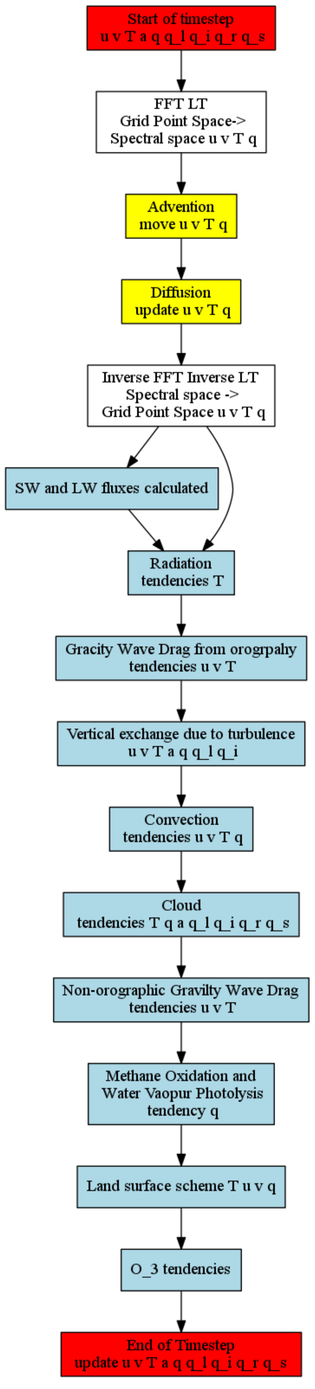
An Investigation of the Timestep of three Atmospheric Models

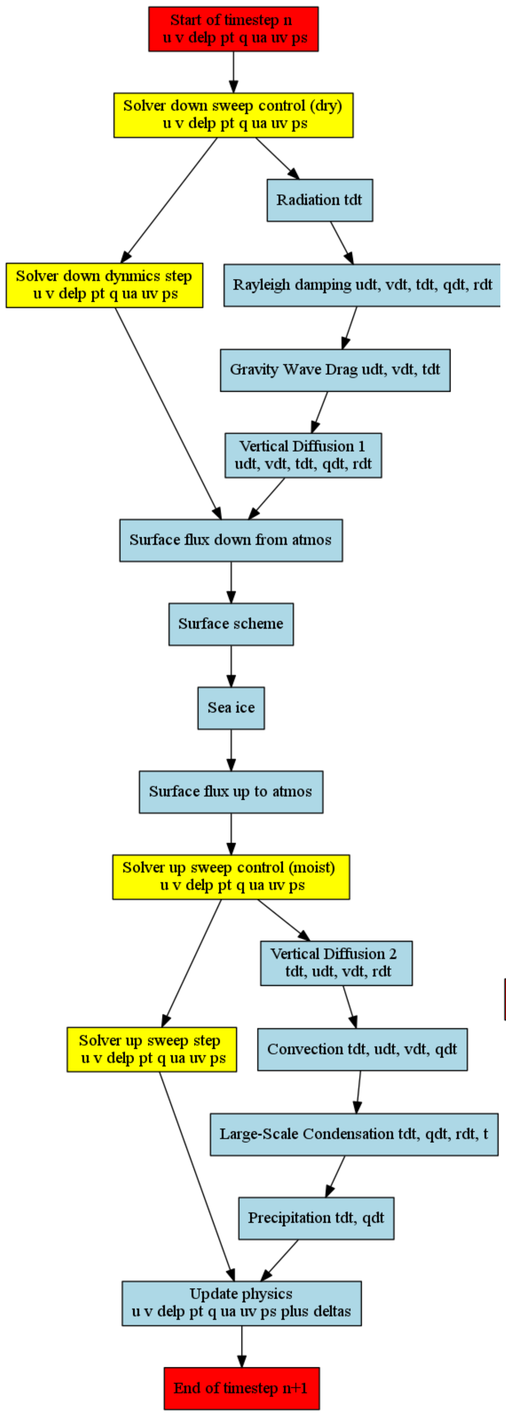
Simon Wilson
NCAS-CMS, Dept. Meteorology, University of Reading

Computational Modelling Services
NATIONAL CENTRE FOR ATMOSPHERIC SCIENCE
NATURAL ENVIRONMENT RESEARCH COUNCIL

