In this example we use the package IntroToSymmetry.m to work out the point group of the PDE,

fxx+fyy-(c^2/4)f=0.

This equation comes up in connection with a model of a laminar jet diffusion flame with convection. See the 1993 paper by S. Mahalingam entitled "Self-similar diffusion flame including effects of streamwise diffusion", Combustion Science and Technology, 89: 363-373.

Clear all symbols in the current context.

In[1]:=
```
ClearAll[Evaluate[Context[] <> "*"]]
```

First read in the package which is located in User Home Folder/Library/*Mathematica*/Applications/SymmetryAnalysis.

In[2]:=
```
Needs["SymmetryAnalysis`IntroToSymmetry`"]
```

Enter the input equation as a string. Don't include the == 0 at the end.

In[3]:=
```
inputequation =
"D[f[x,y],x,x]+D[f[x,y],y,y]-(c^2/4)*f[x,y]";
```

The function f[x, y] is a solution of the equation and this constraint must be applied in the form of a rule to the invariance condition. Be careful to check signs.

In[4]:=
```
rulesarray =
  {"D[f[x,y],y,y]→-D[f[x,y],x,x]+(c^2/4)*f[x,y]"};
```

Enter the list of independent variables.

In[5]:=
```
independentvariables = {"x", "y"};
```

Enter the list of dependent variables.

In[6]:=
```
dependentvariables = {"f"};
```

Enter the list of function and/or constant names that need to be preserved when the equation is converted to generic y1[x1,x2] variables.

In[7]:=
```
frozennames = {"c"};
```

Enter the maximum derivative order of the input equation(s).

In[8]:=
```
p = 2;
```

The maximum derivative order that the infinitesimals are assumed to depend on is specified by the input parameter r. This parameter is only nonzero when the user is looking for Lie contact groups or Lie-Backlund groups. For the usual case where one is searching for point groups set r=0.

In[9]:=
```
r = 0;
```

When searching for Lie-Backlund groups (r=1 or greater one) can, without loss of generality, leave the independent variables untransformed. The corresponding infinitesimals (the xse's) are set to zero by setting xseon=0. If one is searching for point groups then set xseon=1. The choice xseon=1 is also an option when looking for Lie-Backlund groups and this can be useful when looking for contact symmetries.

In[10]:=
```
xseon = 1;
```

When searching for Lie-Backlund groups it is necessary to differentiate the input equation with respect to each independent variable producing derivatives of order p+r. These higher order differential consequences are appended to the set of rules applied to the invariance condition. This process is carried out automatically when internalrules=1. For point groups the equation or equation system is the only rule or set of rules needed and one sets internalrules=0.

In[11]:=
```
internalrules = 0;
```

Now work out the determining equations of the Lie point group that leaves the equation invariant. The output is available as a table of strings called zdeterminingequations.

In[12]:=
```
Timing[FindDeterminingEquations[
  independentvariables, dependentvariables, frozennames,
  p, r, xseon, inputequation, rulesarray, internalrules]]
```

```
The function FindDetermining Equations has
  begun, the memory in use = 92937768, the time used = 1.293549`
```

```
The function FindDeterminingEquations is nearly complete. The invariance condition
  has been created with all rules applied. The final step in the generation of
  the determining equations is to sum together terms in the table of invariance
  condition terms (called infinitesimaltable) that are multiplied by the
  same combination of products of free y derivatives. The result is the table
  infinitesimaltablesums corresponding to matching y-derivative expressions. If
  the invariance condition is long as it often is this process could take a long
  time since it requires sorting through the table infinitesimaltable once for each
  possible combination of y derivative products. This is the rate limiting step
  in the function FindDeterminingEquations.Virtually all other steps are quite
  fast including the generation of the extended derivatives of the infinitesimals.
```

```
The determining equations have been expressed in terms of z-variables, the
  length of zdeterminingequations = 13, the byte count of zdeterminingequations
  = 1720, the memory in use = 93455744, the time used = 1.345578`
```

```
FindDeterminingEquations is done. The memory in use = 93456936, the time used = 1.345749`
```

```
FindDeterminingEquations has finished executing. You can look at the output in the table
  zdeterminingequations. Each entry in this table is a determining equation in
  string format expressed in terms of z-variables. Rules for converting between
  z-variables and conventional variables are contained in the table ztableofrules.
  To view the determining equations in terms of conventional variables use the
  command ToExpression[zdeterminingequations]/.ztableofrules. There are two
  other items the user may wish to look at; the equation converted to generic
  (x1,x2,...,y1,y2,...) variables is designated equationgenericvariables and the
  various derivatives of the equation that appear in the invariance condition
  can be viewed in the table invarconditiontable. Rules for converting between
  z-variables and generic variables are contained in the table ztableofrulesxy.
```

