Malaya Journal of Matematik, Vol. 9, No. 1, 295-301, 2021

https://doi.org/10.26637/MJM0901/0050

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] *SSR College of Arts Commerce and Science, Silvassa, U.T. of DNH and DD-396230, India.*

[2] *Ex Professor. Department of Mathematics, Veer Narmad South Gujarat University, Surat-395007, Gujarat, India.*

***Corresponding author**: [1] vskssr@gmail.com;

Article History: Received **26** December **2020**; Accepted **21** January **2021**

Contents

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 `https://rb.gy/nmufo8`. and can be used, using SageMath Cell, SageMath cloud .

2. Mathematical concepts

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

$$F(x,u,\underset{1}{u},\underset{2}{u},\dots\underset{k}{u}) = u_{i_1,i_2\dots i_l} - f(x,u,\underset{1}{u},\underset{2}{u},\dots,\underset{k}{u}) = 0. \tag{2.1}$$

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

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

$$\mathbb{X} = \xi_i \frac{\partial}{\partial x_i} + \eta \frac{\partial}{\partial u}. \tag{2.2}$$

be the infinitesimal generator of

$$\begin{aligned} x^* &= X(x,u;\varepsilon) \\ u^* &= U(x,u;\varepsilon). \end{aligned} \tag{2.3}$$

where ξ_i and η are infinitesimals.
Let

$$\begin{aligned} \mathbb{X}^{(k)} =& \xi_i \frac{\partial}{\partial x_i} + \eta \frac{\partial}{\partial u} + \eta_i^{(1)}(x,u,\underset{1}{u}) \frac{\partial}{\partial u_i} \\ &+ \dots + \eta^{(k)}_{i_1,i_2,\dots i_j} \frac{\partial}{\partial u_{i_1,i_2,\dots i_j}}. \end{aligned} \tag{2.4}$$

be the kth extended infinitesimal generator of "(2.2)" where $\eta_i^{(1)}$ and $\eta^{(k)}_{i_1,i_2,\dots i_j}$ are given by

$$\begin{aligned} \eta_i^{(1)} &= D_i\eta - (D_i\xi_j)u_j, \quad i = 1,2,\dots,n; \\ \eta^{(k)}_{i_1,i_2,\dots i_k} &= D_{i_k}\eta^{(k-1)} - (D_{i_k}\xi_j)u_{i_1,i_2,\dots i_{k-1}j}, \\ & i_l = 1,2,\dots,n; for\, l = 1,2,\dots,k \\ & with\, k = 2,3,\dots \end{aligned} \tag{2.5}$$

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

$$\mathbb{X}^{(k)} F(x,u,\underset{1}{u},\underset{2}{u},\dots\underset{k}{u}) = 0.$$

when

$$F(x,u,\underset{1}{u},\underset{2}{u},\dots\underset{k}{u}) = 0. \tag{2.6}$$

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
$\frac{\partial^2 u}{\partial xt}, \frac{\partial^2 u}{\partial tx}$	u_xt
$\frac{\partial^2 u}{\partial t^2}$	u_tt
$\frac{\partial^3 u}{\partial xxx}$	u_xxx
$\frac{\partial^3 u}{\partial xxt}, \frac{\partial^3 u}{\partial xtx}, \frac{\partial^3 u}{\partial txx}$	u_xxt
$\frac{\partial^3 u}{\partial xtt}, \frac{\partial^3 u}{\partial txt}, \frac{\partial^3 u}{\partial ttx}$	u_xtt
$\frac{\partial^3 u}{\partial ttt}$	u_ttt
ξ_1	X
ξ_2	T
η	U
$\eta_x^{(1)}, \eta_t^{(1)}$	$U_{[x]}, U_{[t]}$
$\eta_{xx}^{(2)}, \eta_{xt}^{(2)}, \eta_{tt}^{(2)}$	$U_{[xx]}, U_{[xt]}, U_{[tt]}$
$\eta_{xxx}^{(3)}, \eta_{xxt}^{(3)}, \eta_{xtt}^{(3)}, \eta_{ttt}^{(3)}$	$U_{[xxx]}, U_{[xxt]}, U_{[xtt]}, U_{[ttt]}$

