Computer algebra algorithms and routines for the computation of conservation laws and fixing of gauge in differential expressions

Thomas Wolf
Queen Mary & Westfield College, University of London,
Mile End Road, London E1 4NS, UK
email: T.Wolf@maths.qmw.ac.uk

Andreas Brand
Institute for Informatik, Friedrich Schiller Universität Jena
email: maa@cnve.rz.uni-jena.de

Majid Mohammadzadeh
Faculty of Mathematics and Computer Engineering
University of Teacher Education
49 Mofateh Ave, PB 15614, Tehran
email: majid@saba.tmu.ac.ir

May 3, 1999

Abstract

Three different approaches for the determination of conservation laws of differential equations are presented. For three corresponding `REDUCE` computer algebra programs `CONLAW1/2/3` the necessary subroutines are described. One of them simplifies general solutions of overdetermined PDE systems so that all remaining free functions and constants correspond to independent conservation laws. It determines redundant functions and constants in differential expressions and is equally useful for the determination of symmetries or the fixing of gauge freedom in differential expressions.

1 Introduction

The determination of conservation laws (CLs) for single or systems of partial differential equations (PDEs) and of first integrals for ordinary differential equations (ODEs) is of interest for the exact solution of these DEs, for their understanding, classification and for supporting their numerical solution. In this paper we outline three computer algebra programs for the computation of CLs and explain in more detail subroutines to fix gauge freedom in differential expressions which in this context is used to extract individual CLs from the general solution of CL-determining equations.

In the following we adopt the notation of the book of Olver [5]. Independent variables will be denoted by $x = (x^1, x^2, \ldots, x^p)$. The differential equations are $\Delta(x, u^{(n)}) = 0$ (i.e. $\Delta_1 = 0, \ldots, \Delta_q = 0$), for q functions $u = (u^1, u^2, \ldots, u^q)$, $u^{(n)}$ denoting u-derivatives of order up to n. The conservation law that is to be fulfilled by solutions of $\Delta = 0$ is $\mathrm{Div}\, P = 0$ with conserved current $P = (P^1, \ldots, P^p)$. We will use $_J$ as a multiple index denoting partial derivatives, for example, u^α_J will stand for an arbitrary partial derivative, like $\partial^k u^\alpha/(\partial x^1 \partial x^2 \ldots)$.

If the differential equations $\Delta = 0$ result from a variational principle then any Lie-symmetry of $\Delta = 0$ provides a conservation law as is known from Noether's Theorem. Instead, we will not make any restrictive assumptions which leaves us to solve $\mathrm{Div}\, P = 0$ either directly or to determine characteristic functions of conservation laws or to do both at once. A comparison of these different approaches with respect to their complexity, and an extension to find non-local conservation laws and applications to PDEs with parameters will be described elsewhere [9]; here we concentrate on the computer algebra aspects.

2 The mathematical problem and the three approaches

In this section we describe three ways to formulate determining conditions for conservation laws.

The first and most direct approach is to solve

$$\mathrm{Div}\, P = 0 \tag{1}$$

modulo $\Delta = 0$ directly. The corresponding program is `CONLAW1`. The components of the conserved current $P^1, \ldots, P^p$ that are to be calculated are functions of all independent variables x^i, the dependent variables u^α and their derivatives u^α_J up to some order.

Because we are not interested in trivial CLs $P = \texttt{curl}\ V$ but in CLs that solutions of $\Delta = 0$ obey, we use $\Delta = 0$ to eliminate some of the so-called jet-variables u^α_J and substitute them in the determining conditions (1). By that, the conditions (1) have to be fulfilled identically in less variables, they become less restrictive and they may have additional solutions apart from $P = \texttt{curl}\ V$. These extra non-trivial CLs are the ones of interest. We therefore assume $\Delta = 0$ can be solved for leading derivatives u^α_J so that they and all their partial derivatives that occur in (1) can be substituted. We also, w.l.o.g., assume that the P^i do not depend on u-derivatives we substitute, which fixes a kind of equivalence of CLs.

