В.М. Волков

Численные методы.
Конспект лекций в 2-х частях. Часть 1.

Учебно-методическое пособие для студентов специальности
1-31 03 01 «Математика,
1-31 03 01-05 «Математика (информационные технологии)»

МИНСК
2015

УДК 519.6(076.6)
ББК 22.193я73
К79

Р е ц е н з е н т ы:
доктор физико-математических наук, профессор Л.А.Янович
кандидат физико-математических наук, доцент А.И.Шербаф

Волков В.М.
К79 Численные методы. Конспект лекций в 2-х частях. Часть 1.: учебно-методическое пособие / В.М.Волков. – Минск : БГУ, 2015. – 92с.
I33SBN 978-985-518-812-5.

Компактное изложение материала лекций по курсу *численные методы* для студентов, обучающихся по специальности 1-31 03 01 «Математика, 1-31 03 01-05 «Математика (информационные технологии)». Курс лекций состоит из двух частей и рассчитан на два семестра. Первая часть содержит теоретические сведения по численным методам линейной алгебры и решению нелинейных уравнений и систем.

УДК 517.6(075.8)
ББК 22.19я73

ISBN

От автора

Данное пособие содержит материалы для курса лекций "Численные методы" в соответствие с учебной программной дисциплины для студентов механико-математического факультета БГУ. Пособие состоит из двух частей, каждая из которых рассчитана на один семестр в объеме 32 лекционных часа. В первой части излагаются методы решения задач линейной алгебры и систем нелинейных уравнений. Вторая часть посвящена методам решения дифференциальных и интегральных уравнений.

Данное пособие дает краткое изложение основных концепций и ключевых алгоритмов численного анализа линейных и нелинейных алгебраических задач, получивших широкое распространение в практике вычислений и давших начало современным вычислительным технологиям. Особенность данного пособия состоит в том, что рассмотренные алгоритмы приводятся в формате алгебраических операций над векторами и матрицами без детализации структуры алгоритма на уровне поэлементной арифметики. Предполагается при этом, что практические занятия проводятся с использованием среды программирования Matlab, где отсутствует необходимость такой детализации. В конце параграфов приводятся задания для упражнений, многие из которых предполагают использование пакета Matlab, описание некоторых функций которого и примеры реализации алгоритмов можно найти в приложении.

ВВЕДЕНИЕ

Численные методы алгебры играют ключевую роль в современных методах вычислений, поскольку практически все актуальные задачи современной вычислительной математики в конечном итоге замыкаются на решение задач алгебраического плана. Наиболее важные алгебраические задачи, ассоциируемые с компьютерными вычислениями, состоят в решении систем алгебраических уравнений и сопутствующих им проблем (обращение матриц, проблемы собственных значений и собственных векторов, минимизации функционалов и т.п.).

Численные методы линейной алгебры, несмотря на их многообразие и кажущуюся на первый взгляд разобщенность, имеют общую алгоритмическую особенность – решение задачи может быть выражено последовательностью алгебраических операций над матрицами и векторами. Если алгоритм позволяет получить решение за фиксированное число операций, то такой алгоритм называется *прямым*. Если метод основывается на повторяющейся вычислительной процедуре, каждый цикл (итерация) которой дает более точное приближение искомого решения, то такой метод принято называть *итерационным*.

Суть большинства прямых и итерационных численных методов линейной алгебры заключается в достаточно прозрачной идее. Существуют матрицы определенного вида, для которых задачи решения систем ЛАУ и проблема собственных значений в известной мере тривиальны (например, диагональные матрицы, матрицы треугольного вида и т.п.). В силу этого, имея дело с матрицей произвольного вида, естественно попытаться произвести редукцию исходной задачи к эквивалентной или контролируемо приближенной задаче с матрицей, удобной для последующего анализа. Ответ на вопрос о том, всегда ли осуществима и технически реализуема подобного рода редукция, отсылает нас к фундаментальным теоремам линейной алгебры и теории матриц.

Важно отметить отличие методов линейной алгебры и вычислительные аспекты соответствующих проблем. Выводы линейной алгебры базируются на том, что все вычисления выполняются точно. Компьютерные расчеты, в силу приближенности представления действительных чисел и операций с ними, вносят возмущения в вычислительный процесс. При определенных обстоятельствах, если метод имеет тенденцию к вычислительной неустойчивости или матрица задачи имеет плохую *обусловленность*, данные возмущения могут испытывать неконтролируемый рост и приводить к полной потере точности конечного результата или

даже к выходу за пределы стандартного представления действительных чисел с плавающей запятой. По этой причине, наряду с алгоритмическими аспектами численных методов неотъемлемой проблемой является анализ устойчивости и вычислительной погрешности.

Одним из важнейших и одновременно сложных вопросов практического использования численных методов является проблема выбора наиболее адекватного алгоритма для решения конкретной задачи. С этой точки зрения необходимо иметь четкие представления о границах применимости и вычислительной сложности алгоритмов, а также возможностях сокращения вычислительных затрат за счет специфических особенностей решаемой задачи.

Наконец, последнее, но не менее важное. Успешное использование численных методов невозможно без умения проанализировать корректность полученных результатов и, по возможности, оценить погрешность приближенного решения.

1. ЧИСЛЕННЫЕ МЕТОДЫ РЕШЕНИЯ СИСТЕМ ЛИНЕЙНЫХ АЛГЕБРАИЧЕСКИХ УРАВНЕНИЙ.

1.1. Основные понятия

Численные алгоритмы линейной алгебры могут быть представлены в виде последовательности алгебраических операций с векторами и матрицами. Напомним, что под вектором в данном случае понимается совокупность действительных чисел, упорядоченных в виде строки $x=(x_1,x_2,x_3,\cdots x_N)$, или столбца $x'=(x_1,x_2,x_3,\cdots x_N)^T$, N – размерность вектора, $x_k, k=1,2,\ldots,N$ – координаты вектора . Множество всех векторов одинаковой размерности образует векторное пространство, в котором определены следующие операции:

сложение, $x+y=(x_1+y_1,x_2+y_2,\ldots,x_N+y_N)$;

умножение на скаляр, $\alpha\cdot x=(\alpha x_1,\alpha x_2,\ldots,\alpha x_N)$;

скалярное произведение[1], $(x,y)=\sum_{k=1}^{N}x_k y_k$.

Система из K векторов называется *линейно независимой*, если

$\forall\alpha_k\neq 0,\quad k=1,2,\ldots,K,\quad \sum_{k}^{K}\alpha_k x_k\neq 0$, т.е. ни один из векторов системы не может быть представлен в виде линейной комбинации остальных векторов системы. Максимальное число линейно независимых векторов совпадает с размерностью векторного пространства. Любая система из N линейно независимых векторов образуют *базис* в N-мерном векторном пространстве, при этом любой вектор данного векторного пространства может быть однозначно представлен в виде линейной комбинации векторов базиса. Простейшим примером системы линейно независимых векторов N-мерного пространства является *естественный базис:*

$$e_1=(1,0,0,\ldots,0),\ e_2=(0,1,0,\ldots,0),\ \ldots,\ e_N=(0,0,\ldots 0,1).$$

Несложно убедиться, что компоненты вектора являются коэффициентами его представления в виде линейной комбинации векторов естественного базиса: $x=\sum_{k=1}^{N}x_k e_k$.

[1] Скалярное произведение может быть определено также другими способами в соответствии с аксиомами скалярного умножения.

Матрицей называется совокупность чисел a_{km}, упорядоченных в виде прямоугольной таблицы размера $K \times M$:

$$A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1M} \\ a_{21} & a_{22} & \cdots & a_{2M} \\ \cdots & \cdots & a_{km} & \cdots \\ a_{K1} & a_{K2} & \cdots & a_{KM} \end{bmatrix}$$

Матрицу можно рассматривать также как упорядоченную совокупность векторов, образующих ее строки или столбцы. *Индексы* элемента $a_{km} \equiv \{A\}_{km}$ определяют его позицию в k-той строке и m-том столбце (изменениям первого индекса соответствует изменение положения в столбце, а второго – положения в строке). Размерность матрицы $A \in R^{K \times M}$ указывает на то, что матрица имеет K строк и M столбцов соответственно. Если $K = M = N$, то матрица называется *квадратной* и определяет оператор в N-мерном векторном пространстве.

Матрицы, существенная часть элементов которых равна нулю, принято называть *разреженными*. Существует несколько классов матриц специального вида, обладающих рядом полезных свойств с точки зрения матричных вычислений:

- диагональные матрицы: $a_{kk} \neq 0$, $a_{m \neq k} = 0$;
- нижние треугольные матрицы: $a_{k \geq m} \neq 0$, $l_{k<m} = 0$;
- верхние треугольные матрицы: $a_{k>m} = 0$, $a_{k \leq m} \neq 0$;
- ленточные матрицы: $a_{|k-m| \leq p} \neq 0$, $a_{|k-m|>p} = 0$;
- матрицы перестановок, вращений и отображений.

Операции умножения матрицы на скаляр и сложение матриц одинаковой размерности определяются аналогично соответствующим векторным операциям.

Операция умножения матриц сводится к вычислению скалярных произведений векторов-строк первого сомножителя на вектор-столбцы второго. В частности, произведением матрицы $A \in R^{N \times M}$ на матрицу $B \in R^{M \times K}$ является матрица $AB = C \in R^{N \times K}$, элементы которой вычисляются следующим образом

$$c_{nk} = \sum_{m=1}^{M} a_{nm} b_{mk}, \; n = \overline{1, N}, \; k = \overline{1, K}.$$

Произведение матрицы $A \in R^{N \times M}$ на вектор $x \in R^{M \times 1}$ определяется аналогично умножению матриц соответствующих размерностей:

$$Ax = f = (f_1, f_2, \dots f_M)^T, \;\; f_k = \sum_{m=1}^{M} a_{km} x_m, \quad k = \overline{1, M}.$$

В матричном анализе часто фигурируют также операции обращения и транспонирования матриц. Матрица *обратная* заданной матрице A называется такая матрица A^{-1}, для которой выполняется тождество

$$A^{-1}A = E,$$

где E – единичная матрица, все элементы которой, за исключением единичных значений на главной диагонали, равны нулю: $e_{kk} = 1, \;\; e_{k \neq m} = 0$. Для произвольной квадратной *невырожденной* матрицы всегда существует *обратная*. Матрица A^T, *транспонированная* по отношению к заданной матрице A, получается из исходной матрицы заменой ее строк на столбцы: $\{A^T\}_{km} = \{A\}_{mk}$. Если транспонированная матрица совпадает с исходной, то такие матрицы называются *симметричными (симметрическими)*.

Матрицы, которые при транспонировании дают обратную матрицу, $A^T = A^{-1}$, называют *ортогональными*. Операция транспонирования для комплекснозначных матриц обобщается на понятие *сопряжённой* (*Эрмитово-сопряжённой*) матрицы. Сопряжённая матрица получается при транспонировании исходной матрицы с одновременной заменой всех ее элементов на комплексно-сопряженные: $A^* = conj(A^T)$. Матрицы, для которых $A^* = A^{-1}$, называют *унитарными*. Матрицы, для которых $A^* = A$, называются *самосопряжёнными*. Матрицы, для которых $A^* A = AA^*$ называются *нормальными*.

Упражнения:

1. Покажите, что произведение двух нижних (верхних) треугольных матриц есть нижняя (верхняя) треугольная матрица.
2. Докажите справедливость неравенства $\forall x, y \in R^N \;\; (x, y)^2 \leq (x, x)(y, y)$.
3. При каком условии неравенство п.2 обращается в равенство для ненулевых векторов $x, y \in R^N$.

1.2. Нормы векторов и матриц.

Поскольку численные методы по своей природе являются приближенными (по крайней мере, ввиду наличия вычислительной погрешности, обусловленной приближенностью компьютерной арифметики), важное значение имеют оценки погрешности (разности векторов точного и

приближенного решений). Для этих целей используются различные нормы линейного векторного пространства. Напомним, что нормой вектора $x \in R^N$ называется произвольный линейный положительно определенный функционал $\| x \|$ (скалярная функция векторного аргумента), который для любого элемента векторного пространства удовлетворяет трем условиям, известным как *аксиомы нормы*:

1. $\| x \| \geq 0, \quad \| x \| = 0 \Leftrightarrow x = 0$ – *положительная определенность*;
2. $\| \alpha x \| = | \alpha | \cdot \| x \|, \quad \forall \alpha \in R$ – *линейность при умножении на скаляр*;
3. $\| x + y \| \leq \| x \| + \| y \|$ – *неравенство треугольника*.

Векторные нормы, получившие наиболее широкое распространение в численном анализе:

$\| x \|_2 = \sqrt{(x, x)}$ – Евклидова норма (при соответствующем определении скалярного произведения);

$\| x \|_\infty = \max\limits_k \{ | x_k | \}$ – максимальная норма;

$\| x \|_A = \sqrt{(Ax, x)}$ – энергетическая норма, порожденная положительно определенным самосопряженным оператором $A > 0$.

Аксиоматику матричной нормы, в отличие от нормы вектора, полезно пополнить дополнительным условием. Поскольку для матриц определена операция умножения, то естественным является требование $\| AB \| \leq \| A \| \cdot \| B \|$ – *мультипликативность матричной нормы*.

Аксиома мультипликативности матричной нормы тесно связана с требованием *согласованности* норм матриц и векторов. В частности, для мультипликативной нормы автоматически выполняется оценка $\| Ax \| \leq \| A \| \cdot \| x \|$ (вектор x в данном случае может трактоваться как матрица, имеющая размерность $N \times 1$). Для получения *не улучшаемых* оценок произведения матрицы на вектор используют *нормы матриц, подчиненные соответствующим векторным нормам.*

Определение. Норма матрицы A, *подчиненная* векторной норме $\| \cdot \|$ определяется числом

$$\| A \| = \sup_{x \neq 0} \frac{\| Ax \|}{\| x \|} = \sup_{\| x \| = 1} \| Ax \| \qquad (1.1)$$

В случае квадратных матриц из определения подчиненной матричной нормы следует ее согласованность, мультипликативность и минимальность среди всех возможных согласованных норм.

Матричная норма $\| A\|_2 = \max\{\lambda(AA^T)^{1/2}\}$, подчиненная векторной Евклидовой норме, равна максимальному *сингулярному значению* матрицы. Сингулярные значения матрицы численно равны корню квадратному из собственных значений матрицы AA^T. Для симметричной, положительно определенной матрицы сингулярные числа совпадают со спектром собственных значений данной матрицы.

В качестве нормы $\| A\|_\infty$ выступает максимальное значение сумм абсолютных величин элементов строк матрицы:

$$\| A\|_\infty = \max_k \left\{ \sum_{m=1}^{N} | a_{km} | \right\}$$

Выбор конкретной нормы для получения оценок приближенного решения определяется в основном целью исследований и спецификой задачи. При этом следует иметь в виду, что нормы конечномерного линейного векторного пространства эквивалентны. Эквивалентность норм $\|\cdot\|_*$ и $\|\cdot\|_{**}$ понимается в смысле выполнения неравенств вида:

$$c \| x \|_* \le \| x \|_{**} \le C \| x \|_*,$$

где c, C некоторые положительные постоянные. Например, для норм $\|\cdot\|_2$ и $\|\cdot\|_\infty$ имеют место оценки

$$1/\sqrt{N} \| x \|_2 \le \| x \|_\infty \le \| x \|_2 . \qquad (1.2)$$

Упражнения:

1. Показать, что норма единичной матрицы равна единице для произвольной подчиненной матричной нормы.
2. Доказать справедливость неравенства (1.2).
3. Что больше спектральный радиус или любая подчиненная норма матрицы?
4. Приведите примеры матриц и векторов, для которых значения норм $\|\cdot\|_2$ и $\|\cdot\|_\infty$ совпадает.
5. С помощью функции pascal, norm и semilogy построить зависимость норм $\|\cdot\|_2$ и $\|\cdot\|_\infty$ матрицы Паскаля от размерности матрицы.

1.3. Системы линейных алгебраических уравнений. Разрешимость и устойчивость.

Решение системы линейных алгебраических уравнений для заданного вектора f и квадратной матрицы A размерности N состоит в поиске вектора x, удовлетворяющего равенству

$$Ax = f. \tag{1.3}$$

Данная задача для произвольного вектора f имеет единственное решение при условии, что матрица A *не вырожденная* (не имеет линейно зависимых строк). Критерием линейной независимости строк (столбцов) прямоугольной матрицы традиционно служит отличие от нуля *определителя матрицы*. Определитель квадратной матрицы представляет собой алгебраическую сумму всех возможных произведений элементов матрицы, в которые входят по одному элементу каждой строки и каждого столбца. Знаки слагаемых определяются четностью (нечетностью) перестановок порядковых номеров строк (столбцов) из элементов которых берется данный множитель.

Вычислительная сложность нахождения определителя матрицы, согласно определению данной характеристики, имеет порядок $O(NN!)$. В силу этого данный критерий крайне редко применяется в численном анализе непосредственно к исходной матрице системы даже при сравнительно небольших размерностях задачи. Кроме того, нахождение определителя, как вычислительная задача, с прикладной точки зрения не представляет особого интереса в силу приближенности компьютерных вычислений. Поскольку *критерий вырожденности матрицы* состоит в *строгом* равенстве нулю определителя, то вычисление его приближенного значения во многих случаях не позволяет корректно воспользоваться данным критерием. С другой стороны, определитель треугольной матрицы равен произведению диагональных элементов. Поэтому, если в процессе выполнения эквивалентных преобразований, приводящих матицу системы ЛАУ к треугольному виду, все диагональные элементы полученной треугольной матрицы отличны от нуля, то из эквивалентности приведенной и исходной задачи автоматически следует их одновременная разрешимость (или неразрешимость, если на главной диагонали полученной треугольной матрицы окажется хотя бы один нулевой элемент). Опять же, в силу приближенности компьютерных вычислений, возможны нюансы, когда, например, на главной диагонали приводимой матрицы возникают элементы *близкие* к нулю.

Разрешимость системы еще не означает, что в условиях приближенной компьютерной арифметики решение может быть получено с достаточной точностью. С этой точки зрения важным является то, чтобы рассматриваемая задача была *устойчива* к возмущениям *входных данных* – значений элементов матрицы и вектора правой части.

Наряду с системой уравнений (1.3) рассмотрим возмущенную задачу

$$\tilde{A}\tilde{x} = \tilde{f}, \qquad (1.4)$$

где $\tilde{A} = A + \delta A, \quad \tilde{f} = f + \delta f, \quad \tilde{x} = x + \delta x$. δA, δf – возмущения элементов матрицы и вектора правой части задачи соотвественно, а δx – разность между решениями возмущенной и исходной задачи. Задача считается устойчивой, если ее решение непрерывно зависит от возмущений входных данных. Условие устойчивости может быть сформулировано в виде оценки возмущения решения задачи от возмущения входных данных:

$$\| \delta x \| \le M_1 \| \delta f \| + M_2 \| \delta A \|, \qquad (1.5)$$

где M_1, и M_2 некоторые постоянные, зависящие от свойств матрицы A.

1.4. Число обусловленности матрицы. Оценка погрешности решения систем линейных алгебраических уравнений.

Компьютерные вычисления являются приближенными в силу того, что действительные числа представляются конечным числом двоичных (десятичных) разрядов. Относительная погрешность представления действительных чисел в формате double составляет приблизительно $2 \cdot 10^{-14}\%$. Естественно, что и результат арифметических операций с такими приближенными числами также будет приближенным. При решении систем линейных алгебраических уравнений число арифметических операций по порядку величины составляет величину $O(N^3)$. В силу этого анализ ошибки каждой из операций представляется практически неосуществимой задачей. По этой причине для оценки погрешности используют подход, получивший название *метод обратного анализа ошибки*. Суть данного метода состоит в оценке эквивалентных возмущений входных данных задачи, которые при гипотетически точных вычислениях влияют на результат также, как ошибки округления при реальном решении невозмущенной задачи. Обратный анализ ошибки состоит не в

оценке погрешности как таковой, а в выяснении того факта *какой задаче фактически соответствует полученное решение*. Например, пусть при решении задачи $Ax = f$ получено приближенное решение $\tilde{x} = x + \delta x$, с неизвестной погрешностью δx. Подстановка полученного решения в исходное уравнение показывает, что появление погрешности можно связать с возмущениями правой части задачи

$$A(x + \delta x) = f \quad \Rightarrow \quad Ax = f - A\delta x.$$

Таким образом, в условиях приближенности вычислений вместо исходной системы уравнений фактически решается некоторая возмущенная задача, точное решение которой и является приближенным решением, полученным в процессе вычислений. Рассмотрим, как возмущения правой части системы ЛАУ связаны с погрешностью ее решения при условии, что все промежуточные вычисления выполняются точно.

Итак, пусть правая часть системы возмущена или задана с абсолютной погрешностью δf. В силу этого вместо задачи $Ax = f$ мы решаем возмущенную задачу

$$A(x + \delta x) = f + \delta f. \tag{1.6}$$

где δx – абсолютная погрешность решения, ассоциированная с возмущением правой части. Погрешность δx является решением следующей задачи:

$$A\delta x = \delta f. \tag{1.7}$$

Несложные преобразования позволяют получить из уравнения (1.7) оценку нормы абсолютной величины погрешности:

$$A\delta x = \delta f \quad \Rightarrow \quad \| \delta x \| = \| A^{-1} \delta f \| \quad \Rightarrow \quad \| \delta x \| \le \| A^{-1} \| \cdot \| \delta f \|. \tag{1.8}$$

Заметим, что последнее неравенство (1.8) выражает устойчивость решения по правой части, и (согласно условию (1.5)) требование устойчивости эквивалентно требованию ограниченности нормы обратной матрицы системы ЛАУ. Из постановки задачи имеем

$$Ax = f \Rightarrow \quad \| A \| \cdot \| x \| \ge \| f \| \quad \Rightarrow \quad \| x \| \ge \| A \|^{-1} \cdot \| f \|. \tag{1.9}$$

Из неравенств (1.8) и (1.9) следует оценка относительной погрешности возмущенной задачи:

$$\frac{\| \delta x \|}{\| x \|} \le \| A^{-1} \| \cdot \| A \| \cdot \frac{\| \delta f \|}{\| f \|}. \tag{1.10}$$

Оценка (1.10) выражает зависимость относительной погрешности решения возмущенной задачи от относительной величины возмущения правой части в произвольной норме векторного пространства. В частности, согласно (1.10), погрешность возмущенной задачи пропорционально

возрастает с ростом возмущений правой части, причем коэффициент пропорциональности определяется произведением норм матриц A и A^{-1}. Число

$$M_A = \mathrm{cond}(A) = \| A \| \cdot \| A^{-1} \|, \qquad (1.11)$$

характеризующее зависимость относительной погрешности решения системы ЛАУ от величины относительного возмущения правой части, называется *числом обусловленности матрицы*. Если число обусловленности матрицы велико, то говорят, что данная матрица *плохо обусловлена*. Обычно показатель плохой обусловленности матрицы соотносится с относительной погрешностью выполнения арифметических операций. Например, если вычисления производятся с переменными класса double с относительной погрешностью $\sim 10^{-16}$, то, согласно оценке (1.10), число обусловленности $M_A \geq 10^{14}$ является критическим для гарантированного получения относительной погрешности приближенного решения менее одного процента.

Для оценки числа обусловленности удобно использовать спектральную норму матриц, которая является подчиненной для евклидовой векторной нормы. Для симметричных матриц число обусловленности определяется отношением наибольшего и наименьшего по абсолютной величине собственных значений матрицы.

Плохая обусловленность матрицы системы ЛАУ характеризует сильную чувствительность решения задачи к погрешности входных данных (малая ошибка входных данных приводит к существенному изменению решения). Аналогичным образом плохо обусловленные системы ЛАУ реагируют и на погрешности промежуточных вычислений. Так, например, в процессе приведения матрицы системы к треугольному виду выполняется ряд преобразований ее коэффициентов. В итоге, на этапе обратного хода метода Гаусса решается задача с возмущенными коэффициентами матрицы и полная оценка относительной погрешности должна исходить из следующей возмущенной задачи:

$$(A + \delta A)(x + \delta x) = f + \delta f, \qquad (1.12)$$

Если матрица возмущенной задачи (1.12) не вырожденная, то возмущения коэффициентов матрицы могут быть соотнесены с возмущениями правой части и вместо задачи (1.12) достаточно рассмотреть задачу

$$A(x + \delta x) = f + \Delta f, \quad \Delta f = \delta f + \delta A(x + \delta x), \qquad (1.13)$$

для которой имеет место оценка (1.10).

