Ocean Modelling 115 (2017) 86–104

Contents lists available at ScienceDirect

Ocean Modelling

journal homepage: www.elsevier.com/locate/ocemod

Adaptive subdomain modeling: A multi-analysis technique for ocean circulation models

Alper Altuntas, John Baugh*

Department of Civil, Construction, and Environmental Engineering, North Carolina State University, Raleigh, NC, USA

ARTICLE INFO

Article history:
Received 15 December 2016
Revised 19 May 2017
Accepted 23 May 2017
Available online 24 May 2017

Keywords:
Storm surge
Adaptive algorithm
Subdomain modeling
Moving boundaries
ADCIRC

ABSTRACT

Many coastal and ocean processes of interest operate over large temporal and geographical scales and require a substantial amount of computational resources, particularly when engineering design and failure scenarios are also considered. This study presents an adaptive multi-analysis technique that improves the efficiency of these computations when multiple alternatives are being simulated. The technique, called *adaptive subdomain modeling*, concurrently analyzes any number of child domains, with each instance corresponding to a unique design or failure scenario, in addition to a full-scale parent domain providing the boundary conditions for its children. To contain the altered hydrodynamics originating from the modifications, the spatial extent of each child domain is adaptively adjusted during runtime depending on the response of the model. The technique is incorporated in ADCIRC++, a re-implementation of the popular ADCIRC ocean circulation model with an updated software architecture designed to facilitate this adaptive behavior and to utilize concurrent executions of multiple domains. The results of our case studies confirm that the method substantially reduces computational effort while maintaining accuracy.

1. Introduction

A comprehensive evaluation of the damaging effects of coastal hazards on the built and natural environment requires the assessment of many potential model configurations. While modifications corresponding to design and failure scenarios are often local in nature, ocean processes such as tides and hurricanes operate over significantly larger scales. As a result, despite the limited geographic extent of a region of interest, each local configuration requires a large-scale simulation to accurately capture the physics of the hydrodynamic processes involved (Blain et al., 1994).

To address this difference in scale, a prior study presented an exact reanalysis technique, called subdomain modeling, that enables the assessment of multiple local changes without requiring a separate full-scale simulation for each one (Baugh et al., 2015). The technique, implemented in ADCIRC, introduces a new boundary condition type that combines water surface elevation, velocity, and wet/dry status. The workflow begins with the extraction of subdomains from an original full-scale domain. Once subdomain grids are generated, a full-scale simulation is performed to obtain boundary conditions for each subdomain. Local changes corresponding to design and failure scenarios can then be applied to subdomain grids, provided the altered hydrodynamics remain within the boundaries, which are forced with data obtained from the original configuration. The technique substantially reduces the computational effort required to analyze local changes, but requires that users determine *a priori* the size and shape of each subdomain by anticipating the spatial extent of the effects of those changes. The efficiency of the technique may be reduced when subdomains are oversized, whereas, if undersized, the entire exercise may need to be repeated.

In this study, we present an adaptive subdomain modeling (ASM) technique where the sizes and shapes of computationally active regions, called *patches*,[1] of locally modified grids are automatically determined and adaptively adjusted during runtime. The technique is realized by concurrently executing the simulations of multiple child domains, with each instance corresponding to a local scenario, and a full-scale parent domain providing the boundary conditions for the child domains. Initially encompassing only the modified regions, the patches of child domains are dynamically adjusted during runtime depending on the response of the model. An error indicator—a measure of the difference between the

* Corresponding author.
E-mail address: jwb@ncsu.edu (J. Baugh).

http://dx.doi.org/10.1016/j.ocemod.2017.05.009
1463-5003/

[1] The term *patch* has a variety of definitions depending on context. For instance, in GeoClaw, a finite-volume hydrodynamic model with adaptive refinement capability (Berger et al., 2011), the term corresponds to overlapping layers of a computational grid with different refinement levels. Here we use the term to refer to computationally active regions of a grid.

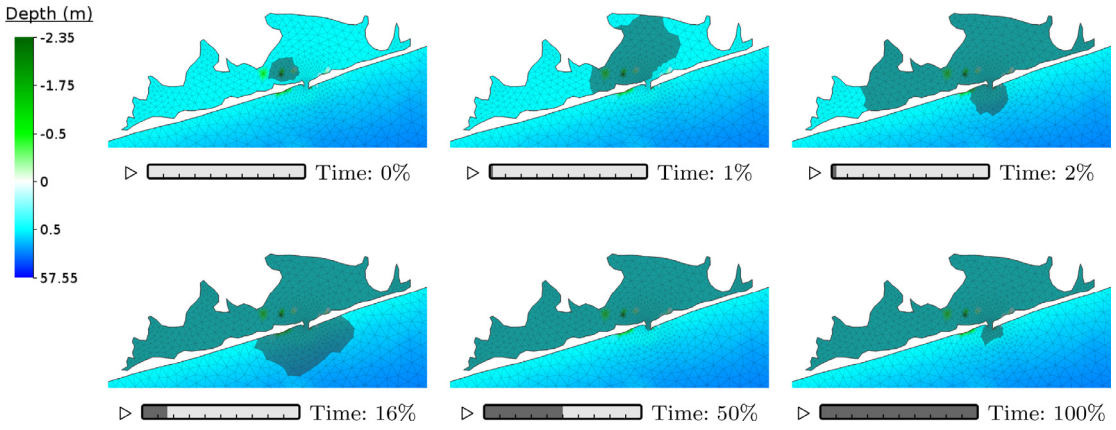

Fig. 1. Expansion and contraction of a locally modified Shinnecock Inlet child domain patch at various timesteps.

solutions of the parent and child domains—is calculated near the boundaries of patches to assess the proximity of altered hydrodynamics to the boundaries. In case the error indicator is determined to be larger than a user-specified tolerance, the patch boundary at that location is moved outward to ensure that the changing hydrodynamics approaching the boundary do not reach it before the next timestep. If the error indicator is determined to be sufficiently small, on the other hand, the patch is contracted to increase computational efficiency. An example of a child domain as a dynamically adjusting patch is shown in Fig. 1.

To accommodate the ASM approach, we make use of a re-implementation of ADCIRC with an updated software architecture that more readily supports adaptivity. In its original form, ADCIRC is based on procedural decomposition, with code that is structured by dividing control flow into subroutines, and where the primary data structures are global and non-reentrant. Our new architecture, on the other hand, is based on data abstraction, where the internal representation of a data type is distinct from its external view. This style of programming enhances modularity, maintainability, and extensibility by ensuring that the changes made within a data structure do not propagate to the rest of the program (Baugh and Rehak, 1992). The new implementation, called ADCIRC++, facilitates adaptive grid behavior and utilizes concurrent executions of multiple domains by means of dynamic containers and object-oriented design principles.

The remainder of the paper is organized as follows. In Section 2, we briefly describe ADCIRC and our original subdomain modeling approach, hereafter referred to as *conventional* subdomain modeling (CSM). In Section 3, the integral components of our new ASM approach are described: the error indicator, adaptivity algorithm, and application of boundary conditions. In Section 4, implementation details of ASM are presented, along with differences between ASM and CSM workflows and a hybrid approach that combines the two. Section 5 includes parametric studies with test cases that serve as a guide for determining the ASM control parameter settings subsequently used, and sensitivity analyses that demonstrate the applicability and computational efficiency of the method. Finally, conclusions and future work are presented.

2. Background

2.1. ADCIRC

ADCIRC is a continuous Galerkin finite element ocean circulation model, widely used by the US Army Corps of Engineers (USACE), Federal Emergency Management Agency (FEMA), and other agencies and institutions to simulate tides and hurricane storm surge (Tanaka et al., 2011). Combined with the flexibility of unstructured triangular meshes, ADCIRC's formulation of the governing equations and optimized numerical algorithms constitute an efficient and versatile modeling system (Luettich et al., 1992). As for the computational process, at every timestep ADCIRC solves the generalized wave continuity equation (GWCE) to obtain water surface elevations, then executes a wetting and drying algorithm to determine the geographic extent of hydrodynamic activity, and finally solves the momentum equations to obtain velocities.

ADCIRC simulations can be performed as three dimensional or two-dimensional depth integrated (2DDI) analyses. The linear system of the GWCE can be configured so that it is based on either consistent or lumped mass matrices, and time discretization may be performed either implicitly or explicitly. The consistent GWCE system is solved using an iterative Jacobi conjugate gradient method. For a 2DDI model with a consistent matrix solver and implicit timestepping scheme—as used in this study—both the GWCE and the momentum equations are discretized in space using the Galerkin finite element method (Tanaka et al., 2011), and the GWCE is discretized in time using a variably weighted three-time-level implicit scheme for the linear terms, while the momentum equations are discretized in time using a two-time-level implicit Crank-Nicolson approximation (Luettich et al., 1992).

2.2. Conventional subdomain modeling and applications

A basis for our adaptive technique is CSM, a static precursor that similarly enables the assessment of local alterations with less computational effort than would be required by repeated simulations on a full-scale grid (Baugh et al., 2015). Local changes can be applied to subdomain grids once they are extracted from an original full-scale grid to simulate design and failure scenarios, provided the subdomains are large enough to fully contain the altered hydrodynamics. The locations of the static boundaries of subdomain grids are predetermined by the user and enforced using boundary conditions that are defined by elevations, velocities, and wet/dry states obtained from the outputs of the original full-scale simulation.

The CSM workflow is as follows:

1. Construct a subdomain
   (a) Locate one or more regions of interest within the original full domain
   (b) Perform a simulation on the full domain to generate boundary conditions for each subdomain
   (c) Preprocess boundary condition files
   (d) Perform simulations on subdomains as a verification step
2. Generate engineering scenarios

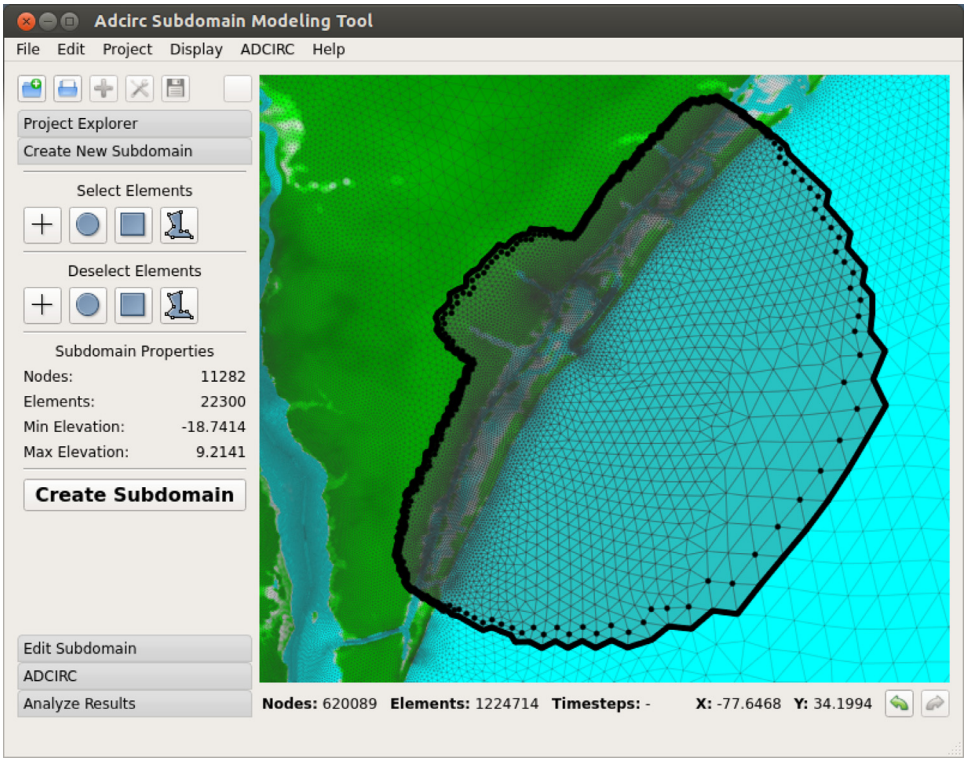

Fig. 2. Subdomain extraction with the SMT user interface.

(a) Alter subdomains to realize engineering design and failure scenarios of interest
(b) Perform simulations on altered subdomains
(c) Check results of altered subdomains as a second verification step

The two verification steps in the workflow are performed to confirm that results from *unaltered* subdomains match their full domain counterparts, and that the new hydrodynamics induced by *altered* subdomains do not propagate to subdomain boundaries.

