Elementary Maple Notes

Marcus Berg, June 2003, updated November 2003

1 General commands (needed for Exercises 1-5)

- `:=`, defines an expression
  Example: `expr:= sin(x)+ x^2;` (note semicolon) defines the expression $sin(x)+x^2$ and calls it `expr`. **NOTE**: the hat `^` means "to the power of".
- `subs`, variable substitution
  Example: `subs(x=sqrt(g),g^2)` substitutes $x=\sqrt{g}$ into the expression g^2, which yields x. **NOTE**: The square root $\sqrt{x}$ is written `sqrt(x)`.
- `f:= x -> f(x)`, defines a map (function)
  Example: `f:= x -> sin(x)` and typing `f(0)` returns `0`. **NOTE**: Doing `f(x):=sin(x)` doesn't work: then `f(0)` does not give 0. The alternative is `f:=sin(x)` and `eval(subs(x=0,f))`, where `eval` tells Maple to evaluate the expression `sin(0)`. The latter is OK if you don't need any values or specific properties but just want to mainpulate abstract expressions.
- `simplify`, simplifies an expression
  Example: `simplify(sin(x)^2+cos(x)^2)` yields 1.
- `diff`, differentiation
  Example: `diff(arctan(x),x)` performs $(\arctan(x))' = 1/(1+x)^2$.
- `integrate`, integration, symbolic or numeric. Example: `integrate(exp(-x^2),x=0..infinity)` performs $\int_0^\infty dx\, e^{-x^2} = \sqrt{\pi}/2$. **NOTE**: "$\infty$" is written "`infinity`".
- `solve`, solves an algebraic system (or single equation). Example: `sol:=solve(x^5=2*x,x);`. This solves $x^5 = 2x$, and returns a list. You can take out elements by `op(1,sol)`, `op(2,sol)` and so on. **NOTE**: The imaginary unit $i=\sqrt{-1}$ is written `I`.
- `plot`, plots in 2d. Example: `plot(sin(x)^2,x=-10..10);`. Click on the right mouse button and choose "Export As..." to save the plot.
- `countourplot`, plots contours.
  Example: `contourplot(sin(x*y),x=0..1,y=0..1,contours=[0.1,0.2,0.3]);` The values are the heights of the contours (try different ones!). **NOTE:** Needs package `plots`, load by typing `with(plots):`
- `plot3d`, draws 3d plots. `plot3d(x^2+y^2,x=-2..2,y=0..2,view=0..1);`
  NOTE: Use the mouse to rotate the plot!
- `for`, iteration (using a dummy variable, e.g. i)
  Example: `for i from 1 to 5 do:  print(i^2); end do:` gives the sequence `1 4 9 16 25`
- `dsolve`, solves an ordinary differential system (or single equation)
  Example: `systemsol:=dsolve({ode1,ic1,ode2,ic2},{f(x),g(x)}, numeric, range=0..1, output=listprocedure);`. Maple uses a range of methods to perform the numeric integration.
  Here e.g. `ode1:=diff(f(x),x)+g(x)=0;` and `ic1:=f(1)=1, D(f)(1)=1;` .
  Then to plot you would do `odeplot(systemsol,[x,f(x)],0..1);`
- `read`, reads a Maple program (text file)
  Example: `read "volume.mpl"`, reads the file `volume.mpl`, a text file containing Maple commands. It doesn't have to have the extension `.mpl`, but it's a good idea to call your Maple programs `exercise1.mpl`, etc. to remember which of your files are Maple programs.
- `#`, adds a comment so that later on you remember what a certain calculation does.
  Example: If you type `# The steps below perform quantization of M-theory`, nothing happens.

2 Plots in LaTeX (needed for Exercise 3)

If you want to include your plot file (an EPS file, i.e. ending in ".eps") in a LaTeX document, insert the command `\usepackage{graphics}` before your `\begin{document}`, then in the text, insert

```
\begin{figure}[h]

\insertgraphics{niceplot.eps}

\caption{This is my figure!}

\end{figure}
```

The `h` means "here", i.e. exactly where it is in the LaTeX file. LaTeX doesn't always succeed in placing it "here", e.g. if you are at the bottom of a page, then it will decide on its own.

