Compiling the model code (and related issues)

1) To compile the model, you must execute a script that will produce and execute a fortran makefile. All of the technical details are hidden under the hood with this procedure, but you can take a look at the compile script to see what's actually going on.

First go to the directory where the compile script is located. For the spectral model, it's:

```
> cd ~/student_models/code/atm_dycores/exp/spectral/
```

To compile the bgrid or fv cores, replace "`spectral`" in the line above with "`bgrid`" or "`fv`", respectively. For GRAM, go to code/spectral_gray/exp/, and for MiMA, go to code/mima/exp/.

Next, execute the compile script:

```
> ./nyu_compilescript
```

(It may also just be called compile_script, for the more advanced models.) The script may need some time to execute, depending how much the source code has been changed. (It will check to see what has been changed since it was last compile, and compile only the parts that have changed, and parts of the code that depend on these.) Warning statements are okay. Errors are not.

With hope that worked. That was deceptively easy ... take a look at the compile script to see what was actually going on, and you'll see that a lot had to be set up to make this work.

2) Why might you need to recompile the code? To change key parameters of the Earth system, such as the radius of the Earth or the rotation rate, no namelist parameters are available. As many parts of the full model depend

on these parameters (i.e. the ocean, sea ice, land, etc..), they are contained in "shared" place in the source code structure. To change these, you need to edit this fortran file.

```
~/student_models/code/atm_dycores/src/shared/constants/constants.f90
```

I think it's pretty straighforward to see what you must change here, but ask me directly if it's not clear. These changes won't go into effect until you compile the code again. A similar strategy should work for MiMA and GRAM, but I haven't tried it yet!

3) Note that when you change a parameter in the fortran, you still need to create a new run_parameters directory, with all the necessary input files (field_table, diag_table, and namelists). You can just copy over another run that has the right parameters. For example, say you are changing the radius of the Earth r0. You first run a control simulation (say: `t42l20e_hs_HC_theory_default`), with the standard value of r0. Now you have doubled the radius of the Earth by modifying the fortran source and compiled the code again. Now set up a place holder for your new run:

```
> cd ~/student_models/run_parameters/
> cp -r t42l20e_hs_HC_theory_default t42l20e_hs_HC_theory_r0x2
```

This will create a new directory, `t42l20e_hs_HC_theory_r0x2`, which has all the same input files. You need a new directory, or the code will try to overwrite your previous work, and balk when it sees that it is there. Unfortunately you can't tell from the namelist that the radius of the Earth has changed, so you need a good naming system.

When you modify the fortran code, complete all the simulations with the new setting before changing the parameters and compiling the code. The system will grab what ever executable is available at the time your run begins.

The same goes for modifying the GRAM or MiMA. Why not create a 4xCO2 run of MiMA. Copy over the starting script, say:

```
> cp -r mima_t42l40cig_flat mima_t42l40cig_flat_4xCO2
```

and then go into the directory mima_t42l40cig_flat_4xCO2/ and modify the CO2 concentration in the namelists, which is in this section:

```
 &rrtm_radiation_nml
     h2o_lower_limit        = 2.e-07,
     co2ppmv                = 300.,
     do_read_ozone          = .true.,
     ozone_file             = 'ozone_1990',
     dt_rad_avg             =  4500,
     dt_rad                 =  4500,
     lonstep                = 4 /
```

Change it so that `co2ppmv = 1200.,` and voila … you've just dumped a solid portion of our coal reserves into the (model) satmosphere.