|  |
| --- |
|  |
| МИНОБРНАУКИ РОССИИ |
| Федеральное государственное бюджетное образовательное учреждение высшего образования  **«МИРЭА – Российский технологический университет»**  **РТУ МИРЭА** |

*Институт искусственного интеллекта Кафедра высшей математики*

# Отчёт по лабораторной работе № 1 Вариант 23

**по дисциплине «Численные методы»**

|  |  |
| --- | --- |
| *Выполнил студент группы КМБО-07- 22* | *Баттур Ц.* |
| *Проверил* | *Алексеев А.А.* |

*Москва 2025*

# Задание

Разработать программу , реализующую указанный алгоритм Гаусса-Жордана для систем линейных алгебраических уравнений

с квадратной матрицей А произвольного порядка п. В разрабатываемой программе предусмотреть вывод расширенной матрицы на каждом шаге преобразований для визуализации и контроля алгоритма. С помощью разработанной программы решить поставленную в варианте задачу. Проверить полученный результат прямой подстановкой с вычислением соответствующей невязки

по указанной норме

Программа, реализующая алгоритм Гаусса и решающая систему линейных алгебраических уравнений, представлена в листинге 1:

**# Gaussian elimination with partial pivoting**

**def Gauss(A):**

**augmented = A.copy()**

**pivot\_row = 0**

**pivot\_cols = []**

**for col in range(n\_cols):**

**if pivot\_row >= n\_rows:**

**break**

**max\_row = np.argmax(np.abs(augmented[pivot\_row:, col])) + pivot\_row**

**if np.abs(augmented[max\_row, col]) < 1e-10:**

**continue**

**augmented[[pivot\_row, max\_row]] = augmented[[max\_row, pivot\_row]]**

**pivot\_cols.append(col)**

**for i in range(pivot\_row + 1, n\_rows):**

**factor = augmented[i, col] / augmented[pivot\_row, col]**

**augmented[i] -= factor \* augmented[pivot\_row]**

**pivot\_row += 1**

**rank = len(pivot\_cols)**

**free\_vars = [c for c in range(n\_cols) if c not in pivot\_cols]**

**num\_free = len(free\_vars)**

**solutions = np.zeros((n\_cols, num\_free), dtype=np.float32)**

**for i, fv in enumerate(free\_vars):**

**solutions[fv, i] = 1.0**

**for r in reversed(range(rank)):**

**pc = pivot\_cols[r]**

**row = augmented[r]**

**known = sum(row[c] \* solutions[c, i] for c in range(pc+1, n\_cols))**

**solutions[pc, i] = -known / row[pc]**

**print("Fundamental system of solutions (columns):")**

**print\_matrix(solutions)**

**return solutions**

*Листинг 1*. Программа, находящая решение системы линейных однородных уравнений методом Гаусса.

Данная программа получает на вход матрицу A, после чего выполняет алгоритм Гаусса для приведения её к ступенчатому виду. На основе полученной формы определяется ранг матрицы, выделяются свободные переменные и, с помощью обратной подстановки, вычисляется фундаментальная система решений однородной системы Ax = 0. Результат выводится в виде матрицы, столбцы которой представляют базис ядра матрицы А.

Тестовый пример:

Шаги Решения представлены в Листинге 2:

