МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Московский авиационный институт (национальный исследовательский университет)

Н.С. Северина, Ю.В. Сластушенский

ЧИСЛЕННЫЕ МЕТОДЫ И АЛГОРИТМЫ. ЛИНЕЙНАЯ И ЭЛЕМЕНТАРНАЯ АЛГЕБРА

Учебное пособие

Утверждено
на заседании редсовета
31 августа 2020 г.

Москва
Издательство МАИ
2021

Северина Н.С., Сластушенский Ю.В. Численные методы и алгоритмы. Линейная и элементарная алгебра: Учеб. пособие. — М.: Изд-во МАИ, 2021. — 104 с.: ил.

Учебное пособие написано на основе лекций, читаемых авторами студентам технических факультетов МАИ, и содержит учебный материал, излагаемый в соответствии с программой преподавания разделов «Вычислительные методы линейной алгебры» и «Численные методы решения нелинейных уравнений и систем нелинейных уравнений» дисциплины «Численные методы», общей для студентов МАИ, обучающихся по различным направлениям подготовки, в том числе по направлению «Прикладная математика и информатика».

Пособие может быть использовано для обучения дисциплине «Численные методы» студентов бакалавриата и специалитета и приобретения ими соответствующих навыков разработки алгоритмов и написания программ. Магистрантам, аспирантам и научным сотрудникам пособие может служить справочником по методам вычислительной математики.

Рецензенты:

кафедра прикладной газовой динамики и горения ТГУ (зав. кафедрой д-р физ.-мат. наук, проф. *Г.Р. Шрагер*);

канд. физ.-мат. наук, доцент *А.Г. Карпенко*

ISBN 978-5-4316-0785-1

ПРЕДИСЛОВИЕ

Учебное пособие написано на основе лекций, практических и лабораторных занятий, проводимых авторами для бакалавров и специалистов технических факультетов, в том числе студентов Института № 8 «Информационные технологии и прикладная математика» Московского авиационного института. Пособие рекомендуется для изучающих соответствующие разделы методов вычислительной математики дисциплины «Численные методы». Особенность пособия состоит в том, что в него не только включён необходимый лекционный материал по изучаемой дисциплине, но и показаны практическое применение рассматриваемых методов на конкретных примерах и техника написания алгоритмов соответствующих программ, их реализующих.

В соответствии с программой подробно изложен материал разделов «Вычислительные методы линейной алгебры» и «Численные методы решения нелинейных уравнений и систем нелинейных уравнений» дисциплины «Численные методы». Теоретические выводы и доказательства излагаются достаточно подробно и доступно для студенческой аудитории. Вычислительные методы рассматриваются в наглядной, доступной, но в то же время математически строгой формах. В конце описания каждого метода предлагается вариант алгоритма программы, этот метод реализующей.

Подробно и доступным языком формулируются пояснения, в том числе с участием смежных дисциплин, которые могут оказаться полезными студенту для более глубокого усвоения основного содержания курса. В результате материал представлен так, что для понимания сути излагаемых методов нет необходимости обращаться к дополнительным источникам. Рассматриваемые вычислительные методы иллюстрируются примерами задач, решаемых с достаточно подробными комментариями, что позволяет использовать пособие и как справочное по методам вычислительной математики. Также пособие может быть использовано для приобретения читателями соот-

ветствующих навыков разработки алгоритмов и написания программ. Для проведения практических занятий рекомендуются задачники, включённые в прилагаемый список литературы по данному курсу.

Учебное пособие предназначено для специалистов, бакалавров, магистров, аспирантов и научных сотрудников, использующих в своей учебной и научной практике изложенные методы вычислительной математики.

1. ОСНОВНЫЕ ПОНЯТИЯ

В настоящее время численные методы являются мощным математическим средством решения многих научно-технических проблем. Это связано как с невозможностью в подавляющем большинстве случаев получить точное аналитическое решение, так и со стремительным развитием ЭВМ. Несмотря на существование многочисленных пакетов прикладных программ, предназначенных для решения задач в различных областях инженерной деятельности, для научных и инженерно-технических работников также необходимо понимание существа основных используемых численных методов и алгоритмов, поскольку зачастую интерпретация результатов расчётов нетривиальна и требует специальных знаний относительно особенностей применяемых методов.

1.1. СТРУКТУРА ПОГРЕШНОСТИ

Использование численных методов неизбежно приводит к возникновению различных видов погрешностей вычислений и алгоритмов. В этой связи важно понимать структуру погрешностей при решении конкретных задач. Различают абсолютную и относительную погрешность.

Определение. Пусть a — точное, вообще говоря, неизвестное числовое значение некоторой величины, а $\bar{a}$ — известное приближённое числовое значение этой величины, тогда число

$$\Delta(a) = |a - \bar{a}|$$

называют *абсолютной погрешностью* числа a, а отношение абсолютной погрешности измерения к модулю опорного значения этой величины

$$\delta(a) = \frac{\Delta(a)}{|a|}$$

называют его *относительной погрешностью.*

В качестве опорного значения может выступать не только точное, но и какое-либо приближённое числовое значение этой величины. Относительная погрешность является безразмерной величиной; её численное значение может указываться, например, в процентах. Нетрудно показать, что при сложении и вычитании складываются абсолютные погрешности, а при делении и умножении — относительные погрешности.

Существуют четыре источника погрешностей, полученных в результате численного решения задачи:

1) физическая и математическая модели изучаемого объекта;
2) исходные данные, входящие в описание задачи;
3) приближённость численного метода, применяемого для решения;
4) ошибки округления в процессе вычисления на ЭВМ.

Первые два источника погрешностей приводят к так называемой неустранимой погрешности. Эта погрешность может присутствовать, даже если решение сформулированной задачи найдено точно.

Погрешность метода возникает из-за того, что точный математический оператор и исходные данные, в частности начальные и краевые условия, заменяются по определённым правилам приближёнными. Так, производные заменяются их разностными аналогами, интегралы — суммами, функции — специальными многочленами. Также при решении многих задач строятся бесконечные итерационные процессы, которые естественным образом прекращаются после конечного числа итераций.

Как правило, погрешность метода может быть оценена и поддаётся контролю. Для некоторых методов ниже такая оценка будет представлена. Погрешность метода следует выбирать так, чтобы она была не более чем на порядок меньше неустранимой погрешности.

Погрешность округления (или вычислительная погрешность) возникает в связи с тем, что вычисления производятся с конечным числом значащих цифр, т.е. в их процессе производятся округления.

Определение. Значащими цифрами в записи приближённого числа называются:

1) все ненулевые цифры;
2) нули, содержащиеся между ненулевыми цифрами;
3) нули в конце числа, являющиеся представителями сохранённых десятичных разрядов при округлении.

Округления производятся по следующим правилам:

1) если в старшем из отбрасываемых разрядов стоит цифра меньше пяти, то содержимое сохраняемых разрядов не изменяется;

2) если в старшем из отбрасываемых разрядов стоит цифра больше пяти, то в младший сохраняемый разряд добавляется единица;

3) если в старшем из отбрасываемых разрядов стоит цифра, равная пяти, и среди остальных отброшенных цифр есть ненулевые, то в младший сохраняемый разряд добавляется единица;

4) если в старшем из отбрасываемых разрядов стоит цифра, равная пяти, и все отброшенные цифры являются нулями, то младший сохраняемый разряд остаётся неизменным, если он чётный, и в младший сохраняемый разряд добавляется единица, если он нечётный (правило чётной цифры).

Правило чётной цифры должно обеспечить компенсацию знаков ошибок. Очевидно, что погрешность, возникающая при округлении, не превышает половины младшего оставляемого разряда.

Пример. Число 2.74 могло быть получено при округлении любых чисел от 2.735000... до 2.745000..., разница между которыми с округлённым числом не превышает 0.005.

В следующем примере значащие цифры подчёркнуты.

Пример. $\underline{2.305}$; $0.0\underline{357}$; $0.00\underline{1123}$; $0.0\underline{35299879} \approx 0.0\underline{35300}$.

При округлении числа 0.035299879 до шести знаков после запятой получается число 0.035300, в котором последние два нуля являются значащими. Если отбросить эти нули, то полученное число 0.0353 не является равнозначным с числом 0.035300 — приближённым значением числа 0.035299879, так как погрешности указанных приближённых чисел отличаются (0.00005 и 0.0000005).

Повторное округление проводить не следует, так как оно может привести к увеличению погрешности.

Пример. Число 2.7346 округляется до 2.73, однако двойное округление приводит сначала к числу 2.735, а затем к числу 2.74, что не только дальше от исходного значения, чем 2.73, но и его погрешность 0.0054 превосходит допустимую величину 0.005.

Легко видеть, что абсолютная погрешность характеризуется числом верных цифр после запятой, а относительная погрешность — числом верных значащих цифр.

Определение. Первые n значащих цифр в записи приближённого числа называются *верными* в узком смысле, если абсолютная погрешность числа не превосходит половины единицы разряда, соответствующего n-й значащей цифре, считая слева направо.

Пример. Определить верные цифры приближённого значения 2.721 числа e, если известно, что $e = 2.718281828\ldots$. Поскольку $|2.721 - e| < 0.003 < 0.005$, то верными являются только три первые цифры 2.72, а последнюю цифру можно отбросить. Пусть $x = 1.10253 \pm 0.00009$. Тогда верными являются только первые четыре значащие цифры x, так как $0.00009 < 0.0005$, а цифры 5 и 3 не удовлетворяют определению.

Таким образом, легко показать, что порядок последней верной значащей цифры числа при его известной абсолютной погрешности Δ можно записать следующей формулой: $p = \left[\log_{10}(2\Delta)\right] + 1$.

Поскольку на современных ЭВМ число записывается, как правило, минимум с 10−12 десятичными знаками, то погрешность единичного округления $\Delta = 10^{-10} \div 10^{-12}$ обычно пренебрежимо мала по сравнению с неустранимой погрешностью и погрешностью метода. Хотя при решении больших задач производятся миллиарды операций, и можно предположить, что ошибки округления могут заметно накапливаться, однако, поскольку они носят случайный характер, обычно происходит их взаимная компенсация. Тем не менее, зачастую строятся специальные алгоритмы, в частности, итерационные, которые малочувствительны к ошибкам округления.

1.2. КОРРЕКТНОСТЬ ЗАДАЧИ

При численном решении основных задач необходимо знать какие-либо входные (исходные) данные — начальные, краевые (граничные) значения искомой функции, коэффициенты и правые части уравнений и т.д. Очевидно, что для исследователя важно знать, существует

ли решение поставленной задачи, единственно ли оно и как оно зависит от входных данных.

Определение. Говорят (следуя Адамару), что математическая задача поставлена *корректно*, если она:

1) разрешима при любых допустимых входных данных;

2) имеет всегда единственное решение;

3) решение непрерывно зависит от входных данных в некоторой разумной топологии, т.е. малому изменению входных данных соответствует малое изменение решения (в этом случае говорят, что задача *устойчива*).

Задача поставлена некорректно, если она не обладает каким-либо из вышеперечисленных свойств, например, если её решение неустойчиво относительно входных данных, т.е. их малому изменению могут соответствовать большие изменения решения.

Известно, что корректной задачей является задача численного интегрирования, а некорректной — задача численного дифференцирования. Классическим примером некорректной задачи является задача Коши для уравнения Лапласа. Эта некорректность исходной задачи проявляется и при её численном решении.

В настоящее время развиты и методы решения некорректных задач. К их числу относятся так называемые методы регуляризации, которые сводят решение исходной задачи к решению близкой к ней вспомогательной с некоторым малым параметром ε, так что при $\varepsilon \to 0$ решение вспомогательной задачи должно стремиться к решению исходной задачи. Для некоторых численных методов ниже будут формулироваться условия корректности и устойчивости.

1.3. СВОЙСТВА МАТРИЦ

Напомним некоторые сведения из линейной алгебры, которые потребуются в дальнейшем. Рассмотрим прямоугольную матрицу:

$$A = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ \dots & \dots & \dots & \dots \\ a_{m1} & a_{m2} & \dots & a_{mn} \end{bmatrix}.$$

Две матрицы $A \equiv \left[a_{ij}\right]$ и $B \equiv \left[b_{ij}\right]$ размерности $m \times n$ равны друг другу, если $a_{ij} = b_{ij}$ для всех $i = 1,\dots,m$ и $j = 1,\dots,n$.

Сумма двух матриц A и B размерности $m \times n$ есть матрица размерности $m \times n$:

$$A + B \equiv \left[a_{ij} \right] + \left[b_{ij} \right] \equiv \left[a_{ij} + b_{ij} \right].$$

Произведение матрицы A на скаляр α есть матрица размерности $m \times n$:

$$\alpha A \equiv \alpha \left[a_{ij} \right] = \left[\alpha a_{ij} \right].$$

Произведение матрицы A размерности $m \times n$ на матрицу B размерности $n \times r$ есть матрица C размерности $m \times r$:

$$C = AB \equiv \left[a_{ij} \right] \left[b_{jk} \right] \equiv \left[c_{ik} \right], \text{ где } c_{ik} = \sum_{j=1}^{n} a_{ij} b_{jk} .$$

Таким образом, элемент c_{ik} матрицы C есть сумма произведений i-й строки матрицы A на соответствующие элементы k-го столбца матрицы B, при этом число столбцов матрицы A должно равняться числу строк матрицы B. Из существования произведения AB вовсе не следует существование произведения BA. Для квадратных матриц ($m = n$) одного порядка существуют и AB, и BA, но, вообще говоря, $AB \neq BA$.

Определение. Определитель (детерминант) квадратной матрицы будем обозначать det A:

$$\det A = \begin{vmatrix} a_{11} \; a_{12} \ldots a_{1n} \\ \ldots \; \ldots \ldots \ldots \\ a_{n1} \; a_{n2} \ldots a_{nn} \end{vmatrix}.$$

Определитель квадратной матрицы с n^2 (действительными или комплексными) числами (элементами a_{ij}) есть сумма вида:

$$\det A = \sum_{j=1}^{n} (-1)^{i+j} a_{ij} M_{ij} ,$$

где M_{ij} — дополнительный минор к элементу a_{ij}. Эта формула называется разложением по строке i. Также справедливо аналогичное разложение по любому столбцу, вследствие чего при транспонировании определитель матрицы не изменяется.

Определение. Минором порядка k матрицы A называется определитель k-го порядка, составленный из элементов, которые находятся на пересечении k строк и k столбцов матрицы A. *Рангом* матрицы A называется такое число r, что все миноры порядка $r+1$ и выше равны нулю. *Дополнительный минор* к элементу a_{ij} матрицы A порядка n есть определитель порядка $n-1$, составленный из всех элементов матрицы A, которые не находятся на пересечении i-й строки и j-го столбца.

Определитель матрицы первого порядка $A=[a_{11}]$ равен a_{11}.

Определитель матрицы второго порядка $A=\begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix}$ равен:

$$\det A=\begin{vmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{vmatrix}=(-1)^{1+1}a_{11}a_{22}+(-1)^{1+2}a_{12}a_{21}=a_{11}a_{22}-a_{12}a_{21}.$$

Пример. Найдём определитель матрицы $A=\begin{pmatrix} 1 & 2 & 3 \\ 3 & -2 & 4 \\ -1 & 0 & 2 \end{pmatrix}$.

Воспользуемся разложением по первой строке:

$$\det A=\begin{vmatrix} 1 & 2 & 3 \\ 3 & -2 & 4 \\ -1 & 0 & 2 \end{vmatrix}=(-1)^{1+1}\cdot 1\cdot\begin{vmatrix} -2 & 4 \\ 0 & 2 \end{vmatrix}+(-1)^{1+2}\cdot 2\cdot\begin{vmatrix} 3 & 4 \\ -1 & 2 \end{vmatrix}+(-1)^{1+3}\cdot 3\cdot\begin{vmatrix} 3 & -2 \\ -1 & 0 \end{vmatrix}=$$

$$=1\cdot(-4)-2\cdot 10+3\cdot(-2)=-30.$$

Определение. Алгебраическим дополнением порядка k матрицы A называется выражение $(-1)^{i+j}M_{ij}$.

Важным частным случаем квадратной матрицы является *диагональная* матрица:

$$A=\begin{bmatrix} a_{11} & 0 & \dots & 0 \\ 0 & a_{22} & \dots & 0 \\ \dots & \dots & \dots & \dots \\ 0 & 0 & \dots & a_{nn} \end{bmatrix}.$$

При $a_{11} = a_{22} = \ldots = a_{nn} = 1$. Такая матрица называется *единичной* и обозначается через *E*. Другим частным случаем квадратной матрицы является *трёхдиагональная* матрица:

$$A = \begin{bmatrix} a_{11} & a_{12} & 0 & \ldots & 0 & 0 \\ a_{21} & a_{22} & a_{23} & \ldots & 0 & 0 \\ \ldots & \ldots & \ldots & \ldots & \ldots & \ldots \\ 0 & 0 & 0 & \ldots & a_{n-1,\,n-1} & a_{n-1,\,n} \\ 0 & 0 & 0 & \ldots & a_{n,\,n-1} & a_{nn} \end{bmatrix}.$$

С такими матрицами часто приходится иметь дело при решении дифференциальных уравнений.

Матрица называется *нижнетреугольной*, если все её элементы, расположенные выше главной диагонали, равны нулю:

$$A = \begin{bmatrix} a_{11} & 0 & \ldots & 0 & 0 \\ a_{21} & a_{22} & \ldots & 0 & 0 \\ \ldots & \ldots & \ldots & \ldots & \ldots \\ a_{n1} & a_{n2} & \ldots & a_{n,n-1} & a_{nn} \end{bmatrix}.$$

Аналогично определяется *верхнетреугольная* матрица.

Квадратная матрица называется *симметричной*, если её элементы удовлетворяют соотношению $a_{ij} = a_{ji}$, $i, j = 1, \ldots, n$.

Если в матрице поменять строки со столбцами, то получим *транспонированную* матрицу A^T. Очевидно, что x^T обозначает вектор-строку. Квадратная матрица *A* симметрична, если $A = A^T$.

Определение. Матрица A^{-1} называется *обратной* матрице *A,* если $AA^{-1} = A^{-1}A = E$. Необходимым и достаточным условием существования обратной матрицы A^{-1} является условие $\det A \neq 0$. Говорят, что в этом случае матрица *A* несобственная, или невырожденная. Для неквадратных матриц и вырожденных матриц обратных матриц не существует.

Матрица, обратная матрице *A*, представима в виде:

$$A^{-1} = \frac{adj(A)}{\det A},$$

где *adj*(*A*) — *присоединённая* матрица (матрица, составленная из алгебраических дополнений для соответствующих элементов транспонированной матрицы).

Пример. Найдём обратную матрицу для $A = \begin{pmatrix} -1 & 2 \\ 3 & -5 \end{pmatrix}$.

$$A^T = \begin{pmatrix} -1 & 3 \\ 2 & -5 \end{pmatrix}. \quad A^{-1} = \frac{\begin{pmatrix} (-1)^{1+1} \cdot (-5) & (-1)^{1+2} \cdot 2 \\ (-1)^{2+1} \cdot 3 & (-1)^{2+2} \cdot (-1) \end{pmatrix}}{5-6} = \frac{\begin{pmatrix} -5 & -2 \\ -3 & -1 \end{pmatrix}}{-1} = \begin{pmatrix} 5 & 2 \\ 3 & 1 \end{pmatrix}.$$

1.4. НОРМЫ ВЕКТОРОВ И МАТРИЦ

В математике зачастую рассматриваются множества, элементами которых являются числа, векторы, матрицы или функции. Сами множества обычно являются линейными нормированными пространствами, ибо в них определены операции сложения элементов и их умножения на число и введена норма каждого элемента. Понятия нормы векторов и матриц необходимы для исследования сходимости численных методов решения задач линейной алгебры.

Определение. Нормой вектора $x = (x_1, x_2, \ldots, x_n)^T$ (обозначают $\|x\|$) в n-мерном вещественном пространстве векторов $x \in R^n$ называется вещественное неотрицательное число, вычисляемое с помощью компонент вектора и удовлетворяющее следующим условиям:

1) $\|x\| \geq 0$, причём $\|x\| = 0$ тогда и только тогда, когда $x = \vartheta$;
2) $\|\alpha \cdot x\| = |\alpha| \cdot \|x\|$ для любых $\alpha \in R$;
3) $\|x + y\| \leq \|x\| + \|y\|$ (неравенство треугольника).

Наиболее употребительными являются следующие нормы векторов:

$$\|x\|_1 = \sum_{i=1}^{n} |x_i|,$$

$$\|x\|_2 = \sqrt{\sum_{i=1}^{n} x_i^2} = \sqrt{(x, x)} \quad \text{(евклидова норма)},$$

$$\|x\|_p = \left[\sum_{i=1}^{n} |x_i|^p \right]^{\frac{1}{p}} \quad (p \geq 3),$$

$$\|x\|_c = \max_i |x_i| \quad \text{(соответствует пределу при } p \to \infty).$$

Очевидно, на множестве действительных чисел $\|x\| = |x|$.

Между разными нормами существует соотношение:

$$\|x\|_1 \geq \|x\|_2 \geq \|x\|_p \geq \|x\|_c .$$

Пример. Вектор $x = (1, -2, 3)^T$ порождает следующие нормы:

$$\|x\|_1 = 1 + 2 + 3 = 6 , \ \|x\|_2 = \sqrt{1 + 4 + 9} = \sqrt{14} \approx 3.742 ,$$

$$\|x\|_3 = \sqrt[3]{1 + 8 + 27} = \sqrt[3]{36} \approx 3.302 , \ \|x\|_4 = \sqrt[4]{1 + 16 + 81} = \sqrt[4]{98} \approx 3.146 ,$$

$$\|x\|_{10} = \sqrt[10]{1 + 1024 + 59049} = \sqrt[10]{60074} \approx 3.005 , \|x\|_c = \max(1, 2, 3) = 3 .$$

Определение. Нормой матрицы $A_{n\times n}$ (обозначается $\|A\|$) с вещественными элементами в пространстве матриц называют вещественное неотрицательное число, вычисляемое с помощью элементов матрицы и обладающее следующими свойствами:

1) $\|A\| \geq 0$, причём $\|A\| = 0$ тогда и только тогда, когда $A = \vartheta$;
2) $\|\alpha \cdot A\| = |\alpha| \cdot \|A\|$ для любых $\alpha \in R$;
3) $\|A + B\| \leq \|A\| + \|B\|$ (неравенство треугольника для сложения);
4) $\|A \cdot B\| \leq \|A\| \cdot \|B\|$ (неравенство треугольника для умножения).

Как видно из последнего свойства (если в качестве матрицы B использовать вектор x), норма матриц должна быть согласована с нормой векторов. Это согласование осуществляется связью:

$$\|A \cdot x\| \leq \|A\| \cdot \|x\| .$$

В пространстве квадратных матриц размерности $n \times n$ наиболее употребительны следующие нормы (согласованные с соответствующими нормами векторов):

$$\|A\|_1 = \max_j \sum_{i=1}^{n} |a_{ij}| ,$$

$$\|A\|_2 = \sqrt{\sum_{i,j=1}^{n} a_{ij}^2} ,$$

$$\|A\|_c = \max_i \sum_{j=1}^{n} |a_{ij}| .$$

Норма $\|A\|_1$ — максимальное число среди сумм модулей элементов столбцов матрицы, норма $\|A\|_c$ — максимальное число среди сумм модулей элементов строк матрицы, норма $\|A\|_2$ — сферическая или евклидова. Отметим, что норма $\|A\|_c$ согласована со всеми приведёнными выше нормами векторов.

Введём понятие предела векторов и матриц.

Определение. Рассмотрим последовательность векторов $x^{(1)}, x^{(2)}, \ldots$ с компонентами $x_1^{(1)}, \ldots, x_n^{(1)}, \; x_1^{(2)}, \ldots, x_n^{(2)}, \ldots$. Если существуют пределы:

$$x_i = \lim_{k\to\infty} x_i^{(k)}, \; i = 1, 2, \ldots, n,$$

то говорят, что вектор с компонентами $x_1, x_2, \ldots, x_n$ является *пределом последовательности* $x^{(1)}, x^{(2)}, \ldots$. Аналогично определяется предел последовательности матриц.

Необходимым и достаточным условием сходимости векторов $x^{(k)}$ к x является условие $\|x^{(k)} - x\| \to 0$, при этом $\|x^{(k)}\| \to \|x\|$. Аналогичное утверждение справедливо и для матриц.

Из сходимости в одной из норм следует сходимость и в остальных.

Пример. Для матрицы $A = \begin{pmatrix} -1 & 2 \\ 3 & -5 \end{pmatrix}$ и вектора $b = \begin{pmatrix} 3 \\ -4 \end{pmatrix}$ вычислить различные нормы $\|\cdot\|_1, \|\cdot\|_2, \|\cdot\|_c$. Проверить выполнение условия согласованности норм $\|Ax\| \le \|A\|\|x\|$ для различных комбинаций норм.

Вычислим соответствующие нормы:

$$\|b\|_1 = |3| + |-4| = 7, \; \|b\|_2 = (3^2 + (-4)^2)^{1/2} = 5, \; \|b\|_c = \max(|3|, |-4|) = 4.$$

$$\|A\|_1 = \max(|-1| + |3|, \; |2| + |-5|) = 7,$$

$$\|A\|_2 = (|(-1)|^2 + 3^2 + 2^2 + (-5)^2)^{1/2} = \sqrt{39} \approx 6.245,$$

$$\|A\|_c = \max(|-1| + |2|, \; |3| + |-5|) = 8.$$

Для проверки условия согласованности вычислим различные нормы вектора $c = Ab = \begin{pmatrix} -11 \\ 29 \end{pmatrix}$:

$$\|c\|_1 = |-11| + |29| = 40,$$

$$\|c\|_2 = ((-11)^2 + 29^2)^{1/2} = \sqrt{962} \approx 31.016 ,$$

$$\|c\|_c = \max(|-11|, |29|) = 29 .$$

Легко убедиться в том, что условие согласованности выполняется для согласованных норм:

$$\|c\|_1 = 40 \le \|A\|_1 \|b\|_1 = 7 \cdot 7 = 49 ,$$

$$\|c\|_2 = \sqrt{962} \le \|A\|_2 \|b\|_2 = \sqrt{39} \cdot 5 = \sqrt{975} ,$$

$$\|c\|_c = 29 \le \|A\|_c \|b\|_c = 8 \cdot 4 = 32 .$$

Кроме того, известно, что матричная норма $\|A\|_c$ согласована со всеми введенными выше нормами векторов. В данном примере это подтверждается выполнением неравенств:

$$\|c\|_1 = 40 \le \|A\|_c \|b\|_1 = 8 \cdot 7 = 56 ,$$

$$\|c\|_2 = \sqrt{962} \le \|A\|_c \|b\|_2 = 8 \cdot 5 = 40 .$$

В то же время использование ряда других комбинаций норм матрицы и вектора приводит в данном случае к нарушению условия согласованности:

$$\|c\|_c = 29 > \|A\|_1 \|b\|_c = 7 \cdot 4 = 28 ,$$

$$\|c\|_c = 29 > \|A\|_2 \|b\|_c = \sqrt{39} \cdot 4 .$$

Рассмотренный пример наглядно иллюстрирует важность использования согласованных норм матрицы и вектора.

2. ЧИСЛЕННЫЕ МЕТОДЫ И АЛГОРИТМЫ ЛИНЕЙНОЙ АЛГЕБРЫ

В главе рассматриваются основные численные методы и алгоритмы вычислительной математики в области линейной алгебры. Сначала приводятся численные методы решения систем линейных алгебраических уравнений. Изложены прямые (метод прогонки, метод исключения Гаусса) и итерационные (метод простой итерации, метод Зейделя) методы. В линейной алгебре эту задачу называют первой основной задачей. К ней примыкают задачи вычисления определителя и вычисления элементов обратной матрицы, которые иногда называют второй и третьей основными задачами линейной алгебры. Последние темы этой главы посвящены задаче отыскания собственных значений и собственных векторов матрицы (метод вращений Якоби, метод степенных итераций). Именно эту задачу в линейной алгебре, как правило, называют второй основной задачей.

2.1. СИСТЕМЫ ЛИНЕЙНЫХ АЛГЕБРАИЧЕСКИХ УРАВНЕНИЙ

Пусть требуется найти решение системы линейных алгебраических уравнений (СЛАУ):

$$Ax = b\,,$$

где $A \equiv \left[a_{ij}\right]$ — квадратная матрица размерности $n \times n$ коэффициентов при неизвестных; $x \equiv \left[x_j\right]$ — вектор-столбец неизвестных; $b \equiv \left[b_j\right]$ — вектор-столбец правых частей системы линейных алгебраических уравнений:

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

Системы линейных уравнений можно было бы рассматривать как частный случай систем нелинейных уравнений. Однако относитель-

ная простота линейных систем обусловила появление специальных высокоэффективных методов их решения.

Вместе с тем, линейные системы представляют большой самостоятельный интерес по двум причинам. Во-первых, к системам линейных уравнений приводят многие практические задачи. Во-вторых, важным и неиссякаемым источником систем линейных алгебраических уравнений являются практически все разделы вычислительной математики: нахождение решения системы линейных алгебраических уравнений необходимо в задачах уточнения корней систем нелинейных уравнений, аппроксимации функций, отыскания собственных значений и собственных векторов матрицы, решения обыкновенных дифференциальных уравнений и уравнений в частных производных и т.д. Методы решения систем линейных уравнений используются и для некоторых классов интегральных и интегродифференциальных уравнений.

С точки зрения классической теории линейных алгебраических систем их решение не вызывает затруднений. Простейший аналитический метод решения СЛАУ — метод последовательного исключения неизвестных.

Пример. Решим последовательным исключением неизвестных систему:

$$\begin{cases} x_1 + 3x_2 - 2x_3 = 5 \\ 3x_1 + 5x_2 + 6x_3 = 7 \\ 2x_1 + 4x_2 + 3x_3 = 8 \end{cases} \rightarrow \begin{cases} x_1 = -3x_2 + 2x_3 + 5 \\ 3x_1 + 5x_2 + 6x_3 = 7 \\ 2x_1 + 4x_2 + 3x_3 = 8 \end{cases} \rightarrow \begin{cases} x_1 = -3x_2 + 2x_3 + 5 \\ -4x_2 + 12x_3 = -8 \\ -2x_2 + 7x_3 = -2 \end{cases} \rightarrow$$

$$\rightarrow \begin{cases} x_1 = -3x_2 + 2x_3 + 5 \\ x_2 = 3x_3 + 2 \\ -2x_2 + 7x_3 = -2 \end{cases} \rightarrow \begin{cases} x_1 = -3x_2 + 2x_3 + 5 \\ x_2 = 3x_3 + 2 \\ x_3 = 2 \end{cases} \rightarrow \begin{cases} x_1 = -3x_2 + 2x_3 + 5 \\ x_2 = 8 \\ x_3 = 2 \end{cases} \rightarrow$$

$$\rightarrow \begin{cases} x_1 = -15 \\ x_2 = 8 \\ x_3 = 2 \end{cases}.$$

Помимо этого, из линейной алгебры известно, что по правилу Крамера система из n линейных алгебраических уравнений с n неизвестными имеет единственное решение, если определитель матрицы системы отличен от нуля ($\det A \neq 0$), и значение каждого из неизвестных вычисляется как отношение двух определителей порядка n:

$$x_j = \frac{\det A_j}{\det A}, \quad j = 1, \ldots, n,$$

где $\det A_j$ — определитель матрицы, получаемой заменой j-го столбца матрицы A столбцом правых частей.