Other approaches calculate characteristic functions Q^ν. A theorem can be proven ([5], p. 272) that for a totally non-degenerate system $\Delta_\nu = 0$, each equivalence class of CLs $\mathrm{Div}\, P = 0$ (i.e. conserved currents differing only by a curl) is determined uniquely by characteristic functions Q^ν satisfying

$$\mathrm{Div}\, P = \sum_\nu Q^\nu \Delta_\nu \tag{2}$$

identically in *all* $x^i, u^\alpha, u^\alpha_J$. Equ. (2) is not solved by simply eliminating Q^1 in terms of P and Δ and other Q^ν as it would be singular for solutions of $\Delta = 0$. To avoid this and because the Q^ν are unique only modulo $\Delta = 0$, we w.l.o.g. ignore dependencies of Q^ν on leading u-derivatives in $\Delta = 0$ and any of their derivatives. A way to calculate the Q^ν is to use the property of the Euler operators $E_\nu = \sum_J (-D)_J \partial/\partial(u^\nu_J)$ which acting on an expression gives identically zero iff this expression is a divergence. The D are total derivatives. Applying this operator on (2) and using $\Delta_\nu = 0$ one obtains as determining conditions for the Q^ν:

$$0 = \sum_{\mu,J} (-D)_J \left(Q^\mu \frac{\partial \Delta_\mu}{\partial(u^\nu_J)} \right) \quad \forall \nu. \tag{3}$$

The second and third approach are to solve identically in $x^i, u_\alpha, u^\alpha_J$ either (3) for Q^ν or (2) for P^i, Q^ν. The corresponding programs are `CONLAW2` for (3) and `CONLAW3` for (2).

The three approaches (1)-(3) differ in the number of equations to be solved or their order or the number of functions to be determined or the number of independent jet-variables or the degree of an ansatz for P, Q in order to obtain the same conservation law.

To obtain solutions of (1)-(3) we assume bounds on the order of u-derivatives on which the P^i and Q^ν may depend. For (1) we assume a bound for P^1 and for (2),(3) we assume a bound for Q^ν. Bounds for the remaining unknown functions follow. Differentiations done in all three conditions (1)-(3) introduce jet-variables (u-derivatives) on which the P^i resp. Q^ν do not depend so that overdetermined conditions result in which there is no unknown function P^i, Q^ν of all jet-variables u^α_J in which the conditions have to be satisfied identically. The resulting overdetermined PDE-systems are investigated with the computer algebra package `CRACK`.

3 The computer algebra problem

The main computer algebra problem is to solve the overdetermined conditions (1)-(3). Steps undertaken include the separation, integration, application of integrability conditions (differential Gröbner Basis), solution of ODEs and other steps which are described in [7],[8].

If the overdetermined system is linear ((1)-(3) are linear in P^i, Q^ν) and not too big - we give an example below for what is currently possible - then `CRACK` will solve the system either completely or partially and return unsolved equations e.g. return the heat equation when investigating conservation laws of the Burgers equation.

In the general solution of the CL-condition(s) a CL is extracted by collecting all terms involving one of the arbitrary constants or arbitrary functions in the solution. If some of them were redundant then CLs extracted would not be independent of each other.

Redundant constants and functions may result because in the process of solving the overdetermined system there is no general rule for what should have a higher priority, integrations or the application of integrability conditions, as there are examples requiring a higher priority for each of them. It therefore may happen that two equations are integrated which are not independent of each other and therefore the constants or functions of integration are not independent of each other. As a consequence the final general solution could have redundant arbitrary constants and functions. For example, in the expression $c_1(x)t + c_2xt + c_3$ with independent variables x, t and arbitrary function $c_1(x)$ and arbitrary constants c_2, c_3 the constant c_2 is redundant as it can be absorbed by $c_1(x)$ through $c_1(x) \to c_1(x) - c_2x$.

Recognizing redundancy can become cumbersome in the case of many independent variables or if arbitrary constants/functions appear non-linearly.

Another application of redundancy recognition is the solution of PDE systems with some gauge freedom where the problem is to eliminate any gauge freedom from the general solution of this system. This can be accomplished by including in the solution terms representing the complete gauge freedom. For example, in the case of conditions (1) the general solution could be augmented by `curl` V and V be added to the list of free constants and functions. In this way trivial CLs could be filtered out as the free constants and functions corresponding to them would be redundant to V.

Although in the case of computing CLs, one easily could drop trivial CLs after they have been computed by checking $\mathrm{Div}\, P = 0$ identically in all jet-variables, such a simple test to eliminate gauge might not be available for other problems.

4 Subroutines

In the following subsections we describe subroutines which extract CLs from the general solution of conditions (1)-(3), subroutines that compute Q^ν from P^i and P^i from Q^ν and subroutines that simplify P^i.

4.1 Identifying redundant constants and functions

The problem of finding the general solution of a PDE system with some existing gauge freedom fixed can be reduced to the problem of finding the general solution of a PDE system without fixing gauge in the following way.

Given a system of DEs $0 = \Omega(f_a, x^i)$ to be solved for the functions $f_a(x^i)$, we assume

$$f_b = F_b(x^i, g_c) \qquad (4)$$

to be a general solution where F_b are differential expressions in x^i, g_c where g_c are arbitrary constants and functions. They may include functions from the original set f_a and constants and functions of integration.

