Абрамов С. А., Баркату М. А.

О дифференциальной сложности вычисления ранга матрицы над кольцом скалярных дифференциальных операторов

Аннотация

Рассматривается один из известных алгоритмов (а именно, алгоритм RowReduction), позволяющих вычислить ранг матрицы с элементами в кольце скалярных линейных дифференциальных операторов. При обсуждении ранга под линейной зависимостью строк или столбцов понимается зависимость над тем же кольцом операторов. Для названного алгоритма известны оценки его дифференциальной сложности, т.е. числа дифференцирований элементов, требующихся алгоритму в худшем случае. Дается модификация этого алгоритма, предполагающая запоминание всех результатов дифференцирования. Рассматривается вопрос о том, в какой мере это запоминание и возможность многократного использования результатов дифференцирования (не предусматриваемого исходной версией алгоритма Row Reduction) понижает дифференциальную сложность.

1. Терминология и обозначения

Ниже кольцо $n \times n$-матриц с элементами в некотором кольце или поле R обозначается через $\mathrm{Mat}_n(R)$. Обозначение $M_{i,*}$ привлекается для матрицы размера $1 \times n$, равной i-й строке матрицы M, имеющей n столбцов. Единичная $n \times n$-матрица обозначается через I_n.

Кольцо скалярных дифференциальных операторов с коэффициентами в дифференциальном поле K с дифференцированием $\partial =' $ обозначается посредством $K[\partial]$; порядок $\mathrm{ord} l$ оператора $l \in K[\partial]$ равен степени l как полинома из $K[\partial]$. Такие операторы-полиномы складываются как обычные полиномы, для умножения действуют законы ассоциативности и дистрибутивности, а также перестановочное правило $\partial a = a\partial + a' = a\partial + \partial(a)$.

Матрицы, не обязательно квадратные, с принадлежащими $K[\partial]$ элементами, мы называем *операторными матрицами*. В тех случаях, когда это не вызывает недоразумений, мы будем использовать термин *оператор* для принадлежащих $\mathrm{Mat}_n(K[\partial])$ (т.е. для квадратных) операторных матриц и термин "матрица" для матриц с элементами из исходного дифференциального поля K.

Оператор L как элемент кольца $\mathrm{Mat}_n(K[\partial])$ имеет вид

$$\begin{pmatrix} L_{11} & \dots & L_{1n} \\ \dots & \dots & \dots \\ L_{n1} & \dots & L_{nn} \end{pmatrix}, \tag{1}$$

$L_{ij} \in K[\partial]$, $i, j = 1, \dots, n$, при этом $d = \max_{i,j} \mathrm{ord} L_{ij}$ — это *порядок* L, мы пишем $\mathrm{ord} L = d$. Этот оператор может также быть представлен в *развернутом* виде

$$A_d \partial^{\,d} + A_{d-1} \partial^{\,d-1} + \dots + A_0, \tag{2}$$

где матрицы

$$A_0, A_1, \dots, A_d \tag{3}$$

принадлежат $\mathrm{Mat}_n(K)$, при этом *ведущая* матрица A_d предполагается ненулевой; *трейлинговая* матрица A_0 ненулевой быть не обязана. При $n = 1$ оператор становится скалярным оператором (имеющим вид полинома от ∂).

2. Ранг оператора; алгоритм RR

Определение 1. Строки $u_1, \dots, u_s$ одинаковой длины, элементы которых принадлежат $K[\partial]$, называются *линейно зависимыми* (более подробно: *линейно зависимыми над* $K[\partial]$), если существуют такие одновременно не равные нулю $f_1, \dots, f_s \in K[\partial]$, что $f_1 u_1 + \dots + f_s u_s = 0$; в противном случае эти строки называются *линейно независимыми* (над $K[\partial]$). Сходным образом определяется линейная независимость столбцов, но в этом случае $f_1, \dots, f_s$ используются как правые множители. Для оператора вида (1) определяется *ранг по строкам* как максимальное число линейно независимых строк этого оператора и *ранг по столбцам* как максимальное число линейно независимых столбцов.

Для произвольной операторной матрицы (в частности, для произвольной квадратной матрицы, т.е. для оператора) ранги по строкам и столбцам равны ([11]), что позволяет говорить просто о *ранге* операторной матрицы или оператора.

Оператор $L \in \mathrm{Mat}_m(K\,[\partial])$ имеет *полный ранг*, если его строки линейно независимы над $K\,[\partial]$.

