Resolving sequences of operators for linear ordinary differential and difference systems

S. A. Abramov
Computing Centre of
the Russian Academy of Science
Vavilova str., 40
Moscow, 119333
Russia
sergeyabramov@mail.ru

M. Petkovšek
University of Ljubljana
Faculty of Mathematics and Physics
Jadranska 19, SI-1000 Ljubljana
Slovenia
Marko.Petkovsek@fmf.uni-lj.si

A. A. Ryabenko
Computing Centre of
the Russian Academy of Science
Vavilova str., 40
Moscow, 119333
Russia
anna.ryabenko@gmail.com

Abstract

We introduce the notion of a resolving sequence of (scalar) operators for a given differential or difference system with coefficients in some differential or difference field $\mathbb{K}$. We propose an algorithm to construct such a sequence, and give some examples of the use of this sequence as a suitable auxiliary tool for finding certain kinds of solutions of differential and (q-)difference systems. Some experiments with our implementation of the algorithm are reported.

1 Introduction

As a rule, both in scientific literature and in practice, algorithms for finding certain kinds of solutions for scalar differential or difference equations appear earlier than for systems of such equations. It may also be that a direct algorithm for systems is known in theory but does not yet have an available computer implementation (e.g., there is no such implementation in commonly used software packages). In this case one makes an effort to find solutions of a system through some auxiliary scalar equations which are constructed for the system.

We will consider linear systems over a differential or difference field $\mathbb{K}$ of characteristic zero supposing that the components of solutions $y = (y_1, \ldots, y_m)^T$ belong to an extension Λ of $\mathbb{K}$. Our approach allows in specific cases to reduce the problem to the search for simpler solutions (on occasion belonging to $\mathbb{K}$). It is based on the notion of a resolving sequence of operators, i.e, such a finite sequence $L_1, \ldots, L_p$ of scalar operators with coefficients in $\mathbb{K}$ that for fixed indices $l_1, \ldots, l_p$, first, the equalities $y_{l_1} = \cdots = y_{l_j} = 0$ imply $L_{j+1}(y_{l_{j+1}}) = 0$ when $j < p$, and second, the equalities $y_{l_1} = \cdots = y_{l_p} = 0$ imply $y_1 = y_2 = \cdots = y_m = 0$.

We will propose an algorithm to construct a resolving sequence for a given system. For the case when $\mathbb{K}$ is the rational function field $K(x)$ over an algebraically closed field K we derive an algorithm to find hypergeometric (other names: exponential, hyperexponential, ...) solutions of systems. (In [25, 23, 21], algorithms for finding hypergeometric solutions of scalar linear difference equations with rational-function coefficients were described. Algorithms for the q-difference scalar equation can be found in [10], [21, Sect. 12], the differential case is considered in [20].)

For differential systems with coefficients in $K(x)$, resolving systems allow to reduce the search for formal exponential-logarithmic solutions to the search for exponential parts of solutions of scalar equations and the subsequent search for regular solutions of systems with coefficients in $K(x)$.

Generally, direct algorithms work faster than algorithms that first uncouple the system. Thus, most probably, the search for hypergeometric solutions of systems will be faster with advent of full direct algorithms for systems. (It is known that work is underway on such an algorithm for normal first-order system $y(x + 1) = A(x)y(x)$, where $A(x)$ is a square matrix whose entries are rational functions [17]). A direct algorithm for finding formal exponential-logarithmic solutions of normal first-order differential systems $y'(x) = A(x)y(x)$ with rational-function coefficients has been proposed in [16], but it does not have yet an available full implementation, although such an implementation will be hopefully available in the future. Currently, our algorithms can be useful for solving systems – the more so as our experiments show that these algorithms work in reasonable time.

If a resolving sequence consists of a single operator (i.e., if $p = 1$) then in the case of a normal first-order system, the row-vector $(\underbrace{0, \ldots, 0}_{l_1-1}, 1, \underbrace{0, \ldots, 0}_{m-l_1})$ is a cyclic vector for the system. Algorithms for constructing a cyclic vector and related uncoupling algorithms for normal first-order systems are well known in computer algebra. The paper [18] contains a review of such algorithms and the corresponding references. In that paper, a new algorithm for constructing a cyclic vector based on fast linear algebra algorithms ([26]) is also proposed. The existence of cyclic-vector algorithms suggests that there is no need for resolving sequences at least in the case of normal first-order systems – the more so as the cyclic vector and the uncoupling are multi-purpose procedures which may be useful not only for finding hypergeometric or formal exponential-logarithmic solutions of systems. Besides, the resolving procedure only solves a part of the problem. Solutions of the operators belonging to the resolving sequence are a “half-finished product”, since we have in addition to find some “simple” solutions of other systems. However, the resolving sequence approach can have some advantage over the cyclic vector approach. First, this sequence is constructed in a single pass, while in practical cyclic vector algorithms numerous random candidates are considered (if a candidate is not appropriate then another one is generated, and all calculations are resumed from the beginning).

Second, asymptotically fast algorithms have advantages over classical algorithms only for inputs of large sizes. However, as for systems of differential or difference equations, a system of, say, twenty equations is quite large for computer algebra algorithms, but twenty is not a large enough size for “fast” linear algebra algorithms. The orientation towards asymptotic complexity estimates does not seem to be very productive here. The use of classical linear algebra algorithms as auxiliary tools for searching for solutions of differential and difference systems is thus logical in many cases.

Third, the resolving system constructed by our algorithm is such that the sum of the orders of its operators is equal to the order of the operator obtained by a cyclic vector algorithm. As a rule, to solve a few equations of small orders is easier than to solve an equation of a large order (this provides stimulation to develop factorization algorithms). Even when a resolving sequence consists of a unique operator, we can profit since such a cyclic vector is of a very simple form, and the scalar equation to solve will not have cumbersome coefficients. We note that if the coefficients of systems are rational functions then we have an opportunity to use various heuristics to obtain operators with possibly less cumbersome coefficients.

As for the second stage when some solutions of additional differential or difference systems have to be found, this can quite often be done in reasonable time. For example, the search for rational solutions of differential or difference systems having rational-function coefficients is not time consuming ([2, 5, 6, 7, 13, 14, 22]).

Our experimental comparison demonstrates a definite advantage of a resolving sequence over a cyclic vector.

Besides, we show that the resolving sequences can be used for higher-order systems of full rank. In so doing we do not suppose that, say, the leading matrix of the system is invertible. In this case, the cyclic vector methods are not applicable. The algorithms are based on the EG-eliminations ([1, 3, 9]). Possibly, these are the first algorithms for higher-order systems.

A preliminary version of our algorithm for finding hypergeometric solutions of normal first-order difference systems was published in [12].

2 First-order normal systems

2.1 The resolving equation and matrix

Let $\mathbb{K}$ be a differential or difference field with derivation δ or, resp., automorphism (shift) σ. In this section we consider systems of the form

$$\delta y = Ay, \tag{1}$$

and

$$\sigma y = Ay. \tag{2}$$

In both cases $A \in \operatorname{Mat}_m(\mathbb{K})$, $y = (y_1, \ldots, y_m)^T$. Systems of this form we call *normal* first-order systems. For systems (1), (2) we will use the short notation $[A]$.

We suppose that Λ is an extension (differential or difference) of $\mathbb{K}$, and will discuss some general facts related to the solutions of $[A]$ which belong to Λ^m.

