Algorithm of Determining equations of infinitesimals for PDE of order one, two, and three using SageMath

Vishwas Khare\(^1\)* and M.G. Timol\(^2\)

Abstract
Algorithm in Computer algebra system is developed for open-source software Sage Math to determine the Lie group infinitesimal transformation of PDE of order one, two, and three in one dependent variable \(u\) and two independent variables \(x\) and \(t\). The application of algorithm is illustrated through examples. The advantage of the present algorithm is that it gives the set of determining equations directly by giving inputs as differential equation also the algorithm is universal as SageMath is a free open-source mathematics software system licensed under the GPL. The algorithm is very useful for researchers working with linear/nonlinear PDE using Lie symmetry method and SageMath software.

Keywords
PDE, Lie symmetry, infinitesimals.

AMS Subject Classification
76M60, 58J70, 35R03, 14B10.

1. Introduction
A group of transformations with one parameter is called continuous if its elements are identified by a continuous parameter [6]. A continuous group of transformations is admitted by a PDE if the PDE remains invariant under that group. The group is called the global transformation group. With a global group of transformation, there is associated an infinitesimal group of transformation that can be found using the concept of invariance of PDE. From a given infinitesimal group one can find the global group and vice versa. To find the infinitesimal group of transformations one has to find and solve the set of determining equations. The process involves many calculations and becomes cumbersome if done manually.

There are many computer algebra systems which provides packages to solve PDE using the symmetry method for example maple and Mathematica etc.

As per the author’s knowledge, no package is available in SageMath to solve PDE at present. In this paper, we have proposed an algorithm in the computer algebra system SageMath software that find determining equations of infinitesimals for solving PDE using symmetry technique. The algorithm presented is universal as SageMath is open-source with the GPL license.

The algorithm presented finds the set of determining equations by giving the inputs as PDE written in solved form as explained in the examples.

The codes given in the algorithm can be downloaded using the link \textit{https://rb.gy/nmufo8}. and can be used, using SageMath Cell, SageMath cloud.
Algorithm of Determining equations of infinitesimals for PDE of order one, two, and three using SageMath — 296/301

2. Mathematical concepts

Consider [2] the kth order PDE written in solved form in terms of some k th order partial derivatives of u:

\[ F(x,u,u_1,u_2,\ldots,u_k) = u_{i_1,i_2,\ldots,i_k} - f(x,u,u_1,u_2,\ldots,u_k) = 0. \tag{2.1} \]

where \( x = (x_1,x_2,\ldots,x_n) \) denotes n independent variables, \( u \) denotes the dependent variable, and \( i \) denotes the set of coordinates corresponding to all jth order partial derivatives of \( u \) with respect to \( x \). The coordinate of \( u \) corresponding to \( \frac{\partial^j}{\partial x_{i_1} x_{i_2} \ldots \partial x_{i_j}} \) is denoted by \( u_{i_1,i_2,\ldots,i_j} \), \( i_j \) = 1,2,\ldots,n for \( j = 1,2,\ldots,k \).

**Theorem 2.1. (Infinitesimal Criterion for Invariance of a PDE)** Let [2]

\[ \xi = \frac{\partial}{\partial x_i} + \eta \frac{\partial}{\partial u}. \tag{2.2} \]

be the infinitesimal generator of

\[ x^i = X(x,u;\varepsilon) \]
\[ u^i = U(x,u;\varepsilon). \tag{2.3} \]

where \( \xi \) and \( \eta \) are infinitesimals.

Let

\[ \xi^{(k)} = \xi_i \frac{\partial}{\partial x_i} + \eta \frac{\partial}{\partial u} + \eta^{(1)}(x,u,u) \frac{\partial}{\partial u_1} + \cdots + \eta^{(k)}(x,u,u) \frac{\partial}{\partial u_{i_1,i_2,\ldots,i_j}}. \tag{2.4} \]

be the kth extended infinitesimal generator of "(2.2)" where \( \eta_i^{(1)} \) and \( \eta_i^{(k)} \) are given by

\[ \eta_i^{(1)} = D_i \eta - (D_i \xi_j) u_j, \quad i = 1,2,\ldots,n; \]
\[ \eta_i^{(k)} = D_i \eta^{(k-1)} - (D_i \xi_j) u_{i_1,i_2,\ldots,i_{k-1},j}, \]
\[ i_j = 1,2,\ldots,n; \text{ for } l = 1,2,\ldots,k \]
\[ \text{with } k = 2,3,\ldots. \tag{2.5} \]

Then "(2.3)" is admitted by PDE "(2.1)" if and only if

\[ \xi^{(k)} F(x,u,u_1,u_2,\ldots,u_k) = 0. \]

when

\[ F(x,u,u_1,u_2,\ldots,u_k) = 0. \tag{2.6} \]

