Symbolic Software for Soliton Theory

W. Hereman[1] and W. Zhuang

Department of Mathematical and Computer Sciences
Colorado School of Mines
Golden, CO 80401, USA

Appeared in: Acta Applicandae Mathematicae **39**, pp. 361-378, 1995

Also: Proceedings of KdV '95, International Symposium,
Amsterdam, The Netherlands, April 23-26, 1995,
Ed: M. Hazewinkel, Kluwer Academic Publishers,
Dortrecht, The Netherlands, 1995, pp. 361-378 (1995).

August 1, 2007

MCS-94-13 September 1994

1991 Mathematics Subject Classification.

Primary 35Q51, 35Q53, 35C05, 68Q40; Secondary 34-04, 35-04

[1]Research Supported in Part by NSF under Grant CCR-9300978

Abstract

Four symbolic programs, in Macsyma or Mathematica language, are presented. The first program tests for the existence of solitons for nonlinear PDEs. It explicitly constructs solitons using Hirota's bilinear method. In the second program, the Painlevé integrability test for ODEs and PDEs is implemented. The third program provides an algorithm to compute conserved densities for nonlinear evolution equations. The fourth software package aids in the computation of Lie symmetries of systems of differential and difference-differential equations. Several examples illustrate the capabilities of the software.

Key words:

soliton theory, symbolic programs, Hirota method, Painlevé test, Lie symmetries, conserved densities.

1 Introduction

The search for special properties (such as integrability), exact solutions, Lie symmetries and conserved densities of nonlinear partial differential equations (PDEs) becomes more and more attractive due to the availability of symbolic manipulation programs such as MACSYMA, MAPLE, REDUCE, MATHEMATICA, AXIOM, MuPAD and DERIVE, which allow to perform the tedious algebra and routine computations.

Four computer programs for symbolic computations relevant to soliton theory are presented. The programs are written in Macsyma and/or MATHEMATICA syntax. They are entirely symbolic and produce exact analytical output.

The first program tests for the existence of solitary wave and soliton solutions of nonlinear PDEs in bilinear form. It also explicitly constructs multi-soliton solutions via Hirota's method. The second program automatically carries out the Painlevé integrability test for ODEs and PDEs. The third program offers a simple algorithm to compute conserved densities for nonlinear evolution equations. The fourth software package aids in the computation of various types of Lie symmetries of ODEs, PDEs and difference-differential equations.

The purpose, methods, algorithms of the four programs are outlined. Implementation issues are briefly discussed and several examples illustrate the capabilities of the software.

2 Hirota's Method

2.1 Purpose

Hirota's method [11, 12] allows one to construct exact soliton solutions of nonlinear evolution and wave equations, provided the equations can be brought in bilinear form.

Our MACSYMA program, called HIROTA_SINGLE.MAX, symbolically computes the one-, two- and three-soliton solutions of well-known nonlinear PDEs. Thus far, the bilinear representation of the PDE must be of Korteweg-de Vries (KdV) type [9]. The program also automatically verifies if three- and four-soliton solutions will exist for such equations. The development and implementation of a comprehensive algorithm that could handle other types of bilinear representations [9] is in progress.

2.2 Algorithm and Implementation

Details about the method can be found in almost any book on soliton theory, [1, 2, 4] here we outline the procedure.

Hirota's method requires: (i) a clever change of dependent variable; (ii) the introduction of a novel differential operator; (iii) a perturbation expansion to solve the resulting bilinear equation.

Our leading example is the Korteweg-de Vries equation,

$$u_t + 6uu_x + u_{3x} = 0. \tag{1}$$

Substitution of

$$u(x,t) = 2\frac{\partial^2 \ln f(x,t)}{\partial x^2} = 2\left(\frac{ff_{xx} - f_x^2}{f^2}\right). \tag{2}$$

into (1) and one integration with respect to x yields,

$$ff_{xt} - f_x f_t + ff_{4x} - 4f_x f_{3x} + 3f_{2x}^2 = 0. \tag{3}$$

This quadratic equation in f can then be written in *bilinear form*,

$$B(f\cdot f) \stackrel{\text{def}}{=} P(D_x, D_t)(f\cdot f) \stackrel{\text{def}}{=} \left(D_x D_t + D_x^4\right)(f\cdot f) = 0, \tag{4}$$

where the new operator is given by

$$D_x^m D_t^n (f\cdot g) = (\partial x - \partial x')^m (\partial t - \partial t')^n f(x,t)\, g(x',t')\Big|_{x'=x,t'=t}. \tag{5}$$

In (4), P should be interpreted as a polynomial in its arguments, B abbreviates the bilinear operator for the KdV equation.

Introducing a book keeping parameter ϵ, we seek a solution

$$f = 1 + \sum_{n=1}^{\infty} \epsilon^n \, f_n. \tag{6}$$

Substituting (6) into (4) and equating to zero the powers of ϵ, yields

$$\begin{aligned}
O(\epsilon^0) &\;:\; B(1{\cdot}1) = 0, & (7)\\
O(\epsilon^1) &\;:\; B(1{\cdot}f_1 + f_1{\cdot}1) = 0, & (8)\\
O(\epsilon^2) &\;:\; B(1{\cdot}f_2 + f_1{\cdot}f_1 + f_2{\cdot}1) = 0, & (9)\\
O(\epsilon^3) &\;:\; B(1{\cdot}f_3 + f_1{\cdot}f_2 + f_2{\cdot}f_1 + f_3{\cdot}1) = 0, & (10)\\
O(\epsilon^4) &\;:\; B(1{\cdot}f_4 + f_1{\cdot}f_3 + f_2{\cdot}f_2 + f_3{\cdot}f_1 + f_4{\cdot}1) = 0, & (11)\\
O(\epsilon^n) &\;:\; B(\sum_{j=0}^{n} f_j{\cdot}f_{n-j}) = 0, \qquad \text{with } \; f_0 = 1, & (12)
\end{aligned}$$

If the original PDE admits a N-soliton solution then (6) will truncate at level $n = N$ provided f_1 is the sum of N exponential terms. For simplicity, consider the case of a three soliton solution ($N = 3$), where

$$f_1 = \sum_{i=1}^{3} \exp(\theta_i) = \sum_{i=1}^{3} \exp\,(k_i\, x - \omega_i\, t + \delta_i), \tag{13}$$

with k_i, ω_i and δ_i constant. Of course, (7) is trivially satisfied, whereas (8) determines the dispersion law, $\omega_i = k_i^3,\; i = 1, 2, 3.$