First of all, recall how to find derivatives and shifts of a scalar t of the form $t = cy$, where c is a given row-vector from $\mathbb{K}^m$, while y is supposed to be a solution of $[A]$ (i.e., how to differentiate, resp., to shift "along the solutions" of $[A]$). Define a sequence $c^{[0]}, c^{[1]}, c^{[2]}, \ldots$ of row-vectors

$$c^{[0]} = c, \;\; c^{[i]} = c^{[i-1]}A + \delta c^{[i-1]} \tag{3}$$

in the differential case, and

$$c^{[0]} = c, \;\; c^{[i]} = \sigma(c^{[i-1]})A \tag{4}$$

in the difference case, $i = 1, 2, \ldots$ In the sequel we will often use a unified form for the systems (1) and (2):

$$\xi y = Ay, \;\; A \in \operatorname{Mat}_m(\mathbb{K}), \;\; y = (y_1, \ldots, y_m)^T, \;\; \xi \in \{\delta, \sigma\}. \tag{5}$$

The known fact is that

$$\xi^i(cy) = c^{[i]}y$$

for any $c \in \mathbb{K}^m$ and any solution y of $[A]$.

It is also well known that for any i, $0 \leqslant i \leqslant m$, one can construct a scalar equation over $\mathbb{K}$ of order $\leqslant m$ which is satisfied by the i-th component y_i of any solution of $[A]$. (The order of a scalar operator $L \in K[\xi]$ is the degree of L as a polynomial in ξ, the order of the equation $L(y) = 0$ is equal to the order of L.) We describe briefly a way to do this.

Since $c^{[0]}, c^{[1]}, \ldots, c^{[m]}$ are linearly dependent over $\mathbb{K}$, there is a least $k \in \{0, 1, \ldots, m\}$ such that $c^{[0]}, c^{[1]}, \ldots, c^{[k]}$ are linearly dependent over $\mathbb{K}$. So there are $u_0, u_1, \ldots, u_k \in \mathbb{K}$, with $u_k \neq 0$, such that $\sum_{j=0}^k u_j c^{[j]} = 0$, hence also $\sum_{j=0}^k u_j c^{[j]} y = \sum_{j=0}^k u_j \xi^j t = 0$. This is a scalar equation of order k satisfied by $t = cy$. In particular, for

$$c = (\underbrace{0, \ldots, 0}_{i-1}, 1, \underbrace{0, \ldots, 0}_{m-i}) \tag{6}$$

we have $t = y_i$, hence

$$\sum_{j=0}^{k} u_j \xi^j y_i = 0 \tag{7}$$

is a scalar equation of order k satisfied by y_i for any solution y of $[A]$.

Definition 1. *Let the row vectors $c^{[0]}, c^{[1]}, \ldots, c^{[k]}$ and $u_0, u_1, \ldots, u_k \in \mathbb{K}$ be constructed as described above for c as given in (6). Then we call (7) the* y_i-resolving equation, *and the full-rank $k \times m$ matrix B whose j-th row, for $j = 1, \ldots, k$, is $c^{[j-1]}$, the* y_i-resolving matrix *of* $[A]$.

2.2 The space of solutions with $y_i = 0$

Here we are interested in the solutions of $[A]$ with $y_i = 0$.

Proposition 1. *Let equation (7) with $k < m$ be the y_i-resolving equation for system (5), $1 \leqslant i \leqslant m$. Then there exist $m-k$ entries $y_{i_1}, \dots, y_{i_{m-k}}$ of y and a system*

$$\xi\tilde{y} = \tilde{A}\tilde{y} \tag{8}$$

where $\tilde{A}$ is an $(m-k) \times (m-k)$ matrix whose entries are in $\mathbb{K}$ with

$$\tilde{y} = (y_{i_1}, \dots, y_{i_{m-k}})^T, \tag{9}$$

such that if the system $[A]$ has a solution with $y_i = 0$ then

- *the entries $y_{i_1}, \dots, y_{i_{m-k}}$ of the solutions satisfy system (8);*
- *each y_j with $j \notin \{i_1, \dots, i_{m-k}\}$ can be expressed as a linear form in $y_{i_1}, \dots, y_{i_{m-k}}$.*

If $k = m$ in (7) then $y_i = 0$ implies $y_j = 0$ for all $j = 1, \dots, m$.

Proof. Note that if $y_i = 0$ then

$$y_i = \xi y_i = \xi^2 y_i = \dots = \xi^{k-1} y_i = 0. \tag{10}$$

Let c be as in (6). Since $c^{[j]}y = \xi^j(cy) = \xi^j y_i = 0$, this gives us a system of k independent linear algebraic equations

$$By = 0 \tag{11}$$

for the unknown y, where B is the y_i-resolving matrix of $[A]$. The matrix B has full rank, and hence there exist $m-k$ entries $y_{i_1}, \dots, y_{i_{m-k}}$ of y such that by means of this system, the other k entries of y can be expressed as linear forms in $y_{i_1}, \dots, y_{i_{m-k}}$ having coefficients from $\mathbb{K}$. Now we can transform $[A]$ as follows: for each $1 \leqslant j \leqslant m$ such that $j \notin \{i_1, \dots, i_{m-k}\}$ we

(a) remove the equation

$$\xi y_j = a_{j1}y_1 + \dots + a_{jm}y_m$$

from $[A]$,

(b) in all the other equations, we replace y_j by the corresponding linear form in $y_{i_1}, \dots, y_{i_{m-k}}$ (in particular, y_i will be replaced by 0, since $i \notin \{i_1, \dots, i_{m-k}\}$ and the system $By = 0$ contains the equation $y_i = 0$).

As a whole, this gives a system of the form (8). If $[A]$ has a solution such that y_i is zero then (9) satisfies the system (8), and each y_j with $j \notin \{i_1, \dots, i_{m-k}\}$ can be expressed as a linear form in $y_{i_1}, \dots, y_{i_{m-k}}$ having coefficients from $\mathbb{K}$.

If $k = m$ then the matrix B in the linear algebraic system (11) is an invertible $m \times m$ matrix, hence $y = 0$. □

Remark 1. *The matrix $\tilde{A}$ of system (8) can be found using matrix multiplication. Let S be the $m \times (m-k)$ matrix whose jth row is the row of the coefficients of the corresponding linear form for y_j in $y_{i_1}, \dots, y_{i_{m-k}}$. In particular, its jth row is equal to $(\underbrace{0, \dots, 0}_{j-1}, 1, \underbrace{0, \dots, 0}_{m-j})$ for each $j \in \{i_1, \dots, i_{m-k}\}$. Let A^- be the matrix consisting of the rows of A having the numbers $i_1, \dots, i_{m-k}$. Then $\tilde{A} = A^- S$.*

2.3 Resolving sequence of operators

Definition 2. *Let $l_1, \dots, l_p$ be pairwise distinct positive integers which do not exceed the number m of solution entries of a given differential or difference system, and let $L_1, \dots, L_p$ be scalar operators from $\mathbb{K}[\xi]$ such that if $y_{l_1} = \dots = y_{l_j} = 0$ (where $j \leqslant p$) for a solution y then*

- *in case of $j = p$, all entries of this solution are equal to zero: $y_1 = y_2 = \dots = y_m = 0$,*
- *in case of $j < p$, for the entry $y_{l_{j+1}}$ of this solution the equality $L_{j+1}(y_{l_{j+1}}) = 0$ holds.*

Then the finite sequence of operators $L_1, \dots, L_p$ is a resolving sequence *for the given system, and the vector $(l_1, \dots, l_p)$ is the* indicator *of this sequence.*

Verification of existence of a resolving sequence for an arbitrary system of the form (5) as well as an algorithm for constructing a resolving sequence are actually contained in Sections 2.1, 2.2. It is easy to describe this algorithm. The description is particularly simple if it does not include the calculation of the indicator:

1. Set $t = 0$.

2. Set $t = t + 1$. Select any y_i from the unknowns under consideration, and set $l_t = i$. Construct L_t as the y_i-resolving operator and the y_i-resolving matrix B of the system $[A]$.

3. If the order of L_t is less than the order of the matrix A then execute the transformations (a), (b) of $[A]$ as it is described in the proof of Proposition 1 (A and $[A]$ will be changed), and go to 2.