**Proof.** For proof see [2]

**Remark 2.2.** Equations "(2.6)" is called invariance condition or linearized symmetry condition.

### Table 1. Table for notations used in algorithm and examples.

| Symbols | Equivalent symbol used in algorithm and examples |
|---------|-------------------------------------------------|
| \( \frac{\partial u}{\partial x} \) | \( u_{,x} \) |
| \( \frac{\partial u}{\partial t} \) | \( u_{,t} \) |
| \( \frac{\partial^2 u}{\partial x^2} \) | \( u_{xx} \) |
| \( \frac{\partial^2 u}{\partial x \partial t} \) | \( u_{xt} \) |
| \( \frac{\partial^2 u}{\partial t^2} \) | \( u_{tt} \) |
| \( \frac{\partial^3 u}{\partial x \partial t^2} \) | \( u_{xtt} \) |
| \( \frac{\partial^3 u}{\partial t^3} \) | \( u_{ttt} \) |

3. Symbols used in Algorithm

4. Algorithm

# Program for finding determining equation for PDE OF ORDER ONE TWO AND THREE
print ("Program find determining equ
of the type u_i=f(u_{k},u,x,t) where i,k can take value x,t,tt,xx,xt,xxx, xxt,xtt and u_i is not equal to u_{k}")
var ("x,t,u,u_x,u_t,u_xx,u_xt,u_txx,u_ttx,u_xxx,u_xtt,u_ttx,u_txx,c,a") function ("X,Y,f,F,U,T,V,w")
# Define function import itertools
@interact
def partial_symmetry(A=input_box (default =u_xxx .label=’Insert u_{,xx}',w=input_box (default =u_t u*u_x,label="Insert f..._right=True)


Algorithm of Determining equations of infinitesimals for PDE of order one, two, and three using SageMath — 297/301

of eq u_i=f(u_j,u_x,u_t)');
W=A(w)
# 1)
D_xU=diff(U(x,t,u),u)+u_x*diff(U(x,t,u),u)+u_xx*diff(U(x,t,u),u)+u_xt*diff(U(x,t,u),u)+u_xxt*diff(U(x,t,u),u)+u_xxt*diff(U(x,t,u),u)+u_xt*diff(U(x,t,u),u)+u_xtt*diff(U(x,t,u),u)
D_tU=diff(U(x,t,u),t)+u_t*diff(U(x,t,u),t)
D_xt=diff(U(x,t,u),x)+u_xt*diff(U(x,t,u),x)+u_xtt*diff(U(x,t,u),x)
D_xxt=diff(U(x,t,u),x)+u_xxt*diff(U(x,t,u),x)+u_xxtt*diff(U(x,t,u),x)
D_xtt=diff(U(x,t,u),x)+u_xtt*diff(U(x,t,u),x)+u_xttt*diff(U(x,t,u),x)
# 2)
D_xT=diff(T(x,t,u),x)+u_x*diff(T(x,t,u),x)+u_xt*diff(T(x,t,u),x)+u_xxt*diff(T(x,t,u),x)+u_xxtt*diff(T(x,t,u),x)+u_xxttt*diff(T(x,t,u),x)+u_xttt*diff(T(x,t,u),x)
D_xt=diff(T(x,t,u),x)+u_xt*diff(T(x,t,u),x)+u_xxt*diff(T(x,t,u),x)+u_xxtt*diff(T(x,t,u),x)
D_xTt=diff(T(x,t,u),x)+u_xT*diff(T(x,t,u),x)+u_xt*diff(T(x,t,u),x)+u_xxt*diff(T(x,t,u),x)+u_xxtt*diff(T(x,t,u),x)
D_xtt=diff(T(x,t,u),x)+u_xtt*diff(T(x,t,u),x)+u_xttt*diff(T(x,t,u),x)+u_xtttt*diff(T(x,t,u),x)
# 3)
D_xX=diff(X(x,t,u),x)+u_x*diff(X(x,t,u),x)+u_xt*diff(X(x,t,u),x)+u_xxt*diff(X(x,t,u),x)+u_xxtt*diff(X(x,t,u),x)
D_xt=diff(X(x,t,u),x)+u_xt*diff(X(x,t,u),x)+u_xxt*diff(X(x,t,u),x)+u_xxtt*diff(X(x,t,u),x)+u_xxttt*diff(X(x,t,u),x)
D_xtt=diff(X(x,t,u),x)+u_xtt*diff(X(x,t,u),x)+u_xttt*diff(X(x,t,u),x)+u_xtttt*diff(X(x,t,u),x)
# A) (For first order pde)
U_x=D_xU u_t=D_xT u_x=D_xX
# B)
U_t=D_tU u_t=U_tD_tT u_x=D_tX
# C)
U_xt=D_xtU u_tt=D_xtT u_xt=D_xtX
# A) (For second order pde)
U_xx=D_xUx u_xt=U_xtT u_xx=D_xX
# B)
U_tt=D_tUt u_xt=U_xtT u_xt=D_xtX
# C)
U_xtt=D_xtUt u_xtt=U_xtT u_xtt=D_xtX
# A) (For third order pde)
U_xxx=D_xUxx u_xt=U_xtT u_xx=D_xX
# B)
U_xtt=D_tUtt u_xtt=U_xtT u_xtt=D_xtX
# C)
U_xttt=D_tUtt u_xttt=U_xtT u_xttt=D_xtX
Algorithm of Determining equations of infinitesimals for PDE of order one, two, and three using SageMath

