Решение СЛАУ прямыми методами

Метод Гаусса

Шерухин Кирилл

5030102/30001
СПБПУ
2024

Содержание

1 Формулировка задачи и ее формализация

Дано: Невырожденная произвольная система линейных алгебраических уравнений из n уравнений и n неизвестных

Цель: Численно получить решение СЛАУ.

План выполнения:

1. Представить данную СЛАУ в матричном виде.
2. Получить решение методом Гаусса.

2 Алгоритм и условия его применимости

2.1 Алгоритм Гаусса

$\forall A \in \mathbb{R}^{n\times n}\ \ B \in \mathbb{R}^{n} : AX = B \iff [A|B]$

Пользуясь эквивалентными операциями над матрицей A:

$$[A|B] \xrightarrow[\text{Операции прямого хода}]{\{T_k\}_n} [U = A^{(n)}|B^{(n)}] \xrightarrow[\text{Операции обратного хода}]{\{G_k\}_n} [E|X]$$

Введём операции прямого хода T_k:

$$\left[\begin{array}{cccccc|c}
a_{00}^{(0)} & \cdots & a_{0k}^{(0)} & a_{0(k+1)}^{(0)} & \cdots & a_{0n}^{(0)} & b_0^{(0)} \\
\vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \vdots \\
0 & \cdots & a_{kk}^{(k)} & a_{k(k+1)}^{(k)} & \cdots & a_{kn}^{(k)} & b_k^{(k)} \\
0 & \cdots & a_{(k+1)k}^{(k)} & a_{(k+1)(k+1)}^{(k)} & \cdots & a_{(k+1)n}^{(k)} & b_{k+1}^{(k)} \\
\vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \vdots \\
0 & \cdots & a_{nk}^{(k)} & a_{n(k+1)}^{(k)} & \cdots & a_{nn}^{(k)} & b_n^{(k)}
\end{array}\right]$$

$$\downarrow T_k$$

$$\left[\begin{array}{cccccc|c}
a_{00}^{(0)} & \cdots & a_{0k}^{(0)} & a_{0(k+1)}^{(0)} & \cdots & a_{0n}^{(0)} & b_0^{(0)} \\
\vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \vdots \\
0 & \cdots & a_{kk}^{(k)} & a_{k(k+1)}^{(k)} & \cdots & a_{kn}^{(k)} & b_k^{(k)} \\
0 & \cdots & 0 & a_{(k+1)(k+1)}^{(k+1)} & \cdots & a_{(k+1)n}^{(k+1)} & b_{k+1}^{(k+1)} \\
\vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \vdots \\
0 & \cdots & 0 & a_{n(k+1)}^{(k+1)} & \cdots & a_{nn}^{(k+1)} & b_n^{(k+1)}
\end{array}\right]
\quad
\begin{array}{l}
\forall i,j : k < i \le n \ \ \& \ \ k < j \le n \\
m_{ik} := \frac{a_{ik}^{(k)}}{a_{kk}^{(k)}} \\
a_{ij}^{(k+1)} = a_{ij}^{(k)} - m_{ik} * a_{kj}^{(k)} \\
b_i^{(k+1)} = b_i^{(k)} - m_{ik} * b_k^{(k)}
\end{array}$$

Введём операции обратного хода G_k:

$$\left[\begin{array}{cccccc|c} a_{00} & \cdots & a^{(k)}_{0(n-k-1)} & a^{(k)}_{0(n-k)} & \cdots & 0 & b^{(k)}_0 \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & \cdots & a_{(n-k-1)(n-k-1)} & a^{(k)}_{(n-k-1)(n-k)} & \cdots & 0 & b^{(k)}_{n-k-1} \\ 0 & \cdots & 0 & a_{(n-k)(n-k)} & \cdots & 0 & b^{(k)}_{n-k} \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & \cdots & 0 & 0 & \cdots & 1 & b^{(1)}_n \end{array}\right]$$

