

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
АДЫГЕЙСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
Инженерно-физический факультет
Кафедра автоматизированных систем обработки информации и
управления

ОТЧЕТ ПО ПРАКТИКЕ

Программная реализация вычисления матрицы
обратную заданной
Вариант 5

1 курс, группа 1ИВТ1-1

Выполнил:

_____ Д. А. Лиев
«___» _____ 2023 г.

Руководитель:

_____ С. В. Теплоухов
«___» _____ 2023 г.

Майкоп, 2023 г.

1. Введение

1.1. Формулировка цели

Целью данной работы является написание программы для нахождения обратной матрицы методом Гаусса.

1.1.1. Теория

В методе Гаусса (и его модификации, такой как метод Гаусса-Жордана), основной целью является приведение матрицы к улучшенному ступенчатому виду или диагональному виду. Это достигается путем применения элементарных преобразований строк матрицы.

Элементарные преобразования строк включают:

- 1) умножение на число, отличное от нуля;
- 2) прибавление к элементам какой-либо строки или какого-либо столбца;
- 3) перемена местами двух строк матрицы.

Целью применения этих преобразований является создание нулевых элементов под главной диагональю или приведение матрицы к диагональному виду, что облегчает нахождение решений систем линейных уравнений или обратных матриц.

Использование метода Гаусса позволяет решать системы линейных уравнений и находить обратные матрицы путем последовательного применения элементарных преобразований строк матрицы до достижения требуемого вида.

1.1.2. Практика

Рассмотрим шаги по нахождению обратной матрицы методом Гаусса подробно:

- 1) предположим, что у нас есть квадратная матрица A размером $n \times n$, для которой мы хотим найти обратную матрицу. Мы будем работать с расширенной матрицей $[A \mid I]$, где I - единичная матрица размером $n \times n$;
- 2) используем метод Гаусса для приведения матрицы A к верхнетреугольному виду, при этом применяем те же самые элементарные преобразования к матрице I . В результате преобразований матрица $[A \mid I]$ превратится в $[U \mid V]$, где U - верхнетреугольная матрица, а V - преобразованная матрица I ;
- 3) затем мы используем метод Гаусса-Жордана для дальнейшего преобразования матрицы U . Нашей целью является приведение матрицы U к диагональному виду, при этом применяем те же самые элементарные преобразования к матрице V . В результате преобразований матрица $[U \mid V]$ превратится в $[I \mid A^{-1}]$, где A^{-1} - обратная матрица, которую мы ищем;

- 4) после завершения преобразований мы получаем обратную матрицу A^{-1} , которая находится в правой части расширенной матрицы. Матрица A^{-1} является обратной матрицей для исходной матрицы A .

Важно отметить, что метод Гаусса-Жордана позволяет найти обратную матрицу только в том случае, если исходная матрица A обратима. Если матрица необратима, то обратной матрицы не существует.

Кроме того, при реализации метода Гаусса-Жордана в вычислительных программах необходимо учитывать особенности работы с плавающей запятой и возможные проблемы с точностью. При нахождении обратной матрицы методом Гаусса-Жордана следует применять методы численного анализа для учета таких проблем и повышения точности вычислений. Пример квадратной матрицы A для преобразования в обратную (Рис. 1.).

