GRTensorII

GRTensorII Release 1.50
For MapleV Releases 3 and 4

A. Introduction and overview

Peter Musgrave
Denis Pollney
Kayll Lake

July 1996

Contents

Queen's University at Kingston, Ontario

GRTensorII is a package for the calculation and manipulation of components of tensors and related objects. Rather than focus upon a specific type or method of calculation, the program has been designed to operate efficiently for a wide range of applications and allows the use of a number of different mathematical formalisms. Algorithms are optimized for the individual formalisms and transformations between formalisms has been made simple and intuitive. Additionally, the package allows for customization and expansion with the ability to define new objects, user-defined algorithms, and add-on libraries.

Regardless of the algorithm or formalism used, it is often the case that only certain simplifications applied at a crucial stages can make some problems tractable. For this reason, GRTensorII has been designed to provide full control of the calculation path and of the simplifications to be performed at each stage. Once a calculation is completed, a variety of commands are available for the manipulation and simplification of results.

In designing the package, emphasis has also been placed on the interface, allowing simple user input, as well as presenting readable output. Metrics and basis vectors are easily defined and can be saved for later use. Calculations are specified in an intuitive manner using a minimal number of commands.

This booklet introduces a number of characteristics and features of the GRTensorII computer algebra package, emphasizing its input and output facilities, the ability to define new tensor objects, and the ability to use and switch between a number of different formalisms (classical tensor methods, general bases, and null tetrads). This booklet itself is intended only as a brief overview. More detailed information regarding the commands introduced here can be found in Booklets B–F:

Booklet *B: Specifying spacetimes*
C: Calculating tensor components
D: Defining new tensors
E: Bases and tetrads
F: Installation and setup

Together these articles are intended to serve as both a user's guide and command reference for the GRTensorII package. Unfortunately because of this dual purpose, there are some shortfalls in their ability to perform each task: Someone looking for tutorial information might find themselves buried in uninteresting technical points, while experienced users will find some sections overly pedagogic.

It is not intended that the booklets be read linearly from start to finish. To begin, only the information from the first sections of the *Specifying spacetimes* and *Calculating tensor components* booklets are needed to perform some sophisticated calculations. Once familiarity with the interface is gained, the ability to define new tensors and build customized libraries, described in Booklet *D: Defining new tensors*, greatly increase the power of the package. For the first time user, probably the best way to approach the package is to study and execute some of the demonstrations available online at the world-wide-web site [1], at the same time referring to these booklets and the online help as necessary.[1]

Release 1.50: GRTensorII was developed from the foundations laid by the GRTensor package for MapleV.2 in 1991. The GRTensorII package was first released in September of 1994. Updates

[1]For MapleV.3 versions, the online help library for GRTensorII is loaded into MapleV by the command `readlib(griihelp)`. It is described with installation instructions in Booklet *F: Installation and setup*. For MapleV.4 versions, the online help is automatically accessible.

 GRTensorII development has been supported by the Natural Science and Engineering Research Council of Canada and the Advisory Research Committee of Queen's University.

have been made intermittently since then, with a new set of release notes accompanying Version 1.1 in the summer of 1995. The current version, Release 1.50, marks an attempt to streamline and document the improvements to the package since its original appearance. Development has concentrated on improving users' ability to define new tensors by incorporating general index symmetries into automatically generated calculation programs. The new release also incorporates a number of internal changes, such as improved algorithms for the calculation of Newman-Penrose coefficients and scalar invariants. The online help library has been expanded and brought up to date and a new documentation set, of which this is the first booklet, has been written with an aim to provide a coherent description of the package as it stands in the summer of 1996.