На практике оценить величину Δf можно на основе полученного приближенного решения задачи $\tilde{x} = x + \delta x$:

$$\Delta f = A\tilde{x} - f \ . \qquad (1.14)$$

Вектор $r = r(\tilde{x}) = A\tilde{x} - f$ принято называть *невязкой* решения системы ЛАУ. Норма данного вектора на практике служит количественной мерой возмущения правой части задачи и используется для оценки погрешности приближенного решения. Очевидно, что невязка точного решения тождественна нулю.

Упражнения:

1. Доказать, что число обусловленности матрицы не меньше единицы.
2. Как изменится число обусловленности матрицы при умножении её на скаляр?
3. Как будут соотноситься числа обусловленности матрицы $A \geq \alpha E$, $0 < \alpha << 1$ и матрицы $B = A + \alpha E$, где E – единичная матрица.
4. Проверьте правильность Вашего ответа на задание 3 с помощью численного эксперимента, используя для генерации матриц A, E соответственно функции pascal(10) и eye(10), полагая $\alpha = 1.e-5.$
5. Число обусловленности симметричной положительно определенной матрицы К=1.e9. Максимальное собственное значение при этом равно 1000. Оценить минимальное и максимальное собственные значения обратной матрицы.

1.5. Геометрический смысл и примеры плохой обусловленности матриц.

Геометрическую интерпретацию плохой обусловленности матриц полезно представить на примере системы двух уравнений.

$$\begin{cases} a_{11}x_1 + a_{12}x_2 = f_1, \\ a_{21}x_1 + a_{22}x_2 = f_2. \end{cases}$$

Каждое их уравнений системы можно трактовать как уравнение прямой на плоскости. Решение рассмотренной системы определяется координатами точки пересечения данных прямых.

Тангенсы углов, образуемых прямыми с осью (O, x_1), соответственно равны

$$\tan(\alpha_1) = -\frac{a_{11}}{a_{12}} \ , \ \tan(\alpha_2) = -\frac{a_{21}}{a_{22}} \ .$$

Вычислим определитель матрицы системы и покажем, что условие $\det(A)=0$ эквивалентно условию $\tan\alpha_1=\tan\alpha_2$:

$$\frac{a_{11}}{a_{12}}=\frac{a_{21}}{a_{22}},\ a_{11}a_{22}=a_{12}a_{21},\Rightarrow \det(A)=a_{11}a_{22}-a_{12}a_{21}=0$$

Таким образом, для системы двух уравнений равенство нулю определителя системы эквивалентно условию параллельности прямых, определяемых каждым из уравнений системы. Следовательно, если определитель равен нулю, то система либо не имеет решений (параллельные прямые не пересекаются), либо имеет бесконечное множество решений (прямые совпадают при $f_1=f_2=0$). Отличие от нуля определителя матрицы эквивалентно тому, что прямые не являются параллельными и имеют единственную точку пересечения, координаты которой и есть искомое решение системы.

Случай плохо обусловленной матрицы соответствует ситуации, когда угол между прямыми, определяемыми уравнениями системы, отличен от нуля, но является малым (прямые почти параллельны). Рассмотрим пример такой системы, полагая

$$a_{11}=-a_{12}=-10^{-6},\quad a_{22}=a_{12}=1,\quad f_1=f_2=1.$$

Несложно убедиться, что угол между прямыми, заданными уравнениями такой системы, составляет приблизительно $2\cdot10^{-6}$ рад., т.е. прямые пересекаются под очень малым углом в точке с координатами $x_1=0,\ x_2=1$. Данная точка соответствует точному решению задачи (на Рис. 1.1 обозначена круглым маркером).

Рассмотрим также возмущенную систему с вектором правой части $f=f+\delta f$, $\delta f_1=0$, $\delta f_2=10^{-5}$. Первое уравнение и определенная им прямая остаются неизменными, а график второй прямой смещается вдоль оси $(0,x_2)$ вверх на расстояние $\|\delta f\|=10^{-5}$. (см. Рис.2.1).

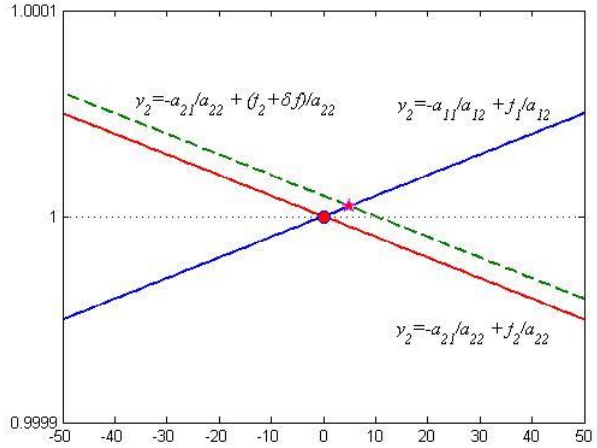

Рис. 2.1. Геометрическая интерпретация чувствительности решения плохо обусловленной системы к возмущениям правой части.

Несмотря на малость смещения второй прямой, точка пересечения прямых перемешается с исходной позиции на весьма существенное расстояние: $\delta x = (5 \cdot 10^{-6}, 5)$, $||\delta x|| \cong 5$. Учитывая неравенство (1.10) , мы можем оценить снизу число обусловленности матрицы задачи:

$$M_A = \mathrm{cond}(A) \geq \frac{||\delta x||}{||x||} \frac{||f||}{||\delta f||} \cong 5 \cdot 10^5.$$

Истинное число обусловленности матрицы рассмотренной задачи $M_A = 10^6$ и полученная оценка в данном случае дает вполне реальное представление о характере обусловленности задачи. Однако, в большинстве случаев неравенство (1.10) дает существенно завышенную оценку истинной погрешности и непригодно для оценки значения числа обусловленности.

В случае систем большей размерности геометрический смысл вырождения и плохой обусловленности матриц также как и в двумерном случае заключается в параллельной или близкой к параллельной ориентации гиперплоскостей, заданных отдельными уравнениями системы. Именно в таких ситуациях небольшие возмущение элементов матрицы или правой части системы вызывают серьезные изменения решения. Критическая ситуация возникает когда число обусловленности матрицы системы оказывается близким к обратной величине относительной погрешности представления действительных чисел в компьютерной арифметике. В таком случает сложно рассчитывать на получение решения задачи с приемлемой точностью.

Примером плохо обусловленной матрицы может служить матрица Гильберта

$$H = \begin{bmatrix} 1 & 1/2 & \cdots & 1/N \\ 1/2 & \ddots & \cdots & \cdots \\ \vdots & \vdots & \vdots & \vdots \\ 1/N & \cdots & \cdots & 1/(2N-1) \end{bmatrix}.$$

Число обусловленности матрицы Гильберта быстро возрастает с ростом размерности и при $N = 12$ превосходит 10^{16}, что делает практически невозможным численный анализ систем ЛАУ с подобной матрицей при использованием стандартного представления действительных чисел.

Упражнения:

1. С помощью функций Matlab cond и hilb постройте зависимость числа обусловленности матрицы Гильберта от ее размерности $N = 2 \div 20$ и изобразите полученную зависимость графически в логарифмическом масштабе с помощью функции semilogy.
2. Используя функции Matlab построить матрицу Гильберта A=hilb(20) и вектор-столбец случайных значений b=rand(20,1). Решите систему уравнений Ax=f: x=A\b. Объясните смысл предупреждения, которое генерируется при решении данной системы.
3. Найти норму разности решений однородной и неоднородной системы уравнений с матрицей Гильберта размерности 10x10. В качестве правой части неоднородной системы взять вектор 1.e-9*rand(10,1). Проверить выполнение оценки (1.10) для рассмотренного примера.

1.6. Прямые методы решения систем ЛАУ. Метод Гаусса

Большинство прямых методов решения систем ЛАУ в той или иной мере наследуют идею алгоритма последовательного исключения неизвестных – метода Гаусса. Идея достаточно прозрачна. Если матрица задачи, например, является верхней треугольной, в которой $a_{k \le m} \neq 0,\ a_{k>m} = 0$, то произвольное уравнение системы имеет вид

$$\sum_{m=k}^{N} a_{km} x_m = f_k\ ,\ k = \overline{1, N}\ .$$

Последнее уравнение системы содержит всего лишь одно неизвестное x_n, и его значение вычисляется первым:

$$x_N = f_N / a_{NN}\ . \qquad (1.15)$$

Остальные неизвестные вычисляются последовательно по явным формулам

$$x_k = \left(f_k - \sum_{m=k+1}^{N} a_{km} x_m \right) a_{kk}^{-1},\ \ k = N-1, N-2, \cdots, 2, 1. \qquad (1.16)$$

Совершенно аналогичный алгоритм может быть построен и для задачи с нижней треугольной матрицей.

Суть метода Гаусса состоит в приведении матрицы задачи к верхнему (нижнему) треугольному виду. Для этих целей используются тождественные преобразования системы. В частности, решение задачи не меняется при замене произвольной строки системы на линейную комбинацию данной строки и произвольного числа других строк системы. Ис-

пользуя такого рода преобразования, можно последовательно, столбец за столбцом, исключить переменные, находящиеся ниже главной диагонали системы ЛАУ. Элементарный шаг такого преобразования можно выразить с помощью операции умножения левой и правой части системы на *элементарную нижнюю треугольную матрицу* вида

$$L_{(k)} = \begin{bmatrix} 1 & 0 & \cdots & 0 & \cdots & 0 \\ 0 & \ddots & 0 & 0 & \cdots & 0 \\ 0 & 0 & 1/a_{kk} & 0 & \cdots & 0 \\ 0 & 0 & -a_{k+1,k}/a_{kk} & 1 & \cdots & \cdots \\ \cdots & \cdots & \cdots & \cdots & \ddots & 0 \\ 0 & \cdots & -a_{N,k}/a_{kk} & \cdots & 0 & 1 \end{bmatrix} \qquad (1.17)$$

Матрицы $L_{(k)}$ отличаются от единичной матрицы тем, что k-ый столбец данной матрицы, начиная с элемента $l_{kk} = a_{kk}^{-1}$, и ниже формируется специальным образом из соответствующих значений элементов преобразуемой матрицы $l_{k+n,k} = -a_{k+n,k}/a_{kk}$, $n = \overline{1, N-k}$. В результате умножения

$$A_{(1)} = L_{(1)} A \qquad (1.18)$$

преобразованная матрица $A_{(1)}$ будет иметь нулевые значения элементов первого столбца ниже главной диагонали: $a_{k>1,1}^{(1)} = 0$. Далее, на основе элементов матриц $A_{(k-1)}$, $k = 2,3,\cdots N$ последовательно формируются матрицы $L_{(k)}$ и находятся

$$A_{(2)} = L_{(2)} A_{(1)},\ A_{(3)} = L_{(3)} A_{(2)},\ \ldots\ A_{(N)} = L_{(N)} A_{(N-1)}. \qquad (1.19)$$

Нетрудно убедиться, что после выполнения описанных действий матрица $A_{(N)} = U$ будет верхней треугольной матрицей с единицами на главной диагонали.

Таким образом, метод последовательного исключения Гаусса состоит в преобразовании исходной системы уравнений путем умножения ее на элементарные треугольные матрицы, в результате чего исходная система уравнений преобразуется в эквивалентную систему с верхней треугольной матрицей

$$L_{(N)} L_{(N-1)} \cdot \ldots \cdot L_{(1)} Ax = Ux = L_{(N)} L_{(N-1)} \cdot \ldots \cdot L_1 f\,. \qquad (1.20)$$

Решение полученной системы уравнений с треугольной матрицей находится с помощью описанного выше алгоритма (1.15), (1.16).

Процедуру (1.18) – (1.20) принято называть *прямым ходом метода Гаусса*, а (1.15), (1.16) – соответственно *обратный ход*. Оценим вычислительные затраты прямого и обратного хода метода Гаусса.

Одна операция умножения $A_{(k)} = L_{(k)} A_{(k-1)}$ имеет вычислительную сложность порядка $O(N^2)$. В самом деле, каждый элемент матричного произведения находится как скалярное произведение двух векторов, причем вектор первого из сомножителей имеет не более двух ненулевых элементов. В результате каждый из N^2 элементов матрицы $A_{(1)}$ вычисляется не более чем за два умножения и одно сложение. Несложно заметить, что в процессе умножения $A_{(k)} = L_{(k)} A_{(k-1)}$ элементы матрицы $A_{(k-1)}$, расположенные выше $k-1$-ой строки и левее $k-1$-го столбца остаются неизменными: $a^{(k)}_{m<k-1,n<k-1} = a^{(k-1)}_{m<k-1,n<k-1}$. По этой причине вычислительная сложность умножения матриц $L_{(k+1)} A_{(k)}$ уменьшается с ростом k и не превосходит $3(N-k)^2$. Таким образом, вычислительные затраты на реализацию прямого хода метода Гаусса составляют $3\sum_{k=0}^{N-1}(N-k)^2 \cong N^3$ операций умножения и сложения. Формирование матриц L_k требует приблизительно $N^2/2$ операций. С учетом вышесказанного для матриц достаточно большой размерности N можно оценить вычислительную сложность прямого хода метода Гаусса величиной $O(N^3)$.

Для обратного хода метода Гаусса из выражения (1.16) легко подсчитать, что для вычисления k-ой компоненты решения требуется $2(N-k)+1$ операций. Суммирование вычислительных затрат для определения всех N неизвестных дает приблизительно N^2 операций, что на порядок меньше вычислительной сложности прямого метода Гаусса.

Относительно применимости метода Гаусса можно утверждать, что данный метод является наиболее универсальным в своем классе и может быть использован для решения произвольных систем ЛАУ с *неособенной* (не вырожденной) матрицей. Тем не менее, источником проблем, с которыми приходится иметь дело при использовании описанного выше алгоритма, являются нулевые (близкие к нулю) значения диагональных элементов, присутствующие в матрице изначально, либо появляющиеся в процессе исключения неизвестных. Как видно из выражения (1.17), при формировании матриц $L_{(k)}$ выполняется деление на a_{kk}, что в случае $a_{kk} = 0$ приводит к потере корректности алгоритма. Для устранения подобных проблем используется алгоритмы перестановок столбцов (строк) матрицы.

Упражнения:

1. С помощью численного эксперимента проверить эквивалентность решений систем линейных алгебраических уравнений $Ax = b,$ и $A^T Ax = A^T b$. В качестве матрицы A использовать матрицу Гильберта 20x20 (A=hilb(20)),вектор правой части – вектор случайных значений (b=randn(20,1)). Объяснить результат численного эксперимента, обращая внимание на предупреждения, генерируемое при выполнении операции численного решения системы уравнений: x=A\b.
2. Повторить численный эксперимент упражнения 1., используя матрицу Паскаля вместо матрицы Гильберта.

1.7.Метод Гаусса с выбором ведущего элемента

При перестановке строк системы ЛАУ решение задачи не изменяться. Данное свойство лежит в основе алгоритмов упорядочения строк матрицы, позволяющих обойти некоторые недостатки метода Гаусса и повысить его вычислительную устойчивость.

Стратегия *частичного упорядочения* состоит в следующем. Прежде чем приступить к формированию матрицы $L_{(k)}$ производится перестановка строк матрицы $A_{(k-1)}$ с номерами k и $n > k$, причем значение n определяется из условия $|a_{kn}^{(k)}| > |a_{k,n\geq k}^{(k)}|$. Таким образом на позиции *ведущего* элемента $a_{kk}^{(k)}$ после перестановки строк оказывается максимальный по модулю из элементов k-го столбца, расположенных ниже главной диагонали. Использование данных перестановок позволяет избежать деления на нуль при формировании матриц $L_{(k)}$. Кроме того, выбор главного элемента на каждом шаге исключения неизвестных во многих случаях предотвращает деление на числа близкие к нулю, что в условиях приближенной компьютерной арифметики способствует повышению вычислительной устойчивости алгоритма последовательного исключения неизвестных.

Алгоритмически перестановку строк матрицы можно реализовать путем умножения *матрицы перестановок* на преобразуемую матрицу[2]. Матрицей перестановок называется матрица, в каждой строке и каждом столбце которой содержится только один ненулевой элемент, равный единице, а остальные элементы равны нулю. Частным случает матрицы

[2] Программная реализация перестановок строк матрицы с помощью современных возможностей объектно- ориентированного программирования может быть реализована более эффективно, нежели непосредственное умножение с матрицей перестановок.

перестановок является единичная матрица. *Элементарной матрицей перестановки* называется матрица P_{nm}, полученная из единичной матрицы путем перестановки в ней строк с номерами n и m. Умножение матрицы P_{nm} на матрицу A приводит к перестановке в последней n-ой и m-ой строк.

Матрицы перестановок обладают рядом замечательных свойств.

- Матрица перестановок является унитарной матрицей: $P^{-1} \equiv P^T$.
- Произведение произвольного числа матриц перестановок является матрицей перестановок.
- Произвольную перестановку строк матрицы можно осуществить с помощью матрицы перестановок, полученной из произведения элементарных матриц перестановок.

Алгоритм частичного упорядочения с выбором главного элемента по столбцам фактически состоит в определении позиции главного элемента и построении элементарной матрицы перестановок. Пусть, например, при исключении неизвестных в системе ЛАУ с матрицей 5x5 после двух шагов исключения неизвестных имеем, что ведущий элемент $a_{33} = 0$, максимальный элемент третьего столбца находится в четвертой строке:

$$A^{(2)} = \begin{bmatrix} 1 & 1 & 3 & 2 & 12 \\ 0 & 1 & 5 & 11 & 7 \\ 0 & 0 & 0 & 6 & 3 \\ 0 & 0 & 19 & 7 & 1 \\ 0 & 0 & 2 & 4 & 9 \end{bmatrix}. \quad P_{34} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}, \; P_{34}A^{(2)} = \begin{bmatrix} 1 & 1 & 3 & 2 & 12 \\ 0 & 1 & 5 & 11 & 7 \\ 0 & 0 & 19 & 7 & 1 \\ 0 & 0 & 0 & 6 & 3 \\ 0 & 0 & 2 & 4 & 9 \end{bmatrix}.$$

Использование элементарной матрицы перестановок P_{34} позволяет предотвратить деление на нуль. Далее, формируется матрица L_3, вычисляется $A^{(3)} = L^{(3)}P_{34}A^{(2)}$ и выполняется следующий шаг исключения.

Замечание 1.1. Кроме рассмотренного алгоритма частичного упорядочения с выбором главного элемента по столбцам существуют аналогичные варианты упорядочения по строкам, а также по строкам и столбцам одновременно.

Замечание 1.2. Надежность алгоритма частичного упорядочения существенно повышается, если матрица системы ЛАУ *масштабирована* таким образом, что максимальные значения модулей элементов в каждой строке и каждом столбце имеют одинаковый порядок. Если матрица не отвечает требованиям масштабирования полезно, по крайней мере, предварительно выполнить нормировку строк, нарушающих баланс матрицы.

Замечание 1.3. Среди немногочисленных случаев, когда частичное упорядочение оказывается излишним, можно отметить диагонально-доминирующие и симметричные положительно-определенные матрицы.

Упражнение:

1. Построить матрицу P_{34} размерности 5x5. Проверить, что данная матрица является ортогональной: $P_{34}^T P_{34} = P_{34}^T P_{34} = E$.

1.8.LU-факторизация.

Как было показано выше, основные вычислительные затраты в методе Гаусса связаны с приведением матрицы системы ЛАУ к треугольному виду. Вычислительная сложность *прямого хода* метода Гаусса на порядок превосходит вычислительные затраты на обратный ход. В связи с этим во многих задачах оказывается целесообразным оптимизировать метод Гаусса с целью сохранения данных, полученных на этапе прямого хода. Данная модификация метода Гаусса получила название LU-*факторизация*. Суть данного метода состоит в разложении матрицы на два сомножителя L и U – соответственно нижнюю и верхнюю треугольные матрицы.

Основной теоретический результат, касающийся существования и единственности представления матрицы вида $A = LU$ заключается в следующем

Теорема 2.1. *Если* $\det(A) \neq 0$*, то существует матрица перестановок* P *такая, что имеет место разложение*

$$PA = LU, \tag{1.21}$$

где L *– нижняя треугольная матрица с отличными от нуля диагональными элементами,* U *– верхняя треугольная матрица с единичной главной диагональю.*

Согласно приведенной теореме LU*-факторизация* может использоваться для произвольной невырожденной матрицы.

Алгоритм вычисления матриц L и U во многом повторяет прямой ход метода Гаусса. В частности, из равенства (1.20) следует

$$U = L_{(N)} L_{(N-1)} \cdot \ldots \cdot L_{(1)} A. \tag{1.22}$$

Умножим равенство (1.22) на произведение матриц $L_{(1)}^{-1} L_{(2)}^{-1} \cdots L_{(N)}^{-1}$, в результате имеем:

$$L_{(1)}^{-1} L_{(2)}^{-1} \cdots L_{(N)}^{-1} U = L_{(1)}^{-1} L_{(2)}^{-1} \cdots L_{(N)}^{-1} L_{(N)} L_{(N-1)} \cdot \ldots \cdot L_{(1)} A = A.$$

Из полученного равенства следует

$$L = L_{(1)}^{-1} L_{(2)}^{-1} \cdots L_{(N)}^{-1}. \qquad (1.23)$$

Относительно элементарных треугольных матриц $L_{(k)}$ известно, что обратные им матрицы $L_{(k)}^{-1}$ также являются элементарными треугольными, причем существует простая связь между элементами матриц $L_{(k)}^{-1}$ и $L_{(k)}$:

$$L_{(k)}^{-1} = \begin{bmatrix} 1 & 0 & \cdots & 0 & \cdots & 0 \\ 0 & \ddots & 0 & 0 & \cdots & 0 \\ 0 & 0 & 1/a_{kk} & 0 & \cdots & 0 \\ 0 & 0 & -a_{k+1,k}/a_{kk} & 1 & \cdots & \cdots \\ \cdots & \cdots & \cdots & \cdots & \ddots & 0 \\ 0 & \cdots & -a_{N,k}/a_{kk} & \cdots & 0 & 1 \end{bmatrix}^{-1} = \begin{bmatrix} 1 & 0 & \cdots & 0 & \cdots & 0 \\ 0 & \ddots & 0 & 0 & \cdots & 0 \\ 0 & 0 & a_{kk} & 0 & \cdots & 0 \\ 0 & 0 & a_{k+1,k} & 1 & \cdots & \cdots \\ \cdots & \cdots & \cdots & \cdots & \ddots & 0 \\ 0 & \cdots & a_{N,k} & \cdots & 0 & 1 \end{bmatrix}. \qquad (1.24)$$

Также как и в методе Гаусса, при использовании алгоритма LU-*факторизации* на этапе формирования матриц $L_{(k)}$ согласно формуле (1.24) может оказаться, что $a_{kk}^{(k)} = 0$, и это приводит к потере корректности алгоритма. Подобные ситуации могут быть исключены с помощью описанной выше стратегии выбора ведущего элемента путем перестановки строк (столбцов) матрицы.

Вычислительная сложность алгоритма LU-факторизации $O(N^3)$, т.е. по порядку величины не превосходит вычислительные затраты в методе Гаусса.

Алгоритм LU разложения полезен в тех случаях, когда требуется решить несколько систем ЛАУ с одной и той же матрицей и разными правыми частями. После того как LU разложение матрицы получено, задача решения системы линейных алгебраических уравнений $Ax = f$ сводится к последовательному решению двух систем с матрицами треугольного вида: $Ly = f$, $Ux = y$. Таким образом, однократное выполнение LU разложения позволяет на порядок сократить вычислительные затраты при серийных расчетах (многократных решениях систем ЛАУ с одинаковой матрицей).

Наряду с LU факторизацией для решения систем линейных алгебраических уравнений можно использовать QR- разложение, где Q и R – соответственно ортогональная и верхняя (правая) треугольная матрицы. Вычислительная сложность алгоритма QR- разложения несколько выше, чем для LU факторизации, но он более устойчив к ошибкам

округления. Подробнее о нем при рассмотрении проблемы собственных значений.

Упражнение

1. С помощью функции lu проведите соответствующую декомпозицию матрицы Пуассона (A=gallery('poisson',5), [L,U]=lu(A)). Сравните структуру матриц A,L,U , используя визуализацию позиций ненулевых элементов с помощью функции spy.

