Особые точки решений линейных обыкновенных дифференциальных систем с полиномиальными коэффициентами*

С. А. АБРАМОВ
Вычислительный центр РАН
e-mail: sergeyabramov@mail.ru

Д. Е. ХМЕЛЬНОВ
Вычислительный центр РАН
e-mail: dennis_khmelnov@mail.ru

УДК 512.628.2

Ключевые слова: компьютерная алгебра, системы дифференциальных уравнений, формальные ряды Лорана, особые точки решений, выявляющий полином, выявляющее преобразование, рациональные решения.

Аннотация

Рассматриваются системы линейных обыкновенных дифференциальных уравнений относительно m неизвестных функций от x. Коэффициенты систем являются полиномами над полем k характеристики 0. Каждая из рассматриваемых систем состоит из m независимых над $k[x, d/dx]$ уравнений, порядки которых произвольны. Предлагается компьютерно-алгебраический алгоритм, который по заданной системе S рассматриваемого вида находит такой полином $d(x) \in k[x] \setminus \{0\}$, что если при некотором $\alpha \in \bar{k}$ система S обладает в $\bar{k}((x-\alpha))^m$ решением и какая-то из компонент этого решения имеет ненулевую полярную часть, то $d(\alpha) = 0$. Если $k \subseteq \mathbb{C}$ и система обладает аналитическим решением с особенностью любого вида (не обязательно полюсом) в α, то равенство $d(\alpha) = 0$ также выполняется.

Abstract

S. A. Abramov, D. E. Khmelnov, On singular points of solutions of linear differential systems with polynomial coefficients, Fundamentalnaya i prikladnaya matematika, vol. 17 (2011/2012), no. 1, pp. 3—21.

We consider systems of linear ordinary differential equations containing m unknown functions of a single variable x. The coefficients of the systems are polynomials over a field k of characteristic 0. Each of the systems consists of m equations independent over $k[x, d/dx]$. The equations are of arbitrary orders. We propose a computer algebra algorithm that, given a system S of this form, constructs a polynomial $d(x) \in k[x] \setminus \{0\}$ such that if S possesses a solution in $\bar{k}((x-\alpha))^m$ for some $\alpha \in \bar{k}$ and a component of this solution has a nonzero polar part, then $d(\alpha) = 0$. In the case where $k \subseteq \mathbb{C}$ and S possesses an analytic solution having a singularity of an arbitrary type (not necessarily a pole) at α, the equality $d(\alpha) = 0$ is also satisfied.

*Исследование частично поддержано РФФИ, грант 10-01-00249-а.

Фундаментальная и прикладная математика, 2011/2012, том 17, № 1, с. 3—21.

1. Введение

Линейные дифференциальные уравнения с переменными коэффициентами и системы таких уравнений встречаются во многих областях математики. Решение систем связано со специфическими сложностями, которые не встречаются в скалярном случае. Рассмотрим уравнение вида

$$P_r(x)y^{(r)} + P_{r-1}(x)y^{(r-1)} + \ldots + P_0(x)y = 0. \tag{1}$$

Сначала предположим, что это скалярное уравнение, где коэффициенты $P_0(x), P_1(x), \ldots, P_r(x)$ — полиномы над полем k характеристики ноль и коэффициент $P_r(x)$ не является тождественно нулевым. Хорошо известно, что если решение уравнения (1) имеет особенность в некоторой точке α (например, это решение является формальным рядом Лорана по степеням $x - \alpha$ с ненулевой полярной частью), то $P_r(\alpha) = 0$. Полином $P_r(x)$ имеет конечное число корней, и все они могут быть исследованы шаг за шагом. Если же (1) является системой, $y = (y_1, y_2, \ldots, y_m)^{\mathrm{T}}$ — вектор-столбец неизвестных функций от x и

$$P_0(x), P_1(x), \ldots, P_r(x) - \tag{2}$$

квадратные матрицы размера $m \times m$ с элементами в $k[x]$, то роль, которую в скалярном случае играли корни полиномиального коэффициента при $y^{(r)}(x)$, теперь могут играть корни определителя *ведущей* матрицы $P_r(x)$, если этот определитель не равен тождественно нулю. Но если он тождественно равен нулю, то ситуация усложняется. Именно она и рассматривается в нашей статье.

В предположении, что уравнения системы независимы над $k[x, d/dx]$, решается задача построения *выявляющего* полинома для данной системы S вида (1), т. е. некоторого полинома $d(x)$, такого что если, например, при каком-нибудь $\alpha \in \bar{k}$ система S обладает решением $y(x) \in \bar{k}((x-\alpha))^m$ и какая-то из компонент этого решения имеет ненулевую полярную часть (в этом случае можно говорить, что α является полюсом $y(x)$), то $d(\alpha) = 0$; у этого полинома могут быть и корни, которые не соответствуют никакому такому α, поэтому такой полином можно было бы назвать и *охватывающим*. Если отказаться от условия, что уравнения исходной дифференциальной системы независимы над $k[x, d/dx]$, то можно столкнуться с ситуацией, при которой множество особых точек решений заданной системы бесконечно: например, для системы

$$\begin{pmatrix} 1 & -1 \\ 0 & 0 \end{pmatrix} y'' + \begin{pmatrix} 0 & 0 \\ 1 & -1 \end{pmatrix} y' = 0,$$

$y = (y_1, y_2)^{\mathrm{T}}$, k — произвольное поле характеристики ноль, любая точка является особой для некоторого решения, для которого $y_1 = y_2$ (при этом уравнения системы независимы над k).

Мы начнём с алгоритма EG_δ выполнения *выявляющего преобразования* исходной системы S по отношению к ведущей матрице (раздел 2.1). Этот алгоритм находит систему S' того же вида (для тех же неизвестных функций), определитель ведущей матрицы которой принадлежит $k[x] \setminus \{0\}$, причём множество

решений S' содержит множество решений системы S в качестве подмножества. Алгоритм Syngsys, опираясь на алгоритм EG_δ, строит выявляющий полином для заданной системы S. Мы предлагаем рандомизированные варианты этих двух алгоритмов (раздел 2.4); выясняется, что рандомизация во многих случаях позволяет снизить степень выявляющего полинома.

Операция дифференциального сдвига, которую мы вводим в разделе 2.1, позволяет использовать в алгоритме EG_δ идеи алгоритмов EG и EG′ [1—4]. Задача построения выявляющего полинома не решается непосредственно с помощью самих алгоритмов EG и EG′, так как они предназначены для рекуррентных систем (см. раздел 3.1). Аналогично дело обстоит с алгоритмом из [10]. Алгоритм из [11] применяется к дифференциальным системам вида (1), но с его помощью возможно выявляющее преобразование системы только по отношению к *трейлинговой* матрице (ненулевой матрице $P_i(x)$ с наименьшим значением индекса).

