GRTensorII

GRTensorII Release 1.50
For MapleV Releases 3 and 4

E. Bases and tetrads

Peter Musgrave
Denis Pollney
Kayll Lake

July 1996

Contents

Queen's University at Kingston, Ontario

The classical method of describing a spacetime is through the use of a covariant tensor, g_{ab}, which specifies the inner product between a pair of vectors in the space. An alternative description is obtained when one specifies a set of coordinate axes (vectors) at each point of the spacetime, as well as their inner product. Together, these are called a basis (or frame, or tetrad in four dimensions).

For the most part, GRTensorII calculations for a basis directly parallel those for a metric and the two formalisms have been described together wherever necessary in Booklets *A–D*. However, to this point most of the examples used in these documents have relied on calculations in metric coordinates. The goal of this booklet is to supplement the previous descriptions by emphasizing some of the unique features of symbolic calculations for spacetimes described by a basis in GRTensorII.

1 Notation

This section defines some notational conventions that are used to describe bases in this set of booklets. A concise summary of basis formalisms (using notation consistent with that described here) can be found in the first part of Chandrasekhar [**?**].

At each point of an n dimensional space, define a set of n independent vectors,

$$e_{(1)}{}^{a}, e_{(2)}{}^{a}, \dots , e_{(n)}{}^{a}.$$

Following convention, the vectors are labeled using bracketed indices, $(1), \dots , (n)$, which we will call the 'basis indices', to distinguish them from the 'tensor indices' which are not bracketed. The basis indices are raised by considering $e_{(a)}{}^{b}$ to form an $n \times n$ matrix whose inverse is given by the matrix $e^{(a)}{}_{b}$.

In addition to the basis vectors, define a symmetric $n \times n$ matrix $\eta^{(a)(b)}$ whose components determine the 'inner product' of basis vectors (a) and (b). The inner product and basis vectors together can be used to define a 2-index tensor via

$$\eta^{(a)(b)} e_{(a)}{}^{c} e_{(b)}{}^{d} =: g^{cd}. \tag{1}$$

This object, along with its matrix inverse, g_{ab}, is used to raise and lower the tensor indices of the basis vectors:

$$e_{(a)b} := e_{(a)}{}^{c} g_{cb}, \qquad e_{(a)}{}^{b} = e_{(a)c} g^{cb}.$$

Then by defining $\eta_{(a)(b)}$ to be the matrix inverse of $\eta^{(a)(b)}$, some manipulation of the above relationships results in the formulae for the raising and lowering of the basis indices:

$$e_{(a)b} = \eta_{(a)(c)} e^{(c)}{}_{b}, \qquad e^{(a)b} = \eta^{(a)(c)} e_{(c)}{}^{b}$$

The collection of vectors, $e_{(a)}{}^{b}$, and inner product, $\eta_{(a)(b)}$, together form a 'basis' which locally describes the spacetime geometry. The tensor g_{ab} is the corresponding spacetime metric that is calculated from the basis via Eq. (1).

Tensors fields can be projected onto the basis field in order to obtain their basis components:

$$T_{(a)(b)} = T_{cd} e_{(a)}{}^{c} e_{(b)}{}^{d}, \qquad T_{ab} = T_{(c)(d)} e^{(c)}{}_{a} e^{(d)}{}_{b}.$$

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

The basis components of a tensor are raised and lowered using the basis inner product,

$$T^{(a)}{}_{(b)} := \eta^{(a)(c)} T_{(c)(b)}, \qquad T^{(a)(b)} := T^{(a)}{}_{(c)} \eta^{(c)(b)}.$$

A computational advantage over tensor methods is offered here, since by choosing basis vectors correctly the inner product $\eta_{(a)(b)}$ can be constructed so as to have a simple form, most often with constant coefficients.[1]

The GRTensorII notation for the basis vectors $e_{(a)}{}^b$ is `e(bdn,up)`, following the convention that basis indices are labeled using `bdn` and `bup` to indicate covariant and contravariant components, respectively. The inner product, $\eta_{(a)(b)}$, is referenced using the object name `e(bdn,bdn)`. The creation of bases for GRTensorII is outlined in Booklet *B: Specifying spacetimes*.

