SOMAR-LES: A framework for multi-scale modeling of turbulent stratified oceanic flows

Vamsi K Chalamalla

University of North Carolina, Chapel Hill, NC 27599, USA

Edward Santilli

Philadelphia University, Philadelphia, PA 19144, USA

Alberto Scotti

University of North Carolina, Chapel Hill, NC 27599, USA

Masoud Jalali

University of California San Diego, La Jolla, CA 92093, USA

Sutanu Sarkar*

University of California San Diego, La Jolla, CA 92093, USA

Abstract

A new multi-scale modeling technique, SOMAR-LES, is presented in this paper. Localized grid refinement gives SOMAR (the Stratified Ocean Model with Adaptive Resolution) access to small scales of the flow which are normally inaccessible to general circulation models (GCMs). SOMAR-LES drives a LES (Large Eddy Simulation) on SOMAR's finest grids, forced with large scale forcing from the coarser grids. Three-dimensional simulations of internal tide generation, propagation and scattering are performed to demonstrate this multi-scale modeling technique. In the case of internal tide generation at a two-dimensional bathymetry, SOMAR-LES is able to balance the baroclinic energy budget and accurately model turbulence losses at only 10% of the computational cost required by a non-adaptive solver running at SOMAR-LES's fine grid resolution.

*Corresponding author
Email address: sarkar@ucsd.edu (Sutanu Sarkar)

This relative cost is significantly reduced in situations with intermittent turbulence or where the location of the turbulence is not known *a priori* because SOMAR-LES does not require persistent, global, high resolution. To illustrate this point, we consider a three-dimensional bathymetry with grids adaptively refined along the tidally generated internal waves to capture remote mixing in regions of wave focusing. The computational cost in this case is found to be nearly 25 times smaller than that of a non-adaptive solver at comparable resolution. In the final test case, we consider the scattering of a mode-1 internal wave at an isolated two-dimensional and three-dimensional topography, and we compare the results with Legg (2014) numerical experiments. We find good agreement with theoretical estimates. SOMAR-LES is less dissipative than the closure scheme employed by Legg (2014) near the bathymetry. Depending on the flow configuration and resolution employed, a reduction of more than an order of magnitude in computational costs is expected, relative to traditional existing solvers.

Keywords: Internal waves, Multi-scale modeling, LES, Turbulence and Mixing.

---

1. Introduction

The range of dynamically active spatial scales in the ocean spans 8 orders of magnitude. The computational cost of General Circulation Models (GCMs) used to study the circulation of the ocean increases rapidly with resolution. The computational cost of the advective terms increases roughly with the third power of the horizontal resolution (the number of grid points increases as the horizontal resolution squared, while the time step decreases linearly with the resolution to maintain a constant CFL). Furthermore, when the horizontal resolution becomes comparable to a suitably defined vertical scale, the hydrostatic approximation, which considerably simplifies the numerical solution of the equations solved by the GCMs, breaks down. For these reasons, GCMs applied to synoptic-scale processes do not resolve scales smaller than $O(10\,\mathrm{Km})$ (Menemen-

lis et al., 2008; Metzger et al., 2010). Physical processes occurring on smaller scales are parameterized with a variety of models. At the opposite end of the scale spectrum, turbulent processes occurring on spatial scales $O(< 10\,\mathrm{m})$ and temporal scales $O(< 1\,\mathrm{hour})$ (for example, unstable fronts (Stamper and Taylor, 2017) or the equatorial undercurrent (Pham et al., 2013)), are usually studied numerically with Direct Numerical Simulation (DNS) or Large Eddy Simulation (LES), at the cost of highly idealizing the geometry and driving mechanism.

When the areas in need of high resolution are known *a priori*, a well developed strategy consists of nesting a high-resolution model within a low-resolution ocean model, with the latter driving the former (Blayo and Debreu, 2006; Debreu et al., 2012). Unfortunately, there are many situations when it is impossible to predict how the areas in need of higher resolution evolve in time. This is the case, for example, when considering the generation of internal waves by tidal flow over complex topography, or when studying the frontal dynamics of river plumes. For these situations, it is preferable to have models whose resolution can be dynamically adapted to accommodate evolving features. There are two basic approaches to this problem. In the first, the model is solved on a single grid, which spans the entire domain. This is the strategy followed by Piggott et al. (2008), which is based on an unstructured grid. Periodically, their model adapts the mesh to represent evolving flow structures such as boundary layers, thus optimizing the computational resources. Unstructured grids allow geometrical complexity, but conservation of mass and momentum during mesh optimization is a delicate issue. In the second approach, a dynamically evolving hierarchy of nested grids is considered with potentially different models solved within each grid. The latter approach has the advantage of allowing the use of a LES model on the grids with the finest resolution. This paper describes such a framework, which combines the recently developed Stratified Ocean Model with Adaptive Refinement (SOMAR) (Santilli and Scotti, 2011, 2015, 2017) with a large eddy simulation model. SOMAR is a non-hydrostatic model that can dynamically increase the resolution in localized regions where nonlinearity transfers energy to finer scales, allowing it to capture both large- and moderate-

scale features of the flow at a reduced computational cost. This allows SOMAR to simultaneously cover large domains and access scales that are not quite small enough to make SOMAR a DNS, but are amenable to be solved with a LES. The combined SOMAR-LES framework thus performs large eddy simulations on the finest grids of the adaptive mesh to model the effects of the unresolved small scale turbulence on the resolved scales of the flow. The SOMAR-LES framework is most efficient when small scale turbulence is localized in patches which are not stationary in either space or time (otherwise traditional grid nesting would suffice). In its current implementation, SOMAR-LES solves the Navier-Stokes equations in the Boussinesq approximation, assuming a linear equation of state for the stratifying agent.

As an application, in this paper we apply the framework to internal tide generation at model topographies, as well as the interaction of mode-1 internal waves with topography, with the intention to showcase the capability of the framework.

The rest of the paper is organized as follows. Section 2 summarizes the basic concepts of adaptive mesh refinement, thus providing a vocabulary for use in the rest of the paper. It also describes the equations solved, and the subgrid closures that are part of the LES package. Section 3 describes a series of test cases, while section 4 discusses the computational cost of our framework as compared to traditional, single-level, solvers. The last section provides a summary of the results presented earlier.

2. Modeling with Adaptive Mesh Refinement

SOMAR-LES is a modular platform that increases the range of dynamic scales that can be directly calculated (as opposed to being parameterized). The main components of this platform are SOMAR (Santilli and Scotti, 2011, 2015, 2017), which includes the Adaptive Mesh Refinement (AMR) machinery to handle the grids and the geometry, and LES which handles the subgrid closures to handle unresolved turbulence. In this section, we describe the major compo-

nents, starting with AMR.

AMR was originally developed by Berger and Oliger (1984) to provide additional resolution and accuracy to localized regions of a computational domain. Although AMR-based codes have been developed for use in ocean modeling (Blayo and Debreu, 1999; Popinet and Rickard, 2007), they do not currently enjoy widespread use. For this reason, we begin with a basic description of the AMR technique, providing a vocabulary that will be used throughout the remainder of this manuscript.

Before beginning, we need to draw a clear distinction between AMR and multigrid (MG) methods. The latter are a well established suite of techniques that are used to accelerate iterative solvers by removing low-frequency errors from an initial guess using a hierarchy of *coarsened* grids (Briggs et al., 2000). These iterative solvers are usually applied to elliptic problems (e.g., when calculating the pressure in incompressible flows), and are commonly found in many established ocean models (Marshall et al., 1998; Fringer et al., 2006). AMR, on the contrary, is used to provide additional resolution to flow features that are often transient and ideally exist over a very small fraction of the computational domain. AMR is most commonly used to time step hyperbolic conservation laws. In short, AMR refines locally in response to transient features generated by the hyperbolic component of the equations, whereas MG coarsens globally to speed up the convergence of the elliptic components. Although SOMAR-LES does make use of both techniques, this paper will not attempt to describe the MG component that is used to solve Poisson problems for the pressure. Instead, we will restrict our attention only to the AMR components of SOMAR-LES.

2.1. Introduction to AMR: Concepts and terminology

Ab initio, we assume that the physical domain is described by an n-dimensional (with $n = 2$ or 3) manifold $\mathfrak{M}$. A physical grid (i.e., a discretization of $\mathfrak{M}$) is mapped into a *logical* grid. AMR deals with logical grids and, in particular, the specific flavor of AMR used by SOMAR is *block-structured AMR*. This type of AMR requires (a) that logical grids are lattices with uniform spacing (though

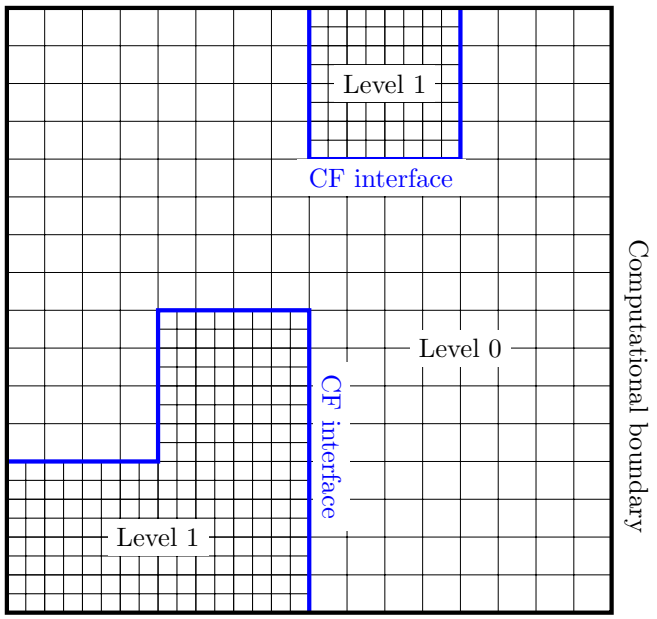


Figure 1: An example set of grids in a block-structured AMR hierarchy in two dimensions. Level 0 (the coarse level) covers the entire computational domain. Level 1 (the fine level) exists in localized regions. The coarse-fine (CF) interface is shown in blue. The refinement ratio tells us how many fine cells replace each coarse cell. In this example, the refinement ratio is 2×2.

not necessarily equal in all directions) and (b) logical grids to be composed of rectangular regions with a minimum size. This minimum size requirement is called the *block factor*. A 2D example set of block-structured grids with a block factor of 8 is shown in figure 1. Notice that none of the rectangular regions of these grids have a side that is less than 8 cells wide. This block factor requirement reduces some of the algorithmic complexity of both the AMR code and the operator stencils. It also allows us to employ the efficient Poisson solving techniques described in the original SOMAR paper (Santilli and Scotti, 2015).

A *level* is defined to be the set of logical grids with the same spacing. The base, or zero, level consists of a single grid, whose preimage covers the manifold. Traditional Navier-Stokes solvers such as MITgcm and ROMS are single-level solvers. Single-level solvers can have non-uniform grid spacing (e.g., by choosing

to "stretch" the grid in some direction), but cannot provide a discontinuous jump in resolution from one cell to the next.

A set of grids such as those in figure 1 are referred to as a *two-level* set of grids because they belong to the union of two levels – a zero level that spans the entire domain and a 1-level that only exists in sparse, localized regions. In the example shown in figure 1, each coarse level cell that is *tagged* for refinement is broken into a 2×2 set of fine level cells. The ratio of fine cells to each coarse cell is called the *refinement ratio*. In SOMAR-LES, it is typical to work in 3D and use a refinement ratio of $4 \times 4 \times 4$ with two levels (a coarse and a fine level), but the machinery is in place to handle arbitrary refinement ratios (in particular, refinements the resolution is increased in one direction but not in another) and any number of levels of refinement. When comparing the efficiency of SOMAR-LES to traditional single-level solvers, we will speak of running the single-level solver with SOMAR-LES's *effective grid* or *effective resolution*. This is the resolution achieved by the finest level. For example, suppose we are operating in a 3D computational domain that is broken into $128 \times 128 \times 256$ cells on the coarse level. If we perform a two-level solve with a refinement ratio of $4 \times 4 \times 1$, then the effective grid will have $512 \times 512 \times 256$ cells resolving the domain. In a performance comparison, the traditional single-level solver would need to be run at this effective resolution.

The coarse level cells exist throughout the domain, even where fine level grids contain more accurate data. This causes refined regions to contain somewhat redundant data. One version of the data (mass, momentum, etc.) exist on the coarse grids, while another more accurate version exist on the fine grids. Because of this, we need to distinguish between two different ways of looking at AMR data. From one point of view, we have *level data*. This is data that lives on a single level's grids. Coarse level data is all of the data on the coarse level (illustrated in the left pane of figure 2) while fine level data is all of the data on the fine level (illustrated in the right pane of figure 2). On the other hand, we have *composite data*. This is data from all levels, stitched together to create the most well-resolved representation of the AMR data available. Composite data

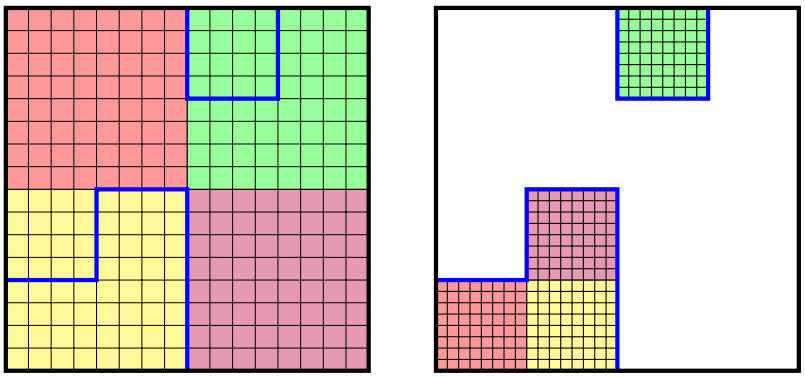


Figure 2: Each level's grids are decomposed into rectangular regions called patches (shown here in red, yellow, purple and green). Each processor is assigned a patch and must perform all computations in that region. The colors were chosen to identify processor assignments and are arbitrary from level to level.

does not have levels, nor does it contain redundant data. This is because we throw away any coarse data that is better represented by fine data. An example of *composite grids* suitable for harboring composite data is shown in figure 1. Notice that the multiple levels have been merged into one set of grids with spatially varying resolution. While SOMAR-LES uses level data to perform its timestepping, the end user would want to analyze the resulting composite data since it contains only the most accurate data at all points in the domain.

We will often speak of AMR data as level data that has been arranged in a logical hierarchy – the fine level sitting in the hierarchy "above" the coarser level. Data on the coarse grids can be interpolated "up" to the fine grids, while data from the fine grids can be averaged "down" to the coarse grids. Data that exists within the computational domain and is not sitting below a finer level is termed *valid* data. All other data is termed *invalid* data. On a given level, valid and invalid data are updated together by the level's time stepper, even though invalid data is eventually replaced by a coarsened (averaged) version of the finer level's more accurate data. For example, in figure 2, we see that level 0 contains a mix of valid and invalid data. The red and purple regions contain

only valid data while the yellow and green regions contain a mix of valid and invalid data.

On any given level, the grids are logically subdivided into disjoint rectangular *patches*. In figure 2, each shaded region represents a patch where computational work must be done. The decomposition of a given level's grid into patches allows to spread out the computations when multiple processing units (cores) are available. Each core handles a subset of patches and processor assignments are represented by color in figure 2. SOMAR uses the Chombo library (Adams et al., 2011) to handle most of the low-level MPI calls, and to perform load balancing. In order to assist this strategy, we maintain a layer of *ghost cells* around each patch (figure 3). These ghosts are filled in different ways, depending on their location in the domain. If a ghost cell coincides with the valid data of another patch at the same level, we perform a simple copy, or data *exchange*, from one patch to another. If a ghost cell lies beyond the computational domain, we use a standard extrapolation technique to enforce the prescribed boundary conditions. If a fine level ghost cell lays within the computational domain but on the coarse side of the CF interface, we use the methods described in the Chombo Design Document (Adams et al., 2011) to interpolate the coarse data up to the fine ghost. Ghost data is considered invalid data since it only exists as a computational convenience.

