Discontinuous Galerkin unsteady discrete adjoint method for real-time efficient tsunami simulations

Accepted for publication in Journal of Computational Physics on August 21, 2012

Sébastien Blaise[a,b], Amik St-Cyr[a,c], Dimitri Mavriplis[d], Brian Lockwood[d]

[a]*Institute for Mathematics Applied to Geosciences, National Center for Atmospheric Research, Boulder CO, USA*
[b]*Institute of Mechanics, Materials and Civil Engineering, Université catholique de Louvain, Louvain-la-Neuve, BELGIUM*
[c]*Royal Dutch Shell, Houston TX, USA*
[d]*Department of Mechanical Engineering, University of Wyoming, Laramie WY, USA*

Abstract

An unsteady discrete adjoint implementation for a discontinuous Galerkin model solving the shallow water wave equations on the sphere is presented. Its use for tsunami simulations is introduced to reconstruct the initial condition automatically from buoy measurements. Based on this feature, a real-time tsunami model is developed, using several numerical tools such as a high-order discretization, *hp*-refinement, parallel dynamic load balancing and adjoint-based data assimilation. The model is able to reconstruct the tsunami source and accurately forecast its far-field propagation (e.g. from Japan to Chile, at a distance of about 17000 km) in a computational time 20 times faster than the physical propagation time, to which the data collecting time needs to be added. The work presented constitutes a step towards an efficient nonlinear tsunami warning model. Additional features could be added for more complete realistic forecasts.

Keywords:
discontinuous Galerkin, discrete adjoint, optimization, dynamic adaptation, tsunami simulation

Introduction

While most popular geophysical flow solvers are based on classical numerical methods such as finite differences, finite volumes or spectral methods, innovative techniques developed in other fields are increasingly being applied to those problems. During the last decade, several studies have focused on unstructured grid methods to increase the model resolution where it is needed (e.g. [1]). Despite improvements in computational power, multi-scale resolution is indeed essential to capture the interaction of phenomena whose spatial scales vary from a few to thousands of kilometers at a bearable computational cost.

The MUSE[1] model has been developed in this framework, with the aim of being an all scales unified simulation system that can exploit the next generation of petascale computers to simulate geophysical flows. It is based on the discontinuous Galerkin method, whose potential has already been demonstrated for marine modeling [2; 3; 4; 5] or atmospheric flows [6; 7; 8]. The main advantages of this method, in addition to the use of unstructured grids, consist of good stability properties thanks to the finite volume fluxes, efficient parallel scaling due the the high locality of the method and high-order accuracy. Further, the fact that no degree of freedom is shared between elements facilitates the implementation of nonconforming edges, characterized by hanging nodes (i.e. a node attached to the middle of an edge) or different polynomial orders on each side of the interface between elements. Hence, the discontinuous Galerkin method is well suited for dynamic adaptation of either the mesh [9; 10; 11], the polynomial order [12] or both [13; 14; 15]. Dynamic refinement concentrates the computing cost not only where, but also when it is needed, as illustrated by hurricane forecasts [16] or tsunami tracking applications [17; 14].

In order to exploit the multi-scale capabilities of the model, it is useful to develop its adjoint. The present study focuses on this task, which is then used to provide the response of a model to perturbations in state variables or design parameters. Adjoints are used in engineering, for example to optimize automatically the shape of aerodynamic parts such as airfoils [18] or to estimate an objective-oriented error in order to selectively adapt dynamically the resolution [19; 20; 15]. Two different adjoint approaches can be used: the discrete adjoint, considered in this study, or the continuous ad-

[1]http://muse.ucar.edu

joint. The continuous adjoint is analytically derived from the flow equations, then discretized while the discrete adjoint is derived at the level of the discrete equations [21]. The continuous adjoint is implemented similarly to any other equation. It can be used in the presence of non-differential operators such as limiters. However, the continuous adjoint equations are not always easy to derive, and the treatment of boundary conditions is considered not trivial. Using the discrete adjoint guarantees that the exact sensitivities of the discrete simulation process are obtained, ensuring robustness in the optimization process. The discrete adjoint has been used in geosciences either for goal-oriented error estimation [22] or data assimilation [23; 24].

