Differential Geometric Computations and Computer Algebra

P. K. H. GRAGERT AND P. H. M. KERSTEN
Department of Applied Mathematics, University of Twente
P.O. Box 217, 7500 AE Enschede, The Netherlands
<P.K.H.Gragert><P.H.M.Kersten>@math.utwente.nl

Abstract—The use of computer algebra in the field of differential geometry and its applications to geometric structures of partial differential equations is discussed. The differential geometric setting is shortly described; a number of programs are slightly touched, some examples given, and an application to the construction of supersymmetric extensions of the Korteweg-de Vries equation is demonstrated.

Keywords—Computer algebra, Differential geometry, Literate programming, Supersymmetry.

1. PLAN OF THE ARTICLE

We start in Section 2 with a historical review, concerning activities of the authors in the field of 'symmetries.' In Section 3, we give an outline of the geometrical algebraic setting, the mathematical background of our computational activities.

Some emphasis is laid on the use of computers in the field of analyzing nonlinear partial differential equations followed by some ideas guiding our interests in this using. This is exemplified next by an example of 'literate programming,' one of the essential activities in using 'symbolic algebra systems,' which is done in Section 5 after giving our small philosophy in computations in Section 4.

Section 6 is about the programs used by the authors. In two of its subsections we give some small but concrete examples of our programming packages for REDUCE 3.5. Some constructive ideas and latest results concerning an n = 2 superextension of the Korteweg-de Vries equation are given in Section 7.

2. HISTORICAL DEVELOPMENT

The activities of our research in the area of symmetry structures started with the classical papers by Estabrook and Wahlquist [1] and Harrison and Estabrook [2]. In the latter, the authors use the nomenclature isovector where they were discussing symmetries. All considerations in the above-mentioned paper were based on exterior differential systems, i.e., ideals of differential forms closed under exterior differentiation.

In terms of an exterior differential system I, which describes (solution structures of) a differential equation, or a system of differential equations in $\mathbb{R}^n$ for some $n, x \in \mathbb{R}^n, x = (x_1, \ldots, x_n)$, the prolongation of an equation is the existence of a 1-form

$$w = dy - X \ dx_1 - T \ dx_2,$$

where $X, T$ are functions defined on $\mathbb{R}^{n+1} = \{(x_1, \ldots, x_n, y)\}$ such that

$$dw \in \langle I, w \rangle;$$

i.e., the ideal $I' = \langle I, w \rangle$ is again closed under exterior differentiation.
An infinitesimal symmetry is defined as a vector-field defined on $\mathbb{R}^n$ such that

$$\mathcal{L}_V I \subset I,$$

where $\mathcal{L}_V$ is the Lie derivative by the vector-field $V$.

In principle, these two conditions (2.2), (2.3) of differential geometric type lead to overdetermined systems of partial differential equations for the coefficients $X, T$ and the components of the vector-field $V$. All the computations described were repeated with pencil and paper. Looking at other examples made it clear that some 'computer help' would be desirable, because the computations involved were more or less algorithmic and rather lengthy. This experience led to the first steps in using the computer algebra system REDUCE.

Now in order to study (2.2),(2.3), one first needs a differential geometry package to derive the partial differential equations which result from these conditions. Further software is needed to study these systems and derive solutions.

The programs allowed (in a local context) to do differential geometric computations: computations on differential forms and vector-fields, such as exterior differentiation, inner products, Lie derivations, and so on. In addition to some tools to get and solve an overdetermined set of partial differential equations and to analyze 'Lie relations,' it means tools to do computations in 'prolongation algebras.'

Nowadays a number of software packages are available to this end and the reader is referred to the overview by Hereman [3] for more details and specifics of each of these packages. Later on (mid eighties), these tools were extended to do analogous computations suitable for the 'super' case. Roughly speaking, this means introduction of 'odd variables.'

Due to influences of Russian mathematicians Vinogradov and Krasil'shchik [4,5], our point of view towards symmetries, conservation laws, prolongation structures, and other geometrical objects gradually changed towards a vector-field approach, the features of which shall be described compactly in Section 3.

In the early nineties, all the essential tools were optimized and extended by 'total derivative operators,' because of application of the theory of 'coverings' (Section 3). A very important remark at this point is that all this 'upgrading' is done in a literate programming style and all the programs are thus developed as 'readable documents.' To be honest, one thing is lacking: user manuals. This lack is felt more and more as a serious omission, and as a consequence, just this year first steps are taken to fill in this gap.