Определение 2. Пусть оператор L вида (1) имеет развернутое представление (2). Если $1 \le i \le n$ и строка $L_{i,*}$ ненулевая, то определим α_i как наибольшее целое k, $0 \le k \le d$, при котором строка $(A_k)_{i,*}$, ненулевая. Для нулевой строки $L_{i,*}$ полагаем $\alpha_i = 0$. Число α_i будем называть *порядком i-й строки* оператора L. Матрица $M \in \mathrm{Mat}_m(K)$, такая, что $M_{i,*} = (A_{\alpha_i})_{i,*}$, $i = 1, \ldots, n$, называется *фронтальной* матрицей оператора L.

Алгоритм RowReduction ([7, 8]), который для краткости мы будем называть алгоритмом RR, позволяет вычислить ранг заданного оператора L. Детальное описание этого алгоритма можно найти в [8, разд. 2.1].

Алгоритм предпринимает перевод исходного ненулевого оператора L в оператор $\breve{L}$ посредством последовательности "элементарных" преобразований, сохраняющих ранг. Этими преобразованиями достигается то, что ненулевые строки фронтальной матрицы оператора $\breve{L}$ оказываются линейно независимыми над K. Это гарантирует равенство ранга исходного оператора числу этих строк.

Опишем здесь этот алгоритм вкратце. Оператор $\breve{L}$ строится на месте оператора L, т.е. L изменяется шаг за шагом, постепенно преобразуясь в $\breve{L}$:

> Проверить, являются ли ненулевые строки фронтальной матрицы оператора L линейно зависимыми над K. Если нет, то $\breve{L} = L$, ранг L равен числу ненулевых строк текущей фронтальной матрицы, и алгоритм останавливается. В противном случае пусть $p_1, \ldots, p_n \in K$ — коэффициенты зависимости (при этом, для нулевых строк фронтальной матрицы соответствующие p_j считаем равными нулю). Из тех строк (1), которым соответствуют ненулевые коэффициенты, выбрать строку, имеющую наибольший порядок (если

имеется несколько строк с таким порядком, то взять любую из них). Пусть это будет строка $L_{i,*}$ оператора L. Она заменяется на

$$\sum_{j=1}^{n} p_j \partial^{\alpha_i - \alpha_j} L_{j,*} \qquad (3).$$

После конечного числа таких шагов получаем оператор, ненулевые строки фронтальной матрицы которого линейно независимы над K. Следовательно, ненулевые строки самого полученного оператора линейно независимы над $K[\partial]$.

3. Алгоритм $\overline{\mathrm{RR}}$. Оценки сложности

Нетрудно заметить, что при вычислении сумм (3) алгоритм RR требует большого числа дифференцирований строк оператора и, тем самым, большого числа дифференцирований элементов поля K. Чтобы его уменьшить, можно сохранять результаты всех дифференцирований строк. Разумеется, это увеличит пространственную сложность алгоритма, но, можно надеяться, что его дифференциальная сложность снизится. Мы показываем, что снижение дифференциальной сложности здесь не слишком большое. Алгоритм с хранением результатов дифференцирования мы будем обозначать посредством $\overline{\mathrm{RR}}$. Ситуация с алгоритмами RR и $\overline{\mathrm{RR}}$ такова (n и d задают размер, т.е. число строк, и порядок оператора, — это такие же величины, как в (1), (2)):

– Легко проверяется, что если не прибегать к хранению результатов дифференцирования, то дифференциальная сложность допускает оценку $\Theta(n^3 d^3)$.

– Если результаты дифференцирования сохраняются, то, как было показано в [2], дифференциальная сложность допускает оценку $O(n^2 d^3)$. В [2] при этом не утверждалось, что эта оценка точна и ставился вопрос о том, можно ли в оценке этой сложности понизить показатель 3 для d до 2?

– Показатель 2 для n и показатель 3 для d не могут быть понижены для дифференциальной сложности $\overline{\mathrm{RR}}$: нами доказана оценка $\Omega(n^2 d^{\,3})$.

– Из оценки $\Omega(n^2d^3)$ и доказанной ранее оценки $O(n^2d^3)$ вытекает, что для рассматриваемой сложности имеет место оценка $\Theta(n^2d^3)$.[1]

Основное утверждение:

Теорема. *Дифференциальная сложность алгоритма* $\overline{\mathrm{RR}}$ *есть* $\Theta(n^2d^3)$.