Пример. Решим методом Крамера систему:

$$\begin{cases} x_1 + 3x_2 - 2x_3 = 5 \\ 3x_1 + 5x_2 + 6x_3 = 7 \\ 2x_1 + 4x_2 + 3x_3 = 8 \end{cases}.$$ Здесь $A = \begin{pmatrix} 1 & 3 & -2 \\ 3 & 5 & 6 \\ 2 & 4 & 3 \end{pmatrix}$, $b = \begin{pmatrix} 5 \\ 7 \\ 8 \end{pmatrix}$, тогда:

$$x_1 = \frac{\begin{vmatrix} 5 & 3 & -2 \\ 7 & 5 & 6 \\ 8 & 4 & 3 \end{vmatrix}}{\begin{vmatrix} 1 & 3 & -2 \\ 3 & 5 & 6 \\ 2 & 4 & 3 \end{vmatrix}} = \frac{60}{-4} = -15, \quad x_2 = \frac{\begin{vmatrix} 1 & 5 & -2 \\ 3 & 7 & 6 \\ 2 & 8 & 3 \end{vmatrix}}{\begin{vmatrix} 1 & 3 & -2 \\ 3 & 5 & 6 \\ 2 & 4 & 3 \end{vmatrix}} = \frac{-32}{-4} = 8, \quad x_3 = \frac{\begin{vmatrix} 1 & 3 & 5 \\ 3 & 5 & 7 \\ 2 & 4 & 8 \end{vmatrix}}{\begin{vmatrix} 1 & 3 & -2 \\ 3 & 5 & 6 \\ 2 & 4 & 3 \end{vmatrix}} = \frac{-8}{-4} = 2.$$

Однако при непосредственном вычислении определителей как алгебраической суммы $n!$ произведений элементов для отыскания решения системы линейных алгебраических уравнений по правилу Крамера потребуется приблизительно $n \cdot n!$ арифметических операций типа умножения, что алгоритмически предельно неэффективно.

Поэтому далее будут рассмотрены наиболее употребительные прямые и итерационные методы. Прямые методы дают решение задачи за конечное (точно определяемое для каждого метода) число операций. Здесь намеренно не употребляется термин «точные методы», так как из-за ошибок округления, имеющих место в любых ЭВМ при вычислениях с конечным числом знаков, точное решение не может быть достигнуто и всегда получается с погрешностями. Из прямых

методов будут рассмотрены метод прогонки для СЛАУ с трёхдиагональной матрицей и метод Гаусса с его модификациями для системы линейных алгебраических уравнений общего вида.

Итерационные методы дают решение как предел бесконечной последовательности приближённых решений, в которых каждое последующее более точное приближение находится по уже найденному предыдущему (или предыдущим) решениям. Из итерационных методов решения систем линейных алгебраических уравнений будут рассмотрены метод простых итераций и метод Зейделя.

2.2. ОБУСЛОВЛЕННОСТЬ МАТРИЦЫ

Другое важное обстоятельство, связанное с решением систем линейных алгебраических уравнений, состоит в следующем. С точки зрения теории линейных систем различаются два случая: когда определитель матрицы системы не равен нулю ($\det A \neq 0$), т.е. СЛАУ является невырожденной, и когда определитель матрицы системы равен нулю ($\det A = 0$) — вырожденная система. Во втором случае система либо не имеет решения (при $b \neq 0$), либо имеет бесконечное множество решений (при $b = 0$). Однако с точки зрения практических вычислений существуют «почти вырожденные» СЛАУ — системы уравнений, у которых определитель близок к нулю, но отличен от нуля ($\det A \approx 0$). Небольшие изменения коэффициентов матрицы системы или правых частей системы в «почти вырожденных» системах могут привести к большим погрешностям решения.

Все эти случаи хорошо иллюстрируются на примере решения системы двух линейных уравнений:

$$\begin{cases} a_{11}x_1 + a_{12}x_2 = b_1 \\ a_{21}x_1 + a_{22}x_2 = b_2 \end{cases}.$$

На рис. 1 каждому из уравнений соответствует прямая на плоскости (x_1, x_2), а точка пересечения этих прямых есть решение системы.

Если $\det A = 0$, то наклоны прямых равны и они либо параллельны, либо совпадают. При $\det A \approx 0$ небольшие погрешности в коэффициентах и правых частях могут привести к большим погрешностям в решении, т.е. к положению точки пересечения.

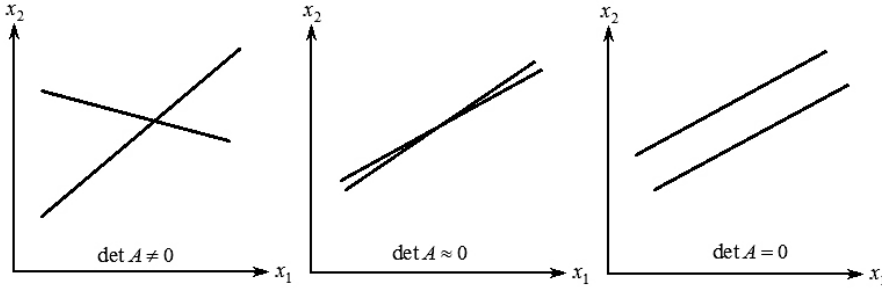

Рис. 1. Геометрическая интерпретация системы двух линейных уравнений

Системы такого типа, в которых малые погрешности в коэффициентах системы или в правых частях (эти погрешности могут быть, в частности, результатом округлений при вычислениях или записи чисел в память ЭВМ) приводят к большим погрешностям в решении, называются плохо обусловленными. Плохо обусловленная система геометрически соответствует почти параллельным прямым.

Определение. Для исследования погрешностей, возникающих при решении СЛАУ, вводят понятие *числа обусловленности* матрицы $cond(A)$:

$$cond(A) = \|A\| \cdot \|A^{-1}\| .$$

При любой норме $cond(A) \geq 1$:

$$cond(A) = \|A\| \cdot \|A^{-1}\| \geq \|A \cdot A^{-1}\| = \|E\| = 1 .$$

Пример. Вычислим число обусловленности матрицы $A = \begin{pmatrix} -1 & 2 \\ 3 & -5 \end{pmatrix}$, взяв в качестве нормы матрицы $\|\cdot\|_c$.

Во-первых, $\|A\|_c = \max(|-1| + |2|,\ |3| + |-5|) = 8$.

Во-вторых, найдём обратную матрицу и вычислим её норму:

$$A^{-1} = \begin{pmatrix} 5 & 2 \\ 3 & 1 \end{pmatrix},\ \|A^{-1}\|_c = \max(|5| + |2|,\ |3| + |1|) = 7 .$$

В результате: $cond(A) = \|A\|_c \cdot \|A^{-1}\|_c = 8 \cdot 7 = 56$.

Число обусловленности характеризует степень зависимости относительной погрешности решения СЛАУ от относительной погреш-

ности входных данных (правые части и элементы матрицы). Можно показать, что для ненулевых векторов x справедливы следующие неравенства:

$$\frac{\|\Delta x\|}{\|x\|} \le cond(A)\frac{\|\Delta b\|}{\|b\|}, \quad \frac{\|\Delta x\|}{\|x\|} \le cond(A)\frac{\|\Delta A\|}{\|A+\Delta A\|}.$$

Чем больше это число, тем хуже обусловленность системы; так, при $cond(A) >> 10^2$ система уже плохо обусловлена. Таким образом, чем больше число обусловленности, тем сильнее влияние погрешности входных данных на конечный результат решения СЛАУ. На практике, однако, вычислители часто ограничиваются проверкой условия $\det A \approx 0$.

Пример. Вычислим число обусловленности матрицы $A = \begin{pmatrix} 1 & 10 \\ 100 & 1001 \end{pmatrix}$, взяв в качестве нормы матрицы $\|\cdot\|_c$.

Легко видеть, что $\det A = 1$.

При этом $\|A\|_c = \max(|1|+|10|,\ |100|+|1001|) = 1101$.

Найдём обратную матрицу и вычислим её норму:

$$A^{-1} = \begin{pmatrix} 1001 & -10 \\ -100 & 1 \end{pmatrix},\ \left\|A^{-1}\right\|_c = \max(|1001|+|-10|,\ |-100|+|1|) = 1011.$$

В результате: $cond(A) = \|A\|_c \cdot \left\|A^{-1}\right\|_c = 1101 \cdot 1011 = 1113111 > 10^6$.

Решим СЛАУ $Ax = b$, где $b = \begin{pmatrix} 11 \\ 1101 \end{pmatrix}$. Несложно показать, что $x = \begin{pmatrix} 1 \\ 1 \end{pmatrix}$.

Внесём небольшое возмущение в вектор правых частей исходной системы уравнений: $\tilde{b} = \begin{pmatrix} 11.01 \\ 1101 \end{pmatrix}$. Теперь новое решение $\tilde{x} = \begin{pmatrix} 11.01 \\ 0 \end{pmatrix}$.

Легко видеть, что для данной плохо обусловленной СЛАУ небольшое возмущение входных данных привело к существенному изменению решения. И действительно:

$$\frac{\|\Delta x\|}{\|x\|} = \frac{10.01}{1} = 10.01 \quad \le \quad cond(A)\frac{\|\Delta b\|}{\|b\|} = 1113111 \cdot \frac{0.01}{1101} = 10.11.$$

Итак, изменив входные данные (в данном случае вектор правых частей) менее чем на 0.001%, можно получить изменение выходных

данных (решения СЛАУ) более чем в 10 раз от исходных, если изначальная матрица коэффициентов плохо обусловлена.

2.3. МЕТОД ПРОГОНКИ

Метод прогонки (или алгоритм Томаса) принадлежит к числу прямых методов решения СЛАУ и используется в тех случаях, когда многие коэффициенты матрицы равны нулю. Это обстоятельство учтено при реализации метода прогонки, в котором исключаются преобразования с нулевыми элементами.

Системы линейных алгебраических уравнений с трёхдиагональной матрицей коэффициентов при неизвестных являются наиболее важным и распространённым случаем систем специального вида. В таких системах отличны от нуля только элементы, лежащие на главной диагонали и на нижней и верхней диагоналях, прилегающих к ней. К системам с трёхдиагональными матрицами приводят, например, задачи о сплайн-интерполяции, о решении разностными методами обыкновенных дифференциальных уравнений и уравнений в частных производных. Метод прогонки является одним из эффективных прямых методов решения СЛАУ с трёхдиагональными матрицами.

Рассмотрим следующую СЛАУ с трёхдиагональной матрицей A:

$$a_1 = 0 \begin{cases} b_1 x_1 + c_1 x_2 = d_1 \\ a_2 x_1 + b_2 x_2 + c_2 x_3 = d_2 \\ \qquad a_3 x_2 + b_3 x_3 + c_3 x_4 = d_3 \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ \qquad\qquad a_{n-1} x_{n-2} + b_{n-1} x_{n-1} + c_{n-1} x_n = d_{n-1} \\ \qquad\qquad\qquad a_n x_{n-1} + b_n x_n = d_n, \qquad c_n = 0 \end{cases},$$

решение которой будем искать в рекуррентном виде (когда каждый член последовательности выражается через один или несколько предыдущих):

$$x_i = P_i x_{i+1} + Q_i,\ i = 1, \ldots, n,$$

где $P_i, Q_i,\ i = 1, \ldots, n$ — прогоночные коэффициенты, подлежащие определению.

Для их определения сначала выразим из первого уравнения СЛАУ x_1 через x_2, получим:

$$x_1 = \frac{-c_1}{b_1} x_2 + \frac{d_1}{b_1} = P_1 x_2 + Q_1, \text{ откуда } P_1 = \frac{-c_1}{b_1}, Q_1 = \frac{d_1}{b_1}.$$

Из второго уравнения СЛАУ, подставив x_1, выразим x_2 через x_3, получим:

$$b_2 x_2 = -a_2 x_1 - c_2 x_3 + d_2 = -a_2 (P_1 x_2 + Q_1) - c_2 x_3 + d_2,$$

$$b_2 x_2 + a_2 P_1 x_2 = -a_2 Q_1 - c_2 x_3 + d_2,$$

$$x_2 = \frac{-c_2}{b_2 + a_2 P_1} x_3 + \frac{d_2 - a_2 Q_1}{b_2 + a_2 P_1} = P_2 x_3 + Q_2,$$

откуда $P_2 = \frac{-c_2}{b_2 + a_2 P_1}, Q_2 = \frac{d_2 - a_2 Q_1}{b_2 + a_2 P_1}$.

Продолжая этот процесс по аналогии (меняются только индексы), получим из i-го уравнения СЛАУ:

$$x_i = \frac{-c_i}{b_i + a_i P_{i-1}} x_{i+1} + \frac{d_i - a_i Q_{i-1}}{b_i + a_i P_{i-1}},$$

следовательно $P_i = \frac{-c_i}{b_i + a_i P_{i-1}}, Q_i = \frac{d_i - a_i Q_{i-1}}{b_i + a_i P_{i-1}}$.

Из последнего уравнения СЛАУ имеем:

$$x_n = \frac{d_n - a_n Q_{n-1}}{b_n + a_n P_{n-1}} = 0 \cdot x_{n+1} + Q_n, \text{ т.е. } P_n = 0 \ (\text{т.к. } c_n = 0),$$

$$Q_n = \frac{d_n - a_n Q_{n-1}}{b_n + a_n P_{n-1}} = x_n.$$

Таким образом, прямой ход метода прогонки по определению прогоночных коэффициентов $P_i, Q_i, i = 1, \ldots, n$ завершён. В результате прогоночные коэффициенты вычисляются по следующим формулам:

$$P_1 = \frac{-c_1}{b_1}, Q_1 = \frac{d_1}{b_1}, \text{ т.к. } a_1 = 0, i = 1;$$

$$P_i = \frac{-c_i}{b_i + a_i P_{i-1}}, Q_i = \frac{d_i - a_i Q_{i-1}}{b_i + a_i P_{i-1}}, i = 2, \ldots, n-1;$$

$$P_n = 0\ ,\ \text{т.к. } c_n = 0\ ,\ Q_n = \frac{d_n - a_n Q_{n-1}}{b_n + a_n P_{n-1}},\ i = n\ .$$

Обратный ход метода прогонки для нахождения решения системы осуществляется в соответствии с выражением $x_i = P_i x_{i+1} + Q_i, i = 1,\ldots,n$:

$$\begin{cases} x_n = P_n x_{n+1} + Q_n = 0 \cdot x_{n+1} + Q_n = Q_n \\ x_{n-1} = P_{n-1} x_n + Q_{n-1} \\ x_{n-2} = P_{n-2} x_{n-1} + Q_{n-2} \\ \ldots\ldots\ldots\ldots\ldots\ldots \\ x_1 = P_1 x_2 + Q_1 \end{cases}.$$

Приведённые формулы — формулы правой прогонки. Аналогично, начиная вывод прогоночных коэффициентов с последнего уравнения СЛАУ, можно вывести формулы левой прогонки.

Общее число совершаемых арифметических операций в методе прогонки равно $8n$, т.е. пропорционально числу уравнений.

Подобные численные методы, для которых число арифметических операций пропорционально размерности n, называют *экономичными*.

Отметим, что $\det A = \prod_{i=1}^{n} [\, a_i P_i - b_i \,]$.

Метод прогонки устойчив, если $|P_i| \le 1,\ i = 1,\ldots,n$. Из чего можно доказать, что для вычислительной устойчивости метода прогонки достаточно выполнения следующих условий:

$$a_i \ne 0\ ,\ c_i \ne 0\ ,\ i = 2,\ldots,n-1\ ;\ |b_i| \ge |a_i| + |c_i|\ ,\ i = 1,\ldots,n\ ,$$

причём строгое неравенство имеет место хотя бы при одном i (условие диагонального преобладания матрицы).

Здесь устойчивость понимается в смысле ненакопления погрешности решения в ходе вычислительного процесса при округлении или при малых погрешностях входных данных (правых частей и элементов матрицы).

Пример. Методом прогонки решить СЛАУ:

$$\begin{cases} 7x_1 - 3x_2 = 1 \\ -4x_1 + 9x_2 + 3x_3 = 23 \\ 3x_2 - 8x_3 + 4x_4 = -2 \\ -2x_3 + 7x_4 + 4x_5 = 42 \\ -5x_4 + 6x_5 = 10 \end{cases}.$$ Здесь $a = \begin{pmatrix} 0 \\ -4 \\ 3 \\ -2 \\ -5 \end{pmatrix}$, $b = \begin{pmatrix} 7 \\ 9 \\ -8 \\ 7 \\ 6 \end{pmatrix}$, $c = \begin{pmatrix} -3 \\ 3 \\ 4 \\ 4 \\ 0 \end{pmatrix}$, $d = \begin{pmatrix} 1 \\ 23 \\ -2 \\ 42 \\ 10 \end{pmatrix}$.

Осуществим прямой ход метода прогонки:

$$P_1 = \frac{-c_1}{b_1} = \frac{3}{7} = 0.429\,,\ Q_1 = \frac{d_1}{b_1} = \frac{1}{7} = 0.143\,;$$

$$P_2 = \frac{-c_2}{b_2 + a_2 P_1} = -0.412\,,\ Q_2 = \frac{d_2 - a_2 Q_1}{b_2 + a_2 P_1} = 3.235\,;$$

$$P_3 = \frac{-c_3}{b_3 + a_3 P_2} = 0.433\,,\ Q_3 = \frac{d_3 - a_3 Q_2}{b_3 + a_3 P_2} = 1.268\,;$$

$$P_4 = \frac{-c_4}{b_4 + a_4 P_3} = -0.652\,,\ Q_4 = \frac{d_4 - a_4 Q_3}{b_4 + a_4 P_3} = 7.261\,;$$

$$P_5 = 0\ (c_5 = 0)\,,\ Q_5 = \frac{d_5 - a_5 Q_4}{b_5 + a_5 P_4} = 5.0\,.$$

Осуществим обратный ход метода прогонки:

$$x_5 = Q_5 = 5.0\,,\ x_4 = P_4 x_5 + Q_4 = 4.0\,,\ x_3 = P_3 x_4 + Q_3 = 3.0\,,$$

$$x_2 = P_2 x_3 + Q_2 = 2.0\,,\ x_1 = P_1 x_2 + Q_1 = 1.0\,.$$

Решение СЛАУ: $x = (1 \quad 2 \quad 3 \quad 4 \quad 5)^T$.

2.4. ПРОГРАММА #01

Ниже предлагается вариант алгоритма программы для решения СЛАУ с трёхдиагональной матрицей методом прогонки.

```
АЛГОРИТМ «Метод прогонки»
ВХОД        n, a[n], b[n], c[n], d[n]
ВЫХОД       i, p[n], q[n], x[n]
НАЧАЛО
    p[1]:=-c[1]/b[1]
    q[1]:=d[1]/b[1]
    ЦИКЛ «Прямой ход» ДЛЯ i ОТ 2 ДО n ПО 1
            p[i]:=-c[i]/(b[i]+a[i]*p[i-1])
            q[i]:=(d[i]-a[i]*q[i-1])/(b[i]+a[i]*p[i-1])
```

```
        x[n]:=q[n]
        ЦИКЛ «Обратный ход» ДЛЯ i ОТ n-1 ДО 1 ПО -1
                x[i]:=p[i]*x[i+1]+q[i]
        ПЕЧАТЬ x
КОНЕЦ
```

2.5. МЕТОД ИСКЛЮЧЕНИЯ ГАУССА

Известно большое число схем метода исключения Гаусса для нахождения решения системы линейных алгебраических уравнений $Ax = b$, приспособленных для ручного или машинного счёта матриц общего или специального вида, где $A \equiv \left[a_{ij}\right]$ — квадратная матрица размерности $n \times n$ коэффициентов при неизвестных; $x \equiv \left[x_j\right]$ — вектор-столбец неизвестных; $b \equiv \left[b_j\right]$ — вектор-столбец правых частей системы.

Таким образом, дана следующая СЛАУ:

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

Метод Гаусса можно интерпретировать как метод, в котором первоначально матрица A с помощью равносильных преобразований для СЛАУ (обмен строк, домножение строки на число, сложение строк и замена одной из них на результат) приводится к верхнетреугольной форме (прямой ход), а далее — к единичной (обратный ход). В обратном ходе определяются неизвестные: очевидно, если матрица СЛАУ единичная, то $x_i = b_i,\ i = 1,\ldots,n$.

Запишем расширенную матрицу системы:

$$\begin{array}{l} \\ \textit{Ведущая строка} \rightarrow \\ \\ \\ \\ \\ \uparrow \textit{Ведущий столбец} \end{array} \begin{array}{c} \overset{x_1 \quad x_2 \quad x_3 \quad \cdots \quad x_n \quad b}{\overline{}} \\ \left[\begin{array}{c|cccccc} a_{11} & a_{12} & a_{13} & \cdots & a_{1n} & b_1 \\ \hline a_{21} & a_{22} & a_{23} & \cdots & a_{2n} & b_2 \\ a_{31} & a_{32} & a_{33} & \cdots & a_{3n} & b_3 \\ \ldots & \ldots & \ldots & \ldots & \ldots & \ldots \\ a_{n1} & a_{n2} & a_{n3} & \cdots & a_{nn} & b_n \end{array}\right] \end{array} \left(-\frac{a_{21}}{a_{11}}\right);\left(-\frac{a_{31}}{a_{11}}\right);\ldots;\left(-\frac{a_{n1}}{a_{11}}\right)$$

$$\xrightarrow[1\text{-й}\ \ шаг]{} .$$

На первом шаге алгоритма Гаусса выберем диагональный элемент $a_{11} \neq 0$ (если он равен 0, то первую строку переставляем с какой-либо нижележащей строкой, где $a_{i1} \neq 0$, $i = 2,\ldots,n$) и объявляем его *ведущим*, а соответствующую строку и столбец, на пересечении которых он стоит — *ведущими*. Обнулим элементы $a_{21},\ldots,a_{n1}$ ведущего столбца. Для этого сформируем числа $(-a_{21}/a_{11}); (-a_{31}/a_{11}); \ldots; (-a_{n1}/a_{11})$, которые соотнесём с ведущей строкой. Умножая ведущую строку на число $(-a_{21}/a_{11})$, складывая со второй и ставя результат на место второй строки, получим вместо элемента a_{21} нуль, а вместо элементов a_{2j}, $j = 2,\ldots,n$, b_2 соответственно элементы $a^1_{2j} = a_{2j} + a_{1j}(-a_{21}/a_{11})$, $j = 2,\ldots,n$, $b^1_2 = b_2 + b_1(-a_{21}/a_{11})$. Продолжаем так вплоть до последней строки: умножая ведущую строку на число $(-a_{n1}/a_{11})$, складывая с n-й строкой и ставя результат на место n-й строки, получим вместо элемента a_{n1} нуль, а остальные элементы этой строки будут иметь вид $a^1_{nj} = a_{nj} + a_{1j}(-a_{n1}/a_{11})$, $b^1_n = b_n + b_1(-a_{n1}/a_{11})$.

Сохраняя на первом шаге ведущую первую строку неизменной, получим в результате первого шага алгоритма Гаусса следующую матрицу:

$$\begin{array}{l} \\ \textit{Ведущая строка} \rightarrow \\ \\ \\ \\ \\ \\ \uparrow \textit{Ведущий столбец} \end{array} \begin{array}{c} \begin{array}{cccccc} x_1 & x_2 & x_3 & \cdots & x_n & b \end{array} \\ \left[\begin{array}{c|c|ccc|c} a_{11} & a_{12} & a_{13} & \cdots & a_{1n} & b_1 \\ \hline 0 & a^1_{22} & a^1_{23} & \cdots & a^1_{2n} & b^1_2 \\ \hline 0 & a^1_{32} & a^1_{33} & \cdots & a^1_{3n} & b^1_3 \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ 0 & a^1_{n2} & a^1_{n3} & \cdots & a^1_{nn} & b^1_n \end{array}\right] \end{array} \begin{array}{l} \left(-\dfrac{a^1_{32}}{a^1_{22}}\right); \ldots; \left(-\dfrac{a^1_{n2}}{a^1_{22}}\right) \\ \xrightarrow[2\text{-й шаг}]{} \end{array}.$$

На втором шаге алгоритма Гаусса в качестве ведущего элемента выбирается элемент $a^1_{22} \neq 0$ (если он равен нулю, то вторую строку взаимно меняем на нижележащую строку, где $a_{i1} \neq 0$, $i = 3,\ldots,n$). Формируются числа $\left(-\dfrac{a^1_{32}}{a^1_{22}}\right); \ldots; \left(-\dfrac{a^1_{n2}}{a^1_{22}}\right)$, которые ставятся около ведущей строки. Умножая ведущую строку на число $\left(-\dfrac{a^1_{32}}{a^1_{22}}\right)$ и складывая результат с третьей строкой, получим вместо элемента a^1_{32} нуль, а вместо эле-

ментов a_{3j}^1, $j = 3,\ldots,n$, b_3^1 — элементы $a_{3j}^2 = a_{3j}^1 + a_{2j}^1\left(-\dfrac{a_{32}^1}{a_{22}^1}\right)$, $j = 3\ldots n$, $b_3^2 = b_3^1 + b_2^1\left(-\dfrac{a_{32}^1}{a_{22}^1}\right)$. И так далее вплоть до последней строки: умножая ведущую строку на число $\left(-\dfrac{a_{n2}^1}{a_{22}^1}\right)$, складывая результат с n-й строкой и ставя полученную сумму на место n-й строки, получим вместо элемента a_{n2}^1 нуль, а вместо элементов a_{nj}^1, b_n^1 — элементы $a_{nj}^2 = a_{nj}^1 + a_{2j}^1\left(-\dfrac{a_{n2}^1}{a_{22}^1}\right)$, $j = 3,\ldots,n$, $b_n^2 = b_n^1 + b_2^1\left(-\dfrac{a_{n2}^1}{a_{22}^1}\right)$.

Сохраняя на втором шаге первую и ведущую вторую строки матрицы неизменными, получим в результате второго шага алгоритма Гаусса следующую матрицу:

$$\begin{array}{c} \begin{array}{cccccc} x_1 & x_2 & x_3 & \cdots & x_n & b \end{array} \\ \left[\begin{array}{cc|c|ccc} a_{11} & a_{12} & a_{13} & \cdots & a_{n1} & b_1 \\ 0 & a_{22}^1 & a_{23}^1 & \cdots & a_{2n}^1 & b_2^1 \\ \hline 0 & 0 & a_{33}^2 & \cdots & a_{3n}^2 & b_3^2 \\ \hline \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & a_{n3}^2 & \cdots & a_{nn}^2 & b_n^2 \end{array}\right] \end{array} \xrightarrow{3\text{-й шаг}} \ldots \xrightarrow{(n-1)\text{-й шаг}}.$$

Ведущая строка →

↑ *Ведущий столбец*

В результате после $(n-1)$-го шага алгоритма Гаусса получаем следующую расширенную матрицу, содержащую верхнетреугольную матрицу СЛАУ:

$$\begin{array}{c} \begin{array}{cccccc} x_1 & x_2 & x_3 & \ldots & x_n & b \end{array} \\ \left[\begin{array}{cccccc} a_{11} & a_{12} & a_{13} & \ldots & a_{1n} & b_1 \\ 0 & a_{22}^1 & a_{23}^1 & \ldots & a_{2n}^1 & b_2^1 \\ 0 & 0 & a_{33}^2 & \ldots & a_{3n}^2 & b_3^2 \\ \ldots & \ldots & \ldots & \ldots & \ldots & \ldots \\ 0 & 0 & 0 & \ldots & a_{nn}^{n-1} & b_n^{n-1} \end{array}\right] \end{array}.$$

Прямой ход алгоритма Гаусса завершён.

Если элементы какой-либо строки матрицы системы в результате преобразований стали равными нулю, а правая часть не равна нулю, то СЛАУ несовместна и решений не имеет. Если элементы какой-либо строки матрицы системы и правая часть в результате преобразований

стали равными нулю, то СЛАУ совместна, но имеет бесконечное множество решений.

Определитель треугольной матрицы равен произведению диагональных элементов. В результате выполнения прямого хода метода исключения Гаусса система линейных уравнений приводится к верхнетреугольной матрице. Следовательно, в результате прямого хода метода Гаусса сразу можно вычислить определитель матрицы A исходной СЛАУ как произведение ведущих элементов:

$$\det A = (-1)^p a_{11} a_{22}^1 a_{33}^2 \cdot \ldots \cdot a_{nn}^{n-1},$$

где p — количество перестановок строк в процессе прямого хода.

В обратном ходе алгоритма Гаусса полученную после прямого хода расширенную матрицу системы возможно аналогичными эквивалентными преобразованиями (если обратить в нуль и все коэффициенты, лежащие выше главной диагонали) привести к диагональной, а затем (если разделить каждое уравнение на соответствующий элемент, стоящий на главной диагонали) и к единичной, что позволит получить решение исходной СЛАУ.

