Manual for CONLAW versions 1,2,3,4

Thomas Wolf
Department of Mathematics
Brock University
St.Catharines
Ontario, Canada L2S 3A1
twolf@brocku.ca

March 20, 2004

1 Purpose

The procedures `CONLAW1, CONLAW2, CONLAW3, CONLAW4` try to find conservation laws for a given single/system of differential equation(s) (ODEs or PDEs)

$$u^\alpha_J = w^\alpha(x, u^\beta, ..., u^\beta_K, ...) \tag{1}$$

`CONLAW1` tries to find the conserved current P^i by solving

$$\text{Div } P = 0 \quad \text{modulo (1)} \tag{2}$$

directly. `CONLAW3` tries to find P^i and the characteristic functions (integrating factors) Q^ν by solving

$$\text{Div } P = \sum_\nu Q^\nu \cdot (u^\nu_J - w^\nu) \tag{3}$$

identically in all u-derivatives. Applying the Euler operator (variational derivative) for each u^ν on (3) gives a zero left hand side and therefore conditions involving only Q^ν. `CONLAW4` tries to solve these conditions identically in all u-derivatives and to compute P^i afterwards. `CONLAW2` does substitutions based on (1) before solving these conditions on Q^ν and therefore computes adjoined symmetries. These are completed, if possible, to conservation laws by computing P^i from the Q^ν.

All four procedures have the same syntax. They have two parameters, both are lists. The first parameter specifies the equations (1), the second specifies the computation to be done. One can either specify an ansatz for P^i, Q^ν or investigate a general situation, only specifying the order of the characteristic functions or the conserved current. For a more detailed description see the file `conca.tex`.

The file `CONLAW0.RED` contains subroutines used in all four versions.

2 The Syntax

The procedures `CONLAWi` i=1,2,3,4 are called through
`CONLAWi(`*problem,runmode*`);`
where i=1,2,3,4. Both parameters *problem, runmode* are lists. The first specifies the DEs to be investigated:
problem ... {*equations, ulist, xlist*}

equations... list of equations, each has the form `df(ui,..)=...` where the LHS (left hand side) `df(ui,..)` is selected such that
- The RHS (right h.s.) of an equations must not include the derivative on the LHS nor a derivative of it.
- The LHS of any equation should not occur in any other equation nor any derivative of the LHS.

If CONLAW3 or CONLAW4 are run where no substitutions are made then the LHS of equations can be `df(ui,..)**n=...` where `n` is a number. No difference is made between equations and constraints.

ulist ... list of function names, which can be chosen freely the number of functions and equations need not be equal

xlist ... list of variable names, which can be chosen freely

The second parameter specifies the calculation to be done.
runmode ... {*minord, maxord, expl, flist, inequ*}

minord ... the minimum of the highest order of derivatives of u
- in `p_t` for CONLAW1 where `t` is the first variable in `xlist` and
- in `q_j` for CONLAW2,CONLAW3,CONLAW4

maxord ... the maximum of the highest order of derivatives of u
- in `p_i` for CONLAW1 where `t` is the first variable in `xlist` and
- in `q_j` for CONLAW2,CONLAW3,CONLAW4

expl ... (t/nil) whether or not the charac. functions `q_i` or conserved current may depend explicitly on the variables of xlist

flist ... a list of unknown functions in any ansatz for `p_i,q_j`, also all parameters and parametric functions in the equation that are to be calculated such that conservation laws exist, if there are no such unknown functions then *flist* is the empty list: `{}`

inequ ... a list of expressions non of which may be identically zero for the conservation law to be found, if there is no such expression then *inequ* is an empty list: `{}`

The procedures `CONLAWi` return a list of conservation laws `{`$C_1, C_2, \ldots$`}`, if no non-trivial conservation law is found they return the empty list `{}`. Each C_i representing a conservation law has the form `{{`$P^1, P^2, \ldots$`},{`$Q^1, Q^2, \ldots$`}}`.

An ansatz for a conservation law can be formulated by specifying one or more of the components P^i for `CONLAW1`, one or more of the functions Q^μ for `CONLAW2, CONLAW4` or one or more of P^i, Q^μ for `CONLAW3`. The P^i are input as `p_i` where `i` in `p_i` stands for a variable name, and the Q^μ are input as `q_i` where `i` stands for an index - the number of the equation in the input list *equations* with which `q_i` is multiplied.