1.9. Разложение Холецкого (метод квадратного корня).

В случае симметричной невырожденной матрицы есть возможность провести факторизацию более эффективно. В частности симметричную матрицу можно представить в виде произведения нижней треугольной матрицы L и транспонированной ей матрицы L^T:

$$a_{km} = a_{mk} \quad \Rightarrow \; A = LL^T . \tag{1.25}$$

Разложение (1.25) аналогично LU разложению, поскольку L^T является верхней треугольной матрицей. Однако, в отличие от LU разложения, факторизация Холецкого реализуется за меньшее число арифметических действий и не требует дополнительных ресурсов памяти для хранения матрицы U.

Элементы матрицы L находятся непосредственно из уравнения (1.25), из которого согласно определению матричного умножения следует

$$a_{mk} = \sum_{n=1}^{k} l_{mn} l_{nk} . \tag{1.26}$$

Заметим, что в скалярном произведении m-ой строки матрицы L на k-*й* столбец матрицы L^T суммирование в (1.26) ведется не по всем компонентам векторов, а только для тех, которые имеют ненулевые значения. В силу этого сумма в равенстве (1.26) при $k = 1$ содержит лишь одно отличное от нуля слагаемое:

$$a_{m1} = l_{m1} l_{11} \quad \Rightarrow \quad l_{11} = \sqrt{a_{11}}, \quad l_{m1} = a_{m1} / l_{11}, \quad m = 2,3,\ldots, N. \tag{1.27}$$

В случае $k = m > 1$ из равенства (1.26) следует

$$a_{mm} = \sum_{n=1}^{m} l_{mn} l_{nm} = \sum_{n=1}^{m} l_{mn}^2 \quad \Rightarrow \quad l_{mm} = \sqrt{a_{mm} - \sum_{n=1}^{m-1} l_{mn}^2} \;, \; m = 2,3,\ldots N . \tag{1.28}$$

Для $1 < k < m$ из равенства (1.26) имеем

$$a_{mk} = \sum_{n=1}^{k} l_{mn} l_{nk} = l_{mk} l_{kk} + \sum_{n=1}^{k-1} l_{mn} l_{nk} \quad \Rightarrow \quad l_{mk} = \left(a_{mk} - \sum_{n=1}^{k-1} l_{mn} l_{nk} \right) l_{kk}^{-1}, \qquad (1.29)$$

$$m = 3,4,\ldots N, \quad k = \overline{2, m-1}.$$

Формулы (1.25) – (1.29) дают явные рекуррентные выражения для вычисления элементов матрицы L. Применимость описанного алгоритма, помимо симметричности матрицы, ограничена также требованием положительности величин $a_{mm} - \sum_{n=1}^{m-1} l_{mn}^2$, стоящих под знаком квадратного корня в выражении (1.28). Данное требование всегда выполняется для положительно определенных матриц с диагональным преобладанием.

Подсчеты вычислительных затрат на реализацию алгоритма факторизации Холецкого позволяют определить, что вычислительная сложность данного метода имеет порядок $O(N^3)$, т.е. сравнима с вычислительными затратами при LU разложении. Более тщательное сравнение этих методов факторизации показывает, что на вычисление элементов матрицы L в методе Холецкого требуется примерно в два раза меньше арифметических операций, чем при LU разложении.

После выполнения разложения Холецкого, как и в случае LU разложения, задача решения системы линейных алгебраических уравнений $Ax = f$ сводится к последовательному решению двух систем с матрицами треугольного вида: $Ly = f$, $L^T x = y$.

Упражнение

С помощью функции chol проведите соответствующую декомпозицию матрицы Пуассона (A=gallery('poisson',5), L=chol(A)). Сравните структуру матриц A,L,L^T, используя визуализацию позиций ненулевых элементов с помощью функции spy. Оцените число ненулевых элементов в матрицах: NZA=length(find(A)), NZL=length(find(L)). Повторите эксперимент с матрицей A=gallery('poisson',20).

1.10.Метод ортогонализации.

Система линейных алгебраических уравнений с невырожденной матрицей $A \in R^{N \times N}$, может быть записана в виде

$$\sum_{m=k}^{N} a_{km} x_m - f_k = 0,\ k = \overline{1,N},$$

или

$$(a_k, y) = 0,\ \ a_k = (a_{k1}, a_{k2}, \ldots a_{kN-1}, a_{kN}, f_k),\ y = (x_1, \ldots x_N, 1)^T,\ k = \overline{1,N}.$$

Таким образом, решение системы линейных алгебраических уравнений сводится к поиску вектора $y \in R^{N+1}$, $y_{N+1} = 1$, который ортогонален линейно независимой системе векторов коэффициентов матрицы, включающих в качестве последних компонент компоненты вектора правой части. Первые N компонент вектора y содержат вектор искомого решения $x \in R^N$, а последняя компонента равна единице.

В силу отмеченных обстоятельств, для решения задачи может быть использован алгоритм ортогонализации, применение которого к системе векторов коэффициентов системы a_k, $k = \overline{1,N}$ позволит получить искомый вектор y. В частности, с точностью до постоянного множителя искомый вектор $y = \alpha w$ может быть вычислен посредством следующего рекурсивного алгоритма, построенного на основе процедуры ортогонализации Грама – Шмита:

$$s_1 = a_1, \quad w_1 = s_1 / \sqrt{(s_1, s_1)},$$

$$s_k = a_k - \sum_{m=1}^{k-1} c_m w_m, \quad c_m = (a_k w_m), \quad w_k = s_k / \sqrt{(s_k, s_k)},\ k = \overline{2, N+1}.$$

Вектор $w_{N+1} = (w_{N+1,1}, w_{N+1,2}, \ldots, w_{N+1,N+1})$ будет ортогонален всем векторам $a_k, k = \overline{1,N}$, и, в силу линейной независимости последних, он будет ненулевым. Искомое решение находится из полученного вектора w_{N+1} путем нормировки его на последнюю компоненту: $x_k = w_{N+1,k} / w_{N+1,N+1}, \quad k = \overline{1,N}$.

Среди прямых методов метод ортогонализации один из наиболее устойчивых к накоплению погрешности, однако по быстродействию он проигрывает методу Гаусса.

Упражнение

1. С помощью функции qr проведите QR-декомпозицию матрицы Пуассона (A=gallery('poisson',5), [Q,R]=chol(A)). Сравните структуру матриц A,Q,R, исполь-

зуя визуализацию позиций ненулевых элементов с помощью функции spy. Сравните результаты QR и LU декомпозиции.

1.11.Метод прогонки.

Рассмотрим решение систем ЛАУ с матрицей трехдиагонального вида. Такого рода системы имеют широкий спектр приложений в численном анализе краевых задач для дифференциальных уравнений второго порядка. Структура системы следующая:

$$\begin{aligned} &a_m x_{m-1} - c_m x_m + b_m x_{m+1} = f_m, \quad m = \overline{2, N-1}, \\ &x_1 = q_1 x_2 + g_1, \quad x_N = q_2 x_{N-1} + g_2. \end{aligned} \tag{1.30}$$

Решение задачи будем искать в виде

$$x_m = \alpha_{m+1} x_{m+1} + \beta_{m+1}, \quad m = N-1, N-2, \ldots 2, \tag{1.31}$$

где коэффициенты α, β подлежат определению. Выразим из (1.31)

$$x_{m-1} = \alpha_m x_m + \beta_m = \alpha_m(\alpha_{m+1} x_{m+1} + \beta_{m+1}) + \beta_m,$$

подставим выражения для x_m и x_{m-1} в (1.30), и, группируя слагаемые, получим

$$\left[\alpha_{m+1}(a_m \alpha_m - c_m) + b_m\right] x_m + \left[\beta_{m+1}(a_m \alpha_m - c_m) + a_m \beta_m - f_m\right] = 0.$$

Приравнивая нулю выражения в квадратных скобках имеем рекуррентные соотношения для определения коэффициентов α, β:

$$\alpha_{m+1} = \frac{b_m}{c_m - a_m \alpha_n}, \quad \beta_{m+1} = \frac{a_m \beta_n - f_m}{c_m - a_m \alpha_n}, \quad m = \overline{2, N-1}. \tag{1.32}$$

Из первого уравнения системы, $x_1 = q_1 x_2 + g_1$, и (1.31) , $m = 1$ следует

$$\alpha_2 = q_1, \ \beta_2 = g_1. \tag{1.33}$$

Аналогично, из последнего уравнения системы, $x_N = q_N x_{N-1} + g_N$, и (1.31) , $m = N-1$, находим

$$x_N = \frac{g_2 + q_2 \beta_N}{1 - q_2 \alpha_N}. \tag{1.34}$$

Реализация описанного алгоритма прогонки осуществляется по челночному принципу:

1) определяются значения коэффициентов α_2 и β_2 согласно (1.33);

2) по формулам (1.32) вычисляются коэффициенты α_m, β_m $m = 2, N-1$;

3) вычисляется значение x_N (1.34);

4) производится расчет решения по рекуррентной формуле (1.31).

Данный метод известен как метод *прогонки* или *алгоритм Томаса*. Для применимости алгоритма прогонки достаточно потребовать, чтобы знаменатели в выражениях (1.32), (1.34) не обращались в нуль. В случае большой размерности системы важно также чтобы коэффициенты α не превосходили по абсолютному значению единицы, поскольку в противном случае алгоритм будет иметь тенденцию к неустойчивости. Достаточным условием устойчивости алгоритма прогонки является следующие ограничения на коэффициенты задачи:

$$a_m \neq 0, b_m \neq 0, \ |c_m| \geq |a_m| + |b_m| \quad m = \overline{2, N-1}, \quad |q_1| \leq 1, \quad |q_2| < 1.$$

Приведенные условия устойчивости выполняются для большинства практически значимых приложений.

Существуют ряд полезных обобщений метода прогонки, например, для случая 5-ти и 7-ми диагональных матриц, блочно-трехдиагональных матриц (*матричная прогонка*), и матриц, соответствующих периодическим краевым условиям вида $x_1 = x_N$ (*циклическая прогонка*).

При более детальном рассмотрении метода прогонки несложно заметить аналогию данного метода и метода LU факторизации. Прямой и обратный ход метода прогонки фактически соответствует решению систем уравнений с двухдиагональными матрицами треугольной структуры.

1.12. Итерационные методы решения систем линейных алгебраических уравнений. Условия сходимости итераций.

Среди недостатков прямых методов решения систем ЛАУ следует отметить сравнительно большие вычислительные затраты, возрастающие в общем случае пропорционально N^3, и склонность к накоплению вычислительной погрешности. В силу этого прямые методы используются, как правило, при решении систем сравнительно небольшой размерности $N \leq 10^2 \div 10^4$.

В большинстве случаев при решении систем ЛАУ большой размерности приходится иметь дело с так называемыми *разреженными матрицами*, количество *ненулевых* элементов которых возрастает пропорционально N при общем числе элементов N^2. Нулевые элементы

матрицы не несут полезной информации и могут быть проигнорированы в матричных вычислениях. В рамках прямых методов игнорирование нулевых элементов матрицы весьма нетривиальная задача. Основная проблема здесь в том, что в процессе матричных преобразований общее число ненулевых элементов не сохраняется и, как правило, возрастает. Например, суммарное число ненулевых элементов разреженных ленточных матриц при LU разложении или при факторизации Холецкого может многократно превосходить число ненулевых элементов исходной матрицы. Предварительное упорядочение элементов матрицы, которые уменьшают ширину ленточной области локализации ненулевых элементов, при использовании алгоритмов факторизации позволяет в ряде случаев снизить остроту проблемы, однако полностью устранить рост числа ненулевых элементов в общем случае, по-видимому, не представляется возможным.

В силу отмеченных выше причин для систем линейных алгебраических уравнений с большими сильно разреженными матрицами более предпочтительными оказываются итерационные методы решения.

Общая схема большинства итерационных методов решения систем ЛАУ

$$Ax = f, \tag{1.35}$$

с невырожденной матрицей A и заданным вектором правой части f имеет вид

$$x^{(k+1)} = Sx^{(k)} + g, \ k = 0,1,2,\ldots, \tag{1.36}$$

где S – *матрица итерационного метода* или *матрицей перехода от k-ой итерации к (k+1)-ой,* $x^{(0)}$ – произвольный вектор (как правило, $x^{(0)} = 0$), называемый *начальным приближением* итерационного процесса. Последовательность $x^{(k)}, k = 0,1,2,\ldots$ будем называть *итерационными приближениями* искомого решения.

Итерационный метод, в котором для вычисления каждого нового итерационно приближения $x^{(k+1)}$ используется только предшествующее значение $x^{(k)}$ называют *итерационным методом первого порядка*, или *одношаговым итерационным методом.*

Для того, чтобы итерационный процесс (1.36) действительно приводил к решению поставленной задачи (1.35) необходимо и достаточно выполнение двух вполне очевидных условий:

1. последовательность векторов $x^{(k)}$, $k = 0,1,2,\ldots$, сходится.
2. предел данной последовательности является решением (1.35).

Из условия *2*. следует, что матрица S и вектор g могут быть заданы в виде:

$$S = E - B^{-1}A,\ g = B^{-1}f, \tag{1.37}$$

где E – единичная матрица, B – произвольная невырожденная матрица, выбор которой призван удовлетворить условие 1. Заметим, что для произвольных невырожденных матриц A и B существует единственный вектор x такое, что $x = Sx + g$, и, с учетом выбора (1.37),

$$B^{-1}Ax = B^{-1}f \quad \Rightarrow Ax = f.$$

Теорема 2.2. *Пусть* $\| S \| \le q < 1$, *x – решение (1.35). Тогда итерационный процесс (1.36), (1.37) сходится со скоростью геометрической прогрессии к вектору* x *и для погрешности итерационного метода* $\delta^{(k)} = x^{(k)} - x$ *выполняется оценка*

$$\| \delta^{(k+1)} \| \le q^k \| \delta^{(0)} \|. \tag{1.38}$$

Доказательство: Вычитая из (1.36) равенство $x = Sx + g$, для погрешности метода имеем:

$$\delta^{(k+1)} = S\delta^{(k)}.$$

Переходя к норме и проводя последовательные оценки получим

$$\left\|\delta^{(k+1)}\right\| = \left\|S\delta^{(k)}\right\| \le \|S\| \cdot \left\|\delta^{(k)}\right\| \le \|S\|^2 \left\|\delta^{(k-1)}\right\| \le \|S\|^{k+1} \left\|\delta^{(0)}\right\| \le q^{k+1} \cdot \left\|\delta^{(0)}\right\|$$

Поскольку $q < 1$, то $\lim\limits_{k\to\infty} \| \delta^{(k+1)} \| = \| \delta^{(0)} \| \lim\limits_{k\to\infty} q^{k+1} = 0$, и тогда $\lim\limits_{k\to\infty} x^{(k)} = x$, что и требовалось доказать.

Разнообразие итерационных методов связано с выбором конкретного вида матрицы B, которую принято называть *предобусловливателем (preconditioner).* Если матрица B одинакова для всех итераций, то такой итерационный процесс называется *стационарным.* Среди *нестационарных* итерационных методов традиционно используются предобусловливатели вида $B\tau_k^{-1}$, где *итерационный параметр* τ_k для каждой итерации выбирается из расчета наибольшей *скорости сходимости*. Под скоростью сходимости стационарного итерационного метода будем понимать скорость убывания погрешности итерационных приближений. Количественной характеристикой скорости сходимости является знаменатель геометрической прогрессии $q = \|S\|$ (см. (1.38)).

Итерационный процесс (1.36), (1.37) с точки зрения алгоритмической реализации удобно представить в виде

$$B\left(x^{(k+1)} - x^{(k)}\right) = -Ax^{(k)} + f. \tag{1.39}$$

При использовании (1.39), в отличие от (1.36), (1.37), не требуется явный вид матрицы B^{-1}, и вычисление очередного итерационного приближе-

ния сводится к решению системы алгебраических уравнений для вычисления вектора поправки $w^{(k+1)} = x^{(k+1)} - x^{(k)}$:

$$Bw^{(k+1)} = -r^{(k)}, \quad r^{(k)} = Ax^{(k)} - f, \quad x^{(k+1)} = x^{(k)} + w^{(k+1)}. \qquad (1.39')$$

Заметим, что вычисляемая согласно (1.39') невязка очередного приближения $r^{(k)}$ может быть использована для оценки его погрешности и формулировки критерия остановки итераций в виде $\| r^{(k)} \| = \| A^{-1}\delta^{(k)} \| \le \varepsilon$.

С точки зрения минимизации вычислительных затрат при выборе матрицы B естественно руководствоваться следующими критериями.

- Система уравнений $Bw = r$ должна иметь существенно более эффективный способ численного решения, нежели исходная система (1.35).
- Норма матрицы итерационного метода $S = E - B^{-1}A$ должна иметь как можно меньшее значение, $\| S \| \le q < 1$.

Сформулированные требования являются во многом взаимоисключающими друг друга. Например, с точки зрения второго требования идеальный выбор матрицы переобуславливателя является $B = A$. Однако такой выбор лишен здравого смысла с точки зрения первого требования. Построение эффективного итерационного метода во многом состоит в поиске разумного компромисса в стремлении минимизации вычислительных затрат на каждой итерации и минимизации числа итераций для достижения заданной точности. По эффективности реализации более предпочтительными являются итерационные методы с диагональной матрицей B, которые принято называть *явными*.

1.13.Метод простой итерации.

Стационарный одношаговый итерационный метод вида

$$x^{(k+1)} = x^{(k)} - \tau(Ax^{(k)} - f) \qquad (1.40)$$

называется методом *простой итерации.* В трактовке общей схемы одношаговых методов (1.36) матрица метода простой итерации имеет вид $S = E - \tau A$. В данном методе фактически отсутствует переобуславливатель. Его роль выполняет единичная матрица и итерационный параметр τ. Согласно представлению (1.37), для метода простой итерации $B = \tau^{-1}E$.

Теорема 2.3. *Пусть* A *– симметричная положительно определенная матрица* $A^T = A > 0$*, тогда итерационный метод (1.40) сходится при условии* $\tau < 2 \| A \|^{-1}$.

Доказательство: Спектральная норма симметричной положительной матрицы определяется модулем ее наибольшего собственного значения: $\| A \| = \max(|\lambda(A)|)$. Если A – *симметричная* матрица, то матрица $S = E - \tau A$ также будет симметричной и $\lambda(S) = 1 - \lambda(A)$. Тогда $\max\{|\lambda(S)|\} = \| S \| = \max\{|1 - \tau\lambda(A)|\}$. Из положительной определенности матрицы A следует, что при $\tau < 2 \| A \|^{-1}$ выполняется оценка $0 < \tau\lambda(A) < 2$, и при этом $\| S \| < 1$, что и требовалось доказать.

Замечание. 1.4. Условие сходимости метода простой итерации может быть сформулировано в виде $2E - \tau A > 0$.

Определим условия, при которых скорость сходимости итерационного метода (1.40) максимальна.

Теорема 2.4. *Пусть* A *– симметричная положительно определенная матрица:* $A^T = A$, $\gamma_1 E \le A \le \gamma_2 E$*, где* γ_1 *и* γ_2 *– минимальное и максимальное собственные значения матрицы* A *соответственно. Оптимальное значение итерационного параметра* τ*, обеспечивающее максимальную скорость сходимости итерационного процесса (1.40), есть величина* $\tau = \tau_0 = 2/(\gamma_1 + \gamma_2)$*, при этом*

$$\| \delta^{(k+1)} \| \le q \| \delta^{(k)} \|, \quad q = \frac{1 - \gamma_1/\gamma_2}{1 + \gamma_1/\gamma_2}. \qquad (1.41)$$

Доказательство: Задача поиска *оптимального значения итерационного параметра* $\tau = \tau_0$, обеспечивающего максимальную скорость сходимости итераций, состоит в определении условия минимума нормы матрицы итераций $\| S \|$, как функции итерационного параметра τ. Найдем явный вид данной функции:

$$\Psi(\tau) = \| S \| = \max\{|\lambda(S)|\} = \max\{|1 - \tau\lambda(A)|\} = \max\{|1 - \tau\gamma_1|, |1 - \tau\gamma_2|\}.$$

Несложно заметить, что $\psi(0) = \psi(2/\gamma_2) = 1$ и $\psi(0 < \tau < 1/\gamma_2) < 1$, следовательно, в интервале значений $\tau \in (0, 2/\gamma_2)$ функция $\psi(\tau)$ принимает минимальное значение. Поскольку функция $\psi(\tau)$ определяется максимальным значением модулей двух линейных функций, то минимум такой функции может достигаться только в точке равенства модулей данных линейных функций. Уравнение $|1 - \tau\gamma_1| = |1 - \tau\gamma_2|$ имеет единственный корень на интервале $\tau \in (0, 2/\gamma_2)$: $\tau = \tau_0 = 2/(\gamma_1 + \gamma_2)$. При этом

$$\psi(\tau_0) = \min_{\tau \in (0, 2/\gamma_2)} \{\| S \|\} = \frac{1 - \gamma_1 / \gamma_2}{1 + \gamma_1 / \gamma_2}.$$

Теорема доказана.

Заметим, что скорость сходимости метода простой итерации зависит от отношения γ_1 / γ_2 даже в случае оптимального выбора итерационного параметра. Для симметричных положительно определенных матриц $\gamma_2 = \| A \|$, $\gamma_1 = 1 / \| A^{-1} \|$. Следовательно $\gamma_1 / \gamma_2 = (\| A \| \cdot \| A^{-1} \|)^{-1} = K_A^{-1}$, где K_A – число обусловленности матрицы A. В случае плохо обусловленных матриц значение K_A велико, и тогда, согласно оценке (1.41),

$$q = \frac{1 - K_A^{-1}}{1 + K_A^{-1}} \to 1.$$

Таким образом, эффективность метода простой итерации может катастрофически ухудшаться при $K_A > 10^2 \div 10^5$.

Упражнения:

1. Используя в качестве примера программу из п.4.2 реализуйте метод простой итерации для системы ЛАУ с матрицей Пуассона (A=gallery('poisson',5)). Для этого, например, в реализации метода минимальных невязок нужно изменить способ задания итерационного параметра, полагая его постоянным tau = 1.9/normest(A);
2. Провести численный эксперимент и убедиться, что при выборе значения итерационного параметра tau = 2.01/normest(A) метод простой итерации расходится, а при tau = 1.99/normtst(A) – сходится.

1.14.Оценка числа итераций и вычислительной сложности итерационных методов

Вычислительные затраты при решении систем ЛАУ с использованием итерационных методов определяются двумя основными факторами. Во-первых, это число итераций, требуемых для достижения заданной точности, а во-вторых – вычислительные затраты на реализацию одной итерации. Именно эти две характеристики позволяет определять и сравнивать эффективность различных итерационных методов. В ряде случаев удается получить вполне реалистичные априорные теоретические оценки скорости сходимости.

Точное решение задачи неизвестно, поэтому для оценки погрешности текущего итерационного приближения используется невязка приближенного решения, которая связана с ошибкой следующим соотношением

$$r^{(k)} = Ax^{(k)} - f = Ax^{(k)} - f - (Ax - f) = A\delta^{(k)} \;\Rightarrow\; \delta^{(k)} = A^{-1}r^{(k)},$$

откуда следует оценка

$$\| \overset{k}{\delta} \| \le \| A^{-1} \| \cdot \| \overset{k}{r} \|.$$

Поскольку при сходимости итерационного процесса норма погрешности убывает пропорционально невязке, то в качестве критерия остановки итераций традиционно используется одно из условий:

$$\| r^{(k)} \| \le \varepsilon, \text{ или } \| r^{(k)} \| / \| f \| \le \varepsilon, \tag{1.42}$$

где ε – некоторая малая величина, определяющая требования к точности искомого приближенного решения. Количество итераций для достижения заданной точности можно оценить, зная норму матрицы итерационного процесса. Если положить, что при нулевом начальном приближении относительная погрешность равна 100% (это всегда имеет место при нулевом начальном приближении), тогда, например, для достижения относительно погрешности

$$\| \delta^{(k)} \| / \| x \| = \| A^{-1}r^{(k)} \| / \| A^{-1} f \| \approx \| r^{(k)} \| / \| f \|$$

с учетом оценки (1.41) потребуется число итераций k, такое, что

$$\| S \|^k \le \varepsilon \quad \Rightarrow \quad k \ge \frac{\log \varepsilon}{\log \| S \|}. \tag{1.43}$$

Норма матрицы итерационного процесса $\| S \|$ характеризует скорость его сходимости. Максимальная скорость сходимости достигается при минимальном значении $\| S \|$. Оценка (1.43) показывает, что минимальное число итераций для достижения заданной точности может неограниченно возрастать при $\| S \| \to 1$.

