`Indiff`: a MAPLE package for over determined differential systems with Lie symmetry

The Short Manual

Elizabeth L. Mansfield
Institute of Mathematics and Statistics,
University of Kent,
Canterbury, CT2 7NF, U.K.
`E.L.Mansfield@ukc.ac.uk`

November 25, 1999

Abstract

We describe the MAPLE package `Indiff`, which is designed to calculate, as far as possible using the available theory, the elimination ideals, integrability conditions and compatibility conditions of a system of nonlinear partial differential equations, *written in terms of the differential invariants of a symmetry group of the system.* It is assumed that all differential equations are polynomial in the independent and dependent variables and the derivative terms.

This report includes examples of use and descriptions of the procedures. The code and input files for the examples discussed are available by anonymous ftp on `ftp.ukc.ac.uk:pub/maths/liz`. Information updates are available on the author's web page[1]. This manual is designed to be read in conjunction with the article *Algorithms for Symmetric Differential Systems*, also available from the author's web page. This article contains an explanation of the mathematics on which this package is based and the precise statement of the algorithms.

The structure of the package is based on the MAPLE package `diffgrob2`, also written by the author and available from the author's web page.

[1] URL: `http://www.ukc.ac.uk/ims/maths/people/E.L.Mansfield.html`

Contents

1 Introduction

There are now several publicly available computer algebra packages which complete a system of partial differential equations to characteristic or involutive form, or one of the variants of these forms. The output of these algorithms have many uses. One major application is finding, in some sense, the "simplest" system having the same set of solutions, from which the solution space may be more easily obtained or described. When these characteristic form algorithms are applied to systems with a symmetry, the expression swell can be so huge that the output is useless. This is the case when, for example, the system is invariant under rotational symmetries, and so there is no obvious term ordering to use in the elimination and simplification calculations.

The motivation behind this package was to analyze systems of partial differential equations which are invariant under a Lie group action. The idea is that if you divide out the symmetry by writing the system in terms of the differential invariants of the group action, the calculations on the invariantized system should then be more manageable. The aim is to complete the system to an invariantized characteristic form. As the formula for invariant differentiation requires prolongations of the group infinitesimals to the desired order, a computer implementation is a necessity for all but the smallest systems and simplest groups.

In order to use this package, it is necessary to know *a priori* the infinitesimal action of the Lie symmetry group on the dependent and independent variables. There are several packages available which can calculate the Lie symmetries of a system of partial differential equations, an excellent survey has been written by Willy Hereman, (Vol. III, Chapter 13 in the CRC Handbook of Lie Group Analysis of Differential Equations, CRC press, Boca Raton 1995).

This manual is designed to be read in conjunction with the article *Algorithms for Symmetric Differential Systems* written by the author and available from the author's home web page. This article will be referred to as **ASDS** in what follows.

The code to compute the prolongations of the infinitesimals was written by Khai Vu for the MAPLE package `Desolv`, (*Computer Algebra and Symmetry: the package Desolv*, PhD Thesis, Monash University, 1998) and has been adapted for use in this package by the author.

2 Installation

The instructions that follow are for UNIX[2]. The package is designed to run under MAPLE V5.

1. Create a directory `indiff` and subdirectories `code` and `src`. Place the source code file `indiff-src` into the `src` directory. Make sure these directories and this file are readable and executable. You may need to use `chmod` to achieve this.

2. `cd` into `indiff` and open a command line version of MAPLE (i.e. not xmaple), and type `!pwd` This will tell you what MAPLE thinks the complete file path of `indiff` is, which may be different from what the system thinks it is. We shall denote this file path by IndiffFilepath in what follows.

3. In your top level directory, create (or open if you already have) a file called `.mapleinit`. The dot is important! MAPLE looks for this file every time you open it. Include the lines
   `indiff:=`IndiffFilepath`:`
   `libname:=indiff,libname:`
   making sure you have all four colons, both back quotes, and the line defining `indiff` goes before the line defining `libname`.

[2]Installation of readlib packages under other operating systems has proved difficult for many of my `diffgrob2` users, regardless of how much consultation I provide. A version of the code which is system independent, but requires being read into MAPLE in its totality every MAPLE session, is available. Check the author's web page for details.

4. From inside `indiff/src`, open MAPLE and type

   `read ‘indiff-src‘;`

   If you get an error message saying that files cannot be opened for writing, check that the directories created in Step 1 are readable and executable.

5. Make sure all the files that have been created by MAPLE inside `indiff/code` are readable and executable.

You can check that the package has installed properly by opening MAPLE and typing `with(Indiff);` You should obtain a list of procedures now available to you. If you obtain the error message “undefined package”, there is something wrong with your `.mapleinit` file, for example it may not be in your top level directory. Other error messages indicate inadequate permissions on the files created inside `code` and so forth.

3 Explanation of necessary inputs, global variables and syntax

In **every** MAPLE session using `Indiff`, the independent variables must be given in a list denoted `vars`, the dependent variables must be given in a list denoted `ukns`, and the group parameter names must be given in a list denoted `GroupP`.

The derivative term

$$\frac{\partial^{|K|}u^\alpha}{\partial x_1^{K_1}\dots x_n^{K_n}}$$

is written here as u^α_K where K is a string consisting of K_1 1’s, K_2 2’s, $\dots$, K_n n’s. If the group is G, then the invariantization of u^α_K is obtained by applying the prolonged group action of G to u^α_K and then evaluating on the frame, or

$$(G\cdot u^\alpha_K)|_{frame}\,.$$

The invariantized form of u^α_K is written as I^α_K, (see **ASDS** §3.4), and is denoted in MAPLE by

`In[u[alpha],K].`

The order in which the independent variables appear in the list `vars` determines the meaning of indices of differentiation. For the invariantized derivative term `DT:=In[v,[1,1,3,4]]`, we have

operand		meaning
`op(0,DT);`	`In`	the *signifier* for an invariantized derivative term
`op(1,DT);`	`v`	the *unknown* being differentiated
`op(2,DT);`	`[1,1,3,4]`	the *index of differentiation* with respect to elements of `vars`

and thus if `vars`$=[x_1,x_2,x_3,x_4]$ then

$$\texttt{In[v,[1,1,3,4]]}\equiv\left(G\cdot\frac{\partial^4 v}{x_1^2x_3x_4}\right)\Bigg|_{frame}$$

The undifferentiated invariantized dependent variable `v` is denoted as `In[v,[]]`. The invariantized independent variable y, written as J_y, is denoted in `Indiff` as `In[y]`.

The package `Indiff` uses symbolic invariantized differentiation. The formulae are given in **ASDS** §3.5. From these formulae it can be seen that MAPLE needs to know

1. the infinitesimal action of the Lie symmetry group on the independent and dependent variables. This information is input in matrix form. The matrix must be denoted `XiPhis`.

