БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

Факультет прикладной математики и информатики

Лабораторная работа №3

Итерационные методы решения систем линейных алгебраических уравнений

Вариант 1

Выполнил:

Черепенников Роман

2 курс 8 группа

Преподаватель:

Радкевич Елена Владимировна

Содержание

Постановка задачи

1. Методом Якоби найти решение СЛАУ с точностью ε = 10^{-5}.
2. Методом Гаусса-Зейделя найти решение СЛАУ с точностью ε = 10^{-5}.

Алгоритм решения

Описание метода нахождения решений системы линейных алгебраических уравнений методом Якоби

Метод Якоби — один из методов приведения системы матрицы к виду, удобному для итерации($x^{k+1} = Bx^k + g$): из 1-го уравнения матрицы выражаем неизвестное x_1, из 2-го выражаем неизвестное x_2 и т.д.

Результатом служит матрица B, в которой на главной диагонали находятся нулевые элементы, а все остальные вычисляются по формуле:

$$b_{ij} = -a_{ij} / a_{ii}; \qquad i,j=1, 2..., n$$

Элементы (компоненты) вектора g вычисляются по следующей формуле:

$$g_i = b_i / a_{ii}; \quad i=1, 2,..., n$$

Расчетная формула метода простой итерации(в координатной форме):

$$x_i^{(n+1)} = b_{i1}x_1^{(n)} +... + b_{i\,i-1}\, x_{i-1}^{(n)} + b_{i\,i+1}\, x_{i+1}^{(n)} +...+b_{in}x_n^{(n)} + g_i$$

Критерий окончания в методе Якоби:

$$\|x^{(n+1)}-x^{(n)}\| < \varepsilon$$

Теорема(о сходимости метода Якоби)ю Метод Якоби для СЛАУ сходится, если элементы матрицы A a_{ij} i, j = 1..n удовлетворяют условию:

$$\sum_{j=1, j\neq i}^{n} \left|\frac{a_{ij}}{a_{ii}}\right| < 1; \; j = 1..n$$

Количество операций вычислим по формуле:

$$k \geq \frac{\lg(\varepsilon * \frac{1-||B||}{||B||})}{\lg(||B||)} - 1$$

Описание метода нахождения решений системы линейных алгебраических уравнений методом Гаусса-Зейделя

Метод Гаусса-Зейделя является модификацией метода Якоби. В данном методе для нахождения значения i-го неизвестного на каждой итерации используются значения предыдущих неизвестных, уже найденные на данной итерации. Общую формулу определения i-го неизвестного на k-й итерации для системы n уравнений можно записать так:

$$x_i^{(n+1)} = b_{i1}x_1^{(n+1)} +... + b_{i\,i-1}\, x_{i-1}^{(n+1)} + b_{i\,i+1}\, x_{i+1}^{(n)} +...+b_{in}x_n^{(n)} + g$$

Листинг программы

```
def jacobi(A, b, eps):
    n = A.shape[0]
    B = np.zeros(A.shape)
    for i in range(n):
        b[i][0] /= A[i][i]
        for j in range(n):
            if i != j:
                B[i][j] = -A[i][j] / A[i][i]
    x_new = b
    x_old = np.zeros(b.shape)
    B_norm = np.linalg.norm(B)
    b_norm = np.linalg.norm(b)
    print("||B|| = ", B_norm, sep=" ")
    print("||b|| = ", b_norm, sep=" ")
    real_iterations = 0
    aprior_iterations = ceil(((log10(eps) + log10(1 - B_norm) -
log10(b_norm)) / log10(B_norm)) - 1)
    while np.linalg.norm(x_old - x_new) >= eps:
        x_old = x_new
        x_new = np.dot(B, x_old) + b
        real_iterations += 1
    print("Априорное число итераций:", aprior_iterations, sep=" ")
    print("Реальное число итераций:", real_iterations, sep=" ")
    return x_new


def gauss_seidel(A, b, eps):
    n = A.shape[0]
    H = np.zeros(A.shape)
    F = np.zeros(A.shape)
    for i in range(n):
        b[i][0] /= A[i][i]
        for j in range(n):
            if i > j:
                H[i][j] = -A[i][j] / A[i][i]
            if j>i:
                F[i][j] = -A[i][j] / A[i][i]
    x_new = b
    x_old = np.zeros(b.shape)
    real_iterations = 0
    while np.linalg.norm(x_old - x_new) >= eps:
        x_old = x_new
        x_new = np.dot(H, x_new) + np.dot(F, x_old) + b
        real_iterations += 1
    print("Реальное число итераций:", real_iterations, sep=" ")
    return x_new

A = np.array([
    [0.6444, 0.0000, -0.1683, 0.1184, 0.1973],
    [-0.0395, 0.4208, 0.0000, -0.0802, 0.0263],
    [0.0132, -0.1184, 0.7627, 0.0145, 0.0460],
    [0.0395, 0.0000, -0.0960, 0.7627, 0.0000],
    [0.0263, -0.0395, 0.1907, -0.0158, 0.5523]
])
b = np.array([
    [1.2677],
    [1.6819],
```

```
    [-2.3657],
    [-6.5369],
    [2.8351]
])

print("Метод Якоби")
x = jacobi(A, b.copy(), 1E-5)
print("Решение ситсемы:")
print(x.tolist())
print()
print("Метод Гаусса-Зейделя")
x = gauss_seidel(A, b.copy(), 1E-5)
print("Решение ситсемы:")
print(x.tolist())
```

Входные данные

A					b
0,6444	0,0000	-0,1683	0,1184	0,1973	1.2677
-0,0395	0,4208	0,0000	-0,0802	0,0263	1.6819
0,0132	-0,1184	0,7627	0,0145	0,0460	-2.3657
0,0395	0,0000	-0,0960	0,7627	0,0000	-6.5369
0,0263	-0,0395	0,1907	-0,0158	0,5523	2.8351

Вывод программы

Метод Якоби

||B|| = 0.6473796747609915

||b|| = 11.369876089029146

Априорное число итераций: 34

Реальное число итераций: 10

Решение ситсемы:

[[0.9982167096784872], [1.9998662172311779], [-2.999759102286751], [-9.000008566860227], [6.007052736929327]]

Метод Гаусса-Зейделя

Реальное число итераций: 10

Решение ситсемы:

[[0.9982167096784872], [1.9998662172311779], [-2.999759102286751], [-9.000008566860227], [6.007052736929327]]

Вывод

При большом числе неизвестных для решения СЛАУ удобнее пользоваться методом простой итерации.

Сравнив априорную оценку и реальное количество итераций в методах Якоби и Гаусса-Зейделя, можно сделать вывод, что методы сходятся значительно быстрее (за значительно меньшее количество итераций).

Метод Якоби и метод Гаусса-Зейделя позволяют находить решение системы с определенной точностью, что сказывается на векторах невязки.