SIAM J. SCI. COMPUT.
Vol. 46, No. 4, pp. B554–B578

IMPLICIT ADAPTIVE MESH REFINEMENT FOR DISPERSIVE TSUNAMI PROPAGATION*

MARSHA J. BERGER† AND RANDALL J. LEVEQUE‡

Abstract. We present an algorithm to solve the dispersive depth-averaged Serre–Green–Naghdi equations using patch-based adaptive mesh refinement. These equations require adding additional higher derivative terms to the nonlinear shallow water equations. This has been implemented as a new component of the open source GeoClaw software that is widely used for modeling tsunamis, storm surge, and related hazards, improving its accuracy on shorter wavelength phenomena. We use a formulation that requires solving an elliptic system of equations at each time step, making the method implicit. The adaptive algorithm allows different time steps on different refinement levels and solves the implicit equations level by level. Computational examples are presented to illustrate the stability and accuracy on a radially symmetric test case and two realistic tsunami modeling problems, including a hypothetical asteroid impact creating a short wavelength tsunami for which dispersive terms are necessary.

Key words. dispersive equations, implicit mehtods, adaptive mesh refinement, subcycling in time, GeoClaw

MSC codes. 65M50, 65M08, 86A15

DOI. 10.1137/23M1585210

See reproducibility of computational results at end of the article.

1. Introduction. Tsunami propagation and inundation is frequently modeled using the depth-averaged nonlinear shallow water equations (SWE). The use of adaptive mesh refinement (AMR) is often crucial for realistic problems, since cell sizes of several km can often be used in the deep ocean, while inundation of coastal regions generally requires a horizontal resolution of 10 m or less. In our version of patch-based mesh refinement, a nested set of finer patches is created and superimposed on the original coarse domain, time-stepping occurs with an appropriate time step for each level patch, and regridding occurs every few time steps so that the finer level patches can track the phenomena of interest in a wave-propagation problem. Several figures in this paper show outlines of the refined patches, providing a better idea of the how the algorithm works. The use of patch-based AMR in computational fluid dynamics has been well documented in [9, 11, 46, 24, 11]; see also [49] for an extensive list of software packages and applications. Our implementation for the SWE is called GeoClaw and is distributed as part of the open source Clawpack software [15, 44]. It uses high-resolution Godunov-type explicit finite volume methods with AMR, as described in detail in [10, 20, 34]. This software is widely used for tsunami modeling (among other applications; see [19]) and has been accepted as a validated model by

*Submitted to the journal's Software, High-Performance Computing, and Computational Science and Engineering section July 11, 2023; accepted for publication (in revised form) May 7, 2024; published electronically August 14, 2024.
https://doi.org/10.1137/23M1585210
Funding: This work was partially supported by the NASA Asteroid Threat Assessment Project (ATAP) through the Planetary Defense Coordination Office and BAERI contract AO9667.
†Flatiron Institute, New York, NY 10010 USA, and Courant Institute, New York University, New York, NY 10012 USA (mberger@flatironinstitute.org).
‡Department of Applied Mathematics, University of Washington, Seattle, WA 98195 USA, and HyperNumerics LLC, Seattle, WA 98125 USA (rjl@uw.edu).

the U.S. National Tsunami Hazard Mitigation Program (NTHMP) after conducting multiple benchmark tests as part of an NTHMP benchmarking workshop [53].

The SWE are a hyperbolic system of equations, for which explicit methods work very well. These equations are a *long wavelength* approximation, where the underlying assumption is that the water depth is much smaller than the horizontal length scale. This is generally suitable for large-scale earthquake-generated tsunamis, for which the wavelength can be 50 to 100s of km, whereas the typical ocean depth is 4 km. But for some earthquake-generated tsunamis dispersive equations are more appropriate, and this is almost always true when modeling phenomena with smaller length scales, such as tsunamis generated by landslides or asteroid impacts. This has been discussed in great detail in previous papers on this topic, several of which show comparisons between simulations with SWE and dispersive models, e.g., [4, 12, 21, 28, 30, 31, 38, 40, 37, 45, 52, 54, 57, 59]. A depth-averaged system of equations is still important for efficient transoceanic propagation. But instead of the SWE we turn to a Boussinesq-type equation, which retains more terms from depth-averaging the three-dimensional (3D) fluid dynamics. These equations, described in more detail below, involve second- and third-order derivatives, and stability constraints typically require solving an implicit system of equations each time step. As a result, solving them can be much more expensive than solving SWE, for which explicit time-stepping can be efficiently used. As with SWE, the use of AMR can dramatically decrease the computational expense, and becomes even more important when solving problems with short wavelength waves, since much higher resolution may be required in some regions to capture these waves accurately. But AMR also becomes much more complex to implement when implicit systems must be solved on the hierarchy of grids that do not cover the full domain at most refinement levels.

The goal of this work is to extend patch-based mesh refinement, as implemented in GeoClaw, to incorporate the solution of an elliptic system each time step. By using highly efficient algebraic multigrid methods and the MPI implementation of PETSc, we have achieved an implementation that can still solve large-scale realistic tsunami modeling problems on a laptop.

We implement a form of the Serre–Green–Naghdi (SGN) equations developed by Tissier et al. [56], which is presented in the next section. In a previous paper [8], we developed a similar implementation for another Boussinesq-type system, the Madsen–Sorensen [43] equations (denoted MS below), but the SGN equations give a more stable and robust computational method with similar results. The elliptic system is formulated and solved separately on each refinement level, but includes all patches at that level. We do this by including the solution variables of the elliptic equation in the vector of state variables, so that they can be used as patch boundary conditions (ghost cells) on finer level patches in the same way as the conserved variables. This allows the AMR procedure to incorporate subcycling in time, so that smaller time steps can be taken on the finer grid patches. We compare results to those obtained with a composite solver that does not involve subcycling, in order to verify the accuracy of our procedure.

There are other Boussinesq solvers that are adaptive in space, including the work by Popinet [52], implemented in the Basilisk software [6]. Our work follows his approach, but with the added component of refinement in time. Other open source tsunami modeling codes that implement dispersive terms (e.g., [1, 16, 18, 48]) do not support general AMR, although many are based on unstructured grids so that much finer grids are used near coastal regions of interest, or they allow static nested levels of refinement. There is also previous work on incompressible flow that uses AMR and

includes subcycling in time. The first such work that we are aware of is the method of Almgren et al. [2] for variable density incompressible flow. Their algorithm used an explicit upwind method for the convective terms, with the viscous term handled implicitly, and the incompressibility constraint imposed by solving an elliptic system. Subsequent recent work by Zeng et al. [58] is of the same nature. Our problem is easier, since our method is less tightly coupled and does not involve an incompressibility constraint, but it still requires solving an elliptic system each time step. We propose a different algorithm that also allows for refinement in time.

Near shore it is necessary to switch from the dispersive SGN equations back to SWE in order to better model wave breaking in very shallow water and to robustly handle wetting and drying during coastal inundation. A variety of criteria have been explored in the literature for optimizing this transition to best capture wave breaking, e.g., [25, 32, 33, 56]. Here we follow [26, 28] and adopt the simplest approach that is widely used, consisting of using SWE in any grid cell where the initial water depth in the ocean at rest for that cell or any of its nearest neighbors is less than some specified tolerance, typically 5 or 10 m depth.

The grid adaptation and placement are handled automatically in GeoClaw using a variety of options, such as wave height, adjoint-based error estimation, or simply forcing a high level of refinement around particular coastal regions of interest. The GeoClaw software incorporates OpenMP for the hyperbolic step by parallelizing over grid patches. The implicit system arising in the Boussinesq equations is solved using an algebraic multigrid preconditioned Krylov solver in PETSc [5]. We use the recently introduced PETSc version 3.20 that allows the use of MPI for the linear solver in combination with the OpenMP parallelization used by GeoClaw.

This paper is organized as follows. Section 2 introduces the depth-averaged SGN equations that have been incorporated into GeoClaw. We also include the radially symmetric 1D version of these equations used to compute a fine grid reference solution for some of our 2D test problems. Section 3 contains a description of the single grid SGN solver, and then the block-structured AMR strategy to solve the implicit system of equations required by the SGN solver. Section 4 contains three computational examples chosen to validate the stability and accuracy of the implementation and to illustrate its effectiveness for realistic problems. We discuss the run times of the new dispersive AMR code and compare them to similarly refined computations using only the SWE.

2. Equations. The 2D nonlinear SWE can be written as

$$\begin{aligned} h_t + (hu)_x + (hv)_y &= 0, \\ (hu)_t + \left(hu^2\right)_x + (huv)_y + gh\eta_x &= 0, \\ (hv)_t + (huv)_x + \left(hv^2\right)_y + gh\eta_y &= 0, \end{aligned} \tag{2.1}$$

where the surface elevation $\eta(x, y, t) = h(x, y, t) + B(x, y)$, with h the fluid depth and $B(x, y)$ the bottom topography. These are the equations solved in GeoClaw in a manner that handles the nonlinearity of wave breaking and on-shore inundation very robustly. However, these equations are nondispersive; the linearized equations have the dispersion relation $\omega(k) = k\sqrt{gh_0}$ and hence constant wave speed $\omega(k)/k = \sqrt{gh_0}$ for all wave numbers k.

The equations (2.1) are written in Cartesian (planar) coordinates and additional terms need to be added for solving real-world problems on the sphere. Coriolis terms can also be added on the sphere, and GeoClaw supports this option, but many experiments both with GeoClaw and by other researchers have concluded that Coriolis

terms have little effect on global tsunami propagation due to the very small fluid velocities in the deep ocean [21, 29]. Friction terms are also added to the momentum equations to model bottom friction based on a tunable Manning coefficient, as described in [34] and used by many standard tsunami modeling codes. We omit all these complications in this paper since they are all handled in the same manner as for SWE in the GeoClaw code.

A number of depth-averaged Boussinesq-type equations have been developed over the past several decades to model dispersive wave propagation; see [42] for one review. These are generally derived by keeping the next order terms in asymptotic expansions in powers of the ratio of water depth to wavelength. Different variants are often compared by computing the dispersion relation for the linearized version of the equations and seeing how well it models that of the Airy solution, which is derived based on the assumption of an incompressible and irrotational flow, allowing the full 3D velocity to vary with depth. The dispersion relation for the Airy solution is

$$\omega(k) = k\sqrt{gh_0 \tanh(kh_0)/kh_0}, \tag{2.2}$$

where k is the wavenumber for a wave with spatial wavelength $2\pi/k$ and ω is the temporal frequency.

In the previous work of Kim et al. [26, 28], GeoClaw was extended to solve the MS equations introduced by Madsen and Sørenson [43] by introducing an implicit solver for an elliptic equation at each time step. The resulting code, BoussClaw, has been used for solving several dispersive landslide-generated tsunami modeling problems (e.g., [23, 26, 36, 27, 35]). However, that extension was implemented only for a single grid patch at a fixed resolution and did not use the AMR aspects of GeoClaw.

In our work on extending the AMR algorithms to work for Boussinesq equations, we started with the BoussClaw code and our first AMR implementation again solved the MS equations. The algorithms and some sample results were presented in [8]. However, these algorithms exhibited poor stability properties at patch interfaces and nonreflecting boundaries, and numerous attempts to make them more stable have not resulted in a sufficiently robust code. We now believe there are inherent stability issues with these equations, which will be explored further elsewhere. Løvholt and Pedersen [39] also observed instabilities for similar equations when modeling flow over topography.