Remark 2. *The simplest way to select y_i on step 2 is just to pick the first unknown from those under consideration. However, it is probably more reasonable to find such a row of the matrix of the system which is the least "cumbersome" of all the rows which contain the largest number of zero entries (the "cumbersome" criterion should be clarified). Then we select the unknown y_i so that ξy_i corresponds to the selected row in the matrix of $[A]$.*

In addition, we propose another description of basically the same algorithm. This version is based entirely on Gaussian elimination of unknowns, and deals with a $(m+1) \times (2m+1)$ matrix V whose rows are filled gradually. If the first $w - 1$ entries of a non-zero row of V are zeros, but the w-th entry is not zero then w is the *level* of that row.

1. Set $t = d = 0$.

2. Set $t = t + 1$. Select any integer i, $1 \leqslant i \leqslant m$, which coincides with no level of the rows of V having numbers $1, \dots, d$, and set $l_t = i$. Set c to be equal to the row (6) and, using the corresponding recurrence relation from (3), (4) find step-by-step $c^{[0]}, c^{[1]}, \dots$ filling the rows of V with numbers $d+1, d+2, \dots$: if $c^{[j]} = (c_{j1}, \dots, c_{jm})$ then the $(d + j + 1)$-st row of V becomes equal to

$$(c_{j1}, \dots, c_{jm}, \underbrace{0, \dots, 0}_{d+j}, 1, \underbrace{0, \dots, 0}_{m-d-j}).$$

After putting the new row into the matrix V we immediately perform eliminations by the rows having smaller numbers. Thus, the rows which are already in

V form a sub-matrix which is in row echelon form. (Elimination by a row produces zero in the column whose number is equal to the level of that row.) At some point we get in V a row having only zeros in the columns with numbers $1, \dots, m$. If it is the sth row then

$$L_t = \sum_{j=d}^{s-1} v_{s,m+j+1} \xi^{j-d}. \tag{12}$$

3. Set $d = d + \operatorname{ord} L_t$. If $d = m$ then a resolving sequence has been found. Otherwise, go to 2.

Remark 3. *(a) The entries $v_{s,m+1}, \dots, v_{s,m+d}$ do not appear in (12). Those useless entries of the sth row need not be calculated: before computing the next L_t, $t > 1$, set $v_{ij} = 0$ for $i = 1, \dots, d$, $j = m+1, m+2, \dots, 2m+1$.*

(b) After computing L_t, $t \geqslant 1$, the l_t-th column of A can be replaced by the zero column. The modified matrix A is used when we compute $c^{[1]}, c^{[2]}, \dots$ for constructing the next operators belonging to the resolving sequence (if A is needed for another computation after constructing the resolving sequence then a copy of the initial A must be stored).

Proposition 2. *The version of the algorithm described below Remark 2 computes a resolving sequence of operators for $[A]$.*

Proof. Let $L_1, \dots, L_p$ be the sequence computed by the described algorithm. Then the statement can be proved by induction on p using Proposition 1 for the inductive step.

The entries $v_{s,m+1}, \dots, v_{s,m+d}$ do not appear in (12) (Remark 3(a)), and the first m entries of each row of V having the number $\leqslant d$ define a linear combination of $y_1, \dots, y_m$, which is equal to zero when we consider the operator L_{t+1}. $\square$

This version of the algorithm is very close to the classical algorithm for transforming a matrix into row echelon form by Gaussian elimination. Some additional work is needed to generate the rows of the matrix V. The complexity of both versions of the algorithm as the number of operations in $\mathbb{K}$ is bounded by Cm^3 with a reasonably small constant C.

The indicator of a resolving sequence of operators is actually not needed in some applications of resolving systems, see, e.g., Sections 4, 5.

3 Higher-order systems

In this section we consider systems of the form

$$A_n \xi^n y + \dots + A_1 \xi y + A_0 y = 0 \tag{13}$$

with $A_i \in \operatorname{Mat}_m(\mathbb{K})$, $i = 0, 1, \dots, n$. Following Definition 2, we will consider resolving sequences of operators and their indicators for such systems.

3.1 The companion matrix

We distinguish two cases, according to whether the leading matrix of the system is invertible or not. If A_n is invertible in $\mathrm{Mat}_m(\mathbb{K})$ then the system (13) is equivalent to the first-order system having mn equations:

$$\xi Y = AY, \tag{14}$$

with a *companion* matrix A:

$$A = \left(\begin{array}{cccc} 0 & I_m & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & I_m \\ \bar{A}_0 & \bar{A}_1 & \dots & \bar{A}_{n-1} \end{array} \right), \tag{15}$$

where $\bar{A}_k = -A_n^{-1}A_k$, $k = 0, 1, \dots, n-1$, and

$$Y = (Y_1, \dots, Y_{nm}) = \\ = \left(y_1 \dots, y_m,\ \xi y_1 \dots, \xi y_m, \dots,\ \xi^{(n-1)} y_1, \dots, \xi^{(n-1)} y_m\right)^T. \tag{16}$$

If the leading matrix A_n in (13) is not invertible but the system is of full rank then the EG-eliminations method [1, 3, 9] allows us to obtain an *embracing* system, i.e., a system of a similar form with an invertible leading matrix, and all solutions of the original system are solutions of the new one. It follows from Definition 2 that any resolving sequence of an embracing system is also a resolving sequence of the original system.

3.2 Constructing resolving sequence without explicit use of the companion matrix

A resolving sequence for the system $\xi Y = AY$ can be constructed by application of (3), (4). However, the first stage of the algorithm, i.e., constructing L_1 can be performed without the implicit use of the huge matrix (15). To do this, we split the row-vector $c^{[i]}$ into n parts of equal length:

$$c^{[i]} = \left(c_0^{[i]},\ c_1^{[i]},\ \dots,\ c_{n-1}^{[i]}\right).$$

We rewrite recursive formulas (3), (4) as

$$\begin{gathered} c_0^{[i]} = c_{n-1}^{[i-1]} \bar{A}_0 + \delta c_0^{[i-1]}, \\ c_1^{[i]} = c_0^{[i-1]} + c_{n-1}^{[i-1]} \bar{A}_1 + \delta c_1^{[i-1]}, \\ \dots \\ c_{n-1}^{[i]} = c_{n-2}^{[i-1]} + c_{n-1}^{[i-1]} \bar{A}_{n-1} + \delta c_{n-1}^{[i-1]} \end{gathered} \tag{17}$$

and

$$\begin{gathered} c_0^{[i]} = \sigma(c_{n-1}^{[i-1]}) \bar{A}_0, \\ c_1^{[i]} = \sigma(c_0^{[i-1]}) + \sigma(c_{n-1}^{[i-1]}) \bar{A}_1, \\ \dots \end{gathered} \tag{18}$$

$$c^{[i]}_{n-1} = \sigma(c^{[i-1]}_{n-2}) + \sigma(c^{[i-1]}_{n-1})\bar{A}_{n-1}$$

for $i = 1, 2, \ldots$ In both cases $c^{[0]} = c$, and all the components of c are equal to 0, save the one belonging to the part $c^{[0]}_0$ which is equal to 1 (note that the total number of components of c is mn).