The version of GRTensorII described in these booklets runs within the MapleV environment. It is supported for both Release 3 and 4 of MapleV. A Mathematica version of the GRTensorII software is also available.[2] It is intended that additions and improvements will continue to be made to the package and the individual 'booklet' format of this documentation has been chosen to facilitate expansion and document supplements. We encourage users to check back periodically for new versions of software, documentation, and demonstrations, at the GRTensorII world-wide-web site [1].[3]

1 Startup

The GRTensorII package is distributed as a set of MapleV libraries ('`.m`' files), the most significant of which is the file `grii.m`. This file contains the full library of GRTensorII commands to be described in this set of booklets, as well as a large standard library of calculable tensors and other objects.[4]

To start the GRTensorII package, in a new MapleV session type the commands:

```
> readlib ( grii ):
> grtensor():
```

The first of these commands loads the GRTensorII library, while the second initializes the package in addition to printing out a title page giving current version information. (Optionally, MapleV.3 users might also consider to execute the command `readlib(griihelp)` which loads the online help library, kept separate because of its size.)

The examples described below are designed to be run in a single GRTensorII session.

2 Calculating components

Tensors are specified by their name and index configuration, using the abbreviations `dn` and `up` to indicate covariant and contravariant indices, respectively. For instance, the Ricci tensor could be referenced

$$\texttt{R(dn,dn)}\quad (= R_{ab}),\qquad \texttt{R(up,dn)}\quad (= R^a{}_b),\qquad \texttt{R(bup,bdn)}\quad (= R^{(a)}{}_{(b)}),$$

[2]The Mathematica port of GRTensorII, called GRTensorM, is available from the world-wide-web site [1]. It differs from the MapleV version in both syntax and functionality. As such, these booklets should not be used as a reference for GRTensorM.

[3]We note parenthetically that some other unaffiliated software sites re-distribute versions of GRTensorII. We cannot assure the recency or integrity of such versions and recommend that GRTensorII software, updates, and documentation be downloaded from the home site.

[4]Instructions for the installation of this library are given in Booklet *F: Installation and setup*.

where the last example uses the labels `bup` and `bdn` to reference the components of the Ricci tensor in terms of a basis rather than metric coordinates.

Three commands are used most frequently within a GRTensorII session:

grcalc calculates the components of tensors;

grdisplay displays the components;

gralter applies simplification routines to tensors;

Thus, to calculate the covariant Ricci tensor for the predefined Kerr metric, we might use the command sequence

```
> qload ( kerr ):
> grcalc ( R(dn,dn) ):
   Calculating g(dn,dn,pdn) for kerr ...  Done.
   Calculating Chr(dn,dn,dn) for kerr ...  Done.
   Calculating detg for kerr ...  Done.
   Calculating g(up,up) for kerr ...  Done.
   Calculating Chr(dn,dn,up) for kerr ...  Done.
   Calculating R(dn,dn) for kerr ...  Done.

> grdisplay ( R(dn,dn) ):
```

For the kerr spacetime:
Covariant Ricci

$$R_{r\,r} = -a^2 \left(r^2 - r^2 \sin(\theta)^2 + 2\,r^2 \cos(\theta)^4 - 3\,r^2 \cos(\theta)^2 + 2\,r^2 \sin(\theta)^2 \cos(\theta)^2 + a^2 \cos(\theta)^4 - a^2 \cos(\theta)^2 + a^2 \cos(\theta)^2 \sin(\theta)^2\right) \Big/ \left(\left(-1 + \cos(\theta)^2\right)\,(r^2 - 2\,m\,r + a^2)\,\left(r^2 + a^2 \cos(\theta)^2\right)^2\right)$$

$R_{\theta\theta} = \ldots$etc.

At this point, since no simplifications of the results have been applied, the components of R_{ab} are not obviously zero as we would expect. The following commands are needed:

```
> gralter ( R(dn,dn), trig ):
> grdisplay ( R(dn,dn) ):
```

For the kerr spacetime:
Covariant Ricci
R(dn,dn) = All components are zero

The first command applies trigonometric simplification to the components of `R(dn,dn)`. Upon using `grdisplay()`, we see that the the simplification has indeed reduced the components to zero.