In this work, we propose to use the adjoint method in the framework of real-time data assimilation, in order to forecast tsunami wave propagation. Such forecasts allow for immediate actions by local authorities to limit the human and material damages caused by potential tsunami-generated inundations. Hence, fast and accurate warnings are essential for the local emergency managers to take appropriate actions [25]. The earliest warnings are based on the analysis of seismic waves [26]. However, large errors can be generated by using indirect measurements of the tsunami generation. Additional valuable information can be obtained based on direct measurements of the elevation of the sea surface. While the distance from the epicenter of the earthquake to the measurement buoys is usually too large to provide a warning forecast for the closest coasts, this method is useful to compute accurately far-field tsunami amplitudes and propagation time. Moreover, even a late forecast for the closest coasts is useful to emergency managers in critical decisions regarding response, recovery and search-and-rescue [25].

Using direct measurements of the sea surface, the Method of Splitting Tsunami MOST[2] numerical model [27] relies upon a linearity assumption to provide a fast estimate of the tsunami behaviour. A set of unit sources covering active subduction zones is defined. For each unit source, a numerical simulation is pre-computed and all the data stored. A tsunami forecast for any initial condition belonging to the space defined by the unit sources is obtained by a linear combination of the pre-computed solutions. The coefficients of the linear combination have to be chosen such that the solution matches measurements, which can be done through an optimization procedure. However, this approach is only reliable if the flow is sufficiently

[2]`http://nctr.pmel.noaa.gov/model.html`

linear for the solution to be expressed as a linear combination of several pre-computed solutions. This linearity assumption is valid offshore, but coastal flows can be highly nonlinear, especially if inundation phenomena are taken into account. In those areas, the model solution can not be expressed as a linear combination of different model runs. For the same reasons, buoy measurements to be used with this method are limited to deep water locations where the dynamics are free from coastal nonlinear effects [25]. Coastal measurements may be easier to obtain and provide data earlier by being closer to the epicenter of the earthquake, allowing faster forecasts.

This study aims at investigating the use of the discrete adjoint to reconstruct the tsunami source without any linearity assumption. It is not intended as a full-fledged tsunami warning model, but rather as an exploratory study to assess the potential of innovative numerical methods, such as high-order discontinuous Galerkin discretizations, dynamic *hp*-adaptation, parallel processing with load-banlancing and adjoint optimization in the framework of the simulation of geophysical flows with focus on the real-time simulation of tsunamis. The first section is a description of the discrete adjoint for a general optimization process, while the second section explains how this concept can be applied in the case of a real-time tsunami application. The third section describes the modifications made in the nonconforming discontinuous Galerkin code to add adjoint capabilities. The last section before the conclusion presents the application of the model to the simulation of the recent 2011 tsunami in Japan, including the reconstruction of the initial condition.

1. Discrete adjoint for minimization

For several engineering applications in computational fluid dynamics, the design optimisation process corresponds to the minimization of a scalar objective function J, subject to the constraint that the discrete flow equations and boundary conditions are satisfied [28]:

$$\text{Find} \qquad \min_{\boldsymbol{\alpha}} J(\mathbf{U}(\boldsymbol{\alpha}), \boldsymbol{\alpha}) \qquad \text{subject to} \qquad \mathbf{N}(\mathbf{U}(\boldsymbol{\alpha}), \boldsymbol{\alpha}) = 0, \tag{1}$$

where $\boldsymbol{\alpha}$ is the vector of the design variables, i.e. the n parameters that can be modified to minimize the objective function. The vector $\mathbf{U}$ contains the degrees of freedom, describing the flow at each spatial node and each time step, while $\mathbf{N}(\mathbf{U}(\boldsymbol{\alpha}), \boldsymbol{\alpha})$ expresses collectively the flow equations and boundary conditions which are satisfied when its value is zero. In order to find this minimum, optimization methods such as the steepest descent [29] or the more advanced BFGS technique [30] require the knowledge of the derivatives

$$\frac{\mathrm{d}J}{\mathrm{d}\alpha_i} = \frac{\partial J}{\partial \mathbf{U}}\frac{\mathrm{d}\mathbf{U}}{\mathrm{d}\alpha_i} + \frac{\partial J}{\partial \alpha_i} \qquad \forall\ i = 1 \to n. \tag{2}$$

While $\frac{\partial J}{\partial \mathbf{U}}$ and $\frac{\partial J}{\partial \alpha_i}$ are generally easy to obtain, the computation of the sensitivity of the flow variables to the parameters $\frac{\mathrm{d}\mathbf{U}}{\mathrm{d}\alpha_i}$ is not straightforward.

