объединенный
институт
ядерных
исследований
дубна

E11-84-400

V.P.Gerdt, A.B.Shvachka, A.Yu.Zharkov

FORMINT – A PROGRAM FOR CLASSIFICATION OF INTEGRABLE NONLINEAR EVOLUTION EQUATIONS

Submitted to "Computer Physics Communications"

1984

PROGRAM SUMMARY

Title of program: FORMINT
Catalogue number:
Program obtainable form: CPC Program library, Queen's University of Belfast, N.Ireland (see application in this issue).
Computer: IBM 360/370.
Operating system: OS.
Programming language used: PL/1 - FORMAC.
High speed storage required: depends on the problem, minimum 160000 bytes.
No. of bits in a word: 32.
No. of lines in combined program and test deck: 344.
Keywords: Evolution equations, formal integrability, conservation laws, Lie-Backlund algebra, Backlund transformations.
Nature of physical problem
Classification of formally integrable evolution equations allows to derive all the equations with the property of nontrivial Lie-Backlund algebra or/and infinite series of nontrivial conservation laws. Some of this equations are interesting from the physical point of view due to their soliton solution.
Method of solution
Classification algorithm is based on the concept of formal integrability proposed in [1]. The most tedious steps of the algorithm are implemented in the program FORMINT written on the language of computer algebra system PL/1 - FORMAC [2].
Restrictions on the complexity of the problem
In some cases the available computer memory is the severest restriction. It may only be avoided if the problem is split into several smaller ones.
Running time
It depends heavily on the form of evolution equation and the nature of subproblem to be solved. It cannot be estimated in advance.

REFERENCES

1. Ibragimov N., Shabat A. Funct. analiz, 1980, 14, p. 79.
2. Bahr K.A. FORMAC 73 User's Manual, Darmstadt: GMD/IFV, 1973.

LONG WRITE-UP

1. INTRODUCTION

In recent years a good deal of attention has been paid to the classification of integrable nonlinear evolution equations

$$u_t = F(u, u_1, \dots, u_n), \quad u_i = \frac{d^i u}{dx^i}. \tag{1}$$

Different criteria of integrability are used for classification of equations (1): the existence of nontrivial symmetries [1,2], conservation laws [3], prolongation structures [4]. In this paper we shall describe classification method based on the concept of formal integrability [5], the latter is one of the strict formulation of the concept of L-A pair [6]. The demand of formal integrability puts strict limitations on the form of the right-hand side of (1) and allows to find all the evolution equations possessing nontrivial Lie-Backlund algebra and infinite series of nontrivial conservation laws. Among them there are equations interesting from the physical point of view due to their soliton solutions, e.g., KdV equation.

Lately the classification of formally integrable evolution equations problem is solved for the third order equations of the form $u_t = u_3 + f(u, u_1, u_2)$ [7]. A higher order equations classification demands tedious computations. To carry out them automatically we suggest the program FORMINT on the language of computer algebra system PL/1 - FORMAC [8]. The program allows one to check up the conditions of formal integrability, to obtain the equations equivalent to those on the F function, to find the nontrivial elements of Lie-Backlund algebra (symmetries), to compute the conservation law densities.

In the second section of this paper the basic concepts and results of the formally integrable system theory are given. In the third section the solving algorithms for the innumerated problems are briefly described. In the forth section one can read the description of the program FORMINT. In the fifth, terminal, section the examples of program usage are given.

2. THEORETICAL BACKGROUND

Equation (1) is called formally integrable if there is the formal series

$$L = \sum_{i=-\infty}^{1} a_i D^i, \quad a_i \equiv a_i(u, u_1, \dots, u_{k_i}), \quad D \equiv \frac{d}{dx}, \quad D^{-1}D = DD^{-1} = 1, \tag{2}$$

meeting the operator relation

$$L_t - [F_*, L] = 0, \tag{3}$$

where $F_* = \sum_{i=0}^{n} \frac{\partial F}{\partial u_i} D^i$, $L_t \equiv [\frac{d}{dt}, L]$, $\frac{d}{dt} \equiv \sum_i (D^i F) \frac{\partial}{\partial u_i}$.

Conservation law density for the equation (1) is called the function $P(u, u_1, \ldots, u_k)$ such that

$$\frac{d}{dt} P \in \mathrm{Im} D. \qquad (4)$$

The notation $\in \mathrm{Im}D$ means that the left-hand side expression is a gradient (i.e., whole derivative with respect to x). If $P \in \mathrm{Im}D$, then the conservation law is called trivial.