The terms generated by $B(f_1, f_1)$ in (9) justify the choice

$$\begin{aligned}
f_2 \;&=\; a_{12} \exp(\theta_1 + \theta_2) + a_{13} \exp(\theta_1 + \theta_3) + a_{23}\, exp(\theta_2 + \theta_3)\\
&=\; a_{12} \exp\left[(k_1 + k_2)x - (\omega_1 + \omega_2)t + \delta_1 + \delta_2\right] + a_{13} \exp\left[(k_1 + k_3)x - (\omega_1 + \omega_3)t + \delta_1 + \delta_3\right]\\
&+\; a_{23} \exp\left[(k_2 + k_3)x - (\omega_2 + \omega_3)t + \delta_2 + \delta_3\right],
\end{aligned} \tag{14}$$

and (9) allows one to calculate the coupling constants

$$a_{ij} = \frac{(k_i - k_j)^2}{(k_i + k_j)^2}, \qquad i, j = 1, 2, 3. \tag{15}$$

Then, $B(f_1{\cdot}f_2 + f_2{\cdot}f_1)$ in (10) motivates the particular solution

$$f_3 = b_{123} \exp(\theta_1 + \theta_2 + \theta_3) = b_{123} \exp\left[(k_1{+}k_2{+}k_3)x - (\omega_1{+}\omega_2{+}\omega_3)t + \delta_1{+}\delta_2{+}\delta_3\right], \tag{16}$$

and one computes

$$b_{123} = a_{12}\, a_{13}\, a_{23} = \frac{(k_1 - k_2)^2\, (k_1 - k_3)^2\, (k_2 - k_3)^2}{(k_1 + k_2)^2\, (k_1 + k_3)^2\, (k_2 + k_3)^2}. \tag{17}$$

Subsequently, (11) allows one to verify that indeed $f_4 = 0$. In the sixth equation of the scheme $B(f_2 \cdot f_3 + f_3 \cdot f_2)$ should equal zero in order to assure that $f_5 = 0$. If so, it will be possible to take $f_i = 0$ for $i > 6$. Finally, setting $\epsilon = 1$ in (6), we obtain

$$\begin{aligned} f &= 1 + \exp\theta_1 + \exp\theta_2 + \exp\theta_3 + a_{12}\exp(\theta_1 + \theta_2) \\ &+ a_{13}\exp(\theta_1 + \theta_3) + a_{23}\exp(\theta_2 + \theta_3) + b_{123}\exp(\theta_1 + \theta_2 + \theta_3), \end{aligned} \tag{18}$$

which upon substitution in (2) generates the well-known three soliton solution of (1).

For the N-soliton solution, Hirota [12] starts with a generalization of (18):

$$f = \sum_{\mu=0,1} \exp\left[\sum_{i<j}^{(N)} A_{ij}\mu_i\mu_j + \sum_{i=1}^{N} \mu_i\theta_i\right], \tag{19}$$

where the θ_i are as in (13) for $i = 1, 2, \ldots, N$, and $a_{ij} = \exp A_{ij}$ and $b_{123} = a_{12}a_{13}a_{23}$. The summation $\sum_{\mu=0,1}$ is over all combinations of $\mu_1 = 0, 1,\ \mu_2 = 0, 1,\ \ldots,\ \mu_N = 0, 1$. The summation $\sum_{i<j}^{(N)}$ stands for all possible combinations under the condition $0 < i < j \leq N$.

Concerning the computer implementation, Hirota's bilinear operators (5) are redefined as

$$Dxt[m, n](f, g) = \sum_{j=0}^{m}\sum_{i=0}^{n} \frac{(-1)^{(m+n-j-i)} m!}{j!(m-j)!} \frac{n!}{i!(n-i)!} \frac{\partial^{i+j} f}{\partial t^i \partial x^j} \frac{\partial^{n+m-i-j} g}{\partial t^{n-i} \partial x^{m-j}}, \tag{20}$$

with similar expression for $Dt[n](f, g), Dx[n](f, g)$ and $Dy[n](f, g)$. The observant reader will recognize the Leibniz formula for the derivatives of products, up to an alteration in signs.

In practice, the dispersion law and the coupling coefficients (15) and (17) can be computed as follows. First determine $P(k, -\omega)$ from

$$B(e^{kx-\omega t+\delta} \cdot 1) = P(k, -\omega)e^{kx-\omega t\delta}, \tag{21}$$

and solve $P(k, -\omega) = 0$ to get the dispersion law. In our example, $P(k, -\omega) = k^4 - \omega k = 0$, yields $\omega = k^3$.

It is easy to show [17] that, once P is known,

$$a_{ij} = -\frac{P(k_i - k_j, -\omega_i + \omega_j)}{P(k_i + k_j, -\omega_i - \omega_j)}, \quad b_{123} = a_{12}a_{13}a_{23}. \tag{22}$$

Note that the knowledge of the forms of P and f in (19) allow one to circumvent the explicit solution of the bilinear scheme (7)-(12).

With the polynomial P at hand, the conditions for the existence of a one- and two-soliton solution are $P(D_x, D_t) = P(-D_x, -D_t)$, and $P(0, 0) = 0$.

To have a N-soliton solution, Hirota [12] and Matsuno [14] showed that the condition

$$S[P, n] = \sum_{\sigma=\pm 1} P\left(\sum_{i=1}^{n} \sigma_i k_i, -\sum_{i=1}^{n} \sigma_i \omega_i\right) \prod_{i<j}^{(n)} P(\sigma_i k_i - \sigma_j k_j, -\sigma_i \omega_i + \sigma_j \omega_j)\sigma_i \sigma_j = 0 \tag{23}$$

must be satisfied for $n = 2, 3, \ldots, N$. Recall that $\sum_{\sigma=\pm 1}$ indicates the summation over all possible combinations of $\sigma_1 = \pm 1,\ \sigma_2 = \pm 1,\ \ldots,\ \sigma_n = \pm 1$ and that $\prod_{i<j}^{(n)}$ refers to the product over all possible combinations of n elements under the condition $i < j$, and all ω_i eliminated via the dispersion law.

The verification that $S[P, n] = 0$ for $n = 3$ and $n = 4$ is nontrivial. The user can choose to test the conditions with constrained random numbers (fast) or symbolically (much slower).

