An Introduction to DifferentialGeometry with Maple
PCMI - July 5, 2013

Getting Started

Synopsis

In this worksheet we will use the ordinary metric in 3-dimensional Euclidean space to illustrate some of the basic functionality and techniques for doing differential geometry calculations in Maple.

The Euclidean Metric

First we load the packages containing the commands we want.

```
> with(DifferentialGeometry): with(Tensor): with(LieAlgebras): with
  (GroupActions):
```

Every DG session begins by creating a computational environment, possibilities include coordinate charts for various types of manifolds, Lie algebras and so on.

Here we use DGsetup to create a 3 dimensional space with coordinates [x, y ,z].

```
> DGsetup([x, y, z], M);
```

$$\textit{frame name: } M \tag{1.1}$$

In differential geometry vectors are represented by directional derivative operators, for example

```
M > X := evalDG(a*D_x + b*D_y + c*D_z);
```

$$X := a\,D_x + b\,D_y + c\,D_z \tag{1.2}$$

The command evalDG is used parse the usual symbols + and * -- to tell Maple to re-interpret these as operations on differential geometry objects.

Here is the standard metric for Euclidean space.

```
M > g := evalDG(dx &t dx + dy &t dy + dz &t dz);
```

$$g := dx \otimes dx + dy \otimes dy + dz \otimes dz \tag{1.3}$$

Use TensorInnerProduct to find the length (squared) of the vector X with respect to the

metric g:

```
M > TensorInnerProduct(g, X, X);
```

$$a^2 + b^2 + c^2 \tag{1.4}$$

Every computer algebra system has a way to compute the curvature of a metric -- DG can do this in a wide range of contexts.

```
M > CurvatureTensor(g);
```

$$0\, D_x \otimes dx \otimes dx \otimes dx \tag{1.5}$$

In differential geometry, calculations which are very difficult in one system of coordinates can become simple in another
set of coordinates. It is important to be able to transform geometric objects from one coordinate system to another.

As an example, let's write our metric in spherical coordinates. First, define the coordinate chart.

```
M > DGsetup([rho, theta, phi], N);
```

$$\textit{frame name: } N \tag{1.6}$$

Create the change of coordinates as a transformation from one space to the other.

```
N > Phi := Transformation(N, M, [x = rho*cos(theta)*sin(phi), y =
    rho*sin(theta)*sin(phi), z = rho*cos(phi)]);
```

$$\Phi := [x = \rho \cos(\theta) \sin(\phi), y = \rho \sin(\theta) \sin(\phi), z = \rho \cos(\phi)] \tag{1.7}$$

In differential geometry, the procedures for transforming geometric objects from one space to another have names like pushforward and pullback.

```
N > g1 := Pullback(Phi, g);
```

$$g1 := d\rho \otimes d\rho + \rho^2 \sin(\phi)^2\, d\theta \otimes d\theta + \rho^2\, d\phi \otimes d\phi \tag{1.8}$$

Even though this metric looks a lot more complicated then our first metric, the curvature tensor still vanishes.

```
N > CurvatureTensor(g1);
```

$$0\, D_\rho \otimes d\rho \otimes d\rho \otimes d\rho \tag{1.9}$$

Isometries and Killing Vectors

Let's return to our original metric.

```
M > g ;
```

$$dx \otimes dx + dy \otimes dy + dz \otimes dz \tag{2.1}$$

A transformation of the space M which leaves the metric g unchanged is called an

isometry of g. The isometries of the Euclidean metric are translations and rotations.

Here are the translations along the x axis. They leave the metric invariant.

```
M > T_x := Transformation(M, M, [x = x + s, y = y, z = z]);
```

$$T_x := [x = x + s, y = y, z = z] \tag{2.2}$$

```
M > Pullback(T_x, g);
```

$$dx \otimes dx + dy \otimes dy + dz \otimes dz \tag{2.3}$$

Here the rotations in the xy plane. They leave the metric invariant

```
M > R_xy := Transformation(M, M, [x = cos(theta)*x + sin(theta)*
    y, y = -sin(theta)*x + cos(theta)*y, z = z]);
```

$$R_xy := [x = \cos(\theta)\, x + \sin(\theta)\, y, y = -\sin(\theta)\, x + \cos(\theta)\, y, z = z] \tag{2.4}$$

```
M > Pullback(R_xy, g);
```

$$dx \otimes dx + dy \otimes dy + dz \otimes dz \tag{2.5}$$

The "infinitesimal" isometries of a metric can be found by the KillingVectors command.

```
M > KV := KillingVectors(g);
```

$$KV := [-z\, D_x + x\, D_z, -z\, D_y + y\, D_z, D_z, -y\, D_x + x\, D_y, D_y, D_x] \tag{2.6}$$

```
M > KV[4], KV[6];
```

$$-y\, D_x + x\, D_y, D_x \tag{2.7}$$

The finite transformations for each of this infinitesimal transformations can be found by solving ordinary differential equations.

```
M > Flow(KV[4], theta);
```

$$[x = -y \sin(\theta) + x \cos(\theta), y = y \cos(\theta) + x \sin(\theta), z = z] \tag{2.8}$$