We have switched to using a different form of Boussinesq equations, an improved form of the SGN equations developed in [56], which includes a parameter α that can be tuned to match the Airy dispersion relation quite well. The dispersion relation for this improved SGN equation is

$$\omega(k) = k\sqrt{\frac{gh_0(1 + (\alpha - 1)(kh_0)^2/3)}{1 + \alpha(kh_0)^2/3}}. \tag{2.3}$$

Using $\alpha = 1$ gives the original SGN equations, but following the choice made by Popinet in the Basilisk code we use $\alpha = 1.153$ (which gives an imperceptible change in practice from the value 1.159 originally suggested in [56]).

For comparison, we also note that the MS equations have a tunable parameter β and dispersion relation

$$\omega(k) = k\sqrt{\frac{gh_0(1 + \beta(kh_0)^2)}{1 + (\beta + 1/3)(kh_0)^2}} \tag{2.4}$$

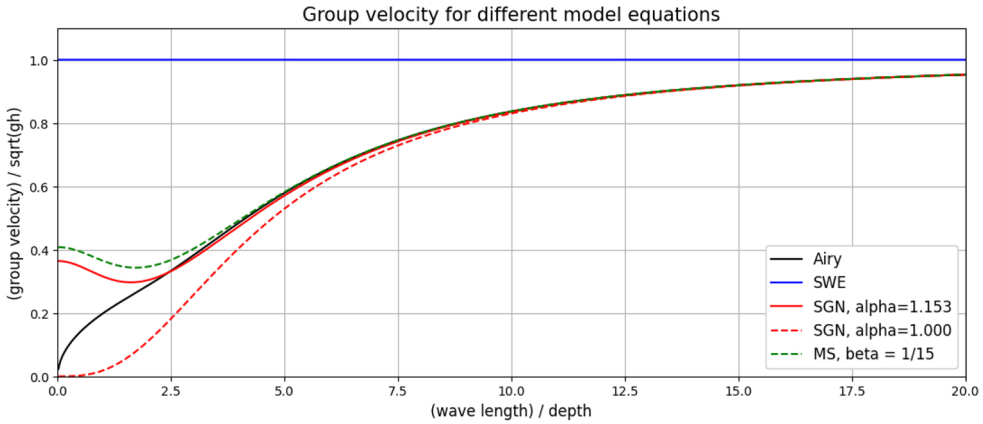

FIG. 1. *The scaled group velocity* $\omega'(k)/\sqrt{gh_0}$ *for the dispersion relations shown in the text, plotted versus* $2\pi/(kh_0)$*, the wavelength relative to the undisturbed fluid depth. In this work we implement SGN with* $\alpha = 1.153$*. (Color figures are online.)*

with the value $\beta = 1/15$ giving the best agreement with the Airy solution. Setting $\beta = 0$ in (2.4) or $\alpha = 1$ in (2.3) gives the dispersion relation of the original SGN equations. Figure 1 shows the group velocity $\omega'(k)$ plotted versus $2\pi/(kh_0)$ (the wavelength divided by the fluid depth) for each set of equations, scaled by $\sqrt{gh_0}$. Note that for wavelengths greater than approximately twice the fluid depth, the SGN equations with $\alpha = 1.153$ give a very good approximation to the Airy solution in the linearized case. In the long-wave limit all of the models asymptote toward the linearized shallow water limit. For very short wavelengths the Airy group velocity approaches 0. So does the group velocity for SGN with $\alpha = 1$, but much too slowly, often resulting in highly oscillatory waves that trail far behind the main wave in an unrealistic manner. By contrast, SGN with the value of $\alpha = 1.153$ has group velocities bounded away from zero. This is evident in some of the examples of section 4, where radially expanding waves leave a quiescent central zone in their wake, which is generally preferable to highly oscillatory waves that are not physically meaningful.

The SGN system of equations has the form of the SWE (2.1) with the addition of source terms:

$$
\begin{aligned}
&h_t + (hu)_x + (hv)_y = 0,\\
&(hu)_t + \left(hu^2\right)_x + (huv)_y + gh\eta_x = h\left(\frac{g}{\alpha}\eta_x - \psi_1\right),\\
&(hv)_t + (huv)_x + \left(hv^2\right)_y + gh\eta_y = h\left(\frac{g}{\alpha}\eta_y - \psi_2\right).
\end{aligned}
\tag{2.5}
$$

The vector $\vec{\psi} = [\psi_1, \psi_2]$ satisfies an elliptic equation of the form

$$
(I + \alpha\mathcal{T})\vec{\psi} = b, \tag{2.6}
$$

where $\vec{u} = [u, v]$ is the velocity vector. These equations agree with equation (4) of [52], but with a typo corrected (there is no h on the left-hand side). The second-order elliptic operator $\mathcal{T}$ and right-hand-side b are written in vector notation in [52]. Here we write them out in component form to clarify the derivatives that must be approximated in the finite difference implementation of these equations.

The components of the differential operator

$$\mathcal{T} = \begin{bmatrix} \mathcal{T}_{11} & \mathcal{T}_{12} \\ \mathcal{T}_{21} & \mathcal{T}_{22} \end{bmatrix}$$

are given by

$$\begin{aligned}
\mathcal{T}_{11} &= -\frac{h^2}{3}\partial_x^2 - hh_x\partial_x + \left(\frac{h}{2}B_{xx} + B_x\eta_x\right), \\
\mathcal{T}_{12} &= -\frac{h^2}{3}\partial_x\partial_y + \frac{h}{2}B_y\partial_x - h\left(h_x + \frac{1}{2}B_x\right)\partial_y + \left(\frac{h}{2}B_{xy} + B_y\eta_x\right), \\
\mathcal{T}_{21} &= -\frac{h^2}{3}\partial_x\partial_y - h\left(h_y + \frac{1}{2}B_y\right)\partial_x + \frac{h}{2}B_x\partial_y + \left(\frac{h}{2}B_{xy} + B_x\eta_y\right), \\
\mathcal{T}_{22} &= -\frac{h^2}{3}\partial_y^2 - hh_y\partial_y + \left(\frac{h}{2}B_{yy} + B_y\eta_y\right).
\end{aligned} \tag{2.7}$$

The right-hand-side b in (2.6) has components

$$\begin{aligned}
b_1 &= \frac{g}{\alpha}\eta_x + 2h\left(\frac{h}{3}\phi_x + \phi\left(h_x + \frac{1}{2}B_x\right)\right) + \frac{h}{2}w_x + w\eta_x, \\
b_2 &= \frac{g}{\alpha}\eta_y + 2h\left(\frac{h}{3}\phi_y + \phi\left(h_y + \frac{1}{2}B_y\right)\right) + \frac{h}{2}w_y + w\eta_y,
\end{aligned} \tag{2.8}$$

where

$$\begin{aligned}
\phi &= \partial_x\vec{u}\cdot\partial_y\vec{u}^{\perp} + (\nabla\cdot\vec{u})^2 \\
&= v_xu_y - u_xv_y + (u_x + v_y)^2, \\
w &= \vec{u}\cdot(\vec{u}\cdot\nabla)\nabla B \\
&= u^2B_{xx} + 2uvB_{xy} + v^2B_{yy}.
\end{aligned} \tag{2.9}$$

2.1. Radial symmetry. For testing purposes it is very convenient to consider problems in which the initial data and bathymetry are radially symmetric, in which case the Boussinesq equations reduce to equations in one space dimension r and time. Numerical solutions obtained with the 2D code can then be compared with fine grid solutions of the radial equations to verify that the complicated matrix equations have been properly implemented. This is done in subsection 4.1 below.

We use $\hat{h}(r,t)$ to denote the depth in radial coordinates and similarly for other variables. The radial velocity $\hat{u}(r,t)$ corresponds to the 2D velocity field $u(x,y,t) = \hat{u}(r,t)\cos(\theta), v(x,y,t) = \hat{v}(r,t)\sin(\theta)$. Converting the Boussinesq system to polar coordinates and assuming there is no variation in the θ direction, we obtain the equations

$$\begin{aligned}
\hat{h}_t + \frac{1}{r}\left(r\hat{h}\hat{u}\right)_r &= 0, \\
(\hat{h}\hat{u})_t + \frac{1}{r}\left(r\hat{h}\hat{u}^2\right)_r + g\hat{h}\hat{\eta}_r &= \hat{h}\left(\frac{g}{\alpha}\hat{\eta}_r - \hat{\psi}\right),
\end{aligned} \tag{2.10}$$

where $\hat{\psi}$ satisfies the 1D elliptic equation

$$(1 + \alpha\hat{\mathcal{T}})\hat{\psi} = \hat{b}. \tag{2.11}$$

In this case the second-order scalar differential operater $\hat{\mathcal{T}}$ is given by

$$\hat{\mathcal{T}} = -\frac{\hat{h}^2}{3}\left(\partial_r^2 + \frac{1}{r}\partial_r - \frac{1}{r^2}\right) - \hat{h}\hat{h}_r\left(\partial_r + \frac{1}{r}\right) + \frac{1}{2}\hat{h}\left(\hat{B}_{rr} - \frac{1}{r}\hat{B}_r\right) + \hat{B}_r\hat{\eta}_r. \tag{2.12}$$

The right-hand side of (2.11) is

$$\hat{b} = \frac{g}{\alpha}\hat{\eta}_r + 2\hat{h}\left(\frac{\hat{h}}{3}\hat{\phi}_r + \hat{\phi}\left(\hat{h}_r + \frac{1}{2}\hat{B}_r\right)\right) + \frac{\hat{h}}{2}\hat{w}_r + \hat{w}\hat{\eta}_r, \tag{2.13}$$

where, analogous to (2.9), $\hat{\phi}$ and $\hat{w}$ are given by

$$\begin{aligned} \hat{\phi} &= (\hat{u}_r)^2 + \frac{1}{r}\hat{u}_r\hat{u} + \frac{1}{r^2}\hat{u}^2, \\ \hat{w} &= \hat{u}^2\hat{B}_{rr}. \end{aligned} \tag{2.14}$$

Note that far from the origin the radial waves should behave like plane waves, and dropping the terms that depend on $1/r$ from these equations reduces them to the plane wave case. This consists of the 1D SWE with a source term ψ_1 in the momentum equation that comes from solving the 1D elliptic equation $(I + \alpha\mathcal{T}_{11})\psi_1 = b_1$, where $\mathcal{T}_{11}$ and b_1 are given by (2.7) and (2.8), respectively.

3. Solution algorithm. We give a high level overview of our approach before giving the details below. Following Popinet and others, we use a splitting method to take a full time step. The elliptic equation (2.6) is discretized using a straightforward second-order finite difference scheme, and the nonlinear SWE are solved in finite volume form as implemented in GeoClaw. A time step starts by solving the elliptic equation, which updates only the momenta. Then the SWE are solved for one time step using the modified momenta and the original depth h to get to the next time level. The details of this splitting method are given in subsection 3.1. In subsection 3.2 we describe the patch-based AMR version of the splitting method. This is the major algorithmic innovation in this paper. We solve an elliptic equation one level at a time, but including all patches at the given level. We also introduce a second "provisional" solve on coarser levels at the next time step. This allows linear interpolation in time to fill in the ghost cells on finer levels, which is needed for subcycling in time. Incorporate the elliptic solution variables from each level into the state vector made this easy to implement in the existing AMR framework. In subsection 3.3 we describe a composite solve of a linear system coupling all levels, and without subcycling in time, which we use for comparison.