In AMR applications, it is convenient to represent the basic dynamical quantities (mass, momentum, etc.) as averages within each cell and use a finite volume timestepping scheme. As illustrated in figure 4, the fluxes of each of these quantities are spatially staggered and computed as averages over each timestep and cell face. By knowing how much of a quantity is inside a cell initially, and computing its flux into and out of neighboring cells, we can properly respect conservation laws as the flow evolves and travels across the coarse/fine (CF) interface. In section 2.3, we will delve into the details of timestepping on an AMR hierarchy of grids.

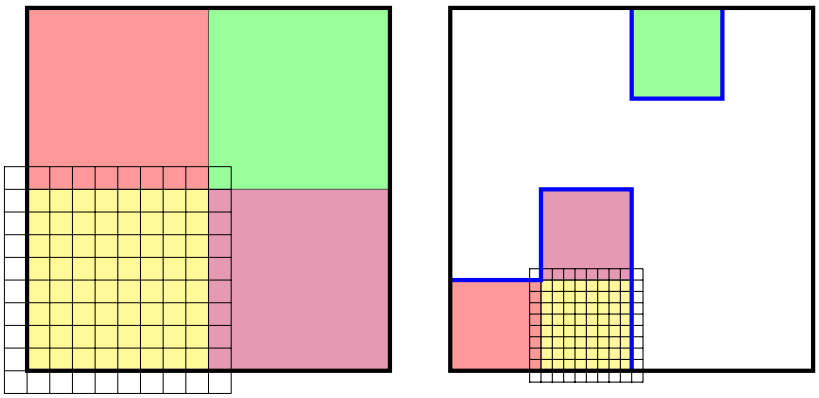


Figure 3: The interior and ghost cells of a single level 0 patch (left) and a single level 1 patch (right). The ghost cells that lay over the interior cells of another patch are filled by an MPI data exchange. The white ghost cells that lay beyond the computational boundary are filled using the boundary condition and a standard extrapolation formula. On the fine level, the white ghost cells that lay within the computational domain are filled using interpolation from the coarser level's data. The other patches have a similar interior/ghost cell structure.

2.2. Tagging cells for refinement

Periodically, SOMAR takes a pause from timestepping to decide if and where it needs to increase resolution. It is very possible and quite common that it reaches a point in the simulation where fine grids covering a certain area are not needed and should be eliminated in the interest of efficiency. This occurs, for example, when a turbulent patch moves away from a given location. On the other hand, if instabilities pump energy to fine-scales, SOMAR must be sure to properly represent those scales with a fine grid. For this purpose, at user-prescribed intervals, SOMAR scans the cells within a level with a flagging routine, which applies a criterion to decide whether the cell needs refinement or not. In case a cell is tagged for refinement, SOMAR will then search if a fine-level grid already exists covering the area. If the search turns out negative, then a new fine-level grid will be added to the existing fine-level grids. Conversely, if an area is not flagged for refinement, the corresponding cells in the fine-level grid that may cover the area are removed from the fine-level grid. Since the latter

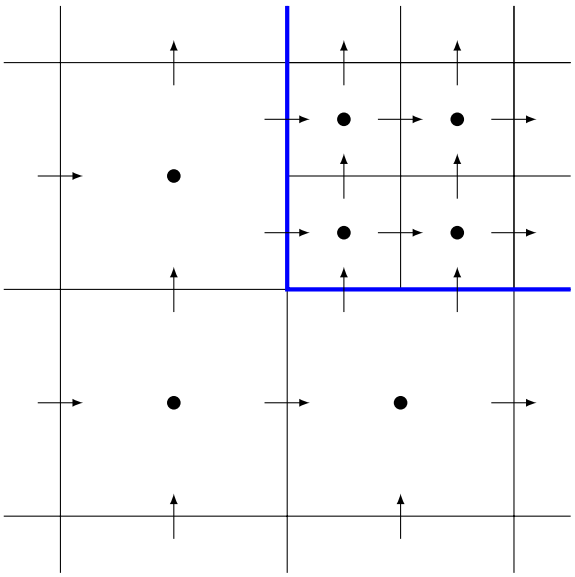


Figure 4: A depiction of various data centerings in 2D. Conserved quantities are stored at the cell centers (black circles) while the associated fluxes into and out of the cells are computed at the face centers (arrows). For example, the fluid's density, ρ, would be located at each of the black circles, representing the average density within the cells. The horizontal mass fluxes are located at each horizontal arrow and are computed to approximate the average value over each face and timestep. The vertical mass fluxes are similarly defined and are located at each vertical arrow. After each timestep, we update the fluid's density at each cell via $\rho \leftarrow \rho + \Delta t\,\mathrm{Div}[\mathrm{fluxes}]$. Other conserved quantities are stored and updated similarly. In 3D, the computation is similar, but here we restrict ourselves to the 2D xz-plane for illustration purposes.

process irreversibly destroys information, the flagging criterion must be carefully designed. The user can choose from a suite of predefined tagging criteria, some of which are detailed below, or can provide their own criterion. Ideally, the method of identifying cells that need refinement must flag cells before the fine-scale structures significantly develop. There are several tactics pre-coded in SOMAR to help identify cells in need of refinement.

1. Undivided differences – If coarse level data appears to be discontinuous due to a lack of resolution between two cell centers, that is if $|\Delta u| = |u_{\mathrm{left}} - u_{\mathrm{right}}| \geq \delta$, where δ is some prescribed tolerance, then we tag both

cells for refinement. We use undivided differences opposed to derivative estimates because derivative estimates will be approximately equal from level to level. For example, suppose we discretize $u(x) = \tanh(mx)$ on a grid with cell size Δx. A second-order estimate of the derivative at $x = 0$ would result in $m + O(\Delta x^2)$, which does not depend on the resolution at the leading order. On the other hand, the undivided difference is $m\Delta x + O(\Delta x^3)$, which diminishes with increasing resolution. It therefore makes sense to say that for some δ, if $|\Delta u| \geq \delta$ then more resolution is needed. This reasoning applies to the discretization of any function as can be shown via a Taylor series expansion about the point in question.

2. Vorticity – We wish to refine regions of high vorticity without eliminating valuable resolution information when estimating $|\nabla \times \mathbf{u}|$. To this end, we scale the x, y, and z components of the vorticity by $\Delta y \Delta z$, $\Delta z \Delta x$, and $\Delta x \Delta y$, respectively. If any of these scaled components rise above a prescribed tolerance, we tag the cell for refinement. Note that in 2D, we only need to specify one tolerance, while in 3D we need one tolerance for each direction.

3. The gradient Richardson number – Turbulence in a stratified flow is expected when the Gradient Richardson Number, defined as the ratio of the local stratification to the horizontal shear squared, becomes less than 0.25 (Miles, 1961; Howard, 1961). In practice, we tag those cells that satisfy $\mathrm{Ri_g} \leq \mathrm{O}(1)$ in anticipation of turbulence. Unlike other tagging strategies, this strategy is robust in the sense that the tolerance is flow-independent. We can use the same threshold, $\mathrm{Ri_g} \leq \mathrm{O}(1)$, for all stratified flows and expect satisfactory results.

4. Permanently tagged cells – We may also have *a priori* knowledge of regions that need finer grids. In this case, the cells covering such regions are pre-tagged.

Note that flagging a cell for refinement does not necessarily imply that the

flow within the cell is or will become turbulent (though it is expected to be so). From a computational point of view the only difference is that within a refined cell we have the possibility of evolving the dynamics more accurately because of the increased resolution, but also because we may decide to use a more sophisticated model (e.g., LES). Although SOMAR-LES is equipped with many versatile tagging strategies, all test cases considered in this study rely on the gradient Richardson number criteria due to its robustness as an indicator of potential turbulence.

2.3. Timestepping in AMR

Every timestep in AMR comes with some overhead. We must first create fine grids that capture the scales of the flow, initializing all data structures in a way that respects conservation laws and incompressibility. Then, we need to evolve both the coarse and fine level's data, finalizing the timestep with a *synchronization* procedure that ensures the composite solution again respects conservation laws and incompressibility. If the fine level grids are chosen wisely and the problem at hand is amenable to AMR, then the cost of each timestep's overhead is eclipsed by the computational savings. In this section, we will briefly walk through this process. Timestepping proceeds recursively through the hierarchical tree that contains the levels. So, it is only necessary to illustrate it assuming two levels. Our timestep will begin when both levels are at time t^n and end when both levels are at time t^{n+1}. We will identify coarse and fine level quantities with a subscript C and F. The coarse level will march forward in time with a single timestep of $\Delta t_C = t^{n+1} - t^n$, while the fine level will march with timesteps of Δt_F. Typically, $\Delta t_F < \Delta t_C$ due to stability constraints.

2.3.1. Initialization of new fine level data

Periodically[1], we must re-evaluate the fine grid structure, deciding which coarse patches would benefit from more resolution and which fine patches are

[1]In SOMAR, the regridding frequency is not hard-coded. In practice, we find that every ~ 10 timesteps gives a good compromise between efficiency and accuracy.

no longer required. This is accomplished by tagging coarse level cells that need refinement and letting the AMR algorithm construct a set of fine patches that refines every tagged cell, respects the block factor requirement, is easily parallelizeable, and minimizes unrequested refinement. The procedure is described in detail by Martin (1998).

When creating a new fine level patch, we fill it using a conservative, limited, linear interpolation of the underlying coarse level data (see Adams et al. (2011)). The interpolator ensures conservation of mass and momentum, and protects against the creation of new extrema, but does not produce a composite velocity field that is incompressible. The trouble lies at the coarse-fine (CF) interface. Consider a velocity field on the coarse level, $\mathbf{u}_C$, that is discretely divergence free. This field has been constructed in such a way that the stencil of $\nabla \cdot \mathbf{u}_C$ that uses only coarse grid data is zero everywhere. However, after local interpolation of $\mathbf{u}_C$ onto a fine patch, we no longer require $\nabla \cdot \mathbf{u}_C = 0$, but $\nabla \cdot \mathbf{u}_{\rm comp} = 0$, where $\mathbf{u}_{\rm comp}$ is the composite velocity field made of the valid data on both the coarse and fine levels. When computing $\nabla \cdot \mathbf{u}_{\rm comp}$ near the CF interface, the stencil of the divergence may lay partially on the fine grid and partially on the coarse grid. This modified stencil produces a non-zero divergence. We repair this problem with a *composite projection* to remove the irrotational component at the CF interface acquired by the interpolator. This is a cell-centered, approximate projection that is well described by Martin (1998).

Once all dynamical quantities have been interpolated onto the new patches and the composite velocity field has been projected, we generate a new estimate of the composite pressure field at t^n. This is accomplished by taking a very small ($\sim \Delta t_C/10$) timestep and projecting the new velocity field. Since this small timestep is performed only to produce a pressure estimate, all other quantities are returned to their original values at t^n upon completion.

2.3.2. The composite timestep – subcycling

Once the fine level is initialized, we are ready to advance the composite solution. To do so, we must first choose suitable timesteps on the coarse and

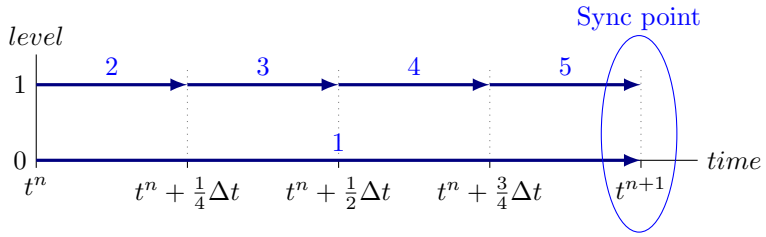


Figure 5: A schematic of a single, composite (subcycled) time step. The composite timestep shown here requires 4 single-level timesteps (represented by 4 right-pointing arrows), each taken in an order shown by the blue numbers above each arrow. Viewed as a predictor-corrector method, the composite timestepping algorithm first computes a coarse level prediction of the state at time t^{n+1}, then generates a finely resolved correction. Once the coarse and fine level solutions arrive at t^{n+1}, they must be *synchronized* (see section 2.3.3).

fine levels, Δt_C and Δt_F. Considering the stability constraints of the coarse and fine levels independently, we generate a Δt_C and $\Delta t_F^\star$. This determines the composite timestep's final time $t^{n+1} = t^n + \Delta t_C$. The fine level must take timesteps that are commensurate with Δt_C to prevent the coarse and fine levels from becoming staggered in time. To accomplish this, we find the smallest integer a such that $\Delta t_C / a \leq \Delta t_F^\star$ and set $\Delta t_F = \Delta t_C / a$. For the flows considered by SOMAR-LES, a often ends up being equal to the horizontal refinement ratio (typically 4) due to the Courant-Friedrichs-Lewy (CFL) stability condition (Courant et al., 1928).

Having chosen a suitable Δt_C and Δt_F, we advance the state variables using a predictor-corrector method. We predict the state at t^{n+1} by advancing the coarse level, then we compute a correction by advancing the more accurate fine level. This sequence is called *subcycling* and is diagrammed in figure 5. At the CF interface, the fine level state's Dirichlet boundary conditions are generated by interpolation of the coarse level state. At an intermediate time t^{n+f} with $0 \leq f \leq 1$, we first interpolate the coarse state in time via $u_C^{n+f} = (1-f)u_C^n + f u_C^{n+1}$, then interpolate u_C^{n+f} in space. When both the coarse and fine levels arrive at t^{n+1}, we must *synchronize* the levels.

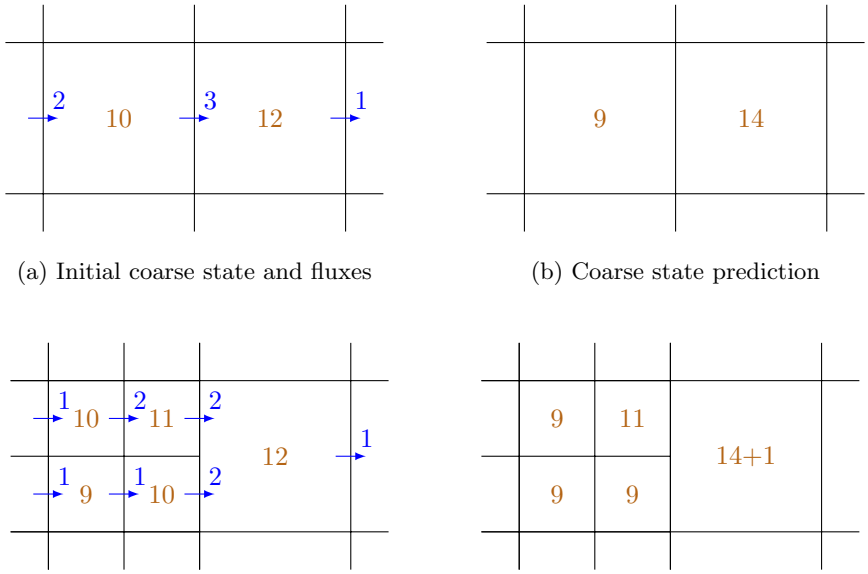
2.3.3. Synchronization

Synchronization is the process of stitching together data over a hierarchy of levels to construct a sensible composite solution. In general, it occurs anytime two or more levels reach the same time. Synchronization consists of three steps:

1. Update the coarse level with an interpolation of the more accurate fine state.
2. Remove conservation law violations at the CF interface (called *refluxing*).
3. Remove the solenoidal component of the final composite velocity.

Of these three steps, refluxing requires some discussion. There is no reason to believe that the coarse level update will generate fluxes at the CF interface that are identical to the sum of the overlying fine level fluxes. The increased accuracy of the fine level update may determine that more or less fluid should be pushed across the interface. When this happens, we end up with a flux mismatch, $\delta F = F_F - F_C$, at the CF interface. The coarse level state must be corrected, or *refluxed*, in valid cells adjacent to the CF interface via $u_C \leftarrow u_C + \Delta t_C \, \delta F$. This is illustrated in figure 6 with sample data and fluxes. The cell averaged data is shown at each cell center while the fluxes produced by the timestepping algorithm is shown at the face centers in blue. For simplicity, we assume no fluxes in the vertical and fix $\Delta t_C = 1$ so that the sum of the fluxes around a cell equals the net change within that cell. For example, suppose the provided state data represents the mass of fluid within each cell (in arbitrary units). Then in figure 6a, the cell on the right initially has a mass of 12 and there is a net flux of 3-1=2 into the cell. So, the final mass in that cell is 12+2=14, shown in figure 6b. However, the fine level updates may generate fluxes at the CF interface that do not add up to the underlying coarse level fluxes. This is shown in figure 6c, where the net flow of mass through the CF interface is 2+2=4, not 3 as predicted by the coarse level update. So, there is a flux mismatch at the CF interface among the coarse and fine level. In figure 6d, we correct (reflux) the final coarse level cell to the right of the CF interface to ensure mass

conservation. After refluxing, we have an even greater need for a composite projection to ensure incompressibility.