Note that for a more complicated metric, it may be the case that it is better to carry out the computation in stages. That is, we can first calculate the Christoffel symbols and apply simplifications, then the Ricci tensor, and finally, for instance, the Ricci scalar.

A number of alternate simplification routines exist within `gralter()`, including trigonometric, radical, expansion and factorization techniques. The command `grmap()` can also be used to apply *any* Maple function to the components of a tensor.

The commands listed in this Section are described in more detail in Booklet *C: Calculating tensor components*. The calculation of objects in a coordinate basis is described in Booklet *E: Bases and tetrads.*

3 Input of spacetimes

Spacetimes are specified within GRTensorII through use of the `makeg()` facility. They can be entered in the following forms:

Metric, g_{ab} A $n \times n$ symmetric matrix;

Line element An equation of the standard form $ds^2 = \ldots$;

Non-holonomic basis A set of basis vectors (covariant or contravariant) with an inner product;

Null tetrad As described by the Newman-Penrose formalism.

As an illustrative (but computationally trivial) example, we consider the twisting Einstein-Maxwell solution described on p. 135 of Kramer, et al. [2],

$$\begin{aligned} k ds^2 = &-2\left(x^a du - \frac{dy}{(a+1)x}\right)\left[dt + a(tdx+dy)/x + f(t)\left(x^a du - \frac{dy}{(a+1)x}\right)\right] \\ &+ (dx^2+dy^2)(t^2+1)/2x^2, \end{aligned}$$

We can enter this metric directly in this form using `makeg()`, giving it the name '`sm`':

```
> makeg ( sm ):
   Makeg 2.0:  GRTensorII metric/basis entry utility
   To quit makeg, type 'exit' at any prompt.

   Do you wish to enter a  1) metric [g(dn,dn)],
                           2) line element [ds],
                           3) non-holonomic basis [e1...e4], or
                           4) null tetrad [l,n,m,mbar]?

> 2:
   Enter coordinates as a LIST (e.g.  [r,theta,phi,t]):
> [ t, x, y, u ]:

   Enter the line element using 'd[coord]' to indicate differentials.
   (for example, r^2*(d[theta]^2 + sin(theta)^2*d[phi]^2)
   [Type 'exit' to quit makeg]
   ds^2 =

> 1/k*(-2*(x^a*d[u] - d[y]/((a+1)*x))*(d[t] + a*(t*d[x]
   + d[y])/x + f(t)*(x^a*d[u]-d[y]/((a+1)*x))) + (d[x]^2 +
   d[y]^2)*(t^2+1)/(2*x^2)):
```

Note that in the above expression the coordinate differentials are labeled using the variables `d[`*coordinate*`]`, allowing the line element to be entered in exactly the form that it is given in [2]. The output from `makeg` is:

$$\begin{aligned} ds^2 = {} & 2\frac{dt\ dy}{k(a+1)x} - 2\frac{x^a dt\ du}{k} + \left(\frac{1}{2}\frac{t^2}{kx^2} + \frac{1}{2}\frac{1}{kx^2}\right) dx^2 + 2\frac{a\ t\ dx\ dy}{k(a+1)x^2} \\ & - 2\frac{x^a a\ t\ dx\ du}{kx} + \left(2\frac{a}{k(a+1)x^2} - 2\frac{f(t)}{k(a+1)^2x^2} + \frac{1}{2}\frac{t^2}{kx^2} + \frac{1}{2}\frac{1}{kx^2}\right) dy^2 \\ & + 2\left(-\frac{x^a a}{kx} + 2\frac{x^a f(t)}{k(a+1)x}\right) dy\ du - 2\frac{(x^a)^2 f(t) du^2}{k} \end{aligned}$$

makeg completed.