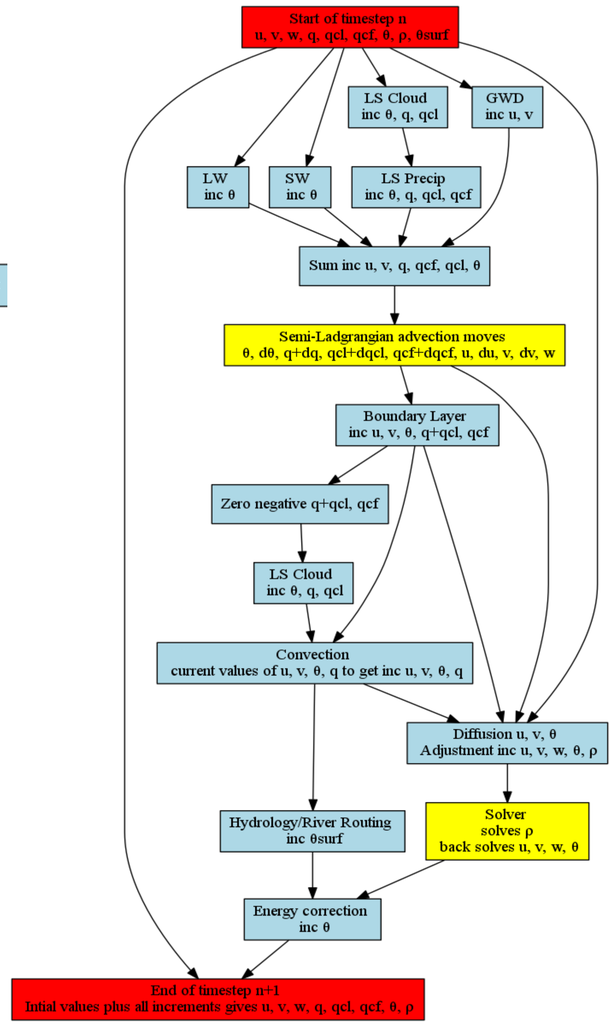
OpenIFS



HiRAM



HadGEM3A



Blue: Physics
Yellow: Dynamics
Red: Start/End

Code Statistics

Model	Number of files	Number of code statements	Number of subroutines	Number of modules
HadGEM3A	1553	217972	2217	628
HiRAM	297	372394	4505	287
OpenIFS	2776	160286	3675	624

Stats generated by git@bitbucket.org:rupertford/codeanalysis.git

Introduction:
As part of IS-ENES2 WP3 on developing convergent model codes, the comprehensive documentation and analysis of several models is required so that modelling groups can be better informed of what each other are doing at a detailed level. This then can be used to develop strategies for convergent model code. For this end, a comparison of the atmosphere timestep and code structure of three different atmosphere models was undertaken. The models studied are the Unified Model from the Met Office in a HadGAM3 configuration; HiRAM from GFDL and OpenIFS from the ECMWF.

Analysis:
As a first step it was decided to develop flowcharts (opposite) in attempt to show the calling sequence and inter-dependencies of the various submodels together with the dependent prognostic variables. These were derived from a combination of human code analysis, reference to the model formulation and technical documentation papers and speaking to model code gurus. Statistics of the code size and structure were also generated.

Subjective comments on code and documentation:
OpenIFS: Reliance on legacy code and structure. Poor code formatting. Best technical documentation of codes, flowchart mainly generated from this.
HiRAM: Best written and formatted code. Sensible use of variable names. Flowchart mainly generated from direct inspection of code and inline documentation.
UM: Reliance on legacy code and structure. Very difficult to read code. Little useful technical documentation. Flow chart mainly generated from personal detailed long term knowledge of UM and a previous flowchart version.

Discussion and future work:
There is clearly a large discrepancy between how the codes are written and documented. While rewriting the code is highly non-trivial, some standardisation in technical documentation is an achievable goal in the short term. A standard system of inlined code documentation can particularly useful, which identifies each submodel component, together with its dependencies and updated prognostics. These can then processed by a tool to generate documentation and flowcharts from the annotated code (similar to Doxygen and Sphinx).