J. Symbolic Computation (2002) **33**, 367–383
doi:10.1006/jsco.2001.0509
Available online at http://www.idealibrary.com on IDEAL®

Size Reduction and Partial Decoupling of Systems of Equations

THOMAS WOLF

Department of Mathematics, Brock University, 500 Glenridge Avenue, St Catherines, Ontario, Canada L2S 3A1

A method is presented that reduces the number of terms of systems of linear equations (algebraic, ordinary and partial differential equations). As a byproduct these systems have a tendency to become partially decoupled and are more likely to be factorizable or integrable. A variation of this method is applicable to nonlinear systems. Modifications to improve efficiency are given and examples are shown. This procedure can be used in connection with the computation of the radical of a differential ideal (differential Gröbner basis).

1. Motivation

Algorithms for applying integrability conditions (IC) to a system of differential equations in a systematic way in order to generate simplified differential equations are implemented in a number of programs (Boulier *et al.*, 1995; Boulier *et al.*, 1997; Hubert, 1999, 2000; Mansfield, 1996; Reid *et al.*, 1996, 2001 and more in Hereman, 1996). Such calculations result in the radical or a (pseudo) differential Gröbner basis of the differential ideal generated by the original system. A common problem of these algorithms, and consequently their implementations, is an explosive expression swell. Optimizations like Buchberger's second criterion (Section 5.5 in Becker and Weispfennig, 1993) and their analogue for differential equations aim to reduce the number of steps to reach a characteristic set (one step = computation of an S-polynomial for algebraic systems or a cross-differentiation of two differential equations for differential systems). These optimizations do not cover other "obvious" simplifications. For a very simple example, consider Df to be a leading derivative of a function f in two equations $0 = Df + A$, $0 = Df + 2A$ with A a sum of a large number of terms. A simplification step in the standard procedure would aim at eliminating Df and get as a consequence the system $0 = Df + A$, $0 = A$ where the big expression A occurs twice. An alternative to be described in this paper would be to try to shorten equations and therefore to get at first the system $0 = Df + A$, $0 = Df$ and in a second length reduction step the system $0 = A$, $0 = Df$.

For a slightly more realistic example, consider two equations $0 = A+C$ and $0 = B-C$, where A, B, C are differential expressions, A, B having only a few terms and C involving many terms. If both equations are long and if they involve high derivatives then they would have a low priority to be used in standard algorithms. Typically, each of them would be paired individually with short low-order equations for their reduction or the generation of IC. A simplification of both equations to the system $0 = A + C$ and

$0 = A + B$ would usually not be found. Instead the expression C would grow in *both* equations when any substitutions of functions are made that occur in C.

Both examples are clear cut situations where big expressions in different equations can cancel each other. Although in practical applications to be described in a later section each length reduction step saves only a small number of terms, these reduction steps can often be repeated many times.

The need to reduce the length of equations comes from the danger of dealing with long equations in elimination algorithms. For elimination algorithms to be finite, i.e. to involve only a finite number of steps, they have to eliminate the leading derivative of equations *first* which involves differentiations and multiplications. Both tend to increase the length of long equations even further. However, elimination algorithms would remain finite if terms other than the leading derivative would be eliminated a *finite* number of times. Flexibility of this kind could be used to prevent excessive expression swell. In this way not only memory is saved. Long expressions also require an increased time to be computed which slows down any future computations in which long expressions are involved.

The procedure to be described in Section 2 is a first step in the direction of an "intelligent" and more efficient computation. It aims at finding equations in the algebraic ideal of the given system with fewer terms. The basic idea is rather straightforward. Any pair of two equations of a given system of equations is checked whether there is a linear combination (with non-vanishing coefficients) of these two equations that is shorter than the longer of the two. If that is the case then the longer one is replaced by the shorter new equation. To find a length-reducing linear combination of a pair of two equations, each term of one equation is divided by each term of the other equation, the quotient is simplified, i.e. common factors of the numerator and denominator are dropped and a counter of the number of occurrences of this quotient is incremented. The quotient occurring most often is picked, its numerator and its denominator are the multipliers of both equations which are then subtracted from each other. By choosing the quotient that occurs frequently a maximum number of terms cancel and the result is as short as possible.

The main content of the first part of the paper is to introduce data structures (L, c_i below) which allow an efficient implementation and to describe some optimizations that speed up the method and that allow one to consider nonlinear equations for length reduction. Beneficial side-effects are discussed such as the increased chance to find ordinary differential equations (ODEs) or to find exact differential equations in a length-reduced system of partial differential equations (PDEs).

In the second half, in Section 3 the method is applied to determining equations of Killing vectors (KVs) and Killing tensors (KTs) in General Relativity. One example is explained in more detail in the appendix where the beneficial side-effects of length reduction become important.

The length reduction module is incorporated in the package Crack for solving overdetermined PDE-systems. To show that the usefulness of the length reduction module is not just based on special features of Crack but is of a more universal nature a test is described at the end of Section 3. In this test other well known programs are used to solve a system of PDEs before and after it has been length reduced. The suitability of length reduction as a pre-processing step is demonstrated.

2. The Term Reduction Method

2.1. introduction

The procedure to be described takes as input two expressions E_1, E_2. Both represent equations $0 = E_1 = E_2$ and are regarded as sums with n_1 and n_2 terms $(n_1, n_2 \geq 1)$. The aim is to multiply each with a different single term and to add the expressions such that the result has fewer terms than $\max(n_1, n_2)$ and can be substituted for the longer of E_1, E_2. For a given system of equations this process is repeated with all possible pairs of equations until no pair can produce an equation that is shorter than the longer of both. The restrictions to multiply only with monomials and to combine only two equations at a time are non-trivial constraints (see the discussion in Section 4).

Example. The method will be illustrated with the following two expressions:

$$E_1 = 2xf + 6yf + 4xg + 5x = \sum_{i=1}^{4} E_{1i} \tag{1}$$

$$E_2 = 3yf - 3xf + 6yg - 7y = \sum_{i=1}^{4} E_{2i} \tag{2}$$

f and g are the unknowns that are to be computed from $0 = E_1$, $0 = E_2$. To explain the method it does not matter if f and g would be replaced by derivatives of unknown functions or by products of powers of different derivatives as long as $f \neq g$. x and y are independent variables or parameters such that $0 = E_1$, $0 = E_2$ are to be satisfied for any value of x and y.

