ПРОГРАММИРОВАНИЕ, 2023, № 5, с. 79–86

КОМПЬЮТЕРНАЯ АЛГЕБРА

УДК 004.421.6+519.61

ЭФФЕКТИВНЫЕ НИЖНИЕ ГРАНИЦЫ ДЛЯ РАНГА МАТРИЦЫ И ПРИЛОЖЕНИЯ

© 2023 г. О. А. Зверков[a],*, А. В. Селиверстов[a],**

[a] *Институт проблем передачи информации им. А.А. Харкевича Российской академии наук*
127051 Москва, Большой Каретный пер., д. 19, Россия

**E-mail: zverkov@iitp.ru*

***E-mail: slvstv@iitp.ru*

Поступила в редакцию 26.06.2022 г.
После доработки 27.07.2022 г.
Принята к публикации 30.10.2022 г.

Предложена эффективно проверяемая нижняя граница для ранга разреженной вполне неразложимой квадратной матрицы, содержащей по два ненулевых элемента в каждой строке и каждом столбце. Ранг такой матрицы равен порядку или отличается на единицу. Построены базисы специального вида в пространствах квадратичных форм от фиксированного числа переменных. Существование таких базисов позволило нам обосновать эвристический алгоритм для решения задачи распознавания, проходит ли данное аффинное подпространство через вершину многомерного единичного куба. В худшем случае этот алгоритм может вернуть уведомление о неопределенности результата вычисления, но для общего подпространства достаточно малой размерности корректно отвергает вход. Алгоритм реализован на языке Python. В ходе тестирования получены оценки времени работы этой реализации алгоритма.

DOI: 10.31857/S0132347423020176, **EDN:** MHEYDZ

1. ВВЕДЕНИЕ

Python – современный язык программирования общего назначения, вышедший в последние годы на первое место в различных рейтингах популярности и особенно востребованный в научных приложениях. Сильными сторонами Python считаются легкость начального освоения, быстрота процесса разработки, лаконичность и удобочитаемость текстов программ. В качестве основного недостатка обычно называется относительно низкая производительность программ, обусловленная в основном интерпретируемой природой языка и динамической типизацией. Однако этот недостаток во многом компенсируется доступностью большого разнообразия библиотек, предоставляющих Python-разработчикам доступ к эффективным реализациям различных алгоритмов, а также возможностью подключения собственных программных модулей, написанных на компилируемых языках [1]. В контексте нашей задачи удобной особенностью Python является встроенная в язык поддержка работы с целыми числами неограниченной битовой длины, а также с рациональными числами.

Ранг $n \times n$ матрицы над полем можно вычислить, используя полиномиальное число процессоров и выполнив на каждом из них лишь $O(\log_2^2 n)$ операций над этим полем [2, 3]. Эффективное распараллеливание возможно не только для вычисления ранга, но также для вычисления обратной матрицы, LDU-разложения квадратной матрицы и разложения Холецкого симметричной положительно определенной матрицы [4, 5]. С другой стороны, сложность вычисления ранга [6] и характеристического многочлена [7–9] близка к сложности матричного умножения. Для циркулянтов известны легко проверяемые условия невырожденности [10]. Теоретический интерес представляет задача проверки линейной независимости системы многочленов от нескольких переменных. Иногда такие многочлены могут быть заданы короткими выражениями или арифметическими схемами, хотя матрицы коэффициентов таких многочленов имеют большой размер. Поэтому важны легко проверяемые оценки ранга матрицы. Обычно вычисления выполняются над полем рациональных чисел, но символьные вычисления можно проводить и над полями алгебраических чисел [11, 12].

Точки детерминантального многообразия соответствуют (с точностью до умножения на ненулевое число) матрицам фиксированного размера, ранг которых ограничен сверху. Детерминантальное многообразие может иметь очень большую степень [13, 14]. Поэтому над алгебраически за-

мкнутым полем нельзя проверить значение ранга матрицы посредством неветвящейся программы небольшого размера. С другой стороны, на этих многообразиях лежат линейные подпространства, что иногда можно использовать для оценки ранга [15].

Используя оценки ранга, мы рассмотрим эвристический алгоритм проверки инцидентности данного подпространства и какой-либо вершины единичного куба. Такая проверка эквивалентна поиску (0, 1)-решения для системы уравнений

$$\begin{cases} x_0 = 1 \\ x_j = \ell_j(x_0, \ldots, x_s), \quad j > s \\ x_i \in \{0,1\}, \quad i > 0, \end{cases}$$

где через ℓ_j обозначены линейные формы [16].