Однако более эффективным способом является последовательная запись решения системы через рекуррентные соотношения, когда из последнего уравнения сразу определяется x_n, из предпоследнего — x_{n-1} и т.д. пока из первого уравнения не определится x_1:

$$\begin{cases} a_{nn}^{n-1} x_n = b_n^{n-1} & \Rightarrow x_n \\ a_{n-1n-1}^{n-2} x_{n-1} + a_{n-1n}^{n-2} x_n = b_{n-1}^{n-2} & \Rightarrow x_{n-1} \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots & \\ a_{11} x_1 + \ldots + a_{1n} x_n = b_1 & \Rightarrow x_1 \end{cases},$$

$$\begin{cases} x_n = \dfrac{b_n}{a_{nn}} \\ x_i = \dfrac{1}{a_{ii}} \left[b_i - \sum\limits_{k=i+1}^{n} a_{ik} x_k \right], \; i = n-1, \ldots, 1 \end{cases}.$$

Важно отметить, что для реализации метода Гаусса требуется порядка n^3 арифметических операций, что относит данный метод к *затратным*. При этом оценка числа операций определяется в основном операциями, затрачиваемыми при выполнении прямого хода метода Гаусса. Обратный ход метода Гаусса (через рекуррент-

ные соотношения) требует порядка n^2 операций. Следовательно, если требуется решить несколько систем линейных алгебраических уравнений $Ax = b$ с одной и той же матрицей и различными правыми частями, то в этом случае целесообразно реализовать алгоритм метода Гаусса в виде двух подпрограмм: первая подпрограмма должна реализовывать прямой ход алгоритма и получать на выходе верхнетреугольную матрицу СЛАУ, а вторая подпрограмма должна реализовывать прямой ход алгоритма для правых частей и, используя полученную верхнетреугольную матрицу, вычислять решение системы для конкретной правой части.

2.6. ОБРАЩЕНИЕ МАТРИЦЫ МЕТОДОМ ГАУССА

Метод исключения Гаусса можно применить и для обращения невырожденной ($\det A \neq 0$) матрицы. Действительно, пусть требуется обратить невырожденную матрицу A. Тогда, сделав обозначение $A^{-1} = X$, можно выписать матричное уравнение $AX = E$, где E — единичная матрица, на основе которого можно записать цепочку СЛАУ:

$$A \cdot \begin{pmatrix} x_{11} \\ x_{21} \\ \dots \\ x_{n1} \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ \dots \\ 0 \end{pmatrix}, \; A \cdot \begin{pmatrix} x_{12} \\ x_{22} \\ \dots \\ x_{n2} \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \\ \dots \\ 0 \end{pmatrix}, \; \dots, \; A \cdot \begin{pmatrix} x_{1n} \\ x_{2n} \\ \dots \\ x_{nn} \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ \dots \\ 1 \end{pmatrix},$$

каждую из которых можно решить методом Гаусса.

Таким образом, для обращения матрицы необходимо решить n систем линейных уравнений n-го порядка с одинаковой матрицей A и различными правыми частями. Так как верхнетреугольная матрица для всех этих СЛАУ будет одной и той же, то прямой ход метода Гаусса применяется для матрицы A только один раз.

Строится следующая расширенная матрица:

$$\left.\begin{array}{cccc|cccc} x_{1n} & x_{2n} & \dots & x_{nn} & & & & \\ \dots & \dots & \dots & \dots & & & & \\ x_{12} & x_{22} & \dots & x_{n2} & & & & \\ x_{11} & x_{21} & \dots & x_{n1} & b^1 & b^2 & \dots & b^n \\ \hline a_{11} & a_{12} & \dots & a_{1n} & 1 & 0 & \dots & 0 \\ a_{21} & a_{22} & \dots & a_{2n} & 0 & 1 & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ a_{n1} & a_{n2} & \dots & a_{nn} & 0 & 0 & \dots & 1 \end{array}\right| .$$

В результате применения $(n-1)$-го шага метода Гаусса получаем:

$$\left.\begin{array}{cccc|c} x_{1n} & x_{2n} & \dots & x_{nn} & \\ \dots & \dots & \dots & \dots & \\ x_{12} & x_{22} & \dots & x_{n2} & \\ x_{11} & x_{21} & \dots & x_{n1} & b^1 \quad b^2 \quad \dots \quad b^n \\ \hline a_{11} & a_{12} & \dots & a_{1n} & b_{11} \quad b_{12} \quad \dots \quad b_{1n} \\ 0 & a_{22}^1 & \dots & a_{2n}^1 & b_{21}^1 \quad b_{22}^1 \quad \dots \quad b_{2n}^1 \\ \dots & \dots & \dots & \dots & \dots \quad \dots \quad \dots \quad \dots \\ 0 & 0 & \dots & a_{nn}^{n-1} & b_{n1}^{n-1} \quad b_{n2}^{n-1} \quad \dots \quad b_{nn}^{n-1} \end{array}\right| .$$

При этом первый столбец $(x_{11} \quad x_{21} \quad \dots \quad x_{n1})^T$ обратной матрицы определяется в обратном ходе метода Гаусса с правой частью b^1, второй столбец $(x_{12} \quad x_{22} \quad \dots \quad x_{n2})^T$ — с правой частью b^2 и так далее. Столбец $(x_{1n} \quad x_{2n} \quad \dots \quad x_{nn})^T$ определяется с правой частью b^n.

Как альтернатива, впрочем, возможно в обратном ходе метода Гаусса эквивалентными преобразованиями привести всю матрицу A из верхнетреугольной непосредственно к единичной, чтобы сразу получить её обратную матрицу, уже не теряя в алгоритмической эффективности.

Пример. Методом Гаусса решить СЛАУ, вычислить определитель и обратную матрицу для матрицы СЛАУ:

$$\begin{cases} x_1 + 3x_2 - x_3 + 2x_4 = 13 \\ 6x_1 - 2x_2 + 2x_4 = 20 \\ 3x_1 - 5x_2 + x_3 + 8x_4 = 7 \\ -x_1 + 4x_2 - 5x_3 + 9x_4 = 7 \end{cases}; \quad A = \begin{pmatrix} 1 & 3 & -1 & 2 \\ 6 & -2 & 0 & 2 \\ 3 & -5 & 1 & 8 \\ -1 & 4 & -5 & 9 \end{pmatrix}, \quad b = \begin{pmatrix} 13 \\ 20 \\ 7 \\ 7 \end{pmatrix}.$$

Прямой ход:

$$\left.\begin{array}{cccc|c} x_1 & x_2 & x_3 & x_4 & b \\ \mathbf{1} & 3 & -1 & 2 & 13 \\ 6 & -2 & 0 & 2 & 20 \\ 3 & -5 & 1 & 8 & 7 \\ -1 & 4 & -5 & 9 & 7 \end{array}\right) (-6/1); (-3/1); (1/1); \xrightarrow[1\text{-й} \quad шаг]{}$$

$$\left.\begin{array}{cccc|c} x_1 & x_2 & x_3 & x_4 & b \\ 1 & 3 & -1 & 2 & 13 \\ 0 & \mathbf{-20} & 6 & -10 & -58 \\ 0 & -14 & 4 & 2 & -32 \\ 0 & 7 & -6 & 11 & 20 \end{array}\right) (-14/20); (7/20); \xrightarrow[2\text{-й} \quad шаг]{}$$

$$\begin{array}{cccc|c} x_1 & x_2 & x_3 & x_4 & b \\ \left(\begin{array}{cccc|c} 1 & 3 & -1 & 2 & 13 \\ 0 & -20 & 6 & -10 & -58 \\ 0 & 0 & \mathbf{-1/5} & 9 & 43/5 \\ 0 & 0 & -39/10 & 15/2 & -3/10 \end{array}\right) & & & & (-39/2); \xrightarrow[3\text{-й шаг}]{} \end{array}$$

$$\begin{array}{cccc|c} x_1 & x_2 & x_3 & x_4 & b \\ \left(\begin{array}{cccc|c} 1 & 3 & -1 & 2 & 13 \\ 0 & -20 & 6 & -10 & -58 \\ 0 & 0 & -1/5 & 9 & 43/5 \\ 0 & 0 & 0 & -168 & -168 \end{array}\right) & & & & . \end{array}$$

Определитель матрицы: $\det A = 1 \cdot (-20) \cdot (-1/5) \cdot (-168) = -672$.
Обратный ход:

$$\begin{cases} -168x_4 = -168 \\ (-1/5)x_3 + 9x_4 = 43/5 \\ -20x_2 + 6x_3 - 10x_4 = -58 \\ x_1 + 3x_2 - x_3 + 2x_4 = 13 \end{cases}; \quad \begin{cases} x_4 = 1 \\ (-1/5)x_3 = 43/5 - 9 \cdot 1 = -2/5, \; x_3 = 2 \\ -20x_2 = -58 - 6 \cdot 2 + 10 \cdot 1 = -60, \; x_2 = 3 \\ x_1 = 13 - 3 \cdot 3 + 2 - 2 \cdot 1 = 4, \; x_1 = 4 \end{cases}.$$

Решение СЛАУ: $x = (4 \quad 3 \quad 2 \quad 1)^T$.
Нахождение обратной матрицы A^{-1}.
Прямой ход:

$$\begin{array}{cccc|cccc} x_{14} & x_{24} & x_{34} & x_{44} & & & & \\ x_{13} & x_{23} & x_{33} & x_{43} & & & & \\ x_{12} & x_{22} & x_{32} & x_{42} & & & & \\ x_{11} & x_{21} & x_{31} & x_{41} & b^1 & b^2 & b^3 & b^4 \\ \left(\begin{array}{cccc|cccc} \mathbf{1} & 3 & -1 & 2 & 1 & 0 & 0 & 0 \\ 6 & -2 & 0 & 2 & 0 & 1 & 0 & 0 \\ 3 & -5 & 1 & 8 & 0 & 0 & 1 & 0 \\ -1 & 4 & -5 & 9 & 0 & 0 & 0 & 1 \end{array}\right) & & & & (-6/1); (-3/1); (1/1); & & & \xrightarrow[1\text{-й шаг}]{} \end{array}$$

$$
\begin{array}{cccc}
x_{14} & x_{24} & x_{34} & x_{44} \\
x_{13} & x_{23} & x_{33} & x_{43} \\
x_{12} & x_{22} & x_{32} & x_{42} \\
x_{11} & x_{21} & x_{31} & x_{41}
\end{array}
\quad
\begin{array}{cccc}
b^1 & b^2 & b^3 & b^4
\end{array}
$$

$$
\left(\begin{array}{cccc|cccc}
1 & 3 & -1 & 2 & 1 & 0 & 0 & 0 \\
0 & \mathbf{-20} & 6 & -10 & -6 & 1 & 0 & 0 \\
0 & -14 & 4 & 2 & -3 & 0 & 1 & 0 \\
0 & 7 & -6 & 11 & 1 & 0 & 0 & 1
\end{array}\right)(-14/20);(7/20);\xrightarrow[2\text{–}й\quad шаг]{}
$$

$$
\begin{array}{cccc}
x_{14} & x_{24} & x_{34} & x_{44} \\
x_{13} & x_{23} & x_{33} & x_{43} \\
x_{12} & x_{22} & x_{32} & x_{42} \\
x_{11} & x_{21} & x_{31} & x_{41}
\end{array}
\quad
\begin{array}{cccc}
b^1 & b^2 & b^3 & b^4
\end{array}
$$

$$
\left(\begin{array}{cccc|cccc}
1 & 3 & -1 & 2 & 1 & 0 & 0 & 0 \\
0 & -20 & 6 & -10 & -6 & 1 & 0 & 0 \\
0 & 0 & \mathbf{-1/5} & 9 & 6/5 & -7/10 & 1 & 0 \\
0 & 0 & -39/10 & 15/2 & -11/10 & 7/20 & 0 & 1
\end{array}\right)(-39/2);\xrightarrow[3\text{–}й\quad шаг]{}
$$

$$
\begin{array}{cccc}
x_{14} & x_{24} & x_{34} & x_{44} \\
x_{13} & x_{23} & x_{33} & x_{43} \\
x_{12} & x_{22} & x_{32} & x_{42} \\
x_{11} & x_{21} & x_{31} & x_{41}
\end{array}
\quad
\begin{array}{cccc}
b^1 & b^2 & b^3 & b^4
\end{array}
$$

$$
\left(\begin{array}{cccc|cccc}
1 & 3 & -1 & 2 & 1 & 0 & 0 & 0 \\
0 & -20 & 6 & -10 & -6 & 1 & 0 & 0 \\
0 & 0 & -1/5 & 9 & 6/5 & -7/10 & 1 & 0 \\
0 & 0 & 0 & -168 & -49/2 & 14 & -39/2 & 1
\end{array}\right).
$$

Обратный ход:

$$
\begin{cases}
-168x_{41} = -49/2 \\
(-1/5)x_{31} + 9x_{41} = 6/5 \\
-20x_{21} + 6x_{31} - 10x_{41} = -6 \\
x_{11} + 3x_{21} - x_{31} + 2x_{41} = 1
\end{cases};
\quad
\begin{cases}
-168x_{42} = 14 \\
(-1/5)x_{32} + 9x_{42} = -7/10 \\
-20x_{22} + 6x_{32} - 10x_{42} = 1 \\
x_{12} + 3x_{22} - x_{32} + 2x_{42} = 0
\end{cases};
$$

$$
\begin{cases}
-168x_{43} = -39/2 \\
(-1/5)x_{33} + 9x_{43} = 1 \\
-20x_{23} + 6x_{33} - 10x_{43} = 0 \\
x_{13} + 3x_{23} - x_{33} + 2x_{43} = 0
\end{cases};
\quad
\begin{cases}
-168x_{44} = 1 \\
(-1/5)x_{34} + 9x_{44} = 0 \\
-20x_{24} + 6x_{34} - 10x_{44} = 0 \\
x_{14} + 3x_{24} - x_{34} + 2x_{44} = 0
\end{cases}.
$$

Отсюда

$$A^{-1} = \begin{pmatrix} x_{11} & x_{12} & x_{13} & x_{14} \\ x_{21} & x_{22} & x_{23} & x_{24} \\ x_{31} & x_{32} & x_{33} & x_{34} \\ x_{41} & x_{42} & x_{43} & x_{44} \end{pmatrix} = \begin{pmatrix} 1/12 & 1/6 & -1/28 & -1/42 \\ 19/48 & -1/12 & 1/112 & -13/168 \\ 9/16 & -1/4 & 25/112 & -15/56 \\ 7/48 & -1/12 & 13/112 & -1/168 \end{pmatrix}.$$

2.7. МЕТОД ГАУССА С ВЫБОРОМ ГЛАВНОГО ЭЛЕМЕНТА

На некотором i-м шаге прямого хода метода исключения Гаусса может оказаться, что коэффициент $a_{ii}^{i-1} \neq 0$, но мал по сравнению с остальными элементами матрицы системы и, в частности, мал по сравнению с элементами i-го столбца ниже. Деление коэффициентов системы на малую величину может привести к значительным ошибкам округления или даже переполнению переменных.

Для уменьшения этих ошибок поступают следующим образом. Среди элементов i-го столбца a_{ki}^{i-1}, $k = i, \ldots, n$ каждой промежуточной матрицы выбирают наибольший по модулю (главный) элемент и путем перестановки i-й строки и строки, содержащей главный элемент, делают главный элемент ведущим. Такая модификация метода исключения Гаусса называется методом Гаусса с выбором главного элемента. Случай появления нулевых ведущих элементов при этом обходится сам собой.

Пример. Методом Гаусса с выбором главного элемента решить СЛАУ, вычислить определитель и обратную матрицу для матрицы СЛАУ:

$$\begin{cases} x_1 + 3x_2 - x_3 + 2x_4 = 13 \\ 6x_1 - 2x_2 + 2x_4 = 20 \\ 3x_1 - 5x_2 + x_3 + 8x_4 = 7 \\ -x_1 + 4x_2 - 5x_3 + 9x_4 = 7 \end{cases}; \quad A = \begin{pmatrix} 1 & 3 & -1 & 2 \\ 6 & -2 & 0 & 2 \\ 3 & -5 & 1 & 8 \\ -1 & 4 & -5 & 9 \end{pmatrix}, \quad b = \begin{pmatrix} 13 \\ 20 \\ 7 \\ 7 \end{pmatrix}.$$

Прямой ход:

$$\begin{array}{c} \begin{matrix} x_1 & x_2 & x_3 & x_4 & b \end{matrix} \\ \left(\begin{array}{cccc|c} 1 & 3 & -1 & 2 & 13 \\ \mathbf{6} & -2 & 0 & 2 & 20 \\ 3 & -5 & 1 & 8 & 7 \\ -1 & 4 & -5 & 9 & 7 \end{array}\right) \end{array} \xrightarrow[1\text{-й шаг}]{}$$

$$\begin{array}{c} \begin{matrix} x_1 & x_2 & x_3 & x_4 & b \end{matrix} \\ \left(\begin{array}{cccc|c} 6 & -2 & 0 & 2 & 20 \\ 0 & 10/3 & -1 & 5/3 & 29/3 \\ 0 & \mathbf{-4} & 1 & 7 & -3 \\ 0 & 11/3 & -5 & 28/3 & 31/3 \end{array}\right) \end{array} \xrightarrow{2\text{-й шаг}}$$

$$\begin{array}{c} \begin{matrix} x_1 & x_2 & x_3 & x_4 & b \end{matrix} \\ \left(\begin{array}{cccc|c} 6 & -2 & 0 & 2 & 20 \\ 0 & -4 & 1 & 7 & -3 \\ 0 & 0 & -1/6 & 15/2 & 43/6 \\ 0 & 0 & \mathbf{-49/12} & 63/4 & 91/12 \end{array}\right) \end{array} \xrightarrow{3\text{-й шаг}}$$

$$\begin{array}{c} \begin{matrix} x_1 & x_2 & x_3 & x_4 & b \end{matrix} \\ \left(\begin{array}{cccc|c} 6 & -2 & 0 & 2 & 20 \\ 0 & -4 & 1 & 7 & -3 \\ 0 & 0 & -49/12 & 63/4 & 91/12 \\ 0 & 0 & 0 & 48/7 & 48/7 \end{array}\right) \end{array}.$$

Произведённое число перестановок за 3 шага: $p = 3$.
Определитель матрицы:

$$\det A = (-1)^3 \cdot 6 \cdot (-4) \cdot (-49/12) \cdot (48/7) = -672 .$$

Обратный ход:

$$\begin{cases} 48/7 x_4 = 48/7 \\ (-49/12)x_3 + (63/4)x_4 = 91/12 \\ -4x_2 + x_3 + 7x_4 = -3 \\ 6x_1 - 2x_2 + 2x_4 = 20 \end{cases};$$

$$\begin{cases} x_4 = 1 \\ (-49/12)x_3 = 91/12 - (63/4)\cdot 1 = -98/12,\ x_3 = 2 \\ -4x_2 = -3 - 2 - 7\cdot 1 = -12,\ x_2 = 3 \\ 6x_1 = 20 + 2\cdot 3 - 2\cdot 1 = 24,\ x_1 = 4 \end{cases}.$$

Решение СЛАУ: $x = (4 \quad 3 \quad 2 \quad 1)^T$.
Нахождение обратной матрицы A^{-1}.

Прямой ход:

$$
\begin{array}{cccc}
x_{14} & x_{24} & x_{34} & x_{44} \\
x_{13} & x_{23} & x_{33} & x_{43} \\
x_{12} & x_{22} & x_{32} & x_{42} \\
x_{11} & x_{21} & x_{31} & x_{41}
\end{array}
\quad
\begin{array}{cccc}
b^1 & b^2 & b^3 & b^4
\end{array}
$$

$$
\left(\begin{array}{cccc|cccc}
1 & 3 & -1 & 2 & 1 & 0 & 0 & 0 \\
\mathbf{6} & -2 & 0 & 2 & 0 & 1 & 0 & 0 \\
3 & -5 & 1 & 8 & 0 & 0 & 1 & 0 \\
-1 & 4 & -5 & 9 & 0 & 0 & 0 & 1
\end{array}\right)
\xrightarrow[1\text{-}й\quad шаг]{}
$$

$$
\begin{array}{cccc}
x_{14} & x_{24} & x_{34} & x_{44} \\
x_{13} & x_{23} & x_{33} & x_{43} \\
x_{12} & x_{22} & x_{32} & x_{42} \\
x_{11} & x_{21} & x_{31} & x_{41}
\end{array}
\quad
\begin{array}{cccc}
b^1 & b^2 & b^3 & b^4
\end{array}
$$

$$
\left(\begin{array}{cccc|cccc}
6 & -2 & 0 & 2 & 0 & 1 & 0 & 0 \\
0 & 10/3 & -1 & 5/3 & 1 & -1/6 & 0 & 0 \\
0 & \mathbf{-4} & 1 & 7 & 0 & -1/2 & 1 & 0 \\
0 & 11/3 & -5 & 28/3 & 0 & 1/6 & 0 & 1
\end{array}\right)
\xrightarrow[2\text{-}й\quad шаг]{}
$$

$$
\begin{array}{cccc}
x_{14} & x_{24} & x_{34} & x_{44} \\
x_{13} & x_{23} & x_{33} & x_{43} \\
x_{12} & x_{22} & x_{32} & x_{42} \\
x_{11} & x_{21} & x_{31} & x_{41}
\end{array}
\quad
\begin{array}{cccc}
b^1 & b^2 & b^3 & b^4
\end{array}
$$

$$
\left(\begin{array}{cccc|cccc}
6 & -2 & 0 & 2 & 0 & 1 & 0 & 0 \\
0 & -4 & 1 & 7 & 0 & -1/2 & 1 & 0 \\
0 & 0 & -1/6 & 15/2 & 1 & -7/12 & 5/6 & 0 \\
0 & 0 & \mathbf{-49/12} & 63/4 & 0 & -7/24 & 11/12 & 1
\end{array}\right)
\xrightarrow[3\text{-}й\quad шаг]{}
$$

$$
\begin{array}{cccc}
x_{14} & x_{24} & x_{34} & x_{44} \\
x_{13} & x_{23} & x_{33} & x_{43} \\
x_{12} & x_{22} & x_{32} & x_{42} \\
x_{11} & x_{21} & x_{31} & x_{41}
\end{array}
\quad
\begin{array}{cccc}
b^1 & b^2 & b^3 & b^4
\end{array}
$$

$$
\left(\begin{array}{cccc|cccc}
6 & -2 & 0 & 2 & 0 & 1 & 0 & 0 \\
0 & -4 & 1 & 7 & 0 & -1/2 & 1 & 0 \\
0 & 0 & -49/12 & 63/4 & 0 & -7/24 & 11/12 & 1 \\
0 & 0 & 0 & 48/7 & 1 & -4/7 & 39/49 & -2/49
\end{array}\right)
$$

Обратный ход:

$$\begin{cases} 48/7x_{41} = 1 \\ (-49/12)x_{31} + (63/4)x_{41} = 0 \\ -4x_{21} + x_{31} + 7x_{41} = 0 \\ 6x_{11} - 2x_{21} + 2x_{41} = 0 \end{cases}; \quad \begin{cases} 48/7x_{42} = -4/7 \\ (-49/12)x_{32} + (63/4)x_{42} = -7/24 \\ -4x_{22} + x_{32} + 7x_{42} = -1/2 \\ 6x_{12} - 2x_{22} + 2x_{42} = 1 \end{cases};$$

$$\begin{cases} 48/7x_{43} = 39/49 \\ (-49/12)x_{33} + (63/4)x_{43} = 11/12 \\ -4x_{23} + x_{33} + 7x_{43} = 1 \\ 6x_{13} - 2x_{23} + 2x_{43} = 0 \end{cases}; \quad \begin{cases} 48/7x_{44} = -2/49 \\ (-49/12)x_{34} + (63/4)x_{44} = 1 \\ -4x_{24} + x_{34} + 7x_{44} = 0 \\ 6x_{14} - 2x_{24} + 2x_{44} = 0 \end{cases}.$$

Отсюда

$$A^{-1} = \begin{pmatrix} x_{11} & x_{12} & x_{13} & x_{14} \\ x_{21} & x_{22} & x_{23} & x_{24} \\ x_{31} & x_{32} & x_{33} & x_{34} \\ x_{41} & x_{42} & x_{43} & x_{44} \end{pmatrix} = \begin{pmatrix} 1/12 & 1/6 & -1/28 & -1/42 \\ 19/48 & -1/12 & 1/112 & -13/168 \\ 9/16 & -1/4 & 25/112 & -15/56 \\ 7/48 & -1/12 & 13/112 & -1/168 \end{pmatrix}.$$

2.8. ПРОГРАММА #02

Ниже предлагается вариант алгоритма программы для решения СЛАУ, вычисления определителя и обратной матрицы для матрицы СЛАУ методом исключения Гаусса или методом Гаусса с выбором главного элемента.

```
АЛГОРИТМ «Метод Гаусса с выбором главного элемента»
ВХОД n, a[n][n], b[n]
ВЫХОД i, j, k, m, p, f, g, h, x[n], y, z[n][n]
НАЧАЛО
    ЦИКЛ «Строки» ДЛЯ i ОТ 1 ДО n ПО 1
        ЦИКЛ «Столбцы» ДЛЯ j ОТ 1 ДО n ПО 1
            ЕСЛИ (i=j) z[i][j]:=1 ИНАЧЕ z[i][j]:=0
    p:=0
    ЦИКЛ «Прямой ход» ДЛЯ k ОТ 1 ДО n-1 ПО 1
        #Для метода Гаусса с выбором главного элемента
        ЕСЛИ (a[k][k]>0) g:=a[k][k] ИНАЧЕ g:=-a[k][k]
```

```
            m:=k
            ЦИКЛ «Строки» ДЛЯ i ОТ k+1 ДО n ПО 1
                ЕСЛИ (a[i][k]>0) f:=a[i][k] ИНАЧЕ f:=-a[i][k]
                ЕСЛИ (f>g) g:=f, m:=i
            ЕСЛИ (g=0)
                ПЕЧАТЬ «Решение не единственно или отсутствует!»
                КОНЕЦ
            ЕСЛИ (m≠k)
                ЦИКЛ «Столбцы» ДЛЯ j ОТ k ДО n ПО 1
                    g:=a[k][j], a[k][j]:=a[m][j], a[m][j]:=g
                g:=b[k], b[k]:=b[m], b[m]:=g
                ЦИКЛ «Столбцы» ДЛЯ j ОТ 1 ДО n ПО 1
                    g:=z[k][j], z[k][j]:=z[m][j], z[m][j]:=g
                p:=p+1
            #Общая часть метода Гаусса
            ЦИКЛ «Строки» ДЛЯ i ОТ k+1 ДО n ПО 1
                h:=-a[i][k]/a[k][k]
                ЦИКЛ «Столбцы» ДЛЯ j ОТ k ДО n ПО 1
                    a[i][j]:=a[i][j]+h*a[k][j]
                b[i]:=b[i]+h*b[k]
                ЦИКЛ «Столбцы» ДЛЯ j ОТ 1 ДО n ПО 1
                    z[i][j]:=z[i][j]+h*z[k][j]
            y:=1-2*(p%2)
            ЦИКЛ «Определитель» ДЛЯ k ОТ 1 ДО n ПО 1
                y:=y*a[k][k]
            ЦИКЛ «Обратный ход» ДЛЯ k ОТ n ДО 1 ПО -1
                    h:=0
                    ЦИКЛ «Строки» ДЛЯ i ОТ k+1 ДО n ПО 1
                        h:=h+a[k][i]*x[i]
                    x[k]:=(b[k]-h)/a[k][k]
                    ЦИКЛ «Столбцы» ДЛЯ j ОТ 1 ДО n ПО 1
                        h:=0
                        ЦИКЛ «Строки» ДЛЯ i ОТ k+1 ДО n ПО 1
                            h:=h+a[k][i]*z[i][j]
                        z[k][j]:=(z[k][j]-h)/a[k][k]
    ПЕЧАТЬ x, y, z
КОНЕЦ
```

2.9. *LU*-РАЗЛОЖЕНИЕ МАТРИЦ

Компьютерная реализация метода Гаусса часто осуществляется с использованием LU-разложения матриц.

LU-разложение матрицы A представляет собой разложение матрицы A в произведение нижней и верхней треугольных матриц:

$$A = LU\,,$$

где L — нижнетреугольная матрица ($l_{ij} = 0$ при $i < j$), а U — верхнетреугольная матрица ($u_{ij} = 0$ при $i > j$). Подобное разложение, очевидно, может быть выполнено бесконечным количеством способов, поэтому для его однозначности положим $l_{ii} = 1$, $i = 1,\ldots,n$.

Искомое LU-разложение может быть построено с использованием описанного выше метода Гаусса. Рассмотрим k-й шаг метода Гаусса, на котором осуществляется обнуление поддиагональных элементов k-го столбца матрицы $A^{(k-1)}$. Как было описано выше, с этой целью используется следующая операция:

$$a_{ij}^{(k)} = a_{ij}^{(k-1)} - \mu_i^{(k)} a_{kj}^{(k-1)},\ \text{где}\ \mu_i^{(k)} = \frac{a_{ik}^{(k-1)}}{a_{kk}^{(k-1)}},\ i = k+1,\ldots,n,\ j = k,\ldots,n\,.$$

В терминах матричных операций такая операция эквивалентна выполнению умножения $A^{(k)} = M_k A^{(k-1)}$, где элементы матрицы M_k определяются следующим образом:

$$m_{ij}^k = \begin{cases} 1, & i = j \\ 0, & i \neq j,\quad j \neq k \\ -\mu_{k+1}^{(k)}, & i \neq j,\quad j = k \end{cases},$$

т.е. матрица M_k имеет вид

$$M_k = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & -\mu_{k+1}^{(k)} & 1 & 0 & 0 \\ \ldots & \ldots & \ldots & \ldots & \ldots & \ldots \\ 0 & 0 & -\mu_n^{(k)} & 0 & 0 & 1 \end{pmatrix}.$$

При этом выражение для обратной операции запишется в форме $A^{(k-1)} = M_k^{-1} A^{(k)}$, где матрица M_k^{-1} имеет вид

$$M_k^{-1} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & \mu_{k+1}^{(k)} & 1 & 0 & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & \mu_n^{(k)} & 0 & 0 & 1 \end{pmatrix}.$$

В результате прямого хода метода Гаусса

$$A^{(n-1)} = M_{n-1} A^{(n-2)} = M_{n-1} M_{n-2} A^{(n-3)} = M_{n-1} M_{n-2} \dots M_2 A^{(1)} =$$

$$= M_{n-1} M_{n-2} \dots M_2 M_1 A^{(0)}$$

или, обратно, $A = A^{(0)} = M_1^{-1} A^{(1)} = M_1^{-1} M_2^{-1} A^{(2)} = M_1^{-1} M_2^{-1} \dots M_{n-1}^{-1} A^{(n-1)}$ получим $A^{(n-1)}$, причём $A^{(n-1)} = U$ — искомая верхнетреугольная матрица, в то время как $M_1^{-1} M_2^{-1} \dots M_{n-1}^{-1} = L$ — искомая нижнетреугольная матрица, имеющая вид

$$L = \begin{pmatrix} 1 & 0 & 0 & 0 & & 0 & 0 \\ \mu_2^{(1)} & 1 & 0 & 0 & & 0 & 0 \\ \mu_3^{(1)} & \mu_3^{(2)} & 1 & 0 & & 0 & 0 \\ \dots & \dots & \mu_{k+1}^{(k)} & 1 & & 0 & 0 \\ \dots & \dots & \dots & \dots & & \dots & \dots \\ \mu_n^{(1)} & \mu_n^{(2)} & \mu_n^{(k)} & \mu_n^{(k+1)} & \dots & \mu_n^{(n-1)} & 1 \end{pmatrix}.$$

Таким образом, искомое разложение $A = LU$ получено.