$$\downarrow G_k$$

$$\left[\begin{array}{cccccc|c} a_{00} & \cdots & a^{(k+1)}_{0(n-k-1)} & 0 & \cdots & 0 & b^{(k+1)}_0 \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & \cdots & a_{(n-k-1)(n-k-1)} & 0 & \cdots & 0 & b^{(k+1)}_{n-k-1} \\ 0 & \cdots & 0 & 1 & \cdots & 0 & b^{(k+1)}_{n-k} \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & \cdots & 0 & 0 & \cdots & 1 & b^{(1)}_n \end{array}\right] \quad \begin{array}{l} \forall i : 0 \le i < n-k \\ b^{(k+1)}_{n-k} := \frac{b^{(k)}_{n-k}}{a^{(k)}_{(n-k)(n-k)}} \\ b^{(k+1)}_i = b^{(k)}_i - a_{i(n-k)} * b^{(k+1)}_{n-k} \\ a^{(k+1)}_{i(n-k)} = 0 \end{array}$$

Вычислительная сложность алгоритма: $O(n^3)$.

2.2 Условия применимости

Чтобы избежать деления на 0 в приведенном алгоритме, необходимо условие для матриц получаемых прямым преобразованием:

$\sqsupset d_k := det(A^{(k)}(1:k, 1:k))$

$\forall\ 0 \le k < n : d_k = 0 \implies a^{(k)}_{kk} \neq 0$

3 Пердварительный анализ задачи

Перед проведением исследований необходимо убедиться, что метод Гаусса для нашей задачи применим. Так как генерировать матрицы для примеров мы будем случайно и использовать собираемся не целые числа, а числа с плавающей точкой, то случаи, когда $d_k = 0$, статистически невероятны. Но можно ожидать сильное отклонение от ожидаемого результата, если $a^{(k)}_{kk}$ значительно меньше других чисел в столбце. Если же матрица всё-таки не будет одволетворять условиям алгоритма Гаусса, то программа выдаст ошибку, и матрица будет отброшена.

4 Тестовый пример

$$\begin{cases} x_1 + 2x_2 + 3x_3 = 4 \\ 5x_1 + 6x_2 + 7x_3 = 8 \\ 9x_1 + 1x_3 = 4 \end{cases} \iff \left[\begin{array}{ccc|c} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \\ 9 & 0 & 1 & 4 \end{array}\right]$$

$$\left[\begin{array}{ccc|c} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \\ 9 & 0 & 1 & 4 \end{array}\right] \xrightarrow{T_0} \left[\begin{array}{ccc|c} 1 & 2 & 3 & 4 \\ 0 & -4 & -8 & -12 \\ 0 & -18 & -26 & -32 \end{array}\right] \xrightarrow{T_1} \left[\begin{array}{ccc|c} 1 & 2 & 3 & 4 \\ 0 & -4 & -8 & -12 \\ 0 & 0 & 10 & 22 \end{array}\right]$$

$$\left[\begin{array}{ccc|c} 1 & 2 & 3 & 4 \\ 0 & -4 & -8 & -12 \\ 0 & 0 & 10 & 22 \end{array}\right] \xrightarrow{G_0} \left[\begin{array}{ccc|c} 1 & 2 & 0 & -2.6 \\ 0 & -4 & 0 & 5.6 \\ 0 & 0 & 1 & 2.2 \end{array}\right] \xrightarrow{G_1} \left[\begin{array}{ccc|c} 1 & 0 & 0 & 0.2 \\ 0 & 1 & 0 & -1.4 \\ 0 & 0 & 1 & 2.2 \end{array}\right] \xrightarrow{G_2} \left[\begin{array}{ccc|c} 1 & 0 & 0 & 0.2 \\ 0 & 1 & 0 & -1.4 \\ 0 & 0 & 1 & 2.2 \end{array}\right]$$

Ответ: $\begin{bmatrix} x_1 & x_2 & x_3 \end{bmatrix} = \begin{bmatrix} 0.2 & -1.4 & 2.2 \end{bmatrix}$

5 Подготовка контрольных тестов

Исследовать алгоритм будем по следующим характеристикам:

1. Зависимость относительной погрешности решения от числа обусловленности матрицы.
   Тестируется на 400 случайных матрицах 6×6 с точным решением X и обусловленностью от 1 до 10^{15}.
   Ожидается возрастание отностительной погрешшности с увеличением числа обусловленности.

