

Research reports in Physics -Nonlinear Dynamics-
Springer Verlag Berlin-Heidelberg-New York
S. Carillo and O.Ragnisco, eds.
pages 131-135, 1990

Algorithms to detect complete integrability in 1+1 - dimension

Waldemar Wiwianka , Benno Fuchssteiner
Department of Mathematics, University of Paderborn, 479 Paderborn,
W-Germany

Abstract.

Algorithms to test and detect complete integrability for nonlinear partial differential equations are given, their implementations in MAPLE are described. For some examples of solvable equations these algorithms are used to find symmetries and master symmetries, yielding the recursive structure of these systems.

1 Introduction

For a dynamical system for $u = u(x, t)$ given by $u_t = K(u)$ we want to find out by use of computer : if the system is completely integrable, and, provided the answer is yes, we want to find the recursion formula for generators of one-parameter symmetry groups.

2 Review of the Problem

The search for the infinitesimal generators of all one-parameter symmetry groups requires the determination of $K^\perp = \{G \mid [G, K] = 0\}$, the commutant of K with respect to the vector field Lie algebra.

It is well known that a hamiltonian system on a $2N$ -dimensional manifold is completely integrable if and only if K^\perp is an N -dimensional abelian Lie algebra of hamiltonian vector fields. For infinite dimensional manifolds the notion of **complete integrability** is not yet completely clear, usually one requires that K^\perp is sufficiently large and abelian. Thus a test for complete integrability is reduced to the construction of sufficiently large abelian subalgebras of K^\perp .

There are many standard mechanisms for the recursive generation of this commutant. For our computational approach we have chosen the mastersymmetries¹ as basic tools.

This for two reasons : Firstly, it seems to be the most general approach. It works in a simple and transparent way even when recursion operators do not exist. Secondly, from the computational point of view, mastersymmetries are simple to handle and give rise to elementary and efficient algorithms.

Before we start with the crucial ansatz for our mastersymmetries we need some basic notation and we have to make some assumptions:

1. For simplicity we restrict our attention to one - component systems of integro - differential equations of polynomial type in one independent variable. To be precise : We consider only $K(u)$ out of the algebra $\mathcal{A}(D, D^{-1}, u)$ generated by D (differentiation with respect to the independent variable x), D^{-1} (inverse of D) and u (field variable). We remark, that under this assumption $K(u)$ is translation invariant, i.e. commutes with u_x .
2. In order to carry out our computations we introduce the algebra $\mathcal{A}(x, D, D^{-1}, u)$ generated by $\mathcal{A}(D, D^{-1}, u)$ and by multiplication with the independent variable x . Members of this Lie algebra are called **vector fields**.
3. We assume that the systems, for which our algorithm has to find the recursive form of the symmetry group, is completely integrable in the following sense:
There is a hereditary algebra $\mathcal{H}er(\tau_n, K_n \mid n \in \mathbb{N})$ of vector fields with $K_0 = u_x$ containing the given $K(u)$. By **hereditary algebra** we mean that the following commutation relations have to be fulfilled for arbitrary n, m :

$$i) \quad [K_n, K_m] = 0 \quad (1)$$

$$ii) \quad [\tau_n, K_m] = (m + \rho)K_{m+n} \quad (2)$$

¹Mastersymmetries exist for all known completely integrable systems, which is no surprise since they are closely connected to the existence of angle variables, at least for finite dimensional invariant submanifolds. (See the articles of B. Fuchssteiner and G. Oevel - B. Fuchssteiner in this volume)

$$iii) \quad [\tau_n, \tau_m] = (m - n)\tau_{n+m} . \quad (3)$$

where ρ is some suitable number. The K_n are the symmetries of the system and the τ_n are called **mastersymmetries**.

Assumption 2 implies in particular that all K_n are elements of $\mathcal{A}(D, D^{-1}, u)$ since they all commute with u_x . We like to remark, that such a hereditary algebra exists for all known **completely integrable** hamiltonian flows on infinite dimensional manifolds. It even exists for nonhamiltonian systems, for example Burgers equation.

Our algorithm is based on the crucial **Assumption** : All mastersymmetries are of the form

$$M = x\tilde{K} + Z$$

where \tilde{K} is a symmetry and Z is some unknown translation invariant vector field.