Lie-Backlund algebra for (1) is the set of functions $H(u, u_1, \ldots, u_m)$ such that

$$H_* F - F_* H = 0. \qquad (5)$$

Algebra is called nontrivial if it contains the elements (symmetries) different from u_1 and F.

The following theorems establish the connections between the concepts introduced above.

<u>Theorem 1.</u> If the equation (1) has the infinite Lie-Backlund algebra, then it is formally integrable [5].

<u>Theorem 2.</u> If the equation (1) has the infinite series of nontrivial conservation laws, then it is formally integrable [7].

<u>Theorem 3.</u> The formal integrability of (1) is equivalent to its property of the conservation law infinite series of the following type [5]

$$\frac{d}{dt} R_i \in \mathrm{Im}D, \quad i = -1, 0, 1, 2, 3, \ldots, \qquad (6)$$

where

$$R_{-1} = \mathrm{Res}(L^{-1}), \quad R_0 = \mathrm{Res}(L^{-1} L_t), \quad R_m = \mathrm{Res}(L^m), \quad m = 1, 2, 3, \ldots \qquad (7)$$

and $\mathrm{Res}(\sum_i a_i D^i) \equiv a_{-1}$.

Theorem 3 allows one to solve the classification problem for formally integrable evolution equations of the order n with the accuracy up to the transformations of the form $u(x,t) = \phi(v(x,t))$. The problem is being solved according to the following scheme.

1) Derivation of all the equations for which several first conditions (6) are done (primary classification).

2) Checking up the higher conditions for the equations derived.

3) Testing of the equations obtained for the property of nontrivial symmetries.

4) Derivation of the Backlund transformations which connect different equations of the list obtained.

In the next section solving algorithms for the problems 1)-4) and their implementation in FORMINT are briefly described.

3. ALGORITHMS

3.1. <u>Testing conditions of formal integrability</u>

To test the formal integrability conditions (6) for the given equation (1) is necessary primarily to express the densities R_i in terms of explicit function $F(u, u_1, \ldots, u_n)$. It can be easily shown that

$$R_{-1} = (F_n)^{-1/n}, \quad R_0 = F_{n-1} / F_n, \qquad (8)$$

where $F_i \equiv \frac{\partial F}{\partial u_i}$, and densities R_m can be derived from the following recurrent relations on the coefficients of the series $L^m = \sum_{i=-\infty}^{m} a_i D^i$

$$a_m = (F_n)^{m/n}; \quad m = 1, 2, 3, \ldots,$$

$$a_i = \frac{(F_n)^{i/n}}{n} D^{-1}[(F_n)^{-\frac{(i+n)}{n}} C_{n+i-1}|_{a_i=0} + \frac{d}{dt} a_{n+1-i}], \qquad (9)$$

$$i = m-1, m-2, m-3, \ldots,$$

where C_{n+i-1} are the coefficients of D^{n+i-1} in the commutator $[L^m, F_*]$, $a_{n+i-1} \equiv 0$ while $i > m-n+1$ and the integration constants in (9) are suggested zeros. The recurrence relations (9) may be obtained from (3) wich is valid not only for L but also for L^m.

Then one has to check up that $S \equiv \frac{dR_i}{dt} \in \mathrm{Im}D$. The relevant algorithm based on the linear dependence of gradient $S(u, u_1, \ldots, u_k)$ on the highest derivative u_k is given below.

1) $S := \frac{dR_i}{dt}$

2) $K := \mathrm{ord}\, S$

3) if $\frac{\partial^2 S}{\partial u_k^2} \neq 0$ then STOP (checking up the linearity condition)

4) $S := S - D \int \frac{\partial S}{\partial u_k} dU_{k-1}$ (reducting the order of S)

5) if $S \neq 0$ then go to 2). (10)

If S expressions contain arbitrary constants or implicit dependencies on u, u_1, ... then the work of algorithm (10) doesn't stop on step 3 but continues after putting the terms in S nonlinearly depending on u_k equal to zero. The algorithms described above are completely implemented in FORMINT.