Хотя уже известны эвристические алгоритмы для решения этой задачи при дополнительных ограничениях [17–19], принято считать, что эта задача трудная для вычисления в худшем случае. Поэтому актуально создание новых алгоритмов, у которых вычислительная сложность в среднем ограничена многочленом от длины входа при других ограничениях.

Говоря о задачах распознавания, мы предполагаем три варианта ответа: вход может быть не только принят или отвергнут, но также возможно явное уведомление о неопределенности выбора. При этом ответ должен быть получен за конечное время и без ошибок, а при выполнении легко проверяемого условия уведомление о неопределенности может быть выдано лишь на малой доле входов среди всех входов данной длины [20, 21]. При оценке алгебраической сложности достаточно требовать, чтобы уведомление о неопределенности выдавалось на множестве входов, на которых обращается в нуль некоторый многочлен, отличный от тождественно нулевого [16, 22].

В разделе 2 представлены теоретические результаты. В разделе 3 описана реализация алгоритма на языке Python. В разделе 4 дано краткое заключение.

2. ТЕОРЕТИЧЕСКИЕ РЕЗУЛЬТАТЫ

Элементами (0, 1)-матрицы служат лишь нули или единицы. Подматрицей называется матрица, полученная удалением некоторых строк и столбцов. Матрица размера $n \times n$ называется вполне неразложимой, когда в ней нет нулевой подматрицы размера $s \times (n - s)$ ни для какого s.

Рассмотрим квадратные матрицы, имеющие в каждой строке и каждом столбце по два ненулевых элемента. Если такого типа (0, 1)-матрица вполне неразложимая, то ее перманент равен двум [23]. В этом случае значение определителя принадлежит множеству $\{-2, 0, 2\}$. Очевидно, вырождена любая такая (0, 1)-матрица над полем характеристики два. Примером вырожденной матрицы над произвольным полем служит 2×2 матрица ранга один, каждый элемент которой равен единице. Другим примером служит (0, 1)-циркулянт

$$\begin{pmatrix} 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \end{pmatrix}.$$

Эта матрица тоже вырожденная, но ее ранг равен трем. Характеристический многочлен этой матрицы равен $x^4 - 4x^3 + 6x^2 - 4x$. Мы дадим нижнюю оценку ранга таких матриц.

Теорема 1. *Дана вполне неразложимая $n \times n$ матрица A, где в каждой строке и каждом столбце ровно по два ненулевых элемента. Ранг матрицы A ограничен снизу:* $\operatorname{rank}(A) \geq n - 1$.

Доказательство. Условие полной неразложимости не нарушается при перестановке строк или столбцов. Поэтому без ограничения общности полагаем, что на главной диагонали матрицы A все элементы ненулевые. Тогда матрица A равна сумме двух невырожденных матриц $A = D + CP$, где обе матрицы C и D диагональные и невырожденные, а через P обозначена матрица перестановки с нулями на главной диагонали. Поскольку матрица A вполне неразложимая, перестановка P состоит из одного цикла длины n.

Для любой матрицы перестановки Q матрица QAQ^{-1} получается из матрицы A согласованной перестановкой строк и столбцов. При этом элементы на главной диагонали остаются на главной диагонали. Поэтому для некоторой матрицы перестановки Q в матрице QAQ^{-1} отличны от нуля элементы на главной диагонали, непосредственно над главной диагональю и еще один элемент в первом столбце и последней строке. Такая матрица QAQ^{-1} имеет вид (для $n > 6$)

$$\begin{pmatrix} * & * & 0 & \cdots & 0 & 0 & 0 \\ 0 & * & * & \cdots & 0 & 0 & 0 \\ 0 & 0 & & \cdots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & * & * & 0 \\ 0 & 0 & 0 & \cdots & 0 & * & * \\ * & 0 & 0 & \cdots & 0 & 0 & * \end{pmatrix},$$

где символом * отмечены ненулевые элементы матрицы. Поскольку при перестановке строк или столбцов ранг матрицы не меняется, ранг матрицы QAQ^{-1} совпадает с рангом матрицы A. Для за-

вершения доказательства теоремы достаточно найти невырожденную $(n-1)\times(n-1)$ подматрицу B в матрице QAQ^{-1}. Выберем подматрицу B, удаляя из матрицы A первую строку и первый столбец. Эта матрица верхнетреугольная. Каждый элемент на главной диагонали матрицы B отличен от нуля. Поэтому матрица B невырожденная. □

В теореме 1 условие полной неразложимости важно. Например, для четного n у блочно-диагональной матрицы размера $n\times n$, на диагонали которой стоят 2×2 блоки по четыре единицы в каждом, ранг равен $n/2$.