A particularly useful form of basis (for general relativity) is given by an inner product whose basis vectors are chosen to satisfy the inner product

$$\eta^{(a)(b)} = \eta_{(a)(b)} := \begin{bmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 \\ 0 & 0 & -1 & 0 \end{bmatrix}$$

In this case the vectors forming the basis are seen to be null. By convention, they are labeled

$$l^a := e_{(1)}{}^a, \qquad n^a := e_{(2)}{}^a, \qquad m^a := e_{(3)}{}^a, \qquad \bar{m}^a := e_{(4)}{}^a.$$

A formalism for studying the geometries specified by such null tetrads was developed by Newman and Penrose [?]. As such, they are often called 'NP tetrads'. The standard GRTensorII library incorporates the full range of objects defined for the Newman-Penrose formalism, as listed in Section 4.

Null tetrads can be created using `makeg()` or automatically from a metric, g_{ab}, using the `nptetrad()` command. Rotations of a null tetrad can be performed using the command `nprotate()`. These commands are described in Booklet *B: Specifying spacetimes*.

2 Calculation of basis components

Calculations performed in a basis are carried out using the methods described in Booklet *C: Calculating tensor components*. A difference between the calculation of basis components and tensor components of an object is the use of basis index labels `bup` and `bdn` (to indicate contravariant and covariant components, respectively) rather than the tensor labels `up` and `dn`. For instance, once a basis is loaded, the command

```
> grcalc ( R(bdn,bdn) ):
```

requests a calculation of the object $R_{(a)(b)}$, the components of the Ricci tensor projected onto the basis.

Note that because the metric can be calculated from the basis via Eq. (1), if the geometry is described by a basis, then either basis or tensor components of an object can be calculated.

[1] In GRTensorII it is not necessary that $\eta_{(a)(b)}$ have constant coefficients. See the demonstration `mixmr.ms` available from the world-wide-web page [?].

That is, having loaded a basis one could also request the calculation

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

of the ordinary tensor components of the Ricci tensor, R_{ab}.[2]

If the basis calculation is requested, the algorithm used by GRTensorII to calculate the basis components of curvature tensors are chosen so as to take maximal use of the basis formalism, ie. they avoid the use of the metric.[3] However, if the tensor components have already been calculated, the program will recognize this and simply perform the conversion

$$R_{(a)(b)} = R_{cd} e_{(a)}{}^{c} e_{(b)}{}^{d}.$$

(The opposite conversion is used when the basis components are known and the tensor components are requested.)

The indices `pbdn` and `cbdn` specify the calculation of the basis components of partial and covariant derivatives, respectively. As described in the Booklet *C: Calculating tensor components*, the command

```
> grcalc ( R(bdn,bdn,pbdn) ):
```

requests the calculation of

$$R_{(a)(b),(c)} := \frac{\partial R_{(a)(b)}}{\partial x^d} e_{(c)}{}^{d},$$

while the command

```
> grcalc ( R(bup,bdn,cbdn) ):
```

requests the calculation of the covariant derivative

$$R^{(a)}{}_{(b);(c)} := R_{(a)(b),(c)} + \gamma^{(a)}{}_{(b)(d)} R^{(d)}{}_{(c)} - \gamma^{(d)}{}_{(b)(c)} R^{(a)}{}_{(d)},$$

defined in terms of the rotation coefficients,

$$\gamma_{(c)(a)(b)} := e_{(c)}{}^{d} e_{(a)d;e} e_{(b)}{}^{e}.$$

(See Section 4 for an alternate definition of this tensor.)

[2] A word of caution should be noted when carrying out such tensor component calculations from a basis. It is often the case that a metric calculated from a basis via Eq. (1) is not automatically expressed in the simplest possible from and could benefit from algebraic simplification. Since later calculations strongly depend on the form of the metric, it is usually worthwhile to perform an explicit simplification of the components of `g(dn,dn)` using the `gralter()` command. See Booklet *C: Calculating tensor components*, in particular the section on simplification strategies.

[3] In the particular case of the Ricci tensor, the definition is in terms of the rotation coefficients:

$$R_{(a)(b)} := -\gamma_{(a)}{}^{(c)}{}_{(b),(c)} + \gamma_{(a)}{}^{(c)}{}_{(c),(b)} + \gamma^{(c)}{}_{(a)(d)}[\gamma_{(b)}{}^{(d)}{}_{(c)} - \gamma_{(b)}{}^{(c)}{}_{(b)}] + \gamma_{(c)(a)(b)}\gamma^{(d)(c)}{}_{(d)} - \gamma_{(c)(a)(d)}\gamma^{(d)(c)}{}_{(b)}$$

See Section 4 for a definition of $\gamma_{(a)(b)(c)}$.

