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

[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.

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

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

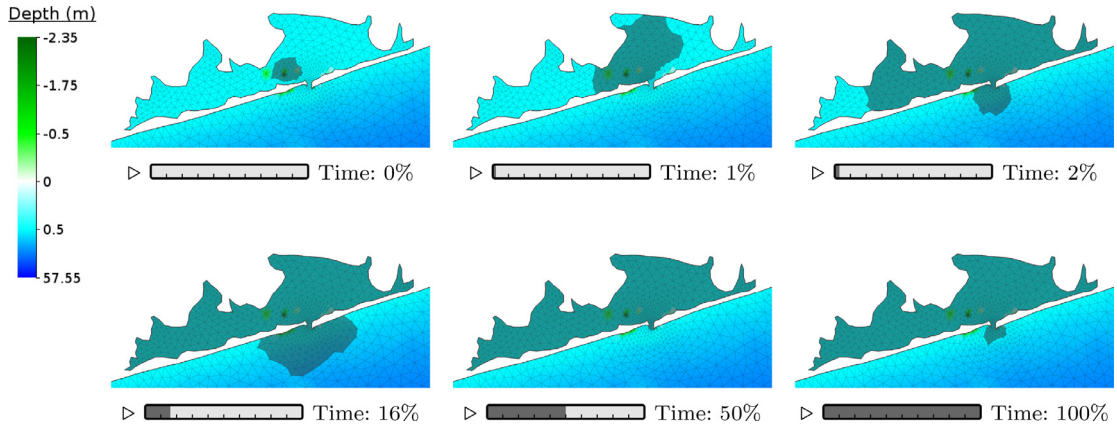

Fig. 1. Expansion and contraction of a locally modified Shinnecock Inlet child domain patch at various timesteps.

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