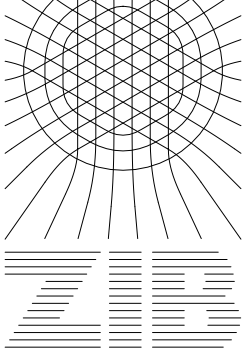
Konrad-Zuse-Zentrum für Informationstechnik Berlin
Heilbronner Str. 10, D-10711 Berlin - Wilmersdorf



Thomas Wolf*

The program CRACK for solving PDEs in General Relativity

Lecture given at the 152. WE-Heraeus-Seminar on
RELATIVITY AND SCIENTIFIC COMPUTING:
Computer Algebra, Numerics, Visualization

* Queen Mary and Westfield College, London und Fellow am Konrad–Zuse–Zentrum

Preprint SC 95–22 (October 1995)

The program CRACK for solving PDEs in General Relativity

Lecture given at the 152. WE-Heraeus-Seminar on RELATIVITY AND SCIENTIFIC COMPUTING: Computer Algebra, Numerics, Visualization

Thomas Wolf
Queen Mary and Westfield College
University of London, London E1 4NS
T.Wolf@maths.qmw.ac.uk

Abstract

In the introduction an approach to solving differential equations is motivated in which non-linear DEs are not attacked directly but properties like infinitesimal symmetries or the existence of an equivalent variational principle are investigated. In the course of such investigations overdetermined PDE-systems are generated which are to be solved (where the term 'overdetermined' just stands for 'more conditions than free functions'). In section 2. algorithms for simplifying and solving overdetermined PDE systems are given together with examples. References for more details of the corresponding program CRACK, written by A. Brand and the author, are given.

In sections 3.-5. applications of the program CRACK are discussed. The first application is the investigation of symmetries of space-time metrics by solving Killing equations for Killing vectors and Killing tensors and their integrability conditions. A program CLASSYM that formulates these equations, written by G. Grebot, is briefly described.

In section 4. an example of the original application of CRACK is discussed which is the determination of symmetries of a PDE system. The problem is to find the symmetries of an unusual unified field theory of gravitational and hadronic interactions.

The application of symmetries with a program APPLYSYM is the content of section 5. where an ODE, resulting from an attempt to generalize Weyl's class of solutions of Einsteins field equations, is solved.

The final section is devoted to future work on, first, making a general PDE-solver more flexible and effective, and secondly, on applying it to more advanced applications. This section contains so far unpublished work. An example requiring the extension of CRACK to deal with non-polynomial non-linearities results from an investigation of interior solutions of Einstein's field equations for a spherically symmetric perfect fluid in shear-free motion by H. Stephani. A possible future application of CRACK is the determination of Killing tensors of higher rank. In the last sub-section an algorithm

for formulating corresponding integrability conditions has been sketched. The maximal number of Killing tensors of rank r in a n-dimensional Riemannian space has been found to be $\frac{1}{r+1}\binom{n+r-1}{r}\binom{n+r}{r}$.

Contents

1 Introduction

Typical applications of computer algebra in classical General Relativity involve mainly the computation of curvature tensors and related invariants and their classification for a given space-time metric. The computer algebra applications to be discussed here concern the solution of differential equations. The usual problem with differential equations in GR is that they are non-linear with no general theory available to solve them. Instead of attacking such equations directly one aims at special properties which the problem might have and which would help to characterize or even to solve it. Special properties could be the possibility of embedding a space into another one or the existence of symmetries which provide first integrals or enable the integration of an ordinary differential equation (ODE) or the transformation of a partial differential equation (PDE) into a simpler form. All such approaches essentially lead to an overdetermined system of differential equations to be solved. By 'overdetermination'

we just mean that more conditions are to be satisfied than there are free functions available. This reflects the fact that the property to be investigated might, but need not be realized.
In order to do a variety of investigations in an effective way, the strategy is to put more emphasis on one program package for solving overdetermined systems of equations and to have relatively straight forward programs to formulate these systems for each individual application. The main purpose of this lecture is therefore to give examples demonstrating the universal character of the tool to encourage its use in similar situations when a smooth differential object is investigated concerning a local property which may, but need not, exist.
In the first part of the lecture an overview of techniques in simplifying and solving DEs is given as they are applied in the program `CRACK` (written by A. Brand and the author) for solving overdetermined DE-systems [25]. Partially, these algorithms can even be useful in cases of hand calculations when no computer is available. A collection of references to other programs which have similar purposes (e.g. symmetry investigation, bringing a DE system into involutive form) is given in [10].
In the remaining sections applications in GR are over-viewed. Other examples, like the determination of Lagrangians for given 2nd order ODEs or the solution of quasilinear first order PDEs will not be given. They can be found in [26].
In section 3. a program `CLASSYM` is described that calculates symmetries of space-times by formulating Killing vector and Killing tensor equations and their integrability conditions which are together passed to `CRACK`.
The following section gives an example for the application of the program `LIEPDE` which also formulates conditions for generators of infinitesimal symmetries, but in this case for symmetries of differential equations. The question to be checked there is whether a field theory has a certain symmetry or not.
The next application is an example for the use of the program `APPLYSYM` which for already calculated symmetries of differential equations determines symmetry and similarity variables and performs the necessary transformations to lower the order of an ODE or to reduce the number of variables of a PDE.
In the final section an outlook for future work is given. The results on integrability conditions and the maximal number of Killing tensors of arbitrary rank have not been reported yet.

2 Contents of `CRACK`

2.1 General remarks

The package `CRACK` attempts the solution of an overdetermined system of ordinary or partial differential equations (ODEs/PDEs) with at most polynomial nonlinearities in the functions to be calculated and their derivatives. Any (differentiable) explicit functions of only the independent variables are allowed.
In the following only a rough overview is given. For details see the manual [25] (see section Availability below). Also for application programs `LIEPDE, LAGRAN, APPLYSYM, ...` check [25].
The input of `CRACK` consists of

- a list of equations in the form of vanishing expressions

- a list of inequalities in a form of non-vanishing expressions which must be satisfied by the solutions
- a list of functions/constants to be calculated and
- a list of independent variables (but only those are necessary on which functions do not already depend).

Returned is a list of solutions which could be an empty list if no solution exists. Each solution contains 4 lists:

- a list of unsolved equations (empty if completely solved)
- a list of computed functions/constants together with the computed expressions they are equal to
- a list of functions/constants which are free or still to be calculated by the unsolved equations
- a list of inequalities for the solution to be valid.

There are about 15 flags which can be used to trace the calculation more or less or to activate or deactivate individual modules. A procedure `CRACKHELP()` lists the flags and their action. The package `CRACK` contains modules for decoupling PDEs, integrating exact PDEs, separating PDEs, solving DEs containing functions of only a subset of all variables and solving standard ODEs. These modules will be described briefly with examples taken from [25].