3.2. Primary classification

Primary classification of evolution equation (1) is done according to the scheme described in 3.1 save for the form of $F(u,u_1,\dots,u_n)$ is completely or partially unknown. One can get differential equations on F_i expressing several first densities in terms of F according to (8), (9) and putting the terms in S nonlinearly depending on u_k equal to zero on the step 3 of algorithm (10). As one solves these equations he can clear up the form of F. As a result one can obtain the list of concrete equations (1) for which several first conditions (6) are done. The most tedious part of computations of this stage-derivation of the equations on F can be carried out by the computer through the usage of various procedures of FORMINT (see example 2, sec.5). As for solving of these equations it must be done by hands.

Evolution equations, derived from the several first conditions (6) are then checked up using FORMINT code by the next conditions of formal integrability: as a result the part of the equations is thrown away. Further investigation of the equations left includes finding their symmetries and Backlund transformations connecting different evolution equations (see below).

3.3. Symmetries

Research algorithm of the symmetry $H(u,u_1,\dots,u_m)$ of the given order for the equation (1) is based on the following relations[5]

$$\frac{\partial H}{\partial u_i} = a_i, \quad i=2,3,\dots,m. \qquad (11)$$

where a_i are the coefficients of the series L^m. One can find $H(u,u_1,\dots,u_m)$ with the accuracy up to addition of the arbitrary function $h(u,u_1)$ by, expressing a_i through F according to (9), checking the compatibility conditions for the system (11)

$$\frac{\partial a_i}{\partial u_j} - \frac{\partial a_j}{\partial u_i} = 0, \quad i \neq j, \qquad (12)$$

and in case they work by integration of (11). After that one must substitute the result into (5), obtain equations on h and solve them. In this important special case when $F = u_n + f(u,u_1,\dots,u_{n-2})$ [1-4] equations (11) are valid for $i=0,1,2,\dots,m$ and H can be obtained by simple integration of (11).

The algorithm described is completely implemented in FORMINT except solving the equations on $h(u,u_1)$. Note that according to theorem 1 the nontrivial symmetry property of the investigated evolution equation works as a solid argument for its formal integrability as there is no example of the evolution equation with nontrivial but finite Lie-Backlund algebra.

3.4. Backlund transformations

Backlund transformation from the evolution equation

$$v_t = G(v,v_1,\dots,v_n) \qquad (13)$$

to equation (1) is the transformation

$$v = \phi(u,u_1,\dots,u_m), \quad m \geq 1 \qquad (14)$$

for wich the relation

$$\phi_* F = G(\phi, D\phi, \dots, D^n\phi) \qquad (15)$$

works. Derivation algorithm for Backlund transformations is based on the following fact. Let $R_i(u,u_1,\dots,u_k)$, $r_i(v,v_1,\dots,v_\ell)$ be conservation law densities for the equations (1), (13) respectively and let them be in nonlinear dependence on u_k and v_ℓ. (One can easily represent them in such a way by adding suitable gradients). In case the transformation (14) exists the following relations

$$m = k-1, \qquad (16)$$

$$r_i(\phi, D\phi, \dots, D^l\phi) - R_i(u,u_1,\dots,u_k) \in \mathrm{Im}D \qquad (17)$$

take place for all i such that $\ell \geq 1$. One can find the order of transformation (14) (or the absence of the latter) by computing several densities R_i, r_i with the help of FORMINT and obtain equations on ϕ function applying algorithm (10) implemented in the program to the left-hand side of (17), the latter evaluated for the simplest R_i, r_i.

It should be mentioned (see[10]) that the formal integrability of the evolution equations derived from the finite number of conditions (6) is proved by Backlund transformation of these to the known formally integrable equations, e.g., KdV equation or its higher analogues. As for the equations nontransformable to the known before, the strict proof of their formal integrability presents a separate problem.

4. PROGRAM DESCRIPTION

FORMINT program with the user commands together is of the following general structure

```
FORMINT: PROCEDURE OPTIONS (MAIN);
FORMAC - OPTIONS;
OPTSET (EXPND; INT);
<INITIAL ASSIGNMENTS>
<F-PROCEDURE DEFINITIONS>
<user program>
END FORMINT;
```

The program includes thirteen function procedures of the F_PROCEDURE type actual parameters of which must be FORMAC expressions. To call this function procedures one uses the statements of the form

LET(<var> = <procname> (<arg 1> , <arg 2>,...,<arg n>));

Through this command the function procedure <procname> value is assigned to FORMAC variable <var>. A brief description of the FORMINT function procedures is given below. The formal parameters with the same meaning for different procedures will be noted similarly. So, symbol S denotes expression $S(u,u_1,\dots,u_k)$; symbol ORDS denotes nonnegative integer equal to $\text{ord}\,S(u,u_1,\dots,u_k) \equiv K$; F is the right-hand side of the evolution equation and N, M, K are nonnegative integers in all cases. Variables $u, u_1, u_2, \dots$ should be represented in the program as UØ, U1, U2, ...