3 Bases in `grdef()`

Tensors and scalars which are not contained in the standard library can be defined for a basis using the `grdef()` command, just as they would in a metric space. The practical difference is that for indexed objects the basis indices are enclosed in round braces, `()`. Thus the basis components of the Ricci tensor, $R_{(a)(b)}$, are referenced using the string

```
R{(a) (b)}
```

rather than

```
R{a b}
```

which refers to the tensor components, R_{ab}. Some other points are also worth noting to make the use of `grdef()` more convenient.

In *four dimensions* objects corresponding to each of the basis vectors are defined by default when the spacetime is loaded or created:

- For contravariant null tetrads the objects `l(up)`, `n(up)`, `m(up)`, and `mbar(up)` are assigned the values of ${e_{(1)}}^a \ldots {e_{(4)}}^a$ respectively. The vectors can be displayed as a group using the object name `nullt(up)`. (For covariantly defined tetrads the `dn` versions of these vectors are assigned.)
- For general contravariant bases, the objects `e1(up)`, `e2(up)`, `e3(up)`, and `e4(up)` are assigned the values of ${e_{(1)}}^a \ldots {e_{(4)}}^a$ respectively. The vectors can be displayed as a group using the object name `basisv(up)`.
- For general covariant bases, the objects `w1(dn)`, `w2(dn)`, `w3(dn)`, and `w4(dn)` are assigned the values of $e_{(1)a} \ldots e_{(4)a}$ respectively. The vectors can be displayed as a group using the object name `basisv(dn)`.

While the information contained in these objects is redundant (in the sense that the components of the basis are already stored in the object `e(bdn,up)`), they are included to facilitate new object definitions in terms of these vectors. For instance, a 'purely-electric' electromagnetic tensor, F_{ab}, is defined by

$$F_{ab} := 4\lambda l_{[a} n_{b]},$$

which can be defined for GRTensorII by using the command

```
> grdef ( `F{[a b]} := 4*lambda*l{[a}*n{b]}` ):
```

An important point is that the information contained in these accessory objects (the vectors of `l(up)`, `e1(up)`, etc.) is copied from `e(bdn,up)` at the time the spacetime is loaded or created, but not afterwards. Thus, if the components of `e(bdn,up)` or `e(bdn,dn)` are modified (using `gralter()`, for example) the modification will not be reflected in the accessory objects. The opposite is also true: algebraic simplification of the `l(up)`... `mbar(up)`, for instance, will not be carried over to the corresponding components of `e(bdn,up)`.

The GRTensorII algorithms for the calculation of curvature tensors rely on the versions of the basis components stored in the objects `e(bdn,up)` and `e(bdn,dn)`. Thus, regardless of the form of the accessory objects, for calculation purposes it should be ensured that the components of `e(bdn,up)` and `e(bdn,dn)` are in the simplest possible form.

The above discussion applies only to four dimensional spacetimes, since the accessory basis vectors are not assigned automatically for any other number of dimensions. In such cases, the Kronecker delta, `kdelta`, can be used to isolate individual basis vectors. For instance, the command

```
> grdef ( `e3{^a} := e{(b) ^a}*kdelta{^(b) $3}` ):
```

assigns the third vector in `e(bdn,up)` to the object `e3(up)`:

$$e3^a := e_{(b)}{}^a \delta_3^{(b)}.$$

The use of `kdelta` to isolate components is described more fully in Booklet *D: Defining new tensors*.

The basis components of derivatives are specified in object definitions just as are the regular tensor derivatives. Thus, a comma, '`,`', indicates a partial derivative, while a semi-colon, '`;`', indicates a covariant derivative. Thus,

$$T_{(a)(b)(c)} := R_{(a)(b);(c)} + R_{(b)(c);(a)}$$

could be defined using the command

```
> grdef ( `T{(a)(b)(c)} := R{(a)(b);(c)} + R{(b)(c);(a)}` ):
```

The form of the following index will indicate which components (either basis or tensor) of the derivative are to be calculated. The operations specified by each type of derivative are listed in Booklet *D: Defining new tensors*.

