

High Performance Multi-Layer Ocean Modeling

Craig C. Douglas^A, Gundolf Haase^B, and Mohamed Iskandarani^C

^A University of Kentucky, Computer Science Department, 325 McVey Hall,
Lexington, KY 40506-0045, USA.

^B Johannes Kepler University Linz, Institute for Analysis and Computational
Mathematics, Department of Computational Mathematics and Optimization,
Altenberger Strasse 69, 4040 Linz, Austria.

^C University of Miami, Rosenstiel School of Marine and Atmospheric Science,
4600 Rickenbacker Causeway, Miami, FL 33149-1098, USA.

Abstract: We are in the process of simulating the wind-driven circulation in the Pacific Ocean. Our aim is to investigate the impact of short scale wind forcing on the oceanic circulation. The model used is the isopycnal version of SEOM which allows baroclinic processes for a relatively low computational cost. Here we present a scalability study of the model on several different parallel computers.

Key words: ocean modeling, iterative methods, numerical linear algebra, partial differential equations, substructuring, parallel computing.

1. Spectral Element Ocean Models (SEOM)

There are three forms of the SEOM that are being investigated. A *single layer model* is really a two (or 1.5) layer approach. It is good for wind circulation and abyssal flows. Global tides and estuarine modeling are adequately accounted for, but not ocean circulation. A *multiple layer model* is accurate enough for wind driven circulation using 2-9 layers. In this paper, five layers are used. Finally, a full *three dimensional continuous stratification model* has gravitational adjustments and is accurate enough for overflows and basin circulation.

The highlights of the SEOM include that it is an h-p type finite element method with C^0 continuity. It has geometric flexibility (see Figure 1) that results in very low phase errors and numerical dissipation. There are cache aware, CPU intensive dense computational kernels requiring $O(KN^3)$ operations that lead to excellent scalability on parallel computers overall.

Motivation for layered SEOM lies in several key points. It is faster as well as both mathematically and computationally simpler than the full SEOM-3D. We avoid cross isopycnal diffusion and pressure gradient errors. However, baroclinic processes possible with 2 layers and eddy resolving simulations can be produced relatively easily and cheaply.

The layered model equations are

$$d\mathbf{u}_k/dt + \mathbf{f} \times \mathbf{u}_k = -\nabla \Phi_k + (\tau_k - \tau_{k+1} + \nabla \cdot (\nu h_k \nabla \mathbf{u}_k))/h_k ; \eta_k + \nabla \cdot [(H_k + \eta_k) \mathbf{u}_k] = 0$$

The Montgomery potential is $\Phi_k = g \zeta_1 + \Phi_{k-1} + g\Delta\rho_k z_k$, $k>0$, where Φ_1 is the barotropic pressure contribution to the Montgomery potential. The thickness anomaly of layer k is $\eta_k = \zeta_k - \zeta_{k+1}$ with $\zeta_{N+1} = 0$. The total depth of the fluid is $H = H_1 + \dots + H_N$. The vertical coordinate of the surface interface of layer k is $z_k = z_{k+1} + h_k$. The stress on the layer k is τ_k , where τ_1 is surface wind stress, τ_{N+1} is the bottom drag coefficient, and τ_{k+1} is the interfacial drag coefficient.

Our model has several limitations worth noting. Layers are not allowed to disappear, even temporarily. When a layer thickness goes below a critical point, entrainment occurs and the model equation for that layer temporarily becomes

$$h_t + \nabla \cdot (h\mathbf{u}) = w^t.$$

The topography is confined to the deepest layer, which leads to thin layers near coasts. Finally, there are no thermodynamics included.

2. Solution Strategies

The time discretization is given by a third order Adams-Bashford explicit on terms except surface gravity waves, which uses a Backward Euler implicit method. The implicit terms are isolated to two dimensional equations. A preconditioned conjugate gradient method is used to solve the implicit problems.

At each time step a filtering step transfers information between layers. For a five layer model, 12 filtering steps must be done per time step. Each layer has to solve

$$-\Delta \mathbf{u} = \mathbf{f} \text{ in } \Omega + \text{boundary conditions with } \mathbf{f} = \nabla \delta - \nabla \times \omega.$$

where ω denotes the filtered vorticity and δ is the filtered divergence field. The filtering is done by series expansion and the Boyd-Vandeven filter in each spectral element. On each of the 5 layers we solve

$$-\Delta \mathbf{u} = \mathbf{f}_u \text{ and } -\Delta \mathbf{v} = \mathbf{f}_v.$$

The spectral elements themselves are interesting from a computational viewpoint. A Gauss-Lobatto discretization is used. Each element provides the support for a Smagorinsky subgrid parameterization that leads to an inner finite element discretization.