4.1. Procedure DX(S, ORDS) returns the value of

$$DS \equiv \sum_{i=0}^{\text{ord}\,S} \frac{\partial S}{\partial u_i} u_{i+1},$$

e.g.,

DS(UØ*U2, 2) → UØ*U3 + U1*U2

4.2. Procedure DT(S, ORDS,F,ORDF) returns the value of

$$\frac{dS}{dt} \equiv \sum_{i=0}^{\text{ord}\,S} (D^i F) \frac{\partial S}{\partial u_i},$$

e.g.,

DT(UØ**2,Ø,U3+3*UØ*U1,3) → 2*UØ*U3+6+UØ**2*U1

4.3. Procedure DEPPART(Y,Z) returns the value equal to the sum of the Y expression terms, the latter not depending on the variable Z, e.g.,

DEPPART(Z*A+SIN(Z)+B,Z) → Z*A+SIN(Z)

4.4. Procedure TRUNC(S, ORDS, M) returns the value equal to the sum of the $S(u,u_1,\dots,u_k)$ expression terms, the latter not depending on u_i, $i > M$, e.g.,

TRUNG(U1+U2+U3+U4,4,2) → U1+U2

4.5. Procedure INT(Y,Z) returns the value of $\int Y dZ$.

In case $Y = \sum_i (a_i Z + b_i)^{c_i}$, where a_i, b_i, c_i are the arbitrary numbers or expressions not depending on Z, the explicit value of untegral returns. In other cases the result is unspecified function INTEG.(Y,Z) with the following differential rule defined in the program

$$\frac{\partial}{\partial Z} \text{INTEG.}(Y,Z) \rightarrow Y$$

Example:

INT(EXP(X) + X**A + 1/(X - 1), X) →

INTEG.(EXP(X), X) + X**(A + 1)/(A + 1) + LN(X - 1)

4.6. Procedure INTU(S) returns the value of $\int S du$ where the expression S may contain the unspecified functions G(T).(UØ) and their derivatives. Before the call of INTU it's necessary to assign nonnegative integers to PL/1 variables NF and ORDG initially equal to zero. NF must be equal to the number of G(I) functions (I = 1, 2, ..., NF) and ORDG is the order of the highest derivative.

Example:

LET(S = UØ + DERIV(G(1).(UØ)*G(2).(UØ), UØ));

NF = 2; ORDG = 1;

INTU(S) → G(1).(UØ)*G(2).(UØ) + 1/2*UØ**2

4.7 Procedure SUBST(S) returns the value of expression S in which unspecified functions G(I).(UØ) are replaced by the expressions given before. The number of replacements must be assigned to PL/1 variable NS which is initially equal to zero. The expressions which should be placed instead of G(I) functions (I=1,2,...,NS) should be assigned to FORMAC variables SUB(I), the symbol ¤ going before UØ in the expression.

Example. Let S be the expression from previous example.

NS=1; LET(SUB(1)=SIN(¤ UØ);

SUBST(S) → UØ+SIN(UØ) * $G(2)^{(1)}$(UØ)+G(2).(UØ)*COS(UØ)

4.8 Procedure INTX(S,ORDS,M) returns the value of $D^{-1}\tilde{S}$, where $\tilde{S} = S$ - TRUNC(S,ORDS,M). In the procedure algorithm (10) is implemented. If the structures nonlinearly depending on the highest derivative u_k arise in the third step of the algorithm they are substracted from the integrand and printed as

ZERO = < nonlinear structure >

One can obtain the equations on the arbitrary parameters or functions inside $\tilde{S}$ or contradictionary equalities meaning that

$S \notin ImD$ if one makes the right-hand side expressions equal to zero.

Example:

```
INTX(UØ*U3+A*U1**2,3,Ø)→UØ*U2-1/2*U1**2

ZERO=A*U1**2
```

4.9 Procedure NLPART(S,ORDS,M) returns the value of expression $S = \bar{S} + DQ$, where S has no linear terms depending on u_i, $i > M$ and Q function is unambiguously determined in case of the execution of algorithm (10), which is implemented in the procedure. In a special case, when $S \in ImD$ and $M = 0$, zero value is returned.