Latest interest has turned to 'coloured' extensions of 'classical' partial differential equations. In our opinion, it is inevitable to use the computer to do those calculations there. In this context, the (first) essential computations were done using Mathematica, because the programming was rather close to mathematics. After initial intermediate results, the separation of a result into a system of equations became too time consuming in Mathematica. It is not totally clear if this explosion of computation-time needed is inherent to Mathematica or to a lack of our knowledge of the deeper ins and outs of Mathematica. Therefore, results of Mathematica were adjusted to the notation of REDUCE (by using Emacs) to produce 'the equations' and to solve them. The efficiency difference between Mathematica and REDUCE to find the coefficients of a big expression (about one megabyte of input) was larger than a factor of 2000.

This finishes a rough description of activities of the authors, and we continue by giving some more details.

### 3. GEOMETRICAL ALGEBRAIC SETTING

The starting point is the infinite jet-bundle $J^\infty(x; u)$, where local coordinates are given by the independent variables $x = (x_1, \ldots, x_n)$, the dependent variables are $u = (u^1, \ldots, u^m)$. Partial derivatives are denoted by $u^\sigma_\alpha$ with a multi-index $\sigma = (\sigma_1, \ldots, \sigma_n), \alpha = 1, \ldots, m$. The total
Differential Geometric Computations

partial derivative operators $D_i$ are given by

$$D_i = \partial_{x_i} + \sum_{\sigma, |\sigma| \geq 0} u_{i+1}^\sigma \partial_{u_\sigma},$$

(3.1)

where $\sigma + i = (\sigma_1, \ldots, \sigma_i + 1, \ldots, \sigma_n)$. These vector-fields commute, i.e.,

$$[D_i, D_j] = 0, \quad (i, j = 1, \ldots, n)$$

(3.2)

where the bracket denotes the Lie bracket of vector-fields on this formally infinite dimensional manifold.

All algebraic/geometric objects are defined in such a way that there are no convergence problems. In classical terms, a partial differential equation is an equation (or system of equations) in $x$ and partial derivatives of an unknown function $u(x)$.

In the jet-bundle formulation, this can be described by an equation $E$,

$$F(x; u) = 0,$$

(3.3)

where $F : J^\infty(x; u) \rightarrow \mathbb{R}$ and $F = (F^1, \ldots, F^l)$ depends on a finite number of variables. The classical partial differentiation of an equation is realized through the infinite prolongation $E^\infty$ of an $E$ which is defined by the system of equations

$$D^\sigma F(x; u) = 0,$$

(3.4)

where $\tau = (\tau_1, \ldots, \tau_n)$ and $D^\tau = D_1^\tau \circ D_2^\tau \cdots D_n^\tau$, $\tau \geq 0$.

A symmetry (actually, an infinitesimal symmetry) is a vector-field defined on $J^\infty(x; u)$ leaving invariant the contact-structure on $J^\infty(x; u)$, (3.1), (3.2) and leaving invariant (3.4).

Since every vector-field of the form

$$\sum_{i=1}^n \xi_i(x; u) D_i$$

(3.5)

satisfies this invariance condition, one can restrict the discussion of symmetries to so-called vertical vector-fields, which turn out to be of the form

$$\exists f \sum_{\sigma, |\sigma| \geq 0} D^{\sigma}(f^\sigma) \partial_{u_\sigma},$$

(3.6)

where $f^\sigma$ are functions defined on $J^\infty(x; u)$, depending on a finite number of variables, denoted $f^\sigma \in C^\infty(J^\infty(x; u))$.

The invariance conditions result in the following equation:

$$l_F(f) = 0,$$

(3.7)

where $l_F$ is the so-called universal linearization of $F$ [4], or Frechet derivative [6], which is given by

$$l_F = |l_F|^\alpha = \left[ \frac{\partial F^\gamma}{\partial u_\sigma} D^\sigma \right] (\alpha = 1, \ldots, m, \gamma = 1, \ldots, l).$$

(3.8)

Equation (3.7) is just the well-known symmetry condition. In effect, doing real computations one has to choose internal coordinates on $E^\infty$ and restrict $D^\sigma$ to the manifold $E^\infty$, but for simplicity reasons we shall not pursue this here.

The local picture of a covering can be described as follows. Start at the prolongation of a differential equation $E^\infty$ in $J^\infty(x; u)$. An $m$-dimensional covering of $E^\infty$ requires the introduction
of $J^\infty(x;u) \otimes \mathbb{R}^s$ with local coordinates $x^i, u_{\sigma}^a, w^1, \ldots, w^s$, and functions $X^r_i$ such that the vector-fields
\[ \tilde{D}_i = D_i + \sum_{r=1}^{s} X^r_i (x;u,w) \partial_{w_r} \] (3.9)
commute, i.e.,
\[ [D_i, D_j] = 0, \] (3.10)
which amounts to the lifting of the total derivative vector-fields $D_i$.