(a) Initial coarse state and fluxes

(b) Coarse state prediction

(c) Initial composite state and fluxes

(d) Final composite state after refluxing

Figure 6: The effect of a composite timestep is illustrated here with sample data and fluxes. The cell averaged data is shown at each cell center while the fluxes produced by the timestepping algorithm is shown at the face centers in blue. For simplicity, we assume no fluxes in the vertical and fix $\Delta t_C = 1$ so that the sum of the fluxes around a cell equals the net change within that cell. In figure 6d, we correct (reflux) the final coarse level cell to the right of the CF interface to ensure mass conservation. This process is described in detail in section 2.3.3.

2.4. The independence of the level solvers

A composite timestep in SOMAR is a collection of single-level timesteps, stitched together at its ends by the initialization and synchronization routines. With this predictor-corrector algorithm, the timesteppers on any given level can operate almost completely independently of one another. The only exception occurs when a CF interface is within the domain's interior. In that case, the fine level timetepper needs Dirichlet boundary condition data from the coarse level. SOMAR-LES takes advantage of this independence by allowing different

timestepping algorithms among the levels.

On the coarse level, it is assumed that the flow over valid cells is laminar, on the assumption that where finer scale eddies (turbulence) exists, the corresponding cells will contain invalid data, that is, will sit below finer grids in the grid hierarchy. Therefore, on the coarse level, we update the state variables using the SOMAR timestepping algorithm detailed in section 2.5.1 and in Santilli and Scotti (2015). On the fine level, we augment the SOMAR algorithm with a Large-Eddy Simulation (LES) to model whatever scales remain unresolved.

The benefit of such a multi-level scheme is that the large scale flow can be modeled efficiently by the large scale solver on a coarse grid. Transient laminar features that require a slight increase in resolution are resolved by creating temporary fine patches as the need arises. When energy is pumped into scales beyond the fine grid's resolution, the LES parameterizes the effects of the unresolved scales. With this scheme, the LES is not restricted to lab-scale flows, but receives realistic forcing through the CF interface from the large scale ocean. In turn, the large scales of the flow are not burdened with artificially high viscosities and diffusivities, but receives realistic, localized mixing and dissipation information from the LES during synchronization for use in the next timestep.

2.5. Physical equations

We now describe the equations solved by SOMAR-LES. Since the levels perform their timesteps independently of each other (with the exception of some information garnered during initialization and synchronization), SOMAR can in principle solve different equations on different levels. SOMAR-LES takes advantage of this by using a set of conservation equations on all levels but the highest. On the highest level (=highest resolution), we introduce explicit subgrid stress and flux terms, i.e. we perform a traditional LES. To accommodate complex topographies, the equations are solved in generalized coordinates: the user can either provide routines that specify the metric tensor of the manifold $\mathfrak{M}$, or simply provide a digital elevation model of the topography and use σ-coordinates, which are predefined in SOMAR-LES. In table 1, we provide a brief description

of the symbols used in our formulation.

$\mathbf{u}$	= velocity vector
(u, v)	= horizontal velocity components
w	= vertical velocity component
$\rho_{\mathrm{bg}}(z)$, ρ_0	= background density profile and reference density (resp.)
$b_{\mathrm{bg}}(z)$	= background buoyancy profile
b^*	= buoyancy deviation
N	= Brunt-Väisäla frequency
p^*	= pressure deviation from hydrostatic
f	= Coriolis parameter
g	= gravitational acceleration
$\hat{\mathbf{k}}$	= vertical unit vector
ν	= kinematic viscosity
κ	= thermal diffusivity
$\boldsymbol{\tau}$	= subgrid stress tensor (modeled via LES at the fine level)
$\boldsymbol{\lambda}$	= subgrid buoyancy flux (modeled via LES at the fine level)
A_C	= a dynamical field, A, discretized on the coarse grid
A_F	= a dynamical field, A, discretized on the fine grid
$A_{F\to C}$	= fine to coarse level interpolation of A
S_{ij}	= rate of strain tensor
ν_{sgs}	= subgrid scale eddy viscosity
κ_{sgs}	= subgrid scale eddy diffusivity
$\overline{A}$	= LES filter applied to A
$\tilde{\overline{A}}$	= high-pass Laplacian filter applied to A
Δ	= LES filter width
C_s	= Smagorinsky coefficient

Table 1: Quantities used in the model formulation. All entries at and below S_{ij} are only used by the LES on the finest level.

2.5.1. Coarse grid equations

The coarse level solves the Navier-Stokes equations in a form amenable to ocean modeling. Specifically, we split the density into a stable, static background stratification and its deviation, both of which are further scaled as buoyancies,

$$b_{\mathrm{bg}}(z) = -g\frac{\rho_{\mathrm{bg}}(z) - \rho_0}{\rho_0} \quad \text{and} \quad b^*(\mathbf{x},t) = -g\frac{\rho(\mathbf{x},t) - \rho_{\mathrm{bg}}(z)}{\rho_0},$$

where ρ_0 is a representative density scale. From this point forward, we will drop the explicit functional dependence and describe the background stratification via the Brunt-Väisäla frequency,

$$N(z) = \sqrt{\frac{db_{\mathrm{bg}}}{dz}} > 0.$$

Removing the background stratification allows us to eliminate the large hydrostatic pressure component and maintain only the deviation from hydrostasy. In SOMAR-LES, the user specifies the functional dependence of N on z in a function. In the examples discussed below, we assume a constant N to allow comparison with published experiments and theory. We employ the Boussinesq and f-plane approximations and assume the buoyancy is linearly related to temperature. For simplicity, we write the equations in their Cartesian form, though the reader should remember that SOMAR-LES internally represents the equations in generalized coordinates. In terms of the quantities defined in table 1, the equations solved on the coarse level are

$$\begin{aligned}
\nabla \cdot \mathbf{u}_C &= 0, && \text{(1a)}\\
\frac{D\mathbf{u}_C}{Dt} &= \nu\nabla^2\mathbf{u}_C - \nabla\cdot(\boldsymbol{\tau})_{F\to C} - f\hat{\mathbf{k}}\times\mathbf{u}_C + b^*_C\hat{\mathbf{k}} - \nabla p^*_C, && \text{(1b)}\\
\frac{Db^*_C}{Dt} &= \kappa\nabla^2 b^*_C - \nabla\cdot(\boldsymbol{\lambda})_{F\to C} + w_C N^2. && \text{(1c)}
\end{aligned}$$

where the subscript C denotes discretization on the coarse level grids.

SOMAR uses second-order, central finite differences for the spatial discretization of viscous terms, computed explicitly for time advancement. Advection terms are computed using the Piecewise-Parabolic Method (PPM) of

Colella and Woodward (1984), a finite volume method, along with the multi-dimensional upwinding scheme of Colella (1990) and Saltzman (1994). Since we want the dissipation of energy to be the result of subgrid parameterizations, we avoid the use of limiters in our finite volume methods. The terms that give rise to buoyancy oscillations and Coriolis effects are treated semi-implicitly (for details, see Santilli and Scotti, 2015; Santilli, 2015).

The inclusion of the subgrid buoyancy flux vector $\boldsymbol{\lambda}$ and the subgrid stress tensor $\boldsymbol{\tau}$ is what makes SOMAR-LES different from the original SOMAR formulation. Both terms are provided by the LES, which runs on the highest level grids, at the beginning of the lower levels' timestep, so they must be interpolated onto the coarser, lower-level grids. At valid level cells (cells with data that will not be superseded by higher-level data), we assume these subgrid fluxes are zero. This assumption is supported by the fact that if sub-coarse grid motions were to exist, a finer level would have been created and that fine level would provide a non trivial $\boldsymbol{\lambda}$ and $\boldsymbol{\tau}$.

During synchronization, all invalid lower-level data will be overwritten with an interpolation of the more accurate overlying higher-level data. At first glance, it may seem that the coarse level has no need for estimates of $\boldsymbol{\lambda}$ and $\boldsymbol{\tau}$. After all, the properly mixed dynamical variables will be computed on the fine level, then averaged down to the coarse level once we reach a sync point. However, these subgrid fluxes have the effect of pushing fluid across the CF interface, into and out of valid coarse cells adjacent to the interface. So, while $(\boldsymbol{\lambda})_{F\rightarrow C}$ and $(\boldsymbol{\tau})_{F\rightarrow C}$ only exist at invalid coarse level cell faces, their divergence extends slightly further into cells that will not be overwritten during synchronization. Providing these estimates on the coarse level helps enforce conservation laws, prevents mismatches at the CF interface among the coarse and fine level data, and allows the coarse level to generate a more accurate pressure estimate throughout the domain. In practice, the LES at the highest level provides a turbulence scheme for the lower levels.

2.5.2. The LES equations

The equations solved on the highest-level are formally identical to the ones solved on the lower-level grids

$$\nabla \cdot \overline{\mathbf{u}}_F = 0, \tag{2a}$$

$$\frac{D\overline{\mathbf{u}}_F}{Dt} = \nu\nabla^2\overline{\mathbf{u}}_F - \nabla\cdot\boldsymbol{\tau} - f\hat{\mathbf{k}}\times\overline{\mathbf{u}}_F + \overline{b}^*_F\hat{\mathbf{k}} - \nabla p^*_F, \tag{2b}$$

$$\frac{D\overline{b}^*_F}{Dt} = \kappa\nabla^2\overline{b}^*_F - \nabla\cdot\boldsymbol{\lambda} + \overline{w}_F N^2, \tag{2c}$$

except that now the dependent variables are understood as implicitly filtered, and $\boldsymbol{\lambda}$ and $\boldsymbol{\tau}$ are modeled instead of being "inherited" by higher-level grids. Also, note that we use a subscript F to denote discretization over the fine level grids. Currently, SOMAR-LES implements "turbulent viscosity" type models, that is

$$\tau_{ij} = -2\nu_{sgs}\overline{S}_{ij} \tag{3}$$

and

$$\lambda_j = -\kappa_{sgs}\frac{\partial\overline{b}^*_F}{\partial x_j}, \tag{4}$$

where ν_{sgs} and κ_{sgs} are respectively the subgrid scale viscosity and diffusivity and $\overline{S}_{ij} = \frac{1}{2}\left(\frac{\partial\overline{u}_{i,F}}{\partial x_j} + \frac{\partial\overline{u}_{j,F}}{\partial x_i}\right)$, is the symmetric rate-of-strain tensor of the filtered velocity field. However, there is nothing in the SOMAR-LES framework that prevents the use of other closure schemes, which can be added writing a suitable C++ class. As for the choice of the eddy viscosity, presently SOMAR-LES offers two choices: one is the standard Smagorinsky model (Smagorinsky, 1963), the other is the filtered structure function (FSF) of Ducros et al. (1996).

In the Smagorinsky model, the subgrid scale viscosity is given by

$$\nu_{sgs}(\mathbf{x},t) = (C_S\Delta)^2\sqrt{2\overline{S}_{ij}\overline{S}_{ij}}, \tag{5}$$

where Δ is the filter width based on the grid size in all directions $(\Delta x\,\Delta y\,\Delta z)^{1/3}$. A subgrid scale Prandtl number $Pr_{sgs} = \nu_{sgs}/\kappa_{sgs}$ of unity is used to calculate the subgrid scale diffusivity κ_{sgs} in all cases. In this model, the Smagorinsky

coefficient C_s is the only tunable parameter. Assuming that the subgrid turbulence is isotropic, the Smagorinsky coefficient C_s, can be calculated in terms of the Kolmogorov constant, obtaining for it the value 0.17. However, several studies have shown that $C_s = 0.17$ leads to excessive dissipation in shear flows and smaller values of the Smagorinsky coefficient between 0.08 and 0.1 have been used in LES of shear flows, e.g. channel flow (Deardorff, 1970; Piomelli et al., 1988) and pipe flow (Brandt, 2005). The value of C_s is found to increase from a small value at early time to $C_s \approx 0.13$ at late time in LES of a temporally evolving shear layer (Vreman et al., 1997), and from 0.04 near the inflow to 0.13 in the region of fully-developed turbulence in a spatially evolving jet (Ribault et al., 1999). LES of a stratified shear layer (Pham and Sarkar, 2014) shows that C_s in the turbulent stage varies from 0.13 in the central core of the shear layer to about 0.08 at the flanks where the turbulence is intermittent. In SOMAR-LES the choice of C_s is left to the user. In the following experiments, we have used a value of 0.085 to allow instabilities and transition to turbulence, an important feature in the problem of turbulence driven by internal tides (Sarkar and Scotti, 2017). Coding of the dynamic Smagorinsky model (Germano et al., 1991), which has no tunable parameters, is left for the future.

In the FSF model, a high-pass Laplacian filter (denoted by $\sim$), iterated three times, is applied to the velocity field before calculating the structure function. A six-neighbor formulation is used to calculate the filtered velocity as shown below (see Ducros et al., 1996, for details)

$$\begin{aligned}\tilde{\overline{u}}(i,j,k) = \overline{u}(i+1,j,k) - 2\overline{u}(i,j,k) + \overline{u}(i-1,j,k) \\ +\overline{u}(i,j+1,k) - 2\overline{u}(i,j,k) + \overline{u}(i,j-1,k) \\ +\overline{u}(i,j,k+1) - 2\overline{u}(i,j,k) + \overline{u}(i,j,k-1).\end{aligned}$$

The subgrid scale viscosity, ν_{sgs}, is given by

$$\nu_{sgs}(\mathbf{x},t) = 0.0014\, C_K^{-3/2}\, \overline{\Delta}\ (\overline{F_2}(\mathbf{x},\Delta,t))^{1/2} \tag{6}$$

where C_K is the Kolmogorov constant set equal to 0.5 (Sreenivasan, 1995) and $\overline{F_2}$ is the filtered structure function given by

$$\begin{aligned}\overline{F_2}(\mathbf{x},t) = \quad & 1/6\Big(||\tilde{\overline{u}}_{i+1,j,k}-\tilde{\overline{u}}_{i,j,k}||^2+||\tilde{\overline{u}}_{i-1,j,k}-\tilde{\overline{u}}_{i,j,k}||^2 \\ & +||\tilde{\overline{u}}_{i,j+1,k}-\tilde{\overline{u}}_{i,j,k}||^2+||\tilde{\overline{u}}_{i,j-1,k}-\tilde{\overline{u}}_{i,j,k}||^2 \\ & +||\tilde{\overline{u}}_{i,j,k+1}-\tilde{\overline{u}}_{i,j,k}||^2+||\tilde{\overline{u}}_{i,j,k-1}-\tilde{\overline{u}}_{i,j,k}||^2\Big).\end{aligned}$$

The FSF model is less dissipative when compared to the classical Smagorinsky model when both are applied to the same flow.

2.6. Turbulent diagnostics

The instantaneous velocity field can be decomposed into mean and fluctuating components, $u = \langle u \rangle + u'$. The calculation of the mean component $\langle u \rangle$ varies according to the flow type. If there is a periodic direction, the mean component is calculated by performing a spatial average in the periodic direction on all grid levels. Then, derived quantities such as turbulent kinetic energy and turbulent production are first calculated on all levels, with the more accurate fine level values averaged down to the coarse grid. If a periodic direction doesn't exist, we average the fine-level solution down to the coarsest grid, then perform a conservative linear interpolation back up to the fine grid. This essentially filters out velocity fluctuations (eddies) that can only be resolved on the refined grids. The interpolated velocity is assumed to be the ensemble average on the fine level. In the case where there is no periodic direction, the derived quantities such as TKE and turbulent production are computed only on the fine level grid and averaged down to the coarse grid.