2.3 Examples

- For the Boussinesq equation (BE) [4]

$$u_{2t} - u_{2x} - 3(u^2)_{2x} - u_{4x} = 0, \tag{24}$$

the bilinear operator is

$$B(f, g) = Dxt[0, 2](f, g) - Dx[2](f, g) - Dx[4](f, g)\ . \tag{25}$$

The BE admits three- and four-soliton solutions, confirmed during testing. With $\theta_i = k_i x - \omega_i t + \delta_i$, the dispersion relation is $\omega = -k\sqrt{1 + k^2}$, and the coupling coefficients of the three-soliton solution are

$$a_{ij} = \frac{\sqrt{1 + k_i^2}\sqrt{1 + k_j^2} - 2k_i^2 + 3k_i k_j - 2k_j^2 - 1}{\sqrt{1 + k_i^2}\sqrt{1 + k_j^2} - 2k_i^2 - 3k_i k_j - 2k_j^2 - 1}, \quad i, j = 1, 2, 3 \quad \text{and} \quad i < j, \tag{26}$$

and $b_{123} = a_{12}\, a_{13}\, a_{23}$.

- In this second example, we illustrate the use of the program HIROTA_SINGLE.MAX in the search for integrable equations. Consider the bilinear equation [10]

$$(kD_t^2 + D_x^3 D_t + D_x^6)(f \cdot f) = 0, \tag{27}$$

with an arbitrary parameter k. It is known that this equation passes the Painlevé test for $k = -1/5$. With our program we calculate the condition for the existence of a three-soliton solution:

$$\frac{5k + 1}{k^3}(2k^2 + 2k + \sqrt{1 - 4k} - 1)(k_1 k_2 k_3)^4 \left[(k_2^2 - k_1^2)(k_3^2 - k_1^2)(k_3^2 - k_2^2)\right]^2 = 0, \tag{28}$$

fueling the conjecture that the existence of a three-soliton solution implies complete integrability [18]. With $k = -1/5$, we continue the calculation of the two- and three-soliton solutions of (27). We obtained completely automatically

$$a_{ij} = \frac{(k_i - k_j)^2(\sqrt{5}k_i^2 + 3k_i^2 + \sqrt{5}k_i k_j + k_i k_j + \sqrt{5}k_j^2 + 3k_j^2)}{(k_i + k_j)^2(\sqrt{5}k_i^2 + 3k_i^2 + \sqrt{5}k_i k_j + k_i k_j + \sqrt{5}k_j^2 + 3k_j^2)}, \tag{29}$$

$$b_{123} = a_{12}\, a_{13}\, a_{23}. \tag{30}$$

Then, with the program, we verified that the equation also admits a four-soliton solution, suggesting that the equation is indeed completely integrable.

3 The Painlevé Test

3.1 Purpose

The MACSYMA program PAINLEVE_SINGLE.MAX allows one to determine whether or not a given single nonlinear ODE or PDE with (real) polynomial terms fulfills the necessary conditions for having the Painlevé property.

A PDE is said to possess the Painlevé property [1] if its solutions in the complex plane are single-valued in the neighborhood of non-characteristic, movable singular manifolds. For ODEs, this means that solutions should have no worse singularities than movable poles. Such equations are prime candidates for being completely integrable.

3.2 Algorithm and Implementation

We illustrate the algorithm for a single PDE. The solution u, say in only two independent variables (x, t), expressed as a Laurent series,

$$u(x,t) = g^{\alpha} \sum_{k=0}^{\infty} u_k(x,t) g^k(x,t) \tag{31}$$

should only have movable poles. In (31), $u_0(t,x) \neq 0$, α a negative integer, and $u_k(x,t)$ are analytic functions in a neighborhood of the singular, non-characteristic manifold $g(x,t) = 0$, with $g_x(x,t) \neq 0$. For an ODE $g(x) = x - x_0$, where x_0 is the initial value for x.

The Painlevé test is carried out in three steps:

Step 1: Determine the negative integer α and u_0 from the leading order "ansatz". This is done by balancing the minimal power terms after substitution of $u \propto u_0 g^{\alpha}$ into the given PDE.

Step 2: Calculate the non-negative integer powers r, called the *resonances*, at which arbitrary functions u_r enter the expansion. This is done by requiring that u_r is arbitrary after substitution of $u \propto u_0 g^{\alpha} + u_r g^{\alpha+r}$ into the equation, only retaining its most singular terms.

Step 3: Verify that the correct number of arbitrary functions u_r indeed exists by substituting the truncated expansion $u(x,t) = g^{\alpha} \sum_{k=0}^{\mathrm{rmax}} u_k(x,t) g^k(x,t)$ into the given equation ($rmax$ is the largest resonance). At non-resonance levels, determine all u_k unambiguously. At resonance levels, u_r should be arbitrary, and since we are dealing with a nonlinear equation, a *compatibility condition* must be satisfied.

An equation (or system) for which the above steps can be carried out consistently, and for which the compatibility conditions at all resonances are satisfied, is said to have the Painlevé property and is conjectured to be completely integrable.

The reader should be warned that the above algorithm does not detect essential singularities. In other words, for an equation to be integrable it is *necessary* but *not* yet *sufficient* that it passes the Painlevé test. There are indeed integrable equations, such as the Dym-Kruskal equation, $u_t = u^3 u_{xxx}$, that only have the Painlevé property after a suitable chance of variables.

3.3 Examples

Instead of just testing some equations for the Painlevé property (see e.g. [8]), we show how to use the symbolic program PAINLEVE_SINGLE.MAX in the investigation of integrability and the search for exact solutions.

- Consider a 2D KdV equation [20] with time-dependent coefficients $a(t)$ and $b(t)$,

$$(u_t + 6uu_x + u_{xxx})_x + a(t)u_x + b(t)u_{yy} = 0, \tag{32}$$

and determine the coefficients for which the equation will pass the Painlevé test.

In this case we seek a solution of the form

$$u(x, y, t) = g(x, y, t)^{\alpha} \sum_{k=0}^{\infty} u_k(x, y, t) g(x, y, t)^k, \tag{33}$$

with singularity $g(x, y, t)$. The program PAINLEVE_SINGLE.MAX determines $\alpha = -2$, $u_0 = -2(g_x)^2$, and computed the resonances $r = 4, 5$ and 6. The compatibility conditions at levels 4 and 5 are identically satisfied. At resonance level $r = 6$ the compatibility condition

$$[a_t + 2a^2]{g_x}^3 + [b_t + 4ab]({g_x}^2 g_{yy} + g_{xx}{g_y}^2 - 2g_x g_{xy} g_y) = 0 \tag{34}$$

must hold irrespective the form of the singularity manifold $g(x, y, t)$. Thus, (32) will have the Painlevé property if

$$a_t + 2a^2 = 0, \quad b_t + 4ab = 0. \tag{35}$$

Two cases are possible. The first case with $a(t) = 0, b(t) = c$, for any constant c, leads to the KdV equation itself ($c = 0$), or to the Kadomtsev-Petviashvili (KP) equation ($c = \pm 1$).