2. the normalization equations which define the moving frame. These are input as a list, which must be denoted `Neqs`, using the symbolic invariantized derivatives.

We explain the necessary inputs using an example. Take the independent variable x, the dependent variable u, and the equiaffine group acting as

$$(x, u(x)) \mapsto \begin{cases} (x+a, u(x)) \\ (x, u(x)+b) \\ (\alpha x, u(x)/\alpha) \\ (x+\beta u(x), u(x)) \\ (x, u(x)+\delta x) \end{cases}$$

We now show how these variables are input in a MAPLE session.

```
> restart:with(Indiff);
```

$$[HNI,\ IKolRitt,\ IdSpoly,\ Idiff,\ Idiffparse,\ Invariantize,\ Iorthreduceall,\ Ireduce,\ Ireduceall,\ Kmat]$$

```
> vars:=[x]:ukns:=[u]:GroupP:=[a,b,alpha,beta,delta]:
```

The infinitesimal group action is given in matrix form which must be denoted `XiPhis`. The (κ, i)th element of `XiPhis` is the infinitesimal of the κth group parameter on the ith variable of the concatenated list `vars, ukns`. Hence, the order of the variables appearing in `vars`, `ukns` and `GroupP` lists affects the order of components in `XiPhis`.

```
> XiPhis:=matrix([[1,0],[0,1],[x,-u(x)],[u(x),0],[0,x]]);
```

$$XiPhis := \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ x & -\mathrm{u}(x) \\ \mathrm{u}(x) & 0 \\ 0 & x \end{bmatrix}$$

The normalization equations, `Neqs`, are the invariantized moving frame equations (see **ASDS** §3.3,§6.0), and are input using the syntax for invariantized derivatives described above. In this example, denoting the group action as $(x, u(x)) \mapsto (y, v(y))$, the moving frame equations are given by $y = 0$, $v = 0$, $v_y = 0$, $v_{yy} = 1$, $v_{yyy} = 0$ whose invariantized forms are $J = 0$, $I = 0$, $I_1 = 0$, $I_{11} = 1$, $I_{111} = 0$. For the record, the frame in terms of the group parameters is $a = x$, $b = u$, $\delta = u_x$, $\alpha = u_{xx}^{-1/3}$ and $\beta = -u_{xxx}/(3u_{xx}^2)$.

```
> Neqs:=[In[x],In[u,[]],In[u,[1]],In[u,[1,1]]-1,In[u,[1,1,1]]]:
```

Finally, `Indiff` needs the correction matrix K for invariantized symbolic differentiation, and for this the highest invariantized derivative terms must be calculated first. The procedure `HNI` calculates the set of highest invariantized derivative terms $\mathcal{HNI}$, denoted `HNIlist`. The information to calculate the map Norder (cf. **ASDS** equation (49)) is also calculated and stored as `DegDiffTable`. Both `DegDiffTable` and `HNIlist` are global variables.

The arguments of the `HNI` procedure give the term ordering, with respect to which the highest derivatives are decided. Detailed information concerning the arguments of the `HNI` procedure is given in the next section.

```
> HNI([[1],[u]],ttdeg):
```

The function `Kmat` calculates $-\mathsf{K}$, (**ASDS** §3.5) the matrix needed for invariant differentiation (**ASDS** equations (21), (22)). This matrix is denoted by `Indiff` as `Kmatrix`, and is a global variable.

```
> Kmat();
```

$$\begin{bmatrix} 1 & 0 & 0 & -\frac{1}{3}\,In_{u,\,[1,\,1,\,1,\,1]} & 1 \end{bmatrix}$$

Summary In order for `Indiff` to calculate anything, it needs the following:

INPUT

`vars`	list of independent variables
`ukns`	list of dependent variables
`GroupP`	list of group parameters
`XiPhis`	matrix of infinitesimal group action
`Neqs`	normalization equations

INITIAL CALCULATION

`HNI`	calculates `HNIlist` and `DegDiffTable`
`Kmat`	calculates `Kmatrix`

As far as `Indiff` is concerned, `vars`, `ukns`, `GroupP`, `HNIlist`, `DegDiffTable`, `XiPhis` and `Neqs` are global variables. That means that as MAPLE performs calculations using `Indiff` code, the current values of these variables are those used. Changing them mid-stream is not advisable.

The matrix `Kmatrix` is also a global variable. This matrix *is* simplified mid-stream by the procedure `IKolRitt` with respect to the system being analyzed, as an efficiency measure. Recalculating `Kmat();` will return `Kmatrix` to its original state as necessary.

4 Explanation of the arguments of the procedures

4.1 Arguments common to all procedures

- `f` is an invariantized differential polynomial.
- `F` is a list or set, `[f1,f2,...]` of invariantized differential polynomials
- `allvars` is a list of lists, used to give the ordering on the independent and dependent variables.
- `termordering` is a string variable, one of `ttdeg` or `mtdeg`

The last two of these are discussed in greater detail in the next subsection.

4.2 Term orderings

Term orderings are vital to the proper functioning of the simplification and cross-differentiation processes used by `Indiff`. The term ordering gives which terms are to be eliminated in a simplification process, and which are those chosen to be made equal and eliminated in a cross-differentiation. Term orderings must satisfy certain criteria to avoid infinite loops in simplification processes. Different orderings affect the final output of algorithms such as `IKolRitt`, not only its form but the use of the output.

There are two pieces of information needed to specify a term ordering in `Indiff`. The first is a list of lists, `allvars`, while the second is the name of the ordering, `termordering`, to be used.

Of all the term orderings known, only those involving "total degree of differentiation" are suitable for this package. In this context, the *degree* of I^α_K is $|K|$, (normally referred to as the *order*, but in the context of term orderings is referred to as the degree!) The two main term orderings are `ttdeg`, which stands for 'traditional total degree' ordering, and `mtdeg`, which stands for 'mixed total degree' ordering. These orderings are given in detail in **ASDS** §6.0. In brief, the `ttdeg` ordering first decides on the basis of total degree of differentiation, and then on a lexicographic ordering, which requires an ordering on the independent and dependent variables. The `mtdeg` ordering requires the dependent variables to be partitioned, and for the elements of the partition to be ordered. Given two derivative terms, the ordering is decided first on the ordering of the partition elements in which the relevant dependent variables are placed, and thereafter is the same as `ttdeg`.