Example:

```
NLPART(UØ*U4,4,Ø)→U2**2
```

4.10 Procedure CFCOM(M,N,K) returns the value of the coefficients of D^k in the commutator $[F_*, L^M]$ for the N order evolution equation with an implicit right-hand side $F(u,u_1,\dots,u_N)$. The result is expressed in terms of coefficients a_i of L^M series, partial derivatives F_i and their gradients which are presented in the program as

$D^j a_i \to A(I,J)$, $D^j F_i \to FF(I,J)$.

Example:

```
CFCOM(1,3,1)→-A(-1,Ø)*FF(3,1)-2*A(Ø,1)*FF(2,Ø)

-A(1,2)*FF(2,Ø)-A(1,1)*FF(1,Ø)+FF(1,1)*A(1,Ø)

-3*A(-1,1)*FF(3,Ø)-3*A(Ø,2)*FF(3,Ø)-A(1,3)*FF(3,Ø)
```

4.11 Procedure CFL(F,ORDF,M,K) helps to compute $a_M, a_{M-1}, \dots, a_k$ $(K \le M)$ coefficients of $L^M = \sum_{i=-\infty}^{M} a_i D^i$ $(M \ge 1)$ series using recurrence relations (9) for the evolution equation with right-hand side F. The coefficients a_i computed are assigned to FORMAC variables A(I,Ø). The value of M is returned.

Example: Computation of L^3 series coefficients for KdV equation

$u_t = u_3 + 3uu_1$

```
LET(DUM=CFL(U3+3*UØ*U1,3,3,-1);
PRINT_OUT(A(3,Ø); A(2,Ø); A(1,Ø); A(Ø,Ø); A(-1,Ø);
A(3,Ø)=1
A(2,Ø)=Ø
A(1,Ø)=3*UØ
A(Ø,Ø)=3*U1
A(-1,Ø)=U2+3/2*UØ**2
```


4.12 Procedure CONDS(F,ORDF,M1,M2,SW) helps to compute densities $R_{M1},\dots,R_{M2}$ $(M1 \le M2)$ determined by (7) and to check up the conditions of formal integrability (6). The densities R_i computed are assigned to FORMAC variables RES(I) and put to print. In addition when SW=1 a suitable condition (6) is checked up for each R_i (if SW $\neq$ 1 this process doesn't take place). If any of the conditions checking doesn't work the following relations on F are printed as

ZERO = < expression >

The value of M2 is returned.

4.13 Procedure SYMMTR (F,ORDF,M) returns the value of M order symmetry for the evolution with right-hand side F if such symmetry exists and zero otherwise. The symmetry can be computed explicitly or in terms of unspecified function H.(UØ,U1). In the latter case the equations on H are printed as

ZERO = < expression >

If the evolution equation has no M order symmetry, the message is printed

NO SYMMETRY OF ORDER M

Examples of using procedures CONDS & SYMMTR are given in the next section.

5. HOW TO USE FORMINT, EXAMPLES

5.1. Checking up the Conditions of Formally Integrability for Equation $u_t = u_5 + uu_1$

```
LET(DUM=CONDS(U5+UØ*U1,5,1,9));
RES(1)=Ø
RES(2)=Ø
RES(3)=3/5*UØ
RES(4)=Ø
RES(5)=Ø
RES(6)=Ø
RES(7)=21/50*UØ**2
RES(8)=Ø
RES(9)=9/25*U1**2
ZERO=9/25*U1**3
```

Contradictionary equality $\frac{9}{25} u_1^3 = 0$ resulting from checking up condition (6) for $i = 9$ means that the evolution equation is not formally integrable despite of the hypothesis of the work [3].

