Geosci. Model Dev. Discuss., 7, C2617–C2619, 2014
www.geosci-model-dev-discuss.net/7/C2617/2014/

Geoscientific Model Development Discussions

Interactive comment on "Firedrake-Fluids v0.1: numerical modelling of shallow water flows using a performance-portable automated solution framework" *by* C. T. Jacobs and M. D. Piggott

A. Dedner (Referee)

A.S.Dedner@warwick.ac.uk

Received and published: 8 December 2014

The authors present a new framework for solving fluid flow problems using UFL and autogenerated code based on Firedrake. They report on the implementation and tests for a viscous shallow water model. After a general introduction the mathematical model is briefly described both in its strong and weak form. Next the author give a short overview of the numerical scheme used and report on validation test cases using some benchmark problems of different complexity.

The paper is well written and easy to read and follow. Overall I have not found any substantial mistakes. But I do have a few concerns.

In summary:

I think that the introduction should be written a bit more balanced and taking into account additional options for users looking for a fluid dynamics solver. The numerical results should be extended to actually demonstrate the claimed flexibility with regards to different computer architectures and the resulting efficiency of the code. The results presented do not go into any detail concerning the machines used (are any computation done on a GPU or in parallel?). Also important information like parallel scaleup or other efficiency measures are not reported on at all. Finally the for me most interesting aspect concerning the flexibility in extending or customizing the code is not addressed. This could include the options for preconditioners used, time integration methods, or the inclusion of the LES model in the code. Both the flexibility with respect to the platform and the flexibility in customizing the model are important points when reporting on software using autogenerated code.

In detail:

In the introduction the authors make a number of unsubstantiated claims concerning the advantage of autogenerated code. Also when comparing their code with other packages they only mention Fenics (which uses a similar approach for code generation) and Fluidity, a package developed in parts at least by the authors themselves. Other packages and options for writing such a code within a general software framework are not touched at all. As major advantages of the autogeneration it is claimed that (a) there is much less coding to do for the user (b) the code can be efficiently used on many different platforms. After making these claims the authors only show results for one single model using two different spatial descritizations. It is unclear on which platforms the results were obtained and if in fact different platforms were or could be easily used - which would be a step towards substantiating claim (b).

Concerning claim (a): in many places in the paper the number of lines of code are compared: a few hundred for the new model compared to many thousands for "hand

written code". In many packages a simple shallow water model as considered here could be easily implemented with a few hundred lines of code (or even less). Packages like DealII, Dune, FreeFem etc. use interfaces to allow users to easily implement their own model problem without much coding required. The number of code lines is from my point of view not the selling point of a UFL bases code generator but the use of the mathematical notation (this is mentioned) and the flexibility to use different and future platforms (mentioned but as already stated not demonstrated). On the other hand, some of the problems with the UFL approach do become apparent in this paper but are not followed up on. It is stated for example that using general diffusion tensors with a discontinuous ansatz space is not yet possible. So if a user wanted to use a better diffusion model, they would be forced to implement quite a few lines of extra code (100 or 1000 perhaps). The authors do not go into detail on why this restriction has to be imposed and if there is some more fundamental problem with the code generation used. An extension is mentioned (an LES model) but how this is implemented within the framework is not covered although this would be very interesting to see.

Some small remarks:

- the observed convergence rates are not clear to me since the temporal discretization is only first order. Nevertheless second order convergence is observed. How is the time step and spatial length scale coupled?

- the ordering in terms between equation (4) and (8) should be made consistent.

- Give some specification of the boundary conditions available and which ones are used in the numerical experiments.

Interactive comment on Geosci. Model Dev. Discuss., 7, 5699, 2014.