The so-called covering condition (3.10) can be written as
\[ D_i(X_j) - D_j(X_i) = [X_i, X_j] = 0, \] (3.11)
whereas in (3.11)
\[ X_i = \sum_{r=1}^{s} X^r_i \partial_{w_r} \]
are in a sense vector-fields vertical to $J^\infty(x;u)$. The commutator in (3.11) is to be taken with respect to the variables $w_r$ ($r = 1, \ldots, s$).

Both (3.7), (3.11) are the vector-field analogues of the classical conditions (2.2), (2.3). It should be noted here that in this formulation, one circumvents differential form computations completely and one only requires (total) derivative vector-fields and $X_i$.

Special types of covering arise in, for instance, the theory of integrable (two-dimensional) systems when one requires $X_i$ ($i = 1, 2$) to be independent of $w$ and condition (3.11) results in a conservation law.

By the notion of covering given above one arrives at a system of equations, called the covering-equation of the original $\mathcal{E}^\infty$, and the notion of symmetry of the covering equation can be defined, generalizing (3.6), leading to the so-called nonlocal symmetries of $\mathcal{E}^\infty$ where the condition (3.7) changes into
\[ Mf) = 0, \] (3.12)
where now $f^\infty \in C^\infty (J^\infty(x;u) \otimes \mathbb{R}^s)$ and
\[ \tilde{I}_f = \left[ I_f \right]^a_{\gamma} = \left[ \frac{\partial F^\gamma}{\partial u^a} \right]. \] (3.13)

In effect, (3.12) just gives the conditions for the shadow of a nonlocal symmetry, and we discard at this place the so-called reconstruction problem [7]; i.e., we are not thinking of the $\partial_{w_r}$-components of the symmetry.

We shall now touch very lightly a very interesting further generalization of covering and symmetry, the deeper geometric structures of which have been recently published [8-10], and which embed the theory of recursion operators in a theory of symmetries.

Let us start in $\mathcal{E}^\infty \subset J^\infty(x;u)$ and introduce the contact forms
\[ z^a_{\sigma} = du^a_{\sigma} + u^a_{\sigma+1} \, dx^i. \] (3.14)
The action of the total derivative operators $D_i$ (in effect $D_i$, the restriction of $D_i$ to the manifold $\mathcal{E}^\infty$) on $z^a_{\sigma}$ can be computed in a standard way leading to
\[ (z^a_{\sigma})_i = D_i (z^a_{\sigma}). \] (3.15)

If we now define
\[ \tilde{D}_i = D_i + (z^a_{\sigma})_i \partial_{z^a_{\sigma}}, \] (3.16)
then it is easy to derive that
\[ [\tilde{D}_i, \tilde{D}_j] = 0 \] (3.17)
and we arrive at a covering of our original equation $E^\infty$ by the contact forms $z^\alpha$, which is called the Cartan covering of $E^\infty$. This is in effect an infinite dimensional covering. Note that due to (3.15) and
\[
\zeta^\beta \wedge \zeta^\gamma = -\zeta^\gamma \wedge \zeta^\beta,
\]
we have a graded covering.

The notion of a graded, nonlocal symmetry can be defined now in a combination of the two previously described types of covering. Now the symmetry conditions read analogously to (3.12), (3.13) and (3.7), (3.8) but $D_i$ or $D_i$ replaced by $D_i$. Symmetries of this covering where $f^{\alpha}$ are linear in $z^\alpha$ represent the recursion operators for symmetries.

We stress at this point that classical, nonlocal, graded symmetries satisfy similar conditions and lead in general to overdetermined systems of partial differential equations for the components of the vector-field. So one can apply the computer programs to solve these equations.

4. SMALL 'PHILOSOPHY'

We are giving some thoughts about our situation: the classical theory of local symmetries is extended in several ways and we want to apply the new theory to interesting equations.

Applying theory means more or less that one has to do some 'computations' in some sense. Our philosophy under this circumstance may be described as follows: if one has to do the same sort of computations with respect to 'different inputs,' then try to use a computer. Or if the computations are lengthy and algorithmical or ..., try to use a computer!

One is just in such a situation being interested in nonnumerical analysis of systems of partial differential equations. One may be interested in symmetries, higher order symmetries, conserved quantities, recursion operators and the like. The theory is then extended to the 'super' case or even further generalized to the 'coloured' case and so on.

For instance, the equation $\lambda^\dagger \Delta = 0$ is on a rather abstract level the condition of a (infinitesimal) symmetry $V$ for a system of differential equations $\Delta$. It means $\Delta$ is given, find a $V$ such that the aforementioned equations 'hold.'

In practice, this problem may be handled as follows: make an 'Ansatz for $V,$' compute the conditions on $V$, which easily becomes a big (several hundreds) overdetermined system of (partial) differential equations, which furthermore has to be solved exactly to get the desired quantity $V$.