2.2 Decoupling

The decoupling module differentiates equations and combines them algebraically to obtain, if possible, decoupled and simplified equations of lower order. This is a reduced form of a general algorithm to bring a system of PDEs into a standard form where all integrability conditions are satisfied by applying a finite number of additions, multiplications and differentiations. This is based on the general theory of involutive systems [19, 23, 11]. Later in this school, David Hartley will give a talk about the related program `EDS` working in the geometrical framework of exterior differential systems [7].
Essential to the theory of the coordinate version of this method is a total ordering of partial derivatives which allows assignment to each PDE of a *Leading Derivative* (LD) according to a chosen ordering of functions and derivatives. Examples for possible orderings are

- lex. order of functions > lex. order of variables (adopted by `CRACK` for decoupling)
- lex. order of functions > total differential order > lex. order of variables
- total order > lex. order of functions > lex. order of variables

or mixtures of them by giving weights to individual functions and variables. Above, the '>' indicate "before" in priority of criteria. The principle is then to

- take two equations at a time and differentiate them as often as necessary to get equal LDs,
- regard these two equations as algebraic equations in the common LD and calculate the remainder w.r.t. the LD, i.e. to generate an equation without the LD by the Euclidean algorithm, and
- add this equation to the system.

Usually pairs of equations are taken first, such that only one must be differentiated. If in such a generation step one of the two equations is not differentiated then it is called a simplification step and this equation will be replaced by the new equation.
The algorithm ends if each combination of two equations yields only equations which simplify to an identity modulo the other equations. More detailed descriptions and related programs are given e.g. in [1, 18, 14, 7]. For other programs see also [10].
In `CRACK`, a reduced version of this algorithm has been implemented, which applies the first of the above orderings with lexicographical ordering of functions having the highest priority. This is done to get decoupled equations, i.e. a system with a "triangular dependence" of the equations on the functions, which is usually easier to solve successively (starting with the equation containing the fewest functions) than are coupled DEs. To save memory not all equations are stored but new equations replace in general older ones.
In the lecture will be shown, how starting from

$$f + f_{,yy}\, f_{,x} = 0$$

$$f_{,y} + f_{,x}^2 = 0$$

for $f(x, y)$, in a few steps the necessary and sufficient condition $f = 0$ follows.

2.3 Integrating exact PDEs

The purpose of the integration module is to decide whether a given differential expression D which involves an arbitrary number of unknown functions is a total derivative of another expression I w.r.t. any variable and to invert the total derivative i.e. to find I and to add appropriate functions/constants of integration.
There exists, of course, always a function I with a total derivative equal to D but the question is whether for *arbitrary* functions f^i the integral I is functionally dependent only on the f^i and their derivatives, and *not on integrals of* f^i.
The algorithm consists of a successive partial integration of the term with the highest derivative (which must occur linearly) of the integration variable x of any f^i. By that the differential order w.r.t. x is reduced successively. This procedure is always applicable because steps involve only differentiations and the polynomial integration ($\int h^n \frac{\partial h}{\partial x} dx = h^{n+1}/(n+1)$) where h is a partial derivative of some function f^i.
The iteration stops if no term with any x-derivative of any f^i is left. If in the remaining un-integrated terms any $f^i(x)$ itself occurs, then I is not expressible with f^i and its derivatives only. In case no $f^i(x)$ occurs then any remaining terms can contain x only explicitly.

Whether they can be integrated depends on their formal integrability. For their integration the REDUCE integrator is applied.
Applying this procedure to

$$D := 2f_{,y}\,g' + 2f_{,xy}\,g + gg'^3 + xg'^4 + 3xgg'^2g'' = 0 \tag{1}$$

it will be shown in the lecture how to obtain

$$I := 2fg + xygg_{,x}^{\,3} + c_1(x) + c_2(y) = 0. \tag{2}$$

This basic integration algorithm is enhanced in a number of ways to increase its success rate. If, after applying the above algorithm, only terms with unknown functions depending on fewer variables remain to be integrated then integration is possible at the price of introducing new equations and new functions but of fewer variables. If, for example, the equation to integrate were

$$\tilde{D} = D + g^2(y^2 + x\sin y + x^2e^y) \tag{3}$$

then the result would be

$$\tilde{I} = I + \frac{1}{3}y^3c_3'' - \cos y(xc_3'' - c_3) + e^y(x^2c_3'' - 2xc_3' + 2c_3) \tag{4}$$

with $c_3 = c_3(x)$, $' \equiv d/dx$ and the *single* additional condition $g^2 = c_3'''$. We integrated an equation in two variables at the price of one extra equation of only one variable which overall is a success. This procedure is optimized in so far as it tries to minimize the number of new functions and equations as can be seen in this example.
Another generalization of integration is to recognize the possibility of writing a given differential expression D as $D = A_{,x} + B_{,y}$ with differential expressions A, B to introduce a new function C with $A = C_{,y}$, $B = -C_{,x}$. If these equations can be solved for different unknown functions, then more functions have been solved for than have been introduced.
A different enhancement of integration is the determination of monomial integrating factors, i.e. products of powers of the independent variables and explicit functions of them.

2.4 Separation of PDEs

It is common practice to have routines in programs like `CRACK` which realize when variables occur only explicitly and then do a separation. For example, from

$$0 = f_{,y} + z(f^2 + g_{,x}) + z^2(g_{,x} + yg^2)$$

for $f = f(x,y)$, $g = g(x)$ with independent variables x, y, z should be concluded after a z-separation, $0 = f_{,y} = f^2 + g_{,x} = g_{,x} + yg^2$ and after a further y-separation $0 = g_{,x} = g^2$.
A more difficult case arises when there is no function of all variables but also no variable which occurs only explicitly. In the lecture a related algorithm will be applied to solve

$$0 = f(x)g(y) - \frac{1}{2}xf'(x) - g'(y) - (1 + x^2)y$$

to obtain the necessary and sufficient solutions $f = 1+x^2, g = 1+y$ and $f = -1-x^2, g = 1-y$. The algorithm developed and implemented in `CRACK` handles situations with an arbitrary number of functions of arbitrary variable dependence.

2.5 Solving standard ODEs

For solving standard ODEs the package `ODESOLVE` by MacCallum [15] is applied. Its applicability is increased by recognizing such PDEs in which there is only one unknown function and all occurring derivatives of this function are only derivatives w.r.t. one variable of only one partial derivative. For example, the PDE for $f(x,y)$

$$0 = f_{,xxy} + f_{,xxyy}$$

can be viewed as a first order ODE in y for $f_{,xxy}$.

After having discussed the contents of `CRACK` we now come to applications.

