An Acceleration Procedure for the Spectral Element Ocean Model Formulation of the Shallow Water Equations

Craig C. Douglas, Gundolf Haase, and Mohamed Iskandarani

Abstract. We develop an overlapping block preconditioner based on an additive Schwarz method for preconditioning the system of equations arising from discretizing the shallow water equations using the spectral ocean element method approach. The system is complicated by being sparse, but having a very large number of nonzeroes in any row of the matrix, which is never stored. The resulting coupled system of equations will be reduced to a Schur complement system with a special structure of the Schur complement. A major section of the computation is shown to be speeded up by a factor of approximately 3, independent of the number of processors used (within reason).

1. Introduction and the numerical model

In this paper we investigate speeding up a major part of the isopycnal (a constant density surface) version of the Spectral Element Ocean Model [**4, 5**], namely the implicit integration of the sea surface height. The novel feature of this model is the combination of isopycnal coordinates in the vertical and spectral element discretization in the horizontal. The benefits of the spectral element discretization include: geometric flexibility, dual h-p paths to convergence, low numerical dispersion and dissipation errors, and dense computational kernels leading to extremely good parallel scalability.

We first define our notation. Let $\vec{u} = (u, v)$ be the velocity vector. Let ζ be the sea surface displacement, and, because of hydrostaticity, also stand for the pressure. Let g be the gravitational acceleration. Let h be the resting depth of the fluid. Let Q be a mass source/sink term. Finally, $\vec{F} = (f^x, f^y)$ is a generalized forcing term for the momentum equations that includes the Coriolis force, non-linear advection, viscous dissipation, and wind forcing.

The shallow water equations govern the evolution of the sea surface height; their vector form is:

$$\left\{ \begin{array}{rcll} \vec{u}_t + g\nabla\zeta & = & \vec{F} & \text{(momentum)}, \\ \zeta_t + \nabla \cdot [(h+\zeta)\vec{u}] & = & Q & \text{(continuity)}. \end{array} \right. \tag{1.1}$$

1991 *Mathematics Subject Classification.* 68W10, 65Y05, 47N40, 76D33.

This research has been partially supported by NSF grants DMS-9707040, ACR-9721388, ACR-9814651, CCR-9902022, and CCR-9988165, National Computational Science Alliance grant OCE980001 (utilizing the University of Illinois Origin 2000.

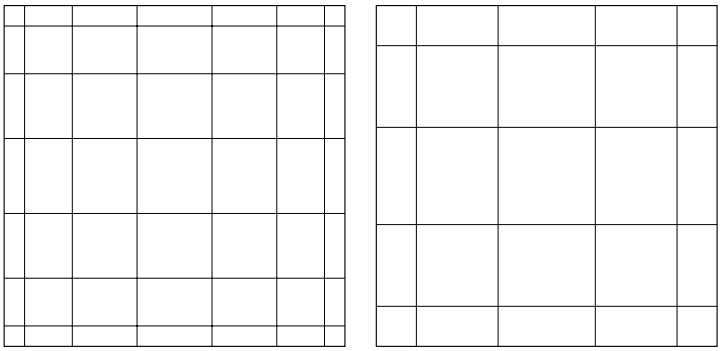

FIGURE 1. Gauss-Lobatto points for the velocity (left) and pressure (right) unknowns in a typical spectral element.

For simplicity, we assume no-slip boundary conditions. The free waves of this system include suface gravity waves which propagate at the speed of $\sqrt{gh}$. Their speed can greatly exceed the advective velocity of the fluid in the deep part of the ocean. As a result there is a significant stability restriction in most 3D ocean models [**4, 5**].

The terms responsible for the gravity wave speed are the pressure gradient term in the momentum equation, $g\nabla\zeta$, and the divergence term in the continuity equation $\nabla\cdot(h\vec{u})$. These terms must be integrated implicitly in order to avoid severe time step restrictions. We adopt a semi-implicit integration procedure where the gravity terms are discretized with a Crank-Nicholson scheme and the remaining terms with a third order Adams Bashforth scheme.