When an operator L_1 is constructed, one constructs other operators of a resolving sequence using the transformed system $[A]$ (this transformation uses the resolving matrix B, see Section 2.2). This transformed system can be obtained without first explicitly constructing the matrix (15): Select $mn - k$ components of Y with indices $i_1 < i_2 < \cdots < i_s \leqslant m(n-1) < i_{s+1} < \cdots < i_{mn-k}$ such that other components can be expressed by them using the system $BY = 0$. Construct the $mn \times (mn-k)$ matrix S as in Remark 1. For $j = 1, \ldots, s$ the jth row of the transformed matrix $\tilde{A}$ is equal to the $(i_j + m)$th row of S: $\tilde{A}_{j,*} = S_{i_j+m,*}$ (we use the standard notation $M_{p,*}$ for the pth row of a matrix M). For $j = s+1, s+2, \ldots, mn-k$, the jth row $\tilde{A}_{j,*}$ of $\tilde{A}$ can be computed as the product $A^{\smile}_{i_j - m(n-1),*}S$, where $A^{\smile}$ is the $m \times mn$ matrix $\begin{pmatrix}\bar{A}_0 & \bar{A}_1 & \ldots & \bar{A}_{n-1}\end{pmatrix}$.

3.3 The final step of constructing a resolving sequence

Let $L_1, \ldots, L_p$ be a resolving sequence for (14) with the indicator $(l_1, \ldots, l_p)$. The elements of this indicator are some of the indices of the elements of $Y = (Y_1, \ldots, Y_{mn})$, and hence belong to the set $\{1, \ldots, mn\}$. Let $l_i = a_i m + b_i$, $0 \leqslant a_i \leqslant n-1$, $1 \leqslant b_i \leqslant m$ for $i = 1, \ldots, p$. In accordance with (16), we have

$$Y_{l_i} = \xi^{a_i} y_{b_i}.$$

Therefore, if we put $\tilde{L}_i = L_i \xi^{a_i}$ then by Definition 2, $\tilde{L}_1, \ldots, \tilde{L}_p$ is a resolving sequence for the original system (13), and $(b_1, \ldots b_p)$ is its indicator.

It will be shown in Sections 4.3, 5.2 that for finding certain kinds of solutions of systems it is sufficient to know a resolving sequence for (14), and that it is even not necessary to know its indicator.

3.4 Resolving sequences and cyclic vectors

If a resolving sequence consists of a single operator L_1 then, as we mentioned in Section 1, the row-vector c which is used to construct L_1 is a cyclic vector. In Section 1 we have compared resolving sequences and cyclic vectors, so it is not necessary to repeat that here. But our attention was aimed at normal systems of the form (5). As for constructing a cyclic vector for a system of the form (13), there the preliminary transformation of the original system to one with an invertible leading matrix gives rise to extra solutions that have to be thrown away – which is not an easy task. In some cases this can annul the possible advantages of the cyclic method.

Besides this, our experiments show that even for normal first-order systems the resolving sequence approach works faster than the cyclic vector approach — see Example 10.

4 Hypergeometric solutions

4.1 First-order normal systems

Let K be an algebraically closed field of characteristic 0. Let $\mathbb{K} = K(x)$. Denote by H_K the K-linear space of finite linear combinations of hypergeometric terms over K (i.e., $\frac{h(x+1)}{h(x)} \in K(x)$ for each hypergeometric term $h(x)$ under consideration) with coefficients in K.

Let E be the shift operator (a concretization of σ): $Ev(x) = v(x+1)$, and let $A(x)$ be an $m \times m$ matrix whose entries are in $K(x)$. We consider systems of the form

$$Ey = A(x)y, \;\; y = (y_1(x), \dots, y_m(x))^T. \tag{19}$$

Using the resolving sequences of operators, we can describe an algorithm which for a given system of the form (19) constructs a basis for the space of its solutions belonging to H_K^m. The basis consists of elements of the form

$$h(x)R(x), \tag{20}$$

where $h(x)$ is a hypergeometric term and $R(x) \in K(x)^m$.

We will say that an element of H_K^m is *related* to a hypergeometric term $h(x)$ if it can be represented in the form (20) (i.e., if each of its nonzero components is similar to $h(x)$).

Remark 4. *If an element of H_K^m is related to a hypergeometric term $h(x)$ then this element is related to any hypergeometric term which is similar to $h(x)$.*

4.2 Reduction to search for rational solutions

Let

$$L_1, \dots, L_p \in K(x)[E] \tag{21}$$

be a resolving sequence for system (19), i.e., for $[A(x)]$.

Proposition 3. *Let (20) be a non-zero solution of (19). Then there exists k, $1 \leqslant k \leqslant p$, such that L_k has a non-zero solution of the form $h(x)f(x)$, $h(x) \in K(x)$.*

Proof. Let $(l_1, \dots, l_p)$ be the indicator of resolving sequence (21). By definition of a resolving sequence (the case $j = p$ of Definition 2), among components of (20) having numbers $l_1, \dots, l_p$ there is at least one non-zero. Take the non-zero component with the minimal index of the form l_k, $1 \leqslant k \leqslant p$. By the case $j < p$ of Definition 2, this component is a solution of L_k. □

Thus, if (20) is a solution of $[A(x)]$ then the term $h(x)$ is similar to a hypergeometric term which satisfies at least one of operators (21). Let $h_j(x)$ satisfy such an operator. We can substitute $y(x) = h_j(x)z(x)$ into $[A(x)]$, where $z(x) = (z_1(x), \dots, z_m(x))^T$ is a new unknown vector. If $\frac{h_j(x+1)}{h_j(x)} = r_j(x) \in K(x)$ then we get the system

$$Ez(x) = \frac{1}{r_j(x)}A(x)z(x). \tag{22}$$

If $R_{j,1}(x),\dots,R_{j,s_j}(x)\in K(x)^m$ is a basis for rational solutions of system (22) then we obtain K-linearly independent hypergeometric solutions

$$h_j(x)R_{j1}(x),\dots,h_j(x)R_{j,s_j}(x) \tag{23}$$

of $[A(x)]$.

Rational solutions of (22) and hypergeometric solutions of the scalar equation $L_s(y)=0$ can be found by the algorithms from [2, 22, 5] resp. from [25, 23, 21]. The algorithm for finding a basis for solutions of $[A(x)]$ belonging to H_K^m is as follows:

1. Construct the resolving sequence (21) and set $\ell=\varnothing$.

2. For $s=1,\dots,p$ compute such a basis b_s for solutions of L_s belonging to H_K, that each element of b_s is a hypergeometric term. Include into ℓ those elements of b_s that are not similar to any of the elements already in ℓ.

3. For each $h_j(x)$ belonging to ℓ, use the rational function $r_j(x)=\frac{h_j(x+1)}{h_j(x)}$ in (22) to construct a basis for the space of those solutions of the system $[A(x)]$ which are related to $h_j(x)$. The union of all such bases gives a basis for the space of solutions of $[A(x)]$ that belong to H_K^m.

Remark 5. *On step 3 of the algorithm, if $h_j(x)\in K(x)$ for some j, then the system transformation leading to (22) is not needed since if a solution is related to a rational function then it is related to* 1. *More generally, if $h_j(x)$ is a hypergeometric term and $\frac{h_j(x+1)}{h_j(x)}=r_j(x)$ then we can construct the rational normal form (RNF) of $r_j(x)$, i.e., represent $r_j(x)$ in the form $U_j(x)\frac{V_j(x+1)}{V_j(x)}$ with $U_j(x),V_j(x)\in K(x)$ where $U_j(x)$ has the numerator and the denominator of minimal possible degrees [11]. We can use $U_j(x)$ instead of $r_j(x)$ in (22). In this case we must replace the hypergeometric term $\frac{1}{V_j(x)}h_j(x)$ in (23) by $h_j(x)$.*

Example 1. Let

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

With this matrix as input the algorithm proceeds as follows:

1. A resolving sequence of operators for $[A(x)]$ is

$$\begin{gathered} L_1=x^2(x-1)(x+3)(x+2)E^3- \\ (x-1)(x+1)(x^4+6x^3+12x^2+8x+4)E^2+ \\ 2x(x+2)(x^4+x^3-x^2-x-1)E- \\ x(x-1)(x+2)(x+1)(x^2-x-1), \\ L_2=E-x \end{gathered} \tag{24}$$

(with the indicator $(1,4)$ which is not of interest to us), $\ell = \varnothing$.

