МОСКОВСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
имени М.В. ЛОМОНОСОВА

Механико-математический факультет

Кафедра вычислительной математики

К.Ю. Богачев

Практикум на ЭВМ. Методы решения линейных систем и нахождения собственных значений

Москва 1998 год

Содержание

ПРЕДИСЛОВИЕ

Настоящее пособие содержит описания алгоритмов, предлагаемых к реализации на ЭВМ студентам механико-математического факультета МГУ на занятиях по "Практикуму на ЭВМ". Для всех алгоритмов приводится необходимое теоретическое обоснование, соответствующие расчетные соотношения и рекомендации по их практическому осуществлению на ЭВМ (организация процесса вычислений, хранения данных и результатов в памяти ЭВМ и т.п.).

Многообразие алгоритмов объясняется, с одной стороны, необходимостью обеспечить преподавателей достаточным набором задач для проведения занятий, а с другой стороны, желанием продемонстрировать различные подходы к решению задачи решения линейных систем и нахождения собственных значений матриц. Алгоритмы требуют разных вычислительных затрат, имеют разную чувствительность к погрешностям во входных данных, их свойства по-разному зависят от числа обусловленности матрицы. "Самого лучшего" метода решения поставленной задачи не существует, и выбор алгоритма зависит от конкретной задачи. Этот выбор будет различным, например, для симметричных и несимметричных матриц, для трехдиагональных и заполненных матриц, и т.д.

Подбор алгоритмов для Практикума диктовался, в основном, возможностью реализации их студентами при существующих ресурсах времени на ЭВМ, что привело к отказу от рассмотрения усложненных подходов. Часть описанных алгоритмов вытеснена из широкой вычислительной практики более эффективными (и более сложными) алгоритмами, рассмотреть которые в курсе "Практикум на ЭВМ" не представляется возможным. Тем не менее, эти алгоритмы представляют интерес для решения определенного круга задач и включены в пособие.

Форма отчетности студентов по Практикуму призвана стимулировать как развитие практических навыков решения математических задач с помощью компьютера, так и создание определенного кругозора в области существующих методов решения поставленной задачи. Поэтому в рамках Практикума студентам предлагается как разработать программу на ЭВМ, реализующую заданный алгоритм, так и письменно ответить хотя бы на половину вопросов из предложенного варианта, составленного из вопросов, приведенных в конце пособия.

В основе настоящего пособия лежат материалы лекций, читавшихся автором в течении 4-х лет в рамках факультативного курса "Практикум на ЭВМ". В электронном варианте оно уже более 5-ти лет используется при проведении занятий со студентами в дисплейном классе.

Предложения, замечания и отмеченные опечатки просьба сообщать автору на кафедру вычислительной математики.

Глава I.

ТОЧНЫЕ МЕТОДЫ РЕШЕНИЯ ЛИНЕЙНЫХ СИСТЕМ

§ 1. МАТРИЧНЫЕ НОРМЫ

Обозначим через $\mathbf{M}_n$ пространство (кольцо) матриц размера $n \times n$ над полем $\mathbf{C}$.

Определение. Нормой на кольце $\mathbf{M}_n$ называется неотрицательный функционал $\| \cdot \| : \mathbf{M}_n \to \mathbf{R}^1_+$, удовлетворяющий следующим условиям для всех $A, B \in \mathbf{M}_n$:

1) $\| A \| = 0 \Leftrightarrow A = 0$,
2) $\| \lambda A \| = |\lambda| \, \| A \|$ для всех $\lambda \in \mathbf{C}$,
3) $\| A + B \| \le \| A \| + \| B \|$,
4) $\| A \, B \| \le \| A \| \, \| B \|$.

Простейшие свойства нормы:

1) $\| I \| \ge 1$ (где I означает единичную матрицу).

Действительно, $\|I\| = \|I \cdot I\| \le \|I\| \, \|I\| = \|I\|^2$.

2) $\|A^{-1}\| \ge \dfrac{\|I\|}{\|A\|}$ и, в частности, $\|A^{-1}\| \, \|A\| \ge 1$.

Доказательство: $\|I\| = \|A \, A^{-1}\| \le \|A\| \, \|A^{-1}\|$.

Поскольку $\mathbf{M}_n$ можно рассматривать как векторное пространство размерности n^2, то на нем можно вести векторные нормы, некоторые из которых оказываются матричными. Для таких норм в проверке нуждается только свойство 4) (поскольку первые три свойства составляют определение векторной нормы).

Пример. $\|A\|_E = \|(a_{ij})_{i,j=1,\dots,n}\|_E = \left(\sum\limits_{i,j=1}^{n} |a_{ij}|^2 \right)^{1/2}$ является матричной нормой. Проверим свойство 4):

$$\|AB\|_E^2 = \sum_{i,j=1}^{n} \Big| \sum_{k=1}^{n} a_{ik} b_{kj} \Big|^2 \le \sum_{i,j=1}^{n} \left(\sum_{k=1}^{n} |a_{ik}|^2 \right) \left(\sum_{m=1}^{n} |b_{mj}|^2 \right) =$$

$$\Big(\sum_{i,k=1}^{n} |a_{ik}|^2 \Big) \Big(\sum_{j,m=1}^{n} |b_{mj}|^2 \Big) = \|A\|_E \|B\|_E .$$

Упражнение. Проверить, что выражение $\|A\|_{\ell_\infty} = \max\limits_{1\le i,j\le n} |a_{ij}|$ не является матричной нормой. Указание: рассмотреть матрицу

$$A = \begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix}.$$

Упражнение. Проверить, что выражение $\|A\| = n\,\|A\|_{\ell_\infty}$ является матричной нормой.

Определение. Пусть $\|\cdot\|$ есть векторная норма на пространсте $\mathbf{C}^n$. Опередим матричную норму $\|\cdot\|$ на $\mathbf{M}_n$ формулой

$$\|A\| = \max_{\|x\|=1} \|A\,x\|.$$

Такая норма называется *подчиненной*, или *операторной*, или *индуцированной* по отношению к векторной норме $\|\cdot\|$.

Проверим вначале, что

$$\|A\| = \max_{\|x\|=1} \|A\,x\| = \max_{x\neq 0} \frac{\|A\,x\|}{\|x\|}.$$

Действительно,

$$\max_{x\neq 0} \frac{\|A\,x\|}{\|x\|} = \max_{x\neq 0} \|A\,\frac{x}{\|x\|}\| \le \max_{\|y\|=1} \|A\,y\| = \|A\|,$$
$$\|A\| = \max_{\|x\|=1} \|A\,x\| = \max_{\|x\|=1} \frac{\|A\,x\|}{\|x\|} \le \max_{x\neq 0} \frac{\|A\,x\|}{\|x\|}$$

Теорема 1. *Так определенный функционал на $\mathbf{M}_n$ действительно является матричной нормой, которая обладает следующими свойствами:*

$$\|Ax\| \le \|A\|\,\|x\| \quad \text{для всех} \quad A \in \mathbf{M}_n,\ x \in \mathbf{C}^n,$$
$$\|I\| = 1.$$

Доказательство. Проверим выполнение свойств 1)-4) из определения матричной нормы.

1) $\|A\| = \max\limits_{\|x\|=1} \|A\,x\| \ge 0$;

2) $\|\lambda A\| = \max\limits_{\|x\|=1} \|\lambda A\,x\| = \max\limits_{\|x\|=1} |\lambda|\,\|A\,x\| = |\lambda| \max\limits_{\|x\|=1} \|A\,x\| = |\lambda|\,\|A\|$.

3) $\|A+B\| = \max\limits_{\|x\|=1} \|(A+B)\,x\| = \max\limits_{\|x\|=1} \|A\,x + B\,x\| \le \max\limits_{\|x\|=1} (\|A\,x\| + \|B\,x\|) \le \max\limits_{\|x\|=1} \|A\,x\| + \max\limits_{\|x\|=1} \|B\,x\| = \|A\| + \|B\|$.

4) $\|A\,B\| = \max\limits_{x\neq 0} \dfrac{\|A\,B\,x\|}{\|x\|} = \max\limits_{x\neq 0} \dfrac{\|A\,B\,x\|}{\|B\,x\|}\dfrac{\|B\,x\|}{\|x\|} \le \max\limits_{y\neq 0} \dfrac{\|A\,y\|}{\|y\|} \max\limits_{x\neq 0} \dfrac{\|B\,x\|}{\|x\|} \le \|A\|\,\|B\|$.

Проверим выполнение дополнительных свойств этой нормы.

$\|A\| = \max\limits_{x\neq 0} \dfrac{\|A\,x\|}{\|x\|}$. Следовательно, для всех $x \in \mathbf{C}^n$ $\dfrac{\|A\,x\|}{\|x\|} \leq \|A\|$, т.е. $\|A\,x\| \leq \|A\|\,\|x\|$.

$\|I\| = \max\limits_{\|x\|=1} \|I\,x\| = \max\limits_{\|x\|=1} \|x\| = 1$.

Определение. *Максимальной столбцовой нормой* называется

$$\|A\|_1 = \max_{1\leq j\leq n} \sum_{i=1}^{n} |a_{ij}|.$$

Лемма 1. *Норма* $\|A\|_1$ *подчинена векторной норме* $\|x\|_1 = \sum\limits_{i=1}^{n} |x_i|$.

Доказательство. Надо проверить, что $\max\limits_{1\leq j\leq n} \sum\limits_{i=1}^{n} |a_{ij}| = \max\limits_{\|x\|_1=1} \|A\,x\|_1$. Запишем матрицу через ее столбцы: $A = [a_1, \ldots, a_n]$, где $a_j = (a_{1j}, \ldots, a_{nj})^t$. Тогда $\|A\|_1 = \max\limits_{1\leq j\leq n} \|a_j\|_1$. С другой стороны, для $x = (x_1, \ldots, x_n)^t \in \mathbf{C}^n$

$$\|A\,x\|_1 = \|\sum_{j=1}^{n} x_j a_j\|_1 \leq \sum_{i=1}^{n} |x_j|\,\|a_j\|_1 \leq \sum_{j=1}^{n} |x_j| \max_{k=1,\ldots,n} \|a_k\|_1 = \|x\|_1\,\|A\|_1.$$

Следовательно, $\max\limits_{\|x\|_1=1} \|A\,x\|_1 = \max\limits_{x\neq 0} \dfrac{\|A\,x\|_1}{\|x\|_1} \leq \|A\|_1$.

Если максимум $\max\limits_{1\leq i\leq n} \|a_i\|_1 = \|A\|_1$ достигается при $i = i_0$, то, выбрав x равным e_{i_0} - стандартному координатному орту, получаем

$$\max_{\|x\|_1=1} \|A\,x\|_1 \geq \|A\,e_{i_0}\|_1 = \|a_{i_0}\| = \|A\|_1.$$

Отсюда вытекает требуемое равенство $\max\limits_{1\leq j\leq n} \sum\limits_{i=1}^{n} |a_{ij}| = \max\limits_{\|x\|_1=1} \|A\,x\|_1$.

Определение. *Максимальной строчной нормой* называется

$$\|A\|_\infty = \max_{1\leq i\leq n} \sum_{j=1}^{n} |a_{ij}|.$$

Лемма 2. *Норма* $\|A\|_\infty$ *подчинена векторной норме* $\|x\|_\infty = \max\limits_{i=1,\ldots,n} |x_i|$.

Задача. Доказать самостоятельно.

Доказательство. Имеем для всех $x \in \mathbf{C}^n$:

$$\|A\,x\|_\infty = \max_{1\leq i\leq n} |\sum_{j=1}^{n} a_{ij} x_j| \leq \max_{1\leq i\leq n} \sum_{j=1}^{n} |a_{ij}|\,|x_j| \leq \max_{1\leq i\leq n} \sum_{j=1}^{n} |a_{ij}|\,\|x\|_\infty = \|A\|_\infty\,\|x\|_\infty.$$

Следовательно, $\max\limits_{\|x\|_\infty=1} \|A\,x\|_\infty \leq \|A\|_\infty$.

Пусть максимум $\max\limits_{1\le i\le n}\sum\limits_{j=1}^{n}|a_{ij}| = \|A\|_\infty$ достигается при $i = i_0$: $\|A\|_\infty = \sum\limits_{j=1}^{n}|a_{i_0 j}|$. Рассмотрим вектор $y = (y_1,\dots,y_n)^t$ с компонентами

$$y_j = \begin{cases} \dfrac{\bar a_{i_0 j}}{|a_{i_0 j}|}, & \text{если} \quad a_{i_0 j}\neq 0,\\ 1, & \text{если} \quad a_{i_0 j}=0. \end{cases}$$

Тогда $\|y\|_\infty = 1$, $y_j a_{i_0 j} = |a_{i_0 j}|$, $j = 1,\dots,n$. Поэтому

$$\max_{\|x\|_\infty=1}\|A\,x\|_\infty \ge \|A\,y\|_\infty = \max_{1\le i\le n}|\sum_{j=1}^{n} a_{ij}y_j| \ge$$
$$|\sum_{j=1}^{n} a_{i_0 j}y_j| = |\sum_{j=1}^{n}|a_{i_0 j}|| = \sum_{j=1}^{n}|a_{i_0 j}| = \|A\|_\infty,$$

т.е. $\max\limits_{\|x\|_\infty=1}\|A\,x\|_\infty \ge \|A\|_\infty$. С учетом доказанного выше это означает требуемое равенство $\max\limits_{\|x\|_\infty=1}\|A\,x\|_\infty = \|A\|_\infty$.

Определение. *Спектральной нормой* называется

$$\|A\|_2 = \max\{\sqrt{\lambda} \,:\, \lambda \text{ собственное значение матрицы } A^*A\}.$$

(здесь A^* означает матрицу, сопряженную к матрице A : $A^* = (\bar A)^t$ в комплексном случае и транспонированную матрицу A^t в вещественном случае). Отметим, что это определение корректно, т.е. указанные квадратные корни всегда существуют. Действительно, для всякого собственного значения λ матрицы A^*A и соответствующего ему собственного вектора x из равенства $A^*A\,x = \lambda\,x$ вытекает $(\,A^*A\,x, x\,) = \lambda(\,x,x\,)$, откуда $\lambda = \dfrac{\|A\,x\|_2}{\|x\|_2}$, и потому $\lambda\in\mathbf{R}$, $\lambda\ge 0$ (здесь $(\cdot,\cdot)$ означает евклидово скалярное произведение в $\mathbf{C}^n$).

Лемма 3. *Норма* $\|A\|_2$ *подчинена евклидовой векторной норме* $\|x\|_2 = (\sum\limits_{i=1}^{n}|x_i|^2)^{1/2}$.

Доказательство. Рассмотрим

$$\max_{\|x\|_2=1}\|A\,x\|_2 = \max_{\|x\|_2=1}(\,A\,x, A\,x\,)^{1/2} = \max_{\|x\|_2=1}(\,A^*A\,x, x\,)^{1/2}.$$

Обозначим $B = A^*A$. Матрица B самосопряжена (симметрична в вещественном случае) : $B^* = (A^*A)^* = A^*(A^*)^* = A^*A = B$. Как показывается в курсе линейной алгебры для матрицы B существует такая унитарная (ортогональная в вещественном случае) матрица U , что $B = U^*\Lambda U$, где $\Lambda = \mathrm{diag}\,(\lambda_1,\dots,\lambda_n)$, λ_j - собственные значения матрицы $B = A^*A$. Поэтому

$$\max_{\|x\|_2=1}\|A\,x\|_2 = \max_{\|x\|_2=1}(\,U^*\Lambda U\,x, x\,)^{1/2} = \max_{\|x\|_2=1}(\,\Lambda U\,x, Ux\,)^{1/2}.$$

Поскольку матрица $U^{-1} = U^*$ является унитарной (ортогональной в вещественном случае) и не изменяет евклидовой длины векторов : $\|U^{-1}\,x\|_2 = \|x\|_2$ для всех x, то (заменяя в последнем неравенстве $y = U\,x$)

$$\max_{\|x\|_2=1} \|A\,x\|_2 = \max_{\|U^{-1}y\|_2=1} (\,\Lambda\,y, y\,)^{1/2} = \max_{\|y\|_2=1} (\,\Lambda\,y, y\,)^{1/2} = \max_{\|y\|_2=1} (\sum_{j=1}^{n} \lambda_j\,|y_j|^2)^{1/2}.$$

Пусть λ_{j_0} - максимальное из λ_j . Тогда

$$\max_{\|x\|_2=1} \|A\,x\|_2 \le \sqrt{\lambda_{j_0}} \max_{\|y\|_2=1} (\sum_{j=1}^{n} |y_j|^2)^{1/2} = \sqrt{\lambda_{j_0}} = \|A\|_2.$$

С другой стороны, если e_{j_0} есть j_0-й координатный орт, то

$$\max_{\|x\|_2=1} \|A\,x\|_2 = \max_{\|y\|_2=1} (\sum_{j=1}^{n} \lambda_j\,|y_j|^2)^{1/2} \ge (\sum_{j=1}^{n} \lambda_j\,|e_{j_0 j}|^2)^{1/2} = \sqrt{\lambda_{j_0}} = \|A\|_2.$$

Из последних двух соотношений вытекает требуемое равенство

$$\max_{\|x\|_2=1} \|A\,x\|_2 = \|A\|_2.$$

Определение. *Спектральным радиусом* матрицы A называется

$$\rho(A) = \max\{\,|\lambda| \;:\; \lambda - \text{собственное значение матрицы } A\,\}.$$

Лемма 4. *Для всякой матрицы $A \in \mathbf{M}_n$ и всякой матричной нормы на $\mathbf{M}_n$ справедливо неравенство $\rho(A) \le \|A\|$.*

Доказательство. Пусть λ - максимальное по модулю собственное значение матрицы A , т.е. $|\lambda| = \rho(A)$, x - соответствующий собственный вектор, т.е. $A\,x = \lambda\,x$. Рассмотрим матрицу $X \in \mathbf{M}_n$, все столбцы которой равны собственному вектору x . Тогда $A\,X = \lambda\,X$. Для всякой матричной нормы $\|\cdot\|$ имеем $|\lambda|\,\|X\| = \|\lambda\,X\| = \|A\,X\| \le \|A\|\,\|X\|$, следовательно, $|\lambda| = \rho(A) \le \|A\|$.

Замечание 1. Если $A = A^*$, то $\rho(A) = \|A\|_2$. Это следует непосредственно из определения спектральной нормы и того, что собственные значения матрицы $A^*A = A^2$ равны квадратам собственных значений матрицы A .

Определение. *Унитарно инвариантной матричной нормой* называется матричная норма $\|\cdot\|$, удовлетворяющая равенству $\|A\| = \|UAV\|$ для всех матриц $A \in \mathbf{M}_n$ и всех унитарных матриц U, $V \in \mathbf{M}_n$.

Лемма 5. *Спектральная норма является унитарно инвариантной.*

Доказательство. Пусть A - произвольная матрица из $\mathbf{M}_n$, U, V - произвольные унитарные матрицы. Собственные значения матрицы

$$(UAV)^*(UAV) = V^*A^*U^*UAV = V^{-1}A^*U^{-1}UAV = V^{-1}A^*AV$$

те же, что и у матрицы A^*A . Следовательно, спектральные нормы матриц UAV и A совпадают.

§ 2. ОБРАТИМОСТЬ МАТРИЦЫ, БЛИЗКОЙ К ОБРАТИМОЙ МАТРИЦЕ

Теорема 1. *Пусть* $\|\cdot\|$ - *матричная норма на* $\mathbf{M}_n$. *Если* $\|A\| < 1$, *то матрица* $I - A$ *обратима, причем* $(I-A)^{-1} = \sum\limits_{k=0}^{\infty} A^k$.

Доказательство. Рассмотрим ряд $\sum\limits_{k=0}^{\infty} A^k$. Поскольку для всякого $p > 0$ $\|\sum\limits_{k=m}^{m+p} A^k\| \le \sum\limits_{k=m}^{m+p} \|A\|^k \to 0$ при $m \to \infty$, то последовательность частичных сумм $s_m = \sum\limits_{k=0}^{m} A^k$ является последовательностью Коши. Так как в силу полноты $\mathbf{C}^{n^2}$ пространство $\mathbf{M}_n$ полно по норме $\|\cdot\|$, то определен предел $B = \lim\limits_{m\to\infty} s_m = \sum\limits_{k=0}^{\infty} A^k$, прием в силу непрерывности нормы $\|B\| \le \sum\limits_{k=0}^{\infty} \|A\|^k = \dfrac{1}{1-\|A\|}$. Имеем: $B(I-A) = \sum\limits_{k=0}^{\infty} A^k(I-A) = \sum\limits_{k=0}^{\infty} A^k - \sum\limits_{k=1}^{\infty} A^k = I$. Аналогично проверяем $(I-A)B = I$. Следовательно, $B = (I-A)^{-1}$.

Теорема 2. *Пусть* $\|\cdot\|$ - *матричная норма на* $\mathbf{M}_n$, A - *обратимая матрица. Если* $B \in \mathbf{M}_n$, $\|B\| < \dfrac{1}{\|A^{-1}\|}$, *то матрица* $C = A + B$ *обратима, причем* $C^{-1} = (A+B)^{-1} = \sum\limits_{k=0}^{\infty} (-1)^k (A^{-1}B)^k A^{-1}$.

Доказательство. Рассмотрим матрицу $D = A^{-1}C = A^{-1}(A+B) = I + A^{-1}B$. Так как по условию $\|A^{-1}B\| \le \|A^{-1}\|\,\|B\| < 1$, то по предыдущей теореме существует матрица $D^{-1} = \sum\limits_{k=0}^{\infty} (-1)^k (A^{-1}B)^k$, обратная к D. Рассмотрим матрицу $\hat{C} = D^{-1}A^{-1}$ и вычислим $\hat{C}\,C = D^{-1}A^{-1}(A+B) = D^{-1}(I + A^{-1}B) = D^{-1}D = I$, $C\,\hat{C} = (A+B)D^{-1}A^{-1} = A\,A^{-1}(A+B)D^{-1}A^{-1} = A(I + A^{-1}B)D^{-1}A^{-1} = A\,A^{-1} = I$. Следовательно, существует $C^{-1} = \hat{C} = D^{-1}A^{-1} = \sum\limits_{k=0}^{\infty} (-1)^k (A^{-1}B)^k A^{-1}$.

§ 3. ОШИБКИ В РЕШЕНИЯХ ЛИНЕЙНЫХ СИСТЕМ

Пусть решается линейная система $A\,x = b$, $A \in \mathbf{M}_n$, $b \in \mathbf{C}^n$. Пусть ß- алгоритм решения этой системы на идеальной вычислительной системе, так, что $x = ß(A,b)$ – точное решение системы и потому $x = ß(A,b)$ – отображение на классе невырожденных матриц ($x = A^{-1}b$). Пусть $\hat{ß}$ - тот же алгоритм, реализованный на реальной вычислительной системе. В результате его проведения получено приближенное решение $\hat{x} = \hat{ß}(A,b)$. Определим матрицу $\hat{A}$ и вектор $\hat{b}$ из условия $\hat{x} = ß(\hat{A},\hat{b})$ ($\hat{b} = \hat{A}^{-1}\hat{x}$), т.е. $\hat{x}$ является точным решением системы

$\hat{A}\hat{x} = \hat{b}$. Обозначим $\hat{A} = A + E$, $\hat{b} = b + e$. Таким образом, точное решение x удовлетворяет системе $A\,x = b$, а приближенное решение $\hat{x}$ удовлетворяет системе $(A + E)\hat{x} = b + e$.

Пусть $\|\cdot\|$ - произвольная матричная норма, согласованная с векторной нормой $\|\cdot\|$. Будем считать, что ошибка, вносимая в матрицу при проведении алгоритма, не очень велика: $\|E\| < \dfrac{1}{\|A^{-1}\|}$. По теореме 2.2 отсюда следует, что матрица $A + E$ обратима и $(A + E)^{-1} = \sum\limits_{k=0}^{\infty}(-1)^k(A^{-1}E)^kA^{-1}$. Вычислим погрешность $x - \hat{x}$:

$$\begin{aligned} x - \hat{x} = \;& A^{-1}b - (A + E)^{-1}(b + e) = (A^{-1} - (A + E)^{-1})b - (A + E)^{-1}e \\ &= -\sum_{k=1}^{\infty}(-1)^k(A^{-1}E)^kA^{-1}b - \sum_{k=0}^{\infty}(-1)^k(A^{-1}E)^kA^{-1}e \\ &= -\sum_{k=1}^{\infty}(-1)^k(A^{-1}E)^kx - \sum_{k=0}^{\infty}(-1)^k(A^{-1}E)^kA^{-1}e. \end{aligned}$$

Следовательно,

$$\begin{aligned} \|x - \hat{x}\| &\le (\sum_{k=1}^{\infty}\|A^{-1}E\|^k)\|x\| + (\sum_{k=0}^{\infty}\|A^{-1}E\|^k)\|A^{-1}e\|, \\ \frac{\|x - \hat{x}\|}{\|x\|} &\le \frac{\|A^{-1}E\|}{1 - \|A^{-1}E\|} + \frac{1}{1 - \|A^{-1}E\|}\frac{\|A^{-1}e\|}{\|x\|}. \end{aligned}$$

Так как $A\,x = b$, то $\|b\| = \|A\,x\| \le \|A\|\,\|x\|$ и $\dfrac{1}{\|x\|} \le \dfrac{\|A\|}{\|b\|}$. Поэтому

$$\begin{aligned} \frac{\|x - \hat{x}\|}{\|x\|} &\le \frac{\|A^{-1}\|\,\|E\|}{1 - \|A^{-1}\|\,\|E\|} + \frac{1}{1 - \|A^{-1}\|\,\|E\|}\|A\|\,\|A^{-1}\|\,\frac{\|e\|}{\|b\|}, \\ \frac{\|x - \hat{x}\|}{\|x\|} &\le \frac{\|A\|\,\|A^{-1}\|\,\dfrac{\|E\|}{\|A\|}}{1 - \|A\|\,\|A^{-1}\|\,\dfrac{\|E\|}{\|A\|}} + \frac{1}{1 - \|A\|\,\|A^{-1}\|\,\dfrac{\|E\|}{\|A\|}}\|A\|\,\|A^{-1}\|\,\frac{\|e\|}{\|b\|} \end{aligned}$$

Определение. *Числом обусловленности* матрицы A по отношению к матричной норме $\|\cdot\|$ называется

$$\kappa(A) = \begin{cases} \|A\|\,\|A^{-1}\|, & \text{если } A \text{ невырождена,} \\ \infty, & \text{если } A \text{ вырождена.} \end{cases}$$

С использованием этого обозначения последнее неравенство можно переписать в виде

$$\frac{\|x - \hat{x}\|}{\|x\|} \le \frac{\kappa(A)}{1 - \kappa(A)\,\dfrac{\|E\|}{\|A\|}}\left(\frac{\|E\|}{\|A\|} + \frac{\|e\|}{\|b\|}\right).$$

Здесь $\dfrac{\|x-\hat{x}\|}{\|x\|}$ называется *относительной погрешностью в решении* x, $\dfrac{\|E\|}{\|A\|}$ называется *относительной погрешностью в матрице* A, $\dfrac{\|e\|}{\|b\|}$ называется *относительной погрешностью в правой части* b.

Часто, чтобы оценить точность полученного приближенного решения $\hat{x}$, вычисляют *вектор невязки* $r = b - A\,\hat{x}$. Оценим относительную погрешность решения через невязку r.

$$x - \hat{x} = A^{-1}b - \hat{x} = A^{-1}(b - A\,\hat{x}) = A^{-1}r,$$
$$\frac{\|x-\hat{x}\|}{\|x\|} = \frac{\|A^{-1}r\|}{\|x\|} \le \frac{1}{\|x\|}\,\|A^{-1}\|\,\|r\| \le \frac{\|A\|}{\|b\|}\,\|A^{-1}\|\,\|r\| = \kappa(A)\frac{\|r\|}{\|b\|}.$$

Пример. Рассмотрим линейную систему

$$A\,x = \begin{pmatrix} 1 & -1 \\ 1 & -1+\varepsilon \end{pmatrix} x = b = \begin{pmatrix} 1 \\ 1 \end{pmatrix}.$$

Ее точное решение $x = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$. Рассмотрим приближенное решение $\hat{x} = \begin{pmatrix} 1+\varepsilon^{-1/2} \\ \varepsilon^{-1/2} \end{pmatrix}$. Тогда вектор невязки $r = \begin{pmatrix} 0 \\ -\varepsilon^{1/2} \end{pmatrix}$, вектор ошибки $x - \hat{x} = \begin{pmatrix} -\varepsilon^{-1/2} \\ -\varepsilon^{-1/2} \end{pmatrix}$. Поэтому

а) относительная величина невязки $\dfrac{\|r\|}{\|b\|} = O(\varepsilon^{1/2}) \to 0$ $(\varepsilon \to 0)$;

б) относительная величина ошибки $\dfrac{\|x-\hat{x}\|}{\|x\|} = O(\varepsilon^{-1/2}) \to \infty$ $(\varepsilon \to 0)$;

Свойства числа обусловленности

1. $\kappa(A) \ge 1$.
2. $\kappa(A) = \kappa(A^{-1})$.
3. $\kappa(AB) \le \kappa(A)\kappa(B)$.

Первые три свойства следуют непосредственно из определения числа обусловленности и основных свойств матричных норм.

4. Если $A = A^*$, то по отношению к спектральной норме $\kappa(A) = \left|\dfrac{\lambda_{\max}(A)}{\lambda_{\min}(A)}\right|$, где $\lambda_{\max}(A)$ и $\lambda_{\min}(A)$ соответственно максимальное и минимальное по модулю собственные значения матрицы A.

Доказательство. В силу замечания 1.1 (см. лемму 1.4) для самосопряженных матриц $\|A\|_2 = \rho(A) = |\lambda_{\max}(A)|$, $\|A^{-1}\|_2 = \rho(A^{-1}) = |\lambda_{\max}(A^{-1})| = |\lambda_{\min}(A)|$. Поэтому $\kappa(A) = \|A\|\,\|A^{-1}\| = \left|\dfrac{\lambda_{\max}(A)}{\lambda_{\min}(A)}\right|$.

5. Для всякой матрицы $A \in \mathbf{M}_n$ число обусловленности относительно любой матричной нормы $\kappa(A) \geq \left|\dfrac{\lambda_{\max}(A)}{\lambda_{\min}(A)}\right|$.