Можно предположить, что задача алгоритмического определения потенциальных особенностей решений систем вида (1), в частности задача построения выявляющего полинома, не рассматривалась в явном виде в литературе, хотя некоторые подходы, вероятно, могут быть извлечены из публикаций [9, 12]. При этом алгоритмы EG_δ и Singsys ориентированы непосредственно на построение выявляющего полинома и основаны на простых вычислениях. Реализация алгоритмов представлена в разделе 2.2.

Если $r = 1$ в (1), то можно указать алгоритм, являющийся альтернативой алгоритму EG_δ. Теоретически система (1) с произвольным r может быть сведена к системе первого порядка. В разделе 2.3 мы показываем, что при больших r алгоритм EG_δ имеет меньшую сложность, чем алгоритм, основанный на переходе к системе первого порядка.

В приложении (раздел 3) рассмотрены некоторые задачи, связанные с алгоритмами EG_δ и Singsys.

Алгоритм EG_δ представляет собой дифференциальный вариант алгоритма EG′. Надо заметить, что название «дифференциальный вариант алгоритма EG′» использовано в [8] для алгоритма решения другой задачи, которая отличается от задачи выявляющего преобразования дифференциальной системы. Мы обсудим это в разделе 3.2 (для алгоритма EG′ в разделе 3.1 введено и использовано новое название: EG_σ). В разделе 3.3 в качестве иллюстрации применения алгоритмов EG_δ и Singsys рассмотрена задача поиска решений в виде рациональных функций для дифференциальных систем.

Если дана неоднородная система с правой частью, принадлежащей $k[x]^m$, то, добавив к $y(x)$ компоненту y_{m+1} со значением 1, $y'_{m+1} = 0$, можно преобразовать данную систему в однородную систему S_1 с числом неизвестных функций и уравнений, равным $m + 1$. Для наших целей достаточно найти выявляющий полином для S_1. Пусть однородная система S_2 получена отбрасыванием правых частей в уравнениях исходной системы. Если уравнения S_2 независимы над $k[x, d/dx]$, то уравнения S_1 также независимы. Исходя из этих соображений, мы ограничиваемся в нашей статье случаем однородной системы.

Краткое изложение результатов этой статьи представлено в [6].

2. Алгоритмы EG_δ и Singsys

Как уже говорилось, мы предполагаем, что уравнения исходной дифференциальной системы независимы над $k[x, d/dx]$.

2.1. Чередование шагов «редукция + дифференциальный сдвиг»

Пусть r, m — произвольные неотрицательные целые числа, а система и соответствующие матрицы $P_0(x), P_1(x), \ldots, P_r(x)$ такие, как в (1) и (2) соответственно. Матрица

$$P(x) = \big(P_r(x) \,|\, P_{r-1}(x) \,|\, \ldots \,|\, P_0(x)\big) \tag{3}$$

называется *явной* матрицей системы, а $P_r(x)$ называется *ведущей частью* явной матрицы.

Пусть i-я строка $P_s(x)$, $0 \leqslant s \leqslant r$, ненулевая и i-е строки $P_{s-1}(x)$, $P_{s-2}(x), \ldots, P_0(x)$ нулевые. Пусть дополнительно t-й элемент, $1 \leqslant t < m$, является последним ненулевым элементом i-й строки $P_s(x)$. Тогда $(r-s)\cdot m + t$ называется *шириной* i-й строки $P(x)$, а последний ненулевой элемент i-й строки — *трейлинговым* элементом этой строки.

Пусть в i-й строке матрицы P есть ненулевые элементы, но i-я строка матрицы $P_r(x)$ нулевая. Пусть $c(x)$ — трейлинговый элемент i-й строки $P(x)$. Поделим дифференциальное уравнение, соответствующее i-й строке матрицы $P(x)$, на $c(x)$, продифференцируем получившееся уравнение и избавимся в нём от знаменателей. Затем заменим i-ю строку $P(x)$ на строку, соответствующую получившемуся дифференциальному уравнению. Эта операция называется *дифференциальным сдвигом* i-й строки $P(x)$.

Если i-е строки $P_r(x), P_{r-1}(x), \ldots, P_{u+1}(x)$, $0 \leqslant u < r$, нулевые и i-я строка $P_u(x)$ ненулевая, то i-я строка $P_{u+1}(x)$ будет ненулевой после дифференциального сдвига i-й строки $P(x)$. Этот факт и следующая лемма оправдывают термин «дифференциальный сдвиг».

Лемма 1. *Пусть i-я строка матрицы $P_r(x)$ нулевая. Тогда дифференциальный сдвиг i-й строки матрицы $P(x)$ уменьшает ширину этой строки.*

Доказательство. Так как мы предполагаем, что строки матрицы $P(x)$ независимы над $k[x, d/dx]$, то найдётся такое $s \geqslant 0$, что i-я строка матрицы $P_s(x)$ ненулевая и i-е строки матриц $P_{s-1}(x), P_{s-2}(x), \ldots, P_0(x)$ нулевые. Пусть i-я строка $P_s(x)$ имеет вид

$$(\underbrace{\ldots, b(x), c(x)}_{t \text{ элементов}}, 0, \ldots, 0),$$

$t > 0$, $c(x) \in k[x] \setminus \{0\}$. Тогда после дифференциального сдвига эта строка будет иметь вид

$$\big(\underbrace{\dots, h(x)\big(b(x)/c(x)\big)'}_{t-1 \text{ элемент}}, 0, 0, \dots, 0\big),$$

где $h(x)$ — полиномиальный множитель, обеспечивающий избавление от знаменателей. При этом i-е строки матриц $P_{s-1}(x), P_{s-2}(x), \dots, P_0(x)$ по-прежнему нулевые. □

Схема алгоритма EG_δ такова. Применяем любой имеющийся метод, чтобы проверить, являются ли строки ведущей части явной матрицы линейно зависимыми над $k(x)$, и если да, то находим коэффициенты зависимости $v_1(x), \dots, v_m(x) \in k[x]$. Из строк явной матрицы, соответствующих ненулевым коэффициентам, выбираем ту, которая имеет наибольшую ширину (пусть это будет i-я строка). Затем заменяем i-ю строку явной матрицы линейной комбинацией её строк с коэффициентами $v_1(x), \dots, v_m(x)$. В результате i-я строка ведущей части становится нулевой. Этот этап называется *редукцией*. Существенно, что ширина ни одной из строк не увеличивается из-за редукции. Теперь производим дифференциальный сдвиг i-й строки и продолжаем процесс до тех пор, пока ведущая часть не станет невырожденной. (Мы никогда не получим в матрице $P(x)$ нулевую строку, так как уравнения исходной системы независимы над $k[x, d/dx]$.)