We use capital letters to denote the numerical solution on a single grid or the coarse grid; below we introduce lower case for the fine grid solution. To enable the subcycling in the AMR implementation, the vector of variables $\vec{\Psi} = (\Psi_1, \Psi_2)$ from the implicit solve are stored along with the conserved variables (H, HU, HV), defining $Q = (H, HU, HV, \Psi_1, \Psi_2)$. The elliptic equations for the dispersive update are solved over the entire domain except when the water depth falls below a threshold. This allows the SWE to handle the wetting and drying on land. They are also better at simulating wave breaking, which does not naturally occur in the dispersive equations. For each of the examples in section 4 we specify the undisturbed water depth used as the threshold for switching from SGN to SWE.

3.1. Splitting method for SGN. A single time step of the fractional step algorithm on a single grid proceeds as follows. We start with $(H, HU, HV)^N$ at time t^N. On a single grid there is no need to store the $\vec{\Psi}^N$ vector, so we simplify the description here by omitting it. The following steps advance the solution to $t^{N+1} = t^N + \Delta t$:

1. Solve the elliptic equations for $\vec{\Psi}^N$ in (2.6). The right-hand-side and matrix coefficients are a function of (H, HU, HV) at time t^N.

We currently use PETSc to solve the sparse system of equations. The matrix elements (in compressed sparse row form) along with the right-hand-side vector are passed to the algebraic multigrid preconditioned Krylov solver in PETSc.

Popinet [52] and others have implemented a multigrid solver and report good convergence results. Since they use a fixed spatial refinement of a factor of 2, the multigrid hierarchy fits nicely into their strategy. We often refine by larger factors, and have not gone this route.

2. Advance the momentum equations using forward Euler and the right-hand side of (2.1) to get

$$\begin{aligned} H^* &= H^N, \\ (HU)^* &= (HU)^N + \Delta t\, H^N \left(\frac{g}{\alpha}\eta_x^N - \Psi_1^N\right), \\ (HV)^* &= (HV)^N + \Delta t\, H^N \left(\frac{g}{\alpha}\eta_y^N - \Psi_2^N\right). \end{aligned} \tag{3.1}$$

In these equations η_x and η_y now stand for centered finite difference approximations to these quantities. We have also experimented with a two-stage Runge–Kutta method but found almost no difference, so it was not worth the computational expense. This is likely due to the first-order errors that are introduced with limiters and the splitting error between the shallow water and dispersive steps. Moreover the SWE already have reduced accuracy with realistic bathymetry.

3. Take a time step Δt with the SWE solver, with initial data $(H, HU, HV)^*$ to obtain the values $(H, HU, HV)^{N+1}$. We denote this update by $(H, HU, HV)^{N+1} = SW((H, HU, HV)^*, \Delta t)$.

At domain boundaries, the conditions for $\vec{\Psi}$, e.g., extrapolation or wall boundaries, are put directly into the matrix equations. For example, the equation for a cell (i, j) adjacent to the right edge of the computational domain would normally include cell $(i+1, j)$ in its stencil. To implement wall boundary conditions, we want to specify $\Psi_{1,i+1,j} = -\Psi_{1,i,j}$ since this is a correction to the momentum in the x-direction. We implement this by negating the matrix coefficient that would correspond to the missing cell, and adding it to cell (i, j)'s matrix entry. Since Ψ_2 updates the y-momentum, tangential to this edge, we want $\Psi_{2,i+1,j} = \Psi_{2,i,j}$ and the matrix coefficient is not negated before adding it to cell (i, j)'s matrix entry. At the top and bottom boundaries, wall boundary conditions are implemented similarly with the negation swapped between Ψ_1 and Ψ_2. For extrapolation boundary conditions, no components are negated. Extrapolation boundary conditions do not absorb outgoing waves as well for SGN as for SWE, where they are routinely used in GeoClaw at open ocean boundaries. The implementation of better absorbing boundary conditions is still work in progress, using a sponge layer as often used in other dispersive solvers, e.g., [37]. Boundary conditions for dispersive equations are still an open area of research [47].

The switch from SGN to SWE is made on a cell-by-cell basis. If the initial water depth for any cell in the 3-by-3 neighborhood surrounding a given cell is below a specified threshold, then the dispersive correction is not applied. This is implemented by setting this row of the linear system for $\vec{\Psi}$ to the corresponding row of the identity matrix and the right-hand side to zero, ensuring a zero correction. The cell can still be used in the stencil for the dispersive terms of a neighboring cell if appropriate. We have not observed any spurious reflections from the interface where the equation changes when using this procedure.

3.2. AMR for SGN. Now suppose we have two grid levels with refinement by a factor of 2 in time. The main changes to the algorithm involve fine grid ghost cells and a second implicit solve on the coarser levels. Figure 2 illustrates the setup, with one grid at the coarsest level, and two adjacent rectangular nonoverlapping fine grid patches at level 2. We denote the coarse grid values at some time t^N as above, using capital letters. We assume that the fine grid is at time t^N, but that on the fine grid we want to take two time steps of $\Delta t/2$ to reach time t^{N+1}. For the fine grid values at time t^N we use lowercase letters with $q^N = (h, hu, hv, \psi_1, \psi_2)^N$.

We always need boundary conditions in ghost cells around the union of fine grid patches when taking an explicit hyperbolic time step, which in general are obtained by space-time interpolation from the underlying coarser level grids. For the SGN equations we also need boundary conditions for the elliptic system of equations that is solved on the union of all grid patches at the fine level. By incorporating $\vec{\Psi}$ in Q, the same space-time interpolation operators provide these necessary ghost cell values.

In the algorithm description below, $\mathcal{I}_f(Q^N)$ denotes the spatial interpolation operator that interpolates from coarse grid values to the ghost cells of a fine grid patch at a single time t^N.

One time step on the coarse grid, coupled with two time steps on the fine grid, is accomplished by the following steps:

1. **Coarse grid step:**
   (a) Take time step Δt on the coarse grid as described above for the single grid algorithm, but denote the result by $(\widetilde{H}, \widetilde{HU}, \widetilde{HV})^{N+1}$ since these provisional values will later be updated. This step is taken on the union of all coarse level patches and does not involve fine patches.
   (b) Solve for a provisional $\widetilde{\Psi}^{N+1} = [\Psi_1, \Psi_2]$. This is the second implicit solve mentioned above. This will be needed for interpolation in time when determining ghost cell values for ψ on the fine grid using $\mathcal{I}_f(Q^N)$ and $\mathcal{I}_f(\widetilde{Q}^{N+1})$.
2. **Fine grid steps:**
   (a) Given $(h, hu, hv)^N$ and ghost cells boundary conditions $\mathcal{I}_f(Q^N)$ on the union of fine grid patches, solve the fine grid elliptic system for ψ^N.
   (b) Update using the source terms

$$(h, hu, hv)^* = (h, hu, hv)^N + \left(0, \frac{\Delta t}{2}\left(\frac{g}{\alpha}\eta_x - \psi_1\right)^N, \frac{\Delta t}{2}\left(\frac{g}{\alpha}\eta_y - \psi_2\right)^N\right).$$

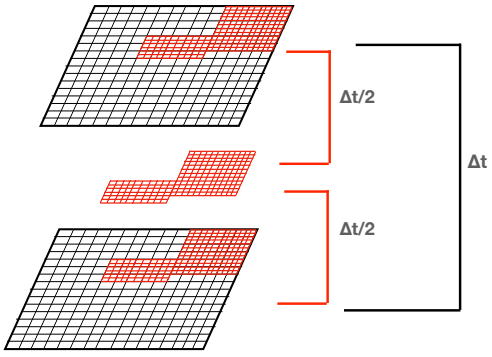

FIG. 2. *There is a coarse grid with cells outlined in black on the base grid. Over part of the domain there are two adjacent fine patches refined by a factor of two, with cells outlined in red. The fine grid time step is half the coarse grid step. Before the fine grid takes a step, two ghost cells (not shown in the figure) are needed to complete the stencil. Figure taken from* [8].

(c) Take a shallow water step:

$$(h, hu, hv)^{N+1/2} = SW((h, hu, hv)^*, \Delta t/2).$$

Note that we use $t^{N+1/2} = t^N + \Delta t/2$ to denote the intermediate time for the fine grid.

(d) Obtain ghost cell values at this intermediate time as $\frac{1}{2}(\mathcal{I}_f(Q^N) + \mathcal{I}_f(\widetilde{Q}^{N+1}))$.

(e) Solve the elliptic system for $\psi^{N+1/2}$ on the fine grid.

(f) Update using the source terms

$$(h, hu, hv)^* = (h, hu, hv)^{N+1/2} + \left(0, \frac{\Delta t}{2}\left(\frac{g}{\alpha}\eta_x - \psi_1\right)^{N+1/2}, \frac{\Delta t}{2}\left(\frac{g}{\alpha}\eta_y - \psi_2\right)^{N+1/2}\right).$$

(g) Take a shallow water step:

$$(h, hu, hv)^{N+1} = SW((h, hu, hv)^*, \Delta t/2).$$

3. **Update coarse grid:**

(a) Define $(H, HU, HV)^{N+1}$ by the provisional values $(\widetilde{H}, \widetilde{HU}, \widetilde{HV})^{N+1}$ where there is no fine grid covering a grid cell, but replacing $(\widetilde{H}, \widetilde{HU}, \widetilde{HV})^{N+1}$ by the conservative average of $(h, hu, hv)^{N+1}$ over fine grid cells that cover any coarse grid cell. In other words, the fine grid values update all underlying coarser cells to get the coarser level ready for its next step.

The final step is applied because the fine grid values $(h, hu, hv)^{N+1}$ are more accurate than the provisional coarse grid values. Note that it is not necessary to update Ψ to Ψ^{N+1} on the coarse grid because this is computed at the start of the next time step.

We then proceed to the next coarse grid time step. At the start of this step, the updated $(H, HU, HV)^{N+1}$ is used to solve for Ψ^{N+1}, and the provisional $\widetilde{\Psi}^{N+1}$ is discarded. Hence two elliptic solves are required on the coarse level each time step, rather than only one as in the single grid algorithm. Luckily this second solve is not needed on the finest grid level.

The algorithm above easly generalizes to refinement factors larger than 2. There will be additional time steps on level 2, and for each time step the ghost cell BCs will be determined by linear interpolation in time between $\mathcal{I}_f(Q^N)$ and $\mathcal{I}_f(\widetilde{Q}^{N+1})$. If there are more than 2 levels, this same idea is applied at the next level. After each time step on level 2, any level 3 grids will be advanced by the necessary number of time steps to reach the advanced time on level 2. In this case we will need two elliptic solves for every time step on level 2, one at the start of a level 2 time step, and one for the provisional values after advancing level 2, to provide interpolated ghost cell values to level 3.

When there are multiple possibly adjacent grids at the same level, as in Figure 2, the border cells of one grid might include another grid's interior cells in its stencil. Its row and column numbers need to be known to build the entries of the matrix. To enable this, grids are preprocessed at every adaptive regridding step by numbering the cells to indicate what equation in the matrix they apply to. This information is saved on each grid patch. Figure 3 illustrates this enumeration for the level 2 grids in Figure 2. For example, the stencil for cell 40 on the left grid involves cell 73 from the right grid. The preprocessing step would fill in these numbers in those ghost cells that come from adjacent grids. If there are no such adjacent grid patches, then the ghost cells would have been interpolated from the coarse grid and are treated as Dirichlet

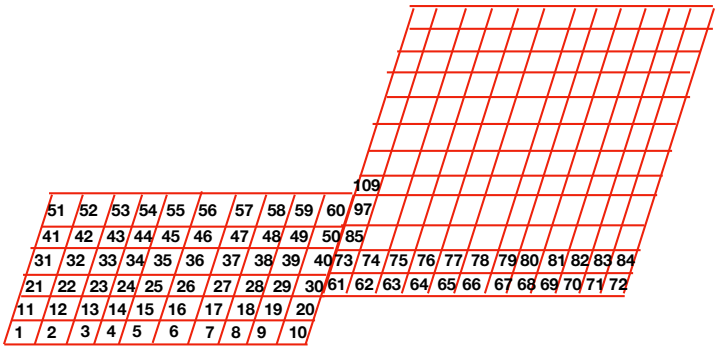

FIG. 3. *Sample enumeration of cells on level* 2 *grids. The numbers are used to map to the row in the linear system.*

conditions. They would be flagged with -1 to indicate that they have no row or column associated with them, and their values are incorporated into the right-hand side of the equations.