An interesting problem is now to generalize $\Delta$ to the 'super' or even 'coloured' case, needing rather lengthy computations. We go into some detail about this problem in Section 5. Furthermore, all questions (and new ones) have to be asked again about the generalized $\Delta$, giving rise to even more bulky computations just indicated.

Another part of this philosophy is: do not provide the computer algebra system with information that is not required for the next step in the calculation, because this information can give rise to expression swell and slows down the computation process. A good example is the use of total derivative operators (Section 6).

Because no 'algebra system' gives you all you need to solve your problems, some programming is necessary. It is worthwhile to adopt a 'literate programming style' described in the next section.

5. NOWEB PROGRAMMING EXAMPLE

From 'information science' we are told to program in a 'structured' way, whatever this may mean. One possibility to obey this command is using a 'WEB' in a way it is meant. Here we give an example of our programming efforts using 'NOWEB' with respect to the 'coloured calculus.'

In the 'coloured calculus' you need products of terms (elements of an infinite jet), where interchanging the order of two adjacent terms must be compensated by a (term dependent) factor of $q$, $q$ an undetermined parameter. You want that different looking terms may cancel or
be added automatically if possible. Therefore you need 'noncommuting objects,' which you have
to bring into a canonical order such that automatic 'simplification' may take place.

Using a 'WEB' means that you develop at the same time a document of your programming
efforts as well as the program itself.

There are different 'WEBs,' which take account of the needs with respect to typesetting of
program-text of the programming language for which they are meant, such as PASCAL, C,
FORTRAN, REDUCE, etc. Two programs makeup a 'WEB' system: one, which prepares out of
your input file an input for \textsc{TeX} or \textsc{LATEX}, and a second, which builds out of the same input file
an input file for the 'programming language.'

'NOWEB' is a 'WEB' which is (more or less) suitable for any computer language. A (NO)WEB
input file is built out of a number of parts (called chunks). A chunk starts with a special mark
(often a ' '), followed by normal text, which should be understood by a (I)\textsc{TEX}-compiler. In
such parts you can use through (I)\textsc{TEX} the full power of mathematical notation to describe, if
meaningful, the mathematical background. It is also a place to write down your considerations
with respect to your way of implementing a program, pre- and postconditions, and the like. If
you do not do this (you may!), you are committing a 'sin.'

Depending on the WEB being used, you may have in a chunk other optional parts. For
NOWEB, an optional part is the definition of a 'module,' by which top down design is made
possible. Here follows a small part out of a NOWEB file for 'coloured action.

Note: ... means: omitted here.

0 We need a total ordering function and start
with the basic order of two objects.

A 'totalOrderList' contains all function-names.
Two identical objects are in good order.

Two function-names are ordered with respect to their position on
the 'totalOrderList'. Note: A wrong spelt function-name gives
a 'recursion-depth runtime error'!

The last pattern uses for identical function-names the order
of derivatives as ordering principle, (higher order first),
otherwise the function-names decides the order relation.

<<totalorder.1>>=
totalOrderList={chilg2,p11,p12,p31,p32,teta2,teta1,
phi,phi1,phi2,phi12,u,w,ksi1,ksi2}

MyOrderQ[x_,x_] := True
MyOrderQ[x_,y_] := And[MemberQ[totalOrderList,x],
MemberQ[totalOrderList,y],
Position[totalOrderList,x] [[1]] [[1]] <
Position[totalOrderList,y] [[1]] [[1]]]
MyOrderQ[x_{{x1___}},y_{{y1___}}] :=
If[x==y,Length[{{x1}}]==Length[{{y1}}],MyOrderQ[x,y]]

0 Now we define the total ordering function to be used later in
the definition of NC, the non-commuting object making function.
\begin{itemize}
\item One term is always ordered, first pattern.
\item Two terms are either ordered or not using ‘MyOrderQ’.
\item More than two terms are decided by recursion, an AND with 
respect to the two first arguments and the total order of all but 
the first argument. Pay attention to the ordering of commands ‘in’ the 'And', it is 'greedy'!!
\end{itemize}

<<totalorder.2>>=

\texttt{totMyOrderQ[x_]:=True}
\texttt{totMyOrderQ[x_,y_]:=MyOrderQ[x,y]}
\texttt{totMyOrderQ[x_,y_,z_]:=And[MyOrderQ[x,y],totMyOrderQ[y,z]]}

...  

...  

We put things together in the output file. The last string 
gives the message, that the whole file has been read.

<<**>>=

<<totalorder.1>>

<<totalorder.2>>

"coloured action loaded"

Assume that the above text is the content of a file ‘coloured_action.nw.’ Now you have two 
options:

\texttt{noweave coloured_action.nw > coloured_action.tex}
\texttt{notangle coloured_action.nw > coloured_action.math}

The first command prepares your input by producing a \LaTeX\ file to finally get a document of 
your programming considerations, whereas the second command delivers a file suitable to load 
into a Mathematica session to use the functions built.