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,ttt and u_i is not equal to
u_k")
var('x,t,u,u_x,u_t,u_xx,u_xt,u_tx,
u_tt,u_xxx,u_xxt,u_xtt,u_ttt,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='Insertu_i '),w=input_box
(default= u_t u*u_x,label='Insert f
```

```
 of eq u_i=f(u_j,u,x,t)')):
W=A (w)
# 1)
D_xU=diff(U(x,t,u),x)+u_x*
 diff(U(x,t,u),u)+(u_xx*diff
(U(x,t,u),u_x)+u_xt*diff(U(x,t,u)
,u_t))+(u_xxx*diff(U(x,t,u),u_xx)+
u_xxt*diff(U(x,t,u),u_xt)+u_xtt
*diff(U(x,t,u),u_tt))
D_tU=diff(U(x,t,u),t)+u_t*diff(
U(x,t,u),u)+(u_tt*diff(U(x,t,u),u_t)
+u_xt*diff(U(x,t,u),u_x))+(u_ttt*
 diff(U(x,t,u),u_tt)+u_xtt*diff
(U(x,t,u),u_xt)+u_xxt*diff(U(x,t,u)
,u_xx))
# 2)
D_xT=diff(T(x,t,u),x)+u_x*diff
(T(x,t,u),u)+(u_xx*diff(T(x,t,u),u_x)
+u_xt*diff(T(x,t,u),u_t))+(u_xxx*diff
(T(x,t,u),u_xx)+u_xxt*diff(T(x,t,u),
u_xt)+u_xtt*diff(T(x,t,u),u_tt))
D_tT=diff(T(x,t,u),t)+u_t*diff
(T(x,t,u),u)+(u_tt*diff(T(x,t,u),u_t)
+u_xt*diff(T(x,t,u),u_x))+(u_ttt
*diff(T(x,t,u),u_tt)+u_xtt*diff
(T(x,t,u),u_xt)+u_xxt*diff(T(x,t,u)
,u_xx))
# 3)
D_xX=diff(X(x,t,u),x)+u_x*diff(
X(x,t,u),u)+(u_xx*diff(X(x,t,u),u_x)
+u_xt*diff(X(x,t,u),u_t))+(u_xxx*
 diff(X(x,t,u),u_xx)+u_xxt*diff
(X(x,t,u),u_xt)+u_xtt*diff(X(x,t,u)
,u_tt))
D_tX=diff(X(x,t,u),t)+u_t*diff(X
(x,t,u),u)+(u_tt*diff(X(x,t,u),u_t)
+u_xt*diff(X(x,t,u),u_x))+(u_ttt*
 diff(X(x,t,u),u_tt)+u_xtt*diff(
X(x,t,u),u_xt)+u_xxt*diff(X(x,t,u)
,u_xx))
# A)(For first order pde)
U_x=D_xU  u_t*D_xT  u_x*D_xX
#print(" Value of U_x is")
#show(U_x)
# B)
U_t=D_tU  u_t*D_tT  u_x*D_tX
#print(" Value of U_t is")
#show(U_t)
# 4)(For second order pde)
D_xUx=diff(U_x,x)+u_x*diff(U_x,u)
+(u_xx*diff(U_x,u_x)
+u_xt*diff(U_x,u_t))+(u_xxx*diff
(U_x,u_xx)+u_xxt
*diff(U_x,u_xt)+u_xtt*diff(U_x,
u_tt))
D_tUt=diff(U_t,t)+u_t*diff(U_t,u)+
(u_tt*diff(U_t,u_t)
+u_xt*diff(U_t,u_x))+(u_ttt*diff
(U_t,u_tt)+u_xtt*diff(U_t,u_xt)+u_xxt
*diff(U_t,u_xx))
D_xUt=diff(U_t,x)+u_x*diff(U_t,u)+
(u_xx*diff(U_t,u_x)+u_xt*diff
(U_t,u_t))+(u_xxx*diff(U_t,u_xx)
+u_xxt*diff(U_t,u_xt)+u_xtt*diff
(U_t,u_tt))

