*Journal of Mathematical Sciences, Vol. 185, No. 3, September, 2012*

# ON SINGULAR POINTS OF SOLUTIONS OF LINEAR DIFFERENTIAL SYSTEMS WITH POLYNOMIAL COEFFICIENTS

**S. A. Abramov and D. E. Khmelnov** UDC 512.628.2

Abstract. 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 $\alpha$, the equality $d(\alpha) = 0$ is also satisfied.

## 1. Introduction

Linear differential equations with variable coefficients and systems of such equations appear in many areas of mathematics. Solving systems leads, however, to specific difficulties which do not appear in the scalar case. Consider the equation

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

First, suppose that this is a scalar equation. In this case, the coefficients $P_0(x), P_1(x), \ldots, P_r(x)$ are polynomials over a field $k$ of zero characteristic, and $P_r(x)$ is not identically zero. It is well known that if a solution of (1) has a singularity at some point $\alpha$ (e.g., this solution is a formal Laurent series in $x - \alpha$ with a nonzero polar part), then $P_r(\alpha) = 0$. The nonzero polynomial $P_r(x)$ vanishes only for a finite set of values of $x$, and all these values can be examined step-by-step. If (1) is instead a system, $y = (y_1, y_2, \ldots, y_m)^{\mathrm{T}}$ is a column vector of unknown functions of $x$, and

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

are square $(m \times m)$-matrices with entries in $k[x]$, then the role that is played by the roots of the polynomial coefficient of $y^{(r)}(x)$ in the scalar case can now be played by the roots of the determinants of the *leading* matrix $P_r(x)$, provided that this determinant is not identically zero. However, if it is identically zero, then the situation becomes difficult. This situation will be studied in this paper.

Under the condition that the system's equations are independent over $k[x, d/dx]$, we solve the problem of the construction of a *revealing* polynomial for a given system $S$ of the form (1), i.e., of such a polynomial $d(x)$ that if, for example, for some $\alpha \in \bar{k}$ the system $S$ has the solution $y(x) \in \bar{k}((x-\alpha))^m$ and some of the components of the solution has a nonzero polar part (in this case we may say that $\alpha$ is the pole of $y(x)$), then $d(\alpha) = 0$; that polynomial may also have roots that do not correspond to any such $\alpha$; therefore, such a polynomial might be called an *embracing* one. If we discard the condition that the equations of the original system are independent over $k[x, d/dx]$, then we may face the situation where the set of the singular points of solutions of a given system is infinite. Let, e.g., the system be

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

$y = (y_1, y_2)^{\mathrm{T}}$, and $k$ be an arbitrary field of characteristic zero. Then any point is singular for some solution such that $y_1 = y_2$ (while the equations of the system are independent over $k$).

Translated from Fundamentalnaya i Prikladnaya Matematika, Vol. 17, No. 1, pp. 3–21, 2011/12.

We start with the algorithm $\mathrm{EG}_\delta$ of performing *revealing transformation* of a given system $S$ with respect to the leading matrix (Sec. 2.1). The algorithm finds a system $S'$ of the same form (and for the same unknown functions) such that the determinant of its leading matrix is in $k[x] \setminus \{0\}$, and the set of the solutions of the system $S'$ contains the set of solutions of the system $S$ as a subset. The algorithm Singsys, based on the algorithm $\mathrm{EG}_\delta$, constructs a revealing polynomial for a given system $S$. We propose randomized versions of these two algorithms (Sec. 2.4); it turns out that the randomization allows one to decrease the degree of the revealing polynomial in many cases.

The operation of differential shift that we introduce in Sec. 2.1 allows one to use the ideas of the algorithms EG and EG′ [1–4] in the algorithm $\mathrm{EG}_\delta$. The problem of constructing the revealing polynomial is not solved directly by means of the algorithms EG and EG′, since they are designed for the recurrence systems (see Sec. 3.1). The situation with the algorithm from [10] is similar. The algorithm from [11] is applicable to the differential systems of the form (1), but in this case it may enable the revealing transformation of the system with respect to its *trailing* matrix (the nonzero matrix $P_i(x)$ with the smallest index).

Note that defining potential singularities of solutions of higher-order linear differential systems does not seem to have been done elsewhere in the literature, although it could be probably derived from the papers [9, 12]. Nevertheless, the algorithms $\mathrm{EG}_\delta$ and Singsys are focused directly on constructing the revealing polynomial and are based on simple calculations. The implementation of the algorithms is presented in Sec. 2.2.

If $r = 1$ in (1), then it is possible to present an algorithm that is an alternative to the algorithm $\mathrm{EG}_\delta$. Theoretically, the system (1) with an arbitrary value of $r$ may be transformed to a first-order system. In Sec. 2.3, we show that for large values of $r$ the algorithm $\mathrm{EG}_\delta$ has less complexity than the algorithm based on the transformation to a first-order system.