The following WWW addresses are suited even better to inform oneself about ‘literate pro-
gramming’ (addresses valid on July 4, 1995):

\begin{itemize}
\item \url{http://info.desy.de/user/projects/LitProg/Tools.html}
\item \url{http://jasper.ora.com/ctan.html}
\end{itemize}

6. SOFTWARE PACKAGES

We will give here a rough description of some software packages developed and already men-
tioned and used by the authors. Documentation as well as programs are available by ‘anonymous 
ftp’ (RWEB, REDUCE 3.5!). Addresses:

WWW: \url{http://www.math.utwente.nl}
FTP: \url{ftp.math.utwente.nl directory: /pub/rweb/*}

A very basic package is the ‘tools’ package, described with some small examples in the next 
section. A package to do specialized calculations in super Lie algebras follows together with a 
very small (trivial) example of automatic Jacobi identity analysis: a parameter has to be adjusted 
to get a Lie algebra. A package to aid solving overdetermined systems of partial differential 
equations is described already in [11]. All this software is meant to run under REDUCE 3.5.

The next two sections give some small examples to give the reader a first feeling of what it is 
all about.
6.1. Tools

The concept of algebraic operator in REDUCE is rather powerful though one concept is not yet implemented in the original distribution or REDUCE 3.5, namely, multilinear operators. This is done in our ‘tools package’ [12].

```
input: multilinear m((a,b));
input: m(a(3)-3*b(6),a(x)+y*b(z));
output: m(a(3),a(x)) + m(a(3),b(z))*y - 3*m(b(6),a(x)) - 3*m(b(6),b(z))*y
input: m(c(3)+a);  
output: m(t)*(c(3) + a)
```

In this example, ‘m’ is declared to be a multilinear algebraic operator with respect to the algebraic operators ‘a’ and ‘b,’ and the two input and output lines give you a good impression of the effect of ‘multilinear.’ For insiders, you may give as a second argument a from ‘simpiden’ different resimplification function.

A convenient procedure is ‘operator_coeff’ to split an algebraic expression of algebraic operators into a list, containing the ‘rest’ and a lists of kernels with its coefficients. We use the ‘multilinear’ operator ‘m’ from the example above.

```
input: operator_coeff(666+m(3*a(4)+z*b(6),m);
output: C666,Cm(b(6>>,z),(m(a(4>),333
```

The second parameter may be an algebraic list of several algebraic operators, sometimes more appropriate. The implementation of the just described procedures also led to an extension of the original ‘COEFF’ procedure:

```
input: multi_coeff (44+(a+b)^2,<a,b3);
output: C44,Cb**2,i),\(b*a,23,<a**2,133
```

REDUCE offers several functions to ‘solve equations,’ nevertheless some specialized procedures to solve (and assign) for operators occurring linearly in an equation are contained too in our ‘tools’ package.

6.2. LIESUPER Package

This package is meant for the analysis of prolongation structures of PDV’s; see [1]. This package implements Lie brackets as a ‘new data-type’ in REDUCE and thus enables symbolic computations in (free) Lie (super) algebras. Further features include automated checking and evaluation of Jacobi identities, basis transformations of Lie (super) algebras, and the possibility to define an integer valued multigrading on the algebra. This last mentioned feature proved to be essential in solving different sorts of problems. This package was used too to solve ‘presentation problems of Lie (super) algebras’; see [13].

We include here a very short example of automatic Jacobi identity analysis.

```
input: load LIESUPER;
input: liebracket lie(x,3,0,nil,a);
meaning: lie: ‘lie’ will be a newly define Lie bracket
  3: basis elements will be x(1), x(2) and x(3)
  0: no odd basis elements
  nil: no extra Lie algebra elements
  a: operator name of parameters
input: [1,2] := x * a(1)*x 1; [2,3] := x 1; lie(x(3),x(1)) := x 2;
meaning: give three commutators. Notice the short notation as
```
well as the long notation.

**input:** `define_used(lie,{3,0}); on solve_parameters;`

**meaning:** Initialization for the analysis, three variables used, and allow parameters to be solved. Without this switch set on, the algebra reduces to the zero algebra.

**input:** `on print_identities;solve_Jacobi_identities_of lie;`

**meaning:** is this a Lie algebra?

**output:** Starting stage 1:
Reordering the commutators...
Searching for identities...
Solving the identities...

```
{1,2,3}
- a(1)*x(2)
*** Solved for: \{a(1)\}
```

1 identities solved of 1

Starting stage 2:
Reordering the commutators...
Searching for identities...