Далее мы рассматриваем поля характеристики нуль. Рассмотрим задачу об инцидентности данного аффинного подпространства и какой-либо вершины n-мерного куба. Метод состоит в попытке построить алгебраическую гиперповерхность малой степени, проходящую через каждую вершину куба, но не пересекающую данное аффинное подпространство. Вычислительная сложность этого метода зависит от степени гиперповерхности. С другой стороны, этот метод эффективен лишь при условии, что размерность аффинного подпространства достаточно мала. Пусть это подпространство определяется системой из m линейных уравнений. Размерность линейного пространства форм степени d от переменных $x_0, ..., x_n$ равна биномиальному коэффициенту $\binom{n+d}{d}$. Поэтому рассматриваемый метод эффективен в среднем, когда число уравнений удовлетворяет неравенству типа $m \ge n - \sqrt{d(d-1)n - o(n)}$, где через d обозначена степень искомой гиперповерхности. Мы ограничимся случаем $d = 2$, когда гиперповерхностью служит квадрика [16].

Пусть n-мерный куб вложен в проективное пространство, а однородные координаты вершин принадлежат множеству $\{0,1\}$, но ни одна из вершин куба не лежит на бесконечно удаленной гиперплоскости $x_0 = 0$. Известно [24], что квадрика, проходящая через каждую вершину этого куба, определяется квадратичной формой типа

$$\lambda_1 x_1(x_1 - x_0) + \cdots + \lambda_n x_n(x_n - x_0).$$

Если подпространство задано системой уравнений $x_j = \ell_j(x_0,\ldots,x_{n-m})$, где $n-m < j \le n$, то существование квадрики, проходящей через каждую вершину куба и пересекающей это подпространство лишь на бесконечности, эквивалентно разрешимости уравнения

$$\sum_{k=1}^{n-m}\lambda_k x_k(x_k - x_0) + \sum_{j=n-m+1}^{n}\lambda_j \ell_j(\ell_j - x_0) = x_0^2$$

относительно переменных $\lambda_1, ..., \lambda_n$. В свою очередь, эта задача эквивалентна разрешимости системы линейных уравнений. По теореме Кронекера–Капелли, эта проверка сводится к сравнению рангов двух матриц. Алгоритм 1 не обеспечивает полиномиальную вычислительную сложность в худшем случае. Но иногда он гораздо эффективнее полного перебора вершин куба.

В алгоритме 1 первые $n-m$ столбцов матрицы A всегда содержат по два ненулевых элемента 1 и -1. В столбце с номером $j > n-m$ элементы равны многочленам второй степени от коэффициентов линейной формы ℓ_j.

Алгоритм 1. Проверка существования вершины куба, инцидентной данному подпространству.

Input: целые числа $0 < m < n$ и линейные формы $\ell_j(x_0,\ldots,x_{n-m})$, где $n-m < j \le n$.

1: Элементами матрицы A служат коэффициенты формы

$$\sum_{k=1}^{n-m}\lambda_k x_k(x_k - x_0) + \sum_{j=n-m+1}^{n}\lambda_j \ell_j(\ell_j - x_0),$$

где строки матрицы A соответствуют мономам второй степени от переменных x_0, ..., x_{n-m}, а столбцы – переменным $\lambda_1, ..., \lambda_n$;

2: Расширенная матрица B получается из матрицы A добавлением столбца, в котором единица стоит в строке, соответствующей моному x_0^2, а остальные элементы этого столбца равны нулю;

3: **if** $\operatorname{rank}(A) = \operatorname{rank}(B)$ **then** вход отвергается

4: **else** уведомление о неопределенности.