3 The calculation of space-time symmetries

3.1 General remarks

In General Relativity the concept of infinitesimal symmetries plays an important role for a number of reasons.
First, the knowledge of the number of Killing vectors (KVs) and their Lie-algebra is a main criterion for classifying space-times and is very helpful to identify a metric as a coordinate-transformed version of another one. Each infinitesimal symmetry furthermore provides a constant of motion of a point particle (a Killing vector K_i provides the first integral $K_i\dot{u}^i$ and a Killing tensor (KT) K_{ij} gives the first integral $K_{ij}\dot{u}^{ij}$).
The calculation of symmetries is often quite lengthy as e.g. Bonnor remarks in [2] where the metric for a 'photon rocket'

$$\begin{aligned} ds^2 &= (1 - a^2r^2\sin^2\theta - 2ar\cos\theta - 2mr^{-1})du^2 + 2dudr \\ &\quad -r^2(d\theta^2 + \sin^2\theta d\phi^2) - 2ar^2\sin\theta d\theta du \end{aligned} \tag{5}$$

$$-\infty < u < \infty,\ \ 0 < r,\ \ 0 \le \theta \le \pi,\ \ 0 \le \phi \le 2\pi,\ \ m = m(u),\ \ a = a(u)$$

is discussed and its Killing vectors are determined. Although the metric does not include many terms and should be easy to handle, the two functions a, m of u already lead to 3 cases

- $a = 0 \rightarrow$ 3 KVs (spherical sym.),
- $a \neq 0, \dot{a} = 0 = \dot{m} \rightarrow$ 2 KVs (axial- & time-s.),
- else 1 KV (axial-s.)

which are to be treated individually.
The calculation of symmetries proceeds in two steps:

- After loading the computer algebra system SHEEP/CLASSI [16], a program `CLASSYM`, written by G. Grebot (for details see [6]), formulates the symmetry conditions and additional integrability conditions as desired. This program has to be able to do calculations in General Relativity efficiently as higher order derivatives of the Riemann tensor may be required and conditions can become quite lengthy.

- The program `CRACK` for solving the PDE system is loaded into REDUCE and called. REDUCE can be started within RCLASSI or `CLASSYM` can save the overdetermined system in REDUCE format to be used by `CRACK` independently in a REDUCE session afterwards.

On a 100 MHz Pentium PC each of the three cases above needs about 5 sec for the KVs to be determined.

3.2 Generation of the conditions

Symmetry conditions can be formulated for the determination of KVs, homothetic KVs, conformal KVs and symmetric rank two KTs. The metric has to be input in CLASSI format using a frame (orthonormal, null, ...) or a coordinate notation. The equations for a Killing vector K are

$$\begin{aligned} 0 &= K_{a;b} + K_{b;a} - 2cg_{ab} \\ c &= K_{a;b}g^{ab}/4 \end{aligned}$$

where g_{ab} is the metric and c the conformal factor which is constant for homothetic KVs and zero for normal KVs.
The program `CRACK` that is to solve the symmetry conditions will add integrability conditions to the PDE-system if necessary, but it will not go for geometrically meaningful conditions which are more likely to be short and useful. The reason is that it is a general purpose tool which knows nothing about the geometric background of the problem. Therefore, already at the first stage when the symmetry conditions are formulated, one has the option of adding up to 5 integrability conditions to the (homothetic, conformal) KV equations. They are

$$\begin{aligned} K_{[a;b]c} &= R_{abcd} + c_{,b}\, g_{ca} - c_{,a}\, g_{bc} \\ 0 &= \mathcal{L}_K R_{abcd} - 2cR_{abcd} \\ 0 &= \mathcal{L}_K R_{abcd;e} - 2cR_{abcd;e} \\ 0 &= \mathcal{L}_K R_{abcd;ef} - 2cR_{abcd;ef} \\ 0 &= \mathcal{L}_K R_{abcd;efg} - 2cR_{abcd;efg} \end{aligned}$$

and similar ones for tetrad coordinates. The individual integrability conditions selected can be contracted at wish. Then all possible contractions with the metric and Ricci tensor and with a vector, which can be specified, are appended to the system as additional equations.
Using the formalism of Hauser & Malhiot [9] the Killing tensor equations

$$0 = K_{(ab;c)} \tag{6}$$

are completed by two integrability conditions (10),(11), formulated with the help of two tensors **L**,**M**:

$$L_{abc} = K_{bc;a} - K_{ac;b} \tag{7}$$

$$
\begin{aligned}
M_{abcd} &= \frac{1}{8}\left(\partial^{pq}_{ab}\partial^{rs}_{cd} + \partial^{rs}_{ab}\partial^{pq}_{cd}\right) L_{pqr;s} && (8)\\
K_{ab;c} &= \frac{1}{3}(L_{cab} + L_{cba}) && (9)\\
L_{abc;d} &= \partial^{pq}_{ab}\left(\frac{5}{8}R_{pqdm}K^{m}{}_{c} + \frac{3}{8}R_{pqcm}K^{m}{}_{d}\right.\\
&\quad \left. + \frac{3}{4}R_{dqcm}K^{m}{}_{p} + \frac{1}{4}R_{cqdm}K^{m}{}_{p}\right) + M_{abcd} && (10)\\
M_{abcd;e} &= \left(\partial^{pq}_{ab}\partial^{rs}_{cd} + \partial^{rs}_{ab}\partial^{pq}_{cd}\right)\left(\frac{1}{8}R_{pqrs;m}K^{m}{}_{e} + \frac{1}{2}R_{pqre;m}K^{m}{}_{s}\right.\\
&\quad -\frac{3}{8}R_{pqrm;e}K^{m}{}_{s} - \frac{1}{4}R_{pqe}{}^{m}L_{rsm} + \frac{1}{3}R_{per}{}^{m}L_{qms}\\
&\quad \left. + \frac{1}{3}R_{pre}{}^{m}L_{sqm} + \frac{1}{24}R_{pqr}{}^{m}(5L_{ems} + 7L_{sem})\right) && (11)
\end{aligned}
$$

L_{abc} has the symmetries $L_{abc} = L_{[ab]c}$, $L_{[abc]} = 0$ and M_{abcd} has the same symmetries as the Riemann tensor. Therefore there are maximally 10 independent components for K_{ab}, 20 for L_{abc} and 20 for M_{abcd} which add up to 50 which is the maximum number of Killing tensors, obtained by space-times with constant curvature. Equations (10),(11) are equally valid for the traceless part of K_{ab}. To formulate integrability conditions one does not have to use these tensors L, M but they are well suited to the problem as can be seen from the solution for flat space [8] by integrating (9)-(11)

$$
\begin{aligned}
M_{abcd} &= A_{abcd} && (12)\\
L_{abc} &= B_{abc} + A_{abcd}x^{d} && (13)\\
K_{bc} &= S_{bc} + \frac{2}{3}B_{a\,(bc)}x^{a} + \frac{1}{3}A_{abcd}x^{a}x^{d}. && (14)
\end{aligned}
$$

If the metric is given in frame components then `CLASSYM` can also formulate the Killing tensor equations in a spinorial form [9]. In a null tetrad k, l, m, m^{*} with $k \cdot l = m \cdot m^{*} = 1$

- the tensor K has 4 real components $K_{kk}, K_{kl}, K_{ll}, K_{mm^{*}}$ and 3 independent complex components K_{km}, K_{lm}, K_{mm}
- L can be expressed in terms of 2 real components $L_{ka}{}^{a}, L_{la}{}^{a}$ and remaining 9 fields
- Because M has the symmetries of the Riemann tensor it can be decomposed into 5 complex fields analogous to the Newman-Penrose components $\psi_0, \ldots, \psi_4$ of the Weyl tensor, the null tetrad components of an analog of the traceless part of the Ricci tensor, and an analog of the curvature scalar.