4. О разностном случае

4.1. Десингуляризация ведущей матрицы

Пусть K — разностное поле с автоморфизмом (сдвигом) σ. Кольцо скалярных разностных операторов — это кольцо $K[\sigma, \sigma^{-1}]$. Несходство разностного и дифференциального случаев заметно в том, что автоморфизм σ имеет обратный в $K[\sigma, \sigma^{-1}]$, тогда как дифференцирование ∂ не обратимо в $K[\partial]$. Алгоритм RR вычисляет $\breve{L}$ вида UL, где $\breve{L}$ имеет невырожденную фронтальную матрицу, при этом оператор U обратим в $K[\partial]$ и оба оператора U, U^{-1} не содержат ∂^{-1}. В разностном случае U^{-1} может содержать σ^{-1}. Вычисление такого U может иметь меньшую сложность. Та версия алгоритма EG-исключений [1, 4, 5], которая изложена в [1], вычисляет в разностном случае эквивалентный исходному оператор (т.е. некоторый оператор $\breve{L}$, который может быть представлен в виде $\tilde{U}L$, где оператор $\tilde{U}$ обратим в $\mathrm{Mat}_n(K[\sigma, \sigma^{-1}])$), имеющий треугольную ведущую матрицу, — аналогичная идея была независимо использована в [10] для построения так называемой слабой формы Попова полиномиальной матрицы. Сдвиговая сложность EG-исключений есть $O(n^2d^2)$ (см., например, [3, 6]).

Заметим, что это соображение не работает в дифференциальном случае: если мы используем здесь EG-исключения, то $\tilde{U}$ не является, вообще говоря, обратимым в $\mathrm{Mat}_n(K[\partial])$.

Нелишне отметить, что при использовании разностной версии алгоритма RR мы получаем сдвиговую сложность $\Theta(n^2d^3)$.

[1] В асимптотических оценках мы, наряду с O-нотацией, используем также Ω- и Θ- нотации (см. [9]).

4.2. Об арифметической сложности

Арифметическая сложность (т.е. сложность по числу арифметических операций в поле K) алгоритма RR есть $\Theta(n^{\omega+1}d^{\,2})$, где ω — показатель матричного умножения, $2 < \omega \leq 3$. Что касается варианта EG-исключений, который упоминался выше, его арифметическая сложность есть $O(n^3d^{\,2})$. Но можно использовать для RR тот же подход к уменьшению арифметической сложности, что и для EG-исключений (см., например, [3]), получая оператор с треугольной фронтальной матрицей. Тогда получим арифметическую сложность $O(n^3d^{\,2})$, как для EG-исключений.

Благодарность

Первый автор благодарит РФФИ за частичную финансовую поддержку, грант 16-01-00174-а.

Литература

1. Abramov S.A. EG–eliminations. *J. of Difference Equations and Applications*, 5(4-5): 393–433, 1999.
2. Abramov S.A. On the differential and full algebraic complexities of operator matrices transformations. In Proc. of CASC'2016, 1–11, 2016.
3. Abramov S.A. Inverse Linear Difference Operators. *Computational Mathematics and Mathematical Physics*, 57, 1887–1898, 2017.
4. Abramov S.A., Bronstein M. On solutions of linear functional systems. *Proc. ISSAC'2001*: 1–6, 2001.
5. Abramov S.A., Bronstein M. Linear algebra for skew-polynomial matrices. Rapport de Recherche INRIA RR-4420, March 2002 `http://www.inria.fr/RRRT/RR-4420.html`
6. Abramov S.A., Khmelnov D.E. On unimodular matrices of difference operators. *CASC 2018 (accepted).*
7. Beckermann B., Cheng H., Labahn G. Fraction-free row reduction of matrices of Ore polynomials. J. Symbolic Comput. **41** (2006),

513–543.

8. Barkatou M.A., El Bacha C., Labahn G., Pflügel E. On simultaneously row and column reduction of higher-order linear differential systems. *J. of Symbolic Comput.*, 49(1): 45–64, 2013.
9. Knuth D.E. Big omicron and big omega and big theta. *ACM SIGACT News*, 8(2): 18–23, 1976.
10. Mulders T., Storjohann A.: On Lattice Reduction for Polynomial Matrices. *J. Symb. Comput.*, 37(4): 485–510, 2004.
11. Кон П. Свободные кольца и их связи. М.: Мир, 1975.

Сведения об авторах

Абрамов Сергей Александрович, д.ф.-м.н.,

Вычислительный центр им. Дородницына,

федеральный исследовательский центр “Информатика и управление”,

гл. научн. сотр.;

sergeybramov@mail.ru

Баркату Мулей Абдельфаттах, д.н.,

Лиможский университет (Франция), профессор,

moulay.barkatou@unilim.fr