Рассмотрим пример, соответствующий прямой на плоскости. Пусть $n = 2$, $m = 1$, линейная форма $\ell_2 = x_0 + x_1$. Получаем уравнение

$$-\lambda_1 x_0 x_1 + \lambda_1 x_1^2 + \lambda_2 x_0 x_1 + \lambda_2 x_1^2 = x_0^2.$$

Здесь три монома от переменных x_0 и x_1. Это x_0^2, x_0x_1 и x_1^2. Соответствующая матрица A содержит два столбца и три строки:

$$A = \begin{pmatrix} 0 & 0 \\ -1 & 1 \\ 1 & 1 \end{pmatrix}.$$

Расширенная матрица равна

$$B = \begin{pmatrix} 0 & 0 & 1 \\ -1 & 1 & 0 \\ 1 & 1 & 0 \end{pmatrix}.$$

Здесь $\mathrm{rank}(A) = 2$ и $\mathrm{rank}(B) = 3$. Поэтому искомые λ_1 и λ_2 не существуют. Ответ алгоритма 1 состоит в уведомлении о неопределенности.

Рассмотрим другой пример, соответствующий прямой в пространстве. Пусть $n = 3$, $m = 2$, линейные формы $\ell_2 = 2x_0 + x_1$ и $\ell_3 = 3x_1$. Получаем уравнение $\lambda_1(-x_0x_1 + x_1^2) + \lambda_2(2x_0^2 + 3x_0x_1 + x_1^2) + \lambda_3(-3x_0x_1 + 9x_1^2) = x_0^2$. Здесь снова три монома от переменных x_0 и x_1. Это x_0^2, x_0x_1 и x_1^2. Но матрица A содержит три столбца и три строки:

$$A = \begin{pmatrix} 0 & 2 & 0 \\ -1 & 3 & -1 \\ 1 & 1 & 9 \end{pmatrix}.$$

Расширенная матрица равна

$$B = \begin{pmatrix} 0 & 2 & 0 & 1 \\ -1 & 3 & -3 & 0 \\ 1 & 1 & 9 & 0 \end{pmatrix}.$$

Матрица A невырожденная, ранги матриц A и B совпадают. Алгоритм 1 отвергает вход.

Следующая теорема обеспечивает достаточное условие типа $m \ge n - \sqrt{2n - o(n)}$, при котором алгоритм 1 отвергает большую долю входов при фиксированных значениях n и m. Доказательство основано на построении базиса специального вида в пространстве квадратичных форм.

Теорема 2. *Даны положительные целые числа n и $m < n$, для которых выполнено неравенство $2n \ge (n - m + 1)(n - m + 2)$. Для почти каждого набора из m линейных форм $\ell_j(x_0, \ldots, x_{n-m})$, где $n - m < j \le n$, найдутся значения коэффициентов $\lambda_1, \ldots, \lambda_n$, для которых выполнено равенство квадратичных форм*

$$\sum_{k=1}^{n-m} \lambda_k x_k (x_k - x_0) + \sum_{j=n-m+1}^{n} \lambda_j \ell_j (\ell_j - x_0) = x_0^2.$$

Если для некоторого $\varepsilon > 0$ коэффициенты линейных форм ℓ_j независимо и равномерно распределены на множестве из $\lceil 2n/\varepsilon \rceil$ чисел, то вероятность отсутствия искомых значений $\lambda_1, \ldots, \lambda_n$ не превышает ε.

Доказательство. Рассмотрим матрицу A из алгоритма 1. По условию теоремы, в матрице A число строк $r = (n - m + 1)(n - m + 2)/2$ не превышает числа столбцов n. В этом случае достаточным условием существования искомых $\lambda_1, \ldots, \lambda_n$ служит равенство $\mathrm{rank}(A) = r$. Каждый элемент матрицы A равен либо константе, либо многочлену второй степени от коэффициентов линейных форм ℓ_j. Следовательно, минор порядка r матрицы A равен многочлену от коэффициентов линейных форм ℓ_j, степень которого не превышает $2r$.

По лемме Шварца–Зиппеля [25], если этот многочлен не равен нулю тождественно, то вероятность обращения в нуль не превышает $2r/\lceil 2n/\varepsilon \rceil \le \varepsilon$.

Осталось показать, что для некоторого набора форм ℓ_j угловой минор порядка r матрицы A отличен от нуля. При этом достаточно рассмотреть набор форм, коэффициенты которых не обязательно принадлежат указанному в условии множеству. Положим $\ell_{n-m+k} = x_0 + x_k$ для $1 \le k \le n - m$. Обозначим через $f(i,k)$ номер пары индексов $1 \le i < k \le n - m$, принимающий значения от 1 до $(n - m)(n - m - 1)/2$. Положим $\ell_j = x_i + x_k$ для $j = 2(n - m) + f(i,k)$. При $j = r$ полагаем $\ell_r = 2x_0$.