Let $\tau = \Delta t$, and let ϕ and ψ denote test functions for the velocity and pressure spaces; then the time discretized variational equations of (1.1) become:

$$\begin{aligned}
\frac{1}{\tau}\int_A u^{n+1}\phi dA &+ \frac{1}{2}\int_A g\zeta_x^{n+1}\phi dA = \frac{1}{\tau}\int_A u^n\phi dA - \frac{1}{2}\int_A g\zeta_x^n\phi dA \\
&+ \sum_{p=0}^{2}\alpha_p\int_A f^{x(n-p)}\phi dA, \\
\frac{1}{\tau}\int_A v^{n+1}\phi dA &+ \frac{1}{2}\int_A g\zeta_y^{n+1}\phi dA = \frac{1}{\tau}\int_A v^n\phi dA - \frac{1}{2}\int_A g\zeta_y^n\phi dA \\
&+ \sum_{p=0}^{2}\alpha_p\int_A f^{y(n-p)}\phi dA, \\
\frac{1}{\tau}\int_A \zeta^{n+1}\psi dA &- \frac{1}{2}\int_A \nabla\psi\cdot h\vec{u}^{n+1}dA = \frac{1}{\tau}\int_A \zeta^n\psi dA + \frac{1}{2}\int_A \nabla\psi\cdot h\vec{u}^n dA \\
&+ \sum_{p=0}^{2}\alpha_p\left(\int_A Q^{n-p}\psi + \nabla\psi\cdot\vec{u}^{n-p}\zeta^{n-p}\right)dA.
\end{aligned}$$

The integration by parts of the continuity equation is important for the solution procedure since it converts the pressure Schur complement into a symmetric positive definite matrix.

For spatial discretization, we employ the spectral element method. The computational domain is divided into elements where the unknowns are interpolated with Legendre cardinal functions collocated at the Gauss-Lobatto points (see Fig. 1). For each element we have

$$\vec{u}(\xi,\eta) = \sum_{k,l=1}^{N^v} \vec{u}_{k,l}\mu_k^v(\xi)\mu_l^v(\eta) \text{ and } \zeta(\xi,\eta) = \sum_{i,j=1}^{N^p} \zeta_{i,j}\mu_i^p(\xi)\mu_j^p(\eta)$$

with μ_k^v and μ_i^p as local 1D basis functions for velocity and pressure, respectively. We use a lower order polynomial for the pressure in order to suppress spurious pressure modes [**5**]. Hence, $N^p = N^v - 2$.

The time-discretized variational equations are reduced to a system of algebraic equations using a Galerkin formulation. Define M^v, and M^p as the mass matrices for the velocity and pressure unknowns, G^x and G^y as the discrete gradient operators in the (x, y) directions, and D^x and D^y as the components of the discrete gradient operator. Then the system of equations can be written as

$$\begin{aligned} \frac{1}{\tau}M^v u + \frac{1}{2}gG^x\zeta &= a, \\ \frac{1}{\tau}M^v v + \frac{1}{2}gG^y\zeta &= b, \\ -\frac{1}{2}D^x u - \frac{1}{2}D^y v + \frac{1}{\tau}M^p\zeta &= c, \end{aligned} \tag{1.2}$$

where a, b, and c are obvious.

The integrals resulting from the spatial discretization are evaluated using Gauss-Lobatto quadrature of the same order as the interpolation polynomial. For both velocity and pressure we get a diagonal matrix.

For the divergence terms and recalling that (i, j) corresponds to velocity points and (k, l) to pressure points, we have

$$D^x_{ij,kl} = h_{kl}G^x_{kl,ij} \text{ and } D^y_{ij,kl} = h_{kl}G^y_{kl,ij}.$$

where the relation $D^s = (G^s)^T H$ holds at a global level due to the continuity of h_{kl}.

By substitution and using the fact that $D^s = (G^s)^T H$ and that M^v is a diagonal matrix, the system (1.2) can easily be reduced to a system of equations in ζ since

$$S\zeta := \left[\frac{M^p}{\tau} + \frac{\tau g}{4}\left((G^x)^T H(M^v)^{-1}G^x + (G^y)^T H(M^v)^{-1}G^y\right)\right]\zeta = f \tag{1.3}$$

with the Schur complement S and the right hand side

$$f := c - \frac{\tau}{2}\left[(G^x)^T H(M^v)^{-1}a + (G^y)^T H(M^v)^{-1}b\right].$$

In our code, the matrix S is not stored due to memory constraints. There exist many entries from nodes of an element to nodes in elements two layers away. We get rows in the sparse matrix that have a very large number of terribly small entries. Simply eliminating the small entries does not work.

Hence, the matrix S is only available via the matrix-vector operation $S \cdot w$. The implemented implicit matrix-vector multiplication is as fast as conventional multiplication since it takes advantage of the local tensor product structure of the matrix.

