Системы уравнений

Система m линейных уравнений с n неизвестными (или, линейная система) в линейной алгебре — это система уравнений вида:

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n = b_1 \\ a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n = b_2 \\ \cdots \\ a_{m1}x_1 + a_{m2}x_2 + \cdots + a_{mn}x_n = b_m \end{cases} \quad (1)$$

Здесь $x_1, x_2, \ldots x_n$ — неизвестные, которые надо определить. $a_{11}, a_{12}, \ldots, a_{mn}$ — коэффициенты системы — и $b_1, b_2, \ldots\ b_m$ — свободные члены — предполагаются известными. Индексы коэффициентов (a_{ij}) системы обозначают номера уравнения (i) и неизвестного (j), при котором стоит этот коэффициент, соответственно.

Система (1) называется однородной, если все её свободные члены равны нулю ($b_1 = b_2 = \cdots = b_m = 0$), иначе — неоднородной.

Система (1) называется квадратной, если число m уравнений равно числу n неизвестных.

Решение системы (1) — совокупность n чисел $c_1, c_2, \ldots, c_n,$ таких что подстановка каждого c_i вместо x_i в систему (1) обращает все её уравнения в тождества.

Система (1) называется совместной, если она имеет хотя бы одно решение, и несовместной, если у неё нет ни одного решения.

Совместная система вида (1) может иметь одно или более решений.

Решения $c_1^{(1)}, c_2^{(1)}, \ldots, c_n^{(1)}$ и $c_1^{(2)}, c_2^{(2)}, \ldots, c_n^{(2)}$ совместной системы вида (1) называются различными, если нарушается хотя бы одно из равенств:

$$c_1^{(1)} = c_1^{(2)}, c_2^{(1)} = c_2^{(2)}, \ldots, c_n^{(1)} = c_n^{(2)}$$

Совместная система вида (1) называется определённой, если она имеет единственное решение; если же у неё есть хотя бы два различных решения, то она называется неопределённой. Если уравнений больше, чем неизвестных, она называется переопределённой.

Матричная форма

Система линейных уравнений может быть представлена в матричной форме как:

$$\begin{pmatrix} a_{11} & a_{12} & \ldots & a_{1n} \\ a_{21} & a_{22} & \ldots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \ldots & a_{mn} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_m \end{pmatrix}$$

или:

$$Ax = B.$$

Умножение матриц

Пусть даны две прямоугольные матрицы размера mxn и nxq соответственно:

$$A = \begin{pmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \dots & a_{mn} \end{pmatrix}, \qquad B = \begin{pmatrix} b_{11} & b_{12} & \dots & b_{1q} \\ b_{21} & b_{22} & \dots & b_{2q} \\ \vdots & \vdots & \ddots & \vdots \\ b_{n1} & b_{n2} & \dots & b_{nq} \end{pmatrix}$$

Тогда матрица C размерностью mxq называется их произведением:

$$\mathrm{C} = \begin{pmatrix} c_{11} & c_{12} & \dots & c_{1q} \\ c_{21} & c_{22} & \dots & c_{2q} \\ \vdots & \vdots & \ddots & \vdots \\ c_{m1} & c_{m2} & \dots & c_{mq} \end{pmatrix}$$

Где:

$$c_{ij} = \sum_{k=1}^{n} a_{ik} b_{kj} \quad i = 1, \dots, m; \quad j = 1,2, \dots, q$$

Операция умножения двух матриц выполнима только в том случае, если число столбцов в первом сомножителе равно числу строк во втором; в этом случае говорят, что форма матриц согласована. В частности, умножение всегда выполнимо, если оба сомножителя — квадратные матрицы одного и того же порядка.

Следует заметить, что из существования произведения AB вовсе не следует существование произведения BA.

Методы решения

Матричный метод

Матричный метод решения (метод решения через обратную матрицу) систем линейных алгебраических уравнений с ненулевым определителем состоит в следующем.
Пусть дана система линейных из n уравнений с n неизвестными:

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + \dots + a_{1n}x_n = b_1 \\ a_{21}x_1 + a_{22}x_2 + \dots + a_{2n}x_n = b_2 \\ \dots \\ a_{n1}x_1 + a_{n2}x_2 + \dots + a_{nn}x_n = b_n \end{cases}$$

Тогда её можно переписать в матричной форме:

$AX = B$, где A — основная матрица системы, B и X — столбцы свободных членов и решений системы соответственно:

$$A = \begin{pmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \dots & a_{mn} \end{pmatrix}, \qquad B = \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_m \end{pmatrix}, \qquad X = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix}$$

Умножим это матричное уравнение слева на A^{-1} — матрицу, обратную к матрице A:

$$A^{-1}(AX) = A^{-1}B$$

Так как $A^{-1}A = E$, получаем $X = A^{-1}B$. Правая часть этого уравнения даст столбец решений исходной системы. Условием применимости данного метода (как и вообще существования решения неоднородной системы линейных уравнений с числом уравнений, равным числу

неизвестных) является невырожденность матрицы A. Необходимым и достаточным условием этого является неравенство нулю определителя матрицы A:

$$\boldsymbol{detA \neq 0}$$

Нахождение обратной матрицы:

Найти обратную матрицу можно с помощью матрицы алгебраических дополнений по формуле:

$$\mathrm{A}^{-1} = \frac{1}{\det A} C^T$$

C^T — транспонированная матрица алгебраических дополнений;

Полученная матрица A^{-1} и будет обратной.