In Appendix (Sec. 3), we consider some problems related to the algorithms $\mathrm{EG}_\delta$ and Singsys.

The algorithm $\mathrm{EG}_\delta$ is a differential version of the algorithm EG′. It should be noted that the notion "a differential version of the algorithm EG′" is used in [8] for an algorithm solving another problem, which is different from the problem of performing a revealing transformation of a differential system. We discuss it in Sec. 3.2 (the new name $\mathrm{EG}_\sigma$ is introduced in Sec. 3.1 for the algorithm EG′). As an illustration of the application of the algorithms $\mathrm{EG}_\delta$ and Singsys, the problem of searching for the rational function solutions of the differential systems is considered in Sec. 3.3.

If an inhomogeneous system with the right-hand side belonging to $k[x]^m$ is given, then by adding a component $y_{m+1}$ to $y(x)$ with value 1, $y'_{m+1} = 0$, one can transform the given system into a homogeneous system $S_1$ with $m + 1$ equations and $m + 1$ unknown functions. For our purposes it is sufficient to find a revealing polynomial for $S_1$. Let the homogeneous system $S_2$ be obtained by dropping the right-hand sides of the equations of the original system. If the equations of $S_2$ are independent over $k[x, d/dx]$, then the equations of $S_1$ are independent as well. For these reasons, we restrict our consideration to homogeneous systems.

A brief description of the results of the paper has been presented in [6].

## 2. Algorithms $\mathrm{EG}_\delta$ and Singsys

As it was stated above, we suppose that the equations of the original differential system are independent over $k[x, d/dx]$.

**2.1. Alternating "Reduction + Differential Shift" Steps.** Let $r$ and $m$ be arbitrary nonnegative integer numbers. A system and the corresponding matrices $P_0(x), P_1(x), \ldots, P_r(x)$ are as in, respectively, (1) and (2). The matrix

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

is called the *explicit* matrix of the system, with $P_r(x)$ called the *leading part* of the explicit matrix.

Let the $i$th row of $P_s(x)$, $0 \le s \le r$, be nonzero, while the $i$th rows of $P_{s-1}(x), P_{s-2}(x), \ldots, P_0(x)$ are zero. Let in addition the $t$th element, $1 \le t < m$, be the last nonzero element of the $i$th row of $P_s(x)$.

Then $(r-s)\cdot m+t$ is the *width* of the $i$th row of $P(x)$, and the last nonzero element of the $i$th row is the *trailing* element of that row.

Let the $i$th row of $P$ be nonzero and the $i$th row of $P_r(x)$ be zero. Let $c(x)$ be the trailing element of the $i$th row of $P(x)$. Divide the differential equation corresponding to the $i$th row of $P(x)$ by $c(x)$, differentiate the received equation, and clear denominators in it. Then replace the $i$th row of $P(x)$ by the row that corresponds to the latter differential equation. This operation is the *differential shift* of the $i$th row of $P(x)$.

If the $i$th rows of $P_r(x), P_{r-1}(x), \ldots, P_{u+1}(x)$, $0 \le u < r$, are zero while the $i$th row of $P_u(x)$ is nonzero, then the $i$th row of $P_{u+1}(x)$ will be nonzero after the differential shift of the $i$th row of $P(x)$. This fact and the following lemma motivate the term "differential shift."

**Lemma 1.** *Let the $i$th row of $P_r(x)$ be zero. Then the differential shift of the $i$th row of $P(x)$ decreases the width of this row.*

*Proof.* The $i$th row of $P(x)$ is nonzero, since we assume that the rows of the matrix $P(x)$ are independent over $k[x, d/dx]$. Therefore, there exists $s$ such that $0 \le s \le m$ and the $i$th row of $P_s(x)$ is nonzero, while the $i$th rows of $P_{s-1}(x), P_{s-2}(x), \ldots, P_0(x)$ are zero. Let the $i$th row of $P_s(x)$ be

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

$t > 0$, $c(x) \in k[x] \setminus \{0\}$. Then after the differential shift this row will be

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

where $h(x)$ is a polynomial factor provided by the clearing denominators. The $i$th rows of $P_{s-1}(x)$, $P_{s-2}(x), \ldots, P_0(x)$ are still zero. □

The scheme of the proposed algorithm $\mathrm{EG}_\delta$ is the following. We use any available method to check whether the rows of the leading part of the explicit matrix are linearly dependent over $k(x)$, and if they are, to find the coefficients $v_1(x), \ldots, v_m(x) \in k[x]$ of a dependence. We select a row with the greatest width from the rows of the explicit matrix that correspond to nonzero coefficients (let it be the $i$th row). We then replace the $i$th row of the explicit matrix by the linear combination of all its rows with the coefficients $v_1(x), \ldots, v_m(x)$. As a result, the $i$th row of the leading matrix becomes zero. This stage is called a *reduction*. It is essential that no single row has increased its width due to the reduction. Then we apply the differential shift to the $i$th row and continue this process until the leading part of the explicit matrix is nonsingular. (We never get the zero row since the equations of the original system are independent over $k[x, d/dx]$.)