2. $b_1 = \left\{\frac{1}{x-1}\right\}$, $b_2 = \{\Gamma(x)\}$, $\ell = \left\{\frac{1}{x-1}, \Gamma(x)\right\}$.

3. For the first element of ℓ, we get $r_1(x) = 1$ since RNF of $\frac{x-1}{x}$ is $1\frac{1/x}{1/(x-1)}$ (Remark 5). The system (22) with $r_1(x) = 1$ has no rational solutions, thus there is no solution of the original system which is related to $\frac{1}{x-1}$.

Since $\frac{\Gamma(x+1)}{\Gamma(x)} = x$ and the RNF of x is $x\frac{1}{1}$, we use $r_2(x) = x$ in (22). This system has a one-dimensional space of rational solutions, generated by the element

$$R(x) = (0, -1, 0, 1)^T.$$

Finally, we obtain the basis of the (one-dimensional) space of all solutions of $[A(x)]$ belonging to H_K^4. It contains the single element

$$\Gamma(x)R(x) = (0, -\Gamma(x), 0, \Gamma(x))^T.$$

Remark 6. *Example 1 shows that the proposed resolving approach is not a modification of the block diagonal form algorithm [15]: if the constructed resolving operator (24) corresponds to a diagonal block of the original system then the system would have a rational solution. However, this is not the case.*

4.3 Hypergeometric solutions of higher-order systems

Let $\frac{h(x+1)}{h(x)} = r(x) \in K(x)$ for a hypergeometric solution $h(x)$ of an operator from a resolving sequence for $A_n(x)y(x+n)+\cdots+A_1(x)y(x+1)+A_0(x)y(x) = 0$ (see Section 3.3). Then the substitution $y(x) = r(x)z(x)$ produces a new system $D_n(x)z(x+n) + \cdots + D_1(x)z(x+1) + D_0(x)z(x) = 0$, where $D_i(x) = r(x+i-1)\dots r(x+1)r(x)A_i(x)$, $i = 0, 1, \dots, n$. For such a substitution, we can also use a hypergeometric solution of an operator from a resolving sequence for the normal first-order system that corresponds to the given higher-order system, since hypergeometric terms $h(x)$ and $h(x+k)$ are similar for any integer k.

In [25, 23, 21], algorithms for finding hypergeometric solutions of scalar linear difference equations with rational-function coefficients were described. Algorithms for the q-difference scalar case are in [10], [21, Sect. 12]. Solutions of this kind can be found in the differential case as well. An algorithm which, for a scalar differential equations with rational-function coefficients constructs its solutions $z(x)$ such that $\frac{z'(x)}{z(x)}$ is a rational function, is published in [20]. Algorithms for finding rational solutions of higher-order systems are described in [7] (the difference case) and in [8] (the differential case).

Example 2. Consider the following second-order system of difference equations with $m = 2$:

$$\begin{pmatrix} -\dfrac{5x^2-1}{x^2-5x+6} & 0 \\ -\dfrac{5x^2-1}{x^2-5x+6} & 0 \end{pmatrix} y(x+2) + \begin{pmatrix} 0 & \dfrac{x^3+x^2-10x+8}{x-3} \\ 0 & -\dfrac{x^3+x^2-10x+8}{x-3} \end{pmatrix} y(x+1) +$$
$$+ \begin{pmatrix} 5x^2+20x+19 & -x^3-x^2+5x-3 \\ 5x^2+20x+19 & x^3+x^2-5x+3 \end{pmatrix} y(x) = 0. \quad (25)$$

The leading matrix of this system is singular. EG-eliminations yield the following embracing system

$$\begin{pmatrix} 0 & -x^2-4x+5 \\ -\dfrac{5x^2-1}{x^2-5x+6} & 0 \end{pmatrix} y(x+2) + \\ + \begin{pmatrix} 0 & x^3+2x^2-8x \\ 0 & \dfrac{x^3+x^2-10x+8}{x-3} \end{pmatrix} y(x+1) + \\ + \begin{pmatrix} 0 & 0 \\ 5x^2+20x+19 & -x^3-x^2+5x-3 \end{pmatrix} y(x) = 0$$

which gives rise to the normal first-order system $Y(x+1) = A(x)Y(x)$ with $Y(x) = (Y_1(x), Y_2(x), Y_3(x), Y_4(x))^T = (y_1(x), y_2(x), y_1(x+1), y_2(x+2))^T$. We can construct a resolving sequence for the new system. Taking $l_1 = 2$ we obtain

$$L_1 = (x^2+4x-5)E^2 + (-x^3-2x^2+8x)E.$$

The matrix of the transformed system is

$$\tilde{A} = \begin{pmatrix} 0 & 1 \\ \dfrac{(x^2-5x+6)(5x^2+20x+19)}{5x^2-1} & 0 \end{pmatrix},$$

and $\tilde{Y} = (Y_1, Y_3)^T$. Constructing a resolving sequence for $[\tilde{A}]$, we obtain for the original system $l_2 = 1$ and

$$L_2 = (5x^2-1)E^2 + (-5x^4+5x^3+51x^2-25x-114).$$

For the equation $L_1(Y_2) = 0$ we get

$$b_1 = \left\{ \frac{\Gamma(x-3)(x-2)}{(x+3)} \right\},$$

and for $L_2(Y_1) = 0$

$$b_2 = \left\{ \Gamma(x-3)\left(x^2-\frac{1}{5}\right), (-1)^x\Gamma(x-3)\left(x^2-\frac{1}{5}\right) \right\}.$$

Thus

$$\ell = \left\{ \frac{\Gamma(x-3)(x-2)}{(x+3)}, \quad (-1)^x\Gamma(x-3)\left(x^2-\frac{1}{5}\right) \right\}.$$

For the first element of ℓ, we get $r_1(x) = x-3$ since RNF of $\frac{x^3-x^2-9x+9}{x^2+2x-8}$ is equal to $(x-3)\frac{(x-1)/(x+4)}{(x-2)/(x+3)}$. We substitute $y(x) = (x-3)z(x)$ into the original system (25). For the obtained system, a basis for the space of rational solutions is

$$\left\{ \left(x^2-\frac{1}{5}, 0\right)^T, \quad \left(0, \frac{x-2}{x+3}\right)^T \right\}.$$

For the second element of ℓ, we get $r_1(x) = -x+3$ and a basis of a space of rational solutions is

$$\left\{ \left(x^2-\frac{1}{5}, 0\right)^T \right\}.$$

Finally, the basis of the space of all solutions belonging to H_K^2 of the original system is

$$\left\{ \left(\left(x^2 - \frac{1}{5}\right) \Gamma(x-3), 0 \right)^T, \left(0, \frac{(x-2)\Gamma(x-3)}{x+3}\right)^T, \right.$$
$$\left. \left((-1)^x \left(x^2 - \frac{1}{5}\right) \Gamma(x-3), 0 \right)^T \right\}.$$

5 Formal exponential-logarithmic solutions of differential systems

5.1 The case of a first-order normal system

Let again K be an algebraically closed field of characteristic 0 and $\mathbb{K} = K(x)$. Let $\delta = \frac{d}{dx} = {}'$, and let $A(x)$ be an $m \times m$ matrix with entries in $K(x)$.

