MTH 437/537 Spring 2009

COMPUTER ALGEBRA IN RELATIVITY

1. INTRODUCTION

Below is an introduction to the use of some computer algebra packages in relativity. The most easily accessible is probably the Maple package GRTensor (more properly GRTensorII), which I discuss first; there is also a *Mathematica* version called GRTensorM (which is not quite as good). I also discuss the Maple package `tensor`, which comes with Maple. Other available packages, discussed elsewhere, include the more specialized packages SHEEP and CLASSI, based on PSL (and hence closely related to REDUCE).

Some of these packages, such as `tensor` and SHEEP, do coordinate computations, whereas CLASSI does frame computations; GRTensor and GRTensorM can do either.

These instructions should work on any campus PCs, such as those in the MLC, which have Maple installed, (requires ONID account), on the COSINe UNIX computers (requires COSINe account), and on the physics UNIX computers (requires physics account).

2. PACKAGES BASED ON MAPLE

On UNIX machines, start Maple by typing `maple` for a shell interface, or `xmaple &` for the worksheet front end under XWindows. On PCs, there should be a Maple icon on the desktop. All Maple commands must end with either a semicolon or a colon; a colon suppresses output. All definitions must use `:=` , not just `=` . Exit Command Line Maple with the command `quit;` .

On COSINe machines, login to `app.science.oregonstate.edu`, not `frontend`.

Recent Maple installations start the Java interface by default, which does *not* work well with GRTensor. On UNIX machines, you can try typing `maple -cx` for the Classic interface, but this is not available on all machines; if you have trouble, use the shell interface. On PCs, use the Start menu to navigate to Programs and then Maple, where you should find shortcuts to both the Classic and Command Line interfaces.

a) GRTensor

GRTensor is freeware; if you have access to Maple elsewhere and would like to obtain a copy of GRTensor, let me know.

To enable Maple to find the GRTensor files on campus PCs, you will first need to add `\\poole\ClassFolders` as a network share (this step may be optional), then in Maple type the following command (note the many backslashes):

```
libname:=libname,"\\\\poole\\ClassFolders\\Math-Dray\\GRtensor\\lib":
```

On COSINe machines, instead type the command:

```
libname:=libname,"/home/math/tevian/GRtensor/lib":
```

and on physics machines type:

```
libname:=libname,"/home/tevian/GRtensor/lib":
```

Then type `readlib(grii):` and `grtensor();` to start GRTensor.

On UNIX machines, you can add the appropriate command to the file `.mapleinit` in your home directory. On PCs, there is a similar file called `maple.ini` , but it's not as straightforward to set up — seek help online or from me.

GRTensor contains an interactive program for constructing your own metric: Simply type `makeg(`*metricname*`);` , where *metricname* is the name you wish to call the metric. You will be prompted to enter the metric in one of several forms, and guided through the necessary steps; don't forget the semi-colon after each input! (To do coordinate-based computations, it is probably easiest to enter the line element; to use an orthonormal basis, choose "non-holonomic basis", then "covariant components", after which you will need to enter the orthonormal basis of 1-forms, followed by the metric, which is typically the diagonal matrix with entries $(-1\ 1\ 1\ 1)$.)

> WARNING: `makeg` does *not* work with the (default) Java interface; either use another interface, or create and load a metric file, as described below.

Some standard metrics can be loaded by first telling GRTensor where to find them with the command (on COSINe machines)

```
grOptionMetricPath:="/home/math/tevian/GRtensor/metrics":
```

after which the file you want can be loaded using `qload` . For example, the Schwarzschild black hole metric can be loaded with `qload(schw)` . On physics machines, you should replace `/home/math/tevian` with `/home/tevian` , and on PCs use `\\\\poole\\ClassFolders\\Math-Dray\\GRtensor\\metrics`

> You can put this command in a file called `grtensor.ini` in your home directory (or the same directory as your `maple.ini` file), but *not* in your `.mapleinit` file.
>
> You can also create metric files of your own by using the canned files as an analogy, then loading your file with `qload` or `grload` . This is not as hard as it sounds, as the file only needs to define the number of coordinates `Ndim_` , list the coordinates `x1_` , etc, and enter the metric components `g11_` , etc; note the trailing underscore in each variable name. For the sphere, the file would be
>
> ```
> Ndim_:=2:
> x1_:=theta:x2_:=phi:
> g11_:=r^2:g22_:=r^2*sin(theta)^2:
> ```
>
> Unspecified metric components are assumed to be zero, so you do not need to explicitly set off-diagonal elements like `g12_` to zero. To use an orthonormal basis, enter the metric components as `eta11_` , etc. (instead of `g11_`), and the basis itself as `bd11_` , etc. For the sphere, the file would be
>
> ```
> Ndim_:=2:
> x1_:=theta:x2_:=phi:
> eta11_:=1:eta22_:=1:
> bd11_:=r:bd22_:=r*sin(theta):
> ```
>
> See the documentation for `qload` or `grload` for help in determining how to load the file once created.