3 special `GRTensor` commands (needed for Exercises 6-8)

The `GRTensor` home page is `http://grtensor.phy.queensu.ca`. The installation instructions tell you to make an `.mapleinit` (for Linux) or `maple.ini` (for Windows) to tell Maple where to find `GRTensor`.

- `makeg`, defines metric
  You enter the coordinates with square brackets: `[x]`.
  After entering it, Maple asks you if you want to save it.
  Example: `d[x]^2 + (d[y] + d[z])^2` has a cross term $2dydz$.
- `qload`, loads previously saved metric
  Example: `qload(schwarzschild)`
- `grcalc`, compute tensor object that `GRTensor` knows
  Example: `grcalc(R(dn,dn))` computes $R_{\mu\nu}$. **NOTE**: "dn" means "down".
  Known tensors: `Chr(up,dn,dn)` is $\Gamma^{\rho}{}_{\mu\nu}$, `R(dn,dn)` is $R_{\mu\nu}$, `Ricciscalar` is R etc.
- `gralter`, apply some rule to a tensor
  Example: `gralter(R(dn,dn),simplify)`.
- `LieD`, take the Lie derivative of a tensor along a vector
  Example: `LieD(v,g(dn,dn))` gives $\mathcal{L}_v g_{\mu\nu}$.
- `grdef`, define your own tensor
  Example: `grdef(T{(a b)}:=F{a c d}*F{b ^c ^d})` defines a stress-energy type tensor $T_{\mu\nu} = F_{\mu\rho\sigma}F_{\nu}{}^{\rho\sigma}$, provided the tensor $F_{\mu\nu\rho}$ was defined previously. **NOTE:** A general 5-tensor in 10 dimensions has 10^5 elements; this makes most computations take too long. Indicating symmetry on the *left-hand* side in the definition, as in this this example (antisymmetry: square brackets `[a b]`, symmetry: round brackets `(a b)`) makes `GRTensor` only calculate the relevant components, e.g. it will compute F_{xyz} and use $F_{xzy} = -F_{xyz}$ rather than computing F_{xzy} as well.

See the `GRTensor` manual and the in-Maple help (if you have installed it) for many more examples.

4 Miscellaneous

- In Mathematica, maps are defined with an underscore "_" instead of the Maple `->`, for example: `f[x_]:=Sin[x]`. Also note the square brackets and the capital "S" in `Sin`.
- An interesting Mathematica package is Ulf Gran's `GAMMA` for gamma matrix algebra in arbitrary dimensions (see hep-th/0105086). It is rule-based and can live without representations.
- Another interesting Mathematica package is `FeynArts` that can do one-loop and some two-loop Feynman diagram computations. Its home page is `www.feynarts.de`, where there are many examples.

Eight Computer Exercises

Marcus Berg, June 2003

The idea is: *when you know how to do it, each of these exercises takes less than ten minutes to do, and the result is correct.* Some of them would take hours to do by hand even if you know how to proceed, and would be very prone to small mistakes.

Note: I have typed solutions for each exercise. Each one is 3-5 lines of Maple commands.

Exercise 1: **Variable transformations and integrals**
Compute the Feynman-parameter integral

$$\int_0^1 dx \int_0^1 dy \int_0^1 dz\ \delta(x+y+z-1)\ \frac{xyz}{(xy+xz+yz)^3}$$

This occurs in the p-dependent two-loop 2-pt diagram in ϕ^4 theory. It may be useful to first perform a variable transformation in the integrand (not by hand!). Hint: try $x = \alpha\beta$, then figure out the other ones.

Exercise 2: **Simple but tedious numerical checks**
Consider a Calabi-Yau manifold. For integer Kähler moduli, the volume of the manifold is a function of integers m_i for $i = 1..3$. Let's say a toy version of the volume is

$$\mathcal{V} = \frac{1}{5}m_1m_2m_3^2 + 3m_1m_2^2m_3 + \frac{1}{4}m_1^2m_2m_3 + \frac{407}{126}m_1m_3^3 + \frac{1}{9}m_1m_2^3 + \frac{1}{3}m_1^3m_2 + \frac{1}{20}m_3^4 + m_2^4 + \frac{1}{2}m_1^4\,.$$

Square-integer values (i.e. 1, 4, 16, ...) can mean preserved supersymmetry. Check whether $\mathcal{V}$ has any square integer values for $0 < m_i < 7$ (harder: up to $m_i < 20$). Note that it is much easier if you set one of the $m_i = 0$, but technically this is not allowed (none of the cycles should shrink to zero), so make sure to try for $m_i > 0$.