Finally, note that for null tetrads, a set of derivatives operators are defined corresponding to partial derivatives along the basis vectors. The operators

$$D := l^a \partial_a, \qquad \Delta := n^a \partial_a, \qquad \delta := m^a \partial_a, \qquad \delta^* := \bar{m}^a \partial_a,$$

are defined by the GRTensorII operators

`Dl[]`, `Dn[]`, `Dm[]`, `Dmbar[]`,

respectively. The operators act on any GRTensorII object placed in the square braces.[4] For example, the right-hand side of the NP equation

$$\delta\tau - \Delta\sigma - (\mu\sigma + \lambda^*\rho) - \tau(\tau + \beta - \alpha^*) + \sigma(3\gamma - \gamma^*) + \kappa\nu^* - \Phi_{02} = 0,$$

could be defined using the command

```
> grdef ( `NPeq1 := Dm[NPtau] - Dn[NPsigma] - (NPmu*NPsigma + NPlambdabar*NPrho)
  - NPtau*(NPtau + NPbeta - NPalphabar) + NPsigma*(3*NPgamma - NPgammabar) +
  NPkappa*NPnubar - Phi02` ):
```

(See Section 4 for definitions of the objects `NPtau`, `NPsigma`, etc.)

[4] GRTensorII operators are described in Booklet *C: Calculating tensor components*.

4 The standard library

The following sections list the objects contained within the standard GRTensorII library which are defined specifically for use with bases. They can be calculated using `grcalc()` once a basis has been created or loaded into the current session (see Booklet *B: Specifying spacetimes*).

Objects not present in the following lists can be defined using the `grdef()` command, as described in the previous section. A further list of tensors calculable from a metric is given in Booklet *C: Calculating tensor components*.

General bases

Definitions listed in the following table are for objects which have not already been defined in the standard library list in Booklet *C: Calculating tensor components*. Basis definitions of the objects given in that table can be accessed by choice of indices. For instance, the basis components of the Riemann tensor, $R^{(a)}{}_{(b)(c)(d)}$ are calculated using the command

```
> grcalc ( R(bup,bdn,bdn,bdn) ):
```

GRTensorII name	Definition
`e(bdn,up)`	basis vectors, ${e_{(a)}}^b$
`eta(bdn,bdn)`	basis inner product, $\eta_{(a)(b)}$
`basisv(up)`	collection of basis vectors, ${e_{(1)}}^a \ldots {e_{(4)}}^a$
`lambda(bdn,bdn,bdn)`	$\lambda_{(a)(b)(c)} := e_{(b)[i,j]}{e_{(a)}}^d{e_{(c)}}^e$
`rot(bdn,bdn,bdn)`	rotation coefficients,
	$\gamma_{(a)(b)(c)} := \frac{1}{2}\left(\lambda_{(a)(b)(c)} + \lambda_{(c)(a)(b)} - \lambda_{(b)(c)(a)}\right)$
`str(bdn,bdn,bdn)`	structure constants,
	$C_{(a)(b)(c)} := \gamma_{(a)(c)(b)} - \gamma_{(a)(b)(c)}$

Four dimensional spacetimes which are specified in the form of a set of *contravariant* basis vectors additionally have the objects `e1(up)`, `e2(up)`, `e3(up)`, and `e4(up)` assigned, corresponding to the four basis vectors, ${e_{(a)}}^b$. If the spacetime was specified by a set of *covariant* basis vectors, then the objects `w1(dn)`, `w2(dn)`, `w3(dn)`, and `w4(dn)` are automatically assigned. For display purposes, the vectors can be accessed as a group using the object `basisv(up)`. See Section 3 for a description of these objects.

The Newman-Penrose formalism

In addition to the objects specified above, a number of special quantities can be calculated for geometries specified by a null tetrad. The following objects are defined in [**?**].

GRTensorII name	Definition
`l(up)`, `n(up)`, `m(up)`, `mbar(up)`	basis vectors,[5] $l^a, n^a, m^a, \bar{m}^a$.
`nullt(up)`	collection of basis vectors, $\{l^a, n^a, m^a, \bar{m}^a\}$.
`testNP(bdn,bdn)`	confirms NP inner product, $testNP_{(a)(b)} := e_{(a)c}e_{(b)}{}^c$.
`NPkappa, NPsigma, NPlambda, NPnu NPrho, NPmu, NPtau, NPpi, NPepsilon, NPgamma, NPalpha, NPbeta`	spin coefficients, $\kappa, \sigma, \dots, \beta$
`NPkappabar, NPsigmabar, NPlambdabar, NPnubar, NPrhobar, NPmubar, NPtaubar, NPpibar, NPepsilonbar, NPgammabar, NPalphabar, NPbetabar`	spin coefficients (complex conjugates) $\bar{\kappa}, \bar{\sigma}, \dots \bar{\beta}$
`Phi00, Phi01, Phi02, Phi10, Phi11, Phi12, Phi20, Phi21 Phi22, Lambda`	Ricci scalars, $\Phi_{00}, \Phi_{01}, \dots, \Phi_{22}, \Lambda$
`Psi0, Psi1, Psi2, Psi3, Psi4`	Weyl scalars, $\Psi_0, \dots, \Psi_4$
`Petrov`	Petrov type
`NPSpin`	collection of spin coefficients, $\{\kappa, \dots, \beta\}$
`NPSpinbar`	collection of spin coefficients $\{\bar{\kappa}, \dots, \bar{\beta}\}$
`RicciSc`	collection of Ricci scalars, $\{\Phi_{00}, \dots, \Phi_{22}, \Lambda\}$
`WeylSc`	collection of Weyl scalars, $\{\Psi_0, \dots, \Psi_4\}$