It can be seen from the definitions of the term orderings (given in **ASDS** §6.0) that they depend on an ordering of the independent and dependent variables. The first is a list of lists, denoted `allvars`,

$$[[i_1, \dots i_n], [u^{j_1}, \dots u^{j_k}], \dots [u^{j_{k+\ell}}, \dots u^q]]$$

where i_ℓ are integers from 1 to p, the number of independent variables, i.e. `nops(vars)`, and the u^ℓ, $\ell = 1, \dots q$ are the q elements of `ukns`. For example

```
[[1,3,4,2],[u1,u3],[u4,u5,u2]]
```

indicates that the ordering on the independent variables is

$$x < z < t < y, \qquad \text{where } \texttt{vars} = [x, y, z, t]$$

while the ordering on the dependent variables is

$$u^1 < u^3 < u^4 < u^5 < u^2$$

The effect of partitioning the dependent variables only comes into play if the ordering is `mtdeg`.

5 Description of the major procedures

5.1 Invariantization

`Invariantize(G);`

`Invariantize(g);`

where `g` is a differential polynomial in the standard MAPLE syntax.

This procedure implements the Fels-Olver-Thomas replacement theorem. Every derivative term in the differential polynomial g, or a list or set of such differential polynomials, G, is replaced by its invariantized form,

$$u^\alpha_K \mapsto \texttt{I[u[alpha],K]}$$

and the result is simplified, algebraically, with respect to the normalization equations, `Neqs`.

The simplification is with respect to the ordering given in the procedure `HNI`. That is, the highest invariantized derivatives are eliminated. MAPLE performs simplification of a polynomial with respect to a set S of polynomials by first converting the set S to an algebraic Gröbner basis and then calculating the unique normal form. It may be assumed without loss of generality that `Neqs` is an algebraic Gröbner basis for the algebraic ideal it generates.

5.2 Invariant differentiation: `Idiff`

`Idiff(f,j);`

where $1 \le j \le p$, the number of independent variables in `vars`.

This procedure implements the invariant differential operator $\mathcal{D}_j$, that is, `f` $\mapsto$ `Idiff(f,j);`. The formulae may be found in **ASDS** equations (21-22). The argument `f` must be an invariantized differential polynomial.

5.3 Calculating highest derivative terms, separants ... `Idiffparse`

The procedure `HNI` calculates the highest terms in the normalization equations. For all other invariantized differential polynomials, the `Idiffparse` procedure is used.

`Idiffparse(f,allvars,termordering,'HIDT','Hpower','Hcoeff',` *'IDT', 'Sep', 'SepIDT'*`);`

The first three arguments are defined in §4. The remaining six arguments are output names, and thus are input with forward quote marks. The last three are optional. Arguments must appear in the argument place given, so if you want *Sep*, you need to also have *IDT*.

`HIDT`	the highest invariantized derivative term in `f`
`Hpower`	the highest power of `HIDT` appearing in `f`
`Hcoeff`	the coefficient of `HIDT^Hpower` in `f`
`IDT`	a list of invariantized derivatives appearing in `f`
`Sep`	the separant of `f`
`SepIDT`	a list of invariantized derivatives appearing in *Sep*

These are defined in **ASDS** §6.0. The procedure returns `HIDT`, `Hpower`, `Hcoeff` (and `Sep`, when it's requested).

5.4 Cross-differentiation: `IdSpoly`

`IdSpoly(f1,f2,allvars,termorder,'f1Sf2');`

The procedure calculates the invariant differential Spolynomial of the invariantized `f1`, `f2`. In words, the highest invariantized derivative terms, HIDTs, are calculated, then `f1`, `f2` are differentiated so that the HIDTs of the differentiated polynomials are equal. These differentiated polynomials are then cross-multiplied in order to eliminate the differentiated HIDTs. As explained at some length in **ASDS** §6.1, this process can only be carried out if the HIDT's of the f_i have sufficiently high order.

5.5 Differential simplification procedures

`Ireduce(f,F,allvars,termorder,'newf',`*options*`);`

The procedure calculates the invariant differential pseudo-reduction of the invariantized differential polynomial `f` with respect to the set of invariantized differential polynomials `F`. By pseudo-reduction is meant, that `f` may be multiplied by nontrivial expressions in order to effect the reduction. As explained at some length in **ASDS** §6.1, this process can only be carried out if the HIDT's of the elements of F have sufficiently high order.

Several options are available with `Ireduce`.

`'algebra'=[A,matrixorder]` As well as differential reduction, `Ireduce` will carry out algebraic simplification with respect to the set `A` of invariantized differential polynomials which are $\mathcal{N}$-bounded (**ASDS** §6.1). The termordering for the algebraic reduction is `matrixorder`, given as for the MAPLE package `Groebner` (cf. the help file `?Groebner[termorder]`)

`'Xset'='X'` All non-numeric factors of expressions with which `f` is multiplied in the course of the pseudo-reduction are collected in the set `X`, a user-defined name.

`Iorthreduce(f,F,allvars,termorder,'newf',`*options*`);`

Same as for `Ireduce`, but `f` may only be multiplied by factors not including any invariantized derivative terms. Only the `'algebra'=[A,matrixorder]` option is available.

`Ireduceall(F,allvars,termorder,'newF',`*options*`);`

`Iorthreduceall(F,allvars,termorder,'newF',`*options*`);`

These procedures simplify every element of `F` with respect to every other element of `F`, until no further reductions are possible, using either `Ireduce` or `Iorthreduce` respectively.

5.6 Basis calculation: `IKolRitt`

`IKolRitt(F,allvars,termorder,'newF',`*options*`);`

The purpose of this algorithm is to find integrability conditions and compatibility conditions of over determined systems of invariantized PDE. The aim is to find complete sets of such conditions. The IKolRitt algorithm is an analogue of the Kolchin Ritt algorithm, which is itself analogous to Buchberger's algorithm for a Gröbner basis for an algebraic polynomial system. IKolRitt is not, however, a naive translation of the Kolchin Ritt algorithm. A full discussion of the issues involved are given in **ASDS** §6. The inevitable caveats for nonlinear differential Gröbner bases apply, and these are discussed fully in the manual for `diffgrob2`.

Several options are available with `IKolRitt`.

The *options* are input in the form `option=`*argument*. The `strategy` and `info` options are input as sets, and one can have as many such options as one wants. Thus, for example, if one wants both the `info` options, `shownew` and `factor`, one inputs `info={'shownew','factor'}`.

The *strategy* options:

1. `strategy={'factor'}` factors each new condition as it is found and asks you to indicate which factor is to be used in the calculation. The factor number should be input followed by a semi-colon. It should be noted that if a product of functions is zero, say $f_1 f_2 = 0$, then in general one can only conclude that one of f_1, f_2 must be zero if the f_i are analytic. However, if one obtains, for example, a cubic in I_1, then the continuity of I_1 ensures that I_1 must be one of the roots of the cubic.