The pre- and post-processing steps of CSM are facilitated by a graphical user interface called SMT, the Subdomain Modeling Tool (Dyer and Baugh, 2016). Multiple subdomains can be visually extracted using a variety of selection tools, as shown in Fig. 2. Once subdomains are defined by the user, the tool automatically generates the required input files for both the subdomains and the full domain.

CSM is incorporated in the official ADCIRC release, beginning with v51.42, and is now in active use by the modeling community. In one application of CSM, Butler et al. (2015) determine spatially varying Manning's n values probabilistically by formulating and solving a stochastic inverse problem. They employ a measure-theoretic framework to address the issue of uncertainties in the inverse problem due to the mapping from parametric data (Manning's n) to observational data (maximum water surface elevation), and due to errors in measurements. Once Manning's n fields are determined probabilistically, the results can then be used for predictive simulations, which may easily become computationally prohibitive. They point out, however, that the use of subdomain modeling can reduce the computational time and allow focusing on specific regions of interest that are prone to hurricane storm surge. Subsequently, one of their co-authors reduces the cost of a series of forward models using the subdomain modeling approach for his Hurricane Gustav Case Study (Graham, 2015), where he extracts a subdomain grid with 15 001 elements from a full-scale grid consisting of 2 720 591 elements. He notes that the runtime required for the full-scale grid is about 3300 CPU-hours, whereas for the subdomains it is only 11 CPU-hours.

In another application of CSM, Haddad et al. (2015) investigate the factors affecting the behavior of storm surge in wetlands by combining field work and numerical modeling. To assess the effects of landscape conditions and surface roughness on water levels, velocities, and wind fields, for instance, they carry out ADCIRC+SWAN simulations with varying Manning's n values, directional surface roughness length coefficients, and dense tree canopies. Since such sensitivity studies require substantial computational resources, they remark on the anticipated value of subdomain modeling in reducing the cost of repeated simulations with adjustments to the grid and the vegetation parameters in the regions of interest.

3. Adaptive subdomain modeling

ASM is an improved and complementary technique to CSM for ocean models that allows the simulation of locally modified child domains to be performed concurrently. Such modifications, for instance, might include changes to bathymetry, bottom friction, and horizontal eddy viscosity that constitute an alternative modeling scenario. By adaptively moving boundaries that are forced with data from the parent, the technique avoids performing computations that are external to a child domain and therefore redundant.

The ASM approach consists of three essential components. First, an error indicator determines the progression of altered hydrodynamics. Second, an adaptivity algorithm for the expansion and contraction of patches manages the insertion and removal of nodes and elements. Finally, boundary conditions are prescribed for accurate computations within child domain patches. In our implementation, the original ADCIRC timestepping loop is modified so that the adaptivity algorithm is executed at the beginning of each timestep to determine and apply any necessary adjustments to patch boundaries. Then, boundary conditions are enforced at

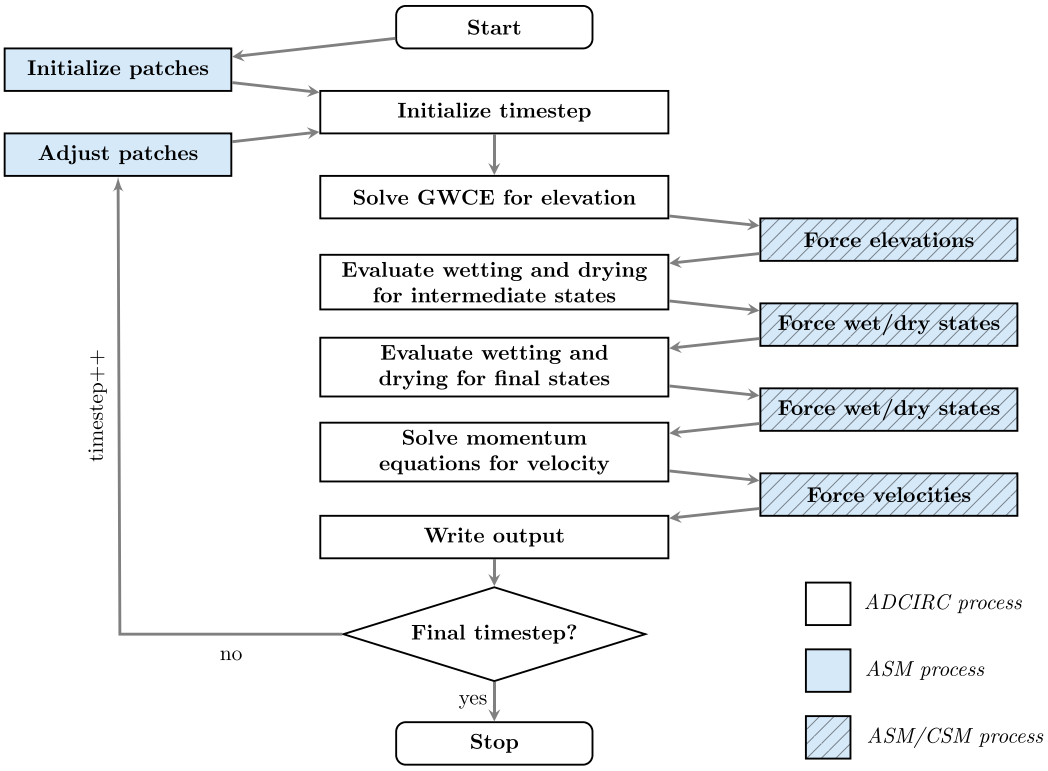

Fig. 3. Flowchart of the modified timestepping loop for adaptive subdomain modeling (ASM steps shaded, steps common to ASM and CSM patterned, and original steps left unshaded).

specified control points as is done in the CSM approach. A flowchart of the modified timestepping loop for ASM is shown in Fig. 3.

3.1. Error indicator

The decision of whether to move a patch boundary is based on an error measure indicative of the altered hydrodynamics, i.e., the differences between the water surface elevations and velocities of the child domains and the parent domain. In other applications, such as adaptive mesh refinement (AMR), error indicators determine how to adjust computational grids, determining where to refine or coarsen them to reduce the numerical error or to increase the computational efficiency. Such indicators may be based on solutions (Behrens, 1998; Eskilsson, 2011; Kubatko et al., 2009), derivatives of solutions (Liang et al., 2007; Tate et al., 2006), mass residuals (Tate et al., 2006; Dietrich et al., 2008; Marrocu and Ambrosi, 1999), or truncation errors (Berger and Colella, 1989; Berger and Oliger, 1984). Thus, their purpose is to guide decisions about mesh refinement and corresponding numerical schemes in an effort to improve overall convergence. In ASM, by way of contrast, the objective is to detect the altered hydrodynamics originating from local changes, and thereby ensure that each locally modified child domain behaves as though it were part of its own full-scale domain in a full-scale simulation. As a result, an error indicator based on differences between the solutions of child domains and parent domains is chosen:

$$\rho = max(\rho_\eta, \rho_u, \rho_v) \tag{1}$$

where

$$\rho_\eta = \left(\frac{|\eta_{child} - \eta_{parent}|}{\sqrt{0.5(|\eta_{child}| + |\eta_{parent}|)}} \right)^2$$

$$\rho_u = \left(\frac{|u_{child} - u_{parent}|}{\sqrt{0.5(|u_{child}| + |u_{parent}|)}} \right)^2 \Delta t$$

$$\rho_v = \left(\frac{|v_{child} - v_{parent}|}{\sqrt{0.5(|v_{child}| + |v_{parent}|)}} \right)^2 \Delta t$$

η : water surface elevation

u, v : x and y velocities

Δt : step size in seconds

Among several forms we have experimentally evaluated, the error indicator shown proves to be both stable and efficient for a wide range of model configurations. For instance, compared with absolute difference as an indicator, this form provides more sensitivity for smaller magnitudes of errors relative to larger ones. As used within our analysis procedure, the ρ indicator in Eq. (1) is calculated at nodes adjacent to the boundaries of child domain patches. The tolerance of a patch boundary node—a control parameter initially set by the user—is then compared with the error indicators of the adjacent nodes to determine whether the patch boundary should be moved at that location.

3.2. Adaptivity algorithm

In the ASM approach, child domain patches are first initialized to include only the nodes whose properties have been modified as part of an alternative modeling scenario, along with a three-layer buffer of surrounding nodes and elements, as shown in Fig. 4. Each layer has a rationale: the first is adjacent to and directly affected by changes to modified nodes, the second assesses the potential for altered hydrodynamics, and the third enforces boundary conditions obtained from the parent domain. Once the initial patch of nodes and elements has been determined and *activated* for each child

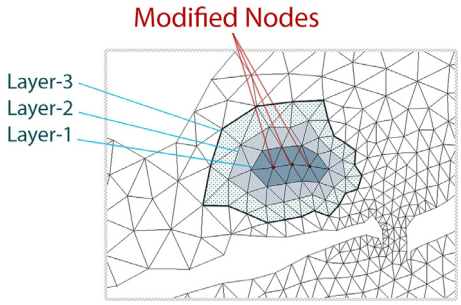

Fig. 4. Initial patch of the modified Shinnecock child domain.

domain, the corresponding systems of equations are constructed, and simulation can begin. Then, during runtime, child domain patches are adaptively adjusted to ensure that they are just large enough to cover the altered hydrodynamics. Control parameters that determine the shapes and sizes of patches are as follows:

- **tolerance** (τ): a parameter that varies with timestep and against which error indicators are compared to determine whether a patch expands; the comparison is of the form $\rho > \tau$. An initial tolerance of τ^0 is set by the user, and subsequent changes are made as necessary by the adaptivity algorithm.
- **minimum activation interval** (θ): the minimum number of timesteps throughout which a newly activated node must stay active; nodes within a patch are referred to as *active*.
- **decay constant** (λ): a parameter controlling the exponential decay of tolerance τ based on a reduction of the form $e^{-\lambda}$. Such reductions are applied after an increase in tolerance to return it over time to its initial setting, τ^0.
- **contraction factor** (σ): a constant set by the user that, along with the initial tolerance τ^0, is compared with error indicators to determine whether a patch contracts; the comparison is of the form $\rho < \sigma\tau^0$ and assumes that σ is less than one.

As an illustration of the relationship between control parameters, Fig. 5 shows a time history of the maximum error indicator at a timestep and its effect on the maximum tolerance for a hypothetical child domain. Each time the tolerance is exceeded by the error indicator near a patch boundary, the boundary is moved outward one layer, and the tolerance of the newly expanded boundary node (e) is set to the sum of the initial tolerance and the error indicator value of the marked node (m) causing the expansion; in other words, $\tau_e^{t+1} = \tau^0 + \rho_m^t$. Increasing the tolerance affords local errors some time to decrease without causing the boundary to be moved outward repeatedly at consecutive timesteps. Locally increased tolerances return to their initial setting over time based on the user-specified exponential decay constant (λ).

3.2.1. Boundary expansion and numerical stability

Before elaborating on the stages of the adaptivity algorithm, we consider the relationship between the boundary expansion procedure and the stability of the numerical scheme, which taken together must ensure that altered hydrodynamics are contained within the patches of child domains throughout the simulation.

The process of expanding patches relies on the assumption that the underlying numerical method employed, in this case by ADCIRC, satisfies the Courant–Friedrichs–Lewy (CFL) condition, which is necessary for the convergence of hyperbolic PDEs. The CFL condition states that a method can only be convergent if the numerical domain of dependence encompasses the analytical domain of dependence of the PDE (LeVeque, 2007). Since hyperbolic PDEs have a finite information propagation speed, their domain of dependence is finite, i.e., the solution at a node depends only on a finite domain (Hoffman and Frankel, 2001). The CFL condition is tested by comparing the Courant number, a ratio of Δt to Δx, against an upper bound. For ADCIRC and its semi-implicit time marching algorithm, the Courant number should be at most 0.5 for open ocean flows, and much less for other situations like near-shore flows with wetting and drying (Dresback, 2005). It is defined as:

$$C_r = \frac{\sqrt{gh}\Delta t}{\Delta x} \tag{2}$$

where $\sqrt{gh}$ is the linear wave celerity, Δt is the step size, and Δx is the distance between two nodes. Note that the given Courant number does not account for velocity but only for celerity since, for ADCIRC simulations, celerity is almost always expected to be greater than velocity by at least an order of magnitude, and hence is more limiting.