A set of four derivative operators, corresponding to partial derivatives along the basis vectors, are also defined for the Newman-Penrose formalism. In the following example the argument

[5] As explained in Section 3, the components of these basis vectors are copied from `e(bdn,up)` when the spacetime is loaded.

'*object*' represents an arbitrary GRTensorII object on which the operator is to act.

GRTensorII name	Definition
`Dl[`*object*`]`	$D := l^c\partial_c$
`Dn[`*object*`]`	$\Delta := n^c\partial_c$
`Dm[`*object*`]`	$\delta := m^c\partial_c$
`Dmbar[`*object*`]`	$\delta^* := \bar{m}^c\partial_c$

Alternate NP definitions

An alternate set of definitions for the Newman-Penrose coefficients is provided by Allen, et al. [**?**]. The definitions aim to gain a computational advantage by avoiding the inversion of the basis vectors (that is, only the covariant components are needed). These algorithms have been incorporated into GRTensorII and tested with results presented in [**?**]. It was found that no consistent computational advantage was gained through use of the alternative formulas, though in certain cases the covariant definitions do outperform the standard NP formulas. The alternate NP algorithms can be accessed by requesting the calculation of the objects listed in the table below. These names of these objects correspond to the NP names listed in the previous section prefixed by a '`C`'. Thus the command

```
> grcalc ( WeylSc ):
```

requests the calculation of the Weyl curvature coefficients using the formulas defined by Newman-Penrose, while the command

```
> grcalc ( CWeylSc ):
```

calculates the same objects using the alternate 'covariant' formulas.[6]

GRTensorII name	Definition

`Ckappa, Csigma, Clambda, Cnu`
`Crho, Cmu, Ctau, Cpi,`
`Cepsilon, Cgamma, Calpha, Cbeta`
	spin coefficients,
	$\kappa, \sigma, \ldots, \beta$

`Ckappabar, Csigmabar,`
`Clambdabar, Cnubar, Crhobar,`
`Cmubar, Ctaubar, Cpibar,`
`Cepsilonbar, Cgammabar,`
`Calphabar, Cbetabar`
	spin coefficients
	(complex conjugates)
	$\bar{\kappa}, \bar{\sigma}, \ldots \bar{\beta}$

[6] Note that if one of the above commands is executed in a given session, then GRTensorII will recognize the Weyl coefficients as having been calculated. Thus the use of the second command will return the previously calculated results. In order to compare the results of each command, they must be issued individually in separate GRTensorII sessions.

`CPhi00, CPhi01, CPhi02, CPhi10, CPhi11, CPhi12, CPhi20, CPhi21 CPhi22, CLambda`	Ricci scalars, $\Phi_{00}, \Phi_{01}, \ldots, \Phi_{22}, \Lambda$
`CPsi0, CPsi1, CPsi2, CPsi3, CPsi4`	Weyl scalars, $\Psi_0, \ldots, \Psi_4$
`CSpin`	collection of spin coefficients, $\{\kappa, \ldots, \beta\}$
`CSpinbar`	collection of spin coefficients $\{\bar{\kappa}, \ldots, \bar{\beta}\}$
`CRicciSc`	collection of Ricci scalars, $\{\Phi_{00}, \ldots, \Phi_{22}, \Lambda\}$
`CWeylSc`	collection of Weyl scalars, $\{\Psi_0, \ldots, \Psi_4\}$

The information contained in this booklet is also available from the following online help pages:

`?grt_basis`, `?grt_objects`, `?grt_operators`, `?grcalc`, `?grdef`, `?petrov`.