Теорема 1. *Алгоритм* EG_δ *заканчивает свою работу.*

Доказательство. Как уже говорилось, ширина ни одной из строк не возрастает в результате редукции. Согласно лемме 1 дифференциальный сдвиг уменьшает ширину соответствующей строки. Таким образом, суммарная ширина всех строк уменьшается на шаге «редукция + дифференциальный сдвиг». □

К сказанному добавим, что поиск коэффициентов $v_1(x), \dots, v_m(x)$ линейной зависимости эквивалентен решению однородной системы линейных алгебраических уравнений с полиномиальными коэффициентами. Эта задача эффективно решается многими различными линейно-алгебраическими алгоритмами, в частности модулярными алгоритмами, которые хорошо справляются с ростом коэффициентов в промежуточных вычислениях. Если получено s линейно независимых решений этой системы линейных алгебраических уравнений, то можно использовать все эти решения для редукций, что приведёт к s нулевым строкам в ведущей матрице. Эти s зависимостей сначала представляются в виде строк матрицы V размера $s \times m$, первая строка матрицы V применяется для обнуления i-й строки ведущей части и дифференциального сдвига в явной матрице. Затем матрица V преобразуется исключением i-х элементов в строках со второй по s-ю с помощью первой строки. В результате все строки матрицы V начиная со второй содержат коэффициенты линейных зависимостей строк ведущей матрицы с номерами $1, \dots, i-1, i+1, \dots, m$. Так можно выполнить s шагов «редукция + дифференциальный сдвиг». Порядок выбора строк в матрице V может быть произвольным, поэтому возможны различные эвристические стратегии выбора.

Если после редукции получилось, что i-е строки матриц $P_r(x), P_{r-1}(x), \dots, P_{u+1}(x)$, $0 \leqslant u < r$, нулевые, i-я строка матрицы $P_u(x)$ ненулевая и при этом

$u < r - 1$, то уравнение, соответствующее i-й строке, можно продифференцировать $r - u - 1$ раз без деления на трейлинговый элемент, и только последнее ($(r-u)$-е) дифференцирование провести с предварительным делением и последующим освобождением от знаменателей (соображение, сообщённое авторам М. Баркату).

Не исключается, что ширина той строки явной матрицы, которую мы заменяем линейной комбинацией других строк, после редукции уменьшится. Тогда делить строку на её на трейлинговый элемент перед дифференцированием не обязательно.

Алгоритм Singsys (от английского выражения «Singularities of solutions of linear ordinary differential systems», т. е. «сингулярности решений линейных обыкновенных дифференциальных систем») состоит в применении алгоритма EG_δ к заданной системе и в последующем нахождении определителя ведущей матрицы получившейся системы с освобождением найденного полинома от квадратов.

Пример 1. Рассмотрим систему S

$$\begin{pmatrix} 2x^2(x+2)(x+1) & -x(x+2)(x+1) \\ 2x^2(x+2) & -x(x+2) \end{pmatrix} y'' + \\ + \begin{pmatrix} 2x(x+1)(x-4) & -x^2 \\ 2x(x-4) & -x(x+4) \end{pmatrix} y' + \begin{pmatrix} -2(x+1)(x-4) & -2 \\ -2x+8 & 2 \end{pmatrix} y = 0, \quad (4)$$

с явной матрицей

$$\begin{pmatrix} 2x^2(x+2)(x+1) & -x(x+2)(x+1) & 2x(x+1)(x-4) & -x^2 & -2(x+1)(x-4) & -2 \\ 2x^2(x+2) & -x(x+2) & 2x(x-4) & -x(x+4) & -2x+8 & 2 \end{pmatrix}. \quad (5)$$

Строки ведущей части матрицы (5) зависимы с коэффициентами $v_1 = -1$, $v_2 = x + 1$. Строки явной матрицы имеют одинаковую ширину. Заменяем вторую строку:

$$\begin{pmatrix} 2x^2(x+2)(x+1) & -x(x+2)(x+1) & 2x(x+1)(x-4) & -x^2 & -2(x+1)(x-4) & -2 \\ 0 & 0 & 0 & -x(x+2)^2 & 0 & 2x+4 \end{pmatrix}.$$

Дифференциальный сдвиг второй строки даёт

$$\begin{pmatrix} 2x^2(x+2)(x+1) & -x(x+2)(x+1) & 2x(x+1)(x-4) & -x^2 & -2(x+1)(x-4) & -2 \\ 0 & -x(x+2) & 0 & -2x & 0 & 0 \end{pmatrix}. \quad (6)$$

Матрица (6) служит явной матрицей системы S'

$$\begin{pmatrix} 2x^2(x+2)(x+1) & -x(x+2)(x+1) \\ 0 & -x(x+2) \end{pmatrix} y'' + \begin{pmatrix} 2x(x+1)(x-4) & -x^2 \\ 0 & -2x \end{pmatrix} y' \\ + \begin{pmatrix} -2(x+1)(x-4) & -2 \\ 0 & 0 \end{pmatrix} y = 0 \quad (7)$$

с невырожденной ведущей матрицей. Эта система является результатом применения EG_δ к системе (4). Определитель ведущей матрицы системы S' равен $-2x^3(x+2)^2(x+1)$. Полином

$$d(x) = x(x+2)(x+1) \tag{8}$$

является результатом работы алгоритма Singsys.

Построенный полином $d(x)$ — выявляющий полином для исходной системы (4), и решения этой системы могут иметь особенности только в точках -2, -1, 0.

Система S', которая строится алгоритмом EG_δ, может иметь больше решений, чем S. Если, например, $k = \mathbb{C}$ и ведущая матрица системы невырожденна, то размерность пространства голоморфных решений в любой области, не содержащей корней определителя ведущей матрицы, равна rm. Но если ведущая матрица вырожденна, то эта размерность может быть меньше rm. Например, система

$$\begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} y' + \begin{pmatrix} 0 & -1 \\ 0 & 1 \end{pmatrix} y = 0 \tag{9}$$

имеет одномерное пространство решений $y = (c, 0)^{\mathrm{T}}$. (Применение EG_δ к этой системе ведёт к построению системы S' вида

$$\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} y' + \begin{pmatrix} 0 & -1 \\ 0 & 0 \end{pmatrix} y = 0,$$

имеющей двумерное пространство решений $y = (c_1 x + c_2, c_1)^{\mathrm{T}}$.) При этом уравнения системы (9), как и строки её явной матрицы, независимы над $k[x, d/dx]$.

Выше мы исходили из предположения, что уравнения заданной системы независимы над $k[x, d/dx]$ (в этом случае пространство решений такой системы имеет размерность, не превышающую rm). Если отказаться от этого предположения, то алгоритм EG_δ позволит выяснить, имеет ли место эта независимость.

