Computational Modelling: g++ Compiler Options

September 28, 2011

Compiling some c++ code file is pretty straight forward if you are not doing anything particularly complicated. However the compiler is 'stupid', it needs to be given exact instructions when your program becomes more complicated so that it can build your program correctly. This involves for example debugging your program, including external libraries or making the compiler optimise your program to make it run faster.

There are far too many compiler options for g++ to go through them all, so here is a list of common ones you should know about:

- **-o** - This specifies the output/executable file that your code is compiled into
- **-w** - Ignore all warning messages - Ideally you should not ignore warnings but fix them, but if you have 100's of warnings that are making it difficult to see the errors in the console, this can be useful
- **-Werror** - Turn all warnings into errors. Warnings suggest problems that should be fixed/changed but don't stop the compilation, errors however stop the compilation of your code
- **-Wall** - Includes extra warnings. This helps catch extra problems that might effect your program.
- **-O1,-O2,-O3** - Optimise code for faster performance. This involves a trade-off between computation speed and memory usage. See http://gcc.gnu.org/onlinedocs/gcc/Optimize-Options.html for more details
- **-l** - Includes an external library. If you have used calls to a library e.g. NAG or GSL, then you need to tell the compiler where the library files are so it can include them in your program
- **-I** - Adds a directory where you store header files. When you use external libraries you will need to also tell the compiler where the libraries header files are.
- **-g** - Produce debugging symbols. Debugging is the process of running your program but stopping it at certain lines in your code, you are then able to step through the code one line at a time to see in realtime what it is doing - Hopefully helping to solve a bug or problem in your code! This is an indepth topic, if you are interested check http://cs.baylor.edu/~donahoo/tools/gdb/tutorial.html

Here's some examples of to use command line options.

- `g++ -w -Wall -a out.a 1.cc 2.cc 3.cc` - Compiles `out.a` from 3 different source code files

- `g++ -w -O2 -o myprog.a myprog.cc` - Compiles the file `myprog.cc` ignoring warnings and doing a level 2 optimisation of the code
- `g++ -I/incudes/ -l/libs/math_library.lib -o myprog2.a myprog.cc` - Compiles the file `myprog.cc` but this time we are using `math_library.lib` in our code so we need to link to the library but we also need to include the library header files in `/includes`

If you want to know more have a look at http://gcc.gnu.org/onlinedocs/gcc-4.5.2/gcc/Option-Summary.html, it has just about all the compiler options available.