Geophysical Research Abstracts
Vol. 14, EGU2012-680, 2012
EGU General Assembly 2012

On the tradeoffs of programming language choice for numerical modelling in geoscience. A case study comparing modern Fortran, C++/Blitz++ and Python/NumPy.

D. Jarecka (1), S. Arabas (1), M. Fijalkowski (), and A. Gaynor (2)

(1) University of Warsaw, Faculty of Physics, Institute of Geophysics, Warsaw, Poland, (2) Rensselaer Polytechnic Institute, Troy, NY, USA

The language of choice for numerical modelling in geoscience has long been Fortran. A choice of a particular language and coding paradigm comes with different set of tradeoffs such as that between performance, ease of use (and ease of abuse), code clarity, maintainability and reusability, availability of open source compilers, debugging tools, adequate external libraries and parallelisation mechanisms. The availability of trained personnel and the scale and activeness of the developer community is of importance as well. We present a short comparison study aimed at identification and quantification of these tradeoffs for a particular example of an object oriented implementation of a parallel 2D-advection-equation solver in Python/NumPy, C++/Blitz++ and modern Fortran.

The main angles of comparison will be complexity of implementation, performance of various compilers or interpreters and characterisation of the "added value" gained by a particular choice of the language. The choice of the numerical problem is dictated by the aim to make the comparison useful and meaningful to geoscientists.

Python is chosen as a language that traditionally is associated with ease of use, elegant syntax but limited performance. C++ is chosen for its traditional association with high performance but even higher complexity and syntax obscurity. Fortran is included in the comparison for its widespread use in geoscience often attributed to its performance. We confront the validity of these traditional views. We point out how the usability of a particular language in geoscience depends on the characteristics of the language itself and the availability of pre-existing software libraries (e.g. NumPy, SciPy, PyNGL, PyNIO, MPI4Py for Python and Blitz++, Boost.Units, Boost.MPI for C++). Having in mind the limited complexity of the considered numerical problem, we present a tentative comparison of performance of the three implementations with different open source compilers including CPython and PyPy, Clang++ and GNU g++, and GNU gfortran.