Лекция 13

Основы матричных вычислений

Рахуба М.В.

26.04.21

Прямые методы для решения линейных систем с большими разреж. матр.

① Ф-ла Шермана – Моррисона

$A = LU$ – $O(n^3)$, $\quad LUx = b$ $\quad O(n^2)$

$b \to \tilde{b}$ – $O(n^2)$

$A \to \tilde{A}$ – $O(n^2)$

$\boxed{\text{Утв 1}}$ Пусть $\det(A) \neq 0$ и $u, v \in \mathbb{R}^n$.

Тогда

1) $(A + uv^T)$ обратима $\Leftrightarrow$ $1 + v^T A^{-1} u \neq 0$

2) $$(A + uv^T)^{-1} = A^{-1} - \frac{(A^{-1}u)(v^T A^{-1})}{1 + v^T A^{-1} u}$$

например, замена в A элемента или строки или столбца

$$(A + uv^T)x = b \Rightarrow x = A^{-1}b - \frac{A^{-1}u \; v^T A^{-1} b}{1 + v^T A^{-1} u}$$

Нужно вычислить $A^{-1}b$, $A^{-1}u$ как решение лин. систем

$\boxed{\text{Лемма}}$ $\quad \det(I + ab^T) = 1 + b^T a$

□

$$(I + ab^T)v = v \;, \quad v \perp b \Rightarrow \lambda = 1$$

$$(I + ab^T)a = (1 + b^T a)\,a \Rightarrow \underline{\lambda = 1 + b^T a}$$

$$\det(I + ab^T) = \prod \lambda_i = 1 + b^T a$$

∎

□ (Упр 1)

1) $\det(A + uv^T) = \det(A)\det(I + A^{-1}uv^T)$

$$= \det(A)\cdot(1 + v^T A^{-1} u)$$

2) $$(A + uv^T)\left(A^{-1} - \frac{A^{-1}uv^TA^{-1}}{1 + \underbrace{v^TA^{-1}u}_{\gamma}}\right) =$$

$$= I - \frac{uv^TA^{-1}}{1+\gamma} + uv^TA^{-1} - \frac{u\overbrace{v^TA^{-1}u}^{\gamma}v^TA^{-1}}{1+\gamma} =$$

$$= I - uv^TA^{-1}\underbrace{\left(\frac{1}{1+\gamma} - 1 + \frac{\gamma}{1+\gamma}\right)}_{0}$$

∎

$$(A + \overset{\in \mathbb{R}^{n\times r}}{U} V^T)^{-1} = A^{-1} - A^{-1} U (I_r + V^T A^{-1} U)^{-1} V^T A^{-1}$$

(Тождество Вудберри)

$\boxed{\text{Замеч.}}$ Бывают малоранговые апдейты и для разложений, например, QR, Холецкий, SVD, ...

② Разреженные матрицы

В python scipy.sparse

Форматы хранения – на семинаре

а) Заполнение в L и U

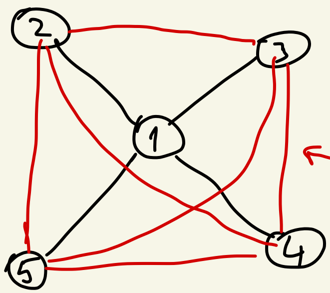
Будем рассматривать $A = A^T > 0$

A в соотв. поставим граф:

$a_{ij} \neq 0 \qquad (i) — (j)$

$$A = \begin{pmatrix} * & * & * & * & * \\ * & * & & & \\ * & & * & & \\ * & & & * & \\ * & & & & * \end{pmatrix}, \quad L = \begin{pmatrix} * & & & & 0 \\ * & * & & & \\ * & F & * & & \\ * & F & F & * & \\ * & F & F & F & * \end{pmatrix}$$

заполнение fill-in

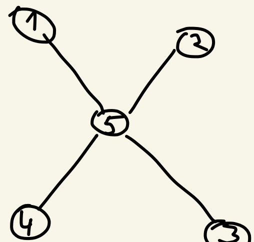


заполнение в L и U. Добавляются рёбра между соседями i с бо́льшим, чем i номером

$$a_{ij}^{(k+1)} = a_{ij}^{(k)} - \frac{a_{ik}^{(k)} a_{kj}^{(k)}}{a_{kk}^{(k)}}$$