For adaptive calculations, the PETSc preconditioner mentioned above can be reused until the grids at that level change. The coarsest level 1 grids never change, but they are so coarse and inexpensive that reuse is not important.

The finite volume method used for SWE in GeoClaw exactly conserves mass on a Cartesian domain with no coastal or onshore points, even when AMR is used (and also near the coast on a uniform grid, but see [34] for more discussion of conservation and the issues that arise when using AMR near the coast). GeoClaw also conserves momentum when used on a flat bottom, but with any realistic topography momentum should not be conserved. The modifications we introduce to solve Boussinesq-type equations add additional source terms only to the momentum equations, and so mass is conserved as well as it is for SWE.

3.3. Composite solution algorithm. To verify the accuracy of our procedure for refinement in time, we also implemented a multilevel composite solve without subcycling. In this composite solve, the unknowns on all levels are coupled together in a single system of equations. In this approach all levels take the same time step. This global time step must then be chosen for stability on the finest level, based on the CFL condition for the explicit SWE solver. This typically means that the Courant number is much smaller than 1 on the coarser levels, requiring more coarse time steps and perhaps diminishing the accuracy due to numerical dissipation. For this reason we prefer to use subcycling in practice.

The composite solve introduces two new types of equations, pictorially illustrated in Figure 4. The first and last rows and columns just inside the border of a fine grid have an equation that says their area-weighted sum should equal the corresponding value on the underlying coarse cell. This equation is associated with the coarse cell index, and it needs to be included in the grid enumeration procedure. Note that this *hidden* coarse cell is also part of its coarse neighbor's stencil that lies outside the fine grid.

Second, all the ghost cells on the fine grid will now have equations for $\vec{\psi}$ that specify linear interpolation in space only from the coarse grid. We use a centrally differenced gradient in x and y on the coarse cell to reconstruct to the fine ghost cell. If the stencil includes a cell that lies outside the computational domain the boundary conditions are applied. Note that the fine grid ghost cell is also referenced in the stencil of the first interior fine cell mentioned above.

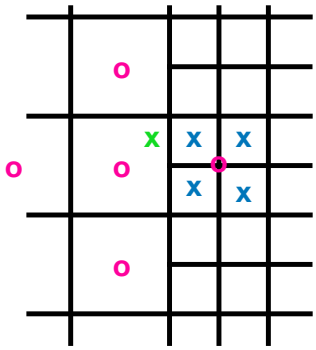

FIG. 4. *For the composite coupled solution, new unknowns and equations are introduced into the elliptic solver. The fine grid ghost cell marked by the green x is a new unknown, and its equation specifies that this value is linearly interpolated from the* 5 *coarse grid values indicated by red circles. Note that the green cell is also part of several fine grid stencils for the interior fine cells adjacent to the patch edge. There is also a new unknown on the coarse grid marked with a red circle that is underneath the fine grid. Its equation specifies that the coarse grid value is the average of the* 4 *fine grid values.*

Except for the border cells mentioned above, the rest of the coarse cells that are *underneath* a fine grid do not need to participate in the composite solve. They are marked with a flag to ignore them in the composite enumeration preprocessing step. After the solve, the coarse cells are updated by the fine grid, and the shallow water step proceeds on each level separately as in the usual patch-based AMR for hyperbolic equations, except with the same time step.

3.4. Algorithmic comparisons. We compare the composite coupled algorithm just described with the patch-based algorithm with subcycling in time described in subsection 3.2. The problem has a flat topography with water depth 4000 m. The radially symmetric initial sea surface is given by $\eta(x, y, 0) = \exp(-(r/2000)^2)$, where $r = \sqrt{x^2 + y^2}$. The initial velocity is 0 everywhere. The simulation is run to time 300 seconds. There are 5 levels of refinement, each with a factor of 2 in x and y. The composite algorithm has a single Δt based on the stability limit for the level 5 grid. The subcycled algorithm refines by a factor of 2 in time as well. We also compare with a uniformly fine solution at the same resolution as the finest level.

Figure 5 shows the results from the three approaches after 300 seconds. To the eye the leading waves are identical. The two adaptively refined simulations show quite different refinement patterns, none of which enforce symmetry, yet the solutions remain remarkably symmetric.

For completeness, we also compared the coupled algorithm with an uncoupled solution and the same fixed (small) time step. This intermediate algorithm was also more accurate than the subcycled algorithm in a few places, at the cost of increased run time due to the much smaller time step needed here, as with the coupled algorithm.

In Figure 6 we compare 4-, 5-, and 6-level adaptive solutions to show convergence, but since the adaptive runs do not refine everywhere and have larger truncation errors at patch boundaries, it is hard to determine the convergence rate. Note the "bumps" in the zoom at level 4, which are gone by levels 5 and 6. All runs show small-scale oscillations that decrease with refinement level. Discrepancies are highlighted in the zooms, but the results are clearly converging to the reference 1D radial solution. There are tiny glitches at patch interfaces that can be seen in the zoomed plots.

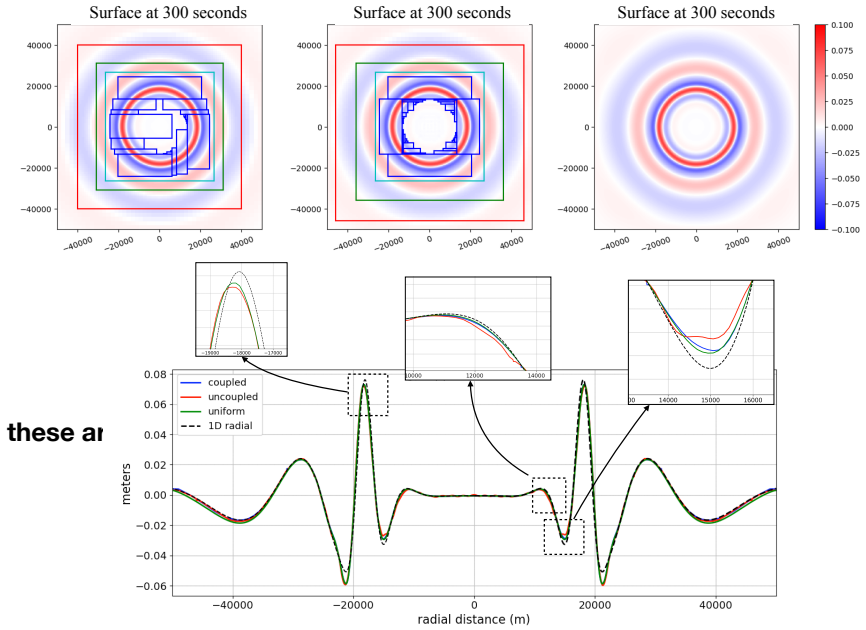

FIG. 5. *Comparison of* 5*-level coupled and uncoupled AMR results with results on a uniform grid having the same resolution as the finest AMR level. Top left is the composite coupled algorithm; middle is the uncoupled algorithm with subcycling in time; right is the equivalently refined uniform grid. The grid refinement patterns are fairly different but the solutions agree very well. Horizontal transects through the* 3 *solutions are shown in the lower plot and compared to the reference* 1*D radial solution. The zoomed sections show the very small differences in the uncoupled solution (red curve) from the others.*

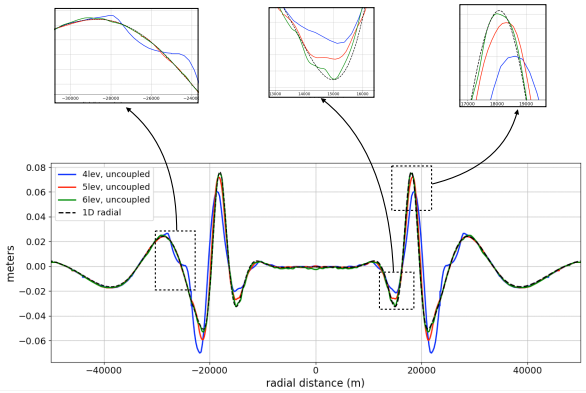

FIG. 6. *Example showing* 4*-,* 5*-, and* 6*-level adaptive runs with subcycling in time converging to the* 1*D radial reference solution.*

4. Computational examples. In this section we present computational results for three different test problems that exercise the new algorithms and software. We have not tried to optimize the runs in a high-performance computing environment, but we do provide timings to demonstrate that it is possible to solve significant problems on a laptop. In all cases the code was run on a MacBook Pro with the Apple M1 chip using six threads for openMP and six MPI processes. The linear systems are solved

using PETSc with a Krylov method and algebraic multigrid as a preconditioner. We reuse the preconditioner for multiple time steps, which helps reduce the run time. We changed the default convergence criteria from a relative tolerance of 10^{-5} to 10^{-9}. Other PETSc parameters can be found in the code repository for this paper [13], along with the source code for all three of these examples.

4.1. Radially symmetric ocean, shelf, and beach. We test both the accuracy and the stability of our AMR implementation for a 2D problem where varying topography, dispersion, and nonlinearity are all important. We construct a radially symmetric "ocean" bounded by a continental shelf followed by a beach. The radial profile of the topography is shown in Figure 7. The ocean has a flat bottom with 3000 m depth out to radius of 40 km, a continental slope from there to 80 km, followed by a flat shelf with depth 100 m. Starting at 100 km a beach with slope 1:200 starts to rise and the initial shoreline is at a radius of 120 km. As initial data we take a stationary Gaussian hump of water with surface elevation

$$\eta(x, y, 0) = 20 \exp(-(r/10000)^2) \tag{4.1}$$

so that the amplitude is 20 m and it decays over roughly 10 km. We use the 1D radially symmetric SGN equations (2.10) to compute a reference solution on this topography. The 1D version of GeoClaw used for this does not support AMR but uses a finite volume grid with variable spacing chosen so that the Courant number is close to 1 everywhere that the depth is greater than 50 m, transitioning to equal grid spacing in shallower water and onshore. A total of 10,000 grid cells were used from $r = 0$ to 124 km, giving a grid spacing that varied from roughly 80 m in the deep ocean to 5 m near shore. The standard GeoClaw wetting-and-drying algorithm is used on shore. With the initial conditions used here, the inundation proceeds onshore about 1.2 km, with a runup elevation of roughly 6 m. The SGN equations are used in water deeper than 5 m, switching to SWE in shallower water and onshore for the inundation modeling.

For the 2D simulation we used topography obtained by rotating the 1D profile. Five levels of refinement were used with mesh spacing ranging from 2 km on level 1 to 5 m on level 5. The refinement ratios going from coarsest level 1 to finest level 5 were 4,5,10,2. The problem was solved on one quadrant of the full 2D domain, for $0 \le x, y \le 126$ km, with solid wall boundary conditions at $x = 0$ and $y = 0$. In order to test the stability and accuracy of AMR near grid interfaces, we focused the refinement on the waves propagating along the diagonal where $x = y$, which hit the shore at $x = y = \sqrt{2} \times 120 \approx 169.7$ km. Levels 3–5 were allowed around the shelf and beach region. The cells flagged for refinement were also restricted so the finest level patches were created only near the diagonal, since this is the region of interest for this comparison.