2. Preconditioners

The system (1.3) is similar to a discretization of the Schroedinger-like equation

$$-\frac{g}{4}\nabla^T\left(h(x)/m(x)\nabla\zeta(x)\right)+\frac{1}{\tau^2}\zeta(x)=\frac{1}{\tau}f(x),\qquad x\in\Omega. \tag{2.1}$$

The matrix $S_{N\times N}$ is clearly symmetric, positive definite. Hence, (1.3) can be solved using a preconditioned CG.

A simple diagonal preconditioner of S is defined using the available matrix-vector multiplication

$$\{C_{ii}\}_{i=1}^N := \{(S\cdot 1)_i\}_{i=1}^N. \tag{2.2}$$

This is a diagonal preconditioner C and is just the lumped Schur complement.

We can derive a far superior preconditioner using an additive Schwarz method. We denote the restriction from a global vector to the local components of an element r by $\mathcal{A}_r : \mathbb{R}^N \mapsto \mathbb{R}^{N_r}$, $r = 1,\ldots,\text{nelem}$.

The original matrix S can no longer be expressed in terms of element matrices. These matrices are no longer local to the element due to the long connections between nodes of all surrounding elements. However, suppose we only consider the local element contributions in the generation of local matrices $\widehat{S}_r$, i.e., we restrict the support of the transformed pressure basis functions to the element. Then we can derive a matrix

$$\begin{aligned}\widehat{S} &= \sum_{r=1}^{\text{nelem}} \mathcal{A}_r^T \widehat{S}_r \mathcal{A}_r \\ &= \sum_{r=1}^{\text{nelem}} \mathcal{A}_r^T \left[\tfrac{M_r^p}{\tau}+\tfrac{\tau g}{4}\left((G_r^x)^T H(M_r^v)^{-1}G_r^x+(G_r^y)^T H(M_r^v)^{-1}G_r^y\right)\right]\mathcal{A}_r.\end{aligned} \tag{2.3}$$

The local matrices are similar to (2.1) in the domain Ω_r with homogeneous Neumann conditions on the element boundary $\partial\Omega_r$.

We now have two ways of calculating the $\widehat{S}$ matrices. First, we can derive $\widehat{S}$ as stiffness matrix of the local problem (2.1). Second, we can use a modified matrix times vector routine that operates only locally in the element. Applying this routine to a vector $[0,\ldots,0,1_j,0,\ldots,0]^T$ results in column entries $S_{r,i,j}$ for all local rows i.

We realized the second version since it is more straightforward even though it consumes more CPU time than the first one. Besides being much easier to implement, it is only a once a run expense. Since a normal run is a very large number of time steps (usually tens of thousands of time steps), the added cost over the first version is negligible over an entire simulation.

The matrices $\widehat{S}_r$ really represent the local properties of S. We find that $S\cdot 1 \equiv \widehat{S}\cdot 1$ holds, i.e., $\widehat{S}$ is some sort of a blockwise lumped S. All matrices $\widehat{S}_r$ are nonsingular as long as τ does not tend to infinity.

We can derive an additive Schwarz style preconditioner for $\widehat{S}$ and also for S. It is defined by the inverse

$$\widehat{C}^{-1} = W^{-1}\sum_{r=1}^{\text{nelem}} \mathcal{A}_r^T \widehat{S}_r^{-1}\mathcal{A}_r. \tag{2.4}$$

Matrix $\widehat{C}^{-1}$ is not block diagonal, but its blocks overlap only in rows/columns shared by more than one element. This requires in (2.4) the diagonal weight matrix

$W = \sum_{r=1}^{\text{nelem}} \mathcal{A}_r^T \mathcal{A}_r$ containing the number of elements a node belongs to (i.e., these are the weights needed for partition of unity). An analysis of the preconditioner $\widehat{C}$ for $\widehat{S}$ can be found in [**1, 2, 7, 8**] and the references therein.

The parallelization of the preconditioner $\widehat{C}^{-1}$ is straightforward and requires only one next neighbor communication per application [**3**].

3. Numerical results and conclusions

We have tested the new solver on a several problems. We report on a typical one here using the NEP08 dataset. We show that when using 2-48 processors that the new preconditioner is much faster than the traditional one used in the field.