The more interesting second case, with

$$a(t) = \frac{1}{2(t - t_0)}, \qquad b(t) = \frac{k}{(t - t_0)^2}, \tag{36}$$

where k and t_0 are integration constants, leads for $k = 0$ and upon integration with respect to x, to the cylindrical KdV equation [1],

$$u_t + 6uu_x + u_{xxx} + \frac{u}{2t} = 0, \tag{37}$$

which describes solitary waves in a channel with slowly varying depth. For $k \neq 0$, upon appropriate scaling of the variable y, equation (32) reduces to the well-known cylindrical KP equation [1],

$$(u_t + 6uu_x + u_{xxx})_x + \frac{1}{2t}u_x + \frac{3\sigma^2}{t^2}y_{yy} = 0, \qquad \sigma^2 = \pm 1, \tag{38}$$

which describes surface waves in a fluid which are characterized by small deviation from axial symmetry. The KdV and KP equations and their cylindrical generalizations (37) and (38) are all known to be completely integrable [1].

- In this second example, we seek an exact solution of the Fitzhugh-Nagumo (FHN) equation with convection term [16],

$$u_t + kuu_x - u_{xx} - u(1-u)(a-u) = 0. \tag{39}$$

For (39), $\alpha = -1$ and u_0 must satisfy $u_0^2 - ku_0 f_x - 2f_x^2 = 0$. The resonances are $r = -1$ and $r = 4 + k(\frac{u_0}{f_x}) = 2 + (\frac{u_0}{f_x})^2$. In order to have integer resonances, we must have $u_0(x,t) = \sqrt{m} f_x$ and $k = \frac{m-2}{\sqrt{m}}$, for any positive integer m.

To find a traveling wave solution, we substitute the Laurent series, truncated at the constant level term,

$$u(x,t) = \sqrt{m}\frac{\partial \ln f(x,t)}{\partial x} + u_1(x,t) = \sqrt{m}\frac{f_x(x,t)}{f(x,t)} + u_1(x,t) \tag{40}$$

into (39) with $k = \frac{m-2}{\sqrt{m}}$, and, we set the different power terms in f separately equal to zero. This results in an overdetermined system in $f(x,t)$ and $u_1(x,t)$:

$$f_t - (1+m)f_{xx} - \sqrt{m}\,[\frac{2}{m}(1+m)\,u_1 - 1 - a]\,f_x = 0, \tag{41}$$

$$f_{xt} - f_{xxx} + \frac{1}{\sqrt{m}}(m-2)\,u_1\,f_{xx} + [3{u_1}^2 - 2(1+a)u_1 + a + \frac{1}{\sqrt{m}}(m-2)\,(u_1)_x]\,f_x = 0, \tag{42}$$

$$(u_1)_t + \frac{1}{\sqrt{m}}(m-2)u_1(u_1)_x - (u_1)_{xx} + u_1(1-u_1)(a-u_1) = 0. \tag{43}$$

The above system is linear if u_1 is any of the constant solutions of the FHN equation (43), i.e. $u_1 = 0, 1, \text{or}\, a$. For $u_1 = 0$ and

$$f = c + \sum_{i=1}^{N} f_i = \sum_{i=1}^{N} \exp(\theta_i) = c + \sum_{i=1}^{N} \exp\left(k_i\,x - \omega_i\,t + \delta_i\right), \tag{44}$$

, we find that $N = 2$ and $k_1 = \frac{1}{\sqrt{m}}$, $\omega_1 = \frac{am-1}{m}$, $k_2 = \frac{a}{\sqrt{m}}$, and $\omega_2 = \frac{a(m-a)}{m}$. Hence,

$$f = c + exp[\frac{1}{\sqrt{m}}x + \frac{a(a-m)}{m}t + \delta_1] + exp[\frac{a}{\sqrt{m}}x + \frac{(1-am)}{m}t + \delta_2]. \tag{45}$$

Returning to $u(x,t)$ via (40) yields

$$u(x,t) = \frac{exp[\frac{1}{\sqrt{m}}x + \frac{(1-am)}{m}t + \delta_1] + a\,exp[\frac{a}{\sqrt{m}}x + \frac{a(a-m)}{m}t + \delta_2]}{c + exp[\frac{1}{\sqrt{m}}x + \frac{(1-am)}{m}t + \delta_1] + exp[\frac{a}{\sqrt{m}}x + \frac{a(a-m)}{m}t + \delta_2]}. \tag{46}$$

This solution reduces for $m = 2$ to the known solution of the FHN equation without convection term.

4 Conserved Densities

4.1 Purpose

Our MATHEMATICA program DENSITY_SINGLE.M allows one to compute polynomial-type conservation laws of single PDEs. A similar program, called DENSITY_SYSTEM.M applies to systems of PDEs.

Recall that for a density ρ, such that there exists an associated flux J, obeying the conservation law

$$\rho_t + J_x = 0, \tag{47}$$

one finds that

$$P = \int_{-\infty}^{+\infty} \rho dx = \text{constant}, \tag{48}$$

provided J vanishes at infinity. The existence of a sufficiently large (in principal infinite) number of conservation laws of type (47) assures completely integrable of an evolution equation. Our MATHEMATICA programs automatically construct the pairs ρ and J.

4.2 Algorithm and Implementation

We explain the algorithm, based on ideas of Kruskal and co-workers [13, 15] and Verheest and Hereman [21], for the KdV equation

$$u_t + uu_x + u_{3x} = 0. \tag{49}$$

Information about the building blocks of $\rho(u, u_x, u_{2x}, \ldots, u_{nx})$ can be obtained from the scaling or symmetry properties of the equation. The scaling of (49) is such that

$$u \sim \frac{\partial^2}{\partial x^2}, \qquad \frac{\partial}{\partial t} \sim \frac{\partial^3}{\partial x^3}. \tag{50}$$

Ignoring the first two trivial conserved densities, corresponding to $\rho_1 = u$, and $\rho_2 = u^2$, we start with building block u^3, which has three factors u.