$$A = \begin{pmatrix} 2 & 1 & -1 \\ 5 & 2 & 4 \\ 7 & 3 & 2 \end{pmatrix}$$

Рис. 1. Квадратная матрица A

2. Ход работы

2.1. Программа

Код выполненной программы:

```
1 import numpy as np
2
3 def gauss_inverse(matrix):
4     n = len(matrix)
5     augmented_matrix = np.concatenate((matrix, np.identity(n)), axis=1)
6
7     print("Исходная матрица A:")
8     print(matrix)
9
10    for i in range(n):
11        max_index = i
12        max_value = augmented_matrix[i, i]
13        for j in range(i+1, n):
14            if augmented_matrix[j, i] > max_value:
15                max_index = j
16                max_value = augmented_matrix[j, i]
17        augmented_matrix[[i, max_index]] = augmented_matrix[[max_index, i]]
```

```

18
19     if augmented_matrix[i, i] == 0:
20         print("Матрица A является вырожденной. Обратной матрицы не
21             ↳ существует.")
22         return None
23
24     augmented_matrix[i] = augmented_matrix[i] / augmented_matrix[i, i]
25
26     print("Преобразование " + str(i+1) + ":")
27     print(np.around(np.concatenate((np.identity(n), augmented_matrix[:,
28         ↳ n:]), axis=1), decimals=decimals))
29
30     for j in range(n):
31         if j != i:
32             augmented_matrix[j] = augmented_matrix[j] -
33                 ↳ augmented_matrix[j, i] * augmented_matrix[i]
34
35     inverse_matrix = augmented_matrix[:, n:]
36
37     return inverse_matrix
38
39 # Выбор режима ввода матрицы
40 mode = input("Выберите способ ввода матрицы:\n1. Сгенерировать случайную
41     ↳ матрицу\n2. Ввести матрицу вручную\nВведите число: ")
42
43 # Генерация случайной матрицы
44 if mode.lower() == "1":
45     # Ввод размерности матрицы с клавиатуры
46     n = int(input("Введите размерность матрицы A: "))
47
48     # Генерация случайной матрицы с целыми числами
49     matrix = np.random.randint(low=1, high=10, size=(n, n))
50
51     print("Сгенерированная матрица A:")
52     print(matrix)
53
54 # Ввод матрицы вручную
55 elif mode.lower() == "2":
56     # Ввод размерности матрицы с клавиатуры
57     n = int(input("Введите размерность матрицы A: "))
58
59     # Ввод элементов матрицы с клавиатуры
60     print("Введите элементы матрицы A:")
61     matrix = np.zeros((n, n), dtype=float)
62     for i in range(n):
63         for j in range(n):

```

```

60         matrix[i, j] = float(input("Элемент [" + str(i+1) + ", " +
        ↪      str(j+1) + "]: "))
61
62     else:
63         print("Некорректный выбор режима.")
64         exit()
65
66     decimals = int(input("Введите количество знаков после запятой для вывода
        ↪      матрицы: "))
67
68     inverse = gauss_inverse(matrix)
69
70     if inverse is not None:
71         print("Обратная матрица A(-1):")
72         print(np.around(inverse, decimals=decimals))

```

2.2. Результат работы программы

Программа работает следующим образом:

- 1) пользователь выбирает способ ввода матрицы: сгенерировать случайную матрицу или ввести матрицу вручную;
- 2) в случае выбора режима "1 (генерирование) программа запрашивает размерность матрицы A и генерирует случайную матрицу с целыми числами;
- 3) в случае выбора режима "2 (ввод вручную) программа запрашивает размерность матрицы A и последовательно запрашивает элементы матрицы от пользователя;
- 4) программа выводит на экран исходную матрицу A;
- 5) программа применяет метод исключения неизвестных Гаусса для преобразования матрицы A в единичную матрицу;
- 6) при каждом преобразовании программа выводит промежуточные результаты, включая преобразованную матрицу с единичной матрицей справа;
- 7) если в процессе преобразования обнаруживается, что матрица A является вырожденной (в строке или столбце только нули), программа выводит сообщение о невозможности вычислить обратную матрицу и завершается;
- 8) в случае успешного преобразования, программа выводит полученную обратную матрицу A^{-1} ;
- 9) пользователю предоставляется возможность указать количество знаков после запятой для округления выводимых чисел;

- 10) конечный результат представляет собой исходную матрицу A , преобразованную матрицу с единичной матрицей справа и полученную обратную матрицу A^{-1} , выводимые на экран.

В качестве примера работы кода была использована матрица A (Рис. 1).
Результат работы программы (Рис. 2.):

```
Выберите способ ввода матрицы:
1. Сгенерировать случайную матрицу
2. Ввести матрицу вручную
Введите число: 2
Введите размерность матрицы A: 3
Введите элементы матрицы A:
Элемент [1, 1]: 2
Элемент [1, 2]: 1
Элемент [1, 3]: -1
Элемент [2, 1]: 5
Элемент [2, 2]: 2
Элемент [2, 3]: 4
Элемент [3, 1]: 7
Элемент [3, 2]: 3
Элемент [3, 3]: 2
Введите количество знаков после запятой для вывода матрицы: 2
Исходная матрица A:
[[ 2.  1. -1.]
 [ 5.  2.  4.]
 [ 7.  3.  2.]]
Преобразование 1:
[[1.  0.  0.  0.  0.  0.14]
 [0.  1.  0.  0.  1.  0. ]
 [0.  0.  1.  1.  0.  0. ]]
Преобразование 2:
[[ 1.  0.  0.  0.  0.  0.14]
 [ 0.  1.  0.  7.  0. -2. ]
 [ 0.  0.  1.  0.  1. -0.71]]
Преобразование 3:
[[ 1.  0.  0. -3.  0.  1.]
 [ 0.  1.  0.  7.  0. -2.]
 [ 0.  0.  1.  1.  1. -1.]]
Обратная матрица A^(-1):
[[ -8. -5.  6.]
 [ 18. 11. -13.]
 [ 1.  1. -1.]]
```

Рис. 2. Результат работы программы