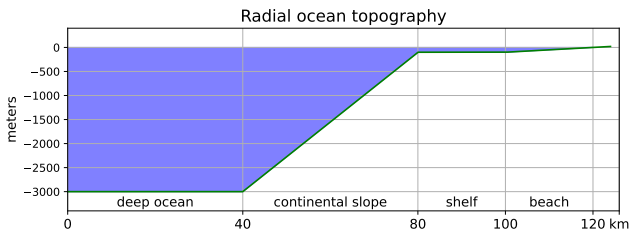

FIG. 7. *Topography for the radially symmetric ocean test case.*

Figures 8 and 9 show the computed solutions at several times, both the 2D surface elevation over the quadrant and the transect along the diagonal. The latter is plotted together with the computed solution to the 1D radial equations to assess the accuracy. Note that by $t = 600$ seconds the dispersion has created an oscillatory wave train that is well captured on level 3. By time $t = 1200$, shoaling on the continental shelf causes the wavelength to decrease, and then by $t = 1800$ the nonlinearity in the shallower water has caused the waves to steepen. Rather than breaking, the dispersion leads to "soliton fission," the appearance of solitary waves that break off from the steepening

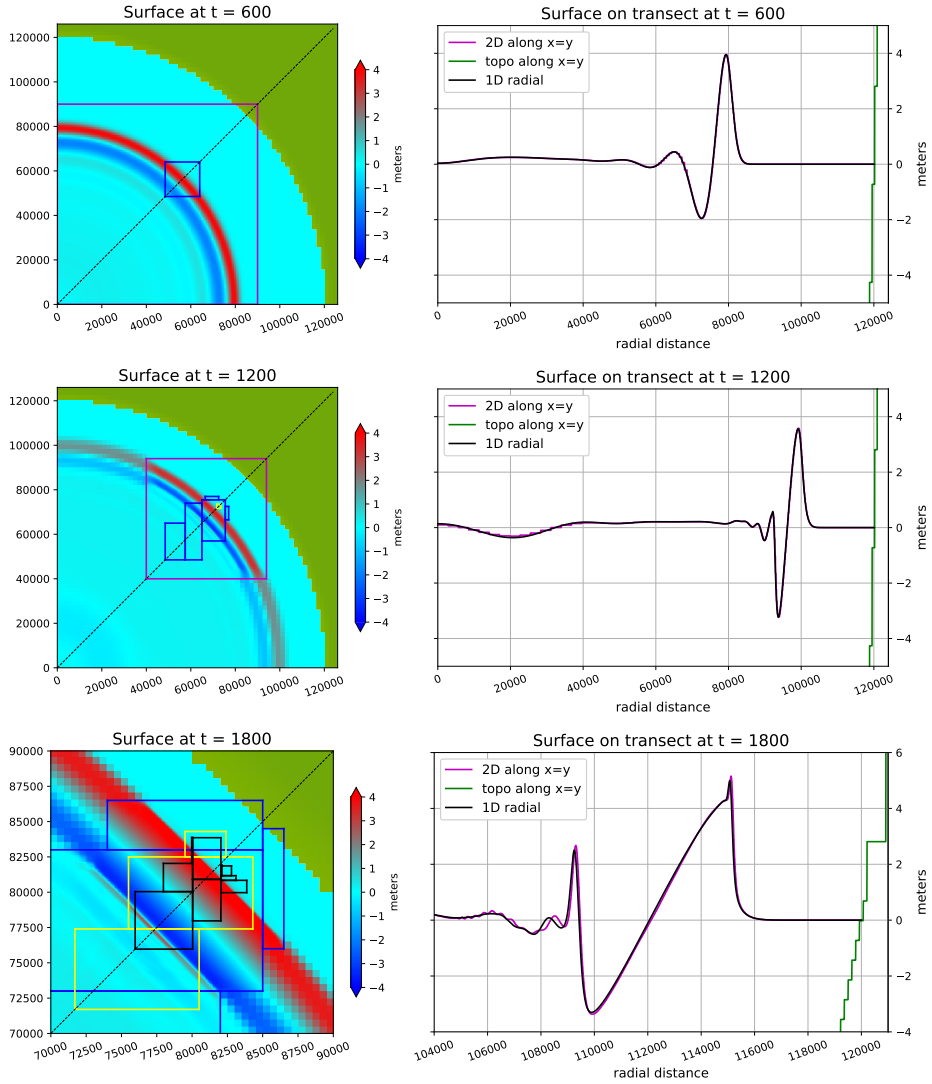

FIG. 8. *Solution to the radially symmetric ocean test problem at several times. The left column shows a colormap of surface elevation $\eta(x, y, t)$ over the positive quadrant. Colored rectangles show grid patches (level 2: magenta; level 3: blue; level 4: yellow; level 5: black) and the dashed line is the $x = y$ diagonal. The right column shows a transect of the 2D solution along the diagonal at each time, along with the solution to the 1D radially symmetric equations at the same time. Plots are zoomed in near the shore at time $t = 1800$ seconds.*

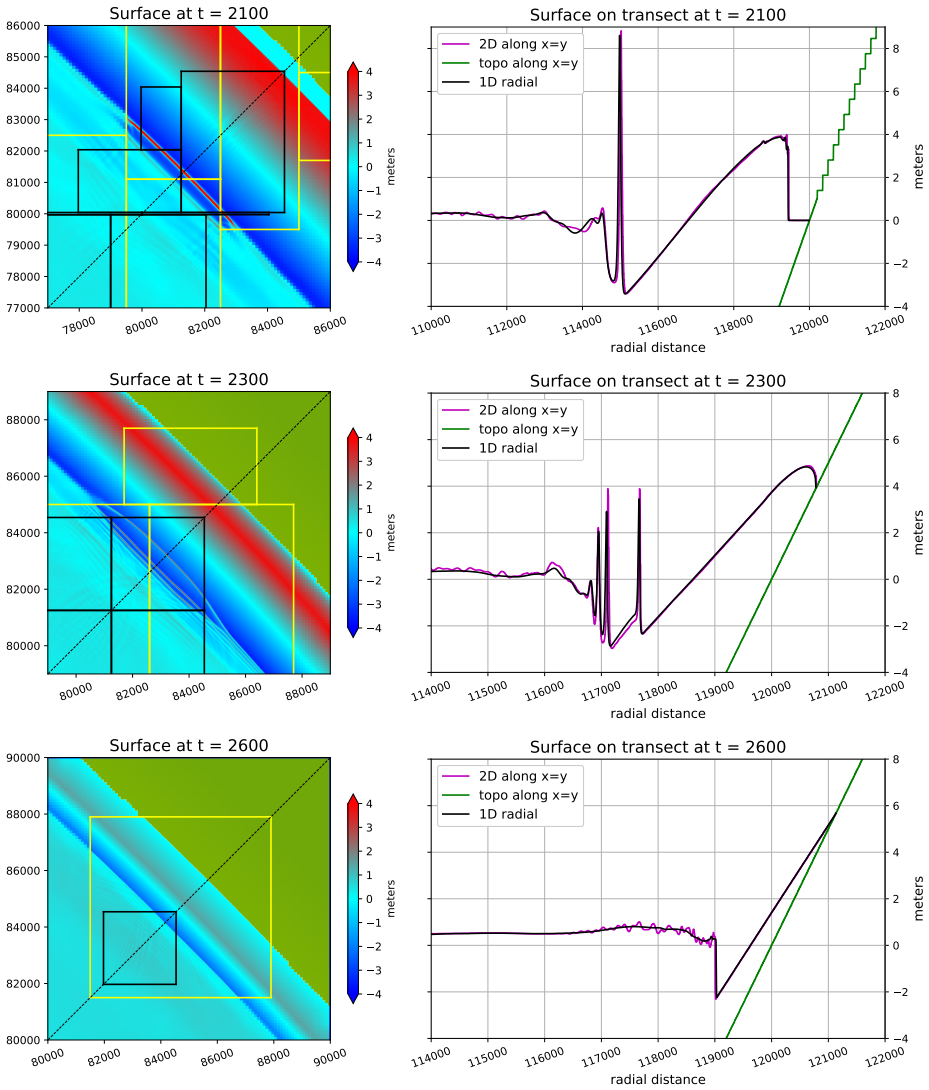

FIG. 9. *Continuation of Figure* 8 *at later times (zoomed in near the shore). Note that the axes change in each plot to better show the signficant features at each time.*

main waves. This phenomenon is frequently observed as real tsunamis approach shore. At $t = 1800$ a soliton is just forming near the main peak. These waves are well captured. At $t = 2100$, the leading wave is in water with depth less than 5 m, where the transition from SGN to SWE occurs, and a shock (hydraulic jump) has formed. In deeper water, at roughly $r = 115$ km, a sharply peaked soliton has formed. Note that this wave is still 100 m wide and less than 10 m high, so it is reasonable that this wave has not yet broken. Also note that the 2D solution agrees very well with the 1D solution in this region where the grid is refined to a similar resolution, in spite of the fact that the 2D wave is moving diagonal to the Cartesian grid here. At $t = 2300$ the wave is moving onshore and has inundated about 800 m inland. The agreement with the 1D radial solution is excellent onshore and quite good in the region of soliton

fission. Finally, at $t = 2600$ the wave has roughly reached the level of maximum runup and is starting to retreat.

In the 2D plots the accuracy decreases dramatically outside of the finer grid patches. Inside the fine grid patches, very small oscillations can be seen radiating from the patch boundaries that also appear in the transect plots, but these oscillations have little effect on the primary waves in the region of interest.

The simulation up to $t = 2300$ required about 6 hours (wall time) on a laptop as described at the beginning of section 4. (For better visibility of both the underlying solution and the grid patches in Figures 8 and 9, we allowed large grid patches, up to 1000×1000 instead of 100×100, reducing the effectiveness of using OpenMP for the explicit shallow water time-stepping and slightly increasing the run time.) A simulation using the SWE with the same refinement levels and strategy required about 30 minutes of wall time. However, the SWE solution (not shown) does not contain the narrow solitons seen in the SGN solution, and so it would not require the same level of refinement to obtain an accurate SWE solution for this problem, potentially further reducing the time required for SWE.

4.2. Japan 2011 earthquake tsunami. The great east Japan (Tohoku) earthquake of March 11, 2011, was studied by Popinet [51] using the SWE. He further used it as a test problem for the dispersive SGN equations in [52], providing some comparisons between the two sets of equations. We present a similar comparison for our AMR implementation of the same version of the SGN equations so that we can compare with his figures. Because the primary tsunami wavelengths generated by this megathrust event were so long, the addition of dispersive terms has relatively little effect on the solution for the most part. This is generally true when modeling tsunamis generated by large subduction zone earthquakes, the source of most devastating historical tsunamis, and the use of the SWE without dispersive terms is often well justified [21, 31]. However, there are some higher frequency waves introduced by the dispersion that match some of the observations slightly better. Below we also use this example to investigate a conjecture made in [3] concerning the arrival times of the peak waves.