Сложность алгоритма зависит от сложности алгоритма расчета определителя O_{det} и равна $O(n^2) \cdot O_{det}$. При вычислении определителя «в лоб» (используя подстановки или разложение на определители меньшего порядка) $O_{det} = O(n!)$. То есть решения системы из 12 уравнений на компьютере с производительностью 1 TFLOPS (1 триллион операций с плавающей точкой в секунду), **решение займет время порядка** $\boldsymbol{20^2 * \frac{20!}{10^{12}}\ c\ =\ 973160803\ c \cong\ 30}$ лет. Не слишком эффективно…

Правило Крамера

Метод Крамера (правило Крамера) — способ решения квадратных систем линейных алгебраических уравнений с ненулевым определителем основной матрицы (причём для таких уравнений решение существует и единственно)

Метод Крамера требует вычисления *n* + 1 определителей размерности $n \times n$. При использовании метода Гаусса для вычисления определителей, метод имеет временную сложность порядка $O(n^4)$, что хуже, чем если бы метод Гаусса напрямую использовался для решения системы уравнений.

Метод Гаусса

Метод Гаусса — классический метод решения системы линейных алгебраических уравнений (СЛАУ). Это метод последовательного исключения переменных, когда с помощью элементарных преобразований система уравнений приводится к равносильной системе ступенчатого (или треугольного) вида, из которого последовательно, начиная с последних (по номеру) переменных, находятся все остальные переменные.

Алгоритм

Алгоритм решения СЛАУ методом Гаусса подразделяется на два этапа.

На первом этапе осуществляется так называемый прямой ход, когда путём элементарных преобразований над строками систему приводят к ступенчатой или треугольной форме, либо устанавливают, что система несовместна. А именно, среди элементов первого столбца матрицы выбирают ненулевой, перемещают его на крайнее верхнее положение перестановкой строк и вычитают получившуюся после перестановки первую строку из остальных строк, домножив её на величину, равную отношению первого элемента каждой из этих строк к первому элементу первой строки, обнуляя тем самым столбец под ним. После того, как указанные преобразования были совершены, первую строку и первый столбец мысленно вычёркивают и продолжают пока не останется матрица нулевого размера. Если на какой-то из итераций среди элементов первого

столбца не нашёлся ненулевой, то переходят к следующему столбцу и проделывают аналогичную операцию.

Пример:

$$\begin{cases} 2x + 3y + z = 11 \\ 3x + y - z = 2 \\ 2x - y + 5z = 15 \end{cases}$$

$$\begin{cases} 2x + 3y + z = 11 \\ 3x + y - z = 2 \ \ | * 2/3 \\ 2x - y + 5z = 15 \ | * 2/2 \end{cases}$$

$$\begin{cases} 2x + 3y + z = 11 \quad (1) \\ 2x + \frac{2y}{3} - \frac{2z}{3} = \frac{4}{3} \ \ | - (1) \\ 2x - y + 5z = 15 \quad | - (1) \end{cases}$$

$$\begin{cases} 2x + 3y + z = 11 \\ 0x - \frac{7y}{3} - \frac{5z}{3} = -\frac{29}{3} \\ 0x - 4y + 4z = 4 \end{cases}$$

$$\begin{cases} \cancel{2x + 3y + z = 1} \\ \frac{7y}{3} - \frac{5z}{3} = -\frac{29}{3} \\ 4y + 4z = 4 \ | * 7/(3 * 4) \end{cases}$$

$$\begin{cases} \frac{7y}{3} - \frac{5z}{3} = -\frac{29}{3} \ | \quad (2) \\ \frac{7y}{3} + \frac{7z}{3} = \frac{7}{3} \ | - (2) \end{cases}$$

$$\begin{cases} \cancel{\frac{7y}{3}} - \cancel{\frac{5z}{3}} = -\cancel{29/3} \\ 0 * y + 4z = 12 \end{cases}$$

$$4z = 12$$

На втором этапе осуществляется так называемый обратный ход, суть которого заключается в том, чтобы выразить все получившиеся базисные переменные через небазисные и построить фундаментальную систему решений, либо, если все переменные являются базисными, то выразить в численном виде единственное решение системы линейных уравнений. Эта процедура начинается с последнего уравнения, из которого выражают соответствующую базисную переменную (а она там всего одна) и подставляют в предыдущие уравнения, и так далее, поднимаясь по «ступенькам» наверх. Каждой строчке соответствует ровно одна базисная переменная, поэтому на каждом шаге, кроме последнего (самого верхнего), ситуация в точности повторяет случай последней строки.