The question is to specify the g_c to fix any redundancy but not to lose generality of the solution. The steps are:

- Formulate a set of conditions

$$0 = F_b(x^i, g_c) - F_b(x^i, \bar{g}_c) \qquad (5)$$

where $\bar{g}_c$ are new functions, each $\bar{g}_c$ having the same variable dependence as g_c. Regard equ. (5) as a system of equations for the set of unknown functions $\{g_c, \bar{g}_c\}$, to be satisfied identically in the x^i.

- Find the general solution of the system (5) as
$$\tilde{g}_c = G_c(x^i, h_d) \tag{6}$$
where $\tilde{g}_c$ is a subset of $\{g_a, \bar{g}_b\}$, and G_c are algebraic or differential expressions in functions h_d which are the remaining $\{g_a, \bar{g}_b\}$ and extra constants and functions of integration. The h_d are arbitrary. Any function g_a or $\bar{g}_a$ appears only once on a left-hand-side (lhs) of (6) or only on right-hand-sides (rhs's).
- If for any index c both, g_c and $\bar{g}_c$ appear only on rhs's of (6) then g_c is redundant and can be set to zero in all F_b in (4) and all G_c in (6).
- If for any index c both, g_c and $\bar{g}_c$ appear only on lhs's of (6) in the equations $g_c = G_c$ and $\bar{g}_c = \bar{G}_c$ then these two equations are replaced by $\bar{g}_c = g_c - G_c + \bar{G}_c$ in (6).
- If for any index c, g_c appears on a lhs of (6) and $\bar{g}_c$ appears only on rhs's then the equation with lhs g_c is solved for $\bar{g}_c$ in terms of g_c and other functions and replaced by the new equation $\bar{g}_c = \bar{G}_c(g_c, \ldots)$. With this new equation $\bar{g}_c$ is substituted on any rhs of (6).
- There remains only the case of $\bar{g}_c$ being on the lhs of an equation and g_c being on rhs's such that the system (6) now has the form
$$\bar{g}_c = \bar{G}_c(x^i, g_a, \bar{h}_b) \tag{7}$$
where $\bar{h}_b$ are arbitrary constants and functions of integration which arose during the solution of (5). $\bar{g}_c$ do not occur on rhs's as they would be redundant and would have been set to zero otherwise.
- Finally, free constants and functions $\bar{h}_b$ on rhs's will be chosen to make as many $\bar{G}_c$ as possible zero and to set the redundant g_c to zero in (4) and (7). As we do not have to know $\bar{h}_b$ explicitly, it is enough to find equations in (7) which include an arbitrary function $\bar{h}_b$ of all variables x^i in this equation. Assuming local solvability of $0 = \bar{G}_c$ for $\bar{h}_b$ we conclude redundancy of g_c.
- All remaining $\bar{h}_b$ which cannot be used to make a rhs zero are set to zero themselves and the final form of (7) $\bar{g}_c = \bar{G}_c(x^i, g_a)$ provides substitutions which turn $F_b(x^i, \bar{g}_c)$ into the gauge fixed final solution $f_b = F_b(x^i, g_c)$.

Two comments:

Although the solvability of (5) for $g_a, \bar{g}_b$ and the solvability of $0 = g_c - G_c(x^i, \bar{g}_c, \ldots)$ for $\bar{g}_c$ cannot be guaranteed, this should in practice not be a problem for the following reasons.

- Usually there is no arbitrary function $g_a, \bar{g}_b$ depending on all (jet-) variables of (5) such that (5) is very overdetermined and therefore easy to solve.

- If the equ.s $0 = \Omega$ are linear in f_a then their solution is linear in the arbitrary functions g_c which is the case for the computation of CLs [1].
- If equ.s $0 = \Omega$ are non-linear in f_a then solving (5) should still be simpler than the solution of $0 = \Omega$ which we assume was possible to derive.
- Equ.s (5) have the special solution $\bar{g}_c = g_c, \ \forall c$.

The above steps for fixing gauge freedom are not only applicable once a general solution of a PDE(-system) $0 = \Omega(f_a, x^i)$ has already been found. For example, the computation of conservation laws for the Burgers equation below returns the heat equation which remains unsolved. In order to find redundancies in constants and functions which turn up in a preliminary solution $f_b = F_b(x^i, g_c)$ and which additionally have to satisfy remaining differential equations $0 = D(x^i, g_c)$, one can extend redundancy conditions (5) by $0 = D(x^i, g_c) - D(x^i, \bar{g}_c)$. These conditions are sufficient but not necessary as only equivalence of $0 = D(x^i, g_c)$ and $0 = D(x^i, \bar{g}_c)$ is required, not equality.