The most direct method, in terms of implementation, is obtained by computing numerically the derivative by means of a finite difference scheme where each parameter is successively perturbed around a base solution $\mathbf{U}(\boldsymbol{\alpha})$:

$$\frac{\mathrm{d}\mathbf{U}}{\mathrm{d}\alpha_i} \simeq \frac{\mathbf{U}(\boldsymbol{\alpha} + \epsilon\boldsymbol{\delta_i}) - \mathbf{U}(\boldsymbol{\alpha})}{\epsilon} \qquad \forall\ i = 1 \to n, \tag{3}$$

where $\boldsymbol{\delta_i}$ is a unit vector such that only the i-th component is non-zero. The perturbation ϵ is chosen sufficiently small to limit the errors due to nonlinear effects, but sufficiently large to reduce numerical roundoff error. This method requires $n+1$ numerical simulations : one to compute the base solution $\mathbf{U}(\boldsymbol{\alpha})$ and n to compute the perturbed solutions.

An alternative consists in a linearization of the flow equations (1) around the base solution:

$$\frac{\partial \mathbf{N}}{\partial \mathbf{U}}\frac{\mathrm{d}\mathbf{U}}{\mathrm{d}\alpha_i} + \frac{\partial \mathbf{N}}{\partial \alpha_i} = 0 \qquad \forall\ i = 1 \to n. \tag{4}$$

By defining [28]

$$\mathbf{u_i} \triangleq \frac{\mathrm{d}\mathbf{U}}{\mathrm{d}\alpha_i}, \quad \mathbf{A} \triangleq \frac{\partial \mathbf{N}}{\partial \mathbf{U}}, \quad \mathbf{f_i} \triangleq -\frac{\partial \mathbf{N}}{\partial \alpha_i} \quad \text{and} \quad \mathbf{g}^T \triangleq \frac{\partial J}{\partial \mathbf{U}}, \tag{5}$$

it is easy to see that (4) corresponds to n linear systems

$$\mathbf{A}\mathbf{u_i} = \mathbf{f_i} \qquad \forall\ i = 1 \to n, \tag{6}$$

whose resolution provides the different vectors $\mathbf{u_i}$ to be introduced in the expression of the derivatives (2):

$$\frac{\mathrm{d}J}{\mathrm{d}\alpha_i} = \mathbf{g}^T\mathbf{u_i} + \frac{\partial J}{\partial \alpha_i} \qquad \forall\ i = 1 \to n. \tag{7}$$

This method does not need the additional parameter ϵ. However, $n+1$ simulations are still needed: one for the base solution, and n simulations using the linearized equations to compute the sensitivity of the flow variables to each of the parameters α_i. If the number of design variables grows, the amount of simulations needed can quickly become unaffordable, and a more efficient method is needed.

Assuming $\mathbf{A}$ is invertible, (6) can be written as

$$\begin{aligned} \mathbf{u_i} &= \mathbf{A}^{-1}\mathbf{f_i} \qquad \forall\ i = 1 \to n, && (8) \\ \mathbf{g}^T\mathbf{u_i} &= \mathbf{g}^T\mathbf{A}^{-1}\mathbf{f_i} \qquad \forall\ i = 1 \to n. && (9) \end{aligned}$$

By defining a last variable $\mathbf{v}^T \triangleq \mathbf{g}^T\mathbf{A}^{-1}$, (9) simplifies into

$$\mathbf{g}^T\mathbf{u_i} = \mathbf{v}^T\mathbf{f_i} \qquad \forall\ i = 1 \to n. \tag{10}$$

