

In this example we use the package IntroToSymmetry.m to work out the point group of the 3-D Compressible Euler equations for isentropic flow.

$$U_t + U \cdot U_x + V \cdot U_y + W \cdot U_z + (1/\rho) \cdot P_x = 0,$$

$$V_t + U \cdot V_x + V \cdot V_y + W \cdot V_z + (1/\rho) \cdot P_y = 0,$$

$$W_t + U \cdot W_x + V \cdot W_y + W \cdot W_z + (1/\rho) \cdot P_z = 0,$$

$$P_t + U \cdot P_x + V \cdot P_y + W \cdot P_z + (5/3) \cdot P \cdot (U_x + V_y + W_z) = 0,$$

$$(\rho)_t + U \cdot (\rho)_x + V \cdot (\rho)_y + W \cdot (\rho)_z + \rho \cdot (U_x + V_y + W_z) = 0.$$

This next command turns off spurious spelling error warnings.

```
In[1]:= Off[General::spell]
```

Clear all symbols in the current context.

```
In[2]:= ClearAll[Evaluate[Context[] <> "*"]]
```

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

```
In[3]:= Needs["SymmetryAnalysis`IntroToSymmetry`"]
```

Enter the list of independent variables.

```
In[4]:= independentvariables = {"x", "y", "z", "t"};
```

Enter the list of dependent variables.

```
In[5]:= dependentvariables = {"U", "V", "W", "P", "rho"};
```

Enter the list of function and/or constant names that need to be preserved when the equation is expressed in generic variables.

```
In[6]:= frozennames = {};
```

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

```
In[7]:= rorder = 1;
```

The maximum derivative order that the infinitesimals are assumed to depend on is specified by the input parameter rorder. 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[8]:= rorder = 0;
```

When searching for Lie-Backlund groups (rorder=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[9]:= xseon = 1;
```

When searching for Lie-Backlund groups it is necessary to differentiate the input equations with respect to all the independent variables to produce derivatives of order  $porder+rorder$ . 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[10]:= internalrules = 0;
```

Construct the array of rules to be applied to the invariance condition.

```
In[11]:= rulesarray=
{"D[U[x,y,z,t],t]->-
U[x,y,z,t]*D[U[x,y,z,t],x]-
V[x,y,z,t]*D[U[x,y,z,t],y]-
W[x,y,z,t]*D[U[x,y,z,t],z]-
(1/rho[x,y,z,t])*D[P[x,y,z,t],x]",

"D[V[x,y,z,t],t]->-
U[x,y,z,t]*D[V[x,y,z,t],x]-
V[x,y,z,t]*D[V[x,y,z,t],y]-
W[x,y,z,t]*D[V[x,y,z,t],z]-
(1/rho[x,y,z,t])*D[P[x,y,z,t],y]",

"D[W[x,y,z,t],t]->-
U[x,y,z,t]*D[W[x,y,z,t],x]-
V[x,y,z,t]*D[W[x,y,z,t],y]-
W[x,y,z,t]*D[W[x,y,z,t],z]-
(1/rho[x,y,z,t])*D[P[x,y,z,t],z]",

"D[P[x,y,z,t],t]->-
U[x,y,z,t]*D[P[x,y,z,t],x]-
V[x,y,z,t]*D[P[x,y,z,t],y]-
W[x,y,z,t]*D[P[x,y,z,t],z]-
(5/3)*P[x,y,z,t]*D[U[x,y,z,t],x]-
(5/3)*P[x,y,z,t]*D[V[x,y,z,t],y]-
(5/3)*P[x,y,z,t]*D[W[x,y,z,t],z]",

"D[rho[x,y,z,t],t]->-
U[x,y,z,t]*D[rho[x,y,z,t],x]-
V[x,y,z,t]*D[rho[x,y,z,t],y]-
W[x,y,z,t]*D[rho[x,y,z,t],z]-
rho[x,y,z,t]*D[U[x,y,z,t],x]-
rho[x,y,z,t]*D[V[x,y,z,t],y]-
rho[x,y,z,t]*D[W[x,y,z,t],z]"};
```

Enter the x-momentum equation as a string.

```
In[12]:= inputequation1=
"D[U[x,y,z,t],t]+
U[x,y,z,t]*D[U[x,y,z,t],x]+
V[x,y,z,t]*D[U[x,y,z,t],y]+
W[x,y,z,t]*D[U[x,y,z,t],z]+
(1/rho[x,y,z,t])*D[P[x,y,z,t],x]";
```

Enter the y-momentum equation as a string.

```
In[13]:= inputequation2=
"D[V[x,y,z,t],t]+
U[x,y,z,t]*D[V[x,y,z,t],x]+
V[x,y,z,t]*D[V[x,y,z,t],y]+
W[x,y,z,t]*D[V[x,y,z,t],z]+
(1/rho[x,y,z,t])*D[P[x,y,z,t],y]";
```

Enter the z-momentum equation as a string.