3.3 Post processing of results

If the symmetry conditions had been solved, a question in connection with KVs is which Lie algebra the symmetry group obeys. A corresponding program `LIEALG` finds the structure constants of the Lie algebra and calls a program of F. Schöbel [20] for classifying Lie algebras up to dimension four.

When all Killing tensors (symmetric, of rank 2) have been determined then a remaining task is to filter out all the irreducible components. The KT equations (6) are always satisfied by the metric tensor g_{ij} and all bivectors $A_{(i}B_{j)}$ where A_i and B_j are KVs. A program `RDUKT` generates all reducible Killing tensors of rank two from the solution of the Killing vector equations. These reducible KTs are used by a program `IRRKT` to find the irreducible content of the general solution of the Killing tensor equations.

3.4 An Example

The Killing equations for the Kimura-metric [12] together with a number of integrability conditions are:

```
DEPEND V0,T,R,H,P$
DEPEND V1,T,R,H,P$
DEPEND V2,T,R,H,P$
DEPEND V3,T,R,H,P$
LISTOFFUNS := LIST(V0,V1,V2,V3)$
KILEQS := LIST(2*DF(V0,T)+2*R**(-1)*V1,B**(1/2)*R**2*DF(V0,R)-B**(-1/2)*R**(-2)*DF(V1,T)
,-B**(1/2)*DF(V2,T)+B**(-1/2)*DF(V0,H),-B**(1/2)*SIN(H)*DF(V3,T)+B**(-1/2)*(SIN( H))**(-1
)*DF(V0,P),-2*DF(V1,R)+2*R**(-1)*V1,-B*R**2*DF(V2,R)-B**(-1)*R**(-2)*DF(V1,H),-B*R**2*SIN(H
)*DF(V3,R)-B**(-1)*R**(-2)*(SIN(H))**(-1)*DF(V1,P),-2*DF(V2,H)-2*R**(-1)*V1,-SIN(H)*DF(V3,
H)-(SIN(H))**(-1)*DF(V2,P),-2*V2*COS(H)*(SIN(H))**(-1)-2*DF(V3,P)-2*R**(-1)*V1,1/2*B*R*DF(
V0,T,R)+B*DF(V0,T)+B*R**(-1)*V1+1/2*R**(-3)*DF(V1,T,2),1/2*B**(3/2)*R**3*DF(V0,R,2)+2*B
**(3/2)*R**2*DF(V0,R)+1/2*B**(1/2)*R**(-1)*DF(V1,T,R)-B**(1/2)*R**(-2)*DF(V1,T),-1/2*B**
(3/2)*DF(V2,T)+1/2*B**(1/2)*R*DF(V0,R,H)+1/2*B**(1/2)*DF(V0,H)+1/2*B**(-1/2)*R**(-3)*DF(V1
,T,H),-1/2*B**(3/2)*SIN(H)*DF(V3,T)+1/2*B**(1/2)*R*(SIN(H))**(-1)*DF(V0,R,P)+1/2*B**(1/2
)*(SIN(H))**(-1)*DF(V0,P)+1/2*B**(-1/2)*R**(-3)*(SIN(H))**(-1)*DF(V1,T,P),-1/2*B**2*R**2*
DF(V2,R)+1/2*B*R**(-1)*DF(V2,T,2)+1/2*R**(-1)*DF(V0,T,H)+1/2*R**(-2)*DF(V1,H),1/2*B**(3/
2)*R*DF(V2,T,R)+1/2*B**(1/2)*R*DF(V0,R,H),1/2*B**(3/2)*R**2*DF(V0,R)+1/2*B**(1/2)*R**(-1)
*DF(V2,T,H)+1/2*B**(1/2)*R**(-2)*DF(V1,T)+1/2*B**(-1/2)*R**(-1)*DF(V0,H,2),-1/2*B**(1/2)
*R**(-1)*COS(H)*DF(V3,T)+1/2*B**(1/2)*R**(-1)*(SIN(H))**(-1)*DF(V2,T,P)-1/2*B**(-1/2)*R**
(-1)*COS(H)*(SIN(H))**(-2)*DF(V0,P)+1/2*B**(-1/2)*R**(-1)*(SIN(H))**(-1)*DF(V0,H,P),-1/2*B
**2*R**2*SIN(H)*DF(V3,R)+1/2*B*R**(-1)*SIN(H)*DF(V3,T,2)+1/2*R**(-1)*(SIN(H))**(-1)*DF(V0
,T,P)+1/2*R**(-2)*(SIN(H))**(-1)*DF(V1,P),1/2*B**(3/2)*R*SIN(H)*DF(V3,T,R)+1/2*B**(1/2)*
R*(SIN(H))**(-1)*DF(V0,R,P),1/2*B**(1/2)*R**(-1)*COS(H)*DF(V3,T)+1/2*B**(1/2)*R**(-1)*SIN
(H)*DF(V3,T,H)-1/2*B**(-1/2)*R**(-1)*COS(H)*(SIN(H))**(-2)*DF(V0,P)+1/2*B**(-1/2)*R**(-1)
*(SIN(H))**(-1)*DF(V0,H,P),1/2*B**(3/2)*R**2*DF(V0,R)+1/2*B**(1/2)*R**(-1)*COS(H)*(SIN(H))
**(-1)*DF(V2,T)+1/2*B**(1/2)*R**(-1)*DF(V3,T,P)+1/2*B**(1/2)*R**(-2)*DF(V1,T)+1/2*B**(-1
/2)*R**(-1)*COS(H)*(SIN(H))**(-1)*DF(V0,H)+1/2*B**(-1/2)*R**(-1)*(SIN(H))**(-2)*DF(V0,P,2)
,1/2*B**(3/2)*R*DF(V2,T,R)+1/2*B**(3/2)*DF(V2,T)-1/2*B**(1/2)*DF(V0,H)-1/2*B**(-1/2)*R**(
-3)*DF(V1,T,H),1/2*B**2*R**3*DF(V2,R,2)+2*B**2*R**2*DF(V2,R)-1/2*R**(-1)*DF(V1,R,H)+R**(-2