The code used runs only in parallel. The speedup formula in two dimensions is

$$S = P / (1 + T_c / (K T_o)) \text{ and } T_c / (K T_o) \sim 1 / (K^{1/2} N^2),$$

where P is the number of processors, K the number of spectral elements per processor, N is the spectral truncation, T_c is the communication time, and T_o is the computational time.

SEOM has a number of advantages over traditional high order finite difference methods. The speedup formula shows that the speedup deteriorates as the second term in the denominator increases. This second term decreases quadratically with the spectral truncation, and like the square root of the number of elements in the partition. The formula also shows the distinguishing property of the spectral element method which gives it its coarse grain character: the communication cost increases only linearly with the order of the method while its computational cost increases cubically, yielding a quadratic ratio between the two. High order finite difference methods, by contrast, show a quadratic increase of the communication cost with the order, since the halo of points needed to be passed between processors increases.

3. Numerical Example

We include one example in this section. It is a wind driven simulation of the northeast Pacific ocean, though a grid covering all oceans is used. The configuration used includes the following features:

- Five layers' thicknesses: 135m, 185m, 230m, 250m, and the rest.
- g' : 0.02, 0.009, 0.004 and 0.002 m/s².
- Topography: $h_m = (h-h_{\max}) 0.8 + h_{\max}$.
- $1460 \text{ m} \leq h_m \leq h_{\max}$.
- Interfacial Drag $v\Delta u/\Delta z = -\gamma\Delta u$, $v = 0.01, 0.001, 0.0001, 0.0001 \text{ m/s}^2$.
- Entrainment: Critical thickness 40m for layer 1 and 50m for other layers.
- 3552 elements and 7th order spectral expansions.
- 64 collocation points within each element.
- High resolution in Pacific (20-30 km), low resolution elsewhere.
- No open boundaries.

Figure 1 shows the surface grid used. It is replicated in the z-axis underwater on each ocean level. Figure 2 shows a sample of sea surface displacement. Data is produced for making movies on any of the layers. The code has been history matched (16 years) for accuracy using satellite data.

Finally, Figure 3 shows how the code performs on 2-48 processors on a variety of machines. These include tightly coupled supercomputers (HP SPP2200 and N-class, SGI Origin 2000, and IBM Ascii Blue Pacific) and a pair of clusters (RR: an Intel based machine and a SUN cluster). The Intel cluster is made from 128 Intel Pentium II's (450 MHz) with 1Gb/s networking. The SUN cluster is made from 20 SUN UltraSparc5's (333 MHz) with 100 Mb/s networking.

There are clear advantages to running on a cluster: cheap supercomputing, fast turnaround time, data is local and readily available, and a flexible configuration. Possibly more important is that the queuing method is negotiable with local colleagues instead of with an impersonal piece of software.

There are also clear advantages to running on a tightly coupled supercomputer: very fast turnaround time if queues are short, usually there is a huge, centralized disk farm, and on large SMP's with a single system image, updating the system can be accomplished in a reasonable amount of time and effort.

4. Ongoing Work

The code runs very well on parallel computers with caches. However, we are actively working on a number of improvements. The first is that there are a number of cache effects that we can draw upon by solving all of the filtering problems simultaneously. This means that multiple right hand sides and solutions will be computed on, which will further localize data.

The second major area involves using subgrid substructuring methods. During 2001 we expect to report on a number of different techniques in two dimensions. We intend to explore extensions to three dimensions later.

Finally, we have preliminary results using a parallel algebraic multigrid procedure that we intend to report on during 2001 and beyond.

Acknowledgments

This research was supported in part by the National Science Foundation (ACR-9721388, DMS-9707040, and CCR-9902022), University of Alaska at Fairbanks (Supercomputer Center), University of New Mexico (AHPCC), University of Illinois (NCSA), University of Kentucky (Computing Center), and the University of California and Department of Energy (Lawrence Livermore National Laboratory).