The parameter values can now be set for the particular Type-III case studied by Siklos and MacCallum [3] (referred to in [2] on p. 135):

```
> a := 1/2:  k := -32/(78*lambda):  f(t) := (13*t^2+17)/32:
```

For this simple metric, it is not difficult to calculate the (rather uninteresting, in this case) components of the Ricci tensor:

```
> grcalc ( R(dn,dn) ):
> grdisplay ( _ ):
```

For the sm metric:

$$R_{ab} = \begin{bmatrix} 0 & 0 & -\frac{13}{8}\frac{1}{x} & \frac{39}{16}\sqrt{x} \\ 0 & -\frac{39}{32}\frac{t^2+1}{x^2} & -\frac{13}{16}\frac{t}{x^2} & \frac{39}{32}\frac{t}{\sqrt{x}} \\ -\frac{13}{8}\frac{1}{x} & -\frac{13}{16}\frac{t}{x^2} & -\frac{65}{192}\frac{5+t^2}{x^2} & -\frac{13}{128}\frac{5+13t^2}{\sqrt{x}} \\ \frac{39}{16}\sqrt{x} & \frac{39}{32}\frac{t}{\sqrt{x}} & -\frac{13}{128}\frac{5+13t^2}{\sqrt{x}} & \frac{39}{256}x(13t^2+17) \end{bmatrix}$$

The specification of metrics and bases is described more fully in Booklet *B: Specifying space-times*. The `grcalc()`, `gralter()` and `grdisplay()` commands are the subject of Booklet *C: Calculating tensor components*.

4 The Newman-Penrose formalism

A more complicated problem is that of determining the Petrov type of this metric. In order to do this in GRTensorII, the metric must be cast in the form of a Newman-Penrose (NP) null tetrad. Since no tetrad obviously presents itself from the form of the metric given above, we can use the command `nptetrad()` to generate one. This command constructs a set of null basis vectors satisfying the properties of a basis for the Newman-Penrose formalism.[5] Naturally, this tetrad may not be optimal for further calculation, but is often a good start to which further simplifications can be applied.

[5] Since the -2 signature convention of the NP formalism conflicts with the +2 Landau-Lifschitz convention recommended for metrics in GRTensorII, the construction of the tetrad also involves a signature change. This is carried out automatically if requested.

```
> nptetrad():
   The metric signature of the sm spacetime is +2.
   In order to create an NP-tetrad, the signature of g(dn,dn) will be changed
   to -2.
   Continue?  (1=yes):
> 1:
   The signature of the sm spacetime is now -2.
```

Null tetrad (covariant components)

$$l_a = \left[\ -\frac{39}{16}\,\lambda\,\sqrt{x} \quad -\frac{39}{32}\,\frac{\lambda\,t}{\sqrt{x}} \quad \frac{13}{256}\,\frac{\lambda\,(\,-7+13\,t^2\,)}{\sqrt{x}} \quad -\frac{507}{512}\,\lambda\,x\,t^2 - \frac{663}{512}\,\lambda\,x\ \right]$$

$$n_a = \left[\ 0 \quad 0 \quad -\frac{2}{3}\,\frac{1}{x^{3/2}} \quad 1\ \right]$$

$$m_a = \left[\ 0 \quad -\frac{1}{8}\,\frac{I\,\sqrt{3}\,\sqrt{13}\,\sqrt{\lambda}\,\sqrt{t^2+1}}{x} \quad \frac{1}{8}\,\frac{\sqrt{3}\,\sqrt{13}\,\sqrt{\lambda}\,\sqrt{t^2+1}}{x} \quad 0\ \right]$$

$$mbar_a = \left[\ 0 \quad -\frac{1}{8}\,\frac{I\,\sqrt{3}\,\sqrt{13}\,\sqrt{\lambda}\,\sqrt{t^2+1}}{x} \quad -\frac{1}{8}\,\frac{\sqrt{3}\,\sqrt{13}\,\sqrt{\lambda}\,\sqrt{t^2+1}}{x} \quad 0\ \right]$$

```
   The null tetrad has been stored as e(bdn,dn).
```