2.0000000 4.0000000 5.0000000 1.0000000 5.0000000

1.0000000 2.0000000 0.0000000 1.0000000 9.0000000

3.0000000 6.0000000 1.0000000 3.0000000 4.0000000

4.0000000 5.0000000 6.0000000 7.0000000 3.0000000

the matrix after 1 step

4.0000000 5.0000000 6.0000000 7.0000000 3.0000000

0.0000000 0.7500000 -1.5000000 -0.7500000 8.2500000

0.0000000 2.2500000 -3.5000000 -2.2500000 1.7500000

0.0000000 1.5000000 2.0000000 -2.5000000 3.5000000

the matrix after 2 step

4.0000000 5.0000000 6.0000000 7.0000000 3.0000000

0.0000000 2.2500000 -3.5000000 -2.2500000 1.7500000

0.0000000 0.0000000 -0.3333333 0.0000000 7.6666667

0.0000000 0.0000000 4.3333333 -1.0000000 2.3333333

the matrix after 3 step

4.0000000 5.0000000 6.0000000 7.0000000 3.0000000

0.0000000 2.2500000 -3.5000000 -2.2500000 1.7500000

0.0000000 0.0000000 4.3333333 -1.0000000 2.3333333

0.0000000 0.0000000 0.0000000 -0.0769231 7.8461538

the matrix after 4 step

4.0000000 5.0000000 6.0000000 7.0000000 3.0000000

0.0000000 2.2500000 -3.5000000 -2.2500000 1.7500000

0.0000000 0.0000000 4.3333333 -1.0000000 2.3333333

0.0000000 0.0000000 0.0000000 -0.0769231 7.8461538

Листинг 2: Пошаговые Преобразования

-385.0000000

137.0000000

23.0000000

102.0000000

1.0000000

Листинг 3: Вывод

Как мы видим, мы нашли матрицу фундаментальной системы решений для данной матрицы используя метод Гаусса.

И мы проверяем результат, умножая матрицу фундаментальной системы решений на данную матрицу.

res\_test = [[0 for x in range(1)] for y in range(5)]

for i in range(len(original\_test)):

    for j in range(len(solutions[0])):

        for k in range(len(solutions)):

            # resulted matrix

            res\_test[i][j] += original\_test[i][k] \* solutions[k][j]

            #print\_matrix(res)

print\_matrix(res\_test)

Листинг 4: Умножение матрицу фундаментальной системы на исходную матрицу

0.0000000

0.0000000

0.0000000

0.0000000

0.0000000

Листинг 5: Результат умножения

Сейчас находим оценки невязки используя норму Фробениуса .

def frobenius(original\_A, solutions):

  residual = original\_A @ solutions

  fro\_norm = np.linalg.norm(residual, 'fro')

  print(f"\nResidual Frobenius norm: {fro\_norm:.7f}")

frobenius(original\_test, solutions)

Листинг 6: Код , который находит норму Фробениуса

Residual Frobenius norm: 0.0000000

Листинг 7: Оценка ошибок

Для:

Шаги решения представлены в Листинге:

 1.0000000 -5.0000000  5.0000000  1.0000000  5.0000000 -5.0000000 -5.0000000  0.0000000  3.0000000

-4.0000000 19.0000000 -25.0000000 -6.0000000 -16.0000000 17.0000000 15.0000000 -5.0000000 -11.0000000

-3.0000000 20.0000000  5.0000000 32.0000000 -40.0000000 45.0000000 45.0000000 40.0000000 -39.0000000

-4.0000000 19.0000000 -22.0000000 -26.0000000  7.0000000 -7.0000000 17.0000000 -29.0000000  4.0000000

 5.0000000 -29.0000000  1.0000000 20.0000000 23.0000000 -18.0000000 -54.0000000 -7.0000000 -3.0000000

the matrix after 1 step

 5.0000000 -29.0000000  1.0000000 20.0000000 23.0000000 -18.0000000 -54.0000000 -7.0000000 -3.0000000

 0.0000000 -4.2000000 -24.2000000 10.0000000  2.4000000  2.6000000 -28.2000000 -10.6000000 -13.4000000

 0.0000000  2.6000000  5.6000000 44.0000000 -26.2000000 34.2000000 12.6000000 35.8000000 -40.8000000

 0.0000000 -4.2000000 -21.2000000 -10.0000000 25.4000000 -21.4000000 -26.2000000 -34.6000000  1.6000000