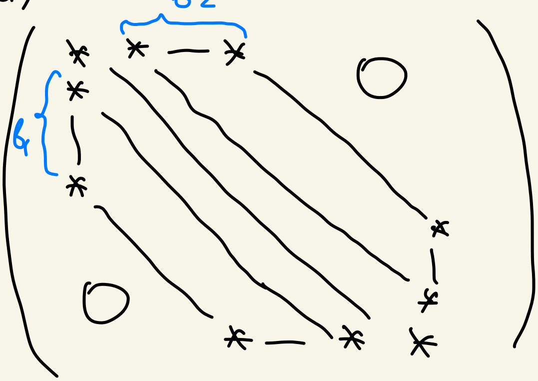
$$PAP^T = \begin{pmatrix} * & & & & * \\ & * & & & * \\ & & * & & * \\ & & & * & * \\ * & * & * & * & * \end{pmatrix}$$

не будет заполнения



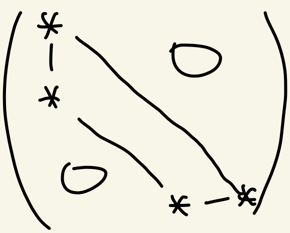
Для ленточных матриц (band)

A =

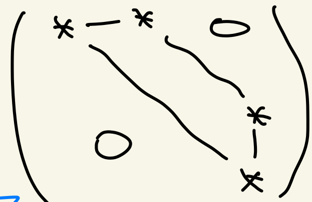


$b = \max(b_1, b_2)$ – ширина ленты (bandwidth)

L =



, U =



нет заполнения

Сложность: $O(nb^2)$

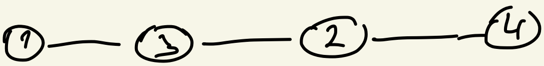
$b = 1$ – метод прогонки (линейные р-ии)

3 Алгоритм поиска P

а) Катхилл – Макки: минимизация ширины ленты b

Граф $G = (V, E)$

Пример матрицы:



$b=2$:

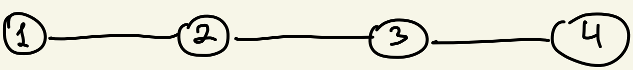
$$\begin{pmatrix} * & 0 & * & 0 \\ 0 & * & * & * \\ * & * & * & 0 \\ 0 & * & 0 & * \end{pmatrix}$$

ширина ленты

$$b = \max_{(u,v)\in E} |\sigma(u) - \sigma(v)|, \quad \sigma : V \to \{1, 2, .., |V|\}$$

Как пронумеровать, чтобы уменьшить b?

Пронумеруем "близкие" друг к другу:



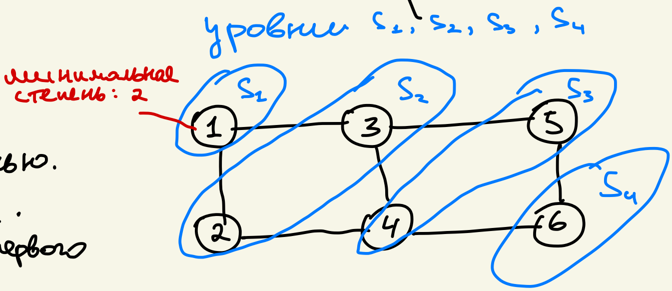
$|2-1| = |3-2| = |4-3| = 1$

$B=1$:

$$\begin{pmatrix} * & * & 0 & 0 \\ * & * & * & 0 \\ 0 & * & * & * \\ 0 & 0 & * & * \end{pmatrix}$$

Алгоритм:

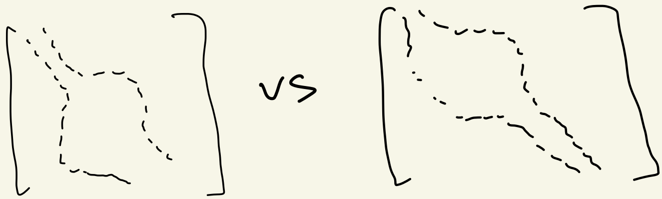
1) Выберем $v_1 \in V$, например, с минимальной степенью. Положим $\sigma(v_1) = 1$. $S_1 = \{v_1\}$ – вершины первого уровня