In summary, the CFL condition limits the maximum stable step size for a computational grid so that the solution at any point does not propagate beyond the domain of dependence, i.e., one layer of

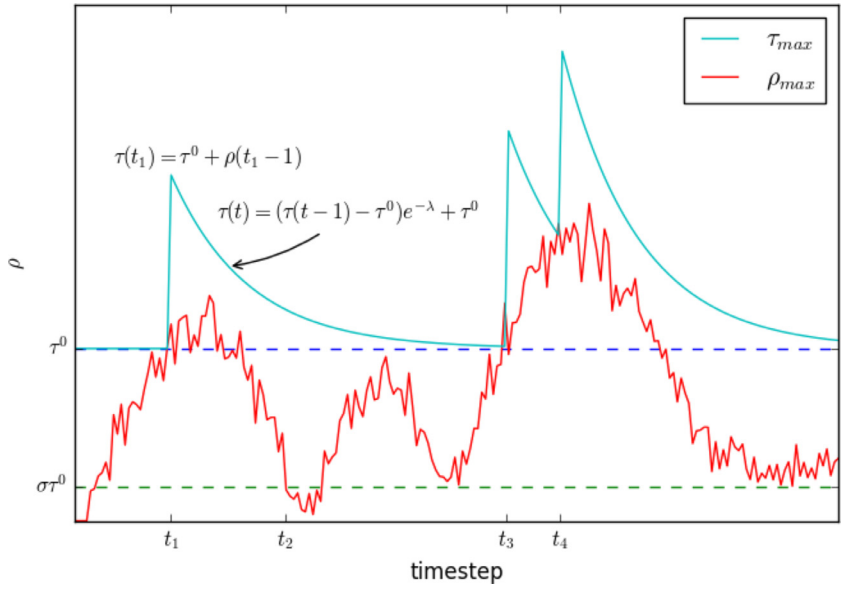

Fig. 5. Relationship between tolerance and error indicator during a simulation. The patch expands at timesteps t_1, t_3, and t_4, and contracts at timestep t_2 since ρ_{max} falls beneath $\sigma\tau^0$.

Expansion criteria: mark a patch boundary node i for expansion if

- $\exists n \in ineitab(i) : \rho_n > \tau_i$ and
- i not adjacent to a grid boundary (except island or mainland)

where $ineitab(i)$ is the set of neighboring internal nodes of i.

Fig. 6. Summary of criteria for expanding a patch.

elements, within a timestep. This restriction also ensures that expanding a child domain by a single layer of elements is sufficient to contain the altered hydrodynamics: such changes are guaranteed to propagate no further than a layer at a time, and therefore cannot reach the boundary. Once differences are detected at a node, the patch is expanded so that *two layers* separate any such nodes from the patch boundary.

As an alternative to the semi-implicit time-marching algorithm, one might consider using an implicit scheme so that the Courant stability constraint can be relaxed (Dresback and Kolar, 2000). In ADCIRC, however, the wetting and drying algorithm imposes an additional restriction on step size, since wetting fronts can propagate only one layer per timestep (Dietrich et al., 2012). As a result, regardless of the time-marching algorithm employed, the step size must be small enough so that the solution is limited to advancing a single layer of elements at a time, further justifying the ASM expansion policy in practice.

3.2.2. Stages of the adaptivity algorithm

The adaptivity algorithm consists of four main stages that are executed in turn at the beginning of each timestep. In the first, the algorithm calculates the error indicator ρ near patch boundaries, marks areas where a tolerance is exceeded for expansion, and areas where the indicators are sufficiently small for contraction. In the remaining three stages it performs the expansions and contractions, and finally carries out a post-processing step to update the affected properties and data structures. Implementation details of each stage are given below, and illustrations of the first three stages are provided in Appendix A.

Stage 1: Assessment of altered hydrodynamics. The algorithm begins by evaluating criteria that determine locations for expansion and contraction on patch boundaries. For expansion, error indicators at nodes *adjacent* to patch boundaries provide a way to gauge whether hydrodynamic changes are impinging, as signaled when indicator ρ exceeds tolerance τ. Such nodes are marked for expansion in a subsequent stage. As patches grow, they may eventually coincide with certain types of boundaries defined in the parent domain, such as mainland or island boundaries. However, if they reach boundary types defined by flux or water surface elevation, such as an open ocean boundary, execution of the child domain is aborted since it will have failed to satisfy the specified tolerance. The criteria for expansion are summarized in Fig. 6.

For contraction, several criteria come into play. The first is that all internal neighbors of a patch boundary node must have error indicator values ρ less than $\sigma\tau^0$, the product of the contraction factor and the initial tolerance. Then, the same test must be satisfied by the neighbors' neighbors of the boundary node, since nodes adjacent to the boundary node are about to become boundaries themselves (once the boundary node under consideration is deactivated). Continuing with other criteria, the minimum activation interval θ must be satisfied to prevent flickering of the nodes and elements, as is done in AMR implementations (Kubatko et al., 2009). Finally, the boundary node under consideration should not be adjacent to a node marked for expansion, and it should not be one of the initially active nodes in the original patch. If these hold, such

Contraction criteria: mark a patch boundary node i for contraction if

- $\forall n \in ineitab(i) : \rho_n < \sigma\tau^0$ and $\forall m \in ineitab(n) : \rho_m < \sigma\tau^0$ and
- i activated at least θ timesteps ago and
- i not adjacent to an expansion node and
- i not included in the initial patch

where $ineitab(i)$ is the set of neighboring internal nodes of i.

Fig. 7. Summary of criteria for contracting a patch.

nodes are marked for contraction in a subsequent stage and added to a *deactivation* list. The criteria for contraction are summarized in Fig. 7.

Stage 2: Expansion. After marking nodes for expansion, the algorithm is ready to make changes to the patch by moving boundaries outward at those locations. The expansion stage, details of which are presented in Appendix B, converts the marked boundary nodes to internal nodes, determines which external nodes comprise the expansion layer, activates them and the elements incident on them, updates the topology (connectivity) of the grid, and sets the tolerances of new patch boundary nodes.

With respect to memory management, optimizations are performed to minimize repetitive allocation and deallocation of nodes and elements. When a node external to a patch is activated for the first time, for instance, space is allocated for it as part of the child domain. The node is also marked as active, but at a later time it might be deactivated, at which point it is removed from the child domain but its underlying space allocation remains. If it is subsequently activated, then, no reallocation of space is required.

After the designated nodes and elements are activated, some bookkeeping and clean-up steps must be performed. The adjacency table is updated to connect the newly activated nodes with the adjacent nodes and the newly activated elements with the nodes incident to them. Patch boundary nodes not marked for expansion that are surrounded by newly activated nodes and elements are converted to internal nodes to prevent them from being treated as boundary conditions and checked for expansion or contraction. Finally, the tolerances of the new boundary nodes are updated.

Stage 3: Contraction. Presented in detail in Appendix C, the contraction stage of the algorithm includes the following major steps: clean up the list of nodes marked for deactivation in the first stage, process that list by actually deactivating nodes in the child domain, and update the adjacency table to disconnect the deactivated nodes and elements from the active nodes.

Before any nodes are deactivated, the algorithm checks for inconsistencies. As a result of the marking in stage 1, some of the patch boundary nodes may become disconnected from internal nodes and remain connected only to patch boundary nodes; such nodes are now added to the deactivation list. Conversely, some nodes are removed from the deactivation list, namely, those that are surrounded by active nodes as a result of expansions, and those that are adjacent to an expansion node or a recently activated node.

At this point, nodes are deactivated by removing them from the patch. Internal nodes that are connected to deactivated nodes are then converted to patch boundary nodes. Additionally, elements incident on deactivated nodes are deactivated by removing them from the patch, and nodal connectivity is updated. Once this process is complete, nodes that are only incident on inactive elements are deactivated. Finally, the connectivity of all affected nodes and elements is updated.

Stage 4: Post-processing. After the expansion and contraction processes are complete, properties and containers of the child domains are updated in this final stage of the algorithm. As part of that process, the system of equations associated with any expanded or contracted patch is reset and resized. Then, auxiliary containers holding nodal data are updated. Finally, patch boundary nodes with tolerances greater than τ^0 are subjected to an exponential decay so that their individual tolerances converge toward the initial tolerance over time. For a patch boundary node j at timestep t, the tolerance is updated as follows:

$$\tau_j^t = (\tau_j^{t-1} - \tau^0)e^{-\lambda} + \tau^0 \tag{3}$$

where λ is the decay constant, τ_j^{t-1} is the tolerance of boundary node j at the previous timestep, and τ^0 is the initial tolerance.

3.3. Boundary conditions

To perform simulations concurrently, an interface is needed between a parent domain and its children. For its basis, we adapt the boundary condition type used in CSM, which incorporates water surface elevation, wet/dry status, and velocity, to realize a one-way hand off from parent to child (Baugh et al., 2015). The conventional approach to subdomain modeling obtains these quantities after completion of a full-scale run, and then applies them to static boundaries of a subdomain. In ASM, of course, boundaries are in motion, but only at the start of a timestep, giving us a static snapshot afterward in which boundary conditions may be applied. To do so, we (a) specify nodal elevations in the implicit GWCE formulation, (b) force wet/dry status on boundary nodes in the wetting and drying routine, and (c) assign boundary velocities outright in the momentum equation solver.

Of the three conditions—water surface elevation, wet/dry status, and velocity—the enforcement of nodal wetting is somewhat less straightforward because, during each timestep, ADCIRC's wetting and drying algorithm (Dietrich et al., 2004) performs several updates to a node before its final wet/dry state is set, and these intermittent changes are spatially dependent on the intermediate states of other, neighboring nodes. A prior study (Baugh et al., 2015) presents an analysis of data dependencies and interactions between the wetting and drying algorithm and the hand off required by subdomain modeling and other mesh partitioning schemes. Included is a proof showing that, for correctness, the multiple *intermediate* wet/dry states of a subdomain boundary node can be set with a single value: the node's *final* wet/dry state at a given timestep from a full run.[2] The implication is that the only data transfer required from one domain to another is that of the final wet/dry states, simplifying communication between domains. Applying this result to ASM, we again note that patch boundary nodes are spatially adjusted at the beginning of a timestep and otherwise remain fixed throughout its execution. Thus, apart from extraction and processing procedures, boundary conditions in ASM can be enforced in the same manner as they are in CSM.

4. Workflow and hybrid approach

Using ASM begins with a modeling step: identifying geographic locations of interest and determining the alternatives to be simulated in concert with an ordinary ADCIRC model. Then, a single input file for each child domain is created: a difference file (`.dif`) containing a list of modified nodes along with new values of their associated properties, e.g., bathymetry, bottom friction, and horizontal eddy viscosity. In contrast with CSM, subdomain boundaries are not defined, and the abbreviated versions of input files ordinarily used by subdomains are not required. Instead, the locations, sizes, and shapes of initial child domain patches are determined automatically, and model parameters are copied as needed from the parent domain.

Typical ASM workflow

0. Begin with an ordinary ADCIRC model
1. Generate ADCIRC++ input files:
   - a difference file (`.dif`) with modified nodes for each child domain
   - a project file (`.prj`) that lists parent and child domains
   - a configuration file (`.cfg`) for each domain that points to standard ADCIRC files, a difference file, and an optional ASM file (`.asm`)
2. Run ADCIRC++

Fig. 8. Summary of a typical workflow for ASM in ADCIRC++.

With respect to other input files, we rely on standard ADCIRC file formats and add some of our own. To organize them, we define the notion of a *project* to be an ordinary ADCIRC model plus some number of child domains: a project file (`.prj`) contains a list of the included domains, i.e., the parent and all of its children. Each domain included in a project file has an associated configuration file (`.cfg`) that points to the locations of standard ADCIRC files used by the domain and a difference file. An optional input file for child domains is the ASM file (`.asm`), where control parameters for adaptive subdomain modeling are set; if missing, default values are assumed.

In addition to being straightforward, the ASM workflow eliminates two verification steps that are required by CSM: confirming the stability of unaltered subdomain grids, and ensuring that altered hydrodynamics do not reach subdomain boundaries. The complete workflow of a typical ADCIRC++ run with ASM is summarized in Fig. 8.

As an optional step in the workflow, users can experiment with ASM control parameters. Although ADCIRC++ provides a set of default values, the efficiency and accuracy of the technique can be optimized by varying them and examining their effects. Doing so requires only a single concurrent execution of a parent domain and some number of child domains with different control parameter settings, so the additional cost is marginal.