The *info* options:

1. `info={'shownew'}` prints out the new conditions as it finds them. This is useful if the computer runs out of memory before it completes the calculation, or if one suspects that a calculation suffers from expansion swell.
2. `info={'Xset'}` collects into a set all the factors by which any condition is multiplied in a pseudo-reduction calculation during the course of the algorithm. This set is the "margin of error" inherent

in any scheme that uses pseudo-reduction, and gives the set of so-called singular integrals that need to be considered when seeking a complete solution set of the input system. This can be achieved by taking each nontrivial member `h` of `Xset` in turn and calculating the algorithm on the input set together with `h`.

- `NonZero = S` It is possible to specify a set S of invariantized differential polynomials which are assumed to be non zero during the course of the calculations. If any invariantized differential polynomial calculated along the way has a factor which is in the set S, that factor is discarded. This option saves on memory and usually helps efficiency.
- `extraReduction=F2` As each new condition is found, it is additionally pseudo-reduced with respect to the list of differential polynomials `F2`, in the order in which they appear in `F2`.
- `extraVarlist=allvars2` The ordering on the variables taken when pseudo-reducing with respect to elements of `F2` is determined by `allvars2` and the termordering in use. If `allvars2` is not present, the ordering on the variables taken is that of `allvars`.

6 Examples of the use of the package and interpretation of the results

6.1 Invariant differentiation calculations

This first example is taken from the MSRI lectures of Peter Olver. It is the Euclidean group action used to study curves in the plane. The group action takes the form

$$(x, u(x)) \mapsto (x\cos\theta - u(x)\sin\theta + a, x\sin\theta + u(x)\cos\theta + b).$$

The three infinitesimal group actions are

$$\partial/\partial x, \qquad \partial/\partial u, \qquad -u\partial/\partial x + x\partial/\partial u.$$

The normalization equations are $J_x = 0$, $I_0 = 0$, $I_1 = 0$. In terms of the group parameters, the left moving frame is given by

$$(a, b) = (x, u(x)), \qquad \theta = \arctan(u_x).$$

```
> restart:with(Indiff);
```

$$[HNI, IKolRitt, IdSpoly, Idiff, Idiffparse, Invariantize, Iorthreduceall, Ireduce, Ireduceall, Kmat]$$

```
> vars:=[x];ukns:=[u];GroupP:=[a,b,theta];
```

$$vars := [x]$$

$$ukns := [u]$$

$$GroupP := [a,\, b,\, \theta]$$

```
> XiPhis:=matrix([[1,0],[0,1],[-u(x),x]]);
```

$$XiPhis := \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ -\mathrm{u}(x) & x \end{bmatrix}$$

```
> Neqs:=[In[x],In[u,[]],In[u,[1]]];
```

$$Neqs := [In_x,\, In_{u,\,[]},\, In_{u,\,[1]}]$$

```
> HNI([[1],[u]],ttdeg);
```

$$\{In_{u,\,[]},\, In_{u,\,[1]}\}$$

```
> Kmat();
```

$$\begin{bmatrix} 1 & 0 & In_{u,\,[1,\,1]} \end{bmatrix}$$

At this point, all the preliminary inputs and calculations having been made. Next we see the action of the invariant differential operator on the higher invariants I_1, I_{11}, and so forth. Since I_1 is a normalized invariant, its derivative is zero.

```
> for j to 5 do Idiff(In[u,[1$j]],1) od;
```

$$0$$

$$In_{u,\,[1,\,1,\,1]}$$

$$In_{u,\,[1,\,1,\,1,\,1]} - 3\, {In_{u,\,[1,\,1]}}^3$$

$$In_{u,\,[1,\,1,\,1,\,1,\,1]} - 10\, {In_{u,\,[1,\,1]}}^2\, In_{u,\,[1,\,1,\,1]}$$

$$In_{u,\,[1,\,1,\,1,\,1,\,1,\,1]} - 15\, In_{u,\,[1,\,1,\,1,\,1]}\, {In_{u,\,[1,\,1]}}^2 - 10\, In_{u,\,[1,\,1]}\, {In_{u,\,[1,\,1,\,1]}}^2$$

We can see what the invariants actually are in terms of derivatives of u, and check the above formulae, using the MAPLE library routine `dchange` (part of the library `PDEtools`) written by E. Cheb-Terrab. The `dchange` procedure performs a change of variable in a differential expression. Information concerning this procedure can be found on the MAPLE help page `?Pdetools,dchange`. Note that the change of variable we use here is the *inverse* of the group action. This is because we are using a *left* moving frame.

```
> with(PDEtools,dchange);
```

$$[dchange]$$

```
> tr:= {y=cos(theta)*(x-a)+sin(theta)*(u(x)-b),
  v(y)=-sin(theta)*(x-a)+cos(theta)*(u(x)-b) };
```

$$tr := \{y = \cos(\theta)\,(x - a) + \sin(\theta)\,(\mathrm{u}(x) - b),\ \mathrm{v}(y) = -\sin(\theta)\,(x - a) + \cos(\theta)\,(\mathrm{u}(x) - b)\}$$

The frame equation for θ is determined from the normalization equation $I_1 = 0$. Thus, we transform u_x using the group action and set the result to zero. This gives $\theta = \arctan(u_x)$.

```
> dchange(tr,diff(v(y),y),[x,u(x)]);
```

$$\frac{-\sin(\theta) + \cos(\theta)\,(\frac{\partial}{\partial x}\,\mathrm{u}(x))}{\cos(\theta) + \sin(\theta)\,(\frac{\partial}{\partial x}\,\mathrm{u}(x))}$$

We give the frame as a list of equations to be used in the `subs` command. The frame is expressed here in a sightly different form to that given above. First of all, the group parameters a and b do not appear in the transformed derivatives of $u(x)$, so we don't bother writing them in the list of substitutions. Secondly, it is simpler to substitute for $\sin\theta$, $\cos\theta$ than to substitute for θ itself and expect MAPLE to perform the trigonometric simplifications.

```
> frame:=sin(theta)=diff(u(x),x)/sqrt(1+diff(u(x),x)^2),
  cos(theta)=1/sqrt(1+diff(u(x),x)^2);
```

$$\mathit{frame} := \sin(\theta) = \frac{\frac{\partial}{\partial x}\,\mathrm{u}(x)}{\sqrt{1 + (\frac{\partial}{\partial x}\,\mathrm{u}(x))^2}},\ \cos(\theta) = \frac{1}{\sqrt{1 + (\frac{\partial}{\partial x}\,\mathrm{u}(x))^2}}$$

