

Earth System Modeling Framework 2003

Python for Assembling Climate Modeling Toolkits

Robert Jacob – ANL and U. of Chicago Computation Institute

Raymond T. Pierrehumbert – University of Chicago, Geophysical Sciences

The Climate System Center: An overview

- NSF Information Technology Research Program
- Goal: Develop the FOAM Atmosphere-Ocean GCM as an alternative to intermediate complexity models. Permit millennial runs on small scale Beowulf clusters without problematic parameterization of large scale dynamics.
- Goal: Gradually move toward development of a modular climate-modeling toolkit, with building blocks assembled into a model using Python, a high-level interpreted language. The user experience will be analogous to the way one uses MATLAB to solve linear algebra problems.

Advantages of Python

- Python is an interpreted object-oriented high-level language with powerful syntax. Yet it is easy to learn. Open source, i.e. no-cost.
- Offers convenient access to full set of operating-system services, e.g. path and directory management.
- Interpreted languages simplify debugging and development
- There is already a rich set of Python modules for data management and graphics (e.g. cdms,vcs, CDAT).

How will we achieve usable model throughput with an interpreted language

- Functions and procedures written in c, c++ or Fortran can be compiled and made accessible as new Python commands using SWIG or PyFort. This does not require any re-coding of the original routines.
- Each Python model-component would be a single call to a compiled procedure that executed a large amount of processing (e.g. a radiation computation for the whole grid). Only "lightweight" loops allowed at the Python level.
- Python can also execute any executable that can be invoked from the Unix command line, and can start up and manage threads.

How we are using Python: Some examples

- Scripting an asynchronously coupled atmosphere/ocean simulation.
- Shallow-water model of the Martian seasonal cycle.
- A barotropic ocean model for exploring numerical analysis issues.
- Python drivers for column physics (e.g. radiation)
- Post-processing of netCDF model output (e.g. computing multi-year composite seasonal cycles)

Some thoughts on our relation to ESMF

- Generating new ideas for climate model design; testing out on a small well-knit friendly user community interested in real climate problems.
- Exploring ideas developed by ESMF community and providing feedback on further design.
- Developing prototype Python implementations of ESMF concepts

- Example: Our Python-based radiative-convective model toolkit provides good testbed for exploring design of the interface between a GCM and its radiation model component.