

Метод Гаусса решения линейной системы с выбором главного элемента по столбцу

7 декабря 2019 г.

1 Алгоритм

Требуется решить систему линейных уравнений $Ax = b$ (A - матрица размера $n \times n$, b - n -мерный вектор свободных членов) методом Гаусса с выбором главного элемента по столбцу. Зададим m - размер блока матрицы. Тогда $n = mk + 1$ и матрица представима в виде:

$$\begin{pmatrix} A_{0,0}^{m \times m} & A_{0,1}^{m \times m} & \dots & A_{0,k-1}^{m \times m} & A_{0,k}^{m \times l} & b_0^m \\ A_{1,0}^{m \times m} & A_{1,1}^{m \times m} & \dots & A_{1,k-1}^{m \times m} & A_{1,k}^{m \times l} & b_1^m \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ A_{k-1,0}^{m \times m} & A_{k-1,1}^{m \times m} & \dots & A_{k-1,k-1}^{m \times m} & A_{k-1,k}^{m \times l} & b_{k-1}^m \\ A_{k,0}^{l \times m} & A_{k,1}^{l \times m} & \dots & A_{k,k-1}^{l \times m} & A_{k,k}^{l \times l} & b_k^l \end{pmatrix}$$

За норму возьмём максимальную норму столбца:

$$\|A^{m \times m}\| = \max_{j=0, \dots, m-1} \sum_{i=0}^{m-1} |a_{i,j}|$$

Найдем в первом столбце обратный блок с минимальной нормой. Если ни у одного блока не посчиталась обратная, метод применить нельзя. Теперь переставляем строку у которой первый блок имеет минимальную обратную с первой. Затем умножаем каждый блок первой строки на минимальную обратную слева.

Умножаем каждый элемент первой строки на обратную первого элемента в этой строке

$$j = 1, \dots, k$$

$$A_{0,j}^{m \times m} = (A_{0,0}^{m \times m})^{-1} A_{0,j}^{m \times m}$$

Умножаем так же блок размера $m \times l$

$$A_{0,k}^{m \times l} = (A_{0,0}^{m \times m})^{-1} A_{0,k}^{m \times l}$$

Умножаем часть вектора b

$$b_0^m = (A_{0,0}^{m \times m})^{-1} b_0^m$$

Получим новую матрицу вида:

$$\begin{pmatrix} E^{m \times m} & A_{0,1}^{m \times m} & \dots & A_{0,k-1}^{m \times m} & A_{0,k}^{m \times l} & b_0^m \\ A_{1,0}^{m \times m} & A_{1,1}^{m \times m} & \dots & A_{1,k-1}^{m \times m} & A_{1,k}^{m \times l} & b_1^m \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ A_{k-1,0}^{m \times m} & A_{k-1,1}^{m \times m} & \dots & A_{k-1,k-1}^{m \times m} & A_{k-1,k}^{m \times l} & b_{k-1}^m \\ A_{k,0}^{l \times m} & A_{k,1}^{l \times m} & \dots & A_{k,k-1}^{l \times m} & A_{k,k}^{l \times l} & b_k^l \end{pmatrix}$$

Теперь вычитаем из i ой строки первую, умноженную на первый коэффициент i ой строки:

$$A_{i,j}^{m \times m} = A_{i,j}^{m \times m} - A_{i,0}^{m \times m} * A_{0,j}^{m \times m}, i = 1, \dots, k-1, j = 1, \dots, k-1$$

$$A_{i,k}^{m \times l} = A_{i,k}^{m \times p} - A_{i,0}^{m \times m} * A_{0,k}^{m \times l}, i = 1, \dots, k-1$$

$$A_{k,i}^{l \times m} = A_{k,i}^{l \times m} - A_{k,0}^{l \times m} * A_{0,i}^{m \times l}, i = 1, \dots, k-1$$

$$A_{k,k}^{l \times l} = A_{k,k}^{l \times l} - A_{k,0}^{l \times m} * A_{0,k}^{m \times l}$$

$$b_i^m = b_i^m - A_{i,1}^{m \times m} * b_0^m, i = 1, \dots, k-1$$

$$b_k^l = b_k^l - A_{k,1}^{l \times m} * b_0^m$$

Получим матрицу вида:

$$\begin{pmatrix} E^{m \times m} & A_{0,1}^{m \times m} & \dots & A_{0,k-1}^{m \times m} & A_{0,k}^{m \times l} & b_0^m \\ 0 & A_{1,1}^{m \times m} & \dots & A_{1,k-1}^{m \times m} & A_{1,k}^{m \times l} & b_1^m \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & A_{k-1,1}^{m \times m} & \dots & A_{k-1,k-1}^{m \times m} & A_{k-1,k}^{m \times l} & b_{k-1}^m \\ 0 & A_{k,1}^{l \times m} & \dots & A_{k,k-1}^{l \times m} & A_{k,k}^{l \times l} & b_k^l \end{pmatrix}$$