In what follows, we perform the two operations needed to compute the invariants. First, we calculate the group action on the j^{th} derivative of $u(x)$, and then evaluate this on the frame. The result is $I_{1\ldots1}$ (j 1's). We denote this here by `II[1...1]` as `I` is a reserved constant in MAPLE.

```
> dchange(tr,diff(v(y),y,y),[x,u(x)]):II[11]:=factor(subs(frame,%));
```

$$\mathit{II}_{11} := \frac{\frac{\partial^2}{\partial x^2}\,\mathrm{u}(x)}{(1 + (\frac{\partial}{\partial x}\,\mathrm{u}(x))^2)^{(3/2)}}$$

```
> dchange(tr,diff(v(y),y,y,y),[x,u(x)]):II[111]:=factor(subs(frame,%));
```

$$\mathit{II}_{111} := \frac{-3\,(\frac{\partial^2}{\partial x^2}\,\mathrm{u}(x))^2\,(\frac{\partial}{\partial x}\,\mathrm{u}(x)) + (\frac{\partial^3}{\partial x^3}\,\mathrm{u}(x)) + (\frac{\partial^3}{\partial x^3}\,\mathrm{u}(x))\,(\frac{\partial}{\partial x}\,\mathrm{u}(x))^2}{(1 + (\frac{\partial}{\partial x}\,\mathrm{u}(x))^2)^3}$$

```
> dchange(tr,diff(v(y),y$4),[x,u(x)]):II[1111]:=factor(subs(frame,%));
```

$$\begin{aligned}\mathit{II}_{1111} := {}&(15\,(\frac{\partial^2}{\partial x^2}\,\mathrm{u}(x))^3\,(\frac{\partial}{\partial x}\,\mathrm{u}(x))^2 - 10\,(\frac{\partial^2}{\partial x^2}\,\mathrm{u}(x))\,(\frac{\partial^3}{\partial x^3}\,\mathrm{u}(x))\,(\frac{\partial}{\partial x}\,\mathrm{u}(x))\\ &- 10\,(\frac{\partial^2}{\partial x^2}\,\mathrm{u}(x))\,(\frac{\partial^3}{\partial x^3}\,\mathrm{u}(x))\,(\frac{\partial}{\partial x}\,\mathrm{u}(x))^3 + (\frac{\partial^4}{\partial x^4}\,\mathrm{u}(x)) + 2\,(\frac{\partial^4}{\partial x^4}\,\mathrm{u}(x))\,(\frac{\partial}{\partial x}\,\mathrm{u}(x))^2\\ &+ (\frac{\partial^4}{\partial x^4}\,\mathrm{u}(x))\,(\frac{\partial}{\partial x}\,\mathrm{u}(x))^4)\Big/(1 + (\frac{\partial}{\partial x}\,\mathrm{u}(x))^2)^{(9/2)}\end{aligned}$$

Next we define a procedure `DD` which is the invariant derivative operator. This is just the operator $\partial/\partial x$ transformed by G and evaluated on the frame. We then proceed to check the formulae derived using the `Idiff` operator above.

```
> DD:=f->diff(f,x)/sqrt(1+diff(u(x),x)^2);
```

$$\mathit{DD} := f \to \frac{\frac{\partial}{\partial x}\,f}{\sqrt{1 + (\frac{\partial}{\partial x}\,\mathrm{u}(x))^2}}$$

```
> factor(DD(II[11])-II[111]);
```

$$0$$

```
> factor(DD(II[111])-II[1111]+3*II[11]^3);
```

$$0$$

```
> dchange(tr,diff(v(y),y$5),[x,u(x)]):II[11111]:=factor(subs(frame,%));
```

$$II_{11111} := \Big((\frac{\partial^5}{\partial x^5} \mathrm{u}(x)) - 20\, (\frac{\partial^3}{\partial x^3} \mathrm{u}(x))^2\, (\frac{\partial}{\partial x} \mathrm{u}(x))^3 + 3\, (\frac{\partial^5}{\partial x^5} \mathrm{u}(x))\, (\frac{\partial}{\partial x} \mathrm{u}(x))^4 + 3\, (\frac{\partial^5}{\partial x^5} \mathrm{u}(x))\, (\frac{\partial}{\partial x} \mathrm{u}(x))^2$$
$$- 10\, (\frac{\partial^3}{\partial x^3} \mathrm{u}(x))^2\, (\frac{\partial}{\partial x} \mathrm{u}(x)) + (\frac{\partial^5}{\partial x^5} \mathrm{u}(x))\, (\frac{\partial}{\partial x} \mathrm{u}(x))^6 - 15\, (\frac{\partial^2}{\partial x^2} \mathrm{u}(x))\, (\frac{\partial^4}{\partial x^4} \mathrm{u}(x))\, (\frac{\partial}{\partial x} \mathrm{u}(x))^5$$
$$- 105\, (\frac{\partial^2}{\partial x^2} \mathrm{u}(x))^4\, (\frac{\partial}{\partial x} \mathrm{u}(x))^3 - 10\, (\frac{\partial^3}{\partial x^3} \mathrm{u}(x))^2\, (\frac{\partial}{\partial x} \mathrm{u}(x))^5$$
$$+ 105\, (\frac{\partial^2}{\partial x^2} \mathrm{u}(x))^2\, (\frac{\partial^3}{\partial x^3} \mathrm{u}(x))\, (\frac{\partial}{\partial x} \mathrm{u}(x))^4 + 105\, (\frac{\partial^2}{\partial x^2} \mathrm{u}(x))^2\, (\frac{\partial^3}{\partial x^3} \mathrm{u}(x))\, (\frac{\partial}{\partial x} \mathrm{u}(x))^2$$
$$- 15\, (\frac{\partial^2}{\partial x^2} \mathrm{u}(x))\, (\frac{\partial^4}{\partial x^4} \mathrm{u}(x))\, (\frac{\partial}{\partial x} \mathrm{u}(x)) - 30\, (\frac{\partial^2}{\partial x^2} \mathrm{u}(x))\, (\frac{\partial^4}{\partial x^4} \mathrm{u}(x))\, (\frac{\partial}{\partial x} \mathrm{u}(x))^3 \Big) \Big/$$
$$(1 + (\frac{\partial}{\partial x} \mathrm{u}(x))^2)^6$$

```
> factor(DD(II[1111])-II[11111]+10*II[11]^2*II[111]);
```

$$0$$

6.2 Analyzing an over determined system

This example analyses the over determined system,

$$\begin{aligned} u_x^2 + u_y^2 - u_t^2 &= 1 \\ u_{xx} + u_{yy} - u_{tt} &= 2/u \end{aligned}$$