Теорема 2. *Пусть уравнения системы зависимы над $k[x, d/dx]$. Тогда на некотором шаге применения алгоритма EG_δ будет получена нулевая строка в явной матрице системы.*

Доказательство. Покажем, что если строки ведущей матрицы системы независимы над $k[x]$, то уравнения системы независимы над $k[x, d/dx]$. Предположим, что уравнения зависимы с операторами-коэффициентами $L_1, L_2, \ldots, L_m \in k[x, d/dx]$, и пусть

$$\operatorname{ord} L_i = l_i, \quad L_i = a_{i,l_i}(x)\frac{d^{l_i}}{dx^{l_i}} + a_{i,l_i-1}(x)\frac{d^{l_i-1}}{dx^{l_i-1}} + \ldots + a_{i,0}(x),$$

$i = 1, 2, \ldots, m$. Положим $l = \max\{l_1, l_2, \ldots, l_m\}$ и

$$b_i(x) = \begin{cases} 0, & \text{если } l_i \neq l, \\ a_{i,l_i}(x), & \text{если } l_i = l. \end{cases}$$

Легко убедиться, что линейная комбинация строк ведущей матрицы, взятых с коэффициентами $b_1(x), b_2(x), \ldots, b_m(x)$, равна нулю. Добавим, что если изначально уравнения системы были зависимыми над $k[x, d/dx]$, то уравнения, соответствующие строкам явной матрицы, зависимы над $k[x, d/dx]$ после каждого шага EG_δ. Из сказанного следует, что в случае зависимых уравнений применение EG_δ приведёт на некотором шаге к появлению в явной матрице нулевой строки. □

К доказанной теореме добавим несколько слов. Тем преобразованиям явной матрицы, которые выполняются алгоритмом EG_δ, соответствуют преобразования исходной системы S, сохраняющие число независимых над $k[x, d/dx]$ уравнений. Если предположить, что это число равно $m_0 \leqslant m$, то с помощью EG_δ можно построить систему S' из m_0 независимых над $k[x, d/dx]$ уравнений, для которой ведущая матрица (размера $m_0 \times m$) имеет над $k[x]$ ранг m_0 и каждое решение системы S является решением системы S'.

Теоретически можно говорить о применении алгоритмов EG_δ и Singsys к дифференциальным системам с любыми аналитическими коэффициентами. Но при таком применении придётся распознавать равенство нулю элементов явной матрицы, что не алгоритмизируется в общем случае. Кроме того, результатом применения Singsys будет аналитическая функция $d(x)$ (можно отказаться от этапа освобождения от квадратов), и не исключается, что уравнение $d(x) = 0$ будет иметь бесконечное множество корней.

2.2. Реализация алгоритмов EG_δ и Singsys

Наша реализация (см. `http://www.ccas.ru/sabramov/singsys/`) алгоритма EG_δ в системе Maple [13] основана на описанной в [4] нашей реализации алгоритма EG'. Главное отличие алгоритмов — шаг сдвига, и предлагаемая реализация включает вспомогательную процедуру дифференциального сдвига строки. Произведены также некоторые упрощения, так как алгоритм EG_δ ориентирован только на выявляющие преобразования системы по отношению к ведущей матрице. Алгоритм реализован в виде процедуры `EG_delta`, входными параметрами которой являются система в виде списка уравнений и список соответствующих неизвестных функций. Опционально может быть указан параметр — имя переменной, в которой будет сохранена явной матрица системы, полученная из исходной в результате работы алгоритма.

Алгоритм Singsys реализован в виде процедуры `Singsys`, использующей процедуру `EG_delta`. Входными параметрами процедуры `Singsys` также являются система в виде списка уравнений и список соответствующих неизвестных функций.

С помощью этих процедур можно найти полином, корни которого определяют потенциальные особые точки решений системы (4) из примера 1, а также получить систему с невырожденной ведущей матрицей, являющуюся результатом соответствующего выявляющего преобразования:

```
> sys := [2*x^2*(x+2)*(x+1)*diff(y1(x),x$2)-
 x*(x+2)*(x+1)*diff(y2(x),x$2)+
 2*x*(x+1)*(x-4)*diff(y1(x),x)-x^2*diff(y2(x),x)-
 2*(x+1)*(x-4)*y1(x)-2*y2(x),
 2*x^2*(x+2)*diff(y1(x),x$2)-x*(x+2)*diff(y2(x),x$2)+
 2*x*(x-4)*diff(y1(x),x)-x*(x+4)*diff(y2(x),x)-
 (2*x-8)*y1(x)+2*y2(x)]:
> vars := [y1(x), y2(x)]:
> Singsys(sys, vars);
```

$$x(x+2)(x+1)$$

```
> EG_delta(sys, vars);
```

$$\left[2x^2(x+2)(x+1)\left(\frac{d^2}{dx^2}y1(x)\right) - x(x+2)(x+1)\left(\frac{d^2}{dx^2}y2(x)\right) + \right.$$
$$+ 2x(x+1)(x-4)\left(\frac{d}{dx}y1(x)\right) - x^2\left(\frac{d}{dx}y2(x)\right) -$$
$$\left. - 2(x+1)(x-4)y1(x) - 2y2(x), -x(x+2)\left(\frac{d^2}{dx^2}y2(x)\right) - 2x\left(\frac{d}{dx}y2(x)\right)\right]$$

Наши эксперименты показывают, что алгоритм позволяет обрабатывать системы с достаточно большими входными параметрами. Например, были проведены две серии экспериментов, для каждой из которых были сгенерированы по семь наборов по десять дифференциальных систем с $m = 10$ и $r = 5, 10, 20, 40, 100, 250, 500$ соответственно. Коэффициенты всех этих систем были случайными полиномами (использовалась стандартная команда `randpoly()` системы Maple), при этом в первой серии системы генерировались так, чтобы число ненулевых коэффициентов составляло 30 %, а во второй серии — 50 %. Результаты экспериментов представлены в таблице. В ячейках указано общее время (в секундах) построения выявляющих полиномов для всех систем в каждом из наборов каждой серии (для всех экспериментов: Maple 13, Windows XP, Pentium 4 1.7 GHz, 1.5 GB RAM).

	5	10	20	40	100	250	500
30 %	3,189	6,468	13,516	29,642	187,375	3305,172	39183,048
50 %	3,578	4,361	11,955	31,875	255,063	5358,843	82052,204

2.3. Системы первого порядка

Если $r = 1$ в (1), т. е. система имеет вид

$$P_1(x)y' + P_0(x)y = 0, \tag{10}$$

и ранг матрицы $P_1(x)$ над $k[x]$ равен s, $0 < s < m$, то операция редукции позволяет переписать систему в виде пары, состоящей из линейных дифферен-

циальной и алгебраической систем $B_1(x)y'+B_0(x)y=0$, $R(x)y=0$, где матрицы $B_1(x)$, $B_0(x)$ имеют размер $s\times m$ и при этом ранг $B_1(x)$ равен s, а матрица $R(x)$ имеет размер $(m-s)\times m$ и её ранг равен $m-s$ в силу независимости над $k((x))$ уравнений исходной системы.