5.2. Primary Classification of Evolution Equations $u_t = F(u, u_1, u_2, u_3)$

The program to obtain the ordinary differential equation of F_3 using the condition $\frac{d}{dt} R_{-1} = DQ_{-1}$, where $R_{-1} = (F_3)^{-1/3}$ is given below.

```
LET(R = DERIV(F.(UØ,U1,U2,U3),U3)**(-1/3);

R = DT(R,3,F.(UØ,U1,U2,U3),3);

R = INTX(R,F,3));
```

Result:

ZERO = (−2Ø/27 $F^{(4^2)^3}$.(UØ,U1,U2,U3)/ $F^{(4)}.^{7/3}$ (UØ,U1,U2,U3)

+ 5/6 $F^{(4^2)}$.(UØ,U1,U2,U3) $F^{(4^3)}$.(UØ,U1,U2,U3)

/ $F^{(4)}.^{4/3}$ (UØ,U1,U2,U3) − 1/6 $F^{(4^4)}$.(UØ,U1,U2,U3)

/ $F^{(4)}.^{1/3}$ (UØ,U1,U2,U3)) $U4^3$

We get an ordinary differential equation on F_3 making the above expression equal to zero:

$$9 F_{3333}(F_3)^2 - 45 F_3 F_{33} F_{333} + 40 (F_{33})^3 = 0 .$$

This equation has the following general solution [11]

$$F_3 = (pu_3^2 + qu_3 + r)^{-3/2} ,$$

where p, q, r are the functions of u, u_1, u_2.

5.3. Computing of the 5th Order Symmetry for Calogero-Degasperis-Fokas Equation [1,12]

$$u_t = u_3 - \frac{1}{8} u_1^3 + (ae^u + be^{-u}) u_1$$

```
LET(F = U3 - U1 ** 3/8 + G(1).(UØ)*U1;

SUB(1) = A*#E ** ¤UØ + B*=E **(- ¤UØ));

NF = 1;   NS = 1;

PRINT_OUT(H(5) = SYMMTR(F,3,5));
```

H(5) = U5 + 5/3 A B U1 + 1Ø/3 A U2 U1 # $E^{UØ}$

+ 5/3 A U3 # $E^{UØ}$ + 5/8 A $U1^3$ # $E^{UØ}$ − 1Ø/3 B U2 U1 # $E^{-UØ}$

+ 5/3 B U3 # $E^{-UØ}$ + 5/8 B $U1^3$ # $E^{-UØ}$ − 5/8 U3 $U1^2$

+ 5/6 A^2 U1 # $E^{2UØ}$ + 5/6 B^2 U1 # $E^{-2UØ}$ − 5/8 $U2^2$ U1

+ 3/128 $U1^5$

ACKNOWLEDGEMENT

We are grateful to Profs. N.N.Govorun, E.P.Zhidkov and V.G.Makhankov for useful discussions, interest to this work and support.

