We would like to thank reviewer for providing useful comments/suggestions towards the improvement of our manuscript. In the following responses, the reviewer's questions are printed in black and our responses are in blue. In the revised manuscript, the changes are highlighted in red.

Response to Reviewer 1

The manuscript describes an extension of a popular wave model with the aim to increase applicability and computational speed through multi-grid nesting and MPI parallelization. The paper is well-written and most sections are properly explained. However, there are still a few section, where more information in necessary. This applies mostly to memory sharing, synchonizaion and several other small details, which could really help a less experienced researcher in understanding the details of this presented technique. Below are a few comments, which can help to improve the overall quality and message of this paper.

Reply: We noticed that the major concern from the reviewer was the clearness of the descriptions about the nesting algorithm and techniques associated with data structures and workflow management. In the revised version, we made a major revision in section 3 and tried to better present the detailed techniques used in the master program. We also clarified a number of confusing statements pointed out by the reviewer. According to the reviewer's suggestions, we tried to use plain language and simple examples, which may help users with less experience in computer programming to understand the model nesting procedures in a parallelized system. The following are detailed responses to the reviewer's questions.

Page 4 line 101-104 :

"is 4 or better" meaning refinement factor >=4 or 4<= ?

Reply: In the revision, we reworded the sentence as "grid refinement ratio (. . .) is 4 or smaller. "

Page 7 line 176-183: Any particular reason why the spherical mode solve the weakly nonlinear equations and not the fully nonlinear equations?

Reply: We think this is a very good question. The spherical model solves the weakly nonlinear Boussinesq equations derived by Kirby et al. (2013). The weakly nonlinear equations were developed because a spherical model is usually used in basin-scale applications where the nonlinearity is relatively weak. However, the recent applications in the National Tsunami Hazard Mitigation Program (NTHMP) have shown that it is more convenient to make multi-grid nesting in the single spherical mode, rather than a combination of the spherical and Cartesian modes, for simulations of transoceanic tsunamis and coastal effects in terms of map projections. We think the reviewer made a good point that the spherical mode should be further developed into a fully nonlinear model, consistent with the Cartesian mode. To address the question, we add a short statement in the future work in the conclusion section:

"As mentioned at the beginning of the paper, a combination of the weakly nonlinear spherical mode and the fully nonlinear Cartesian mode has often been used in the one-way coupling nested grid framework for transoceanic tsunami simulations. However, the nesting interface developed in the present study cannot be used for such mixed-mode applications. It is necessary to further develop the spherical mode based on the fully nonlinear Boussinesq equations. This development may be left as future work, noting that any new developments in the model will not interfere with the nesting interface developed here. Future work may also include the development of an interface for the GPU version of FUNWAVE-TVD and of an adaptive mesh refinement algorithm for the nesting framework. "

Page 10 line 225-228: Is it necessary to exchange the dispersive terms: how does it increase the model efficiency.

Reply: The strategy of directly exchanging dispersive terms can increase the model efficiency by avoiding recalculating those dispersive terms inside the ghost cells.

How does the exchange work for the tridiagonal solver (child grid)?

Reply: In a child grid, the tridiagonal solver is performed after the data exchange

in ghost cells. The tridiagonal matrix is solved for the given boundary conditions (u_α, v_α) at the ghost cells. In the revised version, we clarified this in section 3.1.

Page 12 line 275-276: The restriction operator (The update operator) is not detailed. Which parent grid cells are actually updated? What about the ghost cells? What variables are updated in the Parent grid? Free surface/ velocity/ dispersion terms?

Reply: We added detailed description of the restriction operator.

"The restriction operator can be expressed as

$$\varphi_2 = \frac{1}{MN} \sum_{i=1}^{M} \sum_{j=1}^{N} \tilde{\varphi}_{i,j} \tag{1}$$

where φ_2 represents the averaged value passing to the parent grid, $\tilde{\varphi}_{i,j}$ is the value at (i, j) in the child grid, (M, N) represent the numbers of child grid cells in (x, y) directions embedded in each parent grid cell.

Both the interpolator and the restriction operator are performed to $\eta, u_\alpha, v_\alpha, U$ and V, which pass the values back and forth between the child domain Ω_1 and the parent Ω_0 in the two-way nesting process. It should be pointed out that U and V are not necessarily included in the interpolation/restriction processes because they can be calculated based on η, u_α, and v_α. However, our tests show that directly using the passed U and V can make the model more efficient and does not affect the results much. "

Workload balance and data management

Page 13 line 293-295: This statement is not clear. What terms are pre-computed before the model run?