2.2. the treatment of two equations

Given are two expressions E_1, E_2 with n_1 and n_2 terms, $n_1 \geq n_2$. If each expression is multiplied with a single term (monomial) and both expressions are added, then their sum E_3 can have between $n_1 - n_2$ and $n_1 + n_2$ terms depending on how many terms cancel each other. A way to find the optimal cancellation, i.e. optimal multipliers is to divide each term of expression E_1 by each term of E_2 and to collect the simplified quotients (common factors of numerator and denominator dropped) together with the multiplicity they occur.

Example. E_1, E_2 given in (1), (2) have four terms each $(n_1 = n_2 = 4)$ and the quotients are

$$(E_{1i}/E_{2i}) = \begin{pmatrix} \frac{2x}{3y} & -\frac{2}{3} & \frac{xf}{3yg} & -\frac{2xf}{7y} \\ 2 & -\frac{2y}{x} & \frac{f}{g} & -\frac{6f}{7} \\ \frac{4xg}{3yf} & -\frac{4g}{3f} & \frac{2x}{3y} & -\frac{4xg}{7y} \\ \frac{5x}{3yf} & -\frac{5}{3f} & \frac{5x}{6yg} & -\frac{5x}{7y} \end{pmatrix}.$$

A new equation E_3 would be generated by picking a quotient, say $E_{12}/E_{21} = \frac{4xg}{3yf}$ and using its numerator and denominator to compute $E_3 = 3yf \cdot E_1 - 4xg \cdot E_2$. Because this quotient involves f and g, the new equation $0 = E_3$ is nonlinear in f, g. As a consequence after replacing E_1 by E_3 the new system $0 = E_3 = E_2$ may not be equivalent to the old system $0 = E_1 = E_2$, i.e. E_1 could not automatically be replaced by E_3. The algorithm will therefore not consider quotients that involve any unknowns, here f and g. An effective method to avoid the computation of such quotients will be explained further below.

The quotient occurring most often is $\frac{2x}{3y}$ which is appearing twice and is free of f and g. Each appearance of a quotient means that two terms cancel. We therefore would expect that in the expression $E_3 := 3y \cdot E_1 - 2x \cdot E_2$ two times two terms cancel and therefore E_3 has $4 + 4 - 2 \cdot 2 = 4$ terms. But $E_3 = 6fx^2 + 18fy^2 + 29xy$ has only three terms. When computing $E_3 := 3y \cdot E_1 - 2x \cdot E_2$ the two terms $3y \cdot E_{14} - 2x \cdot E_{24} = 3y \cdot 5x - 2x \cdot (-7y) = 15xy + 14xy$ add up to only one term $29xy$. We could have forecast the saving of one additional term by realizing that the quotient $E_{14}/E_{24} = -\frac{5x}{7y}$ differs from $E_{11}/E_{21} = \frac{2x}{3y}$ only by a numerical factor.

The example implies that we should record all quotients with the multiplicity they occur and group them into classes c_i where all quotients in a class differ by only a numerical factor. Finally, for each class c_i the sum M_i of all the multiplicities of all quotients in the class is recorded too. All classes c_i together with M_i are listed in a list L:

$$\begin{aligned} L &= ((c_1, M_1), (c_2, M_2), \ldots, (c_r, M_r)) \\ c_i &= ((q_{i1}, m_{i1}), (q_{i2}, m_{i2}), \ldots, (q_{is_i}, m_{is_i})). \end{aligned}$$

- q_{ij} are the different quotients such that two quotients q_{ij}, q_{ik} in the same class c_i differ only by a numerical factor.
- s_i is the number of different quotients in the class c_i.
- m_{ij} is the number of how often q_{ij} occurs.
- M_i are defined as $M_i = \sum_{j=1}^{s_i} m_{ij}$.
- L is the complete (unsorted) list of all classes of quotients.

Disregarding quotients involving f or g in the above example we have

$$L = \left(\left(\left(\left(\frac{2x}{3y}, 2\right), \left(-\frac{5x}{7y}, 1\right)\right), 3\right), \left(\left(\left(-\frac{2}{3}, 1\right), (2, 1)\right), 2\right), \left(\left(\left(-\frac{2y}{x}, 1\right)\right), 1\right)\right)$$

with, for example, $q_{11} = \frac{2x}{3y}$ turning up twice, once as E_{11}/E_{21} and once as E_{13}/E_{23}, therefore $m_{11} = 2$, and further $c_1 = ((\frac{2x}{3y}, 2), (-\frac{5x}{7y}, 1))$, $M_1 = m_{11} + m_{12} = 2 + 1 = 3$.

If a quotient q_{ij} is used to combine E_1, E_2 to

$$E_3 = \text{denominator}(q_{ij}) \times E_1 - \text{numerator}(q_{ij}) \times E_2$$

then the number n_3 of terms of E_3 is

$$\begin{aligned} n_3 &= n_1 + n_2 - 2 \times m_{ij} && \text{(due to } m_{ij} \text{ complete cancellations of 2 terms)} \\ &\quad - \sum_{k=1, k \neq j}^{s_i} m_{ik} && \text{(due to savings of one term each time)} \qquad (3) \\ &= n_1 + n_2 - m_{ij} - M_i. \end{aligned}$$

The $\sum m_{ik}$ in equation (3) comes from simplifications such as $15xy + 14xy = 29xy$ which each save one term. In order to be successful and to replace E_1 by E_3 we need to find a quotient q_{ij} such that E_3 has fewer terms than E_1, i.e. $n_3 = n_1 - n_2 - m_{ij} - M_i < n_1$, hence $m_{ij} + M_i > n_2$.

2.3. a pre-processing step

As argued above only quotients q_{ij} should be considered which do not involve any unknowns, i.e. functions or constants that are to be computed from $E_i = 0$. An effective method to even avoid the computation of those quotients requires to re-write expressions E_1, E_2 in the following way. This initial re-writing step also clarifies how the method works for nonlinear expressions E_1, E_2.