This problem is also a good test of the stability of our AMR implementation of the SGN equations on real topography from transoceanic to harbor scale, with realistic refinement patterns and running conditions similar to those often used in practice. Six levels of adaptive refinement were used for the simulations presented below, starting with a level 1 grid having 2 degree resolution (220 km) and using refinement ratios 5,6,4,6,30 at successive levels to reach 1/3 arcsecond (10 m) resolution in Kahului Harbor on the island of Maui, Hawaii. In the deep ocean at most level 4 was used (1 arcminute), with frequent regridding to follow the leading wave traveling to Hawaii without overresolving other regions. We used identical running conditions as in [3] with one exception: there the deep ocean was refined only to level 3 (4 arcminutes) but better resolution is required to model the more oscillatory dispersive waves, and even to capture the SWE waves properly close to the source. In these new computations, 4 levels were allowed in the deep ocean for both the SGN and SWE simulations.

Several different reconstructions of the tsunami source (seafloor motion) from the 2011 earthquake have been developed by different groups. Popinet used a model developed by the UCSB group of Shao et al. [55]. For comparison we present some results with this UCSB model. Then we also present some results using a source model developed by Fujii et al. [17], which was found in [41] to be one of the best at replicating nearfield runup and DART buoy time series in a comparison of GeoClaw results for 10 proposed sources for this event (which also included the UCSB source).

Figure 10 shows the tsunami generated by the UCSB source at 3 hours, as computed with both SWE and SGN. Differences are relatively minor, but the main outgoing wave is seen to have some dispersive ripples in the SGN calculation that are not present with SWE. The differences are more clearly seen when a transect of the surface elevation at a fixed latitude is plotted; see Figure 11. This is similar to Figure 15 in [52]. It should also be noted that the Basilisk [6] model used by Popinet does cell-by-cell refinement on a quadtree structure, and that he refined in somewhat different regions than in our patch-based GeoClaw simulations, so direct comparison is not entirely possible.

Figure 10 also shows the location of two DART sensors that recorded the tsunami passing by. Figure 12 shows the detided DART data observed at these locations (from the data archived with [41]), together with the time series computed with both the SWE and SGN versions of GeoClaw. Here we show results using the Fujii source, which in general matches the DART observations better. As expected, dispersion makes relatively little difference for these long wavelength waves, but does add some oscillations at roughly the same period as seen in the observed data, although the latter is at a sampling rate of 60 seconds and is not well resolved at shorter wavelengths.

The Fujii tsunami source of [17] was used by Arcos and LeVeque [3] to model tide gauge observations and tsunami current speeds at a number of points in Hawaii.

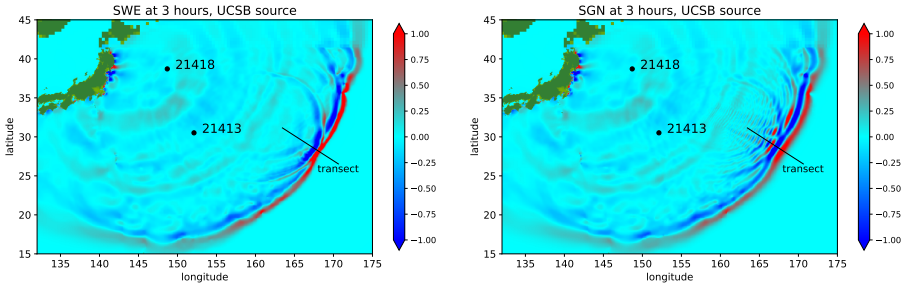

FIG. 10. *Simulations of the* 2011 *Tohoku tsunami* 3 *hours after the earthquake, using the UCSB source model, using SWE (left) or SGN (right). Colors saturate at* ± 0.5 *m relative to sea level in order to accentuate the small oscillatory waves following the main peak in the dispersive SGN simulation. The DART gauge locations are also indicated, for which time series data is plotted in Figure* 12. *Compare this plot to the bottom row of Figures* 12 *and* 13 *in Popinet* [52]. *The black lines show the transect along which the solution is plotted in Figure* 11.

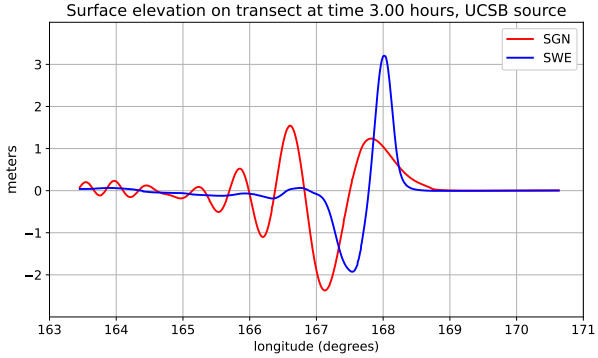

FIG. 11. *Cross section of the solution from Figure* 10 *at* 3 *hours, along the transect shown in that figure, from* $(163.454E, 31.1458N)$ *to* $(170.631E, 26.5282N)$ *as provided by Popinet* [50]. *Compare this plot to Figure* 15 *in Popinet* [52].

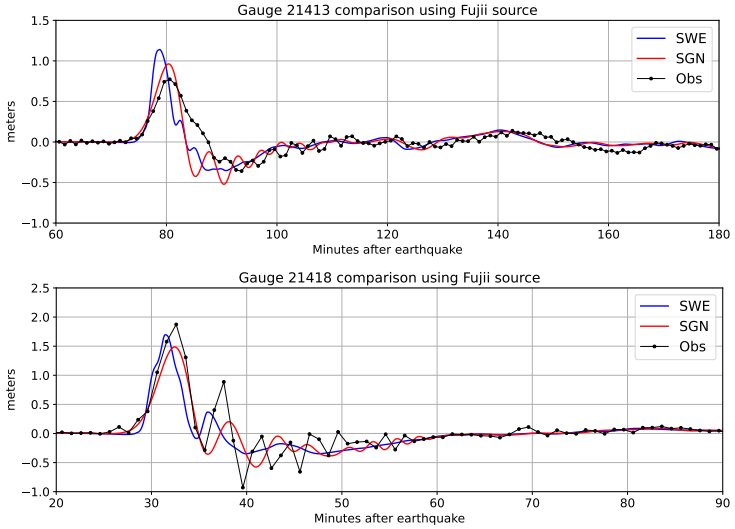

FIG. 12. *DART gauge time series at two locations marked in Figure* 10. *Detided observations (Obs) from March* 11, 2001, *are compared with SWE and SGN computations using the Fujii tsunami source. The SGN results show additional oscillations that appear to match the period but not phase of the underresolved observations.*

In that paper the SWE were used, and many of the computed time series agreed remarkably well with detided observations after shifting the results in time by about 10 minutes. Several possible reasons for this time shift were discussed in [3], including the fact that an instantaneous seafloor displacement was used in the model. The possible effect of dispersion on the time shift was roughly estimated using the average ocean depth and wavelength of the leading wave. It was estimated that the arrival time might be shifted by dispersion by about 1%, less than 5 minutes in Hawaii, where the first wave arrives roughly 460 minutes after the earthquake.

Figure 13 shows the detided observation at one tide gauge located in Kahului Harbor (Maui), along with the computed time series using both the SWE and SGN equations, with no time shift. The top figure shows the computational results obtained with the UCSB source, while the bottom figure shows those obtained with the Fujii source, for comparison to [3]. In the Fujii plot, the computed waveforms look nearly identical except for a small time shift. With SWE, the first maximum tsunami amplitude occurs about 8 minutes early compared to the observation. Using SGN, the peak appears roughly 3 minutes later than with SWE, so only 5 minutes early relative to the observations. This is consistent with other findings in the literature that even with the use of dispersive equations, numerical simulations show waves arriving about 5 minutes earlier than the observations in Hawaii for this event (e.g., [31, 14]). The time series computed with the UCSB source show similar shifting of the first peak, by about 2 minutes. We also note that the differences observed between the SWE and SGN results in each case are small relative to the differences between using the two different source models, neither of which is an exact representation of the actual earthquake.

These calculations were performed on a laptop as described at the beginning of section 4. Using the UCSB source, the shallow water run to 12 hours of simulated time took roughly 23 minutes of wall time, while the SGN run required 145 minutes.

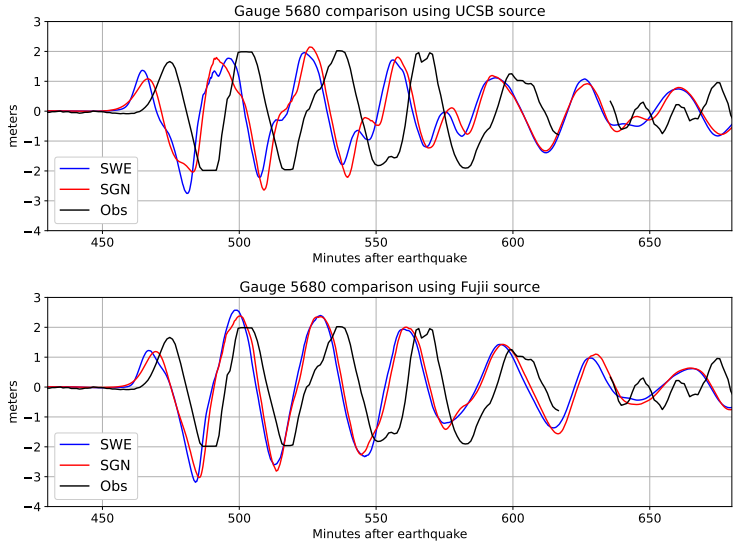

FIG. 13. *Tide gauge time series in Kahului Harbor. Detided observations (Obs) are compared with SWE and SGN computations using the UCSB (top) and Fujii (bottom) tsunami sources. Note that the physical tide gauge bottoms out at* -2 *m while the synthetic gauge is in slightly deeper water, and some observational data is missing after* 600 *minutes.*

4.3. Hypothetical asteroid impact. Next we simulate the tsunami that results from a hypothetical asteroid impact and subsequent crater in the Pacific Ocean off the coast of Washington. The paper [8] contains more details and sample simulation results for this example using the MS equations. Here we use the SGN equations and we will compare to results in [8], since there is no "data" for this experiment. A similar test problem was also used in [12] for a simulation using the nondispersive SWE for a much smaller asteroid. That paper also highlighted the need for dispersive simulations to properly model propagation and possibly runup of asteroid-generated tsunamis.

We use the same problem setup as in [8]. The crater is 1 km deep with a diameter of 3 km, as depicted in Figure 14(a). The crater was placed approximately 150 km west of Grays Harbor, a well-studied area due to the potential for Cascadia subduction zone earthquakes. The initial tsunami surface elevation, shown in Figure 14(b), is taken after 251 seconds of a radially symmetric hydrocode simulation starting with a static crater. The velocity is initialized using an eigenvector from the linearized SWE to give an approximately radially outgoing wave. The simulation uses 7 levels of mesh refinement with refinement factors from coarsest to finest of 5,3,2,2,5,6. Initially there are 5 levels, as shown in Figure 14(c). The last 2 levels are added as the wave shoals on the continental shelf and then approaches shore. The coarsest level has $\Delta y = 10$ arcminutes, and the level-7 grids have $\Delta y = 1/3$ arcsecond. On each grid $\Delta x = 1.5\Delta y$ so that the finest level computational grids are at a resolution of roughly 10 m in both x and y. The equations switch from SGN to SWE where the initial water depth is less than 10 m, regardless of wave height. (We observed no difference in inundation when using 5 or 2 m for the threshold.)