# A)(For second order pde)
U_xx=D_xUx  u_xt*D_xT  u_xx*D_xX
#print(" Value of U_xx is")
#show(U_xx)
# B)
U_tt=D_tUt  u_tt*D_tT  u_xt*D_tX
#print(" Value of U_tt is")
#show(U_tt)
# C)
U_xt=D_xUt  u_tt*D_xT  u_xt*D_xX
#print(" Value of U_xt is")
#show(U_xt)

# 5)(For third order pde)
D_xUxx=diff(U_xx,x)+u_x*diff(U_xx,u)
+(u_xx*diff(U_xx,u_x)+u_xt*diff
(U_xx,u_t))+(u_xxx*diff(U_xx,u_xx)
+u_xxt*diff(U_xx,u_xt)+u_xtt*
 diff(U_xx,u_tt))

D_tUtt=diff(U_tt,t)+u_t*diff(U_tt,u)
+(u_tt*diff(U_tt,u_t)+u_xt*diff
(U_tt,u_x))+(u_ttt*diff(U_tt,u_tt)
+u_xtt*diff(U_tt,u_xt)+u_xxt*diff
(U_tt,u_xx))

D_xUxt=diff(U_xt,x)+u_x*diff(U_xt,u)
+(u_xx*diff(U_xt,u_x)+u_xt*diff
(U_xt,u_t))+(u_xxx*diff(U_xt,u_xx)
+u_xxt*diff(U_xt,u_xt)+u_xtt*
 diff(U_xt,u_tt))

D_tUxt=diff(U_xt,t)+u_t*diff(U_xt,u)
+(u_tt*diff(U_xt,u_t)+u_xt*diff
(U_xt,u_x))+(u_ttt*diff(U_tt,u_tt)
+u_xtt*diff(U_xt,u_xt)+u_xxt*
 diff(U_xt,u_xx))