**Theorem 1.** *The algorithm* $\mathrm{EG}_\delta$ *terminates.*

*Proof.* As we have mentioned, no single row increases its width due to the reduction. By Lemma 1, the differential shift decreases the width of the corresponding row. Thus, the sum of all the widths is decreased by a "reduction + differential shift" step. □

We supplement the above by saying that searching for a linear dependence $\big(v_1(x), \ldots, v_m(x)\big)$ is equivalent to solving a homogeneous system of linear algebraic equations with polynomial coefficients. This problem is efficiently solved by many different computer linear algebra algorithms, in particular, with modular algorithms that resist intermediate coefficient growth well. If we obtain $s$ linearly independent solutions of the linear algebraic system, then it is possible to use all of them for reductions, which yields $s$ zero rows in the leading matrix. To do that, we first represent the $s$ dependencies as rows of an $(s \times m)$-matrix $V$ and use the first row of $V$ to zero the $i$th row of the leading part, and perform the differential shift of the $i$th row in the explicit matrix. We then transform $V$ by eliminating the $i$th element in its rows 2 through $s$, using the $i$th element of the first row as pivot. After this elimination, each

remaining row of $V$ contains the coefficients of a linear dependence of the rows $1, \dots, i-1, i+1, \dots, m$ of the leading matrix. So we may perform $s$ "reduction + differential shift" steps. The order in which we use the rows $V$ is, in fact, arbitrary, whence different heuristic strategies can be used to choose.

If it happens that the $i$th rows of the matrices $P_r(x), P_{r-1}(x), \dots, P_{u+1}(x)$, $0 \le u < r$, are zero after the reduction, while the $i$th row of the matrix $P_u(x)$ is zero and $u < r-1$, then the equation corresponding to the $i$th row may be differentiated $r-u-1$ times without the division by its trailing element, and only the last ($(r-u)$th) differentiation need to be done with the preliminary division and the subsequent clearing of denominators (the consideration was advised to the authors by M. Barkatou).

It is not excluded that the width of the explicit matrix row that is substituted by the linear combination of the other rows decreases after the reduction. In this case, it is not needed to divide the row by the trailing element before the differentiation.

The algorithm Singsys (Singularities of solutions of linear ordinary differential systems) consists in applying the algorithm $\mathrm{EG}_\delta$ to a given system, subsequent computing the determinant of the leading matrix of the transformed system, and freeing the resulting polynomial from squares.

**Example 1.** Let us consider the system $S$

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

with the explicit matrix

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

The rows of the leading part of the matrix (5) are dependent with coefficients $v_1 = -1$, $v_2 = x+1$. The rows of the explicit matrix have the same width. We replace the second row:

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

The differential shift of the second row yields

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

The matrix (6) is the explicit matrix of the system $S'$

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

whose leading matrix is nonsingular. The system is the result of the application of $\mathrm{EG}_\delta$ to the system (4). The determinant of the leading part of the explicit matrix of the system $S'$ is equal to $-2x^3(x+2)^2(x+1)$. The polynomial

$$d(x) = x(x+2)(x+1) \quad (8)$$

is the result of the algorithm Singsys.

The constructed polynomial $d(x)$ is a revealing polynomial for the original system (4), and the solution of the system may have singularities only at the points $-2$, $-1$, $0$.

The system $S'$ that is constructed by Singsys may have more solutions than $S$. If, e.g., $k = \mathbb{C}$ and the leading matrix of a system is nonsingular, then the dimension of the space of holomorphic solutions

at any domain that does not contain the roots of the determinants of the leading matrix is equal to $rm$. But if the leading matrix is singular, then this dimension may be less than $rm$. For example, the system

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

has the one-dimensional solution space $y = (c, 0)^{\mathrm{T}}$. (Applying $\mathrm{EG}_\delta$ to this system produces the system $S'$ of the form

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

having two-dimensional solution space $y = (c_1 x + c_2, c_1)^{\mathrm{T}}$.) Note that the equations of the system (9) as well as the rows of its explicit matrix are independent over $k[x, d/dx]$.

In the above, we assumed that the equations of the original system are independent over $k[x, d/dx]$ (in this case the solution space of the system has a finite dimension, not exceeding $rm$). If we discard the assumption, then the algorithm $\mathrm{EG}_\delta$ allows one to determine whether a dependence exists.

**Theorem 2.** *Let the equations of a system be dependent over $k[x, d/dx]$. Then a zero row will be obtained in the explicit matrix at some step of the execution of the algorithm* $\mathrm{EG}_\delta$.