3. Test cases

To illustrate the capabilities of SOMAR-LES, we select three problems characterized by non-trivial topography and turbulent regions which are not stationary. The first problem concerns the generation of internal tides by barotropic flow over a nominally two-dimensional topography. This is a problem that has

been extensively studied theoretically and numerically (Laurent and Garrett, 2002; Petrelis et al., 2006; Buijsman et al., 2012) . Several observational studies (Polzin et al., 1997; Ledwell et al., 2000; Rudnick et al., 2003; Klymak et al., 2006) indicate enhanced mixing due to internal wave breaking above rough bathymetries. There have been several attempts (Klymak et al., 2006; Carter et al., 2008; Alford et al., 2011) to estimate 'q', the fraction of baroclinic energy which is dissipated locally at turbulent hot spots such as the Hawaiian ridge and Luzon strait. However, there is large uncertainty in this parameter due to the sparseness of direct measurements of dissipation (for an extensive review of the literature see Sarkar and Scotti, 2017, and references therein). The second problem that we consider is similar to the first, but with a three-dimensional topography, a Gaussian ring. Under weakly barotropic forcing, this set up generates internal wave beams that focus into a small area, where wave-wave interaction generates a localized patch of turbulence (Buhler and Muller, 2007; Chalamalla et al., 2016, 2017). It is of interest to test how the regridding algorithm of SOMAR-LES behaves when the region that needs high-resolution has a complex three-dimensional spatial shape. Finally, the third test case considers the interaction of a mode-1 internal tide with an idealized two and three-dimensional isolated bathymetry. This problem has been studied recently by Legg (2014), with the MITgcm model (Marshall et al., 1998). Legg's work is interesting because, in addition to providing a way to quantitatively test the accuracy of SOMAR-LES against a well established model, it also provides a way to compare the computational costs of a computationally mature single-level solver vs. SOMAR-LES.

All the test cases considered in this study are performed with two-level grids with a refinement ratio of 4 in all directions, using the gradient Richardson number Ri_g criteria to tag the coarse level cells for refinement. Regridding is performed every 10 coarse time steps. If the regridding interval is too small, the computational overhead increases, but if the regridding interval is too large, the nonlinear features may move off of the fine grid. The regridding interval has been chosen as 10 as a compromise between these two effects. All test cases

use σ-coordinates over a digital elevation map (DEM), which, for simplicity, is obtained from an analytic expression of the topography. However, SOMAR-LES is able to use DEMs provided as elevation at discrete points. Unless otherwise specified, velocity and other dynamical quantities shown in figures represent composite data.

3.1. Tidal flow over an idealized triangular ridge.

The first set of test cases deals with a smooth two-dimensional bathymetry that has no variation in the third (span-wise) direction. Simulations are performed by time advancing the three-dimensional Navier-Stokes equations under the Boussinesq approximation. Earth's rotation is ignored in this study for simplicity. The barotropic tidal forcing is imposed by adding the following pressure gradient term to r.h.s. of the x-momentum equation (i.e., eq. 2b or 1b),

$$F_0(t) = U_0 \Omega \cos(\Omega t). \tag{7}$$

Here, U_0 is the barotropic velocity amplitude, Ω is the forcing frequency, and t represents dimensional time.

Case	$\boldsymbol{U_0(m/s)}$	$\boldsymbol{l(m)}$	$\boldsymbol{h(m)}$	$\boldsymbol{N^2_\infty(s^{-2})}$	$\boldsymbol{Ex}$	$\boldsymbol{Re_s}$	$\boldsymbol{\gamma}$	**LES**
$SUB1$	0.125	1.9	0.33	5.6	0.066	177	0.57	Off
$SUP1$	0.125	1.9	0.33	67.33	0.066	177	2.17	Off
$CRIT1_{ls}$	0.0175	1900	330	29.26×10^{-8}	0.066	2092	1.0	On
$CRIT2_{ls}$	0.106	1900	330	29.26×10^{-8}	0.4	12669	1.0	On

Table 2: Simulation parameters for different cases of tidal flow simulation at an idealized triangular ridge. Barotropic forcing frequency $\Omega = 1$ rad/s for cases SUB1 and SUP1. For cases $CRIT1_{ls}$ and $CRIT2_{ls}$, the forcing frequency $\Omega = 0.000141$ rad/s, which corresponds to the semi-diurnal barotropic tide.

Case	$\boldsymbol{L_x}$	$\boldsymbol{L_y}$	$\boldsymbol{L_z}$	$\boldsymbol{dx_C}$ $[\boldsymbol{dx_F}]$	$\boldsymbol{dy_C}$ $[\boldsymbol{dy_F}]$	$\boldsymbol{dz_C}$ $[\boldsymbol{dz_F}]$
$SUB1$	40.0	0.25	3.28	0.039 [0.0097]	0.0156 [0.0039]	0.0128[0.0032]
$SUP1$	40.0	0.25	3.28	0.039 [0.0097]	0.0156 [0.0039]	0.0128[0.0032]
$CRIT1_{ls}$	40000.0	250.0	3280.0	39 [9.765]	15.6 [3.9]	12.8 [3.2]
$CRIT2_{ls}$	40000.0	250.0	3280.0	39 [9.765]	15.6 [3.9]	12.8 [3.2]

Table 3: Grid parameters. All units are in meters. The coarse grid is uniform and has $1024 \times 16 \times 256$ grid points in the x, y and z directions, respectively.

The four test cases considered covers different values for the barotropic velocity amplitude U_0, the Brunt-Väisälä frequency of the uniform background stratification N_∞ and topographic horizontal (l) and vertical h scales (table 2). The excursion number, defined by $Ex = U_0/(l\Omega)$, is the ratio of the tidal excursion length (U_0/Ω) of a fluid parcel to the topographic length scale l. Here, l is the half-width of the topography and Ω the frequency of the barotropic forcing (tidal frequency for short). Excursion number is a measure of how strong the fluid advection is relative to the length scale of the topography. The Stokes Reynolds number, defined by $Re_s = U_0\delta_s/\nu$, where δ_s is the Stokes boundary layer thickness given by $\delta_s = \sqrt{2\nu/\Omega}$, is the square root of a Reynolds number based on the excursion length $L = U_0/\Omega$. The slope criticality is measured by the steepness parameter $\gamma \equiv \tan\beta/\tan\alpha$, where β is the topographic slope angle and α is the internal wave characteristic angle. Sponge forcing is applied at the left (x_{min}) and right (x_{max}) boundaries to absorb the internal wave energy radiated outside the domain of interest.

The bathymetry considered in all cases is a smoothed triangular ridge as in previous studies (Rapaka et al., 2013; Jalali et al., 2014; Rapaka and Sarkar, 2016). Around 20% of the bathymetry has linear slope on either side. Cases $SUB1$ and $SUP1$ are at laboratory length scale of $O(1)$ m. Cases $CRIT1_{ls}$ and $CRIT2_{ls}$ have an obstacle with the same triangular shape but at an oceanic length scale of $O(1000)$ m. The background stratification and forcing frequency

are adjusted to achieve both criticality along the linear section of the slope and the target value of excursion number, Ex.

The height of the topography, $h(x)$, varies among different cases. The grid resolution for all the test cases is shown in table 3. Parameters dx_C, dy_C and dz_C represent grid spacings on the coarse level grid in stream-wise (x), spanwise (y) and vertical (z) directions, respectively. Fine grid spacings, dx_F, dy_F and dz_F, are shown in squared brackets in table 3. All simulations discussed in this section are performed with a molecular viscosity of 10^{-6} m^2/s and the ratio of molecular viscosity to diffusivity is equal to 1, except in case $CRIT1_{ls}$ which uses a ratio equal to 7. Periodic boundary conditions are used in the spanwise (y) direction.

In case $SUB1$, the slope is subcritical with respect to the internal wave characteristic angle ($\gamma < 1$). In case $SUP1$, the slope is supercritical with respect to the internal wave characteristic angle ($\gamma > 1$). Cases $CRIT1_{ls}$ and $CRIT2_{ls}$ are the critical slope cases, where the angle of the uniform slope region matches exactly with the internal wave characteristic angle ($\gamma = 1$). In all these cases, the steepness parameter is adjusted by varying the internal wave characteristic angle through the background stratification rather than changing the slope angle.

When the excursion number Ex is small (< 0.1), the excursion length of fluid parcels near the bathymetry is much smaller than the horizontal length scale of the topography. Advection by the barotropic tide affects only weakly the resonant response at critical-slope regions, and as a consequence well-delineated tidal beams emanate from the obstacle (Holloway and Merrifield, 1999; Petrelis et al., 2006; Cole et al., 2013; Gostiaux and Dauxois, 2007; Rapaka et al., 2013). Figure 7 shows snapshots of the stream-wise velocity field for cases $SUB1$ (subcritical) and $SUP1$ (supercritical) at phase $\phi = \pi/2$, when the barotropic velocity amplitude is maximum in the positive x-direction. In both cases, the excursion number is 0.066, strong internal wave beams are generated at the topography and radiate away from the generation region. Baroclinic response is weak in the subcritical case compared to the supercritical case i.e velocity amplitude of tidal

beams is stronger and higher harmonics are more prominent in the supercritical case when compared with the subcritical case. These results agree very well qualitatively with the previous DNS study (see figure 2 of Rapaka et al. (2013)).

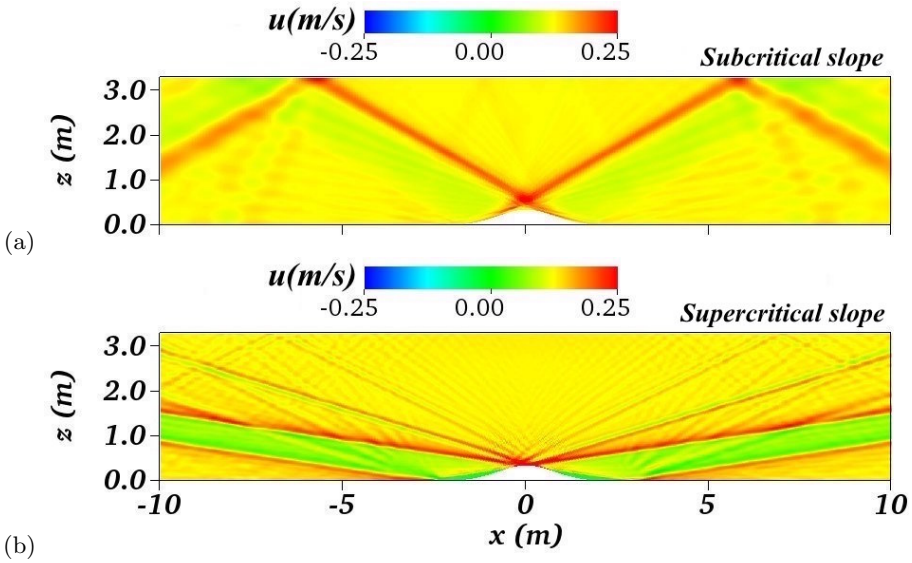


Figure 7: Internal waves beams generated during subcritical (top row, case SUB1) and a supercritical (bottom row, case SUP1) generation. The beams are visualized plotting the intensity of the x component of the velocity when the barotropic velocity is maximum in the positive x-direction.

Conversely, when $Ex = O(1)$, the internal wave beams become less coherent. The wave response and turbulence become more asymmetric with respect to the topography(Jalali et al., 2014). The top row of figure 8 shows xz-plane snapshots of the x-direction velocity at various time instances for case $CRIT2_{ls}$ with an Excursion number of $Ex = 0.4$. Relative to cases SUB1 and SUP1, whose Excursion number is lower, $CRIT2_{ls}$ has incoherent wave beams. The computational mesh on the fine level, represented by a black shaded region, is also shown. The fine level grid adapts in time based on the gradient Richardson number criteria ($Ri_g < 0.25$) as it follows the turbulent patch that sloshes back and forth over the ridge. In the bottom panel, zoomed-in snapshots of turbulent

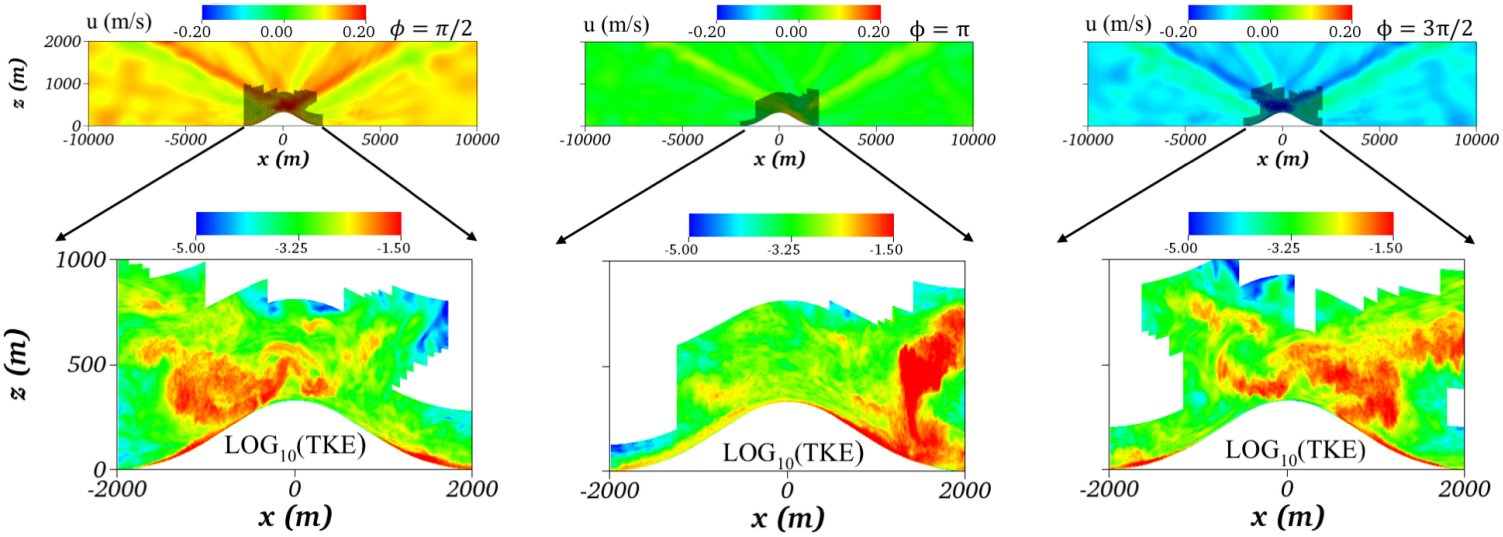


Figure 8: xz-plane snapshots of the streamwise (x-direction) velocity and $LOG_{10}(TKE)$ when the barotropic flow reaches its rightward maximum (left panel) and leftward maximum (right panel), as well as during the reversal through zero velocity (middle), for case $CRIT2_{ls}$ with the $Ex = 0.4$. TKE has units of $\mathrm{m}^2/\mathrm{s}^2$. The computational mesh on the fine level is represented by the shaded region. Since the fine-level mesh follows turbulent spots, at any given time the asymmetry of the mesh over the ridge reflects the underlying asymmetry of turbulence distribution.

kinetic energy ($LOG_{10}(TKE)$) are shown on the fine level. Turbulence intensity is maximum during the flow reversal through zero velocity (middle panel in the bottom row), when the density overturn breaks down into turbulence creating a turbulent patch nearly 500 m tall.

The fine level has very limited extent in the x (streamwise) and z (vertical) directions. Since the turbulence in this case exists only close to the bathymetry, the fine level grid is mostly confined between $x/l = -1$ and $x/l = +1$ in the stream-wise direction, and $z < 3h$ in the vertical direction. However, in the spanwise direction, the fine level grid extends along the whole domain from y_{min} to y_{max}. On an average, the fine level occupies less than 1.95 % of the domain to resolve the nonlinear features in this case, thus saving significant computational cost (More detailed analysis on the computational cost is given in section 4).

The baroclinic energy budget quantifies the conversion from background barotropic tidal energy into baroclinic energy and the partition of the latter between local dissipation and radiated wave flux away from the generation site (The algorithm to calculate baroclinic energy budget on two-level SOMAR grids is described in Appendix A). We compare the results from SOMAR-LES with the numerical experiments of Rapaka and Sarkar (2016) (referred to as RS16), who performed both DNS and LES of the same setup using the Immersed Boundary Method (IBM) to handle the topography. The IBM implementation of Rapaka and Sarkar (2016) employs a structured Cartesian grid with nonuniform grid spacing and a sharp-interface formulation (Mittal et al., 2008) for incorporating the velocity and temperature boundary conditions at the solid-fluid boundary.