One always can regard nonlinear expressions E_1, E_2 as linear homogeneous expressions in some newly defined variables v_a which are linear or nonlinear constructs of the dependent variables. For example, for independent variables x, y, dependent variables $f = \text{const}$, $g = g(x, y)$ and

$$E_1 = 3x + 3\cos(x)fg_x - 12xyg + 6xg - yg + \sin(g_y), \tag{4}$$
$$E_2 = 1 + 4fg_x - 4yg + 2g \tag{5}$$

the related system that is homogeneous and linear in v_i would be

$$E_1 = 3xv_0 + 3\cos(x)v_1 - 12xyv_2 + 6xv_2 - yv_2 + v_3, \tag{6}$$
$$E_2 = v_0 + 4v_1 - 4yv_2 + 2v_2 \tag{7}$$

with $v_0 = 1$, $v_1 = fg_x$, $v_2 = g$, and $v_3 = \sin(g_y)$. After this re-writing the method will investigate the system (6), (7) and avoid quotients q_{ij} that involve any v_l.

2.4. methods to increase speed

The restriction of not multiplying with factors involving dependent variables enables the following major speed up. Instead of investigating the system (4), (5) and computing $6 \times 4 = 24$ quotients we investigate the system (6), (7) and compute only quotients between the terms of the coefficients of the same v_i in E_1 and E_2. This reduces the number of quotients to 1×1 (for v_0) + 1×1 (for v_1) + 3×2 (for v_2) + 3×0 (for v_3) = 8 quotients. For large expressions, or more exactly for a high number of different v_j the speed up is naturally much higher. If we have $r + 1$ new dependent variables $v_0, \ldots, v_r$ and if we denote the number of terms involving v_j in E_i as n_{ij}, $0 \le j \le r$ then instead of computing $\left(\sum_{j=0}^{r} n_{1j}\right) \times \left(\sum_{k=0}^{r} n_{2k}\right)$ quotients the more efficient method only computes $\sum_{j=0}^{r}(n_{1j} \times n_{2j})$ quotients.

Another way of increasing efficiency is based on knowing n_{ij} beforehand. For the terms involving a specific v_j, an upper bound on the maximal number of cancellations is $\min(n_{1j}, n_{2j})$, saving twice as many terms. This value summed over $j = 0 \ldots r$ (for each v_j) gives an upper bound on how many terms can be saved due to cancellations.

This test can be performed without computing any quotients:
If the new dependent variables are $v_0, \ldots, v_r$ and if we have at the beginning

$$\sum_{j=0}^{r} 2\min(n_{1j}, n_{2j}) \le n_2 \left(= \sum_{j=0}^{r} n_{2j} \right)$$

($n_2 + 1$ *is the minimum number of terms to be saved to reach a length reduction*) *then no length reduction is possible.*

This test of a necessary criterion is not only possible at the beginning but also during the computation of quotients. We assume that at first all quotients related to v_0 are computed, then those related to v_1 and so on. When the calculation has reached v_j and the first w terms in E_1 that involve v_j have been processed, i.e. all quotients between each of them and all terms in E_2 with v_j have already been computed then the following holds. At most $\min(n_{1j} - w, n_{2j})$ more cancellations related to v_j are possible and an upper bound $B_j(w)$ of the total number of any cancellations still to be found is given as

$$B_j(w) := \min(n_{1j} - w, n_{2j}) + \sum_{i=j+1}^{r} \min(n_{1i}, n_{2i}).$$

At this moment any quotient q_{kl} that has a chance to provide a length reduction must satisfy

$$M_k + m_{kl} + 2B_j(w) > n_2.$$

All q_{kl} which do not satisfy this condition can be dropped from the list L. For the same reason no new quotients should be added to L as soon as $2B_j(w) \leq n_2$. If the list L becomes empty at any time during the computation of quotients, then the search can stop. In that case no length reduction is possible. By dropping quotients from the list L, the updating of m_{ij} and M_i speeds up.

2.5. a speed up not recommended

When deleting quotients from L that have no chance to give a length reduction then it is little extra effort to check whether any q_{kl} of the remaining quotients in L already satisfies $M_k + m_{kl} > n_2$, and therefore is guaranteed to provide a length reduction. As soon as such a quotient q_{kl} is found the execution could stop. In practical tests it appeared that the negative effects of an early stop dominate. The first length reducing quotient q_{kl} that is found does not have to be the one giving the highest length reduction possible. After using a sub-optimal q_{kl} to compute E_3 and to substitute E_1 no further length reductions may be possible, or even if further length reductions were possible, equations tend to have at least intermediately more terms compared with determining always the optimal quotient that gives the highest length reduction. After a suboptimal length reduction, subsequent pairings with other equations would be slower which would result in an overall slow down. It therefore is recommended to complete the computation of all relevant quotients and not to stop early when the first length reducing q_{kl} is found.

2.6. some time tests

The following tests can only provide some idea of the running times of the length reduction method. They are measured in a 8 MB Reduce 3.6 session running under Linux on a 133 MHz Pentium PC (December 1998). Equations which have been paired had been generated with the Reduce command `RANDPOLY` which allows one to specify the number of terms to be generated. In `RANDPOLY` the randomly generated coefficients may become zero which in that case results in a polynomial with fewer terms than specified. Therefore the number of terms was chosen somewhat larger to be able to drop the surplus terms and get the required size of the polynomial and perform the following statistics.

Performing a length reduction investigation involves no other risk or cost than the computer time that may be lost if no length reduction was possible. Therefore in Figure 1 and Table 1 a statistics of investigations of each time two equations is shown where no length reduction could be found, i.e. the worst possible result. This will be referred to

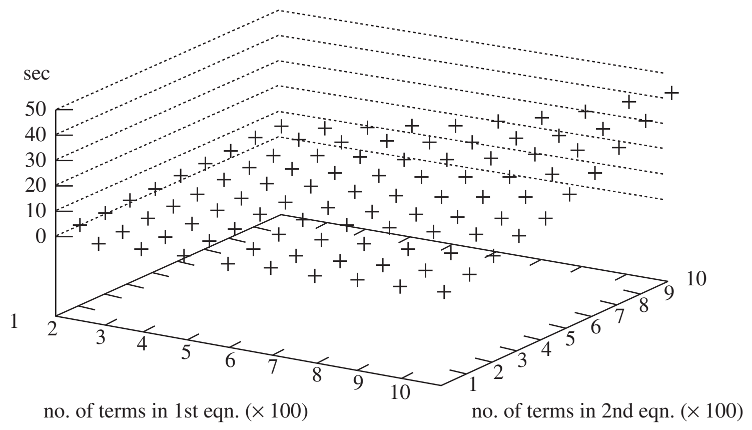

Figure 1. Running times for random polynomials of different sizes.