*Proof.* We will show that if the rows of the leading matrix of the system are independent over $k[x]$, then the equations of the system are independent over $k[x, d/dx]$. Let the equations be dependent with the operator coefficients $L_1, L_2, \ldots, L_m \in k[x, d/dx]$, and

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

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

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

It is easy to see that the linear combination of the rows of the leading matrix taken with the coefficients $b_1(x), b_2(x), \ldots, b_m(x)$ is equal to zero. We mention additionally that if the equations of the system are initially dependent over $k[x, d/dx]$, then the equations that correspond to the rows of the explicit matrix are dependent over $k[x, d/dx]$ after each step of $\mathrm{EG}_\delta$. Then it follows that for the case of dependent equations the use of $\mathrm{EG}_\delta$ will lead to the appearance of a zero row in the explicit matrix at some step. □

We add some more words to the proved theorem. The transformations of the explicit matrix that are performed by the algorithm $\mathrm{EG}_\delta$ correspond to the transformations of the original system $S$, which keep the number of equations independent over $k[x, d/dx]$. If one assumes that the number is equal to $m_0 \leq m$, then, using $\mathrm{EG}_\delta$, one can construct a system $S'$ of $m_0$ equations independent over $k[x, d/dx]$ such that its leading matrix of size $m_0 \times m$ has the rank $m_0$ over $k[x]$, and each solution of the system $S$ is a solution of the system $S'$.

Theoretically, it is possible to declare that the algorithms $\mathrm{EG}_\delta$ and Singsys are applicable to differential systems with any analytic coefficients. However, in this case one must recognize the equality of the explicit matrix elements to zero, which cannot be algorithmically done in the general case. In addition, the result of the Singsys execution may be an analytic function $d(x)$ (we can drop the step of freeing from squares), and it is not excluded that the equation $d(x) = 0$ has infinitely many roots.

**2.2. Implementation of the Algorithms $\mathrm{EG}_\delta$ and Singsys.** Our implementation (see `http://www.ccas.ru/sabramov/singsys/`) of the algorithm $\mathrm{EG}_\delta$ in Maple [13] is based on our implementation of the algorithm $\mathrm{EG}'$ described in [4]. The main difference of the algorithms is the shift step, and the proposed implementation includes the differential shift of the row as an auxiliary procedure. Some simplifications are also performed, since the algorithm $\mathrm{EG}_\delta$ is focused only on the revealing transformations of the system with respect to the leading matrix.

The algorithm is implemented as the procedure `EG_delta`; its input parameters are the system given as the list of its equations and the list of the corresponding unknown functions. The name of a variable that will store the obtained system (like the system $S'$ from Example 1) is an optional parameter.

The algorithm Singsys is implemented as the procedure `Singsys` based on the use of the procedure `EG_delta`. The input parameters of the procedure `Singsys` are also a system given as the list of its equations and the list of the corresponding unknown functions.

With these procedures one can find the polynomial whose roots identify the potential singularities of the solutions of the system (4) from Example 1, and also get the system with a nonsingular leading matrix, which is the result of the corresponding revealing transformation:

```
> 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);
```

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

Our experiments show that the algorithm allows one to work with systems with rather large input parameters. For example, two runs of experiments were executed. Seven sets with 10 differential systems in each were generated for each of the runs; we took $m = 10$ and $r = 5, 10, 20, 40, 100, 250, 500$ correspondingly. The coefficients of all the systems were random polynomials (we used the standard Maple command `randpoly()`); the systems were generated in such a way that the number of nonzero coefficients was 30% in the first run, and 50% in the second run. The results of the experiments are represented in the table. Each cell contains the total time (in seconds) used for constructing the revealing polynomials for all systems in the corresponding set of a run (for all the experiments: 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. First-Order Systems.** If $r = 1$ in (1), i.e., the system is of the form

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

and the rank of $P_1(x)$ over $k[x]$ is equal to $s$, $0 < s < m$, then after the reduction we can rewrite the system as a pair of a first-order linear differential system and a linear algebraic system

$$B_1(x)y' + B_0(x)y = 0, \quad R(x)y = 0,$$

where the matrices $B_1(x)$, $B_0(x)$ have the size $s \times m$, and the rank of $B_1(x)$ is equal to $s$, while the matrix $R(x)$ has the size $(m-s) \times m$. The rank of $R(x)$ is equal to $m - s$, since equations of the original system are independent over $k((x))$.

We can use a version of the approach described in [14]. If we differentiate the system $R(x)y = 0$, then we obtain

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

Using (11), we eliminate in $B_1(x)y' + B_0(x)y = 0$ some of $y_i'$ for $m - s$ values of the index $i$. Then, using equations of the algebraic system $R(x)y = 0$, we eliminate all $y_i$ having the same value of the index $i$. If the obtained differential system has the leading matrix of rank less than $s$, then we repeat these actions (we transform the differential system to a pair that consists of differential and algebraic systems) and so on. Finally, we obtain a first-order differential system with a nonsingular leading matrix. It might be represented by a transformation matrix; the common denominator of the elements of the matrix also makes a contribution to the revealing polynomial.