Statistics for liebracket lie
3 even and 0 odd generators used
3 commutators solved of 3 (100 %)
0 linear dependencies found
1 parameters solved
0 unsolved identities

**result:** only if a(1) is zero it is a Lie algebra

**input:** `print_liebracket lie;`

**output:**
\[
\begin{align*}
[1,2] & := x(3) \\
[1,3] & := - x(2) \\
[2,3] & := x(1)
\end{align*}
\]

**REMARK.** It is possible to produce the complete product table of a simple Lie algebra by calling the procedure `solve_Jacobi_identities_of` once, where the input are the Serre relations only.

There are more valuable procedures included in the package, of which we just mention a procedure which saves the internal state of a 'Lie bracket' to a disk file to be able to resume work in another REDUCE session.

### 6.3. Super Vector-Fields

The construction of the super field package, started by the authors [14], and improved by Roelofs [15], was required by the introduction of superequations [16], and a mathematical description of it was given in [17].

Very briefly, it can be described as follows.

1. A vector space $V$ is a graded vector space if one has $V_0, V_1$ subspaces of $V$ such that

   \[ V = V_0 \oplus V_1. \]

   Elements in $V_0, V_1$ will be called even and odd, respectively.
2. A graded algebra $B$ is a graded vector space ($B = B_0 + B_1$) such that

$$B_iB_j \subset B_{i+j} \quad (i, j = 0, 1 \in \mathbb{Z}_2).$$

3. A graded algebra $B$ is called graded commutative if for any two homogeneous elements $x, y \in B$ one has

$$xy = (-1)^{|x||y|} yx,$$

where $|x| = i$ if $x \in D_i$.

4. The endomorphisms of a graded algebra/vector space $B$ can be graded too. If $B$ is a graded algebra, an operator $\alpha \in \text{End}(B)_i$ is called a (graded) derivation of $B$, if the graded Leibniz rule holds; i.e.,

$$\alpha(xy) = \alpha(x)y + (-1)^{|\alpha||x|} x\alpha(y).$$

The derivations of a graded algebra $B$ constitute a graded Lie algebra in the following way: $\alpha, \beta \in \text{Der}(B)$, define

$$[\alpha, \beta] = \alpha\beta - (-1)^{|\alpha||\beta|} \beta\alpha.$$

Then one has

1. $[x, y] = (-1)^{|x||y|} [y, x].$
2. $(-1)^{|x||z|} [x[y, z]] + (-1)^{|y||z|} [y[z, x]] + (-1)^{|z||x|} [z[x, y]] = 0.$

Proceeding in this way, one constructs the (graded) differential geometric objects as vector-fields, forms, contractions, and Lie derivatives, in a similar way as for the classical case. In [14] we gave a more detailed description of the construction of a graded differential geometry package.

In Section 7 we give an application to the $n = 2$ super KdV construction. This construction has been partly built on a recently constructed Mathematica program by one of the authors (P.K.H.G.).

6.4. Total Derivative Operators

In [18], a software-package has been described using total partial derivative operators $D_i$.

One should realize that, for instance, the symmetry condition and the covering condition are of a type involving the total partial derivative operators as operators on the infinite jet-space. The assumption that the functions involved are defined on a finite jet-space results in a truncation of these operators and a way to solve the conditions. The handling of equations in terms of partial derivative operators and only specifying the so-called highest orders explicitly prevents the computer system from intermediate expression swell and is for this reason very powerful.

Moreover, due to this way of handling, the algebraic structure of the resulting system of equations survives, which is very important from the mathematical point of view. More construction principles have been given in [19].

7. NEW $n = 2$ SUPEREXTENSION OF THE KORTEWEG-DE VRIES EQUATION

The Korteweg-de Vries (KdV) equation

$$u_t = -u_{xxx} + 6uu_x$$

is known to be completely integrable and has a number of interesting structures; e.g., it admits Bäcklund transformations, infinite dimensional prolongation algebra, infinite hierarchies of commuting symmetries and conservation laws, etc.
In recent years, there has been a strong development towards supersymmetric systems [16, 20]. The construction of a supersymmetric extension of an equation can be described by the introduction of an odd variable \( \theta \) and an odd field \( \Phi \) such that

\[
\theta \cdot \theta = 0, \quad \Phi = \varphi + \theta u, \quad xy = (-1)^{|x||y|} yx,
\]

where \( u \) is an even function and \( \varphi \) is an odd function with values in some unspecified Grassmann algebra \( (|\theta| = |\Phi| = 1, |u| = 0) \). In terms of the field \( \Phi \) and the variable \( \theta \), one arrives [16] at the equation

\[
\Phi_t = \Phi_\theta + 3D^2(\Phi \Phi^*_1),
\]

where

\[
D_t = \partial_t + \sum_{i=0}^{\infty} \Phi_{i,t} \partial_{\Phi_i}, \quad D = \partial_\theta + \theta \partial_x + \sum\limits_{i=0}^{\infty} \Phi_{i+1} \partial_{\Phi_i},
\]

and \( \Phi_t = D^t(\Phi) \). In terms of the components \( \varphi, \theta \) (7.1), (7.2) reduces to

\[
\varphi_t = -\varphi_{xxx} + 3\varphi\varphi_{xx}, \quad \theta_t = -\theta_{xxx} + 3\theta\theta_{xx}.
\]