Доказательство вытекает из леммы 1.4: $\|A\| \geq \rho(A) = |\lambda_{\max}(A)|$, $\|A^{-1}\| \geq \rho(A^{-1}) = |\lambda_{\max}(A^{-1})| = |\lambda_{\min}(A)|$; поэтому $\kappa(A) = \|A\|\,\|A^{-1}\| \geq \left|\dfrac{\lambda_{\max}(A)}{\lambda_{\min}(A)}\right|$.

6. Для всякой матрицы $A \in \mathbf{M}_n$ и любых унитарных (ортогональных) матриц $U, V \in \mathbf{M}_n$ число обусловленности относительно спектральной нормы $\kappa(A) = \kappa(UAV)$.

Это свойство непосредственно следует из леммы 1.5.

7. Для всякой невырожденной матрицы $A \in \mathbf{M}_n$

$$\kappa(A) \geq \sup_{\substack{B \in \mathbf{M}_n, \\ B \ -\text{вырожденная}}} \frac{\|A\|}{\|A - B\|}$$

Доказательство. Покажем, что для всякой вырожденной матрицы $B \in \mathbf{M}_n$ справедливо неравенство $\|A - B\| \geq 1/\|A^{-1}\|$. Предположим, что это не так, т.е. существует вырожденная матрица B, такая, что для матрицы $C = A - B$ выполнено $\|C\| < 1/\|A^{-1}\|$. Тогда по теореме 2.2 матрица $A - C = A - (A - B) = B$ обратима, что противоречит вырожденности B. Таким образом, установлено, что для всякой вырожденной матрицы B $\|A^{-1}\| \geq 1/\|A - B\|$. Следовательно, $\kappa(A) = \|A\|\,\|A^{-1}\| \geq \|A\|/\|A - B\|$ для всякой вырожденной $B \in \mathbf{M}_n$. Поскольку левая часть этого неравенства не зависит от B, то из него вытекает требуемое соотношение.

§ 4. МЕТОД ГАУССА

§ 4.1. Алгоритм метода Гаусса

Пусть требуется решить линейную систему $A\,x = b$, $A \in \mathbf{M}_n$:

$$\begin{array}{ccccccccccc} a_{11}x_1 & + & a_{12}x_2 & + & \dots & + & a_{1n}x_n & = & b_1 \\ a_{21}x_1 & + & a_{22}x_2 & + & \dots & + & a_{2n}x_n & = & b_2 \\ \vdots & & \vdots & & \ddots & & \vdots & & \vdots \\ a_{n1}x_1 & + & a_{n2}x_2 & + & \dots & + & a_{nn}x_n & = & b_n \end{array} \qquad (1)$$

Метод Гаусса состоит в том, что элементарными преобразованиями над строками матрицы она приводится к треугольному виду с главной диагональю, состоящей из единичных элементов (прямой ход метода Гаусса); полученная система с треугольной матрицей решается в явном виде (обратный ход метода Гаусса).

Предположим, что $a_{11} \neq 0$. Поделив первое уравнение системы (1) на a_{11}, перепишем его в виде

$$x_1 + c_{12}x_2 + \dots + c_{1n}x_n = y_1, \qquad (2)$$

где $c_{1j} = a_{1j}/a_{11}$, $j = 2, \ldots, n$, $y_1 = b_1/a_{11}$. Умножим уравнение (2) на a_{i1} и вычтем его из i-го уравнения системы (1) ($i = 2, \ldots, n$). В результате система (1) примет вид

$$\begin{array}{ccccccccc} x_1 & + & c_{12}x_2 & + & \ldots & + & c_{1n}x_n & = & y_1 \\ & & a_{22}^{(1)}x_2 & + & \ldots & + & a_{2n}^{(1)}x_n & = & b_2^{(1)} \\ & & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots \\ & & a_{n2}^{(1)}x_2 & + & \ldots & + & a_{nn}^{(1)}x_n & = & b_n^{(1)} \end{array} \tag{3}$$

где $a_{ij}^{(1)} = a_{ij} - c_{1j}a_{i1}$, $\quad b_i^{(1)} = b_i - y_1 a_{i1}$, $\quad i, j = 2, \ldots, n$. Далее этот процесс применяется к подматрице $A^{(1)} = (a_{ij}^{(1)})_{i,j=2,\ldots,n} \in \mathbf{M}_{n-1}$.

Пусть сделаны $k-1$, $k = 1, \ldots, n$ шагов этого процесса, т.е. система (1) преобразована к виду

$$\begin{array}{ccccccccccccc} x_1 & + & c_{12}x_2 & + \ldots & + & c_{1,k-1}x_{k-1} & + & c_{1,k}x_k & + \ldots + & c_{1n}x_n & = & y_1 \\ & & x_2 & + \ldots & + & c_{2,k-1}x_{k-1} & + & c_{2,k}x_k & + \ldots + & c_{2n}x_n & = & y_2 \\ & & \ddots & \vdots & & \vdots & & \vdots & \vdots \ddots \vdots & \vdots & \vdots & \vdots \\ & & & & & x_{k-1} & + & c_{k-1,k}x_k & + \ldots + & c_{k-1,n}x_n & = & y_{k-1} \\ & & & & & & & a_{k,k}^{(k-1)}x_k & + \ldots + & a_{k,n}^{(k-1)}x_n & = & b_k^{(k-1)} \\ & & & & & & & \vdots & \vdots \ddots \vdots & \vdots & \vdots & \vdots \\ & & & & & & & a_{n,k}^{(k-1)}x_k & + \ldots + & a_{n,n}^{(k-1)}x_n & = & b_n^{(k-1)} \end{array} \tag{4}$$

Предположим, что $a_{kk}^{(k-1)} \neq 0$. Поделив k-е уравнение системы (4) на $a_{kk}^{(k-1)}$, перепишем его в виде

$$x_k + c_{k,k+1}x_{k+1} + \ldots + c_{kn}x_n = y_k, \tag{5}$$

где

$$c_{kj} = \frac{a_{kj}^{(k-1)}}{a_{kk}^{(k-1)}}, \quad j = k+1, \ldots, n, \qquad y_k = \frac{b_k^{(k-1)}}{a_{kk}^{(k-1)}}. \tag{6}$$

Умножим уравнение (5) на $a_{ik}^{(k-1)}$ и вычтем его из i-го уравнения системы (4) ($i = k+1, \ldots, n$). В результате система (4) примет вид

$$\begin{array}{ccccccccccccc} x_1 & + & c_{12}x_2 & + \ldots & + & c_{1,k-1}x_{k-1} & + & c_{1,k}x_k & + & c_{1,k+1}x_{k+1} & + \ldots + & c_{1n}x_n & = & y_1 \\ & & x_2 & + \ldots & + & c_{2,k-1}x_{k-1} & + & c_{2,k}x_k & + & c_{2,k+1}x_{k+1} & + \ldots + & c_{2n}x_n & = & y_2 \\ & & \ddots & \vdots & & \vdots & & \vdots & & \vdots \ddots \vdots & \vdots & \vdots & \vdots \\ & & & & & x_{k-1} & + & c_{k-1,k}x_k & + & c_{k-1,k+1}x_{k+1} & + \ldots + & c_{k-1,n}x_n & = & y_{k-1} \\ & & & & & & & x_k & + & c_{k,k+1}x_{k+1} & + \ldots + & c_{k,n}x_n & = & y_k \\ & & & & & & & & & a_{k+1,k+1}^{(k)}x_{k+1} & + \ldots + & a_{k+1,n}^{(k)}x_n & = & b_{k+1}^{(k)} \\ & & & & & & & & & \vdots & \vdots \ddots \vdots & \vdots & \vdots & \vdots \\ & & & & & & & & & a_{n,k+1}^{(k)}x_{k+1} & + \ldots + & a_{n,n}^{(k)}x_n & = & b_n^{(k)} \end{array} \tag{7}$$

где

$$a_{ij}^{(k)} = a_{ij}^{(k-1)} - a_{ik}^{(k-1)} c_{kj}, \quad b_i^{(k)} = b_i^{(k-1)} - a_{ik}^{(k-1)} y_k, \quad i,j = k+1,\ldots,n. \qquad (8)$$

Выражения (6), (8) являются формулами перехода от системы (4) к системе (7). Если обозначить $a_{ij}^{(0)} = a_{ij},\ b_i^{(0)} = b_i,\ i,j = 1,\ldots,n$, то переход от системы (1) к системе (3) будет осуществляться по тем же формулам при $k = 1$.

После проведения вычислений по формулам (6), (8) при $k = 1,\ldots,n$ (которые составляют *прямой ход метода Гаусса*) система (1) примет вид

$$\begin{array}{ccccccccccc} x_1 & + & c_{12}x_2 & + & \ldots & + & c_{1n}x_{n-1} & + & c_{1n}x_n & = & y_1 \\ & & x_2 & + & \ldots & + & c_{2n}x_{n-1} & + & c_{2n}x_n & = & y_2 \\ & & & & \ddots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ & & & & & & x_{n-1} & + & c_{n-1,n}x_n & = & y_{n-1} \\ & & & & & & & & x_n & = & y_n \end{array} \qquad (9)$$

Решение системы (9) с треугольной матрицей может быть найдено непосредственно (методом последовательного исключения неизвестных в порядке $x_n, x_{n-1}, \ldots, x_1$):

$$x_n = y_n, \quad x_i = y_i - \sum_{j=i+1}^{n} c_{ij}x_j, \quad i = n-1,\ldots,1. \qquad (10)$$

Вычисления по формулам (10) составляют *обратный ход метода Гаусса.*

§ 4.2. Оценка количества арифметических операций в методе Гаусса

Здесь и далее при оценке количества арифметических операций мы будем отдельно находить количество аддитивных операций (сложений и вычитаний) и количество мультипликативных операций (умножений и делений). Для упрощения выкладок мы будем находить только главный член асимптотики количества операций при $n \to \infty$.

1. На вычисление c_{kj} при $j = k+1,\ldots,n,\ k = 1,\ldots,n$ по формулам (6) требуется $\sum_{k=1}^{n}(n-k) = \sum_{i=0}^{n-1} i = n(n-1)/2 = O(n^2)$ $(n \to \infty)$ операций деления.

2. На вычисление $a_{ij}^{(k)}$ при $i,j = k+1,\ldots,n,\ k = 1,\ldots,n$ по формулам (8) требуется $\sum_{k=1}^{n}(n-k)^2 = \sum_{i=0}^{n-1} i^2 = (n-1)n(2n-1)/6 = n^3/3 + O(n^2)$ $(n \to \infty)$ операций умножения и столько же операций вычитания.

Итак, на вычисление коэффициентов $c_{kj},\quad j = k+1,\ldots,n,\ k = 1,\ldots,n$ системы (9) требуется $O(n^2) + n^3/3 + O(n^2) = n^3/3 + O(n^2)$ $(n \to \infty)$ мультипликативных операций и столько же аддитивных операций.

3. На вычисление y_k при $k = 1,\ldots,n$ по формулам (6) требуется n операций деления.

4. На вычисление $b_i^{(k)}$ при $i = k+1,\ldots,n,\ k = 1,\ldots,n$ по формулам (8) требуется $\sum_{k=1}^{n}(n-k) = \sum_{i=0}^{n-1} i = n(n-1)/2 = O(n^2)$ $(n \to \infty)$ операций умножения и столько же операций вычитания.

Итак, на вычисление правых частей $y_k, \quad k = 1, \ldots, n$ системы (9) требуется $O(n^2)$ $(n \to \infty)$ мультипликативных операций и столько же аддитивных операций.

Таким образом, прямой ход метода Гаусса требует $n^3/3 + O(n^2) + O(n^2) = n^3/3 + O(n^2)$ $(n \to \infty)$ мультипликативных операций и столько же аддитивных операций.

5. На вычисление решения по формулам (10) (т.е. на проведение обратного хода метода Гаусса) требуется $\sum_{i=1}^{n-1}(n-i) = \sum_{i=1}^{n-1} i = n(n-1)/2 = O(n^2)$ $(n \to \infty)$ операций умножения и столько же операций вычитания.

Следовательно, метод Гаусса требует $n^3/3+O(n^2)+O(n^2) = n^3/3+O(n^2)$ $(n \to \infty)$ мультипликативных операций и столько же аддитивных операций. Всего: $(2/3)\, n^3 + O(n^2)$ $(n \to \infty)$ арифметических операций.

§ 4.3. Представление метода Гаусса в виде последовательности элементарных преобразований

Преобразование системы, задаваемое формулами (6), эквивалентно умножению матрицы системы слева на матрицу

$$D_k = \operatorname{diag}[\, \underbrace{1, \ldots, 1}_{k-1}, \left(a_{kk}^{(k-1)}\right)^{-1}, 1, \ldots, 1\,],$$

(где $\operatorname{diag}[\, d_1, \ldots, d_n\,]$ означает диагональную матрицу с элементами $d_1, \ldots, d_n$ на главной диагонали).

Преобразование системы, задаваемое формулами (8), эквивалентно умножению матрицы системы слева на матрицу

$$L_k = \begin{pmatrix} 1 & & & & \\ & \ddots & & & \\ & & 1 & & \\ & & -a_{k+1,k}^{(k-1)} & 1 & \\ & & \vdots & & \ddots \\ & & -a_{n,k}^{(k-1)} & & 1 \end{pmatrix}$$

(не обозначенные элементы матрицы L_k равны нулю).

Следовательно, прямой ход метода Гаусса эквивалентен умножению матрицы системы (1) последовательно на матрицы $L_k D_k, \quad k = 1, \ldots, n$:

$$L_n D_n \cdot \ldots \cdot L_1 D_1 \cdot Ax = L_n D_n \cdot \ldots \cdot L_1 D_1 \cdot b. \tag{11}$$

причем матрица $U = L_n D_n \cdot \ldots \cdot L_1 D_1 \cdot A$ есть матрица системы (9), т.е. является верхней треугольной с единицами на главной диагонали. Обозначим через $\mathrm{LT}\,(n)$ группу невырожденных верхних треугольных матриц. Тогда для всех $k = 1, \ldots, n \quad D_k, L_k \in \mathrm{LT}\,(n)$ и потому матрица $\hat{L} = L_n D_n \cdot \ldots \cdot L_1 D_1 \in \mathrm{LT}\,(n)$.

Из (11) получаем: $U = \hat{L}A$, откуда $A = LU$, где $L = \hat{L}^{-1} \in \mathrm{LT}(n)$, U верхняя треугольная с единицами на главной диагонали. Полученное представление матрицы A называется *LU-разложением*. Итак, показано, что метод Гаусса эквивалентен построению LU-разложения.

Знание LU-разложения матрицы A может быть полезно, например, в следующей ситуации. Пусть требуется решить ряд систем вида $Ax_m = b_m$, $m = 1, \ldots, M$ с одной и той же матрицей A и разными правыми частями b_m. Применяя M раз метод Гаусса это можно сделать за $(2/3)\,M\,n^3 + O(M\,n^2)$ $(n \to \infty)$ арифметических операций. Предположим, что нам известно LU-разложение матрицы A: $A = LU$. Тогда решение каждой из систем $Ax_m = b_m$ может быть сведено к решению двух систем $Ly_m = b_m$, $Ux_m = y_m$ с треугольными матрицами. Решение системы с треугольной матрицей осуществляется обратным ходом метода Гаусса за $O(n^2)$ $(n \to \infty)$ арифметических операций. Следовательно, если LU-разложение матрицы A уже построено и на это построение потребовалось $d(n)$ арифметических операций, то решение M систем с той же матрицей A потребует $d(n) + O(Mn^2)$ $(n \to \infty)$ арифметических операций. Подчеркнем, что число $d(n)$ не зависит от M. Сейчас мы построим алгоритм для получения LU-разложения и покажем, что $d(n) = (2/3)n^3 + O(n^2)$ $(n \to \infty)$ (т.е. равно числу операций, необходимых для проведения метода Гаусса).

§ 4.4. Алгоритм построения LU-разложения

Пусть требуется найти нижнюю треугольную матрицу $L = (l_{ij})$ и верхнюю треугольную матрицу $U = (u_{ij})$ с единицами на главной диагонали такую, что $A = LU$, т.е.

$$\sum_{j=1}^{n} l_{ij}\,u_{jk} = a_{ik}, \qquad i,k = 1, \ldots, n. \tag{12}$$

Поскольку $l_{ij} = 0$ при $i < j$, $u_{jk} = 0$ при $j > k$, $u_{jj} = 1$, то (12) есть система из n^2 уравнений относительно $n(n+1)/2$ неизвестных l_{ij}, $i \geq j$ и $n(n-1)/2$ неизвестных u_{jk}, $j < k$, всего $n(n+1)/2 + n(n-1)/2 = n^2$ неизвестных. Получим формулы для решения системы (12), которые и составляют алгоритм нахождения LU-разложения.

В силу $l_{ij} = 0$ при $i < j$, $u_{jk} = 0$ при $j > k$ сумма в (12) имеет вид

$$\sum_{j=1}^{\min\{i,k\}} l_{ij}\,u_{jk} = a_{ik}, \qquad i,k = 1, \ldots, n,$$

или

$$\begin{cases} \sum\limits_{j=1}^{k} l_{ij}\,u_{jk} = a_{ik}, & k \leq i, \quad i,k = 1, \ldots, n, \\ \sum\limits_{j=1}^{i} l_{ij}\,u_{jk} = a_{ik}, & k > i, \quad i,k = 1, \ldots, n. \end{cases}$$

Выделим в первой из этих сумм отдельно случай $k = 1$, а во второй - случай $i = 1$, и учтем, что $u_{kk} = 1$ для всех $k = 1, \ldots, n$,

$$\left[\begin{array}{l} \left[\begin{array}{ll} l_{i1} = a_{i1}, & i = 1, \ldots, n, \\ \sum\limits_{j=1}^{k-1} l_{ij}\, u_{jk} + l_{ik} = a_{ik}, & 1 < k \le i, \quad i, k = 2, \ldots, n, \end{array}\right. \\ \left[\begin{array}{ll} l_{11}\, u_{1k} = a_{1k}, & k = 2, \ldots, n, \\ \sum\limits_{j=1}^{i-1} l_{ij}\, u_{jk} + l_{ii}\, u_{ik} = a_{ik}, & k > i > 1, \quad i, k = 2, \ldots, n. \end{array}\right. \end{array}\right.$$

Перегруппируем эти формулы:

$$\left[\begin{array}{l} \left[\begin{array}{ll} l_{i1} = a_{i1}, & i = 1, \ldots, n, \\ u_{1k} = a_{1k}/l_{11}, & k = 2, \ldots, n, \end{array}\right. \\ \left[\begin{array}{ll} l_{ik} = a_{ik} - \sum\limits_{j=1}^{k-1} l_{ij}\, u_{jk}, & 1 < k \le i, \quad i, k = 2, \ldots, n, \\ u_{ik} = (a_{ik} - \sum\limits_{j=1}^{i-1} l_{ij}\, u_{jk})/l_{ii}, & k > i > 1, \quad i, k = 2, \ldots, n. \end{array}\right. \end{array}\right. \tag{13}$$

Процесс вычислений по этим формулам строится следующим образом: вначале по первой из формул (13) вычисляются неизвестные элементы первого столбца матрицы L: $l_{i1},\ i = 1, \ldots, n$, затем по второй из формул (13) вычисляются неизвестные элементы первой строки матрицы U: $u_{1k},\ k = 2, \ldots, n$ (напомним, элемент u_{11} известен, он равен 1). Далее в вычислениях участвуют только третья и четвертая из формул (13). По третьей формуле (13) вычисляются неизвестные элементы второго столбца матрицы L: $l_{i2},\ i = 2, \ldots, n$ (напомним, $l_{12} = 0$, так как L -нижняя треугольная)

$$l_{i2} = a_{i2} - l_{i1}\, u_{12}, \quad i = 2, \ldots, n.$$

По четвертой формуле (13) вычисляются неизвестные элементы второй строки матрицы U: $u_{2k},\ k = 3, \ldots, n$ (напомним, $u_{21} = 0$, так как U -верхняя треугольная, $u_{22} = 1$, так как U имеет единичную главную диагональ)

$$u_{2k} = (a_{2k} - l_{21}\, u_{1k})/l_{22}, \quad k = 3, \ldots, n.$$

Затем по третьей формуле (13) вычисляются неизвестные элементы третьего столбца матрицы L: $l_{i3},\ i = 3, \ldots, n$, а по четвертой формуле (13) вычисляются неизвестные элементы третьей строки матрицы U: $u_{3k},\ k = 4, \ldots, n$ и так далее.

Замечание 1. Организация хранения матриц L и U в памяти. Формулы (13) таковы, что при вычислении элемента l_{ij} или u_{ij} используются значения элемента a_{ij} и вычисленных ранее элементов l_{km}, $m < j$ и u_{km}, $k < i$. Это позволяет хранить нижнюю треугольную матрицу L на месте нижнего треугольника матрицы A: $l_{ij} \equiv a_{ij},\ i \ge j,\ i, j = 1, \ldots, n$, а вернюю треугольную матрицу U (без единичной главной диагонали) - на месте верхнего треугольника матрицы A: $u_{ij} \equiv a_{ij},\ i < j,\ i, j = 1, \ldots, n$.

§ 4.5. Оценка количества арифметических операций в алгоритме построения LU-разложения

1. При фиксированном $k = 1, \ldots, n$ вычисление элементов l_{ik} для всех $i = k, \ldots, n$ по третьей формуле (13) требует $\sum_{i=k}^{n}(k-1) = (n-k+1)(k-1)$ мультипликативных и столько же аддитивных операций. Следовательно, вычисление всех элементов матрицы L требует $\sum_{k=1}^{n}(n-k+1)(k-1) = n\sum_{k=1}^{n}(k-1) - \sum_{k=1}^{n}(k-1)^2 = n\sum_{i=0}^{n-1} i - \sum_{i=0}^{n-1} i^2 = n^2(n-1)/2 - (n-1)n(2n-1)/6 = n^3/2 - n^3/3 + O(n^2) = n^3/6 + O(n^2)$ $(n \to \infty)$ мультипликативных и столько же аддитивных операций.

2. При фиксированном $i = 1, \ldots, n$ вычисление элементов u_{ik} для всех $k = i+1, \ldots, n$ по четвертой формуле (13) требует $\sum_{k=i+1}^{n} i = (n-i)i$ мультипликативных и $\sum_{k=i+1}^{n}(i-1) = (n-i)(i-1)$ аддитивных операций. Следовательно, вычисление всех элементов матрицы U требует $\sum_{i=1}^{n}(n-i)i = n^2(n+1)/2 - n(n+1)(2n+1)/6 = n^3/6 + O(n^2)$ $(n \to \infty)$ мультипликативных и $\sum_{i=1}^{n}(n-i)(i-1) = n^2(n-1)/2 - n(n+1)(2n+1)/6 + n(n+1)/2 = n^3/6 + O(n^2)$ $(n \to \infty)$ аддитивных операций.

Таким образом, алгоритм построения LU-разложения требует для своего проведения выполнения $n^3/3 + O(n^2)$ $(n \to \infty)$ мультипликативных и столько же аддитивных операций, а в сумме — $(2/3)\,n^3 + O(n^2)$ $(n \to \infty)$ арифметических операций.

§ 4.6. Осуществимость метода Гаусса

Теорема 1. *Метод Гаусса осуществим (т.е. возможно LU-разложение) тогда и только тогда, когда для всех $k = 1, \ldots, n$ ее главные угловые миноры* $\det A_k \neq 0$ *, где*

$$A_k = \begin{pmatrix} a_{11} & a_{12} & \ldots & a_{1k} \\ a_{21} & a_{22} & \ldots & a_{2k} \\ \vdots & \vdots & \ddots & \vdots \\ a_{k1} & a_{k2} & \ldots & a_{kk} \end{pmatrix}.$$

— главные угловые подматрицы матрицы A.

Доказательство. Можно показать (мы этого делать не будем), что если $\det A_k \neq 0$ для всех $k = 1, \ldots, n$, то существует и единственно LU-разложение матрицы A.

Обозначим через L_k и U_k главные угловые подматрицы матриц L и U. Так как L – нижняя треугольная, а U – верхняя треугольная матрицы, и $A = LU$, то $A_k = L_k U_k$. Следовательно, $\det A_k = \det L_k \det U_k$. Но матрица U -верхняя треугольная с 1 на главной диагонали, поэтому $\det U_k = 1$ для всех $k = 1, \ldots, n$. Значит, $\det A_k = \det L_k$. Матрица L_k -нижняя треугольная, поэтому $\det L_k = l_{11} \ldots l_{kk}$. Следовательно, $\det A_k = l_{11} \ldots l_{kk}$.

Пусть для всех $k = 1, \ldots, n$ $\det A_k \neq 0$. Тогда $l_{11} = \det A_1 \neq 0$, $l_{kk} = \det A_k / \det A_{k-1} \neq 0$ и в формулах (13) возможно осуществить деление на l_{ii}, $i =$

$1, \ldots, n$. Следовательно, осуществим алгорим построения LU-разложения, а значит, и метод Гаусса.

Пусть осуществим алгорим построения LU-разложения, т.е. в формулах (13) возможно осуществить деление на l_{ii}, $i = 1, \ldots, n$. Следовательно, для всех $i = 1, \ldots, n$ $l_{ii} \neq 0$ и потому $\det A_k = l_{11} \ldots l_{kk} \neq 0$ для всех $k = 1, \ldots, n$.

§ 5. МЕТОДЫ ПОСЛЕДОВАТЕЛЬНОГО ИСКЛЮЧЕНИЯ НЕИЗВЕСТНЫХ ДЛЯ ЛЕНТОЧНЫХ МАТРИЦ

Определение. Матрица $A \in \mathbf{M}_n$ называется *ленточной*, если $a_{ij} = 0$ при $i - j > k_1$ либо $j - i > k_2$. Величина $k_1 + k_2 + 1$ называется *шириной ленты*. Если $k_1 = k_2 = 1$, то матрица называется *трехдиагональной*.

§ 5.1. Метод Гаусса для ленточных матриц

Расчетные формулы метода Гаусса (4.6), (4.8), (4.10) остаются справедливыми и для ленточных матриц, но объем вычислений по ним может быть сокращен при учете структуры матрицы A. В формулах прямого хода (4.6), (4.8) вычисления надо вести для $j = k + 1, \ldots, k + k_2$, $i = k + 1, \ldots, k + k_1$. Ширина ленты системы (4.9), получающейся после прямого хода, равна $k_2 + 1$, поэтому в формуле (4.10) обратного хода метода Гаусса надо учитывать только $k_2 + 1$ слагаемых.

Аналогично тому, как это было сделано выше, можно подсчитать, что

1. На вычисление c_{kj} при $j = k+1, \ldots, k+k_2$, $k = 1, \ldots, n$ по формулам (4.6) требуется $n\,k_2 + O(1)$ операций деления.

2. На вычисление $a_{ij}^{(k)}$ при $i = k+1, \ldots, k+k_1$, $j = k+1, \ldots, k+k_2$, $k = 1, \ldots, n$ по формулам (4.8) требуется $n\,k_1\,k_2 + O(1)$ операций умножения и столько же операций вычитания.

Итак, на вычисление коэффициентов c_{kj}, $j = k + 1, \ldots, k + k_2$, $k = 1, \ldots, n$ системы (4.9) требуется $n(k_1\,k_2 + k_2) + O(1)$ мультипликативных и столько же аддитивных операций.

3. На вычисление y_k при $k = 1, \ldots, n$ по формулам (4.6) требуется n операций деления.

4. На вычисление $b_i^{(k)}$ при $i = k+1, \ldots, k+k_1$, $k = 1, \ldots, n$ по формулам (4.8) требуется $n\,k_1 + O(1)$ операций умножения и столько же операций вычитания.

Итак, на вычисление правых частей y_k, $k = 1, \ldots, n$ системы (4.9) требуется $n(k_1 + 1)$ мультипликативных и столько же аддитивных операций.

Таким образом, прямой ход метода Гаусса требует $n(k_1\,k_2 + k_1 + k_2 + 1)$ мультипликативных операций и столько же аддитивных операций.

5. На вычисление решения по формулам (4.10) (т.е. на проведение обратного хода метода Гаусса) требуется $n\,k_2 + O(1)$ операций умножения и столько же операций вычитания.

Следовательно, метод Гаусса требует $n(k_1\,k_2+k_1+2\,k_2+1)$ мультипликативных операций и столько же аддитивных операций. Всего: $2n(k_1\,k_2+k_1+2\,k_2+1)$ арифметических операций.

Аналогичное сокращение вычислительной работы может быть произведено для алгоритма LU-разложения.

§ 5.2. Алгоритм LU-разложения для трехдиагональных матриц

Пусть требуется трехдиагональную матрицу вида

$$A=\begin{pmatrix} a_1 & c_1 & & & & \\ d_1 & a_2 & c_2 & & & \\ & d_2 & a_3 & \ddots & & \\ & & \ddots & \ddots & c_{n-2} & \\ & & & d_{n-2} & a_{n-1} & c_{n-1} \\ & & & & d_{n-1} & a_n \end{pmatrix} \tag{1}$$

представить в виде $A=LU$, где

$$L=\begin{pmatrix} l_1 & & & & \\ \lambda_1 & l_2 & & & \\ & \lambda_2 & l_3 & & \\ & & \ddots & \ddots & \\ & & & \lambda_{n-1} & l_n \end{pmatrix},\quad U=\begin{pmatrix} 1 & u_1 & & & \\ & 1 & u_2 & & \\ & & 1 & \ddots & \\ & & & \ddots & u_{n-1} \\ & & & & 1 \end{pmatrix}. \tag{2}$$

В этом случае формулы (4.13) алгоритма построения LU-разложения могут быть получены путем непосредственного перемножения матриц L и U и решения получающихся уравнений:

$$\begin{array}{llll} l_1 = a_1, & u_1 = c_1/l_1 & & \\ \lambda_1 = d_1, & l_2 = a_2-\lambda_1 u_1, & u_2 = c_2/l_2 & \\ \quad\vdots & \quad\vdots & \quad\vdots & \\ \lambda_i = d_i, & l_{i+1} = a_{i+1}-\lambda_i u_i, & u_{i+1} = c_{i+1}/l_{i+1} & \\ \quad\vdots & \quad\vdots & \quad\vdots & \\ \lambda_{n-2} = d_{n-2}, & l_{n-1} = a_{n-1}-\lambda_{n-2}u_{n-2}, & u_{n-1} = c_{n-1}/l_{n-1} & \\ \lambda_{n-1} = d_{n-1} & l_n = a_n-\lambda_{n-1}u_{n-1} & & \end{array} \tag{3}$$

Построение LU-разложения по этим формулам требует $n-1$ аддитивных и $2(n-1)$ мультипликативных операций.