From this point we can proceed to determine the Petrov type of our example metric by first calculating the spin coefficients (`NPSpin` and their conjugates `NPSpinbar`) and the Weyl scalars (`WeylSc`) [4].

```
> grcalc ( NPSpin, NPSpinbar ):
> grcalc ( WeylSc ):
> gralter ( _ , factor ):
> grdisplay ( _ ):
```

$$\Psi 0 = -\frac{5}{64}\,\frac{I\,x\,(\,13\,t^2+10\,I\,t+11\,)}{(\,t+I\,)\,(\,-t+I\,)^2}$$

$$\Psi 1 = \frac{5}{52}\,\frac{\sqrt{3}\,\sqrt{13}\,\sqrt{x}\,(\,I\,t-1\,)}{\sqrt{\lambda}\,(\,t^2+1\,)^{3/2}}$$

$$\Psi 2 = 0$$

$$\Psi 3 = 0$$

$$\Psi 4 = 0$$

Note that the calculation of `NPSpin` and `NPSpinbar` did not need to be specified in a separate step, since this the calculation of these objects implicit in the calculation of `WeylSc`. However, in general a good rule of thumb is to calculate these quantities first and simplify them fully (not required in this case) before proceeding to the curvature scalars.

```
> grcalc ( Petrov ):
> grdisplay ( _ ):
```

For the sm spacetime:
Petrov Type = III (or simpler)

Note the caveat 'or simpler' in the final line of output. The calculation of the Petrov type involves determining whether a number of intermediate quantities can be evaluated to zero [5]. It is sometimes the case that complicated ways of expressing zero do not, in fact, reduce to zero within the computer algebra system. In such cases it will return a Petrov type which is unnecessarily specialized. To determine the reliability of the computer's evaluation, the `PetrovReport()` command is provided, which details the intermediate calculations performed in the determination of the Petrov type.

```
> PetrovReport():
```

The conclusion 'Petrov type = III (or simpler)' for the sm metric
was based on the following results:
Weyl scalar Psi0 could not be evaluated to zero.
Weyl scalar Psi1 could not be evaluated to zero.
Weyl scalar Psi2 = 0
Weyl scalar Psi3 = 0
Weyl scalar Psi4 = 0
– – – > Therefore the metric is Petrov III (or simpler).

We see from the previous output that the indeed the calculations have been carried out correctly (all of the terms which the computer believes to be non-zero are actually non-zero).

The use of null tetrads and general bases is treated in more detail in Booklet *E: Bases and tetrads*.

5 Defining new tensors

Finally, we describe a facility which exists within GRTensorII for defining tensors which are not part of the standard object libraries. Here we define the Bel-Robinson tensor,

$$T_{cdef} := C_{acdb} C^{a}{}_{ef}{}^{b} + C^{*}_{acdb} C^{*a}{}_{ef}{}^{b},$$

as well as it's trace and its divergence using the command `grdef()`:

```
> grdef (`T{(c d e f)} := C{a c d b}*C{^a e f^b}+ Cstar{a c d b}
   *Cstar{^a e f^b}`):
> grdef (`TT{(c d)} := T{^a a c d}`):
> grdef (`TC{(b c d)} := T{^a b c d ;a}`):
```

In the strings defining these tensors, contravariant indices are indicated by preceding them with a caret, '`^`', while covariant differentiation is indicated as usual, with a semi-colon, '`;`'. The braces on the left-hand side of the assignment in each command indicate that the newly defined tensors should be assumed to be symmetric under interchange of their indices. The calculation functions which are automatically generated by `grdef()` will take these symmetries into account to improve the efficiency of calculations.