While ASM offers some important advantages, an apparent weakness is the inability of users to alter one or more child domains after reviewing the results produced by another, unless they resort to running another full-scale simulation. However, ASM and CSM are complementary techniques that can be used in combination in cases such as the above, which call for a sequential analysis of subdomains. Using CSM, a single, full-scale simulation can produce one or more conventional, static subdomains, any of which can be used as the parent domain in an ASM simulation with any number of child domains. We refer to this combination as *hybrid* subdomain modeling (HSM), and present examples of its use below.

5. Test cases

In this section, we present the results of two sets of parameter studies on realistic application domains. The first set focuses on ASM control parameters, and for that we perform simulations at two different sites, include a single alternative scenario for each, and vary control parameters to analyze their effects on the accuracy and efficiency of the method. We consider both astronomical tide and meteorological forcing examples. In the second set, we look at applications of ASM to storm surge problems at two sites, varying bathymetric depths and bottom friction values

[2] Verification of the same results using *software model checking* techniques can be found elsewhere (Baugh and Altuntas, 2016).

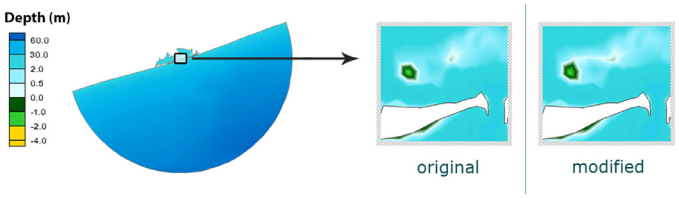

Fig. 9. Case 1: Local change near Shinnecock Inlet.

parametrically while using control parameter settings informed by the first study.

In all cases, errors are determined by comparing ASM results with a separate, independent run of a parent domain that directly incorporates the local change previously simulated by its children.

5.1. ASM control parameters

For evaluating the effects of ASM control parameters on accuracy and efficiency, we look at the following two cases:

1. Shinnecock Inlet on the south shore of Long Island, NY
   A tidal model with a coarse grid, limited area, and a bathymetric change in the inlet
2. Walden Creek at Southport, NC
   Hurricane Fran (1996) simulated on a subdomain around Cape Fear, with an added protective structure

Each case is simulated with a number of concurrent child domains, where each child has the same local change but different values of ASM control parameters. The range of control parameter settings presented reflects a limit for each domain: tolerances smaller than the smallest tolerance and decay constants larger than the largest decay constant result in child domain patches reaching an open ocean boundary, thereby causing the technique to fail. Additional case studies examining the effects of ASM control parameters are presented in the dissertation by Altuntas (2016).

Case 1: Shinnecock Inlet with tidal forcing

As an introductory example, a tidal model developed by the USACE Coastal and Hydraulics Laboratory (Militello and Kraus, 2001; Morang, 1999; Williams et al., 1998) and available on the ADCIRC website (ADCIRC, 2016) is used. Centered on Shinnecock Inlet, New York, the model is realistic, though coarse in time and space, with a grid of 5780 elements and 3070 nodes covering a small area. The total duration of the simulation is 5 days, and the step size is 6 s. Tidal constituents M2, N2, S2, K1, and O1 are applied as tidal potential forcings and tidal boundary forcings. To simulate a small, local change, the bathymetric depths of three nodes near the inlet are reduced by 1 m, as shown in Fig. 9.

As the simulation unfolds, child domain patches with their local changes expand to contain the altered hydrodynamics. For ASM control parameter settings of $\tau^0 = 10^{-3}$, $\sigma = 10^{-2}$, $\lambda = 10^{-4}$, and $\theta = 10$, for instance, the associated patch reaches its maximum size of 910 elements (about 16% of the entire grid) at timestep 2949 (about 4% of simulated time), as shown in Fig. 10. The size of the patch remains mostly the same throughout the simulation, since expansions and contractions come into equilibrium once it covers the maximum region of altered hydrodynamics.

For the parametric study, 120 ($= 5 \times 3 \times 4 \times 2$) child domains are concurrently simulated using all combinations of the following control parameter settings: $\tau^0 = \{10^{-1}, 10^{-2}, 10^{-3}, 10^{-4}, 10^{-5}\}$, $\sigma = \{10^{-1}, 10^{-2}, 10^{-3}\}$, $\lambda = \{10^{-4}, 10^{-5}, 10^{-6}, 10^{-7}\}$, and $\theta = \{1, 10\}$. The results of modified child domains are compared with those of the parent domain from a separate run after the same local change has been made to it. Fig. 11 shows the l^2-norms and max-norms of errors, and the average percentage of active elements for child domains with $\sigma = 10^{-2}$. Results for child domains with other σ values ($10^{-1}, 10^{-3}$), which may be found elsewhere (Altuntas, 2016), demonstrate similar variations in errors and the average percentage of active elements when the remaining ASM parameters are varied.

As seen in the graphs, the initial tolerance setting has the greatest influence on the accuracy of the approach for this simple model. The largest improvements in accuracy for both the l^2-norms and max-norms are observed when reducing the initial tolerance from 10^{-2} to 10^{-3}. Effects of adjustments to the remaining parameters are less significant.

Case 2: Walden Creek and Hurricane Fran (1996)

As an example of meteorological forcing that also happens to use conventional subdomains, a large-scale storm surge model from a prior study (Baugh et al., 2015) is simulated using HSM, a hybrid approach to subdomain modeling that combines ASM and CSM. The full-scale grid from that study consists of 620 089 nodes and 1 224 714 elements encompassing the western North Atlantic Ocean, the Caribbean Ocean, and the Gulf of Mexico. Along the coastlines are external land boundaries having no normal flow and

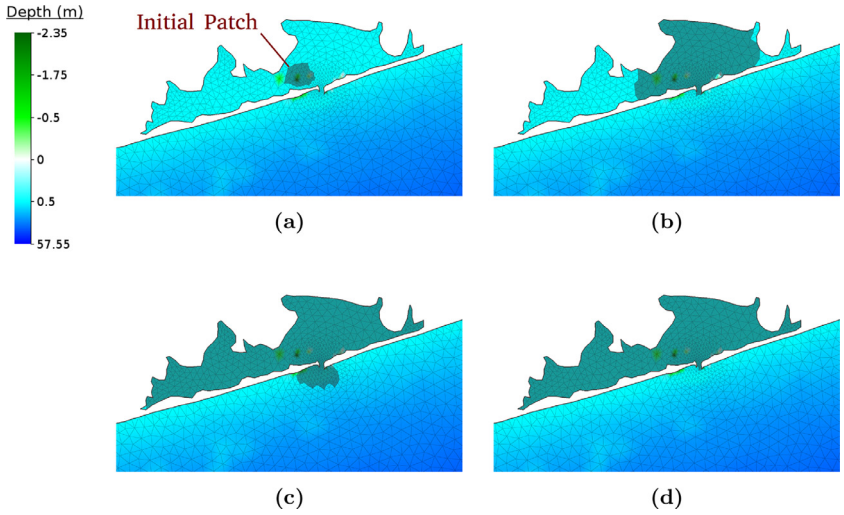

Fig. 10. Case 1: Progression of a Shinnecock child domain patch for $\tau^0 = 10^{-3}$, $\sigma = 10^{-2}$, $\lambda = 10^{-4}$, and $\theta = 10$, with elements in the patch darkened. Snapshots: (a) initial extent, (b) expansion at timestep 812, (c) largest patch (occurring at timestep 2949), and (d) final extent (timestep 71 276).

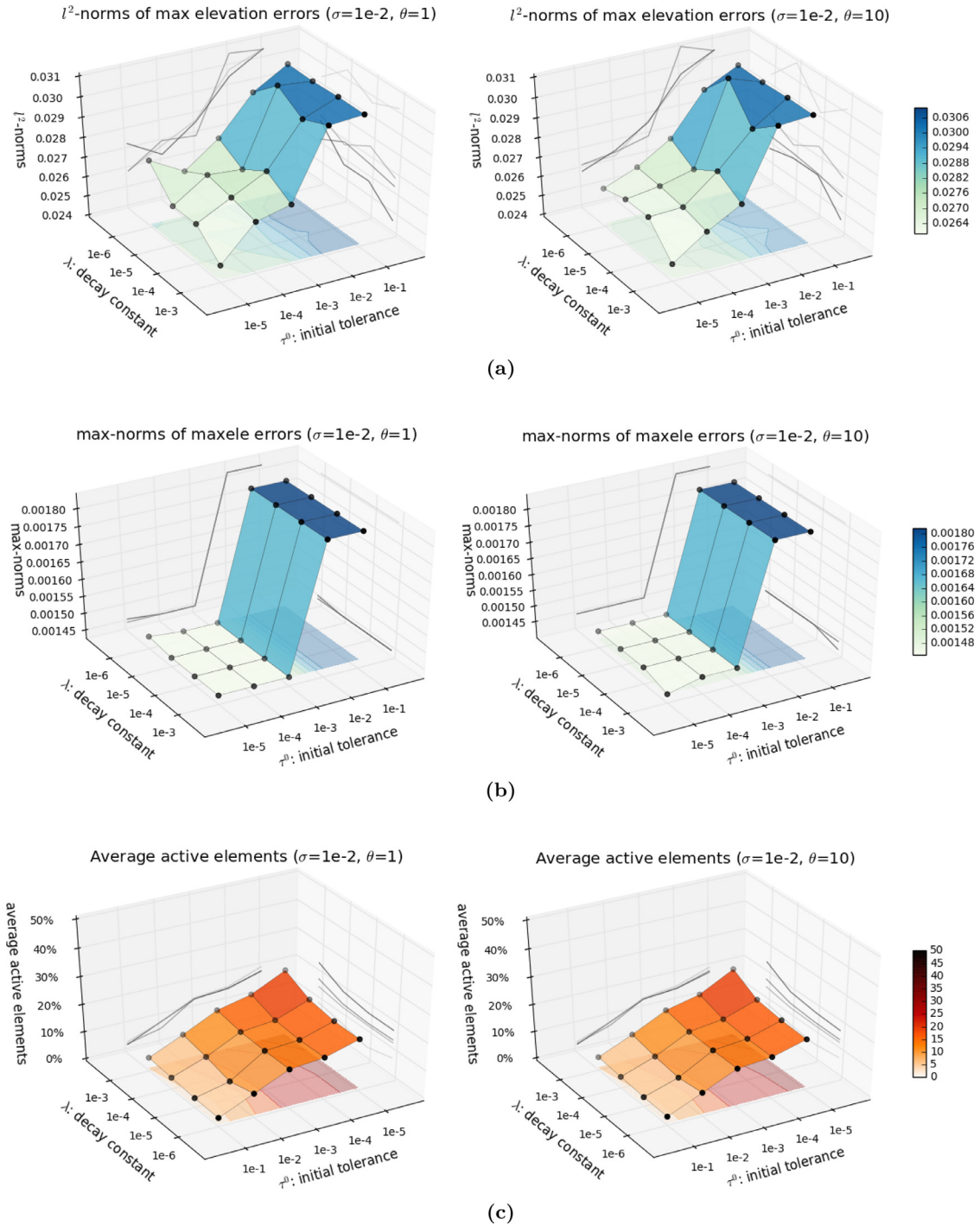

Fig. 11. Case 1: (a) l^2-norms of maximum elevation errors, (b) max-norms of maximum elevation errors, and (c) average percentage of active elements (with λ, τ axes reversed) in Shinnecock child domains with $\sigma = 10^{-2}$.

free tangential slip, and along the eastern edge of the domain is a steady open ocean boundary condition. The specified nodal attributes include surface directional effective roughness length, Manning's n at the sea floor, surface canopy coefficient, and primitive weighting in the continuity equation. For Hurricane Fran, a 0.5-s step size is used to perform a 3.9-day simulation of the event as a 2DDI analysis.

To employ HSM, we first perform a run on the full domain to generate boundary conditions for a circular subdomain consisting of 28 643 nodes and 56 983 elements around Cape Fear, North Carolina. Once extracted, the subdomain and its boundary conditions then serve as a parent domain in an ASM simulation. To generate a local change for testing, we raise the topography in the Walden Creek area north of Southport, as shown in Fig. 12, which results in a 2.5-mile protective structure that prevents flooding in a region zoned for heavy industry and military.