The possibility to fix at least some gauge freedom even in the presence of yet unsolved equations opens the possibility to run a gauge-fixing step during the process of solving overdetermined PDE-systems. By that the number of unknown functions could be reduced and the remaining equations be simplified.

4.2 Computing characteristic functions from conserved currents

The first approach (1) is attractive compared with (2),(3) as it generates only one PDE to be solved which is of first order and involves less jet-variables than approach (2) because it is computed modulo $\Delta = 0$. Also, it has less functions to compute than approach (2). A negative aspect is that it provides only the conserved current P and not the characteristic functions Q.

If expressions $Q^{\nu J}$ in a relation (8) below are known then partial integrations (9) yield the characteristic functions Q^ν and the corresponding conserved current $P - R$:

$$\begin{aligned}
\operatorname{Div} P = 0 \quad & \mod \ \Delta_\nu = 0 \ \leftrightarrow \\
\exists\, Q^{\nu J} : \operatorname{Div} P \ &= \ \sum_{\nu,J} Q^{\nu J} \Delta_\nu^{(J)} && \text{(identically in } \textit{all} \ x, u_J^\alpha) \qquad (8) \\
&= \ \sum_{\nu,J} D_J(Q^{\nu J}\Delta_\nu) - D_J(Q^{\nu J})\Delta_\nu && \text{(repeatedly)} \qquad (9) \\
&= \ \operatorname{Div} R + \sum_\nu Q^\nu \Delta_\nu
\end{aligned}$$

Equation (8) cannot be regarded as a linear algebraic equation to determine $Q^{\nu J}$ as there is the additional requirement that the $Q^{\nu J}$ are non-singular for solutions of $\Delta = 0$. Instead, $\operatorname{Div} P$ is calculated and substitutions of a different form than before are made. For example, if CLs for the Harry Dym equation $0 = \Delta = u_t - u^3 u_{xxx}$ are investigated and if for the derivation of (1) there had been done substitutions $u_t = u^3 u_{xxx}, \ \ u_{tx} = (u^3 u_{xxx})_x, \ldots$ before then now the substitutions would be $u_t = \Delta + u^3 u_{xxx}, \ u_{tx} = \Delta_x + (u^3 u_{xxx})_x, \ldots$ which provide the rhs of (8). The computation of Q^ν and $P^i - R^i$ from P^i is part of `CONLAW1`.

[1] Conditions become non-linear if we want to calculate parameter values such that CLs exist.

4.3 Computing conserved currents from characteristic functions

The inverse computation is necessary in `CONLAW2` where the conserved current P^i has to be computed from Q^μ by integrating $\mathrm{Div}\, P = \sum_\nu Q^\nu \Delta_\nu$.

A direct way is based on a formula given by Anco & Bluman in [1]:

$$\begin{aligned}
P^i &= \int_0^1 \frac{d\lambda}{\lambda}\left(S^i(u) + N^i_\mu(u)u^\mu + N^{ij}_\mu(u)D_j u^\mu + \ldots\right)|_{u\to\lambda u} && (10)\\
S^i(u) &= Q^\nu \frac{\partial\Delta_\nu}{\partial u^\mu_i}u^\mu + Q^\nu\frac{\partial\Delta_\nu}{\partial u^\mu_{ij}}u^\mu_j - \left(Q^\nu\frac{\partial\Delta_\nu}{\partial u^\mu_{ij}}\right)_j u^\mu + \ldots && (11)\\
N^i_\mu(u) &= \frac{\partial Q^\nu}{\partial u^\mu_i}\Delta_\nu - \left(\frac{\partial Q^\nu}{\partial u^\mu_{ij}}\Delta_\nu\right)_j + \left(\frac{\partial Q^\nu}{\partial u^\mu_{ijk}}\Delta_\nu\right)_{jk} - \ldots && (12)\\
N^{ij}_\mu(u) &= \frac{\partial Q^\nu}{\partial u^\mu_{ij}}\Delta_\nu - \left(\frac{\partial Q^\nu}{\partial u^\mu_{ijk}}\Delta_\nu\right)_k + \left(\frac{\partial Q^\nu}{\partial u^\mu_{ijkl}}\Delta_\nu\right)_{kl} - \ldots && (13)
\end{aligned}$$

where summation is done over double indices.