There is a restriction in the structure of all the expressions for `p_i, q_j` that are specified: they must be homogeneous linear in the unknown functions or constants which appear in these expressions. The reason for this restriction is not for CRACK to be able to solve the resulting overdetermined system but for `CONLAWi` to be able afterwards to extract the individual conservation laws from the general solution of the determining conditions.

All such unknown functions and constants must be listed in *flist* (see above). The dependencies of such functions must be defined before calling `CONLAWi`. This is done with the command `DEPEND`, e.g. `DEPEND f,t,x,u$` to specify f as a function of t, x, u. If one wants to have f as a function of derivatives of $u(t, x)$, say f depending on u_{txx}, then one can *not* write

```
DEPEND f,df(u,t,x,2)$
```

but instead

```
DEPEND f,u!‘1!‘2!‘2$
```

if *xlist* has been specified as `{t,x}`, because `t` is the first variable and `x` is the second variable in *xlist* and `u` is differentiated ones wrt. `t` and two times wrt. x we therefore get `u!‘1!‘2!‘2`. The character `!` is the exempt character to allow special characters like ‘ to occur in an identifier name.

It is possible to add extra conditions like PDEs for P^i, Q^μ as a list `cl_condi` of expressions that shall vanish. Remarks:

- The input to `CONLAW1, CONLAW2, CONLAW3, CONLAW4` is the same apart from:
  - an ansatz for Q^ν is ignored in `CONLAW1`
  - an ansatz for P^i is ignored in `CONLAW2, CONLAW4`
  - the meaning of mindensord, maxdensord is different in `CONLAW1` on one hand and `CONLAW2,CONLAW3,CONLAW4` on the other (see above).
- It matters how the differential equations are input, i.e. which derivatives are eliminated. For example, the Korteweg - de Vries equation if input in the form $u_{xxx} = -uu_x - u_t$ instead of $u_t = -uu_x - u_{xxx}$ in `CONLAW1` and choosing `maxdensord`=1 then P^i will be of at most first order, Div P of second order and u_{xxx} will not be substituted and no non-trival conservation laws can be

found. This does not mean that one will not find low order conservation laws at all with the substitution u_{xxx} one only has to go to `maxdensord`=2 with longer computations as a consequence compared to the input $u_t = -uu_x - u_{xxx}$ where `maxdensord`=0 is enough to find non-trivial conservation laws.

- The drawback of using $u_t = \ldots$ compared with $u_{xxx} = \ldots$ is that when seeking all conservation laws up to some order then one has to investigate a higher order ansatz, because with each substitution $u_t = -u_{xxx} + \ldots$ the order increases by 2. For example, if all conservation laws of order up to two in Q^ν are to be determined then in order to include a u_{tt}-dependence the dependence of Q^ν on u_x up to u_{6x} has to be considered.

- Although for any equivalence class of conserved currents P^i differing only by a curl, there exist characteristic functions Q^μ, this need not be true for any particular P^i. Therefore one cannot specify a known density P^i for `CONLAW3` and hope to calculate the remaining P^j and the corresponding Q^μ with `CONLAW3`. What one can do is to use `CONLAW1` to calculate the remaining components P^j and from this a trivial conserved density R and characteristic functions Q^ν are computed such that

$$\mathrm{Div}\,(P - R) = \sum_\nu Q^\nu \cdot (u^\nu_J - w^\nu).$$

- The Q^μ are uniquely determined only modulo $\Delta = 0$. If one makes an ansatz for Q^μ then this freedom should be removed by having the Q^μ independent of the LHS's of the equations and independent of derivatives of the LHS's of them. If the Q^μ were allowed to depend on anything, then (3) could simply be solved for one Q^ν in terms of arbitrary P^j and other arbitrary Q^ρ, giving Q^ν that are singular for solutions of the differential equations as the expression of the differential equation would appear in the denominator of Q^ν.

- Any ansatz for P^i, Q^ν should as well be independent of the LHS's of the equations (1) and independent of derivatives of the LHS's of (1).