Далее можно применить вариант подхода, использованного в [14]. Дифференцируя систему $R(x)y=0$, получаем

$$R(x)y' + R'(x)y = 0. \tag{11}$$

Теперь в системе $B_1(x)y' + B_0(x)y = 0$ с помощью (11) исключаем некоторые y_i' для $m-s$ различных значений индекса i, а затем с помощью уравнений алгебраической системы $R(x)y = 0$ исключаем неизвестные y_i, имеющие те же значения индекса. Если получается дифференциальная система с ведущей матрицей, имеющей меньший ранг, чем s, то повторяем эти действия (переходим от дифференциальной системы к паре, состоящей из дифференциальной и алгебраической систем). В итоге получаем дифференциальную систему первого порядка с невырожденной ведущей матрицей. Этому соответствует матрица преобразования исходной системы, общий знаменатель элементов которой также вносит свой вклад в выявляющий полином.

Известно, что введением дополнительных неизвестных функций можно любую систему вида (1) привести к виду (10), при этом порядок матриц, входящих в систему, становится равным mr. Но также известно, что при больших r переход от системы (1) к системе первого порядка неудобен из-за возникновения матриц больших размеров, работа с которыми трудоёмка. Предположим, что сложность редукции ведущей матрицы порядка m есть m^ω, $2<\omega\leqslant 3$, по числу операций над элементами кольца $k[x]$. Число шагов «редукция + дифференциальный сдвиг» описанного в разделе 2.1 алгоритма EG_δ не превосходит rm^2, и цена каждого шага не превосходит $rm+m^\omega$. Отсюда следует, что сложность EG_δ не превосходит $r^2m^3+rm^{\omega+2}$. С другой стороны, при использовании описанного перехода к системе первого порядка общие затраты на редукции могут быть не меньше чем

$$\sum_{i=1}^{rm} i^\omega \sim \frac{(rm)^{\omega+1}}{\omega+1}.$$

Если, например, считать, что m фиксировано, а r растёт, то сложность EG_δ есть $O(r^2)$, в то время как сложность алгоритма, основанного на сведении к первому порядку, имеет рост, не меньший чем $r^{\omega+1}$, и при этом $\omega>2$.

Скорее всего, оценка rm^2 для числа шагов алгоритма EG_δ является сильно завышенной (наши эксперименты указывают на это). Во всяком случае, если каждый дифференциальный сдвиг приводит к появлению дополнительного решения системы, то число шагов не может превзойти rm.

Нами были проведены эксперименты по сравнению алгоритма Singsys и перехода к системе первого порядка. Для этого были сгенерированы четыре набора

по десять дифференциальных систем с $m = 10$ и $r = 5, 10, 15, 20$ соответственно. Как и в эксперименте из раздела 2.2, коэффициенты всех этих систем были случайными полиномами; системы генерировались так, чтобы число ненулевых коэффициентов составляло 50 %. Результаты экспериментов представлены в таблице. В ячейках указано общее время (в секундах) построения выявляющих полиномов сравниваемыми методами для всех систем в каждом из наборов.

	5	10	15	20
Singsys	6,422	17,375	21,344	29,422
Первый порядок	27,267	731,484	3287,844	6383,953

2.4. Рандомизация

Редукции и дифференциальные сдвиги могут привести к появлению в выявляющем полиноме добавочных множителей, корни которых не являются особенностями решений исходной системы. Некоторая дополнительная работа может позволить избавиться хотя бы от части этих множителей.

Внесём в алгоритмы EG_δ и Singsys элементы случайности. Пусть $0 < p \leqslant 1$. Будем выполнять дифференциальный сдвиг в том виде, как он описан в разделе 2.1, с вероятностью p и будем выполнять дифференцирование без деления на трейлинговый элемент с вероятностью $1-p$. Назовём это p-сдвигом (если $p = 1$, то p-сдвиг — это дифференциальный сдвиг).

С каждым шагом «редукция + p-сдвиг» суммарная ширина всех строк явной матрицы уменьшается на величину, среднее которой не меньше p. Так как $p > 0$, среднее время ожидания завершения выявляющего преобразования явной матрицы, основанного на шагах «редукция + p-сдвиг», будет конечным. Мы получаем рандомизированный вариант алгоритма EG_δ.

При получении выявляющего полинома можно использовать следующую схему: первый раз применяем алгоритм EG_δ так, как описано в разделе 2.1. Затем, не меняя выбранного значения p, несколько раз применяем рандомизированную версию EG_δ (при этом всё, что было выполнено до первого дифференциального сдвига, остаётся без изменений). Каждое такое применение может давать новую ведущую часть явной матрицы. Наибольший общий делитель полученных выявляющих полиномов будет, возможно, иметь меньшую степень. Процесс завершается, когда очередное применение не привело к уменьшению степени выявляющего полинома или когда возник выявляющий полином нулевой степени. Это даёт рандомизированный вариант алгоритма Singsys.

Пример 2. Рассмотрим систему

$$\begin{pmatrix} x & 0 \\ x(x-2) & 0 \end{pmatrix} y'' + \begin{pmatrix} 0 & 0 \\ -x+1 & 0 \end{pmatrix} y' + \begin{pmatrix} 0 & x-2 \\ 0 & -5x+6 \end{pmatrix} y = 0.$$

Использование алгоритма EG_δ даёт

$$\begin{pmatrix} x(x+2)(x-2) & -x(x+2)^3(x-2) \\ x+2 & 0 \end{pmatrix} y'' + \\ + \begin{pmatrix} -x^2-4 & 8(x+2)^2 \\ -1 & (x+2)^2 \end{pmatrix} y' + \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix} y = 0,$$

и полином $d_1(x) = x(x+2)(x-2)$ является результатом работы алгоритма Singsys.

Использование рандомизированного варианта EG_δ позволяет получить ещё и другие системы, в частности

$$\begin{pmatrix} x & 0 \\ -x & -x(x+2)(x-1) \end{pmatrix} y'' + \\ + \begin{pmatrix} 0 & 0 \\ -1 & -2(x+2)(2x-1) \end{pmatrix} y' + \begin{pmatrix} 0 & x-2 \\ 0 & -2x-4 \end{pmatrix} y = 0. \quad (12)$$

Тогда результатом работы рандомизированного варианта алгоритма Singsys является полином $d(x) = x(x+2)$, равный наибольшему общему делителю полинома $d_1(x)$ и полинома $d_2(x) = x(x+2)(x-1)$, соответствующего (12), поскольку другие преобразованные системы, получаемые рандомизированным вариантом EG_δ, уже не приводят к понижению степени выявляющего полинома (этим преобразованным системам соответствуют полиномы $d_3(x) = x(x+2)(x-1)(x^2+4x-2)$ и $d_4(x) = x(x+2)$, причём $d_4(x)$ в данном случае совпадает с итоговым результатом работы рандомизированного варианта алгоритма Singsys).