Для индексов $1 \le i < k \le n - m$ моному x_ix_k соответствует строка, где отличен от нуля лишь элемент в столбце $j = 2(n - m) + f(i,k)$. Этот элемент равен двум. С другой стороны, в столбцах с номерами $1 \le j \le n - m$ по два ненулевых элемента: -1 в строке, соответствующей моному x_0x_j, и 1 в строке, соответствующей моному x_j^2. Переставим строки в матрице A в соответствии с таким порядком мономов: $x_0x_1, \ldots, x_0x_{n-m}, x_1^2, x_2^2, \ldots, x_{n-m}^2$, мономы x_ix_k для $1 \le i < k \le n - m$ и последний x_0^2. Угловая подматрица порядка r примет вид

$$\begin{pmatrix} -I & I & * & 0 \\ I & I & * & 0 \\ 0 & 0 & 2I' & 0 \\ 0 & 0 & 0 & 2 \end{pmatrix},$$

где через I обозначена единичная матрица порядка $n - m$, через I' – единичная матрица порядка $(n - m)(n - m - 1)/2$, а через 0 – нулевая матрица. Эта подматрица невырожденная, поскольку элементарные преобразования строк дают треугольную матрицу с ненулевыми элементами на главной диагонали

$$\begin{pmatrix} -I & I & * & 0 \\ 0 & 2I & * & 0 \\ 0 & 0 & 2I' & 0 \\ 0 & 0 & 0 & 2 \end{pmatrix}.$$

Следовательно, $\mathrm{rank}(A) = r$. □

Заметим, что в доказательстве теоремы 2 используется семейство разреженных матриц. Поэтому разреженность матрицы A не служит препятствием для успешного применения обсуждаемого алгоритма. Более того, для разреженных матриц меньше сложность вычисления ранга, что

позволяет работать в больших размерностях. В доказательстве теоремы 2 рассмотрен частный случай, когда матрица A имеет полный ранг. Поэтому вычисление ранга можно заменить на проверку невырожденности матрицы A. Однако это было бы неоправданным ограничением применимости алгоритма 1.

3. РЕАЛИЗАЦИЯ

Обозначим через L матрицу, составленную из коэффициентов линейных форм ℓ_j в алгоритме 1 и теореме 2. Первая строка матрицы L соответствует форме ℓ_{n-m+1}, а последняя строка – форме ℓ_n. Например, для двух линейных форм $\ell_2 = 2x_0 + x_1$ и $\ell_3 = 3x_1$ эта матрица равна

$$L = \begin{pmatrix} 2 & 1 \\ 0 & 3 \end{pmatrix}.$$

Сопоставление матрице L матрицы B требует меньше времени, чем вычисление рангов матриц A и B. Такое преобразование реализует функция `compose_b()`, которая получает на вход матрицу L и возвращает матрицу B, где обе матрицы представлены массивами NumPy. Порядок строк в B соответствует лексикографическому мономиальному упорядочению, а первая строка матрицы – моному x_0^2. Матрица A легко получается удалением последнего столбца из матрицы B. Функция `compose_b()` реализована на языке Python с использованием библиотеки NumPy.

При использовании целых чисел фиксированной битовой длины, NumPy позволяет значительно повысить эффективность работы с матрицами (как по скорости, так и по памяти) по сравнению со встроенными типами данных языка Python, например списками чисел [26]. При работе с другими типами данных, включая целые неограниченной длины и рациональные числа, эти преимущества в основном утрачиваются. Однако в любом случае NumPy позволяет кратко записывать стандартные операции над матрицами и их частями, например, как в нашем случае, выделение подматрицы, поэлементное сложение и умножение строк и т. п.

Через $s = n - m$ обозначена размерность подпространства, для которого проверяется инцидентность вершине n-мерного куба. Через `height` обозначено число мономов второй степени от переменных $x_0, \ldots, x_s$, равное $(s+1)(s+2)/2$.