)*DF(V1,H),1/2*B*R*DF(V2,R,H)+B*DF(V2,H)+B*R**(-1)*V1-1/2*B**(-1)*R**(-3)*DF(V1,H,2),-1/2*B
*R*COS(H)*DF(V3,R)+1/2*B*R*(SIN(H))**(-1)*DF(V2,R,P)+1/2*B*SIN(H)*DF(V3,H)+1/2*B*(SIN(H))
**(-1)*DF(V2,P)+1/2*B**(-1)*R**(-3)*COS(H)*(SIN(H))**(-2)*DF(V1,P)-1/2*B**(-1)*R**(-3)*(SIN(
H))**(-1)*DF(V1,H,P),1/2*B**(3/2)*R*SIN(H)*DF(V3,T,R)+1/2*B**(3/2)*SIN(H)*DF(V3,T)-1/2*B
**(1/2)*(SIN(H))**(-1)*DF(V0,P)-1/2*B**(-1/2)*R**(-3)*(SIN(H))**(-1)*DF(V1,T,P),1/2*B**2*
R**3*SIN(H)*DF(V3,R,2)+2*B**2*R**2*SIN(H)*DF(V3,R)-1/2*R**(-1)*(SIN(H))**(-1)*DF(V1,R,P)+R
**(-2)*(SIN(H))**(-1)*DF(V1,P),1/2*B*R*COS(H)*DF(V3,R)+1/2*B*R*SIN(H)*DF(V3,R,H)+1/2*B*SIN
(H)*DF(V3,H)+1/2*B*(SIN(H))**(-1)*DF(V2,P)+1/2*B**(-1)*R**(-3)*COS(H)*(SIN(H))**(-2)*DF(V1
,P)-1/2*B**(-1)*R**(-3)*(SIN(H))**(-1)*DF(V1,H,P),1/2*B*R*COS(H)*(SIN(H))**(-1)*DF(V2,R)+1
/2*B*R*DF(V3,R,P)+B*V2*COS(H)*(SIN(H))**(-1)+B*DF(V3,P)+B*R**(-1)*V1-1/2*B**(-1)*R**(-3)*
COS(H)*(SIN(H))**(-1)*DF(V1,H)-1/2*B**(-1)*R**(-3)*(SIN(H))**(-2)*DF(V1,P,2),B**(1/2)*R**
(-1)*COS(H)*DF(V3,T)+1/2*B**(1/2)*R**(-1)*SIN(H)*DF(V3,T,H)-1/2*B**(1/2)*R**(-1)*(SIN(H))**
(-1)*DF(V2,T,P),B*R*COS(H)*DF(V3,R)+1/2*B*R*SIN(H)*DF(V3,R,H)-1/2*B*R*(SIN(H))**(-1)*DF(V2,R
,P),1/2*B**2*R**2*SIN(H)*DF(V3,R)+3/2*R**(-1)*COS(H)*DF(V3,H)+1/2*R**(-1)*COS(H)*(SIN(H))**
(-2)*DF(V2,P)+1/2*R**(-1)*SIN(H)*DF(V3,H,2)-1/2*R**(-1)*(SIN(H))**(-1)*DF(V2,H,P)-1/2*R**(
-2)*(SIN(H))**(-1)*DF(V1,P),-1/2*B**2*R**2*DF(V2,R)-R**(-1)*V2+R**(-1)*COS(H)*(SIN(H))**(-
1)*DF(V3,P)+1/2*R**(-1)*DF(V3,H,P)-1/2*R**(-1)*(SIN(H))**(-2)*DF(V2,P,2)+1/2*R**(-2)*DF(V1
,H),-2*B**2*DF(V0,T)-2*B**2*DF(V1,R),-B**3*R**2*DF(V2,R)-B*R**(-2)*DF(V1,H),-B**3*R**2*SIN
(H)*DF(V3,R)-B*R**(-2)*(SIN(H))**(-1)*DF(V1,P),-B**(5/2)*DF(V2,T)+B**(3/2)*DF(V0,H),-B**(
5/2)*SIN(H)*DF(V3,T)+B**(3/2)*(SIN(H))**(-1)*DF(V0,P),-2*B**2*DF(V0,T)-2*B**2*DF(V2,H)-4
*B**2*R**(-1)*V1,-B**2*SIN(H)*DF(V3,H)-B**2*(SIN(H))**(-1)*DF(V2,P),-B**(5/2)*R**2*DF(V0,R
)+B**(3/2)*R**(-2)*DF(V1,T),-B**(5/2)*SIN(H)*DF(V3,T)+B**(3/2)*(SIN(H))**(-1)*DF(V0,P)+B
```

```
**(1/2)*R**(-2)*SIN(H)*DF(V3,T),-2*B**2*V2*COS(H)*(SIN(H))**(-1)-2*B**2*DF(V0,T)-2*B**2*DF
(V3,P)-4*B**2*R**(-1)*V1,-B**(5/2)*R**2*DF(V0,R)+B**(3/2)*R**(-2)*DF(V1,T),B**(5/2)*DF(V2
,T)-B**(3/2)*DF(V0,H)-B**(1/2)*R**(-2)*DF(V2,T),2*B**2*DF(V1,R)+2*B**2*DF(V2,H),B**2*SIN(H)
*DF(V3,H)+B**2*(SIN(H))**(-1)*DF(V2,P),-B**3*R**2*SIN(H)*DF(V3,R)+B*SIN(H)*DF(V3,R)-B*R**
(-2)*(SIN(H))**(-1)*DF(V1,P),2*B**2*V2*COS(H)*(SIN(H))**(-1)+2*B**2*DF(V1,R)+2*B**2*DF(V3,
P),B**3*R**2*DF(V2,R)-B*DF(V2,R)+B*R**(-2)*DF(V1,H),2*B**2*V2*COS(H)*(SIN(H))**(-1)+2*B**2
*DF(V2,H)+2*B**2*DF(V3,P)+4*B**2*R**(-1)*V1-2*R**(-2)*V2*COS(H)*(SIN(H))**(-1)-2*R**(-2)*DF
(V2,H)-2*R**(-2)*DF(V3,P)-2*R**(-3)*V1)$
SOL:=CRACK(KILEQS,LIST(),LISTOFFUNS,LIST())$
```

On a SUN SPARC II CRACK finds after 45 sec that for $b \neq 0$ this space-time has the 4 Killing vectors

$$\partial_T, \quad \partial_P, \quad -\cos(P)\partial_H + \cot(H)\sin(P)\partial_P, \quad \sin(P)\partial_H + \cot(H)\cos(P)\partial_P$$

and therefore is spherically symmetric and time independent.
Often some symmetries can easily be spotted but it is more difficult to show that there are no additional symmetries. Although having a length of 5 kByte, this example is one of the smaller cases. Killing tensor equations together with integrability conditions easily reach 50 - 100 kByte in length.