Решение линейной системы $A\,x=b$ может быть осуществлено следующим образом:

1. Вначале строится LU-разложение матрицы A по формулам (3), на это требуется $n-1$ аддитивных и $2(n-1)$ мультипликативных операций.

2. Затем находится решение y линейной системы $L\,y = b$ путем последовательного исключения неизвестных, начиная с первого уравнения:

$$y_1 = b_1/l_1, \qquad y_i = (b_i - \lambda_{i-1}y_{i-1})/l_i, \quad i = 2, \ldots, n. \tag{4}$$

На этом этапе требуется произвести $n-1$ аддитивных и $2(n-1)+1$ мультипликативных операций.

3. Искомый вектор x находится как решение линейной системы $U\,x = y$ путем последовательного исключения неизвестных, начиная с последнего уравнения:

$$x_n = y_n, \qquad x_i = y_i - u_i x_{i+1}, \quad i = n-1, \ldots, 1. \tag{5}$$

На этом этапе требуется произвести $n-1$ аддитивных и $n-1$ мультипликативных операций.

Складывая оценки трудоемкости на каждом шаге, находим, что для осуществления алгоритма требуется произвести $3(n-1)$ аддитивных и $5(n-1)+1$ мультипликативных операций.

Замечание 1. Хранение в памяти ЭВМ матриц A, L и U. Матрицу вида (1) в памяти ЭВМ обычно не хранят. Вместо этого запоминают вектора $a = (a_1, \ldots, a_n)^t$, $c = (c_1, \ldots, c_{n-1})^t$, $d = (d_1, \ldots, d_{n-1})^t$. Аналогично, вместо матрицы L хранятся вектора $l = (l_1, \ldots, l_n)^t$, $\lambda = (\lambda_1, \ldots, \lambda_{n-1})^t$, а вместо матрицы U – вектор $u = (u_1, \ldots, u_{n-1})^t$. Как и в обычном алгоритме LU-разложения, матрицы L и U можно хранить на месте матрицы A : вектор l – на месте вектора a , вектор λ – на месте d , u – на месте c.

Замечание 2. Вспомогательный вектор y в приведенном выше алгоритме решения линейной системы может хранится на месте вектора x. Это следует из формул (4), (5), которые можно записать в виде

$$x_1 = b_1/l_1, \quad x_i = (b_i - \lambda_{i-1}x_{i-1})/l_i,\ i = 2, \ldots, n, \quad x_i := x_i - u_i x_{i+1},\ i = n-1, \ldots, 1.$$

§ 5.3. Метод прогонки для трехдиагональных матриц

Пусть требуется трехдиагональную матрицу A вида (1) представить в виде $A = LU$, где

$$L = \begin{pmatrix} l_1 & & & & \\ 1 & l_2 & & & \\ & 1 & l_3 & & \\ & & \ddots & \ddots & \\ & & & 1 & l_n \end{pmatrix}, \quad U = \begin{pmatrix} v_1 & u_1 & & & \\ & v_2 & u_2 & & \\ & & v_3 & \ddots & \\ & & & \ddots & u_{n-1} \\ & & & & v_n \end{pmatrix}.$$

В отличие от LU-разложения, это представление не единственно. Перемножая матрицы L и U, находим уравнения для определения коэффициентов l_i, v_i, u_i :

$$\begin{array}{llll} & l_1 v_1 & = a_1, & l_1 u_1 = c_1 \\ v_{i-1} = d_{i-1}, & u_{i-1} + l_i v_i & = a_i, & l_i u_i = c_i, \quad i = 2, \ldots, n-1 \\ v_{n-1} = d_{n-1}, & u_{n-1} + l_n v_n & = a_n. & \end{array} \tag{6}$$

Число уравнений в этой системе, равное $2 + 3(n-2) + 2 = 3n - 2$, на 1 меньше числа неизвестных l_i, v_i, u_i . Перепишем (6) в виде

$$\begin{array}{llll} v_1 = d_1, & l_1 v_1 & = a_1, & l_1 u_1 = c_1 \\ v_i = d_i, & u_{i-1} + l_i v_i & = a_i, & l_i u_i = c_i, \quad i = 2, \ldots, n-1 \\ & u_{n-1} + l_n v_n & = a_n. & \end{array}$$

Отсюда получаем расчетные формулы:

$$\begin{array}{llll} v_1 = d_1, & l_1 & = a_1 / v_1, & u_1 = c_1 / l_1 \\ v_i = d_i, & l_i & = (a_i - u_{i-1}) / v_i, & u_i = c_i / l_i, \quad i = 2, \ldots, n-1 \\ & l_n v_n & = a_n - u_{n-1}. & \end{array} \tag{7}$$

Из последнего уравнения в (7) мы можем определить только произведение $l_n v_n$. Зафиксировав один из параметров l_n или v_n, мы определим второй. Мы будем считать, что $v_n = 1$. В этом случае, количество арифметических операций, необходимое для осуществления разложения по формулам (7) равно количеству арифметических операций, необходимому для осуществления разложения по формулам (3).

Решение линейной системы $A\,x = b$ может быть осуществлено следующим образом:

1. Вначале строится LU-разложение матрицы A по формулам (7), на это требуется $n-1$ аддитивных и $2(n-1)$ мультипликативных операций.

2. Затем находится решение y линейной системы $L\,y = b$ путем последовательного исключения неизвестных, начиная с первого уравнения:

$$y_1 = b_1 / l_1, \qquad y_i = (b_i - y_{i-1}) / l_i, \quad i = 2, \ldots, n. \tag{8}$$

На этом этапе требуется произвести $n-1$ аддитивных и $(n-1)+1$ мультипликативных операций.

3. Искомый вектор x находится как решение линейной системы $U\,x = y$ путем последовательного исключения неизвестных, начиная с последнего уравнения:

$$x_n = y_n / v_n, \qquad x_i = (y_i - u_i x_{i+1}) / v_i, \quad i = n-1, \ldots, 1. \tag{9}$$

На этом этапе требуется произвести $n-1$ аддитивных и $2(n-1)$ мультипликативных операций (с учетом $v_n = 1$).

Складывая оценки трудоемкости на каждом шаге, находим, что для осуществления алгоритма требуется произвести $3(n-1)$ аддитивных и $5(n-1)+1$ мультипликативных операций.

Замечание 3. Хранение в памяти ЭВМ матриц A, L и U осуществляется так, как описано в Замечании 1. Вместо матрицы A запоминают вектора $a = (a_1, \ldots, a_n)^t$, $c = (c_1, \ldots, c_{n-1})^t$, $d = (d_1, \ldots, d_{n-1})^t$, вместо матрицы L – вектор $l = (l_1, \ldots, l_n)^t$, а вместо матрицы U – вектора $v = (v_1, \ldots, v_n)^t, u = (u_1, \ldots, u_{n-1})^t$. Аналогично алгоритму LU-разложения, матрицы L и U можно хранить на месте матрицы A: вектор l – на месте вектора a, вектор v – на месте d (последняя, не определяемая однозначно компонента v_n вектора v не хранится; ее можно считать равной 1), u – на месте c.

Замечание 4. Вспомогательный вектор y в приведенном выше алгоритме решения линейной системы может хранится на месте вектора x. Это следует из формул (8), (9), которые можно записать в виде

$$x_1 = b_1/l_1, \quad x_i = (b_i - x_{i-1})/l_i,\ i = 2, \ldots, n, \quad x_i := (x_i - u_i x_{i+1})/v_i,\ i = n-1, \ldots, 1.$$

(здесь считается $v_n = 1$.)

§ 6. ЗАДАЧА ОБРАЩЕНИЯ МАТРИЦЫ

Рассмотрим задачу нахождения матрицы, обратной к данной.

Случай произвольного алгоритма. Пусть ß – некоторый алгоритм решения линейных систем вида $Ax = b$, так, что $x = ß(A, b)$. Тогда j-й столбец x_j матрицы A^{-1} равен $x_j = ß(A, e_j)$, где $e_j = (\underbrace{0, \ldots, 0}_{j-1}, 1, 0, \ldots, 0)^t$ есть j-й орт стандартного базиса. Если алгоритм ß требует для своего проведения $d(n)$ арифметических операций, то этот способ нахождения обратной матрицы потребует $n\,d(n)$ арифметических операций. Например, если ß – это метод Гаусса, то потребуется $2/3\,n^4 + O(n^3)$ арифметических операций.

Случай специального алгоритма. Многие алгоритмы решения линейных систем (в частности, все алгоритмы, рассматриваемые нами) обладают следующим свойством: алгоритм (по крайней мере его самая трудоемкая с вычислительной точки зрения часть) состоит в проведении над системой преобразований, которые выполняются над матрицей системы и правой частью независимо. Эта особенность позволяет вместо правой части – вектора b – рассматривать набор правых частей, т.е. матрицу B. Преобразования алгоритма выполняются над матрицей системы и набором правых частей. Таким образом, такой алгоритм решения системы $Ax = b$ может быть преобразован в алгоритм решения матричного уравнения $AX = B$, где X, B – $n \times n$ матрицы. Обычно алгоритм решения системы $Ax = b$ требует $O(n^3)$ арифметических операций для проведения преобразований над матрицей и $O(n^2)$ арифметических операций для проведения преобразований над правой частью. Поэтому алгоритм решения матричной системы $AX = B$ требует $O(n^3) + n \cdot O(n^2) = O(n^3)$ арифметических операций.

Использование LU-разложения для обращения матрицы. Пусть для матрицы A возможно осуществить LU-разложение. Действительно, если $A = LU$, то $A^{-1} = U^{-1}L^{-1}$. Матрицы, обратные к L и U, строятся, например, описанным выше способом. Поскольку системы с матрицами L и U решаются методом последовательного исключения неизвестных за соответственно $n(n-1)+n = n^2+O(n)$ и $n(n-1) = n^2+O(n)$ действий (см. формулы (4.10)), то матрицы U^{-1} и L^{-1} могут быть вычислены с затратой $n^3+O(n^2)$ арифметических операций. Следовательно, для вычисления обратной матрицы требуется: $2/3\,n^3+O(n^2)$ арифметических операций для построения LU-разложения, $n^3+O(n^2)$ арифметических операций для вычисления U^{-1} и L^{-1}, $n^3+O(n^2)$ арифметических операций для вычисления $A^{-1} = U^{-1}L^{-1}$; всего: $8/3\,n^3+O(n^2)$ арифметических операций.

§ 7. МЕТОД ГАУССА С ВЫБОРОМ ГЛАВНОГО ЭЛЕМЕНТА

Теорема 4.1 показывает, что метод Гаусса в изложенном выше виде применим не ко всем невырожденным матрицам. Например, если в системе (4.1) $a_{11} = 0$, то нельзя осуществить первый же шаг алгоритма. Модернизируем алгоритм следующим образом.

Уравнения в системе (4.1) равноправны, мы можем их занумеровать в произвольном порядке. Присвоим номер 1 тому уравнению, в котором коэффициент при x_1 отличен от 0. Если такого уравнения не нашлось, то матрица A имеет нулевой первый столбец, т.е. вырождена. После этой перенумерации уравнений мы сделаем первый шаг метода Гаусса, т.е. перейдем от системы (4.1) к системе (4.3). Далее в подматрице $A^{(1)} = (a_{ij}^{(1)})_{i,j=2,\dots,n} \in \mathbf{M}_{n-1}$ присвоим номер 2 тому уравнению, в котором коэффициент при x_2 отличен от 0, и сделаем следующий шаг метода Гаусса. Затем этот процесс применяется к подматрице $A^{(2)} \in \mathbf{M}_{n-2}$ и так далее.

Если уравнений, в которых коэффициент при x_1 отличен от 0, несколько, то с вычислительной точки зрения не безразлично, какое из этих уравнений получит номер 1. Пусть погрешность в элементе a_{ij} матрицы A равна ε_{ij}, т.е. вместо точной матрицы A рассматривается матрица $\hat{A}$, элементы которой содержат вычислительные погрешности: $\hat{a}_{ij} = a_{ij} + \varepsilon_{ij}$. Для простоты будем считать, что элементы первого столбца известны точно: $\varepsilon_{i1} = 0$, $i = 1,\dots,n$. Из формул для элементов матрицы $A^{(1)}$ (см. (4.8), (4.6)):

$$\hat{a}_{ij}^{(1)} = \hat{a}_{ij} - \hat{c}_{1j}\hat{a}_{i1} = \hat{a}_{ij} - \frac{\hat{a}_{1j}}{\hat{a}_{11}}\,\hat{a}_{i1} = \hat{a}_{ij} - \hat{a}_{1j}\,\frac{\hat{a}_{i1}}{\hat{a}_{11}} = a_{ij} + \varepsilon_{ij} - (a_{1j}+\varepsilon_{1j})\,\frac{a_{i1}}{a_{11}}$$
$$= a_{ij} - a_{1j}\,\frac{a_{i1}}{a_{11}} + \varepsilon_{ij} - \varepsilon_{1j}\,\frac{a_{i1}}{a_{11}} = a_{ij}^{(1)} + \varepsilon_{ij}^{(1)}, \quad i,j = 2,\dots,n,$$

где

$$\varepsilon_{ij}^{(1)} = \varepsilon_{ij} - \varepsilon_{1j}\,\frac{a_{i1}}{a_{11}}, \quad i,j = 2,\dots,n, \tag{1}$$

Эти равенства показывают, как преобразуется погрешность после шага алгоритма. Из соотношений (1) вытекает, что если отношение $\frac{a_{i1}}{a_{11}}$ очень велико, то вычислительная погрешность, вносимая на шаге алгоритма, может быть недопустимо большой.

Из (1) следует, что погрешность будет наименьшей, если модуль отношения $\frac{a_{i1}}{a_{11}}$ наименьший из возможных. Это будет в том случае, если a_{11} – наибольший по модулю элемент в первом столбце. Поэтому описанный выше алгоритм преобразуем к следующему виду.

Присвоим номер 1 тому уравнению, в котором коэффициент при x_1 наибольший по модулю. Этот коэффициент отличен от нуля, так как противное означало бы, что матрица A имеет нулевой первый столбец, т.е. вырождена. После этой перенумерации уравнений мы сделаем первый шаг метода Гаусса, т.е. перейдем от системы (4.1) к системе (4.3). Далее в подматрице $A^{(1)} = (a_{ij}^{(1)})_{i,j=2,\dots,n} \in \mathbf{M}_{n-1}$ присвоим номер 2 тому уравнению, в котором коэффициент при x_2 наибольший по модулю, и сделаем следующий шаг метода Гаусса. Затем этот процесс применяется к подматрице $A^{(2)} \in \mathbf{M}_{n-2}$ и так далее. Этот алгоритм называется *методом Гаусса с выбором главного элемента по столбцу.*

Можно преобразовать метод Гаусса и по-другому. Неизвестные в системе (4.1) равноправны, мы можем их занумеровать в произвольном порядке. Присвоим номер 1 той неизвестной, при которой коэффициент в первой строке отличен от 0. Если такой неизвестной не нашлось, то матрица A имеет нулевую первую строку, т.е. вырождена. После этой перенумерации неизвестных мы сделаем первый шаг метода Гаусса, т.е. перейдем от системы (4.1) к системе (4.3). Далее в подматрице $A^{(1)} = (a_{ij}^{(1)})_{i,j=2,\dots,n} \in \mathbf{M}_{n-1}$ присвоим номер 2 той неизвестной, при которой коэффициент в первой строке матрицы $A^{(1)}$ (т.е. во второй строке матрицы A) отличен от 0, и сделаем следующий шаг метода Гаусса. Затем этот процесс применяется к подматрице $A^{(2)} \in \mathbf{M}_{n-2}$ и так далее.

Если неизвестных, при которых коэффициент отличен от 0, несколько, то с вычислительной точки зрения не безразлично, какая из них получит номер 1. Рассуждениями, аналогичными вышеприведенным, можно установить, что погрешность, вносимая на шаге алгоритма, будет минимальной, если если a_{11} – наибольший по модулю элемент в первой строке. Поэтому описанный выше алгоритм преобразуем к следующему виду.

Присвоим номер 1 той неизвестной, при которой коэффициент в первой строке наибольший по модулю. Этот коэффициент отличен от нуля, так как противное означало бы, что матрица A имеет нулевую первую строку, т.е. вырождена. После этой перенумерации неизвестных мы сделаем первый шаг метода Гаусса, т.е. перейдем от системы (4.1) к системе (4.3). Далее в подматрице $A^{(1)} = (a_{ij}^{(1)})_{i,j=2,\dots,n} \in \mathbf{M}_{n-1}$ присвоим номер 2 той неизвестной, при которой коэффициент в первой строке матрицы $A^{(1)}$ (т.е. во второй строке матрицы A) наибольший по модулю, и сделаем следующий шаг метода Гаусса. Затем этот процесс применяется к подматрице $A^{(2)} \in \mathbf{M}_{n-2}$ и так далее. Этот алгоритм

называется *методом Гаусса с выбором главного элемента по строке.*

Для уменьшения вычислительной погрешности используют следующую комбинацию приведенных выше методов. В качестве a_{11} выбирается элемент, имеющий наибольший модуль среди всех элементов матрицы. Если этот элемент есть a_{ij}, то меняются номера у 1-й и i-й строк и у 1-го и j-го столбцов. После этой перенумерации уравнений и неизвестных делается первый шаг метода Гаусса, т.е. осуществляется переход от системы (4.1) к системе (4.3). Далее в подматрице $A^{(1)} = (a_{ij}^{(1)})_{i,j=2,\ldots,n} \in \mathbf{M}_{n-1}$ выбирается элемент a_{ij} с наибольшим модулем среди всех элементов матрицы $A^{(1)}$ и меняются номера у 1-й и i-й строк и у 1-го и j-го столбцов матрицы $A^{(1)}$ (т.е. у 2-й и i-й строк и у 2-го и j-го столбцов матрицы A). Затем этот процесс применяется к подматрице $A^{(2)} \in \mathbf{M}_{n-2}$ и так далее. Этот алгоритм называется *методом Гаусса с выбором главного элемента по всей матрице.*

Вычислим дополнительные (по сравнению с обычным методом Гаусса) затраты вычислительной работы на решение системы по этим алгоритмам. На k-ом шаге ($k = 1, \ldots, n$) метода Гаусса с выбором главного элемента по столбцу или строке требуется $n-k$ операций сравнения элементов матрицы A для нахождения максимального по модулю элемента. В методе Гаусса с выбором главного элемента по всей матрице это число равно $(n-k)^2$. Следовательно, в первых двух методах дополнительно требуется $\sum_{k=1}^{n}(n-k) = n(n-1)/2 = O(n^2)$ $(n \to \infty)$ операций сравнения, а в последнем методе – $\sum_{k=1}^{n}(n-k)^2 = (n-1)n(2n-1)/6 = n^3/3 + O(n^2)$ $(n \to \infty)$ операций сравнения.

На большинстве ЭВМ операция сравнения двух чисел с плавающей точкой выполняется за время, по порядку равное времени вычитания этих чисел. (Это связано с тем, что вместо сравнения двух чисел выполняется операция вычитания одного числа из другого и сравнения результата с нулем. Поскольку сам результат нигде не запоминается и от него используется лишь его знак, то операция сравнения обычно осуществляется быстрее операции вычитания, однако следующая за операцией сравнения команда условного перехода с лихвой компенсирует эту разницу.) Поэтому в методе Гаусса с выбором главного элемента по столбцу или строке количество операций асимптотически то же, что в обычном методе Гаусса: $2/3\, n^3 + O(n^2)$. В методе Гаусса с выбором главного элемента по всей матрице количество операций асимптотически в полтора раза больше, чем в обычном методе Гаусса: $n^3 + O(n^2)$. По этой причине этот метод обычно применяется тогда, когда с помощью других методов не удалось получить приемлемого по точности результата из-за сильного роста вычислительной погрешности (такая ситуация возникает, если матрица A имеет большое число обусловленности).

Теорема 1. *Метод Гаусса с выбором главного элемента по столбцу осуществим тогда и только тогда, когда* $\det A \neq 0$.

Доказательство. Шаг метода Гаусса переводит невырожденную матрицу

в невырожденную. Действительно, после перехода от матрицы (4.4) к матрице (4.7) по формулам (4.6), (4.8) определитель матрицы (4.4) равен определителю матрицы (4.7), умноженному на $a_{kk}^{(k-1)}$ (множитель возникает при вычислении по формулам (4.6), при преобразовании матрицы по формулам (4.8) определитель не изменяется, так как они задают элементарные преобразования матрицы). Согласно правилам вычисления определителей, определитель матрицы (4.4), получающейся после $k-1$ шагов метода Гаусса, равен определителю матрицы $A^{(k-1)} = (a_{ij}^{(k-1)})_{i,j=k,\ldots,n}$. Следовательно, невырожденность матрицы A эквивалентна невырожденности матриц $A^{(k)}$ для всех $k = 1, \ldots, n$.

Очередной, k-й шаг метода Гаусса с выбором главного элемента по столбцу возможен тогда и только тогда, когда первый столбец матрицы $A^{(k-1)}$ ненулевой, т.е. эта матрица невырождена (см. подробное обоснование этого при построении метода).

Таким образом, осуществимость всех n шагов метода Гаусса эквивалентна невырожденности матриц $A^{(k)}$ для всех $k = 1, \ldots, n$, что эквивалентно невырожденности матрицы A.

Теорема 2. *Метод Гаусса с выбором главного элемента по строке осуществим тогда и только тогда, когда* $\det A \neq 0$.

Доказательство повторяет доказательство предыдущей теоремы. Изменения только в том, что очередной, k-й шаг метода Гаусса с выбором главного элемента по строке возможен тогда и только тогда, когда первая строка матрицы $A^{(k-1)}$ ненулевая, т.е. эта матрица невырождена (см. подробное обоснование этого при построении метода).

Теорема 3. *Метод Гаусса с выбором главного элемента по всей матрице осуществим тогда и только тогда, когда* $\det A \neq 0$.

Доказательство повторяет доказательство теоремы 1. Изменения только в том, что очередной, k-й шаг метода Гаусса с выбором главного элемента по всей матрице возможен тогда и только тогда, матрица $A^{(k-1)}$ ненулевая (см. подробное обоснование этого при построении метода).

Замечание 1. Программная реализация методов Гаусса с выбором главного элемента. При реализации этих методов можно переставлть не строки или столбцы матрицы, а их номера. Сделать это можно, например, следующим способом.

Рассмотрим метод Гаусса с выбором главного элемента по всей матрице. Пусть массив `indi` длиной n содержит номер строки матрицы A, массив `indj` длиной n содержит номер столбца матрицы A. Вначале `indi(i)=i`, `indj(j)=j`, `i,j`$= 1, \ldots, n$. Обращение к элементам матрицы A происходит следующим образом: элемент a_{ij} есть `a(indi(i),indj(j))`. Для того, чтобы переставить местами

i_1-ю и i_2-ю строки матрицы A, достаточно переставить местами i_1-й и i_2-й элементы массива `indi`; для того, чтобы переставить местами j_1-й и j_2-й столбцы матрицы A, достаточно переставить местами j_1-й и j_2-й элементы массива `indj`.

Существенным недостатком такого способа реализации перестановок строк и столбцов является замедление доступа к элементам массива.

Замечание 2. Для матриц A произвольного вида методы Гаусса с выбором главного элемента практически вытеснили обычный метод Гаусса из вычислительной практики. Совершенно иная ситуация для случая ленточных матриц A. Дело здесь в том, что перестановка строк или столбцов в таких матрицах приводит к увеличению ширины ленты, что часто недопустимо (поскольку вместо матрицы хранится только ее лента).

§ 8. МЕТОД ЖОРДАНА (ГАУССА-ЖОРДАНА)

Пусть требуется решить линейную систему $A\,x = b,\ A \in \mathbf{M}_n$ вида (4.1). Первый шаг метода Жордана сопадает с первым шагом метода Гаусса: система (4.1) преобразуется к виду

$$\begin{array}{ccccccccc} x_1 & + & a_{12}^{(1)} x_2 & + & \ldots & + & a_{1n}^{(1)} x_n & = & b_1^{(1)} \\ & & a_{22}^{(1)} x_2 & + & \ldots & + & a_{2n}^{(1)} x_n & = & b_2^{(1)} \\ & & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots \\ & & a_{n2}^{(1)} x_2 & + & \ldots & + & a_{nn}^{(1)} x_n & = & b_n^{(1)} \end{array} \qquad (1)$$

по тем же формулам

$$\begin{array}{llll} a_{1j}^{(1)} = a_{1j}/a_{11}, & b_1^{(1)} = b_1/a_{11}, & & j = 2, \ldots, n, \\ a_{ij}^{(1)} = a_{ij} - a_{1j}^{(1)} a_{i1}, & b_i^{(1)} = b_i - b_1^{(1)} a_{i1}, & & i, j = 2, \ldots, n. \end{array}$$

После $k-1$, $k = 1, \ldots, n$ шагов метода Жордана система (4.1) преобразована к виду

$$\begin{array}{ccccccccc} x_1 & & & + a_{1,k}^{(k-1)} x_k & + \ldots + & a_{1n}^{(k-1)} x_n & = & b_1^{(k-1)} \\ & x_2 & & + a_{2,k}^{(k-1)} x_k & + \ldots + & a_{2n}^{(k-1)} x_n & = & b_2^{(k-1)} \\ & & \ddots & \vdots \quad \vdots & \vdots \ddots \vdots & \vdots \quad \vdots & \vdots & \vdots \\ & & x_{k-1} & + a_{k-1,k}^{(k-1)} x_k & + \ldots + & a_{k-1,n}^{(k-1)} x_n & = & b_{k-1}^{(k-1)} \\ & & & a_{k,k}^{(k-1)} x_k & + \ldots + & a_{k,n}^{(k-1)} x_n & = & b_k^{(k-1)} \\ & & & \vdots \quad \vdots & \vdots \ddots \vdots & \vdots \quad \vdots & \vdots & \vdots \\ & & & a_{n,k}^{(k-1)} x_k & + \ldots + & a_{n,n}^{(k-1)} x_n & = & b_n^{(k-1)} \end{array} \qquad (2)$$

Предположим, что $a_{kk}^{(k-1)} \neq 0$. Поделив k-е уравнение системы (2) на $a_{kk}^{(k-1)}$, перепишем его в виде

$$x_k + a_{k,k+1}^{(k)} x_{k+1} + \ldots + a_{kn}^{(k)} x_n = b_k^{(k)}, \qquad (3)$$

где

$$a_{kj}^{(k)} = \frac{a_{kj}^{(k-1)}}{a_{kk}^{(k-1)}}, \quad j = k+1, \ldots, n, \qquad b_k^{(k)} = \frac{b_k^{(k-1)}}{a_{kk}^{(k-1)}}. \tag{4}$$

Умножим уравнение (3) на $a_{ik}^{(k-1)}$ и вычтем его из i-го уравнения системы (2), $i = 1, \ldots, n$. В результате система (2) примет вид

$$\begin{array}{rcccccccccc}
x_1 & & & & + & a_{1,k+1}^{(k)} x_{k+1} & + \ldots + & a_{1n}^{(k)} x_n & = b_1^{(k)} \\
& x_2 & & & + & a_{2,k+1}^{(k)} x_{k+1} & + \ldots + & a_{2n}^{(k)} x_n & = b_2^{(k)} \\
& & \ddots & & \vdots & \vdots & \vdots \ddots \vdots & \vdots & \vdots & \vdots \\
& & & x_{k-1} & + a_{k-1,k+1}^{(k)} x_{k+1} & + \ldots + & a_{k-1,n}^{(k)} x_n & = b_{k-1}^{(k)} & & \\
& & & & x_k + & a_{k,k+1}^{(k)} x_{k+1} & + \ldots + & a_{k,n}^{(k)} x_n & = b_k^{(k)} & \\
& & & & & a_{k+1,k+1}^{(k)} x_{k+1} & + \ldots + & a_{k+1,n}^{(k)} x_n & = b_{k+1}^{(k)} & \\
& & & & & \vdots & \vdots \ddots \vdots & \vdots & \vdots & \vdots \\
& & & & & a_{n,k+1}^{(k)} x_{k+1} & + \ldots + & a_{n,n}^{(k)} x_n & = b_n^{(k)}
\end{array} \tag{5}$$

где

$$\begin{aligned}
a_{ij}^{(k)} &= a_{ij}^{(k-1)} - a_{ik}^{(k-1)} a_{kj}^{(k)}, \qquad i = 1, \ldots, n, \ i \neq k, \ \ j = k+1, \ldots, n. \\
b_i^{(k)} &= b_i^{(k-1)} - a_{ik}^{(k-1)} b_k^{(k)}, \qquad i = 1, \ldots, n, \ i \neq k.
\end{aligned} \tag{6}$$

Выражения (4), (6) являются формулами перехода от системы (2) к системе (5). Если обозначить $a_{ij}^{(0)} = a_{ij}$, $b_i^{(0)} = b_i$, $i, j = 1, \ldots, n$, то переход от системы (4.1) к системе (1) будет осуществляться по тем же формулам при $k = 1$.

После проведения вычислений по формулам (4), (6) при $k = 1, \ldots, n$ матрица системы (4.1) станет единичной матрицей. Следовательно, правая часть системы содержит искомое решение: $x_i = b_i^{(n)}$, $i = 1, \ldots, n$.

Метод Жордана удобно применять для нахождения обратной матрицы. При этом вместо правой части b используется набор правых частей, состоящий из n столбцов единичной матрицы, над которыми одновременно производятся преобразования, задаваемые соотношениями (4), (6). После проведения n шагов метода Жордана этот набор будет состоять из столбцов обратной матрицы A^{-1}.

Поскольку на каждом шаге метода Жордана подматрица $A^{(k-1)} = (a_{ij}^{(k-1)})_{i,j=k,\ldots,n}$ – та же, что на соответствующем шаге метода Гаусса, то метод Жордана осуществим тогда и только тогда, когда осуществим метод Гаусса, т.е. когда все главные угловые миноры матрицы A отличны от нуля.

Оценка количества арифметических операций в методе Жордана

1. На вычисление $a_{kj}^{(k)}$ при $j = k+1, \ldots, n$, $k = 1, \ldots, n$ по формулам (4) требуется $\sum_{k=1}^{n}(n-k) = n(n-1)/2 = O(n^2)$ $(n \to \infty)$ операций деления.