2. Зависимость отностительной невязки СЛАУ от числа обусловленности матрицы.
   Тестируется на тех же 400 случайных матрицах 6×6 с точным решением X и обусловленностью от 1 до 10^{15}.
   Ожидается стабилизация относительной невязки с увеличением числа обусловленности.

3. Зависимость количества операций от размера матрицы.
   Тестируется на случайных матрицах размерностью от 1×1 до 200×200 с точным решением X и обусловленностью 30.
   Ожидается, что график покажет ассимптотику $O(n^3)$.

4. Зависимость отностительной ошибки решения от отностительной ошибки коэффициентов матриц с большим и маленьким числом обусловленности.
   Тестируется на 2-х случайных матрицах 6×6 с точным решением X и обусловленностью 100 и 10^7 соответственно. Диапазон δA — от 0 до 0.7 с делением диапазона на 200 точек.
   Ожидается возрастание отностельной ошибки решение с увелиением отностительной ошибки матрицы, большая ошибка решения для матрицы с большим числом обусловленности.

Как норму для векторов и матриц будем брать L_2.
Для вычисления ошибок решения необходимо знать само точное решение для СЛАУ, которую мы исследуем. Строго зададим его и будем генерировать A и B, исходя из него:

$$X = [123123; 1376785; 726129; 76878238; 34234; 4234234]^T$$

6 Модульная структура программы

Программный код оформим на языке C, он будет состоять из:

Самого алгоритма Гаусса, принимающего матрицу коэффициентов A, вектор свободных коэффициентов B, размер системы, и как выходные параметры — решение и количество операций, которые были произведены:

```
int gauss(double** mat, double* b, int n, double** res, int* actionsCnt);
```

Теста, принимающего множество расширенных матриц из файла по пути *in* и возвращающего множество решений для этих систем по пути *out* в csv формате:

```
void procces_err_to_cond(char* in, char* out);
```

Теста, принимающего множество расширенных матриц из файла по пути *in* и возвращающего множество операций, которые потребовались для их решения, в файл по пути *out* в csv формате:

```
void process_time_to_N(char* in, char* out);
```

7 Исследование метода

Из рис. 1 и 2 видно, что, как и ожидалось, с возрастанием числа обусловленности возраствает отностиельная ошибка решения, а невязка — стабилизируется на примерно $10^{15\pm2}$. Это прямо следует из самого определения числа обусловленности для матрицы (максимальное изменение функции: $\max(\frac{||\Delta Y||}{||Y||}/\frac{||\Delta X||}{||X||})$) и специфики алгоритма Гаусса: так как число обусловленности для $A \in \mathbb{R}^{n\times n}$ — отношение максимального и минимального собственного значения матрицы, то деление малого числа на большое становится более вероятным с возрастанием числа обусловленности.

Для исследования ассимптотики алгоритма, воспользуемся предположением, что она имеет вид $N = \mathcal{O}(size(A)^\alpha)$, тогда:

$$\frac{log(N)}{log(size(A))} = \frac{log(\mathcal{O}(size(A)^\alpha))}{log(size(A))} \approx \alpha$$

Из рис 3. можно заметить что при увеличении $size(A)$, α начнает стремиться к 3, как и ожидалось теоретически.