Introducing this identity in (7), it is possible to compute equivalently the derivative $\frac{\mathrm{d}J}{\mathrm{d}\alpha_i}$ by

$$\frac{\mathrm{d}J}{\mathrm{d}\alpha_i} = \mathbf{v}^T\mathbf{f_i} + \frac{\partial J}{\partial \alpha_i} \qquad \forall\ i = 1 \to n, \tag{11}$$

subject to the constraint directly derived from the definition of $\mathbf{v}^T$:

$$\mathbf{A}^T\mathbf{v} = \mathbf{g}. \tag{12}$$

After the computation of the base solution, there is only one linear system, i.e. equation (12) to solve. Whatever the number of design parameters: the knowledge of $\mathbf{v}$ allows the computation of the derivatives of the objective function J with regards to each of the n designs parameters α_i using vector dot products.

2. Application in a real-time Tsunami alert system

The procedure considered in this work for the real-time prediction of tsunamis is summarized in Figure 1. When an earthquake occurs, its effect on the perturbation of the sea level is quickly measured by the NOAA DART[3] tsunameters located around the epicenter. Those measurements are available in real-time, but they need to be filtered to remove the tidal signal and earthquake waves, which can be done by different approaches (e.g. [31; 32]).

[3]`http://nctr.pmel.noaa.gov/Dart`

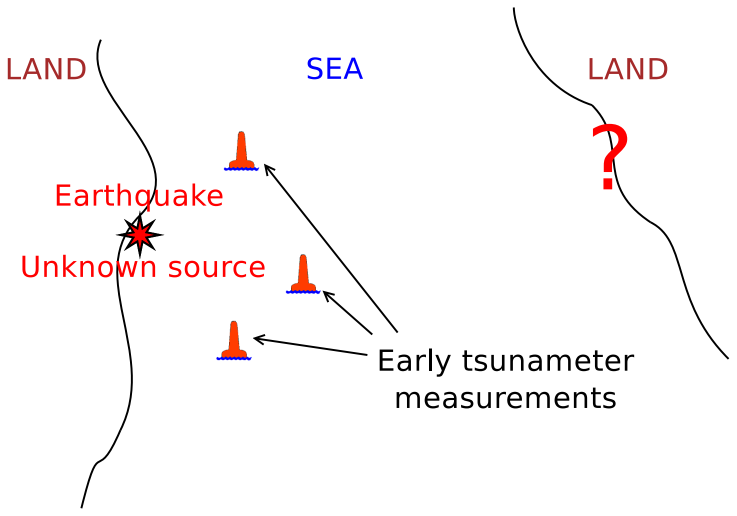

Figure 1: Real-time tsunami warning procedure. Once early measurements are available, the source of the tsunami is reconstructed to provide an initial condition to simulate the propagation of the tsunami through the world ocean.

Once early buoy measurements have been collected and filtered, they can be used in the process of reconstructing the tsunami source, i.e. the initial perturbation of the free-surface of the ocean generated by the earthquake. To this aim, we define an objective value to minimize, corresponding to the time-integration of the model error on the free-surface elevation η at the position of the n_b tsunameters:

$$J(\mathbf{U}(\boldsymbol{\alpha})) = \sum_{j=1}^{n_b} \left[\int_{t=0}^{t} \left(\eta^{\text{model}}(\mathbf{U}(\boldsymbol{\alpha}), \mathbf{x_j}, t) - \eta_j^{\text{meas}}(t) \right)^2 \mathrm{d}t \right], \tag{13}$$

where $\mathbf{x_j}$ is the coordinate of the j-th buoy. The way of computing η^{model} from the vector of degrees of freedom $\mathbf{U}$ will be discussed in the following section. In order to minimize this objective function, design parameters to act on are needed. To this aim, a set of unit sources (initial perturbations of the sea level) is considered. Those unit sources η^i_{unit} have to be chosen such that their adequate linear combination forms a good approximation of the tsunami initial perturbation η^{init}:

$$\eta^{\text{init}}(\mathbf{x}) \simeq \sum_{i=1}^{n} \alpha_i \eta_i^{\text{unit}}(\mathbf{x}). \tag{14}$$

Hence, the factors α_i multiplying each of those unit sources to establish the initial condition characterize the parameter space used to minimize the objective function $J(\mathbf{U}(\boldsymbol{\alpha}))$. It is now possible to use the concepts introduced in section 1 to minimize this objective and build the best approximation possible of the tsunami source given the parameters space. The number of parameters being generally high, it is more efficient to use the adjoint technique in the computation of the derivatives $\frac{\mathrm{d}J}{\mathrm{d}\alpha_i}$ rather than finite differenciation or a direct linearization.