It is well known that any system of the form (1) can be transformed to the form (10) by introducing some new unknown functions, and the order of matrices $P_0(x)$, $P_1(x)$ will be equal to $mr$. However, it is also well known that for large values of $r$ such a transformation from the system (1) to the first-order system is not convenient, since it leads to matrices of large size, and work with such matrices is quite expensive. Suppose that complexity (the number of operations in $k[x]$ in the worst case) of the reduction of the leading matrix of order $m$ is $m^\omega$, $2 < \omega \le 3$. The number of steps "reduction + differential shift" of the algorithm $\mathrm{EG}_\delta$ (Sec. 2.1) does not exceed $rm^2$, and the cost of each step is at most $rm + m^\omega$. Thus, the complexity of $\mathrm{EG}_\delta$ does not exceed $r^2m^3 + rm^{\omega+2}$. On the other hand, if we use the described transformation of the original system to the first-order system, then the total cost of reductions can be not less than

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

If we suppose that $m$ is fixed and $r$ tends to $\infty$, then the complexity of $\mathrm{EG}_\delta$ is $O(r^2)$, while the complexity of the algorithm based on the transformation to the first-order system grows as $r^{\omega+1}$, $\omega > 2$, or even faster.

The upper bound $rm^2$ for the number of steps of $\mathrm{EG}_\delta$ is most likely too large (our experiments confirm this). In any case, if each differential shift causes an additional solution of the system to appear, then the number of steps cannot be bigger than $rm$.

Some experiments on a comparison of the algorithm Singsys and the algorithm based on the transformation to the first-order system were done. For this, 4 sets with 10 differential systems in each were generated; we took $m = 10$ and $r = 5, 10, 15, 20$ correspondingly. As in the experiment from Sec. 2.2, the coefficients of all the systems were random polynomials; the systems were generated in such a way that the number of nonzero coefficients was 50%. The results of the experiments are represented in the table. Each cell contains the total time (in seconds) used for constructing the revealing polynomials for all systems in the corresponding set by one of the comparing algorithms.

| | 5 | 10 | 15 | 20 |
|---|---|---|---|---|
| Singsys | 6.422 | 17.375 | 21.344 | 29.422 |
| First order | 27.267 | 731.484 | 3287.844 | 6383.953 |

**2.4. Randomization.** Reductions and differential shifts may lead to some excess factors in the revealing polynomial, the roots of which are not singularities of original system solutions. Some additional work allows one to avoid at least a part of them.

We introduce into the algorithms $\mathrm{EG}_\delta$ and Singsys some randomness. Let $0 < p \le 1$. The differential shift will be performed as described in Sec. 2.1 with probability $p$. The corresponding equation will be differentiated without the preliminary division by the trailing coefficient with probability $1 - p$. This shift will be called the *p-shift* (if $p = 1$, then $p$-shift is the differential shift).

After each step "reduction + $p$-shift," the sum of widths of all rows is decreased by at least $p$ in average. Since $p > 0$, the total average time of performing the revealing transformation of the explicit matrix based on "reduction + $p$-shift" steps is finite. Thus, we obtain a randomized version of $\mathrm{EG}_\delta$.

The following scheme can be used for constructing a revealing polynomial. First, we apply the version of $\mathrm{EG}_\delta$ described in Sec. 2.1. Then we apply the randomized version of $\mathrm{EG}_\delta$ a few times without changing the value $p$ (and everything that was computed before the first differential shift remains the same). Each such application can produce a new leading part of the explicit matrix. The greatest common divisor of the obtained revealing polynomials may have a smaller degree. We terminate the process when the current application does not change this degree or when the revealing polynomial is of zero degree. This gives a randomized version of Singsys.

**Example 2.** Consider the system

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

Using $\mathrm{EG}_\delta$, we obtain

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

and $d_1(x) = x(x+2)(x-2)$ is the result of Singsys.

Using the randomized version of $\mathrm{EG}_\delta$, we obtain other systems as well, in particular,

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

The result of the randomized Singsys is equal to $d(x) = x(x+2)$, this polynomial is equal to the greatest common divisor of $d_1(x)$ and $d_2(x) = x(x+2)(x-1)$ (the latter polynomial corresponds to (12)). Further systems that we could get, using the randomized $\mathrm{EG}_\delta$, have revealing polynomials $d_3(x) = x(x+2)(x-1)(x^2+4x-2)$ and $d_4(x) = x(x+2)$ and do not lead to decreasing the degree of the resulting revealing polynomial; note that $d_4(x)$ coincides in this case with the final result of the randomized Singsys.

The randomized versions of the algorithms $\mathrm{EG}_\delta$ and Singsys are included in our implementation, described in Sec. 2.2. The procedures `EG_delta` and `Singsys` accept an optional parameter that determines if the randomization should be used. The probability of the division in the differential shift in the randomized version of the algorithm $\mathrm{EG}_\delta$ is taken to be 1/2.