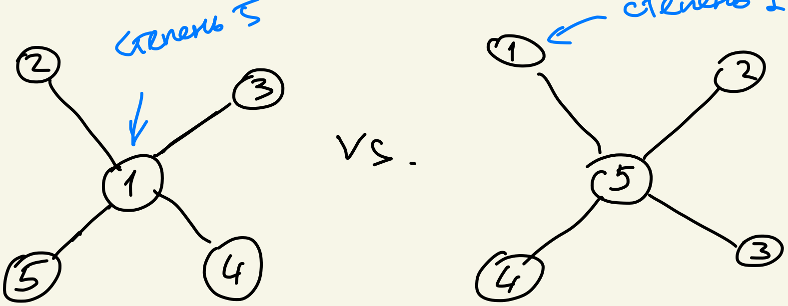
2) Обозначим S_2 соседей S_1. Для каждой вершины из S_1 (по возраст.) пронумеруем её соседей из S по возрастанию их степени.

3) Для каждой $v \in S_2$ найдём ещё не пронумер. соседей. Обозначим их S_3. Далее аналогично

$\boxed{\text{Замечание}}$ Эмпирически замечено, что лучше в конце алгоритма "перевернуть" нумерацию вершин. Называется обратный алгоритм Катхилла-Макки (reversed)



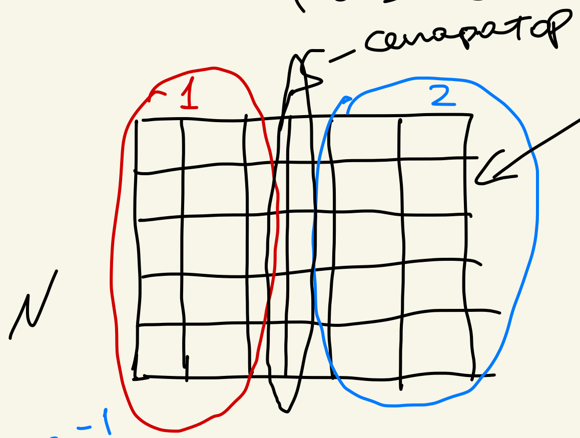
б) Minimal degree ordering



Сначала выбираем вершины с наименьшей степенью.

Approximate MD (AMD) - один из лучших алгоритмов.

в) Вложенное рассечение (nested dissection)



пример графа

S: в 1 и 2 не сильно отличается кол-во вершин

$A_{s1}A_1^{-1}\cdot$

$$\begin{bmatrix} A_1 & 0 & A_{1s} \\ 0 & A_2 & A_{2s} \\ A_{s1} & A_{s2} & A_s \end{bmatrix} \overset{A_{s2}A_2^{-1}\cdot}{\sim} \begin{bmatrix} A_1 & 0 & A_{1s} \\ 0 & A_2 & A_{2s} \\ 0 & A_{s2} & A_s - A_{s1}A_1^{-1}A_{1s} \end{bmatrix}$$

$$\sim \begin{bmatrix} A_1 & 0 & A_{1s} \\ 0 & A_2 & A_{2s} \\ 0 & 0 & \underbrace{A_s - A_{s1}A_1^{-1}A_{1s} - A_{s2}A_2^{-1}A_{2s}}_{\text{плотный блок}} \end{bmatrix}$$

Продолжаем разбиение для A_1, A_2 аналогично

4 МНК для разреж. матриц

$$\|Ax - b\|_2 \to \min$$

$A = QR$, Q может быть плотной даже при разреж. R

$$\underbrace{A^TA}_{R^TR}\, x = A^T b$$

т.к. система нормал. уравн. может привести к потере точности

$$A^TA(x_0 + \Delta x) = A^T b$$

$$A^TA\,\Delta x = A^T b - A^TA x_0$$

можно продолжить

Литература:

G. Golub, Ch. Van Loan Matrix Computations 4th edition, Sec. 11.1