Для оценки времени работы функции `compose_b()` при n = `height` генерировалась $(n-s)\times(s+1)$ матрица L, элементами которой служат случайно выбранные целые числа из отрезка $[-10^9, 10^9]$. Время, затраченное на вычисление матрицы B, показано в таблице 1. В третьем столбце таблицы 1 указаны результаты для 64-битовых целых чисел, а в последнем столбце – для целых чисел неограниченной битовой длины, вычисления над которыми реализованы в языке Python. Мы использовали версии Python 3.10.4 и NumPy 1.22.4. Вычисления проведены на персональном компьютере на базе процессора Intel® Core i5-3570 и с оперативной памятью 16 гигабайтов. Поэтому для больших матриц вычисления с числами неограниченной битовой длины не были проведены из-за переполнения оперативной памяти, хотя для 64-битовых чисел вычисления продолжены вплоть до значений $s = 300$.

Одновременное вычисление рангов матриц A и B по данной на вход матрице B также реализовано на языке Python. Для вычислений с рациональными числами используется класс Fraction из стандартной библиотеки Python.

Вычисление ранга основано на приведении матрицы к ступенчатому виду. Поскольку матрица A служит подматрицей в B, для обеих матриц A и B такое преобразование можно сделать одновременно. Это позволяет примерно вдвое уменьшить время работы по сравнению с независимым вычислением рангов двух матриц.

Чтобы оценить время для вычисления функции `rank_ab()`, при n = `height` генерировались $(n-s)\times(s+1)$ матрицы L, элементами которых служат случайно выбранные целые числа из отрезка $[-N, N]$. Время вычисления для разных значений границы диапазона $N \in \{10^3, 10^6, 10^9\}$ показано в таблице 2. Чем больше размер матрицы, тем быстрее возрастает время работы при увеличении средней битовой длины элементов матрицы.

Отметим, что реализованный в библиотеке NumPy метод вычисления ранга, основанный на сингулярном разложении матрицы, в этой задаче применять небезопасно из-за возможных ошибок, связанных с использованием чисел с плавающей запятой. Например, при значениях $k \geq 16$ вычисляемый в NumPy 1.22.4 таким неточным способом ранг диагональной 2×2 матрицы с элементами 1 и 10^k на главной диагонали оказывается равным единице. Поэтому для вычисления ранга мы использовали новую реализацию известного алгоритма Гаусса приведения матрицы к ступенчатому виду, которая позволяет работать с числами неограниченной битовой длины.

В итоге реализацией алгоритма 1 служит функция `may_have_01solution()`, которая получает на вход матрицу L. Значение равно `True`, когда алгоритм 1 дал бы неопределенный ответ. Значение `False` означает, что алгоритм 1 отвергает вход. Входом может служить матрица с рациональными элементами.

Таблица 1. Время в секундах для вычисления матрицы B при различных значениях s и n для двух типов целых чисел: 64-битовых и неограниченной битовой длины

Параметры		Тип целых чисел	
s	n	64-битовые	без ограничений
20	231	0.01	0.02
30	496	0.03	0.08
40	861	0.07	0.20
50	1326	0.14	0.50
60	1891	0.25	1.00
70	2556	0.43	1.90
80	3321	0.68	3.20
90	4186	1.10	5.00
100	5151	1.50	8.20
110	6216	2.20	12
120	7381	3.00	16
130	8646	4.20	24
140	10011	5.40	36
150	11476	7.50	51
160	13041	9.50	66
170	14706	13	89
180	16471	15	
190	18336	20	
200	20301	23	
210	22366	29	
220	24531	34	
230	26796	43	
240	29161	50	
250	31626	62	
260	34191	83	
270	36856	89	
280	39621	100	
290	42486	130	
300	45451	170	

Листинги обсуждаемых функций и примеры их использования доступны по адресу http://lab6.iitp.ru/-/qualg

4. ЗАКЛЮЧЕНИЕ

Реализован на языке Python эвристический алгоритм полиномиального времени для распознавания подпространств достаточно малой размерности, инцидентных какой-либо из вершин единичного многомерного куба, хотя в худшем случае эта задача считается вычислительно трудной. Алгоритм либо корректно отвергает вход, либо сообщает об отсутствии очевидного препятствия к инцидентности. В частности, полученные ограничения на сложность этой задачи могут быть полезны для оценки надежности криптографических протоколов, основанных на поиске (0, 1)-решения системы уравнений [19, 27].