Lie Algebras

Because the isometries of any metric form a (Lie) group, the infinitesimal isometries of a metric inherit the structure of a Lie algebra. Let's see what this structure is for the Killing vectors of our metric g.

```
M > KV;
```

$$[-z\, D_x + x\, D_z, -z\, D_y + y\, D_z, D_z, -y\, D_x + x\, D_y, D_y, D_x] \tag{3.1}$$

```
M > LD1 := LieAlgebraData(KV, alg1):
M > DGsetup(LD1):
alg1 > MultiplicationTable("LieTable");
```

(3.2)

$$\begin{array}{c|cccccc} & e1 & e2 & e3 & e4 & e5 & e6 \\ \hline e1 & 0 & -e4 & e6 & e2 & 0 & -e3 \\ e2 & e4 & 0 & e5 & -e1 & -e3 & 0 \\ e3 & -e6 & -e5 & 0 & 0 & 0 & 0 \\ e4 & -e2 & e1 & 0 & 0 & e6 & -e5 \\ e5 & 0 & e3 & 0 & -e6 & 0 & 0 \\ e6 & e3 & 0 & 0 & e5 & 0 & 0 \end{array} \tag{3.2}$$

There is a very highly developed structure theory for studying Lie algebras and this structure theory is implemented in Maple. The first step is to find the Levi decomposition.

```
alg > Levi := LeviDecomposition(alg1);
```

$$Levi := [[e3, e5, e6], [e1, e2, e4]] \tag{3.3}$$

At each step one obtains a new basis for the Lie algebra in which the structure of the Lie algebra becomes more transparent.

```
alg1 > newBasis := [op(Levi[2]), op(Levi[1])];
```

$$newBasis := [e1, e2, e4, e3, e5, e6] \tag{3.4}$$

```
alg1 > LD2 := LieAlgebraData(newBasis, alg2):
alg > DGsetup(LD2);
```

$$Lie\ algebra: alg2 \tag{3.5}$$

```
alg2 > MultiplicationTable("LieTable");
```

$$\begin{array}{c|cccccc} & e1 & e2 & e3 & e4 & e5 & e6 \\ \hline e1 & 0 & -e3 & e2 & e6 & 0 & -e4 \\ e2 & e3 & 0 & -e1 & e5 & -e4 & 0 \\ e3 & -e2 & e1 & 0 & 0 & e6 & -e5 \\ e4 & -e6 & -e5 & 0 & 0 & 0 & 0 \\ e5 & 0 & e4 & -e6 & 0 & 0 & 0 \\ e6 & e4 & 0 & e5 & 0 & 0 & 0 \end{array} \tag{3.6}$$

The Killing Form of the semi-simple part is negative-definite and so the semi-simple part is the Lie algebra so(3).

```
alg2 > Killing([e1,e2,e3]);
```

$$\begin{bmatrix} -4 & 0 & 0 \\ 0 & -4 & 0 \\ 0 & 0 & -4 \end{bmatrix} \tag{3.7}$$

Invariant Metrics

In the previous section we started with a metric and found the isometry group and the Lie algebra of infinitesimal isometries or Killing vectors.

This construction can be reversed. We can start with a Lie algebra of vector fields and ask for the most general metric for which the given vector fields are Killing vectors. This construction is one of the most important methods for finding solutions to the Einstein equations and other geometrical equations in physics.

Here are our vector fields:

```
alg2 > KV;
```

$$[-z\,D_x + x\,D_z,\ -z\,D_y + y\,D_z,\ D_z,\ -y\,D_x + x\,D_y,\ D_y,\ D_x] \tag{4.1}$$

Here are the symmetric tensors:

```
M > SymTen := GenerateSymmetricTensors([dx, dy, dz], 2);
```

$$SymTen := \left[dx \otimes dx,\ \frac{1}{2}\,dx \otimes dy + \frac{1}{2}\,dy \otimes dx,\ \frac{1}{2}\,dx \otimes dz + \frac{1}{2}\,dz \otimes dx,\ dy \otimes dy,\ \frac{1}{2}\,dy \otimes dz + \frac{1}{2}\,dz \otimes dy,\ dz \otimes dz\right] \tag{4.2}$$

Here are the (invariant) symmetric tensors for which the KV are Killing vectors:

```
M > InvariantGeometricObjectFields(KV, SymTen, output = "list");
```

$$[dx \otimes dx + dy \otimes dy + dz \otimes dz] \tag{4.3}$$

Summary

- The DG interface uses standard notation and conventions of modern differential geometry.
- DG supports a wide range of commands for working with vector fields, differential forms and tensors
- One can move effortlessly between differential geometry and Lie algebra calculations
- Maple can solve the types of large overdetermined systems of over-determined partial differential equations that arise when calculating Killing vectors, invariant tensors and many other important geometric structures.