```
In[14]:= inputequation3=
"D[W[x,y,z,t],t]+
U[x,y,z,t]*D[W[x,y,z,t],x]+
V[x,y,z,t]*D[W[x,y,z,t],y]+
W[x,y,z,t]*D[W[x,y,z,t],z]+
(1/rho[x,y,z,t])*D[P[x,y,z,t],z]";
```

Enter the pressure equation (actually the energy equation) as a string.

```
In[15]:= inputequation4=
"D[P[x,y,z,t],t]+
U[x,y,z,t]*D[P[x,y,z,t],x]+
V[x,y,z,t]*D[P[x,y,z,t],y]+
W[x,y,z,t]*D[P[x,y,z,t],z]+
(5/3)*P[x,y,z,t]*D[U[x,y,z,t],x]+
(5/3)*P[x,y,z,t]*D[V[x,y,z,t],y]+
(5/3)*P[x,y,z,t]*D[W[x,y,z,t],z]";
```

Enter the continuity equation as a string.

```
In[16]:= inputequation5=
"D[rho[x,y,z,t],t]+
U[x,y,z,t]*D[rho[x,y,z,t],x]+
V[x,y,z,t]*D[rho[x,y,z,t],y]+
W[x,y,z,t]*D[rho[x,y,z,t],z]+
rho[x,y,z,t]*D[U[x,y,z,t],x]+
rho[x,y,z,t]*D[V[x,y,z,t],y]+
rho[x,y,z,t]*D[W[x,y,z,t],z]";
```

Now work out the determining equations of the Lie point group which leaves equation1 invariant. The Mathematica function Timing outputs the time required for the FindDeterminingEquations function to execute.

```
In[17]:= Timing[FindDeterminingEquations[
independentvariables,dependentvariables,frozennames,porder,rorder,xseon,
inputequation1,rulesarray,internalrules]]
```

The function FindDetermining Equations has begun,  
the memory in use = 92950872, the time used = 3.6929390000000004`

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 = 38, the byte count of zdeterminingequations = 12264, the memory in use = 94855136, the time used = 3.9736780000000005`

FindDeterminingEquations is done. The  
memory in use = 94858424, the time used = 3.9740010000000003`

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[17]= {0.294162, Null}
```

The determining equations of equation1.

```
In[18]:= zdeterminingequations1=zdeterminingequations;
```

Now work out the determining equations of the Lie point group which leaves equation2 invariant.

```
In[19]:= Timing[FindDeterminingEquations[
independentvariables,dependentvariables,frozensnames,porder,rorder,xseon,
inputequation2,rulesarray,internalrules]]
```

The function FindDetermining Equations has begun,

```
the memory in use = 94939080, the time used = 3.9898879999999997`
```

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 = 38, the byte count of zdeterminingequations
= 12264, the memory in use = 95201336, the time used = 4.281542`
```

FindDeterminingEquations is done. The memory in use = 95203744, the time used = 4.281816`

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[19]= {0.302821, Null}
```

The determining equations of equation2.

```
In[20]:= zdeterminingequations2=zdeterminingequations;
```

Now work out the determining equations of the Lie point group which leaves equation3 invariant.

```
In[21]:= Timing[FindDeterminingEquations[
independentvariables,dependentvariables,frozensnames,porder,rorder,xseon,
inputequation3,rulesarray,internalrules]]
```

The function FindDetermining Equations has

```
begun, the memory in use = 95087248, the time used = 4.293192`
```

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 = 38, the byte count of zdeterminingequations = 12264, the memory in use = 95290208, the time used = 4.5766160000000005`

FindDeterminingEquations is done. The memory in use = 95292616, the time used = 4.576892`

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[21]= {0.294153, Null}
```

The determiningequations of equation3.

```
In[22]:= zdeterminingequations3=zdeterminingequations;
```

Now work out the determining equations of the Lie point group which leaves equation4 invariant.

```
In[23]:= Timing[FindDeterminingEquations[
independentvariables,dependentvariables,frozennames,porder,rorder,xseon,
inputequation4,rulesarray,internalrules]]
```

The function FindDetermining Equations has begun,  
the memory in use = 95064272, the time used = 4.594429000000001`

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 = 51, the byte count of zdeterminingequations = 18880, the memory in use = 96661824, the time used = 5.039969`

FindDeterminingEquations is done. The memory in use = 96664704, the time used = 5.0402960000000006`

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[23]= {0.463422, Null}
```

The determiningequations of equation3.

```
In[24]:= zdeterminingequations4=zdeterminingequations;
```

Now work out the determining equations of the Lie point group which leaves equation5 invariant.

```
In[25]:= Timing[FindDeterminingEquations[
independentvariables,dependentvariables,frozensnames,porder,rorder,xseon,
inputequation5,rulesarray,internalrules]]
```

The function FindDetermining Equations has

```
begun, the memory in use = 96461392, the time used = 5.058441`
```

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 = 57, the byte count of zdeterminingequations = 18904, the memory in use = 96634112, the time used = 5.4886170000000005`

FindDeterminingEquations is done. The memory in use = 96636904, the time used = 5.488901`

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[25]= {0.441507, Null}
```

The determining equations of equation3.