To save you typing, I predefined a variable `volume`, get a copy from me and type `read "volume.mpl"`.

Exercise 3: **Visualization**
The potential

$$V(x,y) = \frac{1}{2}x^2 + \frac{1}{2}y^2 + \frac{1-m}{\sqrt{(x+m)^2+y^2}} + \frac{m}{\sqrt{(x+m-1)^2+y^2}}$$

describes a test particle moving in the presence of two big masses m and $1-m$. What does this function look like? (You could either do a couple of 2d plots or a 3d plot, whatever you like better.) How much energy, i.e. height, does the test particle need to pass between the two masses (just read off in the figure)? Print one plot. Include it in one of your own LaTeX files.

Exercise 4: **ODEs**
Consider this ordinary differential equation (ODE) system with three functions $R(q), W(q), A(q)$ and a constant parameter p:

$$\begin{aligned}
\frac{1}{4}e^{2A}W'(R''W' + R'W'' + R'W'A') - p^2R &= 0;\\
W'(4W - \frac{1}{2}W'') - V' &= 0;\\
A' + 4\frac{W'}{W} &= 0;
\end{aligned}$$

where the potential $V(q)$ is a given function,

$$V = \frac{1}{8}(1+\cosh^4 q)^2 - \frac{1}{4}(\cosh^6 q + \sinh^2 q) \; .$$

This system describes fluctuations of a scalar $R(r)$ in the background of a domain wall metric $ds^2 = e^{2A(r)}dx^i dx^i + dr^2$ and domain wall scalar $q(r)$.[1] Choose boundary conditions at $q = 0.2$ *such that* R *vanishes at infinity*, and integrate from some cutoff $\epsilon > 0$, for example $\epsilon = 0.01$. Plot the fluctuation $R(q)$. This is the object that in AdS/CFT would give the correlator (which would be basically R'/R).

To start with you can put the boundary momentum p to 1, but make the plots for a few different boundary momenta p, e.g. $p = 0.1, 1, 10$.

To save you some typing, I predefined the system; get it from me and just type `read ‘‘ode.mpl’’`.

Exercise 5: **Special functions**
Here's a type of expression you might find in string amplitudes:

$$A = K_1 B(a,b) + K_2 B(a+1,b+2) + K_3 B(a+3,b+2);$$

where K_i, $i = 1,2,3$ are kinematic factors, and $B(a,b)$ is the beta function $B(a,b) = \Gamma(a)\Gamma(b)/\Gamma(a+b)$. Reduce this to a single beta function with an overall factor that is a polynomial in the K_i.

(Hint: Maple has a range of built-in special functions and knows many identities for them, but it isn't always clever in applying them. Although $B(a,b)$ exists in Maple as `Beta(a,b)`, using `simplify` directly doesn't work. Why?)

Exercise 6: **Curvature computations** (requires a tensor package)
Show that the Kerr metric is a vacuum solution of Einstein's equations. Make sure you understand what conventions are used and how they relate to the ones you are used to. Check that the vector

$$v = \frac{\partial}{\partial t} + \Omega_H \frac{\partial}{\partial \phi}$$

(Wald (12.3.20)), which is tangent to null geodesics generating the horizon, is a Killing vector. Here $\Omega_H = a/(r_+^2 + a^2)$ is the limit angular velocity, and $r_+ = m + \sqrt{m^2 - a^2}$.

(Hint: in `GRTensor`, the Kerr metric is pre-defined, just type `qload(kerr)`.)

Exercise 7: **Curvature computations II** (requires a tensor package)
Compute the Ricci scalar $R(y)$ of a domain wall in 5 dimensions

$$ds^2 = e^{2A(y)}dx^i dx^i + e^{2B(y)}dy^2 \qquad i = 1,..4$$

in terms of the functions $A(y)$ and $B(y)$.

Does the Ricci scalar diverge at some point? (Answer in terms of properties of $A(y)$ and $B(y)$.)