4 Symmetries of a field theory

In this section an example for the application of the program LIEPDE is given which formulates symmetry conditions for DE-systems and then calls CRACK to solve them.
The following PDE is a field equation for the axial vector component of the torsion in a vacuum within an unusual unified field theory of gravitational and hadronic interactions (see [4] for details). In

$$\Delta K - d\delta K/2 + *(KdK) = 0$$

K is a 1-form, d is the exterior derivative, $*$ is the Hodge dual operator, and δ is the adjoint of d under $*$. The first two terms are invariant under the conformal group of transformations and the question is whether the new non-linear third term violates this symmetry.
Using the REDUCE package EXCALC by E. Schrüfer, D. Hartley brought the above system into coordinate notation such that it can be understood by LIEPDE [17]:

$$\begin{aligned} 0 &= 3k^t{}_{,tt} - 2k^t{}_{,xx} - 2k^t{}_{,yy} - 2k^t{}_{,zz} - k^x{}_{,tx} - k^y{}_{,ty} - k^z{}_{,tz} \\ &\quad - 2k^x{}_{,y}\,k^z + 2k^x{}_{,z}\,k^y - 2k^y{}_{,z}\,k^x + 2k^y{}_{,x}\,k^z - 2k^z{}_{,x}\,k^y + 2k^z{}_{,y}\,k^x \\ 0 &= k^t{}_{,tx} - 3k^x{}_{,xx} - 2k^x{}_{,yy} - 2k^x{}_{,zz} - k^y{}_{,xy} - k^z{}_{,xz} + 2k^x{}_{,tt} \\ &\quad - 2k^t{}_{,y}\,k^z + 2k^t{}_{,z}\,k^y + 2k^y{}_{,t}\,k^z - 2k^z{}_{,t}\,,k^y - 2k^y{}_{,z}\,k^t + 2k^z{}_{,y}\,k^t \end{aligned} \tag{15}$$

with 2 more equations (x, y, z-permutations of (15)). As reported in [17], an older version of CRACK needed 3/4 h to do a Lie-point-symmetry analysis where the symmetry generators had still to be specialized. The recent version (July 95) takes 4 min on a 100 MHz Pentium to do a general point-symmetry analysis (formulation + solution of the symmetry conditions).

The system is found to have the 11 symmetries

$$
\begin{array}{ll}
x\partial_t + t\partial_x - k^x\partial_{k^t} - k^t\partial_{k^x}\,, & y\partial_x - x\partial_y + k^y\partial_{k^x} - k^x\partial_{k^y}\,,\\
y\partial_t + t\partial_y - k^y\partial_{k^t} - k^t\partial_{k^y}\,, & z\partial_y - y\partial_z + k^z\partial_{k^y} - k^y\partial_{k^z}\,,\\
z\partial_t + t\partial_z - k^z\partial_{k^t} - k^t\partial_{k^z}\,, & x\partial_z - z\partial_x + k^x\partial_{k^z} - k^z\partial_{k^x}\,,\\
t\partial_t + x\partial_x + y\partial_y + z\partial_z - k^t\partial_{k^t} - k^x\partial_{k^x} - k^y\partial_{k^y} - k^z\partial_{k^z}\,, & \\
\partial_t\,,\ \ \partial_x\,,\ \ \partial_y\,,\ \ \partial_z &
\end{array}
$$

which shows that the conformal symmetry is preserved.

5 Applying symmetries of differential equations

Two ways to apply known infinitesimal symmetries of differential equations are a) to generalize a known special solution by one free parameter for each known symmetry or b) to calculate symmetry and similarity variables and to transform the DE. The second method effectively lowers the order of an ODE by one or reduces the number of variables of a PDE to obtain special solutions.
The following ODE for $h = h(\rho)$ resulted from an attempt to generalize Weyl's class of solutions of Einstein's field equations ([13])

$$0 = 3\rho^2 h\, h'' - 5\rho^2\, h'^2 + 5\rho\, h\, h' - 20\rho\, h^3 h' - 20\, h^4 + 16\, h^6 + 4\, h^2. \tag{16}$$

where $' = d/d\rho$. Calling `LIEPDE` through

```
depend h,r;
prob:={{-20*h**4+16*h**6+3*r**2*h*df(h,r,2)+5*r*h*df(h,r)
-20*h**3*r*df(h,r)+4*h**2-5*r**2*df(h,r)**2}, {h}, {r}};
sym:=liepde(prob,{0,nil,nil});
```

gives the two symmetries $-\rho^3\partial_\rho + h\rho^2\,\partial_h$ and $\rho\partial_\rho$. Corresponding finite transformations can be calculated with `APPLYSYM` through

```
newde:=APPLYSYM(de,rest sym);
```

If in the following interactive session (for details see [27]) the user wants to find symmetry- and similarity variables and specifies a linear combination of these two symmetries or one of them (to get the result below the first symmetry is used) and answers several times with 'yes' to the choice the program offers then `APPLYSYM` returns the finite transformation

$$\rho = (2\,u)^{-1/2}, \qquad h = (2\,u)^{1/2}\, v \tag{17}$$

and the new ODE

$$0 = 3u''v - 16u'^3v^6 - 20u'^2v^3 + 5u' \tag{18}$$

where $u = u(v)$ and $' = d/dv$.
Using one symmetry we reduced the 2. order ODE (16) to a first order ODE (18) for u' plus one integration. The second symmetry can be used by hand to reduce the remaining ODE to

an integration too by introducing a variable w through $v^3 d/dv = d/dw$, i.e. $w = -1/(2v^2)$. With

$$p = du/dw \tag{19}$$

the remaining ODE is

$$0 = 3\,w\,\frac{dp}{dw} + 2\,p\,(p+1)(4\,p+1)$$

with the solution

$$\tilde{c}w^{-2}/4 = \tilde{c}v^4 = \frac{p^3(p+1)}{(4\,p+1)^4}, \quad \tilde{c} = const.$$

Writing (19) as $p = v^3(du/dp)/(dv/dp)$ we get u by integration and with (17) finally a parametric solution for ρ, h:

$$\begin{aligned} \rho &= \left(\frac{3c_1^2(2p-1)}{p^{1/2}(p+1)^{1/2}} + c_2\right)^{-1/2} \\ h &= \frac{(c_2 p^{1/2}(p+1)^{1/2} + 6c_1^2 p - 3c_1^2)^{1/2} p^{1/2}}{c_1(4p+1)} \end{aligned}$$

where $c_1, c_2 = const.$ and $c_1 = \tilde{c}^{1/4}$.

6 Future work

6.1 General remarks

The contents of this and the next subsection had not been published yet. They are two more examples for potential applications of a program solving overdetermined DE-systems.
Future work can be grouped into two categories. The first is to make the solution process of differential equations more efficient by putting more intelligence/algorithms into solving DE-systems, e.g. enabling the program to find appropriate coordinate transformations. A further need is, to be able to deal not only with polynomially non-linear equations. An example where these two extensions would be necessary is given in the next subsection.
The other category is to try more advanced applications of the program; one example is given in the last subsection.