Figure 1: Computational grid on the ocean surface

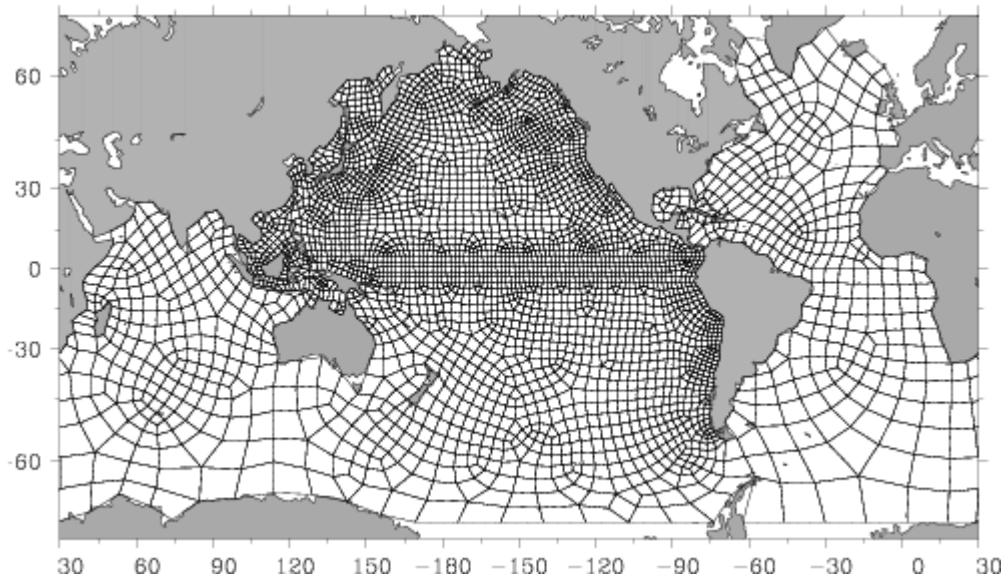


Figure 2: Sea surface displacement

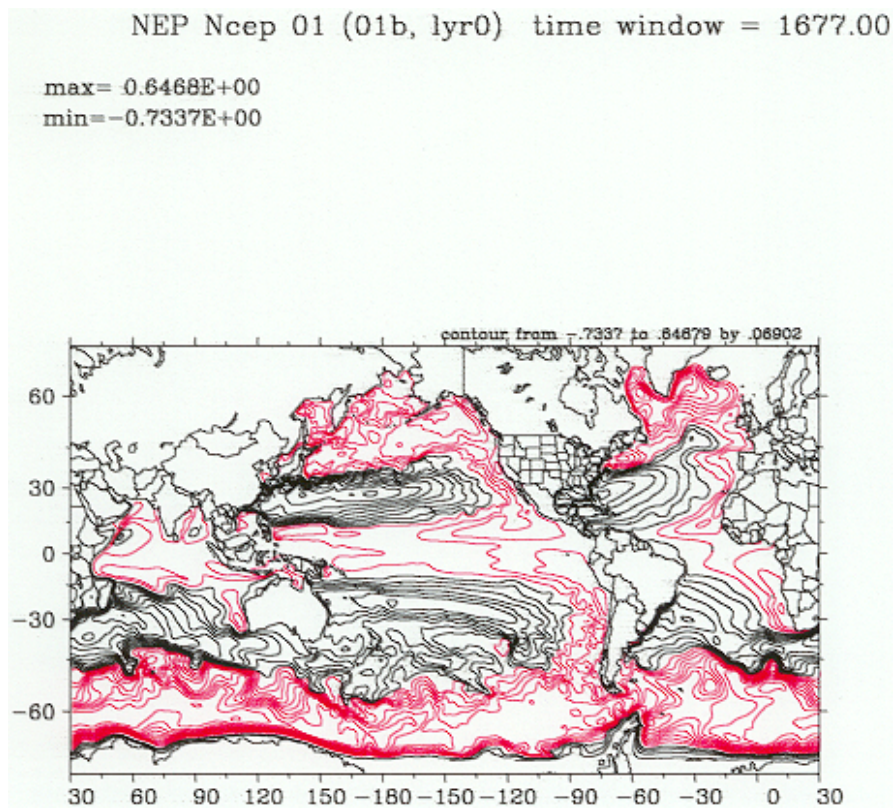


Figure 3: Wall clock times for 200 time steps