```
In[26]:= zdeterminingequations5=zdeterminingequations;
```

Collect the determining equations together.

```
In[27]:= zdeterminingequations6=
Join[zdeterminingequations1,
zdeterminingequations2,zdeterminingequations3,
zdeterminingequations4,zdeterminingequations5];
```

How many are there?

```
In[28]:= Length[zdeterminingequations6]
```

```
Out[28]= 222
```



In[29]:= **ztableofrules**

Out[29]= {z1 → x, z2 → y, z3 → z, z4 → t, z5 → U[x, y, z, t], z6 → V[x, y, z, t],  
z7 → W[x, y, z, t], z8 → P[x, y, z, t], z9 → rho[x, y, z, t]}

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

$$\text{time/timeref} = ((\text{number of equations}) / (\text{number of equationsref}))^n$$

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[30]:= Timing[SolveDeterminingEquations[
independentvariables,dependentvariables,rorder,xseon,zdeterminingequations6,order=2]
```

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 = 55`

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 = 101137352, The time = 5.811451`

The number of unknown polynomial coefficients = 495

The number of equations for the polynomial coefficients = 2097

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 = 111630200, The time = 6.0372200000000005`

`Solve` has finished.

The function `SolveDeterminingEquations` is finished executing.

The memory in use = 117468504, The time = 6.360132`

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[30]= {0.87591, Null}
```

Here are the infinitesimal transformation functions for the independent variables.

In[31]= **xsefunctions**

```
Out[31]= {xse1[z1_, z2_, z3_, z4_, z5_, z6_, z7_, z8_,
          z9_]=a10 + a11*z1 + a13*z2 + a16*z3 + a110*z4 + a111*z1*z4,
          xse2[z1_, z2_, z3_, z4_, z5_, z6_, z7_, z8_, z9_]=a20 -
          a13*z1 + a11*z2 + a26*z3 + a210*z4 + a111*z2*z4,
          xse3[z1_, z2_, z3_, z4_, z5_, z6_, z7_, z8_, z9_]=a30 -
          a16*z1 - a26*z2 + a11*z3 + a310*z4 + a111*z3*z4,
          xse4[z1_, z2_, z3_, z4_, z5_, z6_, z7_, z8_, z9_]=a40 + a410*z4 + a111*z4^2}
```

and for the dependent variables.

In[32]= **etafunctions**

```
Out[32]= {eta1[z1_, z2_, z3_, z4_, z5_, z6_, z7_, z8_, z9_]=a110
          + a111*z1 + (a11 - a410)*z5 - a111*z4*z5 + a13*z6 + a16*z7,
          eta2[z1_, z2_, z3_, z4_, z5_, z6_, z7_, z8_, z9_]=a210 + a111*z2
          - a13*z5 + (a11 - a410)*z6 - a111*z4*z6 + a26*z7,
          eta3[z1_, z2_, z3_, z4_, z5_, z6_, z7_, z8_, z9_]=a310 + a111*z3
          - a16*z5 - a26*z6 + (a11 - a410)*z7 - a111*z4*z7,
          eta4[z1_, z2_, z3_, z4_, z5_, z6_, z7_, z8_, z9_]=b436*z8 - 5*a111*z4*z8,
          eta5[z1_, z2_, z3_, z4_, z5_, z6_, z7_,
          z8_, z9_]=(-2*a11 + 2*a410 + b436)*z9 - 3*a111*z4*z9}
```

In[33]= **infinitesimalgroups1 = infinitesimalgroups /.**

```
{z1 -> x, z2 -> y, z3 -> z, z4 -> t, z5 -> U, z6 -> V, z7 -> W, z8 -> P, z9 -> rho};
```

Check the groups. xse's are on the left, eta's are on the right.

In[34]= **Column[infinitesimalgroups1]**

```
{ {1, 0, 0, 0}, {0, 0, 0, 0, 0} }
{ {x, y, z, 0}, {U, V, W, 0, -2 rho} }
{ {y, -x, 0, 0}, {V, -U, 0, 0, 0} }
{ {z, 0, -x, 0}, {W, 0, -U, 0, 0} }
{ {t, 0, 0, 0}, {1, 0, 0, 0, 0} }
{ {t x, t y, t z, t^2}, {-t U + x, -t V + y, -t W + z, -5 P t, -3 rho t} }
Out[34]= { {0, 1, 0, 0}, {0, 0, 0, 0, 0} }
{ {0, z, -y, 0}, {0, W, -V, 0, 0} }
{ {0, t, 0, 0}, {0, 1, 0, 0, 0} }
{ {0, 0, 1, 0}, {0, 0, 0, 0, 0} }
{ {0, 0, t, 0}, {0, 0, 1, 0, 0} }
{ {0, 0, 0, 1}, {0, 0, 0, 0, 0} }
{ {0, 0, 0, t}, {-U, -V, -W, 0, 2 rho} }
{ {0, 0, 0, 0}, {0, 0, 0, P, rho} }
```

Check the infinitesimals in the determining equations.