Once the initial condition is reconstructed, a complete simulation of the propagation of the tsunami through the whole ocean can be performed to anticipate the effects of the tsunami on remote coastal areas.

3. Improving the model with adjoint capabilities

The MUSE shallow water wave model, described in detail by [14], is used in this work. It is a discontinuous Galerkin model, solving the shallow water wave equations on the sphere. It captures the dynamically varying key aspects of the flows by having the advantageous ability to locally modify the mesh as well as the order of interpolation within each element. The computational load is efficiently distributed amongst processors in parallel using a weighted recursive coordinate bisection strategy. The flows are expressed in three-dimensional Cartesian coordinates but tangentially constrained to the sphere by adding a Lagrange multiplier to the system of equations [33]. The model has been validated on classical atmospheric test cases such as those described by [34], and on the simulation of the February 2010 Chilean tsunami propagation [14]. Its multiscale strategy was able to reduce the computational time by an order of magnitude on the tsunami simulation, demonstrating its potential towards multi-resolution oceanic and atmospheric applications.

The modification of the code to obtain an adjoint model begins with the linearization of the discrete governing equations to derive the linear system given by equation (6). Using the third-order strong stability preserving Runge-Kutta scheme (SSP33, see [14] for more details), the discrete flow equations and boundary conditions can be expressed as

$$\mathbf{N} = \begin{bmatrix} \mathbf{I} & & & & & & & \\ \mathbf{I} & -\mathbf{I} & & & & & & \\ \frac{3}{4}\mathbf{I} & \frac{1}{4}\mathbf{I} & -\mathbf{I} & & & & & \\ \frac{1}{3}\mathbf{I} & & \frac{2}{3}\mathbf{I} & -\mathbf{I} & & & & \\ & & & \ddots & & & & \\ & & & & \mathbf{I} & -\mathbf{I} & & \\ & & & & \frac{3}{4}\mathbf{I} & \frac{1}{4}\mathbf{I} & -\mathbf{I} & \\ & & & & \frac{1}{3}\mathbf{I} & & \frac{2}{3}\mathbf{I} & -\mathbf{I} \\ & & & & & & & & \ddots \end{bmatrix} \begin{bmatrix} \mathbf{U}^0 \\ \mathbf{U}^{0,1} \\ \mathbf{U}^{0,2} \\ \vdots \\ \mathbf{U}^n \\ \mathbf{U}^{n,1} \\ \mathbf{U}^{n,2} \\ \vdots \\ \vdots \end{bmatrix} + \begin{bmatrix} \mathbf{U}^{\text{init}} \\ \mathbf{F}(\mathbf{U}^0)\Delta t \\ \frac{1}{4}\mathbf{F}(\mathbf{U}^{0,1})\Delta t \\ \frac{2}{3}\mathbf{F}(\mathbf{U}^{0,2})\Delta t \\ \vdots \\ \mathbf{F}(\mathbf{U}^n)\Delta t \\ \frac{1}{4}\mathbf{F}(\mathbf{U}^{n,1})\Delta t \\ \frac{2}{3}\mathbf{F}(\mathbf{U}^{n,2})\Delta t \\ \vdots \end{bmatrix} = 0, \qquad (15)$$

where $\mathbf{U}^{a,b}$ is the vector of the discrete degrees of freedom at the b-th Runge-Kutta stage of the a-th time iteration (no b means the vector corresponds to the beginning of a complete Runge-Kutta iteration), while $\mathbf{I}$ is the identity matrix of the corresponding size. The time step is denoted Δt, and $\mathbf{F}(\mathbf{U}^{a,b})$ is the discrete right hand side of the governing equations at the corresponding time iteration (see [14] for details). In the case of an explicit discontinuous Galerkin code, no matrix needs to be built, and the equation can be computed locally element by element. The solution at any time iteration depends only on the solutions computed at the previous steps. Hence, the equations are advanced in time in a forward step-by-step fashion.

