Investigating DEs with CRACK and related programs

Thomas Wolf
School of Mathematical Sciences, QMW,
University of London, London E1 4NS

Andreas Brand
Institut für Informatik, Friedrich Schiller Universität Jena,
07740 Jena, Germany

April 18, 1995

1 Introduction

The aim of the package to be described is to try modularizing the investigation of differential equations for which there are no complete algorithms available yet. All, that is available for such problems are algorithms for special situations, e.g. when first integrals with a simple structure exist (e.g. polynomial in first derivatives) or when the problem has infinitesimal symmetries. In all such cases, finally a system of differential equations has to be solved which is overdetermined in the sense that more conditions have to be satisfied than there are unknown functions. To do a variety of such investigations efficiently, like a symmetry analysis, application of symmetries, determination of first integrals, differential factors, equivalent Lagrangians, the strategy is to have one package (CRACK) for simplifying DEs and solving simple DEs as effective as possible and to use this program as the main tool for all the above mentioned investigations. For each investigation there is then only a short program necessary to just formulate the necessary conditions in form of an overdetermined DE-system and to call CRACK to solve this, possibly in a number of successive calls. The examples below shall indicate the range of possible applications.

2 Examples

Example 1.
The program `LIEPDE` formulates conditions for point- and contact-symmetries

for single ODEs/PDEs and systems of them and calls CRACK to solve these conditions. By splitting the whole calculation in a succession of formulating conditions and solving them to simplify the formulation of the next conditions, the efficiency could be increased.

The following are the Karpman equations which play a role in plasma physics. In their real form, which is used to determine symmetries, they are [3]

$$\rho_{,t} + w_1\rho_{,z} + \frac{1}{2}\left[s_1(2\rho_{,x}\phi_{,x} + 2\rho_{,y}\phi_{,y} + \rho\phi_{,xx} + \rho\phi_{,yy}) + s_2(2\rho_{,z}\phi_{,z} + \rho\phi_{,zz})\right] = 0$$

$$\phi_{,t} + w_1\phi_{,z} - \frac{1}{2}\left[s_1\left(\frac{\rho_{,xx}}{\rho} + \frac{\rho_{,yy}}{\rho} - \phi_{,x}^2 - \phi_{,y}^2\right) + s_2\left(\frac{\rho_{,zz}}{\rho} - \phi_{,z}^2\right)\right] + a_1\nu = 0$$

$$\nu_{,tt} - (w_2)^2(\nu_{,xx} + \nu_{,yy} + \nu_{,zz}) - 2a_2\rho(\rho_{,xx} + \rho_{,yy} + \rho_{,zz}) - 2a_2(\rho_{,x}^2 + \rho_{,y}^2 + \rho_{,z}^2) = 0$$

with the three functions ρ, ϕ, ν of four variables t, x, y, z and with constant parameters a_i, s_i, w_i.

The corresponding input for LIEPDE is

```
depend r,x,y,z,tt;
depend f,x,y,z,tt;
depend v,x,y,z,tt;

de := {{df(r,tt)+w1*df(r,z)+s1*(df(r,x)*df(f,x)+df(r,y)*df(f,y)+r*
        df(f,x,2)/2+r*df(f,y,2)/2)+s2*(df(r,z)*df(f,z)+r*df(f,z,2)/2),
        df(f,tt)+w1*df(f,z)-(s1*(df(r,x,2)/r+df(r,y,2)/r-df(f,x)**2
        -df(f,y)**2)+s2*(df(r,z,2)/r-df(f,z)**2))/2+a1*v,
        df(v,tt,2)-w2**2*(df(v,x,2)+df(v,y,2)+df(v,z,2))
        - 2*a2*r*(df(r,x,2)+df(r,y,2)+df(r,z,2))
        - 2*a2*(df(r,x)**2+df(r,y)**2+df(r,z)**2)},
       {r,f,v},  {x,y,z,tt}};
mo := {0, nil, nil};
LIEPDE(de,mo);
```

The general solution for the symmetry generators then reads

$$\begin{aligned}
\xi^x &= -yc_1 + c_2 & \eta^r &= 0\\
\xi^y &= xc_1 + c_3 & \eta^\phi &= a_1c_6t^2 + a_1c_7t + c_8\\
\xi^z &= c_4 & \eta^v &= -c_6t - c_7\\
\xi^t &= c_5 & &
\end{aligned}$$

with constants $c_1, \ldots, c_8$. The corresponding 8 symmetry generators are

$$\begin{aligned}
&X_1 = \partial_x, \quad X_3 = \partial_z, \quad X_5 = y\partial_x - x\partial_y, \quad X_7 = a_1t\partial_\phi - \partial_\nu,\\
&X_2 = \partial_y, \quad X_4 = \partial_t, \quad X_6 = \partial_\phi, \quad X_8 = a_1t^2\partial_\phi - 2t\partial_\nu.
\end{aligned}$$

The time for this calculation (without outputing intermediate steps) on a PC486 with 33 MHz and 4MB RAM is less than 5 min.

Example 2.
Two ways to apply known infinitesimal symmetries of differential equations are a) to generalize an additionally 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 methods effectively lowers the order of an ODE by one or reduces the number of variables of a PDEs to obtain special solutions.

The following ODE for $h = h(\rho)$ resulted from an attempt to generalize Weyl's class of solutions of Einsteins field equations ([4])

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

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 [5]) 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

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

and the new ODE

$$0 = 3u''v - 16u'^3v^6 - 20u'^2v^3 + 5u'$$

where $u = u(v)$ and $' = d/dv$. Using the second symmetry, this first order equation for u' can be integrated by hand to obtain the parametric solution:

$$\begin{aligned} \rho &= \left(\frac{3c_1^2(2p-1)}{p^{1/2}(p+1)^{1/2}} + c_2\right)^{-1/2} \\ h &= \frac{(c_2p^{1/2}(p+1)^{1/2} + 6c_1^2p - 3c_1^2)^{1/2}p^{1/2}}{c_1(4p+1)} \end{aligned}$$

where $c_1, c_2 = const$ (details in [5]).