The expansion and contraction of a child domain with $\tau^0 = 10^{-5}$, $\sigma = 10^{-2}$, $\lambda = 5 \times 10^{-4}$, and $\theta = 10$ is shown in Fig. 13. Since, in this case, local changes are made to dry nodes, the extent of the patch remains the same as its initial extent until timestep 539 602 (about 80% of simulated time). Once surge effects reach the locally modified region, however, the patch begins to expand

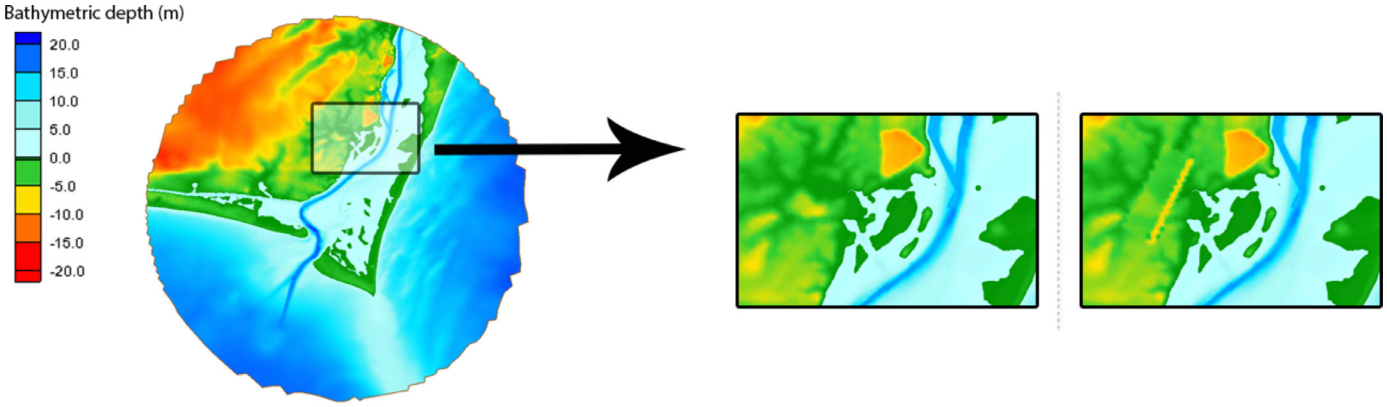

Fig. 12. Case 2: Local change at Walden Creek in the Cape Fear subdomain (Baugh et al., 2015).

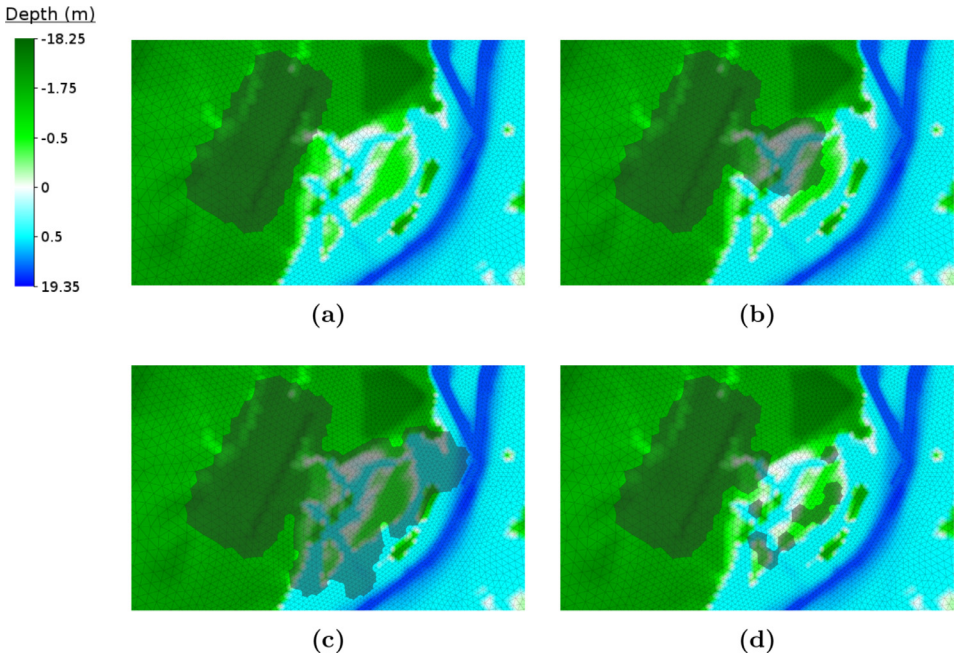

Fig. 13. Case 2: Progression of a Walden Creek child domain patch for $\tau^0 = 10^{-5}$, $\sigma = 10^{-2}$, $\lambda = 5 \times 10^{-4}$, and $\theta = 10$, with elements in the patch darkened. Snapshots: (a) initial extent, (b) extent at timestep 546 520, (c) largest extent (occurring at timestep 570 694), and (d) final extent (timestep 668 367).

to accommodate the altered hydrodynamics induced by the structure. As the storm surge retreats and the effects of the local change dissipate, the patch begins to contract.

For the parametric study, 144 child domains are concurrently simulated using all combinations of the following control parameter settings: $\tau^0 = \{10^{-1}, 10^{-2}, 10^{-3}, 10^{-4}, 10^{-5}, 10^{-6}\}$, $\sigma = \{10^{-1}, 10^{-2}, 10^{-3}\}$, $\lambda = \{5 \times 10^{-3}, 5 \times 10^{-4}, 5 \times 10^{-5}, 5 \times 10^{-6}\}$, $\theta = \{10, 100\}$. Once again, to evaluate the influence of those settings, we perform a baseline run of the parent domain after the same local change has been made to it. Fig. 14 shows the l^2-norms and max-norms of maximum elevation errors, and the average percentage of active elements for child domains with $\sigma = 10^{-2}$. The graphs for the remaining child domains (with $\sigma = 10^{-1}$ and $\sigma = 10^{-3}$) demonstrate similar variations in the accuracy and efficiency of the method (Altuntas, 2016).

Changes in initial tolerance and the decay constant have the most significant influences while activation interval and the contraction factor are less significant. As the initial tolerance is reduced, the child domain patches expand further and accuracy improves. Similarly, as the decay constant is increased, the local tolerances converge to the initial tolerance more quickly, and so once again the accuracy improves. The maximum error in maximum elevations is 0.69 cm for the best combination of settings ($\tau^0 = 10^{-6}$, $\sigma = 10^{-3}$, $\lambda = 5 \times 10^{-3}$ and $\theta = 10$), 3.7 cm for the worst ($\tau^0 = 10^{-1}$, $\sigma = 10^{-1}$, $\lambda = 5 \times 10^{-6}$ and $\theta = 100$), and less than 1 cm for most of them.

Discussion

The foregoing study on control parameters suggests that accuracy generally improves with (a) reductions in the initial tolerance and (b) increases in the decay constant, as one might expect, but there are caveats. For the initial tolerance, the largest improvements in accuracy seem to occur when it is reduced from 10^{-2} to 10^{-3} or 10^{-4}. Further reductions, however, to say 10^{-5}, improve accuracy only marginally while degrading the computational efficiency significantly and increasing the risk of patches reaching the grid boundary.

In addition to parameter settings, some modeling scenarios are inherently either more or less likely to produce early termination as a result of patches reaching a boundary. For instance, storm surge simulations focusing on overland flows and local changes in *topography* are usually more robust even for very low initial tolerances, as seen in the Walden Creek example with Hurricane Fran.

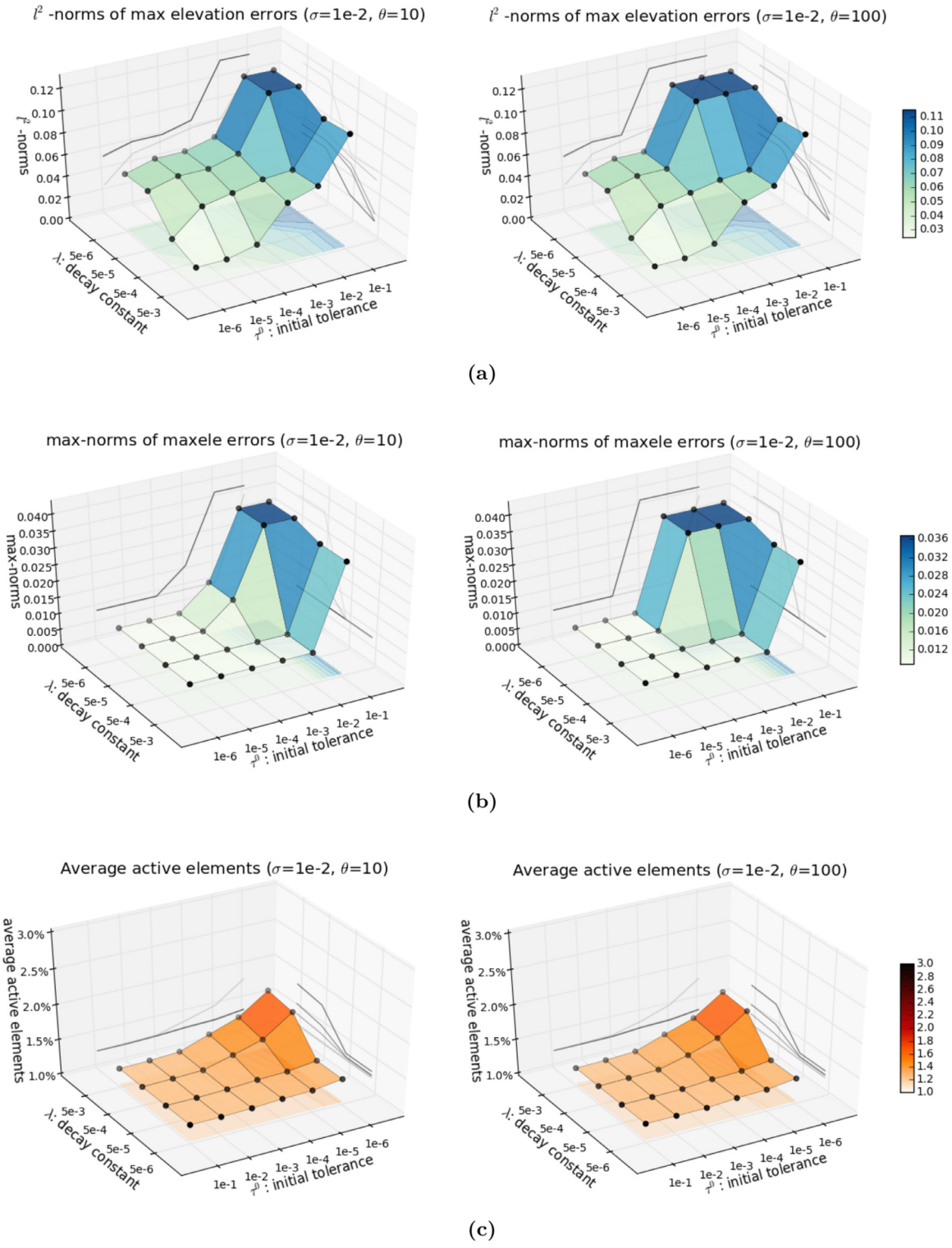

Fig. 14. Case 2: (a) l^2-norms of maximum elevation errors, (b) max-norms of maximum elevation errors, and (c) average percentage of active elements (with λ, τ axes reversed) in Walden Creek child domains with $\sigma = 10^{-2}$.

In other cases, such as the Brunswick intake canal problem presented in the next section, local changes in *bathymetry* do influence the hydrodynamics early in the simulation, but less stringent tolerances still allow patches to expand and accommodate those influences appropriately, long before any surge effects come into play. As a result, the ASM technique can accurately simulate diverse modeling conditions, even when local changes occur near grid boundaries, as demonstrated here.

Based on these and other studies we have performed on the accuracy and efficiency of the method (Altuntas, 2016), a good balance seems to be found with ASM control parameter values of $\tau^0 = 10^{-3}$, $\sigma = 10^{-1}$, $\lambda = 10^{-4}$, and $\theta = 10$, so these constitute our default settings for ADCIRC++.

5.2. Applications and performance

To further demonstrate subdomain modeling and its computational advantages, we look at the following additional cases:

3. Brunswick intake canal at Southport, NC
   Hurricane Irene (2011) simulated on a more refined Cape Fear subdomain, with a range of depths and Manning's n values along the Brunswick nuclear power plant's intake canal
4. Silver Lake at Wilmington, NC
   Hurricane Fran (1996) simulated on a small portion of the Cape Fear River, with varying values of Manning's n and depth on a part of the river bank

As before, each case is simulated with a number of concurrent child domains, but here the local changes are *problem domain* changes that are carried out using a constant set of ASM control parameter values. By again making use of conventional, static subdomains for the parents, they also demonstrate the complementary benefits of subdomain modeling approaches realized by HSM.