Рандомизированные версии алгоритмов EG_δ и Singsys включены в нашу реализацию, описанную в разделе 2.2. Процедуры `EG_delta` и `Singsys` имеют опциональный параметр, определяющий, надо ли использовать рандомизацию. Вероятность деления при выполнения дифференциального сдвига в рандомизированной версии алгоритма EG_δ взята равной $1/2$.

Применим рандомизированную версию для системы из примера 2.

```
> sys := [x*diff(y1(x),x$2)+(x-2)*y2(x),
 (x^2-2*x)*diff(y1(x),x$2)+(1-x)*diff(y1(x),x)+
 (6-5*x)*y2(x)]:
> Singsys(sys, vars, "random");
```

$$x(x+2)$$

```
> Singsys(sys, vars, "random");
```

$$x(x+2)(x-2)$$

```
> Singsys(sys, vars);
```

$$x(x+2)(x-2)$$

Это говорит о том, что рандомизация может позволить получить выявляющий полином меньшей степени, но может дать тот же самый результат, что и нерандомизированный вариант.

3. Приложение: совместная работа алгоритмов EG_δ и EG_σ

3.1. Алгоритм EG_σ

В дальнейшем нам будут встречаться рекуррентные системы вида

$$Q_l(n)z(n+l) + Q_{l-1}(n)z(n+l-1) + \ldots + Q_t(n)z(n+t) = 0, \qquad (13)$$

где $l \geqslant t$ — произвольные целые числа, $z(n) = \big(z(n)_1, \ldots, z_m(n)\big)^{\mathrm{T}}$ — вектор-столбец неизвестных последовательностей, а $Q_l(n), \ldots, Q_t(n)$ — квадратные матрицы размера $m \times m$ с элементами из $k[n]$. Ненулевые матрицы $Q_l(n)$ и $Q_t(n)$ называются *ведущей* и *трейлинговой* матрицами системы (13). Во многих случаях естественно рассматривать систему (13) совместно с конечным множеством *линейных ограничений*, т. е. линейных соотношений, каждое из которых содержит конечное множество величин $z_j(i)$. Пусть S и S' — системы вида (13), а C и C' — конечные множества линейных ограничений. Мы будем говорить, что системы (S, C) и (S', C') эквивалентны, если пространство решений S, удовлетворяющих C, совпадает с пространством решений S', удовлетворяющих C'. Конечное множество линейных ограничений может быть легко принято во внимание при определении различных свойств системы и при вычислении её решений. Алгоритмы EG [1] и EG′ [2—4] решают задачу нахождения системы (S', C'), эквивалентной исходной системе $(S, \varnothing)$ и такой, что ведущая или трейлинговая матрица S' невырожденна (EG′ — улучшенная версия EG).

Как мы упоминали в разделе 1, алгоритм EG_δ использует основные идеи EG и EG′.

Представляется логичным дать единообразные названия основанным на сходных идеях алгоритмам, касающимся дифференциальных и разностных систем. Поэтому для EG′ мы вводим новое название EG_σ. Обозначения δ и σ для отображений, которые обладают свойствами дифференцирования и соответственно сдвига, используются, например, в теории полиномов Оре.

3.2. Рекуррентные системы для коэффициентов решений в виде формальных рядов Лорана

Дифференциальные системы с полиномиальными коэффициентами индуцируют рекуррентные системы вида (13) для коэффициентов их решений в виде

рядов. Для построения индуцированной рекуррентной системы могут использоваться преобразования

$$x \to \phi^{-1}, \quad \frac{d}{dx} \to (n+1)\phi \tag{14}$$

(ϕ — оператор сдвига: $\phi(z_n) = z_{n+1}$). Подстановка $x + \alpha$ вместо x в дифференциальную систему сводит исследование решений в точке α к исследованию решений в точке 0. Корни определителей матриц $Q_l(n)$ и $Q_t(n)$ (в случае если эти матрицы невырожденны) важны для выяснения структуры пространства решений исходной системы в 0.

Алгоритмы EG_σ и Singsys могут работать вместе. При этом важно, что если заданная дифференциальная система имеет невырожденную ведущую матрицу, то это не гарантирует, что ведущая матрица индуцированной рекуррентной системы также является невырожденной [1, пример 8], и наоборот: для дифференциальной системы

$$\begin{pmatrix} x & 0 \\ 0 & 0 \end{pmatrix} y' + \begin{pmatrix} 0 & x \\ 1 & 1 \end{pmatrix} y = 0$$

соответствующая индуцированная рекуррентная система

$$\begin{pmatrix} n & 0 \\ 1 & 1 \end{pmatrix} z(n) + \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix} z(n-1) = 0$$

имеет невырожденную ведущую матрицу. Существуют простые примеры, в которых ведущие матрицы дифференциальной и индуцированной рекуррентной систем обе являются (не)вырожденными.

В случае вырожденности ведущей (трейлинговой) матрицы индуцированной системы мы применяем EG_σ. Возникающее при этом конечное множество C линейных соотношений может позволить отсеять некоторые из корней определителя результирующей невырожденной матрицы, не являющихся порядками решений в виде лорановых рядов.

Пример 3. Индуцированной рекуррентной системой для дифференциальной системы

$$\begin{pmatrix} x & x \\ x & x \end{pmatrix} y' + \begin{pmatrix} 1 & 1+x^2 \\ -x & 0 \end{pmatrix} y = 0 \tag{15}$$

служит

$$\begin{pmatrix} n+1 & n+1 \\ n & n \end{pmatrix} z(n) + \begin{pmatrix} 0 & 0 \\ -1 & 0 \end{pmatrix} z(n-1) + \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix} z(n-2) = 0.$$

Выявляющее преобразование последней системы по отношению к ведущей матрице приводит к соотношению

$$\begin{pmatrix} n+2 & 0 \\ n & n \end{pmatrix} z(n) + \begin{pmatrix} 0 & n+1 \\ -1 & 0 \end{pmatrix} z(n-1) = 0, \tag{16}$$

