Министерство образования Республики Беларусь

Учреждение образования “Белорусский государственный университет информатики и радиоэлектроники”

Факультет компьютерных систем и сетей

Кафедра информатики

Дисциплина Методы численного анализа

**Отчет**

к лабораторной работе №1

**Системы линейных алгебраический уравнений**

*Вариант 26*

Выполнил:

студент гр.453504

Сухов Н.Ю.

Минск, 2016

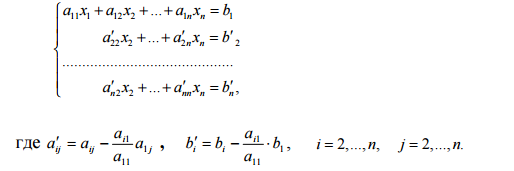
**Введение**

Методы решения систем линейных уравнений делятся на две группы. Это так называемые прямые методы и итерационные методы. К прямым методам относятся метод квадратного корня, метод исключения Гаусса и его модификации.

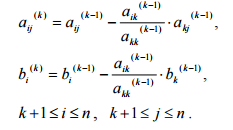
**Метод Гаусса** является одним из самых распространенных методов решения систем линейных уравнений, по этой причине для выполнения данной работы использовался именно он. Этот метод, который также называют методом последовательного исключения неизвестных, известен в различных вариантах. Он состоит из прямого и обратного ходов.

Вычисления с помощью метода Гаусса на этапе прямого хода заключаются в последовательном исключении неизвестных из системы для преобразования ее к равносильной системе с верхней треугольной (трапециевидной) матрицей. Вычисления значений неизвестных производят на этапе обратного хода.

Пусть система содержит n уравнений с n неизвестными. На первом шаге надо исключить неизвестное из уравнений с номерами i = 2, 3, …, n. Пусть элемент 0 . Он называется ведущим элементом первого шага. Вычтем последовательно из второго, третьего, …, n-го уравнений системы первое уравнение, умноженное соответственно на . Это позволит обратить в нуль коэффициенты при во всех уравнениях, кроме первого. В результате получим равносильную систему:



Продолжим дальше, предполагая, что 10.PNG и исключим неизвестное из уравнений, начиная с третьего и так далее.. На k-ом шаге предполагаем, что ведущий элемент k-го шага 11.PNG , и, продолжая процесс, получаем формулы для преобразования элементов матрицы на данном шаге:



Это так называемый прямой ход метода Гаусса. Если на каком-то шаге получается невыполнимое равенство, это означает, что система не имеет решений. В противном случае после (n-1)-го шага исключений можем получить треугольную систему, из последнего уравнения которой мы найдем . Подставляя его в предпоследнее уравнение, найдем . И так далее. Этот процесс называется обратным ходом метода Гаусса.

Возможно также ситуация, когда в процессе прямого хода получается трапециевидная система, где в последнем оставшемся уравнении имеется более одной переменной. В этом случае придаем всем оставшимся в последнем уравнении неизвестным, кроме первого, произвольные значения , затем из последнего уравнения через выражается оставшееся неизвестное и подставляется в предыдущие уравнения. Постепенно все неизвестные выражаются через параметры , которые могут иметь произвольные числовые значения. Таким образом, в последнем случае система имеет бесконечно много решений.

Замечание. При реализации метода Гаусса на каждом шаге производится деление на соответствующий ведущий элемент, поэтому предполагается, что эти элементы не должны быть равными нулю. В противном случае проводится перенумерация уравнений и неизвестных.

**Метод ведущего (разрешающего) элемента**

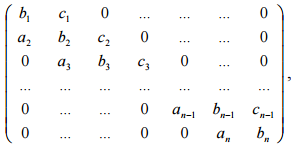
Метод ведущего элемента представляет собой модификацию метода Гаусса. Выбираем в матрице А любой не равный нулю элемент , который называют ведущим элементом. Строка и столбец, в котором находится этот элемент, называют рабочими. То есть рабочими будут l-й столбец и k-ая строка. Исключаем неизвестное из всех уравнений, кроме k-го. В матрице новой системы l-й столбец принимает вид:

1.PNG

Помечаем рабочие строку и столбец. В последующих действиях они больше не могут являться рабочими. Опять ищем ведущий элемент и исключаем соответствующую переменную. Так продолжаем до тех пор, пока не получится уравнение с одним неизвестным. Из этого уравнения находим значение неизвестного и подставляем его в предыдущие уравнения для обратного хода. Продолжаем процесс, пока не получим значения всех неизвестных.