Case 3: Brunswick intake canal and Hurricane Irene (2011)

We again apply HSM in the context of hurricane storm surge, but this time using a more refined grid of the western North Atlantic: NC Mesh Version 9.98 with 622 946 nodes and 1 230 430 elements. Otherwise, the extent and model parameters mostly correspond to those given in Case 2. For Hurricane Irene, a best-track file from the NOAA NHC online data archive is used for the meteorological forcing, and a 0.5-s step size is used to perform an 8-day simulation of the event as a 2DDI analysis. For this example, no tidal forcing is applied, thereby eliminating a long tidal spin-up run.

Working in the same area as before, we perform a full-scale run to obtain boundary conditions for a circular subdomain consisting of 39 234 nodes and 78 114 elements around Cape Fear, North Carolina. Our area of focus this time is the intake canal of the Brunswick nuclear power plant, where we vary bottom surface conditions for a set of child domains. Throughout its length, we use a constant Manning's n value of either 0.012, 0.024, 0.048, or 0.096, which ranges from constructed channel conditions to ones that are unmaintained and have dense brush and weeds. Simultaneously, 17 different changes in depth, from −2 m to 2 m, are made to the original bathymetry of the canal in the parent domain. Fig. 15 shows the circular subdomain and the local modification for one of the child domains where the depths of all 244 nodes are increased by 2 m.

With recording stations shown in Fig. 16, a simulation of the resulting 68 child domains is performed using the following ASM control parameter settings: $\tau^0 = 10^{-3}$, $\sigma = 10^{-2}$, $\lambda = 10^{-5}$, and $\theta = 100$. The expansion and contraction of the child domain with

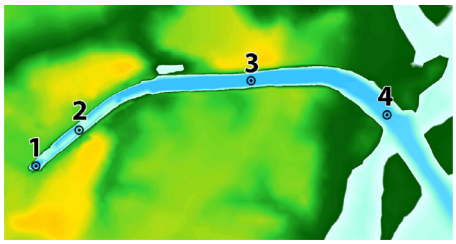

Fig. 16. Case 3: Recording stations at the intake canal.

canal depth increased by 2 m and Manning's n value set to 0.096 is shown in Fig. 17.

Fig. 18 shows the maximum water surface elevations at each recording station for four selected child domains with the following pairs of changes in depths and Manning's n values: (−2 m, 0.012), (−1.5 m, 0.012), (−1.5 m, 0.096), and (2 m, 0.096).

Case 4: Silver Lake and Hurricane Fran (1996)

Using the same grid from Case 2 encompassing the western North Atlantic Ocean, the Caribbean Ocean, and the Gulf of Mexico (Baugh et al., 2015), we generate a small subdomain consisting of 11 255 nodes and 22 223 elements on the Cape Fear River near Silver Lake in Wilmington, NC, as shown in Fig. 19.

As a hypothetical problem context, a development activity adjacent to the river seeks materials ecologically best suited to lowering water velocities during a hurricane event. To simulate a range of such materials, Manning's n values are varied along two rows of nodes (50 nodes in total) in the region shown in the figure. The following Manning's n values are considered: 0.015, 0.041, 0.067, 0.093, 0.119, 0.145, 0.172, 0.198, 0.224, and 0.250. Additionally, to simulate the effects of planned gabion walls of different sizes, the inner row of nodes closer to the river is adjusted in height ranging from 0 m to 0.5 m increases. With recording stations shown in Fig. 20, a simulation of the resulting 60 child domains is performed using the following ASM control parameter settings: $\tau^0 = 10^{-4}$, $\sigma = 10^{-1}$, $\lambda = 10^{-4}$, and $\theta = 100$. The expansion and contraction of the child domain where the Manning's n values of the nodes are set to 0.25, and the inner row of nodes is raised by 0.5 m is shown in Fig. 21.

Fig. 22 shows the velocities of all child domains at the recording stations. As indicated by the plots, both roughness and raised topography can be used, whether separately or in combination, to reduce water velocities during the hurricane event simulated.

Performance

The computational efficiency of ASM depends on numerous factors, including control parameter settings, model settings, and the spatial and temporal extent of the impacts of local changes. We

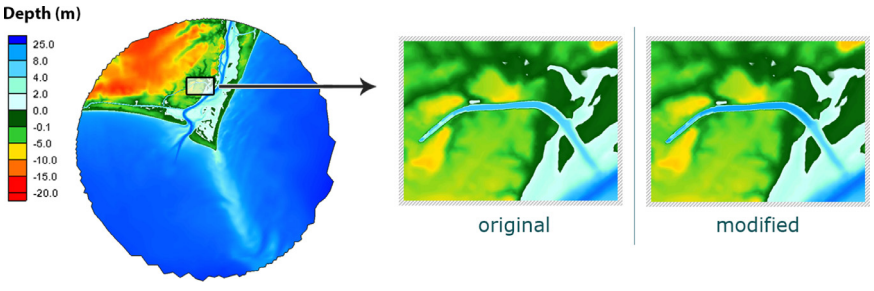

Fig. 15. Case 3: Local change at Brunswick intake canal in the refined Cape Fear subdomain.

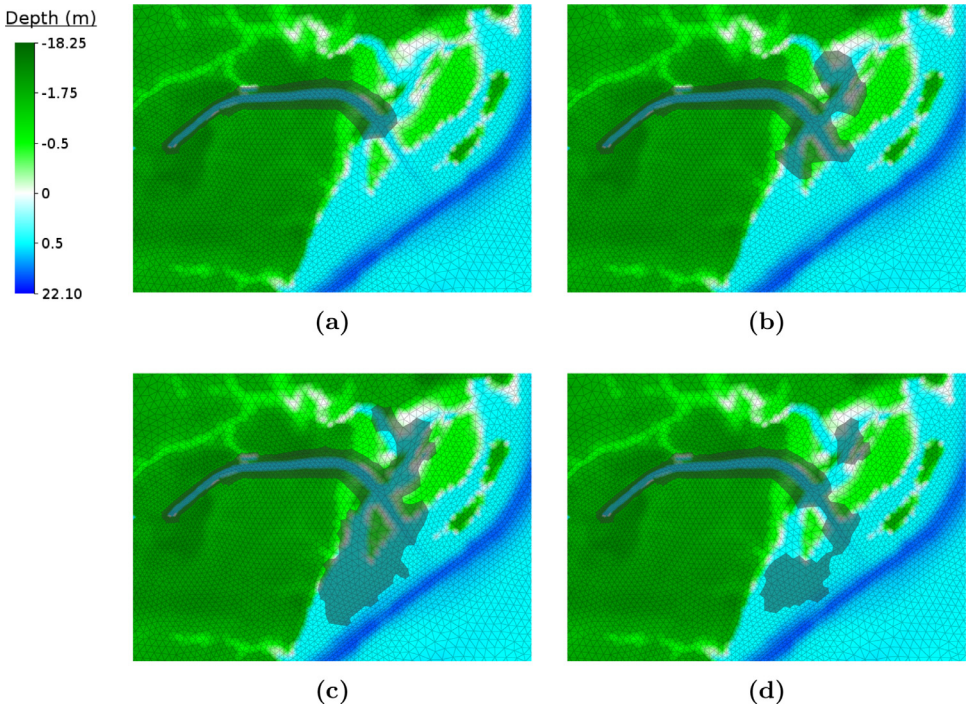

Fig. 17. Case 3: Progression of a Brunswick child domain patch where the canal depth is increased by 2 m and Manning's n value set to 0.096, with elements in the patch darkened. Snapshots: (a) initial extent, (b) extent at timestep 988 919, (c) largest extent (occurring at timestep 1 262 107), and (d) final extent (timestep 1 367 123).

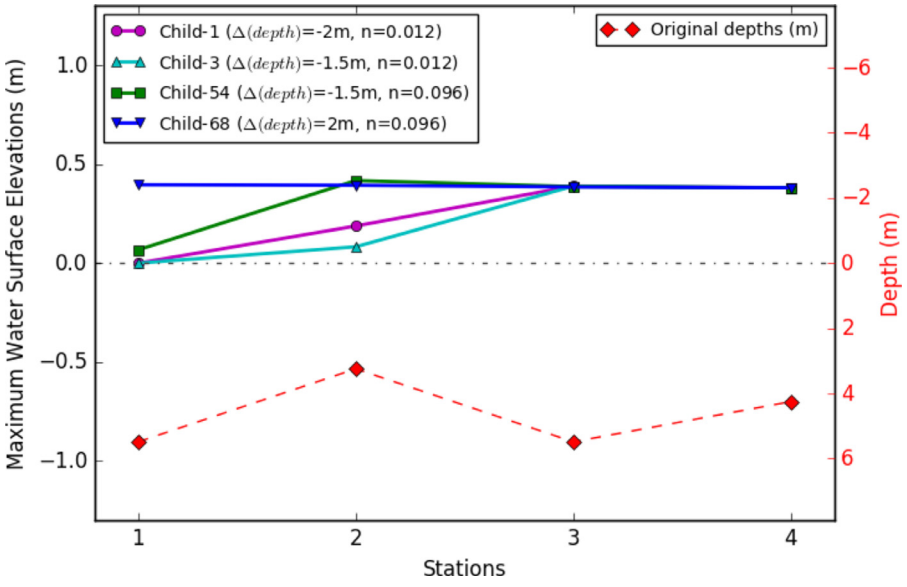

Fig. 18. Case 3: Water surface elevations at intake canal recording stations for four selected child domains with varying depths and Manning's n values.

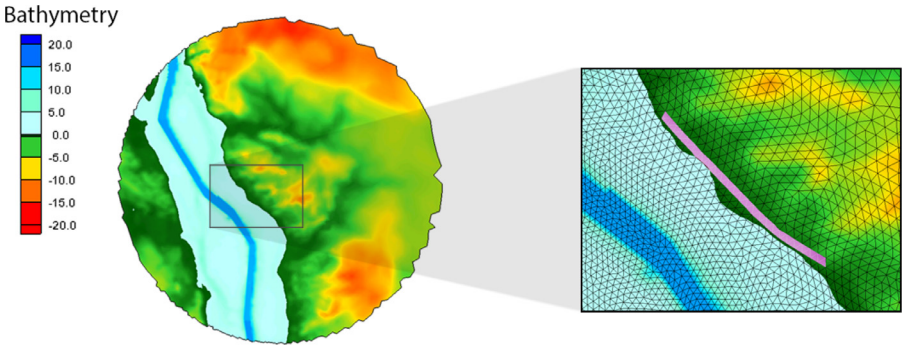

Fig. 19. Case 4: Local changes on the Cape Fear River.

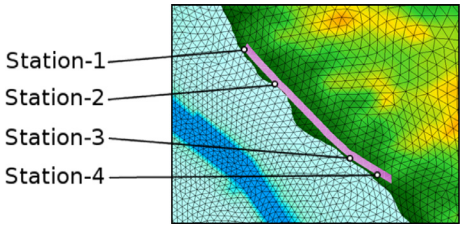

Fig. 20. Case 4: Recording stations along the river bank.

Table 1
Comparison of computational costs for Case 3.

Runtime	CPU hours	% of full scale
Full-scale grid	1488	100
CSM subdomain	102	6.88
ASM child domain	0.76	0.051

Table 2
Comparison of computational costs for Case 4.

Runtime	CPU hours	% of full scale
Full-scale grid	897	100
CSM subdomain	9.32	1.04
ASM child domain	0.19	0.021

evaluate the performance of the technique for the two cases in this section by comparing runtimes on a 64-core AMD Opteron Processor 6274 workstation using a serial prototype of ADCIRC++, an unoptimized pre-release version that nevertheless comes within about 15% of the (serial) performance of ADCIRC itself.

Tables 1 and 2 compare the computational costs of full-scale runs, CSM subdomains, and ASM child domains for the Brunswick intake canal and Silver Lake test cases. ASM child domain runtimes are average values that exclude the computational cost of the parent domains.

In both cases, the selected tolerances lead to high accuracy, with errors less than a millimeter. The runtime of a child domain is only a tiny fraction of a subdomain run, which itself is already a fraction of a full-scale run, so the combination constitutes a highly efficient use of resources. In terms of a cost breakdown for the components of ASM, it should be noted that little if any price is paid for adaptivity since memory management is optimized and changes are infrequent (typically less than once every thousand timesteps on average). As a result, costs for ASM subdomains are largely proportional to their extent.