6.2 Extending capabilities in dealing with DEs

A method to derive new interior solutions of Einstein's field equations for a spherically symmetric perfect fluid in shear-free motion has been given by H. Stephani in [21]. As described there in more detail, it relies on finding explicit solutions of the innocent looking, but non-linear, ODE

$$L_{,xx} = L^2 F(x), \tag{20}$$

where $F(x)$ is an arbitrary function which can be chosen freely (it will determine the equation of state of the matter but this shall not keep us for now from choosing any appropriate one) and $L = L(x)$ is the function to calculate which has to depend on at least one constant of

integration (which in the further process is taken to be an arbitrary function of time t). To solve (20) completely up to integrations with Lie's method of infinitesimal symmetries, at least two symmetries have to be found. As described in [21], the condition on F that at least one point symmetry exists, is

$$0 = 4(gx+k)(y')^{5/2} - \left(\frac{y}{y'}\right)''' \tag{21}$$

for $y = y(x)$, where $' = d/dx$, g, k are free constants and $F = \int (y')^{5/2} dx$. So far, we have not got an overdetermined problem. This results when we demand that (20) should have two symmetry generators which then can be chosen to satisfy $[X_1, X_2] = X_1$. Translating this condition into a condition for y gives a second ODE ([22]):

$$0 = 4c(y')^{5/2} - \left(\frac{1}{y'}\right)''' \tag{22}$$

with the free constant c. The question now is which common solutions do (21),(22) have other than $y = const.$ Because `CRACK` is currently only applicable to polynomial non-linearities, (21),(22) have to be squared and then the calculation (solving (22) for $y^{(4)}$ and substituting it, solving (21) for $y^{(3)}$ and substituting it ...) blows up expressions very quickly. Furthermore, due to squaring, only necessary consequences would be derived, not sufficient ones.
After a substitution $y = (gw + gx + k)/c$ with $w = w(x)$ in (21), (22), which is not recognized automatically so far, the expressions shrink. Unexpectedly, a single first order ODE results which is equivalent to (21),(22) (with $g, k, c \neq 0$) which does not contain x explicitly and therefore reduces to an integration. After a change of the constant parameters $g, c \to r, s, t$: $c = t^{14}$, $g = s^2$, $s^{-7}t^{35} = r^3$ and a rescaling $wr \to w$, $xr \to x$ the problem has the form

$$0 = w^3 z^3 - 12z^2 + 4 \tag{23}$$

$$z = \sqrt{w' + 1}. \tag{24}$$

Solving $z(w)$ from (23) and $dw/dx = w'(w)$ from (24) leaves finally the integral

$$\int \frac{dw}{w'(w)} = x + const \tag{25}$$

to be calculated which unfortunately is not expressible in elementary functions. For some special values of parameters g, k, c similar calculations give

$$g = 0 \neq c \;\; \to \;\; y = \left(\left(1008c^5 x + p\right)^{1/7} + k \right)/c, \quad p = const.$$

$$g = c = k = 0 \;\; \to \;\; y \text{ is solution of } y^{(3)} = \frac{3y''^2}{2y'}$$

$$\text{which is } y = \begin{cases} px + q \\ 1/(px+q) + r \end{cases} \quad p, q, r = const.$$

A more detailed analysis of the results obtained will be published elsewhere.

6.3 Advanced symmetry investigations

6.3.1 Prolongation of symmetry conditions

Although integrability conditions to the Killing vector and Killing tensor equations that are formulated in the current form of the program `CLASSYM` are quite helpful for solving the full system automatically, these integrability conditions are higher order equations for $\mathbf{K}$ and are rather long. An alternative way would be, for example, to look at the Killing tensor equations (9) and integrability conditions (10),(11) as a first order system for the components of the tensors $\mathbf{K}, \mathbf{L}, \mathbf{M}$. Although both ways of looking at it seem to be identical and substituting $\mathbf{L}$ from (7) and $\mathbf{M}$ from (8) into (9)-(11) convert one form into the other, both are not identical from a computational point of view.
For example, in the simpler case of two functions $k(x), m(x)$ and two ODEs $m = D1(k),\ \ 0 = D2(m)$ with differential expressions $D1(k), D2(m)$, the best strategy is to solve first $0 = D2(m)$ and then $m = D1(k)$ and not to substitute m first and then to solve the much harder problem $0 = D2(D1(k))$. To make matters not too simple, if on the other hand there is a further equation $0 = D3(k)$ then it would make sense to investigate the system $0 = D2(D1(k)),\ 0 = D3(k)$. The question in which order to do integrations and substitutions is therefore a global question of the structure of the whole DE system and not only a question of the single computational step to be done next.
In any case, having equations given in the form (7)-(11) for the components of $\mathbf{K}, \mathbf{L}, \mathbf{M}$ leaves all options open why such a formulation should be preferred. The price for more flexibility is to formalize and implement 'higher substitution skills' which still has to be done. The simplification of working with $\mathbf{K}, \mathbf{L}, \mathbf{M}$ is best seen when many curvature tensor components vanish. For the flat case the equations are integrated instantly ((12)-(14) for rank two, as given in [8], and [28] for arbitrary rank).

6.3.2 Killing tensors of arbitrary rank

The problem of going up in the rank of Killing tensors to be determined is of interest in its own but is also of practical interest. One should expect a strong increase of complexity of the calculational effort going from Killing tensors of rank two to rank three but applications are not restricted to 4-dimensional space-time symmetries. For example, symmetry investigations of van Elst [5] with 2 and 3-dimensional Jacobi metrics h_{ij} defining a purely kinetic Lagrangian $L = h_{ij}\dot{x}^i\dot{x}^j$ which have been done with `CLASSYM` and `CRACK` are candidates to be repeated with Killing tensors of higher rank.
For a symmetric tensor $K_{i_1\ldots i_r}$ of rank r to be a Killing tensor and to describe first integrals $K_{i_1\ldots i_r}\dot{x}^{i_1}\ldots\dot{x}^{i_r}$ of geodesic motion, it has to satisfy

$$K_{(i_1\ldots i_r;i_{(r+1)})} = 0. \tag{26}$$

An equivalent formulation of (26) together with integrability conditions in the form of structural equations [8]

$$(F_A)_{;k} = \sum_B \Gamma_{kAB} F_B \tag{27}$$

is desirable because (27) are homogeneous linear first order PDEs expressing *all* first order derivatives of *all* functions F_A through expressions algebraic in these functions. Also, integrability conditions of (27) are easily obtained by calculating $(F_A)_{;[kl]}$ at first from (27) with first

order derivatives substituted through (27) and secondly using the RICCI identity involving the Riemann tensor $\mathbf{R}$

$$2W_{i_1\ldots i_p;[cd]} = W_{bi_2\ldots i_p}R^b{}_{i_1cd} + \ldots + W_{i_1\ldots i_{(p-1)}b}R^b{}_{i_pcd}. \tag{28}$$