A slightly more compact formulation (and way to compute P^i) is

$$\begin{aligned}
V &= Q^\nu\Delta_\nu,\\
W^i &= n(i)u^\mu\frac{\partial V}{\partial u^\mu_i} + \\
&\quad n(ij)\left(u^\mu_j - u^\mu D_j\right)\frac{\partial V}{\partial u^\mu_{ij}} + \\
&\quad n(ijk)\left(u^\mu_{jk} - u^\mu_j D_k + u^\mu D_j D_k\right)\frac{\partial V}{\partial u^\mu_{ijk}} + \\
&\quad \vdots\\
T^i &= x^i\int_0^1 d\lambda\lambda^{p-1}V|_{u\to 0, x\to\lambda x}\\
P^i &= T^i + \int_0^1\frac{d\lambda}{\lambda}W^i|_{u\to\lambda u} && (14)
\end{aligned}$$

where in W^i it is summed over equal indices (not the $i, j, k, \ldots$ in n) and $n(i, j, \ldots) = \prod_a r_a!/(\sum_b r_b)!$ with r_a being the multiplicities of different arguments $i, j, \ldots$ of n (e.g. $n(i) = 1, n(i, i) = 1, n(1, 2) = 1/2$) which also occur in (11) - (13). p is the number of variables x^i and T^i are non-zero only if $u \equiv 0$ does not solve $\Delta = 0$ (T^i have to enter (10) in that case as well).

Although being an elegant formula there may be problems in computing the integral analytically. More seriously, the integral may be singular for $\lambda = 0, 1$. That is the case, for example, for the non-polynomial characteristic functions of the Harry-Dym equations in the next section. Although in some cases it might help do take $P^i = \int^1 \frac{d\lambda}{\lambda}W^i|_{u\to\lambda u}$ this need not always be the case.

Because of these potential difficulties the default procedure to compute P^i is to use the integration module of `CRACK` to x^1-integrate $\sum_\nu Q^\nu\Delta_\nu$, to x^2-integrate the remaining unintegrated terms and so on. In case, terms remain after the last x^p-integration, the process is restarted on the remaining terms until all terms are integrated or at most a fixed number of times. If this method does not work because not all determining conditions had been solved as, for example, for the Burgers equation below then (14) is used.

4.4 The simplification of P in two variables

After deleting trivial CLs and identifying equivalent CLs through the computation of characteristic functions Q it remains to simplify the conserved current P through the addition of some curl: $P \rightarrow P + \mathrm{curl} V$. This is done if there are only two independent variables, say x^1, x^2. The aim is to lower the order of x^2-derivatives in P^1 through changes $P^1 \rightarrow P^1 - D_2 R$, $P^2 \rightarrow P^2 + D_1 R$. R is found by repeated partial integration of terms in P^1 with highest x^2-derivatives of u. For that, partial integration routines of CRACK are used which are limited in applicability to expressions at most polynomially non-linear in u and derivatives of u.

5 Examples

Computation times refer to a 24 MB REDUCE 3.7 session under LINUX on a 133 MHz Pentium PC with the Jan. 1998 version of CRACK.

Example 1:
The advantage of using the package CRACK for solving determining equations is that they can be PDEs and do not have to be restricted to algebraic equations for coefficients of a polynomial ansatz for the CL. By that it is possible to find non-polynomial CLs and CLs that have an explicit x^i dependence. An example is the Harry Dym equation

$$\Delta = u_t - u^3 u_{xxx}, \qquad u = u(t,x)$$

which was used below to substitute u_t and derivatives of u_t. These calculations were done with CONLAW1.
P^t of order 0: time to formulate (1): 0.32 sec, to solve (1): 1.34 sec, CLs:

$$\begin{array}{rclcl}
2u^{-2} \cdot \Delta &=& D_t(-2u^{-1}) &+& D_x({u_x}^2 - 2uu_{xx}) \\
2u^{-3} \cdot \Delta &=& D_t(-u^{-2}) &+& D_x(-2u_{xx}) \\
2xu^{-3} \cdot \Delta &=& D_t(-xu^{-2}) &+& D_x(2u_x - 2xu_{xx}) \\
2x^2u^{-3} \cdot \Delta &=& D_t(-x^2u^{-2}) &+& D_x(4xu_x - 2x^2u_{xx} - 4u)
\end{array}$$

P^t of order 1: time to formulate (1): 0.32 sec, to solve (1): 2.6 sec, CLs:

$$(2uu_{xx} - {u_x}^2)u^{-2} \cdot \Delta =$$

$$D_t(-{u_x}^2u^{-1}) + D_x((2u_tu_x - {u_{xx}}^2u^3 + u_{xx}{u_x}^2u^2 - {u_x}^4u/4)u^{-1})$$

P^t of order 2: time to formulate (1): 0.7 sec, to solve (1): 158 sec, CL:

$$(-8u_{xxxx}u^3 - 16u_{xxx}u_xu^2 - 12{u_{xx}}^2u^2 + 12u_{xx}{u_x}^2u - 3{u_x}^4)u^{-2} \cdot \Delta =$$

$$D_t((-4{u_{xx}}^2u^2 - 3u_{xx}{u_x}^5tu - {u_x}^4)u^{-1}) +$$
$$D_x((8u_{tx}u_{xx}u^2 + 3u_{tx}{u_x}^5tu - 8u_tu_{xxx}u^2 - 8u_tu_{xx}u_xu + 4u_t{u_x}^3 +$$
$$4{u_{xxx}}^2u^5 + 4{u_{xx}}^3u^4 - 6{u_{xx}}^2{u_x}^2u^3 + 3u_{xx}{u_x}^4u^2)u^{-1})$$