TEST RUN OUTPUT

EQUATION DU / DT = F

F = U5 + 1Ø A B U1 # $E^{UØ}$ + 5 A U3 # $E^{2UØ}$ + 15 A U1 U2 # $E^{2UØ}$ + 5 B U3

$E^{-UØ}$ + 5 U3 U2 − 5 U3 $U1^2$ − 5 U1 $U2^2$ + 5 A^2 U1 # $E^{4UØ}$ + 5 B^2 U1

$E^{-2UØ}$ + $U1^5$

COMPUTING CONSERVATION LAW DENSITY RES(5)
CHECKING CORRESPONDING INTEGRABILITY CONDITION

RES(5) = −25 A B $U1^2$ # $E^{UØ}$ + 5 A B^2 − 5/3 A $U1^4$ # $E^{2UØ}$ + 3Ø A $U2^2$

$E^{2UØ}$ − 5/3 B $U1^4$ # $E^{-UØ}$ + 15 B $U2^2$ # $E^{-UØ}$ + 5 A^2 B # $E^{3UØ}$ −

5Ø A^2 $U1^2$ # $E^{4UØ}$ + 5/3 A^3 # $E^{6UØ}$ − 2Ø B^2 $U1^2$ # $E^{-2UØ}$ + 5/3 B^3

$E^{-3UØ}$ − 25 $U1^2$ $U2^2$ − 5 $U3^2$ − 5/3 $U1^6$ + 25/3 $U2^3$

COMPUTING SEVENTH-ORDER SYMMETRY H

H = U7 + 28 A B U3 # $E^{UØ}$ + 7Ø A B U1 U2 # $E^{UØ}$ + 14 A B $U1^3$ # $E^{Ø}$ + 7Ø A

U3 U3 # $E^{2UØ}$ + 56 A U3 $U1^2$ # $E^{2UØ}$ + 35 A U4 U1 # $E^{2UØ}$ + 7 A U5 # E^{2}

UØ + 112 A U1 $U2^2$ # $E^{2UØ}$ + 28 A B^2 U1 + 42 A $U1^3$ U2 # $E^{2UØ}$ + 7 B

U3 U2 # $E^{-UØ}$ − 7 B U3 $U1^2$ # $E^{-UØ}$ − 7 B U4 U1 # $E^{-UØ}$ + 7 B U5 # E

$$^{-U\O} - 14\,B\,U1\,U2^{2}\ \#\ E^{-U\O} + 14\,U3\,U4 - 14\,U3\,U1^{2}\,U2 + 14\,U3\,U1^{4}$$

$$- 28\,U3\,U2^{2} - 28\,U4\,U1\,U2 + 7\,U5\,U2 - 7\,U5\,U1^{2} - 28/3\,U1\,U2^{3} + 28\,A^{2}$$

$$B\,U1\ \#\ E^{3U\O} + 14\,A^{2}\,U3\ \#\ E^{4U\O} + 98\,A^{2}\,U1\,U2\ \#\ E^{4U\O} + 7\O\,A^{2}\,U1^{3}$$

$$\#\ E^{4U\O} + 28/3\,A^{3}\,U1\ \#\ E^{6U\O} + 14\,B^{2}\,U3\,E^{-2U\O} - 28\,B^{2}\,U1\,U2\ \#\ E$$

$$^{-2U\O} + 7\,B^{2}\,U1^{3}\ \#\ E^{-2U\O} + 28/3\,B^{3}\,U1\ \#\ E^{-3U\O} - 21\,U3^{3}\,U1$$

$$+ 28\,U1^{3}\,U2^{2} - 4/3\,U1^{7}$$

REFERENCES

1. Fokas A.S. J.Math.Phys., 1980, 21, p.1318.
2. Fujimoto A., Watanabe Y. Math.Jap., 1983, 28, p. 43.
3. Abellanas L.A., Galindo A. J.Math.Phys., 1983, 24, p. 504.
4. Leo M. et al. J.Math.Phys., 1983, 24, p.1720.
5. Ibragimov N.H., Shabat A.B. Funct.Anal., 1980, 14, p. 79. (in Russian).
6. Lax P.D. Comm.Pure and Appl.Math., 1968, 21, p. 467.
7. Svinolupov S.I., Sokolov V.V. Funct.Anal., 1982, 16, p. 86 (in Russian).
8. Bahr K.A. FORMAC 73 User's Manual, Darmstadt: GMD/IFV, 1973.
9. Svonilupov S.I., Sokolov V.V., Yamilov R.I. DAN SSSR, 1983, 271, p. 21 (in Russian).
10. Ibragimov N.H., Shabat A.B. Funct.Anal., 1980, 14, p.25.
11. Kamke E. Differentialgleichungen Losungsmethoden und Losungen, I. Gewöhnliche Differentialgleichungen, Akad.Verl. Ges. Geest & Portig K.G., Leipzig, 7th edition.
12. Calogero F., Degasperis A. J.Math.Phys., 1981, 22, p. 23.

Received by Publishing Department
on June 11, 1984.

Гердт В.П., Швачка А.Б., Жарков А.Ю.

E11-84-400

FORMINT - программа для классификации интегрируемых нелинейных эволюционных уравнений

Программа написана на языке аналитических вычислений PL/1-FORMAC и предназначена для классификации формально интегрируемых нелинейных эволюционных уравнений. Для заданного уравнения программа позволяет проверять необходимые условия формальной интегрируемости, получать соотношения на правую часть уравнения, эквивалентные этим условиям, вычислять симметрии и плотности законов сохранения.

Работа выполнена в Лаборатории вычислительной техники и автоматизации ОИЯИ.

Препринт Объединенного института ядерных исследований. Дубна 1984

Gerdt V.P., Shvachka A.B., Zharkov A.Yu.

E11-84-400

FORMINT - a Program for Classification of Integrable Nonlinear Evolution Equation

The program is created in the language of computer algebra system PL/1 - FORMAC intended for classification of formally integrable nonlinear evolution equations. For a given equation the program allows to check up necessary conditions of formal integrability, to derive relations on the right-hand side part of equation equivalent to these conditions, to compute symmetries and conservation laws densities.

The investigation has been performed at the Laboratory of Computing Techniques and Automation, JINR.

Preprint of the Joint Institute for Nuclear Research. Dubna 1984