which is invariant under the Poincaré group of translations in x, y and t, rotations in (x, y), and boosts in (y, t) and (x, t). This group leaves the dependent variable u invariant. First, we input the necessary quantities and perform the initial calculations of `HNI` and `Kmat`.

```
> restart:with(Indiff);
```

$$[HNI,\ IKolRitt,\ IdSpoly,\ Idiff,\ Idiffparse,\ Invariantize,\ Iorthreduceall,\ Ireduce,\ Ireduceall,\ Kmat]$$

```
> vars:=[x,y,t];ukns:=[u];GroupP:=[k1,k2,k3,theta1,theta2,theta3];
```

$$vars := [x,\ y,\ t]$$

$$ukns := [u]$$

$$GroupP := [k1,\ k2,\ k3,\ \theta 1,\ \theta 2,\ \theta 3]$$

```
> XiPhis:=matrix([[1,0,0,0],[0,1,0,0],[0,0,1,0],[-y,x,0,0],[t,0,x,0],[0,t,y,0]]);
```

$$XiPhis := \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ -y & x & 0 & 0 \\ t & 0 & x & 0 \\ 0 & t & y & 0 \end{bmatrix}$$

```
> Neqs:=[In[x],In[y],In[t],In[u,[2]],In[u,[3]],In[u,[3,3]]];
```

$$Neqs := [In_x,\ In_y,\ In_t,\ In_{u,[2]},\ In_{u,[3]},\ In_{u,[3,3]}]$$

```
> HNI([[1,3,2],[f],[u]],mtdeg);
```

$$\{In_{u,[2]},\ In_{u,[3]},\ In_{u,[3,3]}\}$$

```
> Kmat();
```

$$\begin{bmatrix} 1 & 0 & 0 & \dfrac{In_{u,[1,2]}}{In_{u,[1]}} & -\dfrac{In_{u,[1,3]}}{In_{u,[1]}} & \dfrac{{In_{u,[1,3]}}^2}{In_{u,[1]}\,In_{u,[2,3]}} - \dfrac{1}{2}\dfrac{In_{u,[1,3,3]}}{In_{u,[2,3]}} \\ 0 & 1 & 0 & \dfrac{In_{u,[2,2]}}{In_{u,[1]}} & -\dfrac{In_{u,[2,3]}}{In_{u,[1]}} & \dfrac{In_{u,[1,3]}}{In_{u,[1]}} - \dfrac{1}{2}\dfrac{In_{u,[2,3,3]}}{In_{u,[2,3]}} \\ 0 & 0 & 1 & \dfrac{In_{u,[2,3]}}{In_{u,[1]}} & 0 & -\dfrac{1}{2}\dfrac{In_{u,[3,3,3]}}{In_{u,[2,3]}} \end{bmatrix}$$

As a quick check, we note that the invariant differentiation of the normalization equations are zero.

```
> seq(Idiff(Neqs[6],j),j=1..3);
```

$$0,\ 0,\ 0$$

```
> seq(Idiff(Neqs[5],j),j=1..3);
```

$$0,\ 0,\ 0$$

```
> seq(Idiff(Neqs[4],j),j=1..3);
```

$$0,\ 0,\ 0$$

The invariantization of the input system is trivial so we input it directly, choosing the positive square root. Further, we note that $I_1 = 1$ implies that $I_{11} = 0$, simplifying further the invariantization of the second equation.

```
> g1:=In[u,[1]]-1;
```

$$g1 := In_{u,[1]} - 1$$

```
> Idiff(g1,1);
```

$$In_{u,[1,1]}$$

```
> g2:=In[u,[]]*In[u,[2,2]]-2;
```

$$g2 := In_{u,[]}\, In_{u,[2,2]} - 2$$

We perform `IKolRitt` on the invariantized system, with the option that $I_0 \neq 0$. We also ask MAPLE to print the `Xset`, the set of all nontrivial expressions by which a differential polynomial is multiplied in the course of a pseudo-reduction. These quantities are implicitly assumed to be non zero. The output is discussed in **ASDS** §5.1.2.

```
> IKolRitt([g1,g2],[[1,3,2],[u]],mtdeg,'BIGTEST',
  info={'Xset'},NonZero={In[u,[]]});

the Xset is generated by   {In[u,[]], In[u,[2, 3]]}
```

$$\begin{aligned}&[In_{u,[2]},\, In_{u,[3]},\, In_{u,[1]} - 1,\, In_{u,[]}\, In_{u,[2,2]} - 2,\, {In_{u,[2,2]}}^2 - 4\,\%1,\, 2\, In_{u,[]}\,\%1 - In_{u,[2,2]},\\ &\quad In_{u,[3,3]},\, In_{u,[1,2]},\, In_{u,[1,3]},\, In_{u,[1,1]},\, In_{u,[2,2,2]} - In_{u,[]}\, In_{u,[2,2,3]}\, In_{u,[2,3]},\\ &\quad In_{u,[2,3,3]} - In_{u,[2,2,2]},\, In_{u,[3,3,3]} - In_{u,[2,2,3]},\, In_{u,[1,2,2]} + 3\,\%1,\\ &\quad In_{u,[1,2,3]} + In_{u,[2,3]}\, In_{u,[2,2]},\, In_{u,[1,3,3]} + \%1,\, In_{u,[1,1,2]},\, In_{u,[1,1,3]},\, In_{u,[1,1,1]}]\\ &\quad \%1 := {In_{u,[2,3]}}^2\end{aligned}$$

```
> time();
```

$$22.289$$

The calculation was performed on a 500MHz Pentium III processor having 512Mb RAM, with MAPLE V5 running under Red Hat Linux 6.0.

6.3 A classification problem

We show a simple classification problem, that of classical symmetries of the equation $u_t = (f(u)u_x)_x$. The determining equations for the infinitesimals $\xi(x,t,u)$, $\tau(x,t,u)$ and $\phi(x,t,u)$, where the invariant surface condition for the symmetry is

$$\xi u_x + \tau u_t = \phi,$$

can be obtained using any of several packages, and is

$$\begin{gathered}\xi_u = 0 \qquad \tau_u = 0\\ \tau_x = 0, \qquad \phi_{uu} = 0\\ f(2\xi_x - \tau_t) - f_u\phi\\ f(2\phi_{xu} - \xi_{xx}) + 2f_u\phi_x + \xi_t = 0\\ f\phi_{xx} - \phi_t = 0\end{gathered}$$

The group under which this system is invariant is the equivalence group of $u_t = (f(u)u_x)_x$. This consists of all translations and scalings in t and u. These induce an action on f, $\xi(x,t,u)$, $\tau(x,t,u)$ and $\phi(x,t,u)$, which is given by

$$\begin{array}{rcl} x & \mapsto & x+k_1 \\ t & \mapsto & \lambda t+k_2 \\ u & \mapsto & \mu u+k_3 \\ f & \mapsto & \mu f/\lambda \\ \xi & \mapsto & \xi \\ \tau & \mapsto & \lambda\tau \\ \phi & \mapsto & \mu\phi. \end{array}$$