2. На вычисление $a_{ij}^{(k)}$ при $i = 1, \ldots, n,\ i \neq k,\ j = k+1, \ldots, n,\ k = 1, \ldots, n$ по формулам (6) требуется $\sum_{k=1}^{n}(n-k)(n-1) = (n-1)^2 n/2 = n^3/2 + O(n^2)$ $(n \to \infty)$ операций умножения и столько же операций вычитания.

3. На вычисление $b_k^{(k)}$ при $k = 1, \ldots, n$ по формулам (4) требуется n операций деления.

4. На вычисление $b_i^{(k)}$ при $i = 1, \ldots, n,\ i \neq k,\ k = 1, \ldots, n$ по формулам (6) требуется $\sum_{k=1}^{n}(n-1) = n(n-1) = O(n^2)$ $(n \to \infty)$ операций умножения и столько же операций вычитания.

Таким образом, метод Жордана требует $O(n^2) + n^3/2 + n + O(n^2) = n^3/2 + O(n^2)$ $(n \to \infty)$ мультипликативных операций и столько же аддитивных операций. Всего: $n^3 + O(n^2)$ $(n \to \infty)$ арифметических операций.

По аналогии с методом Гаусса можно строить *метод Жордана с выбором главного элемента*. Именно, в подматрице $A^{(k-1)} = (a_{ij}^{(k-1)})_{i,j=k,\ldots,n}$ (совпадающей с подматрицей из метода Гаусса) той же процедурой, что и в методе Гаусса, выбирается главный элемент.

§ 9. ПОЛОЖИТЕЛЬНО ОПРЕДЕЛЕННЫЕ МАТРИЦЫ

Определение. Матрица $A \in \mathbf{M}_n$ называется *положительно определенной* (обозначается $A > 0$), если для всех $x \in \mathbf{C}^n$ выражение (Ax, x) вещественно и $(Ax, x) > 0$ для всех $x \in \mathbf{C}^n,\ x \neq 0$ (здесь $(\cdot, \cdot)$ означает обычное скалярное произведение в $\mathbf{C}^n$, $(x, y) = y^* x$, где $y^* = (\overline{y_1}, \ldots, \overline{y_n})$, черта над символом обозначает, как обычно, знак комплексного сопряжения). Если рассматриваемая матрица A вещественна, то часто положительно определенной называется матрица A, для которой $(Ax, x) > 0$ для всех $x \in \mathbf{R}^n,\ x \neq 0$.

Замечание 1. Если матрица $A \in \mathbf{M}_n$ – самосопряженная (т.е. $A^* = A$), то выражение (Ax, x) вещественно для всех $x \in \mathbf{C}^n$.

Действительно, $(Ax, x) = (x, A^* x) = (x, Ax) = \overline{(Ax, x)}$ и потому (Ax, x) вещественно.

Лемма 1. *Если матрица A положительно определена, то она невырождена.*

Доказательство. Предположим противное, $\det A = 0$. Тогда линейная система $Ax = 0$ имеет решение $x \in \mathbf{C}^n,\ x \neq 0$. Для этого x выражение $(Ax, x) = (0, x) = 0$, что противоречит положительной определенности матрицы A.

Лемма 2. *Если матрица A положительно определена, то для нее существует LU-разложение.*

Доказательство. В соответствии с теоремой 4.1 нам надо проверить, что главные угловые миноры положительно определенной матрицы A отличны от нуля.

Определим отображение $x_{(k)}$ пространства $\mathbf{C}^n \to \mathbf{C}^k$, $k \le n$ действующее по правилу: для всякого $x = (x_1, \ldots, x_n)^t \in \mathbf{C}^n$ $x_{(k)} = (x_1, \ldots, x_k)^t \in \mathbf{C}^k$. Это отображение есть отображение "на", т.е. для каждого элемента $x = (x_1, \ldots, x_k)^t \in \mathbf{C}^k$ найдется элемент $\hat{x} \in \mathbf{C}^n$, являющийся прообразом x при этом отображении (например, $\hat{x} = (x_1, \ldots, x_k, 0, \ldots, 0)^t \in \mathbf{C}^n$).

Поскольку матрица A положительно определена, то для всякого $k = 1, \ldots, n$ и всякого $x \in \mathbf{C}^n$, такого, что $x_{(k)} \neq 0$ выражение $(Ax_{(k)}, x_{(k)})$ вещественно и положительно. По правилу перемножения матриц $(Ax_{(k)}, x_{(k)}) = (A_k x_{(k)}, x_{(k)})_k$, где

$$A_k = \begin{pmatrix} a_{11} & a_{12} & \ldots & a_{1k} \\ a_{21} & a_{22} & \ldots & a_{2k} \\ \vdots & \vdots & \ddots & \vdots \\ a_{k1} & a_{k2} & \ldots & a_{kk} \end{pmatrix}$$

-главный угловая подматрица A , $(\cdot, \cdot)_k$ – обычное скалярное произведение в пространстве $\mathbf{C}^k$, $(x, y)_k = y^* x$, $x, y \in \mathbf{C}^k$. Следовательно, выражение $(A_k x, x)_k$ вещественно и положительно для всех $x \in \mathbf{C}^k$, т.е. матрицы $A_k \in \mathbf{M}_k$, $k = 1, \ldots, n$ положительно определены. Пользуясь леммой 1, получаем, что матрицы A_k, $k = 1, \ldots, n$ невырождены. Из теоремы 4.1 теперь вытекает требуемый результат.

Лемма 3. *Самосопряженная матрица $A \in \mathbf{M}_n$ положительно определена тогда и только тогда, когда все ее собственные значения вещественны и положительны.*

Доказательство. Пусть $A \in \mathbf{M}_n$ самосопряженная положительно определеная матрица, λ – ее собственное значение, $x \neq 0$ – соответствующий собственный вектор: $Ax = \lambda x$. Умножим это равенство скалярно на x, получим $(Ax, x) = \lambda(x, x)$ и $\lambda = \dfrac{(Ax, x)}{\|x\|^2}$. Поскольку (Ax, x) вещественно и положительно, то $\lambda > 0$.

Пусть $A \in \mathbf{M}_n$ самосопряженная матрица и $\lambda_i > 0$, $i = 1, 2, \ldots, n$ – ее собственные значения. В курсе линейной алгебры было доказано, что всякая самосопряженная (симметричная в вещественном случае) матрица диагонализируема в евклидовом базисе, т.е. существует ортонормированный базис $x_1, x_2, \ldots, x_n$, $(x_i, x_j) = \delta_{ij}$, состоящий из собственных векторов матрицы A: $Ax_i = \lambda_i x_i$. Пусть $x \neq 0$ – произвольный вектор, $x = \sum_{i=1}^n c_i x_i$ – его разложение по базису $\{x_i\}$, причем $\|x\|^2 = \sum_{i=1}^n |c_i|^2 \neq 0$. Рассмотрим выражение $(Ax, x) = (A\sum_{i=1}^n c_i x_i, \sum_{i=1}^n c_i x_i) = (\sum_{i=1}^n c_i \lambda_i x_i, \sum_{i=1}^n c_i x_i) = \sum_{i=1}^n \lambda_i |c_i|^2$. Следовательно, (Ax, x) вещественно. Поскольку $\lambda_i > 0$ и не все c_i равны 0, то (Ax, x) положительно. Итак, для всякого вектора $x \neq 0$ выражение (Ax, x) вещественно и положительно, что и означает положительную определенность матрицы A.

§ 10. МЕТОД ХОЛЕЦКОГО (КВАДРАТНОГО КОРНЯ)

Пусть требуется решить линейную систему $A\,x = b$ с самосопряженной (симметричной в вещественном случае) матрицей $A \in \mathbf{M}_n$, $A^* = A$.

§ 10.1. Разложение Холецкого

Обозначим через $\mathrm{RT}(n)$ подгруппу невырожденных верхних треугольных матриц в $\mathbf{M}_n$, а через $\mathrm{UT}(n)$ – подгруппу в $\mathrm{RT}(n)$ матриц с единицами на главной диагонали.

Теорема 1. *Пусть матрица A – самосопряженная ($A^* = A$) и все ее главные угловые миноры отличны от нуля. Тогда существуют матрица $R = (r_{ij}) \in \mathrm{RT}(n)$ с вещественными положительными элементами на главной диагонали ($r_{ii} > 0$ для всех $i = 1, \ldots, n$) и диагональная матрица D с вещественными равными по модулю единице диагональными элементами ($d_{ii} \in \{-1, 1\}$ для всех $i = 1, \ldots, n$) такие, что $A = R^* D R$.*

Доказательство. По теореме 4.1 для матрицы A осуществимо LU-разложение, т.е. существуют $L \in \mathrm{LT}(n)$ и $U \in \mathrm{UT}(n)$ такие, что $A = LU$. Поскольку матрица $L = (l_{ij})$ невырождена, то $l_{ii} \neq 0$, $i = 1, \ldots, n$ и матрица

$$\hat{D} = \mathrm{diag}\,(l_{11}, \ldots, l_{nn}) \tag{1}$$

обратима, $\hat{D}^{-1} = \mathrm{diag}\,(l_{11}^{-1}, \ldots, l_{nn}^{-1})$. Положим $\hat{L} = L\hat{D}^{-1} \in \mathrm{LT}(n)$. Тогда по правилам перемножения матриц $\hat{l}_{ii} = 1$, $i = 1, \ldots, n$.

Подставим это представление матрицы $L = \hat{L}\hat{D}$ в LU-разложение матрицы A: $A = \hat{L}\hat{D}U$. Так как $A = A^*$, то $A = \hat{L}\hat{D}U = A^* = U^*\hat{D}^*\hat{L}^*$. Поэтому $U = \hat{D}^{-1}\hat{L}^{-1}U^*\hat{D}^*\hat{L}^*$,

$$U(\hat{L}^*)^{-1} = \hat{D}^{-1}\hat{L}^{-1}U^*\hat{D}^*. \tag{2}$$

Заметим, что $\hat{L}^* \in \mathrm{RT}(n)$, причем главная диагональ этой матрицы состоит из единиц. Следовательно $\hat{L}^* \in \mathrm{UT}(n)$. Поэтому в левой части равенства (2) стоит матрица $U(\hat{L}^*)^{-1} \in \mathrm{UT}(n)$. В право же части равенства (2) стоит произведение нижних треугольных матриц, которое является опять нижней треугольной матрицей, т.е. принадлежит $\mathrm{LT}(n)$. Поэтому из (2) вытекает

$$U(\hat{L}^*)^{-1} \in \mathrm{UT}(n) \cap \mathrm{LT}(n).$$

Единственной матрицей, которая принадлежит одновременно подгруппам $\mathrm{UT}(n)$ и $\mathrm{LT}(n)$ является I – единичная матрица. Следовательно,

$$U(\hat{L}^*)^{-1} = \hat{D}^{-1}\hat{L}^{-1}U^*\hat{D}^* = I. \tag{3}$$

Таким образом, $U = \hat{L}^*$ и $A = U^*\hat{D}U$. Далее, из (3)

$$I = \hat{D}^{-1}\hat{L}^{-1}U^*\hat{D}^* = \hat{D}^{-1}\hat{L}^{-1}(\hat{L}^*)^*\hat{D}^* = \hat{D}^{-1}\hat{L}^{-1}\hat{L}\hat{D}^* = \hat{D}^{-1}\hat{D}^*$$

т.е. $\hat{D} = \hat{D}^*$. В силу (1) получаем $l_{kk} = \overline{l_{kk}}$, т.е. l_{kk} – вещественные для всех $k = 1, \ldots, n$. Представив матрицу $\hat{D}$ в виде $\hat{D} = |\hat{D}|^{1/2} D\, |\hat{D}|^{1/2}$, где

$$D = \operatorname{diag}(\operatorname{sign} l_{11}, \ldots, \operatorname{sign} l_{nn}), \quad |\hat{D}|^{1/2} = \operatorname{diag}(\sqrt{|l_{11}|}, \ldots, \sqrt{|l_{nn}|}),$$

получаем $A = U^* |\hat{D}|^{1/2} D\, |\hat{D}|^{1/2} U$. Обозначим $R = |\hat{D}|^{1/2} U \in RT(n)$. Тогда $A = R^* D R$, причем диагональные элементы r_{ii} матрицы R равны $\sqrt{|l_{ii}|} > 0$. Следовательно, полученное разложение является требуемым.

Замечание 1. Если матрица A – вещественная, то все участвующие в теореме 1 матрицы вещественные.

Замечание 2. Если в условиях теоремы 1 матрица A положительно определена, то матрица D в теореме 1 – единичная, т.е. разложение, даваемое этой теоремой, имеет вид $A = R^* R$.

Действительно, если матрица A положительно определена и для нее справедливо разложение $A = R^* D\, R$, то для всякого $x \in \mathbf{C}^n$, $x \neq 0$ выражение $(Ax, x) = (R^* D\, Rx, x) = (D\, Rx, Rx)$ вещественно и положительно. Для всякого $y \in \mathbf{C}^n$ положим $x = R^{-1} y$. Тогда выражение $(Dy, y) = (Ax, x)$ вещественно и положительно для всех $y \in \mathbf{C}^n$, $y \neq 0$, т.е. матрица D положительно определена.

Поскольку $D = \operatorname{diag}(d_{11}, \ldots, d_{nn})$, то $(Dy, y) = \sum_{j=1}^n d_{jj} y_j \overline{y_j} = \sum_{j=1}^n d_{jj} |y_j|^2$. По доказанному, эта сумма вещественна и положительна. Выбирая здесь $y = e_k$, $k = 1, \ldots, n$, где e_k – орты стандартного базиса, находим, что все d_{kk}, $k = 1, \ldots, n$ должны быть вещественными и положительными. Поскольку $|d_{kk}| = 1$, то это означает, что $d_{kk} = 1$, $k = 1, \ldots, n$.

Замечание 3. Если матрица A положительно определена, то для нее выполнены условия теоремы 1. Это вытекает из леммы 9.2.

§ 10.2. Алгоритм построения разложения Холецкого

Построим алгоритм нахождения разложения из теоремы 1.

Пусть для самосопряженной матрицы A ($A^* = A$) требуется найти верхнюю треугольную матрицу $R = (r_{ij})$ с вещественными положительными элементами на главной диагонали ($r_{ii} > 0$ для всех $i = 1, \ldots, n$) и диагональную матрицу D с равными по модулю единице диагональными элементами ($|d_{ii}| = 1$ для всех $i = 1, \ldots, n$) такую, что $A = R^* D R$.

Элемент (k, j) матрицы DR равен $(DR)_{kj} = \sum\limits_{i=1}^{n} d_{ki} r_{ij} = d_{kk} r_{kj}$, так как матрица D – диагональная; элемент (i, k) матрицы R^* равен $(R^*)_{ik} = \overline{r_{ki}}$; элемент (i, j) матрицы $R^* DR$ равен $(R^* DR)_{ij} = \sum\limits_{k=1}^{n} (R^*)_{ik} (DR)_{kj} = \sum\limits_{k=1}^{n} \overline{r_{ki}} d_{kk} r_{kj}$. Следовательно, равенство $A = R^* DR$ дает нам уравнения

$$\sum_{k=1}^{n} \overline{r_{ki}} d_{kk} r_{kj} = a_{ij}, \qquad i, j = 1, \ldots, n. \tag{4}$$

Поскольку матрицы A и R^*DR – самосопряженные, то уравнение с номером (j,i) получается из уравнения с номером (i,j) путем комплексного сопряжения и не дает ничего нового. Поэтому система (4) эквивалентна системе

$$\sum_{k=1}^{n} \overline{r_{ki}} d_{kk} r_{kj} = a_{ij}, \qquad i \le j, \;\; i,j = 1,\dots,n. \tag{5}$$

Таким образом, (5) представляет собой систему из $n(n+1)/2$ уравнений с $n(n+1)/2$ неизвестными r_{ij}, $i \le j$ (напомним, $R \in \mathrm{RT}(n)$ и $r_{ij} = 0$ при $i > j$) и n неизвестными d_{kk}, $k = 1,\dots,n$ (при этом $r_{kk} > 0$ и $d_{kk} \in \{-1, 1\}$).

Получим формулы для решения системы (5), которые и составляют алгоритм метода Холецкого.

Перепишем (5) в виде

$$\sum_{k=1}^{i} \overline{r_{ki}} d_{kk} r_{kj} + \sum_{k=i+1}^{n} \overline{r_{ki}} d_{kk} r_{kj} = a_{ij}, \qquad i \le j, \;\; i,j = 1,\dots,n. \tag{6}$$

Поскольку матрица R – верхняя треугольная, то $r_{ki} = 0$ при $k > i$ и вторая из сумм в (6) равна нулю. Следовательно, система (5) эквивалентна следующей

$$\sum_{k=1}^{i-1} \overline{r_{ki}} d_{kk} r_{kj} + r_{ii} d_{ii} r_{ij} = a_{ij}, \qquad i \le j, \;\; i,j = 1,\dots,n, \tag{7}$$

(здесь считается, что сумма вида $\sum_{k=1}^{i-1}$ равна нулю, если верхний предел суммирования меньше нижнего; это позволяет не рассматривать отдельно случай $i = 1$; также в (7) мы учли, что r_{ii} – вещественный элемент). Выделим в сумме (7) отдельно случай $i = j$

$$\begin{cases} r_{ii}^2 d_{ii} &= a_{ii} - \sum\limits_{k=1}^{i-1} |r_{ki}|^2 d_{kk}, & i = 1,\dots,n, \\ r_{ii} d_{ii} r_{ij} &= a_{ij} - \sum\limits_{k=1}^{i-1} \overline{r_{ki}} d_{kk} r_{kj}, & i < j, \;\; i,j = 1,\dots,n, \end{cases}$$

Отсюда получаем расчетные формулы:

$$\begin{cases} d_{ii} &= \operatorname{sign}(a_{ii} - \sum\limits_{k=1}^{i-1} |r_{ki}|^2 d_{kk}), & i = 1,\dots,n, \\ r_{ii} &= \sqrt{|a_{ii} - \sum\limits_{k=1}^{i-1} |r_{ki}|^2 d_{kk}|}, & i = 1,\dots,n, \\ r_{ij} &= (a_{ij} - \sum\limits_{k=1}^{i-1} \overline{r_{ki}}\, d_{kk} r_{kj})/(r_{ii} d_{ii}), & i < j, \;\; i,j = 1,\dots,n. \end{cases} \tag{8}$$

Процесс вычислений по этим формулам строится следующим образом: вначале вычисляются неизвестные элементы первых строк матриц D и R:

$$d_{11} = \operatorname{sign} a_{11}, \quad r_{11} = \sqrt{|a_{11}|}, \quad , r_{1j} = a_{1j}/(r_{11} d_{11}), \;\; j = 2,\dots,n;$$

потом по формулам (8) при $i = 2$ вычисляются неизвестные элементы вторых строк матриц D и R:

$$d_{22} = \operatorname{sign}(a_{22} - |r_{12}|^2 d_{11}), \quad r_{22} = \sqrt{|a_{22} - |r_{12}|^2 d_{11}|},$$
$$r_{2j} = (a_{2j} - \overline{r_{12}}\, d_{11} r_{1j})/(r_{22} d_{22}), \quad j = 3, \ldots, n;$$

затем по формулам (8) при $i = 3$ вычисляются неизвестные элементы третьих строк матриц D и R и так далее.

Замечание 4. Организация хранения матриц A, R и D в памяти.

Поскольку для самосопряженной матрицы $a_{ji} = \overline{a_{ij}}$, то можно вместо всей матрицы A хранить только ее верхний треугольник: a_{ij}, $i \le j$, $i, j = 1, \ldots, n$.

Формулы (8) таковы, что при вычислении элемента r_{ij} используются значения элемента a_{ij} и вычисленных ранее элементов r_{km}, $k < i$. Это позволяет хранить вернюю треугольную матрицу R на месте верхнего треугольника матрицы A: $r_{ij} \equiv a_{ij}$, $i \le j$, $i, j = 1, \ldots, n$. Матрица D обычно хранится в виде отдельного вектора $d = (d_{11}, \ldots, d_{nn})^t$.

Замечание 5. Случай положительной матрицы A.

В силу замечания 2 в этом случае разложение матрицы имеет более простой вид $A = R^* R$. Формулы (8) тоже упрощаются:

$$\left[\begin{array}{ll} r_{ii} = \sqrt{(a_{ii} - \sum\limits_{k=1}^{i-1} |r_{ki}|^2)}, & i = 1, \ldots, n, \\ r_{ij} = (a_{ij} - \sum\limits_{k=1}^{i-1} \overline{r_{ki}}\, r_{kj})/r_{ii}, & i < j, \quad i, j = 1, \ldots, n. \end{array}\right. \tag{9}$$

Если матрица A еще и вещественная, то в формулах (9) можно убрать знак модуля и комплексного сопряжения:

$$\left[\begin{array}{ll} r_{ii} = \sqrt{(a_{ii} - \sum\limits_{k=1}^{i-1} r_{ki}^2)}, & i = 1, \ldots, n, \\ r_{ij} = (a_{ij} - \sum\limits_{k=1}^{i-1} r_{ki} r_{kj})/r_{ii}, & i < j, \quad i, j = 1, \ldots, n. \end{array}\right.$$

§ 10.3. Оценка количества арифметических операций в алгоритме построения разложения Холецкого

1. Вычисление элемента d_{ii}, $i = 1, \ldots, n$ по формулам (8) требует $i-1$ мультипликативных и столько же аддитивных операций. Следовательно, вычисление всех элементов матрицы D требует $\sum_{i=1}^{n}(i-1) = n(n-1)/2 = O(n^2)$ мультипликативных и столько же аддитивных операций.

2. Вычисление элемента r_{ii} по формулам (8) требует одной операции извлечения корня и $i-1$ мультипликативных и столько же аддитивных операций (умножение на $d_{ii} \in \{-1, 1\}$ мы за операцию не считаем). При фиксированном $i = 1, \ldots, n$ вычисление элементов r_{ij} для всех $j = i+1, \ldots, n$ по формулам (8) требует $1 + \sum_{j=i+1}^{n}(i-1) = (n-i)(i-1) + 1$ мультипликативных и $\sum_{j=i+1}^{n}(i-1) = (n-i)(i-1)$ аддитивных операций. Следовательно, вычисление всех элементов матрицы R требует n операций извлечения корня, $\sum_{i=1}^{n}((n-i)(i-1) + (i-1) + 1) = n^3/6 + O(n^2)$ $(n \to \infty)$ мультипликативных и $\sum_{i=1}^{n}((n-i)(i-1)+(i-1)) = n^3/6+O(n^2)$ $(n \to \infty)$ аддитивных операций (подробное вычисление см. при подсчете количества арифметических операций для алгоритма LU-разложения).

На вычислительных машинах, имеющих аппаратную поддержку для вычисления трансцендентных функций вещественного аргумента, операция извлечения корня выполняется за время, по порядку равное времени деления двух вещественных чисел. Поэтому на таких вычислительных системах операции извлечения корня в описанном выше алгоритме займут время, равное времени выполнения $O(n)$ мультипликативных операций.

На вычислительных машинах, не имеющих аппаратной поддержки для вычисления трансцендентных функций вещественного аргумента, операции извлечения корня в описанном выше алгоритме займут время, равное времени выполнения $O(\mathrm{const}\, n) = O(n)$ мультипликативных операций.

Таким образом, алгоритм построения разложения Холецкого требует для своего проведения выполнения $n^3/6 + O(n^2)$ $(n \to \infty)$ мультипликативных и столько же аддитивных операций, а в сумме – $n^3/3 + O(n^2)$ $(n \to \infty)$ арифметических операций, т.е. асимптотически вдвое меньше, чем в методе Гаусса или алгоритме построения LU-разложения.

§ 11. МЕТОД ОРТОГОНАЛИЗАЦИИ

Пусть требуется решить линейную систему

$$A\,x = b \tag{1}$$

с матрицей $A = (a_{ij}) \in \mathbf{M}_n$ и правой частью $b = (b_1, \ldots, b_n)^t$. Пусть $x = (x_1, \ldots, x_n)^t$ – точное решение этой системы. Введем следующие вектора из $\mathbf{C}^{n+1}$:

$$Y = (x_1, \ldots, x_n, 1)^t \in \mathbf{C}^{n+1}, \tag{2}$$

$$A_i = (a_{i1}, \ldots, a_{in}, -b_i)^t \in \mathbf{C}^{n+1},\ i = 1, \ldots, n. \tag{3}$$

С использованием этих обозначений система (1) может быть записана в виде n условий ортогональности

$$(Y, A_i) = 0, \qquad i = 1, \ldots, n, \tag{4}$$

где $(\cdot, \cdot)$ – обычное евклидово скалярное произведение в $\mathbf{C}^{n+1}$. Соотношения (4) дают новое определение понятию решения системы (1): решить систему (1) – это значит найти вектор Y вида (2), ортогональный всем векторам (3).

Будем строить последовательность подпространств

$$\mathbf{C}^{n+1} = \mathbf{E}^{(0)} \supset \mathbf{E}^{(1)} \supset \ldots \supset \mathbf{E}^{(n)},$$

где $\mathbf{E}^{(k)} = \langle e^{(k)}_{k+1}, \ldots, e^{(k)}_{n+1} \rangle$ – $(n-k+1)$-мерное подпространство в $\mathbf{C}^{n+1}$, состоящее из векторов, ортогональных векторам $A_1, \ldots, A_k$ (в частности, все вектора одномерного подпространства $\mathbf{E}^{(n)}$ ортогональны $A_1, \ldots, A_n$, т.е. $\mathbf{E}^{(n)}$ содержит решение, которое выделяется из всех других векторов этого подпространства тем, что его последняя координата равна 1).

В качестве базиса начального пространства $\mathbf{E}^{(0)}$ возьмем стандартный базис $\mathbf{C}^{n+1}$: $e^{(0)}_1 = (1, 0, \ldots, 0), \ldots, e^{(0)}_{n+1} = (0, \ldots, 0, 1)$.

Для всех $k = 1, \ldots, n$ базис $e^{(k)}_{k+1}, \ldots, e^{(k)}_{n+1}$ каждого следующего подпространства $\mathbf{E}^{(k)}$ строится по предыдущему базису $e^{(k-1)}_k, \ldots, e^{(k-1)}_{n+1}$ подпространства $\mathbf{E}^{(k-1)}$ по формулам

$$e^{(k)}_i = e^{(k-1)}_i - \frac{(A_k, e^{(k-1)}_i)}{(A_k, e^{(k-1)}_k)}\, e^{(k-1)}_k, \quad i = k+1, \ldots, n. \tag{5}$$

Проверим, что $\mathbf{E}^{(k)}$ состоит из векторов, ортогональных векторам $A_1, \ldots, A_k$.

а) Базис пространства $\mathbf{E}^{(k)}$ получается как линейная комбинация базиса пространства $\mathbf{E}^{(k-1)}$, все элементы которого ортогональны векторам $A_1, \ldots, A_{k-1}$. Следовательно, вектора из $\mathbf{E}^{(k)}$ ортогональны векторам $A_1, \ldots, A_{k-1}$.

б) Проверим, что все элементы базиса пространства $\mathbf{E}^{(k)}$ ортогональны вектору A_k. $(A_k, e^{(k)}_i) = (A_k, e^{(k-1)}_i) - \frac{(A_k, e^{(k-1)}_i)}{(A_k, e^{(k-1)}_k)} (A_k, e^{(k-1)}_k) = (A_k, e^{(k-1)}_i) - (A_k, e^{(k-1)}_i) = 0, \quad i = k+1, \ldots, n.$

Лемма 1. *Для всех $k = 0, \ldots, n$ и всех $i = k+1, \ldots, n+1$ вектор $e^{(k)}_i$ имеет не более $k+1$ отличных от нуля компонент. Именно, могут быть отличными от нуля компоненты этого вектора с номерами $1, \ldots, k$ и i.*

Доказательство. При $k = 0$ это следует из выбора $e^{(0)}_i$, $i = 1, \ldots, n$. Предположим, что утверждение леммы справедливо для $e^{(k-1)}_i$, $i = k, \ldots, n$. Покажем,

что оно верно для $e_i^{(k)}$, $i = k+1, \ldots, n$. Действительно, по предположению у вектора $e_k^{(k-1)}$ только компоненты $1, \ldots, k$ могут быть отличны от нуля. Поэтому из формул (5) получаем, что вектор $e_i^{(k)}$ получается из вектора $e_i^{(k-1)}$ изменением не более чем первых k компонент. По предположению у вектора $e_i^{(k-1)}$ возможно отличны от нуля только компоненты $1, \ldots, k-1$ и i. Следовательно, у $e_i^{(k)}$, $i = k+1, \ldots, n$ могут быть отличны от нуля компоненты с номерами $1, \ldots, k$ и i.

Следствие 1. По доказанному в лемме 1 вектор $e_i^{(k)}$ получается из $e_i^{(k)}$ изменением компонент $1, \ldots, k$ (остальные компоненты у вектора $e_k^{(k-1)}$ нулевые). Следовательно, у всех $e_{n+1}^{(k)}$, $k = 1, \ldots, n$ $(n+1)$-я компонента равна 1. Поэтому вектор $e_{n+1}^{(n)}$ является решением задачи (4).

Оценка количества арифметических операций в методе ортогонализации

Оценим трудоемкость вычислений по формулам (5) для фиксированного k, а затем просуммируем полученные оценки по всем $k = 1, \ldots, n$.

1. Знаменатель $(A_k, e_k^{(k-1)})$ в (5) от i не зависит и вычисляется один раз для каждого k. В силу леммы 1 у вектора $e_k^{(k-1)}$ только компоненты $1, \ldots, k$ могут быть отличны от нуля, поэтому на вычисление скалярного произведения $(A_k, e_k^{(k-1)})$ потребуется k операций умножения и $k-1$ операций сложения.

2. На вычисление скалярного произведения $(A_k, e_i^{(k-1)})$ в (5) по лемме 1 потребуется k операций умножения и $k-1$ операций сложения (поскольку у вектора $e_i^{(k-1)}$ только компоненты $1, \ldots, k-1$ и i могут быть отличны от нуля). Следовательно, на вычисление этих скалярных произведений для всех $i = k+1, \ldots, n$ потребуется $\sum_{i=k+1}^{n} k = k(n-k)$ операций умножения и $\sum_{i=k+1}^{n}(k-1) = (k-1)(n-k)$ операций сложения.

3. На вычисление дроби $\dfrac{(A_k, e_i^{(k-1)})}{(A_k, e_k^{(k-1)})}$ для всех $i = k+1, \ldots, n$ в (5) потребуется $n-k$ операций деления.