Keeping (50) in mind, this is equivalent to two factors u and two derivations (to be distributed over these two factors as uu_{2x} or as u_x^2), or one factor u and four derivations (obviously u_{4x}). But, uu_{2x} can be integrated by parts to yield $(uu_x)_x - u_x^2$, and $u_{4x} = (u_{3x})_x$. Removing any density (or part thereof) that is a total x–derivative, only u_x^2 must be kept. Therefore, ρ_3 is a linear combination of the building blocks u^3 and u_x^2. It is then straightforward to find the appropriate numerical coefficient, resulting in $\rho_3 = u^3 - 3u_x^2$.

In the computer implementation, the building blocks of $\rho(u, u_x, u_{2x}, \ldots, u_{nx})$ can easily be computed. To compute the building blocks of, say, ρ_6 one proceeds as follows:

(i) Start with u^6, divide it by u and, taking into account the scaling (50), compute $(u^5)_{2x}$. This produces the following list of terms $[u^3u_x^2, u^4u_{2x}] \longrightarrow [u^3u_x^2]$, where in the second list we have removed the terms that can be written either as a total derivative with respect to x, or as a total derivative up to terms appearing earlier in the list.

Next, divide u^6 by u^2, and compute $(u^4)_{4x}$, the corresponding lists are
$[u_x^4, uu_x^2 u_{2x}, u^2 u_{2x}^2, u^2 u_x u_{3x}, u^3 u_{4x}] \longrightarrow [u_x^4, u^2 u_{2x}^2]$.

In a similar fashion, proceed with $(\frac{u^6}{u^3})_{6x} = (u^3)_{6x}$, $(\frac{u^6}{u^4})_{8x} = (u^2)_{8x}$, and $(\frac{u^6}{u^5})_{10x} = (u)_{10x}$, to obtain the lists:
$[u_{2x}^3, u_x u_{2x} u_{3x}, uu_{3x}^2, u_x^2 u_{4x}, uu_{2x} u_{4x}, uu_x u_{5x}, u^2 u_{6x}] \longrightarrow [u_{2x}^3, uu_{3x}^2]$,
$[u_{4x}^2, u_{3x} u_{5x}, u_{2x} u_{6x}, u_x u_{7x}, uu_{8x}] \longrightarrow [u_{4x}^2]$, and $[u_{10x}] \longrightarrow [\,]$,
where [ ] refers to the empty list.
Gathering the terms in the simplified lists yields

$$\rho_6 = u^6 + c_1 u^3 u_x^2 + c_2 u_x^4 + c_3 u^2 u_{2x}^2 + c_4 u_{2x}^3 + c_5 uu_{3x}^2 + c_6 u_{4x}^2, \tag{51}$$

where the constants c_i have to be determined.

(ii) Now compute $\frac{\partial}{\partial t}\rho_6$, and replace $u_t, u_{xt}, \ldots, u_{nx,t}$ by $-(uu_x + u_{xxx})$ and it differential consequences.

(iii) Integrate the result with respect to x, carry out all integrations by parts, and require that the part that can no longer be integrated vanishes. Setting equal to zero all the coefficients of the independent combinations involving powers of u and its derivatives with respect to x, leads to a linear system in the unknowns $c_1, c_2, \ldots, c_6$. After solving that system, we found

$$\rho_6 = u^6 - 60u^3 u_x^2 - 30u_x^4 + 108u^2 u_{2x}^2 + \frac{720}{7} u_{2x}^3 - \frac{648}{7} uu_{3x}^2 + \frac{216}{7} u_{4x}^2. \tag{52}$$

(iv) Once the constants c_i are determined, the form of the flux J_6 can be computed by substituting these constants into the integrable part of ρ_6, and reversing the sign.

For simplicity, let us illustrate steps (ii)-(iv) for ρ_3. Applying step (i) leads to $\rho_3 = u^3 + c[1]u_x^2$. After replacement of u_t and u_{xt} in $\frac{\partial \rho_3}{\partial t}$ by $-(uu_x + u_{xxx})$ and $-(uu_x + u_{xxx})_x$, and integration with respect to x, we obtain

$$\frac{\partial \rho_3}{\partial t} = -[\frac{3}{4}u^4 + (c_1 - 3)uu_x^2 + 3u^2 u_{xx} - c_1 u_{xx}^2 + 2c_1 u_x u_{xxx}]_x - (c_1 + 3)u_x^3. \tag{53}$$

The last term in (53) must vanish. Hence, $c_1 = -3$, and the expression $[\ldots]$ gives

$$J_3 = \frac{3}{4}u^4 - 6uu_x^2 + 3u^2 u_{xx} + 3u_{xx}^2 - 6u_x u_{xxx}, \tag{54}$$

corresponding to $\rho_3 = u^3 - 3u_x^2$.

4.3 Examples

• Let us study the conservation laws, and consequently the integrability of a generalized Schamel (gS) equation [21]

$$n^2 u_t + (n+1)(n+2)u^{\frac{2}{n}} u_x + u_{xxx} = 0, \tag{55}$$

with n positive integer. We our MATHEMATICA program DENSITY_SINGLE.M we computed $\rho_1 = u, \rho_2 = u^2, \rho_3 = \frac{1}{2}u_x^2 - \frac{n^2}{2}u^{2+\frac{2}{n}}$. It has be shown [21] that these are the only conserved densities. Hence, establishing that for arbitrary values of n the gS equation is not completely integrable.

• Our MATHEMATICA program DENSITY_SYSTEM.M, was used to compute the first five nontrivial conserved densities of a modified vector derivative nonlinear Schrödinger equation (MVDNLS) [22],

$$\frac{\partial \mathbf{B}_\perp}{\partial t} + \frac{\partial}{\partial x}(B_\perp^2 \mathbf{B}_\perp) + \alpha \mathbf{B}_{\perp 0}\mathbf{B}_{\perp 0} \cdot \frac{\partial \mathbf{B}_\perp}{\partial x} + \mathbf{e}_x \times \frac{\partial^2 \mathbf{B}_\perp}{\partial x^2} = \mathbf{0}. \tag{56}$$

The parameter α characterizes the extra term which distinguishes the MVDNLS from the derivative NLS. Equation (56), which comes from plasma physics, can be replaced by

$$u_t + \Big(u(u^2+v^2) + \beta u - v_x\Big)_x = 0, \quad v_t + \Big(v(u^2+v^2) + u_x\Big)_x = 0, \tag{57}$$

where u and v denote the components of $\mathbf{B}_\perp$ parallel and perpendicular to $\mathbf{B}_{\perp 0}$, and $\beta = \alpha B_{\perp 0}^2$.