Example 3.
To find similarity and symmetry variables in the previous example the program has to solve first order linear PDEs. The corresponding procedure can also be used alone to solve quasilinear first order PDEs. Though the solution of the related characteristic ODE-system can not be assured and by that the solution of the PDE, the program is successful in more than half of the quasilinear first order PDEs in [1].

The following equation comes up in the elimination of resonant terms in normal forms of singularities of vector fields ([6]).

$$0 = xf_{,x} + yf_{,y} + 2zf_{,z} - 2f - xy \qquad \text{for} \qquad f = f(x,y,z).$$

The input

```
QUASILINPDE(df(f,x)*x+df(f,y)*y+2*df(f,z)*z-2*f-x*y,f,{x,y,z});
```

produces the result:

```
The general solution of the PDE is given through

          - log(z)*x*y + 2*f    sqrt(z)*x   sqrt(z)*y
0 = ff(---------------------,-----------,-----------)
                 z                 z           z

with arbitrary function ff(..).
```

The second PDE solved below is equation 3.12 from [1]:

$$0 = xw_{,x} + (ax+by)w_{,y} + (cx+dy+fz)w_{,z}\,, \quad a,b,c,d,f \;=\; \text{const}$$

`QUASILINPDE` returns that the general solution of the PDE is given through

$$0 = F\left(\frac{ax+by-y}{x^b(b-1)},\ \frac{cx(b-f)+dy(1-f)-adx}{x^f(b-f)(f-1)} + \frac{z}{x^f},\ w\right)$$

with arbitrary function F. This means that w is an arbitrary function of the first two arguments of the function F.

Example 4.
To solve equations of motion in General Relativity, symmetries of space-times have to be determined. Symmetries are also an essential criterion for classifying solutions of Einsteins field equations. With programs written by Guy Grebot ([7]) in the computer algebra system `CLASSI` the conditions for symmetries (Killing vectors, homothetic and conformal Killing vectors, Killing tensors) can be formulated and afterwards `CRACK` be called to solve them. In the simplest case of Killing vectors these are 10 first order linear PDEs for 4 functions of 4 variables. The Killing equations for Kimura-metric 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)*(SI
N(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)*(S
IN(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 SPARK 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 not more than these symmetries.

Example 5.
When numerical methods are used to solve DEs it might be interesting to know a corresponding Lagrangian in order to use other more efficient numerical methods. The program **LAGRAN** aims at finding Lagrangians polynomial in the first derivative for second order ODEs. For example, to investigate the 6'th transcendental Painlevé equation

$$\begin{aligned} y'' = & \ \frac{1}{2}\left(\frac{1}{y}+\frac{1}{y-1}+\frac{1}{y-x}\right)y'^2 - \left(\frac{1}{x}+\frac{1}{x-1}+\frac{1}{y-x}\right)y' \\ & + \frac{y(y-1)(y-x)}{x^2(x-1)^2}\left(a+\frac{bx}{y^2}+\frac{c(x-1)}{(y-1)^2}+\frac{dx(x-1)}{(y-x)^2}\right) \end{aligned} \tag{1}$$

concerning equivalence to a Lagrangian of the structure

$$L = u(x,y)(y')^2 + v(x,y)$$

the input would be

```
depend y,x;
de:={df(y,x,2) = (1/y+1/(y-1)+1/(y-x))*df(y,x)**2/2 -
     (1/x+1/(x-1)+1/(y-x))*df(y,x) + y*(y-1)*(y-x)/x**2/(x-1)**2*
     (a+b*x/y**2+c*(x-1)/(y-1)**2+d*x*(x-1)/(y-x)**2) , y, x };
LAGRAN(de,{0,{}});
```

for which **LAGRAN** finds

$$\begin{aligned} L \ = \ & 1/[xy(x-y)(x+1)(x-1)(y-1)]\cdot[(x+1)(x-1)^2x^2y'^2 \\ & -2a(xy+x+y)(x-y)(y-1)y + 2b(x+y+1)(x-y)(y-1)x \\ & +2c(x+y)(x-y)(x-1)y - 2d(x-1)(y+1)(y-1)xy]. \end{aligned}$$

3 Availability

The programs run under `REDUCE 3.4.1` or later versions and are available by ftp from galois.maths.qmw.ac.uk (138.37.80.15), directory `ftp/pub/crack`. Manual files `CRACK.TEX`, `APPLYSYM.TEX` give more details on the above applications.

References

[1] E. Kamke, Lösungsmethoden und Lösungen von Differentialgleichungen, Partielle Differentialgleichungen erster Ordnung, B. G. Teubner, Stuttgart (1979).

[2] T. Wolf, An efficiency improved program LIEPDE for determining Lie - symmetries of PDEs, Proceedings of the workshop on Modern group theory methods in Acireale (Sicily) Nov. (1992)

[3] V.I. Karpman, Phys. Lett. A 136, 216 (1989)

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

[5] T. Wolf, Programs for Applying Symmetries of PDEs, to be publ. in Proceedings of ISSAC 95 (1995)

[6] C. Herssens, P. Bonckaert, Limburgs Universitair Centrum/Belgium, private communication.

[7] T. Wolf, G. Grebot, Automatic Symmetry Investigation of Space-Time Metrics, Int. J. of Mod. Phys. D, Vol. 3, No. 1 (1994)