4. На вычисление вектора $e_i^{(k)}$ по формуле (5) при условии, что дробь $\dfrac{(A_k, e_i^{(k-1)})}{(A_k, e_k^{(k-1)})}$ уже вычислена, потребуется k операций умножения и столько же операций сложения (поскольку в силу леммы 1 у вектора $e_k^{(k-1)}$ только компоненты $1, \ldots, k$ могут быть отличны от нуля). Следовательно, на проведение этих вычислений для всех $i = k+1, \ldots, n$ потребуется $\sum_{i=k+1}^{n} k = k(n-k)$ операций умножения и столько же операций сложения.

Итак, при фиксированном $k = 1, \ldots, n$ трудоемкость формул (5) составляет $k + k(n-k) + (n-k) + k(n-k) = 2k(n-k) + n$ мультипликативных операций и $k - 1 + (k-1)(n-k) + k(n-k) = 2k(n-k) + 2k - n - 1$ аддитивных операций. Следовательно, трудоемкость всего метода ортогонализации составляет

$\sum_{k=1}^{n}(2k(n-k)+n) = 2n\sum_{k=1}^{n}k - 2\sum_{k=1}^{n}k^2 + n^2 = 2n\cdot n(n+1)/2 - 2n(n+1)(2n+1)/6 + n^2 = n^3 + O(n^2) - \frac{2}{3}n^3 + O(n^2) = n^3/3 + O(n^2)$ мультипликативных операций и $\sum_{k=1}^{n}(2k(n-k)+2k-n-1) = n^3/3 + O(n^2)$ аддитивных операций. Таким образом, метод ортогонализации асимптотически требует такого же количества арифметических операций (суммарно $\frac{2}{3}n^3 + O(n^2)$ $(n \to \infty)$), как и метод Гаусса.

МЕТОДЫ РЕШЕНИЯ ЛИНЕЙНЫХ СИСТЕМ, ОСНОВАННЫЕ НА УНИТАРНЫХ ПРЕОБРАЗОВАНИЯХ МАТРИЦ

Каждый из изложенных выше методов решения линейных систем может быть представлен в виде последовательности элементарных преобразований матрицы (см., например, такое представление в §4 для метода Гаусса). Каждое из преобразований задается некоторой матрицей P, так что применение этого преобразования эквивалентно умножению (слева) исходной матрицы A на матрицу P. Таким образом, каждый шаг приведенных выше алгоритмов есть переход от матрицы A к матрице $A := PA$. О числе обусловленности этой новой матрицы $A := PA$ можно лишь утверждать, что $\kappa(PA) \le \kappa(P)\kappa(A)$. Поэтому может случиться так, что в процессе проведения преобразований число обусловленности матрицы возрастает и на каждом шаге метод будет вносить все большую вычислительную погрешность. В результате может оказаться, что исходная матрица имела приемлемое число обусловленности, однако после нескольких шагов алгоритма она уже имеет слишком большое число обусловленности, так что последующие шаги алгоритма приведут к появлению очень большой вычислительной погрешности.

Возникает идея подбирать матрицы преобразования P так, чтобы число обусловленности матрицы в процессе преобразований не возрастало. Лемма 1.5 указывает нам пример таких матриц: если матрица преобразования P унитарна (ортогональна в вещественном случае), то относительно спектральной нормы $\kappa(PA) = \kappa(A)$.

Излагаемые ниже метод вращений и метод отражений представляют собой алгоритмы подбора унитарных матриц преобразований P, таких, что в результате всех этих преобразований исходная матрица A приводится к треугольному виду. Система с треугольной матрицей затем решается, например, обратным ходом метода Гаусса. Несмотря на то, что трудоемкость этих методов больше, чем метода Гаусса (соответственно в 3 и 2 раза), эти методы получили широкое распространение в вычислительной практике благодаря своей устойчивости к накоплению вычислительной погрешности.

§ 12. МЕТОД ВРАЩЕНИЙ

В этом методе в качестве элементарного преобразования матрицы выбирается умножение ее на матрицу вращения.

§ 12.1. Матрица элементарного вращения и ее свойства

Определение. *Элементарным вращением* $T_{ij} = T_{ij}(\varphi)$ называется преобразование пространства, задаваемое матрицей $T_{ij} = (t_{kl})_{k,l=1,\dots,n}$, в которой только следующие элементы отличны от нуля: $t_{ii} = \cos\varphi$, $t_{jj} = \cos\varphi$, $t_{ij} = -\sin\varphi$, $t_{ji} = \sin\varphi$, $t_{kk} = 1$ для всех $k = 1, \dots, n, k \neq i, j$:

$$T_{ij} = \begin{pmatrix} 1 & & & & & & & & & \\ & \ddots & & & & & & & & \\ & & 1 & & & & & & & \\ & & & \cos\varphi & & & & -\sin\varphi & & \\ & & & & 1 & & & & & \\ & & & & & \ddots & & & & \\ & & & & & & 1 & & & \\ & & & \sin\varphi & & & & \cos\varphi & & \\ & & & & & & & & 1 & \\ & & & & & & & & & \ddots \\ & & & & & & & & & & 1 \end{pmatrix}. \qquad (1)$$

Если $\langle e_1, \dots, e_n \rangle$ – базис $\mathbf{C}^n$ ($e_k = (\underbrace{0, \dots, 0}_{k-1}, 1, 0, \dots, 0)^t$), то T_{ij} является вращением в подпространстве $\langle e_i, e_j \rangle$ и не изменяет подпространства $\langle e_1, \dots, e_{i-1}, e_{i+1}, \dots, e_{j-1}, e_{j+1}, \dots, e_n \rangle$ (другими словами, T_{ij} изменяет только i-ю и j-ю координаты векторов). Поэтому для изучения свойств преобразования T_{ij} достаточно изучить свойства преобразования

$$T = \begin{pmatrix} \cos\varphi & -\sin\varphi \\ \sin\varphi & \cos\varphi \end{pmatrix}$$

в двумерном пространстве.

Лемма 1. *Матрица T_{ij} является ортогональной матрицей.*

Доказательство. Следует из ортогональности матрицы T.

Лемма 2. *Для всякого вектора* $\mathbf{r} = \begin{pmatrix} x \\ y \end{pmatrix} \in \mathbf{R}^2$, $\mathbf{r} \neq 0$ *существует матрица* $T = \begin{pmatrix} \cos\varphi & -\sin\varphi \\ \sin\varphi & \cos\varphi \end{pmatrix}$, *такая, что* $T\mathbf{r} = \|\mathbf{r}\| \begin{pmatrix} 1 \\ 0 \end{pmatrix}$, *где* $\|\mathbf{r}\| = \sqrt{x^2 + y^2}$ – *евклидова длина вектора* $\mathbf{r}$. *При этом трудоемкость построения матрицы T составляет 4 мультипликативные операции, одну аддитивную и одну операцию извлечения корня.*

Доказательство. Достаточно положить

$$\cos\varphi = \frac{x}{\sqrt{x^2+y^2}}\,, \quad \sin\varphi = -\frac{y}{\sqrt{x^2+y^2}}\,.$$

Лемма 3. *Для всякого вектора* $x \in \mathbf{R}^n$, $x \neq 0$ *существуют* $n-1$ *матриц* $T_{12} = T_{12}(\varphi_{12})$, $T_{13} = T_{13}(\varphi_{13})$, $\ldots$, $T_{1n} = T_{1n}(\varphi_{1n})$, *таких, что* $T_{1n}\ldots T_{13}T_{12}x = \|x\|\,e_1$, *где* $\|x\| = \|x\|_2 = \sqrt{\sum\limits_{k=1}^{n} x_k^2}$ *– евклидова длина вектора* x, $e_1 = (1,0,\ldots,0)^t$ *– первый координатный орт.*

Доказательство. Если вектор $\mathbf{r} = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \neq 0$, то по лемме 2 существует матрица элементарного вращения

$$T = T(\varphi_{12}) = \begin{pmatrix} \cos\varphi_{12} & -\sin\varphi_{12} \\ \sin\varphi_{12} & \cos\varphi_{12} \end{pmatrix},$$

такая, что $T\mathbf{r} = \|\mathbf{r}\| \begin{pmatrix} 1 \\ 0 \end{pmatrix}$. Тогда матрица

$$T_{12} = T_{12}(\varphi_{12}) = \begin{pmatrix} \cos\varphi_{12} & -\sin\varphi_{12} & & & \\ \sin\varphi_{12} & \cos\varphi_{12} & & & \\ & & 1 & & \\ & & & \ddots & \\ & & & & 1 \end{pmatrix}$$

переводит вектор x в вектор $x^{(2)} = T_{12}x = (\sqrt{x_1^2+x_2^2}, 0, x_3, \ldots, x_n)^t$. Если вектор $\mathbf{r} = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = 0$, то преобразование не осуществляется ($T_{12} = I$ – единичной матрице).

После $k-1$ шагов этого процесса ($k = 1, \ldots, n-1$) вектор x преобразован к виду $x^{(k)} = T_{1k}\ldots T_{12}x = (\sqrt{\sum_{i=1}^{k} x_i^2}, 0, \ldots, 0, x_{k+1}, \ldots, x_n)^t$. Если вектор

$$\mathbf{r} = \begin{pmatrix} \sqrt{\sum_{i=1}^{k} x_i^2} \\ x_{k+1} \end{pmatrix} \in \mathbf{R}^2, \quad \mathbf{r} \neq 0,$$

то по лемме 2 существует матрица элементарного вращения

$$T = T(\varphi_{1,k+1}) = \begin{pmatrix} \cos\varphi_{1,k+1} & -\sin\varphi_{1,k+1} \\ \sin\varphi_{1,k+1} & \cos\varphi_{1,k+1} \end{pmatrix},$$

такая, что $T\mathbf{r} = \|\mathbf{r}\| \begin{pmatrix} 1 \\ 0 \end{pmatrix}$. Тогда матрица

$$T_{1,k+1} = T_{1,k+1}(\varphi_{1,k+1}) = \begin{pmatrix} \overbrace{\cos\varphi_{1,k+1} & & & -\sin\varphi_{1,k+1}}^{k+1} & & \\ & 1 & & & & \\ & & \ddots & & & \\ & & & 1 & & \\ \sin\varphi_{1,k+1} & & & \cos\varphi_{1,k+1} & & \\ & & & & 1 & \\ & & & & & \ddots \\ & & & & & & 1 \end{pmatrix}$$

переводит вектор $x^{(k)}$ в вектор

$$x^{(k+1)} = T_{1,k+1}x^{(k)} = T_{1,k+1}T_{1k}\dots T_{12}x = (\sqrt{\sum_{i=1}^{k+1} x_i^2}, 0, \dots, 0, x_{k+2}, \dots, x_n)^t.$$

Если вектор $\mathbf{r} = 0$, то преобразование не осуществляется ($T_{1,k+1} = I$ – единичной матрице).

После $n-1$ шагов этого процесса вектор x будет преобразован к виду $x^{(n)} = T_{1n}\dots T_{12}x = (\sqrt{\sum_{i=1}^{n} x_i^2}, 0, \dots, 0)^t = \|x\|\, e_1$.

Лемма 4. *Произведение матрицы элементарного вращения на вектор может быть вычислено за 4 умножения и 2 сложения.*

Доказательство. Произведение $y = T_{ij}(\varphi_{ij})x$ матрицы элементарного вращения T_{ij} на вектор x имеет следующие компоненты:

$$y_k = x_k,\ k = 1, \dots, n,\ k \neq i, j,\quad y_i = x_i\cos\varphi_{ij} - x_j\sin\varphi_{ij},\quad y_j = x_i\sin\varphi_{ij} + x_j\cos\varphi_{ij}.$$

При осуществлении вычислений по этим формулам надо выполнить 4 умножения и 2 сложения.

Замечание 1. Для вычисления произведения матрицы из $\mathbf{M}_n$ общего вида на вектор требуется выполнить n^2 умножений и $n(n-1)$ сложений.

Лемма 5. *Произведение матрицы элементарного вращения $T_{ij} \in \mathbf{M}_n$ на матрицу размера $n \times m$ может быть вычислено за $4m$ умножений и $2m$ сложений.*

Доказательство. Пусть $n \times m$ матрица $Y = T_{ij}X$ есть произведение матрицы элементарного вращения $T_{ij} \in \mathbf{M}_n$ на $n \times m$ матрицу X. Запишем матрицы $X = (x_{ij})$ и $Y = (y_{ij})$ через их столбцы: $X = [x^{(1)}, \dots, x^{(m)}]$, $Y = [y^{(1)}, \dots, y^{(m)}]$,

где $x^{(k)} = (x_{1k}, \ldots, x_{nk})^t$, $y^{(k)} = (y_{1k}, \ldots, y_{nk})^t$, $k = 1, \ldots, m$. Согласно определению произведения матриц $Y = T_{ij}X = [T_{ij}x^{(1)}, \ldots, T_{ij}x^{(m)}]$, т.е. $y^{(k)} = T_{ij}x^{(k)}$, $k = 1, \ldots, m$. Таким образом, для вычисления матрицы $Y = T_{ij}X$ надо вычислить m произведений $T_{ij}x^{(k)}$ матрицы T_{ij} на вектора $x^{(k)}$, $k = 1, \ldots, m$. Доказываемое утверждение теперь вытекает из леммы 4.

Замечание 2. Для вычисления произведения двух матриц из $\mathbf{M}_n$ общего вида требуется выполнить n^3 умножений и $n^2(n-1)$ сложений.

§ 12.2. Алгоритм метода вращений

Пусть требуется решить линейную систему $Ax = b$, $A \in \mathbf{M}_n(\mathbf{R}^n)$ вида (4.1).

Обозначим $a_1 = (a_{11}, \ldots, a_{n1})^t$ – первый столбец матрицы A. Согласно лемме 3 существуют $n-1$ матриц $T_{12} = T_{12}(\varphi_{12})$, $T_{13} = T_{13}(\varphi_{13})$, $\ldots$, $T_{1n} = T_{1n}(\varphi_{1n})$, таких, что $T_{1n} \ldots T_{13}T_{12}a_1 = \|a_1\| e_1$ (причем значения углов φ_{1k}, $k = 2, \ldots, n$ определяются леммами 2, 3). Умножим систему $Ax = b$ на $T_{1n} \ldots T_{13}T_{12}$ слева, получим

$$A^{(1)}x = b^{(1)},$$

где

$$A^{(1)} = T_{1n} \ldots T_{13}T_{12}A = \begin{pmatrix} \|a_1\| & c_{12} & \ldots & c_{1n} \\ 0 & a_{22}^{(1)} & \ldots & a_{2n}^{(1)} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & a_{n2}^{(1)} & \ldots & a_{nn}^{(1)} \end{pmatrix}, \quad b^{(1)} = T_{1n} \ldots T_{13}T_{12}b.$$

Далее процесс применяется к подматрице $(a_{ij}^{(1)})_{i,j=2,\ldots,n}$.

Пусть сделаны $k-1$, $k = 1, \ldots, n-1$ шагов этого процесса, т.е. система преобразована к виду

$$A^{(k-1)}x = b^{(k-1)}, \tag{2}$$

где

$$A^{(k-1)} = \prod_{i=k-1}^{1} \prod_{j=n}^{i+1} T_{ij}A, \quad b^{(k-1)} = \prod_{i=k-1}^{1} \prod_{j=n}^{i+1} T_{ij}b, \tag{3}$$

$$A^{(k-1)} = \begin{pmatrix} \|a_1\| & c_{12} & c_{13} & \ldots & c_{1,k-1} & c_{1k} & \ldots & c_{1n} \\ & \|a_1^{(1)}\| & c_{23} & \ldots & c_{2,k-1} & c_{2k} & \ldots & c_{2n} \\ & & \|a_1^{(2)}\| & \ldots & c_{3,k-1} & c_{3k} & \ldots & c_{3n} \\ & & & \ddots & \vdots & \vdots & \ddots & \vdots \\ & & & & \|a_1^{(k-2)}\| & c_{k-1,k} & \ldots & c_{k-1,n} \\ & & & & & a_{kk}^{(k-1)} & \ldots & a_{kn}^{(k-1)} \\ & & & & & \vdots & \ddots & \vdots \\ & & & & & a_{nk}^{(k-1)} & \ldots & a_{nn}^{(k-1)} \end{pmatrix}. \tag{4}$$

(здесь $\prod\limits_{j=n}^{i+1}$ означает, что сомножители берутся в порядке $n, \dots, i+1$).

Обозначим

$$a_1^{(k-1)} = (a_{kk}^{(k-1)}, \dots, a_{nk}^{(k-1)})^t \tag{5}$$

– первый столбец подматрицы $(a_{ij}^{(k-1)})_{i,j=k,\dots,n}$. Согласно лемме 3 существуют $n-k$ матриц

$$T_{k,k+1} = T_{k,k+1}(\varphi_{k,k+1}),\ T_{k,k+2} = T_{k,k+2}(\varphi_{k,k+2}),\ \dots,\ T_{k,n} = T_{kn}(\varphi_{kn}),$$

таких, что

$$T_{kn} \dots T_{k,k+2} T_{k,k+1} a_1^{(k-1)} = \|a_1^{(k-1)}\|\, e_1^{(n-k+1)}, \tag{6}$$

(значения углов φ_{kj}, $j = k+1, \dots, n$ определяются леммами 2, 3), здесь $e_1^{(m)} = (1, 0, \dots, 0)^t \in \mathbf{R}^m$. Умножим систему (2) на $T_{kn} \dots T_{k,k+2} T_{k,k+1}$ слева, получим

$$A^{(k)} x = b^{(k)},$$

где

$$A^{(k)} = \prod_{j=n}^{k+1} T_{kj} A^{(k-1)} = \prod_{i=k}^{1} \prod_{j=n}^{i+1} T_{ij} A, \quad b^{(k)} = \prod_{j=n}^{k+1} T_{kj} b^{(k-1)} = \prod_{i=k}^{1} \prod_{j=n}^{i+1} T_{ij} b, \tag{7}$$

$$A^{(k)} = \begin{pmatrix} \|a_1\| & c_{12} & c_{13} & \dots & c_{1,k-1} & c_{1k} & c_{1,k+1} & \dots & c_{1n} \\ & \|a_1^{(1)}\| & c_{23} & \dots & c_{2,k-1} & c_{2k} & c_{2,k+1} & \dots & c_{2n} \\ & & \|a_1^{(2)}\| & \dots & c_{3,k-1} & c_{3k} & c_{3,k+1} & \dots & c_{3n} \\ & & & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ & & & & \|a_1^{(k-2)}\| & c_{k-1,k} & c_{k-1,k+1} & \dots & c_{k-1,n} \\ & & & & & \|a_1^{(k-1)}\| & c_{k,k+1} & \dots & c_{k,n} \\ & & & & & & a_{k+1,k+1}^{(k)} & \dots & a_{k+1,n}^{(k)} \\ & & & & & & \vdots & \ddots & \vdots \\ & & & & & & a_{n,k+1}^{(k)} & \dots & a_{nn}^{(k)} \end{pmatrix}. \tag{8}$$

Отметим, что в (7) каждая из $n-k$ матриц элементарных вращений T_{kj} такова, что $j > k$ и потому в (7) она умножается только на подматрицу $(a_{ij}^{(k-1)})_{i,j=k,\dots,n}$ матрицы $A^{(k-1)}$ размера $n-k+1$ (остальная часть $A^{(k-1)}$ в преобразовании (7) не участвует).

После $n-1$ шагов этого процесса (т.е. перехода от матриц и правых частей (3), (4) к (7), (8)) система примет вид

$$R\,x = y, \tag{9}$$

где

$$R = A^{(n-1)} = \prod_{i=n-1}^{1} \prod_{j=n}^{i+1} T_{ij} A, \quad y = b^{(n-1)} = \prod_{i=n-1}^{n-1} \prod_{j=n}^{i+1} T_{ij} b, \tag{10}$$

$$R = \begin{pmatrix} \|a_1\| & c_{12} & c_{13} & \dots & c_{1,n-2} & c_{1,n-1} & c_{1n} \\ & \|a_1^{(1)}\| & c_{23} & \dots & c_{2,n-2} & c_{2,n-1} & c_{2n} \\ & & \|a_1^{(2)}\| & \dots & c_{3,n-2} & c_{3,n-1} & c_{3n} \\ & & & \ddots & \vdots & \vdots & \vdots \\ & & & & \|a_1^{(n-3)}\| & c_{n-2,n-1} & c_{n-2,n} \\ & & & & & \|a_1^{(n-2)}\| & c_{n-1,n} \\ & & & & & & a_{nn}^{(n-1)} \end{pmatrix} \tag{11}$$

(напомним, определения векторов $a_1^{(k-1)}$, $k = 1, \dots, n-1$ даются в (5), где считаем, что $a_1^{(0)} = a_1$).

Система (9) с верхней треугольной матрицей R решается обратным ходом метода Гаусса.

§ 12.3. Оценка количества арифметических операций в методе вращений

Оценим трудоемкость k-го шага алгоритма, а затем просуммируем полученные оценки по всем $k = 1, \dots, n-1$.

1. На вычисление $n-k$ матриц $T_{k,k+1}, \dots, T_{kn}$, участвующих в (6), согласно лемме 2 требуется $4(n-k)$ мультипликативных, $(n-k)$ аддитивных и $n-k$ операций извлечения корня.

2. На вычисление компонент $k, \dots, n$ k-го столбца матрицы $A^{(k)}$, равных компонентам вектора $\|a_1^{(k-1)}\| e_1^{(n-k+1)}$ требуется (для вычисления длины вектора (5)) $n-k+1$ операций умножения, $n-k$ операций сложения и одна операция извлечения корня. Столбец k вычисляется именно этим способом (а не по общим формулам (7)) для сокращения количества арифметических операций и уменьшения вычислительной погрешности.

3. Поскольку в формуле (7) каждая из $n-k$ матриц элементарных вращений умножается на подматрицу $(a_{ij}^{(k-1)})_{i=k,\dots,n,\ j=k+1,\dots,n}$ матрицы $A^{(k-1)}$ размера $(n-k+1)\times(n-k)$ (k-й столбец матрицы $A^{(k)}$ уже вычислен в пункте 2), то согласно лемме 5 на это требуется $(n-k)4(n-k) = 4(n-k)^2$ умножений и $(n-k)2(n-k) = 2(n-k)^2$ сложений.

4. На вычисление новой правой части по формуле (7) согласно лемме 4 требуется $4(n-k)$ умножений и $(n-k)$ сложений.

Итак, на k-ом шаге алгоритма требуется выполнить $4(n-k) + (n-k+1) + 4(n-k)^2 + 4(n-k) = 4(n-k)^2 + 9(n-k) + 1$ мультипликативных операций, $(n-k) + (n-k) + 2(n-k)^2 + (n-k) = 2(n-k)^2 + 3(n-k)$ аддитивных операций и $n-k+1$ операций извлечения корня.

Следовательно, всего для проведения алгоритма требуется выполнить

$$\sum_{k=1}^{n-1} (4(n-k)^2 + 9(n-k) + 1) = 4n(n-1)(2n-1)/6 + 9n(n-1)/2 + n - 1$$

$$= \frac{4}{3}n^3 + O(n^2)\ (n \to \infty)$$

мультипликативных операций, $\sum_{k=1}^{n-1}(2(n-k)^2+3(n-k)) = \frac{2}{3}n^3 + O(n^2)\ (n \to \infty)$ аддитивных операций и $\sum_{k=1}^{n-1}(n-k+1) = O(n^2)\ (n \to \infty)$ операций извлечения корня (которые по трудоемкости по порядку можно сравнить с операциями деления).

На решение системы (9) с верхней треугольной матрицей R обратным ходом метода Гаусса требуется $O(n^2)\ (n \to \infty)$ арифметических операций.

Таким образом, на решение линейной системы методом вращений требуется $\frac{4}{3}n^3 + O(n^2)\ (n \to \infty)$ мультипликативных операций (что в 4 раза больше, чем в методе Гаусса), и $\frac{2}{3}n^3 + O(n^2)\ (n \to \infty)$ аддитивных операций (что в 2 раза больше, чем в методе Гаусса).

Теорема 1 (О QR-разложении). *Всякая невырожденная вещественная матрица A может быть представлена в виде $A = Q\,R$, где матрица Q – ортогональная, а матрица R – верхняя треугольная с положительными элементами на главной диагонали. Это разложение единственно.*

Доказательство. Проведем для матрицы A алгоритм метода вращений, осуществимый для всякой невырожденной матрицы. Обозначим в (10) $\hat{Q} = \prod\limits_{i=n-1}^{1}\prod\limits_{j=n}^{i+1} T_{ij}$. Как произведение ортогональных матриц, матрица $\hat{Q}$ ортогональна. Тогда (10) имеет вид $R = \hat{Q}A$, откуда $A = (\hat{Q})^{-1}R = QR$, где $Q = (\hat{Q})^t = (\hat{Q})^{-1}$. Если $a_{nn}^{(n-1)} > 0$, то матрица R, имеющая вид (11), удовлетворяет условиям теоремы. Если $a_{nn}^{(n-1)} < 0$, то положим $D = \mathrm{diag}(1,\ldots,1,\mathrm{sign}(a_{nn}^{(n-1)}))$. Матрица D ортогональна и $D^2 = I$. Поэтому $A = (QD)(DR)$, где $Q := QD$ и $R := DR$ удовлетворяют условиям теоремы.

Предположим, что возможно два различных разложения $A = QR$ и $A = Q'R'$, удовлетворяющих условиям теоремы. Тогда $QR = Q'R'$ и $(Q')^{-1}Q = R^{-1}R'$. В левой части последнего равенства стоит ортогональная матрица, а в правой – верхняя треугольная. Пересечение группы ортогональных матриц и группы верхних треугольных матриц состоит из матриц вида $D = \mathrm{diag}(d_1,\ldots,d_n)$, где $d_i \in \{-1,1\}$, $i = 1,\ldots,n$ (проверить самостоятельно). Поскольку диагональные элементы матрицы $R^{-1}R'$ равны произведениям диагональных элементов матриц R и R', то они положительны. Следовательно $R^{-1}R' = I$, т.е. $R = R'$. Также $(Q')^{-1}Q = I$, т.е. $Q = Q'$. Полученное противоречие доказывает теорему.

Замечание 3. QR-разложение матрицы A может быть использовано, например, для тех же целей, что и LU-разложение. Именно, пусть стоит задача решить серию систем вида $Ax_j = b_j$, $j = 1,\ldots,m$ с одной и той же матрицей A и разными правыми частями b_j. Построим QR-разложение матрицы A (которое, в отличие от LU-разложения, существует для всякой невырожденной матрицы). Для ортогональной матрицы Q легко находится обратная $Q^{(-1)} = Q^t$. Поэтому

x_j находятся как решения системы $R\,x_j = Q^t\,b_j$ с верхней треугольной матрицей R, например, обратным ходом метода Гаусса.

Замечание 4. QR-разложение матрицы A используется в QR-алгоритме нахождения собственных значений матрицы A.

§ 12.4. Построение QR-разложения методом вращений

Построение QR-разложения. Хранение матриц Q и R в памяти. Пусть стоит задача построить QR-разложение для матрицы A. Будем действовать как в теореме 1. Проведем для матрицы A метод вращений и получим в результате матрицу R из (11). При этом матрица Q равна (см. доказательство теоремы 1)

$$Q = (\prod_{i=n-1}^{1}\prod_{j=n}^{i+1} T_{ij})^t = \prod_{i=1}^{n-1}\prod_{j=i+1}^{n} T_{ij}^t. \qquad (12)$$

Возможны два способа хранения матриц Q и R в памяти.

1. Матрица R хранится на месте верхнего треугольника матрицы A и получается из нее последовательным применением элементарных вращений (см. выше алгоритм метода вращений). Для хранения матрицы Q выделяется отдельная матрица Q, которая равна единичной перед первым шагом алгоритма. На шаге k, $k = 1, \ldots, n-1$ эта матрица умножается справа на матрицу $\prod\limits_{j=n}^{k+1} T_{kj}$:

$$Q := Q \prod_{k+1}^{j=n} T_{kj}$$

(см. (7), (12)). Произведение матрицы элементарного вращения на матрицу вычисляется по алгоритму из леммы 5 с затратой $4n$ умножений и $2n$ сложений. Следовательно, произведение $n(n-1)/2$ матриц вращения в (12) может быть вычислено за $2n^2(n-1) = 2n^3 + O(n^2)$ $(n \to \infty)$ умножений и $n^2(n-1) = n^3 + O(n^2)$ $(n \to \infty)$ сложений.

2. Как и в первом способе, матрица R хранится на месте верхнего треугольника матрицы A. Для хранения же матрицы Q отдельная память не выделяется. Заметим, что на шаге k, $k = 1, \ldots, n-1$ мы использовали $n-k$ элементарных вращений $T_{k,k+1}, \ldots, T_{kn}$ и каждая из этих матриц целиком определяется единственным параметром – значением угла φ_{kj} : $T_{kj} = T_{kj}(\varphi_{kj})$, $j = k+1, \ldots, n$. При этом после преобразования (6), (7), т.е. перехода от матрицы (4) к матрице (8), в k-ом столбце матрицы $A^{(k)}$ образовались $n-k$ нулевых элементов $a_{jk}^{(k)} = 0$, $j = k+1, \ldots, n$. Поэтому возможно вместо матрицы Q вида (12) хранить на месте нижнего треугольника матрицы A набор параметров, с помощью которых можно вычислять тригонометрические функции углов φ_{ij}, $j < i$, $i = 2 \ldots, n$, $j = 1, \ldots, n-1$, задающих матрицы T_{ij}. Конечно, проще всего было

бы хранить сами эти углы φ_{ij}, но это требует вычисления обратных тригонометрических функций, что довольно медленно и вносит большую вычислительную погрешность. На практике на месте a_{ij}, $j < i$, $i = 2\ldots, n$, $j = 1, \ldots, n-1$ хранят $\cos\varphi_{ij}$ или $\sin\varphi_{ij}$ – тот который имеет наименьший модуль. При этом на месте двух младших битов мантиссы этой величины хранятся признак того, что было запомнено: sin или cos, и знак не запомненной тригонометрической функции. Измененение двух младших битов мантиссы у $\cos\varphi_{ij}$ или $\sin\varphi_{ij}$ вносит погрешность, намного меньшую чем погрешность, с которой они вычислены. Запоминание значения $\cos\varphi_{ij}$ или $\sin\varphi_{ij}$ с наименьшим модулем уменьшает погрешность при вычислении по формулам $\sin\varphi_{ij} = \pm\sqrt{1-\cos^2\varphi_{ij}}$ или $\cos\varphi_{ij} = \pm\sqrt{1-\sin^2\varphi_{ij}}$. Большинство современных микропроцессоров (Intel 80x86, Motorola 68xxx, SPARC, PowerPC) поддерживают стандарт ANSI/IEEE 754-1985 при работе с данными с плавающей точкой. Для таких процессоров младшие биты мантиссы являются младшими битами числа с плавающей точкой.