The energy conversion C increases as a function of steepness parameter γ as shown previously, e.g Petrelis et al. (2006); Echeverri et al. (2009); Rapaka et al. (2013) and takes values within the range reported in those studies. The present geometry and flow parameters match cases simulated by Rapaka and Sarkar (2016). The overall agreement of the baroclinic budget statistics with the results of Rapaka and Sarkar (2016) is found to be good as shown by Table

Case	$\boldsymbol{C}$	$\boldsymbol{C_{nl}}$	$\boldsymbol{M}$	$\boldsymbol{M_{adv}}$	$\boldsymbol{\varepsilon_{bc}}$	$\boldsymbol{P}$	**Residual**	$\boldsymbol{q \equiv 1 - M/C}$
SUB1	0.678	0.001	0.676	-	.001	.001	-0.0006	0.003
RS16	0.689	-	0.691	-	-	-	-0.022	-.002
SUP1	0.985	-	0.917	-	-	-	-0.036	0.069
RS16	1.009	-	0.962	-	-	-	0.02	0.046
$\mathbf{CRIT1_{ls}}$	0.789	-	0.714	-	-	-	-0.041	0.095
RS16	0.704	-	0.645	-	-	-	-0.052	0.084
$\mathbf{CRIT2_{ls}}$	0.736	0.034	0.637	0.069	0.025	0.036	-0.0025	0.13
RS16	0.721	-	0.612	-	-	-	-0.098	0.15

Table 4: Baroclinic budget comparison for all cases. Budget statistics are obtained by averaging over 3 tidal cycles for cases $SUB1, SUP1$, 2 tidal cycles for $CRIT1_{ls}$ and 1 tidal cycle for case $CRIT2_{ls}$. All quantities reported in this table have units of [m^2/s^3] except 'q', which is a nondimensional quantity. Only quantities that are larger than the absolute value of the residual are shown.

4. In the subcritical case $SUB1$, baroclinic energy conversion(C), and radiated wave flux(M) agree quite well. In the supercritical case $SUP1$, the baroclinic energy conversion C is higher than in the subcritical case and matches well with RS16. Among the large scale cases, case $CRIT1_{ls}$ has slightly higher conversion and wave flux compared to RS16 while turbulent production is in very good agreement. For the critical case $CRIT2_{ls}$ we have excellent agreement with RS16. Residual term, which is the difference between left hand side and right hand side of equation A.4, represents how well the baroclinic energy budget is balanced. Residual is small in all test cases indicating that the budget is well balanced. Nonlinear conversion C_{nl} and advective flux M_{adv} are negligible in all test cases except in the case $CRIT2_{ls}$ (table 4).

3.2. Tidal flow past a three-dimensional Gaussian ring

In this test case, we consider a three-dimensional Gaussian ring bathymetry, shown in figure 9. The steepness parameter is $\gamma = \tan\beta/\tan\alpha$ where the bathymetry has angle β and the internal wave characteristic has angle α with respect to the horizontal. The steepness parameter γ varies over the slope with $\gamma_{max} = 1.1$. Due to the three-dimensional nature of the bathymetry and the resonant interaction of barotropic tide and the bathymetry, the internal tidal beams focus at a point above the generation region leading to velocity intensification and subsequently wave breaking (Buhler and Muller, 2007).

This test case demonstrates that the unique capabilities of SOMAR-LES such as adaptive refinement and turbulence modeling can enable the identification and resolution of remote mixing by internal tides.

Parameter	**Value**	**Remark**
U_0	0.008 m/s	barotropic tidal amplitude
Ω	0.04 rad/s	forcing frequency
N_∞^2	0.0049 s^{-2}	background stratification
h	0.5 m	height of the topography
$\Delta x_C[\Delta x_F]$	0.156 [0.039] m	x-direction grid spacing
$\Delta y_C[\Delta y_F]$	0.156 [0.039] m	y-direction grid spacing
$\Delta z_C[\Delta z_F]$	0.14 [0.035] m	z-direction grid spacing
$L_x \times L_y \times L_z$	20 m $\times$ 20 m $\times$ 9 m	domain size

Table 5: Simulation parameters for the tidal flow past a three-dimensional topography in the shape of a circular ring with a Gaussian cross-section. The simulations are performed with a molecular viscosity of $1 \times 10^{-6} m^2/s$ and the ratio of molecular viscosity to diffusivity equal to 7. The coarse grid is uniform and has $128 \times 128 \times 64$ grid points in the x, y and z directions, respectively.

3.2.1. Forcing and boundary conditions

Tidal forcing with amplitude U_0 is applied by adding a forcing term to the momentum equations (i.e., eq. 2b or 1b). Since the objective of this simulation is to demonstrate the remote mixing due to internal wave focusing, free slip boundary conditions for the velocity are imposed at the bottom boundary to minimize fine-scale shear and turbulence near the generation region. No-flux boundary conditions are applied on the buoyancy at the bottom boundary. Parabolic sponge forcing is applied to the momentum equations to absorb the internal wave energy radiated outside the domain of interest.

3.2.2. Results

Figure 10 shows xz-plane snapshots of the stream-wise (x-direction) velocity field at different time instances. The coarse level grid spans the entire domain, whereas the fine level grid is adaptively created in localized regions where the gradient Richardson number Ri_g drops below 2.0. Compared to earlier test cases, we use a higher threshold of Ri_g to ensure that the fine level grids encompass the internal wave beams all the way up to the focusing region. We found that if we do not resolve the internal wave beams as they propagate, wave beam intensification will be compromised, leading to the reduction or elimination of turbulence in the focusing region. The simulation parameters and grid resolutions are given in table 5. When the barotropic tide interacts with the topography, resonant interaction between the critical region of the topography and the tide generates intensified tidal beams. The axisymmetry of the obstacle is imprinted on the generated wave field. For example, in figure 10, which shows the x-direction velocity field on an xz-plane, internal tides are generated at $x = -5$ and $+5$ m. These tides which are generated on the opposite ends of the circular ring, radiate away from the generation site and focus at a point above the bathymetry. The adaptive refinement feature of SOMAR ensures that the fine level grids track the internal tide as it moves away from the generation region. As shown in figure 10, the fine level domain (shaded regions) evolves as the internal waves radiate away from the generation region. Initially, at time

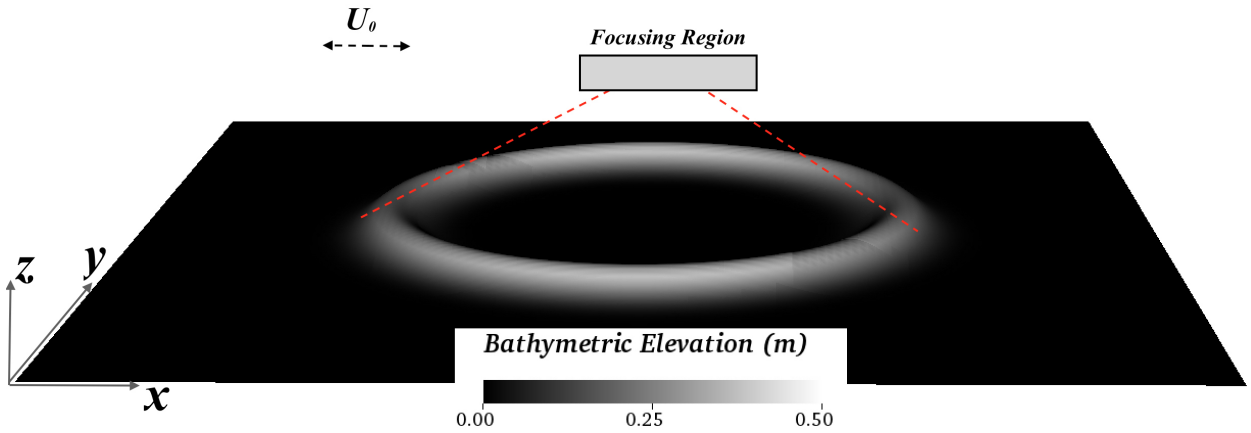


Figure 9: Schematic of the geometry and wave focusing. The pseudocolor plot show a prospective view of the bottom geometry, color coded with the elevation of the topography. Red colored lines show the approximate beam path. The beams focus in the shaded region creating overturning and turbulence.

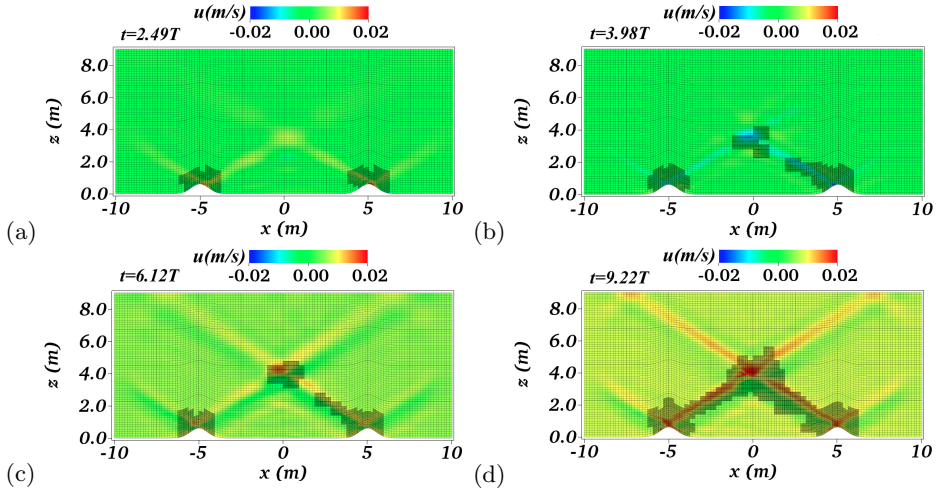


Figure 10: Time evolution of the x-direction velocity field as the internal tides approach the focusing region on a plane parallel to the direction of the forcing barotropic flow. Shaded regions represent the fine level mesh which evolves in time as internal tides propagate away from the generation site and focus.

$t = 2.49T$ (T being the period of the barotropic forcing) shown in figure 10(a), the fine level exists very close to the bathymetry. At a later time, when the internal tides form and radiate away from the generation site, the fine level grid evolves in such a way that it envelopes the internal tides up to the focusing point.

SOMAR-LES envelopes the propagating beams with high-resolution grids: Figure 11(a) shows internal wave focusing occurring at a point approximately 4 m above the topography. Note that only the region covered by the high-resolution grid is shown. The velocity is intensified in the focusing region, leading to density overturns and fine-scale shear. This results in enhanced values of the calculated subgrid-viscosity (figure 11(b)) near the generation site and where the beams converge. Near the topography, the resonant interaction of the tide at the near-critical region of the topography leads to velocity intensification, which creates high shear and turbulence. However, the most striking phenomenon in this flow occurs when internal tides focus at a point above the topography. As shown in

figure 11(b), at $\approx z = 4$ m, the subgrid scale viscosity is elevated by more than an order of magnitude compared to the background molecular viscosity. The buoyancy contours (represented by black lines) show density overturns near the focusing point.

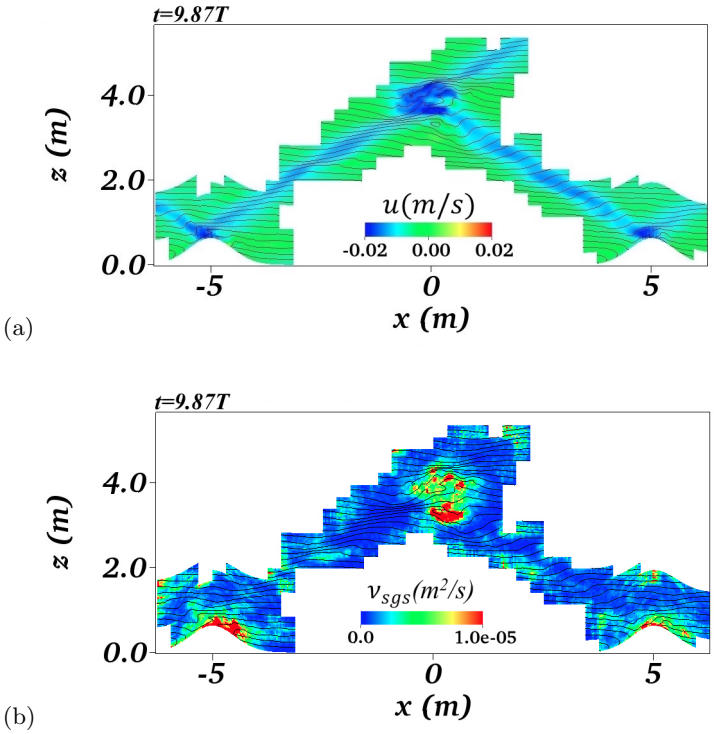


Figure 11: Snapshots of the x-direction velocity and subgrid scale viscosity on the fine level grid at time $t = 9.87T$ on the same plane shown in figure 10. Note that only the regions covered by the fine level grid are shown.

3.3. Low mode wave scattering

So far we have considered internal waves that were generated by the interaction of a barotropic flow with a topographic relief. Of course, SOMAR-LES is not limited to this type of problems. In this section, we consider the scattering of a mode-1 internal wave by an isolated topographic ridge. This example shows how propagating internal waves can be added via properly chosen bound-

ary forcing. The problem has been studied with a traditional single-level solver by Legg (2014). Thus we can compare our results, and also contrast the computational cost of our adaptive grid approach relative to a traditional single level solver. For a linearly stratified fluid, a mode-1 wave is introduced at the left boundary with streamwise velocity of the form (Legg and Adcroft, 2003)

$$u = u_0 \cos(mz - mH_0) \sin(kx - \Omega t), \tag{8}$$

where u_0 is the amplitude, k and m are horizontal and vertical wave numbers, H_0 is the water depth away from the topographic relief, Ω is the frequency of the mode-1 wave and t is the dimensional time. For a mode-1 wave with vertical wavenumber $m = \pi/H_0$ and streamwise wavenumber $k = sm$, the wave characteristic slope s is given by:

$$s = \sqrt{\frac{\Omega^2 - f^2}{N^2 - \Omega^2}}, \tag{9}$$

where N is the buoyancy frequency and f is the Coriolis frequency.

Two test cases are considered in this study. The first case has a two-dimensional bathymetry with no variation in the spanwise direction. The shape of the bathymetry is sinusoidal, expressed as

$$h = \begin{cases} \frac{h_0}{2}\{1 + \cos[2\pi(x - 3\lambda_x)/(2L) - \pi]\}, & \text{if } 3\lambda_x < x < 3\lambda_x + 2L \\ 0, & \text{otherwise} \end{cases}$$

where h_0 is the maximum height of the bathymetry, $\lambda_x = 2\pi/k$ is the streamwise wavelength and L is the half-width of the bathymetry. The second test case considers a three-dimensional bathymetry expressed as a solid of revolution

$$h = \begin{cases} \frac{h_0}{2}\{1 + \cos[2\pi(d - L)/(2L) - \pi]\}, & \text{if } d < L \\ 0, & \text{otherwise} \end{cases}$$

where $d = \sqrt{(x - (x_1 + L))^2 + (y - (y_1 + L))^2}$, $x_1 = 3\lambda_x$ and $y_1 = x_1/2$. These two cases correspond to H1000crit and H3000crit3D of Legg (2014), respectively. The bathymetry is characterized by the steepness parameter $\gamma =$

$(dh/dx)/s$, given by the ratio of topographic slope to wave characteristic slope. In both cases, the maximum steepness parameter γ_{max} is equal to 1, which corresponds to a critical slope. Another important non-dimensional parameter in this case is the incoming wave Froude number defined by $Fr_0 = u_0/C_p$, which is the ratio of the amplitude of the incoming wave velocity (in the x-direction) u_0 to the incoming wave phase speed (in the x-direction) $C_p = \Omega/k$. Rewriting the Froude number using the definitions of C_p, k, m, we obtain the equation for incoming wave Froude number $Fr_0 = \dfrac{u_0 \pi s}{\Omega H_0}$, which is a measure of nonlinearity of the incoming wave.