Reply: We rewrote this statement, providing more details on how to get the parent-child proximity:

"For efficient communication between the parent and child grids in the MPI-based parallel communication system, parent-child spatial proximity is created at the beginning of the model run. The parent-child proximity is associated with the

parent processor IDs and the child processor IDs at the nesting boundaries in the image distribution, as shown in Fig. 3. The parent-child proximity records the spatial relation of processor IDs between the parent and child grids and is saved in a parameter array. Therefore, the communication between the parent and child grids can be carried out straightforwardly using MPI_SEND and MPI_IRECV (MPI library) with the existing parameter array whenever a communication is needed. In this example, along the west boundary of the child grid, the child processors with ID=1, 2, and 3 communicate directly with parent processors with ID = 4 and 5. In particular, the variables in the ghost cells located in child processor ID=1 are evaluated by the interpolator in the parent processor ID=4, and so on."

The child-parent proximity (if proximity means boundaries) changes at each Parent time step – Same for the restriction process.

Reply: In the revision, we used child-parent spatial-proximity to avoid confusing. It is saved in a parameter array which won't change with time.

Personally I think the implementation is not very detailed considering that it's the main contribution from the paper. I don't understand how the communication between the child and the parent grid is straightforward...

Reply: We hope the rewriting based on the last question (parent-child proximity) can help to understand better the communication algorithm.

When the authors talk about shared array allocation, do all processors have a copy of all the model variables (Parents and child grids) + the boundary condition of child grid ? How does the synchronization work?

Reply: The shared array allocation here means that an allocated array (a variable) can be shared by all grids (parent and child), and thus that no additional allocation is needed for a particular grid. To make the definition more clearly, we use the example of the rectangular-shaped hump presented in 4.1.

" A strategy of shared array allocation is used, whereby the arrays are allocated

with the maximum dimension of all grids at the initialization stage, and grids at all levels share the same memory allocations. For example, in the case presented later in section 4.1, the nesting system has four grid levels, the dimensions of which are (48, 48), (73, 73), (92, 92), and (91, 91) for grids 1-4, respectively. A two-dimensional array for η will be allocated in η(92,92), the maximum dimension among the four grids. The four grids share the same array η(92,92), while the grids with a dimension smaller than (92,92) only use part of allocation, i.e., (48,48) for grid 1, (73,73) for grid 2, and (91, 91) for grid 4. There is no additional array allocation needed for a specific child grid with such a shared allocation strategy. It can apparently save a significant amount of computer memory and does not create an extra burden for a large number of nested grids. "

The nesting processes are actually sequential. The synchronization is performed at each grid level as in a single grid FUNWAVE model. No processor is idle because of the equal computational load.

We are sorry that the description in the last manuscript is not clear. In the revision, we used the example above to explain this concept.

How is the MPI implementation optimized for nested grid?

Reply: As mentioned in the last question, the calculations from one grid to another are sequential. The grid partition is performed at each grid level with the equal division (not exactly due to non-divisible grid numbers). There is no additional optimization for MPI partitions for nested grids.

In section 3.4, we added

"In summary, the hierarchical-type grid refinement processes are sequential and the synchronization is conducted at each grid level. Therefore, there is no additional optimization for MPI partitions needed for the nesting framework. "

How do the authors synchronize the parent and child grid after each parent time step? Do the authors use a fixed time step for the Child grid? If they use the CFL condition for the parent and the child solutions, does this involve that some type of synchronization is required before the update step.

Reply: We used CFL-criterion. Δt for the first grid level is determined by the CFL-criterion and is time varying. Because the subgrid ratio is an integer number, the time steps for child grid levels are reduced correspondingly based on the criterion. In the revised version, we added the following description.

"Based on the assumption that the grid refinement factor equals the time refinement factor, and the linear relation between time step and grid spacing holds in the CFL criterion, the child time step between two parent time levels is constant and can be written as

$$\Delta t_{\text{child}} = \Delta t_{\text{parent}}/s \tag{2}$$

where, Δt_{parent} and Δt_{child} denote the parent time step and child time step, respectively. "

Because the hierarchical-type grid refinement processes are sequential there is no synchronization involved in the nesting process.

Application

4.1 Evolution of an initial rectangular-shaped hump

Symmetry test is okay

Figure 7 : maybe include a diagonal transect and plot (a) (b) and (c) in the same figure. For better comparison.

Reply: We updated Figure 7 as suggested and also added a subplot showing a detailed comparison between different grid configurations along a transect.

Page18 line 349-352: The whole solution depends on grid resolution not only dispersive effects.

Reply: We agree with the reviewer that the solution is not only dependent on dispersive effects but also the grid resolution which can resolve the dispersive undulations.

ADDITIONAL CORRECTIONS

1) We re-drawed Figures 3, 8 and 11 to make consistency with the text.

2) We found an error in the number of ghost cells used in the nesting boundary. It should be 4, not 3. We corrected Figure 3, which is related to the issue and added the text (in section 3.1)

"Four ghost cells are used along Γ. Here, we want to mention that the number of ghost cells in the MPI-based parallelization is three, as required by the higher-order numerical schemes used in the model. The nesting scheme needs one more cell for the requirement because of re-calculating rather than passing values of the ghost cells. "

3) Some typos were corrected.

We want to thank the reviewer again for providing such a detailed and constructive review.