Для оценки числа арифметических операций на одной итерации прежде всего требуется учитывать конкретный вид матриц A и B. В большинстве случаев целесообразность в использовании итерационных методов ассоциируется с большой размерностью и разреженностью матрицы A. Введем понятие к*оэффициента заполнения разреженной матрицы,* численно равного отношению количества ненулевых элементов к их общему числу. Например, коэффициент заполнения диагональной матрицы D: $M_D = N / N^2 = N^{-1}$, N – размерность матрицы.

Положим для начала, что $B = E$. В этом случае одна итерация согласно (1.40) включает одно умножение матрицы A на вектор,

$Q(Ax) = 2N(N-1)M_A$, и сложение трех векторов, $Q(a+b+c) = 3N$. Если $M_A \cong 1$, то каждая итерация (1.40) требует порядка $Q(N^2)$ арифметических операций. При количестве итераций $k = O(N)$ вычислительная сложность итерационного процесса оценивается числом арифметических операций порядка $(2N(N-1)M_A + 3N)k = O(N^3)$, т.е. сопоставима с вычислительной сложностью метода Гаусса.

Заметим, что в случае разреженной матрицы вычислительная сложность итерационного метода строго пропорциональна коэффициенту заполнения, а для метода Гаусса, как правило, вычислительные затраты растут быстрее, чем заполнение матрицы. Полученные оценки показывают, что преимущества итерационного метода по сравнению с методом Гаусса будут проявляться наиболее заметно при $M_A << 1$, $k << N$.

С другой стороны, целесообразность использования переобуславливателя $B \neq E$ связана с возможностью уменьшения числа итераций для достижения заданной точности решения (переобуславливатель – единственный свободный параметр итерационного процесса (2.39), выбор которого способен влиять на норму матрицы итерационного процесса). Вычислительная сложность отдельной итерации при этом возрастает ровно настолько, сколько требуется для решения системы уравнений вида $Bx = w$. Таким образом, использование переобуславливателя способно реально снизить вычислительные затраты на поиск приближенного решения, если обеспеченное им относительное уменьшение числа итераций превосходит относительный рост вычислительной сложности отдельной итерации.

Упражнения:

1. С помощью функции normest оценить норму матрицы $S = E - \tau A, \tau = 1/\|A\|$ для метода простой итерации решения системы ЛАУ с матрицей Пуассона (см. упражнение в предыдущем параграфе). Для формирования единичной матрицы используйте функцию speye. Оцените число итераций для достижения заданной точности $\varepsilon = 10^{-5}$. Сравнить полученную оценку с реальным числом итераций при решении данной системы ЛАУ.
2. При решении системы ЛАУ 1024x1024 методом простых итераций за 50 итераций достигается относительная погрешность приближенного решения 1.e-3. Сколько итераций потребуется для достижения точности 1.e-6 (считать скорость сходимости итераций равномерной).

1.15.Выбор оптимальных итерационных параметров. Метод минимальных невязок.

Вычисление оптимального значения итерационного параметра при решении систем линейных алгебраических уравнений методом простой итерации требует знания спектра матрицы системы. Определение границ спектра матрицы – отдельная непростая задача. Рассмотрим один способ оптимизации итерационного параметра, для которого не требуется симметричность матрицы системы и знание границ её спектра.

Для решения системы ЛАУ (1.35) с положительно определенной матрицей A используем итерационный метод

$$x^{(k+1)} = x^{(k)} - \tau_{k+1}(Ax^{(k)} - f). \qquad (1.44)$$

В отличие от метода простой итерации (1.40) в итерационном процессе (1.44) используется переменный итерационный параметр τ_{k+1}. Определим, каким должно быть значение итерационного параметра τ_{k+1}, чтобы некоторая норма погрешности $\| x^{(k+1)} \| = \| x - x^{(k)} \|$ для очередного итерационного приближения имела минимальное значение.

Для погрешности итерационного метода (1.44) имеем

$$\delta^{(k+1)} = (E - \tau_{k+1}A)\delta^{(k)}. \qquad (1.45)$$

Умножим скалярно уравнение (1.45) само на себя, предварительно умножив его слева на матрицу A:

$$\left(A\delta^{(k+1)}, A\delta^{(k+1)}\right) = \| A\delta^{(k+1)} \|^2 = \left((A - \tau_{k+1}AA)\delta^{(k)}, (A - \tau_{k+1}AA)\delta^{(k)}\right) =$$
$$= \left(A\delta^{(k+1)}, A\delta^{(k+1)}\right) + \tau_{k+1}^2\left(AA\delta^{(k)}, AA\delta^{(k)}\right) - 2\tau_{k+1}\left(AA\delta^{(k)}, A\delta^{(k)}\right).$$

Условие минимума нормы погрешности определим из равенства нулю ее производной:

$$\frac{d}{d\tau_{k+1}} \| A\delta^{(k+1)} \| = 2\tau_{k+1}(AA\delta^{(k)}, AA\delta^{(k)}) - 2(AA\delta^{(k)}, A\delta^{(k)}) = 0.$$

Из последнего равенства имеем

$$\tau_{k+1} = \frac{(AA\delta^{(k)}, A\delta^{(k)})}{(AA\delta^{(k)}, AA\delta^{(k)})}.$$

Учитывая, что $\delta^{(k)} = A^{-1}r^{(k)}$, получаем выражение для оптимального значения итерационного параметра

$$\tau_{k+1} = \frac{(Ar^{(k)}, r^{(k)})}{(Ar^{(k)}, Ar^{(k)})}, \; r^{(k)} = Ax^{(k)} - f. \qquad (1.46)$$

Таким образом, мы приходим к итерационному методу с оптимальным выбором итерационного параметра, обеспечивающего на каждой итерации минимальное значение нормы погрешности $\| A\delta^{(k)} \|$:

$$x^{(k+1)} = x^{(k)} - \frac{(Ar^{(k)}, r^{(k)})}{(Ar^{(k)}, Ar^{(k)})} r^{(k)}. \qquad (1.47)$$

Итерационный метод (1.47) называется метод *минимальных невязок*, поскольку $\| A\delta^{(k)} \| = \| r^{(k)} \|$, и каждая итерация (1.47) соответствует нахождению следующего приближения, минимизирующего норму невязки $\| r^{(k+1)} \|$. Скорость сходимости метода минимальных невязок (1.47) сопоставима со скоростью сходимости метода простой итерации с оптимальным параметром, однако, в отличие от последнего в данном случае не требуется симметричности матрицы системы и определения границ ее спектра.

Задача выбора оптимального набора итерационных параметров, обеспечивающих минимизацию погрешности решения после k итераций, позволяет добиться лучших результатов в ускорении сходимости, нежели рассмотренная выше оптимизация итерационного параметра на каждом итерационном шаге в отдельности. Пусть требуется определить набор итерационных параметров $\tau_1, \tau_2, \cdots \tau_k$, минимизирующих норму матрицы

$$S_k = (E - \tau_k A)(E - \tau_{k-1} A) \ldots (E - \tau_1 A).$$

Если матрица A симметричная, $\gamma_1 E \le A \le \gamma_2 E$, то матрица S_k также симметрична и ее норма численно равна ее спектральному радиусу. Задача минимизации матричного полинома S_k эквивалентна минимизации многочлена

$$P_k(\lambda) = (1 - \tau_k \lambda)(1 - \tau_{k-1} \lambda) \ldots (1 - \tau_1 \lambda), \quad \gamma_1 \le \lambda \le \gamma_2.$$

Поскольку $P_k(0) = 1$, то искомый многочлен совпадает с многочленом Чебышева

$$P_k(\lambda) = H_k(\lambda) = (-1)^k q_k \cos\left(k \arccos \frac{2\lambda - (\gamma_1 + \gamma_2)}{\gamma_2 - \gamma_1} \right),$$

и его корни

$$\lambda = \lambda_m = \tau_m^{-1} = \frac{\gamma_1 + \gamma_2}{2} + \frac{\gamma_1 + \gamma_2}{2} \cos \frac{(2m-1)\pi}{2k}, \quad m = 1,2,\ldots k.$$

Здесь $q_k = \dfrac{2q^k}{1+q^{2k}}$, $q = \dfrac{1-\sqrt{K_A^{-1}}}{1+\sqrt{K_A^{-1}}}$, $K_A = \dfrac{\gamma_2}{\gamma_1}$ – число обусловленности матрицы A. Искомый набор итерационных параметров

$$\tau_m = \frac{2}{\gamma_1+\gamma_2}\frac{1}{1+q_0 t_m}, \quad q_0 = \frac{1-K_A^{-1}}{1+K_A^{-1}}, \quad t_m = \cos\frac{(2m-1)\pi}{2k}. \qquad (1.48)$$

Средняя скорость сходимости итерационного метода (1.44) с набором итерационных параметров (1.48) (данный метод обычно называют *явный итерационный метод с чебышевским набором итерационных параметров*) будет определяться казателем геометрической прогрессии

$$q = \frac{1-\sqrt{K_A^{-1}}}{1+\sqrt{K_A^{-1}}}.$$

Зависимость скорости сходимости итерационного метода (1.44) с оптимальным набором параметров (1.48) от числа обусловленности матрицы решаемой системы существенно ниже, чем в методе минимальных невязок. Однако, для вычисления оптимального набора итерационных параметров (1.48) требуется определить точные границы спектра матрицы A. Кроме того, для обеспечения вычислительной устойчивости такого итерационного метода требуется упорядочение параметров. В силу отмеченных сложностей на практике более широкое распространение получили методы *сопряженных градиентов*, обеспечивающие сравнимую скорость сходимости, но не требующие для своего использования границ спектра матрицы системы.

Упражнение:

1. С помощью численных экспериментов сравнить число итераций для достижения заданной точности в методе простой итерации с оптимальным параметром и методе минимальных невязок при решении систем ЛАУ с матрицей Пуассона. Для расчета оптимального значения итерационного параметра $\tau = \tau_0 = 2/(\gamma_1+\gamma_2)$ (см. условие теоремы 4.2) спектр матрицы Пуассона вычислить с помощью функции eig и определить его границы с помощью функций min, max. Используйте в качестве примера реализации метода минимальных невязок программу из приложения 4.2.

1.16.Градиентные методы. Методы наискорейшего спуска.

В случае симметричной положительно-определенной матрицы A задача поиска решения системы линейных алгебраических уравнений

$$Ax = f \tag{1.49}$$

эквивалентна задаче минимизации функционала (квадратичной формы)

$$F(x) = \frac{1}{2} x^T Ax - f^T x + c. \tag{1.50}$$

Использование данной эквивалентности лежит в основе так называемых *градиентных* итерационных методов для систем ЛАУ с положительно определенной симметричной матрицей. Наиболее распространенные из градиентных методов являются методы *наискорейшего спуска* и метод *сопряженных градиентов*.

Несложно вычислить градиент квадратичной формы и убедиться, что при $A = A^T$ для любого вектора x градиент совпадает с невязкой системы:

$$\text{grad}(F(x)) = \frac{1}{2} Ax + \frac{1}{2} A^T x - f = Ax - f = r.$$

Идея итерационного метода наискорейшего спуска состоит в том, что, стартуя с произвольного начального приближения $x = x^{(0)}$, для достижения минимума квадратичной формы (1.50) нам следует двигаться в направлении наискорейшего убывания данного функционала. Отсюда следует *структура метода наискорейшего спуска*

$$x^{(n+1)} = x^{(n)} - \tau_{n+1} r^{(n)} \tag{1.51}$$

т.е. каждое новое приближение находится как текущее значение вектора $x^{(n)}$ плюс некоторый вектор $-\tau_{n+1} r^{(n)}$, имеющий *направление противоположное градиенту* квадратичной формы в текущей точке, т.е. направление, в котором происходит *наискорейшее убывание минимизируемого функционала*. Таким образом, градиент функционала позволяет задать *направление поиска решения* на каждой текущей итерации.

Отметим, что выбранное нами направление поиска, вообще говоря, не совпадает с направлением на глобальный минимум функционала. Более того, двигаясь в выбранном направлении поиска, мы можем обнаружить вначале убывание функционала, а затем его возрастание. Определим стратегию поиска минимума функционала состоящую в том, чтобы движение в выбранном направлении поиска продолжалось до тех пор, пока *не будет достигнут минимум функционала* вдоль выбранного

направления поиска. Отсюда величина шага τ_{n+1} определяется исходя из условия минимума функционала при новом значении $x = x^{(n+1)}$.

Для вычисления величины шага определим условие минимума функционала в направлении, противоположном вектору невязки $r^{(n)}$:

$$\frac{dF(x^{(n+1)})}{d\tau} = \text{grad}(F(x^{(n+1)}))^T \frac{d}{d\tau} x^{(n+1)} = r^{T(n+1)} r^{(n+1)} = (r_{n+1}, r_n) = 0.$$

Таким образом, величина шага τ_{n+1} должна быть такой, чтобы новая невязка $r^{(n+1)}$ была ортогональна невязке на предыдущем итерационном приближении. Для вычисления τ_{n+1} умножим слева уравнение (1.51) на A и вычтем из полученного равенства f.

$$Ax^{(n+1)} - f = Ax^{(n)} - f - \tau_{n+1} Ar^{(n)} \quad \Rightarrow \quad r^{(n+1)} = r^{(n)} - \tau_{n+1} Ar^{(n)}$$

Последнее равенство умножим скалярно на r_n, в результате чего, учитывая требование ортогональности r_n и r_{n+1}, имеем

$$(r^{(n)}, r^{(n)}) - \tau_{n+1}(Ar^{(n)}, r^{(n)}) = 0 \quad \Rightarrow \quad \tau_{n+1} = \frac{(r^{(n)}, r^{(n)})}{(Ar^{(n)}, r^{(n)})}. \qquad (1.52)$$

Метод наискорейшего спуска (1.51), (1.52) и по структуре и по скорости сходимости аналогичен методу минимальных невязок (1.47).

Упражнения:

1. По аналогии с упражнением предыдущего параграфа сравните число итераций для достижения заданной точности в методе минимальных невязок и наискорейшего спуска при решении систем ЛАУ с матрицей Пуассона. (см. приложение 4.2.).
2. Проверьте гипотезу о том, что метод наискорейшего спуска сойдется за одну итерацию, если в качестве начального приближения использовать один из собственных векторов матрицы системы уравнений. Для вычисления собственного вектора используйте функцию eig.

1.17.Методы сопряженных градиентов.

Выбор направления для минимизации функционала в методе наискорейшего спуска представляется вполне разумным, но вряд ли оптимальным. Наблюдая за траекторией сходимости решения системы двух уравнений (см. Рис. 1.2.) можно заметить, что каждое новое приближение (за исключением некоторых частных случаев выбора начального приближения) находится в направлении, которое не совпадает с направлением на точку глобального минимума функционала. Более того,

в методе наискорейшего спуска одно и то же направление поиска может использоваться на нескольких итерациях.

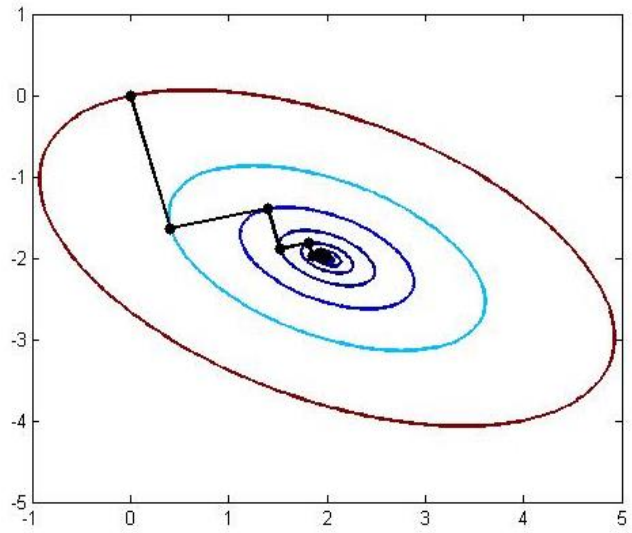

Рис. 1.2. Траектория сходимости итерационных приближений по методу наискорейшего спуска на фоне изолиний минимизируемого функционала для системы двух линейных алгебраических уравнений.

Среди градиентных итерационных методов наибольшее распространение получил метод *сопряженных градиентов (Conjugate Gradients)*. Его отличительная особенность состоит в том, что теоретически он позволяет получить *точное решение* за фиксированное число итераций, не превосходящее размерности матрицы (т.е. он аналогичен прямым методам решения СЛАУ).

Структура метода сопряженных градиентов (CG) во многом напоминает метод наискорейшего спуска. В частности, первая итерация CG метода выполняется по тому же алгоритму, что и для метода скорейшего спуска

$$r^{(0)} = Ar^{(0)} - f; \tag{1.53}$$

$$\sigma_0 = (r^{(0)}, r^{(0)}); \tag{1.54}$$

$$\tau_1 = \sigma_0 / (Ar^{(0)}, r^{(0)}); \tag{1.55}$$

$$x^{(1)} = x^{(0)} - \tau_1 r^{(0)}. \tag{1.56}$$

Далее, *стратегия выбора направления* для минимизации квадратичной формы меняется, и последующие итерации вычисляются по следующему алгоритму (вначале приведем структуру алгоритма, а затем рассмотрим идеи, положенные в основу данного метода):

$$r^{(k)} = Ax^{(k)} - f; \tag{1.57}$$

$$\sigma_k = (r^{(k)}, r^{(k)}); \tag{1.58}$$

$$\beta_{k+1} = \sigma_k / \sigma_{k-1}; \tag{1.59}$$

$$d^{(n+1)} = r^{(n)} + \beta_{n+1} d^{(n)}; \tag{1.60}$$

$$\tau_{k+1} = d^{(k)} / (d^{(k+1)}, Ad^{(k+1)}); \quad (1.61)$$

$$x^{(k+1)} = x^{(k)} - \tau_{n+1} d^{(k+1)}; \quad (1.62)$$

Итерации выполняются до тех пор, пока невязка $r^{(k)}$ для полученного приближения $x^{(k)}$ не станет меньше заданного малого числа ε.

Рассмотрим концептуальные соображения, положенные в основу метода сопряженных градиентов. Заметим, что направление поиска глобального минимума квадратичной формы определяется векторами $d^{(k)}$, которые конструируется по рекуррентным формулам (1.59) – (1.60) с учетом невязок задачи на двух предыдущих итерациях. *Идея, приводящая к методу сопряженных градиентов, состоит в нахождении такого набора линейно независимых направлений* $d^{(k)}$*, чтобы в каждом из этих направлений делать лишь один шаг*. Это возможно лишь в том случае, если на каждом итерационном приближении происходит *полное* исключение погрешности вдоль текущего направления поиска $d^{(k)}$.

Предположим, что нами выбрана совокупность направлений поиска $d^{(k)}$, $k = \overline{0, N-1}$, составляющих систему линейно независимых векторов. Пусть вектор погрешности начального приближения итерационно процесса $e^{(0)} = x^{(0)} - x$. Представим данный вектор в виде разложения по базису $d^{(k)}$, $k = \overline{0, N-1}$:

$$e^{(0)} = \sum_{k=0}^{N-1} \tau_k d^{(k)}. \quad (1.63)$$

Пусть $(d^{(k)}, Ad^{(m)}) = 0$, $k \neq m$, т.е. направления поиска *A-ортогональны*, где A – симметричная положительно определенная матрица рассматриваемой системы ЛАУ. Если потребовать, чтобы на текущей итерации полностью устранялась *k*-я компонента погрешности, то каждое новое приближение должно находиться по следующему алгоритму

$$x^{(k)} = x^{(k-1)} - \tau_k d^{(k)},$$

и тогда

$$e^{(k)} = \sum_{m=k+1}^{N-1} \tau_m d^{(m)}. \quad (1.64)$$

Умножая уравнение (1.64) слева на матрицу *A*, а затем, умножая полученное равенство скалярно на $d^{(k+1)}$, с учетом $Ae^{(k)} = e^{(k)}$ имеем:

$$(d^{(k+1)}, Ae^{(k)}) = (d^{(k+1)}, r^{(k)}) = \sum_{m=k+1}^{N} \tau_m (d^{(k+1)}, Ad^{(m+1)}) = \tau_{k+1} (d^{(k+1)}, Ad^{(k+1)}), \quad (1.65)$$

$$\tau_{k+1} = \frac{(d^{(k+1)}, r^{(k)})}{(d^{(k+1)}, Ad^{(k+1)})}. \quad (1.66)$$

Полученное выражение дает величину τ_{k+1} – шага текущего итерационного приближения, позволяющего полностью исключить погрешность искомого решения вдоль вектора $d^{(k+1)}$. Следовательно, для полноты алгоритма нам необходимо построить процедуру вычисления системы A-ортогональных векторов $d^{(k)}$, $k = \overline{0, N-1}$ для поиска минимума квадратичной формы.

С учетом A-ортогональности направлений поиска, для этого достаточно взять любой набор линейно независимых векторов и применить к нему алгоритм ортогонализации. В методе сопряженных градиентов для построение требуемой системы А-ортогональных векторов используется тот факт, что погрешность на $k+1$-й итерации А-ортогональна вектору поиска $d^{(k+1)}$ (поскольку $k+1$-я итерация исключает погрешность вдоль вектора поиска $d^{(k+1)}$, а остальные векторы $d^{(m>k+1)}$ в разложении (1.64) А-ортогональны направлению d_{k+1}):

$$(e^{(k+1)}, Ad^{(k+1)}) = (Ae^{(k+1)}, d^{(k+1)}) = (r^{(k+1)}, d^{(k+1)}) = 0. \quad (1.67)$$

Следовательно, невязки метода сопряженных градиентов образуют систему линейно-независимых векторов $r^{(k)}$, *ортогональных* направлениям поиска $d^{(k)}$. Кроме того, невязка $r^{(k)}$ ортогональна всем предыдущим направлениям поиска $d^{(m<k)}$. В самом деле

$$Ae^{(k+1)} = A(x^{(k+1)} - x) = A(x^{(k)} - \tau_{k+1} d^{(k+1)} - x) = r^{(k+1)} = r^{(k)} - \tau_{k+1} Ad^{(k+1)},$$

$$r^{(k)} = r^{(k-1)} - \tau_k Ad^{(k)}. \quad (1.68)$$

Умножая скалярно последнее равенство на $d^{(m<k)}$, с учетом А-ортогональности направлений поиска и тождества (1.67), имеем

$$(r^{(k)}, d^{(m<k)}) = 0. \quad (1.69)$$

Проведем процедуру А-ортогонализации базиса образуемого векторами невязок. В качестве первого вектора используем вектор невязки нулевого приближения

$$d^{(1)} = r^{(1)}.$$

Для получения следующего направления поиска, согласно стандартной процедуре ортогонализации, следует взять невязку $r^{(1)}$ и вычесть из неё составляющие, *которые не A-ортогональны вектору* $d^{(1)}$:

$$d^{(2)} = r^{(1)} - \beta_1 d^{(1)},\ (d^{(2)}, Ad^{(1)}) = (r^{(1)}, Ad^{(1)}) - \beta_1 (d^{(1)}, Ad^{(1)}),$$

$$\beta_1 = \frac{(r^{(1)}, Ad^{(1)})}{(d^{(1)}, Ad^{(1)})}.$$

Повторяя данную процедуру, приходим к формулам построения A-ортогональной системы векторов $d^{(k)}$:

$$d^{(k+1)} = r^{(k)} - \sum_{m=0}^{k} \beta_{m+1} d^{(m)}. \qquad (1.70)$$

Для определения коэффициентов β_m, $\overline{m = 1, k+1}$, заметим, что невязки образуют ортогональную систему. В самом деле, умножая (2.52) скалярно на $r^{(k+1)}$ с учетом (1.67), (1.69) имеем $(r^{(k)}, r^{(k+1)}) = 0$. Аналогично имеем

$$(r^{(k)}, r^{(m<k)}) = 0. \qquad (1.71)$$

Далее, умножая (1.70) скалярно на Ad_k:

$$(d^{(k+1)}, Ad^{(k)}) = (r^{(k)}, Ad^{(k)}) - \beta_{k+1} (d^{(k)}, Ad^{(k)}) = 0,$$

$$\beta_{k+1} = \frac{(r^{(k)}, Ad^{(k)})}{(d^{(k)}, Ad^{(k)})}. \qquad (1.72)$$

Умножая тождество (1.70) на $r^{(m>k)}$ с учетом (1.71) имеем

$$(r^{(m>k)}, Ad^{(k)}) = 0. \qquad (1.73)$$

При скалярном умножении (1.70) на $Ad^{(m)}$:

$$\forall m < k : (d^{(k+1)}, Ad^{(m)}) = (r^{(k)}, Ad^{(m)}) - \beta_{m+1} (d^{(m)}, Ad^{(m)}) = 0, \quad \Rightarrow \quad \beta_{m+1} = 0.$$

Последние тождества показывают, что в процедуре A-ортогонализации нет необходимости хранить и использовать все предыдущие направления поиска. Данное обстоятельство обеспечивает высокую эффективность алгоритма.