The normalization equations for the translations have little effect as the system is autonomous, and these could be ignored. The remaining equations are all for the arbitrary function f, namely $I_f = 1$, $I_{f,3} = 1$. The whole purpose of the exercise is to find the conditions on f for which extra symmetries exist. These conditions will be differential invariants of the equivalence group, expressible in terms of f. In fact the frame is $\lambda = f$, $\mu = f_u/f$. The second order invariant in f is `In[f,[3,3]]` $= ff_{uu}/f_u^2$. The main classification condition to be found is $(ff_{uu}/f_u^2)_u = 0$. The secondary classification condition is $ff_{uu}/f_u^2 = 7/4$.

```
> restart:with(Indiff);
```

$$[\mathit{HNI},\ \mathit{IKolRitt},\ \mathit{IdSpoly},\ \mathit{Idiff},\ \mathit{Idiffparse},\ \mathit{Invariantize},\ \mathit{Iorthreduceall},\ \mathit{Ireduce},\ \mathit{Ireduceall},\ \mathit{Kmat}]$$

```
> vars:=[x,t,u];ukns:=[xi,tau,phi,f];GroupP:=[k1,k2,k3,lambda,mu];
```

$$\mathit{vars} := [x,\ t,\ u]$$

$$\mathit{ukns} := [\xi,\ \tau,\ \phi,\ f]$$

$$\mathit{GroupP} := [\mathit{k1},\ \mathit{k2},\ \mathit{k3},\ \lambda,\ \mu]$$

```
> XiPhis:=matrix([[1,0,0,0,0,0,0],[0,1,0,0,0,0,0],[0,0,1,0,0,0,0],
  [0,t,0,0,tau(x,t,u),0,-f(x,t,u)],[0,0,u,0,0,phi(x,t,u),0]]);
```

$$\mathit{XiPhis} := \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & t & 0 & 0 & \tau(x,t,u) & 0 & -\mathrm{f}(x,t,u) \\ 0 & 0 & u & 0 & 0 & \phi(x,t,u) & 0 \end{bmatrix}$$

```
> Neqs:=[In[x],In[t],In[u],In[f,[]]-1,In[f,[3]]-1];
```

$$\mathit{Neqs} := [\mathit{In}_x,\ \mathit{In}_t,\ \mathit{In}_u,\ \mathit{In}_{f,\,[]} - 1,\ \mathit{In}_{f,\,[3]} - 1]$$

```
> HNI([[3,1,2],[f],[xi,tau,phi]],mtdeg);
```

$$\{\mathit{In}_{f,\,[]},\ \mathit{In}_{f,\,[3]}\}$$

```
> Kmat();
```

$$\begin{bmatrix} 1 & 0 & 0 & -In_{f,[1]} & In_{f,[1]} - In_{f,[1,3]} \\ 0 & 1 & 0 & -In_{f,[2]} & In_{f,[2]} - In_{f,[2,3]} \\ 0 & 0 & 1 & -1 & 1 - In_{f,[3,3]} \end{bmatrix}$$

```
> GG:=[diff(xi(x,t,u),u),diff(tau(x,t,u),u),diff(tau(x,t,u),x),
  diff(phi(x,t,u),u,u),diff(f(x,t,u),x),diff(f(x,t,u),t),
  f(x,t,u)*(2*diff(xi(x,t,u),x)-diff(tau(x,t,u),t))-phi(x,t,u)*diff(f(x,t,u),u),
  f(x,t,u)*(2*diff(phi(x,t,u),x,u)-diff(xi(x,t,u),x,x))
  +2*diff(f(x,t,u),u)*diff(phi(x,t,u),x)+diff(xi(x,t,u),t),
  f(x,t,u)*diff(phi(x,t,u),x,x)-diff(phi(x,t,u),t)];
```

$$\begin{aligned} GG := [&\frac{\partial}{\partial u}\xi(x,t,u), \frac{\partial}{\partial u}\tau(x,t,u), \frac{\partial}{\partial x}\tau(x,t,u), \frac{\partial^2}{\partial u^2}\phi(x,t,u), \frac{\partial}{\partial x}\mathrm{f}(x,t,u), \frac{\partial}{\partial t}\mathrm{f}(x,t,u), \\ &\mathrm{f}(x,t,u)\,(2\,(\frac{\partial}{\partial x}\xi(x,t,u)) - (\frac{\partial}{\partial t}\tau(x,t,u))) - \phi(x,t,u)\,(\frac{\partial}{\partial u}\mathrm{f}(x,t,u)), \\ &\mathrm{f}(x,t,u)\,(2\,(\frac{\partial^2}{\partial x\,\partial u}\phi(x,t,u)) - (\frac{\partial^2}{\partial x^2}\xi(x,t,u))) + 2\,(\frac{\partial}{\partial u}\mathrm{f}(x,t,u))\,(\frac{\partial}{\partial x}\phi(x,t,u)) \\ &+ (\frac{\partial}{\partial t}\xi(x,t,u)), \mathrm{f}(x,t,u)\,(\frac{\partial^2}{\partial x^2}\phi(x,t,u)) - (\frac{\partial}{\partial t}\phi(x,t,u))] \end{aligned}$$

```
> IGG:=map(Invariantize,GG);
```

$$\begin{aligned} IGG := [&In_{\xi,[3]}, In_{\tau,[3]}, In_{\tau,[1]}, In_{\phi,[3,3]}, In_{f,[1]}, In_{f,[2]}, 2\,In_{\xi,[1]} - In_{\tau,[2]} - In_{\phi,[]}, \\ &2\,In_{\phi,[1,3]} - In_{\xi,[1,1]} + 2\,In_{\phi,[1]} + In_{\xi,[2]}, In_{\phi,[1,1]} - In_{\phi,[2]}] \end{aligned}$$