Table 1. Timings of unsuccessful length reduction attempts of one equation with varying length and a second equation that has either 10 or 1000 terms.

No. of terms of the first equation	100	200	300	400	500	600	700	800	900	1000
Time in sec (no success) if Second equation has 10 terms	0.07	0.23	0.45	0.70	0.72	1.01	1.36	1.8	2.2	2.7
Time in sec (no success) if Second equation has 1000 terms	4.18	6.23	9.35	12.6	15.4	20.3	24.4	30.2	36.4	43.8

as unsuccessful pairings in contrast to successful pairings where a length reduction was possible. Both equations are random polynomials of degree up to 5 and with up to eight variables. Times are obtained by averaging 20 runs. The individual times in these runs differ typically by up to 30%. The results confirm an overall dependence time $\propto$ (terms of equation (1)) $\times$ (terms of equation (2)).

Test results shown in Table 2 are based on pairing equations which are polynomials of seventh degree with each 300 terms but with a varying number of variables. Times are averaged again over 20 runs.

The effect of efficiency improvements as described in the above sub-section to detect the non-existence of length reductions early can be seen clearly from the second row in Table 2. As more independent variables occur the number of different quotients q_{kl} increases and the average frequency for each quotient to appear becomes smaller. This in turn rules out many quotients early in the computation and it becomes clear earlier that no quotient will result in a length reduction if that is the case.

Usually length reductions do not happen with random polynomials of that size. In order to measure computing times for pairings when length reductions were possible, pairs of polynomials had been constructed in the following way: a multiple of one random polynomial P_1 of 300 terms is added to another random polynomial P_2 and terms in excess of 300 terms are dropped to obtain a polynomial P_3 with 300 terms. Length reductions between polynomials P_1 and P_3 are investigated which produced the third row in Table 2. Two trends seem to be present, one lowering the time with an increase of the number of variables (mainly effective between four and five variables) due to a decrease of potentially

Table 2. A comparison between average running times of unsuccessful and successful pairings of two equations.

No. of variables	4	5	6	7	8
Time in msec (unsuccessful)	4720	2461	1673	1656	1640
Time in msec (successful)	5706	4968	5137	5388	5728

successful quotients and another trend slightly increasing the time with the number of variables.

To summarize this sub-section, the main feature found in this study is that computing times are lower in unsuccessful attempts to shorten equations and only when a length reduction becomes possible, i.e. when computing times are of less importance, then they are increased. The speed up measures become increasingly efficient if more unknowns (like f and g in the first example) are present which is the case for PDEs with many different partial derivatives acting as different unknowns v_j during length reduction.

2.7. THE ORDER OF PAIRINGS OF EQUATIONS

If more than two equations are given then the question arises in which order they should be paired to search for length reductions. Given that we combine only two equations at a time and multiplying them only with a monomial, we cannot expect results that are invariant against combining equations in a different order. The following criteria serve only as a suggestion but they proved to be useful in applications. According to them pairs of equations are picked with the following priorities:

- There should be as *few* as possible dependent variables v_i in the shorter equation which do not occur in the longer equation.
- The shorter of both equations should be as short as possible.
- The longer of both equations should be as short as possible.

The first two rules maximizes the chance to find a reduction of terms. The third rule reduces computation times. The second rule has a higher priority than the third rule because the shorter the equations are, the more useful they are potentially in reducing the length of other equations.

In the following table the above priority list is compared with the same list, only modified by exchanging in the first rule 'as few as possible' with 'as many as possible'. The equations are a set of first-order PDEs resulting from investigating in General Relativity the Kimura metric (Kimura, 1976) with respect to KTs (see Section 3). Because sin and cos occur in these differential equations, both length reductions are performed once with the simplification rule $\cos(x)^2 \Rightarrow 1 - \sin(x)^2$ and once with the simplification rule $\sin(x)^2 \Rightarrow 1 - \cos(x)^2$. It becomes apparent that these simplifications are not equivalent in their effect.

As it was to be expected, the '... as few as ...' rule performed better (i.e. resulted in shorter length reduced systems) than the '... as many as ...' rule. What also becomes apparent is that choosing accidentally the less effective simplification rule $\sin(x)^2 \Rightarrow 1 -$

Table 3. A comparison of different simplification rules and different rules for the pairing of equations.

Simplification used	Original system		Length red. system using the rule "... as many as ..."		Length red. system using the rule "... as few as ..."	
	No. of eqn.	Terms	No. of eqn.	Terms	No. of eqn.	Terms
$\sin(x)^2 \Rightarrow 1-\cos(x)^2$	48	607	25	117	21	81
$\cos(x)^2 \Rightarrow 1-\sin(x)^2$	48	464	25	108	21	74

Table 4. The chances for random bilinear polynomials in up to five variables to be factorizable in dependence on the number of terms.

No. of terms	2	3	4	5	6
Factorizable equations in %	55.4	13.6	2.3	0.22	1.3×10^{-3}

$\cos(x)^2$ is less critical when length reduction is performed than without length reduction:

$$\frac{\text{no. of terms of original system using cos-rule}}{\text{no. of terms of original system using sin-rule}} = \frac{607}{464}$$
$$> \frac{\text{no. of terms reduced system using cos-rule}}{\text{no. of terms of reduced system using sin-rule}} = \frac{81}{74}.$$

2.8. BENEFICIAL SIDE EFFECTS

The reduction of length is not only useful for saving memory, and as shorter expressions are quicker to process later on, also for saving time. In this section we want to explain further benefits.

(1) Given a set of PDEs, the length reduced system is more likely to contain ODEs or to contain integrable exact PDEs.
(2) Length-reduced polynomially nonlinear equations are much more likely to be algebraically factorizable.
(3) Length reduction of a system of equations has in general the side effect of partially decoupling the system.