Подобное LU-разложение матрицы $A_{n \times n}$, однажды полученное (с использованием порядка n^3 арифметических операций), затем может быть эффективно использовано при решении систем линейных алгебраических уравнений вида $Ax = b$ (а также нахождении обратной матрицы A^{-1}).

Действительно, подставляя LU-разложение в СЛАУ, получим:

$$LUx = b, \text{ или } Ux = L^{-1} b.$$

В результате процесс решения СЛАУ сводится к двум простым этапам.

На первом этапе решается СЛАУ $Lz = b$. Поскольку матрица системы нижнетреугольная, решение можно записать в явном виде:

$$z_1 = b_1,\ z_i = b_i - \sum_{j=1}^{i-1} l_{ij} z_j,\ i = 2,\ldots,n.$$

На втором этапе решается СЛАУ $Ux = z$. Поскольку матрица системы верхнетреугольная, то здесь, как и на предыдущем этапе, решение представляется в явном виде:

$$x_n = \frac{z_n}{u_{nn}},\ x_i = \frac{1}{u_{ii}}(z_i - \sum_{j=i+1}^{n} u_{ij} x_j),\ i = n-1,\ldots,1.$$

Отметим, что второй этап эквивалентен обратному ходу методу Гаусса, тогда как первый соответствует преобразованию правой части СЛАУ в процессе прямого хода. Оба этапа требуют использования порядка n^2 арифметических операций.

2.10. МЕТОД ПРОСТОЙ ИТЕРАЦИИ (ЛИНЕЙНЫЕ СИСТЕМЫ)

Метод простой итерации относится к итерационным методам решения систем линейных алгебраических уравнений. Итерационными называются методы последовательных приближений, в которых при вычислении последующего приближения решения используются предыдущие, уже известные приближённые решения. Принципиально точного решения x^* итерационные методы не дают, поскольку оно достигается как предел последовательности приближений $x^{(k)}, k = 0,1,2,...$ при $k \to \infty$. Однако итерационные методы обладают свойством, позволяющим получить решение с любой наперёд заданной точностью, если доказана сходимость метода. Поэтому при решении любой задачи итерационным методом неизбежно возникает необходимость задаться требуемым уровнем точности ответа, т.е. задать допустимую погрешность решения ε.

Следует отметить, что при большом числе уравнений прямые методы решения СЛАУ (за исключением метода прогонки) теряют в эффективности реализации на ЭВМ прежде всего из-за сложности хранения и обработки матриц большой размерности. В то же время

характерной особенностью ряда часто встречающихся в прикладных задачах СЛАУ является разреженность матриц. Число ненулевых элементов таких матриц относительно мало по сравнению с их размерностью. Поэтому для решения СЛАУ с разреженными матрицами предпочтительнее использовать итерационные методы.

Пусть дана СЛАУ с невырожденной матрицей ($\det A \neq 0$):

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

Приведём эту СЛАУ к следующему эквивалентному виду:

$$\begin{cases} x_1 = \beta_1 + \alpha_{11}x_1 + \alpha_{12}x_2 + \ldots + \alpha_{1n}x_n \\ x_2 = \beta_2 + \alpha_{21}x_1 + \alpha_{22}x_2 + \ldots + \alpha_{2n}x_n \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ x_n = \beta_n + \alpha_{n1}x_1 + \alpha_{n2}x_2 + \ldots + \alpha_{nn}x_n \end{cases},$$

или, в векторно-матричной форме:

$$x = \beta + \alpha x \text{, где } x = \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix}, \quad \beta = \begin{pmatrix} \beta_1 \\ \vdots \\ \beta_n \end{pmatrix}, \quad \alpha = \begin{pmatrix} \alpha_{11} & \cdots & \alpha_{1n} \\ \vdots & \cdots & \vdots \\ \alpha_{n1} & \ldots & \alpha_{nn} \end{pmatrix}.$$

Такое приведение может быть выполнено различными способами, в том числе через тождество $x = x + C(Ax - b)$, где C — выбираемое в соответствии с возможными дополнительными требованиями число или, в самом общем случае, матрица.

Однако одним из наиболее распространённых способов является выделение диагональных элементов. Разрешим СЛАУ относительно неизвестных при ненулевых диагональных элементах $a_{ii} \neq 0$, $i = 1, \ldots, n$ (если какой-либо коэффициент на главной диагонали равен нулю, достаточно соответствующее уравнение поменять местами с любым другим уравнением). Получим следующие выражения для компонентов вектора β и матрицы α эквивалентной системы:

$$\beta_i = \frac{b_i}{a_{ii}}; \; \alpha_{ij} = -\frac{a_{ij}}{a_{ii}}, \; i \neq j; \; \alpha_{ij} = 0, \; i = j; \; i, j = 1, \ldots, n,$$

а СЛАУ в эквивалентном виде запишется как

$$x_i = \left[\frac{b_i}{a_{ii}} - \sum_{j=1, j\neq i}^{n} \frac{a_{ij}}{a_{ii}} x_j \right], \; i = 1,...,n .$$

При таком способе приведения исходной СЛАУ к эквивалентному виду метод простой итерации носит название метода Якоби.

Вне зависимости от способа приведения СЛАУ к эквивалентному виду метод простой итерации имеет следующий вид:

$$\begin{cases} x^{(0)} = \beta \\ x^{(1)} = \beta + \alpha x^{(0)} \\ x^{(2)} = \beta + \alpha x^{(1)} \\ \ldots\ldots\ldots\ldots\ldots\ldots \\ x^{(k)} = \beta + \alpha x^{(k-1)} \end{cases},$$

где k — число исполненных итераций.

Теперь очевидно преимущество итерационных методов по сравнению, например, с методом Гаусса. В вычислительном процессе участвуют только произведения матрицы на вектор, что позволяет работать только с ненулевыми элементами матрицы, значительно оптимизируя процесс хранения и обработки матриц в ЭВМ.

При построении любого итерационного процесса для нахождения точного решения x^* некоторой задачи необходимо дать ответ на три вопроса: с чего он начинается, т.е. значение нулевого приближения $x^{(0)}$; при каких условиях он сходится к решению, т.е. $\lim\limits_{k\to\infty} \left\| x^{(k)} - x^* \right\| = 0$; когда его нужно останавливать, т.е. требуемая точность ε достигнута $\left\| x^{(k)} - x^* \right\| \le \varepsilon$.

В качестве нулевого приближения $x^{(0)}$ вектора неизвестных можно принять вектор $x^{(0)} = \beta$, $(x_1^{(0)} \;\; x_2^{(0)} \;\; \ldots \;\; x_n^{(0)})^T = (\beta_1 \;\; \beta_2 \;\; \ldots \;\; \beta_n)^T$, хотя если итерационный процесс сходится, то он сойдётся и для любого другого начального вектора.

Имеет место следующее достаточное условие сходимости метода простой итерации. Метод простой итерации сходится к единственному решению СЛАУ при любом начальном приближении $x^{(0)}$, если какая-либо норма матрицы α эквивалентной системы меньше единицы: $\|\alpha\| < 1$.

Действительно, согласно выбранному итерационному методу, верно $x^{(k)} = \beta + \alpha x^{(k-1)}$, а согласно исходному уравнению, $x^* = \beta + \alpha x^*$ суть тождество, если x^* — искомое решение, т.к. $x = \beta + \alpha x$ эквивалентно $Ax = b$.

Тогда, после вычитания,

$$x^{(k)} - x^* = \alpha(x^{(k-1)} - x^*),$$

а значит, верно и $\left\|x^{(k)} - x^*\right\| = \left\|\alpha(x^{(k-1)} - x^*)\right\|$.

Согласно 4-му свойству норм для матриц,

$$\left\|x^{(k)} - x^*\right\| = \left\|\alpha(x^{(k-1)} - x^*)\right\| \le \|\alpha\| \cdot \left\|x^{(k-1)} - x^*\right\|.$$

Применяя это соотношение само к себе k раз, можно получить:

$$\left\|x^{(k)} - x^*\right\| \le \|\alpha\| \cdot \left\|x^{(k-1)} - x^*\right\| \le \|\alpha\|^2 \cdot \left\|x^{(k-2)} - x^*\right\| \le \ldots \le \|\alpha\|^k \cdot \left\|x^{(0)} - x^*\right\|.$$

Отсюда очевидно, что при $k \to \infty$ будет верно $\left\|x^{(k)} - x^*\right\| \to 0$, если

$$0 \le \left\|x^{(k)} - x^*\right\| \le \|\alpha\|^k \cdot \left\|x^{(0)} - x^*\right\| \to 0,$$

а $\|\alpha\|^k \to 0$ при $k \to \infty$ только при $\|\alpha\| < 1$.

Таким образом, доказано, что $\|\alpha\| < 1 \Rightarrow \lim\limits_{k\to\infty} \left\|x^{(k)} - x^*\right\| = 0$.

При этом скорость сходимости не меньше скорости геометрической прогрессии со знаменателем $q \le \|\alpha\|$.

Если используется метод Якоби, то достаточным условием сходимости является диагональное преобладание матрицы A, т.е. $|a_{ii}| > \sum\limits_{j=1, i\ne j}^{n} |a_{ij}|, i = 1, \ldots, n$ (для каждой строки матрицы A модули элементов, стоящих на главной диагонали, больше суммы модулей недиагональных элементов). Очевидно, что в этом случае $\|\alpha\|_c$ меньше единицы и, следовательно, итерационный процесс сходится.

Ещё раз отметим, что рассмотренные формы условия сходимости метода простой итерации являются лишь достаточными. Их выполнение гарантирует сходимость метода, но их невыполнение в общем случае не означает, что метод простой итерации расходится.

Приведём также необходимое и достаточное условие сходимости метода простой итерации (оно редко используется в практике вычислений). Для сходимости итерационного процесса необходимо и

достаточно, чтобы спектр матрицы α эквивалентной системы лежал внутри круга с радиусом, равным единице, т.е. чтобы максимальное по модулю собственное значение матрицы (спектральный радиус матрицы α) $\rho(\alpha) = \max_i |\lambda_i| < 1$.

При выполнении достаточного условия сходимости оценка погрешности решения на k-й итерации $\left\|x^{(k)} - x^*\right\|$ даётся выражением

$$\left\|x^{(k)} - x^*\right\| \le \varepsilon^{(k)} = \frac{\|\alpha\|}{1 - \|\alpha\|}\left\|x^{(k)} - x^{(k-1)}\right\|,$$

где x^* — точное решение СЛАУ.

Действительно, согласно 3-му свойству норм для векторов, верно

$$\left\|x^{(k)} - x^*\right\| = \left\|\alpha(x^{(k-1)} - x^*)\right\| \le \|\alpha\| \cdot \left\|x^{(k-1)} - x^*\right\| =$$

$$= \|\alpha\| \cdot \left\|x^{(k-1)} - x^{(k)} + x^{(k)} - x^*\right\| \le$$

$$\le \|\alpha\| \cdot \left(\left\|x^{(k-1)} - x^{(k)}\right\| + \left\|x^{(k)} - x^*\right\|\right) =$$

$$= \|\alpha\| \cdot \left\|x^{(k)} - x^{(k-1)}\right\| + \|\alpha\| \cdot \left\|x^{(k)} - x^*\right\|.$$

Выражая $\left\|x^{(k)} - x^*\right\|$, получаем $\left\|x^{(k)} - x^*\right\| \le \frac{\|\alpha\|}{1 - \|\alpha\|} \cdot \left\|x^{(k)} - x^{(k-1)}\right\| = \varepsilon^{(k)}$.

Таким образом, процесс итераций останавливается при выполнении условия $\varepsilon^{(k)} \le \varepsilon$, где ε — задаваемая вычислителем точность.

Принимая во внимание, что $\left\|x^{(k)} - x^*\right\| \le \frac{\|\alpha\|^k}{1 - \|\alpha\|}\left\|x^{(1)} - x^{(0)}\right\|$, можно получить априорную оценку необходимого для достижения заданной точности числа итераций. При использовании в качестве начального приближения вектора β такая оценка определится неравенством $\frac{\|\alpha\|^{k+1}}{1 - \|\alpha\|}\|\beta\| \le \varepsilon$, откуда получаем априорную оценку числа итераций k при $\|\alpha\| < 1$: $k + 1 \ge \frac{\lg\varepsilon - \lg\|\beta\| + \lg(1 - \|\alpha\|)}{\lg\|\alpha\|}$. Следует подчеркнуть, что это неравенство даёт завышенное число итераций k, поэтому редко используется на практике.

Поскольку $\|\alpha\| < 1$ является только достаточным (не необходимым) условием сходимости метода простой итерации, то итераци-

онный процесс может сходиться и в случае, если оно не выполнено. Тогда критерием окончания итераций может служить неравенство $\left\|x^{(k)} - x^{(k-1)}\right\| \le \varepsilon$.

2.11. МЕТОД ЗЕЙДЕЛЯ

Метод простой итерации довольно медленно сходится. Для его ускорения существует итерационный метод Зейделя, который использует очевидную возможность улучшения сходимости итерационного процесса — немедленное введение в расчёт вновь вычисленных компонент вектора $x^{(k)}$. Метод Зейделя заключается в том, что при вычислении компонента $x_i^{(k)}$ вектора неизвестных на k-й итерации уже вычисленные значения $x_1^{(k)}, x_2^{(k)}, \ldots, x_{i-1}^{(k)}$ используются на той же k-й итерации. Значения для остальных компонент $x_i^{(k-1)}, x_{i+1}^{(k-1)}, \ldots, x_n^{(k-1)}$ берутся из предыдущей итерации. Так же как и в методе простой итерации строится эквивалентная СЛАУ, за начальное приближение принимается вектор $x^{(0)} = \beta$, строится итерационный процесс $x^{(k)} = \beta + \alpha x^{(k-1)}$. Отличие в том, что метод Зейделя для вектора $x^{(k)}$ на k-й итерации имеет вид:

$$\begin{cases} x_1^{(k)} = \beta_1 + \alpha_{11}x_1^{(k-1)} + \alpha_{12}x_2^{(k-1)} + \ldots + \alpha_{1n}x_n^{(k-1)} \\ x_2^{(k)} = \beta_2 + \alpha_{21}x_1^{(k)} + \alpha_{22}x_2^{(k-1)} + \ldots + \alpha_{2n}x_n^{(k-1)} \\ x_3^{(k)} = \beta_3 + \alpha_{31}x_1^{(k)} + \alpha_{32}x_2^{(k)} + \alpha_{33}x_3^{(k-1)} + \ldots + \alpha_{13}x_n^{(k-1)} \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ x_n^{(k)} = \beta_n + \alpha_{n1}x_1^{(k)} + \alpha_{n2}x_2^{(k)} + \ldots + \alpha_{nn-1}x_{n-1}^{(k)} + \alpha_{nn}x_n^{(k-1)} \end{cases},$$

или, короче:

$$x_i^{(k)} = \beta_i + \sum_{j=1}^{i-1} \alpha_{ij}x_j^{(k)} + \sum_{j=i+1}^{n} \alpha_{ij}x_j^{(k-1)}, \quad i = 1,\ldots,n$$

$$(x_i^{(k)} = \left[\frac{b_i}{a_{ii}} - \sum_{j=1}^{i-1} \frac{a_{ij}}{a_{ii}} x_j^{(k)} - \sum_{j=i+1}^{n} \frac{a_{ij}}{a_{ii}} x_j^{(k-1)} \right], \ i = 1,\ldots,n$$

для метода Якоби).

Отсюда видно, что $x^{(k)} = \beta + Bx^{(k)} + Cx^{(k-1)}$, где B — нижнетреугольная матрица с диагональными элементами, равными нулю, а

C — верхнетреугольная матрица с диагональными элементами, отличными от нуля, и верно $\alpha = B + C$. Следовательно,

$$(E - B)x^{(k)} = \beta + Cx^{(k-1)},$$

откуда, так как матрица $(E - B)$ всегда невырождена:

$$x^{(k)} = (E - B)^{-1}\beta + (E - B)^{-1}Cx^{(k-1)}.$$

Таким образом, метод Зейделя эквивалентен методу простой итерации с матрицей правых частей $\tilde{\alpha} = (E - B)^{-1}C$ и вектором правых частей $\tilde{\beta} = (E - B)^{-1}\beta$, и, следовательно, условия сходимости и формулы погрешности метода Зейделя можно исследовать с помощью формул, выведенных для метода простой итерации, в которых вместо матрицы α подставлена матрица $\tilde{\alpha} = (E - B)^{-1}C$, а вместо вектора β — вектор $\tilde{\beta} = (E - B)^{-1}\beta$.

Для практических вычислений важно, что в качестве достаточных условий сходимости метода Зейделя могут быть использованы условия, приведённые выше для метода простой итерации ($\|\alpha\| < 1$ или, если используется эквивалентная СЛАУ в форме Якоби — диагональное преобладание матрицы A). В случае выполнения этих условий для оценки погрешности на k-й итерации можно использовать выражение:

$$\left\|x^{(k)} - x^{*}\right\| \le \frac{\|C\|}{1 - \|\alpha\|} \cdot \left\|x^{(k)} - x^{(k-1)}\right\| \le \frac{\|\alpha\|}{1 - \|\alpha\|} \cdot \left\|x^{(k)} - x^{(k-1)}\right\| = \varepsilon^{(k)}.$$

Отметим, что, как и метод простой итерации, метод Зейделя может сходиться и при нарушении условия $\|\alpha\| < 1$. В этом случае $\varepsilon^{(k)} = \left\|x^{(k)} - x^{(k-1)}\right\|$.

Кроме того, при использовании метода Зейделя может быть поставлена задача об отыскании на каждой итерации оптимальной последовательности уточнения компонент вектора неизвестных $x^{(k)}$. Удовлетворительные методы построения такой последовательности неизвестны. На практике иногда используется упорядочение компонент по убыванию разности их значений на двух последовательных итерациях.

Итерационный процесс метода Зейделя можно обобщить и записать в следующей форме:

$$x_i^{(k)} = \omega\left(\beta_i + \sum_{j=1}^{i-1}\alpha_{ij}x_j^{(k)} + \sum_{j=i+1}^{n}\alpha_{ij}x_j^{(k-1)}\right) + (1 - \omega)x_i^{(k-1)},\ i = 1,\ldots,n,$$

вводя в итерационный процесс на k-м шаге для вычисления $x_i^{(k)}$ значение $x_i^{(k-1)}$, что не имеет места ни в методе простой итерации (с преобразованием методом Якоби), ни в методе Зейделя.

В этом случае итерационный процесс при $\omega > 1$ называют методом верхней релаксации, при $\omega = 1$ (метод Зейделя) — методом полной релаксации и при $\omega < 1$ — методом нижней релаксации.

Пример. Методом простой итерации и методом Зейделя решить СЛАУ $Ax = b$ с точностью $\varepsilon = 0.2$:

$$\begin{cases} 5x_1 - x_2 + 2x_3 + x_4 = 7 \\ 4x_1 + 11x_2 - 2x_3 - 3x_4 = 10 \\ 6x_1 - 3x_2 + 16x_3 - 4x_4 = 15 \\ 7x_1 + 2x_2 + 4x_3 - 14x_4 = -1 \end{cases}; \quad A = \begin{pmatrix} 5 & -1 & 2 & 1 \\ 4 & 11 & -2 & -3 \\ 6 & -3 & 16 & -4 \\ 7 & 2 & 4 & -14 \end{pmatrix}, \quad b = \begin{pmatrix} 7 \\ 10 \\ 15 \\ -1 \end{pmatrix}.$$

Приведем СЛАУ к эквивалентному виду $x = \beta + \alpha x$ методом Якоби:

$$\begin{cases} x_1 = \dfrac{7}{5} + \dfrac{1}{5}x_2 - \dfrac{2}{5}x_3 - \dfrac{1}{5}x_4 \\ x_2 = \dfrac{10}{11} - \dfrac{4}{11}x_1 + \dfrac{2}{11}x_3 + \dfrac{3}{11}x_4 \\ x_3 = \dfrac{15}{16} - \dfrac{6}{16}x_1 + \dfrac{3}{16}x_2 + \dfrac{4}{16}x_4 \\ x_4 = \dfrac{1}{14} + \dfrac{7}{14}x_1 + \dfrac{2}{14}x_2 + \dfrac{4}{14}x_3 \end{cases};$$

$$\alpha = \begin{pmatrix} 0 & 1/5 & -2/5 & -1/5 \\ -4/11 & 0 & 2/11 & 3/11 \\ -3/8 & 3/16 & 0 & 1/4 \\ 1/2 & 1/7 & 2/7 & 0 \end{pmatrix}, \quad \beta = \begin{pmatrix} 7/5 \\ 10/11 \\ 15/16 \\ 1/14 \end{pmatrix}.$$

Норма матрицы α $q = \|\alpha\|_c = \dfrac{13}{14} = 0.929 < 1$, следовательно, достаточное условие сходимости выполнено. Начальное приближение:

$$\alpha = \begin{pmatrix} 0 & 0.200 & -0.400 & -0.200 \\ -0.364 & 0 & 0.182 & 0.273 \\ -0.375 & 0.188 & 0 & 0.250 \\ 0.500 & 0.143 & 0.286 & 0 \end{pmatrix}, \quad x^{(0)} = \beta = \begin{pmatrix} 1.400 \\ 0.909 \\ 0.938 \\ 0.071 \end{pmatrix},$$

$$\varepsilon^{(0)} = \frac{q}{1-q}\left\|x^{(0)}\right\| = \frac{0.929}{0.071} \cdot 1.4 = 18.200 > \varepsilon.$$

Далее итерационный процесс метода простой итерации выглядит следующим образом:

$$x^{(1)} = \beta + \alpha x^{(0)},$$

$$\begin{cases} x_1^{(1)} = 1.400 + 0.200 \cdot 0.909 - 0.400 \cdot 0.938 - 0.200 \cdot 0.071 = 1.193 \\ x_2^{(1)} = 0.909 - 0.364 \cdot 1.400 + 0.182 \cdot 0.938 + 0.273 \cdot 0.071 = 0.590 \\ x_3^{(1)} = 0.938 - 0.375 \cdot 1.400 + 0.188 \cdot 0.909 + 0.250 \cdot 0.071 = 0.601 \\ x_4^{(1)} = 0.071 + 0.500 \cdot 1.400 + 0.143 \cdot 0.909 + 0.286 \cdot 0.938 = 1.169 \end{cases},$$

$$\varepsilon^{(1)} = \frac{q}{1-q}\left\|x^{(1)} - x^{(0)}\right\| = 14.270 > \varepsilon;$$

далее аналогично:

$$x^{(2)} = \beta + \alpha x^{(1)} = \begin{pmatrix} 1.044 \\ 0.904 \\ 0.893 \\ 0.924 \end{pmatrix}, \ \varepsilon^{(2)} = 4.077 > \varepsilon;$$

$$x^{(3)} = \beta + \alpha x^{(2)} = \begin{pmatrix} 1.039 \\ 0.944 \\ 0.946 \\ 0.978 \end{pmatrix}, \ \varepsilon^{(3)} = 0.702 > \varepsilon;$$

$$x^{(4)} = \beta + \alpha x^{(3)} = \begin{pmatrix} 1.015 \\ 0.970 \\ 0.969 \\ 0.996 \end{pmatrix}, \ \varepsilon^{(4)} = 0.341 > \varepsilon;$$

$$x^{(5)} = \beta + \alpha x^{(4)} = \begin{pmatrix} 1.007 \\ 0.988 \\ 0.988 \\ 0.994 \end{pmatrix}, \ \varepsilon^{(5)} = 0.241 > \varepsilon;$$

$$x^{(6)} = \beta + \alpha x^{(5)} = \begin{pmatrix} 1.004 \\ 0.994 \\ 0.994 \\ 0.998 \end{pmatrix}, \ \varepsilon^{(6)} = 0.075 < \varepsilon.$$

Таким образом, вычислительный процесс завершён за 6 итераций. Решение СЛАУ: $x^* \approx x^{(6)} = (1.004 \quad 0.994 \quad 0.994 \quad 0.998)^T$.

Отметим, что точное решение исходной СЛАУ в данном случае известно: $x^* = (1 \quad 1 \quad 1 \quad 1)^T$. Отсюда следует, что заданной точности $\varepsilon = 0.2$ удовлетворяло решение, полученное уже на пятой итерации: $\left\|x^{(5)} - x^*\right\| < \varepsilon$. Но в силу использования для вычисления погрешности оценочного выражения процесс останавливается только на шестой итерации (т.к. $\varepsilon^{(5)} > \varepsilon$).

Отметим также, что априорная оценка необходимого количества итераций в данной задаче даёт $k \geq \dfrac{\lg 0.2 - \lg 1.4 + \lg 0.071}{\lg 0.929} - 1 = 60.869$, что представляет собой неадекватно завышенное значение.

Итерационный процесс метода Зейделя выглядит следующим образом.

Условие сходимости $q = \|\alpha\|_c = 0.929 < 1$ и начальное приближение те же.

$$x^{(0)} = \beta = \begin{pmatrix} 1.400 \\ 0.909 \\ 0.938 \\ 0.071 \end{pmatrix},$$

$$\begin{cases} x_1^{(1)} = 1.400 + 0.200 \cdot 0.909 - 0.400 \cdot 0.938 - 0.200 \cdot 0.071 = 1.193 \\ x_2^{(1)} = 0.909 - 0.364 \cdot 1.193 + 0.182 \cdot 0.938 + 0.273 \cdot 0.071 = 0.665 \\ x_3^{(1)} = 0.938 - 0.375 \cdot 1.193 + 0.188 \cdot 0.665 + 0.250 \cdot 0.071 = 0.633 \\ x_4^{(1)} = 0.071 + 0.500 \cdot 1.193 + 0.143 \cdot 0.665 + 0.286 \cdot 0.633 = 0.944 \end{cases},$$

$$\varepsilon^{(1)} = \frac{q}{1-q}\left\|x^{(1)} - x^{(0)}\right\| = 11.338 > \varepsilon;$$

далее аналогично:

$$x^{(2)} = \begin{pmatrix} 1.091 \\ 0.885 \\ 0.930 \\ 1.009 \end{pmatrix}, \quad \varepsilon^{(2)} = 3.863 > \varepsilon;$$

$$x^{(3)} = \begin{pmatrix} 1.003 \\ 0.989 \\ 0.999 \\ 1.000 \end{pmatrix}, \ \varepsilon^{(3)} = 1.351 > \varepsilon ;$$

$$x^{(4)} = \begin{pmatrix} 0.998 \\ 1.000 \\ 1.001 \\ 0.999 \end{pmatrix}, \ \varepsilon^{(4)} = 0.152 < \varepsilon .$$

Таким образом, вычислительный процесс завершён за 4 итерации. Решение СЛАУ: $x^* \approx x^{(4)} = (0.998 \quad 1.000 \quad 1.001 \quad 0.999)^T$.

Легко видеть, что метод Зейделя в данном случае сходится быстрее метода простой итерации.

2.12. ПРОГРАММА #03

Ниже предлагается вариант алгоритма программы для решения СЛАУ методом простой итерации или методом Зейделя.

```
АЛГОРИТМ «Метод простой итерации или метод Зейделя»
ВХОД        n, c[n][n], d[n], e
ВЫХОД       i, j, f, g, h, a[n][n], b[n], x[n], k
НАЧАЛО
    #Приведение к эквивалентному виду методом Якоби
    ЦИКЛ «Строки» ДЛЯ i ОТ 1 ДО n ПО 1
            ЕСЛИ (c[i][i]=0) КОНЕЦ
            ЦИКЛ «Столбцы» ДЛЯ j ОТ 1 ДО n ПО 1
                    ЕСЛИ (i≠j) a[i][j]:=-c[i][j]/c[i][i] ИНАЧЕ a[i][j]:=0
            b[i]:=d[i]/c[i][i]
    g:=0
    ЦИКЛ «Строки» ДЛЯ i ОТ 1 ДО n ПО 1
            h:=0
            ЦИКЛ «Столбцы» ДЛЯ j ОТ 1 ДО n ПО 1
                    ЕСЛИ (a[i][j]>0) h:=h+a[i][j] ИНАЧЕ h:=h-a[i][j]
            ЕСЛИ (h>g) g:=h
```

```
        ЕСЛИ (g>=1)
                ПЕЧАТЬ «Условие сходимости не выполнено!»
                КОНЕЦ
        f:=0
        ЦИКЛ «Строки» ДЛЯ i ОТ 1 ДО n ПО 1
                x[i]:=b[i]
                ЕСЛИ (x[i]>f) f:=x[i]
                ЕСЛИ (-x[i]>f) f:=-x[i]
        f:=f*g/(1-g)
        k:=0
        ЦИКЛ «Итерация» ПОКА (f>e)
                ЦИКЛ «Строки» ДЛЯ i ОТ 1 ДО n ПО 1
                        y[i]:=x[i]
                ЦИКЛ «Строки» ДЛЯ i ОТ 1 ДО n ПО 1
                        x[i]:=b[i]
                        #Для метода простой итерации
                        ЦИКЛ «Элемент» ДЛЯ j ОТ 1 ДО n ПО 1
                                x[i]:=x[i]+a[i][j]*y[j]
                        #Альтернативно для метода Зейделя
                        ЦИКЛ «Элемент» ДЛЯ j ОТ 1 ДО i-1 ПО 1
                                x[i]:=x[i]+a[i][j]*x[j]
                        ЦИКЛ «Элемент» ДЛЯ j ОТ i ДО n ПО 1
                                x[i]:=x[i]+a[i][j]*y[j]
                f:=0
                ЦИКЛ «Строки» ДЛЯ i ОТ 1 ДО n ПО 1
                        ЕСЛИ (x[i]-y[i]>f) f:=x[i]-y[i]
                        ЕСЛИ (y[i]-x[i]>f) f:=y[i]-x[i]
                f:=f*g/(1-g)
                k:=k+1
        ПЕЧАТЬ x, k
КОНЕЦ
```

2.13. СОБСТВЕННЫЕ ЗНАЧЕНИЯ И СОБСТВЕННЫЕ ВЕКТОРЫ МАТРИЦ

Рассмотрим матрицу $A_{n\times n}$ в n-мерном вещественном пространстве R^n векторов $x = (x_1, x_2, \ldots, x_n)^T$.

Определение. Собственным вектором x матрицы A называется ненулевой вектор ($x \neq \vartheta$), удовлетворяющий равенству:

$$Ax = \lambda x,$$

где λ — *собственное значение* матрицы A, соответствующее рассматриваемому собственному вектору x.