At this point we would like to mention some known results related to the algorithm presented here. In Paderborn, our group together with some students has developed and implemented similar algorithms in case of lattice - systems and for quantum mechanical spin - $\frac{1}{2}$ - chains (see [?] , [?] , [?] , [?]). Although these algorithms technically differ drastically from the present one, they look very similar on the level of the user interface. The only apparent difference is that in case of a one dimensional lattice, for example, the continuous parameter x has to be replaced by the discrete n describing the lattice position.

3 The special ansatz $M = x\tilde{K} + Z$

Summing up , it may be said that the expected mastersymmetry always has a term breaking translation invariance and having a coefficient out of the the symmetry group generators plus some unknown term which has to be translation invariant.

By study of the action of the translation group it can be proved that the mastersymmetries have to have the special form which we assumed. Let us go successively through the necessary arguments.

1. As the angle variables are scalar fields, linear in t , they give rise to time dependent symmetry group generators (also linear in t) on the vector field side. So we have to look at two crucial classes of vectorfields. The first are the time independent symmetry group generators $K_0, K_1, K_2, \dots, K_n$ and the second are the time dependent $G_0, G_1, G_2, \dots, G_n$ coming from the angle variables and being linear in t . Obviously we will find the generator of the translation invariance among the K 's. We set $K_0 = u_x$.
2. As the K_i belong to the action variables and since these generate an abelian symmetry group it must hold $[K_i, u_x] = 0$ for all i . This is equivalent to the fact, that the partial derivative of K_i with respect to x , vanishes $\frac{\partial}{\partial x} K_i = 0$. This means the K_i cannot depend explicitly on x .
3. Because of the linearity in t the Poisson bracket between the angle variables and the action variables lead to non vanishing action variables. By use of the Lie algebra homomorphism, which is given from the hamiltonian formulation, we carry over these facts to the vector fields. So every commutator between some τ_i and some K_j has to lead to a non vanishing linear combination of the K_n . So, for the special $K_0 = u_x$ we have $[\tau_i, u_x] = \sum_n \alpha_n K_n$. If we choose an suitable base of the τ_i we can expect that

$$[\tau_i, u_x] = \rho K_i \tag{4}$$

which yields the ansatz we made.

4. This last formula we can write as $\frac{\partial}{\partial x} \tau_i = K_i$, which shows that τ_i depends explicitly on x and that this dependence is of first order. With that we get $\tau_i = xA_i + Z_i$ where A_i and Z_i are translation invariant. From the comparison with $\frac{\partial}{\partial x} \tau_i = K_i$ we conclude $A_i = K_i$ and so we obtain $\tau_i = xK_i + Z_i$. with which it is also shown that the explicit time dependence has to be in Z_i .
5. Now using the fact that the coefficient of the linear term in t has to be a symmetry, i.e. commutes with all K_i , we can drop this term without changing the result of the commutator (4). The t -independent term obtained by this is our required mastersymmetry.

In case we have several independent variables x_1, \dots, x_n the role of u_x will be taken by the generators $u_{x_1}, u_{x_2}, \dots, u_{x_r}$. This results in $\tau_i = \sum_n x_n \tilde{K}_n + Z_n$, where \tilde{K}_n is a linear combination of the K_n .

4 The design of the algorithms

For the algorithmic search of symmetries the **highest-order-term projection** is an essential tool². For the definition of that projection we introduce an order \prec among the elements of $\mathcal{A} = \mathcal{A}(x, D, D^{-1}, u)$ having the following properties.

1. For every fixed $A \in \mathcal{A}$ the space $\{B \mid B \prec A\}$ is finite-dimensional.
2. The order is semi-compatible with the Lie algebra structure in \mathcal{A} , that means whenever $[A, X] = B$ then there is some $\tilde{X} \prec B$ such that $X - \tilde{X} \in A^\perp$.
3. \prec is a linear order on the set \mathcal{M} in \mathcal{A} where \mathcal{M} is the set generated by x, D, D^{-1} and u due algebraic operation given by multiplication.