The focus of this simulation is the Northern Pacific Ocean, with particular interest in the Gulf of Alaska, and the equatorial and northern coastal wave guide. We use a domain including all of the oceans globally in order to avoid open boundaries. The grid emphasizes our areas of interests which are discretized with small elements with a typical grid resolution ranging from 20 km in the Gulf of Alaska to about 35 km in the parts of the Pacific Ocean that are of interest for long term study of sea surface heights. The element sizes are increased gradually away from the Pacific Ocean in order to reduce the cost of the global simulation. The grid contains 3552 elements, 176377 velocity nodes, and 90459 pressure nodes.

Our focus is comparing the new preconditioner $\widehat{C}$ from (2.4) with the old preconditioner C from (2.2). The preconditioned CG stops when the relative error measured in the $\|\cdot\|_{SC^{-1}S}$ norm, respective in the $\|\cdot\|_{S\widehat{C}^{-1}S}$ norm, is reduced by 10^{-4}. We note that the solution of the shallow water equations consummed a large portion of the model CPU time and improvements in the model performance translates into a substantial reduction in the overall cost.

we are interested in the parallel performance of the two preconditioners. The domain is partitioned into p subdomains, where $p \in \{2, 4, 6, 8, 12, 16, 32, 48\}$. One subdomain is used per processor. We used METIS [**6**] to calculate the subdomains. METIS is only used once per domain and number of processors as a preprocessing step. Hence, its cost is zero in a simulation.

We have run tests on several platforms: SGI Origin2000 (IRIX), SGI 3800 (IRIX), HP N-Class 4400 (HP-UX), and a cluster of dual Intel Pentium III (Linux). In each case we see a speedup of between 2.4 and 3.1 for 2 to 48 processors. For a single processor, we see a speedup of between 2.8 and 3. The number of Uzawa interations is reduced by a roughly a factor of 4.

The reduction of the coupled system (1.2) to the reduced system (1.3) does not provide an explicit matrix. The local approximation of the matrix of the reduced system (2.3) is the right way to derive a good preconditioner for CG.

The preconditioner setup costs approximately 10 Uzawa iterations. This can be neglected with respect to the overall solution time. In a production run of 14 to 16 years, we have to solve (1.2) for up to a few million time steps using thousands of CPU hours. The setup costs simply are not measurable overall.

The new preconditioner $\widehat{C}^{-1}$ from (2.4) reduces the number of iterations in the Uzawa algorithm by a factor of 4 and the wall clock CPU time by a factor of 3 in the examples studied to date.

References

1. M. Dryja and O. B. Widlund, *Some domain decomposition algorithms for elliptic problems*, Iterative Methods for Large Linear Systems (New York), Academic Press, 1989, pp. 273–291.
2. M. Griebel and P. Oswald, *On the abstract theory of additive and multiplicative Schwarz algorithms*, Numer. Math. **70** (1995), 163–180.
3. Gundolf Haase, *New matrix-by-vector multiplications based on a non-overlapping domain decomposition data distribution*, Proceedings of the Euro-Par97 (Heidelberg), Lecture Notes in Computer Sciences, vol. 1300, Springer, 1997, pp. 726–733.
4. D. B. Haidvodel and A. Beckmann, *Numerical ocean circulation modeling*, Environmental Science and Management, vol. 2, Imperial College Press, London, 1999.
5. M. Iskandarani, D. B. Haidvogel, and J. P. Boyd, *A staggered spectral element model with application to the oceanic shallow water equations*, International Journal for Numerical Methods in Fluids **20** (1995), 393–414.
6. G. Karypis, *METIS serial graph partitioning and matrix ordering software*, In URL http://www-users.cs.umn.edu/∼karypis/metis/metis/main.shtml, Department of Computer Science and Engineering, University of Minnesota, Minneapolis, MN, USA.
7. B. F. Smith, P. E. Bjørstad, and W. D. Gropp, *Domain decomposition: Parallel multilevel methods for elliptic partial differential equations*, Cambridge University Press, New York, 1996.
8. J. Xu, *Iterative methods by space decomposition and subspace correction: A unifying approach*, SIAM Review **34** (1992), 581–613.

University of Kentucky, Department of Computer Science, 325 McVey Hall-CCS, Lexington, KY 40506-0045, USA and Yale University, Department of Computer Science, P.O. Box 208285, New Haven, CT 06520-8285, USA.

E-mail address: douglas@ccs.uky.edu or douglas-craig@cs.yale.edu

Johannes Kepler University of Linz, Institute of Analysis and Computational Mathematics, Altenberger Str. 69, A–4040 Linz, Austria.

E-mail address: ghaase@numa.uni-linz.ac.at

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

E-mail address: MIskandarani@rsmas.miami.edu