\[ U_{,tt} = D_{,t} U_{,tt} + D_{,t} U_{,xt} + D_{,t} X \]

\[ X_2 = (x, t, u) \cdot \text{diff}(W, x) + (x, t, u) \cdot \text{diff}(W, t) + (x, x, u) \cdot \text{diff}(W, x) + (x, t) \cdot \text{diff}(W, u) + (x, u) \cdot \text{diff}(W, u) + (x, x, x) \cdot \text{diff}(W, x) + (x, x, t) \cdot \text{diff}(W, u) + (x, t, t) \cdot \text{diff}(W, u) + (x, x, x, x) \cdot \text{diff}(W, t) + (x, x, t, t) \cdot \text{diff}(W, u) + (x, t, t, t) \cdot \text{diff}(W, u) \]

\# print("The value of X2 is ")
\# show(X2==0)

if (A==u_{,x}) :
    K=X2(u_{,x}=w).simplify_full()
elif (A==u_{,t}) :
    K=X2(u_{,t}=w).simplify_full()
elif (A==u_{,xx}) :
    K=X2(u_{,xx}=w).simplify_full()
elif (A==u_{,xt}) :
    K=X2(u_{,xt}=w).simplify_full()
elif (A==u_{,tt}) :
    K=X2(u_{,tt}=w).simplify_full()
elif (A==u_{,xxx}) :
    K=X2(u_{,xxx}=w).simplify_full()
elif (A==u_{,xxt}) :
    K=X2(u_{,xxt}=w).simplify_full()
elif (A==u_{,xtt}) :
    K=X2(u_{,xtt}=w).simplify_full()
K=(numerator(K))
\# print("The coefficient of \(u_{,x}^3\) is ")
\# show(K==0)

print("The determining equations are given by")
F=[1,2,3,4]
E=[1,2]
L=[u_{,x},u_{,t},u_{,xx},u_{,xt},u_{,tt},u_{,xxx},u_{,xxt},u_{,xtt},u_{,ttt}]
I=[]
J=[]
v=[]

for i,j,k in itertools.product(L,L,L):
    if i!=j and j!=k:
        for a,b,c in itertools.product(F,F,F):
            s=\"i^\"a\+j^\"b\+k^\"c\"\n            e=\"i^\"a\+j^\"b\n            d=\"i\"a\n            I.append(s)
            J.append(e)
            v.append(d)

for m in I :
    if ((K.coefficient(m))!=0):
        # print("The coefficient of ",m," is")
        show(K.coefficient(m)==0)
        K=(K (K.coefficient(m)*m))
        .simplify_full()
for m in J :
    if ((K.coefficient(m))!=0):
        # print("The coefficient of ",m," is")
        show(K.coefficient(m)==0)
        K=(K (K.coefficient(m)*m))
        .simplify_full()
for m in v :
    if ((K.coefficient(m))!=0):
        # print("The coefficient of \(u_{,x}^0\) is")
        show(K==0)

5. Flowchart of algorithm

![Flowchart of algorithm](image)
Example 1 Consider the first order PDE \[ u_t = u_x^2. \] (5.1)
which is a nonlinear PDE.
The invariance condition in this case is
\[ X^{(1)}(u_t - u_x^2) = 0 \text{ when } u_t = u_x^2. \] (5.2)
where
\[ X^{(1)} = X \frac{\partial}{\partial x} + T \frac{\partial}{\partial t} + U \frac{\partial}{\partial u_x} + U [t] \frac{\partial}{\partial u_t} \] (5.3)

Input: We give input as \( u_t \) which is LHS and \( u_x^2 \) which is RHS of equation \( u_t = u_x^2 \) written in solved form.