Таблица 2. Время в секундах для вычисления ранга матрицы B при различных значениях параметров s, n и N

Параметры		N		
s	n	10^3	10^6	10^9
5	21	0.02	0.02	0.02
6	28	0.04	0.05	0.07
7	36	0.09	0.10	0.19
8	45	0.21	0.34	0.50
9	55	0.45	0.77	1.20
10	66	0.91	1.70	2.70
11	78	1.80	3.50	5.70
12	91	3.30	6.90	12
13	105	6	13	23
14	120	11	24	42
15	136	18	42	76
16	153	30	73	130
17	171	50	120	230
18	190	79	200	380
19	210	120	330	610
20	231	190	510	960

Алгоритм использует вычисление ранга матрицы. Полученные в теореме 1 оценки ранга неразложимой разреженной матрицы позволяют, в частности, тестировать вычисление ранга матрицы большого порядка.

Авторы благодарны анонимному рецензенту за сделанные замечания, позволившие улучшить статью.

СПИСОК ЛИТЕРАТУРЫ

1. *Геворкян М.Н., Королькова А.В., Кулябов Д.С., Севастьянов Л.А.* Пример модульного расширения системы компьютерной алгебры // Программирование. 2020. № 2. С. 30–37. DOI: Перевод: Programming and Computer Software. 2020. V. 46. № 2. P. 98–104. https://doi.org/10.31857/S0132347420020065
2. *Chistov A.L.* Fast parallel calculation of the rank of matrices over a field of arbitrary characteristic // In: *L. Budach* (eds) Fundamentals of Computation Theory. FCT 1985. Lecture Notes in Computer Science, vol. 199. Springer, Berlin, Heidelberg, 1985. P. 63–69. https://doi.org/10.1007/BFb0028792
3. *Mulmuley K.* A fast parallel algorithm to compute the rank of a matrix over an arbitrary field // Combinatori-

ca. 1987. V. 7. № 1. P. 101–104.
https://doi.org/10.1007/BF02579205

4. *Malaschonok G., Tchaikovsky I.* About big matrix inversion // In: *Abramov S.A., Sevastyanov L.A.* (eds) Computer algebra. Moscow: MAKS Press, 2021. P. 81–84.
https://doi.org/10.29003/m2019.978-5-317-06623-9

5. *Малашонок Г.И., Сидько А.А.* Суперкомпьютерная среда выполнения для рекурсивных матричных алгоритмов // Программирование. 2022. № 2. С. 33–46. DOI: Перевод: Programming and Computer Software. 2022. V. 48. P. 90–101.
https://doi.org/10.31857/S0132347422020091

6. *Cheung H.Y., Kwok T.C., Lau L.C.* Fast matrix rank algorithms and applications // Journal of the ACM. 2013. V. 60. № 5. Article № 31. P. 1–25.
https://doi.org/10.1145/2528404

7. *Abdeljaoued J., Malaschonok G.I.* Efficient algorithms for computing the characteristic polynomial in a domain // Journal of Pure and Applied Algebra. 2001. V. 156. P. 127–145.
https://doi.org/10.1016/S0022-4049(99)00158-9

8. *Переславцева О.Н.* О вычислении характеристического полинома матрицы // Дискретная математика. 2011. Т. 23. № 1. С. 28–45. DOI: Перевод: Discrete Mathematics and Applications. 2011. V. 21. № 1. P. 109–128.
https://doi.org/10.4213/dm1128

9. *Neiger V., Pernet C.* Deterministic computation of the characteristic polynomial in the time of matrix multiplication // Journal of Complexity. 2021. V. 67. № 101572. P. 1–35.
https://doi.org/10.1016/j.jco.2021.101572

10. *Chen Z.* On nonsingularity of circulant matrices // Linear Algebra and its Applications. 2021. V. 612. P. 162–176.
https://doi.org/10.1016/j.laa.2020.12.010

11. *Алаев П.Е., Селиванов В.Л.* Поля алгебраических чисел, вычислимые за полиномиальное время. I // Алгебра и логика. 2019. Т. 58. № 6. С. 673–705. DOI: Перевод: Algebra Logic. 2020. V. 58. P. 447–469.
https://doi.org/10.33048/alglog.2019.58.601

12. *Алаев П.Е., Селиванов В.Л.* Поля алгебраических чисел, вычислимые за полиномиальное время. II // Алгебра и логика. 2021. Т. 60. № 6. С. 533–548. DOI: Перевод: Algebra Logic. 2022. V. 60. P. 349–359.
https://doi.org/10.33048/alglog.2021.60.601

13. *Harris J., Tu L.W.* On symmetric and skew-symmetric determinantal varieties // Topology. 1984. V. 23. № 1. P. 71–84.
https://doi.org/10.1016/0040-9383(84)90026-0