Apply the randomized version to the system from Example 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);
```

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

This shows that the randomization may allow one to obtain a revealing polynomial of smaller degree; however, it may give the same result as the nonrandomized version.

## 3. Appendix: Combined Work of the Algorithms $\mathrm{EG}_\delta$ and $\mathrm{EG}_\sigma$

**3.1. Algorithm $\mathrm{EG}_\sigma$.** Further on, we will deal with recurrence systems of the form

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

where $l \geq t$ are arbitrary integers, $z(n) = \big(z(n)_1, \ldots, z_m(n)\big)^{\mathrm{T}}$ is a column vector of unknown sequences, and $Q_l(n), \ldots, Q_t(n)$ are square $(m \times m)$-matrices with entries in $k[n]$. The nonzero matrices $Q_l(n)$ and $Q_t(n)$ are called *leading* and *trailing* matrices of the system (13). In many cases, it is natural to consider the system (13) together with a finite set of *linear constraints*, i.e., linear relations, each of which contains a finite set of variables $z_j(i)$. Let $S$ and $S'$ be systems of the form (13) and $C$ and $C'$ be finite sets of linear constraints. We then say that the systems $(S, C)$ and $(S', C')$ are equivalent if the space of solutions of $S$ that satisfy $C$ is the same as the space of solutions of $S'$ that satisfy $C'$. Finitely many linear constraints are easily taken into account when determining various properties of a system and when computing its solutions. The algorithms EG [1] and EG′ [2–4] solve the problems of computing a system $(S', C')$ that is equivalent to $(S, \varnothing)$ and such that the leading or trailing matrix of $S'$ is nonsingular (EG′ is an improved version of EG).

As we have mentioned in Sec. 1, the algorithm $\mathrm{EG}_\delta$ uses the main ideas of EG and EG′.

We propose a consistent new name for these two similar algorithms treating differential and difference systems. That is why we introduce the new name $\mathrm{EG}_\sigma$ for the algorithm EG′. The symbols $\delta$ and $\sigma$ for the mappings that possess the properties of the differentiation and the shift correspondingly are used, e.g., in the theory of Ore polynomials.

**3.2. Recurrence Systems for Coefficients of Formal Laurent Series Solutions.** Linear recurrence systems can appear when handling differential systems with polynomial coefficients. Such differential systems induce recurrence systems of the form (13) for the coefficients of their series solutions. The transformation

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

can be used to construct the recurrence system ($\phi$ is the shift operator: $\phi(z_n) = z_{n+1}$). The substitution $x + \alpha$ for $x$ into the differential system reduces the computation of solutions at the point $\alpha$ to that of solutions at the point 0. The roots of the determinants of the matrices $Q_l(n)$ and $Q_t(n)$ (when those matrices are nonsingular) are always important for determining the structure of the solution space of the original differential system at 0.

The algorithms $\mathrm{EG}_\sigma$ and Singsys often work in combination. It should be noted that if a given differential system has a nonsingular leading matrix, then there is no guarantee that the leading matrix of the induced recurrence system is also nonsingular [1, Example 8], and vice versa: for the differential system

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

the corresponding induced recurrence system

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

has a nonsingular leading matrix. There exist also simple examples where the leading matrices of the differential and induced recurrence systems are both (non)singular.

Thus, in the case where the leading (trailing) matrix of the induced system is singular, we use $\mathrm{EG}_\sigma$. The supplementing finite set of linear constraints $C$ helps to exclude some of the roots of the determinant of the resulting nonsingular matrix, which are not the orders of Laurent series solutions.

**Example 3.** An induced recurrence system for the differential system

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

is

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

The revealing transformation of the latter system with respect to the leading matrix leads to

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

accompanied by the linear constraint

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

The determinant of the leading matrix of the system (16) has the roots 0, $-2$, but the maximal root does not correspond to any Laurent series solution of the system (15): the relation (17) and the first equation of the system (16) show that if $z(-1) = z(-2) = 0$, then $z(0) = 0$ as well. As for the root $-2$, the corresponding Laurent series solutions may be easily constructed. Choose $z(-2)$ so that

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

holds. As a basis solution of the algebraic system we may take, e.g., $z_1(-2) = (1, -1)^{\mathrm{T}}$. From (16) we obtain

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

leading to $z_1(-1) = (0, -1)^{\mathrm{T}}$. For $n = 0$ the system (16) takes the form

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

and jointly with (17) it gives $z_1(0) = (1/2, 1/2)^{\mathrm{T}}$. Using (16), we obtain

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

for $n \geq 1$.