Output: infinitesimals \( X, T, U \) are found by solving following set of equations.
\[ -2U_x - X_t = 0 \]
\[ -T_t + U_x + 2X_t = 0 \]
\[ 2T_x + X_u = 0 \]
\[ T_u = 0 \]
\[ U_t = 0 \] (5.4)
Solving the above determining equations we get infinitesimals [3].

Remark 5.1. For second order pde use symbol \( u_{xt} \) for \( u_{xt} \) and \( u_{tx} \)

Example 2 Consider the second order PDE [1]
\[ u_t = u_{xx} + F(u)u'' \neq 0. \] (5.5)
which is heat equation with source.
The invariance condition in this case is
\[ X^{(2)}(u_{xx} + F(u)u'' - u_t) = 0 \text{ when } u_{xx} = u_t - F(u). \] (5.6)
where
\[ X^{(2)} = X^{(1)} + U_{[x]} \frac{\partial}{\partial u_{xx}} + U_{[u]} \frac{\partial}{\partial u_{xt}} + \]
\[ U_{[t]} \frac{\partial}{\partial u_t} \] (5.7)

Input: We give input as \( u_{xx} \) which is LHS and \( u_t - F(u) \) which is RHS of equation \( u_{xx} = u_t - F(u) \) written in solved form.
Output: infinitesimals \( X, T, U \) are found by solving following set of equations.
\[ T_{uu} = 0 \]
\[ -2T_{uu} - 2X_u = 0 \]
\[ -2T_{u} = 0 \]
\[ 3F(u)X_u + 2U_{xx} - X_{xx} + X_t = 0 \]
\[ U_{uu} - 2X_{uu} = 0 \]
\[ -X_{uu} = 0 \]
\[ F(u)T_u - T_{xx} + T_t - 2X_t = 0 \]
\[ -2T_t = 0 \]
\[ UF_u(u) - F(u)U_x + 2F(u)X_x + U_{xx} - U_t = 0 \] (5.8)
Solving the above determining equations we get infinitesimals [1].

Remark 5.2. For third order pde use symbol
1. \( u_{xtt} \) for \( u_{txt} \) or \( u_{ttx} \) or \( u_{xtt} \)
2. \( u_{xxt} \) for \( u_{xxt} \) or \( u_{txx} \) or \( u_{txt} \)

Example 3 Consider the Korteweg-De Vries Equation [1]
\[ u_t + uu_x + u_{xxx} = 0. \] (5.9)
which is a third order nonlinear PDE.
The invariance condition in this case is
\[ X^{(3)}(u_t + uu_x + u_{xxx}) = 0 \text{ when } u_{xxx} = -u_t - uu_x. \] (5.10)
Algorithm of Determining equations of infinitesimals for PDE of order one, two, and three using SageMath — 300/301

where

\[
X^{(3)} = X^{(2)} + U_{[xx]} \frac{\partial}{\partial u_{xx}} + U_{[xt]} \frac{\partial}{\partial u_{xt}} + U_{[tt]} \frac{\partial}{\partial u_{tt}}
\]

\[+ U_{[ux]} \frac{\partial}{\partial u_{ux}} + U_{[ut]} \frac{\partial}{\partial u_{ut}} + U_{[tt]} \frac{\partial}{\partial u_{tt}} \tag{5.11}
\]

**Input:** We gives input as \( u_{xxx} \) which is LHS and \(-u_t-u_{xx} \) which is RHS of equation \( u_{xxx}=-u_t-u_{xx} \).

The algorithm given in the paper gives the output as the determining equations for finding infinitesimals by giving inputs as PDE in two independent variables \( x,t \) and dependent variable \( u \), of order one, two, and three written in the solved form.

The algorithm is very useful for researchers working with PDE using Lie symmetry method and open source SageMath software. The results can be extended for higher-order PDEs.

**References**

[1] Daniel J. Arigo, Symmetry Analysis of Differential Equations An Introduction, John Wiley and Sons, 2015.

[2] G.W. Bluman, S.Kumei, Symmetry and Differential equations,Applied Mathematical Sciences, Volume 81.

[3] Peter E. Hydon, Symmetry Methods for Differential Equations,Cambridge University Press,2005.
Algorithm of Determining equations of infinitesimals for PDE of order one, two, and three using SageMath — 301/301

[4] Razvan A. Mezei, An Introduction to SAGE Programming with Application to SAGE Interacts for Numerical Methods, John Wiley and Sons, 2016.

[5] Ted Kosan, Sage for Newbies, v1.23-02/17/08.

[6] R. Seshadri, T.Y. Na, Group Invariance in Engineering Boundary Value Problems, Springer 1985.

**********
ISSN(P):2319 – 3786
Malaya Journal of Matematik
ISSN(O):2321 – 5666
**********