We now load the Kerr-Newman metric and calculate these objects:

```
> qload(newkn):
```

Coordinates:
$x^a = [\ r\ u\ \phi\ t\ ]$
Line element

$$ds^2 = \frac{(r^2+u^2)dr^2}{r^2-2mr+a^2+Q^2} + \frac{(r^2+u^2)du^2}{a^2-u^2} + \frac{(a^2-u^2)\left(r^2+a^2+\frac{(a^2-u^2)(2mr-Q^2)}{r^2+u^2}\right)d\phi^2}{a^2}$$
$$- \frac{(a^2-u^2)(2mr-Q^2)d\phi dt}{a(r^2+u^2)} + \left(-1+\frac{2mr-Q^2}{r^2+u^2}\right)dt^2$$

Kerr–Newman Solution in Boyer–Lindquist coordinates ($u = a * \cos(\theta)$)

```
> grcalc ( T(dn,dn,dn,dn) ):
```

We display a component (e.g. T_{rrrr}):

```
> grcomponent ( T(dn,dn,dn,dn), [r,r,r,r] ):
```

$$6\frac{m^2r^2 - 2mrQ^2 + m^2u^2 + Q^4}{(r^2+u^2)^2(r^2-2mr+a^2+Q^2)^2}$$

We now check that ${T^a}_{acd} = 0$ and that $\nabla_a {T^a}_{bcd} = 0$ in vacuum:

```
> grcalc ( TT(dn,dn) ):
> grdisplay( _ ):
```

TT(dn,dn) = All components are zero

```
> Q:=0:
> grcalc ( TC(dn,dn,dn) ):
> grdisplay( _ ):
```

TC(dn,dn,dn) = All components are zero

A more thorough discussion of the definition of tensors in GRTensorII is to be found in Booklet *D: Defining new tensors.*

Conventions

GRTensorII can perform tensor calculations in any number of dimensions and for symmetric metrics of arbitrary signature. Tensor indices generally take values from $1 \ldots n$ in an n-dimensional spacetime.

In its definitions of curvature tensors, GRTensorII follows the Landau-Lifshitz spacelike conventions. That is, the curvature (Riemann) tensor is defined by

$$R^a{}_{bcd} := \frac{\partial \Gamma^a_{bd}}{\partial x^c} - \frac{\partial \Gamma^a_{bc}}{\partial x^d} + \Gamma^a_{ec}\Gamma^e_{bd} - \Gamma^a_{ed}\Gamma^e_{bc},$$

the Ricci tensor by

$$R_{ab} := R^c{}_{acb},$$

and the Einstein tensor by

$$G_{ab} := R_{ab} - \frac{1}{2} R g_{ab}.$$

Although spacetimes of any signature can be defined, for four dimensional Lorentzian metrics the signature +2 is recommended. Most standard curvature calculations will be carried out correctly regardless of the signature convention. However, some tensors (in particular the vector operators of Booklet *C: Calculating tensor components*) which depend on the normalization of timelike or spacelike vectors can be affected by the choice of signature. If a signature other than +2 is employed, users should carefully check tensor definitions for signature dependent terms.

The only deviation from this standard signature is in the specification of the Newman-Penrose formalism, which follows the original definitions of [4], and thus requires a -2 signature for spacetimes.

References

[1] GRTensorII software and documentation can be obtained free of charge from the ftp site at `astro.queensu.ca` in the `/pub/grtensor` directory, or from the world-wide-web page `http://astro.queensu.ca/~grtensor/`.

[2] D. Kramer, H. Stephani, Herlt E., and M. MacCallum. *Exact Solutions of Einstein's Field Equations.* Cambridge University Press, Cambridge, 1980.

[3] M. A. H. MacCallum and S. T. C. Siklos. Algebraically-special hypersurface-homogeneous Einstein spaces in general relativity. *J. Geom. Phys*, 8:221–242, 1981.

[4] E. T. Newman and R. Penrose. An approach to gravitational radiation by a method of spin coefficients. *J. Math. Phys.*, 3:896–902, 1962. (Errata 4:998, 1963).

[5] F. W. Letniowski and R. G. McLenaghan. An improved algorithm for quartic equation classification and Petrov classification. *Gen. Rel. Grav.*, 20:463–483, 1988.