If the number of terms of an equation is lowered to, say, n and if the equation is linear then n is necessarily an upper bound for the number of different functions and different derivatives that occur. The length reduction method as described above is indiscriminate to different functions or different derivatives. On average therefore a consequence of a reduction of the number of terms will be a reduction of the number of different functions and the number of different derivatives that occur (see the comparison between Tables 6 and 7 below) which in turn provides the above side effects.

ABOUT 1. If the number of terms in a length-reduced equation got very small (say less than four) then the chance increases that they involve only one differentiation variable, or that all derivatives can be looked at as derivatives with respect to only one variable of a common partial derivative, like $\partial_y f, \partial_{xy} f, \partial_{xxy} f$ are all x-derivatives of $\partial_y f$. In these cases the equation has been reduced to an ODE. (A proper ODE can of course only appear if the

Table 5. The number of length reduction steps and the computation time for the length reduction of different KV and KT determining systems of equations.

Equations	Original		Shortened		No. of	Time
	Eqn.	Terms	Eqn.	Terms	red. steps	in sec
KV for Taub-NUT (p. 170 in Hawking and Ellis, 1973) + IC	49	1929	44	932	192	120
KT for the Kerr metric (p. 161 in Hawking and Ellis, 1973)	20	1154	20	750	36	87
KT for the Kimura metric (Kimura, 1976) + CT	48	464	21	74	93	19.5
KV for the Barnes metric (Barnes, 1979)	10	66	10	39	14	2.3
KV for pp waves (p. 178 in Hawking and Ellis, 1973)	10	58	10	37	13	1.7

Table 6. The original set of conditions for KTs in the Kimura metric.

No. of terms	Dependent variables	No. of terms	Dependent variables
2	k01, k00	9	k22, k11, k00, k13, k03, k33, k23
2	k11	9	k22, k11, k01, k00, k02, k03, k33
2	k22, k12	9	k22, k11, k01, k00, k02, k03, k33
3	k13, k33, k23	9	k22, k33, k23
3	k12, k00, k02	9	k11, k01, k00
3	k00, k13, k03	9	k11, k01, k00
3	k11, k01	10	k22, k12, k11, k00, k02, k33, k23
3	k22, k01, k02	11	k22, k12, k13, k33, k23
3	k12, k11	11	k22, k12, k13, k33, k23
3	k11, k13	12	k22, k12, k11, k01, k00, k13, k33
4	k01, k02, k03, k33	14	k22, k12, k11, k01, k13, k02, k03, k33
4	k11, k01, k00	14	k22, k12, k11, k01, k13, k02, k03, k33
4	k12, k01, k02	18	k12, k11, k01, k00, k13, k02, k03
4	k01, k13, k03	18	k12, k11, k01, k00, k13, k02, k03
4	k02, k03, k23	18	k12, k11, k01, k00, k13, k02, k03
4	k22, k12, k11	18	k12, k11, k01, k00, k13, k02, k03
4	k22, k13, k23	19	k22, k12, k11, k00, k13, k02, k03, k33, k23
5	k22, k12, k33, k23	19	k22, k12, k11, k00, k13, k02, k03, k33, k23
5	k12, k11, k13, k33	21	k22, k12, k11, k01, k00, k13, k02, k03, k33
5	k12, k13, k23	21	k22, k12, k11, k01, k00, k13, k02, k03, k33
8	k11, k01, k00	21	k22, k12, k13, k02, k03, k33, k23
8	k12, k11, k00, k13, k02, k03	21	k22, k12, k13, k02, k03, k33, k23
8	k12, k11, k00, k13, k02, k03	21	k22, k12, k13, k02, k03, k33, k23
8	k11, k01, k00	21	k22, k12, k13, k02, k03, k33, k23

differential ideal of the original system does contain ODEs, but that is guaranteed for all the typical sources of over-determined PDE-systems, like the computation of infinitesimal symmetries and of conservation laws.)

Similarly the chance increases to obtain an exact differential equation. In order for a differential expression $(P(f^i)$ that involves functions f^i and that satisfies $0 = P$ to be a total x derivative of some expression $I(f^i)$, the identity $P = dI/dx$ has to be satisfied identically in all functions f^i and in all their derivatives. The fewer different functions

Table 7. The length-reduced conditions for KTs in the Kimura metric.

No. of terms	Dependent variables	No. of terms	Dependent variables
2	k01, k00	4	k22, k13, k33, k23
2	k11	4	k11, k01, k00
2	k22, k12	4	k01, k02, k03, k33
3	k13, k33, k23	4	k12, k01, k02
3	k12, k00, k02	4	k01, k13, k03
3	k00, k13, k03	4	k02, k03, k23
3	k11, k01	4	k22, k12, k11
3	k22, k01, k02	5	k22, k33, k23
3	k12, k11	5	k12, k11, k13, k33
3	k11, k13	5	k12, k13, k23
4	k22, k33, k23		

f^i and the fewer different derivatives with respect to variables other than x occur, the less restrictive is this assumption of exactness and the more likely it will be satisfied.

The conclusion is not that length reduction is the best way to proceed in order to integrate. We only say that the chance increases to find an integrable PDF in a length reduced system. In the appendix this statement is illustrated by a sequence of 10 integrations which a system of KT equations admits after being reduced in length.

To explain the usefulness of integrability let us consider an extreme hypothetical example. Gröbner Basis techniques aim at equations with a low differential order. This is a good strategy but not the only way to go. For example, knowing that $0 = \partial^{10} f / \partial x^{10}$ is included in the differential ideal would be very useful as well. After integration, substitution of f and direct separation with respect to x (sometimes called splitting or fragmentation) a highly over determined system for the 10 functions of integration results. Although an expensive Gröbner basis computation might have provided an ODE of lower order than 10, this information is usually gained faster by integrating $0 = \partial^{10} f / \partial x^{10}$ and solving the over determined system that resulted from direct separation for the 10 functions of integration. The key to this speed up would be to sacrifice the minimal differential order for a reduction in the number of terms.

About 2. In a small experiment we want to show that the chance for an algebraic factorization increases with a reduction of the number of terms. Homogeneous quadratic polynomials with up to five variables and coefficients in the interval $-9, \ldots, -1, 1 \ldots 9$ have been randomly generated for each length from two terms up to six terms. The percentage of factorizable polynomials is given in Table 4. The chance to have a non-trivial factorization (non-numeric factors) decreases surprisingly quick with an increasing number of terms. Although this test is not proving anything it may serve as an illustration.