The wave height after 20 and 30 minutes is shown in Figure 15. Both compare extremely well with the comparable figures in [8]. There are more grid patches in this calculation than that paper, since a smaller maximum patch size was specified to allow for higher parallelism. Even the region with soliton fission, shown in Figure 16

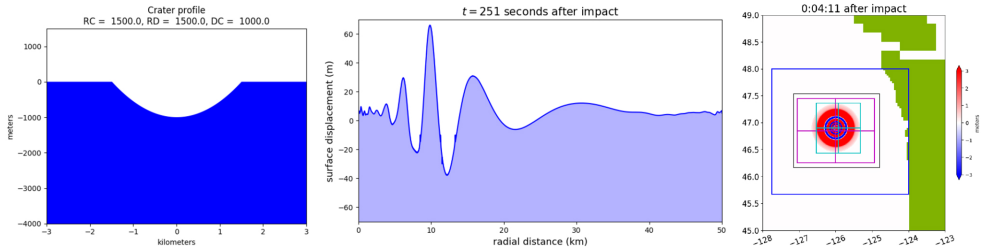

FIG. 14. (a) *Initial conditions of a static crater of depth* 1 *km and diameter* 3 *km, which were the initial conditions for the hydrocode simulation.* (b) *The radially symmetric results of the hydrocode simulation at* 251 *seconds, used to initialize the GeoClaw Boussinesq simulation.* (c) *Radially symmetric GeoClaw initial conditions and mesh, with the initial grid patches outlined.*

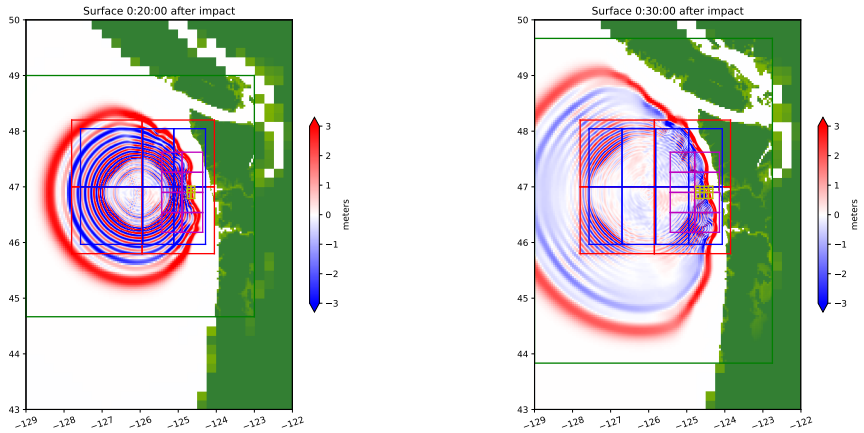

FIG. 15. *Surface elevation computed using SGN at the indicated times. The outlines of the rectangles show the grid patch boundaries, and are color-coded by level of refinement as described in the text.*

as the waves approach shore, are amazingly similar, so we also show transects of the solution at a fixed latitude to see the differences. Since the nonlinear and dispersive terms in SGN and MS differ, one might have expected more difference in this area. The refinement to finer levels at later times is focused on Grays Harbor, and the plots show that the two peninsulas at the entrance to the harbor (containing the communities of Ocean Shores and Westport, Washington) are completely inundated. In both figures the patch boundaries are color-coded by level: level 2 is in blue, 3 is black, 4 is magenta, 5 is cyan, 6 is black, and 7 is green.

The simulation up to 50 minutes required about 30 minutes (wall time) on a laptop as described at the beginning of section 4. A simulation using the SWE with the same refinement levels and strategy (not shown) required about 4 minutes.

5. Conclusions and future work. We have developed a high-fidelity tool for solving depth-averaged SGN equations, a form of Boussinesq equations that can be used to accurately model short wavelength tsunamis or other related dispersive wave problems. Building on the open source GeoClaw software, patch-based mesh refinement in space and time is implemented for solving realistic ocean-scale problems. The major new algorithmic advance is the introduction of an implicit solver in the AMR code to solve an elliptic equation on all grid patches at a given level of refinement. The elliptic equation is solved before taking each shallow water time step. It must also be

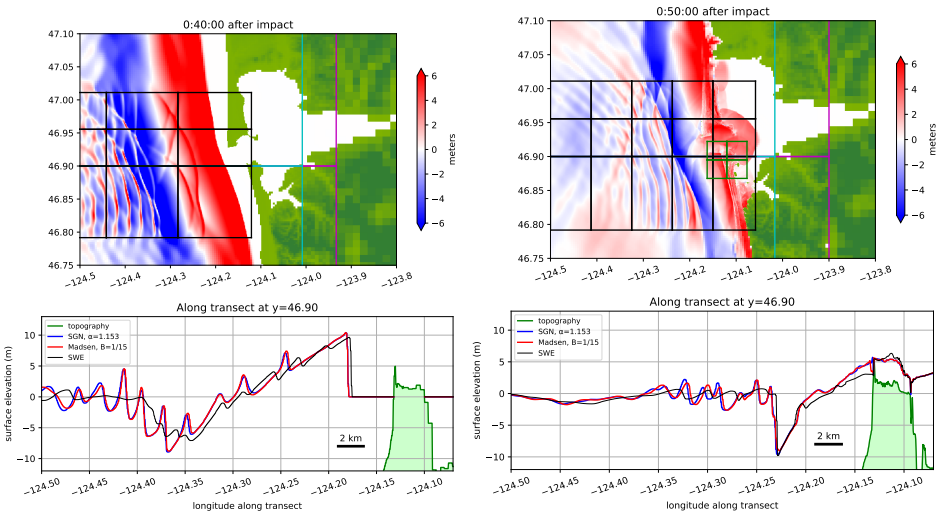

FIG. 16. *Top: Surface elevation computed using SGN at the indicated times. The outlines of the rectangles show the grid patch boundaries, and are color-coded by level of refinement as described in the text. Bottom: Surface elevation along a transect at latitude* 46.90 *cutting through Westport, Washington, highlighting the very small differences between the MS and SGN solutions. The solution computed with SWE is also shown for comparison and does not exhibit the solitary wave formation seen in the dispersive simulations. Note that at* 50 *minutes the Westport peninsula is refined to one higher level, so the topography is finer in the transect.*

solved provisionally at the end of each time step on all but the finest AMR level in order to provide values for space-time interpolation to ghost cells needed to solve the elliptic equation on the finer grid levels, in exactly the same way that ghost cell values for the depth and momentum are interpolated. This approach allows subcycling in time, so that the equations are advanced on each level using a time step based on the CFL restriction of the SWE, allowing optimal efficiency and minimal numerical dissipation. By using PETSc for solving the elliptic equations, in conjunction with OpenMP for the explicit hyperbolic steps, an efficient code has been developed in the GeoClaw framework. In the near future this will be merged into the GeoClaw code base and better documented for general use by the scientific community. We also plan to apply this software to several modeling problems, in particular for the NASA Asteroid Threat Assessment Project that originally motivated and partially funded this work.

In future research, we plan to better explore the transition between SGN and SWE in the nearshore region in order to better model breaking waves, which can have an impact on the resulting onshore inundation. We would also like to reduce the generation of reflections at interfaces between different refinement levels. Although these reflections are generally minor, as seen in the examples we have presented, they do tend to be larger than those seen when solving SWE. Better interpolation procedures may be required for the SGN equations since they involve higher order spatial derivatives. In addition it may be beneficial to increase the accuracy of the finite difference discretization of the dispersive terms.

We also plan to compare the Boussinesq approach to modeling dispersion, which requires implicit algorithms, with hyperbolic formulations (e.g., [7, 22]) that model dispersion by introducing relaxation source terms. This requires somewhat reduced time steps relative to SWE, but it has been found that these equations can still be solved efficiently with explicit methods, unlike SGN. The algorithms presented in this

paper and the resulting GeoClaw code can provide useful reference solutions for the further exploration of hyperbolic approaches.

Code and data availability. The GeoClaw setup for the examples presented in section 4 can be found in a GitHub repository created for this paper, https://github.com/rjleveque/ImplicitAMR-paper, and is permanently archived on Zenodo [13]. This includes standard Clawpack-style `setrun.py` files that show the AMR resolutions, refinement regions, and other parameters used in these examples. The software described in this paper and used for the experiments presented here has recently been merged into GeoClaw and is part of release v5.10.0 [15].

Reproducibility of computational results. This paper has been awarded the "SIAM Reproducibility Badge: Code and data available" as a recognition that the authors have followed reproducibility principles valued by SISC and the scientific computing community. Code and data that allow readers to reproduce the results in this paper are available at https://github.com/rjleveque/ImplicitAMR-paper and in the supplementary materials (ImplicitAMR-paper.zip [local/web 174KB]).

Acknowledgments. Barry Smith was instrumental in building the new version of PETSc (now included in v3.20) that allows OpenMP code in combination with MPI, and assisting us in its use. We had helpful discussions with Donna Calhoun on the implementation of the SGN equations. Stéphane Popinet provided information about his simulations in [52] and about his Basilisk code.

REFERENCES

[1] ADCIRC Developers, *ADCIRC Software*, adcirc.org.

[2] A. Almgren, J. B. Bell, P. Colella, L. Howell, and M. Welcome, *A conservative adaptive projection method for the variable density incompressible Navier-Stokes equations*, J. Comput. Phys., 142 (1998), pp. 1–46.

[3] M. E. M. Arcos and R. J. LeVeque, *Validating velocities in the GeoClaw tsunami model using observations near Hawaii from the* 2011 *Tohoku tsunami*, Pure Appl. Geophys., 172 (2015), pp. 849–867, https://doi.org/10.1007/s00024-014-0980-y.

[4] T. Baba, N. Takahashi, Y. Kaneda, K. Ando, D. Matsuoka, and T. Kato, *Parallel implementation of dispersive tsunami wave modeling with a nesting algorithm for the* 2011 *Tohoku tsunami*, Pure Appl. Geophys., 172 (2015), pp. 3455–3472, https://doi.org/10.1007/s00024-015-1049-2.

[5] S. Balay, W. Gropp, L. C. McInnes, and B. F. Smith, *PETSc, Portable, Extensible Toolkit for Scientific Computation*, Argonne National Laboratory, 1998.

[6] *Basilisk Software*, http://basilisk.fr/, 2023.

[7] C. Bassi, L. Bonaventura, S. Busto, and M. Dumbser, *A hyperbolic reformulation of the Serre-Green-Naghdi model for general bottom topographies*, Comput. & Fluids, 212 (2020), 104716, https://doi.org/10.1016/j.compfluid.2020.104716.

[8] M. Berger and R. LeVeque, *Towards adaptive simulations of dispersive tsunami propagation from an asteroid impact*, in Proceedings of the International Congress of Mathematicians, 2022.

[9] M. J. Berger and P. Colella, *Local adaptive mesh refinement for shock hydrodynamics*, J. Comput. Phys., 82 (1989), pp. 64–84.

[10] M. J. Berger, D. L. George, R. J. LeVeque, and K. T. Mandli, *The GeoClaw software for depth-averaged flows with adaptive refinement*, Adv. Water Resources, 34 (2011), pp. 1195–1206, https://doi.org/10.1016/j.advwatres.2011.02.016.

[11] M. J. Berger and R. J. LeVeque, *Adaptive mesh refinement using wave-propagation algorithms for hyperbolic systems*, SIAM J. Numer. Anal., 35 (1998), pp. 2298–2316.

[12] M. J. Berger and R. J. LeVeque, *Modeling Issues in Asteroid-Generated Tsunamis*, NASA Technical Memorandum NASA/CR-2018-219786, ARC-E-DAA-TN53167, 2018, http://hdl.handle.net/2060/20180006617.

[13] M. J. Berger and R. J. LeVeque, *rjleveque/ImplicitAMR-paper*, Zenodo, https://doi.org/10.5281/zenodo.12709451.