При втором способе хранения матрицы Q не только экономится n^3 ячеек памяти, но и экономится $2n^3+O(n^2)$ $(n\to\infty)$ умножений и $n^3+O(n^2)$ $(n\to\infty)$ сложений на построение матрицы Q. Второму способу хранения благоприятствует также то обстоятельство, что редко требуется знать матрицу Q "саму по себе". Обычно требуется уметь вычислять ее произведения на вектор и матрицу. Для того, чтобы вычислить произведение матрицы Q вида (12) на некоторую матрицу B требуется вычислить $n(n-1)/2$ произведений матриц элементарных вращений T_{ij} на B :

$$QB = \prod_{i=1}^{n-1}\prod_{j=i+1}^{n}(T_{ij}B).$$

По лемме 5 на это потребеутся $2n^2(n-1)$ умножений и $n^2(n-1)$ сложений. По сравнению с количеством операций, необходимых для вычисления произведения двух матриц Q и B произвольного вида, число умножений тут в 2 раза больше, а число сложений совпадает. Если таких произведений требуется вычислить не очень много, то второй способ предпочтительнее первого.

§ 12.5. Оценка количества арифметических операций в алгоритме построения QR-разложения методом вращений

Трудоемкость алгоритма построения QR-разложения складывается из количества арифметических операций, необходимых для проведения алгоритма метода вращений, и количества арифметических операций, необходимых для построения матрицы Q.

Если для Q используется второй способ хранения, то дополнительных действий для ее построения не требуется. Следовательно, в этом случае для построения QR-разложения надо выполнить $\frac{4}{3}n^3+O(n^2)$ $(n\to\infty)$ мультипликативных операций и $\frac{2}{3}n^3+O(n^2)$ $(n\to\infty)$ аддитивных операций.

Если для Q используется первый способ хранения, то как показано выше для ее построения дополнительно к $\frac{4}{3}n^3 + O(n^2)$ $(n \to \infty)$ мультипликативным и $\frac{2}{3}n^3 + O(n^2)$ $(n \to \infty)$ аддитивным операциям, необходимых для проведения алгоритма метода вращений, требуется $2n^3 + O(n^2)$ $(n \to \infty)$ умножений и $n^3 + O(n^2)$ $(n \to \infty)$ сложений, всего $\frac{10}{3}n^3 + O(n^2)$ $(n \to \infty)$ мультипликативных операций и $\frac{5}{3}n^3 + O(n^2)$ $(n \to \infty)$ аддитивных операций.

§ 13. МЕТОД ОТРАЖЕНИЙ

Всюду в данном параграфе под нормой вектора будет пониматься евклидова норма, а под нормой матрицы – спектральная норма.

Лемма 1. *Спектральная норма всякой унитарной (ортогональной в вещественном случае) матрицы равна* 1.

Доказательство. Поскольку унитарные матрицы сохраняют евклидову длину вектора, по определению спектральной нормы получаем для всякой унитарной матрицы U:

$$\|U\| = \sup_{x \neq 0} \frac{\|Ux\|}{\|x\|} = \sup_{x \neq 0} \frac{\|x\|}{\|x\|} = 1.$$

Лемма 2. *Собственные значения всякой унитарной матрицы по модулю равны* 1.

Доказательство. Пусть λ – произвольное собственное значение матрицы U. По лемме 1.4 $|\lambda| \leq \|U\| = 1$ – по предыдущей лемме. С другой стороны, λ^{-1} является собственным значением матрицы U^{-1}, которая тоже унитарна. Опять по лемме 1.4 и лемме 1 $|\lambda^{-1}| \leq \|U^{-1}\| = 1$, т.е. $|\lambda| \geq 1$. Следовательно, $|\lambda| = 1$.

Лемма 3. *Собственные значения всякой самосопряженной (симметричной в вещественном случае) матрицы A (т.е. $A^* = A$) вещественны.*

Доказательство. Пусть λ – произвольное собственное значение матрицы A, $x \neq 0$ – отвечающий ему собственный вектор, т.е. $Ax = \lambda x$. Умножим это равенство скалярно на x: $(Ax, x) = \lambda(x, x)$, откуда $\lambda = (Ax, x)/\|x\|^2$. В силу замечания 9.1 выражение (Ax, x) вещественно для самосопряженной матрицы A. Следовательно, λ вещественно.

§ 13.1. Матрица отражения и ее свойства

Определение. *Матрицей отражения* называется матрица вида $U = U(x) = I - 2xx^*$, где x – единичный вектор (т.е. $\|x\| = 1$). (Напомним, что $x^* = (\overline{x}_1, \ldots, \overline{x}_n)$ – "матрица" размера $1 \times n$, $x = (x_1, \ldots, x_n)^t$ "матрица" размера $n \times 1$ и потому xx^* – матрица размера $n \times n$.)

Установим основные свойства матрицы отражения.

Лемма 4. *Матрица отражения является самосопряженной матрицей.*

Доказательство. Вычислим сопряженную матрицу для матрицы отражения $U(x)$

$$(U(x))^* = (I - 2xx^*)^* = I - 2(x^*)^* x^* = I - 2xx^* = U(x),$$

что и означает самосопряженность матрицы $U(x)$.

Лемма 5. *Матрица отражения является унитарной матрицей.*

Доказательство. Вычислим для матрицы отражения $U(x)$

$$U(x)U^*(x) = U(x)^2 = (I - 2xx^*)(I - 2xx^*) = I - 4xx^* + 4xx^*xx^* = I - 4xx^* + 4x1x^* = I,$$

поскольку $x^*x = (x, x) = \|x\|^2 = 1$. Это равенство и означает унитарность матрицы $U(x)$.

Лемма 6. *Собственные значения матрицы отражения равны либо* 1, *либо* -1.

Доказательство. Из лемм 2 и 4 вытекает, что собственные значения матрицы отражения по модулю равны 1. Из лемм 3 и 5 следует, что они вещественны. Значит, собственные значения есть либо 1 либо -1.

Лемма 7. *Матрица отражения* $U(x)$ *имеет собственное значение* -1 *кратности* 1, *которому отвечает собственный вектор* x, *и собственное значение* 1 *кратности* $n - 1$, *которому отвечает собственное подпространство* $\langle x \rangle^\perp = \{y : (y, x) = 0\}$.

Доказательство. Имеем

$$U(x)x = (I - 2xx^*)x = x - 2xx^*x = x - 2x = -x,$$

поскольку $x^*x = (x, x) = \|x\|^2 = 1$. Следовательно, x – собственный вектор, отвечающий собственному значению -1.

Далее, для всех $y \in \langle x \rangle^\perp$

$$U(x)y = (I - 2xx^*)y = y - 2xx^*y = y,$$

поскольку $x^*y = (y, x) = 0$. Следовательно, y -собственный вектор, отвечающий собственному значению 1. Такие вектора $y \in \langle x \rangle^\perp$ образуют $(n-1)$-мерное подпространство.

Лемма 8. *Геометрический смысл преобразования, задаваемого матрицей отражения $U(x)$: отражение относительно гиперплоскости $\langle x\rangle^\perp$.*

Доказательство. Всякий вектор $z \in \mathbf{C}^n$ может быть представлен в виде $z = \alpha x + y$, где $y \in \langle x\rangle^\perp$. Здесь компонента αx параллельна x, а компонента y ортогональна x, т.е. лежит в гиперплоскости $\langle x\rangle^\perp$. В силу леммы 7 $U(x)z = U(x)(\alpha x + y) = -\alpha x + y$, т.е. вектор z отразился относительно гиперплоскости $\langle x\rangle^\perp$.

Лемма 9. *Пусть e – произвольный единичный вектор: $\|e\| = 1$. Тогда для всякого вектора $y \in \mathbf{C}^n$ существует вектор $x \in \mathbf{C}^n$, $\|x\| = 1$ такой, что $U(x)y = \|y\|e$.*

Доказательство. Так как вектора y и $\|y\|e$ должны быть получены друг из друга отражением относительно гиперплоскости $\langle x\rangle^\perp$, то вектор $y - \|y\|e$ должен быть параллелен x, т.е. $x = \alpha(y - \|y\|e)$. Коэффициент α найдем из условия $\|x\| = 1$. Получаем

$$x = \pm\frac{y - \|y\|e}{\|y - \|y\|e\|}.$$

Лемма 10. *Произведение матрицы отражения на вектор может быть вычислено за $2n + O(1)$ $(n \to \infty)$ сложений и столько же умножений (точнее,за $2n + 1$ умножение и $2n - 1$ сложение).*

Доказательство. Для матрицы отражения $U(x)$ и произвольного вектора y имеем

$$U(x)y = (I - 2xx^*)y = y - 2x(x^*y) = y - 2x(y, x).$$

На вычисление скалярного произведения (y, x) требуется n умножений и $n - 1$ сложение. На вычисление коэффициента $\alpha = 2(y, x)$ требуется еще одно умножение. На вычисление линейной комбинации $y - \alpha x$ требуется n умножений и столько же сложений. Складывая эти оценки, находим, что всего необходимо $n + 1 + n = 2n + 1$ умножение и $n - 1 + n = 2n - 1$ сложений.

Лемма 11. *Произведение матрицы отражения $U(x) \in \mathbf{M}_n$ на матрицу размера $n \times m$ может быть вычислено за $2nm + O(m)$ $(n, m \to \infty)$ сложений и столько же умножений (точнее,за $(2n + 1)m$ умножение и $(2n - 1)m$ сложение).*

Доказательство. Пусть $n \times m$ матрица $B = U(x)A$ есть произведение матрицы отражения $U(x) \in \mathbf{M}_n$ на $n \times m$ матрицу A. Запишем матрицы $A = (a_{ij})$ и $B = (b_{ij})$ через их столбцы: $A = [a^{(1)}, \ldots, a^{(m)}]$, $B = [b^{(1)}, \ldots, b^{(m)}]$, где $a^{(k)} = (a_{1k}, \ldots, a_{nk})^t, b^{(k)} = (b_{1k}, \ldots, b_{nk})^t, k = 1, \ldots, m$. Согласно определению произведения матриц $B = U(x)A = [U(x)a^{(1)}, \ldots, U(x)a^{(n)}]$, т.е. $b^{(k)} = U(x)a^{(k)}$, $k = 1, \ldots, m$. Таким образом, для вычисления матрицы $B = U(x)A$ надо вычислить m произведений $U(x)a^{(k)}$ матрицы $U(x)$ на вектора $a^{(k)}$, $k = 1, \ldots, m$. Доказываемое утверждение теперь вытекает из леммы 10.

§ 13.2. Алгоритм метода отражений

Пусть требуется решить линейную систему $A\,x = b$, $A \in \mathbf{M}_n$ вида (4.1).

Обозначим $a_1 = (a_{11}, \ldots, a_{n1})^t$ – первый столбец матрицы A. Согласно лемме 9 существует вектор $x^{(1)} \in \mathbf{C}^n$, равный

$$x^{(1)} = \pm \frac{a_1 - \|a_1\| e_1}{\|a_1 - \|a_1\| e_1\|},$$

такой, что $U(x^{(1)})a_1 = \|a_1\| e_1$, где $e_1 = (1, 0, \ldots, 0) \in \mathbf{C}^n$, $U_1 = U(x^{(1)})$ – матрица отражения. Умножим систему $A\,x = b$ на $U(x^{(1)})$ слева, получим

$$A^{(1)} x = b^{(1)},$$

где

$$A^{(1)} = U(x^{(1)})A = \begin{pmatrix} \|a_1\| & c_{12} & \ldots & c_{1n} \\ 0 & a_{22}^{(1)} & \ldots & a_{2n}^{(1)} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & a_{n2}^{(1)} & \ldots & a_{nn}^{(1)} \end{pmatrix}, \quad b^{(1)} = U(x^{(1)})b.$$

Далее процесс применяется к подматрице $(a_{ij}^{(1)})_{i,j=2,\ldots,n}$.

Пусть сделаны $k-1$, $k = 1, \ldots, n$ шагов этого процесса, т.е. система преобразована к виду

$$A^{(k-1)} x = b^{(k-1)}, \tag{1}$$

где

$$A^{(k-1)} = \prod_{i=k-1}^{1} U_i\, A, \quad b^{(k-1)} = \prod_{i=k-1}^{1} U_i\, b, \tag{2}$$

$$A^{(k-1)} = \begin{pmatrix} \|a_1\| & c_{12} & c_{13} & \ldots & c_{1,k-1} & c_{1k} & \ldots & c_{1n} \\ & \|a_1^{(1)}\| & c_{23} & \ldots & c_{2,k-1} & c_{2k} & \ldots & c_{2n} \\ & & \|a_1^{(2)}\| & \ldots & c_{3,k-1} & c_{3k} & \ldots & c_{3n} \\ & & & \ddots & \vdots & \vdots & \ddots & \vdots \\ & & & & \|a_1^{(k-2)}\| & c_{k-1,k} & \ldots & c_{k-1,n} \\ & & & & & a_{kk}^{(k-1)} & \ldots & a_{kn}^{(k-1)} \\ & & & & & \vdots & \ddots & \vdots \\ & & & & & a_{nk}^{(k-1)} & \ldots & a_{nn}^{(k-1)} \end{pmatrix}, \tag{3}$$

$$U_i = \begin{pmatrix} I_{i-1} & 0 \\ 0 & U(x^{(i)}) \end{pmatrix},$$

здесь $I_{i-1} \in \mathbf{M}_{i-1}$ – единичная матрица размера $(i-1)\times(i-1)$, $U(x^{(i)}) \in \mathbf{M}_{n-i+1}$ – матрица отражения размера $(n-i+1)\times(n-i+1)$, построенная по вектору

$$x^{(i)} = \pm \frac{a_1^{(i-1)} - \|a_1^{(i-1)}\| e_1^{(n-i+1)}}{\|a_1^{(i-1)} - \|a_1^{(i-1)}\| e_1^{(n-i+1)}\|} \in \mathbf{C}^{n-i+1},$$

где $e_1^{(m)} = (1, 0, \ldots, 0) \in \mathbf{C}^m$.

Обозначим

$$a_1^{(k-1)} = (a_{kk}^{(k-1)}, \ldots, a_{nk}^{(k-1)})^t \in \mathbf{C}^{n-k+1} \tag{4}$$

– первый столбец подматрицы $(a_{ij}^{(k-1)})_{i,j=k,\ldots,n}$. Согласно лемме 9 существует матрица отражения

$$U(x^{(k)}) = I - 2x^{(k)}(x^{(k)})^*, \quad x^{(k)} = \pm\frac{a_1^{(k-1)} - \|a_1^{(k-1)}\|e_1^{(n-k+1)}}{\|a_1^{(k-1)} - \|a_1^{(k-1)}\|e_1^{(n-k+1)}\|} \in \mathbf{C}^{n-k+1}, \tag{5}$$

такая, что

$$U(x^{(k)})a_1^{(k-1)} = \|a_1^{(k-1)}\|e_1^{(n-k+1)}. \tag{6}$$

Положим

$$U_k = \begin{pmatrix} I_{k-1} & 0 \\ 0 & U(x_k) \end{pmatrix}. \tag{7}$$

Покажем, что матрица U_k является унитарной, т.е. $U_k^* = U_k^{-1}$. По правилам перемножения блочных матриц

$$U_k^* = \begin{pmatrix} I_{k-1}^* & 0 \\ 0 & U^*(x_k) \end{pmatrix} = \begin{pmatrix} I_{k-1} & 0 \\ 0 & U(x_k) \end{pmatrix} = U_k, \tag{8}$$

$$U_k U_k^* = U_k U_k = \begin{pmatrix} I_{k-1}^2 & 0 \\ 0 & U^2(x_k) \end{pmatrix} = \begin{pmatrix} I_{k-1} & 0 \\ 0 & I_{n-k+1} \end{pmatrix} = I_n = I, \tag{9}$$

что и означает $U_k^* = U_k^{-1}$, т.е. унитарность матрицы U_k.

Умножим систему (1) на U_k слева, получим

$$A^{(k)}x = b^{(k)},$$

где

$$A^{(k)} = U_k A^{(k-1)} = \prod_{i=k}^{1} U_i\, A, \quad b^{(k)} = U_k b^{(k-1)} = \prod_{i=k}^{1} U_i\, b, \tag{10}$$

$$A^{(k)} = \begin{pmatrix} \|a_1\| & c_{12} & c_{13} & \ldots & c_{1,k-1} & c_{1k} & c_{1,k+1} & \ldots & c_{1n} \\ & \|a_1^{(1)}\| & c_{23} & \ldots & c_{2,k-1} & c_{2k} & c_{2,k+1} & \ldots & c_{2n} \\ & & \|a_1^{(2)}\| & \ldots & c_{3,k-1} & c_{3k} & c_{3,k+1} & \ldots & c_{3n} \\ & & & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ & & & & \|a_1^{(k-2)}\| & c_{k-1,k} & c_{k-1,k+1} & \ldots & c_{k-1,n} \\ & & & & & \|a_1^{(k-1)}\| & c_{k,k+1} & \ldots & c_{k,n} \\ & & & & & & a_{k+1,k+1}^{(k)} & \ldots & a_{k+1,n}^{(k)} \\ & & & & & & \vdots & \ddots & \vdots \\ & & & & & & a_{n,k+1}^{(k)} & \ldots & a_{nn}^{(k)} \end{pmatrix}. \tag{11}$$

Отметим, что при умножении матрицы U_k вида (5) на матрицу $A^{(k-1)}$ вида (3) она умножается только на подматрицу $(a_{ij}^{(k-1)})_{i,j=k,\dots,n}$ матрицы $A^{(k-1)}$ размера $n-k+1$ (остальная часть $A^{(k-1)}$ в преобразовании (10) не участвует).

Вычисления по формулам (5) осуществляются следующим образом: вначале вычисляются числа

$$s_k = \sum_{j=k+1}^{n} |a_{jk}^{(k-1)}|^2, \tag{12}$$

$$\|a_1^{(k-1)}\| = \sqrt{|a_{kk}^{(k-1)}|^2 + s_k}. \tag{13}$$

затем – вектор

$$x^{(k)} = (a_{kk}^{(k-1)} - \|a_1^{(k-1)}\|, a_{k+1,k}^{(k-1)}, \dots, a_{nk}^{(k-1)})^t \in \mathbf{C}^{n-k+1} \tag{14}$$

и его норма

$$\|x^{(k)}\| = \sqrt{|x_1^{(k)}|^2 + s_k}. \tag{15}$$

Теперь можно вычислить искомый вектор $x^{(k)}$:

$$x^{(k)} := x^{(k)}/\|x^{(k)}\|, \quad \text{т.е.} \quad x_j^{(k)} := x_j^{(k)}/\|x^{(k)}\|,\ j = 1, \dots, n-k+1. \tag{16}$$

После n шагов этого процесса (т.е. перехода от матриц и правых частей (2), (3) к (10), (11)) система примет вид

$$R\,x = y, \tag{17}$$

где

$$R = A^{(n)} = \prod_{i=n}^{1} U_i\,A, \quad y = b^{(n)} = \prod_{i=n}^{1} U_i\,b, \tag{18}$$

$$R = \begin{pmatrix} \|a_1\| & c_{12} & c_{13} & \dots & c_{1,n-2} & c_{1,n-1} & c_{1n} \\ & \|a_1^{(1)}\| & c_{23} & \dots & c_{2,n-2} & c_{2,n-1} & c_{2n} \\ & & \|a_1^{(2)}\| & \dots & c_{3,n-2} & c_{3,n-1} & c_{3n} \\ & & & \ddots & \vdots & \vdots & \vdots \\ & & & & \|a_1^{(n-3)}\| & c_{n-2,n-1} & c_{n-2,n} \\ & & & & & \|a_1^{(n-2)}\| & c_{n-1,n} \\ & & & & & & \|a_1^{(n-1)}\| \end{pmatrix} \tag{19}$$

(напомним, определения векторов $a_1^{(k-1)}$, $k = 1, \dots, n$ даются в (4), где считаем, что $a_1^{(0)} = a_1$).

Система (17) с верхней треугольной матрицей R решается обратным ходом метода Гаусса.

§ 13.3. Оценка количества арифметических операций в методе отражений

Оценим трудоемкость k-го шага алгоритма, а затем просуммируем полученные оценки по всем $k = 1, \ldots, n$.

1. На вычисление матрицы $U(x_k)$ по формулам (5) требуется

а) $n-k$ умножений и $n-k-1$ сложений для вычисления s_k в (12);

б) одно умножение, одно сложение и одна операция извлечения корня для вычисления $\|a_1^{(k-1)}\|$ в (13);

в) одно вычитание для построения вектора $x^{(k)}$ в (14);

г) одно умножение, одно сложение и одна операция извлечения корня для вычисления $\|x^{(k)}\|$ в (15);

д) $n-k+1$ делений для построения вектора $x^{(k)}$ в (16).

Всего для построения матрицы $U(x_k)$ требуется $(n-k)+1+1+(n-k+1) = 2(n-k)+3$ мультипликативных, $(n-k-1)+1+1+1 = n-k+2$ аддитивных операций и $1+1=2$ операции извлечения корня.

2. Компоненты $k, \ldots, n$ k-го столбца матрицы $A^{(k)}$, равные компонентам вектора $\|a_1^{(k-1)}\| e_1^{(n-k+1)}$, уже вычислены в (13). Столбец k вычисляется не по общим формулам (10) для сокращения количества арифметических операций и уменьшения вычислительной погрешности.

3. Поскольку в формуле (10) матрица U_k вида (5) умножается на матрицу $A^{(k-1)}$ вида (3), то при вычислениях по (10) надо умножить матрицу отражения $U(x^{(k)}) \in \mathbf{M}_{n-k+1}$ на подматрицу $(a_{ij}^{(k-1)})_{i=k,\ldots,n,\, j=k+1,\ldots,n}$ матрицы $A^{(k-1)}$ размера $(n-k+1)\times(n-k)$ (k-й столбец матрицы $A^{(k)}$ уже вычислен в пункте 2). Согласно лемме 11 на это требуется $2(n-k)(n-k+1)+O(n-k) = 2(n-k)^2+O(n-k)$ $(n\to\infty)$ умножений и столько же сложений.

4. На вычисление новой правой части по формуле (10) согласно лемме 10 требуется $2(n-k+1)+O(1)$ $(n\to\infty)$ умножений и столько же сложений.

Итак, на k-ом шаге алгоритма требуется выполнить $2(n-k)+3+2(n-k)^2+O(n-k)+2(n-k+1)+O(1) = 2(n-k)^2+O(n-k)$ $(n\to\infty)$ мультипликативных операций, $n-k+2+2(n-k)^2+O(n-k)+2(n-k+1)+O(1) = 2(n-k)^2+O(n-k)$ $(n\to\infty)$ аддитивных операций и 2 операции извлечения корня.

Следовательно, всего для проведения алгоритма требуется выполнить

$$\sum_{k=1}^{n}(2(n-k)^2+O(n-k)) = 2n(n-1)(2n-1)/6+O(n^2) = \frac{2}{3}n^3+O(n^2)\ (n\to\infty)$$

мультипликативных и столько же аддитивных операций, и $\sum_{k=1}^{n}(2) = 2n$ операций извлечения корня (которые по трудоемкости по порядку можно сравнить с операциями деления).

На решение системы (17) с верхней треугольной матрицей R обратным ходом метода Гаусса требуется $O(n^2)$ $(n\to\infty)$ арифметических операций.

Таким образом, на решение линейной системы методом отражений требуется $\frac{2}{3}n^3+O(n^2)$ $(n\to\infty)$ мультипликативных и столько же аддитивных операций (что в 2 раза больше, чем в методе Гаусса).

Теорема 1 (О QR-разложении). *Всякая невырожденная матрица $A \in \mathbf{M}_n$ может быть представлена в виде $A = Q\,R$, где матрица Q – унитарная, а матрица R – верхняя треугольная с вещественными положительными элементами на главной диагонали. Это разложение единственно.*

Доказательство проходит аналогично доказательству теоремы 12.1. Проведем для матрицы A алгоритм метода отражений, осуществимый для всякой невырожденной матрицы. Обозначим в (18) $\hat{Q} = \prod\limits_{i=n}^{1} U_i$. Как произведение унитарных матриц, матрица $\hat{Q}$ унитарна. Тогда (5) имеет вид $R = \hat{Q}A$, откуда $A = (\hat{Q})^{-1}R = QR$, где $Q = (\hat{Q})^t = (\hat{Q})^{-1}$. Здесь матрица Q унитарна, а матрица R имеет вид (19) и потому удовлетворяет условиям теоремы.

Предположим, что возможно два различных разложения $A = QR$ и $A = Q'R'$, удовлетворяющих условиям теоремы. Тогда $QR = Q'R'$ и $(Q')^{-1}Q = R^{-1}R'$. В левой части последнего равенства стоит унитарная матрица, а в правой – верхняя треугольная. Пересечение группы унитарных матриц и группы верхних треугольных матриц состоит из матриц вида $D = \mathrm{diag}(d_1, \ldots, d_n)$, где $d_j = e^{i\varphi_j}$, $j = 1, \ldots, n$ (проверить самостоятельно). Поскольку диагональные элементы матрицы $R^{-1}R'$ равны произведениям диагональных элементов матриц R^{-1} и R', то они вещественны и положительны. Следовательно $R^{-1}R' = I$, т.е. $R = R'$. Полученное противоречие доказывает теорему.

Замечание 1. Справедливы замечания 12.3 и 12.4 о применении QR-разложения.

§ 13.4. Построение QR-разложения методом отражений

Построение QR-разложения. Хранение матриц Q и R в памяти. Пусть стоит задача построить QR-разложение для матрицы A. Будем действовать как в теореме 1. Проведем для матрицы A метод отражений и получим в результате матрицу R из (19). При этом матрица Q равна (см. доказательство теоремы 1)

$$Q = (\prod_{i=n}^{1} U_i)^t = \prod_{i=1}^{n} U_i^t = \prod_{i=1}^{n} \bar{U}_i. \tag{20}$$

Возможны два способа хранения матриц Q и R в памяти.

1. Матрица R хранится на месте верхнего треугольника матрицы A и получается из нее последовательным применением матриц отражения (см. выше алгоритм метода отражений). Для хранения матрицы Q выделяется отдельная матрица Q, которая равна единичной перед первым шагом алгоритма. На шаге k, $k = 1, \ldots, n$ эта матрица умножается справа на матрицу U_k :

$$Q := Q\,U_k$$

(см. (10), (20)). Матрица U_k вида (5) умножается по алгоритму из леммы 11 на матрицу Q произвольного вида за $2n(n-k+1)+O(n) = 2n(n-k)+O(n)$ $(n \to \infty)$ умножений и такого же количества сложений (поскольку для вычисления произведения QU_k матрицы Q на матрицу U_k вида (5) надо вычислить произведение подматрицы $(q_{ij})_{i=1,\dots,n,\, j=k,\dots,n}$ размера $n \times (n-k+1)$ на матрицу отражения $U(x^{(k)}) \in \mathbf{M}_{n-k+1}$ размера $(n-k+1) \times (n-k+1)$).

Следовательно, произведение n матриц отражения в (20) может быть вычислено за $\sum_{k=1}^{n}(2n(n-k)+O(n)) = 2nn(n-1)/2 + O(n^2) = n^3 + O(n^2)$ $(n \to \infty)$ умножений и столько же сложений.

2. Как и в первом способе, матрица R хранится на месте верхнего треугольника матрицы A. Для хранения же матрицы Q отдельная память не выделяется. Заметим, что на шаге k, $k = 1, \dots, n$ мы использовали матрицу U_k, получающуюся в (7) из матрицы отражения $U(x^{(k)})$, которая в свою очередь целиком определяется вектором $x^{(k)} \in \mathbf{C}^{n-k+1}$ из (5). При этом после преобразования (10), т.е. перехода от матрицы (3) к матрице (11), в k-ом столбце матрицы $A^{(k)}$ образовались $n-k$ нулевых элементов $a_{jk}^{(k)} = 0$, $j = k+1, \dots, n$. Поэтому возможно вместо матрицы Q вида (20) хранить на месте нижнего треугольника матрицы A набор векторов $x^{(k)}$, $k = 1, \dots, n$, задающий матрицы отражения $U(x^{(k)})$. Формула (14) подсказывает удобный способ организации такого хранения: на шаге k $x_1^{(k)} \equiv a_{kk}^{(k-1)}, \dots, x_{n-k+1}^{(k)} \equiv a_{nk}^{(k-1)}$, а элемент $a_{kk}^{(k)} = \|a_1^{(k-1)}\|$ хранится в виде $(k-1)$-ой компоненты дополнительного вектора D. В итоге после n шагов процесса на месте исходной $n \times n$ матрицы A и дополнительного вектора D длины n будет находиться следующая информация: верхний треугольник матрицы R : $r_{ij} = a_{ij}$, $i < j$, $i = 1, \dots, n$, $j = 2, \dots, n$, диагональ матрицы R : $r_{ii} = d_i$, $i = 1, \dots, n$, набор векторов $x^{(k)}$, $k = 1, \dots, n$, $x_1^{(k)} \equiv a_{kk}, \dots, x_{n-k+1}^{(k)} \equiv a_{nk}$.

При втором способе хранения матрицы Q не только экономится n^3 ячеек памяти, но и экономится $n^3 + O(n^2)$ $(n \to \infty)$ умножений и такое же количество сложений на построение матрицы Q. Второму способу хранения благоприятствует также то обстоятельство, что редко требуется знать матрицу Q "саму по себе". Обычно требуется уметь вычислять ее произведения на вектор и матрицу. Для того, чтобы вычислить произведение матрицы Q вида (20) на некоторую матрицу B требуется вычислить $QB = \prod_{i=1}^{n}(U_i B)$. На это нужно $n^3 + O(n^2)$ $(n \to \infty)$ умножений и столько же сложений (см. подсчет количества операций при рассмотрении первого способа хранения, в котором фактически вычислялось произведение матрицы вида (20) и единичной матрицы). Это количество совпадает с количеством арифметических операций, необходимых для вычисления произведения двух матриц Q и B произвольного вида. В силу этого почти всегда используется второй способ хранения матрицы Q.