6. Conclusions and future work

Subdomain modeling techniques, whether adaptive or conventional, are designed to assess the effects of incremental changes at an incremental computational cost. Motivated by engineering design and failure scenarios, such techniques also support scientific studies where one or more local properties of a physical domain are varied over a meaningful range of values. Subdomain modeling is predicated on the observation that many changes of interest induce responses with a local extent and without producing effects far from their origins—at least at the space and time scales of interest. Thus, we can eliminate calculations that fall outside the sphere of influence of those changes.

The adaptive approach presented in this study offers new, attractive features that complement conventional subdomain modeling. By automatically adjusting boundaries in response to domain changes, ASM relieves users from determining the sizes and shapes of subdomain grids, provides greater performance gains, and eliminates the verification steps required by CSM. Error indicator settings and other control parameters determine the behavior of the algorithm, allowing users to tailor its accuracy and efficiency according to their needs. Most importantly, the overall computational approach, where parent and child domains are analyzed concur-

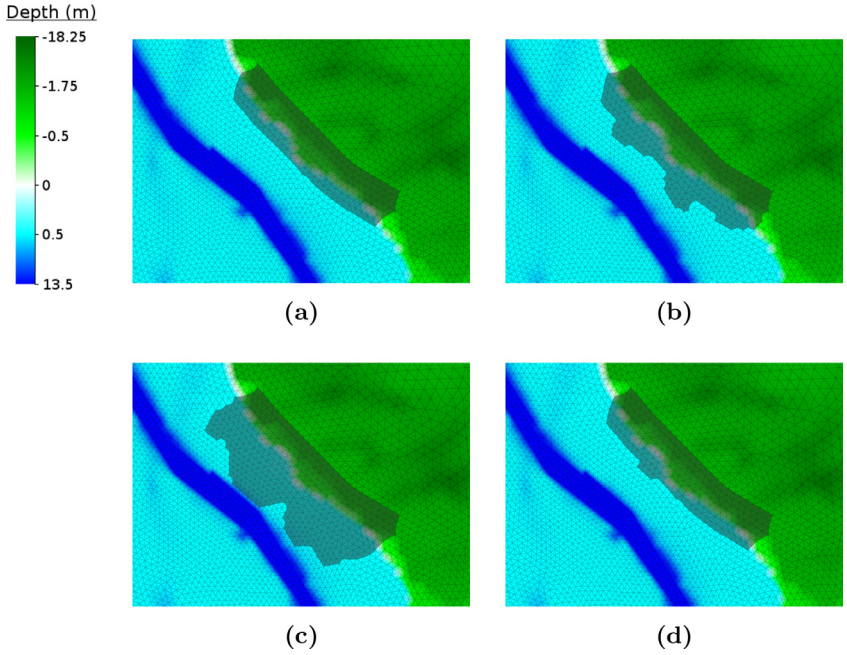

Fig. 21. Case 4: Progression of a Silver Lake child domain patch where the Manning's n values of two rows of nodes are set to 0.25, and the inner row of nodes is raised by 0.5 m. Snapshots: (a) initial extent, (b) extent at timestep 541 351, (c) largest extent (occurring at timestep 578 931), and (d) final extent (timestep 663 082).

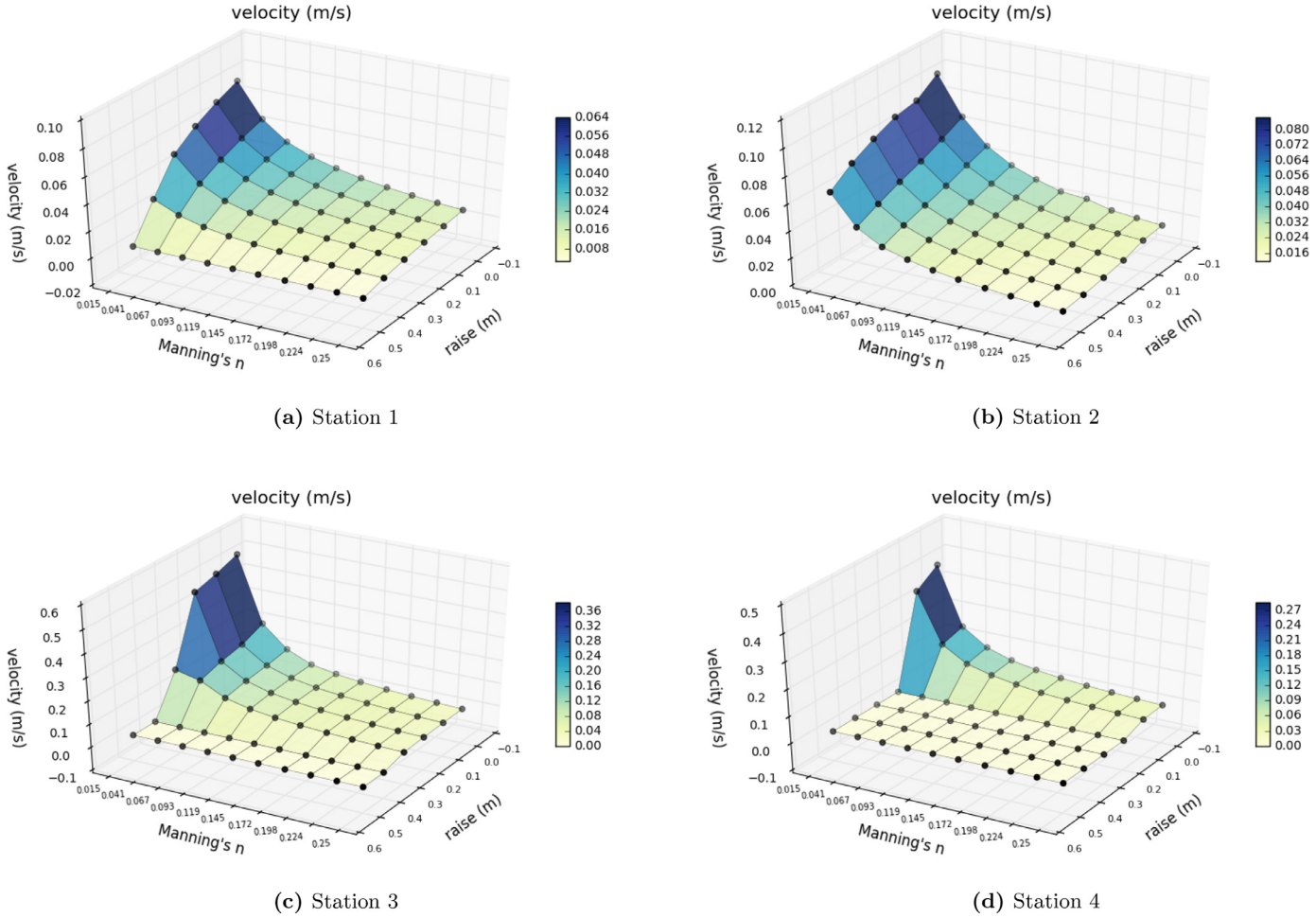

Fig. 22. Case 4: Velocities at recording stations for varying elevation raises and Manning's n values.

rently, imposes no arbitrary limitation on what is considered a parent, so users can employ conventional subdomains as parents and do so hierarchically to any degree of nesting desired, giving rise to the combined HSM approach we describe.[3]

The dynamic nature of ASM patches and other features, such as inter-domain communication, call for a software architecture that can accommodate them. We knew from the beginning we wanted to take advantage of ADCIRC's mature and well-tested formulation because of its many modeling strengths, but we also knew it would require an overhaul of static data structures that were conceived under a different set of assumptions. After hand translating parts of the code with some success, we grew confident we could implement about eighty percent of its features and create an adaptive code that, while not highly optimized, could serve as a prototype for ADCIRC++ and a proof of concept for ASM.

More recent versions of ADCIRC++ incorporate inter- and intra-domain parallelism on multicore architectures, which is realized by a thread pool and a timestepping routine that allows each concurrent domain to be executed with one or more dedicated threads. A phasing mechanism prevents a parent domain from updating itself until children access the data they need and likewise prevents children from moving ahead of the parent domain. To minimize false sharing among threads, decomposition of a grid into multiple patches is performed by METIS, a graph partitioning library also used by ADCIRC. Ongoing efforts are focused on improving the performance of intra-domain parallelism on large numbers of processors.

With regard to future directions, we anticipate using ASM to facilitate new population-based optimization strategies that might result in next-generation decision-support systems. More generally, we expect to continue our focus on tools and techniques for engineering users of large-scale storm surge models.

Acknowledgments

The authors thank Yoonhee Park for motivating the Silver Lake case study and offering the perspective of a landscape architect.

Team members of the Computational Modeling Group at NCSU provided helpful and constructive feedback during the course of this research, including Tristan Dyer, Fatima Bukhari, Yuliya Siamashka, Tucker Whitesides, and Xing Liu, and their contributions are appreciated.

[3] As in CSM, if (conventional) subdomains are in fact used, we note that they can be sized either more or less conservatively, and that the CSM technique actually allows boundary conditions for multiple subdomains to be produced in a single run at no extra cost other than file space (Baugh et al., 2015).

Appendix A. Illustrations of the first three stages of the adaptivity algorithm

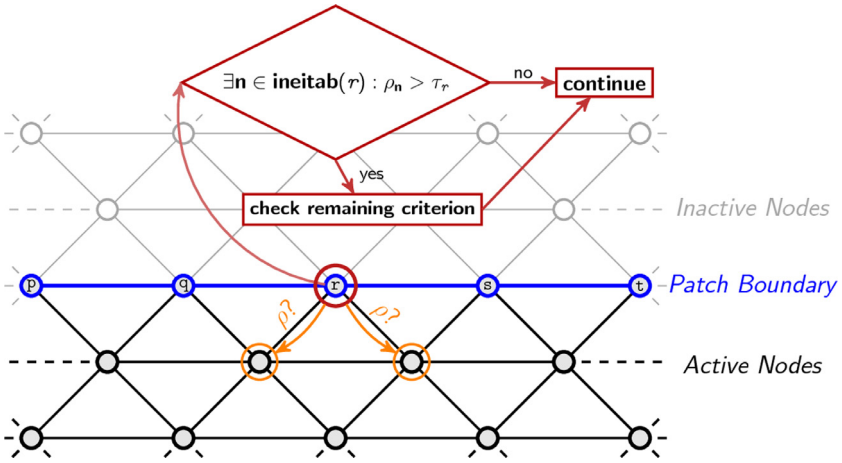

Fig. A.23. Illustration of criteria, evaluated at the first stage, for expanding a patch.

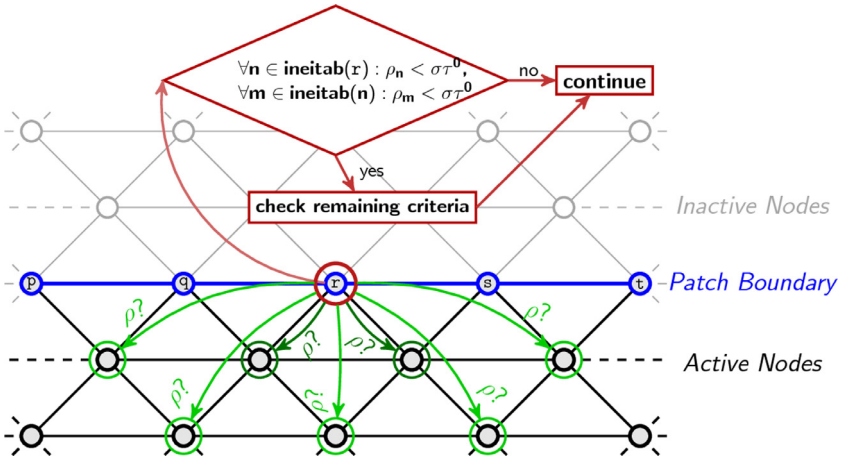

Fig. A.24. Illustration of criteria, evaluated at the first stage, for contracting a patch.