The first 5 conserved densities were computed. The first three are

$$\rho_1 = u^2 + v^2, \tag{58}$$

$$\rho_2 = \frac{1}{2}(u^2+v^2)^2 - uv_x + u_x v + \beta u^2, \tag{59}$$

$$\rho_3 = \frac{1}{4}(u^2+v^2)^3 + \frac{1}{2}(u_x^2 + v_x^2) - u^3 v_x + v^3 u_x + \frac{\beta}{4}(u^4 - v^4), \tag{60}$$

the remaining two can be found in [22].

5 Lie Point Symmetries

5.1 Purpose

The program SYMMGRP.MAX [3, 5, 6, 7] automatically computes the determining equations for the coefficients in the vector field that realizes the Lie algebra of point symmetries. With a feedback mechanism, these determining equations can then be solved explicitly.

The program SYMMGRP.MAX has been adapted in two ways. It now allows one to compute non-classical symmetries of systems of differential equations, and Lie-point symmetries (classical and non-classical) of system of difference-differential equations. Examples of the latter include the 1D Toda lattice, $u_{tt}(n) - e^{u(n-1)-u(n)} + e^{u(n)-u(n+1)} = 0$, the 2D Toda lattice where $u_{xt}(n)$ replaces $u_{tt}(n)$, and the discrete version of the KdV equation, $u_t(n) + u(n)[u(n-1) - u(n+1)] = 0$.

5.2 Algorithm and Implementation

The method used in the algorithm or our program SYMMGRP.MAX [3], is based on prolonged vector fields described in [19], from which we also adopted the notations and terminology.

For notational simplicity, let us consider the case of Lie-point symmetries. We start with a system of m differential equations,

$$\Delta^i(x, u^{(k)}) = 0, \quad i = 1, 2, ..., m, \tag{61}$$

of order k, with p independent and q dependent (real) variables, denoted by $x = (x_1, x_2, ..., x_p) \in \mathbb{R}^p$, $u = (u^1, u^2, ..., u^q) \in \mathbb{R}^q$. We stress that m, k, p and q are *arbitrary* positive integers. The partial derivatives of u^l are represented using a multi-index notation, for $J = (j_1, j_2, ..., j_p) \in \mathbb{N}^p$, we denote $u^l_J \equiv \dfrac{\partial^{|J|} u^l}{\partial {x_1}^{j_1} \partial {x_2}^{j_2} ... \partial {x_p}^{j_p}}$, where $|J| = j_1 + j_2 + ... + j_p$. Finally, let $u^{(k)}$ denote a vector whose components are all the partial derivatives of order 0 up to k of all the u^l.

The group transformations have the form $\tilde{x} = \Lambda_g(x, u)$, $\tilde{u} = \Omega_g(x, u)$, where the functions Λ_g and Ω_g are to be determined. Note that the subscript g refers to the group parameters. Instead of looking for a Lie group G, we look for its Lie algebra $\mathcal{L}$, realized by vector fields of the form

$$\alpha = \sum_{i=1}^{p} \eta^i(x, u) \frac{\partial}{\partial x_i} + \sum_{l=1}^{q} \varphi_l(x, u) \frac{\partial}{\partial u^l}. \tag{62}$$

The problem is now reduced to finding the coefficients $\eta^i(x, u)$ and $\varphi_l(x, u)$. In essence, the computer constructs the k^{th} prolongation $\text{pr}^{(k)}\alpha$ of the vector field α, applies it to the system (61), and requests that the resulting expression vanishes on the solution set of (61). Although that sounds straightforward, the method involves tedious calculations because the length and complexity of the expressions increase rapidly as p, q, m, and especially k, increase. Here are the details and the steps to be performed:

1. Construct the k^{th} prolongation of the vector field α in (62) by means of the formula

$$\text{pr}^{(k)}\alpha = \alpha + \sum_{l=1}^{q} \sum_{J} \psi_l^J(x, u^{(k)}) \frac{\partial}{\partial u^l_J}, \quad 1 \leq |J| \leq k, \tag{63}$$

where the coefficients ψ_l^J are defined as follows. The coefficients of the first prolongation are

$$\psi_l^{J_i} = D_i \varphi_l(x,u) - \sum_{j=1}^{p} u_{J_j}^l D_i \eta^j(x,u), \tag{64}$$

where J_i is a $p-$tuple with 1 on the i^{th} position and zeros elsewhere, and D_i is the total derivative operator

$$D_i = \frac{\partial}{\partial x_i} + \sum_{l=1}^{q} \sum_{J} u_{J+J_i}^l \frac{\partial}{\partial u_J^l}, \quad 0 \leq |J| \leq k. \tag{65}$$

The higher-order prolongations are defined recursively as

$$\psi_l^{J+J_i} = D_i \psi_l^J - \sum_{j=1}^{p} u_{J+J_j}^l D_i \eta^j(x,u), \quad |J| \geq 1. \tag{66}$$

2. Apply the prolonged operator $\mathrm{pr}^{(k)}\alpha$ to each equation $\Delta^i(x, u^{(k)})$ and require that

$$\mathrm{pr}^{(k)}\alpha \ \Delta^i \ |_{\Delta^j=0} \ = 0 \quad i,j = 1,...,m. \tag{67}$$

The condition (67) expresses that $\mathrm{pr}^{(k)}\alpha$ vanishes on the solution set of the originally given system (61). Precisely, this condition assures that α is an infinitesimal symmetry generator of the group transformation; $\tilde{x} = \Lambda_g(x,u)$, $\tilde{u} = \Omega_g(x,u)$, i.e., that $u(x)$ is a solution of (61) whenever $\tilde{u}(\tilde{x})$ is one.

3. Choose, if possible, m components of the vector $u^{(k)}$, say $v^1, ..., v^m$, such that:

   (a) Each v^i is equal to a derivative of a u^l $(l = 1, ..., q)$ with respect to at least one variable x_i $(i = 1, ..., p)$.

   (b) None of the v^i is the derivative of another one in the set.

   (c) The system (61) can be solved algebraically for the v^i in terms of the remaining components of $u^{(k)}$, which we denote by w. Hence, $v^i = S^i(x,w)$, $i = 1, ..., m$.