This is the classifying condition we should be looking for

```
> Idiff(In[f,[3,3]],3);
```

$$In_{f,[3,3,3]} + In_{f,[3,3]} - 2\,In_{f,[3,3]}{}^2$$

We are now ready to use IKolRitt to process the system. We know we don't want to consider f linear, so we set `In[f,[3,3]]` to be non zero (recall `In[f,[3,3]]`$=ff_{uu}/f_u^2$). The term ordering used must be the same as that used in `HNI`. The classifying condition comes out as a factor in the very first condition found.

```
> IKolRitt(IGG,[[3, 2, 1], [f], [xi, tau], [phi]], mtdeg,'BB',
  info={'shownew','Xset'},strategy={'factor'},NonZero= {In[f,[3,3]]});
```

```
In[xi,[2]]*(In[f,[3, 3, 3]]+In[f,[3, 3]]-2*In[f,[3, 3]]^2)


here are the factors

factor number   1

In[xi,[2]]
```

```
factor number   2

In[f,[3, 3, 3]]+In[f,[3, 3]]-2*In[f,[3, 3]]^2
```

```
>  2;
```

```
the Xset is generated by   {4*In[f,[3, 3]]-7, -In[f,[3, 3]]-In[f,[3,
3, 3]]+2*In[f,[3, 3]]^2}
```

$$[In_{f,[]} - 1,\ In_{f,[1]},\ In_{f,[2]},\ In_{f,[3]} - 1,\ In_{f,[1,1]},\ In_{f,[1,2]},\ In_{f,[2,2]},\ In_{f,[1,3]},\ In_{f,[2,3]},\ In_{f,[3,3,3]} + In_{f,[3,3]} - 2\,{In_{f,[3,3]}}^2,\ In_{\xi,[3]},\ In_{\tau,[1]},\ In_{\tau,[3]},\ -7\,In_{\xi,[1,1]} - In_{\xi,[2]} + 4\,In_{f,[3,3]}\,In_{\xi,[1,1]},\ 2\,In_{\xi,[1,1,1]} - 2\,In_{\xi,[1,2]} + In_{\tau,[2,2]},\ 2\,In_{\xi,[1]} - In_{\tau,[2]} - In_{\phi,[]}]$$

The output is quite disappointing, as the classifying condition appears in the `Xset`. This means that despite choosing it to be zero at some point, nevertheless, it had already been assumed to be nonzero in an earlier pseudo-reduction. The problem can be addressed in two different ways. One possibility is to try a different term ordering, in fact setting `allvars` to be `[[1, 3, 2], [f], [xi, tau], [phi]]` will give better output. If you do this, then `HNI`, `Kmat` must be recalculated with the new ordering. Alternatively we can put the classifying condition into the input system. This second option we do next. It is necessary to recalculate `Kmatrix` before using `IKolRitt` a second time.

```
>  Kmat();
```

$$\begin{bmatrix} 1 & 0 & 0 & -In_{f,[1]} & In_{f,[1]} - In_{f,[1,3]} \\ 0 & 1 & 0 & -In_{f,[2]} & In_{f,[2]} - In_{f,[2,3]} \\ 0 & 0 & 1 & -1 & 1 - In_{f,[3,3]} \end{bmatrix}$$

```
>  IKolRitt([op(IGG),-In[f,[3, 3, 3]]- In[f,[3, 3]]+2*In[f,[3, 3]]^2],
   [[3, 2, 1], [f], [xi, tau], [phi]], mtdeg,'BB',info={'shownew','Xset'},
   strategy={'factor'},NonZero={In[f,[3,3]],4*In[f,[3, 3]]-7});
```

```
4*In[f,[3, 3]]*In[tau,[2, 2]]-8*In[f,[3, 3]]*In[xi,[1,
2]]-7*In[tau,[2, 2]]+16*In[xi,[1, 2]]


-In[xi,[2]]


the Xset is generated by   {4*In[f,[3, 3]]-7}
```

$$[In_{f,[]} - 1,\ In_{f,[1]},\ In_{f,[2]},\ In_{f,[3]} - 1,\ In_{f,[1,1]},\ In_{f,[1,2]},\ In_{f,[2,2]},\ In_{f,[1,3]},\ In_{f,[2,3]},$$
$$-In_{f,[3,3]} - In_{f,[3,3,3]} + 2\,{In_{f,[3,3]}}^2,\ In_{\xi,[2]},\ In_{\xi,[3]},\ In_{\tau,[1]},\ In_{\tau,[3]},$$
$$4\,In_{f,[3,3]}\,In_{\xi,[1,1]} - 7\,In_{\xi,[1,1]},\ 4\,In_{f,[3,3]}\,In_{\tau,[2,2]} - 7\,In_{\tau,[2,2]},\ 2\,In_{\xi,[1,1,1]} + In_{\tau,[2,2]},$$
$$2\,In_{\xi,[1]} - In_{\tau,[2]} - In_{\phi,[]}]$$

It is clear that the other classifying condition is $I_{f,3,3} = 7/4$ and we next calculate the result with this condition in the input system.Again we must recalculate `Kmat`.

```
> Kmat();
```

$$\begin{bmatrix} 1 & 0 & 0 & -In_{f,[1]} & In_{f,[1]} - In_{f,[1,3]} \\ 0 & 1 & 0 & -In_{f,[2]} & In_{f,[2]} - In_{f,[2,3]} \\ 0 & 0 & 1 & -1 & 1 - In_{f,[3,3]} \end{bmatrix}$$

```
> IKolRitt([op(IGG),4*In[f,[3, 3]]-7], [[1, 3, 2], [f], [xi, tau], [phi]], mtdeg,
  'BB',info={'shownew','Xset'},strategy={'factor'});
```

```
In[tau,[2, 2]]


the Xset is generated by   {}
```

$$[In_{f,[]} - 1,\ In_{f,[2]},\ In_{f,[3]} - 1,\ In_{f,[1]},\ In_{f,[2,2]},\ In_{f,[2,3]},\ 4\,In_{f,[3,3]} - 7,\ In_{f,[1,2]},\ In_{f,[1,3]},\ In_{f,[1,1]},$$
$$-In_{\xi,[2]},\ In_{\xi,[3]},\ In_{\tau,[3]},\ In_{\tau,[1]},\ In_{\tau,[2,2]},\ -In_{\xi,[1,1,1]},\ 2\,In_{\xi,[1]} - In_{\tau,[2]} - In_{\phi,[]}]$$

7 Notes

1. Currently, `Indiff` can handle functions of the dependent variables. The syntax for the invariantized form of $\partial f(u_1, \ldots)/\partial u_1^{n_1} \ldots$ is `In[f,[`u_1`,...],[`n_1`,...]]`, so that the third argument is an additive multi-index of differentiation and the second argument a list of elements of `ukns`. However, if the system contains both invariantized independent variables and arbitrary functions of the dependent variables, then loss of information can occur at present. In `allvars`, the names of functions of functions should be immediately to the right of the list of integers giving the ordering on the independent variables, and to the left of any remaining lists of dependent variables.

2. Variables which are not declared in either `vars` or `ukns` are assumed to be constants as far as invariant differentiation is concerned.

3. It is well known that `Ireduceall` can lead to loss of information. Work is in progress to enable `Indiff` to alert the user when this happens.

See **ASDS** for a Bibliography and the Acknowledgements.