14. *Harris J.* Algebraic geometry. Springer-Verlag New York, 1992. 330 p.
https://doi.org/10.1007/978-1-4757-2189-8

15. *Rubei E.* Affine subspaces of matrices with constant rank // Linear Algebra and its Applications. 2022. V. 644. № 1. P. 259–269.
https://doi.org/10.1016/j.laa.2022.03.002

16. *Селиверстов А.В.* Двоичные решения для больших систем линейных уравнений // Прикладная дискретная математика. 2021. № 52. С. 5–15.
https://doi.org/10.17223/20710410/52/1

17. *Кузюрин Н.Н.* Полиномиальный в среднем алгоритм в целочисленном линейном программировании // Сибирский журнал исследования операций. 1994. Т. 1. № 3. С. 38–48.

18. *Kuzyurin N.N.* An integer linear programming algorithm polynomial in the average case // In: *Korshunov A.D.* (eds.) Discrete Analysis and Operations Research. Mathematics and Its Applications. V. 355. P. 143–152. Springer, Dordrecht, 1996.
https://doi.org/10.1007/978-94-009-1606-7

19. *Pan Y., Zhang F.* Solving low-density multiple subset sum problems with SVP oracle // Journal of Systems Science and Complexity. 2016. V. 29. P. 228–242.
https://doi.org/10.1007/s11424-015-3324-9

20. *Рыбалов А.Н.* О генерической сложности проблемы о сумме подмножеств для полугрупп целочисленных матриц // Прикладная дискретная математика. 2020. № 50. С. 118–126.
https://doi.org/10.17223/20710410/50/9

21. *Рыбалов А.Н.* О генерической сложности проблемы вхождения для полугрупп целочисленных матриц // Прикладная дискретная математика. 2022. № 55. С. 95–101.
https://doi.org/10.17223/20710410/55/7

22. *Селиверстов А.В.* Эвристические алгоритмы распознавания некоторых кубических гиперповерхностей // Программирование. 2021. № 1. С. 65–72. DOI: Перевод: Programming and Computer Software. 2021. V. 47. № 1. P. 50–55.
https://doi.org/10.31857/S0132347421010106

23. *Minc H.* (0,1)-matrices with minimal permanents // Israel Journal of Mathematics. 1973. V. 15. P. 27–30.
https://doi.org/10.1007/BF02771770

24. *Seliverstov A.V., Lyubetsky V.A.* About forms equal to zero at each vertex of a cube // Journal of Communications Technology and Electronics. 2012. V. 57. № 8. P. 892–895.
https://doi.org/10.1134/S1064226912080049

25. *Schwartz J.T.* Fast probabilistic algorithms for verification of polynomial identities // Journal of the ACM. 1980. V. 27. № 4. P. 701–717.
https://doi.org/10.1145/322217.322225

26. *Harris C.R., Millman K.J., van der Walt S.J. et al.* Array programming with NumPy // Nature. 2020. V. 585. № 7825. P. 357–362.
https://doi.org/10.1038/s41586-020-2649-2

27. *Chen Y.A., Gao X.S.* Quantum algorithm for Boolean equation solving and quantum algebraic attack on cryptosystems // Journal of Systems Science and Complexity. 2022. V. 35. P. 373–412.
https://doi.org/10.1007/s11424-020-0028-6

Effective Lower Bounds on the Matrix Rank and Their Applications

© 2023 г. O. A. Zverkov[a,#], and A. V. Seliverstov[a,##]

[a] *Institute for Information Transmission Problems (Kharkevich Institute), Russian Academy of Sciences, Bol'shoi Karetnyi per. 19/1, Moscow, 127051 Russia*

[#] *e-mail: zverkov@iitp.ru*

[##] *e-mail: slvstv@iitp.ru*

We propose an efficiently verifiable lower bound on the rank of a sparse fully indecomposable square matrix that contains two non-zero entries in each row and each column. The rank of this matrix is equal to its order or differs from it by one. Bases of a special type are constructed in the spaces of quadratic forms in a fixed number of variables. The existence of these bases allows us to substantiate a heuristic algorithm for recognizing whether a given affine subspace passes through a vertex of a multidimensional unit cube. In the worst case, the algorithm may output a computation denial warning; however, for the general subspace of sufficiently small dimension, it correctly rejects the input. The algorithm is implemented in Python. The running time of its implementation is estimated in the process of testing.