§ 13.5. Оценка количества арифметических операций в алгоритме построения QR-разложения методом отражений

Трудоемкость алгоритма построения QR-разложения складывается из количества арифметических операций, необходимых для проведения алгоритма метода отражений, и количества арифметических операций, необходимых для построения матрицы Q.

Если для Q используется второй способ хранения, то дополнительных действий для ее построения не требуется. Следовательно, в этом случае для построения QR-разложения надо выполнить $\frac{2}{3}n^3+O(n^2)$ $(n\to\infty)$ мультипликативных операций и такое же количество аддитивных операций.

Если для Q используется первый способ хранения, то как показано выше для ее построения дополнительно к $\frac{2}{3}n^3+O(n^2)$ $(n\to\infty)$ мультипликативным и $\frac{2}{3}n^3+O(n^2)$ $(n\to\infty)$ аддитивным операциям, необходимых для проведения алгоритма метода отражений, требуется $n^3+O(n^2)$ $(n\to\infty)$ умножений и $n^3+O(n^2)$ $(n\to\infty)$ сложений, всего $\frac{5}{3}n^3+O(n^2)$ $(n\to\infty)$ мультипликативных и столько же аддитивных операций.

Приведение матрицы к почти треугольному виду унитарным подобием

Определение. Матрица B называется *подобной* матрице A, если существует невырожденная матрица C такая, что $A=C\,B\,C^{-1}$.

В курсе алгебры доказывается, что подобные матрицы имеют один и тот же набор собственных значений.

Определение. Матрица B называется *унитарно подобной* матрице A, если матрица C в определении выше унитарная.

В силу свойства 6 числа обусловленности (см. §3) у унитарно подобных матриц числа обусловленности совпадают. Поэтому именно преобразование унитарного подобия будет вносить наименьшую вычислительную погрешность.

Отметим еще одно свойство унитарного подобия: если матрица A самосопряженная, то унитарно подобная ей матрица B также самосопряженная. Действительно, $B^*=(CAC^{-1})^*=(C^{-1})^*A^*C^*=CAC^{-1}=B$.

Рассмотренные выше алгоритмы решения линейных систем работали единообразным способом: они приводили исходную матрицу к более простому виду (треугольному) с помощью преобразований, сохраняющих решение системы; затем решение системы с более простой матрицей находилось в явном виде. Пусть стоит задача найти собственные значения матрицы. Попробуем действовать по той же схеме: приведем исходную матрицу к более простому виду с помощью преобразований подобия, сохраняющих собственные значения; затем для этой более простой матрицы тем или иным способом найдем ее собственные значения, которые совпадают с собственными значениями исходной матрицы. Для того, чтобы вносить меньшую вычислительную погрешность, будем использовать преобразования унитарного подобия.

Простейшее рассмотрение алгоритмов метода вращений и отражений показывает, что вид этой более простой матрицы не может быть треугольным. Действительно, пусть, например, в методе вращений при умножении на матрицу T_{12} слева элемент $(2,1)$ исходной матрицы становится равным нулю. Тогда при умножении на матрицу $T_{12}^{-1} = T_{12}^t$ справа этот элемент может измениться и перестать быть равным нулю.

Определение. Матрица $A = (a_{ij})$ называется *почти треугольной*, если $a_{ij} = 0$ при $i > j+1,\ j = 1,\dots,n-2,\ i = 3,\dots,n$.

Оказывается, всякую матрицу можно привести к почти треугольному виду с помощью унитарного подобия.

§ 14. ПРИВЕДЕНИЕ МАТРИЦЫ К ПОЧТИ ТРЕУГОЛЬНОМУ ВИДУ УНИТАРНЫМ ПОДОБИЕМ МЕТОДОМ ВРАЩЕНИЙ

Пусть требуется привести вещественную матрицу A к почти треугольному виду.

Всюду ниже мы будем часто пользоваться тем фактом, что при умножении матрицы A на матрицу элементарного вращения T_{ij} слева изменяются только строки i и j матрицы A, а при умножении на T_{ij} справа изменяются только столбцы i и j матрицы A.

§ 14.1. Случай произвольной матрицы

Обозначим $a_1 = (a_{21},\dots,a_{n1})^t$. Согласно лемме 12.3 существуют $n-2$ матриц $T_{2j} = T_{2j}(\varphi_{2j})$, $j = 3,\dots,n$, таких, что $T_{2n}\dots T_{24}T_{23}a_1 = \|a_1\|\,e_1^{(n-1)}$ (причем значения углов φ_{2j}, $j = 3,\dots,n$ определяются леммами 12.2, 12.3). Умножим матрицу A на $T_{2n}\dots T_{24}T_{23}$ слева, получим

$$\widehat{A}^{(1)} = T_{2n}\dots T_{24}T_{23}A = \begin{pmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ \|a_1\| & \hat{a}_{22}^{(1)} & \dots & \hat{a}_{2n}^{(1)} \\ 0 & \hat{a}_{32}^{(1)} & \dots & \hat{a}_{3n}^{(1)} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & \hat{a}_{n2}^{(1)} & \dots & \hat{a}_{nn}^{(1)} \end{pmatrix}. \qquad (1)$$

Умножим матрицу $\widehat{A}^{(1)}$ на $(T_{2n}\dots T_{24}T_{23})^* = T_{23}^tT_{24}^t\dots T_{2n}^t$ справа, получим (с учетом того, что при умножении справа на T_{2j}, $j = 3,\dots,n$ первый столбец

матрицы $\widehat{A}^{(1)}$ не изменяется)

$$A^{(1)} = \widehat{A}^{(1)} T_{23}^t T_{24}^t \dots T_{2n}^t = \begin{pmatrix} a_{11} & a_{12}^{(1)} & \dots & a_{1n}^{(1)} \\ \|a_1\| & a_{22}^{(1)} & \dots & a_{2n}^{(1)} \\ 0 & a_{32}^{(1)} & \dots & a_{3n}^{(1)} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & a_{n2}^{(1)} & \dots & a_{nn}^{(1)} \end{pmatrix}. \tag{2}$$

Пусть сделаны $k-1$, $k = 1, \dots, n-1$ шагов этого процесса, т.е. матрица преобразована к виду

$$A^{(k-1)} = \prod_{i=k}^{2} \prod_{j=n}^{i+1} T_{ij} A \prod_{i=2}^{k} \prod_{j=i+1}^{n} T_{ij}^t, \tag{3}$$

$$A^{(k-1)} = \begin{pmatrix} a_{11} & c_{12} & c_{13} & \dots & c_{1,k-1} & a_{1k}^{(k-1)} & \dots & a_{1n}^{(k-1)} \\ \|a_1\| & a_{22}^{(1)} & c_{23} & \dots & c_{2,k-1} & a_{2k}^{(k-1)} & \dots & a_{2n}^{(k-1)} \\ & \|a_1^{(1)}\| & a_{33}^{(2)} & \dots & c_{3,k-1} & a_{3k}^{(k-1)} & \dots & a_{3n}^{(k-1)} \\ & & \|a_1^{(2)}\| & \ddots & \vdots & \vdots & \ddots & \vdots \\ & & & \ddots & a_{k-1,k-1}^{(k-2)} & a_{k-1,k}^{(k-1)} & \dots & a_{k-1,n}^{(k-1)} \\ & & & & \|a_1^{(k-2)}\| & a_{kk}^{(k-1)} & \dots & a_{kn}^{(k-1)} \\ & & & & & a_{k+1,k}^{(k-1)} & \dots & a_{k+1,n}^{(k-1)} \\ & & & & & \vdots & \ddots & \vdots \\ & & & & & a_{nk}^{(k-1)} & \dots & a_{nn}^{(k-1)} \end{pmatrix}. \tag{4}$$

Обозначим

$$a_1^{(k-1)} = (a_{kk}^{(k-1)}, \dots, a_{nk}^{(k-1)})^t \in \mathbf{R}^{n-k} \tag{5}$$

– часть первого столбца подматрицы $(a_{ij}^{(k-1)})_{i,j=k,\dots,n}$. Согласно лемме 12.3 существуют $n-k-1$ матриц $T_{k+1,j} = T_{k+1,j2}(\varphi_{k+1,j})$, $j = k+2, \dots, n$ таких, что

$$T_{k+1,n} \dots T_{k+1,k+3} T_{k+1,k+2} a_1^{(k-1)} = \|a_1^{(k-1)}\| \, e_1^{(n-k)}, \tag{6}$$

(значения углов $\varphi_{k+1,j}$, $j = k+2, \dots, n$ определяются леммами 12.2, 12.3). Умножим матрицу (3) на $T_{k+1,n} \dots T_{k+1,k+3} T_{k+1,k+2}$ слева, получим

$$\widehat{A}^{(k)} = \prod_{j=n}^{k+2} T_{k+1,j} A^{(k-1)}, \tag{7}$$

где

$$\widehat{A}^{(k)} = \begin{pmatrix} a_{11} & c_{12} & c_{13} & \dots & c_{1,k-1} & c_{1k} & a^{(k-1)}_{1,k+1} & \dots & a^{(k-1)}_{1n} \\ \|a_1\| & a^{(1)}_{22} & c_{23} & \dots & c_{2,k-1} & c_{2k} & a^{(k-1)}_{2,k+1} & \dots & a^{(k-1)}_{2n} \\ & \|a^{(1)}_1\| & a^{(2)}_{33} & \dots & c_{3,k-1} & c_{3k} & a^{(k-1)}_{3,k+1} & \dots & a^{(k-1)}_{3n} \\ & & \|a^{(2)}_1\| & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ & & & \ddots & a^{(k-2)}_{k-1,k-1} & c_{k-1,k} & a^{(k-1)}_{k-1,k+1} & \dots & a^{(k-1)}_{k-1,n} \\ & & & & \|a^{(k-2)}_1\| & a^{(k-1)}_{kk} & a^{(k-1)}_{k,k+1} & \dots & a^{(k-1)}_{kn} \\ & & & & & \|a^{(k-1)}_1\| & \widehat{a}^{(k)}_{k+1,k+1} & \dots & \widehat{a}^{(k)}_{k+1,n} \\ & & & & & & \widehat{a}^{(k)}_{k+2,k+1} & \dots & \widehat{a}^{(k)}_{k+2,n} \\ & & & & & & \vdots & \ddots & \vdots \\ & & & & & & \widehat{a}^{(k)}_{n,k+1} & \dots & \widehat{a}^{(k-1)}_{nn} \end{pmatrix}. \tag{8}$$

Отметим, что в (7) каждая из $n-k-1$ матриц элементарных вращений $T_{k+1,j}$ такова, что $j > k+1$ и потому в (7) она умножается только на подматрицу $(a^{(k-1)}_{ij})_{i=k+1,\dots,n,\, j=k,\dots,n}$ матрицы $A^{(k-1)}$ размера $(n-k)\times(n-k+1)$ (остальная часть $A^{(k-1)}$ в преобразовании (7) не участвует).

Умножим матрицу $\widehat{A}^{(k)}$ на $(T_{k+1,n}\dots T_{k+1,k+2})^* = T^*_{k+1,k+2}\dots T^*_{k+1,n}$ справа, получим из (8) (с учетом того, что при умножении справа на $T_{k+1,j}$, $j = k+2,\dots,n$ столбцы $1,\dots,k$ матрицы $\widehat{A}^{(k)}$ не изменяются)

$$A^{(k)} = \widehat{A}^{(k)} \prod_{j=k+2}^{n} T^t_{k+1,j} = \prod_{j=n}^{k+2} T_{k+1,j} A^{(k-1)} \prod_{j=k+2}^{n} T^t_{k+1,j}, \tag{9}$$

$$A^{(k)} = \begin{pmatrix} a_{11} & c_{12} & c_{13} & \dots & c_{1,k-1} & c_{1k} & a^{(k)}_{1,k+1} & \dots & a^{(k)}_{1n} \\ \|a_1\| & a^{(1)}_{22} & c_{23} & \dots & c_{2,k-1} & c_{2k} & a^{(k)}_{2,k+1} & \dots & a^{(k)}_{2n} \\ & \|a^{(1)}_1\| & a^{(2)}_{33} & \dots & c_{3,k-1} & c_{3k} & a^{(k)}_{3,k+1} & \dots & a^{(k)}_{3n} \\ & & \|a^{(2)}_1\| & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ & & & \ddots & a^{(k-2)}_{k-1,k-1} & c_{k-1,k} & a^{(k)}_{k-1,k+1} & \dots & a^{(k)}_{k-1,n} \\ & & & & \|a^{(k-2)}_1\| & a^{(k-1)}_{kk} & a^{(k)}_{k,k+1} & \dots & a^{(k)}_{kn} \\ & & & & & \|a^{(k-1)}_1\| & a^{(k)}_{k+1,k+1} & \dots & a^{(k)}_{k+1,n} \\ & & & & & & a^{(k)}_{k+2,k+1} & \dots & a^{(k)}_{k+2,n} \\ & & & & & & \vdots & \ddots & \vdots \\ & & & & & & a^{(k)}_{n,k+1} & \dots & a^{(k-1)}_{nn} \end{pmatrix}. \tag{10}$$

Отметим, что в (9) каждая из $n-k-1$ матриц элементарных вращений $T^t_{k+1,j}(\varphi_{k+1,j}) = T_{k+1,j}(-\varphi_{k+1,j})$ такова, что $j > k+1$ и потому в (9) она умножается только на подматрицу $(\widehat{a}^{(k-1)}_{ij})_{i=1,\dots,n,\, j=k+1,\dots,n}$ матрицы $\widehat{A}^{(k-1)}$ размера $n\times(n-k)$ (остальная часть $\widehat{A}^{(k-1)}$ в преобразовании (9) не участвует).

После $n-2$ шагов этого процесса (т.е. перехода от матриц (3), (4) к (9), (10)) матрица примет требуемый почти треугольный вид

$$R = A^{(n-2)} = \prod_{i=n-1}^{2} \prod_{j=n}^{i+1} T_{ij} A \prod_{i=2}^{n-1} \prod_{j=i+1}^{n} T_{ij}^t, \tag{11}$$

$$R = \begin{pmatrix} a_{11} & c_{12} & c_{13} & \dots & c_{1,n-2} & a_{1,n-1}^{(n-2)} & a_{1n}^{(n-2)} \\ \|a_1\| & a_{22}^{(1)} & c_{23} & \dots & c_{2,n-2} & a_{2,n-1}^{(n-2)} & a_{2n}^{(n-2)} \\ & \|a_1^{(1)}\| & a_{33}^{(2)} & \dots & c_{3,n-2} & a_{3,n-1}^{(n-2)} & a_{3n}^{(n-2)} \\ & & \|a_1^{(2)}\| & \ddots & \vdots & \vdots & \vdots \\ & & & \ddots & a_{n-2,n-2}^{(n-3)} & a_{n-2,n-1}^{(n-2)} & a_{n-2,n}^{(n-2)} \\ & & & & \|a_1^{(n-3)}\| & a_{n-1,n-1}^{(n-2)} & a_{n-1,n}^{(n-2)} \\ & & & & & a_{n,n-1}^{(n-2)} & a_{nn}^{(n-2)} \end{pmatrix} \tag{12}$$

(напомним, определения векторов $a_1^{(k-1)}$, $k = 1, \dots, n-2$ даются в (5), где считаем, что $a_1^{(0)} = a_1$).

Оценка количества арифметических операций в алгоритме приведения матрицы к почти треугольному виду унитарным подобием методом вращений

Оценим трудоемкость k-го шага алгоритма, а затем просуммируем полученные оценки по всем $k = 1, \dots, n-2$.

1. На вычисление $n-k-1$ матриц $T_{k,k+1}, \dots, T_{kn}$, участвующих в (6), согласно лемме 12.2 требуется $4(n-k-1)$ мультипликативных, $2(n-k-1)$ аддитивных и $n-k$ операций извлечения корня.

2. На вычисление компонент $k+1, \dots, n$ k-го столбца матрицы $\widehat{A}^{(k)}$, равных компонентам вектора $\|a_1^{(k-1)}\| e_1^{(n-k)}$ требуется (для вычисления длины вектора (5)) $n-k$ операций умножения, $n-k-1$ операций сложения и одна операция извлечения корня. Столбец k вычисляется именно этим способом (а не по общим формулам (7)) для сокращения количества арифметических операций и уменьшения вычислительной погрешности.

3. Поскольку в формуле (7) каждая из $n-k-1$ матриц элементарных вращений умножается на подматрицу $(a_{ij}^{(k-1)})_{i=k+1,\dots,n,\, j=k+1,\dots,n}$ матрицы $A^{(k-1)}$ размера $(n-k)\times(n-k)$ (k-й столбец матрицы $\widehat{A}^{(k)}$ уже вычислен в пункте 2), то согласно лемме 12.5 на это требуется $(n-k-1)4(n-k) = 4(n-k)^2 - 4(n-k)$ умножений и $(n-k-1)2(n-k) = 2(n-k)^2 - 2(n-k)$ сложений.

4. Поскольку матрица, транспонированная к матрице элементарного вращения, опять является матрицей элементарного вращения:

$$T_{k+1,j}^t(\varphi_{k+1,j}) = T_{k+1,j}(-\varphi_{k+1,j})$$

и в формуле (9) каждая из этих $n-k-1$ матриц элементарных вращений умножается только на подматрицу $(\widehat{a}_{ij}^{(k-1)})_{i=1,\dots,n,\, j=k+1,\dots,n}$ матрицы $\widehat{A}^{(k-1)}$ размера

$n\times(n-k)$, то согласно лемме 12.5 на это требуется $(n-k-1)4n = 4n(n-k)-4n$ умножений и $(n-k-1)2n = 2n(n-k)-2n$ сложений.

Итак, на k-ом шаге алгоритма требуется выполнить $4(n-k-1)+(n-k)+4(n-k)^2-4(n-k)+4n(n-k)-4n = 4n(n-k)+4(n-k)^2+(n-k)-4n-4$ мультипликативных операций, $2(n-k-1)+(n-k-1)+2(n-k)^2-2(n-k)+2n(n-k)-2n = 2n(n-k)+2(n-k)^2+(n-k)-2n-3$ аддитивных операций и $n-k$ операций извлечения корня.

Следовательно, всего для проведения алгоритма требуется выполнить

$$\sum_{k=1}^{n-2}(4n(n-k)+4(n-k)^2+(n-k)-4n-4) = 4n((n-1)(n-2)/2-1)$$
$$+4((n-1)(n-2)(2n-3)/6-1)+(n-1)(n-2)/2-1-4(n-2)(n+1)$$
$$= 2n^3+O(n^2)+\tfrac{4}{3}n^3+O(n^2)+O(n^2) = \tfrac{10}{3}n^3+O(n^2)\ (n\to\infty)$$

мультипликативных операций, $\sum_{k=1}^{n-2}(2n(n-k)+2(n-k)^2+(n-k)-2n-3) = \frac{5}{3}n^3+O(n^2)$ $(n\to\infty)$ аддитивных операций и $\sum_{k=1}^{n-2}(n-k) = O(n^2)$ $(n\to\infty)$ операций извлечения корня (которые по трудоемкости по порядку можно сравнить с операциями деления).

Таким образом, на приведение матрицы к почти треугольному виду унитарным подобием методом вращений требуется $\frac{10}{3}n^3+O(n^2)$ $(n\to\infty)$ мультипликативных операций и $\frac{5}{3}n^3+O(n^2)$ $(n\to\infty)$ аддитивных операций. Заметим, что это количество операций в два с половиной раза больше, чем нужно для решения линейной системы методом вращений.

Теорема 1. *Всякая невырожденная вещественная матрица A может быть представлена в виде $A = Q\,R\,Q^t$, где матрица Q – ортогональная, а матрица R – верхняя почти треугольная.*

Доказательство. Проведем для матрицы A изложенный выше алгоритм, осуществимый для всякой невырожденной матрицы. Обозначим в (11) $\hat{Q} = \prod\limits_{i=n-1}^{2}\prod\limits_{j=n}^{i+1} T_{ij}$. Как произведение ортогональных матриц, матрица $\hat{Q}$ ортогональна. Тогда (11) имеет вид $R = \hat{Q}A\hat{Q}^t$, откуда $A = (\hat{Q})^{-1}R(\hat{Q}^t)^{-1} = QRQ^t$, где $Q = (\hat{Q})^t = (\hat{Q})^{-1}$ – ортогональная матрица. Матрица R, имеющая вид (12), удовлетворяет условиям теоремы.

Замечание 1. Как отмечалось выше, построенное в теореме 1 разложение используется в ряде алгоритмов нахождения собственных значений матрицы.

Хранение матриц Q и R в памяти осуществляется одним из способов, изложенных при обсуждении алгоритма построения QR-разложения для матрицы A методом вращений.

Трудоемкость алгоритма построения описанного выше разложения складывается из количества арифметических операций, необходимых для проведения самого алгоритма, и количества арифметических операций, необходимых

для построения матрицы Q. Подробные выкладки были проведены при обсуждении алгоритма построения QR-разложения методом вращений.

§ 14.2. Случай симметричной матрицы

Рассмотрим ситуацию, когда описанный выше метод приведения к почти треугольному виду применяется к симметричной матрице $A \in \mathbf{M}_n$. Согласно (1), (2) $A^{(1)} = Q_1 A Q_1^t$, где $Q_1 = T_{2n} \dots T_{24} T_{23}$ – ортогональная матрица, т.е. $A^{(1)}$ и A – унитарно подобны. Следовательно, $A^{(1)}$ – симметричная матрица. Согласно (7), (9) на k-ом ($k = 1, \dots, n-2$) шаге алгоритма $A^{(k)} = Q_k A^{(k-1)} Q_k^t$, где $Q_k = T_{k+1,n} \dots T_{k+1,k+3} T_{k+1,k+2}$ – ортогональная матрица. Следовательно, $A^{(k)}$ и A унитарно подобны, и $A^{(k)}$ – симметричная матрица для всякого $k = 1, \dots, n-2$. Таким образом, $R = A^{(n-2)}$ – почти треугольная и симметричная, т.е. трехдиагональная матрица.

Запишем описанный выше процесс приведения симметричной матрицы к трехдиагональному виду так, чтобы максимально уменьшить объем вычислительной работы за счет использования симметрии.

Лемма 1. *Для всякой матрицы элементарного вращения $T_{ij} \in \mathbf{M}_n$ и всякой симметричной матрицы $A \in \mathbf{M}_n$ матрица $B = T_{ij} A T_{ij}^t$ может быть вычислена за $4n + 8$ умножений и $2n + 4$ сложений.*

Доказательство. Матрица $\widehat{B} = T_{ij} A$ согласно лемме 12.5 вычисляется за $4n$ умножений и $2n$ сложений. Матрица $B = (b_{kl}) = T_{ij} A T_{ij}^t$ унитарно подобна A и потому симметрична:

$$b_{kl} = b_{lk}, \quad k, l = 1, \dots, n.$$

С другой стороны, матрица $B = \widehat{B} T_{ij}^t$ получается из матрицы $\widehat{B} = (\hat{b}_{kl})$ изменением элементов, расположенных только в i-ом и j-ом столбцах:

$$b_{kl} = \hat{b}_{kl}, \quad l \neq i, j, \quad k, l = 1, \dots, n.$$

Из последних двух равенств получаем:

$$b_{kl} = \hat{b}_{kl}, \quad (k,l) \neq (i,i),\ (i,j),\ (j,i)\ (j,j) \quad k, l = 1, \dots, n,$$

т.е. у матриц B и $\widehat{B}$ отличаются только 4 элемента с индексами (i,i), (i,j), (j,i), (j,j). Эти элементы получаются умножением i-й и j-й строки матрицы $\widehat{B}$ на матрицу T_{ij} справа и вычисляются по формулам

$$\begin{aligned} b_{ii} &= \hat{b}_{ii} \cos\varphi_{ij} + \hat{b}_{ij} \sin\varphi_{ij}, \quad & b_{ij} &= \hat{b}_{ii} \sin\varphi_{ij} - \hat{b}_{ij} \cos\varphi_{ij}, \\ b_{ji} &= \hat{b}_{ji} \cos\varphi_{ij} + \hat{b}_{jj} \sin\varphi_{ij}, \quad & b_{jj} &= \hat{b}_{jj} \sin\varphi_{ij} - \hat{b}_{jj} \cos\varphi_{ij}. \end{aligned}$$

Эти вычисления требуют дополнительно 8 умножений и 4 сложения. Складывая это с трудоемкостью построения матрицы $\widehat{B}$, получаем требуемую оценку.

Замечание 2. Для несимметричной матрицы A вычисление матрицы $B = T_{ij}AT_{ij}^t$ требует $8n$ умножений и $4n$ сложений (см. лемму 12.5).

Обозначим $a_1 = (a_{21}, \dots, a_{n1})^t$. Согласно лемме 12.3 существуют $n-2$ матриц $T_{2j} = T_{2j}(\varphi_{2j})$, $j = 3, \dots, n$, таких, что $T_{2n} \dots T_{24}T_{23}a_1 = \|a_1\| e_1^{(n-1)}$ (причем значения углов φ_{2j}, $j = 3, \dots, n$ определяются леммами 12.2, 12.3). Обозначим $A_3^{(1)} = T_{23}AT_{23}^t$, $A_4^{(1)} = T_{24}A_3^{(1)}T_{24}^t$, $\dots$, $A_n^{(1)} = T_{2n}A_{n-1}^{(1)}T_{2n}^t$. Все матрицы $A_j^{(1)}$, $j = 3, 4, \dots, n$ унитарно подобны A и потому симметричны. Согласно (1), (2) $A^{(1)} = A_n^{(1)}$. В силу симметрии матрицы $A^{(1)}$ вместо (2) для нее справедливо более точное равенство

$$A^{(1)} = T_{2n}(\dots(T_{24}(T_{23}AT_{23}^t)T_{24}^t)\dots)T_{2n}^t = \begin{pmatrix} a_{11} & \|a_1\| & 0 & \dots & 0 \\ \|a_1\| & a_{22}^{(1)} & a_{23}^{(1)} & \dots & a_{2n}^{(1)} \\ 0 & a_{32}^{(1)} & a_{33}^{(1)} & \dots & a_{3n}^{(1)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & a_{n2}^{(1)} & a_{n3}^{(1)} & \dots & a_{nn}^{(1)} \end{pmatrix}. \quad (13)$$

Таким образом, у матрицы $A^{(1)}$ необходимо с помощью леммы 1 вычислить только подматрицу $(a_{ij}^{(1)})_{i,j=2,\dots,n} \in \mathbf{M}_{n-1}$ (так как остальные элементы уже вычислены).

Пусть сделаны $k-1$, $k = 1, \dots, n-1$ шагов этого процесса, т.е. матрица преобразована к виду (3), где

$$A^{(k-1)} = \begin{pmatrix} a_{11} & \|a_1\| & & & & & & \\ \|a_1\| & a_{22}^{(1)} & \|a_1^{(1)}\| & & & & & \\ & \|a_1^{(1)}\| & a_{33}^{(2)} & \ddots & & & & \\ & & \|a_1^{(2)}\| & \ddots & \|a_1^{(k-3)}\| & & & \\ & & & \ddots & a_{k-1,k-1}^{(k-2)} & \|a_1^{(k-2)}\| & & \\ & & & & \|a_1^{(k-2)}\| & a_{kk}^{(k-1)} & \dots & a_{kn}^{(k-1)} \\ & & & & & a_{k+1,k}^{(k-1)} & \dots & a_{k+1,n}^{(k-1)} \\ & & & & & \vdots & \ddots & \vdots \\ & & & & & a_{nk}^{(k-1)} & \dots & a_{nn}^{(k-1)} \end{pmatrix}. \quad (14)$$

Введем обозначение (5). Согласно лемме 12.3 существуют $n-k-1$ матриц $T_{k+1,j} = T_{k+1,j2}(\varphi_{k+1,j})$, $j = k+2, \dots, n$ таких, что справедливо равенство (6).

Обозначим $A_{k+2}^{(k)} = T_{k+1,k+2}A^{(k-1)}T_{k+1,k+2}^t$, $A_{k+3}^{(k)} = T_{k+1,k+3}A_{k+2}^{(k)}T_{k+1,k+3}^t$, $\dots$, $A_n^{(k)} = T_{k+1,n}A_{n-1}^{(k)}T_{k+1,n}^t$. Все матрицы $A_j^{(k)}$, $j = k+1, k+2, \dots, n$ унитарно подобны $A^{(k-1)}$ и потому симметричны. Согласно (7), (9) $A^{(k)} = A_n^{(k)}$. В силу симметрии матрицы

$$A^{(k)} = T_{k+1,n}(\dots(T_{k+1,k+3}(T_{k+1,k+2}A^{(k-1)}T_{k+1,k+2}^t)T_{k+1,k+3}^t)\dots)T_{k+1,n}^t \quad (15)$$

вместо (10) для нее справедливо более точное равенство

$$A^{(k)} = \begin{pmatrix} a_{11} & \|a_1\| & & & & & & & \\ \|a_1\| & a_{22}^{(1)} & \|a_1^{(1)}\| & & & & & & \\ & \|a_1^{(1)}\| & a_{33}^{(2)} & \ddots & & & & & \\ & & \|a_1^{(2)}\| & \ddots & \|a_1^{(k-3)}\| & & & & \\ & & & \ddots & a_{k-1,k-1}^{(k-2)} & \|a_1^{(k-2)}\| & & & \\ & & & & \|a_1^{(k-2)}\| & a_{kk}^{(k-1)} & \|a_1^{(k-1)}\| & & \\ & & & & & \|a_1^{(k-1)}\| & a_{k+1,k+1}^{(k)} & \dots & a_{k+1,n}^{(k)} \\ & & & & & & a_{k+2,k+1}^{(k)} & \dots & a_{k+2,n}^{(k)} \\ & & & & & & \vdots & \ddots & \vdots \\ & & & & & & a_{n,k+1}^{(k)} & \dots & a_{nn}^{(k-1)} \end{pmatrix}. \tag{16}$$

Таким образом, у матрицы $A^{(k)}$ необходимо с помощью леммы 1 вычислить только подматрицу $(a_{ij}^{(k)})_{i,j=k+1,\dots,n} \in \mathbf{M}_{n-k-1}$ (так как остальные элементы уже вычислены).