при этом возникает дополнительное линейное соотношение

$$z_1(0) + z_2(0) + z_2(-2) = 0. \tag{17}$$

Определитель ведущей матрицы системы (16) имеет корни 0, -2, но большему из этих корней не отвечает никакое лораново решение системы (15): соотношение (17) и первое уравнение системы (16) показывают, что если $z(-1) = z(-2) = 0$, то и $z(0) = 0$. Что касается корня -2, то соответствующие лорановы решения строятся легко. Выбираем $z(-2)$ так, чтобы выполнялось соотношение

$$\begin{pmatrix} 0 & 0 \\ -2 & -2 \end{pmatrix} z(-2) = 0.$$

В качестве базисного решения этой алгебраической системы можем взять, например, $z_1(-2) = (1, -1)^{\mathrm{T}}$. Из (16) получаем

$$\begin{pmatrix} 1 & 0 \\ -1 & -1 \end{pmatrix} z(-1) + \begin{pmatrix} 0 & 0 \\ -1 & 0 \end{pmatrix} z(-2) = 0,$$

откуда следует, что $z_1(-1) = (0, -1)^{\mathrm{T}}$. При $n = 0$ система (16) принимает вид

$$\begin{pmatrix} 2 & 0 \\ 0 & 0 \end{pmatrix} z(0) + \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix} z(-1) = 0$$

и вместе с (17) даёт $z_1(0) = (1/2, 1/2)^{\mathrm{T}}$. Используя (16), получаем

$$z(n) = \begin{pmatrix} 0 & -\frac{n+1}{n+2} \\ \frac{1}{n} & \frac{n+1}{n+2} \end{pmatrix} z(n-1)$$

для $n \geqslant 1$.

Таким образом, в точке $x = 0$ дифференциальная система (15) имеет одномерное пространство решений в виде лорановых рядов, его базис может быть задан рядом

$$\begin{pmatrix} 1 \\ -1 \end{pmatrix} x^{-2} + \begin{pmatrix} 0 \\ -1 \end{pmatrix} x^{-1} + \sum_{n=0}^{\infty} \begin{pmatrix} z_1(n) \\ z_2(n) \end{pmatrix} x^n,$$

где

$$\begin{pmatrix} z_1(0) \\ z_2(0) \end{pmatrix} = \begin{pmatrix} 1/2 \\ 1/2 \end{pmatrix}$$

и

$$\begin{pmatrix} z_1(n) \\ z_2(n) \end{pmatrix} = \begin{pmatrix} 0 & -\frac{n+1}{n+2} \\ \frac{1}{n} & \frac{n+1}{n+2} \end{pmatrix} \begin{pmatrix} z_1(n-1) \\ z_2(n-1) \end{pmatrix} \tag{18}$$

при $n \geqslant 1$.

Включение в множество C также дополнительных соотношений с нецелыми значениями аргумента n позволяет отсеивать некоторые значения λ, являющиеся кандидатами на роль показателя в (19).

В [8, раздел 6] содержится наблюдение, что тем преобразованиям, которые выполняются алгоритмом EG_σ над индуцированной рекуррентной системой, отвечают вполне определённые преобразования исходной дифференциальной системы. Это даёт, в терминологии авторов статьи [8], дифференциальный вариант EG_σ, который работает без перехода к рекуррентной системе.

Такого рода подход может быть полезен, когда требуется найти небольшое число членов лорановых рядов. Но когда это число велико, индуцированная рекуррентная система выглядит как более эффективное средство (в примере 3 мы получили удобную рекуррентную формулу (18)). Кроме этого, исходный вариант EG_σ даёт дополнительно конечное множество C линейных соотношений, которое в некоторых случаях позволяет не рассматривать часть корней определителя результирующей невырожденной ведущей матрицы рекуррентной системы (в примере 3 это множество состоит из одного-единственного соотношения (17), которое позволило не рассматривать корень -2 и избежать появления лишних решений).

Главная задача, которая решается в [8], — это построение всех регулярных решений исходной дифференциальной системы, т. е. решений вида

$$y(x) = x^\lambda v(x), \tag{19}$$

где $\lambda \in \bar{k}$, $v(x) \in \bar{k}[[x]]^m[\log x]$. Так как множество соотношений C не рассматривается, отсеивание появляющихся лишних решений предлагается выполнять с помощью подстановки в уравнения исходной системы. Если индуцированная рекуррентная система и множество C построены, то, как нам представляется, это отсеивание обходится дороже, чем учёт соотношений, подобных (17). (В [8] все степенные ряды, которые входят в регулярное решение, задаются в усечённом виде, что делает отсеивающую проверку-подстановку ещё более непростым делом.)

Ранее задача построения регулярных решений дифференциальных систем высокого порядка была полностью решена в [5], где на соответствующем этапе алгоритма вполне успешно применялась оригинальная версия EG_σ. Реализация алгоритма из [5] в виде процедур `RegularSolution` и `ExtendRegularSolution` входит в пакет `LinearFunctionalSystems` системы Maple начиная с версии 10.

3.3. Решения в виде рациональных функций

Пусть найдены корни выявляющего полинома $d(x)$ для заданной дифференциальной системы S. Для каждого из этих корней мы можем найти (применяя EG_σ к соответствующим индуцированным рекуррентным системам) нижние границы порядков полюсов решений S в виде формальных рядов Лорана в этой точке или обнаружить, что таких решений не существует. В последнем случае S не имеет рациональных решений (т. е. решений, все компоненты которых являются рациональными функциями). Иначе найденные нижние границы позволяют получить *границу знаменателя* рациональных решений S, т. е. такую рациональную функцию $U(x)$, что компоненты любого рационального решения $\big(y_1(x), y_2(x), \ldots, y_m(x)\big)^{\mathrm{T}}$ могут быть представлены в виде $y_i(x) = p_i(x)U(x)$, где $p_i(x) \in k[x]$, $i = 1, 2, \ldots, m$. Затем можно выполнить подстановку $\tilde{y}(x)U(x)$ вместо $y(x)$ в S и найти полиномиальные решения полученной системы. При этом

верхние границы степеней полиномиальных решений могут быть найдены с помощью построения рекуррентной системы для коэффициентов таких решений, выявляющего преобразования этой системы по отношению к трейлинговой матрице с помощью EG_σ, и исследования целых корней определителя полученной трейлинговой матрицы (см. [1, разделы 3.3, 3.4]).

Пример 4. Найдём рациональные решения системы (4). Построением индуцированных рекуррентных систем в точках -2, -1, 0 (корнях полинома (8)), выявляющими преобразованиями этих систем по отношению к ведущим матрицам с помощью EG_σ и анализом корней определителей результирующих ведущих матриц находится граница знаменателей $U(x) = x/(x+2)^2$ рациональных решений системы (4).