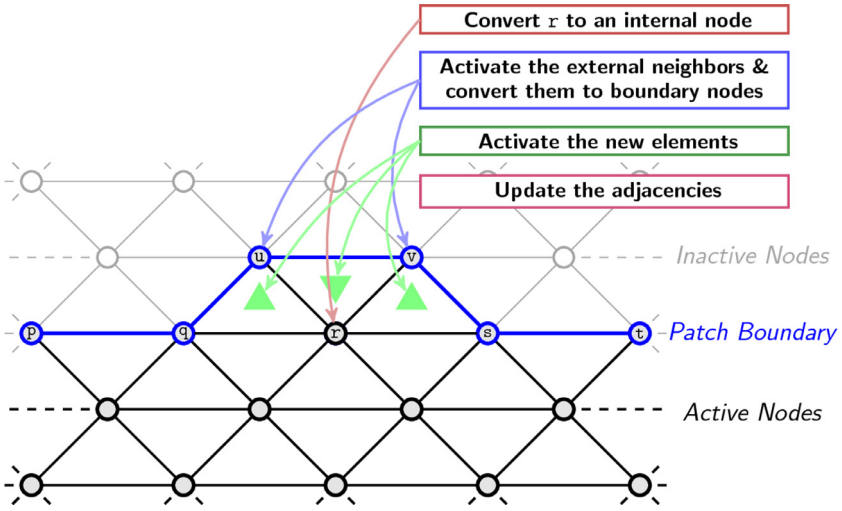

Fig. A.25. Stage 2: Main steps of the expansion of a child domain patch, where the patch boundary node r is marked for expansion at the first stage of the algorithm.

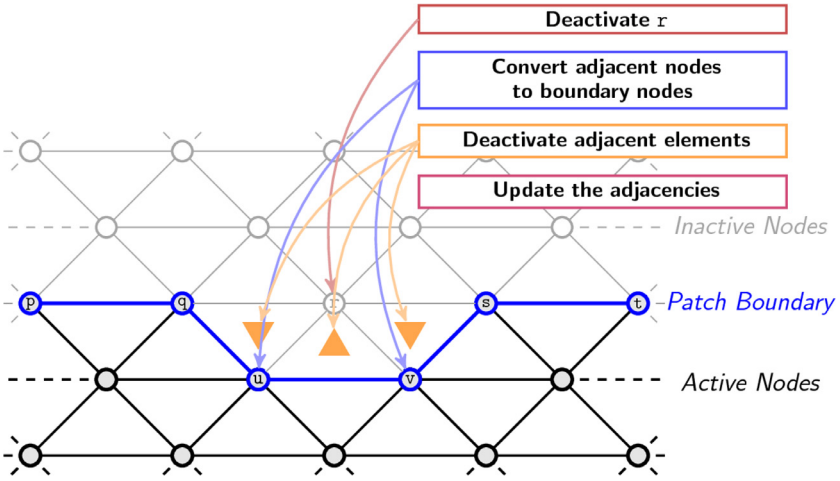

Fig. A.26. Stage 3: Main steps of the contraction of a child domain patch, where the patch boundary node r is marked for contraction at the first stage of the algorithm.

Appendix B. Expansion algorithm

Stage 2: Expansion Algorithm

- Let `expansionNodes` be the set of patch boundary nodes where the boundary is to be expanded. The set is assembled in Stage 1
- Let `nodesToAllocate` and `nodesToActivate` be empty sets of nodes
- Let `neighbors(i)` be the set of nodes adjacent to node `i`
- Let `elements(i)` be the set of elements incident on node `i`
- Let `nodes(i)` be the list of nodes of element `i`

```
for i in expansionNodes do
    Convert i to internal node                          ▷ to remove it from the set of patch boundary nodes
    for j in neighbors(i) do
        if j is not instantiated before then
            nodesToAllocate.insert(j)                   ▷ copy the external neighbor from parent grid
        else if j is inactive then                      ▷ (i.e., copied at a previous timestep)
            nodesToActivate.insert(j)

for i in nodesToAllocate do
    Copy i from parent
    Insert i to patch                                   ▷ and so activate i
    if i is mainland boundary or island boundary then
        Update the relevant vectors and parameters

for i in nodesToActivate do
    Insert i to patch                                   ▷ and so activate i
    Copy time-varying properties of node i from parent

for i in (nodesToActivate+nodesToAllocate) do
    for j in elements(i) do                             ▷ loop through elements connected to i
        if (j is not instantiated before) and (all three nodes of j are instantiated) then
            Copy j from parent
        if (j is inactive) and (all three nodes of j are active) then
            Insert j to patch                           ▷ and so activate j
            for k in nodes(j) do                        ▷ loop through nodes of element j
                Update elements(k)

for i in (nodesToActivate+nodesToAllocate) do
    Update neighbors(i)
    for j in neighbors(i) do
        Update neighbors(j)

for i in (nodesToActivate+nodesToAllocate) do
    if i is connected to at least one inactive node then
        Convert i to boundary node
    else                                                ▷ i.e., surrounded by active nodes
        Convert i to internal node
```

Fig. B.27. Expansion stage of the adaptivity algorithm.

Appendix C. Contraction algorithm

Stage 3: Contraction Algorithm

- Let `contractionNodes` be the set of patch boundary nodes where the boundary is to be contracted. This set is assembled in Stage 1
- Let `patch.boundaryNodes` be the set of boundary nodes of `patch`
- Let `deactivatedElements` be an empty set of elements
- Let `neighbors(i)` be the set of nodes adjacent to node `i`
- Let `elements(i)` be the set of elements incident on node `i`
- Let `nodes(i)` be the list of nodes of element `i`

```
1: for i in patch.boundaryNodes do
     if (i is connected only to patch boundary nodes) and (i is not initially active) then
       contractionNodes.insert(i)

2: for i in contractionNodes do
     if (i is not at the boundary anymore)
      or (i is next to an expansion node)
      or (i is next to a recently activated node) then
       contractionNodes.remove(i)

3: for i in contractionNodes do
     Deactivate i
     Convert i to internal node                ▷ to remove it from the set of patch boundary nodes
     for j in neighbors(i) do
       Update neighbors(j)
       if (j is internal node) then
         Convert j to boundary node
     for k in elements(i) do
       Deactivate k
       deactivatedElements.insert(k)
       for l in nodes(k) do                    ▷ loop through nodes of element k
         Update elements(l)

4: for i in deactivatedElements do
     if any node j of i is not connected to any active element then
       Deactivate j
       Convert j to internal node              ▷ to remove it from the set of patch boundary nodes
       for k in neighbors(j) do
         Update neighbors(k)
         if (k is internal node) then
           Convert k to boundary node
     if any two nodes of i are disconnected then   ▷ i.e., do not have a common element anymore
       Update neighbors of both nodes
```

Fig. C.28. Contraction stage of the adaptivity algorithm.

References

ADCIRC, 2016. The official ADCIRC website. http://adcirc.org. Accessed: 2016-12-14.

Altuntas, A., 2016. An Adaptive Multi-Analysis Technique and Software Architecture for Ocean Circulation Models. North Carolina State University Ph.D. thesis.

Baugh, J., Altuntas, A., 2016. Modeling a discrete wet-dry algorithm for hurricane storm surge in Alloy. In: Abstract State Machines, Alloy, B, TLA, VDM, and Z: 5th International Conference, ABZ 2016, Linz, Austria, May 23–27, 2016, Proceedings. Springer International Publishing, Cham, pp. 256–261.

Baugh, J., Altuntas, A., Dyer, T., Simon, J., 2015. An exact reanalysis technique for storm surge and tides in a geographic region of interest. Coastal Eng. 97, 60–77.

Baugh, J.W., Rehak, D.R., 1992. Data abstraction in engineering software development. J. Comput. Civil Eng. 6, 282–301.

Behrens, J., 1998. Atmospheric and ocean modeling with an adaptive finite element solver for the shallow-water equations. Appl. Numer. Math. 26, 217–226.

Berger, M.J., Colella, P., 1989. Local adaptive mesh refinement for shock hydrodynamics. J. Comput. Phys. 82, 64–84.

Berger, M.J., George, D.L., LeVeque, R.J., Mandli, K.T., 2011. The GeoClaw software for depth-averaged flows with adaptive refinement. Adv. Water Resour. 34, 1195–1206.

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

Blain, C., Westerink, J., Luettich, R., 1994. The influence of domain size on the response characteristics of a hurricane storm surge model. J. Geophys. Res. 99, 18467–18479.

Butler, T., Graham, L., Estep, D., Dawson, C., Westerink, J., 2015. Definition and solution of a stochastic inverse problem for the Manning's n parameter field in hydrodynamic models. Adv. Water Resour. 78, 60–79.

Dietrich, J., Kolar, R., Dresback, K., 2008. Mass residuals as a criterion for mesh refinement in continuous Galerkin shallow water models. J. Hydraul. Eng. 134, 520–532.

Dietrich, J., Kolar, R., Luettich, R.A., 2004. Assessment of ADCIRC's wetting and drying algorithm. Dev. Water Sci. 55, 1767–1778.

Dietrich, J.C., Tanaka, S., Westerink, J.J., Dawson, C., Luettich Jr, R., Zijlema, M., Holthuijsen, L.H., Smith, J., Westerink, L., Westerink, H., 2012. Performance of the unstructured-mesh, SWAN+ADCIRC model in computing hurricane waves and surge. J. Sci. Comput. 52, 468–497.

Dresback, K., Kolar, R., 2000. An implicit time-marching algorithm for 2-D GWC shallow water models. In: Bentley, L.R. (Ed.), Computational Methods in Water Resources XIII, 2, pp. 913–920.

Dresback, K.M., 2005. Algorithmic Improvements and Analyses of the Generalized Wave Continuity Equation Based Model ADCIRC. University of Oklahoma Ph.D. thesis.

Dyer, T., Baugh, J., 2016. SMT: an interface for localized storm surge modeling. Adv. Eng. Software 92, 27–39.
Eskilsson, C., 2011. An hp-adaptive discontinuous Galerkin method for shallow water flows. Int. J. Numer. Methods Fluids 67, 1605–1623.
Graham, L.C., 2015. Adaptive Measure-Theoretic Parameter Estimation for Coastal Ocean Modeling. The University of Texas at Austin Ph.D. thesis.
Haddad, J., Lawler, S., Ferreira, C.M., et al., 2015. Wetlands as a nature-based coastal defense: a numerical modeling and field data integration approach to quantify storm surge attenuation for the Mid-Atlantic region. In: OCEANS 2015-MTS/IEEE Washington. IEEE, pp. 1–6.
Hoffman, J.D., Frankel, S., 2001. Numerical Methods for Engineers and Scientists. CRC Press.
Kubatko, E.J., Bunya, S., Dawson, C., Westerink, J.J., 2009. Dynamic p-adaptive Runge–Kutta discontinuous Galerkin methods for the shallow water equations. Comput. Methods Appl. Mech. Eng. 198, 1766–1774.
LeVeque, R.J., 2007. Finite Difference Methods for Ordinary and Partial Differential Equations: Steady-State and Time-Dependent Problems, vol. 98. SIAM.
Liang, Q., Zang, J., Borthwick, A.G., Taylor, P.H., 2007. Shallow flow simulation on dynamically adaptive cut cell quadtree grids. Int. J. Numer. Methods Fluids 53, 1777–1799.
Luettich, R.A., Westerink Jr, J.J., Scheffner, N.W., 1992. ADCIRC: An Advanced Three--Dimensional Circulation Model for Shelves, Coasts and Estuaries, Report 1: Theory and Methodology of ADCIRC-2DDI and ADCIRC-3DL. Technical Report, DRP-92-6. U.S. Army Engineers Waterways Experiment Station, Vicksburg, MS. Dredging Research Program.
Marrocu, M., Ambrosi, D., 1999. Mesh adaptation strategies for shallow water flow. Int. J. Numer. Methods Fluids 31, 497–512.
Militello, A., Kraus, N.C., 2001. Site Investigation, Report 4: Evaluation of Flood and Ebb Shoal Sediment Source Alternatives for the West of Shinnecock Interim Project, New York. Technical Report. Shinnecock Inlet, New York,. DTIC Document.
Morang, A., 1999. Site Investigation, Report 1: Morphology and Historical Behavior. Technical Report. Shinnecock Inlet, New York,. DTIC Document.
Tanaka, S., Bunya, S., Westerink, J.J., Dawson, C., Luettich, R.A., 2011. Scalability of an unstructured grid continuous Galerkin based hurricane storm surge model. J. Sci. Comput. 46, 329–358.
Tate, J.N., Berger, R., Stockstill, R.L., 2006. Refinement indicator for mesh adaption in shallow-water modeling. J. Hydraul. Eng. 132, 854–857.
Williams, G.L., Morang, A., Lillycrop, L., 1998. Site Investigation, Report 2: Evaluation of Sand Bypass Options. Technical Report. Shinnecock Inlet, New York,. DTIC Document.