Пример:

$$4z = 12$$

$$z = 3$$

Далее из зачеркнутых уравнений:

$$y = \frac{-29 + 5z}{3} = 2$$

$$x = \frac{11 - z - 3z}{2} = 1$$

Ответ:

$$X = \begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix}$$

Метод Гаусса требует порядка $O(n^3)$ действий. Компьютер в 1TFLOPS решит систему 20x20 за время $20^3/10^{12}$ c =8 нс. **Быстрее метода с обратной матрицей в $\mathbf{10^{17}}$ раз**.

Используя метод Гаусса можно найти обратную матрицу:

При последовательной подстановке столбцов единичной матрицы размера матрицы А в качестве правой части, матрица из векторов X полученных в результате решения соответствующих систем будет обратной к матрице А.

$$\begin{pmatrix} 2 & 3 & 1 \\ 3 & 1 & -1 \\ 2 & -1 & 5 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} \Rightarrow \begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} -\frac{1}{12} \\ \frac{17}{48} \\ \frac{5}{48} \end{pmatrix}$$

$$\begin{pmatrix} 2 & 3 & 1 \\ 3 & 1 & -1 \\ 2 & -1 & 5 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix} \Rightarrow \begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} \frac{1}{3} \\ -\frac{1}{6} \\ -\frac{1}{6} \end{pmatrix}$$

$$\begin{pmatrix} 2 & 3 & 1 \\ 3 & 1 & -1 \\ 2 & -1 & 5 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} \Rightarrow \begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} \frac{1}{12} \\ -\frac{5}{48} \\ \frac{7}{48} \end{pmatrix}$$

Тогда:

$$C = \begin{pmatrix} -\frac{1}{12} & \frac{1}{3} & \frac{1}{12} \\ \frac{17}{48} & -\frac{1}{6} & -\frac{5}{48} \\ \frac{5}{48} & -\frac{1}{6} & \frac{7}{48} \end{pmatrix} = A^{-1}$$

Проверка:

$$AC = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

Задание:

1) Написать функцию реализующую умножение двух матриц по определению
   Заголовок функции такого вида:

```
function C = matrix_multiply(A,B)
```

2) Написать функцию реализующую метод Крамера для системы произвольной размерности
   Заголовок функции такого вида:

```
function X = CramersMethod(A,B)
```

3) Написать функцию реализующую метод Гаусса для системы произвольной размерности
   Заголовок функции такого вида:

```
function X = GaussMethod(A,B)
```

4) Написать функцию реализующую вычисление обратной матрицы, используя уже написанную функцию из пункта 3)
   Заголовок функции такого вида:

```
function X = GaussInverse(A,B)
```

5) Сравнить скорость работы функций с встроенными в MATLAB функциями для решения систем и нахождения обратных матриц.

Указание:

Для работы с произвольными матрицами нужно проходить последовательно по их строкам или столбцам. Для этого в MATLAB существует оператор `for`
Оператор `for` имеет синтаксис:

```
for i=A
   %некоторые вычисления
end
```

где i – переменная, в которую записывается следующее значение из массива A.
При этом значения в массиве A не меняются при изменении переменной i внутри цикла.
`%некоторые вычисления` – вычисления, которые будут повторяться при каждой *итерации*
Пример:

1) Вывести все значения массива A

```
for i=A
    disp(i)
end
```

2) Сложить числа от 1 до 1000

```
Sum = 0;
for i=1:1000
    Sum = Sum+1;
end
```

3) Вывести элементы матрицы по строкам

```
for i=1:size(A,1) %size(A,1) - значит взять первое измерение матрицы - то
                          есть число строк
    for j=1:size(A,2) %size(A,2) - значит взять второе измерение матрицы - то
                                 есть число столбцов
        disp(A(i,j))
    end
end
```

Рассмотрим логику работы этого кода:
Сначала переменной i присваивается значение 1, следующим шагом программа попадает в новый цикл: в нем переменная j меняется от 1 до количества столбцов в матрице A при постоянной переменной i. Затем переменной i присваивается значение 2 и т.д.

Для измерения производительности скрипта или команды нужно использовать функции измеряющие время. В MATLAB встроены функции `tic` и `toc`. Функция `tic засекает начальное значение времени, функция toc засекает конечное и вычитает из него начальное значение полученное функцией toc.`
`Пример:`

```
tic
X = A\B; %долгие вычисления
toc
```

Вывод:

```
Elapsed time is 0.000809 seconds.
```

Можно получить возвращаемое функцией toc значение:

```
tic
X = A\B; %долгие вычисления
t=toc
disp t
```

Вывод:

```
    0.00080523831011399
```