   (d) The derivatives of v^i, $v_J^i = D_J S^i(x,w)$, where $D_J \equiv D_1^{j_1} D_2^{j_2} ... D_p^{j_p}$, can all be expressed in terms of the components of w and their derivatives, without ever reintroducing the v^i or their derivatives.

The requirements in step 3 put some restrictions on the system (61), but for many systems the choice of the appropriate v^i is quite obvious. For example, for a system of evolution equations

$$\frac{\partial u^i}{\partial t}(x_1, ..., x_{p-1}, t) = F^i(x_1, ..., x_{p-1}, t, u^{(k)}), \quad i = 1, ..., m, \tag{68}$$

where $u^{(k)}$ involves derivatives with respect to the variables x_i but not t, an appropriate choice is $v^i = \frac{\partial u^i}{\partial t}$.

4. Use $v^i = S^i(x, w)$ to eliminate all v^i and their derivatives from the expression (67), so that all the remaining variables are now independent of each other. It is tacitly assumed that the resulting expression is now a polynomial in the u^l_J.

5. Obtain the determining equations for $\eta^i(x, u)$ and $\varphi_l(x, u)$ by equating to zero the coefficients of all functionally independent expressions (monomials) in the remaining derivatives u^l_J.

In the above algorithm the variables x_i, u^l, and u^l_J are treated as independent; the dependent ones are η^i and φ_l.

In summary, the result of implementing (63) is a system of linear homogeneous PDEs for η^i and φ_l, in which x and u are independent variables. These are the so-called determining or defining equations for the symmetries of the system. Solving these by hand, interactively or automatically with a symbolic package, will give the explicit forms of the $\eta^i(x, u)$ and $\varphi_l(x, u)$.

The procedure, which is thoroughly discussed in [19], consists of two major steps: *deriving* the determining equations, and *solving* them explicitly. Details on how this can be done by computer can be found in [5, 6, 7]. In these papers we also presented a survey of available codes, including a discussion of their strengths and weaknesses.

5.3 Examples

- Consider the Dym-Kruskal equation [1]

$$u_t - u^3 u_{xxx} = 0. \tag{69}$$

The program SYMMGRP.MAX automatically computes the determining equations for the coefficients eta[1] $= \eta^x$, eta[2] $= \eta^t$ and phi[1] $= \varphi^u$ of the vector field

$$\alpha = \eta^x \frac{\partial}{\partial x} + \eta^t \frac{\partial}{\partial t} + \varphi^u \frac{\partial}{\partial u}. \tag{70}$$

There are only eight determining equations,

$$\frac{\partial \text{eta2}}{\partial u[1]} = 0, \quad \frac{\partial \text{eta2}}{\partial x[1]} = 0, \quad \frac{\partial \text{eta1}}{\partial u[1]} = 0, \quad \frac{\partial^2 \text{phi1}}{\partial u[1]^2} = 0, \tag{71}$$

$$\frac{\partial^2 \text{phi1}}{\partial u[1] \partial x[1]} - \frac{\partial^2 \text{eta1}}{\partial x[1]^2} = 0, \quad \frac{\partial \text{phi1}}{\partial x[2]} - u[1]^3 \frac{\partial^3 \text{phi1}}{\partial x[1]^3} = 0, \tag{72}$$

$$\begin{aligned} 3u[1]^3 \frac{\partial^3 \text{phi1}}{\partial u[1] \partial x[1]^2} + \frac{\partial \text{eta1}}{\partial x[2]} - u[1]^3 \frac{\partial^3 \text{eta1}}{\partial x[1]^3} &= 0, \\ u[1] \frac{\partial \text{eta2}}{\partial x[2]} - 3u[1] \frac{\partial \text{eta1}}{\partial x[1]} + 3 \,\text{phi1} &= 0. \end{aligned} \tag{73}$$

These determining equations are easily solved explicitly with the feedback mechanism within SYMMGRP.MAX. The general solution, rewritten in the original variables, is

$$\eta^x = k_1 + k_3\, x + k_5\, x^2\,, \tag{74}$$
$$\eta^t = k_2 - 3k_4\, t\,, \tag{75}$$
$$\varphi^u = (k_3 + k_4 + 2k_5\, x)\, u\,, \tag{76}$$

where $k_1, ..., k_5$ are arbitrary constants. The five infinitesimal generators then are

$$\begin{array}{lll} G_1 = \partial_x\,, & & G_2 = \partial_t\,, \\ G_3 = x\partial_x + u\partial_u\,, & & G_4 = -3t\partial_t + u\partial_u\,, \\ G_5 = x^2\partial_x + 2xu\partial_u\,. & & \end{array} \tag{77}$$

Clearly, (69) is invariant under translations (G_1 and G_2) and scaling (G_3 and G_4). Computation of the flow corresponding to G_5 requires integration of the system

$$\begin{array}{lllll} \frac{d\tilde{x}}{d\epsilon} = \tilde{x}^2, & & \tilde{x}(0) & = & x, \\ \frac{d\tilde{t}}{d\epsilon} = 0, & & \tilde{t}(0) & = & t, \\ \frac{d\tilde{u}}{d\epsilon} = 2\tilde{x}\tilde{u}, & & \tilde{u}(0) & = & u, \end{array} \tag{78}$$

where ϵ is the parameter of the transformation group.

One readily obtains $\tilde{x}(\epsilon) = x/(1-\epsilon x)$, $\tilde{t}(\epsilon) = t$, and $\tilde{u}(\epsilon) = u/(1-\epsilon x)^2$. We therefore conclude that for any solution $u = f(x,t)$ of equation (69) the transformed solution

$$\tilde{u}(\tilde{x},\tilde{t}) = (1+\epsilon\tilde{x})^2 f(\frac{\tilde{x}}{1+\epsilon\tilde{x}}, \tilde{t}) \tag{79}$$

will solve $\tilde{u}_{\tilde{t}} - \tilde{u}^3\tilde{u}_{\tilde{x}\tilde{x}\tilde{x}} = 0$.

- As a second example, we take the nonlinear Schrödinger (NLS) equation [1],

$$iu_t + u_{xx} + u|u|^2 = 0. \tag{80}$$

The complex equation can be replaced by a coupled system,

$$\begin{array}{lll} v_t + w_{xx} + w(v^2+w^2) & = & 0, \\ w_t - v_{xx} - v(v^2+w^2) & = & 0. \end{array} \tag{81}$$

for the real and imaginary parts v, w of the complex variable u.