We consider systems of the form

$$y' = A(x)y, \quad y = (y_1(x), \dots, y_m(x))^T. \tag{26}$$

Using the resolving sequences of operators, we can propose an algorithm which for a given system of the form (26) constructs a basis for the space of its formal exponential-logarithmic solutions. Any solution $y(x)$ of this basis can be represented in the form

$$e^{Q(x^{-1/q})} x^\gamma \Phi(x^{1/q}), \tag{27}$$

where $Q(x^{-1/q})$ is a polynomial in $x^{-1/q}$ without constant term, $q \in \mathbb{Z}_{>0}$, $\gamma \in K$, $\Phi(t) = (\Phi_1(t), \dots, \Phi_m(t))^T$ is a column-vector with

$$\Phi_i(t) = \sum_{j=0}^{k} \left(\sum_{n=0}^{\infty} v_{i,j}(n)\, t^n \right) \ln^j t, \tag{28}$$

where $k \in \mathbb{Z}_{\geqslant 0}$, $v_{i,j}(n) \in K$. In the case when $q = 1$ and $Q = 0$ the solution (27) is a *regular solution.*

Let

$$L_1, \dots, L_p \in K(x) \left[\frac{d}{dx} \right] \tag{29}$$

be a resolving sequence for system (26), i.e., for $[A(x)]$. Similarly the difference case (see Proposition 3), if (27) is a solution of $[A(x)]$ then there is a solution of at least one of operators (29) with the same pair (q, Q). Let (q_j, Q_j) correspond to a solution of such an operator. We can substitute $y(x) = e^{Q_j(1/t)} z(t), x = t^{q_j}$ into $[A(x)]$, where $z(t) = (z_1(t), \dots, z_m(t))^T$ is a new unknown vector. Then we get the system

$$\frac{dz}{dt} = \left(q_j\, t^{q_j - 1} A(t^{q_j}) - \left(\frac{d}{dt} Q_j(1/t) \right) I_m \right) z, \tag{30}$$

where I_m is the $m \times m$ identity matrix. If $t^{\gamma_{j,1}} R_{j,1}(t), \dots, t^{\gamma_{j,s_j}} R_{j,s_j}(t)$ (where $R_{j,i} \in K((t))^m[\ln t]$) is a basis for regular solutions of system (30) then we obtain K-linearly independent formal exponential-logarithmic solutions

$$e^{Q_j(x^{-1/q_j})} x^{\gamma_{j,1}/q_j} R_{j,1}(x^{1/q_j}), \dots, e^{Q_j(x^{-1/q_j})} x^{\gamma_{j,s_j}/q_j} R_{j,s_j}(x^{1/q_j}) \tag{31}$$

of $[A(x)]$.

An algorithm for finding regular solutions of a differential system was proposed in [4]. The algorithm for finding a basis for formal exponential-logarithmic solutions of $[A(x)]$ is as follows:

1. Construct the resolving sequence (29) and set $\ell = \varnothing$.

2. For $s = 1, \dots, p$ compute b_s which is a set of all pairs (q, Q) corresponding to formal exponential-logarithmic solutions of L_s. Include into ℓ those elements of b_s that are not in ℓ.

3. For each pair (q_j, Q_j) belonging to ℓ, construct a basis for the space of regular solutions of (30). The union of all bases (31) gives a basis for the space of formal exponential-logarithmic solutions of $[A(x)]$.

5.2 The case of higher-order systems

Suppose that we are interested in formal exponential-logarithmic solutions of a full rank system

$$A_n(x)y^{(n)}(x) + \cdots + A_1(x)y'(x) + A_0(x)y(x) = 0.$$

The substitution $y(x) = e^{Q(1/t)}z(t), x = t^q$ yields a system of the form

$$D_n(t)z^{(n)}(t) + \cdots + D_1(t)z'(t) + D_0(t)z(t) = 0,$$

which is the result of some equivalent transformations of the system

$$A_n(t^q)Z_n + \cdots + A_1(t^q)Z_1 + A_0(t^q)Z_0 = 0,$$

where

$$Z_0 = z(t),$$

$$Z_i = \frac{1}{qt^{q-1}}\left(Z_{i-1}\frac{d}{dt}Q(1/t) + \frac{d}{dt}Z_{i-1}\right), \quad i = 1, \dots, n.$$

If $y_i(x)$ is a component of the vector (27) then $y_i^{(k)}(x)$ can be represented in the analogous form with the same pair (q, Q) for any non-negative integer k. Due to this we can use a resolving sequence for the corresponding first-order normal differential system instead of a resolving sequence for the original higher-order system (see Section 3.3).

Our substitutions transform the original system into new systems with coefficients in $K(t)$. To find formal exponential-logarithmic solutions of the original system it will be sufficient to find regular solutions of the obtained systems (note that those systems can have singular leading matrices). The algorithms from [4] can be used.

Example 3. The leading matrix of the system

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

is singular. EG-eliminations yield the following embracing system

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

Taking $l_1 = 1$ we obtain a resolving sequence which consists of the single operator

$$L_1 = x^5(x-2)^2(x+2)\frac{d^4}{dx^4} + x^4(x-2)(7x^2-2x-40)\frac{d^3}{dx^3} + \\ + 2x^3(4x^3-9x^2-30x+80)\frac{d^2}{dx^2} + (x-2)^3(x+2)^2\frac{d}{dx}.$$

For this operator we get

$$\ell = \left\{ (q=1, Q=0)\,, \left(q=3, Q=-\frac{3}{2}\frac{4^{\frac{1}{3}}}{x^{\frac{2}{3}}}\right), \right. \\ \left. \left(q=3, Q=-\frac{3}{2}\frac{(-1)^{\frac{2}{3}}4^{\frac{1}{3}}}{x^{\frac{2}{3}}}\right), \left(q=3, Q=\frac{3}{2}\frac{(-1)^{\frac{1}{3}}4^{\frac{1}{3}}}{x^{\frac{2}{3}}}\right) \right\}.$$

For $(q=1, Q=0)$ a substitution is not needed, and we test the existence of regular solutions of the original system. There is no such solution.

For the next three pairs, we substitute $y(x) = e^{Q(1/t)}z(t), x = t^3$ into the original system. We get three new systems whose spaces of regular solutions are one-dimensional and have the bases

$$\left\{ \left(-\frac{4^{1/3}}{2} + O(t), t^5 + O(t^6) \right)^T \right\}, \left\{ \left(-\frac{(-1)^{2/3}4^{1/3}}{2} + O(t), t^5 + O(t^6) \right)^T \right\}, \\ \left\{ \left(\frac{(-1)^{1/3}4^{1/3}}{2} + O(t), t^5 + O(t^6) \right)^T \right\}.$$

Thus, the space of formal exponential-logarithmic solutions of the original system has the basis

$$\left\{ e^{-\frac{3}{2}\frac{4^{1/3}}{x^{2/3}}} \left(-\frac{4^{1/3}}{2} + O(x^{1/3}), x^{5/3} + O(x^2) \right)^T, \right. \\ e^{-\frac{3}{2}\frac{(-1)^{2/3}4^{1/3}}{x^{2/3}}} \left(-\frac{(-1)^{2/3}4^{1/3}}{2} + O(x^{1/3}), x^{5/3} + O(x^2) \right)^T, \\ \left. e^{\frac{3}{2}\frac{(-1)^{2/3}4^{1/3}}{x^{2/3}}} \left(\frac{(-1)^{1/3}4^{1/3}}{2} + O(x^{1/3}), x^{5/3} + O(x^2) \right)^T \right\}.$$

6 Implementation and experiments

6.1 Procedure ResolvingSequence

We have implemented the algorithm from Section 3 in Maple 2015 ([27]) as the procedure `ResolvingSequence`. The arguments of this procedure are a system of the form (5) or (13), a name of the unknown function, and the standard Maple representation of an Ore algebra generated by the procedure `SetOreRing` from the package `OreTools`. A system of the form (5) is represented by a square matrix with rational-function entries, while a system of the form (13) is represented by a homogeneous linear equation with matrix coefficients. The output is a list of resolving equations, i.e., a resolving sequence

$L_1, \dots, L_p$ having the indicator $(l_1, \dots, l_p)$ is represented as the list of equations $L_1(y_{l_1}) = 0, \dots, L_p(y_{l_p}) = 0$. For matrix calculations we use procedures from the package `LinearAlgebra`.