Выражение (1.72) для вычисления коэффициентов β_k можно упростить, используя выражение (1.68) , из которого при скалярном умножении на $r^{(k)}$ следует $(r^{(k)}, Ad^{(k)}) = -(r^{(k)}, r^{(k)}) / \tau_k$, и тогда

$$\beta_{k+1} = \frac{(r^{(k)}, Ad^{(k)})}{(d^{(k)}, Ad^{(k)})} = -\frac{(d^{(k)}, Ad^{(k)})}{(d^{(k)}, r^{(k-1)})} \frac{(r^{(k)}, r^{(k)})}{(d^{(k)}, Ad^{(k)})} = -\frac{(r^{(k)}, r^{(k)})}{(d^{(k)}, r^{(k-1)})}$$

С учетом того, что $(r^{(k)}, d^{(k)}) = 0$, $d^{(k)} = r^{(k-1)} - \beta_k d^{(k-1)}$, имеем

$$\beta_{k+1} = -\frac{(r^{(k)}, r^{(k)})}{(r^{(k-1)}, r^{(k-1)})} \text{ и } \tau_{k+1} = \frac{(r^{(k)}, r^{(k)})}{(d^{(k+1)}, Ad^{(k+1)})}$$

Таким образом, мы получили расчетные формулы метода сопряженных градиентов (1.47) – (1.52), использующего в качестве направлений поиска минимума функционала полную системы А-ортогональных векторов. При этом на каждом шаге итерационного метода невязка приближенного решения *ортогональна* направлению поиска, т.е. направления поиска являются сопряженными с вектором градиента в точке текущего итерационного приближения. Отсюда происходит название данного метода – *метод сопряженных градиентов*.

Изложенный формализм, приводящий к методу сопряженных градиентов, является не единственным. К аналогичным результатам приводит построение итерационного метода, обеспечивающего минимизацию ошибки в энергетической норме $\|\cdot\|_A$ на подпространствах Крылова, ассоциированных с матрицей A системы ЛАУ:

$$V_k = span\{r_0, Ar_0, A^2 r_0, \cdots, A^k r_0\}.$$

Скорость сходимости метода сопряженных градиентов не хуже, чем для метода с чебышевским набором оптимальных итерационных параметров (1.48), однако в отличие от последнего для него не требуется знания точных границ спектра матрицы системы. Благодаря высокой скорости сходимости и самодостаточности, на сегодняшний день метод сопряженных градиентов и его модификации является одними из наиболее популярных и совершенных итерационных методов.

Упражнения:

1. Изучите пример реализации итерационных методов в приложении 4.2. Выполните программу и проанализируйте результаты численного эксперимента. Используйте функцию pcg и сравните число итераций в методе сопряженных градиентов, реализованном в программе приложения 4.2 и функции pcg.
2. При решении системы ЛАУ 120x120 методом *сопряженных градиентов* за 50 итераций достигается относительная погрешность приближенного решения 1.e-3. Сколько итераций потребуется для достижения точности 1.e-9. (вычислительной погрешностью метода пренебречь).

3.7 Неявные итерационные методы. Переобуславливатель.

В случае плохо обусловленных матриц, $K_A >> 1$, сходимость итерационных методов вида (1.39) с оператором $B = \tau_k^{-1} E$ может оказаться очень медленной (см., например, оценку (1.41)). Выбор оптимальных итерационных параметров во многих практических задачах не способен обеспечить достижения желаемой скорости сходимости. Отмеченный недостаток присущ всем явным методам, включая метод сопряженных градиентов, метод минимальных невязок и др. Решение данной проблемы в большинстве случаев может быть получено с использованием неявных итерационных методов или итерационных методов с *переобуславливателем*.

Назначение *переобуславливателя* прочитывается в самом названии. Его функции состоят в том, чтобы от системы с плохо обусловленной матрицей перейти к решению системы с матрицей более приемлемой обусловленности. В самом деле, *неявный* итерационный метод вида

$$\tau_{k+1}^{-1} B(x^{(k+1)} - x^{(k)}) = Ax^{(k)} - f, \qquad (1.73)$$

эквивалентен *явному* итерационному методу

$$\tau_{k+1}^{-1}(x^{(k+1)} - x^{(k)}) = Dx^{(k)} - g, \qquad (1.74)$$

где $D = B^{-1}A, \quad g = B^{-1} f$. Основное функциональное назначение матрицы B в том, чтобы в итерационных процессах (1.73) и (1.74) достичь существенного уменьшения числа обусловленности матрицы $D = B^{-1}A$ по сравнению с числом обусловленности исходной матрицы A.

Второй важный момент при выборе предобусловливателя состоит в вычислительной эффективности – возможности вычисления матрицы B^{-1} (или решения системы $Bx = w$) *с вычислительными затратами существенно меньшими* по сравнению вычислительными затратами на обращение матрицы A (или решение системы $Ax = f$).

С точки зрения основного функционального назначения идеальным переобуславливателем является матрица $B = A$. Однако с точки зрения вычислительной эффективности такой переобуславливатель оказывается абсолютно бесполезным, поскольку, по сути, этот выбор возвращает нас снова к необходимости решения системы вида $Ax = f$.

Рассмотрим наиболее простые неявные итерационные методы.

Метод Якоби. $B = diag(a_{11}, a_{22}, \ldots, a_{NN})$. В качестве переобуславливателя используется диагональная матрица, элементы которой совпадают с диагональными элементами матрицы A. Выбор диагонального пере-

обуславливателя практически не увеличивает вычислительную сложность отдельной итерации по сравнению с явным методом. Данный метод может оказаться полезным для разреженных матриц с диагональным преобладанием в случае, когда диагональные элементы матрицы A существенно отличаются друг от друга. Такие матрицы возникают, например, при дискретизации многомерных уравнений математической физики с сильно неоднородными коэффициентами. Если диагональные элементы матрицы A одинаковы (или почти совпадают), то метод Якоби не имеет преимуществ по сравнению с явным методом.

Метод Зейделя (Гаусса-Зейделя). Матрица B имеет треугольный вид и строится непосредственно из соответствующих элементов матрицы A. В силу треугольности матрицы B, данный метод имеет небольшой рост вычислительных затрат на одну итерацию и примерно вдвое (в отдельных случаях и более чем вдвое) сокращает число итераций для достижения заданной точности по сравнению с явным методом. Использование данного метода практически всегда имеет положительный эффект по сравнению с явным методом и методом Якоби. Большего эффекта можно достичь если чередовать на четных и нечетных итерациях соответственно нижние и верхние треугольные матрицы (данная модификация называется *симметричным методом Гаусса-Зейделя* или *попеременно треугольным методом*).

Метод последовательной верхней релаксации. В некотором роде является обобщением метода Зейделя и метода Якоби. Переобуславливатель строится из верхней треугольной части матрицы A без главной диагонали $\overline{R} = \{A\}_{k<m}$ и диагональных элементов матрицы

$$D = diag(a_{11}, a_{22}, \ldots, a_{NN}):$$

$$B = D + \omega\overline{R},\ \tau_k = \omega.$$

Рассмотрим вопрос о сходимости метода последовательной верхней релаксации

$$Bx^{(k+1)} = Bx^{(k)} - \omega(Ax^{(k)} - f). \qquad (1.75)$$

Теорема 2.5. *В случае симметричной положительной матрицы* $A = A^T > 0$ *итерационный метод последовательной верхней релаксации (1.75) сходится при* $0 < \omega < 2$.

Доказательство. По аналогии с доказательством сходимости явного метода простой итерации, условие сходимости неявного метода (1.75) может быть сформулировано в виде $2B - \omega A = 2(D + \omega\overline{R}) - \omega A > 0$, т.е. условие сходимости равносильно положительной определенности мат-

рицы $2(D+\omega\overline{R})-\omega A$, или $((2(D+\omega\overline{R})-\omega A)x,x)>0$. Учитывая, что $A=\overline{R}+D+\overline{R}^T$, имеем

$$(2(D+\omega\overline{R})-\omega A)x,x)=(2(D+\omega\overline{R})-\omega(\overline{R}+D+\overline{R}^T))x,x)=((2-\omega)Dx,x).$$

Поскольку из положительной определенности матрицы A следует положительная определенность матрицы $D>0$, то положительность матрицы $(2-\omega)D>0$ имеет место в интервале $0<\omega<2$, что и требовалось доказать.

При $\omega=1$ метод последовательной верхней релаксации совпадает с методом Гаусса-Зейделя, при $\omega=0$ – метод совпадает с методом Якоби. Оптимальное значение параметра ω обычно лежит в интервале $1.5<\omega<1.8$. В большинстве случаев метод последовательной верхней релаксации превосходит по эффективности методы Якоби и Зейделя. Метод весьма популярен для решения систем линейных алгебраических уравнений, возникающих при разностной аппроксимации многомерных задач математической физики. Существуют модификации данного метода, основанные на чередовании верхней и нижней треугольных матриц в переобуславливателе B.

В последние годы широкое распространение получили неявные итерационные методы, основанные на аддитивном представлении и приближенной факторизации матрицы задачи при формировании переобуславливателей. Данная техника может успешно использоваться в итерационных методах минимальных невязок, наискорейшего спуска и сопряженных градиентов. Например, неявный аналог метода минимальных невязок может быть получен из формул явного метода (1.47), в котором системная матрица A заменяется на матрицу $B^{-1}A$:

$$r^{(k)}=Ax^{(k)}-f,\ Bw^{(k)}=r^{(k)},\ x^{(k+1)}=x^{(k)}-\tau_{k+1}w^{(k)},$$
$$\tau_{k+1}=(Aw^{(k)},w^{(k)})/(Aw^{(k)},Aw^{(k)}).$$

Упражнения:

1. Оценить во сколько раз число итераций для достижения заданной точности метода простой итерации будет превосходить число итераций метода с оптимальным набором чебышевских параметров если число обусловленности матрицы системы уравнений $K_A=100;10000$.
2. При каких начальных приближениях итерационный метод наискорейшего спуска сойдется за одну итерацию?

2. ВЫЧИСЛЕНИЕ СОБСТВЕННЫХ ЗНАЧЕНИЙ И СОБСТВЕННЫХ ВЕКТОРОВ МАТРИЦ

2.1. Свойства собственных значений и собственных векторов. Преобразование подобия

Пусть дана квадратная невырожденная матрица $A \in R^{N \times N}$. Число λ называется *собственным значением матрицы* A, если существует такой ненулевой вектор x, удовлетворяющий равенству

$$Ax = \lambda x. \qquad (2.1)$$

Вектор $x \neq 0$, удовлетворяющий равенству (2.1), называется *собственным вектором матрицы* A. Совокупность всех собственных значений называется *спектром матрицы*. Уравнение (2.1) имеет нетривиальные решения тогда и только тогда, когда

$$\det(A - E\lambda) = 0. \qquad (2.2)$$

Функция $P(\lambda) = \det(A - E\lambda) = \lambda^N + a_1\lambda^{N-1} + a_2\lambda^{N-2} + \ldots + a_{N-1}\lambda + a_N$ называется *характеристическим многочленом* матрицы. Степень характеристического многочлена равна порядку матрицы, а множество его корней совпадает со спектром. Таким образом, задача поиска собственных значений может быть сведена к задаче вычисления корней характеристического многочлена. Число собственных значений матрицы, с учетом возможной кратности корней характеристического полинома, равно порядку матрицы.

Спектр действительной матрицы составляют действительные или комплексные числа, образующие комплексно-сопряженные пары. Собственные вектора матрицы определяются с точностью до постоянного множителя. Обычно используют нормированные значения собственных векторов: $x = x / \| x \|$. Отметим важные свойства собственных векторов и собственных значений.

1. Собственными значениями положительно определенной симметричной матрицы являются действительные положительные числа, причем среди них отсутствуют кратные:

$$\forall A = A^T \in R^{N \times N} : (Ax, x) \geq \delta > 0 \Rightarrow \lambda_k \in R,\ k = \overline{1, N},\ 0 < \lambda_1 < \lambda_2 < \ldots < \lambda_N$$

.

2. Множество собственных векторов симметричной невырожденной матрицы $A \in R^{N \times N}$ образуют ортогональный базис в пространстве R^N.

3. Если λ – собственное значение матрицы A, то $\lambda - \alpha$ – собственное значение матрицы $A - \alpha E$.

4. Собственные значения диагональной и треугольной матрицы совпадают с элементами главной диагонали данной матрицы.

5. Если λ, и $x-$ собственное значение и собственный вектор матрицы A, тогда λ^{-1}и x – собственное значение и собственный вектор матрицы A^{-1} соответственно.

6. Для произвольной невырожденной матрицы $P \in R^{NxN}$, собственные матрицы $A \in R^{NxN}$ и матрицы $B = PAP^{-1}$ – совпадают. Матрицы A и B в этом случае называются *подобными*, а преобразования вида $B = PAP^{-1}$ называются *преобразованиями подобия*.

7. $\forall A, B \in R^{NxN}$ собственные значения матриц ABи BA совпадают.

Перечисленные свойства играют важную роль при разработке численных методов решения проблемы собственных значений и собственных векторов. Важно отметить также класс матриц, для которых задача отыскания собственных значений и собственных векторов имеет наиболее простое решение. Это так называемые матрицы *простой структуры*.

Определение. *Если существует невырожденная матрица P такая, что матрица $D = PAP^{-1}$ является диагональной, то матрица A называется матрицей простой структуры.*

Иными словами, матрица простой структуры подобна диагональной матрице, элементами которой являются собственные значения исходной матрицы.

Для решения проблемы собственных значений, как правило, используются преобразования подобия с использованием ортогональных матриц, для которых обратная матрица совпадает с транспонированной. Используются также треугольные матрицы специального вида, для которых обратная матрица может быть выписана в явном виде без дополнительных вычислений.

Теорема 2.1. *Матрица $A \in R^{NxN}$ подобна диагональной матрице в том и только в том случае, если она имеет систему N линейно независимых векторов.*

Доказательство. Докажем вначале, что если все собственные векторы матрицы являются линейно независимыми, то данная матрица – матрица простой структуры. Пусть x_k, λ_k собственные векторы и собственные значения матрицы A. Пусть матрица $P = [x_1, x_2, \ldots, x_N]$ – матрица, столбцами которой являются собственные векторы матрицы A. По

определению $Ax_k = \lambda_k x_k$, $k = \overline{1, N}$. Последнее равенство для всех $k = \overline{1, N}$ может быть представлено в эквивалентном виде

$$AP = DP = PD, \tag{2.3}$$

где $D = diag(\lambda_1, \lambda_2, \ldots, \lambda_N)$ – диагональная матрица, элементами которой являются собственные значения. По условию теоремы столбцы матрицы P линейно независимы. В силу этого матрица P является не вырожденной и существует обратная матрица P^{-1}. Умножая равенство (2.3) слева на матрицу P^{-1}, имеем:

$$P^{-1}AP = P^{-1}PD = D, \tag{2.4}$$

Полученное тождество означает, что матрица A подобна диагональной матрице, что и требовалось доказать.

Докажем теперь обратное. Пусть матрица A подобна диагональной матрице D, элементами которой являются собственные значения матрицы A. Тогда выполняется равенство (2.4), умножая которое слева на матрицу P, получаем, что в этом случае выполняется и равенство (2.3) . Несложно заметить, что решением матричного уравнения $AP = DP$ является матрица P, столбцами которой являются собственные векторы матрицы A. *В* силу существования обратной матрицы P^{-1}, матрица P является не вырожденной, откуда следует линейная независимость ее строк и столбцов. Теорема доказана.

Теорема 2.2. *(теорема Гершгорина). Все собственные значения произвольной матрицы* $A \in R^{N \times N}$ *лежат на комплексной плоскости в объединении кругов* $\Lambda_k = \{z : |z - a_{kk}| \le R_k\}, k = \overline{1, N}, \quad R_k = \sum_{m \ne k} |a_{km}|$.

Круги Гершгорина Λ_k имеют центры в точках комплексной плоскости, соответствующих диагональным элементам матрицы, а их радиусы численно равны сумме модулей недиагональных элементов соответствующей строки.

Упражнения:

1. Доказать, что отношение подобия матриц является отношением эквивалентности.
2. Доказать, что спектральный радиус матрицы не превосходит любую ее норму.
3. Доказать, что если матрица имеет диагональное преобладание и диагональные элементы положительны, то данная матрица положительно определенная.
4. Доказать, что диагональные элементы положительно определенной матрицы положительны.

2.2. Каноническая форма Фробениуса. Метод Данилевского.

Метод Данилевского может быть использован для решения полной проблемы собственных значений и собственных векторов. Он основан на приведении матрицы к канонической форме Фробениуса с использованием преобразования подобия. Под канонической формой Фробениуса понимают матрицу вида

$$\Phi = \begin{bmatrix} p_1 & p_2 & & \cdots & p_n \\ 1 & 0 & & \cdots & 0 \\ 0 & 1 & & \cdots & 0 \\ \cdots & \cdots & \ddots & \cdots & \cdots \\ 0 & 0 & \cdots & 1 & 0 \end{bmatrix}.$$

Примечательно то, что элементы первой строки матрицы Фробениуса совпадают с коэффициентами ее характеристического многочлена:

$$\det(\Phi - \lambda E) = (-1)^N \left(\lambda^N - p_1\lambda^{N-1} - \ldots - p_k\lambda^{N-k} - \ldots - p_N\right) = (-1)^N P_N(\lambda).$$

Таким образом, после приведения матрицы к форме Фробениуса с использованием преобразований подобия, задача вычисления собственных значений сводится к нахождению корней уравнения $P_N(\lambda) = 0$.

Приведение исходной матрицы A к каноническому виду Фробениуса осуществим построчно, начиная с последней строки. Для этого умножим матрицу A справа на матрицу

$$M_{N-1} = \begin{bmatrix} 1 & 0 & 0 & \cdots & 0 & 0 \\ 0 & 1 & 0 & & 0 & 0 \\ 0 & 0 & 1 & & 0 & 0 \\ \vdots & & & \ddots & & \vdots \\ -\dfrac{a_{N,1}}{a_{N,N-1}} & -\dfrac{a_{N,2}}{a_{N,N-1}} & -\dfrac{a_{N,3}}{a_{N,N-1}} & \cdots & \dfrac{1}{a_{N,N-1}} & -\dfrac{a_{N,N}}{a_{N,N-1}} \\ 0 & 0 & 0 & \cdots & 0 & 1 \end{bmatrix}$$

Заметим, что обратная матрица для M_{N-1} имеет вид

$$M_{N-1}^{-1}=\begin{bmatrix} 1 & 0 & 0 & \dots & 0 & 0 \\ 0 & 1 & 0 & & 0 & 0 \\ 0 & 0 & 1 & & 0 & 0 \\ \vdots & & & \ddots & & \vdots \\ a_{N,1} & a_{N,2} & a_{N,3} & \dots & a_{N,N-1} & a_{N,N} \\ 0 & 0 & 0 & \dots & 0 & 1 \end{bmatrix}.$$

В результате умножения $A^{(1)}=AM_{N-1}$ последняя строка матрицы $A^{(1)}$ принимает вид матрицы Фробениуса, а умножение $A^{(1)}=M_{N-1}^{-1}A^{(1)}$ не изменяет последней строки матрицы $A^{(1)}$. При этом собственные значения матрицы $A^{(1)}=M_{N-1}^{-1}\cdot A\cdot M_{N-1}$ совпадают с собственными значениями матрицы A. Выполняя для матрицы A цепочку преобразований подобия с использованием матрицам $M_{N-1}, M_{N-2},\dots$, приводим исходную матрицу к каноническому виду Фробениуса:

$$\begin{aligned} &A^{(1)}=M_{N-1}^{-1}\cdot A\cdot M_{N-1};\\ &A^{(2)}=M_{N-2}^{-1}\cdot A^{(1)}\cdot M_{N-2};\\ &\dots\dots\dots\dots\dots\dots\dots\dots\\ &A^{(N-1)}=\underbrace{M_1^{-1}\cdots M_{N-2}^{-1}M_{N-1}^{-1}}_{S^{-1}}\cdot A\cdot\underbrace{M_{N-1}M_{N-2}\cdots M_1}_{S}=S^{-1}\cdot A\cdot S=\Phi. \end{aligned} \qquad (2.5)$$

Матрицы M_{N-k}, $k=1,2,\dots,N-1$, конструируются из элементов матриц $A^{(k-1)}$, $A^{(0)}=A$, полученных на предыдущем шаге преобразований, путем замены в единичной матрице $(N-k)$-*й* строки на строку вида:

$$\left(-\frac{a_{N-k+1,1}^{(k-1)}}{a_{N-k+1,N-k}^{(k-1)}},-\frac{a_{N-k+1,2}^{(k-1)}}{a_{N-k+1,N-k}^{(k-1)}},\dots,-\frac{a_{N-k+1,N-k-1}^{(k-1)}}{a_{N-k+1,N-k}^{(k-1)}},\frac{1}{a_{N-k+1,N-k}^{(k-1)}},-\frac{a_{N-k+1,N-k+1}^{(k-1)}}{a_{N-k+1,N-k}^{(k-1)}},\dots-\frac{a_{N-k+1,N}^{(k-1)}}{a_{N-k+1,N-k}^{(k-1)}}\right)$$

Аналогичным образом конструируется матрица A_{N-k}^{-1} $(N-k)$-*я* строка которой имеет вид

$$\left(a_{N-k+1,1}^{(k-1)},a_{N-k+1,2}^{(k-1)},\dots,a_{N-k+1,N-k-1}^{(k-1)},a_{N-k+1,N-k}^{(k-1)},\dots,a_{N-k+1,N}^{(k-1)}\right).$$

Несложно убедиться, что на k-*ом* шаге преобразований в преобразуемой матрице все строки ниже $(N-k+1)$-*й* строки остаются неизменными.

Важно отметить, что собственные векторы матрицы Фробениуса находятся относительно просто. В самом деле, система уравнений для собственного вектора, соответствующего собственному значению λ, имеет вид

$$\begin{pmatrix} p_1 & p_2 & & \dots & p_N \\ 1 & 0 & & \dots & 0 \\ 0 & 1 & & \dots & 0 \\ \dots & \dots & \ddots & \dots & \dots \\ 0 & 0 & \dots & 1 & 0 \end{pmatrix} \cdot \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ \vdots \\ y_N \end{pmatrix} = \lambda \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ \vdots \\ y_N \end{pmatrix},$$

из которой следует рекуррентная зависимость между компонентами собственного вектора:

$$y_{k-1} = \lambda y_k, \quad k = N, N-1, N-2, \dots, 2. \tag{2.6}$$

Отсюда, в силу произвольности нормировки собственных векторов, можно положить $y_N = 1$ и вычислить остальные компоненты согласно (2.6):

$$y = (\lambda^{N-1}, \lambda^{N-2}, \dots, \lambda, 1)^T.$$

Поскольку

$$\Phi y = S^{-1} \cdot A \cdot Sy = \lambda y, \tag{2.7}$$

умножая (2.7) слева на матрицу S, имеем $A \cdot Sy = \lambda Sy \Rightarrow Ax = \lambda x,$ где $x = Sy$ – собственный вектор матрицы A.

Следует отметить, что с появлением более эффективных методов расчета собственных значений матрицы, метод Данилевского может быть использован для нахождения корней полинома.

2.3. Степенной метод.