Example 2:
The Burgers equation in the form

$$\Delta = u_t - u_{xx} - \frac{1}{2}u_x^2 = 0, \qquad u = u(t,x) \tag{15}$$

is an example for the case that the determining equations cannot be solved completely. It has zeroth order CLs

$$fe^{u/2}\Delta = D_t(2fe^{u/2}) + D_x(e^{u/2}(2f_x - fu_x)) \tag{16}$$

with $f = f(t,x)$ satisfying the linear reverse heat equation $0 = f_t + f_{xx}$. This CL is also an example that CONLAW allows the computation of CLs with non-rational terms which is not possible with approaches based on a polynomial ansatz. A remaining linear PDE and the occurrence of free functions in the CL indicates linearizability of $\Delta = 0$ which is the case with the Burgers equation.

Example 3:
The MVDNLS equations (Modified Vector Derivative Nonlinear Schrödinger equations) describe oblique propagation of magnetohydrodynamic waves in warm plasmas [6]. For functions $u = u(t,x),\ v = v(t,x)$ and $b =$ const. they are

$$\begin{aligned} \Delta_1 &= u_t + [u(u^2+v^2) + bu - v_x]_x && (17)\\ \Delta_2 &= v_t + [v(u^2+v^2) + u_x]_x. && (18) \end{aligned}$$

Both equations have the form of CLs. Using the abbreviations (introduced by hand afterwards)

$$\begin{aligned} E &= -v_x + u(u^2+v^2)\\ F &= \ \ u_x + v(u^2+v^2-b)\\ G &= 2u_{xx} + 6v_x(u^2+v^2) - 3u(u^2+v^2)^2 - 2bu^3\\ H &= 2v_{xx} - 6u_x(u^2+v^2) - 3v(u^2+v^2)^2 + 2bv^3\\ I &= b(u^4-v^4) + (u^2+v^2)^3 - 2u_x^2 - 2v_x^2 \end{aligned}$$

and using equ.s (17), (18) to substitute for u_t, v_t, further CLs calculated by CONLAW2/3 have the characteristics $\{Q^1, Q^2\}$:

$$\{u,v\},\ \ \{E,F\},\ \ \{G,H\}, \tag{19}$$

$$\{(bt-2x)E - 2tG + b(bt-x)u + v,\ \ (bt-2x)F - 2tH + b(bt-x)v - u\}, \tag{20}$$

$$\{-H_x + 2uvH + (b+2u^2)G + uI,\ \ G_x + 2uvG + 2v^2H + vI\}. \tag{21}$$

CONLAW2 can compute one more CL with Q^1, Q^2 of 4'th order and 36 terms each. Run times are listed in table 1.

Apart from (20) these CLs are given in [6] where also a bi-Hamiltonian structure is provided. Although from the resulting recursion operator, an infinite sequence of conserved densities can be calculated, the CL (20) is not contained in that sequence and is new - it has an explicit t,x-dependence.

In the scope of CONLAW1 to find CLs with P^1 of order 1 are CLs (19),(20) and if equations (17),(18) are used to substitute u_{xx}, v_{xx} then also (21) is included. Such a run of CONLAW1 returns a differential Gröbner Basis of 2 equations for one function in 3 variables and 2 equations for one function in 2 variables, which could not be solved completely because one of the ODEs is a second order ODE that could not be solved automatically.

CONLAW	order of P^t for `CONLAW1`, order of Q for `CONLAW2/3`									
	0		1		2		3		4	
	t_1	t_2	t_1	t_2	t_1	t_2	t_1	t_2	t_1	t_2
1	0.15	2.9	0.15	1977						
2	1.7	2.0	2.7	16	4.5	194	8.5	722	17	2784
3	0.17	4.5	0.18	11.7	0.3	28.5	0.6	377	1.9	low memory

Table 1: Run times t_1 to formulate and t_2 to solve determining conditions of CLs of the MVDNLS equations

6 Comparison of the three methods

The determining equations (1)-(3) differ in the number of functions, number of variables and their order.