Now, SYMMGRP.MAX quickly generates the twenty determining equations for the coefficients of the vector field

$$\alpha = \eta^x \frac{\partial}{\partial x} + \eta^t \frac{\partial}{\partial t} + \varphi^v \frac{\partial}{\partial v} + \varphi^w \frac{\partial}{\partial w}. \tag{82}$$

The first eleven single-term determining equations are similar to (71), and provide information about the dependencies of the $\eta's$ and the $\phi's$ on x, t, v and w, and their linearity in the latter two independent variables. The remaining nine determining equations are more complicated, but the entire system is readily solved.

In the original variables, the solution reads

$$\eta^x = k_1 + 2k_4\, t + k_5\, x\,, \tag{83}$$

$$\eta^t = k_2 + 2k_5\, t\,, \tag{84}$$

$$\varphi^v = k_3\, w - k_4\, xw - k_5\, v\,, \tag{85}$$

$$\varphi^w = -k_3\, v + k_4\, xv - k_5\, w\,, \tag{86}$$

where $k_1, ..., k_5$ are arbitrary constants. As in the previous examples, the complete symmetry algebra is spanned by five vector fields (generators):

$$\begin{array}{rcll} G_1 & = & \partial_x\,, & G_2 = \partial_t\,, \\ G_3 & = & w\partial_v - v\partial_w\,, & G_4 = 2t\partial_x - x(w\partial_v - v\partial_w)\,, \\ G_5 & = & x\partial_x + 2t\partial_t - v\partial_v - w\partial_w\,. & \end{array} \tag{87}$$

Clearly, (80) is invariant under translations in space and time (G_1 and G_2). Generator G_3 corresponds to adding an arbitrary constant to the phase of u. The Galilean boost is generated by G_4. Finally, G_5 indicates invariance of the equation under scaling (or dilation).

References

[1] M.J. Ablowitz and P.A. Clarkson, *Solitons, Nonlinear Evolution Equations and Inverse Scattering*, London Mathematical Society Lecture Notes on Mathematics **149** (Cambridge University Press, Cambridge, UK, 1991).

[2] M.J. Ablowitz and H. Segur, *Solitons and the Inverse Scattering*, *SIAM Studies in Appl. Math.* **4** (SIAM, Philadelphia, 1981).

[3] B. Champagne, W. Hereman and P. Winternitz, *The Computer Calculation of Lie Point Symmetries of Large Systems of Differential Equations*, Comp. Phys. Comm. **66** (1991) 319-340.

[4] P.G. Drazin and R.S. Johnson, *Solitons: An Introduction* (Cambridge University Press, Cambridge, 1989).

[5] W. Hereman, *SYMMGRP.MAX and Other Symbolic Programs for Symmetry Analysis of Partial Differential Equations*, in: *"Exploiting Symmetry in Applied and Numerical Analysis"*, Lectures in Applied Mathematics **29**, E. Allgower, K. Georg and R. Miranda, eds. (American Mathematical Society, Providence, Rhode Island, 1993) 241-257.

[6] W. Hereman, *Review of Symbolic Software for the Computation of Lie Symmetries of Differential Equations*, Euromath Bulletin **1**, no. 2 (1994) 45-82.

[7] W. Hereman, *Symbolic Software for Lie Symmetry Analysis*, in: *"New Trends in Theoretical Developments and Computational Methods"*, CRC Handbook of Lie Group Analysis of Differential Equations, Vol. 3, N.H. Ibragimov, ed. (CRC Press, Boca Raton, Florida, 1995) Chapter 13, in press.

[8] W. Hereman and S. Angenent, *The Painlevé Test for Nonlinear Ordinary and Partial Differential Equations*, MACSYMA Newsletter **6** (1989) 11-18.

[9] W. Hereman and W. Zhuang, *Symbolic Computation of Solitons with Macsyma*, in *"Computational and Applied Mathematics II: Differential Equations"*, W.F. Ames and P.J. van der Houwen, eds. (North Holland, Amsterdam, The Netherlands, 1992) 287-296.

[10] J. Hietarinta, *A Search for Bilinear Equations Passing Hirota's Three-Soliton Condition, Part I. KdV-Type Bilinear Equations*, J. Math. Phys. **28** (1987) 1732-1742.

[11] R. Hirota, *Direct Method of Finding Exact Solutions of Nonlinear Evolution Equations*, in: *"Bäcklund Transformations, the Inverse Scattering Method, Solitons, and Their Applications"*, Lecture Notes in Mathematics **515**, R.M. Miura, ed. (Springer-Verlag, Berlin, 1976) 40-68.

[12] R. Hirota, *Direct Methods in Soliton Theory*, in: *"Solitons"*, Topics in Physics **17**, R.K. Bullough and P.J. Caudrey, eds. (Springer-Verlag, Berlin, 1980) 157-176.

[13] M.D. Kruskal, R.M. Miura, C.S. Gardner and N.J. Zabusky, *Korteweg-de Vries Equations and Generalizations. V. Uniqueness and Nonexistence of Polynomial Conservation Laws*, J. Math. Phys. **11** (1970) 952-960.

[14] Y. Matsuno, *Bilinear Transformation Method* (Academic Press, Orlando, 1984).

[15] R.M. Miura, C.S. Gardner and M.D. Kruskal, *Korteweg-de Vries Equations and Generalizations. II. Existence of Conservation Laws and Constants of Motion*, J. Math. Phys. **9** (1968) 1204-1209.

[16] J.D. Murray, *Mathematical Biology*, Biomathematics Texts **19** (Springer Verlag, Berlin, 1989).

[17] A.C. Newell, *Solitons in Mathematics and Physics* (SIAM, Philadelphia, 1985).

[18] A.C. Newell and Y.B. Zeng, *The Hirota Conditions*, J. Math. Phys. **29** (1987) 2016-2021.

[19] P.J. Olver, *Applications of Lie Groups to Differential Equations*, Graduate Texts in Mathematics **107** (Springer Verlag, New York, 1986 & 1993, 2nd edition).

[20] W.-H. Steeb and N. Euler, *Nonlinear Evolution Equations and Painlevé Test* (World Scientific, Singapore, 1988).

[21] F. Verheest and W. Hereman, *Conservation Laws and Solitary Wave Solutions for Generalized Schamel Equations*, Physica Scripta (1994) in press.

[22] R. Willox, W. Hereman and F. Verheest, *Complete Integrability of a Modified Vector Derivative Nonlinear Schrödinger Equation*, Physica D (1994) submitted.