It is now possible to derive $\mathbf{A}$ and $\mathbf{f_i}$ needed to build the linear system (6) corresponding to the linearized equations :

$$\mathbf{A} = \begin{bmatrix} \mathbf{I} & & & \\ & \begin{bmatrix} & \mathbf{A_b^0} & \end{bmatrix} & & \\ & & \ddots & \\ & & & \begin{bmatrix} & \mathbf{A_b^n} & \end{bmatrix} \\ & & & & \ddots \end{bmatrix}, \ \mathbf{f_i} = \begin{bmatrix} -\frac{\partial \mathbf{U}^{\text{init}}}{\partial \alpha_i} \\ \mathbf{0} \\ \vdots \\ \\ \\ \end{bmatrix}, \qquad (16)$$

with

$$\mathbf{A_b^n} = \begin{bmatrix} \mathbf{I} + \frac{\partial \mathbf{F}(\mathbf{U_i}^n)}{\partial \mathbf{U_i}^n}\Delta t & -\mathbf{I} & & \\ \frac{3}{4}\mathbf{I} & \frac{1}{4}\left(\mathbf{I} + \frac{\partial \mathbf{F}(\mathbf{U_i}^{n,1})}{\partial \mathbf{U_i}^{n,1}}\Delta t\right) & -\mathbf{I} & \\ \frac{1}{3}\mathbf{I} & & \frac{2}{3}\left(\mathbf{I} + \frac{\partial \mathbf{F}(\mathbf{U_i}^{n,2})}{\partial \mathbf{U_i}^{n,2}}\Delta t\right) & -\mathbf{I} \end{bmatrix}. \qquad (17)$$

After some manipulations, this linear system can be expressed as

$$\begin{bmatrix} \mathbf{I} & & & & & & & & \\ \mathbf{I} & -\mathbf{I} & & & & & & & \\ \frac{3}{4}\mathbf{I} & \frac{1}{4}\mathbf{I} & -\mathbf{I} & & & & & & \\ \frac{1}{3}\mathbf{I} & & \frac{2}{3}\mathbf{I} & -\mathbf{I} & & & & & \\ & & & \ddots & & & & & \\ & & & & \mathbf{I} & -\mathbf{I} & & & \\ & & & & \frac{3}{4}\mathbf{I} & \frac{1}{4}\mathbf{I} & -\mathbf{I} & & \\ & & & & \frac{1}{3}\mathbf{I} & & \frac{2}{3}\mathbf{I} & -\mathbf{I} & \\ & & & & & & & & \ddots \end{bmatrix} \begin{bmatrix} \mathbf{u_i}^{0} \\ \mathbf{u_i}^{0,1} \\ \mathbf{u_i}^{0,2} \\ \vdots \\ \mathbf{u_i}^{n} \\ \mathbf{u_i}^{n,1} \\ \mathbf{u_i}^{n,2} \\ \vdots \\ \vdots \end{bmatrix} + \begin{bmatrix} \frac{\partial \mathbf{U}^{\text{init}}}{\partial \alpha_i} \\ \frac{\partial \mathbf{F}(\mathbf{U}^0)}{\partial \mathbf{U}^0} \mathbf{u_i^0} \Delta t \\ \frac{1}{4}\frac{\partial \mathbf{F}(\mathbf{U}^{0,1})}{\partial \mathbf{U}^{0,1}} \mathbf{u_i^{0,1}} \Delta t \\ \frac{2}{3}\frac{\partial \mathbf{F}(\mathbf{U}^{0,2})}{\partial \mathbf{U}^{0,2}} \mathbf{u_i^{0,2}} \Delta t \\ \vdots \\ \frac{\partial \mathbf{F}(\mathbf{U}^n)}{\partial \mathbf{U}^n} \mathbf{u_i^n} \Delta t \\ \frac{1}{4}\frac{\partial \mathbf{F}(\mathbf{U}^{n,1})}{\partial \mathbf{U}^{n,1}} \mathbf{u_i^{n,1}} \Delta t \\ \frac{2}{3}\frac{\partial \mathbf{F}(\mathbf{U}^{n,2})}{\partial \mathbf{U}^{n,2}} \mathbf{u_i^{n,2}} \Delta t \\ \vdots \end{bmatrix} = 0, \tag{18}$$