For example, for the MVDNLS equations (17),(18) the condition (1) for CLs with P^1 of order 2 and the conditions (2),(3) for CLs with Q^μ of order 3 have the following characteristics:

(1): 1 condition in 12 variables $(t, x, u, v, u_x, v_x, \ldots, u_{4x}, v_{4x})$, 2 of which occur only explicitly (u_{4x}, v_{4x}), with 55 terms linear in functions P^t of 8 variables $(t, x, u, v, u_x, v_x, u_{xx}, v_{xx})$ and P^x of 10 variables $(t, x, u, v, \ldots, u_{xxx}, v_{xxx})$ and their 1st order derivatives. The unsymmetry in the dependencies of P^t, P^x at the beginning of `CONLAW1` is necessary because of the unsymmetry in using (17), (18) to substitute a first order t-derivative of u by a second order x-derivative.

(2): 1 condition in 22 variables $(t, x, u, v, \ldots, u_{(3)}, v_{(3)})$, 6 of which occur only explicitly (2nd order derivatives of u_t, v_t), with 37 terms linear in functions P^t, P^x of 14 variables $(t, x, u, v, \ldots, u^{(2)}, v^{(2)})$ and their 1st order derivatives, and furthermore functions Q^1, Q^2 of 10 variables $(t, x, u, v, \ldots, u_{xxx}, v_{xxx})$.

(3): 2 coupled conditions in 14 variables $(t, x, u, v, u_x, v_x, \ldots, u_{5x}, v_{5x})$, 4 of which occur only explicitly $(u_{4x}, v_{4x}, u_{5x}, v_{5x})$, with 131 and 132 terms linear in functions Q^1, Q^2 of 10 variables $(t, x, u, v, \ldots, u_{xxx}, v_{xxx})$ and their 1st and 2nd order derivatives.

The following are general features of equations (1)-(3).

Equ. (1) is of first order and therefore only highest order u-derivatives which are not substituted due to $0 = \Delta$ are not variables to the P^i and can be used for direct separation. Equ. (1) therefore is only weakly overdetermined with the application of integrability conditions playing an important role. A general problem with computing a differential Gröbner Basis is that the complexity of these calculations depends heavily on the total ordering of derivatives of functions P, Q chosen for which there is currently no complete theory available. Choices made by the program can be particularly good or bad for the problem at hand.

In contrast, equ.s (3) are of higher order with more jet-variables that occur only explicitly and that can be used for direct separation. Although these equations are of higher order they are highly overdetermined and simpler to solve in general. An efficient way of doing direct separations and handling large equations is of importance for this approach.

Finally, in equ.s (2) the P^i depend initially on all jet-variables (apart from highest order u-derivatives), also those substituted through $0 = \Delta$ on which the Q^μ do not depend. On the other hand the Q^μ do depend on highest order u-derivatives initially. The efficiency in solving (2) therefore depends on the efficiency of a module for indirect separation, i.e. on a module for handling equations which have no function depending on all variables but which have also no variable occurring only explicitly so that no direct separation with respect to any variable is possible. Such a module is described in [7].

To solve the overdetermined system of all three approaches, all techniques are used, only some are used more often in one approach than in the other.

There is another issue. If the order of derivatives w.r.t. different variables differs, like, for the Harry Dym equation $0 = u_t - u^3 u_{xxx}$, then it matters whether this equation is used to do substitutions $u_t = u^3 u_{xxx}$ or $u_{xxx} = u_t/u^3$. Substituting u_t gives a lower increase in complexity when successively higher order ansätze for P or Q are made. On the other hand one has to go to higher orders of P and Q to cover the same equivalence classes of CLs compared to substituting u_{xxx}. As equ.s (3) involve already higher order u-derivatives, a further increase could explode the size of (3) even more.

Another relation between (2) and (3) is that one could look at (3) as resulting from a differential-Gröbner-Basis calculation done with (2), with the aim to eliminate the P^i first. It is of course more efficient to exploit knowledge of the structure of (2) and to apply the Euler operator to write down (3) directly rather than to do the differential Gröbner Basis calculation step by step with (2). On the other hand `CRACK` includes a number of modules to take advantage of special situations (e.g. to integrate exact PDEs or to recognize and solve PDEs that are ODEs for some partial derivatives and to solve them using `ODESOLVE` [4]). For a concrete problem it is very likely that there exists a quicker way to solve (2) than to eliminate at first all P^i. The question which of the `CONLAW` programs is more effective depends on the effectiveness of different submodules of the program `CRACK` which solves (1)-(3). With the current version of `CRACK` (Jan. 1998), programs `CONLAW1/3` are better for simpler CL problems and `CONLAW2` is better for larger problems.