Пусть требуется найти максимальное по абсолютной величине собственное значение матрицы $A \in R^{NxN}$, причем известно, что искомое собственное значение *простое действительное* (кратности единица). Предположим, что A – матрица простой структуры и

$$|\lambda_1| > |\lambda_2| \geq |\lambda_3| \geq \ldots \geq |\lambda_N|.$$

Заметим, что при умножении матрицы на ее собственный вектор последний преобразуется в коллинеарный вектор $Ax_k = \lambda_k x_k$, причем длина полученного при этом вектора изменяется пропорционально соответствующему собственному значению λ_k. Данное свойство собственных векторов лежит в основе *степенного метода*.

Для матриц простой структуры система собственных векторов образует базис в пространстве R^N, и любой вектор этого пространства может быть представлен в виде линейной комбинации векторов данного базиса:

$$y = \sum_{k=1}^{N} \alpha_k x_k \Rightarrow Ay = \sum_{k=1}^{N} \alpha_k \lambda_k x_k$$

Из последнего равенства следует, что в разложении по собственным векторам при умножении матрицы на вектор наибольший рост (наименьшее убывание) испытывает составляющая, соответствующая максимальному собственному значению. Рассмотрим последовательность

$$y^{(k)} = Ay^{(k-1)} = \lambda_1^k \left(\alpha_1 x_1 + \sum_{m=2}^{N} \alpha_m \left(\frac{\lambda_m}{\lambda_1} \right)^k x_m \right). \qquad (2.8)$$

Поскольку $\lambda_{m\neq 1} / \lambda_1 < 1$, то при $k \to \infty$ последовательность $y^{(k)}$ сходится к собственному вектору x_1. Компоненты вектора $y^{(k)}$, соответствующие другим собственным векторам стремятся к нулю со скоростью геометрической прогрессии. Очевидно, что скорость сходимости последовательности определяется отношением λ_2 / λ_1 – знаменателем геометрической прогрессии самой медленной из компонент $y^{(k)}$.

Заметим, что асимптотика нормы $\overset{k}{y}$ определяется также значением λ_1^k, которое в пределе $k \to \infty$ стремиться к нулю или бесконечности, в зависимости от величины λ_1. В силу этого для практического использования итерационного процесса (2.8) необходима нормировка промежуточных результатов. В качестве нормировочного коэффициента наиболее походящий выбор в использовании максимальной по абсолютной величине координаты вектора $y^{(k)}$. Алгоритм степенного метода следующий:

1. Формируем произвольный вектор $y^{(0)}$;
2. Вычисляем вектор $y^{(1)} = Ay^{(0)}$;

3. Находим отношение произвольных ненулевых компонент векторов $y^{(1)}$ и $y^{(0)}$

$$\lambda^{(1)} = y_m^{(1)} / y_m^{(0)}; \quad . \tag{2.9}$$

В качестве расчетной компоненты $y_m^{(1)}$ лучше всего брать максимальную по модулю компоненту: $| y_m^{(1)} | = || y^{(1)} ||_\infty$.

4. Производим нормировку вектора $\overset{1}{y}$: $y^{(1)} = y^{(1)} / \lambda^{(0)}$.
5. Повторяем процедуру п.п. 2, 3, 4, используя в качестве y вектор, полученный на предыдущем шаге п. 4.;
6. Повторение процедуры проводим до тех пор, пока различия меду двумя последовательными приближениями $\lambda^{(k+1)}$ и $\lambda^{(k)}$ не станут меньше некоторого заданного малого числа: $\left|\lambda^{(k)} - \lambda^{(k+1)}\right| \le \varepsilon$.

После того, как условие п.6 выполнено, итерационный процесс прекращается. Последнее полученное значение $\lambda^{(k)}$ является приближением искомого максимального по модулю собственного значения матрицы A, а вектор $y^{(k)}$ – собственный вектор, соответствующий данному собственному значению.

Таким образом, использование описанного выше метода позволяет определить приближенно как собственный вектор, соответствующий максимальному собственному значению, так и величину данного собственного значения:

$$x_1 = \lim_{k\to\infty} || Ay^{(k)} ||_\infty^{-1} Ay^{(k)}, \tag{2.10}$$

$$\lambda_1 = \lim_{k\to\infty} \lambda^{(k)}. \tag{2.11}$$

После того как наибольшее собственное значение определено, данный подход может быть использован для вычисление других собственных значений и собственных векторов. Например, для вычисления наименьшего собственного значения и соответствующего ему собственного вектора описанную выше процедуру следует применить к матрице $\tilde{A} = A - \lambda_1 E$. В соответствии со свойством сдвига собственных значений, собственные значения матриц $\tilde{A}$ и A связаны соотношением $\tilde{\lambda}_m = \lambda_m - \lambda_1$. В силу этого наибольшее по абсолютной величине соб-

ственное значение матрицы $\tilde{A}$: $\tilde{\lambda}_N = \lambda_N - \lambda_1$. Если собственное значение λ_N не кратное, то оно окажется максимальным по модулю собственным значением матрицы $\tilde{A}$ и может быть вычислено с помощью степенного метода, а вслед за ним и минимальное собственное значение $\lambda_N = \tilde{\lambda}_N + \lambda_1$ матрицы A.

Заметим, что сходимость степенного метода имеет скорость геометрической прогрессии со знаменателем $q = |\lambda_2| / |\lambda_1|$. Использование свойства сдвига собственных значений может оказаться очень полезным для ускорения сходимости степенного метода, когда собственные значения λ_1 и λ_2 близки по абсолютной величине.

Для случая симметричных матриц скорость сходимости степенного метода может быть существенно улучшена, если для определения приближенного значения максимального по модулю собственного числа использовать вместо (2.9) соотношение Релея

$$\lambda^{(k)} = \frac{(Ay^{(k)}, y^{(k)})}{(y^{(k)}, y^{(k)})}.$$

Однако, следует иметь в виду, что соотношение Релея дает лучшее по сравнению с (2.9) приближение для собственного значения, в то время как скорость сходимости собственного вектора остается прежней.

К недостаткам степенного метода следует отнести тот факт, что он не гарантирует результата в случаях, если матрица не является матрицей простой структуры.

Упражнения:

1. Реализуйте степенной метод вычисления максимального по модулю собственного значения и соответствующего собственного вектора матрицы. Примените его к вычислению максимального собственного значения и собственного вектора матрицы Пуассона. Сравните результаты с результатами, полученными с помощью функции eig.
2. С помощью численных экспериментов оцените, как изменяется максимальное собственное значение матрицы Пуассона при изменении размерности матрицы в четыре и шестнадцать раз. Заметь, что размерность матрицы Пуассона определяется квадратом значения параметра *N* функции: A=gallery('poisson',N).

2.4.Метод вращений.

Наиболее эффективный подход к проблеме собственных значений основан на использовании преобразований подобия, позволяющих привести исходную матрицу к треугольному, диагональному или блочно-диагональному виду. Поскольку преобразование подобия не меняет спектр матрицы, то применение такого рода преобразований во многих случаях приводит к решению *полной проблемы собственных значений.* Наиболее эффективны преобразования подобия в случае симметричных матриц. Однако во многих случаях достаточно предположить, что среди собственных значений матрицы отсутствуют кратные. В этом случае существует преобразование подобие, приводящее матрицу к диагональному виду.

Один из способов построения искомого преобразования подобия состоит в использовании элементарных матриц плоских вращений T_{km} :

$$T_{km} = \begin{bmatrix} 1 & 0 & \cdots & \cdots & \cdots & \cdots & 0 \\ 0 & \ddots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \cdots & \cdots & \cos\alpha & \cdots & -\sin\alpha & \cdots & \cdots \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ \cdots & \cdots & \sin\alpha & \cdots & \cos\alpha & \cdots & \cdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & 0 \\ 0 & \cdots & \cdots & \cdots & \cdots & 0 & 1 \end{bmatrix} \qquad (2.12)$$

Матрица T_{km} (для определенности пусть $k < m$) отличается от единичной матрицы только элементами $t_{kk} = t_{mm} = \cos\alpha$ и $t_{mk} = -t_{km} = \sin\alpha$. Несложно убедиться, что матрицы T_{km} являются ортогональными. В силу этого преобразования $A^{(1)} = T_{km}^T A T_{km}$, называемое преобразованием плоских вращений, или преобразованием Гивенса, является преобразованием подобия. При умножении произвольной матрицы слева (или справа) на матрицу T_{km} результирующая матрица отличается от исходной лишь элементами строк с номерами k и m (или столбцами с соответствующими номерами).

Полагая $A = A^T$, рассмотрим произведения:

$$B = AT_{km}: \begin{array}{l} b_{nk} = a_{nk}\cos\alpha + a_{nm}\sin\alpha, \\ b_{nm} = -a_{nk}\sin\alpha + a_{nm}\cos\alpha, \quad b_{ij} = a_{ij}, \quad j \neq k, m' \end{array}$$

$$C = T_{km}A: \begin{array}{l} c_{kn} = a_{kn}\cos\alpha + a_{mn}\sin\alpha, \\ c_{nm} = -a_{kn}\sin\alpha + a_{km}\cos\alpha, \quad c_{ij} = a_{ij}, \quad i \neq k, m. \end{array}$$

$A^{(1)} = T_{km}^T A T_{km}$:

$$a_{km}^{(1)} = b_{km}\cos\alpha + b_{mm}\sin\alpha =$$

$$= (-a_{kk}\sin\alpha + a_{km}\cos\alpha)\cos\alpha - (a_{mk}\sin\alpha - a_{mm}\cos\alpha)\sin\alpha =$$

$$= a_{km}(\cos^2\alpha - \sin^2\alpha) + (a_{mm} - a_{kk})\sin\alpha\cos\alpha =$$

$$= a_{km}\cos 2\alpha + \frac{1}{2}(a_{mm} - a_{kk})\sin 2\alpha.$$

Определим угол вращения таким образом, чтобы $a_{km}^{(1)} = 0$. С учетом последнего равенства данное условие приводит к выражению для угла α :

$$a_{km}\cos 2\alpha + \frac{1}{2}(a_{mm} - a_{kk})\sin 2\alpha. = 0 \quad \Rightarrow \quad \tan 2\alpha = \frac{2a_{km}}{a_{kk} - a_{mm}} = \omega.$$

Используя тригонометрические тождества, имеем:

$$s = \sin\alpha = sign(\omega)\sqrt{\frac{\sqrt{1+\omega^2} - 1}{2\sqrt{1+\omega^2}}},\ c = \cos\alpha = \sqrt{\frac{\sqrt{1+\omega^2} + 1}{2\sqrt{1+\omega^2}}}. \qquad (2.13)$$

При выбранном угле поворота преобразование $A^{(1)} = T_{km}^T A T_{km}$ в отношении симметричной матрицы A обладает замечательным свойством. В результате данного преобразования уменьшается общая сумма квадратов недиагональных элементов результирующей матрицы. Многократное применение такого рода преобразования с матрицами вращения T_{km} такими, что на текущем шаге $|a_{km}| = |a_{mk}| = \max_{i \neq j} |a_{ij}|$ приводит к сходимости последовательности матриц $A^{(n)}$, $n = 1,2,3,\ldots$ к матрице диагонального вида, при этом на диагонали результирующей матрицы будут находиться приближенные значения собственных чисел исходной матрицы A Пример реализации метода вращений в среде Matlab приведен в Приложении.

Упражнения:

1. Показать, что при умножении $B = AT_{km}$ имеет место тождество

$$b_{nm}^2 + b_{nk}^2 = a_{nm}^2 + a_{nk}^2.$$

2. Доказать, что $T_{km}(\alpha) = T_{km}^T(-\alpha)$.

2.5. Понятие о QR-алгоритме

Если говорить о наиболее успешных методах решения полной проблемы собственных значений, то нельзя не сказать о методах, основанных на QR-алгоритме. Схема QR-алгоритма выглядит следующим образом.

Вначале матрица представляется в виде произведения ортогональной матрицы Q ($Q^{-1} = Q^T$) и верхней треугольной матрицы R:

$$A = QR.$$

Затем вычисляется матрица

$$A_1 = RQ = Q^T QRQ = Q^T AQ.$$

Далее, повторяем разложение полученной матрицы на произведение ортогональной и правой треугольной с последующей перестановкой сомножителей. В итоге получаем последовательность подобных матриц A_k, $k = 1,2,\ldots$:

$$A_{k+1} = R_k Q_k = Q_k^T Q_k R_k Q_k = Q_k^T A_k Q_k.$$

В процессе описанных повторений возможно два сценария. Если исходная матрица A – матрица простой структуры, у которой отсутствуют кратные собственные значения, то поддиагональные элементы матрицы A_k стремятся к нулю и в пределе при больших k матрицы A_k стремятся к верхней треугольной матрице, на диагонали которой, в силу подобия всех матриц последовательности A_k, будут находиться собственные значения исходной матрицы A. Если же среди собственных значений матрицы A имеются кратные и (или) комплексные, то в пределе последовательности будет наблюдаться сходимость A_k к блочно-треугольному виду, причем размерность диагональных блоков определяется кратностью соответствующих им собственных значений. Комплексно-сопряженным парам собственных значений соответствуют блоки 2×2.

Для разложения матрицы на множители используем преобразование *отражения* или *Хаусхолдера*, которому соответствует ортогональная матрица

$$H = E - 2ww^T,$$

где w – произвольный вектор единичной длины, $w^T w = 1$. Умножение такой матрицы на произвольный вектор равносильно преобразованию отражения относительно плоскости, ортогональной вектору w.

Если в качестве вектора w взять нормированный вектор $w_1 = (a_1 + \|a_1\|e_1) / \|a_1 + \|a_1\|e_1\|$, где $e_1 = (1,0,0,\ldots 0)^T$, $a_1 = (a_{11}, a_{21}, \ldots, a_{N1})^T$ –

первый столбец матрицы A, то умножение матрицы *Хаусхолдера* $H_1 = E - 2w_1 w_1^T$ слева на матрицу A приведет к исключению в ее первом столбце поддиагональных элементов. Продолжая данный процесс с соответствующим образом сконструированными матрицами Хаусхолдера можно за N шагов привести произвольную матрицу к верхнему треугольному виду. Произведение же использованных в процессе преобразований матриц Хаусхолдера даст ортогональную матрицу Q в QR разложении.

Две важные модификации описанного алгоритма наиболее часто используются для повышения его эффективности. Во-первых, разложение матрицы на множители весьма трудоемкий процесс, вычислительная сложность которого порядка $O(N^3)$, причем такое разложение требуется проводить многократно. В связи с этим исходную матрицу вначале разумно с помощью ортогональных преобразований Хаусхолдера привести к форме Хессенберга, имеющей почти треугольный вид, где дополнительно одна нижняя диагональ отлична от нуля. Примечательно, что для матрицы Хессенберга QR разложение может быть выполнено с вычислительными затратами порядка $O(N^2)$, т.е. на порядок более экономично по сравнению с матрицей полного вида. Кроме того, вид матрицы Хессенберга сохраняется для всей последовательности матриц A_k в процессе выполнения QR алгоритма. Последнее означает, что приведение матрицы к виду Хессенберга нужно провести однократно перед началом QR алгоритма, и гарантировать тем самым получение на последующих итерациях существенной экономии вычислительных затрат.

Необходимость использования еще одной модификации алгоритма связана со сравнительно низкой скоростью сходимости в случае наличия в спектре матрицы близких по модулю собственных значений. Как правило, скорость убывания поддиагональных элементов характеризуется величиной $a_{nm}^{(k)} = O\left(\left|\lambda_m\right|^k \left|\lambda_n\right|^{-k}\right)$, и если матрица имеет близкие по модулю собственные значения, то сходимость алгоритма может быть очень медленной. В связи с этим существенное ускорение сходимости может быть получено при использовании QR алгоритма со сдвигом. Суть данной модификации состоит в том, что если есть хорошее приближение к собственному значению $\tilde{\lambda}$, то вместо матрицы A_k на очередном шаге используют матрицу $A_k - \tilde{\lambda}E$. Возникающий при этом сдвиг спектра матрицы позволяет существенно усилить отношения между собственными значениями матрицы, близкими по модулю к $\tilde{\lambda}$.

В качестве параметра сдвига $\tilde{\lambda}$ разумно использовать диагональные элементы матриц A_k. Например, пусть матрица имеет два максимальных по модулю собственные значения $\lambda_m = -\lambda_n \cong \tilde{\lambda} > 0$. Отношение их модулей равно единице и соответствующий поддиагональный элемент матрицы A_k, $a_{nm}^{(k)} = O\left(|\lambda_m|^{-k}|\lambda_n|^{k}\right)$, практически не будет стремиться к нулю при реализации QR алгоритма. Тем не менее, в процессе итераций мы можем взять в качестве параметра сдвига максимальный по модулю диагональный элемент A_k, который с высокой степенью вероятности обеспечит хорошее приближение к $\tilde{\lambda}$. Выполняя преобразование сдвига $\overline{A}_k = A_k - \tilde{\lambda}E$ мы получим, что соответствующие собственные значения матрицы $\overline{A}_k$ примут значения $\overline{\lambda}_m = \lambda_m + \tilde{\lambda} \cong 2\tilde{\lambda}$, $\overline{\lambda}_n = \lambda_n + \tilde{\lambda} \cong 0$. Теперь их отношение близко к нулю, что положительно скажется на динамике сходимости соответствующего матричного элемента, $\overline{a}_{nm}^{(k)} = O\left(\left|\overline{\lambda}_m\right|^{-k}\left|\overline{\lambda}_n\right|^{k}\right)$.

Упражнения:

1. Доказать, что произведение произвольного числа ортогональных матриц является ортогональной матрицей.
2. Показать, что матрица Хаусхолдера является ортогональной.
3. Построить матрицу Хаусхолдера, порожденную вектором $e_1 = (1,0,0,\ldots 0)^T$: $H = E - 2e_1e_1^T$. Чему будут равны вектора $He_1, He_2, \ldots$
4. Доказать, что преобразование Хаусхолдера сохраняет углы между векторами.

3. РЕШЕНИЕ НЕЛИНЕЙНЫХ УРАВНЕНИЙ И СИСТЕМ.

3.1.Отделение корней. Корни полиномов. Кратные корни.

Рассмотрим уравнение вида

$$f(x) = 0, \quad x \in [a,b], \tag{3.1}$$

где $f(x)$ – заданная функция, непрерывная на отрезке $[a,b]$. Задача состоит в отыскании нулей функции, т.е. значения переменной x, принадлежащих заданной области определения, при которой функция прини-

мает нулевое значение. Примером такой задачи моет служить задача определения корней полинома:

$$f(x) = P_n(x) = p_0 + p_1 x + p_2 x^2 + \ldots + p_n x^n .$$

Говоря о решении нелинейных уравнений, прежде всего, необходимо выяснить вопрос о существовании корней, их количестве, кратности и уточнить область их локализации, т.е. определить подобласти, в которых имеет место единственный корень уравнения – *отделить корни*.

Для решения нелинейных уравнений и систем традиционно используются итерационные методы. В отличие от итерационных методов для линейных систем в нелинейном случае имеет место две принципиальные особенности. Во-первых, в вопросе сходимости итерационных методов принципиальное значение приобретает выбор начального приближения. Во-вторых, решения задачи, вообще говоря, не единственно и наряду с вопросом сходимости итераций необходимо решение вопроса об их сходимости к определенному корню. Перечисленные особенности убеждают в необходимости рассматривать процедуру отделения корней как неотъемлемую составляющую любого из методов решения нелинейных уравнений.

Процедура отделения корней не имеет универсальных рецептов теоретического решения. Даже при отделении корней полинома мы располагаем весьма скромными возможностями, например, в решении вопроса о количестве действительных корней на заданном отрезке. Для этих целей можно воспользоваться *теоремой Штурма*, а выяснить вопрос о наличии кратных корней можно на основании наличия соответствующих корней у наибольшего общего делителя полинома и его производной.

На практике задача отделения корней, как правило, решается путем табличного или графического представления функции, причем графический способ во многом представляется более предпочтительным. В частности, современные средства компьютерной графики позволяют таким бесхитростным и весьма наглядным способом во многих случаях не только отделить, но и найти искомые корни с вполне приемлемой точностью.

В качестве примера на рис. 3.1 представлен график полинома $P_5(x) = x^5 - 2x^4 - x^3 + 2x^2$.

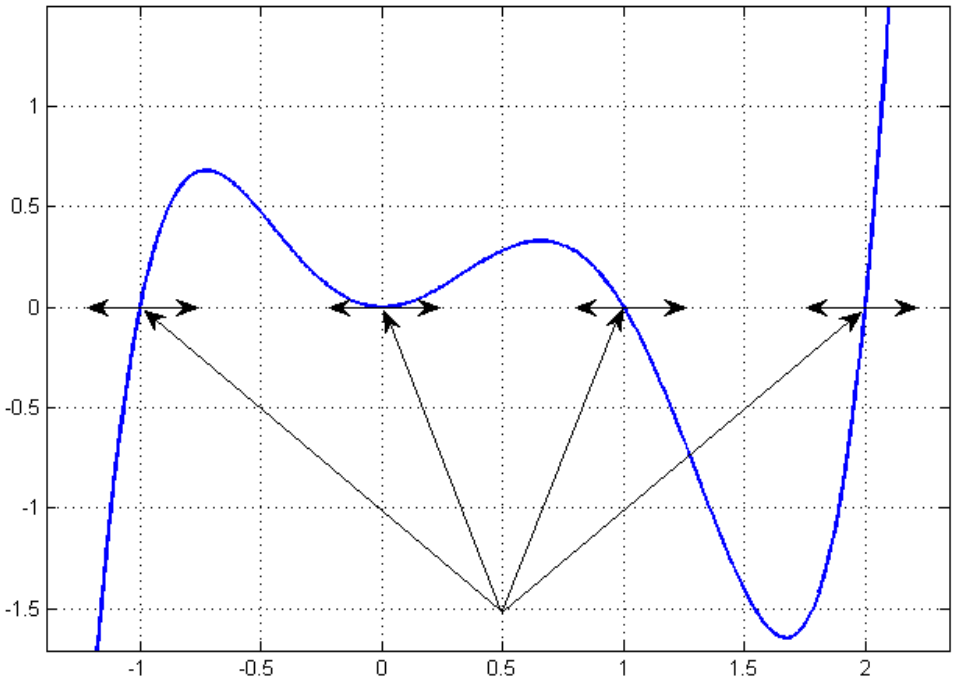

Рис. 3.1. Графический способ отделения корней.

При соответствующем ракурсе и акцентированном увеличении области локализации корней, несложно убедиться, что представленный на графике полином имеет корни в трех точках пересечения графика с осью *Ox*, $x = \{-1,1,2\}$, и кратный корень в точке касания графика с осью *Ox*, $x = 0$. Напомним, что корень уравнения $f(x) = 0$, $x = x^*$ имеет кратность *p*, если имеет место тождество $f^{(p-1)}(x^*) = 0$ или $f^{(p-1)}(x^*) = 0$, где $f^{(m)}(x^*)$ – производная порядка m функции $f(x)$ в точке $x = x^*$. Корни, кратность которых равна единице, принято называть *простыми*.

К недостаткам графического способа следует отнести необходимость "ручной" работы с графиком и отсутствие возможности автоматизировать процедуру отделения корней.

Если вычислить значения функции на множестве точек отрезка

$$\{x_m = a + \delta \cdot m, \quad m = \overline{0, M}, \delta = (b-a)/M\},$$

то в качестве критерия наличия корня нечетной кратности на элементарном отрезке $[x_k, x_{k-1}]$ может служить отрицательный знак или равенство нулю произведения значений функции на концах отрезка:

$f(x_k) \cdot f(x_{k-1}) \le 0$. Такой способ прекрасно алгоритмизируется, но его использование затруднительно для отделения корней четной кратности.
Для идентификации корней четной кратности можно провести поиск нулей первой производной функции. Нули производной функции потенциально могут быть кратными корнями самой функции. Следует также иметь в виду, что задача отыскания корней четной кратности по своей природе является неустойчивой в том смысле, что малые возмущения функции могут приводить либо к потере корня, либо к превращению кратного корня в пару простых корней (см. Рис. 3.2).