About 3. The fewer different functions occur in each equation (on average) the more the system is (at least partially) decoupled and the fewer steps are needed in a subsequent computation to get a differential Gröbner Basis or characteristic system. This means that an elimination algorithm performed afterwards has less work to do, i.e. needs fewer steps. A sparsely occupied system also opens the possibility to choose an appropriate total ordering on which an elimination algorithm is based. For example, if a function turns up in only few equations and with very few different derivatives then one could give this function a high priority, i.e. the lexicographical ordering of functions which plays a role in any total ordering could give this function a high priority to be eliminated first because this will probably take the fewest steps to eliminate this function.

3. Applications

One of the applications to which the above length reduction procedure has been applied successfully is the computation of KVs $K_i(x^p)$ and rank 2 KTs $K_{ij}(x^p)$ of given space times with a metric g_{ij} in General Relativity. KVs and KTs provide conservation laws of the form $K_i u^i =$ const. and $K_{ij}u^i u^j =$ const. where u^i is the 4-velocity of geodesic motion in a curved space-time. If enough KVs (i.e. symmetries of the space time) and KTs are found then the equations of free (geodesic) motion of test particles in a curved space can be integrated which is a major step towards the physical interpretation of a space time metric. KVs and their Lie algebras are also widely used to classify space time metrics.

The determining equations for KVs are

$$K_{i;j} + K_{j;i} = 0, \qquad i, j = 1 \ldots 4 \tag{8}$$

and for KTs they are

$$K_{ij;l} + K_{jl;i} + K_{li;j} = 0, \qquad i, j, l = 1 \ldots 4 \tag{9}$$

where ';' is the covariant derivative. Conditions (8), (9) are generated for a given space–time with metric tensor g_{ij} by the program Classym described in Wolf and Grebot (1994). Examples are shown in Table 5 (not selected from a larger set, but as they appeared in applications). The systems of equations determine either KVs or KTs for different space time metrics. One system is extended by integrability conditions (IC) and one system by contractions (CT) with the metric tensor. The number of equations and terms before and after length reduction are shown. Running times in a Reduce 3.6 session running under Linux on a 133 MHz Pentium PC are given in the right column.

The benefits of a length reduction for a program like Crack (Wolf, 1996) are manifold. Tables 6 and 7 show the number of terms and the occurring dependent variables in each of the KT equations for the Kimura metric before and after the reduction of length.

The equations of Table 7 (after dropping a single linear dependent equation with four terms) are explicitly given in the appendix. It is indicated there how the sparse occurrence of different derivatives leads to integrable ODEs and exact differential equations.

3.1. length reduction as an adjunct to elimination algorithms

The main purpose of the term reduction method is to shorten and simplify systems of equations without imposing any risk of an intermediate length increase. This risk is present when using standard Gaussian elimination or standard Gröbner basis methods. For that reason and the benefits discussed above it would not matter if the term reduction method would be comparatively time consuming. The following run time tests show that term reduction can even be time saving if it is used in connection with computing differential Gröbner bases or the radical of a differential ideal.

The program `rif` Reid *et al.* (1996, 2001) has been applied by Allan Wittkopf to the KT conditions for the Kimura metric (see Table 3, the shortened form is given in the appendix). Because we only provide the relative speed up in Tables 8, 9 the following

Table 8. Relative speed up of computing times (cpu time) for the program `rif` for different versions of the KT conditions of the kimura metric.

	`rif`, with b pivots (%)	`rif`, without b pivots (%)
$t_{\text{kimu2}}/t_{\text{kimu1}}$	96.8	93.2
$t_{\text{kimu3}}/t_{\text{kimu1}}$	86.6	73.3

Table 9. Relative speed up of computing times (cpu time) for the program `diffalg` for different versions of the KT conditions of the Kimura metric.

	`diffalg` (%)
$t_{\text{kimu2}}/t_{\text{kimu1}}$	83.7
$t_{\text{kimu3}}/t_{\text{kimu1}}$	40.1

hardware and software specifications are not of much relevance. They are added only for completeness. Computing times of the program rif are measured on a PII 333 MHz Linux system with 384 MB RAM (although only about 3 MB was needed), running in XMaple release 5.0.

Differential Gröbner Bases have been computed, once in the generic case where the constant b is treated like a dependent variable and a second time where b is regarded as an arbitrary constant, ignoring b pivots.

In the Tables 8 and 9 kimu1 is the form of the system with 607 terms (see Table 3), kimu2 is a pre-optimal form of the system with 128 terms which resulted when a non-optimal pairing of equations was used and finally kimu3 is the system with 74 terms (see appendix).

The Maple program `diffalg` has been applied by Evelyne Hubert (see Boulier *et al.* (1995); Boulier *et al.* (1997); Hubert (1999, 2000) and the URL `http://www-sop.inria.fr/cafe/evelyne.hubert/webdiffalg`). `diffalg` was run on a PC with dual Intel 400 MHz CPUs, 512 MB RAM, 128 MB swap, Asus P2L7-DS motherboard, 4.3 GB Seagate hard drive and Intel Ether Express pro 10/100 network card under LINUX.

A potential gain of a reduction of the number of terms is the possibility to choose a more appropriate total ordering in a Gröbner Basis computation when a system is already partially decoupled. For example, dependent variables which turn up in fewer equations could be given a higher lexicographical priority than other dependent variables. So far no advantage has been taken of this opportunity in the package Crack.

4. Possible Extensions

The manipulations studied in this paper to achieve a reduction of length seem very special and one might seek a more comprehensive theory, for example, covering all possible linear combinations of any number of equations multiplied not only with monomials but with any number of terms. This problem is harder than it looks. It would include being able to decide whether any given linear algebraic system could be solved without intermediate memory increase. Already the question of combining three equations to kill at least five terms (four terms could be killed by combining two equations twice) has a much larger search space than combining two equations. Although not proven, the author expects that in generic practical applications the frequency of reducing the length of any one of three equations which cannot be reached by combining repeatedly two equations is low (which, of course, may be different in specific applications). To give an example, the system $0 = a + b - c - d + g$, $0 = c + d - e - f + h$, $0 = e + f - a - b + k$ gives a shortened equation only if all three equations are combined (added).