The algorithms for constructing resolving sequences in the differential and difference cases have much in common. They can be described in a unified way using Ore noncommutative polynomials (the use of such polynomials in computer algebra is described, e.g., in [19]). For this reason, the algorithm can also be used in the q-difference case — see Example 8.

Example 4. Applying `ResolvingSequence` to the matrix $A(x)$ from Example 1

```
> A1 := Matrix([[(x-1)/x, 0, -(x-1)/(x+1), 0],
               [1, 0, 2/(x+1), -x],
               [ -1, 1, x-1, 1],
               [-(x+2)/x, (x+1)/x, (x^2-x-1)/((x+1)*x),
                                               (x^2+x+1)/x]]):
```

we get:

```
> ResolvingSequence(A1, y(x), OreTools:-SetOreRing(x, 'shift'));
```

$$\begin{aligned}[\ & (-x^6 - x^5 + 4x^4 + 3x^3 - 3x^2 - 2x)y_1(x) + \\ & (2x^6 + 6x^5 + 2x^4 - 6x^3 - 6x^2 - 4x)y_1(x+1) + \\ & (-x^6 - 6x^5 - 11x^4 - 2x^3 + 8x^2 + 8x + 4)y_1(x+2) + \\ & (x^5 + 4x^4 + x^3 - 6x^2)y_1(x+3) = 0, \\ & -xy_4(x) + y_4(x+1) = 0\ ]\end{aligned}$$

Example 5. We applied `ResolvingSequence` to a system with a 16×16 matrix. We cannot present here that matrix and the corresponding result since they are too large. The matrix is such that 80% of its entries are zeros. The maximum degree of the numerators of the entries is 13. The maximum degree of their denominators is 11. The procedure finds a resolving sequence in 43.767 CPU sec[1].

Example 6. For the system from Example 2

```
> Syst1:=<<-(5*x^2-1)/(x^2-5*x+6)|0>,
          <-(5*x^2-1)/(x^2-5*x+6)|0>>.y(x+2)+
         <<0|(x^3+x^2-10*x+8)/(x-3)>,
          <0|-(x^3+x^2-10*x+8)/(x-3)>>.y(x+1)+
         <<5*x^2+20*x+19|-x^3-x^2+5*x-3>,
          <5*x^2+20*x+19|x^3+x^2-5*x+3>>.y(x)=0:
```

we set the last (optional) argument of `ResolvingSequence` as 'select_indicator' = 2 and get

```
> ResolvingSequence(Syst1, y(x),
                    OreTools:-SetOreRing(x, 'shift'),
                    'select_indicator' = 2);
```

$$\begin{aligned}[\ & (-x^3 - 2x^2 + 8x)y_2(x+1) + (x^2 + 4x - 5)y_2(x+2) = 0, \\ & (-5x^4 + 5x^3 + 51x^2 - 25x - 114)y_1(x) + (5x^2 - 1)y_1(x+2) = 0\ ]\end{aligned}$$

[1]by Maple 2015, Ubuntu 8.04.4 LTS, AMD Athlon(tm) 64 Processor 3700+, 3GB RAM

Example 7. For the differential system from Example 3 we obtain

```
> Syst2 :=<<x^5|0>,<0|0>>.diff(y(x),x$2)+
          <<0|0>,<0|1>>.diff(y(x),x)+
          <<0|-x+2>,<x+2|0>>.y(x)=0:
  ResolvingSequence(Syst2, y(x),
                    OreTools:-SetOreRing(x, 'differential'));
```

$$\begin{aligned}[\quad & (x^5 - 2x^4 - 8x^3 + 16x^2 + 16x - 32) \frac{d}{dx} y_1(x) + \\ & (8x^6 - 18x^5 - 60x^4 + 160x^3) \frac{d^2}{dx^2} y_1(x) + \\ & (7x^7 - 16x^6 - 36x^5 + 80x^4) \frac{d^3}{dx^3} y_1(x) + \\ & (x^8 - 2x^7 - 4x^6 + 8x^5) \frac{d^4}{dx^4} y_1(x) = 0 \quad]\end{aligned}$$

Example 8. It is possible to use the procedure for q-difference linear systems:

```
>Syst3 := <<0|0>,<1|0>>.y(q^3*x) = <<-1|0>,<1|2>>.y(q^2*x) +
          <<1|-1>,<x*q|-2*q>>.y(q*x) + <<x|q>,<0|0>>.y(x):
 Syst3;
```

$$\begin{bmatrix} 0 & 0 \\ 1 & 0 \end{bmatrix} \cdot y(xq^3) = \begin{bmatrix} -1 & 0 \\ 1 & 2 \end{bmatrix} \cdot y(xq^2) + \begin{bmatrix} 1 & -1 \\ xq & -2q \end{bmatrix} \cdot y(xq) + \begin{bmatrix} x & q \\ 0 & 0 \end{bmatrix} \cdot y(x)$$

```
> ResolvingSequence(Syst3, y(x),
        OreTools:-SetOreRing([x, q], 'qshift'));
```

$$\left[-q^2 x\, y_1(xq^2) - y_1(xq^3) + y_1(xq^4) = 0, -q\, y_2(xq) + y_2(xq^2) = 0 \right]$$

The code and examples of using the procedure `ResolvingSequence` are available from http://www.ccas.ru/ca/doku.php/resolvingsequence.

6.2 Hypergeometric solutions

We have implemented the algorithm to find hypergeometric solutions of a difference system. The implemented procedures are put together in the package `LRS` (Linear Recurrence Systems). The main procedure of the package is `HypergeometricSolution`.

To find a basis of hypergeometric solutions of resolving equations, the procedure `hypergeomsols` from the package `LREtools` is used. It implements the algorithm from [23]. To find a basis of rational solutions of a first-order difference system, we use the procedure `RationalSolution` from the package `LinearFunctionalSystems`. This procedure implements the algorithms from [1, 2, 3, 24]. For higher-order systems, the procedure `RationalSolution` from our package `LRS` is used, it implements the algorithms from [7]. To perform RNF transformation, we use the procedure `RationalCanonicalForm` from the package `RationalNormalForms`. It implements the algorithms from [11].

The arguments of the procedure `HypergeometricSolution` can be in two different forms. The first one is the normal form: a square matrix with rational-function entries and a name of an independent variable. The second one is a homogeneous linear equation with matrix coefficients and an unknown function. The output is a list of vectors whose entries are hypergeometric terms.

Example 9. Applying `HypergeometricSolution` to the matrix `A1` from Example 4 we get the result in 0.303 CPU sec:

```
> st := time():
  LRS:-HypergeometricSolution(A1, x);
  time()-st;
```

$$\left[\begin{bmatrix} 0 \\ -\Gamma(x) \\ 0 \\ \Gamma(x) \end{bmatrix}\right]$$

$$0.303$$

Example 10. We applied `HypergeometricSolution` to the system mentioned in Example 5. The procedure finds a two-dimensional hypergeometric solutions space in 346.046 CPU sec. For normal systems we implemented not only the resolving procedure, but also the search for hypergeometric solutions based on the cyclic vector approach. A comparison of the timings obtained with the two implementations is given in Table 1, with time needed for the initial phase in which a resolving sequence resp. a cyclic vector is constructed shown in the first column of the table.

	initial phase	total time
Resolving sequence approach	43.767 sec	346.046 sec
Cyclic vector approach	94.776 sec	1063.747 sec

Table 1: CPU times needed to find a basis of the hypergeometric solutions space for the system from Example 5.