 0.0000000  0.8000000  4.8000000 -3.0000000  0.4000000 -1.4000000  5.8000000  1.4000000  3.6000000

the matrix after 2 step

 5.0000000 -29.0000000  1.0000000 20.0000000 23.0000000 -18.0000000 -54.0000000 -7.0000000 -3.0000000

 0.0000000 -4.2000000 -24.2000000 10.0000000  2.4000000  2.6000000 -28.2000000 -10.6000000 -13.4000000

 0.0000000  0.0000000 -9.3809524 50.1904762 -24.7142857 35.8095238 -4.8571429 29.2380952 -49.0952381

 0.0000000  0.0000000  3.0000000 -20.0000000 23.0000000 -24.0000000  2.0000000 -24.0000000 15.0000000

 0.0000000  0.0000000  0.1904762 -1.0952381  0.8571429 -0.9047619  0.4285714 -0.6190476  1.0476190

the matrix after 3 step

 5.0000000 -29.0000000  1.0000000 20.0000000 23.0000000 -18.0000000 -54.0000000 -7.0000000 -3.0000000

 0.0000000 -4.2000000 -24.2000000 10.0000000  2.4000000  2.6000000 -28.2000000 -10.6000000 -13.4000000

 0.0000000  0.0000000 -9.3809524 50.1904762 -24.7142857 35.8095238 -4.8571429 29.2380952 -49.0952381

 0.0000000  0.0000000  0.0000000 -3.9492386 15.0964467 -12.5482234  0.4467005 -14.6497462 -0.7005076

 0.0000000  0.0000000  0.0000000 -0.0761421  0.3553299 -0.1776650  0.3299492 -0.0253807  0.0507614

the matrix after 4 step

 5.0000000 -29.0000000  1.0000000 20.0000000 23.0000000 -18.0000000 -54.0000000 -7.0000000 -3.0000000

 0.0000000 -4.2000000 -24.2000000 10.0000000  2.4000000  2.6000000 -28.2000000 -10.6000000 -13.4000000

 0.0000000  0.0000000 -9.3809524 50.1904762 -24.7142857 35.8095238 -4.8571429 29.2380952 -49.0952381

 0.0000000  0.0000000  0.0000000 -3.9492386 15.0964467 -12.5482234  0.4467005 -14.6497462 -0.7005076

 0.0000000  0.0000000  0.0000000  0.0000000  0.0642674  0.0642674  0.3213368  0.2570694  0.0642674

the matrix after 5 step

 5.0000000 -29.0000000  1.0000000 20.0000000 23.0000000 -18.0000000 -54.0000000 -7.0000000 -3.0000000

 0.0000000 -4.2000000 -24.2000000 10.0000000  2.4000000  2.6000000 -28.2000000 -10.6000000 -13.4000000

 0.0000000  0.0000000 -9.3809524 50.1904762 -24.7142857 35.8095238 -4.8571429 29.2380952 -49.0952381

 0.0000000  0.0000000  0.0000000 -3.9492386 15.0964467 -12.5482234  0.4467005 -14.6497462 -0.7005076

 0.0000000  0.0000000  0.0000000  0.0000000  0.0642674  0.0642674  0.3213368  0.2570694  0.0642674

Листинг 8: Шаги матрицы используя алгоритма Гаусса.

Fundamental system of solutions (columns):

982.0000000 2784.0000000 2764.0000000 751.0000000

162.0000000 458.0000000 457.0000000 125.0000000

-31.0000000 -89.0000000 -88.0000000 -24.0000000

-7.0000000 -19.0000000 -19.0000000 -4.0000000

-1.0000000 -5.0000000 -4.0000000 -1.0000000

 1.0000000  0.0000000  0.0000000  0.0000000

 0.0000000  1.0000000  0.0000000  0.0000000

 0.0000000  0.0000000  1.0000000  0.0000000

 0.0000000  0.0000000  0.0000000  1.0000000

Листинг 9: Матрица фундаментальной системы решения

Residual Frobenius norm: 0.0000000

Листинг 10: Оценка ошибок