[14] K. F. Cheung, Y. Bai, and Y. Yamazaki, *Surges around the Hawaiian Islands from the* 2011 *Tohoku Tsunami*, J. Geophys. Res. Oceans, 118 (2013), pp. 5703–5719, https://doi.org/10.1002/jgrc.20413.

[15] Clawpack Development Team, *Clawpack Software Version* 5.10.0, 2024, http://www.clawpack.org, https://doi.org/10.17605/osf.io/kmw6h.

[16] *COULWAVE: Cornell University Long and Intermediate Wave Modeling Package*, http://isec.nacse.org/models/.

[17] Y. Fujii, K. Satake, S. Sakai, M. Shinohara, and T. Kanazawa, *Tsunami source of the* 2011 *off the Pacific coast of Tohoku earthquake*, Earth Planets Space, 63 (2011), pp. 815–820.

[18] *FUNWAVE-TVD*, https://fengyanshi.github.io/build/html/index.html.

[19] *GeoClaw*, http://www.geoclaw.org.

[20] D. L. George, *Augmented Riemann solvers for the shallow water equations over variable topography with steady states and inundation*, J. Comput. Phys., 227 (2008), pp. 3089–3113, https://doi.org/10.1016/j.jcp.2007.10.027.

[21] S. Glimsdal, G. K. Pedersen, C. B. Harbitz, and F. Løvholt, *Dispersion of tsunamis: Does it really matter?*, Nat. Hazards Earth Syst. Sci., 13 (2013), pp. 1507–1526, https://doi.org/10.5194/nhess-13-1507-2013.

[22] J.-L. Guermond, C. Kees, B. Popov, and E. Tovar, *Hyperbolic relaxation technique for solving the dispersive Serre-Green-Naghdi equations with topography*, J. Comput. Phys., 450 (2022), 110809, https://doi.org/10.1016/j.jcp.2021.110809.

[23] S. S. Gylfadóttir, J. Kim, J. K. Helgason, S. Brynjólfsson, A. Höskuldsson, T. Jóhannesson, C. Bonnevie Harbitz, and F. Løvholt, *The* 2014 *Lake Askja rockslide-induced tsunami: Optimization of numerical tsunami model using observed data*, J. Geophys. Res. Oceans, 122 (2017), pp. 4110–4122, https://doi.org/10.1002/2016JC012496.

[24] R. D. Hornung and S. R. Kohn, *Managing application complexity in the samrai object-oriented framework*, Concurr. Comput. Pract. Exp., 14 (2002), pp. 347–368.

[25] M. Kazolea and M. Ricchiuto, *On wave breaking for Boussinesq-type models*, Ocean Model., 123 (2018), pp. 16–39, https://doi.org/10.1016/j.ocemod.2018.01.003.

[26] J. Kim, *Finite Volume Methods for Tsunamis Generated by Submarine Landslides*, Ph.D. thesis, University of Washington, 2014, http://hdl.handle.net/1773/25374.

[27] J. Kim, F. Løvholt, D. Issler, and C. F. Forsberg, *Landslide material control on tsunami genesis—The Storegga slide and tsunami (*8,100 *years BP)*, J. Geophys. Res. Oceans, 124 (2019), pp. 3607–3627, https://doi.org/10.1029/2018JC014893.

[28] J. Kim, G. K. Pedersen, F. Løvholt, and R. J. LeVeque, *A Boussinesq type extension of the GeoClaw model—A study of wave breaking phenomena applying dispersive long wave models*, Coast. Eng., 122 (2017), pp. 75–86, https://doi.org/10.1016/j.coastaleng.2017.01.005.

[29] J. Kirby, N. Pophet, F. Shi, and S. Grilli, *Basin scale tsunami propagation modeling using Boussinesq models: Parallel implementation in spherical coordinates*, in Proceedings of the WCCE-ECCE-TCCE Joint Conference on Earthquake and Tsunami, 2009.

[30] J. T. Kirby, S. T. Grilli, J. Horrillo, P. L.-F. Liu, D. Nicolsky, S. Abadie, B. Ataie-Ashtiani, M. J. Castro, L. Clous, C. Escalante, et al., *Validation and intercomparison of models for landslide tsunami generation*, Ocean Model., 170 (2022), 101943.

[31] J. T. Kirby, F. Shi, B. Tehranirad, J. C. Harris, and S. T. Grilli, *Dispersive tsunami waves in the ocean: Model equations and sensitivity to dispersion and Coriolis effects*, Ocean Model., 62 (2013), pp. 39–55, https://doi.org/10.1016/j.ocemod.2012.11.009.

[32] D. G. Korycansky and P. J. Lynett, *Offshore breaking of impact tsunami: The Van Dorn effect revisited*, Geophys. Res. Lett., 32 (2005), https://doi.org/10.1029/2004GL021918.

[33] D. G. Korycansky and P. J. Lynett, *Run-up from impact tsunami*, Geophys. J. Int., 170 (2007), pp. 1076–1088, https://doi.org/10.1111/j.1365-246X.2007.03531.x.

[34] R. J. LeVeque, D. L. George, and M. J. Berger, *Tsunami modelling with adaptively refined finite volume methods*, Acta Numer., 20 (2011), pp. 211–289, https://doi.org/10.1017/S0962492911000043.

[35] Z. Liu, J.-S. L'Heureux, S. Glimsdal, and S. Lacasse, *Modelling of mobility of Rissa landslide and following tsunami*, Comput. Geotech., 140 (2021), 104388, https://doi.org/10.1016/j.compgeo.2021.104388.

[36] F. Løvholt, S. Bondevik, J. S. Laberg, J. Kim, and N. Boylan, *Some giant submarine landslides do not produce large tsunamis*, Geophys. Res. Lett., 44 (2017), pp. 8463–8472, https://doi.org/10.1002/2017GL074062.

[37] F. Løvholt, S. Glimsdal, P. Lynett, and G. Pedersen, *Simulating tsunami propagation in fjords with long-wave models*, Nat. Hazards Earth Syst. Sci., 15 (2015), pp. 657–669, https://doi.org/10.5194/nhess-15-657-2015.

[38] F. Løvholt, P. Lynett, and G. Pedersen, *Simulating run-up on steep slopes with operational Boussinesq models; capabilities, spurious effects and instabilities*, Nonlinear Process. Geophys., 20 (2013), pp. 379–395, https://doi.org/10.5194/npg-20-379-2013.

[39] F. Løvholt and G. Pedersen, *Instabilities of Boussinesq models in non-uniform depth*, Internat. J. Numer. Methods Engrg., 61 (2009), pp. 606–637, https://doi.org/10.1002/fld.1968.

[40] F. Løvholt, G. Pedersen, C. B. Harbitz, S. Glimsdal, and J. Kim, *On the characteristics of landslide tsunamis*, Philos. Trans. Roy. Soc. A, 373 (2015), 20140376.

[41] B. T. MacInnes, A. R. Gusman, R. J. LeVeque, and Y. Tanioka, *Comparison of earthquake source models for the* 2011 *Tohoku event using tsunami simulations and near field observations*, Bull. Seis. Soc. Amer., 103 (2013), pp. 1256–1274.

[42] P. A. Madsen and H. A. Schäffer, *Higher-order Boussinesq-type equations for surface gravity waves: Derivation and analysis*, Phil. Trans. Roy. Soc. A, 356 (1998), pp. 3123–3181, https://doi.org/10.1098/rsta.1998.0309.

[43] P. A. Madsen and O. R. Sørensen, *A new form of the Boussinesq equations with improved linear dispersion characteristics. Part* 2. *A slowly-varying bathymetry*, Coast. Eng., 18 (1992), pp. 183–204, http://www.sciencedirect.com/science/article/pii/037838399290019Q.

[44] K. T. Mandli, A. Ahmadia, M. Berger, D. Calhoun, D. George, Y. Hadjimichael, D. Ketcheson, G. Lemoine, and R. LeVeque, *Clawpack: Building an open source ecosystem for solving hyperbolic PDEs*, PeerJ Comput. Sci., 2 (2016), e68, https://doi.org/10.7717/peerj-cs.68.

[45] M. Matsuyama, M. Ikeno, T. Sakakiyama, and T. Takeda, *A study of tsunami wave fission in an undistorted experiment*, Pure Appl. Geophys., 164 (2007), pp. 617–631, https://doi.org/10.1007/s00024-006-0177-0.

[46] R. Meakin, *An efficient means of adaptive refinement within systems of overset grids*, in Proceedings of the 12th Computational Fluid Dynamics Conference, 1995, 1722.

[47] S. Noelle, M. Parisot, and T. Tscherpel, *A class of boundary conditions for time-discrete Green–Naghdi equations with bathymetry*, SIAM J. Numer. Anal., 60 (2022), pp. 2681–2712.

[48] G. Pedersen and F. Løvholt, *Documentation of a Global Boussinesq Solver (GloBouss)*, Matematisk Institutt, Universitetet i Oslo, 2008, https://www.duo.uio.no/handle/10852/10184.

[49] T. Plewa, T. Linde, and V. G. Weirs, eds., *Adaptive Mesh Refinement: Theory and Applications*, Springer, New York, 2005, https://doi.org/10.1007/b138538.

[50] S. Popinet, *private communication*, July, 2023.

[51] S. Popinet, *Adaptive modelling of long-distance wave propagation and fine-scale flooding during the Tohoku tsunami*, Nat. Hazards Earth Syst. Sci., 12 (2012), pp. 1213–1227, https://doi.org/10.5194/nhess-12-1213-2012.

[52] S. Popinet, *A quadtree-adaptive multigrid solver for the Serre-Green-Naghdi equations*, J. Comput. Phys., 302 (2015), pp. 336–358.

[53] *Proceedings and Results of the* 2011 *NTHMP Model Benchmarking Workshop*, NOAA Special Report, U.S. Department of Commerce/NOAA/NTHMP, 2011, https://purl.fdlp.gov/GPO/gpo44987.

[54] L. Schambach, S. T. Grilli, J. T. Kirby, and F. Shi, *Landslide tsunami hazard along the upper US east coast: Effects of slide deformation, bottom friction, and frequency dispersion*, Pure Appl. Geophys., 176 (2019), pp. 3059–3098.

[55] G. Shao, X. Li, C. Ji, and T. Maeda, *Focal mechanism and slip history of the* 2011 *MW* 9.1 *off the Pacific coast of Tohoku Earthquake, constrained with teleseismic body and surface waves*, Earth Planet Space, 63 (2011), pp. 559–564, https://doi.org/10.5047/eps.2011.06.028.

[56] M. Tissier, P. Bonneton, F. Marche, F. Chazel, and D. Lannes, *A new approach to handle wave breaking in fully non-linear Boussinesq models*, Coast. Eng., 67 (2012), pp. 54–66, https://doi.org/10.1016/j.coastaleng.2012.04.004.

[57] S. N. Ward and E. Asphaug, *Asteroid impact tsunami: A probabilistic hazard assessment*, Icarus, 145 (2000), pp. 64–78, https://doi.org/doi:10.1006/icar.1999.6336.

[58] Y. Zeng, H. Liu, Q. Gao, A. Almgren, A. P. S. Bhalla, and L. Shen, *A consistent adaptive level set framework for incompressible two-phase flows with high density ratios and high Reynolds numbers*, J. Comput. Phys., 478 (2023), 111971.

[59] H. Zhou, C. Moore, Y. Wei, and V. Titov, *A nested-grid Boussinesq-type approach to modelling dispersive propagation and runup of landslide-generated tsunamis*, Nat. Hazards Earth Syst. Sci., 11 (2011), pp. 2677–2697.