Thus, the differential system (15) has a one-dimensional space of Laurent series solutions in the point $x = 0$, its basis may be given by the series

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

where

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

and

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

when $n \geq 1$.

The additional inclusion of linear constraints with noninteger values of the argument $n$ into the set $C$ enables avoiding some values of $\lambda$, which are candidates for the role of an exponent in (19).

In [8, Sec. 6], there is the observation that the transformations performed by $\mathrm{EG}_\sigma$ in the recurrence system correspond to some well-defined transformations in the original differential systems. This yields, according to the authors of the paper [8], a differential version of $\mathrm{EG}_\sigma$ that works without involvement of the recurrence system.

This approach can be useful if a small number of terms of Laurent series is needed. However, when this number is large, the induced recurrence system seems a more reasonable and effective tool (in Example 3

we obtained the recurrent formula (18), which is convenient for computation). Moreover, the original version of $\mathrm{EG}_\sigma$ gives the supplementing finite set of linear constraints $C$ that allows one to avoid the consideration of some of the roots of the determinant of the resulting nonsingular leading matrix of the recurrence system (in Example 3 the set consisted of the only one relation (17) that enabled to avoid the consideration of the root $-2$ and, hence, to avoid extra solutions).

The main problem solved in [8] is the construction of regular solutions of the original differential system, i.e., the solutions of the form

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

where $\lambda \in \bar{k}$, $v(x) \in \bar{k}[[x]]^m[\log x]$. Since the set of constraints $C$ is not considered, the removal of the resulting extra solutions is proposed to be done with the substitution to the equations of the original system. But if the recurrence system and the set $C$ are constructed, then, as it seems to us, this removal is costlier than taking into account the constraints similar to (17). In [8], all the series appearing in a regular solution are represented in a truncated form, which makes the substitution for removing the extra solution even more difficult.

The problem of the construction of regular solutions of higher-order differential systems was first solved completely in [5], where the original version of $\mathrm{EG}_\sigma$ was successfully used at an appropriate step of the algorithm. The implementation of the algorithm from [5] as the procedures `RegularSolution` and `ExtendRegularSolution` is a part of `LinearFunctionalSystems` package in Maple starting from release 10.

**3.3. Rational Function Solutions.** Let the roots of the revealing polynomial $d(x)$ for a given differential system $S$ be found. For each of the roots we can find (applying $\mathrm{EG}_\sigma$ to the corresponding induced recurrence system) a lower bound for the valuations of Laurent series solutions of $S$ at this point or to detect that there is no such solution. In the latter case, $S$ has no rational (i.e., rational function) solution. Otherwise, the lower bounds found allow one to obtain a *denominator bound* for rational solutions of $S$, i.e., a rational function $U(x)$ such that the components of any rational solution $\big(y_1(x), y_2(x), \dots, y_m(x)\big)^{\mathrm{T}}$ can be represented in the form $y_i(x) = p_i(x)U(x)$, where $p_i(x) \in k[x]$, $i = 1, 2, \dots, m$. Then one can substitute $\tilde{y}(x)U(x)$ for $y(x)$ into $S$ and find the polynomial solutions of the obtained system. An upper bound for degrees of the polynomial solutions can be found by constructing a recurrence system for the coefficients of such solutions, performing a revealing transformation of the system with respect to its trailing matrix with $\mathrm{EG}_\sigma$ and investigating the integer roots of the determinant of the obtained trailing matrix (see [1, Secs. 3.3, 3.4]).

**Example 4.** Let us find rational solutions of the system (4). By constructing induced recurrence systems at the points $-2$, $-1$, and 0 (the roots of the polynomial (8)), performing a revealing transformation of the system with respect to its leading matrix with $\mathrm{EG}_\sigma$ and analyzing the roots of the determinants of the resulting leading matrices, the denominator bound $U(x) = x/(x+2)^2$ of rational solutions of the system (4) is found.

It remains to find polynomial solutions of the differential system that is obtained by the substitution $\tilde{y}(x)x/(x+2)^2$ for $y(x)$ into the system (4). After all the calculations, we obtain the space of all rational solutions of the system (4):

$$y = \left( \frac{x(c_1 + 4xc_3 + x^2c_3)}{(x+2)^2}, \frac{c_2x}{(x+2)} \right)^{\mathrm{T}}.$$

The computer algebra system Maple contains the `LinearFunctionalSystems` package, which provides the procedures for finding solutions of linear systems of ordinary equations (differential ones as well) based on the construction of induced recurrence systems and the revealing transformations of these systems with $\mathrm{EG}_\sigma$. The procedure `RationalSolutions` from the package finds rational solutions only for the systems of the form (1) having a nonsingular leading matrix $P_r(x)$, since it uses the procedure `UniversalDenominator` of the same package, which can be used to construct denominator bounds only of such systems. In our new implementation, the procedure `UniversalDenominator` uses the procedure