Free slip boundary conditions are specified for horizontal velocities (u,v) at the top and bottom boundary. Sponge forcing is applied at the right boundary to absorb transmitted waves. A rigid lid boundary condition is applied at the top and bottom boundaries. Periodic boundary conditions are used in the spanwise direction. A no-flux boundary condition is applied for buoyancy at the bottom boundary.

Case	$\mathbf{u_0}(\mathbf{m/s})$	$\mathbf{h_0}, \mathbf{L}(\mathbf{m})$	$\mathbf{N^2_\infty}(\mathbf{s^{-2}})$	$\mathbf{f}(\mathbf{s^{-1}})$	**Fr**	Remark
H1000crit	0.12	1000, 10550	6.4×10^{-7}	0.00008	0.0839	2D bathymetry
H3000crit3D	0.12	3000, 31950	6.4×10^{-7}	0.00008	0.0839	3D bathymetry

Table 6: Simulation parameters for different cases considered to study low mode wave interaction with isolated topography. Background molecular viscosity and diffusivity are set to $1 \times 10^{-4} m^2/s$ in the horizontal directions and $1 \times 10^{-5} m^2/s$ in the vertical direction in order to be consistent with simulations performed by Legg (2014). Incoming mode-1 wave frequency Ω is given by 1.41×10^{-4} rad/s and streamwise wavelength $\lambda_x = 63753$ m.

Case	$\mathbf{N_x}$	$\mathbf{N_y}$	$\mathbf{N_z}$	$\mathbf{dx(m)}$	$\mathbf{dy(m)}$	$\mathbf{dz(m)}$
H1000crit	1024	256	64	436.67 (109.16)	500.2 (125)	73.44 (18.36)
H3000crit3D	1024	512	64	436.67 (109.16)	498.07 (124.52)	73.44 (18.36)

Table 7: Grid parameters. Stream-wise domain length $L_x = 7\lambda_x$ and vertical domain length outside the bathymetry region $H_0 = 4700$ m. Spanwise domain length $L_y = 2\lambda_x$ in case 1 and $4\lambda_x$ in case 2.

In all cases, the mode-1 wave introduced along the left boundary has a Froude number of $Fr_0 = 0.0839$. As the wave approaches the ridge, owing to the fact that fluid depth varies above the bathymetry, conservation of the energy flux requires that the Froude number changes. Assuming that the wave retains the mode-1 structure, the Froude number at a particular location can be written as $Fr = \dfrac{u\pi s}{\Omega H}$, where u and H are velocity amplitude and fluid depth at that location. As the wave approaches the left face of the ridge, the fluid depth H decreases and a result the velocity amplitude and Froude number increase leading to nonlinear interactions with the sloping topography. Legg (2014) argues that for critical slopes (i.e when $\gamma_{max} = 1$), wave breaking and turbulence occurs at all incoming wave Froude numbers. Also, the fraction of the wave energy that will be transmitted is estimated to be $1 - h_0/H_0$.

Figure 12(a) shows the streamwise velocity field at time $t = 7T$ ($T = 2\pi/\Omega$ being the period of the incoming mode-1 wave). The collection of fine level grids, created in the regions where $Ri_g < 0.25$, are also shown in figure 12(a). As the incident wave interacts with the topography, the simple mode-1 structure is distorted, as some energy is transferred to higher modes, characterized by multiple zero-crossings along a vertical profile (figure 12(a)). At this particular moment, nonlinear wave breaking and turbulence is concentrated on the stoss (left) side of the ridge as evident from density overturns in figure 12(b). Regions where density overturns prominently occur coincide with regions where the gradient Richardson number Ri_g drops below 0.25 as shown in figure 12(b),

and thus are covered with fine-level grid cells.[2] The position of the grid cells follows the evolution of the turbulent region as the wave moves over and past the ridge.

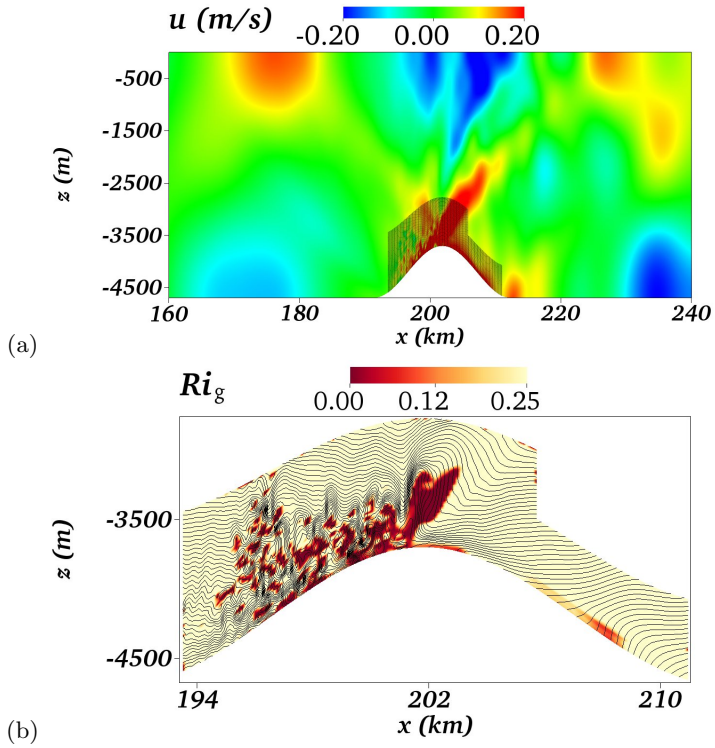


Figure 12: (a) H1000Crit: xz-plane ($y = L_y/2$) snapshot of horizontal velocity field u at $t = 7T$; the shaded area shows the location of the fine-level grid at this particular time. (b) Gradient Richardson number (Ri_g) and buoyancy isocontours on the fine level at $t = 7T$. Note the asymmetry in the fine-grid distribution which reflects the asymmetric distribution of turbulent features at this particular time during the cycle.

750 To give a more quantitative comparison with Legg (2014) results, we consider the energy budget. Note that since we use SOMAR-LES to solve for the buoyancy deviation b^*, we consider the available potential energy, rather than

[2]In all the figures shown in this section, vertical coordinate z is translated by $-H_0$ in order to do a qualitative comparison of our results with Legg (2014).

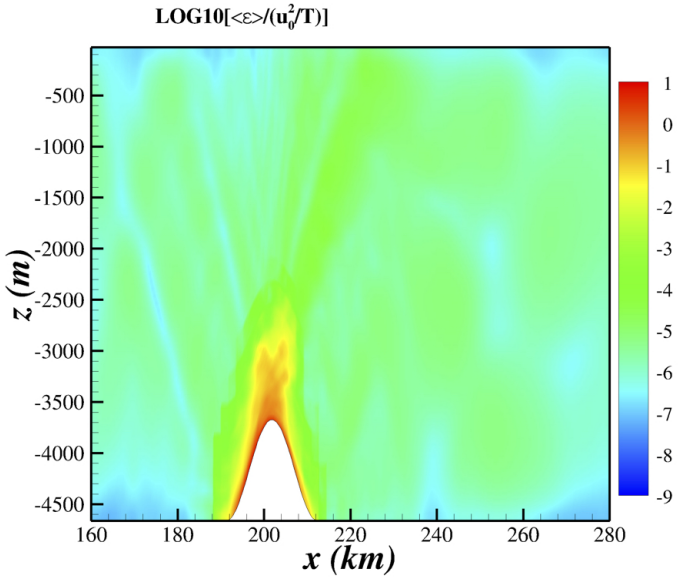


Figure 13: H1000Crit: xz-plane ($y = L_y/2$) snapshot of $LOG10\Big[\langle\varepsilon\rangle/(u_0^2/T)\Big]$, where $\langle\varepsilon\rangle$ is cycle averaged dissipation.

the total potential energy in the budget (for details, see Scotti and White, 2014). Assuming a constant background stratification N_∞, the volume integrated energy equation is given by

$$\frac{\partial}{\partial t}\langle KE+APE\rangle = (Adv+P_{trans}+D)_{x_1}-(Adv+P_{trans}+D)_{x_2}-\varepsilon-\varepsilon_{APE}, \quad (10)$$

where the kinetic and available potential energy sum is

$$\langle KE+APE\rangle = \int_V \frac{1}{2}(u^2+v^2+w^2)dV + \int_V \frac{b^{*2}}{2N_\infty^2}dV, \quad (11)$$

the horizontal divergence of advective, pressure and diffusion transport terms are

$$Adv = \int_{-H}^{0}\int_{0}^{L_y}(KE+APE)u\,dydz, \quad (12)$$

$$P_{trans} = \int_{-H}^{0}\int_{0}^{L_y}p^*u\,dydz, \quad (13)$$

$$D = \int_{-H}^{0}\int_{0}^{L_y}\left[(\nu+\nu_{sgs})\frac{\partial KE}{\partial x}+(\kappa+\kappa_{sgs})\frac{\partial APE}{\partial x}\right]dydz, \quad (14)$$

and the dissipation rate of kinetic energy ε and available potential energy ε_{APE} are given by

$$\varepsilon = \int_V \left\{ \nu \frac{\partial u_i}{\partial x_j} \frac{\partial u_i}{\partial x_j} - \tau_{ij} \overline{S}_{ij} \right\} dV$$

$$\varepsilon_{APE} = \int_V (\kappa + \kappa_{sgs}) \frac{|\nabla b^*|^2}{N_\infty^2} dV.$$

We use V to represent the volume of integration with limits $(x_1, 0, -H)$ to $(x_2, L_y, 0)$, where $x_1 = 3\lambda_x$ and $x_2 = 3\lambda_x + 2L$. There is an additional vertical transport term at the top boundary that is considered in Legg (2014). However, in SOMAR-LES, this vertical transport term is zero due to the rigid lid boundary condition applied at the top boundary as opposed to the free surface boundary condition used in Legg (2014). This vertical transport at the top boundary is found to be negligible in Legg (2014), thus, excluding this term in SOMAR-LES does not change any conclusions drawn from the energy budget analysis. To be consistent with Legg (2014), we calculate statistics by time averaging over two wave periods. Figure 13 shows the cycle-averaged dissipation normalized by u_0^2/T. Energy is dissipated prominently over the ridge and more so towards the stoss face of the ridge as the incoming wave is incident upon the bathymetry from this side. At steady state, transient terms become negligible, and so do the diffusive term D. The primary balance is thus between the net energy loss and energy transport terms as shown below:

$$\varepsilon + \varepsilon_{APE} \approx (Adv + P_{trans} + D)_{x_1} - (Adv + P_{trans} + D)_{x_2}. \tag{15}$$

Here, $(Adv + P_{trans} + D)_{x_1}$ and $(Adv + P_{trans} + D)_{x_2}$ are the energy transport terms at locations $x = x_1$ and $x = x_2$. When the incoming wave interacts with the bathymetry, some fraction of the energy is reflected back, some fraction of the energy is transmitted past the bathymetry and the rest is lost to dissipation and background mixing. The energy transport at location x_1 includes the combination of incoming and reflected wave energies. To compute the actual incoming wave energy, without the reflection part, an additional simulation

without the bathymetry is performed. Thus, the fraction of energy that is reflected back is given by

$$R = \frac{(Adv + P_{trans} + D)_0 - (Adv + P_{trans} + D)_{x_1}}{(Adv + P_{trans} + D)_0}, \tag{16}$$

where $(Adv + P_{trans} + D)_0$ is the energy transport at location x_1 in the case of flat-bottomed simulation.

The fraction of wave energy that is transmitted past the bathymetry is given by

$$T = \frac{(Adv + P_{trans} + D)_{x_2}}{(Adv + P_{trans} + D)_0}, \tag{17}$$

where $(Adv + P_{trans} + D)_{x_2}$ is the energy transport at a location past the bathymetry. The fraction of incident wave energy that is lost to heat and mixing near the bathymetry is given by

$$(\varepsilon + \varepsilon_{APE})^* = (\varepsilon + \varepsilon_{APE})/(Adv + P_{trans} + D)_0. \tag{18}$$

For case H1000crit, the energy budget terms are computed by averaging over two wave periods. At the steady state, the transient terms dKE/dt and $dAPE/dt$ are very small compared to other terms. A balance is primarily established among the energy transmitted/reflected and the local loss of energy to heat and background mixing. The fraction of energy that SOMAR-LES reflects away from the bathymetry is $R = 0.18$ and the fraction that is transmitted past the bathymetry is $T = 0.78$. These values are in close agreement with the theoretical predictions of $R = h_0/H_0 \approx 0.21$ and $T = 1 - h_0/H_0 \approx 0.79$. However, the fraction of energy that is lost near the bathymetry $(\varepsilon + \varepsilon_{APE})^*$ is 26% smaller than the net energy loss reported by Legg (2014) under the same conditions. We note that the algebraic-equation parameterization of Klymak and Legg (2010) is used in Legg (2014) to compute turbulent viscosity in contrast to the full LES model used here. That parameterization assumes that the Thorpe and Ozmidov scales are equal. However, recent studies (Chalamalla and Sarkar, 2015; Mater et al., 2015; Scotti, 2015; Jalali et al., 2017) show that this

assumption leads to an overestimate of dissipation rate especially when the turbulence is driven by the breaking of internal waves. The smaller dissipation rate in SOMAR-LES compared to Legg (2014) is consistent with the above studies.

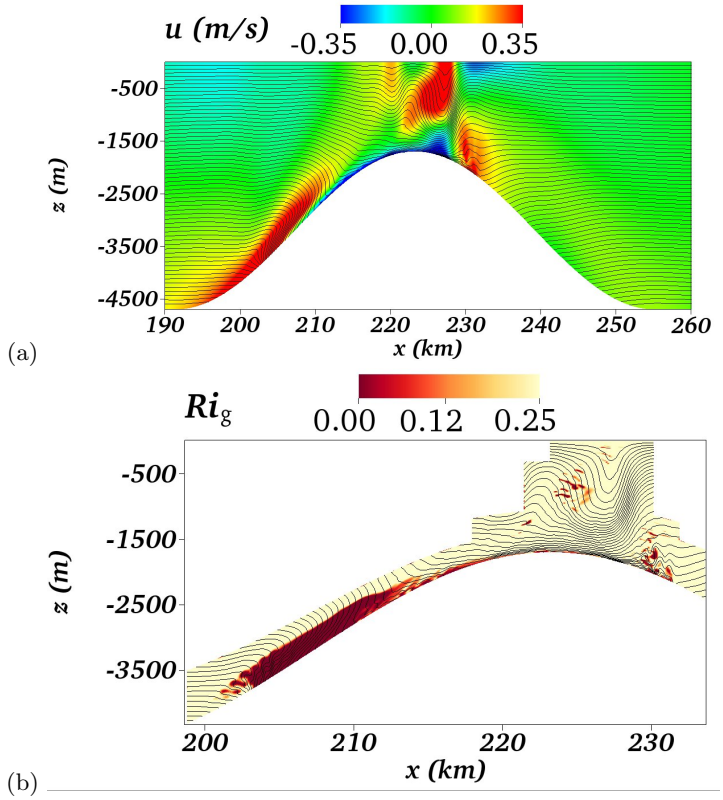


Figure 14: H3000crit3D: (a) xz-plane ($y = L_y/2$) snapshot of streamwise velocity u at t = 6.9T, (b) gradient Richardson number is shown on the fine level domain. Black colored lines in both the sub plots are isocontours of buoyancy.

The second case in this series of numerical experiments (H3000crit3D) considers a three-dimensional bathymetry scattering the same planar internal wave considered in H1000crit. Figure 14(a) shows the streamwise velocity field along with buoyancy isocontours at time $t = 6.9T$. Similar to the two-dimensional bathymetry, the velocity intensifies and high-mode wave beams appear near the bathymetry as shown in figure 14(a). Overall, the solution compares well with Legg (2014) (figure 17a of that paper), i.e both of them show velocity

intensification and formation of small scale features at similar locations near the bathymetry. However, isopycnal displacement and small scale features are more prominent in SOMAR-LES, as seen in figure 14(b), which shows the gradient Richardson number and buoyancy isocontours on the fine level grid. As detailed below when we discuss the computational cost, our effective resolution is finer than Legg's, while at the same time LES is less dissipative than the closure employed by Legg, and thus we expect to observe sharper features in our simulations.