Example 11. For the system from Example 6 we get

```
> st := time();
 LRS:-HypergeometricSolution(Syst1, y(x));
 time()-st;
```

$$\left[\begin{bmatrix} (-1)^x\Gamma(x-3)\left(x^2-\frac{1}{5}\right) \\ 0 \end{bmatrix}, \begin{bmatrix} \Gamma(x-3)\left(x^2-\frac{1}{5}\right) \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ \frac{\Gamma(x-3)(x-2)}{x+3} \end{bmatrix}\right]$$

$$0.296$$

The code and examples of applications of `HypergeometricSolution` are available from http://www.ccas.ru/ca/doku.php/lrs.

6.3 Formal exponential-logarithmic solutions

We have implemented the algorithm for finding formal exponential–logarithmic solutions of a differential system as the procedure `FormalSolution`.

In the implementation, we use the procedure `EG_delta` from the package `EG` to construct an embracing system. This package is described in [9], and the code of the package and some examples are available from http://www.ccas.ru/ca/doku.php/eg. We use also the procedure `gen_exp` from the package `DEtools` for constructing all exponential parts $Q(x^{-1/q})$ for solutions of the resolving equations, the procedure `dchange` from the package `PDEtools` to substitute $y(x) = e^{Q(1/t)}z(t), x = t^q$, and the procedure `RegularSolutions` from the package `LinearFunctionalSystems` for finding a basis of the space of regular solutions.

The arguments of the procedure `FormalSolution` can be in two different forms. The first one is the normal form: a square matrix with rational-function entries and a name of an independent variable. The second one is a homogeneous linear equation with matrix coefficients and an unknown function. The output is a list of pairs. The first element of a pair is an expression of the form $e^{Q(x^{-1/q})}$, the second element is a vector whose entries are truncated series $x^{\gamma/q}\,\Phi_i(x^{1/q})$, see (27) and (28).

Example 12. For the differential system

```
> Syst4 := <<-15*x+4        | 60*x^3-120*x^2>,
             <60*x^4+120*x^3|   -15*x+4     >>.y(x) +
            <<60*x^3|   0   >,
             <  0   | 60*x^3>>.diff(y(x), x):
```

we have

```
> FormalSolution(Syst4, y(x));
```

$$\left[\left[e^{-\frac{1}{4x}+\frac{1}{30x^2}}, \begin{bmatrix} \ln(x)+O(x)) \\ \\ \dfrac{1}{2}+O(x) \end{bmatrix}\right], \left[e^{-\frac{1}{4x}+\frac{1}{30x^2}}, \begin{bmatrix} 1+O(x) \\ \\ O(x) \end{bmatrix}\right]\right]$$

Example 13. We tested `FormalSolution` for a third-order system with 6×6 matrix coefficients, the leading matrix is not invertible. The procedures find a resolving sequence (it consists of four resolving equations) in 428.027 CPU sec, two different exponential parts for the resolving equations in 0.127 sec, eight-dimensional regular solution space in 5.117 sec and two-dimensional solution space with exponential part $e^{-\frac{2601}{92416x}-\frac{621}{1216x^2}}$ in 0.613 sec. The total time is 434.746 sec.

The code of the procedure together with examples of the use are available from http://www.ccas.ru/ca/doku.php/formalsolution.

References

[1] S. Abramov. EG-Eliminations, *J. Difference Equations Appl.* **5** (1999) 393–433.

[2] S. Abramov, M.A. Barkatou. Rational solutions of first-order linear difference systems, *ISSAC'98 Proceedings* (1998) 124–131.

[3] S. Abramov, M.A. Bronstein. On solutions of linear functional systems, *ISSAC'2001 Proceedings* (2001) 1–6.

[4] S. Abramov, M. Bronstein, D. Khmelnov. On regular and logarithmic solutions of ordinary linar differential systems. In *Proc. of CASC'05* Lecture Notes in Computer Science **3718**, Springer Verlag (2005) 1–12.

[5] S. Abramov, A. Gheffar, D. Khmelnov. Factorization of polynomials and gcd computations for finding universal denominators. CASC'2010 Proceedings (2010) 4–18

[6] S. Abramov, A. Gheffar, D. Khmelnov. Rational Solutions of linear difference equations: universal denominators and denominator bounds. Programming and Computer Software, No 2 (2011) 78–86.

[7] S. Abramov, D. Khmelnov. Denominators of rational solutions of linear difference systems of arbitrary order. Programming and Computer Software, No 2 (2012) 84–91.

[8] S. Abramov, D. Khmelnov. On singular points of solutions of linear differential systems with polynomial coefficients, *J. of Mathematical Sciences* **185** No. 3 (2012) 347–359 (Translated from *Fundamentalnaya i Prikladnaya Matematika* **17** No. 1, (2011/12) 3–21)

[9] S. Abramov, D. Khmelnov. Linear differential and difference systems: EG_δ- and EG_σ-eliminations. Programming and Computer Software, No 2 (2013) 91-109.

[10] S. Abramov, P. Paule, M. Petkovšek. q-Hypergeometric solutions of q-difference equations, *Disctrete Math.* 180, (1998) 3–22.

[11] S. Abramov, M. Petkovšek. Rational normal forms and minimal representation of hypergeometric terms, *J. Symb. Comp.* **33** (2002) 521–543.

[12] S. Abramov, M. Petkovšek, A. Ryabenko, Hypergeometric solutions of first-order linear difference systems with rational-function coefficients, *CASC'2015 Proceedings*, Lecture Notes in Computer Science **9301**, Springer Verlag (2015) 1–14.

[13] M.A. Barkatou. A fast algorithm to compute the rational solutions of systems of linear differential equations. RR 973–M– Mars 1997, IMAG–LMC, Grenoble (1997).

[14] M.A. Barkatou. Rational solutions of matrix difference equations: problem of equivalence and factorization. In: ISSAC'99 Proceedings, ACM Press (1999) 277–282

[15] M.A. Barkatou. An algorithm for computing a companion block diagonal form for a system of linear differential equations, *AAECC* **4** (1993) 185–195.

[16] M.A. Barkatou. An algorithm to compute the exponential part of a formal fundamental matrix solution of a linear differential system, *AAECC* **8** (1997) 1–23.

[17] M.A. Barkatou. Hypergeometric solutions of linear difference systems, http://www.ricam.oeaw.ac.at/conferences/aca08/aadios.html

[18] A. Bostan, F. Chyzak, E. de Panafieu. Complexity estimates for two uncoupling algorithms, *ISSAC'2013 Proceedings* (2013) 85–92.

[19] M. Bronstein, M. Petkovšek. An introduction to pseudo-linear algebra. Theoret. Comput. Sci. **157** (1996) 3–33.

[20] T. Cluzeau, M. van Hoeij. A modular algorithm to compute the exponential solutions of a linear differential operator, *J. Symb. Comp.* **38** (2004) 1043–1076.

[21] T. Cluzeau, M. van Hoeij. Computing hypergeometric solutions of linear difference equations, *AAECC* **17** (2006) 83–115.

[22] M. van Hoeij. Rational solutions of linear difference equations. In: ISSAC'98 Proceedings, ACM Press (1998) 120–123

[23] M. van Hoeij. Finite singularities and hypergeometric solutions of linear recurrence equations, *J. Pure Appl. Algebra* **139** (1999), 109–131.

[24] D. Khmelnov. Search for polynomial solutions of linear functional systems by means of induced recurrences, *Programming Comput. Software* **30**, no. 2 (2004) 61–67.

[25] M. Petkovšek. Hypergeometric solutions of linear recurrences with polynomial coefficients, *J. Symb. Comp.* **14** (1992) 243–264.

[26] A. Storjohann. High-order lifting and integrality certification *J. Symb. Comp.* **36** (2003) 613–648.

[27] Maple online help: http://www.maplesoft.com/support/help/