Для исследования верхней границы погрешностей решения применим формулу:

$$\delta X \leq cond(A)\delta A$$

Из рис. 4 можно заметить, что, как и ожидалось, ошибка решения сильно зависит от числа обусловенности матрицы A, но стоит заметить, что даже довольно малые ошибки делают неиспользуемым решение для обоих матриц. В общем, погрешность, действительно, хорошо оценивается теоретическим неравенством, но из-за возможности деления на малые числа есть отдельные выбросы за верхнюю границу, этот вид погрешности учесть общим неравенством не представляется возможным.

8 Визуальные приложения

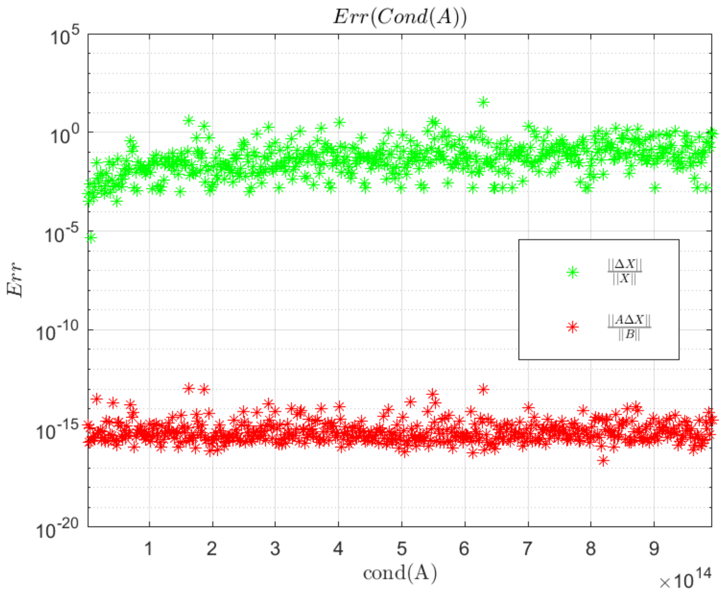

Рис. 1: Зависимость относительной ошибки и невязки от числа обусловленности

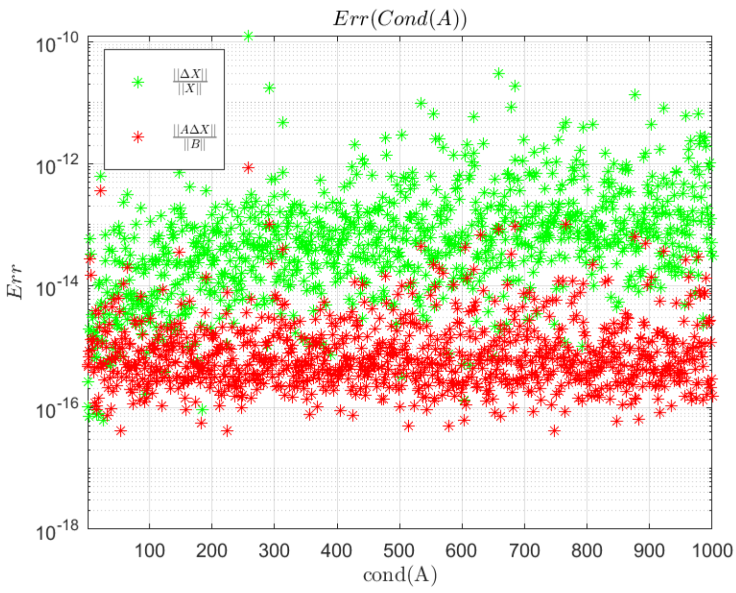

Рис. 2: Зависимость относительной ошибки и невязки от числа обусловленности (меньший масштаб)