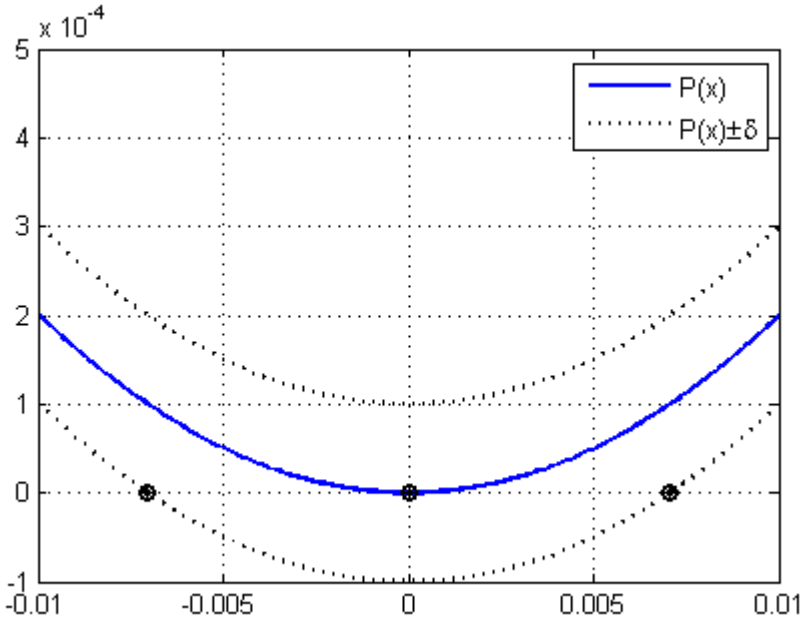

Рис. 3.2. Неустойчивость корня четной кратности при возмущении коэффициентов полинома.

Если задача стоит в отыскании нескольких корней в заданной области с учетом их кратности, то полезным оказывается после вычисления очередного корня исключить его из дальнейшего рассмотрения задачи. Например, найден корень уравнения (3.1) $x = x_1$. Для нахождения других корней уравнения (3.1) в ряде случае удобно рассматривать далее уравнение

$$\frac{f(x)}{x - x_1} = 0, \quad x \in [a,b], \tag{3.2}$$

начиная с процедуры отделения его корней. В случае полиномиального уравнения такая процедура равносильна поэтапному понижению степени полинома. Однако, следует иметь в виду, что корни уравнения вычисляются, вообще говоря, приближенно и процедура исключения корней (3.2) может приводить к возмущениям последующих корней. В силу этого, как правило, после вычисления решений нелинейного уравнения требуется их проверка, а при обнаружении грубости приближения, производится "полировка" корней. В качестве критерия точности решения $x = x^*$ традиционно используется невязка, в качестве которой, очевидно, оценивается значения самой функции, $r(x^*) = f(x^*) \cong 0$.

3.2.Метод дихотомии (бисекций).

Данный метод является надежным и достаточно эффективным способом определения простых корней нелинейного уравнения. Если уравнение (3.1) имеет на отрезке $[a,b]$ один простой корень (или корень нечетной кратности), то непрерывная функция $f(x)$ на концах данного отрезка имеет место неравенство $f(a)f(b) \le 0$. Суть метода дихотомии состоит в делении отрезка пополам и выборе той его половинки, на которой находится искомый корень уравнения.

Пусть для определенности $f(a) < 0$, $f(b) > 0$. Вычислим значение функции в середине отрезка, $\bar{x} = \frac{b-a}{2}$. Если $f(\bar{x}) < 0$, то, очевидно, что искомый корень принадлежит отрезку $[\bar{x}, b]$ (в противном случае корень будет принадлежать другой половине отрезка $[a, \bar{x}]$). Таким образом, дальнейшая задача снова состоит в определении нуля функции на отрезке, но длина отрезка в два раза меньше первоначальной. Далее, присваиваем переменной a новое значение $a = \bar{x}$ (или $b = \bar{x}$, если $f(\bar{x}) > 0$) и повторяем деление отрезка. Заметим, что на каждом шаге такого деления мы добиваемся локализации корня на отрезке, длина которого с каждой итерацией уменьшается вдвое. Поскольку длина отрезка, на котором локализован корень, ассоциируется с точность решения поставленной задачи, то мы имеем гарантированную сходимость итерационного процесса со скоростью геометрической прогрессии со знаменателем $q = \frac{1}{2}$. Несложно убедиться, что за 52 итерации такой итерационный процесс сжимает единичный отрезок до размеров, сопоставимых с относительной

погрешностью представления действительных чисел в классе double: $(1/2)^{52} \cong 2.22e-16$.

3.3.Метод простой итерации. Условие и скорость сходимости.

Исходное уравнение (3.1) заменим эквивалентным уравнением вида

$$x = \varphi(x), \tag{3.3}$$

и рассмотрим итерационный процесс

$$x_{k+1} = \varphi(x_k),\ k = 0,1,2,\ldots \quad . \tag{3.4}$$

Итерационный процесс вида (3.4) называется методом простой итерации. Итерационный процесс (3.4) будем называть сходящимся, если при заданном начальном приближении x_0 последовательность $\{x_k\}$ имеет предел при $k \to \infty$. Функция $\varphi(x)$ выбирается из тех соображений, чтобы итерационный процесс (3.4) сходился и корни уравнения (3.3) совпадали с корнями уравнения (3.1).

В силу отмеченных обстоятельств в качестве функции $\varphi(x)$ удобно использовать функцию вида

$$\varphi(x) = x + \tau(x) f(x), \tag{3.5}$$

где $\tau(x)$ не имеет нулей в области поиска корней уравнения (3.1). В простейшем случае в качестве $\tau(x)$ можно использовать некоторый постоянный итерационный параметр τ, с помощью которого можно управлять сходимостью итераций. Рассмотрим вопрос о сходимости итерационного процесса (3.4) . Пусть R – некоторая ограниченная окрестность искомого решения. В случае функции одной переменной это отрезок с центров в точке $x = a$ и длиной $2r$: $R(a,r) = \{x : |x - a| \le r\}$.

Теорема 3.1. *Пусть на отрезке* $R(a,r)$ *функция* $\varphi(x)$ *липшиц-непрерывна,* $|\varphi(x') - \varphi(x'')| \le q|x' - x''|,\ 0 < q < 1$ *и*

$$|\varphi(a) - a| \le (1-q)r. \tag{3.6}$$

Тогда уравнение (3.3) имеет единственное решение $x = x^*$, *к которому сходится итерационный процесс (3.4) при любом начальном приближении* $x_0 \in R(a,r)$, *и для погрешности выполняется оценка*

$$|x_k - x^*| \le q^k |x_0 - x^*|,\ k = 0,1,2,\ldots \tag{3.7}$$

Доказательство. Вначале докажем, что условия теоремы гарантируют принадлежность всех итерационных приближений отрезку $R(a,r)$. Пусть $x_k \in R(a,r)$. Из (3.4), (3.6) следует

$$x_{k+1} - a = \varphi(x_k) - \varphi(a) + \varphi(a) - a,$$
$$|x_{k+1} - a| \le |\varphi(x_k) - \varphi(a)| + |\varphi(a) - a| \le qr + (1-q)r \le r.$$

Таким образом, если $x_k \in R(a,r)$, то $x_{k+1} \in R(a,r)$, что согласно принципу математической индукции доказывает принадлежность всех итераций отрезку $R(a,r)$.

Оценим разность двух последовательных приближений:

$$x_{k+1} - x_k = \varphi(x_k) - \varphi(x_{k-1}) \Rightarrow |x_{k+1} - x_k| \le q|x_k - x_{k-1}| \le q^k |x_1 - x_0|.$$

Тогда, учитывая, что $\forall p \in N$,

$$x_{k+p} - x_k = \sum_{m=1}^{p} (x_{k+m} - x_{k+m-1}),$$

имеем

$$|x_{k+p} - x_k| \le |x_1 - x_0| \sum_{m=1}^{p} q^{k+m-1} = q^k \frac{1-q^p}{1+q} |x_1 - x_0| \le \frac{q^k}{1-q} |x_1 - x_0|.$$

Из последнего неравенства следует, что фундаментальная последовательность $\{x_k\}$ сходится и $\lim\limits_{k\to\infty} x_k = x^* \in R(a,r)$, $x^* = \varphi(x^*)$.

Таким образом, последовательность $\{x_k\}$ сходится к решению уравнения (3.3). Единственность решения x^* на отрезке $R(a,r)$ докажем методом от противного. Пусть x^{**} также решение (3.3). Тогда

$$x^* - x^{**} = \varphi(x^*) - \varphi(x^{**}) \Rightarrow |x^* - x^{**}| \le q|x^* - x^{**}|.$$

Поскольку $q<1$, то последнее равенство может быть выполнено лишь когда $x^* = x^{**}$, что доказывает единственности решения.

Для доказательства оценки (3.7) достаточно оценить разность

$$x_{k+1} - x^* = \varphi(x^*) - \varphi(x_{k-1}) \Rightarrow |x_{k+1} - x^*| \le q|x_k - x^*| \le q^k |x_1 - x^*|.$$

Теорема доказана.

Замечание. Если функция $\varphi(x)$ непрерывно дифференцируемая, то для сходимости метода простых итераций условие липшиц-непрерывности с постоянной $0<q<1$ можно заменить условием $|\varphi'(x)| < 1$, $x \in R(a,r)$.

Как видно из оценки погрешности (3.7), метод простой итерации имеет скорость сходимости геометрической прогрессии со знаменателем, величина которого определяется максимальным значением производной

функции $\varphi(x)$ в окрестности искомого решения. Выбор функции $\varphi(x)$ в виде (3.5) , $\tau(x) \equiv \tau = cont$, позволяет за счет выбора величины и знака итерационного параметра τ добиться не только выполнения условия сходимости итераций, но и обеспечить приемлемую скорость сходимости метода.

Метод простой итерации сохраняет свою структуру при решении систем нелинейных уравнений. Для систем уравнений функции $f(x)$ и $\varphi(x)$, равно как и их аргументы, понимается как вектор-функции и векторы соответственно:

$$f(x) = \left(f_1(x), f_2(x), \ldots, f_N(x)\right)^T,$$
$$\varphi(x) = \left(\varphi_1(x), \varphi_2(x), \ldots, \varphi_N(x)\right)^T,\ x = \left(x_1, x_2, \ldots, x_N\right)^T.$$

Условие сходимости метода простой итерации для систем нелинейных уравнений состоит в выполнении следующей оценки для нормы матрицы Якоби в некоторой окрестности искомого корня:

$$\|J(x)\| \le q < 1,\ J(x) = \begin{pmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \cdots & \frac{\partial f_1}{\partial x_2} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} & \cdots & \frac{\partial f_2}{\partial x_N} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial f_N}{\partial x_1} & \frac{\partial f_N}{\partial x_2} & \cdots & \frac{\partial f_N}{\partial x_N} \end{pmatrix},\ \left\|x - x^*\right\| \le \delta. \qquad (3.8)$$

Пример. Попытаемся найти действительные корни уравнения

$$f(x) = \sqrt[3]{x} - x^2 = 0.$$

Несложно убедиться, что данное уравнение имеет два корня $x^* = 0$, $x^{**} = 1$. Будем использовать для поиска корней итерационный процесс

$$x_{k+1} = \varphi(x_k) = x_k + \tau f(x_k). \qquad (3.9)$$

Для исследования сходимости (3.9) оценим значения модуля производных $\varphi'(x)$ в окрестности корней:

$$\varphi'(x) = 1 - \tau\left(2x - \frac{1}{3\sqrt[3]{x^2}}\right),\ \varphi'(1) = 1 - \tau\frac{5}{3},\quad \forall \tau \quad \lim_{x \to 0}\left|\varphi'(x)\right| = \infty.$$

Таким образом, для итерационного процесса (3.9) условие сходимости в любой, сколь угодно малой окрестности корня $x = 0$ *не выполнено,* и получить значение данного корня при любом начальном приближении и при любом значении параметра τ оказывается невозможным.

Для второго корня условие сходимости выполняется при $0 < \tau < \frac{6}{5}$.

Минимальное значение $|\varphi'(x)| = 0$ достигается при $\tau = \tau_0 = \frac{3}{5}$. Данное значение итерационного параметра обеспечивает наибольшую скорость сходимости итераций при начальном приближении в достаточной близости от искомого корня. В этом несложно убедиться на основе численных экспериментов. Например, при выборе начального приближения $x_0 = 1/2$ динамика сходимости итераций при различных значениях итерационного параметра представлена на рис. 3.3.

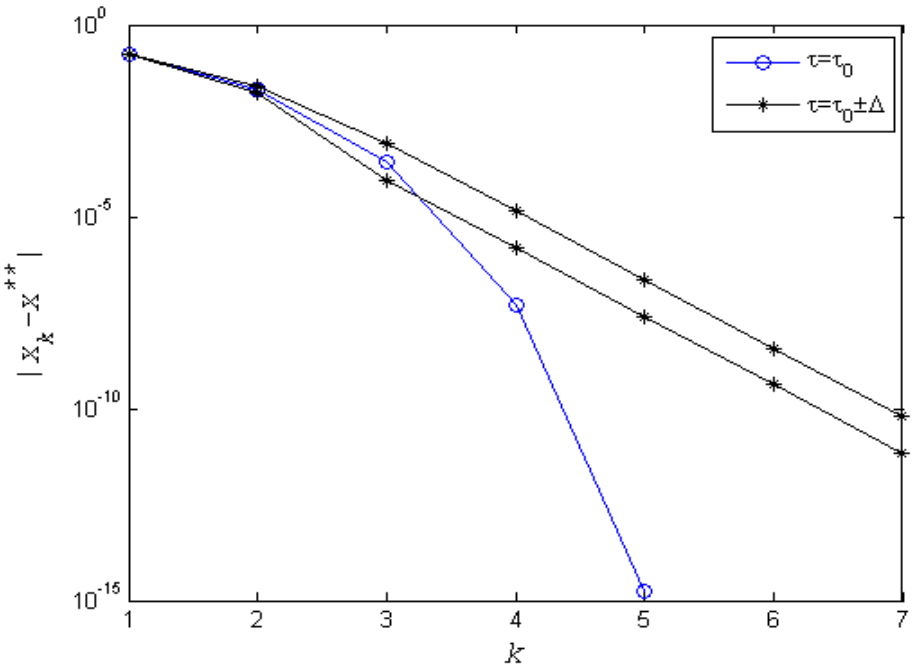

Рис. 3.3. Сходимость итерационного процесса (3.9) при различных значениях итерационного параметра: $\tau = \tau_0$, $\tau = \tau_0 \pm \delta$, $\tau_0 = 3/5$, $\delta = 10^{-10}$.

Из рисунка видно, что при $\tau = \tau_0$ погрешность искомого решения убывает быстрее, чем при других, близких к этому, значениях итерационного параметра. При этом легко видеть, что с приближением к корню скорость сходимости итерационного процесса с оптимальным параметром увеличивается.

3.4. Метод Ньютона. Квадратичная скорость сходимости.

Пусть $x = x^*$ простой корень уравнения (3.1), $f(x^*) = 0$, $f'(x^*) \neq 0$, значение которого предстоит вычислить. Возьмем некоторое начальное приближение $x = x_0$ и выразим $f(x_0)$ отрезком степенного ряда:

$$f(x_0) = f(x^*) + f'(x_0)(x_0 - x^*) + O\left((x_0 - x^*)^2\right).$$

Отсюда

$$x^* = x_0 - \frac{f(x_0)}{f'(x_0)} + O\left((x_0 - x^*)^2\right). \qquad (3.10)$$

Рассмотрим последовательность приближений вида (3.10), отбрасывая члены второго порядка малости:

$$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)}. \qquad (3.11)$$

Итерационный процесс (3.11) впервые был предложен Исааком Ньютоном в 1669 году и сейчас известен как *метод Ньютона*.

Для выяснения вопроса о сходимости метода Ньютона заметим, что итерационный процесс (3.11) имеет вид метода простой итерации (3.4), (3.5) с функцией

$$\varphi(x) = x + \tau(x) f(x), \quad \tau(x) = -\frac{1}{f'(x)}.$$

Вычислим производную $\varphi'(x)$:

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

Для дважды непрерывно дифференцируемой функции $f(x)$, $f'(x) \neq 0$, имеем $\lim\limits_{x \to x^*} \varphi'(x) = 0$, и вблизи корня $x = x^*$ найдется окрестность $\left[x^* - \delta, x^* + \delta\right]$, в которой будет выполнено условие сходимости итераций, $|\varphi'(x)| < 1$. Следовательно, при выполнении отмеченных выше ограничений на функцию $f(x)$, метод Ньютона сходится, если начальное приближение оказывается в достаточной близости от искомого корня.

Из выражения (3.10) следует, что если на k-ой итерации погрешность $\delta_k = \left|x_k - x^*\right|$, то на следующей (k+1)-ой итерации погрешность будет

$$\delta_{k+1} = \left|x_{k+1} - x^*\right| = O\left((x_k - x^*)^2\right) = O(\delta_k^2). \qquad (3.12)$$

Таким образом, в отличие от метода простой итерации, где погрешность убывает по закону геометрической прогрессии (3.7) с коэффициентом $q \le |\varphi'(x)| < 1$, для метода Ньютона имеет место *квадратичный* закон убывания погрешности (3.12).

Метод Ньютона можно рассматривать как оптимизированный вариант метода простой итерации, в котором итерационный параметр τ выбирается из условия минимума $|\varphi'(x)|$. В самом деле, рассмотрим метод простой итерации с функцией $\varphi(x) = x + \tau f(x)$. Выберем значение итерационного параметра τ, при котором производная функции $\varphi(x)$ будет равная нулю:

$$\varphi'(x) = 1 + \tau f'(x) = 0, \Rightarrow \tau = -\frac{1}{f'(x)} \Rightarrow \varphi(x) = x - \frac{f(x)}{f'(x)}.$$

Рассмотренный в предыдущем параграфе пример показывает, что выбор итерационного параметра из условия $\varphi'(x) = 0$ позволяет качественного улучшения сходимости метода простых итераций.

В случае кратных корней метод Ньютона в виде (3.11) не имеет квадратичной сходимости. Напомним, что корень x^* имеет кратность p, если $f(x^*) = 0,\ f'(x^*) = 0,\ f''(x^*) = 0, \ldots,\ f^{(p-1)}(x^*) = 0,\ \ f^{(p)}(x^*) \neq 0.$
В связи с этим хотелось бы отметить модифицированный метод Ньютона с параметром:

$$x_{k+1} = x_k - \tau \frac{f(x_k)}{f'(x_k)}. \qquad (3.13)$$

Выбор значения итерационного параметра $\tau = p$ позволяет достичь квадратичной скорости сходимости (3.13) при нахождении корней кратности p. Данная модификация может оказаться полезной для расширения области сходимости итераций. С этой целью следует использовать (3.13) с параметром $\tau < 1$. При этом важно помнить, что уменьшение итерационного параметра расширяет область сходимости, но в общем случае ведет к увеличению числа итераций для достижения заданной точности.

Метод Ньютона имеет аналогичную структуру при *решении систем нелинейных уравнений*. Для систем уравнений $f(x)$ – вектор функция, $f(x) = \left(f_1(x), f_2(x), \ldots, f_N(x)\right)^T$, где $x = \left(x_1, x_2, \ldots, x_N\right)^T$. Роль производной в этом случае выполняет матрица Якоби (см. (3.8)). Удобно использовать следующую формулировку метода Ньютона для систем нелинейных уравнений:

$$J(x^k)\,\Delta^{k+1} = -f(x^k), \quad x^{k+1} = x^k + \Delta^{k+1}. \qquad (3.14)$$

На каждой итерации (3.14) для вычисления очередного приближения x^{k+1} необходимо вначале решить систему линейных алгебраических уравнений с матрицей Якоби для вычисления приращения $\Delta = x^{k+1} - x^{k+1}$. Условие сходимости итераций состоит в требовании *невырожденности* матрицы Якоби в некоторой окрестности искомого корня, $\|x - x^*\| \le \delta$.

Упражнение.

Приведите пример уравнения $f(x) = 0$, для которого метод Ньютона не сходится при любом начальном приближении.

3.5.Модификации метода Ньютона.

В ряде случаев использование метода Ньютона в классической формулировке (3.11) , (3.14) бывает затруднено или невозможно. Например, затруднения могут быть связаны с вычислением на каждой итерации производной функции (или Якобиана, в случае нелинейных систем). Одна из полезных модификаций в этом случае состоит в том, что вычисленное на одном из очередных приближений значение производной (матрица Якоби) может в неизменном виде использоваться не для одной, а для нескольких итераций, вплоть до достижения полной сходимости. Вычислительная сложность одной итерации в этом случае может существенно снизиться, однако общее число итераций увеличится из-за снижения скорости сходимости.

Бывают ситуации, когда функция $f(x)$ не имеет аналитического выражения. Например, это может быть в случае, если зависимость $f(x)$ сама по себе является решением некоторой задачи, и мы можем вычислить значения этой функции для любого значения аргумента, но получить ее выражение, пригодное для аналитического дифференцирования, не представляется возможным.

В данном случае может быть полезной модифицированный метод Ньютона, получивший название *метод секущих*. Суть данной модификации состоит в том, что аналитические значения производной функции (якобиана), заменяются соответствующими разностными приближениями. Для этого (ограничимся скалярным случаем) вместо одного начального приближения задается два близких друг к другу начальных значения x_0 и $x_1 \ne x_0$. Пусть для определенности $x_1 < x_0$. Приближенное зна-

чение производной функции $f(x)$ на отрезке $[x_1, x_0]$ выразим посредством разделенной разности: $f'(x) \cong \dfrac{f(x_1) - f(x_0)}{x_1 - x_0}$. Данное выражение естественным образом обобщается на случай произвольных последовательных приближений x_{k-1}, x_k: $f'(x) \cong \dfrac{f(x_k) - f(x_{k-1})}{x_k - x_{k-1}}$. Использование данного представления в (3.11) приводит к модифицированной схеме метода Ньютона – *методу секущих*, в котором не требуется непосредственного вычисления производной $f'(x)$:

$$x_{k+1} = x_k - \frac{f(x_k)}{f(x_k) - f(x_{k-1})}(x_k - x_{k-1}). \qquad (3.15)$$

Отличие метода секущих (3.15) от классического варианта метода Ньютона (3.11) легко прояснить на основе геометрической интерпретации данных подходов, представленных на рис. 3.5.

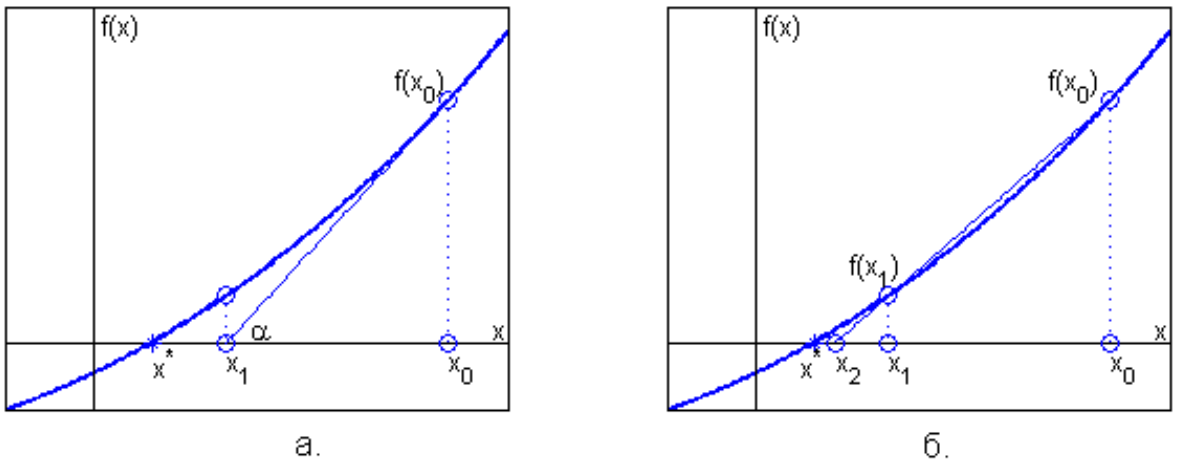

Рис. 3.4. Геометрическая интерпретация **а.** метод Ньютона (метод касательных); **б.** метод секущих.

В методе Ньютона (3.11) очередное приближение x_{k+1} находится на пересечении оси (O,x) и касательной, проведенной к графику функции в точке предыдущего итерационного приближения x_k (см. рис.3.4 а.). Это непосредственно вытекает из прямоугольного треугольника с вершинами в точках $(x_0,0), (x_1,0), (x_1, f(x_1))$, в котором $\operatorname{tg}\alpha = f'(x_0)$.

Аналогичные геометрические построения для модифицированного метода (3.15) показывают, что очередное итерационное приближение

находится на пересечении оси (O,x) и прямой, проходящей через точки графика, соответствующие двум предыдущим последовательным приближениям (см. рис. 3.4 б.).