System (7.4) is known as the one-dimensional \( (n = 1) \) supersymmetric extension of the KdV equation. It is completely integrable in the sense that it admits an infinite dimensional prolongation algebra, an infinite hierarchy of even and odd symmetries, master symmetries and recursion operators [8, 9]. Here we are interested in an \( n = 2 \) extension of the KdV equation. Until now there are three distinct superextensions known, which can be termed completely integrable. The construction requires the introduction of two odd variables \( \theta_1, \theta_2 \) and the even superfield \( \Phi \)

\[
\Phi = \omega + \theta_1 \xi_2 + \theta_2 \xi_1 + \theta_2 \theta_1 u,
\]

where, as in (7.5), \( \xi_1, \xi_2, \theta_1, \theta_1 \) are odd and \( \Phi, \omega, u \) are even.

In terms of the superfield formulation, the supersystem reads

\[
\Phi_t = D_x \left( -D_x^2 \Phi + 3D_1 D_2 \Phi + \frac{1}{2} (a - 1) D_1 D_2 \Phi^2 + a \Phi^3 \right),
\]

where

\[
D_1 = \partial_{\theta_1} + \theta_1 D_x, \quad D_2 = \partial_{\theta_2} + \theta_2 D_x.
\]

It has been proved that for values \( a = -2, 1, 4 \), the system is completely integrable. Written in component \( u, \xi_1, \xi_2, \omega \) the system (7.6) reads

\[
\begin{align*}
u_t &= D_x \left( -u_{xxx} + 3u^2 - 3\xi_1 \xi_{1x} - 3\xi_2 \xi_{2x} - (a + 1) u_x \right. \\
+ \left. (a + 2) \omega \xi_{xx} + 3a u \omega + 6a \omega \xi_2 \xi_1 \right) \\
\xi_{1t} &= D_x \left( -\xi_{1xx} + 3u \xi_1 + 3a \omega^2 \xi_1 - (a + 2) \omega \xi_{2x} - (a - 1) \omega \xi_2 \right) \\
\xi_{2t} &= D_x \left( -\xi_{2xx} + 3u \xi_2 + 3a \omega^2 \xi_2 + (a + 2) \omega \xi_{1x} + (a - 1) \omega \xi_1 \right) \\
\omega_t &= D_x \left( -u \omega + au + (a + 2) u \omega + (a - 1) \xi_2 \xi_1 \right).
\end{align*}
\]

Note that systems (7.4), (7.8) have been constructed in such a way that they reduce to the classical KdV equation (7.1) in absence of \( \varphi \) and \( \xi_1, \xi_2, \omega \), respectively.

In order to construct \( (n = 2) \) supersymmetric extensions of the KdV equation, we follow the lines as presented in [20]. The scheme for the construction is as follows.

1. Since KdV equation (7.2) admits a scaling, we require the scaling to be extendible to the super case, i.e.,

\[
[u] = 2, \quad [t] = -3, \quad [x] = -1, \quad [\theta_1] = [\theta_2] = -\frac{1}{2}, \quad [\xi_1] = [\xi_2] = \frac{3}{2}, \quad [\omega] = 1.
\]
so $|\Psi| = 1$, since all terms in (7.5) are of this degree. The degree of $\vartheta_1, \vartheta_2$ is induced by point 2 of this scheme.

2. Construct a general system of equations

$$
\begin{align*}
  u_t &= f_1[u, \xi_1, \xi_2, \omega] \\
  \xi_{1t} &= f_2[u, \xi_1, \xi_2, \omega] \\
  \xi_{2t} &= f_3[u, \xi_1, \xi_2, \omega] \\
  \omega_1 &= f_4[u, \xi_1, \xi_2, \omega],
\end{align*}
$$

(7.10)

where $f_1, \ldots, f_4$ are graded functions [10] in the algebra generated by the jet variables $u, u_x, \ldots, \omega, \omega_x, \ldots, \xi_1, \xi_{1x}, \ldots, \xi_2, \xi_{2x}, \ldots$ of degrees 5, 4.5, 4.5, 4, respectively, since $[u_t] = 3 + 2 = 5$, etc.

The first requirement is of course reduction to (7.1) in absence of $\xi_1, \xi_2, \omega$.

The system (7.10) now contains 83 constants to be determined by requirements 3 and 4 below.