We will not describe the details of this order here, since these are technical and implemented by a set of nested programs. The basic principle is that differentiation increases the order. However, since integration is allowed implementation of this order is not trivial from the syntactic point of view.

Once, such an order found, the solution X of the division problem

$$[A, X] = B \tag{5}$$

for given A and B can be achieved algorithmically by means of linear algebra (whenever such a solution exists, of course). One has to work with a suitable base of the linear space $\{B \mid B \prec A\}$. However, such an algorithm based on solving linear equations is too slow to yield an efficient tool. Therefore we have introduced the **highest-order-projection** $HO(A)$ mapping every sum $A_1 + A_2 + \dots + A_n$ onto its maximal summand A_k (i.e. $A_j \prec A_k$ for all $j \neq k$). Instead of solving (5) we solve first the following approximate division problem:

$$HO([HO(A), HO(X)]) = HO(B) \tag{6}$$

²It is sufficient to think in terms with the highest x - derivatives.

The sub-program that does the job is called $CS(B, A)$ and is the heart of the whole matter. The efficient implementation of this routine is the most difficult part of our implementation, since a great number of special cases has to be considered. Once we have this routine it is easy to find symmetries and mastersymmetries. We start with a symmetry \tilde{K} for the vector field K , i.e. $[\tilde{K}, K] = 0$ where \tilde{K} and K are both assumed to be translation invariant. Our mastersymmetry M is of the form $M = x\tilde{K} + Z$ with translation invariant Z .

Since M is a mastersymmetry the commutation with K must be another symmetric

$K_1 = [M, K] = SYM(K, \tilde{K})$ which we denote by $SYM(K, \tilde{K})$, since finding this later on has to be implemented as a sub routine of our program package.

An essential property of the order relation is that it is defined in such a way that the highest-order-term of $[K, M]$ is of the same order as that of $[K, x\tilde{K}]$. Therefore the knowledge of the known term \tilde{K} yields the highest-order-term of the commutator $[M, K]$ (up to multiplication by a suitable factor). Thus the highest-order-term of the unknown mastersymmetry M is independent of the unknown quantity Z . So for given \tilde{K} we find $SYM(K, \tilde{K})$ by **Algorithm 1**. It looks as if the algorithm can only be successful if a symmetry \tilde{K} is already known. That is not so, because at the beginning of our computation we can always choose $\tilde{K} = K$ and then compute the symmetries recursively.

We observe that the implementation of SYM is not really necessary since it is only a special case of **Algorithm 2** (which will be needed anyway).

Algorithm 1 - (SYM) :

Input : $SYM(K, \tilde{K})$

Step 0 : $K_1 := [x\tilde{K}, K]$

Step 1 : $P := [K_1, K]$

if $P \neq 0$

then Goto Step 2

else Output $\rightarrow SYM(K, \tilde{K}) = K_1$

Step 2 : $X := CS(P, K)$

if there is no solution

then Output $\rightarrow no\ solution$

else Goto Step 3

Step 3 : $K_1 := K_1 - X$

Goto Step 1

Algorithm 2 - (GHO) :

Input : $GHO(E, K)$ or $GHO(E, K, X)$

Step 0 : if number of arguments = 3
then $S := X$

else $S := CS(HO(E), HO(K))$

Step 1 : $P := [S, E]$

if $P \neq K$

then Goto Step 2

else Output $\rightarrow GHO(E, K) = S$

Step 2 : $X := CS(HO(E), HO(P - K))$

if there is no solution

then Output $\rightarrow no\ solution$

else Goto Step 3

Step 3 : $S := S - X$

Goto Step 1

$CS()$ denotes the crucial algorithm which produces the solution of our approximate division problem. Clearly this algorithm ($GHO =$ given highest order) produces, if existent, for given E, K and X the solution Y of $[E, Y] = K$ such that $HO(Y) = HO(X)$.

Now $SYM(K, \tilde{K}) = GHO(0, K, [x\tilde{K}, K])$. And the mastersymmetry we are looking for is obtained by the call $GHO(K, SYM(K, \tilde{K}))$, or, at the beginning, when a second symmetry \tilde{K} is unknown by $GHO(K, SYM(K, K))$.

The programs are implemented in MAPLE, a formula manipulation system developed by the University of Waterloo.

References