В заключение, несколько слов о методе Ньютона с параметром (3.13). В случае системы нелинейных уравнений метод Ньютона фактически состоит в поиске глобальных минимумов функции нескольких переменных $F(x) = f(x)^T \cdot f(x)$, $x = \left(x^1, x^2, \ldots, x^N\right)^T$. Шаг метода Ньютона (3.14) состоит в приращении аргументов данной функции и можно показать, что приращение этой *положительной* функции при достаточной малости приращения аргумента будет *отрицательным*. В самом деле

$$\Delta F = F(x+\Delta) - F(x) \cong \text{grad}(F) \cdot \Delta = 2\left(f^T \cdot J\right)\left(-J^{-1} f\right) = -2F .$$

Таким образом, шаг приращения Δ_{k+1} очередного приближения в методе Ньютона (3.14) имеет направление, совпадающее с направлением *убывания* функции $F(x) = f(x)^T \cdot f(x)$, глобальные минимумы которой совпадают с корнями уравнения $f(x) = 0$. Если шаг в данном направлении будет достаточно большим, то может оказаться, что $F(x+\Delta) > F(x)$ и новое приближение окажется дальше от искомого минимума (корня) нежели предыдущее.

Приведенные выше нестрогие рассуждения позволяют прояснить проблему отсутствия *глобальной сходимости* классической схемы метода Ньютона и изыскать возможности ее модификация для устранения данного недостатка. В частности, введение итерационного параметра позволяет фактически регулировать величину очередного приращения и вместо шага Δ_{k+1} использовать шаг :

$$J(x_k)\Delta_{k+1} = -f(x_k), \quad x_{k+1} = x_k + \tau\Delta_{k+1} \quad . \qquad (3.16)$$

При разумной стратегии выбора величины итерационного параметра τ возможно достижение практически *глобальной* сходимости модернизированного метода Ньютона (3.16).

Упражнения:

1. При каких значениях *p* метод простой итерации $x_{k+1} = x_k + p\left(1 - \sqrt{x_k}\right)$ сходится при выборе начального приближения $x_0 \geq 1$. При каком значении параметра *p* скорость сходимости рассмотренного метода будет сравнима со скоростью сходимости метода Ньютона.

2. При решении нелинейного уравнения методом ньютона за первые двенадцать итераций получено приближенное решение с погрешностью $\delta = 2 \cdot 10^{-4}$. Сколько всего итераций потребуется для получения решения с погрешностью $\varepsilon \le 10^{-7}$.

3.6.Понятие о методах нелинейной оптимизации.

Задачи оптимизации имеют широкий круг приложений в математических методах планирования, управления и других областях естественных и социальных наук. Примером такого рода задач может быть задача вычисления значений параметров в некоторой зависимости, при которых данная зависимость максимально приближает некоторые данные, полученные экспериментальным путем (*нелинейная подгонка*).

Задачи оптимизации близки к задачам решения уравнений (линейных и нелинейных). В ряде случаев речь идет даже не об аналогии, а об эквивалентности подобных задач, что было показано при рассмотрении градиентных методов решения систем линейных и нелинейных уравнений. В силу этого численные методы оптимизации (линейной и нелинейной) имеют много общего с численными решения соответствующих систем уравнений.

В общем виде задача оптимизации состоит в поиске абсолютных экстремумов функции многих переменных, например, точки глобального минимума, $F(x) \to \min, x = \left(x^1, x^2, \ldots, x^N\right)^T$, в ограниченной или неограниченной области N-мерного пространства. В дальнейшем мы будем понимать под задачей оптимизации задачу поиска абсолютного минимума некоторой функции. Минимизируемую функцию называют *целевой функцией,* которая формируется на основе определенных *критериев оптимальности.* Решением задачи оптимизации будем называть вектор $x^* : \forall x \ne x^*, F(x^*) \le F(x)$.

На пространство независимых переменных могут налагаться *ограничения в виде равенств,* $a_k x = 0, k = \overline{1, K}$ и (*или) неравенств,* $b_m x > 0, m = \overline{1, M}, \quad a \le x \le b$. При наличии таких условий задача называется *оптимизация с ограничениями* (*условная оптимизация*), в противном случае оптимизация называется *безусловной*.

Вначале остановимся на методах решения задач безусловной оптимизации и рассмотрим пример, когда целевая функция зависит от одной переменной и область определения ограничена отрезком. В этом случае простейшая стратегия определения минимума может быть основана на идее *общего поиска*. Отрезок области определения целевой функции

разбивается на равные интервалы, в граничных узлах интервалов вычисляются значения функции, среди которых выбирается минимальное значение. Точность решения в этом случае определяется длиной двух интервалов разбиения, а вычислительная сложность характеризуется числом вычислений значений целевой функции. Несложно убедиться, что вычислительная сложность метода общего поиска линейно зависит от требуемой точности решения задачи.

Если целевая функция *унимодальна,* т.е. имеет на отрезке оптимизации один минимум и монотонна по обе стороны от экстремума, то для поиска минимума может быть использован метод деления отрезка пополам, который полностью аналогичен рассмотренному выше методу дихотомии при поиске нулей функции.

Если целевая функция $F(x)$ дважды дифференцируема, то для решения задачи оптимизации можно воспользоваться известными критериями минимума функции: $F'(x)=0$, $F''(x)<0$. Тем самым задача минимизации $F(x)\to\min$ сводится к поиску корней уравнения $F'(x)=0$. Задавая некоторое начальное приближение $x=x_0$, мы можем использовать итерационный метод Ньютона с параметром:

$$\Delta_{k+1}=x_{k+1}-x_k=-\frac{F'(x_k)}{F''(x_k)},\quad x_{k+1}=x_k+\tau_{k+1}\Delta_{k+1}. \quad (3.17)$$

Для коррекции приращения аргумента функции используем критерии: $F''(x_k)<0$, $F(x_{k+1})<F(x_k)$. При выполнении обоих критериев итерационный параметр полагается равным единице. Если нарушен первый критерий, то $\tau=-1$, если не выполняется второе условие, то $\tau=1/2$. Коррекция итерационного параметра производится с целью предотвращения сходимости метода к локальному максимуму.

Методы решения многомерных задач оптимизации могут быть основаны на их редукции к последовательности аналогичных одномерных задач. Примером является метод покоординатного спуска, когда в пространстве независимых переменных фиксируются $N-1$ координата и рассматривается задача поиска минимума функции вдоль оставшейся одной независимой переменной, например, вдоль координаты x_1. На следующем шаге полученное значение переменной x_1, минимизизирующее целевую функцию на заданной прямой, параллельной соответствующей координатной оси, фиксируется и рассматривается задача оптимизации вдоль другой координаты. Циклическое повторение данной процедуры поочередно для всех координат позволяет приблизить решение задачи

оптимизации, причем при наличии в ограниченной области единственной точки минимума целевой функции такой итерационный процесс гарантированно сходится.

3.7.Градиентные методы минимизации функции.

По определению, градиент функции $F(x): R^N \to R$ – это вектор $g(x) = \nabla F = \left(\frac{\partial F}{\partial x_1}, \frac{\partial F}{\partial x_2}, \ldots, \frac{\partial F}{\partial x_N} \right)^T$. Направление градиента функции совпадает с направлением ее наискорейшего роста. Естественно использовать данное свойства при выборе направления поиска минимума функции в методах оптимизации. Очевидно, что минимума функции можно достичь, двигаясь в направлении противоположном градиенту. В точке локального минимума градиент функции обращается в нуль.

Рассмотрим градиентные методы, используемые для минимизации функции среднеквадратичного отклонения при построении регрессионных моделей – нелинейной подгонке. Типичная задача нелинейной подгонки состоит в поиске такого вектора параметров модели $x \in R^N$, который обеспечивает минимальное среднеквадратичное отклонение результатов моделирования $v_m(x) \in R^M$ и отвечающих им экспериментальных данных $s_m \in R^M$:

$$F(x) = \frac{1}{2} \sum_{m=1}^{M} \left(v_m(x) - s_m \right)^2 = \frac{1}{2} \sum_{m=1}^{M} f_m^2(x), \quad f_m = v_m(x) - s_m . \tag{3.18}$$

Метод Ньютона для нахождения минимума функции $F(x)$ имеет вид:

$$x^{(k+1)} = x^{(k)} - H^{-1}(x^{(k)}) \nabla F(x^{(k)}), \tag{3.19}$$

где $H(x)-$ матрица Гесса для функции $F(x)$, которая фактически есть матрица Якоби для её градиента $\nabla F = \left(\frac{\partial F}{\partial x_1}, \frac{\partial F}{\partial x_2}, \ldots, \frac{\partial F}{\partial x_N} \right)^T$:

$$H(x)=\begin{pmatrix} \dfrac{\partial^2 F}{\partial x_1 \partial x_1} & \dfrac{\partial^2 F}{\partial x_1 \partial x_2} & \cdots & \dfrac{\partial^2 F}{\partial x_1 \partial x_N} \\ \dfrac{\partial^2 F}{\partial x_2 \partial x_1} & \dfrac{\partial^2 F}{\partial x_2 \partial x_2} & \cdots & \dfrac{\partial^2 F}{\partial x_2 \partial x_N} \\ \vdots & \vdots & \ddots & \vdots \\ \dfrac{\partial^2 F}{\partial x_N \partial x_1} & \dfrac{\partial^2 F}{\partial x_N \partial x_2} & \cdots & \dfrac{\partial^2 F}{\partial x_N \partial x_N} \end{pmatrix}.$$

С учетом (3.18) вычислим градиент и матрицу Гесса функции $F(x)$:

$$\nabla F = J^T(x) f(x),\ H(x) = J^T(x)J(x) + Q(x),\ Q(x) = \sum_{m=1}^{M} H_m(x) f_m(x). \quad (3.20)$$

Здесь $J(x)$ – Якобиан вектора $f(x)$, $H_m(x)$ – матрица Гесса для функции $f_m(x)$.

По сути, метод Ньютона (3.19), (3.20) является классическим методом Ньютона для нахождения нулей градиента целевой функции $F(x)$. Его реализация осложняется необходимостью вычисления на каждой итерации матрицы Гесса. На практике матрица Гесса вычисляется посредством нахождения производных с помощью конечных разностей, что сопряжено с большими вычислительными затратами.

Одна из модификаций метода Ньютона, позволяющая повысить его эффективность, состоит в возможности пренебречь в выражении для вычисления матрицы Гесса слагаемым $Q(x)$, что допустимо при условии $\left\| J^T(x)J(x) \right\| >> \left\| Q(x) \right\|$. Данная модификация называется методов *Ньютона-Гаусса* и имеет вид

$$x^{(k+1)} = x^{(k)} - \left[J^T(x^{(k)}) J(x^{(k)}) \right]^{-1} J^T(x^{(k)}) f(x^{(k)}). \quad (3.21)$$

Метод *Ньютона–Гаусса* в ряде случаев дает хорошие результаты, демонстрируя квадратичную сходимость, как и метод Ньютона. Однако, в случае, кода условия для упрощенного вычисления матрицы Гесса не выполняются, в методе Ньютона–Гаусса могут проявляться проблемы сходимости. В связи с этим на практике более широкое применение получили другие модификации градиентных методов, в частности, метод *Левенберга–Марквардa*.

Заметим, что в методе Ньютона–Гаусса (3.21) приращение вектора x на каждой итерации, $\Delta^{(k+1)} = x^{(k+1)} - x^{(k)}$, находится из решения системы линейных алгебраических уравнений

$$J^T(x^{(k)})J(x^{(k)})\Delta^{(k+1)} = -J^T(x^{(k)})f(x^{(k)}).$$

В итерационном методе Левенберга – Марквардта приращение $\Delta^{(k+1)} = x^{(k+1)} - x^{(k)}$ определяется из системы уравнений

$$\left(J^T(x^{(k)})J(x^{(k)}) + \lambda_k E\right)\Delta^{(k+1)} = -J^T(x^{(k)})f(x^{(k)}),\ x^{(k+1)} = x^{(k)} + \Delta^{(k+1)} \quad (3.22)$$

Несложно заметить, что если $\lambda_k = 0$ итерационный метод (3.22) совпадает с методом Ньютона – Гаусса, в то время как при $\lambda_k >> \left\|J^T(x^{(k)})J(x^{(k)})\right\|$ приращение $\Delta^{(k+1)}$ стремится к направлению наискорейшего спуска. Если выбрать значение λ_k слишком большим, то при расчете вектора $\Delta^{(k+1)}$ теряется информация о кривизне целевой функции, которая определяется первым слагаемым в скобках выражения (3.22). Рациональный способ выбора параметров λ_k состоит в увеличении его значения до тех пор, пока выполнятся монотонность убывания целевой функции от итерации к итерации: $F(x^{(k+1)}) < F(x^{(k)})$.

Дальнейшее улучшение алгоритма Левенберга – Марквардта состоит в использовании в (3.22), вместо единичной матрицы E, диагональной матрицы, элементами которой являются диагональные элементы матрицы $J^T(x^{(k)})J(x^{(k)})$. Это позволяет гибко регулировать длину приращения $\Delta^{(k+1)}$, увеличивая ее в области медленного изменения целевой функции, и уменьшая вблизи крутых перепадов.

Алгоритм Левенберга – Марквардта широко используется, например, в системе Matlab, в приложениях для решения задач оптимизации и нелинейной подгонки и др.

4. ПРИЛОЖЕНИЕ

4.1.Функции Matlab для решения задач линейной алгебры.

Для получения более подробной информации о назначении и порядке использования функций Matlab используйте справочную систему:

help <имя функции>, или

doc <имя функции>.

Функции Формирование матриц.

rand – массив случайных чисел с равномерной плотностью вероятности на отрезке [0,1].
randn – массив случайных чисел с нормальной плотностью вероятности на отрезке [-1,1].
pascal – матрица Паскаля.
eye, speye– единичная матрица.
hilb – матрица Гильберта.
ones – массив, все элементы которого равны единице.
zeros– массив, все элементы которого равны нулю.
gallery – галерея матриц .

Операции с матрицами.

diag – построение диагональной матрицы или вектор диагональных элементов матрицы.
triu – верхняя треугольная матрица.
tril – нижняя треугольная матрица.
kron – тензорное (кронекеровское) произведение матриц.
sparse – преобразование в формат разреженной матрицы.
reshape – преобразование размерности массива.

Решение систем ЛАУ и факторизация матриц.

linsolve – решение систем ЛАУ прямым методом.
x=A\b – решение системы ЛАУ Ax=b методом Гаусса.
lu – LU разложение.
chol – разложение Холецкого.
qr – QR разложение.
inv – вычисление обратной матрицы.

pcg – решение системы ЛАУ методом сопряженных градиентов.
gmres – решение системы ЛАУ обобщенным методом минимальных невязок.

Собственные значения и нормы

eig – полный набор собственных значений и собственных векторов.
eigs – максимальные собственные значения и соответствующие им собственные векторы.
norm, normest – норма вектора или матрицы.
cond, condest – число обусловленности.

Другие функции.
plot – рисование графика функции, заданной массивом значений.
semilogy – рисование графика функции в логарифмическом масштабе по оси ординат ***y***.
spy – визуализация структуры разреженной матрицы.
mesh – визуализация функции двух переменных в виде поверхности.
contour – рисование изоуровней функции двух пременных.
meshgrid – построение многомерной сетки узлов для функций нескольких переменных.
roots – вычисление корней полинома.
solve – решение нелинейных уравнений и систем.

Некоторые полезные конструкции языка программирования Matlab

a:d:b – вектор, первый элемент которого равен **a** а каждый последующий больше предыдущего на величину **d.** Последний элемент не превосходит **b**.

for n=N:m:K
…
end - цикл с переменной **n**, принимающей значения от **N** до **K** с шагом **m.**

for (s> S & k<K)
…
end - цикл, выполнение которого продолжается пока **s> S** и **k<K**

A(:) – формирование вектор столбца из массива **A** произвольной размерности.
A(3:end-1,:) – возвращает матрицу из строк матрицы **A**, начиная с третьей и заканчивая предпоследней строкой.
A(:,3) – возвращает третий столбец матрицы.

```
if I==K
    A(I,K)=0;
 elseif  I>K
     A(I,K)=-1;
  else
      A(I,K)=1;
end
```

Условное выполнение фрагментов программы. В приведенном примере диагональным элементам матрицы присваиваются нулевые значения, а верхней и нижней треугольной части матрицы присваиваются значения ±1 соответственно.
A.*B – поэлементное умножение элементов массива **A** на соответствующие элементы массива **B.**
A.^2 – возведение в квадрат массива **A** поэлементно**.**
f=@(x) sin(x).*exp(-x.^2) – задание анонимной функции.
g=f(t) – вычисление значений ананимной функции **f=@(x)** от значений аргументов, заданных массивом **t.**

4.2. Реализация итерационных методов минимальных невязок и сопряженных градиентов

Приведенная программа позволяет убедиться в выполнении оценок скорости сходимости итерационных методов сопряженных градиентов и минимальных невязок.

```
% Демонстрация зависимости скорости сходимости итерационных
% методов минимальных невязок и сопряженных градиентов от
% числа обусловленности матрицы A для системы ЛАУ Ax=f.
% A - Матрица Пуассона - пятидиагональная матрица, которая
% возникает при разностной аппроксимации двумерного уравнения
% Пуассона. Число обусловленности пропорционально размерности
% матрицы Пуассона
m = 0;
for n = 5:20;
```

```
    m = m+1
    N = n^2;
    A = gallery('poisson',n); % Матрица Пуассона NxN
    f = rand(N,1);            % Вектор правой части
    x = zeros(N,1);           % Начальное приближение
    epsilon = 1.e-6;          % Минимальная норма невязки
    r = A*x-f;                % Начальная невязка решения
    f_norm = norm(f);         % Норма правой части системы
    rel_res = norm(r)/f_norm; % Относительная невязка решения
    k = 0;                    % Счетчик итераций
    % Итерационнный метод минимальных невязок
    % см. формулу (1.47)
    while (rel_res>epsilon & k<10000) %Критерий остановки
        Ar = A*r;
        tau = (Ar.'*r)/(Ar.'*Ar); % Итерационный параметр
        x = x-tau*r;              % очередное приближения
        r = A*x-f;                % Вычисление невязки
        rel_res = norm(r)/f_norm; % Относительная невязка
        k = k+1;                  % Счетчик итераций
    end
% при окончании цикла переменная k содержит число
% итераций для достижения заданной точности

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    % Решение системы A*x=f методом сопряженных градиентов
    % см. формулы (1.53) - (1.62)
    k_cg = 0;                 % Счетчик итераций
    x = zeros(N,1);           % Начальное приближение
    epsilon = 1.e-6;          % Минимальная норма невязки
    r = A*x-f;                % Начальная невязка решения
    Ar = A*r;                 % Произведение A на невязку
    f_norm = norm(f);         % Норма правой части системы
    rel_res = norm(r)/f_norm; % Относительная невязка решения
    sigma0 = r'*r;            % Сигма (1.54)
    tau = sigma0/(Ar'*r);     % (1.55)
    x = x-tau*r;              % (1.56)
    d=r;


       while (rel_res>epsilon & k_cg<10000) %Критерий оста-
новки итераций
            Ar = A*r;
            r = A*x-f;                % (1.57)
            sigma = r'*r;             % (1.58)
            betta = sigma/sigma0;     % (1.59)
            d = r+betta*d;            % (1.60)
            Ad = A*d;
            tau = sigma/(Ad.'*d);     % (1.61)
```

```
            x = x-tau*d;                % (1.61)
            rel_res = norm(r)/f_norm; % Относительная невязка
            k_cg = k_cg+1;              % приращение счетчика
            sigma0 = sigma;             % обновление  sigma0
    end

%%%%%%%%%%%%%%%% см.  также help pcg %%%%%%%%%%%%%%%%%%%%%%%%
    display( ['число итераций: ',num2str(k)])
    NN(m) = n;
    K_mr(m) = k;
    K_cg(m) = k_cg;
    CN(m) = condest(A);
end
% Построение графиков зависимости количества итераций
% от числа обусловленности матрицы системы
plot(CN,K_mr,'.-',CN,K_cg,'o-')
grid on
title('Convergense rate vs. Cond. Numb.')
legend('Min. Res.','Conj. Grad')
xlabel('Condition Number K_A')
ylabel('Iterations to convergence')
```

4.3.Метод преобразований Гивенса для вычисления собственных значений матрицы

```
% Проблема собственных значений
% Метод Вращений
N = 22;       % Задание размерности матрицы
A = rand(N); %Формирование матрицы случайных значений
A0 = A*A';   %Формирование симметричной матрицы
A = A0;
Err = 1;      %Инициация начального значения погрешности
kk = 0;       %Инициация счетчика итераций
while Err>1.e-9 % Начало цикла итерационного процесса (пока Err>1/e9)
    kk = kk+1;    % Приращение счетчика итераций
for k = 2:N       % Вниз по столбцам
%Выбор максимального по модулю элемента к-ой строки слева
%от главной диагонали для исключения поддиагональных элементов
[mm,m] = find((abs(A(k,1:k-1))-max(abs(A(k,1:k-1))))==0);
        w = 2*A(k,m)/(A(k,k)-A(m,m));   %Тангенс дв-го угла вращения
        ss = sqrt(1+w^2);
        s = sign(w)*sqrt((ss-1)/(2*ss));%Синус угла вращения
        c = sqrt((ss+1)/(2*ss));        %Косинус угла вращения
        T = eye(N);                     %%%%%%%%%%%%%%%%%%%%%%%
        T(m,m) = c;                     %Формирование матрицы
        T(k,k) = c;                     %вращения
        T(k,m) = -s;                    %T_km
        T(m,k) = s;                     %%%%%%%%%%%%%%%%%%%%%%%
        A = T'*A*T;     % Преобразование Гивенса (плоских вращений)
```

```
 end
   AA = A-diag(diag(A));  % Выделение НЕдиагональных элементов матрицы
   Err = sum(sum(AA.^2)); % Выч. суммы квадратов недиагональных элем.
end                       % Конец цикла итерационного процесса


S=sparse(A);          % Формат разреженной матрицы
spy(abs(S)>1e-5)      % Визуализация структуры разреженной матрицы
{'Iterations' kk}     % Вывод числа итераций
'eigenvalues'
Labd = sort(diag(A),'ascend') % Спектр матрицы в порядке возрастания
Labd_f = eig(A0)               % Спектр матрицы с помощью функц. e
```

Литература

1. *Бахвалов Н. С., Жидков Н. П., Кобельков Г. М.* Численные методы. М. : БИНОМ. Лаб. знаний, 2003. 636 с.
2. *Самарский А.А., Гулин А.В*. Численные методы. М.: Наука, 1989. 432 с.
3. *Крылов В. И., Бобков В. В., Монастырный П. И.* Начала теории вычислительных методов. Линейная алгебра и нелинейные уравнения. Минск: Наука и техника, 1985. 279 с.
4. *Фадеев Д.К., Фадеева В.Н.* Вычислительные методы линейной алгебры. СПт. 2002.
5. *Воеводин В.В., Кузнецов Ю.А.* Матрицы и вычисления. М., Наука, 1984. 320 с.
6. Press, W. H., Teukolsky, S. A., Vetterling, W. T., & Flannery, B. P.. Numerical Recipes: The art of scientific computing. New York. 1997. 973p
7. Lindfield G. R., Penny J. E. T. Numerical methods: using MATLAB Third edition. – Academic Press, 2012. 534 p.

Оглавление

Учебное издание

Волков Василий Михайлович

ЧИСЛЕННЫЕ МЕТОДЫ. I

учебно-методическое пособие
для студентов, обучающихся по специальности
1-31 03 01 «Математика,
1-31 03 01-05 «Математика (информационные технологии)»

Редактор *Е. В. Павлова*
Технический редактор *Т. К. Раманович*
Компьютерная верстка *А. А. Микулевича*
Корректор *Т. Н. Крюкова*

Подписано в печать 00.03.2013. Формат 60×84/16. Бумага офсетная.
Печать офсетная. Усл. печ. л. 0,0. Уч.-изд. л. 0,0. Тираж 100 экз. Заказ

Белорусский государственный университет.
ЛИ № 02330/0494425 от 08.04.2009.
Пр. Независимости, 4, 220030, Минск.

Республиканское унитарное предприятие
«Издательский центр Белорусского государственного университета».
ЛП № 02330/0494178 от 03.04.2009
Ул. Красноармейская, 6, 220030, Минск.