  } |