# A) (For third order pde)
U_xxx=D_xUxx  u_xxt*D_xT  u_xxx*D_xX

U_ttt=D_tUtt  u_ttt*D_tT  u_xtt*D_tX

U_xxt=D_xUxt  u_xtt*D_xT  u_xxt*D_xX
```

```
U_xtt=D_tUxt  u_xtt*D_tT  u_xxt*D_tX
 #
X2=X(x,t,u)*diff(W,x)+T(x,t,u)*
diff(W,t)+U(x,t,u)*diff(W,u)+(U_x)
*diff(W,u_x)+(U_t)*diff(W,u_t)+
(U_xx)*diff(W,u_xx)+(U_xt)*diff
(W,u_xt)+(U_tt)*diff(W,u_tt)+
((U_xxx)*diff(W,u_xxx)+(U_xxt)*
diff(W,u_xxt)+(U_xtt)*diff
(W,u_xtt)+(U_ttt)*diff(W,u_ttt))
#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()
elif (A==u_ttt):
   K=X2(u_ttt=w).simplify_full()
K=(numerator(K))
#show(K.coefficient(u_x^3))
# print("The value of K 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
                e=i^a*j^b
                d=i^a
                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 "
         ,m,"is")
        show(K.coefficient(m)==0)
        K=(K (K.coefficient(m)*m)).
        simplify_full()
        #print("The coefficient of
        u_x^0","is")
        show(K==0)
```

5. Flowchart of algorithm

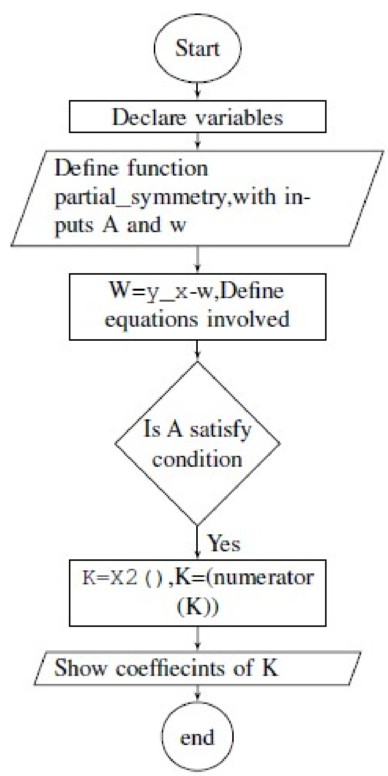


Figure 1. Flowchart of algorithm

Example 1 Consider the first order PDE [3]

$$u_t = u_x^2 . \tag{5.1}$$

which is a nonlinear PDE.
The invariance condition in this case is

$$\mathbb{X}^{(1)}(u_t - u_x^2) = 0 \,\text{when}\, u_t = u_x^2 . \tag{5.2}$$

where

$$\mathbb{X}^{(1)} = X\frac{\partial}{\partial x} + T\frac{\partial}{\partial t} + U\frac{\partial}{\partial u} + U_{[x]}\frac{\partial}{\partial u_x} + U_{[t]}\frac{\partial}{\partial u_t} \tag{5.3}$$

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.

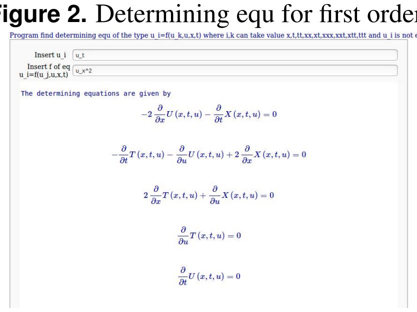


Figure 2. Determining equ for first order PDE

Output: infinitesimals X, T, U are found by solving following set of equations.

$$\begin{aligned} -2U - x - X_t &= 0 \\ -T_t - U_u + 2X_x &= 0 \\ 2T_x + X_u &= 0 \\ T_u &= 0 \\ U_t &= 0 \end{aligned} \tag{5.4}$$

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]

$$u_t = u_{xx} + F(u), F'' \neq 0 . \tag{5.5}$$

which is heat equation with source.
The invariance condition in this case is

$$\mathbb{X}^{(2)}(u_{xx} + F(u) - u_t) = 0 \,\text{when}\, u_{xx} = u_t - F(u) . \tag{5.6}$$

where

$$\mathbb{X}^{(2)} = \mathbb{X}^{(1)} + U_{[xx]}\frac{\partial}{\partial u_{xx}} + U_{[xt]}\frac{\partial}{\partial u_{xt}} + U_{[tt]}\frac{\partial}{\partial u_{tt}} . \tag{5.7}$$

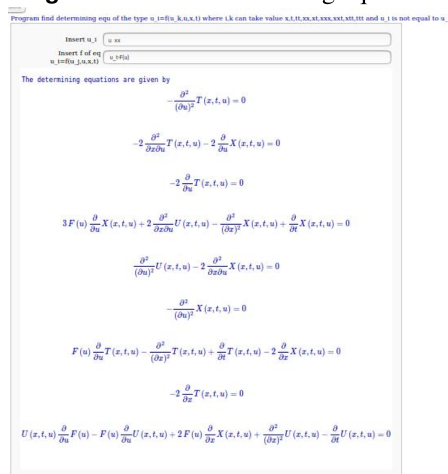


Figure 3. Set of Determining equations