Собственные значения матрицы A с действительными элементами могут быть вещественными различными, вещественными кратными, комплексными попарно сопряженными, комплексными кратными.

Классический аналитический способ нахождения собственных значений и собственных векторов известен и заключается в следующем. Для однородной СЛАУ $(A - \lambda E)x = \vartheta$ ненулевые решения ($x \neq \vartheta$, а именно такие решения и находятся) имеют место только при $\det(A - \lambda E) = 0$.

Определение. Уравнение $\det(A - \lambda E) = 0$ называют *характеристическим уравнением*, а выражение в левой части — *характеристическим многочленом*.

Затем каким-либо способом находят решения $\lambda_1, \lambda_2, \ldots, \lambda_n$ полученного алгебраического уравнения n-й степени (предположим, что они вещественны и различны), после чего, решая для различных собственных значений $\lambda_j, j = 1, \ldots, n$ однородную СЛАУ $(A - \lambda_j E)x^j = \vartheta, j = 1, \ldots, n$, получают линейно независимые собственные векторы $x^j, j = 1, \ldots, n$, которые соответствуют собственным значениям $\lambda_j, j = 1, \ldots, n$. Обычно полученные собственные векторы $x^j, j = 1, \ldots, n$ затем для однозначности нормируются.

Попарно различным собственным значениям соответствуют линейно независимые собственные векторы; k-кратному корню характеристического уравнения, построенного для данной матрицы $A_{n \times n}$, соответствует не более k линейно независимых собственных векторов.

Пример. Аналитически вычислить собственные значения и собственные векторы матрицы $A = \begin{pmatrix} 1 & 2 \\ 2 & -2 \end{pmatrix}$.

Запишем характеристическое уравнение $\det(A - \lambda E) = 0$:

$$\det(A - \lambda E) = \det\left[\begin{pmatrix} 1 & 2 \\ 2 & -2 \end{pmatrix} - \begin{pmatrix} \lambda & 0 \\ 0 & \lambda \end{pmatrix}\right] = \det\begin{pmatrix} 1 - \lambda & 2 \\ 2 & -2 - \lambda \end{pmatrix} = 0,$$

$$(1-\lambda)(-2-\lambda)-2\cdot 2=0\,,\ \lambda^2+\lambda-6=0\,,\ \lambda_1=2,\lambda_2=-3\,.$$

Собственным значениям $\lambda_1=2,\lambda_2=-3$ соответствуют следующие нормированные (по евклидовой норме) собственные векторы (решения однородных СЛАУ $\left(A-\lambda_j E\right)x^j=\vartheta, j=1,\ldots,n$):

$$\begin{pmatrix}1-\lambda_1 & 2\\ 2 & -2-\lambda_1\end{pmatrix}\begin{pmatrix}x_1^1\\ x_2^1\end{pmatrix}=\begin{pmatrix}-1 & 2\\ 2 & -4\end{pmatrix}\begin{pmatrix}x_1^1\\ x_2^1\end{pmatrix}=\begin{pmatrix}0\\ 0\end{pmatrix},$$

$$-x_1^1+2x_2^1=0\,,\ \begin{pmatrix}x_1^1\\ x_2^1\end{pmatrix}=\begin{pmatrix}\dfrac{2}{\sqrt{5}}\\ \dfrac{1}{\sqrt{5}}\end{pmatrix};$$

$$\begin{pmatrix}1-\lambda_2 & 2\\ 2 & -2-\lambda_2\end{pmatrix}\begin{pmatrix}x_1^2\\ x_2^2\end{pmatrix}=\begin{pmatrix}4 & 2\\ 2 & 1\end{pmatrix}\begin{pmatrix}x_1^2\\ x_2^2\end{pmatrix}=\begin{pmatrix}0\\ 0\end{pmatrix},$$

$$2x_1^2+x_2^2=0\,,\ \begin{pmatrix}x_1^2\\ x_2^2\end{pmatrix}=\begin{pmatrix}-\dfrac{1}{\sqrt{5}}\\ \dfrac{2}{\sqrt{5}}\end{pmatrix}.$$

Легко видеть, что скалярное произведение полученных векторов $x^1\cdot x^2=\dfrac{2}{\sqrt{5}}\cdot\dfrac{1}{\sqrt{5}}+\dfrac{-1}{\sqrt{5}}\cdot\dfrac{2}{\sqrt{5}}=0$, т.е. собственные векторы ортогональны.

Если количество линейно независимых собственных векторов матрицы $A_{n\times n}$ совпадает с размерностью пространства R^n, то их можно принять за новый базис, в котором матрица $A_{n\times n}$ примет диагональный вид $\Lambda=U^{-1}\cdot A\cdot U$, на главной диагонали которой находятся собственные значения, а столбцы матрицы преобразования U являются собственными векторами матрицы A.

Определение. Матрицы Λ и A, удовлетворяющие указанному равенству $\Lambda=U^{-1}\cdot A\cdot U$, называются *подобными*. Собственные значения и определитель подобных матриц Λ и A совпадают. Собственные векторы матрицы A связаны с собственными векторами матрицы Λ соотношением $x=\Lambda y$.

Симметрическая матрица A ($A = A^T$) имеет полный спектр $\lambda_j, j = 1,\ldots,n$ вещественных собственных значений; k-кратному корню характеристического уравнения симметрической матрицы соответствует ровно k линейно независимых собственных векторов.

Таким образом, симметрическая матрица A всегда имеет ровно n ортогональных собственных векторов, приняв которые за новый базис (т.е. построив матрицу преобразования U, взяв в качестве её столбцов координатные столбцы собственных векторов), можно преобразовать симметрическую матрицу к диагональному виду с помощью преобразования $\Lambda = U^{-1} \cdot A \cdot U$.

Определение. Матрица U, для которой верно $U^{-1} = U^T$, называется *ортогональной*. Все собственные значения ортогональной матрицы по модулю равны единице, строки (столбцы) ортогональной матрицы U попарно ортогональны, суммы квадратов элементов каждой строки (столбца) ортогональной матрицы равны единице, определитель ортогональной матрицы равен ± 1. Если матрица U ортогональна, то и матрица U^{-1} тоже ортогональна.

Для симметрической матрицы A матрица преобразования U является ортогональной, и, следовательно, её преобразование к диагональному виду имеет вид: $\Lambda = U^T \cdot A \cdot U$.

Пример. Проверим подобие матриц $A = \begin{pmatrix} 1 & 2 \\ 2 & -2 \end{pmatrix}$ и $\Lambda = \begin{pmatrix} 2 & 0 \\ 0 & -3 \end{pmatrix}$.

Составим матрицу преобразования U, взяв в качестве её столбцов координатные столбцы собственных векторов матрицы A:

$$U = \begin{pmatrix} \frac{2}{\sqrt{5}} & -\frac{1}{\sqrt{5}} \\ \frac{1}{\sqrt{5}} & \frac{2}{\sqrt{5}} \end{pmatrix}.$$

Действительно:

$$U^T \cdot A \cdot U = \begin{pmatrix} \frac{2}{\sqrt{5}} & \frac{1}{\sqrt{5}} \\ -\frac{1}{\sqrt{5}} & \frac{2}{\sqrt{5}} \end{pmatrix} \cdot \begin{pmatrix} 1 & 2 \\ 2 & -2 \end{pmatrix} \cdot \begin{pmatrix} \frac{2}{\sqrt{5}} & -\frac{1}{\sqrt{5}} \\ \frac{1}{\sqrt{5}} & \frac{2}{\sqrt{5}} \end{pmatrix} =$$

$$= \begin{pmatrix} \frac{4}{\sqrt{5}} & \frac{2}{\sqrt{5}} \\ \frac{3}{\sqrt{5}} & -\frac{6}{\sqrt{5}} \end{pmatrix} \cdot \begin{pmatrix} \frac{2}{\sqrt{5}} & -\frac{1}{\sqrt{5}} \\ \frac{1}{\sqrt{5}} & \frac{2}{\sqrt{5}} \end{pmatrix} = \begin{pmatrix} \frac{10}{5} & \frac{0}{5} \\ \frac{0}{5} & \frac{-15}{5} \end{pmatrix} = \begin{pmatrix} 2 & 0 \\ 0 & -3 \end{pmatrix} = \Lambda .$$

2.14. МЕТОД ВРАЩЕНИЙ ЯКОБИ

Метод вращений Якоби применим только для действительных симметрических матриц A_{nxn} ($A = A^T$) и решает полную проблему собственных значений и собственных векторов таких матриц (в том смысле, что находятся все искомые значения и векторы). Он основан на отыскании с помощью итерационных процедур матрицы U в преобразовании подобия $\Lambda = U^{-1} \cdot A \cdot U$, сохраняющем собственные числа; а поскольку для симметрических матриц A матрица преобразования подобия U является ортогональной ($U^{-1} = U^T$), то $\Lambda = U^T \cdot A \cdot U$, где Λ — диагональная матрица с собственными значениями на главной диагонали:

$$\Lambda = \begin{pmatrix} \lambda_1 & \dots & 0 \\ \dots & \ddots & \dots \\ 0 & \dots & \lambda_n \end{pmatrix}.$$

Пусть дана симметрическая матрица A. Требуется для неё вычислить с точностью ε все собственные значения и соответствующие им собственные векторы. Итерационный алгоритм метода вращений имеет следующий вид:

$$\begin{cases} A^{(0)} = A, V^{(0)} = E \\ A^{(1)} = U^{(0)T} \cdot A^{(0)} \cdot U^{(0)}, V^{(1)} = V^{(0)} \cdot U^{(0)} \\ A^{(2)} = U^{(1)T} \cdot A^{(1)} \cdot U^{(1)}, V^{(2)} = V^{(1)} \cdot U^{(1)} \\ \dots\dots\dots\dots\dots\dots \\ A^{(k)} = U^{(k-1)T} \cdot A^{(k-1)} \cdot U^{(k-1)}, V^{(k)} = V^{(k-1)} \cdot U^{(k-1)} \end{cases},$$

где k — число исполненных итераций.

Пусть известна матрица $A^{(k)}$ на k-й итерации (для $k = 0$ $A^{(0)} = A$).

Во-первых, выбирается максимальный по модулю недиагональный элемент $a_{ij}^{(k)}$ матрицы $A^{(k)}$: $\left|a_{ij}^{(k)}\right| = \max\limits_{l<m} \left|a_{lm}^{(k)}\right|$.

Во-вторых, ставится задача найти такую ортогональную матрицу $U^{(k)}$, чтобы в результате преобразования подобия $A^{(k+1)} = U^{(k)T} \cdot A^{(k)} \cdot U^{(k)}$ произошло обнуление элемента $a_{ij}^{(k+1)}$ матрицы $A^{(k+1)}$. В качестве ортогональной матрицы выбирается матрица элементарного вращения (дающая название методу), осуществляющая поворот на подлежащий определению угол вращения $\varphi^{(k)}$ в плоскости (x_i, x_j) в n-мерном вещественном пространстве R^n. Матрица элементарного вращения имеет в пространстве R^n следующий вид:

$$U^{(k)} = \begin{array}{c} \begin{matrix} & & i & & j & & \end{matrix} \\ \begin{pmatrix} 1 & & \vdots & & \vdots & & \\ & \ddots & \vdots & & \vdots & 0 & \\ & & 1 & \vdots & \vdots & & \\ \cdots\cdots & & \cos\varphi^{(k)} & \cdots\cdots & -\sin\varphi^{(k)} & \cdots\cdots & \\ & & \vdots & 1 & \vdots & & \\ & & \vdots & \ddots & \vdots & & \\ & & \vdots & & 1 & \vdots & \\ \cdots\cdots & & \sin\varphi^{(k)} & \cdots\cdots & \cos\varphi^{(k)} & \cdots\cdots & \\ & & \vdots & & \vdots & 1 & \\ & 0 & \vdots & & \vdots & & \ddots \\ & & \vdots & & \vdots & & 1 \end{pmatrix} \end{array} \begin{matrix} \\ \\ \\ i \\ \\ \\ \\ j \\ \\ \\ \end{matrix} .$$

В матрице вращения $U^{(k)}$ на пересечении i-й строки и j-го столбца находится элемент $u_{ij}^{(k)} = -\sin\varphi^{(k)}$, симметрично относительно главной диагонали (j-я строка, i-й столбец) расположен элемент $u_{ji}^{(k)} = \sin\varphi^{(k)}$; диагональные элементы $u_{ii}^{(k)}$ и $u_{jj}^{(k)}$ равны соответственно $u_{ii}^{(k)} = u_{jj}^{(k)} = \cos\varphi^{(k)}$; другие диагональные элементы равны единице $u_{mm}^{(k)} = 1, m = 1, \ldots, n,\ m \neq i,\ m \neq j$; остальные элементы в матрице вращения равны нулю.

Угол вращения $\varphi^{(k)}$ определяется из условия обнуления элемента $a_{ij}^{(k+1)} = 0$ матрицы $A^{(k+1)}$. Его легко выразить, если рассмотреть процедуру поворота только в плоскости (x_i, x_j) пространства R^n:

$$\begin{pmatrix} \cos\varphi & \sin\varphi \\ -\sin\varphi & \cos\varphi \end{pmatrix} \cdot \begin{pmatrix} a_{ii} & a_{ij} \\ a_{ji} & a_{jj} \end{pmatrix} \cdot \begin{pmatrix} \cos\varphi & -\sin\varphi \\ \sin\varphi & \cos\varphi \end{pmatrix} =$$

$$= \begin{pmatrix} \cos\varphi & \sin\varphi \\ -\sin\varphi & \cos\varphi \end{pmatrix} \cdot \begin{pmatrix} a_{ii}\cos\varphi + a_{ij}\sin\varphi & -a_{ii}\sin\varphi + a_{ij}\cos\varphi \\ a_{ji}\cos\varphi + a_{jj}\sin\varphi & -a_{ji}\sin\varphi + a_{jj}\cos\varphi \end{pmatrix},$$

$$-a_{ii}\sin\varphi\cos\varphi + a_{ij}\cos^2\varphi - a_{ji}\sin^2\varphi + a_{jj}\sin\varphi\cos\varphi = 0\ ,$$

$$a_{ij}(\cos^2\varphi - \sin^2\varphi) + (a_{jj} - a_{ii})\sin\varphi\cos\varphi = 0,$$

$$a_{ij}\cos 2\varphi + (a_{jj} - a_{ii})\frac{\sin 2\varphi}{2} = 0,\ 2a_{ij}\cos 2\varphi = (a_{ii} - a_{jj})\sin 2\varphi,$$

$$\begin{bmatrix} \tan 2\varphi = \dfrac{2a_{ij}}{(a_{ii} - a_{jj})},\ a_{ii} \neq a_{jj} \\ \cos 2\varphi = 0,\ a_{ii} = a_{jj} \end{bmatrix}, \begin{bmatrix} \varphi = \dfrac{1}{2}\arctan\dfrac{2a_{ij}}{(a_{ii} - a_{jj})},\ a_{ii} \neq a_{jj} \\ \varphi = \dfrac{\pi}{4},\ a_{ii} = a_{jj} \end{bmatrix}.$$

Таким образом, угол вращения $\varphi^{(k)} = \frac{1}{2}\arctan\frac{2a_{ij}^{(k)}}{a_{ii}^{(k)} - a_{jj}^{(k)}}$, причём если $a_{ii}^{(k)} = a_{jj}^{(k)}$, то в этом случае $\varphi^{(k)} = \frac{\pi}{4}$.

В результате строится новая матрица $A^{(k+1)} = U^{(k)T} \cdot A^{(k)} \cdot U^{(k)}$, в которой элемент $a_{ij}^{(k+1)} = a_{ji}^{(k+1)} \approx 0$. Очевидно, что матрица $A^{(k+1)}$ также симметрическая.

Кроме того, можно видеть, что в матрицах $A^{(k+1)}$ и $V^{(k+1)}$ за одну итерацию происходит изменение только элементов *i*-х и *j*-х строк и столбцов, что при желании позволяет значительно оптимизировать процессы перемножения матриц $A^{(k+1)} = U^{(k)T} \cdot A^{(k)} \cdot U^{(k)}$ и $V^{(k+1)} = V^{(k)} \cdot U^{(k)}$ (с порядка n^3 арифметических операций до $4n$).

В качестве критерия окончания итераций используется условие малости среднеквадратичного недиагональных элементов матрицы $A^{(k+1)}$:

$$\varepsilon^{(k+1)} = t\left(A^{(k+1)}\right) = \left(\sum_{l,m;l<m}\left(a_{lm}^{(k+1)}\right)^2\right)^{1/2}.$$

Если $\varepsilon^{(k+1)} = t\left(A^{(k+1)}\right) > \varepsilon$, то итерационный процесс продолжается. Если $\varepsilon^{(k+1)} = t\left(A^{(k+1)}\right) \le \varepsilon$, то итерационный процесс останавливается, и в качестве искомых собственных значений принимаются диагональные элементы матрицы $A^{(k+1)}$: $\lambda_1 \approx a_{11}^{(k+1)}$, $\lambda_2 \approx a_{22}^{(k+1)}, \ldots, \lambda_n \approx a_{nn}^{(k+1)}$.

Итоговая матрица $A^{(k+1)}$ может быть представлена как

$$A^{(k+1)} = U^{(k)T} \cdot A^{(k)} \cdot U^{(k)} =$$

$$= U^{(k)T}U^{(k-1)T}\ldots U^{(0)T} \cdot A^{(0)} \cdot U^{(0)}U^{(1)}\ldots U^{(k)} = V^{(k+1)T} \cdot A^{(0)} \cdot V^{(k+1)},$$

из чего следует, что координатными столбцами собственных векторов матрицы A в единичном базисе будут столбцы матрицы $V^{(k+1)} = U^{(0)}U^{(1)}\ldots U^{(k)}$, т.е. $v^1 = \left(v_{11}, v_{21}, \ldots, v_{n1}\right)^T$, $v^2 = \left(v_{12}, v_{22}, \ldots, v_{n2}\right)^T$,

..., $v^n = \left(v_{1n}, v_{2n}, ..., v_{nn}\right)^T$, причём эти собственные векторы будут ортогональны между собой:

$$\left(v^l, v^m\right) \approx 0, \ \ l \neq m \, .$$

Пример. Вычислить методом вращений Якоби собственные значения и собственные векторы матрицы $A = \begin{pmatrix} 4 & 2 & 1 \\ 2 & 5 & \mathbf{3} \\ 1 & 3 & 6 \end{pmatrix} = A^{(0)}$ с точностью $\varepsilon = 0.2$.

Отметим, что

$$\varepsilon^{(0)} = t\left(A^{(0)}\right) = \left(\sum_{l,m;\, l<m} \left(a_{lm}^{(0)}\right)^2 \right)^{1/2} = (2^2 + 1^2 + 3^2)^{1/2} = \sqrt{14} = 3.742 > \varepsilon \, .$$

Начнём итерационный процесс. Итерация $k = 1$. Выбираем максимальный по модулю недиагональный элемент матрицы $A^{(0)}$, т.е. находим $a_{ij}^{(0)}$, такой что $\left|a_{ij}^{(0)}\right| = \max\limits_{l<m} \left|a_{lm}^{(0)}\right|$. Им является элемент $a_{23}^{(0)} = 3$.

Находим соответствующую этому элементу матрицу вращения:

$$\varphi^{(0)} = \frac{1}{2} \arctan \frac{2 \cdot 3}{5 - 6} = -0.703 \, ,$$

$$U^{(0)} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\varphi^{(0)} & -\sin\varphi^{(0)} \\ 0 & \sin\varphi^{(0)} & \cos\varphi^{(0)} \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0.763 & 0.646 \\ 0 & -0.646 & 0.763 \end{pmatrix} .$$

Вычисляем матрицу $A^{(1)}$:

$$A^{(1)} = U^{(0)T} A^{(0)} U^{(0)} = \begin{pmatrix} 4 & 0.880 & \mathbf{2.056} \\ 0.880 & 2.459 & 0.000 \\ 2.056 & 0.000 & 8.541 \end{pmatrix}.$$

В полученной матрице с точностью до ошибок округления элемент $a_{23}^{(1)} = 0$. Сумма квадратов недиагональных элементов матрицы:

$$\varepsilon^{(1)} = t\left(A^{(1)}\right) = \left(\sum_{l,m;\, l<m} \left(a_{lm}^{(1)}\right)^2 \right)^{1/2} = (0.880^2 + 2.056^2 + 0^2)^{1/2} = 2.236 > \varepsilon \, ,$$

следовательно, итерационный процесс необходимо продолжить.

Итерация $k = 2$. Находим $a_{13}^{(1)} = 2.056$, $\left|a_{13}^{(1)}\right| = \max\limits_{l,m;\, l<m} \left|a_{lm}^{(1)}\right|$.

$$\varphi^{(1)} = \frac{1}{2}\arctan\frac{2\cdot 2.056}{4-8.541} = -0.368,$$

$$U^{(1)} = \begin{pmatrix} \cos\varphi^{(1)} & 0 & -\sin\varphi^{(1)} \\ 0 & 1 & 0 \\ \sin\varphi^{(1)} & 0 & \cos\varphi^{(1)} \end{pmatrix} = \begin{pmatrix} 0.933 & 0 & 0.360 \\ 0 & 1 & 0 \\ -0.360 & 0 & 0.933 \end{pmatrix}.$$

Вычисляем матрицу $A^{(2)}$:

$$A^{(2)} = U^{(1)T} A^{(1)} U^{(1)} = \begin{pmatrix} 3.208 & \mathbf{0.821} & 0.000 \\ 0.821 & 2.459 & 0.316 \\ 0.000 & 0.316 & 9.334 \end{pmatrix},$$

$$\varepsilon^{(2)} = t\left(A^{(2)}\right) = \left(\sum_{l,m;l<m} \left(a_{lm}^{(2)}\right)^2\right)^{1/2} = (0.821^2 + 0^2 + 0.316^2)^{1/2} = 0.880 > \varepsilon.$$

Итерация $k = 3$. Находим $a_{12}^{(2)} = 0.821, \left|a_{12}^{(2)}\right| = \max\limits_{l,m;\, l<m} \left|a_{lm}^{(2)}\right|$.

$$\varphi^{(2)} = \frac{1}{2}\arctan\frac{2\cdot 0.821}{3.208-2.459} = 0.571,$$

$$U^{(2)} = \begin{pmatrix} \cos\varphi^{(2)} & -\sin\varphi^{(2)} & 0 \\ \sin\varphi^{(2)} & \cos\varphi^{(2)} & 0 \\ 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} 0.841 & -0.541 & 0 \\ 0.541 & 0.841 & 0 \\ 0 & 0 & 1 \end{pmatrix}.$$

Вычисляем матрицу $A^{(3)}$:

$$A^{(3)} = U^{(2)T} A^{(2)} U^{(2)} = \begin{pmatrix} 3.735 & 0.000 & 0.171 \\ 0.000 & 1.931 & 0.266 \\ 0.171 & 0.266 & 9.334 \end{pmatrix},$$

$$\varepsilon^{(3)} = t\left(A^{(3)}\right) = \left(\sum_{l,m;l<m} \left(a_{lm}^{(3)}\right)^2\right)^{1/2} = (0^2 + 0.171^2 + 0.266^2)^{1/2} = 0.316 > \varepsilon.$$

Итерация $k = 4$. Находим $a_{23}^{(3)} = 0.266, \left|a_{23}^{(3)}\right| = \max\limits_{l,m;\, l<m} \left|a_{lm}^{(3)}\right|$.

$$\varphi^{(3)} = \frac{1}{2}\arctan\frac{2\cdot 0.266}{1.931-9.334} = -0.036,$$

$$U^{(3)} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\varphi^{(3)} & -\sin\varphi^{(3)} \\ 0 & \sin\varphi^{(3)} & \cos\varphi^{(3)} \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0.999 & 0.036 \\ 0 & -0.036 & 0.999 \end{pmatrix}.$$

Вычисляем матрицу $A^{(4)}$:

$$A^{(4)} = U^{(3)T} A^{(3)} U^{(3)} = \begin{pmatrix} 3.735 & -0.006 & 0.171 \\ -0.006 & 1.921 & 0.000 \\ 0.171 & 0.000 & 9.343 \end{pmatrix},$$

$$\varepsilon^{(4)} = t\left(A^{(4)}\right) = \left(\sum_{l,m;l<m} \left(a_{lm}^{(4)}\right)^2 \right)^{1/2} = ((-0.006)^2 + 0.171^2 + 0^2)^{1/2} = 0.171 < \varepsilon .$$

Теперь итерационный процесс можно остановить.

В качестве искомых собственных значений могут быть приняты диагональные элементы матрицы $A^{(4)}$: $\lambda_1 \approx 3.735$, $\lambda_2 \approx 1.921$, $\lambda_3 \approx 9.343$.

Собственные векторы определяются из произведения:

$$V^{(4)} = U^{(0)} U^{(1)} U^{(2)} U^{(3)} = \begin{pmatrix} 0.785 & -0.517 & 0.341 \\ 0.217 & 0.745 & 0.630 \\ -0.580 & -0.421 & 0.697 \end{pmatrix},$$

таким образом:

$$v^1 = \begin{pmatrix} 0.785 \\ 0.217 \\ -0.580 \end{pmatrix}, \; v^2 = \begin{pmatrix} -0.517 \\ 0.745 \\ -0.421 \end{pmatrix}, \; v^3 = \begin{pmatrix} 0.341 \\ 0.630 \\ 0.697 \end{pmatrix}.$$

Полученные собственные векторы ортогональны в пределах заданной точности, т.е. $\left(v^1, v^2\right) = \left(v^1, v^3\right) = \left(v^2, v^3\right) \approx 0$.

Полученные собственные значения удовлетворяют уравнению $Ax = \lambda x$ в пределах заданной точности:

$$Av^1 - \lambda_1 v^1 = \begin{pmatrix} 0.062 \\ 0.103 \\ 0.122 \end{pmatrix}, \; Av^2 - \lambda_2 v^2 = \begin{pmatrix} -0.005 \\ -0.001 \\ 0.004 \end{pmatrix}, \; Av^3 - \lambda_3 v^3 = \begin{pmatrix} 0.134 \\ 0.037 \\ -0.099 \end{pmatrix}.$$

2.15. ПРОГРАММА #04

Ниже предлагается вариант алгоритма программы для нахождения собственных значений и собственных векторов симметрической матрицы методом вращений Якоби.

```
АЛГОРИТМ «Метод вращений Якоби»
ВХОД      n, a[n][n], e
ВЫХОД     i, j, p, l, m, f, g, h, u[n][n], b[n][n], w[n], v[n][n], k
```

```
НАЧАЛО
        ЦИКЛ «Строки» ДЛЯ i ОТ 1 ДО n ПО 1
                ЦИКЛ «Столбцы» ДЛЯ j ОТ 1 ДО n ПО 1
                        ЕСЛИ (i=j) v[i][j]:=1 ИНАЧЕ v[i][j]:=0
        f:=0
        ЦИКЛ «Строки» ДЛЯ i ОТ 1 ДО n ПО 1
                ЦИКЛ «Столбцы» ДЛЯ j ОТ i+1 ДО n ПО 1
                        f:=f+a[i][j]*a[i][j]
        f:=КОРЕНЬ(f)
        k:=0
        ЦИКЛ «Итерация» ПОКА (f>e)
                g:=0, l=1, m=2
                ЦИКЛ «Строки» ДЛЯ i ОТ 1 ДО n ПО 1
                        ЦИКЛ «Столбцы» ДЛЯ j ОТ i+1 ДО n ПО 1
                                ЕСЛИ (a[i][j]>g) g:=a[i][j], l=i, m=j
                                ЕСЛИ (-a[i][j]>g) g:=-a[i][j] , l=i, m=j
                ЦИКЛ «Строки» ДЛЯ i ОТ 1 ДО n ПО 1
                        ЦИКЛ «Столбцы» ДЛЯ j ОТ 1 ДО n ПО 1
                                ЕСЛИ (i=j) u[i][j]:=1 ИНАЧЕ u[i][j]:=0
                ЕСЛИ (a[l][l]=a[m][m]) h:=ПИ()/4
                ИНАЧЕ h:=АРКТАНГЕНС(2*a[l][m]/(a[l][l]-a[m][m]))/2
                u[l][l]:=КОСИНУС(h)
                u[l][m]:=-СИНУС(h)
                u[m][l]:=СИНУС(h)
                u[m][m]:=КОСИНУС(h)
                ЦИКЛ «Строки» ДЛЯ i ОТ 1 ДО n ПО 1
                        ЦИКЛ «Столбцы» ДЛЯ j ОТ 1 ДО n ПО 1
                                ЕСЛИ (i=l)ИЛИ(i=m)ИЛИ(j=l)ИЛИ(j=m)
                                        b[i][j]:=0
                                        ЦИКЛ «Элемент» ДЛЯ p ОТ 1 ДО n ПО 1
                                                b[i][j]:=b[i][j]+u[p][i]*a[p][j]
                ЦИКЛ «Строки» ДЛЯ i ОТ 1 ДО n ПО 1
                        ЦИКЛ «Столбцы» ДЛЯ j ОТ 1 ДО n ПО 1
                                ЕСЛИ (i=l)ИЛИ(i=m)ИЛИ(j=l)ИЛИ(j=m)
                                        a[i][j]:=0
                                        ЦИКЛ «Элемент» ДЛЯ p ОТ 1 ДО n ПО 1
```

```
                                    a[i][j]:=a[i][j]+b[i][p]*u[p][j]
            ЦИКЛ «Строки» ДЛЯ i ОТ 1 ДО n ПО 1
                ЦИКЛ «Столбцы» ДЛЯ j ОТ 1 ДО n ПО 1
                    ЕСЛИ (i=l)ИЛИ(i=m)ИЛИ(j=l)ИЛИ(j=m)
                        b[i][j]:=0
                        ЦИКЛ «Элемент» ДЛЯ p ОТ 1 ДО n ПО 1
                            b[i][j]:=b[i][j]+v[i][p]*u[p][j]
            ЦИКЛ «Строки» ДЛЯ i ОТ 1 ДО n ПО 1
                ЦИКЛ «Столбцы» ДЛЯ j ОТ 1 ДО n ПО 1
                    ЕСЛИ (i=l)ИЛИ(i=m)ИЛИ(j=l)ИЛИ(j=m)
                        v[i][j]:=b[i][j]
            f:=0
            ЦИКЛ «Строки» ДЛЯ i ОТ 1 ДО n ПО 1
                ЦИКЛ «Столбцы» ДЛЯ j ОТ i+1 ДО n ПО 1
                    f:=f+a[i][j]*a[i][j]
            f:=КОРЕНЬ(f)
            k:=k+1
        ЦИКЛ «Строки» ДЛЯ i ОТ 1 ДО n ПО 1
            w[i]:=a[i][i]
        ПЕЧАТЬ w, v, k
КОНЕЦ
```

2.16. МЕТОД СТЕПЕННЫХ ИТЕРАЦИЙ

Рассмотренный выше метод вращения Якоби решает полную проблему собственных значений и собственных векторов матриц (симметрических). Однако зачастую не нужно находить все собственные значения (спектр) и все собственные векторы, а стоит задача найти лишь максимальное по модулю собственное значение матрицы (спектральный радиус) и соответствующий ему собственный вектор. Такая задача называется частичной проблемой собственных значений и собственных векторов. Для её решения существует метод степенных итераций (или степенной метод).