Out[12]= {0.068599, Null}

In[13]:= **equationgenericvariables**

Out[13]= D[y1[x1,x2],x1,x1]+D[y1[x1,x2],x2,x2]-(c^2/4)*y1[x1,x2]

In[14]:= **invarconditiontable**

Out[14]= $\left\{0, 0, -\frac{c^2}{4}, 0, 0, 1, 0, 1\right\}$

The program expresses the determining equations in terms of zvariables. Here is the correspondence between z-variables and conventional variables.

In[15]:= **ztableofrules**

Out[15]= {z1 → x, z2 → y, z3 → f[x, y]}

Here are the determining equations expressed in terms of z-variables.The equations in the table can be distinguished by the == 0 at the end of each item.

In[16]:=
```
Column[zdeterminingequations]
```

Out[16]=
```
-4*Derivative[0, 0, 1][xse1][z1, z2, z3] == 0
-4*Derivative[0, 0, 1][xse2][z1, z2, z3] == 0
4*Derivative[0, 0, 1][xse2][z1, z2, z3] == 0
-2*Derivative[0, 0, 2][xse1][z1, z2, z3] == 0
-2*Derivative[0, 0, 2][xse2][z1, z2, z3] == 0
2*Derivative[0, 0, 2][eta1][z1, z2,
  z3] - 4*Derivative[0, 1, 1][xse2][z1, z2, z3] == 0
4*Derivative[0, 1, 0][xse2][z1, z2,
  z3] - 4*Derivative[1, 0, 0][xse1][z1, z2, z3] == 0
-4*Derivative[0, 1, 0][xse1][z1, z2,
  z3] - 4*Derivative[1, 0, 0][xse2][z1, z2, z3] == 0
2*Derivative[0, 0, 2][eta1][z1, z2,
  z3] - 4*Derivative[1, 0, 1][xse1][z1, z2, z3] == 0
-4*Derivative[0, 1, 1][xse1][z1, z2,
  z3] - 4*Derivative[1, 0, 1][xse2][z1, z2, z3] == 0
-(c^2*eta1[z1, z2, z3])/2 + (c^2*z3*Derivative[0, 0, 1][eta1][z1, z2, z3])/2
  - c^2*z3*Derivative[0, 1, 0][xse2][z1, z2, z3] + 2*Derivative[0, 2,
  0][eta1][z1, z2, z3] + 2*Derivative[2, 0, 0][eta1][z1, z2, z3] == 0
-(c^2*z3*Derivative[0, 0, 1][xse1][z1, z2, z3])/2 -
  2*Derivative[0, 2, 0][xse1][z1, z2, z3] + 4*Derivative[1, 0,
  1][eta1][z1, z2, z3] - 2*Derivative[2, 0, 0][xse1][z1, z2, z3] == 0
(-3*c^2*z3*Derivative[0, 0, 1][xse2][z1, z2, z3])/2 +
  4*Derivative[0, 1, 1][eta1][z1, z2, z3] - 2*Derivative[0, 2,
  0][xse2][z1, z2, z3] - 2*Derivative[2, 0, 0][xse2][z1, z2, z3] == 0
```

How many determining equations are there?

In[17]:=
```
Length[zdeterminingequations]
```

Out[17]= 13

Now solve the determining equations in terms of multivariable polynomials of a selected order. The Mathematica function Solve uses Gaussian elimination to solve a large number of linear equations for the polynomial coefficients. The time roughly follows

time/timeref=((number of equations)/(number of equationsref))^n

where the exponent is between 2.4 and 2.7. The Mathematica function Timing outputs the time required for the SolveDeterminingEquations function to execute.