Many standard tensors are predefined in terms of the metric you have entered or loaded, including the inverse metric `invmetric` , the Christoffel symbols `Chr2` , the *covariant* (all indices down!) Riemann tensor `Riemann` , the Ricci tensor `Ricci` , the Ricci scalar `Ricciscalar` , and the Einstein tensor `Einstein` ; the metric itself is called `metric` .

You can calculate any of these tensors using `grcalc` , and see the result of the calculation using `grdisplay` . For instance, to find the Ricci scalar, type

```
grcalc(Ricciscalar);
grdisplay(Ricciscalar);
```

When working with an orthonormal basis, it is necessary to refer to e.g. the (covariant) Riemann tensor explicitly as `R(bdn,bdn,bdn,bdn)` ; "`Riemann`" is an alias for the coordinate components `R(dn,dn,dn,dn)` .

You may need to use the `gralter` and/or `grmap` commands to fully simplify your results.

Further documentation can be obtained by typing `?grtensor` .

b) Tensor

The package `tensor` is distributed with Maple; if you have Maple, you have `tensor`.

The information in this section is a summary of the online documentation which can be obtained by typing `?tensor` or `help(tensor);` .

Type the command `with(tensor):` to load the `tensor` package, then choose coordinates with a command like `coord:=[theta,phi]:` . Tensors are created using the somewhat messy `create` command, in which you must first explicitly specify which indices are "up" (+1) and "down" (−1), then give a matrix of components. For instance, the (covariant, that is, both indices "down") metric for the sphere can be entered as

```
g:=create([-1,-1],array(symmetric, [[r^2,0],[0,r^2*sin(theta)^2]]));
```

It is essential that the metric be defined as a *symmetric* array. The Ricci scalar `R` is now computed by first successively computing the inverse metric `ginv` , the partial derivatives of the metric `dg` and `ddg` , the Christoffel symbols `Gam` , the Riemann tensor `Rie` , and the Ricci tensor `Ric` , as follows; any names can be used for the intermediate steps.

```
ginv := invert(g,´detg´):
dg := d1metric(g,coord):  ddg := d2metric(dg,coord):
Gam := Christoffel1(dg):
Rie := Riemann(ginv,ddg,Gam):
Ric := Ricci(ginv,Rie):
R := Ricciscalar(ginv,Ric):
```

Finally, to display the result of this computation, use the command `get_compts(R);` .

WARNING: `tensor` defines the Ricci tensor and scalar to be −1 times our definitions.

3. PACKAGES BASED ON MATHEMATICA

Start *Mathematica* by typing `math` for a shell interface, or `mathematica &` for the notebook front end under XWindows. In a notebook, all commands must be entered by holding down the shift key while typing the return key. Note that all *Mathematica* commands are written with square brackets, not parenthesis. An optional semicolon at the end of a command suppresses output. Exit *Mathematica* with the command `Quit` .

a) GRTensorM

GRTensorM is a (much older) translation of GRTensor into *Mathematica*.

GRTensorM does not appear to work on *Mathematica* versions 6 or later, and will therefore not work on physics machines or in the MLC; the following instructions are for COSINe machines only.

GRTensorM is freeware; if you have access to a suitable version of *Mathematica* elsewhere and would like to obtain a copy of GRTensorM, let me know.

To enable *Mathematica* to find the GRTensorM files, type

```
$Path=AppendTo[$Path,"/home/math/tevian/GRtensor"]
```

(You can add the appropriate command to the file `init.m` in your home directory.) Then type `<<grii/grt.m` to start GRTensorM.

Due to the age of this code, you will get some error messages on startup, which I believe you can safely ignore.

Most commands are the same as in GRTensor, so long as you remember to change parentheses to square brackets. However, I do not know how to enter the metric by hand, so you will need to use the `makeg` command. If you are using the notebook interface, this will pop up several new windows with questions for you to answer, each of which requires you to first click on the empty box, then enter your answer, then click on OK.

WARNING: Unlike the newer GRTensor, in GRTensorM you must save your metric before using it. To do this, *before* running `makeg` , type `SetOptions[grii,MetricPath->"~"]` , which tells GRTensorM to put such files in your home directory; you may enter any other directory, such as `"/tmp"` instead of your home directory `"~"` . If you then run the command `grmake[test]` , after answering the questions a file named `test.g` will be created in the appropriate directory. Don't forget to delete unwanted files when done!

In order to use system metric files, you must (re)set `MetricPath` by typing the command `SetOptions[grii,MetricPath->"~tevian/GRtensor/grii/metrics/"]`; you could now load the Schwarzschild metric with the command `qload[schw]` .

You may need to replace `~tevian` with `/home/math/tevian` .

The `SetOptions` commands can *not* go in your `init.m` file!

To calculate a tensor, use `grcalc` and `grdisplay` as before; you may also need to use `grmap` to simplify your result. For instance, to calculate the Ricci scalar type `grcalc[Ricciscalar]` , and to see the result type `grdisplay[Ricciscalar]` ; you may need to try things like `grmap[Simplify,Ricciscalar]` (followed by another `grdisplay` command) in order to get the final answer.

Some additional help is available by typing `?grtensor` .