Теперь повторяем алгоритм для подматрицы размера $(k-1) \times (k-1)$ блоков. Общие формулы на шаге p ($p=0, \dots, k-1$):

$$j = p+1, \dots, k$$

$$A_{p,j}^{m \times m} = (A_{p,p}^{m \times m})^{-1} A_{p,j}^{m \times m}$$

Умножаем так же блок размера $m \times l$

$$A_{p,k}^{m \times l} = (A_{p,p}^{m \times m})^{-1} A_{p,k}^{m \times l}$$

Умножаем часть вектора b

$$b_p^m = (A_{p,p}^{m \times m})^{-1} b_p^m$$

$$A_{i,j}^{m \times m} = A_{i,j}^{m \times m} - A_{i,p}^{m \times m} * A_{p,j}^{m \times m}, i = p+1, \dots, k-1, j = p+1, \dots, k-1$$

$$A_{i,k}^{m \times l} = A_{i,k}^{m \times p} - A_{i,p}^{m \times m} * A_{p,k}^{m \times l}, i = p+1, \dots, k-1$$

$$A_{k,i}^{l \times m} = A_{k,i}^{l \times m} - A_{k,p}^{l \times m} * A_{p,i}^{m \times l}, i = p+1, \dots, k-1$$

$$A_{k,k}^{l \times l} = A_{k,k}^{l \times l} - A_{k,p}^{l \times m} * A_{p,k}^{m \times l}$$

$$b_i^m = b_i^m - A_{i,p}^{m \times m} * b_p^m, i = p+1, \dots, k-1$$

$$b_k^l = b_k^l - A_{k,p}^{l \times m} * b_p^m$$

После прохождения алгоритма получаем верхнетругольную матрицу с единицами на диагонали.

$$\begin{pmatrix} E^{m \times m} & A_{0,1}^{m \times m} & \dots & A_{0,k-1}^{m \times m} & A_{0,k}^{m \times l} & b_0^m \\ 0 & E^{m \times m} & \dots & A_{1,k-1}^{m \times m} & A_{1,k}^{m \times l} & b_1^m \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \dots & E^{m \times m} & A_{k-1,k}^{m \times l} & b_{k-1}^m \\ 0 & 0 & \dots & 0 & E^{l \times l} & b_k^l \end{pmatrix}$$

1.1 Обртанный ход

$$b_i = b_i - A_{i,j} * b_j, j = k, \dots, 0, i = i+1, \dots, 0$$

После выполнения алгоритма в векторе b будет исходный ответ.

2 Работа с блоками

Для работы с блоками матрицы пригодятся две функции - для получения и изменения блоков:

```
void GetBlock(double* A, double* block, const int x, const int y, const int x1, const int y1, const int x2, const int y2)
void PutBlock(double* A, double* block, const int x, const int y, const int x1, const int y1, const int x2, const int y2)
```

По переданным индексам x, y и индексам конца $x1, y1$ кладет часть матрицы A в $block$. Во втором случае блок записывается в матрицу A на место x, y . Предполагается, что матрица A лежит в памяти построчно.

3 Сложность блочного алгоритма

Сложность операций которые мы используем:

Нахождение обратной матрицы размера n : $8/3 * (n^3)$

Вычитание двух матриц размера m на n : $m * n$

Перемножение двух матриц $m \times n$ и $n \times k$: $2m * n * k$

Пусть $k = n/m$, тогда

Сложность нахождения обратных матриц:

$$\sum_{i=0}^k (k-i) * (8/3 * m^3) = 8/6 * k * (k+1) * (m^3) = 8/6 * m^3 * k^2 = 8/6 * m * n^2$$

Сложность перемножения первой строки матрицы на обратный блок:

$$\sum_{i=0}^k (k-i) * (2m^3) = m^3 * k^2 = m * n^2$$

Сложность вычитания и умножения строк:

$$\sum_{i=1}^k (\sum_{s=i+1}^k (k-i) * (2m^3 + m^2)) = \sum_{i=0}^k (k-i)^2 * (2m^3) = 2/3 * m^3 * k^3 = 2/3 * n^3$$

Сложность обратного хода: $\sum_{i=0}^k (i) * (2 * m^2 + m^2)$

$$\sum = 8/6 * m^3 * k^2 + m^3 * k^2 + 2/3 * m^3 * k^3 = 2/3 * m^3 * k^3 + 7/3 * m^3 * k^2$$

При $m = 1, k = n$: $\sum = 2 * n^3/3$

При $m = n, k = 1$: $\sum = 3 * n^3$

4 Параллельная реализация

Пусть p - количество потоков. Тогда каждый i -й поток будет работать с блочными строками с номерами $k \equiv i \pmod p$

В ходе алгоритма в начале, каждый поток находит обратную матрицу с минимальной нормой среди своих строк.

Точка синхронизации.

Затем поток индекс которого сравним с данным шагом k находит строку с наименьшей нормой обратного первого блока, переносит на k -ое место. Домножает строку на данную обратную матрицу.

Точка синхронизации.

Каждый поток копирует k -ую строку. После вычитает сохраненную строку из своих строк, домноженную на нужный коэффициент ($k \equiv i \pmod p$ поток не вычитает свою k -ую строку из самой себя). Из остаточной неполной строки вычитает первый поток.

$$(A_{k,0}^{l \times m} \quad A_{k,1}^{l \times m} \quad \dots \quad A_{k,k-1}^{l \times m} \quad A_{k,k}^{l \times l} \quad b_k^l)$$

Точка синхронизации и можно повторять шаг алгоритма.

Первый поток считает остаточный неполный крайний блок, домножает ответ на обратную.

Точка синхронизации перед обратным ходом.

В обратном ходе нужно k точек синхронизации.

По итогу получается 3 точки синхронизации на шаг прямого хода, одна на шаг обратного, одна перед обратным, всего $4k + 1$ точек синхронизации.