Остаётся найти полиномиальные решения дифференциальной системы, получаемой подстановкой $\tilde{y}(x)x/(x+2)^2$ вместо $y(x)$ в систему (4). После всех вычислений получаем пространство рациональных решений системы (4):

$$y = \left(\frac{x(c_1 + 4xc_3 + x^2 c_3)}{(x+2)^2},\ \frac{c_2 x}{(x+2)} \right)^{\mathrm{T}}.$$

Уже упоминавшийся пакет `LinearFunctionalSystems`, содержащийся в системе Maple, предоставляет процедуры для поиска решений систем обыкновенных уравнений (в том числе и дифференциальных), основанные на построении индуцированных рекуррентных систем и выявляющих преобразованиях этих систем с помощью EG_σ. Процедура `RationalSolutions` этого пакета находит рациональные решения лишь таких систем вида (1), для которых ведущая матрица $P_r(x)$ изначально невырожденна, эта процедура использует в своей работе процедуру `UniversalDenominator` того же пакета, применимую для построения границ знаменателей только таких систем. В нашей новой реализации процедура `UniversalDenominator` использует процедуру `Singsys`, описанную в разделе 2.2. Граница знаменателей разыскивается так, как написано в начале этого раздела. В итоге процедуры `UniversalDenominator` и `RationalSolutions` применимы для любых систем вида (1), уравнения которых независимы над $k[x, d/dx]$. Используя `sys` и `vars` из раздела 2.2, получаем рациональные функции из примера 4:

```
> LinearFunctionalSystems[UniversalDenominator](sys, vars);
```

$$\frac{x}{(x+2)^2}$$

```
> LinearFunctionalSystems[RationalSolutions](sys, vars);
```

$$\left[\frac{(\_c_1 + 4x\_c_3 + x^2\_c_3)x}{(x+2)^2},\ \frac{x\_c_2}{x+2} \right]$$

Наши эксперименты показывают, что при поиске рациональных решений систем имеет смысл использовать рандомизированный вариант алгоритма Singsys. Например, был проведён эксперимент, для которого были сгенерированы три

набора по десять дифференциальных систем с $m = 10$ и $r = 5, 10, 15$ соответственно. Все системы были построены таким образом, что имели случайно сгенерированные рациональные решения. Был произведён поиск рациональных решений всех систем в каждом наборе с использованием построения выявляющего полинома с помощью рандомизированной и нерандомизированной версии Singsys. Результаты экспериментов представлены в таблице. В ячейках указано общее время (в секундах) поиска рациональный решений всех систем в каждом из наборов, вдобавок в скобках указано соответствующее время выполнения вспомогательной операции построения выявляющих полиномов. Дополнительные затраты на рандомизацию, как правило, окупаются за счёт экономии на последующих этапах, которые являются гораздо более затратными как с точки зрения времени выполнения, так и с точки зрения потребляемой памяти.

	5	10	15
Без рандомизации	336,531 (7,547)	1128,096 (14,345)	3305,061 (24,936)
С рандомизацией	292,704 (20,640)	958,890 (41,859)	2887,798 (184,046)

Алгоритм из [7] также предназначен для поиска рациональных решений, но только систем вида $y' = My + N$, где $M \in \mathrm{Mat}_m\big(k(x)\big)$, $N \in k(x)^m$. Если рациональное решение такой системы имеет полюс в $\alpha \in k$, то $g(\alpha) = 0$, где $g(x)$ — полином, являющийся общим знаменателем элементов матрицы M и вектора N.

Авторы признательны М. Баркату и Э. Пфлюгелю за полезные дискуссии и советы.

Литература

[1] Abramov S. EG-eliminations // J. Differ. Equ. Appl. — 1999. — Vol. 5. — P. 393—433.

[2] Abramov S., Bronstein M. On solutions of linear functional systems // Proc. of the 2001 Int. Symp. on Symbolic and Algebraic Computation, London, Ontario, Canada, July 22—25, 2001 / B. Mourrain, ed. — New York: ACM Press, 2001. — P. 1—6.

[3] Abramov S. A., Bronstein M. Linear algebra for skew-polynomial matrices: Rapport de Recherche INRIA. — RR-4420. — March 2002.

[4] Abramov S., Bronstein M., Khmelnov D. Regularization of linear recurrence systems // Trans. A. M. Liapunov Inst. — 2003. — Vol. 4. — P. 158—171.

[5] Abramov S., Bronstein M., Khmelnov D. On regular and logarithmic solutions of ordinary linear differential systems // Computer Algebra in Scientific Computing. Ann. Int. Workshop in Computer Algebra in Scientific Computing. Kalamata, Greece, 12—16 September 2005. — (Lect. Notes Comput. Sci.; Vol. 3718). — Berlin: Springer, 2005. — P. 1—12.

[6] Abramov S., Khmelnov D. Desingularization of leading matrices of systems of linear ordinary differential equations with polynomial coefficients // Int. Conf. «Differential Equations and Related Topics» Dedicated to I. G. Petrovskii, Moscow, MSU, May 30 — June 4, 2011. Book of Abstracts. — 2012. — P. 5.

[7] Barkatou M. A. On rational solutions of systems of linear differential equations // J. Symbol. Comput. — 1999. — Vol. 28. — P. 547—567.

[8] Barkatou M. A., El Bacha C., Cluzeau T. Simple forms of higher-order linear differential systems and their applications in computing regular solutions // J. Symbol. Comput. — 2011. — Vol. 46. — P. 633—658.

[9] Barkatou M. A., El Bacha C., Pflügel E. Simultaneously row- and column-reduced higher-order linear differential systems // Symbolic and Algebraic Computation, Int. Symp., ISSAC 2010, Munich, Germany, July 25—28, 2010, Proceedings / W. Koepf, ed. — New York: ACM Press, 2010. — P. 45—52.

[10] Beckermann B., Cheng H., Labahn G. Fraction-free row reduction of matrices of skew polynomials // Proc. of the 2002 Int. Symp. on Symbolic and Algebraic Computation / T. Mora, ed. — New York: ACM Press, 2002. — P. 8—15.

[11] Beckermann B., Cheng H., Labahn G. Fraction-free row reduction of matrices of Ore polynomials // J. Symbol. Comput. — 2006. — Vol. 41. — P. 513—543.

[12] Davies P., Cheng H., Labahn G. Computing Popov form of general Ore polynomial matrices // Milestones in Computer Algebra, MICA 2008. A Conf. in Honour of Keith Geddes' 60th Birthday. Stonehaven Bay, Trinidad and Tobago, 1—3 May 2008. — 2008. — P. 149—156.

[13] Maple online help. — `http://www.maplesoft.com/support/help/`.

[14] Quéré M. P., Villard G. An algorithm for the reduction of linear DAE // Int. Symp. on Symbolic and Algebraic Computation, ISSAC'95. Montreal, Canada, juillet 1995. — New York: ACM Press, 1995. — P. 223—231.