Пусть дана произвольная матрица A и пусть её собственные значения упорядочены по абсолютным величинам: $|\lambda_1| > |\lambda_2| \geq \ldots \geq |\lambda_n|$. Тогда, выбрав некоторый начальный вектор $y^{(0)}$, например, вектор

$y^{(0)} = (1\,1 \ldots 1)^T$, компоненты которого равны единице, можно для определения λ_1 построить следующий итерационный процесс:

$$y^{(k)} = Ay^{(k-1)},\ \lambda_1^{(k)} = \frac{y_j^{(k)}}{y_j^{(k-1)}},$$

где $y_j^{(k-1)}$, $y_j^{(k)}$ — соответствующие компоненты векторов $y^{(k-1)}$, $y^{(k)}$. При этом в качестве номера j может использоваться любое число из диапазона $j = 1, \ldots, n$.

Поскольку вектор $y^{(k)}$ на k-й итерации может быть представлен в виде $y^{(k)} = Ay^{(k-1)} = \ldots = A^k y^{(0)}$, рассматриваемый итерационный процесс получил название метод степенных итераций. Из-за своей конструкции метод применяется в первую очередь для разреженных матриц.

При выполнении условия $|\lambda_1| > |\lambda_2| \geq \ldots \geq |\lambda_n|$ итерационный процесс сходится к искомому собственному значению λ_1 и соответствующему ему собственному вектору $y^{(k)}$, причём скорость сходимости определяется отношением $\frac{|\lambda_2|}{|\lambda_1|}$ (чем оно меньше, тем выше скорость сходимости).

В качестве критерия завершения вычислений используется следующее условие:

$$\varepsilon^{(k)} = \left|\lambda_1^{(k)} - \lambda_1^{(k-1)}\right| \leq \varepsilon,$$

где ε — заданная вычислителем точность расчёта.

Приведённый алгоритм обладает существенным практическим недостатком, связанным с сильным изменением компонентов итерируемого вектора $y^{(k)}$ в ходе итерационного процесса. Очевидно, что раз $\left|\frac{y_j^{(k)}}{y_j^{(k-1)}}\right| \approx |\lambda_1|$, то это приводит к неограниченному возрастанию (при $|\lambda_1| > 1$) или убыванию (при $|\lambda_1| < 1$) компонентов $y^{(k)}$ по мере увеличения числа итераций k. Во избежание этого при проведении компьютерных расчётов обычно применяется метод степенных итераций с нормировкой итерируемого вектора. С этой целью алгоритм модифицируется следующим образом:

$$z^{(k)} = Ay^{(k-1)},\ \lambda_1^{(k)} = \frac{z_j^{(k)}}{y_j^{(k-1)}},\ y^{(k)} = \frac{z^{(k)}}{\left\|z^{(k)}\right\|}.$$

При этом в качестве начального приближения $y^{(0)}$ берётся вектор с единичной нормой.

Пример. Вычислить степенным методом (с нормировкой итерируемого вектора) спектральный радиус матрицы $A = \begin{pmatrix} 4 & 2 & 1 \\ 2 & 5 & 3 \\ 1 & 3 & 6 \end{pmatrix}$ с точностью $\varepsilon = 0.2$.

В качестве начального приближения собственного вектора возьмём $y^{(0)} = (1 \quad 1 \quad 1)^T$. Реализуем итерационный процесс, полагая $j = 1$:

$$z^{(k)} = Ay^{(k-1)},\ \lambda_1^{(k)} = \frac{z_1^{(k)}}{y_1^{(k-1)}},\ y^{(k)} = \frac{z^{(k)}}{\left\|z^{(k)}\right\|_c}.$$

$$z^{(1)} = Ay^{(0)} = (7 \quad 10 \quad 10)^T,\ \lambda_1^{(1)} = \frac{z_1^{(1)}}{y_1^{(0)}} = \frac{7}{1} = 7,$$

$$y^{(1)} = \frac{z^{(1)}}{\left\|z^{(1)}\right\|_c} = (0.7 \quad 1 \quad 1)^T;$$

$$z^{(2)} = Ay^{(1)} = (5.8 \quad 9.4 \quad 9.7)^T,\ \lambda_1^{(2)} = \frac{z_1^{(2)}}{y_1^{(1)}} = 8.286,$$

$$y^{(2)} = \frac{z^{(2)}}{\left\|z^{(2)}\right\|_c} = (0.598 \quad 0.969 \quad 1)^T,\ \varepsilon^{(2)} = \left|\lambda_1^{(2)} - \lambda_1^{(1)}\right| = 1.286 > \varepsilon;$$

$$z^{(3)} = Ay^{(2)} = (5.330 \quad 9.041 \quad 9.505)^T,\ \lambda_1^{(3)} = \frac{z_1^{(3)}}{y_1^{(2)}} = 8.914,$$

$$y^{(3)} = \frac{z^{(3)}}{\left\|z^{(3)}\right\|_c} = (0.561 \quad 0.951 \quad 1)^T,\ \varepsilon^{(3)} = \left|\lambda_1^{(3)} - \lambda_1^{(2)}\right| = 0.628 > \varepsilon;$$

$$z^{(4)} = Ay^{(3)} = (5.145 \quad 8.877 \quad 9.414)^T,\ \lambda_1^{(4)} = \frac{z_1^{(4)}}{y_1^{(3)}} = 9.146,$$

$$y^{(4)} = \frac{z^{(4)}}{\left\|z^{(4)}\right\|_c} = (0.547 \quad 0.943 \quad 1)^T,\ \varepsilon^{(4)} = \left|\lambda_1^{(4)} - \lambda_1^{(3)}\right| = 0.262 > \varepsilon;$$

$$z^{(5)} = Ay^{(4)} = (5.072 \quad 8.808 \quad 9.375)^T,\ \lambda_1^{(5)} = \frac{z_1^{(5)}}{y_1^{(4)}} = 9.280,$$

$$y^{(5)} = \frac{z^{(5)}}{\left\|z^{(5)}\right\|_c} = (0.541 \quad 0.939 \quad 1)^T, \ \varepsilon^{(5)} = \left|\lambda_1^{(5)} - \lambda_1^{(4)}\right| = 0.104 < \varepsilon .$$

Таким образом, полученное на 5-й итерации значение $\lambda_1^{(5)} = 9.280$ удовлетворяет заданной точности и может быть взято в качестве приближённого значения λ_1, и искомое значение спектрального радиуса $\rho(A) = \max_i |\lambda_i| = |\lambda_1| \approx \left|\lambda_1^{(5)}\right| = 9.280$.

Отметим, что это значение $\lambda_1^{(5)}$ с точностью $\varepsilon = 0.2$ совпадает со значением $\lambda_3 \approx 9.343$, вычисленным методом вращений Якоби, так же как и соответствующий ему собственный вектор $y^{(5)} = (0.541 \quad 0.939 \quad 1)^T$ представляет собой вектор $v^3 = (0.341 \quad 0.630 \quad 0.697)^T$ после перенормировки.

2.17. ПРОГРАММА #05

Ниже предлагается вариант алгоритма программы для нахождения максимального по модулю собственного значения матрицы и соответствующего ему собственного вектора методом степенных итераций.

```
АЛГОРИТМ «Метод степенных итераций»
ВХОД        n, a[n][n], e
ВЫХОД       i, j, f, r, z[n], w, y[n], k
НАЧАЛО
    w:=0
    ЦИКЛ «Строки» ДЛЯ i ОТ 1 ДО n ПО 1
            y[i]:=1
    f:=2*e
    k:=0
    ЦИКЛ «Итерация» ПОКА (f>e)
            r:=0
            ЦИКЛ «Строки» ДЛЯ i ОТ 1 ДО n ПО 1
                    z[i]:=0
                    ЦИКЛ «Элемент» ДЛЯ j ОТ 1 ДО n ПО 1
                            z[i]:=z[i]+a[i][j]*y[j]
                    ЕСЛИ (z[i]>r) r=z[i]
                    ЕСЛИ (-z[i]>r) r=-z[i]
            f:=w
```

```
            w:=z[1]/y[1]
            ЦИКЛ «Строки» ДЛЯ i ОТ 1 ДО n ПО 1
                  y[i]:=z[i]/r
            f:=w-f
            ЕСЛИ (f<0) f:=-f
            k:=k+1
      ПЕЧАТЬ w, y, k
КОНЕЦ
```

2.18. *QR*-РАЗЛОЖЕНИЕ МАТРИЦ

Для решения полной проблемы собственных значений для несимметричных матриц $A_{n\times n}$ эффективным является подход с использованием *QR*-разложения матриц, основанный на приведении матриц к подобным, имеющим треугольный или квазитреугольный вид, и позволяющий находить как вещественные, так и комплексные собственные значения. В основе *QR*-алгоритма лежит представление матрицы в виде $A = QR$, где Q — ортогональная матрица ($Q^{-1} = Q^T$), а R — верхнетреугольная. Такое разложение существует для любой квадратной матрицы.

Одним из возможных подходов к построению *QR*-разложения является использование преобразования Хаусхолдера, позволяющего обратить в нуль группу поддиагональных элементов столбца матрицы. Преобразование Хаусхолдера осуществляется с использованием матрицы Хаусхолдера, имеющей следующий вид:

$$H = E - \frac{2}{v^T v} vv^T ,$$

где v — произвольный ненулевой вектор-столбец, E — единичная матрица, vv^T — квадратная матрица того же размера.

Легко убедиться, что любая матрица такого вида является симметричной и ортогональной. При этом произвол в выборе вектора v даёт возможность построить матрицу, отвечающую некоторым дополнительным требованиям.

Рассмотрим случай, когда необходимо обратить в нуль все элементы какого-либо вектора кроме первого, т.е. построить матрицу Хаусхолдера такую, что $\tilde{b} = Hb$, $b = (b_1, b_2, ..., b_n)^T$, $\tilde{b} = (\tilde{b}_1, 0, ..., 0)^T$.

Тогда вектор v определяется следующим образом: $v = b + sign(b_1)\|b\|_2 e_1$, где $\|b\|_2 = \left(\sum_i b_i^2\right)^{1/2}$ — евклидова норма вектора, и $e_1 = (1, 0, ..., 0)^T$.

Применяя описанную процедуру с целью обнуления поддиагональных элементов для каждого из столбцов исходной матрицы, можно за фиксированное число шагов $n-1$ получить её QR-разложение.

Действительно, положим $A_0 = A$ и на первом шаге построим преобразование Хаусхолдера H_1 ($A_1 = H_1 A_0$), переводящее матрицу A_0 в матрицу A_1 с нулевыми элементами первого столбца под главной диагональю:

$$A_0 = \begin{pmatrix} a_{11}^0 & a_{12}^0 & \dots & a_{1n}^0 \\ a_{21}^0 & a_{22}^0 & \dots & a_{2n}^0 \\ \dots & \dots & \dots & \dots \\ a_{n1}^0 & a_{n2}^0 & \dots & a_{nn}^0 \end{pmatrix} \xrightarrow[H_1]{} A_1 = \begin{pmatrix} a_{11}^1 & a_{12}^1 & \dots & a_{1n}^1 \\ 0 & a_{22}^1 & \dots & a_{2n}^1 \\ \dots & \dots & \dots & \dots \\ 0 & a_{n2}^1 & \dots & a_{nn}^1 \end{pmatrix}.$$

Ясно, что матрица Хаусхолдера H_1 должна определяться по первому столбцу матрицы A_0, т.е. в качестве вектора b в выражении берётся вектор $(a_{11}^0, a_{21}^0, ..., a_{n1}^0)^T$ размерности n. Тогда компоненты вектора v вычисляются как

$$v_1^1 = a_{11}^0 + sign(a_{11}^0)\left(\sum_{j=1}^{n} (a_{j1}^0)^2\right)^{1/2}, \quad v_i^1 = a_{i1}^0, \quad i = 2, \dots, n, \text{ и}$$

$$H_1 = E - 2\frac{v^1 v^{1T}}{v^{1T} v^1}.$$

На втором шаге рассматриваемого процесса строится преобразование Хаусхолдера H_2 ($A_2 = H_2 A_1$), обнуляющее расположенные ниже главной диагонали элементы второго столбца матрицы A_1. Взяв в качестве вектора b вектор $(a_{22}^1, a_{32}^1, ..., a_{n2}^1)^T$ размерности $n-1$, получим следующие выражения для компонентов вектора v:

$$v_1^2 = 0, \quad v_2^2 = a_{22}^1 + sign(a_{22}^1)\left(\sum_{j=2}^{n} (a_{j2}^1)^2\right)^{1/2}, \quad v_i^2 = a_{i1}^1, \quad i = 3, \dots, n.$$

Повторяя этот процесс ровно $n-1$ раз вплоть до $A_{n-1} = H_{n-1} H_{n-2} ... H_1 A$, получим искомое разложение $A = QR$, где

$$R = A_{n-1} \text{ и } Q = (H_{n-1} H_{n-2} ... H_1)^{-1} = (H_{n-1} H_{n-2} ... H_1)^T = H_1 H_2 ... H_{n-1}.$$

Следует отметить определённое сходство рассматриваемого процесса с алгоритмом метода Гаусса. Отличие заключается в том, что здесь обнуление поддиагональных элементов соответствующего столбца осуществляется с использованием ортогонального преобразования.

Процедура QR-разложения многократно используется в итерационном QR-алгоритме вычисления собственных значений произвольных матриц.

Строится следующий итерационный процесс:

$$A^{(0)} = A\,,\ A^{(0)} = Q^{(0)}R^{(0)}\,,\ A^{(1)} = R^{(0)}Q^{(0)}\,,\ \ldots,\ A^{(k)} = Q^{(k)}R^{(k)}\,,$$

$$A^{(k+1)} = R^{(k)}Q^{(k)}\,.$$

Таким образом, каждая итерация реализуется в два этапа. На первом этапе осуществляется QR-разложение матрицы $A^{(k)}$ в произведение ортогональной $Q^{(k)}$ и верхнетреугольной $R^{(k)}$ матриц, а на втором полученные матрицы перемножаются в обратном порядке.

Легко видеть, что матрицы $A^{(k+1)}$ и $A^{(k)}$ подобны. Действительно, учитывая ортогональность $Q^{(k)}$ ($Q^{(k)T}Q^{(k)} = E$), можно записать:

$$A^{(k+1)} = R^{(k)}Q^{(k)} = E \cdot R^{(k)}Q^{(k)} = Q^{(k)T}Q^{(k)} \cdot R^{(k)}Q^{(k)} = Q^{(k)T}A^{(k)}Q^{(k)}\,.$$

Соответственно, любая из матриц $A^{(k)}$ подобна матрице A.

При отсутствии у матрицы A_{nxn} кратных собственных значений при $k \to \infty$ последовательность $A^{(k)}$ сходится либо к верхнетреугольной матрице, если все собственные значения вещественны, либо к верхней квазитреугольной матрице, если имеются комплексно-сопряженные пары собственных значений. Таким образом, каждому вещественному собственному значению будет соответствовать столбец матрицы со стремящимися к нулю поддиагональными элементами, и само собственное значение принимается равным диагональному элементу данного столбца. Каждой комплексно-сопряженной паре соответствует диагональный блок размерностью 2×2, т.е. матрица $A^{(k)}$ имеет блочно-диагональную структуру. Принципиально то, что элементы этих блоков изменяются от итерации к итерации без видимой закономерности, в то время как комплексно-сопряженные собственные значения $\lambda_{1,2}^{(k)}$, определяемые каждым блоком, имеют

тенденцию к сходимости. Это обстоятельство необходимо учитывать при формировании критерия выхода из итерационного процесса. Если в ходе итераций прослеживается комплексно-сопряженная пара собственных значений, соответствующая блоку матрицы, образуемому элементами j-го и $(j+1)$-го столбцов $a_{jj}^{(k)}, a_{j(j+1)}^{(k)}, a_{(j+1)j}^{(k)}, a_{(j+1)(j+1)}^{(k)}$, то, несмотря на значительное изменение в ходе итераций самих этих элементов, собственные значения, соответствующие данному блоку, начиная с некоторого k отличаются незначительно. Они определяются из решения квадратного уравнения

$$(a_{jj}^{(k)} - \lambda^{(k)})(a_{(j+1)(j+1)}^{(k)} - \lambda^{(k)}) = a_{j(j+1)}^{(k)} a_{(j+1)j}^{(k)}.$$

Например, матрица

$$A^{(k)} = \begin{pmatrix} x & x & x & x & x & x \\ 0 & x & x & x & x & x \\ 0 & x & x & x & x & x \\ 0 & 0 & 0 & x & x & x \\ 0 & 0 & 0 & 0 & x & x \\ 0 & 0 & 0 & 0 & x & x \end{pmatrix}_{6\times 6}$$

имеет два действительных собственных значения $\lambda_1 \approx a_{11}^{(k)}, \lambda_4 \approx a_{44}^{(k)}$ и две пары комплексно-сопряжённых собственных значений $\lambda_{2,3}$ и $\lambda_{5,6}$, определяемых из уравнений $(a_{22}^{(k)} - \lambda)(a_{33}^{(k)} - \lambda) = a_{23}^{(k)} a_{32}^{(k)}$ и $(a_{55}^{(k)} - \lambda)(a_{66}^{(k)} - \lambda) = a_{56}^{(k)} a_{65}^{(k)}$.

Поскольку невозможно заранее учесть, какие элементы матрицы A_{nxn}, лежащие на нижней диагонали, прилегающей к главной, стремятся к нулю при $k \to \infty$, то в качестве простейшего критерия окончания итераций можно использовать следующее неравенство:

$$\varepsilon^{(k)} = \left(\sum_{i>j+1}^{n} (a_{ij}^{(k)})^2 \right)^{1/2} \le \varepsilon .$$

Существенным недостатком QR-алгоритма вычисления собственных значений матриц является большое число операций (пропорциональное n^3), необходимое для QR-факторизации матрицы на каждой итерации.

3. ЧИСЛЕННЫЕ МЕТОДЫ И АЛГОРИТМЫ ЭЛЕМЕНТАРНОЙ АЛГЕБРЫ

В главе рассматриваются основные численные методы и алгоритмы вычислительной математики в области элементарной алгебры. Сначала излагаются численные методы уточнения корней нелинейных уравнений. В общем случае такие уравнения не имеют аналитических формул для своих корней или эти формулы слишком громоздки и неудобны для практического использования. В частности, в начале 19-го века было доказано, что нелинейные уравнения выше четвертой степени неразрешимы в радикалах, даже если использовать радикалы произвольной степени. Рассматриваемые численные методы уточнения корней нелинейных уравнений носят итерационный характер и в принципе пригодны для отыскания корней уравнений любого вида. Подробно изложены метод половинного деления (дихотомии), метод простой итерации, метод Ньютона (касательных) и метод секущих (хорд). Для уточнения корней системы нелинейных уравнений приводятся обобщения метода простой итерации и метода Ньютона, рассмотренных для одного нелинейного уравнения.

3.1. РЕШЕНИЕ НЕЛИНЕЙНЫХ УРАВНЕНИЙ

Пусть требуется найти решение нелинейного (алгебраического или трансцендентного) уравнения

$$f(x) = 0 ,$$

т.е. все те значения x (корни), которые обращают это уравнение в тождество.

Численное решение нелинейных уравнений вида $f(x) = 0$ заключается в нахождении значений x, удовлетворяющих данному уравнению с заданной точностью ε, и состоит из следующих основных этапов:

1) отделение (изоляция, локализация) корней уравнения;

2) уточнение с помощью некоторого вычислительного алгоритма конкретного выделенного корня с заданной точностью ε.

Целью первого этапа является нахождение отрезков из области определения функции $f(x)$, внутри которых содержится только один корень решаемого уравнения. Иногда вычислители ограничиваются рассмотрением лишь какой-нибудь части области определения, вызывающей по тем или иным соображениям интерес. В общем случае этап отделения корня не может быть алгоритмизирован. Для отдельных классов уравнений (наиболее известным из которых является класс алгебраических уравнений) разработаны специальные приёмы отделения корней, существенно облегчающие данное отделение. Нередко отделение корней нелинейных уравнений выполняется «вручную» с использованием всей возможной информации о функции $f(x)$. В ряде случаев приближённое значение корня может быть определено из физических соображений, если речь идет о решении нелинейного уравнения, связанного с конкретной прикладной задачей. В целом, для реализации данного этапа используются способы, которые можно разделить на графические и аналитические.

При аналитическом способе отделения корней полезна следующая теорема. Непрерывная строго монотонная функция $f(x)$ имеет, и притом единственный, нуль на отрезке $[a,b]$ тогда и только тогда, когда на его концах она принимает значения разных знаков. В случае если на концах интервала функция имеет одинаковые знаки, то на этом интервале корни либо отсутствуют, либо их чётное число. Достаточным признаком монотонности функции $f(x)$ на отрезке $[a,b]$ является сохранение знака производной функции.

Графический способ отделения корней, обладающий большой наглядностью, целесообразно использовать в том случае, когда имеется возможность построения графика функции $y = f(x)$. Наличие графика исходной функции даёт непосредственное представление о количестве и расположении нулей функции, что позволяет определить промежутки, внутри которых содержится только один корень. Если построение графика функции $y = f(x)$ вызывает затруднение, часто оказывается удобным преобразовать исходное уравнение к эквивалентному виду $f_1(x) = f_2(x)$ и построить графики функций $y = f_1(x)$ и $y = f_2(x)$. Абсциссы точек пересечения этих графиков будут соответствовать значениям корней решаемого уравнения.

Так или иначе, по завершении первого этапа должны быть определены отрезки, на каждом из которых содержится только один корень уравнения.

Пример. Осуществить изоляцию корней уравнения $f(x) = e^{2x} + 3x - 4 = 0$.

Применим аналитический способ. Найдём производную:

$$f'(x) = 2e^{2x} + 3 > 0 \text{ для } \forall x .$$

Функция $f(x)$ строго монотонно возрастающая и имеет не более одного корня. Для его локализации применим графический способ. Приведём исходное уравнение к следующему эквивалентному виду:

$$e^{2x} = 4 - 3x .$$

Теперь построим графики функций $f_1(x) = e^{2x}$ и $f_2(x) = 4 - 3x$ (рис. 2).

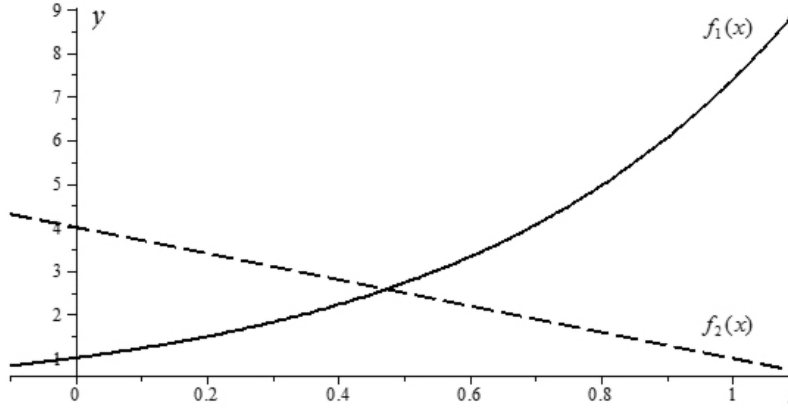

Рис. 2. Графический способ локализации корней уравнения

Из них определяем, что у решаемого уравнения имеется один корень (точка пересечения графиков), который находится в отрезке $0.4 \le x^* \le 0.6$ (т.к. $f_1(0.4) < f_2(0.4)$ и $f_1(0.6) > f_2(0.6)$). В дальнейшем осуществим уточнение значения корня в этом отрезке с требуемой точностью, пользуясь итерационными методами, приведёнными ниже.

На втором этапе уточнения рассматривается некоторый промежуток, который содержит только один корень решаемого уравнения. При нахождении корня используют два типа методов: прямые и итерационные. В прямых методах корень уравнения принципиально находится за конечное, заранее известное число операций. Прямыми методами удаётся решить только некоторые простейшие алгебраические и тригонометрические уравнения. Поэтому для уточнения корня с требуемой точностью ε обычно применяется какой-либо итерационный метод, заключающийся в построении числовой последовательности $x^{(k)}$, $k = 0,1,2,\ldots$, сходящейся в пределе $k \to \infty$ к искомому корню x^* исходного уравнения. Одна из точек промежутка принимается за начальное приближение корня. В этом случае точное решение — предел этой последовательности — принципиально не может быть достигнуто за конечное, заранее известное число операций.

Определение. Важной характеристикой итерационных методов является *порядок сходимости* процесса. Говорят, что метод имеет *n*-й порядок сходимости, если $\left| x^{(k)} - x^* \right| = C \left| x^{(k-1)} - x^* \right|^n$, где C — константа, не зависящая от n. При $n = 1$ имеем сходимость первого порядка, или линейную сходимость, при $n = 2$ — второго порядка, или квадратичную, и т.д.

Определение. Говорят, что метод является *n-шаговым*, если для построения итерационной последовательности $x^{(k)}$, $k = 0,1,2,\ldots$ необходимо вычислять функцию $f(x)$ в n предыдущих точках последовательности, например, одношаговым, если нужно вычислять функцию в одной точке последовательности, двухшаговым — если в двух точках, и т.д.

3.2. МЕТОД ПОЛОВИННОГО ДЕЛЕНИЯ (ДИХОТОМИИ)

Процесс уточнения корня уравнения $f(x) = 0$ на отрезке $[a,b]$ методом половинного деления при условии, что функция $f(x)$ непрерывна на этом отрезке и $f(a) \cdot f(b) < 0$, заключается в следующем.

Исходный отрезок $[a^{(0)}, b^{(0)}]$ делится пополам. Если $f\left(\frac{a+b}{2}\right) = 0$, то $x^* = \frac{a+b}{2}$ является корнем уравнения. Если $f\left(\frac{a+b}{2}\right) \neq 0$, то приближением корня является $x^{(0)} = \frac{a+b}{2}$, после чего выбирается

та из половин отрезка $\left[a, \frac{a+b}{2}\right]$ или $\left[\frac{a+b}{2}, b\right]$, на концах которой функция $f(x)$ имеет противоположные знаки. Новый отрезок $[a^{(1)}, b^{(1)}]$ снова делится пополам и проводится то же рассмотрение, и т.д.

В результате на каком-то этапе k либо находится точный корень уравнения, либо имеется последовательность вложенных друг в друга отрезков $[a^{(0)}, b^{(0)}]$, $[a^{(1)}, b^{(1)}]$, $[a^{(2)}, b^{(2)}]$, ..., $[a^{(k)}, b^{(k)}]$, для которых $f(a^{(k)}) \cdot f(b^{(k)}) < 0$, $k = 0, 1, 2, \ldots$, а приближением корня является середина последнего отрезка

$$x^{(k)} = \frac{a^{(k)} + b^{(k)}}{2}.$$

Итерационный процесс останавливается, если длина отрезка $[a^{(k)}, b^{(k)}]$ станет меньше 2ε, т.е.

$$\varepsilon^{(k)} = \frac{b^{(k)} - a^{(k)}}{2} = \frac{b-a}{2^{k+1}}.$$

Очевидно, что с увеличением k погрешность стремится к нулю не медленнее геометрической прогрессии со знаменателем 1/2. Дихотомия проста и надёжна, всегда сходится, хотя и медленно, и устойчива к ошибкам округления.

Пример. Осуществить уточнение значения корня уравнения $f(x) = e^{2x} + 3x - 4 = 0$ в отрезке $[0.4, 0.6]$ с точностью $\varepsilon = 10^{-3}$ методом половинного деления.

Результаты вычислений приведены в табл. 1.

Итоговое приближение корня $x^* \approx x^{(7)} = 0.4742$.

Таблица 1

k	$a^{(k)}$	$b^{(k)}$	$f(a^{(k)})$	$f(b^{(k)})$	$\frac{a^{(k)} + b^{(k)}}{2}$	$f\left(\frac{a^{(k)} + b^{(k)}}{2}\right)$	$\varepsilon^{(k)}$
0	0.4000	0.6000	–0.5745	1.1201	0.5000	0.2183	0.1000
1	0.4000	0.5000	–0.5745	0.2183	0.4500	–0.1904	0.0500
2	0.4500	0.5000	–0.1904	0.2183	0.4750	0.0107	0.0250
3	0.4500	0.4750	–0.1904	0.0107	0.4625	–0.0906	0.0125
4	0.4625	0.4750	–0.0906	0.0107	0.4688	–0.0402	0.0062
5	0.4688	0.4750	–0.0402	0.0107	0.4719	–0.0148	0.0031
6	0.4719	0.4750	–0.0148	0.0107	0.4734	–0.0024	0.0016
7	0.4734	0.4750	–0.0020	0.0107	**0.4742**	0.0042	**0.0008**

3.3. ПРОГРАММА #06

Ниже предлагается вариант алгоритма программы для уточнения значения корня уравнения методом половинного деления.

```
АЛГОРИТМ «Метод половинного деления»
ВХОД        f(), a, b, e
ВЫХОД       x, k
НАЧАЛО
    ЕСЛИ (f(a)*f(b)≥0) КОНЕЦ
    x:=(a+b)/2
    k:=0
    ЦИКЛ «Итерация» ПОКА ((b-a)/2>e)
            ЕСЛИ (f(x)=0) ПРЕРВАТЬ
            ЕСЛИ (f(a)*f(x)<0) b:=x
            ЕСЛИ (f(x)*f(b)<0) a:=x
            x:=(a+b)/2
            k:=k+1
    ПЕЧАТЬ x, k
КОНЕЦ
```

3.4. МЕТОД ПРОСТОЙ ИТЕРАЦИИ (НЕЛИНЕЙНЫЕ УРАВНЕНИЯ)

При использовании метода простой итерации уравнение $f(x) = 0$ заменяется эквивалентным уравнением с выделенным линейным членом:

$$x = \varphi(x) .$$

Решение ищется путём построения последовательности по формуле:

$$x^{(k+1)} = \varphi(x^{(k)}),\ k = 0, 1, 2, \ldots ,$$

начиная с некоторого заданного значения $x^{(0)}$. Если $\varphi(x)$ — непрерывная функция, а $x^{(k)}$, $k = 0, 1, 2, \ldots$ — сходящаяся последовательность, то значение $x^* = \lim\limits_{k \to \infty} x^{(k)}$ является решением уравнения $f(x) = 0$.