For Killing vectors the structural equations take the well known form

$$\begin{aligned} K_{i;j} &= \omega_{ij} \\ \omega_{ij;k} &= K_m R^m{}_{kji} \end{aligned}$$

with the Killing vector K_i and the antisymmetric Killing bivector $\omega_{ij} = -\omega_{ji}$ being the F_A of (27).
For the case of rank two Killing tensors Collinson [3] derived integrability conditions but as higher order equations for $\mathbf{K}$. As shown above, Hauser and Malhiot [8],[9] introduced new tensors L, M to formulate structural equations for rank two Killing tensors, equ. (7)-(11). Their definition is especially well suited to do investigations in a null tetrad.
An algorithm for the formulation of structural equations for higher rank Killing tensors can be formulated (see [28]) by taking the F_A to be the components of $\mathbf{K}$ and its symmetrized covariant derivatives of order up to r. By proving that $K_{i_1\ldots i_r;(i_{(r+1)}\ldots i_{(2r+1)})}$ can be expressed as a linear combination of $K_{i_1\ldots i_r}$, $K_{i_1\ldots i_r;i_{(r+1)}}$, $\ldots$, $K_{i_1\ldots i_r;i_{(r+1)}\ldots i_{(2r-1)}}$ it can be shown that the resulting system of structural equations is finite with $Z_n^r := \frac{1}{r+1}\binom{n+r-1}{r}\binom{n+r}{r}$ many equations and as many functions F_A and at most as many solutions, which is obtained in flat space (see [28]). Each solution corresponds to a rank r Killing tensor - a non-reducible one or a reducible one, e.g. a symmetrized product of Killing vectors. For $r = 1, 2$, Z_n^r reduces to known numbers which for $r = 2$ is given e.g. in [24]. For rank 3 Killing tensors in space time we get $Z_4^3 = 175$.

7 Availability

The programs run under `REDUCE 3.5` or later versions and are available by anonymous ftp from euclid.maths.qmw.ac.uk (138.37.80.16) from the directories `pub/crack` and `pub/classym`.

8 Acknowledgment

The author acknowledges gratefully the support as Zuse-fellow obtained from the Konrad-Zuse-Zentrum Berlin, especially repeated discussions with Herbert Melenk and Winfried Neun.

References

[1] Bocharov, A.V. and Bronstein, M.L., *Efficiently Implementing Two Methods of the Geometrical Theory of Differential Equations: An Experience in Algorithm and Software Design*, Acta. Appl. Math. 16 (1989) 143–166.

[2] Bonnor, B., The photon rocket, Class. Quantum Grav. 11 (1994) 2007-2012.

[3] Collinson, C. D., *Special quadratic first integrals of geodesics*, J. Phys. A: Gen. Phys., **4**, (1971), 756-760.

[4] Drechsler, W. D., Class. Quant. Grav. 6 (1989) 623-657.

[5] van Elst, H., *Jacobi Metric for the Bartnik/McKinnon SU(2)-EYM Field* Gen. Rel. Grav. **25** (1993), 1295.

[6] Grebot, G., *Automatic symmetry investigation in GR:* `CLASSYM`*, an utility for* `CRACK`, preprint (1995), obtainable together with `CLASSYM`.

[7] Hartley, D., *Causal structure and integrability in moving frames using REDUCE*, talk given at the 152. WE-Heraeus-Seminar on RELATIVITY AND SCIENTIFIC COMPUTING, to be published in the same book as this article by Springer.

[8] Hauser, I. and Malhiot, R. J., *Structural equations for Killing tensors of order two. I* J. Math. Phys. **16**, No. 1 (1975), 150-152.

[9] Hauser, I. and Malhiot, R. J., *Structural equations for Killing tensors of order two. II* J. Math. Phys. **16**, No. 8 (1975), 1625-1629.

[10] Hereman, W., *Volume 3 of CRC Handbook of Lie Group Analysis of Differential Equations*, Ed. by N.H. Ibragimov, CRC Press (1995).

[11] Janet, M., *Leçons sur les systèmes d'équations aux dérivées*, Gauthier–Villars, Paris (1929).

[12] Kimura, M., *On quadratic first integrals in static spherically symmetric spare-times, having spacial parts of non-constant curvature, I.*, Tensor, N. S., **30** (1976), 27-43.

[13] Kubitza, M., FSU Jena, private communication

[14] Mansfield, E., *ncdg: a MAPLE package for analysing systems of PDE referred to a moving frame*, Department of Mathematics, University of Exeter preprint M94/39 (1994).

[15] MacCallum, M. A. H., *An Ordinary Differential Equation Solver for REDUCE*, Proc. ISAAC'88, Springer Lect. Notes in Comp Sci. 358, 196-205.

[16] MacCallum, M.A.H. and Skea, J.E.F., *SHEEP: a Computer Algebra System for General Relativity*, Lectures given at the First Brazilian School for Computer Algebra, CBPF Rio de Janeiro (1989).

[17] Melenk, H., The Complexity Barrier in REDUCE a Case Study, Technical Report TR 94-6 (1994) Konrad-Zuse-Zentrum Berlin.

[18] Reid, G.J., *A triangularization algorithm which determines the Lie symmetry algebra of any system of PDEs*, J. Phys. A: Math. Gen. **23** (1990) L853-L859.

[19] Riquier, C., *Les systèmes d'équations aux dérivées partielles*, Gauthier–Villars, Paris (1910).

[20] Schöbel, F., *The Symbolic Classification of Real Four Dimensional Lie Algebras*, Universität Leipzig Preprint (1992).

[21] Stephani, H., *A new solution of Einstein's field equations for a spherically symmetric perfect fluid in shear motion* J. Phys. A: Math. Gen. **16** (1983) 3529-3532.

[22] Stephani, H., private communication.

[23] Thomas, J., *Differential Systems*, AMS, Colloquium publications, v. 21, N.Y. (1937).

[24] Thomas, T. Y., *The fundamental theorem on quadratic first integrals*, Proc. **32**, (1946), 10-15.

[25] Wolf, T. and Brand, A., *The Computer Algebra Package* `CRACK` *for Investigating PDEs*, Manual for the package `CRACK` in the REDUCE network library and in Proceedings of ERCIM School on Partial Differential Equations and Group Theory, April 1992 in Bonn, GMD Bonn.

[26] Wolf, T. and Brand, A., *Examples of the investigation of differential equations with modularized programs* to be published in special issue of Mathematical and Computer Modelling,

[27] Wolf, T., *Programs for Applying Symmetries of PDEs*, Proceedings of ISSAC 95, p. 7, ACM Press (1995)

[28] Wolf, T., *Structural equations for Killing tensors of arbitrary rank*, preprint (1995).