После $n-2$ шагов этого процесса (т.е. перехода от матриц (3), (14) к (15), (16)) матрица примет требуемый трехдиагональный вид (11), где

$$R = \begin{pmatrix} a_{11} & \|a_1\| & & & & & & \\ \|a_1\| & a_{22}^{(1)} & \|a_1^{(1)}\| & & & & & \\ & \|a_1^{(1)}\| & a_{33}^{(2)} & \ddots & & & & \\ & & \|a_1^{(2)}\| & \ddots & \|a_1^{(n-4)}\| & & & \\ & & & \ddots & a_{n-2,n-2}^{(n-3)} & \|a_1^{(n-3)}\| & & \\ & & & & \|a_1^{(n-3)}\| & a_{n-1,n-1}^{(n-2)} & a_{n,n-1}^{(n-2)} \\ & & & & & a_{n,n-1}^{(n-2)} & a_{nn}^{(n-2)} \end{pmatrix} \tag{17}$$

(напомним, определения векторов $a_1^{(k-1)}$, $k = 1, \dots, n-2$ даются в (5), где считаем, что $a_1^{(0)} = a_1$).

Оценка количества арифметических операций в алгоритме приведения симметричной матрицы к трехдиагональному виду унитарным подобием методом вращений

Оценим трудоемкость k-го шага алгоритма, а затем просуммируем полученные оценки по всем $k = 1, \dots, n-2$.

1. На вычисление $n-k-1$ матриц $T_{k,k+1}, \dots, T_{kn}$, участвующих в (6), согласно лемме 12.2 требуется $4(n-k-1)$ мультипликативных, $2(n-k-1)$ аддитивных и $n-k$ операций извлечения корня.

2. На вычисление компонент $k+1, \dots, n$ k-го столбца и k-ой строки матрицы $A^{(k)}$, равных компонентам вектора $\|a_1^{(k-1)}\| e_1^{(n-k)}$ требуется (для вычисления

длины вектора (5)) $n-k$ операций умножения, $n-k-1$ операций сложения и одна операция извлечения корня. k-й столбец и k-ая строка вычисляются именно этим способом для сокращения количества арифметических операций и уменьшения вычислительной погрешности.

3. Вычисления по формуле (15) можно представить как последовательное вычисление матриц $A_l^{(k)}$, $l = k+1, k+2, \ldots, n$, у которых нужно вычислить только подматрицу $((a_l^{(k)})_{ij})_{i,j=k+1,\ldots,n} \in \mathbf{M}_{n-k-1}$. Согласно лемме 1 на вычисление каждой подматрицы требуется $4(n-k-1)+8 = 4(n-k)+4$ умножений и $2(n-k-1)+4 = 2(n-k)+2$ сложений. На вычисление всех $n-k-1$ подматриц требуется, таким образом, $(n-k-1)(4(n-k)+4) = 4(n-k)^2-4$ умножений и $(n-k-1)(2(n-k)+2) = 2(n-k)^2-2$ сложений.

Итак, на k-ом шаге алгоритма требуется выполнить $4(n-k-1)+(n-k)+4(n-k)^2-4 = 4(n-k)^2+5(n-k)-8$ мультипликативных операций, $2(n-k-1)+(n-k-1)+2(n-k)^2-2 = 2(n-k)^2-3(n-k)-5$ аддитивных операций и $n-k$ операций извлечения корня.

Следовательно, всего для проведения алгоритма требуется выполнить

$$\sum_{k=1}^{n-2}(4(n-k)^2+5(n-k)-8) = 4(n(n-1)(2n-1)/6-1)+5(n(n-1)/2-1)-8(n-2)$$
$$= \frac{4}{3}n^3+O(n^2)+O(n^2)+O(n) = \frac{4}{3}n^3+O(n^2) \ \ (n\to\infty)$$

мультипликативных операций, $\sum_{k=1}^{n-2}(2(n-k)^2-3(n-k)-5) = \frac{2}{3}n^3+O(n^2)$ $(n\to\infty)$ аддитивных операций и $\sum_{k=1}^{n-2}(n-k) = O(n^2)$ $(n\to\infty)$ операций извлечения корня (которые по трудоемкости по порядку можно сравнить с операциями деления).

Таким образом, на приведение симметричной матрицы к трехдиагональному виду унитарным подобием методом вращений требуется $\frac{4}{3}n^3+O(n^2)$ $(n\to\infty)$ мультипликативных операций и $\frac{2}{3}n^3+O(n^2)$ $(n\to\infty)$ аддитивных операций. Заметим, что это количество операций в два с половиной раза меньше, чем требуется для приведения произвольной матрицы к почти треугольному виду унитарным подобием методом вращений и совпадает количеством операций, необходимым для решения линейной системы методом вращений.

Теорема 2. *Всякая невырожденная вещественная симметричная матрица A может быть представлена в виде $A = QRQ^t$, где матрица Q – ортогональная, а матрица R – трехдиагональная.*

Доказательство Совпадает с доказательством теоремы 1.

Хранение матриц Q и R в памяти осуществляется одним из способов, изложенных при обсуждении алгоритма построения QR-разложения для матрицы A методом вращений. Для симметричных матриц A удобно применять второй способ. Действительно, так как на шаге k, $k = 1, \ldots, n-2$ мы использовали $n-k-1$ элементарных вращений $T_{k+1,k+2}, \ldots, T_{k+1,n}$ для получения нулевых

элементов $a_{jk}^{(k)} = 0$, $a_{kj}^{(k)} = 0$, $j = k+2, \ldots, n$, то можем хранить, например, $\cos\varphi_{k+1,j}$ на месте $a_{jk}^{(k)} = 0$, а $\sin\varphi_{k+1,j}$ – на месте $a_{kj}^{(k)} = 0$.

Трудоемкость алгоритма построения описанного выше разложения складывается из количества арифметических операций, необходимых для проведения самого алгоритма, и количества арифметических операций, необходимых для построения матрицы Q. Подробные выкладки были проведены при обсуждении алгоритма построения QR-разложения методом вращений.

§ 15. ПРИВЕДЕНИЕ МАТРИЦЫ К ПОЧТИ ТРЕУГОЛЬНОМУ ВИДУ УНИТАРНЫМ ПОДОБИЕМ МЕТОДОМ ОТРАЖЕНИЙ

Пусть требуется привести матрицу A (не обязательно вещественную) к почти треугольному виду.

Всюду ниже мы будем часто пользоваться следующими фактами.

1. Если по произвольной матрице $U_k \in \mathbf{M}_k$ (размера $k\times k$) построить матрицу $U \in \mathbf{M}_n$ (размера $n \times n$) по формуле

$$U = \begin{pmatrix} I_{n-k} & 0 \\ 0 & U_k \end{pmatrix}, \tag{1}$$

где $I_{n-k} \in \mathbf{M}_{n-k}$ – единичная матрица размера $(n-k)\times(n-k)$, то при умножении матрицы A на матрицу U слева изменяются только последние k строк матрицы A, а при умножении на U справа изменяются только последние k столбцов матрицы A. Это следует из определения умножения матриц.

2. Если матрица $U_k \in \mathbf{M}_k$ в (1) самосопряженная, то матрица $U \in \mathbf{M}_n$, полученная в (1), также самосопряженная. Это доказано при рассмотрении алгоритма метода отражений, см. (13.8).

3. Если матрица $U_k \in \mathbf{M}_k$ в (1) унитарна, то матрица $U \in \mathbf{M}_n$, полученная в (1), также унитарна. Это доказано при рассмотрении алгоритма метода отражений, см. (13.9).

§ 15.1. Случай произвольной матрицы

Обозначим $a_1 = (a_{21}, \ldots, a_{n1})^t$. Согласно лемме 13.9 существует вектор $x^{(1)} \in \mathbf{C}^n$, равный

$$x^{(1)} = \pm\frac{a_1 - \|a_1\|e_1}{\|a_1 - \|a_1\|e_1\|},$$

такой, что $U(x^{(1)})a_1 = \|a_1\|e_1$, где $e_1 = (1, 0, \ldots, 0) \in \mathbf{C}^{n-1}$, $U(x^{(1)}) \in \mathbf{M}_{n-1}$ – матрица отражения. Положим

$$U_1 = \begin{pmatrix} 1 & 0 \\ 0 & U(x^{(1)}) \end{pmatrix}. \tag{2}$$

Как отмечалось выше, матрица U_1 является унитарной.

Умножим матрицу A на U_1 слева, получим матрицу $\widehat{A}^{(1)}$ вида (14.1) (поскольку первая строка матрицы A не изменяется). Умножим матрицу $\widehat{A}^{(1)}$ на $U_1^* = U_1$ справа, получим матрицу (14.2) (с учетом того, что при умножении справа на U_1 первый столбец матрицы $\widehat{A}^{(1)}$ не изменяется).

Пусть сделаны $k-1$, $k = 1, \dots, n-1$ шагов этого процесса, т.е. матрица преобразована к виду

$$A^{(k-1)} = \prod_{i=k}^{2} U_i A \prod_{i=2}^{k} U_i \tag{3}$$

где $A^{(k-1)}$ имеет вид (14.4),

$$U_i = \begin{pmatrix} I_i & 0 \\ 0 & U(x^{(i)}) \end{pmatrix},$$

здесь $I_i \in \mathbf{M}_i$ – единичная матрица размера $i \times i$, $U(x^{(i)}) \in \mathbf{M}_{n-i}$ – матрица отражения размера $(n-i) \times (n-i)$, построенная по вектору

$$x^{(i)} = \pm \frac{a_1^{(i-1)} - \|a_1^{(i-1)}\| e_1^{(n-i)}}{\|a_1^{(i-1)} - \|a_1^{(i-1)}\| e_1^{(n-i)}\|} \in \mathbf{C}^{n-i},$$

где $e_1^{(m)} = (1, 0, \dots, 0) \in \mathbf{C}^m$,

$$a_1^{(i-1)} = (a_{i+1,i}^{(i-1)}, \dots, a_{ni}^{(i-1)})^t \in \mathbf{C}^{n-i}.$$

Обозначим через $a_1^{(k-1)}$ часть первого столбца подматрицы $(a_{ij}^{(k-1)})_{i,j=k,\dots,n}$, см. (14.5). Согласно лемме 13.9 существует матрица отражения

$$U(x^{(k)}) = I - 2x^{(k)}(x^{(k)})^*, \quad x^{(k)} = \pm \frac{a_1^{(k-1)} - \|a_1^{(k-1)}\| e_1^{(n-k)}}{\|a_1^{(k-1)} - \|a_1^{(k-1)}\| e_1^{(n-k)}\|} \in \mathbf{C}^{n-k}, \tag{4}$$

такая, что

$$U(x^{(k)}) a_1^{(k-1)} = \|a_1^{(k-1)}\| e_1^{(n-k)}. \tag{5}$$

Положим

$$U_k = \begin{pmatrix} I_k & 0 \\ 0 & U(x_k) \end{pmatrix}. \tag{6}$$

Как отмечалось выше, матрица U_k является самосопряженной и унитарной.

Умножим матрицу (3) на U_k слева, получим

$$\widehat{A}^{(k)} = U_k A^{(k-1)}, \tag{7}$$

где матрица $\widehat{A}^{(k)}$ имеет вид (14.8) Отметим, что в (7) первые k строк у матриц $\widehat{A}^{(k)}$ и $A^{(k-1)}$ совпадают. Другими словами, преобразование (7) заключается в

умножении матрицы $U(x_k) \in \mathbf{M}_{n-k}$ на подматрицу $(a_{ij}^{(k-1)})_{i=k+1,\dots,n,\, j=k,\dots,n}$ матрицы $A^{(k-1)}$ размера $(n-k)\times(n-k+1)$ (остальная часть $A^{(k-1)}$ в преобразовании (7) не участвует).

Умножим матрицу $\widehat{A}^{(k)}$ на $U_k^* = U_k$ справа, получим из (14.8) (с учетом того, что при умножении справа на U_k столбцы $1,\dots,k$ матрицы $\widehat{A}^{(k)}$ не изменяются)

$$A^{(k)} = \widehat{A}^{(k)} U_k = U_k A^{(k-1)} U_k, \tag{8}$$

где матрица $A^{(k)}$ имеет вид (14.10). Отметим, что в (8) первые k столбцов у матриц $A^{(k)}$ и $\widehat{A}^{(k)}$ совпадают. Другими словами, преобразование (8) заключается в умножении матрицы $U(x_k) \in \mathbf{M}_{n-k}$ на подматрицу $(\widehat{a}_{ij}^{(k-1)})_{i=1,\dots,n,\, j=k+1,\dots,n}$ матрицы $\widehat{A}^{(k-1)}$ размера $n\times(n-k)$ (остальная часть $\widehat{A}^{(k-1)}$ в преобразовании (8) не участвует).

Вычисления по формулам (4) осуществляются следующим образом: вначале вычисляются числа

$$s_k = \sum_{j=k+2}^{n} |a_{jk}^{(k-1)}|^2, \tag{9}$$

$$\|a_1^{(k-1)}\| = \sqrt{|a_{k+1,k}^{(k-1)}|^2 + s_k}. \tag{10}$$

затем – вектор

$$x^{(k)} = (a_{k+1,k}^{(k-1)} - \|a_1^{(k-1)}\|, a_{k+2,k}^{(k-1)}, \dots, a_{nk}^{(k-1)})^t \in \mathbf{C}^{n-k} \tag{11}$$

и его норма

$$\|x^{(k)}\| = \sqrt{|x_1^{(k)}|^2 + s_k}. \tag{12}$$

Теперь можно вычислить искомый вектор $x^{(k)}$:

$$x^{(k)} := x^{(k)}/\|x^{(k)}\|, \quad \text{т.е.} \quad x_j^{(k)} := x_j^{(k)}/\|x^{(k)}\|,\ j = 1,\dots,n-k. \tag{13}$$

После $n-2$ шагов этого процесса (т.е. перехода от матриц (3), (14.4) к (8), (14.10)) матрица примет требуемый почти треугольный вид (14.12), где

$$R = A^{(n-2)} = \prod_{i=n-2}^{1} U_i A \prod_{i=1}^{n-2} U_i. \tag{14}$$

Оценка количества арифметических операций в алгоритме приведения матрицы к почти треугольному виду унитарным подобием методом отражений

Оценим трудоемкость k-го шага алгоритма, а затем просуммируем полученные оценки по всем $k = 1,\dots,n-2$.

1. На вычисление матрицы $U(x_k)$ по формулам (4) требуется

а) $n-k-1$ умножений и $n-k-2$ сложений для вычисления s_k в (9);

б) одно умножение, одно сложение и одна операция извлечения корня для вычисления $\|a_1^{(k-1)}\|$ в (10);

в) одно вычитание для построения вектора $x^{(k)}$ в (11);

г) одно умножение, одно сложение и одна операция извлечения корня для вычисления $\|x^{(k)}\|$ в (12);

д) $n-k$ делений для построения вектора $x^{(k)}$ в (13).

Всего для построения матрицы $U(x_k)$ требуется $(n-k-1)+1+1+(n-k) = 2(n-k)+1$ мультипликативных, $(n-k-2)+1+1+1 = n-k+1$ аддитивных операций и $1+1=2$ операции извлечения корня.

2. Компоненты $k+1,\dots,n$ k-го столбца матрицы $A^{(k)}$, равные компонентам вектора $\|a_1^{(k-1)}\|\, e_1^{(n-k)}$, уже вычислены в (10). Столбец k вычисляется не по общим формулам (7) для сокращения количества арифметических операций и уменьшения вычислительной погрешности.

3. Поскольку в формуле (7) матрица $U(x_k) \in \mathbf{M}_{n-k}$ умножается на подматрицу $(a_{ij}^{(k-1)})_{i=k+1,\dots,n,\, j=k+1,\dots,n}$ матрицы $A^{(k-1)}$ размера $(n-k)\times(n-k)$ (k-й столбец матрицы $\widehat{A}^{(k)}$ уже вычислен в пункте 2), то согласно лемме 13.11 на это требуется $2(n-k)^2+O(n-k)$ $(n\to\infty)$ умножений и столько же сложений.

4. Поскольку в формуле (8) матрица $U(x_k) \in \mathbf{M}_{n-k}$ умножается на подматрицу $(\widehat{a}_{ij}^{(k-1)})_{i=1,\dots,n,\, j=k+1,\dots,n}$ матрицы $\widehat{A}^{(k-1)}$ размера $n\times(n-k)$, то согласно лемме 13.11 на это требуется $2(n-k)n+O(n-k)$ $(n\to\infty)$ умножений и столько же сложений.

Итак, на k-ом шаге алгоритма требуется выполнить $2(n-k)+1+2(n-k)^2+2n(n-k)+O(n-k) = 2n(n-k)+2(n-k)^2+O(n-k)$ мультипликативных операций, $n-k+1+2(n-k)^2+2n(n-k)+O(n-k) = 2n(n-k)+2(n-k)^2+O(n-k)$ аддитивных операций и 2 операции извлечения корня.

Следовательно, всего для проведения алгоритма требуется выполнить

$$\begin{aligned}&\sum_{k=1}^{n-2}\bigl(2n(n-k)+2(n-k)^2+O(n-k)\bigr)\\ &= 2n((n-1)(n-2)/2)+2((n-1)(n-2)(2n-3)/6)+O(n^2)\\ &= n^3+O(n^2)+\tfrac{2}{3}n^3+O(n^2) = \tfrac{5}{3}n^3+O(n^2)\ (n\to\infty)\end{aligned}$$

мультипликативных операций, столько же аддитивных операций и $2(n-2)$ операций извлечения корня (которые по трудоемкости по порядку можно сравнить с операциями деления).

Таким образом, на приведение матрицы к почти треугольному виду унитарным подобием методом отражений требуется $\frac{5}{3}n^3+O(n^2)$ $(n\to\infty)$ мультипликативных операций и столько же аддитивных операций. Заметим, что это количество операций в два с половиной раза больше, чем нужно для решения линейной системы методом отражений.

Теорема 1. *Всякая невырожденная матрица A может быть представлена в виде $A = Q\,R\,Q^t$, где матрица Q – унитарная, а матрица R – верхняя почти треугольная.*

Доказательство. Проведем для матрицы A изложенный выше алгоритм, осуществимый для всякой невырожденной матрицы. Обозначим в (14) $\hat{Q} = \prod\limits_{i=n-2}^{1} U_i$. Как произведение унитарных матриц, матрица $\hat{Q}$ унитарна. Тогда (14) имеет вид $R = \hat{Q}A\hat{Q}^t$, откуда $A = (\hat{Q})^{-1}R(\hat{Q}^t)^{-1} = QRQ^t$, где $Q = (\hat{Q})^t = (\hat{Q})^{-1}$ – унитарная матрица. Матрица R, имеющая вид (14.12), удовлетворяет условиям теоремы.

Замечание 1. Как отмечалось выше, построенное в теореме 1 разложение используется в ряде алгоритмов нахождения собственных значений матрицы.

Хранение матриц Q и R в памяти осуществляется одним из способов, изложенных при обсуждении алгоритма построения QR-разложения для матрицы A методом отражений.

Трудоемкость алгоритма построения описанного выше разложения складывается из количества арифметических операций, необходимых для проведения самого алгоритма, и количества арифметических операций, необходимых для построения матрицы Q. Подробные выкладки были проведены при обсуждении алгоритма построения QR-разложения методом отражений.

§ 15.2. Случай самосопряженной матрицы

Рассмотрим ситуацию, когда описанный выше метод приведения к почти треугольному виду применяется к самосопряженной матрице $A \in \mathbf{M}_n$.

Согласно (14.1), (14.2) $A^{(1)} = U_1 A U_1^t$, где U_1 – унитарная матрица, т.е. $A^{(1)}$ и A – унитарно подобны. Следовательно, $A^{(1)}$ – самосопряженная матрица. Согласно (7), (8) на k-ом ($k = 1, \ldots, n-2$) шаге алгоритма $A^{(k)} = U_k A^{(k-1)} U_k^t$, где U_k – унитарная матрица. Следовательно, $A^{(k)}$ и A унитарно подобны, и $A^{(k)}$ – самосопряженная матрица для всякого $k = 1, \ldots, n-2$. Таким образом, $R = A^{(n-2)}$ – почти треугольная и самосопряженная, т.е. трехдиагональная матрица.

Запишем описанный выше процесс приведения самосопряженной матрицы к трехдиагональному виду так, чтобы максимально уменьшить объем вычислительной работы за счет использования симметрии.

Лемма 1. *Для всякой матрицы отражения $U = U(x) \in \mathbf{M}_n$ и всякой самосопряженной матрицы $A \in \mathbf{M}_n$ матрица $B = UAU^* = UAU$ может быть вычислена за $2n^2 + O(n)$ умножений и столько же сложений.*

Доказательство. Поскольку $U(x) = I - 2xx^*$, то

$$B = (I-2xx^*)A(I-2xx^*) = (I-2xx^*)(A-2Axx^*) = A-2Axx^*-2xx^*A+4xx^*Axx^*.$$

Обозначим

$$y = Ax \in \mathbf{C}^n \tag{15}$$

В силу самосопряженности матрицы A имеем $y = Ax = A^*x = (x^*A)^*$,

$$4xx^*Axx^* = 2xx^*Axx^* + 2xx^*Axx^* = 2xx^*yx^* + 2xy^*xx^*$$

и

$$B = A - 2yx^* - 2xy^* + 2xx^*yx^* + 2xy^*xx^* = A - 2(I - xx^*)yx^* - 2xy^*(I - xx^*)$$

Обозначим

$$z = 2(I - xx^*)y = 2y - x(x^*y) = 2y - 2(x, y)x. \tag{16}$$

Тогда

$$B = A - zx^* - xz^* \tag{17}$$

После этих преобразований мы можем сформулировать алгоритм вычисления матрицы B:

1) Вычисляется вектор y по формуле (15). На это требуется $n^2 + O(n)$ мультипликативных операций и столько же аддитивных операций.

2) Вычисляется вектор z по формуле (16). На вычисление $\alpha = 2(x, y)$ – удвоенного евклидова скалярного произведения, требуется $n + O(1)$ мультипликативных операций и столько же аддитивных операций; на вычисление $z = 2y - \alpha x$ требуется $2n$ мультипликативных операций и n аддитивных операций. Общее число операция, необходимое для вычисления вектора z – $3n + O(1)$ мультипликативных и $2n + O(1)$ аддитивных операций.

3) Вычисляется матрица B по формуле (17). Матрица B как унитарно подобная A самосопряжена, поэтому по формуле (17) вычисляются только $n(n+1)/2$ элементов верхнего треугольника матрицы B. На вычисление каждого элемента матрицы B по формуле (17) надо выполнить 2 умножения и 2 вычитания, поэтому трудоемкость вычисления B по формуле (17) равна $n(n+1) = n^2 + O(n)$ мультипликативным и $n^2 + O(n)$ аддитивным операциям.

Таким образом, этот алгоритм требует $n^2 + O(n) + 3n + O(1) + n^2 + O(n) = 2n^2 + O(n)$ мультипликативных и столько же аддитивных операций. Лемма доказана.

Замечание 2. Для несамосопряженной матрицы A вычисление матрицы $B = UAU$ требует $4n^2 + O(n)$ умножений столько же сложений (см. лемму 13.11).

Обозначим $a_1 = (a_{21}, \ldots, a_{n1})^t$. Согласно лемме 13.9 существует вектор $x^{(1)} \in \mathbf{C}^n$, равный

$$x^{(1)} = \pm \frac{a_1 - \|a_1\| e_1}{\|a_1 - \|a_1\| e_1\|},$$

такой, что $U(x^{(1)})a_1 = \|a_1\| e_1$, где $e_1 = (1, 0, \ldots, 0) \in \mathbf{C}^{n-1}$, $U(x^{(1)}) \in \mathbf{M}_{n-1}$ – матрица отражения. Введем матрицу U_1 как в (2) и вычислим матрицу

$A^{(1)} = U_1 A U_1$. В силу самосопряженности матрицы $A^{(1)}$ вместо (14.2) для нее справедливо более точное равенство

$$A^{(1)} = U_1 A U_1 = \begin{pmatrix} a_{11} & \|a_1\| & 0 & \dots & 0 \\ \|a_1\| & a_{22}^{(1)} & a_{23}^{(1)} & \dots & a_{2n}^{(1)} \\ 0 & a_{32}^{(1)} & a_{33}^{(1)} & \dots & a_{3n}^{(1)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & a_{n2}^{(1)} & a_{n3}^{(1)} & \dots & a_{nn}^{(1)} \end{pmatrix}. \tag{18}$$

Таким образом, у матрицы $A^{(1)}$ необходимо с помощью леммы 1 вычислить только подматрицу $(a_{ij}^{(1)})_{i,j=2,\dots,n} \in \mathbf{M}_{n-1}$ (так как остальные элементы уже вычислены).

Пусть сделаны $k-1$, $k = 1, \dots, n-1$ шагов этого процесса, т.е. матрица преобразована к виду (14.3), где матрица $A^{(k-1)}$ имеет вид (14.14).

Введем обозначение (14.5). Согласно лемме 13.9 существует матрица отражения (4) такая, что выполнено (5). Определим U_k равенством (6). Вычислим матрицу

$$A^{(k)} = U_k A^{(k-1)} U_k. \tag{19}$$

Матрица $A^{(k)}$ унитарно подобна самосопряженной матрице $A^{(k-1)}$. Поэтому она самосопряжена и вместо (14.10) для нее справедливо более точное равенство (14.16). Таким образом, у матрицы $A^{(k)}$ необходимо с помощью леммы 1 вычислить только подматрицу $(a_{ij}^{(k)})_{i,j=k+1,\dots,n} \in \mathbf{M}_{n-k-1}$ (так как остальные элементы уже вычислены).

После $n-2$ шагов этого процесса (т.е. перехода от матриц (14.3), (14.14) к (19), (14.16)) матрица примет требуемый трехдиагональный вид (14.17).

Оценка количества арифметических операций в алгоритме приведения самосопряженной матрицы к трехдиагональному виду унитарным подобием методом отражений

Оценим трудоемкость k-го шага алгоритма, а затем просуммируем полученные оценки по всем $k = 1, \dots, n-2$.

1. На вычисление матрицы $U(x_k)$ по формулам (4) требуется $2(n-k)+1$ мультипликативных, $n-k+1$ аддитивных операций и 2 операции извлечения корня (см. вычисления при оценке количества арифметических операций в алгоритме приведения матрицы к почти треугольному виду унитарным подобием методом отражений).

2. Компоненты $k+1, \dots, n$ k-го столбца матрицы $A^{(k)}$, равные компонентам вектора $\|a_1^{(k-1)}\| e_1^{(n-k)}$, уже вычислены в (10). Столбец k вычисляется не по общим формулам (7) для сокращения количества арифметических операций и уменьшения вычислительной погрешности.

3. Поскольку в формуле (19)

$$(a_{ij}^{(k)})_{i=k+1,\dots,n,\, j=k+1,\dots,n} = U(x_k)(a_{ij}^{(k-1)})_{i=k+1,\dots,n,\, j=k+1,\dots,n} U(x_k), \tag{20}$$

то в силу леммы 1 на вычисление подматрицы (20) матрицы $A^{(k)}$ требуется $2(n-k)^2 + O(n-k)$ мультипликативных и столько же аддитивных операций.

Итак, на k-ом шаге алгоритма требуется выполнить $n-k+1+2(n-k)^2 + O(n-k) = 2(n-k)^2 + O(n-k)$ мультипликативных операций, $n-k+1+2(n-k)^2 + O(n-k) = 2(n-k)^2 + O(n-k)$ аддитивных операций и 2 операции извлечения корня.

Следовательно, всего для проведения алгоритма требуется выполнить

$$\sum_{k=1}^{n-2}(2(n-k)^2+O(n-k)) = 2((n-1)(n-2)(2n-3)/6)+O(n^2) = \tfrac{2}{3}n^3+O(n^2)\ (n\to\infty)$$

мультипликативных операций, столько же аддитивных операций и $2(n-2)$ операций извлечения корня (которые по трудоемкости по порядку можно сравнить с операциями деления).

Таким образом, на приведение матрицы к почти треугольному виду унитарным подобием методом отражений требуется $\frac{2}{3}n^3 + O(n^2)$ $(n \to \infty)$ мультипликативных операций и столько же аддитивных операций. Заметим, что это количество операций в два с половиной раза меньше, чем требуется для приведения произвольной матрицы к почти треугольному виду унитарным подобием методом отражений и совпадает количеством операций, необходимым для решения линейной системы методом отражений.

Теорема 2. *Всякая невырожденная самосопряженная матрица A может быть представлена в виде $A = Q\,R\,Q^t$, где матрица Q – унитарная, а матрица R – трехдиагональная.*

Доказательство Совпадает с доказательством теоремы 1.

Хранение матриц Q и R в памяти осуществляется одним из способов, изложенных при обсуждении алгоритма построения QR-разложения для матрицы A методом отражений.

Трудоемкость алгоритма построения описанного выше разложения складывается из количества арифметических операций, необходимых для проведения самого алгоритма, и количества арифметических операций, необходимых для построения матрицы Q. Подробные выкладки были проведены при обсуждении алгоритма построения QR-разложения методом отражений.

Глава II.

МЕТОДЫ НАХОЖДЕНИЯ СОБСТВЕННЫХ ЗНАЧЕНИЙ

§ 1. ТОЧНЫЕ И ИТЕРАЦИОННЫЕ МЕТОДЫ

Определение. Метод решения линейной системы называется **точным**, если при отсутствии округлений точное решение системы находится этим методом за конечное число арифметических операций (например, для метода Гаусса это $\frac{2}{3}n^3 + O(n^2)$).

На реальной вычислительной машине точный метод дает некоторое приближение к точному решению системы. Мера близости оценена в § I.3.

Все описанные выше методы являются точными.

Определение. Метод решения линейной системы называется **итерационным**, если он состоит в вычислении последовательности $\{x_k\}$, сходящейся к точному решению: $x_k \to x$ при $k \to \infty$. Итерационный метод за конечное число арифметических операций дает только некоторое приближение x_{k_0} к точному решению.

Теория итерационных методов будет изложена в курсе "Численные методы".

Определение. Метод нахождения собственных значений называется **итерационным**, если он состоит в вычислении последовательности $\{\lambda_k\}$, сходящейся к точному собственному значению: $\lambda_k \to \lambda$ при $k \to \infty$. Итерационный метод за конечное число арифметических операций дает только некоторое приближение λ_{k_0} к точному собственному значению.

Теорема 1. (Без доказательства.) *Не может существовать точного метода нахождения всех собственных значений произвольной матрицы $A \in \mathbf{M}_n$ при $n \geq 5$. Другими словами, за конечное число арифметических операций нельзя найти все собственные значения произвольной матрицы $A \in \mathbf{M}_n$ при $n \geq 5$.*