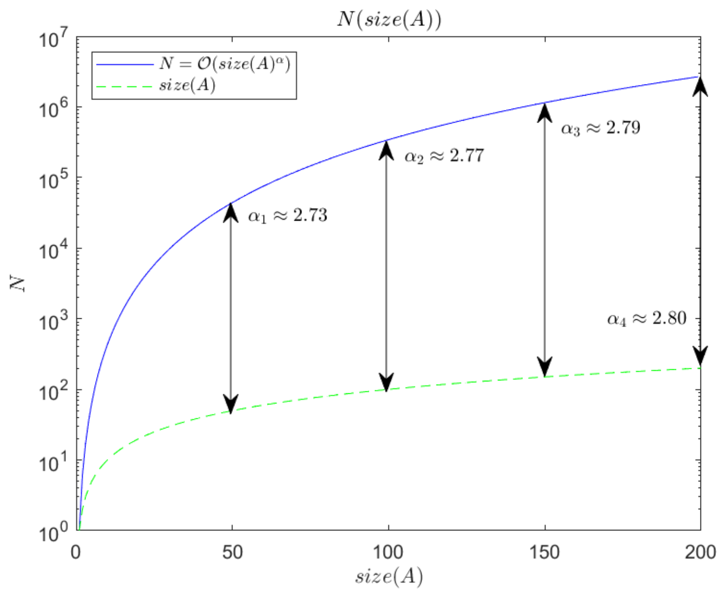

Рис. 3: Зависимость количества операций алгоритма от размера матрицы

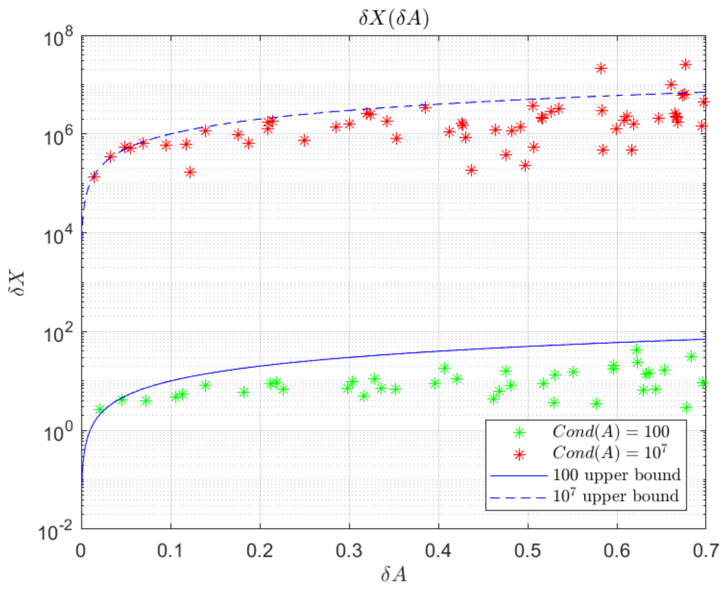

Рис. 4: Зависимость относительной ошибки решения от ошибки матриц с большим и маленьким числом обусловленности

9 Выводы

Алгоритм Гаусса сильно зависит от числа обусловленности матрицы, при больших его значениях погрешность получаемого решения может не удволетворять поставленному задачей (или вообще не решить систему, если довольно строгие в промышленных задачах условия применимости не выполняются: нули перед некоторыми коэффициентами бывают очень часто), на это можно повлиять, модифицируя алгоритм выбором наибольшего числа в строке/столбце, но это может повлиять на ассимптотику алгоритма и всё-равно не дать необходимой точности.

Рис. 4, не столько показывает невозможность использовать алгоритм Гаусса для системы с ошибкой в матрице A, сколько важность точного измерения этих коэффициентов в промышленных задачах. Скорее всего, другие алгоритмы покажут примерно такие же данные на этом тесте, возможно лишь с меньшим разбросом точек при $\delta A > 0.4$.

Так же неоспоримым недостатком алгоритма Гаусса является кубическая ассимптотика от размера решаемой матрицы, для больших систем алгоритм не применим.

Подводя итоги, можно сказать, что алгоритм Гаусса использовать не стоит. Есть множетсво других прямых методов, которые лишены его недостатков, приведенных выше. Например все методы, использующие ортогональные матрицы.