7 Syntax of CONLAW

Example: The input to find CLs with Q of order 0-4 for the MVDNLS equations (17),(18) is

```
depend u,x,t;
depend v,x,t;
conlaw2({{df(u,t) = - df( u*(u**2+v**2) + b*u - df(v,x) ,x),
          df(v,t) = - df( v*(u**2+v**2) + df(u,x)        ,x) },
         {u,v}, {t,x}
        },
        {0, 4, t, {}, {}});
```

In `REDUCE` lists are enclosed in `{ }`. The input of `CONLAWi` (i=1,2,3) consists of two lists, the first encodes the PDE problem. It contains a list of equations with the derivative to be substituted on the left hand side, a list of functions and a list of independent variables. The second parameter to `CONLAWi` is a list that specifies the CLs to be computed. Its first two elements are the minimum and maximum order of P^1 in the case of `CONLAW1` and the order of Q^μ in the case of `CONLAW2/3`. The third element is `t` or `nil` and specifies whether the CL may depend explicitly on the x^i or not. The fourth element is a list of functions to be determined in an ansatz made for P^i or Q^μ and the last element is a list of inequalities to be satisfied.

More details about investigating an ansatz is given in a manual file that comes with the three `CONLAW` files.

8 Summary

Supplied with subroutines to fix gauge freedom in differential expressions the programs `CONLAW1/2/3` proved to be a efficient tool for the computation of CLs of differential equations. Compared with other programs, a list of which and a short description is given in [2], the programs `CONLAWi` show the following new features:

- By solving systems of overdetermined differential equations it is possible to find CLs with non-polynomial, even non-rational P, Q.
- It is possible to find CLs with an explicit dependence of P, Q on the independent variables.
- There is no limit on the number of DEs nor the number of independent variables to be investigated for CLs other than a limit through the complexity of computations.
- It is possible to determine values of parameters in the DE such that CLs exist (examples in [9]).
- For each of the programs `CONLAWi` an ansatz for P^i and/or Q^μ can be input to specify CLs to be calculated.

Compared with the program of Göktaş and Hereman, `CONLAW` is able to find more general CLs and to make a definitive statement if local CLs do not exist and the order is not too high to complete the computations.

The strength of the program described in [2] is to get sometimes higher in the order that still can be handled by concentrating on polynomial CLs having to solve algebraic systems for coefficients of a polynomial ansatz. They were also able to extend applicability to differential-difference systems [3].

The comparison of the three approaches (1)-(3) showed that each of them has advantages in special circumstances. It also serves as a comparison between using a general purpose program to find the quickest way of solving overdetermined PDE systems directly (`CONLAW1/3`) and an approach to derive integrability conditions by applying extra information about the structure of the PDE system (`CONLAW2`).

The programs including a manual and a test file are available via ftp from `lie.maths.qmw.ac.uk`, directory `pub/compalg`. The package will be submitted to the REDUCE network library.

9 Acknowledgement

The authors want to thank Malcolm MacCallum for comments on a first version of the manuscript. Further, the support at a research visit of TW at CAN Netherlands/Amsterdam and discussions with Jan Sanders and Willy Hereman and multiple visits at the Konrad Zuse Institute/Berlin and discussions with Winfried Neun are gratefully acknowledged. Frank Verheest, Willy Sarlet, Micheline Musette and the Relativity group at Hall University are thanked for suggesting PDEs to test the code.

References

[1] Anco, S.C. and Bluman, G. (1997). *Direct Construction of Conservation Laws from Field Equations* Phys. Rev. Let. **78**, no 15, 2869-2873.

[2] Göktaş, Ü. and Hereman, W. (1996) *Symbolic Computation of Conserved Densities for Systems of Nonlinear Evolution Equations.* Journal of Symbolic Computation, vol. 24, pp. 591-621 (1997).

[3] Göktaş, Ü., Hereman, W. and Erdmann, G. (1997) *Computation of conserved densities for systems of nonlinear differential-difference equations.*, Physics Letters A, vol. 236, pp. 30-38 (1997).

[4] MacCallum, MAH. (1988) *An Ordinary Differential Equation Solver for REDUCE* Proc. ISSAC‘88, Springer Lect. Notes in Comp. Sc., 358, 196-205.

[5] Olver, P.J. (1986). *Applications of Lie Groups to Differential Equations.* Grad. Texts in Math. Berlin: Springer-Verlag.

[6] Willox, R., Hereman, W. and Verheest, F. (1995). *Complete Integrability of a Modified Vector Derivative Nonlinear Schrödinger Equation.* Physica Scripta **52**, 21-26.

[7] Wolf, T. and Brand, A. (1992) *The Computer Algebra Package CRACK for Investigating PDEs*, manual + software in the `REDUCE` network library.

[8] Wolf, T. (1996) *The program CRACK for solving PDEs in General Relativity* in *Relativity and Scientific Computing* Eds. F.W.Hehl, R.A.Puntigam, H.Ruder, Springer, 241-257.

[9] Wolf, T. To be published elsewhere.