SI^2-SSE: Interdisciplinary Software Infrastructure for Differential Geometry, Lie Theory, and Applications

Ian M. Anderson and Charles G. Torre

Utah State University

OCI-1148331

February 2015

1. Overview

- The DifferentialGeometry (DG) software provides advanced mathematical functionalities for teaching and research in the areas of differential geometry, Lie theory, general relativity, and geometric methods for differential equations.
- DG contains extensive libraries and databases taken from mathematics and physics literature.
- The user interface uses notation and conventions from modern textbooks.
- Documentation serves not only as a manual but as an encyclopedia for the broader mathematics/physics communities.
- Development is driven by the research interests of the user community.

2. Sustainability

- Fully integrated with state of the art symbolic engine.
- DG is included in Maple distributed library.
- Cross-platform installations and compatibility.
- No black boxes - extensive documentation, all code and computations are user accessible and can be edited.
- Community-driven functionality.
- All compiled libraries now freely and publicly available.
- Open source development via new applications package.

3. Dissemination

- Regular tutorials at research conferences and institutes, e.g., PCMI, GR20, SRNI.
- Collaborations with researchers.
- Extensive layered documentation.
- Education: tutorials and lessons, classroom integration.
- DG website: how to's, advanced applications, DG worksheet archive.
- Demonstrate advanced functionalities via published research results.

4. Metrics, Feedback and Impact

- Website statistics ($>$ 2000 downloads).
- Bibliography of publications ($>$ 50).
- Tutorials at conferences, workshops and other venues.
- MaplePrimes.
- Advisory panel.

5. Accomplishments

- 20% increase in commands, 50% increase in exact solutions library.
- New environments for Lie algebras, homogeneous spaces, principal bundles.
- Structure theory for real and complex Lie algebras.
- Unique capabilities for Cartan normal connections and parabolic geometry.
- Critical components recoded to eliminate efficiency bottlenecks.
- Several new research results in mathematics and physics obtained using DG.

A Sample Computation

A sample calculation with the DG software illustrating some interdisciplinary features of the project.

First load some packages.

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

Define a coordinate system.

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

frame name: M (1)

Define a metric.

```
M > g := evalDG(1/y^2*(dx &t dx + dy&t dy) + dz &s dz);
```

$$g := \frac{dx\,dx}{y^2} + \frac{dy\,dy}{y^2} + dz\,dz \quad (2)$$

Use the Maple pdsolve command to integrate the Killing equations for the infinitesimal isometries.

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

$$KV := \left[D_z, \left(\frac{x^2}{2} - \frac{y^2}{2}\right) D_x + y\,x\,D_y, x\,D_x + y\,D_y, D_x\right] \quad (3)$$

Calculate the infinitesimal holonomy.

```
M > InfinitesimalHolonomy(g, [x = 0, y = 1, z = 0]);
```

$$\left[\left[\begin{array}{ccc} 0 & -\frac{1}{y^2} & 0 \\ \frac{1}{y^2} & 0 & 0 \\ 0 & 0 & 0 \end{array}\right]\right] \quad (4)$$

Find the structure equations for the Lie algebra of infinitesimal isometries.

```
M > LD := LieAlgebraData(KV, alg);
```

$$LD := [[e2, e3] = -e2, [e2, e4] = -e3, [e3, e4] = -e4] \quad (5)$$

```
M > DGsetup(LD);
```

Lie algebra: alg (6)

Find the Levi decomposition of the isometry algebra.

```
alg > LeviDecomposition(alg);
```

$$[[e1], [e2, e3, e4]] \quad (7)$$

Find the isotropy subalgebra. We see that our manifold is a **decomposable Riemannian symmetry space**.

```
alg > Iso := IsotropySubalgebra(KV, [x = 0, y = 1, z = 0],
      output = [alg]);
```

$$Iso := \left[e2 + \frac{1}{2}\,e4\right] \quad (8)$$

```
alg > Query(Iso, evalDG([e1, (1/2)*e2-(1/4)*e4, e3]),
      "SymmetricPair");
```

true (9)

Documentation

Tensor[PetrovType] - **determine the Petrov type of the Weyl tensor**

Calling Sequences

PetrovType(**g**, **pt**, **options**)
PetrovType(**NTetrad**, **pt**, **options**)
PetrovType(**NPWeyl**, **pt**, **options**)
PetrovType(**Fr**, **pt**, **options**)