Условия сходимости метода и оценка его погрешности определяются теоремой. Пусть функция $\varphi(x)$ определена и дифференцируема на отрезке $[a,b]$. Тогда если выполняются условия:

$$\varphi(x) \in [a,b],\ \forall\, x \in [a,b]\,,$$

$$\exists\, q : \left|\varphi'(x)\right| \le q < 1,\ \forall\, x \in (a,b)\,,$$

т.е. $\varphi(x)$ — сжимающее отображение, то уравнение $f(x) = 0$ имеет, и притом единственный на $[a,b]$, корень x^*, к которому сходится определяемая методом простой итерации последовательность $x^{(k)}$, $k = 0,1,2,\ldots$, начинающаяся с любого $x^{(0)} \in [a,b]$, причём сходимость линейная. Обычно берут $x^{(0)} = \dfrac{a+b}{2}$.

Действительно, поскольку $x^{(k)} = \varphi(x^{(k-1)})$ и для корня верно $x^* = \varphi(x^*)$, то согласно теореме Лагранжа о конечном приращении для дифференцируемой на $[a,b]$ функции $\varphi(x)$:

$$x^* - x^{(k)} = \varphi(x^*) - \varphi(x^{(k-1)}) = \varphi'(\xi)\cdot(x^* - x^{(k-1)}),\ \xi \in [x^*, x^{(k-1)}]\,,$$

$$\left|x^* - x^{(k)}\right| \le \left|\max_{\xi\in[a,b]} \varphi'(\xi)\right| \cdot \left|x^* - x^{(k-1)}\right|.$$

Положив $q = \left|\max\limits_{\xi\in[a,b]} \varphi'(\xi)\right|$, получим:

$$\left|x^* - x^{(k)}\right| \le q \cdot \left|x^* - x^{(k-1)}\right| \le q^2 \cdot \left|x^* - x^{(k-2)}\right| \le \ldots \le q^k \cdot \left|x^* - x^{(0)}\right|,$$

что означает, что $\lim\limits_{k\to\infty} \left|x^* - x^{(k)}\right| = 0$ только если $q < 1$ (достаточное условие). Если же $\left|\varphi'(x)\right| > 1$, то итерации могут не сходиться.

Четыре случая взаимного расположения линий $y = x$ и $y = \varphi(x)$ вблизи корня уравнения $x = \varphi(x)$ и соответствующие им итерационные процессы показаны на рис. 3.

Рис. 3.1,*а* и 3.1,*б* соответствуют случаю $\left|\varphi'(x)\right| < 1$ — процесс итераций сходится. При этом в первом случае $\varphi'(x) > 0$ и сходимость носит односторонний характер, а во втором $\varphi'(x) < 0$ и сходимость носит двусторонний характер. Рис. 3.1,*в* и 3.1,*г* соответствуют случаю $\left|\varphi'(x)\right| > 1$ — процесс итераций расходится, при этом имеет место односторонняя и двусторонняя расходимость соответственно.

Для метода простой итерации справедливы оценки погрешности:

$$\left|x^* - x^{(k)}\right| \le \frac{q}{1-q}\left|x^{(k)} - x^{(k-1)}\right| \ \text{ и } \ \left|x^* - x^{(k)}\right| \le \frac{q^k}{1-q}\left|x^{(1)} - x^{(0)}\right|.$$

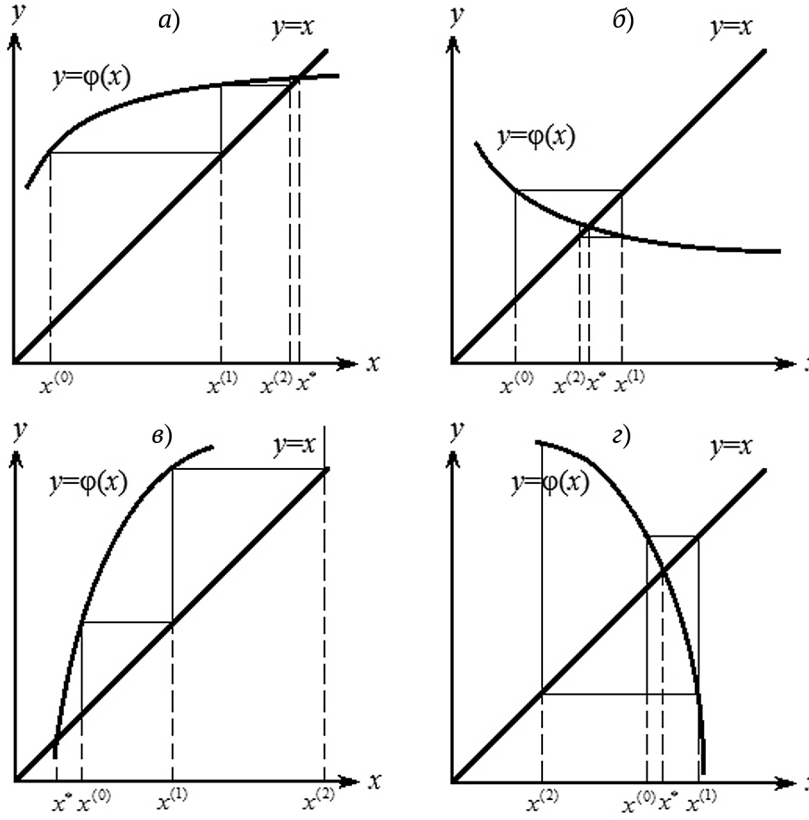

Рис. 3. Геометрическая интерпретация метода простой итерации

Действительно, согласно теореме Лагранжа:

$$x^* - x^{(k)} = \varphi(x^*) + \varphi(x^{(k)}) - \varphi(x^{(k)}) - \varphi(x^{(k-1)}) =$$

$$= \varphi'(\xi_1) \cdot (x^* - x^{(k)}) + \varphi'(\xi_2) \cdot (x^{(k)} - x^{(k-1)}),$$

$$\left|x^* - x^{(k)}\right| \le q \cdot \left|x^* - x^{(k)}\right| + q \cdot \left|x^{(k)} - x^{(k-1)}\right|, \text{ т.е.}$$

$$\left|x^* - x^{(k)}\right| \le \frac{q}{1-q} \cdot \left|x^{(k)} - x^{(k-1)}\right|.$$

Таким образом, в качестве условия окончания итераций в практических вычислениях используется правило $\varepsilon^{(k)} = \dfrac{q}{1-q} \cdot \left| x^{(k)} - x^{(k-1)} \right|$. При этом ясно, что скорость сходимости метода простой итерации зависит от величины q: чем меньше q, тем быстрее сходится метод. Поскольку исходное уравнение $f(x) = 0$ может быть преобразовано к виду $x = \varphi(x)$ многими способами, то, очевидно, для метода простой итерации целесообразно брать то уравнение $x = \varphi(x)$, для которого q имеет наименьшее значение.

Если непосредственное преобразование уравнения $f(x) = 0$ к виду $x = \varphi(x)$ не позволяет получить уравнение, для которого выполняются условия сходимости метода простой итерации, то можно преобразовать уравнение $f(x) = 0$ к следующему эквивалентному уравнению:

$$x = x - \lambda f(x).$$

Для данного уравнения $\varphi(x) = x - \lambda f(x)$. Здесь $\lambda > 0$ — параметр, который подбирается таким образом, чтобы в нужной области выполнялось неравенство $\left|\varphi'(x)\right| = \left|1 - \lambda f'(x)\right| \le q < 1$, т.е. $\lambda < \dfrac{sign\left(f'(x)\right)}{\max\left|f'(x)\right|}$.

В общем случае можно вообще положить $\varphi(x) = x + \psi(x) \cdot f(x)$, где $\psi(x)$ — непрерывная произвольная знакопостоянная функция.

Пример. Осуществить уточнение значения корня уравнения $f(x) = e^{2x} + 3x - 4 = 0$ в отрезке [0.4, 0.6] с точностью $\varepsilon = 10^{-3}$ методом простой итерации.

Уравнение $f(x) = e^{2x} + 3x - 4 = 0$ можно перезаписать в виде:

$$x = \frac{4 - e^{2x}}{3} \quad \text{или} \quad x = \frac{\ln(4 - 3x)}{2}.$$

Из двух этих вариантов приемлемым является второй вариант, так как положив $\varphi(x) = \dfrac{\ln(4-3x)}{2}$, будем иметь:

$$\varphi(x) \in [0.4,\ 0.55],\ \forall x \in [0.4,\ 0.55];$$

$$\varphi'(x) = -\frac{3}{2(4-3x)},$$

в отрезке $[0.4,\ 0.55]$ $\left|\varphi'(x)\right| < \left|-\dfrac{3}{2(4-3\cdot 0.55)}\right| \approx 0.64 = q$.

Условия сходимости выполнены. Причём для их выполнения исходный отрезок [0.4,0.6] был сокращён до [0.4,0.55]. Это допустимый приём в подобных задачах, если отбрасываемый промежуток не содержит искомого корня (что очевидно, т.к. $f(0.55) > 0$ или $f_1(0.55) > f_2(0.55)$).

В качестве начального приближения положим

$$x^{(0)} = (0.4 + 0.55)/2 = 0.475 .$$

Последовательные приближения $x^{(k)}$ вычисляются по формуле

$$x^{(k+1)} = \varphi(x^{(k)}),\ k = 0,1,2,\ldots,\ \text{где}\ \varphi(x^{(k)}) = \frac{\ln(4 - 3x^{(k)})}{2} .$$

Итерации завершаются при выполнении условия

$$\varepsilon^{(k+1)} = \frac{q}{1-q} \cdot \left| x^{(k+1)} - x^{(k)} \right| \le \varepsilon .$$

Результаты вычислений приведены в табл. 2.

Итоговое приближение корня $x^* \approx x^{(4)} = 0.4738$.

Таблица 2

k	$x^{(k)}$	$\varphi(x^{(k)})$	$\varepsilon^{(k)}$
0	0.4750	0.4729	–
1	0.4729	0.4741	0.0037
2	0.4741	0.4734	0.0021
3	0.4734	0.4738	0.0012
4	**0.4738**		**0.0007**

3.5. ПРОГРАММА #07

Ниже предлагается вариант алгоритма программы для уточнения значения корня уравнения методом простой итерации.

```
АЛГОРИТМ «Метод простой итерации»
ВХОД      f(), q, a, b, e
ВЫХОД     r, y, x, k
НАЧАЛО
    ЕСЛИ (q>=1) КОНЕЦ
```

```
    x:=(a+b)/2
    r:=2*e
    k:=0
    ЦИКЛ «Итерация» ПОКА (r>e)
            y:=x
            x:=f(y)
            ЕСЛИ (x<a)ИЛИ(x>b) КОНЕЦ
            ЕСЛИ (x>y) r:=x-y ИНАЧЕ r:=y-x
            r:=r*q/(1-q)
            k:=k+1
    ПЕЧАТЬ x, k
КОНЕЦ
```

3.6. МЕТОД НЬЮТОНА (КАСАТЕЛЬНЫХ)

При нахождении корня уравнения $f(x) = 0$ методом Ньютона итерационный процесс определяется формулой:

$$x^{(k+1)} = x^{(k)} - \frac{f(x^{(k)})}{f'(x^{(k)})}, \quad k = 0, 1, 2, \ldots$$

Действительно, разложение функции $f(x)$ по формуле Тейлора в точке $x^{(k+1)}$ примет вид:

$$f(x^{(k+1)}) = f(x^{(k)}) + f'(x^{(k)})(x^{(k+1)} - x^{(k)}) + O((x^{(k+1)} - x^{(k)})^2).$$

Исходя из требования $f(x^{(k+1)}) = 0$ при $k \to \infty$ и отбрасывая слагаемые порядка выше первого, получаем:

$$f(x^{(k)}) + f'(x^{(k)})(x^{(k+1)} - x^{(k)}) = 0, \; x^{(k+1)} = x^{(k)} - \frac{f(x^{(k)})}{f'(x^{(k)})}.$$

Так как была использована лишь линейная часть разложения в ряд, то для $x^{(k+1)}$ вместо корня x^* получим лишь его приближённое уточнённое значение, после чего процесс необходимо повторить, что позволяет построить последовательность приближений.

Геометрически процесс (рис. 4) означает замену на каждой итерации кривой $y = f(x)$ касательной к ней в точке $(x^{(k)}, f(x^{(k)}))$ и определение значения $x^{(k+1)}$ как координаты точки пересечения

касательной и оси абсцисс; с этой интерпретацией связано второе название метода — метод касательных.

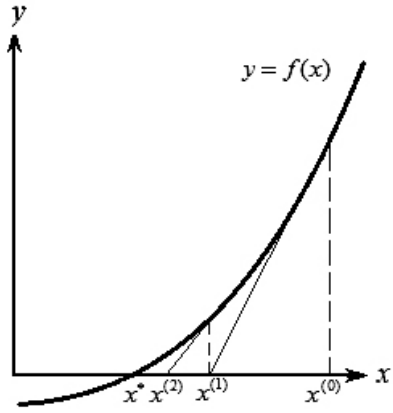

Рис. 4. Геометрическая интерпретация метода Ньютона

Достаточное условие сходимости метода Ньютона получается из соответствующего условия для метода простой итерации: можно видеть, что метод Ньютона представляет собой специальный случай метода простой итерации, в котором $\varphi(x) = x - f(x) / f'(x)$.

Используя условие сходимости метода простой итерации $|\varphi'(x)| < 1$, получаем:

$$\varphi'(x) = 1 - \frac{\left(f'(x)\right)^2 - f(x)f''(x)}{\left(f'(x)\right)^2} = \frac{-f(x)f''(x)}{\left(f'(x)\right)^2},$$

$$\left|\frac{-f(x)f''(x)}{\left(f'(x)\right)^2}\right| < 1, \ \left|f(x)f''(x)\right| < \left(f'(x)\right)^2.$$

Для начала процесса вычислений требуется задание начального приближения $x^{(0)}$. Условия выбора начального приближения $x^{(0)}$ определяются следующей теоремой. Пусть на отрезке $[a,b]$ функция $f(x)$ имеет первую и вторую производные постоянного знака и пусть $f(a) \cdot f(b) < 0$. Тогда если точка $x^{(0)}$ выбрана на $[a,b]$ так, что $f(x^{(0)}) \cdot f''(x^{(0)}) > 0$, то начатая с неё последовательность $x^{(k)}$, опре-

деляемая методом Ньютона, монотонно сходится к корню $x^* \in [a, b]$ уравнения $f(x) = 0$. Обычно берут $x^{(0)} = a$ либо $x^{(0)} = b$ — в зависимости от того, в какой точке выполнено $f(x^{(0)}) \cdot f''(x^{(0)}) > 0$.

В качестве условия окончания итераций в практических вычислениях используется правило $\varepsilon^{(k)} = \left| x^{(k)} - x^{(k-1)} \right|$.

Метод Ньютона имеет вблизи корня второй порядок сходимости: на каждой итерации ошибка меняется пропорционально квадрату ошибки на предыдущей итерации. Очевидно, метод Ньютона является одношаговым.

Пример. Осуществить уточнение значения корня уравнения $f(x) = e^{2x} + 3x - 4 = 0$ в отрезке [0.4, 0.6] с точностью $\varepsilon = 10^{-3}$ методом Ньютона.

Для корректного использования данного метода необходимо сперва определить поведение первой и второй производной функции $f(x)$ на промежутке уточнения корня и правильно выбрать начальное приближение $x^{(0)}$. Для функции $f(x) = e^{2x} + 3x - 4 = 0$ имеем:

$f'(x) = 2e^{2x} + 3$, $f''(x) = 4e^{2x}$ — положительные во всей области определения функции, удовлетворяющие условию сходимости $\left| f(x) f''(x) \right| < \left(f'(x) \right)^2$, условие $f(0.4) \cdot f(0.6) < 0$ также выполнено.

В качестве начального приближения можно выбрать правую границу отрезка $[a, b]$ $x^{(0)} = 0.6$, для которой выполняется неравенство $f(0.6) \cdot f''(0.6) > 0$.

Дальнейшие вычисления проводятся по формуле

$$x^{(k+1)} = x^{(k)} - \frac{f(x^{(k)})}{f'(x^{(k)})},$$

где $f(x^{(k)}) = e^{2x^{(k)}} + 3x^{(k)} - 4$, $f'(x^{(k)}) = 2e^{2x^{(k)}} + 3$.

Итерации завершаются при выполнении условия

$$\varepsilon^{(k+1)} = \left| x^{(k+1)} - x^{(k)} \right| < \varepsilon.$$

Результаты вычислений приведены в табл. 3.

Итоговое приближение корня $x^* \approx x^{(3)} = 0.4737$.

Таблица 3

k	$x^{(k)}$	$f(x^{(k)})$	$f'(x^{(k)})$	$-f(x^{(k)}) / f'(x^{(k)})$	$\varepsilon^{(k)}$
0	0.6000	1.1201	9.6402	–0.1162	–
1	0.4838	0.0831	8.2633	–0.0101	0.1162
2	0.4738	0.0005	8.1585	–0.0001	0.0101
3	**0.4737**				**0.0001**

3.7. ПРОГРАММА #08

Ниже предлагается вариант алгоритма программы для уточнения значения корня уравнения методом Ньютона.

```
АЛГОРИТМ «Метод Ньютона»
ВХОД        f(), g(), h(), a, b, e
ВЫХОД       r, y, x, k
НАЧАЛО
    ЕСЛИ (f(a)*h(a)>0) x:=a ИНАЧЕ ЕСЛИ (f(b)*h(b)>0) x:=b
    ИНАЧЕ КОНЕЦ
    r:=2*e
    k:=0
    ЦИКЛ «Итерация» ПОКА (r>e)
            y:=x
            x:=y-f(y)/g(y)
            ЕСЛИ (x>y) r:=x-y ИНАЧЕ r:=y-x
            k:=k+1
    ПЕЧАТЬ x, k
КОНЕЦ
```

3.8. МЕТОД СЕКУЩИХ (ХОРД)

Использование метода Ньютона предполагает вычисление на каждой итерации значения функции и её производной. Иногда, например если функция $f(x)$ задана таблично, непосредственное вычисление производной затруднительно. Тогда, заменяя производную функции приближённым разностным отношением $f'(x^{(k)}) \approx \dfrac{f(x^{(k)}) - f(x^{(k-1)})}{x^{(k)} - x^{(k-1)}}$

и подставляя его в формулу метода Ньютона, получаем итерационную формулу метода секущих:

$$x^{(k+1)} = x^{(k)} - \frac{f(x^{(k)})(x^{(k)} - x^{(k-1)})}{f(x^{(k)}) - f(x^{(k-1)})},\ k = 1, 2, \ldots$$

Геометрически процесс (рис. 5) означает замену на каждой итерации касательной кривой $y = f(x)$ в точке $[x^{(k)},\ f(x^{(k)})]$ на секущую, проведённую через две точки $[x^{(k)},\ f(x^{(k)})]$, $[x^{(k-1)},\ f(x^{(k-1)})]$, с чем связано название метода.

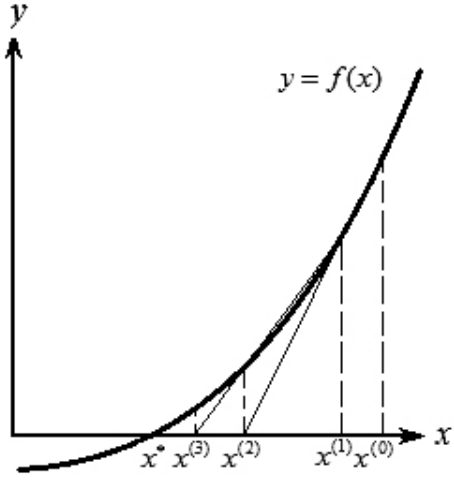

Рис. 5. Геометрическая интерпретация метода секущих

Использование этого метода избавляет от необходимости расчёта производной функции в процессе вычислений за счёт некоторой потери точности.

Метод является двухшаговым: как видно из его формулы, результат после $(k+1)$-го шага зависит от результатов k-го и $(k-1)$-го шагов. Соответственно, для выполнения первой итерации требуется задание двух начальных точек $x^{(0)}$ и $x^{(1)}$.

Условия сходимости метода секущих аналогичны условиям сходимости метода Ньютона. Выбор начальной точки $x^{(0)}$ осуществляется

по тому же принципу, что и в методе Ньютона. Вторая начальная точка $x^{(1)}$ выбирается в непосредственной близости от $x^{(0)}$, желательно между точкой $x^{(0)}$ и искомым корнем x^*. В качестве условия окончания итераций в практических вычислениях используется правило $\varepsilon^{(k)} = \left|x^{(k)} - x^{(k-1)}\right|$.

Порядок сходимости метода секущих равен $\frac{\sqrt{5}}{2} \approx 1,62$.

Пример. Осуществить уточнение значения корня уравнения $f(x) = e^{2x} + 3x - 4 = 0$ в отрезке [0.4, 0.6] с точностью $\varepsilon = 10^{-3}$ методом секущих.

В качестве начальных точек зададим $x^{(0)} = 0.6$ и $x^{(1)} = 0.55$.

Дальнейшие вычисления проводятся по формуле

$$x^{(k+1)} = x^{(k)} - \frac{f(x^{(k)})(x^{(k)} - x^{(k-1)})}{f(x^{(k)}) - f(x^{(k-1)})}, \text{ где } f(x^{(k)}) = e^{2x^{(k)}} + 3x^{(k)} - 4.$$

Итерации завершаются при выполнении условия

$$\varepsilon^{(k+1)} = \left|x^{(k+1)} - x^{(k)}\right| < \varepsilon.$$

Результаты вычислений приведены в табл. 4.

Итоговое приближение корня $x^* \approx x^{(4)} = 0.4737$. Можно видеть, что метод секущих в данном случае сходится несколько медленнее метода Ньютона.

Таблица 4

k	$x^{(k)}$	$f(x^{(k)})$	$\varepsilon^{(k)}$
0	0.6000	1.1201	–
1	0.5500	0.6542	–
2	0.4798	0.0501	0.0702
3	0.4740	0.0024	0.0058
4	**0.4737**		**0.0003**

3.9. ПРОГРАММА #09

Ниже предлагается вариант алгоритма программы для уточнения значения корня уравнения методом секущих.

```
АЛГОРИТМ «Метод секущих»
ВХОД        f(), c, d, e
ВЫХОД       r, z, y, x, k
НАЧАЛО
    y:=c
    x:=d
    r:=2*e
    k:=1
    ЦИКЛ «Итерация» ПОКА (r>e)
            z:=y
            y:=x
            x:= y-f(y)*(y-z)/(f(y)-f(z))
            ЕСЛИ (x>y) r:=x-y ИНАЧЕ r:=y-x
            k:=k+1
    ПЕЧАТЬ x, k
КОНЕЦ
```

3.10. РЕШЕНИЕ СИСТЕМ НЕЛИНЕЙНЫХ УРАВНЕНИЙ

Пусть требуется найти решение системы из n нелинейных уравнений с n неизвестными, которую можно записать в следующем виде:

$$\begin{cases} f_1(x_1, x_2, ..., x_n) = 0 \\ f_2(x_1, x_2, ..., x_n) = 0 \\ ..... \\ f_n(x_1, x_2, ..., x_n) = 0 \end{cases},$$

или, более коротко, в векторной форме:

$$\mathbf{f}(\mathbf{x}) = \mathbf{0},$$

где $\mathbf{x}$ — вектор неизвестных величин, $\mathbf{f}$ — вектор-функция:

$$\mathbf{x} = \begin{pmatrix} x_1 \\ x_2 \\ ... \\ x_n \end{pmatrix}, \ \mathbf{f} = \begin{pmatrix} f_1(\mathbf{x}) \\ f_2(\mathbf{x}) \\ ... \\ f_n(\mathbf{x}) \end{pmatrix}, \ \mathbf{0} = \begin{pmatrix} 0 \\ 0 \\ ... \\ 0 \end{pmatrix}.$$

Численное решение этой системы заключается в нахождении значений $x_1, x_2, \ldots, x_n$, удовлетворяющих с заданной точностью ε каждому из n заданных уравнений.

В редких случаях для решения такой системы удаётся применить метод последовательного исключения неизвестных и свести решение исходной задачи к решению одного нелинейного уравнения с одним неизвестным. Значения других неизвестных величин тогда находятся соответствующей подстановкой в конкретные выражения. Однако в подавляющем большинстве случаев для решения систем нелинейных уравнений используются итерационные методы, заключающиеся в построении последовательности векторов $\mathbf{x}^{(k)}$, $k = 0, 1, 2, \ldots$, сходящейся в пределе $k \to \infty$ к искомому корню $\mathbf{x}^* = (x_1^*, x_2^*, \ldots, x_n^*)^T$ исходной системы.

Как и в случае одного нелинейного уравнения, предполагается, что сперва ищется некоторое изолированное решение нелинейной системы, которое затем уточняется с заданной точностью ε в некоторой выпуклой области $G \in R^n$. Локализация решения может осуществляться на основе специфической информации по конкретной решаемой задаче (например, по физическим соображениям) или с помощью методов математического анализа. При решении системы из двух уравнений достаточно часто удобным является графический способ, когда месторасположение корней определяется как точки пересечения кривых $f_1(x_1, x_2) = 0$ и $f_2(x_1, x_2) = 0$ на плоскости (x_1, x_2). Однако для систем высоких порядков удовлетворительных методов отделения корней не существует.

Пример. Локализовать положительное решение системы нелинейных уравнений:

$$\begin{cases} f_1(x_1, x_2) = 0.1x_1^2 + x_1 + 0.2x_2^2 - 0.3 = 0 \\ f_2(x_1, x_2) = 0.2x_1^2 + x_2 - 0.1x_1x_2 - 0.7 = 0 \end{cases}.$$

Для определения выпуклой области $G \in R^2$, содержащей искомое решение, применяем графический способ.

На рис. 6 можно видеть графики кривых функций $f_1(x_1, x_2) = 0$ и $f_2(x_1, x_2) = 0$.

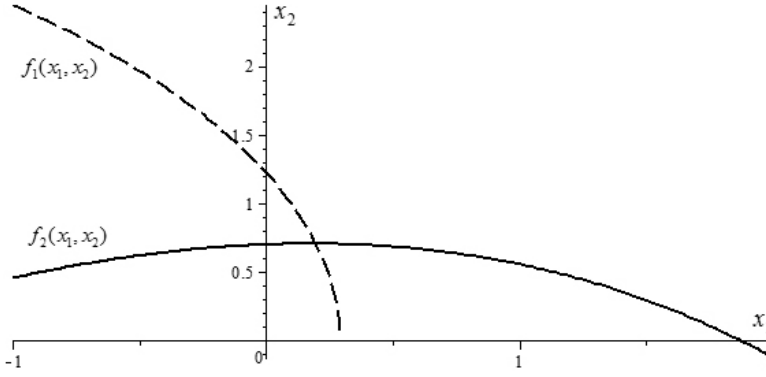

Рис. 6. Графический способ локализации корней системы двух уравнений

Построив на плоскости (x_1, x_2) в интересующей нас области кривые $f_1(x_1, x_2) = 0$ и $f_2(x_1, x_2) = 0$, определяем, что положительное решение системы уравнений находится, например, в выпуклой области, представляющей собой прямоугольник $G = \{0 < x_1 < 0.5;\ 0.5 < x_2 < 1\}$.

3.11. МЕТОД ПРОСТОЙ ИТЕРАЦИИ (НЕЛИНЕЙНЫЕ СИСТЕМЫ)

При использовании обобщённого метода простой итерации система уравнений $\mathbf{f}(\mathbf{x}) = \mathbf{0}$ приводится к эквивалентной системе специального вида:

$$\begin{cases} x_1 = \varphi_1(x_1, x_2, ..., x_n) \\ x_2 = \varphi_2(x_1, x_2, ..., x_n) \\ ..... \\ x_n = \varphi_n(x_1, x_2, ..., x_n) \end{cases},$$

или, в векторной форме:

$$\mathbf{x} = \varphi(\mathbf{x}),\ \varphi(\mathbf{x}) = \begin{pmatrix} \varphi_1(\mathbf{x}) \\ \varphi_2(\mathbf{x}) \\ ... \\ \varphi_n(\mathbf{x}) \end{pmatrix},$$

где функции $\varphi_1(\mathbf{x})$, $\varphi_2(\mathbf{x})$, ..., $\varphi_n(\mathbf{x})$ — определены и непрерывны в некоторой окрестности $G \in R^n$ искомого изолированного решения $\mathbf{x}^* = (x_1^*, x_2^*, ..., x_n^*)^T$.

Если выбрано некоторое начальное приближение $\mathbf{x}^{(0)} = (x_1^{(0)}, x_2^{(0)}, ..., x_n^{(0)})^T$ из $G \in R^n$, то последующие приближения $\mathbf{x}^{(k)} = (x_1^{(k)}, x_2^{(k)}, ..., x_n^{(k)})^T$ в методе простой итерации находятся по формулам:

$$\begin{cases} x_1^{(k+1)} = \varphi_1(x_1^{(k)}, x_2^{(k)}, ..., x_n^{(k)}) \\ x_2^{(k+1)} = \varphi_2(x_1^{(k)}, x_2^{(k)}, ..., x_n^{(k)}) \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ x_n^{(k+1)} = \varphi_n(x_1^{(k)}, x_2^{(k)}, ..., x_n^{(k)}) \end{cases}, \quad k = 0, 1, 2, ...,$$

или, в векторной форме:

$$\mathbf{x}^{(k+1)} = \varphi(\mathbf{x}^{(k)}), \quad k = 0, 1, 2, ...$$

Если эта последовательность сходится, то она сходится к искомому решению $\mathbf{x}^* = (x_1^*, x_2^*, ..., x_n^*)^T$, причём сходимость линейная.