It seems to be a general phenomenon that the computational complexity increases drastically when any method of investigating systems of equations is generalized whereas the success rate of the generalized method increases only marginally.

To give an example, in determining Lie-symmetries and conservation laws of generic PDEs (not the rare fully integrable PDEs) the success rate in finding symmetries when going from point symmetries to higher order symmetries or the success rate in finding conservation laws when going from zeroth order integrating factors to higher order conservation laws increases only marginally. On the other hand the complexity in solving the related over determined system of conditions grows drastically when the order of the ansatz is increased. The growth in complexity comes from the task to determine an increased number of constants or functions or in determining functions of more independent variables. The decreasing success rate results from a relatively more over determined problem: having to satisfy not only the more restrictive conditions of a more general ansatz but also the conditions that the solution may not be decomposable into simpler cases.

5. Summary

Although conceptually simple, the method explained in this paper proved to be very useful. It is fast and it has zero risk of increasing the size of the system of equations during the computation or as a result of it. This is an important feature for very memory intensive computations. In such a case, computation times are of less concern. Still, it was possible to show that even for other well known programs the time saving using term reduction could be higher than the additional cost.

As discussed in the subsection ‘Other benefits’, size reduced equations on average are more likely an ODE or a total derivative and therefore easily integrable. They are also more likely to be algebraically factorizable. The fact that the system becomes partially decoupled opens the possibility to select a total ordering according to which the system is already close to a characteristic form. This has the potential to reduce subsequent computations to obtain a characteristic system to a high extent.

Acknowledgements

The author thanks Allan Wittkopf for many discussions. He, together with Evelyne Hubert are thanked for running the tests mentioned in the text with their systems. The Symbolic Computation Group at the University of Waterloo is thanked for its hospitality during the authors sabbatical in the fall of 1998 when the manuscript was prepared. This work was partially supported by the Natural Sciences and Engineering Research Council of Canada under CRD grant no. 661-090/93 and partially supported by the Engineering and Physical Sciences Research Council England under grant no. GR/L99036.

References

Barnes, A. (1979). On space-times admitting a three-parameter isometry group with two-dimensional null orbits. *J. Phys. A: Math. Gen.*, **12**, 1493–1497.

Becker, T., Weispfennig, V. (1993). *Gröbner Bases,* volume 141 of *Graduate Texts in Mathematics.* 222. Springer Verlag.

Boulier, F., Lazard, D., Ollivier, F., Petitot, M. (1995). *Computing Representations for Radicals of Finitely Generated Differential Ideals, Proceedings of ISSAC'95*, pp. 158–166. ACM Press.

Boulier, F., Lazard, D., Ollivier, F., Petitot, M. (1997). Computing representations for radicals of finitely generated differential ideals. Technical Report IT-306, LIFL.

Hawking, S. W., Ellis, G. F. R. (1973). *The Large Scale Structure of Space-time.* Cambridge University Press.

Hereman, W. (1996). Symbolic software for lie symmetry analysis. In Ibragimov, N H ed., *CRC Handbook of Lie Group Analysis of Differential Equations*, volume 3, Chapter 13, pp. 367–413. Boca Raton, Florida, CRC Press.

Hubert, E. (1999). Essential components of algebraic differential equations. *J. Symbolic Computation*, **28**, 657–680.

Hubert, E. (2000). Factorisation-free decomposition algorithms in differential algebra. *J. Symbolic Computation*, **29**, 641–662.

Kimura, M. (1976). On Quadratic First Integrals in Static Spherically Symmetric Space-times, having Spacial Parts of non-constant Curvature, I. *I. Tensor, N. S.*, **30**, 27–43.

Mansfield, E. L. (1996). The differential algebra package diffgrob2. *Mapletech*, **3**, 33–37.

Reid, G. J., Wittkopf, A. D., Boulton, A. (1996). Reduction of systems of nonlinear partial differential equations to simplified involutive forms. *Eur. J. Appl. Math.*, **7**, 604–635.

Reid, G. J., Lin, P., Wittkopf, A. D. (2001). Differential-elimination completion algorithms for differential algebraic equations and partial differential algebraic equations. *Studies Appl. Math.*, **106**, 1–45.

Wolf, T., Grebot, G. (1994). Automatic symmetry investigation of space–time metrics. *Int. J. Mod. Phys. D.*, **3**, 323–326.

Wolf, T. (1996). The program CRACK for solving PDEs in General Relativity. In Hehl, F. W., Puntigam, R. A., Ruder, H. eds, *Relativity and Scientific Computing: Computer Algebra, Numerics, Visualization*, pp. 241–258. Springer Verlag.

Appendix: Length Reduced Conditions for Killing Tensors of the Kimura Metric

The size reduced determining conditions for KTs in the Kimura metric read:

$$0 = k_{22,h} + 2b^2r^3k_{12} \tag{A1}$$

$$0 = rk_{11,r} + 2k_{11} \tag{A2}$$

$$0 = k_{00,t} - 2br^3k_{01} \tag{A3}$$

$$0 = 2\cos(h)\sin(h)k_{23} + k_{33,p} + 2\sin(h)^2b^2r^3k_{13} \tag{A4}$$

$$0 = rk_{11,p} + 2rk_{13,r} - 2k_{13} \tag{A5}$$

$$0 = rk_{11,h} + 2rk_{12,r} - 2k_{12} \tag{A6}$$

$$0 = 2k_{02,h} + k_{22,t} + 2b^2r^3k_{01} \tag{A7}$$

$$0 = 2rk_{01,r} + rk_{11,t} - 2k_{01} \tag{A8}$$

$$0 = k_{00,p} + 2k_{03,t} - 2br^3k_{13} \tag{A9}$$

$$0 = k_{00,h} + 2k_{02,t} - 2br^3k_{12} \tag{A10}$$

$$0 = 6\cos(h)\sin(h)k_{23} - \sin(h)^2k_{22,p} - 2\sin(h)^2k_{23,h} + k_{33,p} \tag{A11}$$