4. Analysis of computational cost

The primary objective in developing the SOMAR-LES model is to resolve both large scale features and model small scale turbulence simultaneously, at a reduced computational cost compared to existing single-level solvers. Quantitative comparison of SOMAR-LES results with previous studies in Section 3 showed good quantitative agreement of baroclinic energy conversion, radiated wave flux, turbulent production and scattering intensities. In this section, the focus is on the computational cost relative to traditional single-level techniques.

In situations where any portion of the domain may, at any given time, host fine-scale structures, single-level solvers must cover the whole domain with a fine grid. With SOMAR-LES, provided fine-scale structures are not "densely" distributed at all times in the domain, fine grids are only deployed where and when necessary. SOMAR-LES is most efficient where the average fraction Λ of the physical domain covered by higher-level grids is not large. Of course, when $\Lambda \to 1$, SOMAR-LES can still be run as a single-level solver, though other, more specialized codes, could outperform it.[3]

Even in those situations where the location of fine-scale structures can be predicted *a priori* and captured with grid stretching techniques, SOMAR-LES

[3]For example, when simulating a turbulent channel flow with periodic conditions in the horizontal, a dedicated channel solver which uses FFTs along x and y to solve the pressure Poisson problem will be faster than SOMAR.

can be more accurate. Indeed, the (often hidden) decrease in accuracy of a stretched grid arises because the global time step is determined by the CFL condition on the grid's smallest cells. This causes the Courant number, defined at each cell by $\sigma = u\Delta t/\Delta x$, to be non-uniform across the domain, leading to increased numerical errors over cells where σ is low. This is because the numerical dissipation of finite volume methods such as PPM is $O((1-\sigma)^p)$, where $p > 0$ is determined by the specific PPM implementation (For details see Durran2010:Lauritzen2007). Thus, cells where σ is not close to 1 experience significant level of numerical diffusion .

In SOMAR-LES, each level runs on its own time-step (this is known as sub-cycling) determined by the CFL condition applied to the level grids. Thus, the base grid can advance on a relatively large time step, and on each level the Courant number can be maintained within tighter limits, minimizing numerical errors. For example, in case $CRIT1_{ls}$, the coarse-level time step Δt is 4 times greater than the fine-level time step. Thus, only a small fraction of the computational domain is impacted by the time step reduction, while the coarse level advances with larger time steps and Courant numbers close to unity.

We now estimate SOMAR's efficiency by comparison with a single-level (non-AMR) solver, and for simplicity we assume a 2-level simulation run over a specific length of time, which we assume to be a tidal period. We begin by defining the following quantities:

$\mathcal{N}_c$ = Number of grid cells needed to span the domain at coarse resolution

$\mathcal{N}_f$ = Number of grid cells needed to span the domain at fine resolution

$\mathcal{N}_{sl}$ = Number of grid cells used by a single-level (non-AMR) solver

$\mathcal{T}_c$ = Number of time steps per tidal cycle at coarse resolution

$\mathcal{T}_f$ = Number of time steps per tidal cycle at fine resolution

Λ = Average fraction of domain covered by fine grids

With these definitions, we can estimate SOMAR's total number of cells updated

per tidal cycle. The computational work per cycle is

$$W_{\text{SOMAR}} \sim \mathcal{N}_c\mathcal{T}_c + \Lambda\mathcal{N}_f\mathcal{T}_f.$$

For test case $CRIT1_{ls}$, the fine grid resolution is 4 times greater than the coarse grid resolution in all directions, so $\mathcal{N}_f = 4^3\mathcal{N}_c$. To maintain stability, the fine grid Δt must be 4 times smaller than that on the coarse grid, so $\mathcal{T}_f = 4\mathcal{T}_c$. With these substitutions, we see that

$$W_{\text{SOMAR}} \sim (1 + 4^4\Lambda)\mathcal{N}_c\mathcal{T}_c.$$

On the other hand, computational work per cycle in IBM (Rapaka and Sarkar, 2016) study of this case was

$$W_{\text{IBM}} \sim \mathcal{N}_{sl}\mathcal{T}_f = 4\mathcal{N}_{sl}\mathcal{T}_c.$$

The IBM's timestep was approximately equal to SOMAR-LES's fine grid timestep because the IBM's grids were stretched in such a way as to make it's finest resolution roughly equal to that of SOMAR-LES's fine level resolution.

In the SOMAR-LES, case $CRIT1_{ls}$ used $\mathcal{N}_c = 1024 \times 16 \times 256$ with $\Lambda \approx 0.0046$. The single-level IBM solver used $\mathcal{N}_{sl} = 896 \times 64 \times 384$ cells. Thus, the ratio of computational work per tidal cycle in SOMAR-LES to the IBM, $W_{\text{SOMAR}}/W_{\text{IBM}} \sim 10\%$.

In the second test case of flow past a three-dimensional obstacle, the internal tides focus at a region far above the generation site, leading to wave breaking and turbulence. Apart from the generation region, which requires fine resolution in order to resolve the nonlinear features during the generation of internal tides, the focusing region above the bathymetry also requires fine grid resolution to resolve wave breaking and turbulence. In summary, the remote mixing process unique to these kind of flows demand finer grid resolution in multiple regions of the computational domain, which cannot be achieved by the existing grid stretching techniques. SOMAR, with its adaptive grids, is able to achieve fine-scale resolutions at multiple, evolving regions of interest at a reduced computational cost compared to the traditional single-level solvers. If we were to run

the same simulation with a single-level solver at the resolution of the fine grid, we would need both more cells per time step and more time steps to maintain stability. In total, the computational work is reduced by nearly 25 times using two-level grids. This effect is made more striking when we consider the fact that the cost of solving Poisson's equation rises nonlinearly with the number of cells in the domain.

The last series of numerical experiments provide an interesting way of comparing our AMR strategy with a computationally mature single-level solver, the MITgcm. To simulate the scattering of an internal wave over a three-dimensional topography, SOMAR-LES uses approximately 50,000 CPU hours to complete 12 tidal periods. For the same test case, Legg (2014) reports that MITgcm uses nearly 200,000 CPU hours.

The effective horizontal grid resolution in SOMAR-LES is nearly three times finer compared to the resolution employed by Legg (2014). Thus, assuming that the MITgcm scales linearly, the cost of running the MITgcm at the effective resolution used in our test case would be over 2 million CPU hours (accounting for the reduction in time step). The order of magnitude increase in efficiency of SOMAR-LES relative to the MITgcm is not solely due to a much more efficient use of grid points[4]. To solve the Poisson problem for pressure, SOMAR-LES uses a V-cycle scheme, whose cost is $O(\mathcal{N}\log(\mathcal{N}))$, where $\mathcal{N}$ is the total number of grid points. In the future, we plan to use a full-multigrid scheme, whose cost is $O(\mathcal{N})$ (see Briggs et al., 2000, for a cost analysis of these multigrid techniques). For comparison, the MITgcm uses a Conjugate Gradient method, whose cost is $O(\mathcal{N}\sqrt{\mathfrak{k}})$. Here, $\mathfrak{k}$ is the condition number of the Laplacian to be inverted. Due to geometric constraints, $\mathfrak{k}$ is typically $O(L_{\text{horiz}}/\Delta z)$ in the ocean, where Δz is the vertical grid spacing and L_{horiz} is the horizontal length scale of the problem. Thus, the Poisson solver of SOMAR-LES deals better with high aspect-ratio

[4]The two codes were run on similar, though not identical, supercomputing clusters. While the difference in hardware may account for some of the performance difference, it is clear that the performance gain in SOMAR-LES is mostly algorithmic.

geometries typical of oceanic simulations.

5. Summary and Discussion

We have demonstrated a new multi-scale modeling technique, SOMAR-LES, and demonstrated its utility when applied to problems involving the interaction of a barotropic and/or baroclinic flows with a nontrivial bottom bathymetry. Two "levels" of grids at coarse and fine resolution are used to perform three-dimensional, turbulence-resolving simulations. The grid resolution on the coarse level is adequate to model the large scale baroclinic flow, but is unsuitable for modeling the effects of small scale turbulence generated during internal tide breaking. Thus, the fine level, which has four times finer grid resolution in all directions, is utilized in localized patches to perform a large eddy simulation that models the small-scale turbulence.

In the first set of test cases, a smoothed triangular bump is considered. Four different simulations are performed with different slope criticality and excursion numbers. Cases $SUB1$ and $SUP1$ are laboratory-scale simulations while cases $CRIT1_{ls}$ and $CRIT2_{ls}$ are at larger oceanic scale. Results from these simulations are compared with our previous DNS and LES. Overall, all the important statistics including conversion, wave flux, the fraction of converted energy dissipated locally, and turbulent production computed using SOMAR-LES agree well with our previous results on a single-level stretched grid. The agreement is excellent in the sub-critical case. In the supercritical case, there is only a 2% difference in the conversion (C) and a 4% difference in the radiated flux (M). Differences in the critical-slope cases range from 2 % to 10%. The local energy loss increases with obstacle steepness and tidal excursion number to a maximum of 15% in the simulated cases.

Another test case is considered with a three-dimensional bathymetry in the form of a circular ring with a Gaussian cross-section. Previous studies (Buhler and Muller, 2007; Shmakova et al., 2016) have reported instabilities due to internal wave focusing above similar three-dimensional topographies. This unique

flow phenomenon where turbulence is generated at remote locations due to wave breaking, requires that internal tides are well resolved all along their path up to the breaking point near the focusing region. Such a high resolution is computationally very expensive with solvers that use single-level grids. SOMAR-LES, with the capability of adaptive grid refinement and large eddy simulation, is able to track and resolve the internal tides as they move away from their generation site. In this test case, the computational cost savings are shown to be quite substantial.

In the third set of simulations, the scattering of a mode-1 baroclinic wave by an isolated two- and three-dimensional topography is considered. The fraction of the incoming wave energy that is transmitted past the bathymetry agrees well with theoretical estimates and the numerical results of Legg (2014). The fraction of wave energy that is reflected back agrees well with the theoretical estimates, however, it is larger than the numerical results of Legg (2014). This is because the fraction of energy that is lost by SOMAR-LES to dissipation and background mixing near the bathymetry is smaller than the value reported by Legg (2014) due to the less dissipative nature of LES vis-a-vis the turbulence closure employed by Legg (2014). Relative to traditional single-level solvers, we show that the computational cost can be reduced by more than an order of magnitude via adaptive mesh refinement.

In summary, this novel modeling technique is able to accurately model internal waves and turbulence at a reduced computational cost. The magnitude of computational cost savings depends on the flow physics. SOMAR-LES is most effective in reducing the computational cost in flow situations where the location of turbulence is intermittent in space and time. With the help of adaptive refinement features and an LES model, nonlinear features can be tracked and turbulence can be modeled as the flow evolves in both space and time. Thus, SOMAR-LES can be used to tackle problems that so far have been computationally prohibitive.

Acknowledgment

This work was supported in part by NSF grants OCE-1459774 and OCE-1459506 and ONR grant N000141512578. We wish to thank two anonymous referees for their valuable comments.

Appendix A. Flow analysis on data spread over nested grids

Special care must be taken when post-processing data which belongs to grids with different resolution. Failure to properly incorporate the coarse and fine data into post-processing efforts can lead to filtering of high wavenumber modes. In this section, we will describe how the energy budget calculation is performed on the adaptive mesh and provide justification for our algorithmic decisions.

Appendix A.1. Computing energy integrals using adaptive, finite volume data

SOMAR-LES is typically setup to produce multiple levels of data. The coarse level (i.e., level-0), which extends over the entire problem's domain, contains data produced by the large-scale solver. The finer level, which only exists in small, possibly disconnected, regions of the domain, contains data that is more accurate for two major reasons. First, the time-stepper experiences significantly less truncation error. For example, with a uniform refinement ratio of 4, a level-1 solution has at least 16 times less numerical error than the coarser level-0 solution. Second, since the grid is more refined, it can harbor modes that are sub-grid to the coarser level. Since SOMAR-LES uses block-structured adaptive mesh refinement, there is, in a sense, some data duplication. The data that lives on the fine level overshadows less resolved data on the coarse level. This less-resolved data is often called *invalid data* since it is not as accurate as its fine-level version, but this invalid data still serves a significant purpose. Each invalid cell contains the average of all of the fine-level cells above it. This makes line and volume integration significantly simpler – whatever needs to be integrated can be averaged down to the coarse level, where we can sum over any sub-domain with impunity. Since the coarse level extends over the entire

domain, we will not run into missing regions of data as we would on the fine level. Integrals carried out in this way are guaranteed to be as accurate as possible since they effectively use the most accurate data available.

965

This "average then sum" method does not work for quantities that are not linearly related to the velocity and buoyancy fields, such as the kinetic energy. Averaging the velocity to the coarse level before squaring leads to a filtering of the high wavenumber modes. Consider a single, 1D coarse grid cell containing a single velocity value, u_C. Above it, on a grid refined by two, are two fine grid cells containing velocity values $u_{F,l}$ and $u_{F,r}$. If we attempt to compute an energy integral on the coarse level, we get

$$\begin{aligned}\int E_c\,dx &= \frac{1}{2}u_C^2\Delta x_C \\ &= \frac{1}{2}\left(\frac{u_{F,l}+u_{F,r}}{2}\right)^2\Delta x_C \\ &= \frac{1}{2}u_{F,l}^2\Delta x_F + \frac{1}{2}u_{F,r}^2\Delta x_F - \frac{1}{8}\left(u_{F,l}-u_{F,r}\right)^2\Delta x_C \\ &\leq \frac{1}{2}u_{F,l}^2\Delta x_F + \frac{1}{2}u_{F,r}^2\Delta x_F \equiv \int E_F\,dx,\end{aligned}$$

where we used the identities $\Delta x_C = 2\Delta x_F$ and $u_C = (u_{F,l}+u_{F,r})/2$. Notice that the energy that is filtered out, $(u_{F,l}-u_{F,r})^2\Delta x_C/8$ is of the form $\nu(\partial u_F/\partial x)^2$. This is a viscous dissipation as derived in Kundu et al. (2012, sections 4.8 and 11.10), with an effective viscosity of $\Delta x_C^3/32$. To correctly compute the integral
970 of the energy, or any other derived quantity for that matter, we must first compute the energy function on all levels, then average the energy function down rather than the velocity, then integrate.

Appendix A.2. Baroclinic Energy Budget Equations

In order to quantify the energy conversion from barotropic to baroclinic motions in tidal flow over two-dimensional obstacles, the velocity and pressure fields are split into barotropic (capitals), baroclinic (subscript *bc*), and three-

dimensional fluctuation (primed) fields,

$$u(x,y,z,t) = U(x,t) + u_{bc}(x,z,t) + u'(x,y,z,t) \tag{A.1}$$

$$w(x,y,z,t) = W(x,z,t) + w_{bc}(x,z,t) + w'(x,y,z,t) \tag{A.2}$$

$$p^*(x,y,z,t) = P^*(x,t) + p_{bc}(x,z,t) + p'(x,y,z,t), \tag{A.3}$$

where p^* stands for the deviation from hydrostatic pressure. The obstacle extends along the x direction, uniform along the y (while the averaged flow is two-dimensional, turbulent fluctuations are three-dimensional). First, the fluctuations (primed quantities) are removed by computing the Reynolds average of each state variable. Then, U and P^* are calculated by depth averaging the corresponding Reynolds average. The vertical barotropic component is computed via $W = -\frac{\partial}{\partial x}([z - h(x)]\,U)$, so that the barotropic velocity field remains divergence-free. The terms u_{bc}, w_{bc}, and p_{bc} are the remaining baroclinic components. We use the baroclinic energy equation used in several previous studies (Cummins and Oey, 1997; Kurapov et al., 2003; Kang and Fringer, 2012; Buijsman et al., 2012; Rapaka et al., 2013; Jalali et al., 2014; Rapaka and Sarkar, 2016) to compute the budget,

$$\frac{\partial}{\partial t}(E_k + E_p) = C + C_{nl} - \nabla \cdot \boldsymbol{F} - \varepsilon_{bc} - P. \tag{A.4}$$

Here, E_k and E_p are baroclinic kinetic energy and available potential energy respectively. C is the energy conversion from the barotropic tide to baroclinic motions. C_{nl} is the nonlinear conversion. Flux term ($\nabla \cdot F = M + M_{adv}$), is the sum of wave flux M and advective flux M_{adv}. Detailed discussion of the baroclinic energy budget and the importance of each term is beyond the scope of this paper, and is available in the literature cited above. At steady state, cycle-averaged values of the transient terms $\frac{\partial}{\partial t}(E_k + E_p)$ are negligible. The energy conversion ($C + C_{nl}$) is in balance with radiated wave flux (M), the advective flux (M_{adv}), the baroclinic dissipation (ε_{bc}) and the turbulent production (P). Conversion C and radiated wave flux M are the dominant terms. Also, the fraction of energy that is dissipated locally at the generation site is given by $q = 1 - M/C$.