Exercise 8: **Tensor manipulations** (requires a tensor package)

Let's define a toy version of a tensor structure that occurs in string amplitudes, using polarization 4-vectors e_1^μ and e_2^μ and momentum 4-vectors k_1^μ and k_2^μ :

$$t_{\mu\nu\rho\sigma} = (k_{1\mu}e_{1\nu} - k_{1\nu}e_{1\mu})(k_{2\rho}e_{2\sigma} - k_{2\sigma}e_{2\rho}) + (k_{2\mu}e_{2\nu} - k_{2\nu}e_{2\mu})(k_{1\rho}e_{1\sigma} - k_{1\sigma}e_{1\rho})$$

Compute $t_{\mu\nu}{}^{\mu\nu}$ (using a flat Euclidean metric), $t_{(\mu\nu)\rho\sigma}$, and $t_{\mu\nu\rho\sigma}t^{\mu\nu\rho\sigma}$. Here antisymmetrization is defined as $A_{(\mu\nu)} := (A_{\mu\nu} - A_{\nu\mu})/2$. You can use $k_i \cdot e_i = 0$ if this helps.

[1] for simplicity, I have set the effective potential to zero, it would have appeared in the R'' equation. Also I have set $\alpha = 1$. It is convenient to use the background scalar $q(r)$ itself as dependent variable, so the above equation are, as stated, functions of q.

Solutions to Computer Exercises

Marcus Berg, June 2003

Exercise 1:
One has to play around a little bit to find a good transformation, such as $x = \alpha\beta$, $y = (1-\alpha)\beta$, $z = 1-\beta$.

```
f:=x*y*z/(x*y + x*z + y*z)^3
Jac:=alpha*beta+(1-alpha)*beta;
g:=simplify(Jac*subs(x=alpha*beta,y=(1-alpha)*beta,z=1-beta,f));
answer:=integrate(g,beta=0..1);
```

The transformation takes care of the delta function, and the result is 1/2.

Exercise 2:
First copy the file `volume.mpl` to your directory. This contains the definition

```
V:=1/5*m1*m2*m3^2+3*m1*m2^2*m3+1/4*m1^2*m2*m3 +407/126*m1*m3^3
+1/9*m1*m2^3+1/3*m1^3*m2+1/20*m3^4+m2^4+1/2*m1^4;
```

Then do

```
read ''volume.mpl''; maxint:=10;
for m1 from 1 to maxint do: for m2 from 1 to maxint do: for m3 from 1 to maxint do:
if type(sqrt(volume),integer) then print(m1,m2,m3,volume); end if;
od; od; od;
```

You wouldn't really need to do the "if" statement, if you did the easy version, up to $m_i < 7$, you could just look through the 343 possibilities. The answer is (not counting many answers for setting some m_i to zero): $(m_1, m_2, m_3) = (7, 3, 6)$. This gives $\mathcal{V} = 8100 = 90^2$. Note: you can also use the shorthand "`od`" (`do` backwards) for `end do`, and similarly for `if`.

Exercise 3:
After a little experimenting, I thought this view was pretty good:

```
V:=x^2/2+y^2/2+(1-m)/sqrt((x+m)^2+y^2)+m/sqrt((x+m-1)^2+y^2);
with(plots):
Vn:=subs(m=0.7,V):
plot3d(Vn,x=-2..2,y=-2..2,view=1..3);
contourplot(Vn,x=-2..2,y=-2..2,contours=[1.8,1.9,2.0,2.1]);
```

Exercise 4:
This also requires a little playing to get the right initial conditions. I set everything to be one except the derivative of R at $q = 0.2$, which I modified to get zero at "infinity". To me, "infinity" meant $q = 10$ or so. In practice, you can check how much the quantity you are interested in (here R'/R) changes if you make the solution be zero further out, but if there are no small or big parameters in the problem, usually roughly $\infty \approx 10$. Again, one can check the accuracy by trial and error, going further out and reading off e.g. `systemsol[3](10)`, this gives the value of the third function (in my case $R(q)$) at $q = 10$.

```
read ''ode.mpl'';  intrange:=0.01..10; p:=1;
icR:=R(0.2)=1,D(R)(0.2)=-1.717066995036805; icA:=A(0.2)=1; icW:=W(0.2)=1;
systemsol:=dsolve({odeR,icR,odeA,icA,odeW,icW},{R(q),A(q),W(q)},
    numeric,range=intrange,output=listprocedure);
plots[odeplot](systemsol,[q,R(q)],intrange);
```

This gives $R(q = 10) \approx 10^{-14}$. Fixing it too much makes R'/R do strange things, though.