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.

$$\begin{aligned} T_{uu} &= 0 \\ -2T_{xu} - 2X_u &= 0 \\ -2T_u &= 0 \\ 3F(u)X_u + 2U_{xu} - X_{xx} + X_t &= 0 \\ U_{uu} - 2X_{xu} &= 0 \\ -X_{uu} &= 0 \\ F(u)T_u - T_{xx} + T_t - 2X_x &= 0 \\ -2T_x &= 0 \\ UF_u(u) - F(u)U_u + 2F(u)X_x + U_{xx} - U_t &= 0 \end{aligned} \tag{5.8}$$

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]

$$u_t + uu_x + u_{xxx} = 0 . \tag{5.9}$$

which is a third order nonlinear PDE.
The invariance condition in this case is

$$\mathbb{X}^{(3)}(u_t + uu_x + u_{xxx}) = 0 \,\text{when}\, u_{xxx} = -u_t - uu_x . \tag{5.10}$$

where

$$\mathbb{X}^{(3)} = \mathbb{X}^{(2)} + U_{[xxx]}\frac{\partial}{\partial u_{xxx}} + U_{[xxt]}\frac{\partial}{\partial u_{xxt}} + U_{[xtt]}\frac{\partial}{\partial u_{xtt}} + U_{[ttt]}\frac{\partial}{\partial u_{ttt}} \quad (5.11)$$

Input: We gives input as `u_{xxx}` which is LHS and `-u_t-uu_x` which is RHS of equation `u_{xxx}=-u_t-uu_x`. **Output: infinitesimals X, T, U are found by solving following set of equations.**

Figure 4. Determining equ for third order PDE

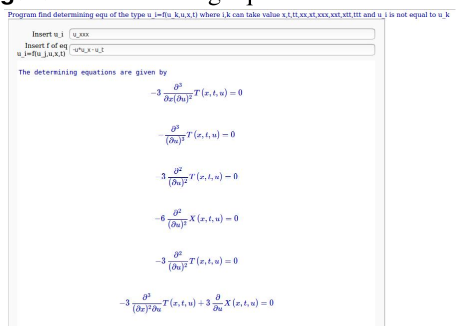


Figure 5. Determining equ for third order PDE

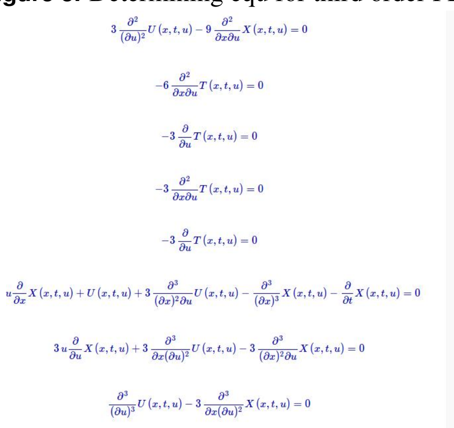
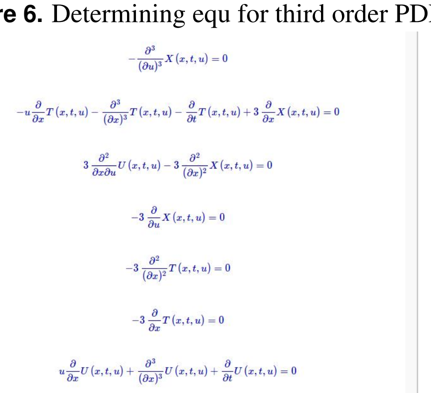


Figure 6. Determining equ for third order PDE



$$
\begin{aligned}
-3T_{xuu} &= 0\\
-T_{uuu} &= 0\\
-3T_{uu} &= 0\\
-6X_{uu} &= 0\\
-3T_{xxu}+3X_u &= 0\\
-3U_{uu}-9X_{xu} &= 0\\
-6T_{xu} &= 0\\
-3T_u &= 0\\
-3T_{xu} &= 0\\
-3T_u &= 0\\
2ux_x+U+3U_{xxu}-X_{xxx}-X_t &= 0\\
3uX_u+3U_{xuu}-3X_{xxu} &= 0\\
U_{uuu}-3X_{xuu} &= 0\\
-X_{uuu} &= 0\\
-uT_x-T_{xxx}-T_t+3X_x &= 0\\
3U_{xu}-3X_{xx} &= 0\\
-3X_u &= 0\\
-3T_{xx} &= 0\\
-3T_x &= 0\\
uU_x+U_{xxx}+U_t &= 0
\end{aligned} \quad (5.12)
$$

Solving the above determining equations we get infinitesimals [1].

Remark 5.3. *If PDE contains functions other than F, f then codes given in the algorithm can be modified and the new functions can be added in # Define function code.*

6. Conclusion

Finding infinitesimals associated with infinitesimal transformation admitted by PDE is the key step to solve PDEs using the Lie symmetry method.
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 Mathematcal Sciences*, Volume 81.

[3] Peter E. Hydon, Symmetry Methods for Differential Equations,*Cambrige University Press*,2005.

[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
⋆⋆⋆⋆⋆⋆⋆⋆⋆