- If in equation (3) the right hand side is of order m then from the conserved current P^i a trivial conserved current can be subtracted such that the remaining conserved current is at most of order m. If the right hand side is linear in the highest derivatives of order m then subtraction of a trivial conserved current can even achieve a conserved current of order $m - 1$. The relevance of this result is that if the right hand side is known to be linear in the highest derivatives then for P^i an ansatz of order $m - 1$ is only necessary. To take advantage of this relation if the right hand side is known to be linear in the highest derivatives, a flag `quasilin_rhs` can be set to `t` (see below).

3 Flags, parameters

```
LISP (PRINT_:= NIL/0/1/ ...)$
```

`print_=nil` suppresses all CRACK output, for `print_=n` (n an integer) `CRACK` prints only equations with at most n factors in its terms.

```
CRACKHELP()$
```

to show other flags controling the solution of the overdetermined PDE-system,

```
OFF BATCH_MODE$
```

to solve the system of conditions with `CRACK` interactively.

```
LISP(QUASILIN_RHS:=T)$
```

reduces in the ansatz for P^i the order to $m-1$ if the order of the right hand side is m. This can be used to speed up computations if the right hand side is known to be linear in the highest derivatives (see the note above).

4 Requirements

`REDUCE 3.6` and the files `CRACK.RED, CONLAW0.RED`, one of the files `CONLAW1.RED, CONLAW2.RED, CONLAW3.RED, CONLAW4.RED` depending which program should be used and all files `CR*.RED` which are read in from `CRACK.RED`.

One either has to read in files with

```
IN "crack.red","conlaw0.red","conlaw1.red"$
```

(and appropriate paths) or compile them before with

```
FASLOUT "crack"$
IN "crack"$
FASLEND$
FASLOUT "conlaw0"$
IN "conlaw0.red"$
FASLEND$
FASLOUT "conlaw1"$
IN "conlaw1.red"$
FASLEND$
BYE$
```

and load them afterwards with `LOAD crack,conlaw0,conlaw1$`
`conlaw2, conlaw3, conlaw4` are treated like `conlaw1`.

5 Examples

Below a `CRACK`-procedure `nodepnd` is used to clean up after each run and delete all dependencies of each function in the list of functions in the argument of `nodepnd`.

More details concerning these examples are given when running the file `conlaw.tst`.
`lisp(print_:=nil);` to suppress output from CRACK

- a single PDE:

```
depend u,x,t$
conlaw1({{df(u,t)=-u*df(u,x)-df(u,x,3)}, {u}, {t,x}},
        {0, 1, t, {}, {}})$
nodepnd {u}$
```

- a system of equations:

```
depend u,x,t$
depend v,x,t$
conlaw1({{df(u,t)=df(u,x,3)+6*u*df(u,x)+2*v*df(v,x),
          df(v,t)=2*df(u,x)*v+2*u*df(v,x)           },
         {u,v}, {t,x}},
        {0, 1, t, {}, {}})$
nodepnd {u,v}$
```

- a system of equations with ansatz:

```
depend u,x,t$
depend v,x,t$
depend r,t,x,u,v,u!‘2,v!‘2$
q_1:=r*df(u,x,2)$
conlaw2({{df(u,t)=df(v,x),
          df(v,t)=df(u,x) }, {u,v}, {t,x}},
        {2, 2, t, {r}, {r}})$
nodepnd {u,v,r}$
```

- for the determination of parameters, such that conservation laws exist:

```
depend u,x,t;
conlaw1({{df(u,t)=-df(u,x,5)-a*u**2*df(u,x)
                   -b*df(u,x)*df(u,x,2)-c*u*df(u,x,3)},
         {u}, {t,x}},
        {0, 1, t, {a,b,c}, {}});
nodepnd {u};
```

- for first integrals of an ODE-system including the determination of parameter values s,b,r such that conservation laws exist:

```
depend {x,y,z},t;
depend a1,x,t;
depend a2,y,t;
depend a3,z,t;

p_t:=a1+a2+a3;
conlaw2({{df(x,t) = - s*x + s*y,
          df(y,t) = x*z + r*x - y,
          df(z,t) = x*y - b*z},
         {x,y,z},{t}
        },
        {0,0,t,{a1,a2,a3,s,r,b},{}});
nodepnd {x,y,z,a1,a2,a3};
```