Parameters

g	-	a metric tensor **g** of Lorentz signature $(1,-1,-1,-1)$ on a four-dimensional manifold **M**
Fr	-	the name of an initialized anholonomic frame, created from a null tetrad
NTetrad	-	a list of 4 vectors defining a null tetrad
NPWeyl	-	a table, the Newman-Penrose Weyl scalars
pt	-	(optional) a list of equations specifying the coordinates of a point of **M**
options	-	the keyword arguments **output**, **parameters**, **auxiliaryequations**

Description

- The Petrov classification of the Weyl tensor of the metric **g** is an algebraic classification based upon the eigenvalues and eigenvectors for the Weyl tensor, the principal null directions associated to the Weyl tensor, or the factorized form of the Weyl spinor. The Weyl tensor is determined to be one of the following types **"O"**, **"N"**, **"D"**, **"I"**, **"II"**, **"III"**. See Stephani, Kramer, et al for details.
- Because the algorithm is based upon the calculation of the Newman-Penrose Weyl scalars, it is best to use one of the last 3 calling sequences. The second and fourth calling sequences use a null tetrad $[L, N, M, \overline{M}]$, where L and N are real, M is the complex conjugate of M, and

$$g(L, N) = 1\,, \quad g(M, \overline{M}) = -1,$$

where g is the metric.
- Note that the Weyl scalars can often be simplified through the application of an appropriate null tetrad rotation. The first calling sequence **PetrovType(g, pt, option)** will automatically generate a null tetrad (if possible) but the result may not be optimal for calculating all the invariants needed to determine the Petrov type.
- It is possible that the Petrov type can vary from point to point on the manifold **M**. The command **PetrovType** returns, if possible, the Petrov type at a generic point. If the command fails, try to calculate Petrov type at a specific point with the optional argument **pt**. Constraints on the coordinates values can be imposed with assuming.
- The values of the invariants used to test for a given Petrov type will be returned with the keyword argument **output = petrovtype**, where **petrovtype** is one of **"O"**, **"N"**, **"D"**, **"I"**, **"II"**, **"III"**.
- If the metric or null tetrad depends upon a number of parameters (either constants or functions), then the keyword argument **parameters** = *ParList*, where *ParList* is the list of parameters, will invoke case-splitting with respect to these parameters. Special values of the parameters, where the Petrov type changes, are calculated. Additional algebraic or differential conditions may be imposed upon the parameters using the keyword argument **auxiliaryequations** = *EqList*. The case-splitting algorithm is as follows. First, the Petrov type for generic values of the parameters, independent of the auxiliary equations is determined. Then the algebraic and differential

Advanced Applications

THE DIFFERENTIALGEOMETRY SOFTWARE PROJECT

Cartan Subalgebras, Compact Roots and the Satake Diagrams for su(2,2).

Synopsis
In this worksheet we use the 15-dimensional real Lie algebra $su(2,2)$ to illustrate a some important points regarding the general structure theory and classification of real semi-simple Lie algebras.

1. Recall that a real semi-simple Lie algebra $\mathfrak{g}$ is called a compact Lie algebra if the Killing form is negative definite. If $\mathfrak{g}$ is compact, the all the root vectors for any Cartan subalgebra are pure imaginary. However, the converse is not true --- just because the root vectors are pure imaginary it is not true that the Lie algebra is compact.

2. A real semi-simple Lie algebra $\mathfrak{g}$ is called a split Lie algebra if there exists a Cartan subalgebra such that the root vectors are all real. Again, it is not true that if the root vectors are all real with respect one choice of Cartan subalgebra, then they are real with respect to any other choice.

3. Points 1 and 2 reflect the fact that for complex Lie algebras, all Cartan subalgebras are equivalence in the sense that they may be maped into each other by a Lie algebra automorphism. This is not true for real semi-simple Lie algebras.

4. To properly describe the structure theory for real Lie algebras, one must first calculate a Cartan decomposition $\mathfrak{g} = \mathfrak{t} + \mathfrak{p}$. Then one must chose a Cartan subalgebra $\mathfrak{h}$ such that $\mathfrak{h} = (\mathfrak{h} \cap \mathfrak{t}) \oplus (\mathfrak{h} \cap \mathfrak{p})$ and such that $\mathfrak{h} \cap \mathfrak{p}$ is of maximal dimension. With such a Cartan subalgebra one obtains the correct number of pure imaginary (or compact) roots and one can draw the proper Satake diagram.

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

The Matrix Algebra su(2, 2)

The matrix algebra $su(2,2)$ is the 15-dimensional real Lie algebra of 4×4 complex matrices M which are skew-hermitian with respect to the indefinite inner product