Exercise 5:
This is quick, one just has to realize that an expression is not a function:

```
B:= (a,b) -> GAMMA(a)*GAMMA(b)/GAMMA(a+b);
A:=K1*B(a,b)+K2*B(a+3,b+2)+K3*B(a+1,b+2);
simplify(A);
```

Actually the end factor is not quite a Beta function, but can easily be reduced to $B(a, b)$ by $\Gamma(z+1) = z\Gamma(z)$, e.g. by `simplify(GAMMA(a+b+5)/GAMMA(a+b))`.

Exercise 6:
Make sure you have the package `GRTensor` loaded. Wald's vector (12.3.20) is special because it is tangent to null geodesics generating the horizon, but for the purpose of checking whether it's a Killing vector, Ω_H is just some constant so we don't even have to put it in explicitly.

```
qload(kerr);  grcalc(R(dn,dn));
gralter(R(dn,dn),simplify);  grdisplay(R(dn,dn));
grdef(‘v{^a}:=[0,0,OmegaH,1]‘);
grcalc(LieD[v,g(dn,dn)]); grdisplay(LieD[v,g(dn,dn)]);
```

A quick alternative is `grcalc(KillingTest[v])` ! If you have the help files installed, you can learn more about this command by typing ?`killing`.

Exercise 7:
Type `makeg(warped)`, then enter

```
exp(2*A(y))*(d[x1]^2+d[x2]^2+d[x3]^2+d[x4]^2)+exp(2*B(y))*d[y]^2;
```

Finish the `makeg` (no, this doesn't count as extra lines!) and then type

```
grcalc(Ricciscalar); Rscalar:=grcomponent(Ricciscalar);
```

We see that it blows up if $B(y) \to -\infty$. We would want to solve for A and B and Φ, but this would typically require some ansatz, such as in `hep-th/0004165`. If you are interested: try their ansatz (they have two), with general exponents, and solve for those exponents.

Exercise 8:
Define a Euclidean metric `eucl`, then type `qload(eucl)` as usual.

```
grdef(‘t{a b c d}:=(k1{a}*f1{b}-k1{b}*f1{a})*(k2{c}*f2{d}-k2{d}*f2{c})
+(k3{a}*f3{b}-k3{b}*f3{a})*(k4{c}*f4{d}-k4{d}*f4{c})‘);
```

Note that you can't use `e1`, etc as names since those already exist. To calculate this tensor using `grcalc`, we have to define those 4-vectors. First one can try just putting in general components, that works fairly well for $t_{\mu\nu}{}^{\mu\nu}$ and $t_{(\mu\nu)\rho\sigma}$, but not so well for $t_{\mu\nu\rho\sigma}t^{\mu\nu\rho\sigma}$, which then becomes too long (over 50 000 words). One reasonable compromise, which is not completely general, is to pick k_1, k_2 and k_3 to be orthogonal, and also to impose transversality $k_1 \cdot e_1$ by putting some components to zero:

```
grdef(‘k1{a}:=[k1t,k1x,0,0]‘); grdef(‘k2{a}:=[k2t,0,k2y,0]‘);
grdef(‘k3{a}:=[k3t,0,0,k3z]‘); grdef(‘k4{a}:=[k4t,k4x,k4y,k4z]‘);
grdef(‘f1{a}:=[0,0,f1y,f1z]‘); grdef(‘f2{a}:=[0,f2x,0,f2z]‘);
grdef(‘f3{a}:=[0,f3x,f3y,0]‘); grdef(‘f4{a}:=[f4t,f4x,f4y,f4z]‘);
```

Note that the Maple code above can be written in two lines! Hence I still have 3 lines left to make it 5:

```
grdef(‘tcontr:=t{a b ^a ^b}‘);
grdef(‘tanti{[a b] c d}:=t{[a b] c d}‘);
grdef(‘tsquared:=t{a b c d}*t{^a ^b ^c ^d}‘);
```

Then to check them, you would type e.g. `grcalc(tcontr); grdisplay(tcontr);`. In this exercise, it is not clear exactly how useful these fairly long expressions will be, but one can at least check coefficients if one also does it by hand. For this kind of problem, `Cadabra` by Kasper Peeters would be a better option.