Appendix A.3. The Baroclinic Energy Analysis Algorithm

An algorithm for computing the baroclinic energy budget of a 3D, adaptive, finite-volume dataset is required. Suppose we have the velocity components, u, v, and w, hydrostatic buoyancy deviation, b^*, and hydrostatic pressure deviation, p^*, at many time steps throughout several tidal cycles of interest. The first thing we do is to average fine-level data down to the coarse level to ensure the invalid data properly represents the finer data. Next, we compute the barotropic velocity and pressure as described in Appendix A.2. This requires vertical integration, which means we must perform this decomposition on the coarse level then interpolate the results up to the finer level. Although this seems to fly in the face of our "derive *then* average down" rule from Appendix A.1, it is appropriate in this case since the barotropic fields are large scale fields and do not have fine-grained, localized details. Note that we also compute the vertical barotropic velocity component in this way, computing W on the coarse level then interpolating to the fine level. We use the conservative, limited, piecewise trilinear interpolation scheme described in the Chombo design document (Adams et al., 2011).

With the barotropic velocity and pressure available on all levels, we can compute the baroclinic velocity and pressure components, pressure flux, dissipation, conversion, and production terms of the energy budget, then average down to the coarsest level and spatially integrate over the region of interest. In short, the procedure consists of the following 7 steps.

1. Average down to correct the coarse level.
2. Compute barotropic components on the coarse level and interpolate up.
3. Separate the barotropic from the baroclinic components on all levels.
4. Compute derived quantities on all levels.
5. Average derived quantities down to coarse level.

6. Perform spatial integration of derived quantities on coarse level.

7. Perform time series analysis on results.

As a final note, whenever we need the divergence of a derived quantity (for example, when computing the wave flux term, $\nabla \cdot F$), we compute the flux F in step 4 on all levels, then apply the divergence on the coarse level in step 6. Due to the commutativity of sums and divergences, this is identical to computing the divergence on all levels before averaging down, but much simpler to implement numerically.

References

Adams, M., Colella, P., Graves, D.T., Johnson, J., Keen, N., Ligocki, T.J., Martin, D.F., McCorquodale, P., Modiano, D., Schwartz, P., Sternberg, T., Straalen, B.V., 2011. Chombo software package for AMR applications - Design document. Lawrence Berkeley National Laboratory Technical Report LBNL-6616E .

Alford, M.H., MacKinnon, J.A., Nash, J.D., Simmons, H., Pickering, A., Klymak, J.M., Pinkel, R., Sun, O., Rainville, L., Musgrave, R., Beitzel, T., Fu, K., Lu, C., 2011. Energy flux and dissipation in Luzon Strait: Two tales of two ridges. J. Phys. Oceanogr. 41, 2211–2222.

Berger, M.J., Oliger, J., 1984. Adaptive mesh refinement for hyperbolic partial differential equations. J. Comp. Phys. 53, 484 – 512.

Blayo, E., Debreu, L., 1999. Adaptive mesh refinement for finite-difference ocean models: First experiments. J. Phys. Oceanogr. 29, 1239–1250. http://dx.doi.org/10.1175/1520-0485(1999)029<1239:AMRFFD>2.0.CO;2.

Blayo, E., Debreu, L., 2006. Nesting ocean models, in: Ocean Weather Forecasting. Springer, pp. 127–146.

Brandt, T., 2005. A posteriori study on modeling and numerical error in LES applying the Smagorinsky model. Lect. Notes Comput. Sci. Eng. 56, 173–189.

Briggs, W., Henson, V., McCormick, S., 2000. A Multigrid Tutorial, Second Edition. Society for Industrial and Applied Mathematics. second edition. http://epubs.siam.org/doi/pdf/10.1137/1.9780898719505.

Buhler, O., Muller, C.J., 2007. Instability and focusing of internal tides in the deep ocean. J. Fluid Mech. 588, 1–28.

Buijsman, M.C., Legg, S., Klymak, J., 2012. Double-ridge internal tide interference and its effect on dissipation in Luzon Strait. J. Phys. Oceanogr. 42, 1337–1356.

Carter, G.S., Merrifield, M.A., Becker, J.M., Katsumata, K., Gregg, M.C., Luther, D.S., Levine, M.D., Boyd, T.J., Firing, Y.L., 2008. Energetics of M_2 barotropic-to-baroclinic tidal conversion at the Hawaiian Islands. J. Phys. Oceanogr. 38, 2205–2223.

Chalamalla, V.K., Hurley, M.W., Passaggia, P.Y., Scotti, A., 2016. Internal wave focusing above a three dimensional topography. APS-DFD https://doi.org/10.1103/APS.DFD.2016.GFM.P0027.

Chalamalla, V.K., Hurley, M.W., Passaggia, P.Y., Scotti, A., 2017. Experimental study of internal wave focusing above a three dimensional topography. *(In preperation)* .

Chalamalla, V.K., Sarkar, S., 2015. Mixing, dissipation rate, and their overturn-based estimates in a near-bottom turbulent flow driven by internal tides. J. Phys. Oceanogr. 45, 1969–1987.

Cole, S.T., Rudnick, D.L., Hodges, B.A., Martin, J.P., 2013. Observations of tidal internal wave beams at Kauai Channel, Hawaii. J. Phys. Oceanogr. 39, 421–436.

Colella, P., 1990. Multidimensional upwind methods for hyperbolic conservation laws. J. Comp. Phys. 87, 171–200.

Colella, P., Woodward, P.R., 1984. The Piecewise Parabolic Method (PPM) for Gas-Dynamical Simulations. J. Comp. Phys. 54, 174–201.

Courant, R., Friedrichs, K., Lewy, H., 1928. Über die partiellen differenzengleichungen der mathematischen physik. Mathematische Annalen 100, 32–74.

Cummins, P.F., Oey, L.Y., 1997. Simulation of barotropic and baroclinic tides off Northern British Columbia. J. Phys. Oceanogr. 27, 762–781.

Deardorff, J.W., 1970. A numerical study of three-dimensional turbulent channel flow at large Reynolds numbers. J. Fluid Mech. 41, 453–480.

Debreu, L., Marchesiello, P., Penven, P., Cambon, G., 2012. Two-way nesting in split-explicit ocean models: algorithms, implementation and validation. Ocean Modell. 49, 1–21.

Ducros, F., Comte, P., Lesieur, M., 1996. Large-eddy simulation of transition to turbulence in a boundary layer developing spatially over a flat plate. J. Fluid Mech. 326, 1–36.

Echeverri, P., Flynn, M.R., Winters, K.B., Peacock, T., 2009. Low-mode internal tide generation by topography: an experimental and numerical investigation. J. Fluid Mech. 636, 91–108.

Fringer, O., Gerritsen, M., Street, R., 2006. An unstructured-grid, finite-volume, nonhydrostatic, parallel coastal ocean simulator. Ocean Modell. 14, 139 – 173.

Germano, M., Piomelli, U., Moin, P., Cabot, W.H., 1991. A dynamic subgrid-scale eddy viscosity model. Phys. Fluids A 3, 1760–1765.

Gostiaux, L., Dauxois, T., 2007. Laboratory experiments on the generation of internal tidal beams over steep slopes. Phys. Fluids 19, 028102.

Holloway, P.E., Merrifield, M.A., 1999. Internal tide generation by seamounts, ridges, and islands. J. Geophys. Res. 104, 25937–25951.

Howard, L.N., 1961. Note on a paper of John W. Miles. J. Fluid Mech. 10, 509–512.

Jalali, M., Chalamalla, V.K., Sarkar, S., 2017. On the accuracy of overturn-based estimates of turbulent dissipation at rough topography. J. Phys. Oceanogr. 47, 513–532.

Jalali, M., Rapaka, N.R., Sarkar, S., 2014. Tidal flow over topography: effect of excursion number on wave energetics and turbulence. J. Fluid Mech. 750, 259–283.

Kang, D., Fringer, O., 2012. Energetics of barotropic and baroclinic tides in the Monterey Bay area. J. Phys. Oceanogr. 42, 272–290.

Klymak, J.M., Legg, S.M., 2010. A simple mixing scheme for models that resolve breaking internal waves. Ocean Modell. 33, 224–234.

Klymak, J.M., Moum, J.N., Nash, J.D., Kunze, E., Girton, J.B., Carter, G.S., Lee, C.M., Sanford, T.B., Gregg, M.C., 2006. An estimate of tidal energy lost to turbulence at the Hawaiian Ridge. J. Phys. Oceanogr. 36, 1148–1164.

Kundu, P., Cohen, I., Dowling, D., 2012. Fluid Mechanics. Academic Press.

Kurapov, A.L., Egbert, G.D., Allen, J.S., Miller, R.N., Erofeeva, S.Y., Kosro, P.M., 2003. The M2 internal tide off Oregon: Inferences from data assimilation. J. Phys. Oceanogr. 33, 1733–1757.

Laurent, L.C.S., Garrett, C., 2002. The role of internal tides in mixing the deep ocean. J. Phys. Oceanogr. 32, 2882–2899.

Ledwell, J.R., Montgomery, E.T., Polzin, K.L., Laurent, L.C.S., Schmitt, R.W., Toole, J.M., 2000. Evidence for enhanced mixing over rough topography in the abyssal ocean. Nature 403, 179–182.

Legg, S., 2014. Scattering of low-mode internal waves at finite isolated topography. J. Phys. Oceanogr. 44, 359–383.

Legg, S., Adcroft, A., 2003. Internal wave breaking at concave and convex continental slopes. J. Phys. Oceanogr. 33, 2224–2246.

Marshall, J., Jones, H., Hill, C., 1998. Efficient ocean modeling using non-hydrostatic algorithms. J. Mar. Syst. 18, 115 – 134.

Martin, D.F., 1998. An Adaptive Cell–Centered Projection Method for the Incompressible Euler Equations. Ph.D. thesis. University of California at Berkeley.

Mater, B.D., Venayagamoorthy, S., Laurent, L.S., Moum, J., 2015. Biases in thorpe-scale estimates of turbulence dissipation. part I: Assessments from large-scale overturns in oceanographic data. J. Phys. Oceanogr. 45, 2497–2521.

Menemenlis, D., Campin, J.M., Heimbach, P., Hill, C., Lee, T., Nguyen, A., Schodlok, M., Zhang, H., 2008. ECCO2: High resolution global ocean and sea ice data synthesis. Mercator Ocean Q. Newsl. 31, 13–21.

Metzger, E., Hurlburt, H., Xu, X., Shriver, J.F., Gordon, A., Sprintall, J., Susanto, R.v., Van Aken, H., 2010. Simulated and observed circulation in the Indonesian Seas: 1/12 global HYCOM and the INSTANT observations. Dyn. Atmos. Oceans 50, 275–300.

Miles, J.W., 1961. On the stability of heterogeneous shear flows. J. Fluid Mech. 10, 496–508.

Mittal, R., Dong, H., Bozkurttas, M., Najjar, F.M., Vargas, A., von Loebbecke, A., 2008. A versatile sharp interface immersed boundary method for incompressible flows with complex boundaries. J. Comp. Phys. 227, 4825–4852.

Petrelis, F., Smith, S.G.L., Young, W.R., 2006. Tidal conversion at submarine ridge. J. Phys. Oceanogr. 36, 10531071.

Pham, H.T., Sarkar, S., 2014. Large eddy simulations of a stratified shear layer. J. Fluids Eng. 136, 060913.

Pham, H.T., Sarkar, S., Winters, K.B., 2013. Large-eddy simulation of deep-cycle turbulence in an Equatorial Undercurrent model. J. Phys. Oceanogr. 43, 2490–2502.

Piggott, M., Gorman, G., Pain, C., Allison, P., Candy, A., Martin, B., Wells, M., 2008. A new computational framework for multi-scale ocean modelling based on adapting unstructured meshes. Int. J. Numer. Methods Fluids 56, 1003–1015.

Piomelli, U., Moin, P., Ferziger, J.H., 1988. Model consistency in large eddy simulation of turbulent channel flows. Phys. Fluids 31, 1884–1891.

Polzin, K.L., Toole, J.M., Ledwell, J.R., Schmitt, R.W., 1997. Spatial variability of turbulent mixing in the abyssal ocean. Science 276, 93–96.

Popinet, S., Rickard, G., 2007. A tree-based solver for adaptive ocean modelling. Ocean Modell. 16, 224–249.

Rapaka, N.R., Gayen, B., Sarkar, S., 2013. Tidal conversion and turbulence at a model ridge: direct and large eddy simulations. J. Fluid Mech. 715, 181–209.

Rapaka, N.R., Sarkar, S., 2016. An immersed boundary method for direct and large eddy simulation of stratified flows in complex geometry. J. Comp. Phys. 322, 511–534.

Ribault, C.L., Sarkar, S., Stanley, S., 1999. Large eddy simulation of a plane jet. Phys. Fluids 11, 3069–3083.

Rudnick, D.L., Boyd, T.J., Brainard, R.E., Carter, G.S., Egbert, G.D., Gregg, M.C., Holloway, P.E., Klymak, J.M., Kunze, E., Lee, C.M., Levine, M.D., Luther, D.S., Martin, J.P., Merrifield, M.A., Moum, J.N., Nash, J.D., Pinkel, R., Rainville, L., Sanford, T.B., 2003. From tides to mixing along the Hawaiian Ridge. Science 301, 355–357.

Saltzman, J., 1994. An unsplit 3d upwind method for hyperbolic conservation laws. J. Comp. Phys. 115, 153–168.

Santilli, E., 2015. The Stratified Ocean Model with Adaptive Refinement (SOMAR). Ph.D. thesis. University of North Carolina at Chapel Hill Graduate School.

Santilli, E., Scotti, A., 2011. An efficient method for solving elliptic equations on highly anisotropic grids. J. Comp. Phys. 230(23), 8342–8359.

Santilli, E., Scotti, A., 2015. The Stratified Ocean Model with Adaptive Refinement (SOMAR). J. Comp. Phys. 291, 60–81.

Santilli, E., Scotti, A., 2017. UNC-CFD/somar: SOMAR first release.

Sarkar, S., Scotti, A., 2017. From topographic internal gravity waves to turbulence. Ann. Rev. Fluid Mech. 49, 195–220.

Scotti, A., 2015. Biases in thorpe-scale estimates of turbulence dissipation. part II: Energetics arguments and turbulence simulations. J. Phys. Oceanogr. 45, 2522–2543.

Scotti, A., White, B., 2014. Diagnosing mixing in stratified turbulent flows with a locally defined available potential energy. J. Fluid Mech. 740, 114–135.

Shmakova, N., Flór, J.B., Voisin, B., Sommeria, J., Viboud, S., 2016. High stokes number wave focusing by a circular ridge: Internal, inertial and inertia-gravity waves. VIIIth Int. Symp. on Stratified Flows .

Smagorinsky, J.S., 1963. General circulation experiments with the primitive equations, part I: The basic experiment. Mon. Weather Rev. 91, 99–152.

Sreenivasan, K.R., 1995. On the universality of the Kolmogorov constant. Phys. Fluids 7, 2778–2784.

Stamper, M.A., Taylor, J.R., 2017. The transition from symmetric to baroclinic instability in the Eady model. Ocean Dyn. 67, 65–80.

Vreman, B., Geurts, B., Kuerten, H., 1997. Large-eddy simulation of the turbulent mixing layer. J. Fluid Mech. 339, 357–390.