3. The existence of two odd symmetries of (7.10)

\begin{align*}
  Y_{1/2} &= \xi_{1x} \partial_u + u \partial_{\xi_1} + \omega x \partial_{\xi_2} + \xi_2 \partial_{\omega} \\
  \overline{Y}_{1/2} &= -\xi_{2x} \partial_u + \omega x \partial_{\xi_1} - u \partial_{\xi_2} + \xi_1 \partial_{\omega}
\end{align*}

(7.11)

with $[Y_{1/2}, \overline{Y}_{1/2}] = [\overline{Y}_{1/2}, \overline{Y}_{1/2}] = -2\partial_x$.

4. We require the existence of a generalized symmetry of (7.10), i.e., a vertical vector-field

$$
Y^5 = g_1 \partial_u + g_2 \partial_{\xi_1} + g_3 \partial_{\xi_2} + g_4 \partial_{\omega},
$$

(7.12)

where $g_1, g_2, g_3, g_4$ are graded functions of degree 7, 6.5, 6.5, 6 which reduces to the classical higher order (first one) of KdV:

$$
Y^{5c} = (u_5 - 10u_3u - 20u_2u + 30u_1u^2) \partial_u
$$

(7.13)

in absence of $\xi_1, \xi_2, \omega$.

In order to have $g_1, g_2, g_3, g_4$, we have to introduce a huge number of coefficients (322) to be determined in an appropriate way.

From the construction scheme (1)-(4) we obtain the following result in addition to (7.8): there exists a supersymmetric extension of (7.1) of the form

\begin{align*}
  u_t &= -u_3 + 6u_1u + (c_5 - c_6)\xi_1\xi_{2x} + cs\xi_{1x}\xi_2 - (-3 + a_3)\xi_{1xx}\xi_1 + c_6\xi_{1xx}\xi_2 + a_3\xi_{2xx}\xi_2 \\
  &\quad -c_5w_{xx} - c_5u_2w_x \\
  \xi_{1t} &= -\xi_{1xx} + (3 + a_3)u_1\xi_2 - c_6u_2\xi_2 - (-3 + a_3)u_2\xi_1 + c_6u_2\xi_2 + (-c_5 - c_6)w_{xx}\xi_2 \\
  &\quad -a_3w_2\xi_{2x} - (c_6 - c_6)w_{xx}\xi_1 + a_3w_{xx}\xi_2 \\
  \xi_{2t} &= -\xi_{2xx} + (c_5 - c_6)u_1\xi_2 + (6 - a_3)u_2\xi_2 + (-c_5 - c_6)u_2\xi_1 + a_3u_2\xi_2 + (3 - a_3)w_{xx}\xi_2 \\
  &\quad +(-2c_5 + c_6)w_2\xi_{2x} + (-3 + a_3)w_{xx}\xi_1 - c_6w_{xx}\xi_2 \\
  \omega_t &= -w_{xxx} + (3 - a_3)\xi_{1xx} - (c_5 - c_6)\xi_{1xx}\xi_1 + a_3\xi_{1xx}\xi_2 - c_6\xi_{2xx}\xi_2 + 3w_{xx} - c_5w_{xx},
\end{align*}

(7.14)

where $a_3, a_5, a_6$ are parameters satisfying one condition

$$
-3a_3 + a_5^2 - c_5c_6 + c_6^2 = 0.
$$

(7.15)
In terms of the even superfield $\bar{\Phi}$, this system reads
\[ \Phi_t = -\Phi_{xxz} - 3\Phi_{21} \Phi_z + (3 - a_3) \Phi_{12} \Phi_2 - c_3 \Phi_z \Phi_x + (c_5 - c_6) \Phi_{1z} \Phi_1 - c_6 \Phi_{2z} \Phi_2 \] (7.16)

together with (7.15). Note that in (7.16)
\[ \Phi_1 = D_1 \Phi, \quad \Phi_2 = D_2 \Phi, \quad \Phi_{21} = D_1 D_2 \Phi, \quad \Phi_x = D_1 D_1 \Phi = D_2 D_2 \Phi \]

with $D_1 = \partial_{\phi_1} + \phi_1 \partial_{\phi_2}$, $D_2 = \partial_{\phi_2} + \phi_2 \partial_{\phi_1}$. For each value of the parameters in (7.15) we have a system which is supersymmetric in the sense that there exist odd symmetries, and which admits a generalized symmetry.

In order to obtain this result we have to construct the solution of a huge system of “nonlinear” algebraic equations for the 83 coefficients of the system (7.10) and the 322 coefficients of the vector-field $Y^5$. We claim that all these systems are completely integrable in the sense that they admit infinite hierarchies of symmetries, master-symmetries, and recursion operators. The results will be published elsewhere [21].

REFERENCES

12. [GR].