`Singsys` described in Sec. 2.2. Denominator bounds are constructed as described in the beginning of that section. This yields that the procedures `UniversalDenominator` and `RationalSolutions` are applicable to any system of the form (1) with equations independent over $k[x, d/dx]$. Taking `sys` and `vars` from Sec. 2.2, we get the rational functions from Example 4:

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

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

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

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

Our experiments show that it is worth using the randomized version of the algorithm Singsys in the search for the rational solution of the systems. For example, we executed an experiment, for which 3 sets with 10 differential systems in each were generated; we took $m = 10$ and $r = 5, 10, 15$ correspondingly. All the systems were constructed to have randomly generated rational solutions. The rational solutions were found for all systems in each set using the revealing polynomial construction with the randomized and nonrandomized versions of Singsys. The results of the experiments are represented in the table. Each cell contains the total time (in seconds) used for searching for the rational solution of all systems in the corresponding set by one of the approach; additionally the total time used for the auxiliary operation of the revealing polynomial construction is given in the brackets. Extra costs for the randomization are usually compensated by the savings at the next steps, which are much costlier, both in terms of the calculation time and the memory used.

| | 5 | 10 | 15 |
|---|---|---|---|
| Nonrandomized | 336.531 (7.547) | 1128.096 (14.345) | 3305.061 (24.936) |
| Randomized | 292.704 (20.640) | 958.890 (41.859) | 2887.798 (184.046) |

The algorithm from [7] is also designed to search for rational solutions, but only of systems of the form $y' = My + N$, where $M \in \mathrm{Mat}_m\big(k(x)\big)$, $N \in k(x)^m$. If the rational solution of such a system has a pole in $\alpha \in \bar{k}$, then $g(\alpha) = 0$, and $g(x)$ is a polynomial that is the common denominator of the elements of the matrix $M$ and the vector $N$.

The authors would thank M. Barkatou and E. Pflügel for useful discussions and helpful comments.

This work was supported in part by a grant from RFBR, Project No. 10-01-00249.

## REFERENCES

1. S. Abramov, "EG-eliminations," *J. Differ. Equ. Appl.*, **5**, 393–433 (1999).
2. S. Abramov and M. Bronstein, "On solutions of linear functional systems," in: B. Mourrain, ed., *Proc. of the 2001 Int. Symp. on Symbolic and Algebraic Computation, London, Ontario, Canada, July 22–25, 2001*, ACM Press, New York (2001), pp. 1–6.
3. S. A. Abramov and M. Bronstein, *Linear Algebra for Skew-Polynomial Matrices*, Rapport de Recherche INRIA, RR-4420 (March 2002).
4. S. Abramov, M. Bronstein, and D. Khmelnov, "Regularization of linear recurrence systems," *Trans. A. M. Liapunov Inst.*, **4**, 158–171 (2003).

5. S. Abramov, M. Bronstein, and D. Khmelnov, "On regular and logarithmic solutions of ordinary linear differential systems," in: *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, Springer, Berlin (2005), pp. 1–12.
6. S. Abramov and D. Khmelnov, "Desingularization of leading matrices of systems of linear ordinary differential equations with polynomial coefficients," in: *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. M. A. Barkatou, "On rational solutions of systems of linear differential equations," *J. Symbol. Comput.*, **28**, 547–567 (1999).
8. M. A. Barkatou, C. El Bacha, and T. Cluzeau, "Simple forms of higher-order linear differential systems and their applications in computing regular solutions," *J. Symbol. Comput.*, **46**, 633–658 (2011).
9. M. A. Barkatou, C. El Bacha, and E. Pflügel, "Simultaneously row- and column-reduced higher-order linear differential systems," in: W. Koepf, ed., *Symbolic and Algebraic Computation, Int. Symp., ISSAC 2010, Munich, Germany, July 25–28, 2010, Proceedings*, ACM Press (2010), pp. 45–52. ACM Press, New York (2010), pp. 45–52.
10. B. Beckermann, H. Cheng, and G. Labahn, "Fraction-free row reduction of matrices of skew polynomials," in: T. Mora, ed., *Proc. of the 2002 Int. Symp. on Symbolic and Algebraic Computation*, ACM Press, New York (2002), pp. 8–15.
11. B. Beckermann, H. Cheng, and G. Labahn, "Fraction-free row reduction of matrices of Ore polynomials," *J. Symbol. Comput.*, **41**, 513–543 (2006).
12. P. Davies, H. Cheng, and G. Labahn, "Computing Popov form of general Ore polynomial matrices," in: *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), pp. 149–156.
13. Maple online help, `http://www.maplesoft.com/support/help/`.
14. M. P. Quéré and G. Villard, "An algorithm for the reduction of linear DAE," *Int. Symp. on Symbolic and Algebraic Computation, ISSAC'95. Montreal, Canada, juillet 1995*, ACM Press, New York (1995), pp. 223–231.

S. A. Abramov
Computing Centre of the Russian Academy of Science,
Vavilov str., 40, 119333, GSP-1, Moscow, Russia
E-mail: `sergeyabramov@mail.ru`

D. E. Khmelnov
Computing Centre of the Russian Academy of Science,
Vavilov str., 40, 119333, GSP-1, Moscow, Russia
E-mail: `dennis_khmelnov@mail.ru`