$$0 = 2\cos(h)\sin(h)k_{02} + 2k_{03,p} + k_{33,t} + 2\sin(h)^2b^2r^3k_{01} \tag{A12}$$

$$0 = 2rk_{12,h} + rk_{22,r} + 2b^2r^4k_{11} - 4k_{22} \tag{A13}$$

$$0 = 2\cos(h)k_{03} - \sin(h)k_{02,p} - \sin(h)k_{03,h} - \sin(h)k_{23,t} \tag{A14}$$

$$0 = rk_{01,p} + rk_{03,r} + rk_{13,t} - 4k_{03} \tag{A15}$$

$$0 = rk_{01,h} + rk_{02,r} + rk_{12,t} - 4k_{02} \tag{A16}$$

$$0 = rk_{00,r} + 2rk_{01,t} + br^5k_{11,r} - 4k_{00} \tag{A17}$$

$$\begin{aligned}0 &= 2\cos(h)\sin(h)^2k_{22} - 4\cos(h)k_{33} - \sin(h)^3k_{22,h}\\ &\quad + 2\sin(h)k_{23,p} + \sin(h)k_{33,h}\end{aligned} \tag{A18}$$

$$0 = 2\cos(h)\sin(h)rk_{12} + 2rk_{13,p} + rk_{33,r} + 2\sin(h)^2b^2r^4k_{11} - 4k_{33} \tag{A19}$$

$$0 = 2\cos(h)rk_{13} - \sin(h)rk_{12,p} - \sin(h)rk_{13,h} - \sin(h)rk_{23,r} + 4\sin(h)k_{23}. \tag{A20}$$

One effect of reducing the number of terms is that fewer different derivatives of different dependent variables k_{ij} occur in each equation. As a consequence the chance for the equations to be exact or to be simple ODEs increases. The package Crack is able to integrate 10 of the above equations and as a consequence to express 10 of the unknown functions k_{ij} in terms of new unknown functions of integration of only three variables: integration of (A2) to solve for k_{11}, of (A5) to solve for k_{13}, of (A6) to solve for k_{12}, of (A1) to solve for k_{22}, of (A8) to solve for k_{01}, of (A3) to solve for k_{00}, of (A9) to solve for k_{03}, of (A7) to solve for k_{02}, of (A19) to solve for k_{33} and of (A20) to solve for k_{23}. These integrations and substitutions are a first step in a longer calculation which finally gives the following general solution. The 13 free constants $c_1 \dots c_{13}$ stand for 13 KTs, i.e. 13 conserved first integrals of geodesic motion in the curved space described by the Kimura metric:

$$\begin{aligned}
k_{00} &= (3b^2r^4t^2c_1 + 48b^2r^4tc_2 + 3br^2c_1 - br^4c_3 - 3br^4t^2c_4 - r^2c_4)/(6b)\\
k_{01} &= (brtc_1 + 8brc_2 - rtc_4)/(2b)\\
k_{02} &= \cos(p)r^4c_{12} + \sin(p)r^4c_{11}\\
k_{03} &= (-\cos(h)^3\cos(p)r^4c_{11} + \cos(h)^3\sin(p)r^4c_{12} - \cos(h)^2\sin(h)r^4c_{13}\\
&\qquad + \cos(h)\cos(p)r^4c_{11} - \cos(h)\sin(p)r^4c_{12} + \sin(h)r^4c_{13})/\sin(h)\\
k_{11} &= -c_4/(3b^2r^2)\\
k_{12} &= 0\\
k_{13} &= 0\\
k_{22} &= (6\cos(p)^2r^4c_6 + 6\cos(p)\sin(p)r^4c_5 - 3b^2r^4c_1t^2 - 48b^2r^4tc_2 + 3br^4t^2c_4\\
&\qquad - 6r^4c_7 - 2r^2c_4)/6\\
k_{23} &= (2\cos(h)^2\cos(p)r^4c_8 - 2\cos(h)^2\sin(p)r^4c_9 + 2\cos(h)\cos(p)^2\sin(h)r^4c_5\\
&\qquad - 2\cos(h)\cos(p)\sin(h)\sin(p)r^4c_6 - \cos(h)\sin(h)r^4c_5 - 2\cos(p)r^4c_8\\
&\qquad + 2\sin(p)r^4c_9)/2\\
k_{33} &= (12\cos(h)^5\cos(p)r^4c_9 + 12\cos(h)^5\sin(p)r^4c_8 + 6\cos(h)^4\cos(p)^2\sin(h)r^4c_6\\
&\qquad + 6\cos(h)^4\cos(p)\sin(h)\sin(p)r^4c_5 - 6\cos(h)^4\sin(h)r^4c_7 + 6\cos(h)^4\sin(h)r^4c_{10}\\
&\qquad - 24\cos(h)^3\cos(p)r^4c_9 - 24\cos(h)^3\sin(p)r^4c_8 - 6\cos(h)^2\cos(p)^2\sin(h)r^4c_6\\
&\qquad - 6\cos(h)^2\cos(p)\sin(h)\sin(p)r^4c_5 + 3\cos(h)^2\sin(h)b^2r^4t^2c_1 - 12\sin(h)r^4c_7\\
&\qquad + 48\cos(h)^2\sin(h)b^2r^4tc_2 - 3\cos(h)^2\sin(h)br^4t^2c_4 - 6\cos(h)^2\sin(h)r^4c_6\\
&\qquad + 18\cos(h)^2\sin(h)r^4c_7 - 12\cos(h)^2\sin(h)r^4c_{10} + 2\cos(h)^2\sin(h)r^2c_4\\
&\qquad + 12\cos(h)\cos(p)r^4c_9 + 12\cos(h)\sin(p)r^4c_8 - 3\sin(h)b^2r^4t^2c_1 + 6\sin(h)r^4c_{10}\\
&\qquad - 48\sin(h)b^2r^4tc_2 + 3\sin(h)br^4t^2c_4 + 6\sin(h)r^4c_6 - 2\sin(h)r^2c_4)/(6\sin(h)).
\end{aligned}$$

What makes this result and the Kimura metric interesting is that two of these KTs are non-trivial, i.e. they are not just symmetrized products of the four KVs of this metric (see, for example, Kimura, 1976).

Received 28 May 1999
Accepted 20 June 2001