**Метод прогонки**

Метод прогонки также является модификацией метода Гаусса для систем специального вида, так называемых трехдиагональных систем. К необходимости решения такого рода систем приводят, в частности, задачи построения кубических сплайнов и разностные схемы решения краевых задач для дифференциальных уравнений. Трехдиагональной системой называют систему линейных уравнений с матрицей вида:

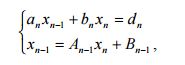


Суть метода прогонки заключается в построении рекуррентной последовательности для нахождения прогоночных коэффициентов Ai и Bi. При этом каждое неизвестное представляется в виде:

3.PNG

Процесс нахождения коэффициентов продолжается для 4.PNG

Для нахождения используем данное уравнение и оставшееся последнее уравнение. Получаем систему



решая которую и находим

Проводя обратный ход метода прогонки, последовательно находим значения неизвестных.

**Метод квадратного корня**

Этот метод применяется при решении систем вида

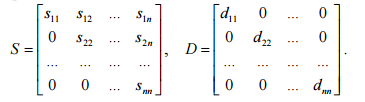
6.PNG

с неособенной симметрической матрицей. Если матрица А не является симметрической, то без предварительного преобразования системы к виду

7.PNG

метод применять нельзя. Однако преобразование системы к указанному выше виду связано с выполнением большого числа дополнительных операций умножения и сложения, число которых намного превосходит число аналогичных операций, необходимых при решении системы с симметрической матрицей по методу квадратного корня. Поэтому выполнять указанное преобразование и затем применять к решению системы метод квадратного корня, как правило не целесообразно.

Пусть матрица А симметрическая. Схема метода квадратного корня строится на идее представления матрицы в виде произведения треугольных и диагональных матриц, а именно: находим такую правую треугольную матрицу S и диагональную матрицу D с элементами ±1 по главной диагонали, чтобы имело место равенство A=S\*DS, где приняты обозначения



Отметим, что метод квадратного корня очень эффективен при решении систем с положительно определенной симметрической матрицей. Такие системы, как правило, возникают при решении задач минимизации положительно определенных квадратичных форм. Главными требованиями к исходной матрице являются симметричность и положительная определенность. Симметричность проверяется простым сравнением соответствующих элементов. Положительная определенность матрицы определяется с помощью критерия Сильвестра.

*Критерий Сильвестра.* Для того чтобы матрица была положительно определенной, необходимо и достаточно, чтобы все главные диагональные миноры матрицы были строго положительны.

**3. Распечатка кода программы**

Код программы для решения системы 1 методом Гаусса:

import numpy

def main():

print("Start working")

A = numpy.array([[3, 1, 3], [-1, 3, 1], [3, -1, 3]], "f")

b = numpy.array([5, 5, 1], "f")

for k in range(len(A) - 1):

for i in range(1 + k, len(A)):

temp = A[i][k]

b[i] = b[i] - temp / A[k][k] \* b[k]

for j in range(k, len(A[i])):

A[i][j] = A[i][j] - temp / A[k][k] \* A[k][j]

x = numpy.zeros(3)

for i in range(len(A) - 1, -1, -1):

x[i] = b[i]

for j in range(len(A) - 1, i, -1):

x[i] -= A[i][j] \* x[j]

x[i] /= A[i][i]

print("\ntriangular matrix A:")

print(A)

print("\nchange vector b:")

print(b)

print("\nresult:")

print(x)

if \_\_name\_\_ == "\_\_main\_\_":

main()

Код программы для решения системы 2 методом Гаусса:

import numpy

def main():

print("Start working")

A = numpy.array([[1, 2, 3], [1, 1, 4], [1, -2, -5]], "f")

b = numpy.array([1, -4, 5], "f")

for k in range(len(A) - 1):

for i in range(1 + k, len(A)):

temp = A[i][k]

b[i] = b[i] - temp / A[k][k] \* b[k]

for j in range(k, len(A[i])):

A[i][j] = A[i][j] - temp / A[k][k] \* A[k][j]

x = numpy.array([0, 0, 0], "f")

for i in range(len(A) - 1, -1, -1):

x[i] = b[i]

for j in range(len(A) - 1, i, -1):

x[i] -= A[i][j] \* x[j]

x[i] /= A[i][i]

print("Triangular matrix A: ")

for i in A:

print(i)

print("\nb: ")

for i in b:

print(i)

print("\nRoots: ")

for i in x:

print(i)

if \_\_name\_\_ == "\_\_main\_\_":

main()

Код программы для решения системы 3 методом Гаусса:

import sympy

def main():

A = sympy.Matrix([[5, 2, -3, 4], [2, 3, -2, 1], [1, -4, 1, 2]])

b = sympy.Matrix([1, 2, 3])

print("\nInitial matrix:")

for i in range(A.rows):

print(A.row(i))

c = A.cols

for k in range(A.rows - 1):

for i in range(1 + k, A.rows):

temp = A[c \* i + k]

b[i] = b[i] - temp / A[c \* k + k] \* b[k]

for j in range(k, A.cols):

