Nonlinear Shallow Water Testbed Model

Chris Eldred, Colorado State University

A) Project Overview

- Subset of the Global Circulation Model development being done at the Center for Multiscale Modeling of Atmospheric Processes (CMMAP)
- Global cloud resolving model based on a system of equations that filters vertically propagating sound waves. The vorticity and divergence are predicted, and the model is implemented on a geodesic grid.
- CSU Team Members: David Randall, Celal Konor, Ross Heikes, Don Dazlich, many others
- We have had SciDAC support since 2000. Ario Arakawa was involved until the current grant.
- My funding comes from the DOE Computational Science Fellowship
- Going to restrict myself to what I am working on

B) Science Lesson

- Nonlinear Shallow Water Testbed Model (NTM)
  - Has two components: Shallow Water Model and Eigensolver
  - Shallow Water Model
    - Solves the nonlinear shallow water equations on an f-plane (soon to be beta-plane and sphere as well)
    - Variety of different schemes implemented (all finite-volume/finite-difference)
    - Variety of different variable staggerings implemented (A,C,Z,Z*)
    - Variety of different grids implemented (planar square, hexagon, triangle) under a generic Voronoi mesh framework
  - Eigensolver
    - Determines the eigenstructure of the linear shallow water equations for a given grid and variable staggering
    - Important for understanding the effects of discretization on waves in the atmosphere

C) Parallel Programming Model + Computational Methods

- Written in Fortran 90 with Cheetah as a templating language (code-autogeneration, controlled with cfg files)
- Fortran namelists used to control run-time behavior
- Uses PETSC and SLEPC for:
  - Parallel vector management (decomposition, indexing, etc.)
  - Linear solver (Poisson problem, associed with vorticity-divergence formulation)
  - Sparse eigenvalue solver (SLEPC)
- Python is used as an analysis and visualization system
  - Uses Numpy, Scipy and Matplotlib
  - Scripts written to parse output (text right now, soon to be CF-compliant NetCDF) and generate plots and reports
- Would like scalability out to ~1000 cores (past that not useful)
- Currently only tested on my workstation

E) I/O Patterns and Strategy

- Right now write 1 file per MPI process
  - Python scripts put it back together
- Need to explore pNetCDF, probably using specific MPI I/O processes to overlap I/O and computation
- No input files (other than namelist)
- Output size is controlled by size of grid and number of variables used by the scheme, typically <1GB for the grids I have been testing
- No checkpoint/restart capabilities right now
  - Would like to put this in later

F) Visualization and Analysis

- How do you explore the data generated?
  - Python scripts produce plots and generate reports
  - Plotting is automated based on the variables defined by the scheme
  - Generic Voronoi mesh plotting code written on top of Matplotlib
- Do you have a visualization workflow?
  - 1 script does everything at this point
    - Would like to use a cfg file to control what is plotted/analyzed and how
- Plans to incorporate movies of field evolution

G/I) Performance and Scaling

- Haven't had an issue with speed using workstation for test cases (small though)
- Unsure of performance/scaling bottlenecks
- PETSC has proven scalability well beyond what I need
  - Just need to make sure my code doesn't slow it down
- Need to start profiling (Tau?)
- This is something I haven't really explored yet
- Would like to be scalable out to ~1000 cores in the next year

H) Tools

- Debugging:
  - Gdb for single-core stuff
  - Combo of gdb and print statements for parallel stuff
    - Need a better debugging solution here
- Other tools
  - Use makefiles for compilation
  - Uses Cheetah (python templating engine) to auto-generate code based on scheme/variable configuration
- Plans to move project to GitHub or Google Code and use Git for code management and bug-tracking

J) Roadmap

- Primary: need to finish parallelization of shallow water model and eigensolver
- Long-Term: Want to have shallow water model and eigensolver fully implemented for scalable, 1000 core performance in the next year
- Would like to add cubed-sphere grids and finite element-type methods (spectral element, discontinuous Galerkin)
- NTM should be a useful framework for testing and evaluating different schemes and grids for the non-linear shallow water equations