Достаточное условие сходимости итерационного процесса метода простой итерации формулируется следующим образом. Пусть вектор-функция $\varphi(\mathbf{x})$ непрерывна вместе со своей производной (матрицей Якоби первых производных системы функций)

$$\varphi'(\mathbf{x}) = \begin{bmatrix} \dfrac{\partial \varphi_1(\mathbf{x})}{\partial x_1} & \dfrac{\partial \varphi_1(\mathbf{x})}{\partial x_2} & \cdots & \dfrac{\partial \varphi_1(\mathbf{x})}{\partial x_n} \\ \dfrac{\partial \varphi_2(\mathbf{x})}{\partial x_1} & \dfrac{\partial \varphi_2(\mathbf{x})}{\partial x_2} & \cdots & \dfrac{\partial \varphi_2(\mathbf{x})}{\partial x_n} \\ \cdots & \cdots & \cdots & \cdots \\ \dfrac{\partial \varphi_n(\mathbf{x})}{\partial x_1} & \dfrac{\partial \varphi_n(\mathbf{x})}{\partial x_2} & \cdots & \dfrac{\partial \varphi_n(\mathbf{x})}{\partial x_n} \end{bmatrix}$$

в ограниченной замкнутой выпуклой области $G \in R^n$ и $\max\limits_{\mathbf{x} \in G} \|\varphi'(\mathbf{x})\| \le q < 1$, где q — некоторая постоянная. Тогда если $\mathbf{x}^{(0)} \in G$ и все последовательные приближения векторов $\mathbf{x}^{(k)} = (x_1^{(k)}, x_2^{(k)}, ..., x_n^{(k)})^T$ также содержатся в $G \in R^n$, то итерационный процесс $\mathbf{x}^{(k+1)} = \varphi(\mathbf{x}^{(k)})$, $k = 0, 1, 2, ...$ сходится к единственному решению $\mathbf{x}^* = (x_1^*, x_2^*, ..., x_n^*)^T$ уравнения $\mathbf{x} = \varphi(\mathbf{x})$ в области $G \in R^n$.

В этом случае также справедливы оценки погрешности $(\forall k \in N)$:

$$\left\|\mathbf{x}^{(*)} - \mathbf{x}^{(k)}\right\| \le \frac{q}{1-q}\left\|\mathbf{x}^{(k)} - \mathbf{x}^{(k-1)}\right\|, \quad \left\|\mathbf{x}^* - \mathbf{x}^{(k)}\right\| \le \frac{q^k}{1-q}\left\|\mathbf{x}^{(1)} - \mathbf{x}^{(0)}\right\|.$$

Таким образом, в качестве условия окончания итераций в практических вычислениях используется правило $\varepsilon^{(k)} = \dfrac{q}{1-q} \cdot \left\| \mathbf{x}^{(k)} - \mathbf{x}^{(k-1)} \right\|$.

Следует отметить, что в случае, когда при анализе сходимости конкретной итерационной схемы непосредственная проверка условия $\max\limits_{\mathbf{x} \in G} \left\| \varphi'(\mathbf{x}) \right\| \le q < 1$ является затруднительной, можно вместо этого определить норму «мажорирующей» матрицы $\mathbf{M}(\mathbf{x})$ с элементами $m_{ij}(\mathbf{x}) = \max\limits_{\mathbf{x} \in G} \left| \dfrac{\partial \varphi_i(\mathbf{x})}{\partial x_j} \right|$, так как $\max\limits_{\mathbf{x} \in G} \left\| \varphi'(\mathbf{x}) \right\| \le \left\| \mathbf{M}(\mathbf{x}) \right\|$. Тогда если $\left\| \mathbf{M}(\mathbf{x}) \right\| \le q < 1$, то последовательные приближения $\mathbf{x}^{(k)}$ сходятся к искомому решению $\mathbf{x}^*$.

Как и ранее, метод простой итерации может сходиться и при нарушении условия $q < 1$.

Сходимость метода простой итерации вновь можно ускорить методом Зейделя, который заключается в том, что при вычислении компонента $x_i^{(k)}$ вектора неизвестных на k-й итерации уже вычисленные значения $x_1^{(k)}, x_2^{(k)}, \ldots, x_{i-1}^{(k)}$ используются на той же k-й итерации. Значения для остальных компонент $x_i^{(k-1)}, x_{i+1}^{(k-1)}, \ldots, x_n^{(k-1)}$ берутся из предыдущей итерации. В этом случае итерационный процесс имеет вид:

$$\begin{cases} x_1^{(k+1)} = \varphi_1(x_1^{(k)}, x_2^{(k)}, x_3^{(k)}, \ldots, x_n^{(k)}) \\ x_2^{(k+1)} = \varphi_2(x_1^{(k+1)}, x_2^{(k)}, x_3^{(k)}, \ldots, x_n^{(k)}) \\ x_3^{(k+1)} = \varphi_3(x_1^{(k+1)}, x_2^{(k+1)}, x_3^{(k)}, \ldots, x_n^{(k)}) \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ x_n^{(k+1)} = \varphi_n(x_1^{(k+1)}, x_2^{(k+1)}, x_3^{(k+1)}, \ldots, x_{n-1}^{(k+1)}, x_n^{(k)}) \end{cases}, \quad k = 0, 1, 2, \ldots$$

Сходимость этого процесса тоже линейная. Так же, как и при решении систем линейных уравнений, при использовании здесь метода Зейделя может быть поставлена задача об отыскании на каждой итерации оптимальной последовательности уточнения компонент вектора неизвестных $\mathbf{x}^{(k)}$.

Пример. Найти положительное решение системы нелинейных уравнений с точностью $\varepsilon = 10^{-4}$ методом простой итерации:

$$\begin{cases} f_1(x_1, x_2) = 0.1x_1^2 + x_1 + 0.2x_2^2 - 0.3 = 0 \\ f_2(x_1, x_2) = 0.2x_1^2 + x_2 - 0.1x_1x_2 - 0.7 = 0 \end{cases}.$$

Преобразуем исходную систему уравнений к следующему виду:

$$\begin{cases} x_1 = 0.3 - 0.1x_1^2 - 0.2x_2^2 = \varphi_1(x_1, x_2) \\ x_2 = 0.7 - 0.2x_1^2 + 0.1x_1x_2 = \varphi_2(x_1, x_2) \end{cases}.$$

Проверим выполнение условия $\max\limits_{\mathbf{x}\in G}\|\varphi'(\mathbf{x})\| \le q < 1$ в прямоугольной области $G = \{0 < x_1 < 0.5;\ 0.5 < x_2 < 1\}$ (т.е. $|x_1 - 0.25| \le 0.25$, $|x_2 - 0.75| \le 0.25$).

Для этого найдём $\max\limits_{\mathbf{x}\in G}\|\varphi'(\mathbf{x})\| = \max\limits_{\mathbf{x}\in G}\left\{\max\limits_{i=1..n}\sum\limits_{j=1}^{n}\left|\dfrac{\partial\varphi_i(x_1, x_2)}{\partial x_j}\right|\right\}$.

Определим частные производные $\varphi'(\mathbf{x})$:

$$\frac{\partial\varphi_1(x_1, x_2)}{\partial x_1} = -0.2x_1,\ \frac{\partial\varphi_1(x_1, x_2)}{\partial x_2} = -0.4x_2,$$

$$\frac{\partial\varphi_2(x_1, x_2)}{\partial x_1} = -0.4x_1 + 0.1x_2,\ \frac{\partial\varphi_2(x_1, x_2)}{\partial x_2} = 0.1x_1.$$

Очевидно, в области $G = \{0 < x_1 < 0.5;\ 0.5 < x_2 < 1\}$ имеем:

$$\left|\frac{\partial\varphi_1(x_1, x_2)}{\partial x_1}\right| + \left|\frac{\partial\varphi_1(x_1, x_2)}{\partial x_2}\right| = |-0.2x_1| + |-0.4x_2| \le 0.1 + 0.4 = 0.5,$$

$$\left|\frac{\partial\varphi_2(x_1, x_2)}{\partial x_1}\right| + \left|\frac{\partial\varphi_2(x_1, x_2)}{\partial x_2}\right| = |-0.4x_1 + 0.1x_2| + |0.1x_1| \le 0.15 + 0.05 = 0.2.$$

Таким образом, $\max\limits_{\mathbf{x}\in G}\|\varphi'(\mathbf{x})\| = \max(0.5, 0.2) = 0.5 = q < 1$.

Следовательно, если последовательные приближения $(x_1^{(k)}, x_2^{(k)})$ не покинут области $G = \{0 < x_1 < 0.5;\ 0.5 < x_2 < 1\}$ (что легко обнаружить в процессе вычислений), то итерационный процесс будет сходящимся.

В качестве начального приближения примем $x_1^{(0)} = 0.25$, $x_2^{(0)} = 0.75$.

Последующие приближения определяем как

$$\begin{cases} x_1^{(k+1)} = \varphi_1(x_1^{(k)}, x_2^{(k)}) \\ x_2^{(k+1)} = \varphi_2(x_1^{(k)}, x_2^{(k)}) \end{cases},\ k = 0, 1, 2, \ldots,$$

где

$$\varphi_1(x_1^{(k)}, x_2^{(k)}) = 0.3 - 0.1x_1^{(k)\,2} - 0.2x_2^{(k)\,2},$$

$$\varphi_2(x_1^{(k)}, x_2^{(k)}) = 0.7 - 0.2x_1^{(k)\,2} + 0.1x_1^{(k)}x_2^{(k)}.$$

Итерации завершаются при выполнении условия

$$\varepsilon^{(k+1)} = \frac{q}{1-q}\left\|\mathbf{x}^{(k+1)} - \mathbf{x}^{(k)}\right\| \le \varepsilon\,, \text{ где } \left\|\mathbf{x}^{(k+1)} - \mathbf{x}^{(k)}\right\| = \max_i \left|x_i^{(k+1)} - x_i^{(k)}\right|.$$

Результаты вычислений приведены в табл. 5.

Итоговое приближение корня $x_1^* \approx 0.1964$, $x_2^* \approx 0.7062$.

Таблица 5

k	$x_1^{(k)}$, $x_2^{(k)}$	$\varphi_1(x_1^{(k)}, x_2^{(k)})$, $\varphi_2(x_1^{(k)}, x_2^{(k)})$	$\varepsilon^{(k)}$
0	0.25000, 0.75000	0.18125, 0.70625	–
1	0.18125, 0.70625	0.19696, 0.70623	0.06875
2	0.19696, 0.70623	0.19639, 0.70615	0.01571
3	0.19637, 0.70615	0.19641, 0.70615	0.00059
4	**0.19641**, **0.70615**		**0.00005**

3.12. ПРОГРАММА #10

Ниже предлагается вариант алгоритма программы для уточнения значения корней системы из двух уравнений методом простой итерации или методом Зейделя.

```
АЛГОРИТМ «Метод простой итерации или метод Зейделя»
ВХОД        f(,), g(,), q, a, b, c, d, e
ВЫХОД       r, s, u, v, x, y, k
НАЧАЛО
    ЕСЛИ (q>=1) КОНЕЦ
    x:=(a+b)/2, y:=(c+d)/2
    r:=2*e
    k:=0
    ЦИКЛ «Итерация» ПОКА (r>e)
            u:=x, v:=y
            #Для метода простой итерации
            x:=f(u,v), y:=g(u,v)
            #Альтернативно для метода Зейделя
            x:=f(u,v), y:=g(x,v)
            ЕСЛИ (x<a)ИЛИ(x>b)ИЛИ(y<c)ИЛИ(y>d) КОНЕЦ
```

```
            ЕСЛИ (x>u) r:=x-u ИНАЧЕ r:=u-x
            ЕСЛИ (y>v) s:=y-v ИНАЧЕ s:=v-y
            ЕСЛИ (s>r) r:=s
            r:=r*q/(1-q)
            k:=k+1
      ПЕЧАТЬ x, y, k
КОНЕЦ
```

3.13. МЕТОД НЬЮТОНА (ОБОБЩЁННЫЙ)

Решение системы нелинейных уравнений $\mathbf{f}(\mathbf{x}) = \mathbf{0}$ обобщённым методом Ньютона сводится к решению последовательности линейных задач, дающих в пределе решение исходной задачи. Линейная задача получается путем выделения из нелинейных уравнений главной линейной части.

Если определено начальное приближение $\mathbf{x}^{(0)} = (x_1^{(0)}, x_2^{(0)}, ..., x_n^{(0)})^T \in G$, то итерационный процесс нахождения решения системы уравнений методом Ньютона можно представить в следующем виде:

$$\begin{cases} x_1^{(k+1)} = x_1^{(k)} + \Delta x_1^{(k)} \\ x_2^{(k+1)} = x_2^{(k)} + \Delta x_2^{(k)} \\ \quad \ldots\ldots\ldots\ldots\ldots\ldots \\ x_n^{(k+1)} = x_n^{(k)} + \Delta x_n^{(k)} \end{cases}, \quad k = 0, 1, 2, \ldots$$

Здесь значения приращений $\Delta x_1^{(k)}$, $\Delta x_2^{(k)}$,..., $\Delta x_n^{(k)}$ определяются из решения системы линейных алгебраических уравнений, все коэффициенты которой выражаются через известное предыдущее приближение $\mathbf{x}^{(k)} = (x_1^{(k)}, x_2^{(k)}, ..., x_n^{(k)})$:

$$\begin{cases} f_1(\mathbf{x}^{(k)}) + \dfrac{\partial f_1(\mathbf{x}^{(k)})}{\partial x_1}\Delta x_1^{(k)} + \dfrac{\partial f_1(\mathbf{x}^{(k)})}{\partial x_2}\Delta x_2^{(k)} + \ldots + \dfrac{\partial f_1(\mathbf{x}^{(k)})}{\partial x_n}\Delta x_n^{(k)} = 0 \\ f_2(\mathbf{x}^{(k)}) + \dfrac{\partial f_2(\mathbf{x}^{(k)})}{\partial x_1}\Delta x_1^{(k)} + \dfrac{\partial f_2(\mathbf{x}^{(k)})}{\partial x_2}\Delta x_2^{(k)} + \ldots + \dfrac{\partial f_2(\mathbf{x}^{(k)})}{\partial x_n}\Delta x_n^{(k)} = 0 \\ \qquad\qquad \ldots\ldots\ldots\ldots\ldots\ldots \\ f_n(\mathbf{x}^{(k)}) + \dfrac{\partial f_n(\mathbf{x}^{(k)})}{\partial x_1}\Delta x_1^{(k)} + \dfrac{\partial f_n(\mathbf{x}^{(k)})}{\partial x_2}\Delta x_2^{(k)} + \ldots + \dfrac{\partial f_n(\mathbf{x}^{(k)})}{\partial x_n}\Delta x_n^{(k)} = 0 \end{cases}.$$

Эта линейная система получается в результате разложения вектор-функции $\mathbf{f}(\mathbf{x})$ по многомерной формуле Тейлора в точке $x^{(k+1)}$ с отбрасыванием слагаемых порядка выше первого и исходя из требования $\mathbf{f}(\mathbf{x}^{(k+1)}) = \mathbf{0}$ при $k \to \infty$.

В векторно-матричной форме расчётные формулы имеют вид

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \Delta\mathbf{x}^{(k)}, \quad k = 0, 1, 2, \ldots,$$

где вектор приращений

$$\Delta\mathbf{x}^{(k)} = \begin{pmatrix} \Delta x_1^{(k)} \\ \Delta x_2^{(k)} \\ \ldots \\ \Delta x_n^{(k)} \end{pmatrix}$$

находится из решения уравнения

$$\mathbf{f}(\mathbf{x}^{(k)}) + \mathbf{J}(\mathbf{x}^{(k)})\Delta\mathbf{x}^{(k)} = \mathbf{0}.$$

Здесь

$$\mathbf{J}(\mathbf{x}) = \begin{bmatrix} \dfrac{\partial f_1(\mathbf{x})}{\partial x_1} & \dfrac{\partial f_1(\mathbf{x})}{\partial x_2} \ldots \dfrac{\partial f_1(\mathbf{x})}{\partial x_n} \\ \dfrac{\partial f_2(\mathbf{x})}{\partial x_1} & \dfrac{\partial f_2(\mathbf{x})}{\partial x_2} \ldots \dfrac{\partial f_2(\mathbf{x})}{\partial x_n} \\ \ldots\ldots\ldots\ldots\ldots\ldots \\ \dfrac{\partial f_n(\mathbf{x})}{\partial x_1} & \dfrac{\partial f_n(\mathbf{x})}{\partial x_2} \ldots \dfrac{\partial f_n(\mathbf{x})}{\partial x_n} \end{bmatrix}$$

есть матрица Якоби первых производных вектор-функции $\mathbf{f}(\mathbf{x})$.

Выражая вектор приращений $\Delta\mathbf{x}^{(k)}$ из этого уравнения, итерационный процесс нахождения решения в итоге можно записать в виде

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} - \mathbf{J}^{-1}(\mathbf{x}^{(k)})\mathbf{f}(\mathbf{x}^{(k)}), \quad k = 0, 1, 2, \ldots,$$

где $\mathbf{J}^{-1}(\mathbf{x})$ — матрица, обратная матрице Якоби. Эта формула есть обобщение формулы метода Ньютона для нелинейных уравнений $x^{(k+1)} = x^{(k)} - \dfrac{f(x^{(k)})}{f'(x^{(k)})}$ на случай систем нелинейных уравнений.

Однако при реализации алгоритма метода Ньютона в большинстве случаев предпочтительным является не вычисление обратной матрицы $\mathbf{J}^{-1}(\mathbf{x}^{(k)})$, а непосредственное нахождение из системы линейных

алгебраических уравнений значений приращений $\Delta x_1^{(k)}$, $\Delta x_2^{(k)}$,..., $\Delta x_n^{(k)}$ и вычисление через них нового приближения. Для решения таких СЛАУ можно привлекать самые разные численные методы, как прямые, так и итерационные, при необходимости — с учётом размерности *n* решаемой задачи и специфики возникающих матриц Якоби **J**(**x**) (например, симметрии, разреженности и т.п.).

Использование метода Ньютона для нелинейных систем предполагает дифференцируемость функций $f_1(\mathbf{x})$, $f_2(\mathbf{x})$, ..., $f_n(\mathbf{x})$ и невырожденность матрицы Якоби ($\det \mathbf{J}(\mathbf{x}^{(k)}) \neq 0$) на каждой итерации. Тогда в случае, если начальное приближение $\mathbf{x}^{(0)} \in G$ выбрано в достаточно малой окрестности искомого корня, итерации метода Ньютона сходятся к точному решению $\mathbf{x}^* = (x_1^*, x_2^*, ..., x_n^*)^T$, причём сходимость квадратичная.

В практических вычислениях в качестве условия окончания итераций обычно используется критерий $\varepsilon^{(k)} = \left\| \mathbf{x}^{(k)} - \mathbf{x}^{(k-1)} \right\| \leq \varepsilon$, где ε — заданная вычислителем точность.

Пример. Найти положительное решение системы нелинейных уравнений с точностью $\varepsilon = 10^{-4}$ методом Ньютона:

$$\begin{cases} f_1(x_1, x_2) = 0.1x_1^2 + x_1 + 0.2x_2^2 - 0.3 = 0 \\ f_2(x_1, x_2) = 0.2x_1^2 + x_2 - 0.1x_1x_2 - 0.7 = 0 \end{cases}.$$

Решение ищем в области $G = \{0 < x_1 < 0.5;\ 0.5 < x_2 < 1\}$.

За начальное приближение примем $x_1^{(0)} = 0.25$, $x_2^{(0)} = 0.75$.

Для системы из двух уравнений расчётные формулы метода Ньютона удобно записать в виде, разрешённом относительно $x_1^{(k+1)}$, $x_2^{(k+1)}$ согласно правилу Крамера для решения СЛАУ:

$$\begin{cases} x_1^{(k+1)} = x_1^{(k)} - \dfrac{\det \mathbf{A}_1^{(k)}}{\det \mathbf{J}^{(k)}} \\ x_2^{(k+1)} = x_2^{(k)} - \dfrac{\det \mathbf{A}_2^{(k)}}{\det \mathbf{J}^{(k)}} \end{cases}, \quad k = 0, 1, 2, \ldots,$$

где матрица Якоби

$$\mathbf{J}^{(k)} = \begin{bmatrix} \dfrac{\partial f_1(x_1^{(k)}, x_2^{(k)})}{\partial x_1} & \dfrac{\partial f_1(x_1^{(k)}, x_2^{(k)})}{\partial x_2} \\ \dfrac{\partial f_2(x_1^{(k)}, x_2^{(k)})}{\partial x_1} & \dfrac{\partial f_2(x_1^{(k)}, x_2^{(k)})}{\partial x_2} \end{bmatrix}$$

и матрицы

$$\mathbf{A}_1^{(k)} = \begin{bmatrix} f_1(x_1^{(k)}, x_2^{(k)}) & \dfrac{\partial f_1(x_1^{(k)}, x_2^{(k)})}{\partial x_2} \\ f_2(x_1^{(k)}, x_2^{(k)}) & \dfrac{\partial f_2(x_1^{(k)}, x_2^{(k)})}{\partial x_2} \end{bmatrix},$$

$$\mathbf{A}_2^{(k)} = \begin{bmatrix} \dfrac{\partial f_1(x_1^{(k)}, x_2^{(k)})}{\partial x_1} & f_1(x_1^{(k)}, x_2^{(k)}) \\ \dfrac{\partial f_2(x_1^{(k)}, x_2^{(k)})}{\partial x_1} & f_2(x_1^{(k)}, x_2^{(k)}) \end{bmatrix}.$$

В итоге, воспользовавшись формулой определителя для матриц размерности 2×2, последующие приближения определяем как

$$\begin{cases} x_1^{(k+1)} = x_1^{(k)} - \dfrac{f_1(x_1^{(k)}, x_2^{(k)}) \cdot \dfrac{\partial f_2(x_1^{(k)}, x_2^{(k)})}{\partial x_2} - f_2(x_1^{(k)}, x_2^{(k)}) \cdot \dfrac{\partial f_1(x_1^{(k)}, x_2^{(k)})}{\partial x_2}}{\dfrac{\partial f_1(x_1^{(k)}, x_2^{(k)})}{\partial x_1} \cdot \dfrac{\partial f_2(x_1^{(k)}, x_2^{(k)})}{\partial x_2} - \dfrac{\partial f_2(x_1^{(k)}, x_2^{(k)})}{\partial x_1} \cdot \dfrac{\partial f_1(x_1^{(k)}, x_2^{(k)})}{\partial x_2}} \\ x_2^{(k+1)} = x_2^{(k)} - \dfrac{f_2(x_1^{(k)}, x_2^{(k)}) \cdot \dfrac{\partial f_1(x_1^{(k)}, x_2^{(k)})}{\partial x_1} - f_1(x_1^{(k)}, x_2^{(k)}) \cdot \dfrac{\partial f_2(x_1^{(k)}, x_2^{(k)})}{\partial x_1}}{\dfrac{\partial f_1(x_1^{(k)}, x_2^{(k)})}{\partial x_1} \cdot \dfrac{\partial f_2(x_1^{(k)}, x_2^{(k)})}{\partial x_2} - \dfrac{\partial f_2(x_1^{(k)}, x_2^{(k)})}{\partial x_1} \cdot \dfrac{\partial f_1(x_1^{(k)}, x_2^{(k)})}{\partial x_2}} \end{cases}.$$

В рассматриваемом примере:

$$f_1(x_1^{(k)}, x_2^{(k)}) = 0.1x_1^{(k)\,2} + x_1^{(k)} + 0.2x_2^{(k)\,2} - 0.3,$$

$$f_2(x_1^{(k)}, x_2^{(k)}) = 0.2x_1^{(k)\,2} + x_2^{(k)} - 0.1x_1^{(k)}x_2^{(k)} - 0.7,$$

$$\frac{\partial f_1(x_1^{(k)}, x_2^{(k)})}{\partial x_1} = 0.2x_1^{(k)} + 1, \quad \frac{\partial f_1(x_1^{(k)}, x_2^{(k)})}{\partial x_2} = 0.4x_2^{(k)},$$

$$\frac{\partial f_2(x_1^{(k)}, x_2^{(k)})}{\partial x_1} = 0.4x_1^{(k)} - 0.1x_2^{(k)}, \quad \frac{\partial f_2(x_1^{(k)}, x_2^{(k)})}{\partial x_1} = 1 - 0.1x_1^{(k)}.$$

Подставляя в правые части расчётных формул выбранные значения $x_1^{(0)}$, $x_2^{(0)}$, получим приближение $x_1^{(1)}$, $x_2^{(1)}$, используемое, в свою очередь, для нахождения $x_1^{(2)}$, $x_2^{(2)}$ и т.д.

Итерации завершаются при выполнении условия

$$\varepsilon^{(k+1)} = \left\|\mathbf{x}^{(k+1)} - \mathbf{x}^{(k)}\right\| \le \varepsilon \text{, где } \left\|\mathbf{x}^{(k+1)} - \mathbf{x}^{(k)}\right\| = \max_i \left|x_i^{(k+1)} - x_i^{(k)}\right|.$$

Результаты вычислений приведены в табл. 6.

Итоговое приближение корня $x_1^* \approx 0.1964$, $x_2^* \approx 0.7062$.

Таблица 6

k	$x_1^{(k)}$ $x_2^{(k)}$	$f_1(x_1^{(k)}, x_2^{(k)})$ $f_2(x_1^{(k)}, x_2^{(k)})$	$\frac{\partial f_1(x_1^{(k)}, x_2^{(k)})}{\partial x_1}$ $\frac{\partial f_2(x_1^{(k)}, x_2^{(k)})}{\partial x_1}$	$\frac{\partial f_1(x_1^{(k)}, x_2^{(k)})}{\partial x_2}$ $\frac{\partial f_2(x_1^{(k)}, x_2^{(k)})}{\partial x_2}$	$\varepsilon^{(k)}$
0	0.25000 0.75000	0.06875 0.04375	1.05000 0.02500	0.30000 0.97500	–
1	0.19696 0.70649	0.00066 0.00033	1.03939 0.00813	0.28260 0.98030	0.05304
2	0.19641 0.70615	0.00000 0.00000	1.03928 0.00795	0.28246 0.98036	0.00054
3	**0.19641** **0.70615**				**0.00000**

3.14. ПРОГРАММА #11

Ниже предлагается вариант алгоритма программы для уточнения значения корней системы из двух уравнений методом Ньютона.

```
АЛГОРИТМ «Метод Ньютона»
ВХОД     f(,), g(,), f1(,), g1(,), f2(,), g2(,), a, b, c, d, e
ВЫХОД    r, s, u, v, x, y, k
НАЧАЛО
    x:=(a+b)/2, y:=(c+d)/2
    r:=2*e
    k:=0
    ЦИКЛ «Итерация» (ПОКА r>e)
        u:=x, v:=y
        ЕСЛИ (f1(u,v)*g2(u,v)-f2(u,v)*g1(u,v)=0) КОНЕЦ
        x:=u-(f(u,v)*g2(u,v)-g(u,v)*f2(u,v))/(f1(u,v)*g2(u,v)-f2(u,v)*g1(u,v))
        y:=v-(f1(u,v)*g(u,v)-f(u,v)*g1(u,v))/(f1(u,v)*g2(u,v)-f2(u,v)*g1(u,v))
        ЕСЛИ (x>u) r:=x-u ИНАЧЕ r:=u-x
```

```
            ЕСЛИ (y>v) s:=y-v ИНАЧЕ s:=v-y
            ЕСЛИ (s>r) r:=s
            k:=k+1
        ПЕЧАТЬ x, y, k
КОНЕЦ
```

3.15. МЕТОД ЭКСТРАПОЛЯЦИИ ЭЙТКЕНА

Предположим, что отношение $\frac{x^{(k-1)} - x^*}{x^{(k)} - x^*} = q$ есть величина постоянная и неизменная в некотором итерационном процессе.

Тогда верно и $\frac{x^{(k-1)} - x^*}{x^{(k)} - x^*} = q = \frac{x^{(k-2)} - x^*}{x^{(k-1)} - x^*}$. Выражая отсюда x^*, имеем:

$$x^* = \frac{x^{(k)}x^{(k-2)} - (x^{(k-1)})^2}{x^{(k)} - 2x^{(k-1)} + x^{(k-2)}}.$$

Полученное таким образом значение x^* можно тогда принять за следующее приближённое значение $x^{(k+1)}$.

Описанный трёхшаговый способ ускорения сходимости итерационной последовательности называется экстраполяцией Эйткена и пригоден как для решения одного нелинейного уравнения, так и для решения систем нелинейных уравнений.

Однако исходное предположение также означает, что метод применим только к процессам с линейной сходимостью (метод простой итерации, метод Зейделя), чья сходимость в результате повышается до квадратичной, и неприменим, например, к методу Ньютона, методу секущих и т.п.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. *Формалёв В.Ф., Ревизников Д.Л.* Численные методы. — М.: Физматлит, 2004.
2. *Пирумов У.Г.* Численные методы: Учеб. пособие для студ. втузов. — М.: Дрофа, 2007.
3. *Гидаспов В.Ю., Иванов И.Э., Ревизников Д.Л., Стрельцов В.Ю., Формалёв В.Ф.* Численные методы: Сборник задач / Под ред. У.Г. Пирумова. — М.: Дрофа, 2007.
4. *Демидович Б.П., Марон И.А.* Основы вычислительной математики. — М.: ГИФМЛ, 1963.
5. *Бахвалов Н.С.* Численные методы. — М.: Наука, 1975.
6. *Калиткин Н.Н.* Численные методы. — М.: Наука, 1978.
7. *Турчак Л.И.* Основы численных методов. — М.: Наука, 1980.
8. *Волков Е.А.* Численные методы. — М.: Наука, 1987.
9. *Самарский А.А., Гулин А.В.* Численные методы. — М.: Наука, 1989.
10. *Каханер Д., Моулер К., Нэш С.* Численные методы и программное обеспечение. — М.: Мир, 1998.
11. *Рябенький В.С.* Введение в вычислительную математику: Учеб. пособие. — М.: Физматлит, 2000.
12. *Вержбицкий В.М.* Основы численных методов. Учеб. пособие для вузов. — М.: Высшая школа, 2002.

ОГЛАВЛЕНИЕ

Тем. план 2020, ч. 3, поз. 12

Северина Наталья Сергеевна
Сластушенский Юрий Викторович

ЧИСЛЕННЫЕ МЕТОДЫ И АЛГОРИТМЫ.
ЛИНЕЙНАЯ И ЭЛЕМЕНТАРНАЯ АЛГЕБРА

Редактор *Е.В. Дмитриева*
Компьютерная верстка *О.А. Пелипенко*

Сдано в набор 18.12.2020. Подписано в печать 27.04.2021.
Бумага писчая. Формат 60×84 1/16. Печать цифровая.
Усл. печ. л. 6,04. Уч.-изд. л. 6,5. Тираж 100 экз.
Заказ 017/855.

Издательство МАИ
(МАИ), Волоколамское ш., д. 4
Москва, А-80, ГСП-3 125993

Типография Издательства МАИ
(МАИ), Волоколамское ш., д. 4
Москва, А-80, ГСП-3 125993