A[c \* i + j] -= temp / A[c \* k + k] \* A[c \* k + j]

x = [sympy.Symbol("x" + str(i)) for i in range(1, c + 1)]

for i in range(A.rank() - 1, -1, -1):

x[i] = b[i]

for j in range(A.cols - 1, i, -1):

x[i] -= A[c \* i + j] \* x[j]

x[i] /= A[c \* i + i]

print("\nTriangular matrix A:")

for i in range(A.rows):

print(A.row(i))

print("\nb:")

for i in b:

print(i)

print("\nx:")

for i in x:

print(i)

if \_\_name\_\_ == "\_\_main\_\_":

main()

Здесь количество уравнений меньше количества неизвестных поэтому система будет иметь бесконечно много решений.Так как ранг матрицы системы равен 2 то система будет иметь 2 базисные и 2 свободные переменные. В качестве свободных возьмём x3 и x4.И в этом случае в методе Гаусса мы приходим к трапецивидной системе.

Код программы для однородной системы 4 методом Гаусса:

import sympy

def main():

A = sympy.Matrix([[3, 5], [6, 10]])

b = sympy.Matrix([0, 0])

print("\nInitial matrix:")

for i in range(A.rows):

print(A.row(i))

c = A.cols

for k in range(A.rows - 1):

for i in range(1 + k, A.rows):

temp = A[c \* i + k]

b[i] = b[i] - temp / A[c \* k + k] \* b[k]

for j in range(k, A.cols):

A[c \* i + j] -= temp / A[c \* k + k] \* A[c \* k + j]

x = [sympy.Symbol("x" + str(i)) for i in range(1, c + 1)]

for i in range(A.rank() - 1, -1, -1):

x[i] = b[i]

for j in range(A.cols - 1, i, -1):

x[i] -= A[c \* i + j] \* x[j]

x[i] /= A[c \* i + i]

print("\nTriangular matrix A:")

for i in range(A.rows):

print(A.row(i))

print("\nb:")

for i in b:

print(i)

print("\nx:")

for i in x:

print(i)

if \_\_name\_\_ == "\_\_main\_\_":

main()

Код программы для решения нахождения нетривиальных решений однородной системы 5 методом Гаусса:

from sympy import \*

def GaussianMethod(A, b):

c = A.cols

for k in range(A.rows - 1):

for i in range(1 + k, A.rows):

temp = A[c \* i + k]

b[i] = b[i] - temp / A[c \* k + k] \* b[k]

for j in range(k, A.cols):

A[c \* i + j] -= temp / A[c \* k + k] \* A[c \* k + j]

x = [Symbol("x" + str(i)) for i in range(1, c + 1)]

for i in range(A.rank() - 1, -1, -1):

x[i] = b[i]

for j in range(A.cols - 1, i, -1):

x[i] -= A[c \* i + j] \* x[j]

x[i] /= A[c \* i + i]

print("x:")

for i in x:

print(i)

def main():

a = Symbol("a")

A = Matrix([[a, 5, -4],

[7, 2, -1],

[4, -3, a]])

print("\nInitial matrix:")

for i in range(A.rows):

print(A.row(i))

aValue = solve(Eq(A.det(), 0), a)

print("\na value: ")

print(aValue)

for i in aValue:

Asubs = A.subs(a, i)

b = zeros(1, A.rows)

print("")

print("a = ", i)

GaussianMethod(Asubs, b)

if \_\_name\_\_ == "\_\_main\_\_":

main()

Здесь мы сначала находим параметр **a** при котором система будет нетривиально совместной,приравняв определитель матрицы системы к нулю.Далее найдя значения параметра **a**, подставим его значения в систему и найдём её решения методом Гаусса.

**Вывод:** Метод Гаусса - это классический метод решения СЛАУ. Это метод последовательного исключения переменных,когда с помощью элементарных преобразований система сводится к верхнетреугольному или трапецивидному виду, из которых обратным ходом последовательно находятся неизвестные.Этот метод универсальны по сравнению с методом Крамера и матричным методом,т.к не нужно считать определители и обратные матрицы.Что является трудоёмким даже для систем начиная с n>5.Так же метод является универсальным. Так методы прогонки и квадратного корня более просты в реализации, но применимы только для трёхдиагональных систем и для системы с симметрической матрицей соответственно. Единственным серьёзным недостатком метода Гаусса является необращение в нуль диагональных элементов.Поэтому используются различные модификации метода Гаусса,например,с выбором главного элемента по столбцу или во всей матрице.