In[18]:=
```
Timing[SolveDeterminingEquations[
  independentvariables, dependentvariables,
  r, xseon, zdeterminingequations, order = 4]]
```

```
The variable powertablelength is the number of terms required for each multivariate
  polynomial used for the infinitesimals. This number is determined by the choice
  of polynomial order and the number of zvariables. The time needed to solve the
  determining equations increases as powertable increases. powertablelength = 35

The polynomial expansions have been substituted into the determining
  equations. It is now time to collect the coefficients of various powers
  of zvariables into a table called table of coefficientsall. This step
  uses the function CoefficientList and is a fairly time consuming procedure.

The memory in use = 94836872, The time = 1.3952589999999998`

The number of unknown polynomial coefficients = 105

The number of equations for the polynomial coefficients = 225

Now it we are ready to use the function Solve
  to find the nonzero polynomial coefficients corresponding to
  the symmetries of the input equation(s). This can take a while.

The memory in use = 94968360, The time = 1.399741`

Solve has finished.

The function SolveDeterminingEquations is finished executing.

The memory in use = 95453624, The time = 1.451768`

You can look at the output in the tables xsefunctions and etafunctions.
  Each entry in these tables is an infinitesimal function in string format
  expressed in terms of z-variables and the group parameters. The output
  can also be viewed with the group parameters stripped away by looking
  at the table infinitesimalgroups. In either case you may wish to convert
  the z-variables to conventional variables using the table ztableofrules.

Keep in mind that this function only finds solutions of the determining equations that
  are of polynomial form. The determining equations may admit solutions that
  involve transcendental functions and/or integrals. Note that arbitrary functions
  may appear in the infinitesimals and that these can be detected by running the
  package function SolveDeterminingEquations for several polynomial orders. If
  terms of ever increasing order appear, then an arbitrary function is indicated.
```

Out[18]=
```
{0.081798, Null}
```

Here are the infinitesimals for the independent variables expressed in terms of z-variables.

In[19]:=
```
xsefunctions
```

Out[19]=
```
{xse1[z1_, z2_, z3_]=a10 + a15*z2, xse2[z1_, z2_, z3_]=a20 - a15*z1}
```

and the infinitesimals for the dependent variables.

```
In[20]:= etafunctions

Out[20]= {eta1[z1_, z2_, z3_]=b115*z3}
```

Here is a list of the various infinitesimal groups.

```
In[21]:= infinitesimalgroupsxy =
           infinitesimalgroups /. {z1 → x, z2 → y, z3 → f, z4 → fx, z5 → fy};

In[22]:= Column[infinitesimalgroupsxy]

         {{1, 0}, {0}}
Out[22]= {{y, -x}, {0}}
         {{0, 1}, {0}}
         {{0, 0}, {f}}
```

Check the solution in the determining equations.

```
In[23]:= Simplify[ToExpression[zdeterminingequations]]

Out[23]= {True, True, True, True, True, True, True, True, True, True, True, True, True}
```

Note that the infinitesimal polynomials truncate in spite of the linearity of the equation. In other words the order of the solution does not continue to increase as the order of the trial polynomial is increased. The reason for this is that the presence of f[x,y] in the last term of the equation rather than a derivative of f[x,y] makes it impossible for a polynomial of any order to satisfy the equation. Nevertheless the equation must admit the group {{0,0},{g[x,y]}} where g[x,y] is any solution of the equation. Now generate the commutator table.

```
In[24]:= MakeCommutatorTable[
          independentvariables, dependentvariables, infinitesimalgroupsxy]

         MakeCommutatorTable has finished executing. You can look at the output
           in the table commutatortable. To present the output in the most readable
           form you may want view it as a matrix using MatrixForm[commutatortable].
           Occasionally the entries in the commutatortable will have terms that cancel.
           To get rid of these terms use the function Simplify before viewing the table.
```

Put the commutator table in readable form.

```
In[25]:= MatrixForm[commutatortable]
```

Out[25]//MatrixForm=

{0, 0} {0}	{0, -1} {0}	{0, 0} {0}	{0, 0} {0}
{0, 1} {0}	{0, 0} {0}	{-1, 0} {0}	{0, 0} {0}
{0, 0} {0}	{1, 0} {0}	{0, 0} {0}	{0, 0} {0}
{0, 0} {0}	{0, 0} {0}	{0, 0} {0}	{0, 0} {0}

```
In[26]:= MaxMemoryUsed[]

Out[26]= 97 771 344
```