which is very similar in shape to the discrete flow system (15). Hence the linearization of the model to obtain $\frac{\mathrm{d}\mathbf{U}}{\mathrm{d}\alpha_i}$ consists in

1. using $\frac{\partial \mathbf{U}^{\text{init}}}{\partial \alpha_i}$ instead of $\mathbf{U}^{\text{init}}$ as initial condition,
2. replacing the discrete right hand side by its derivative with respect to $\mathbf{U}^{a,b}$, multiplied by $\mathbf{u_i}^{a,b}$, at each Runge-Kutta stage b of the a-th time-iteration. Note that the base solution $\mathbf{U}$ either needs to be computed simultaneously or reloaded at each time iteration from a pre-computed flow simulation.

The drawback of this approach is the requirement of a model run per dimension of the parameter space. To obtain the derivatives $\frac{\mathrm{d}\mathbf{U}}{\mathrm{d}\alpha_i}$ for any i with a single model run, one needs to solve the linear system for the dual problem (12). To compute $\mathbf{g} = \frac{\partial J}{\partial \mathbf{U}}$ at the discrete level, the discrete version of the objective function (13) at a time iteration a, corresponding to the time t^a is constructed as:

$$\begin{aligned} J(\mathbf{U}(\boldsymbol{\alpha})) = \sum_{j=1}^{n_b} \sum_{k=1}^{a} \Delta t \Bigg[& \frac{1}{6} \left(\eta^{\text{model}}(\mathbf{U}(\boldsymbol{\alpha}), \mathbf{x_j}, t^k) - \eta_j^{\text{meas}}(t^k) \right)^2 \\ & + \frac{1}{6} \left(\eta^{\text{model}}(\mathbf{U}(\boldsymbol{\alpha}), \mathbf{x_j}, t^{k,1}) - \eta_j^{\text{meas}}(t^{k,1}) \right)^2 \\ & + \frac{2}{3} \left(\eta^{\text{model}}(\mathbf{U}(\boldsymbol{\alpha}), \mathbf{x_j}, t^{k,2}) - \eta_j^{\text{meas}}(t^{k,2}) \right)^2 \Bigg]. \end{aligned} \tag{19}$$

The sum over the time steps corresponds to the time integration of the error. The associated coefficients (i.e. 1/6, 1/6 and 2/3) have been derived from the SSP33 Runge-Kutta time discretization scheme to ensure that this time integration is done similarly to the time integration of the discrete equations.

The elevation of the sea surface computed by the model at the coordinate $\mathbf{x_j}$, time iteration a and Runge-Kutta step b is computed using the vector product

$$\eta^{\text{model}}(\mathbf{U}(\boldsymbol{\alpha}), \mathbf{x_j}, t^{a,b}) = \boldsymbol{\eta}^{a,b}(\mathbf{U}(\boldsymbol{\alpha})) \cdot \boldsymbol{\phi}(\mathbf{x_j}), \tag{20}$$

where $\boldsymbol{\eta}^{a,b}(\mathbf{U}(\boldsymbol{\alpha}))$ is the subset of $\mathbf{U}(\boldsymbol{\alpha})$ related to the degrees of freedom for the elevation of the free-surface at time iteration a and Runge-Kutta step b, while the vector $\boldsymbol{\phi}$ contains the associated shape functions which are zero everywhere except in the element to which they belong (see [14] for details). The vector $\mathbf{g}$ is obtained by a differentiation of the objective function (19) with respect to $\mathbf{U}$, leading to the linear system (12) for the adjoint problem:

$$\mathbf{A}^T = \begin{bmatrix} \mathbf{I} & & & & \\ & \begin{bmatrix} (\mathbf{A_b^0})^T \end{bmatrix} & & & \\ & & \ddots & & \\ & & & \begin{bmatrix} (\mathbf{A_b^n})^T \end{bmatrix} & \\ & & & & \ddots \end{bmatrix}, \quad \mathbf{g} = \begin{bmatrix} \mathbf{0} \\ \sum_{j=1}^{n_b} \frac{2}{6}\Delta t \left(\boldsymbol{\eta}^0 \boldsymbol{\phi}(\mathbf{x_j}) - \eta_j^{\text{meas}}(t^0)\right) \boldsymbol{\phi}(\mathbf{x_j}) \\ \sum_{j=1}^{n_b} \frac{2}{6}\Delta t \left(\boldsymbol{\eta}^{0,1} \boldsymbol{\phi}(\mathbf{x_j}) - \eta_j^{\text{meas}}(t^{0,1})\right) \boldsymbol{\phi}(\mathbf{x_j}) \\ \sum_{j=1}^{n_b} \frac{4}{3}\Delta t \left(\boldsymbol{\eta}^{0,2} \boldsymbol{\phi}(\mathbf{x_j}) - \eta_j^{\text{meas}}(t^{0,2})\right) \boldsymbol{\phi}(\mathbf{x_j}) \\ \vdots \\ \sum_{j=1}^{n_b} \frac{2}{6}\Delta t \left(\boldsymbol{\eta}^n \boldsymbol{\phi}(\mathbf{x_j}) - \eta_j^{\text{meas}}(t^n)\right) \boldsymbol{\phi}(\mathbf{x_j}) \\ \sum_{j=1}^{n_b} \frac{2}{6}\Delta t \left(\boldsymbol{\eta}^{n,1} \boldsymbol{\phi}(\mathbf{x_j}) - \eta_j^{\text{meas}}(t^{n,1})\right) \boldsymbol{\phi}(\mathbf{x_j}) \\ \sum_{j=1}^{n_b} \frac{4}{3}\Delta t \left(\boldsymbol{\eta}^{n,1} \boldsymbol{\phi}(\mathbf{x_j}) - \eta_j^{\text{meas}}(t^{n,2})\right) \boldsymbol{\phi}(\mathbf{x_j}) \\ \vdots \end{bmatrix}. \tag{21}$$

After some manipulations, this linear system can be expressed as

$$\begin{bmatrix} \mathbf{I} & \mathbf{I} & \frac{3}{4}\mathbf{I} & \frac{1}{3}\mathbf{I} & & & & & \\ & -\mathbf{I} & \frac{1}{4}\mathbf{I} & & & & & & \\ & & -\mathbf{I} & \frac{2}{3}\mathbf{I} & & & & & \\ & & & -\mathbf{I} & & & & & \\ & & & & \ddots & \mathbf{I} & \frac{3}{4}\mathbf{I} & \frac{1}{3}\mathbf{I} & \\ & & & & & -\mathbf{I} & \frac{1}{4}\mathbf{I} & & \\ & & & & & & -\mathbf{I} & \frac{2}{3}\mathbf{I} & \\ & & & & & & & -\mathbf{I} & \\ & & & & & & & & \ddots \end{bmatrix} \begin{bmatrix} \mathbf{v_i}^0 \\ \mathbf{v_i}^{0,1} \\ \mathbf{v_i}^{0,2} \\ \vdots \\ \mathbf{v_i}^n \\ \mathbf{v_i}^{n,1} \\ \mathbf{v_i}^{n,2} \\ \vdots \\ \vdots \end{bmatrix} + \begin{bmatrix} \left(\frac{\partial \mathbf{F}(\mathbf{U_i}^0)}{\partial \mathbf{U_i}^0}\right)^t \mathbf{v_i^{0,1}} \Delta t \\ \frac{1}{4}\left(\frac{\partial \mathbf{F}(\mathbf{U_i}^{0,1})}{\partial \mathbf{U_i}^{0,1}}\right)^t \mathbf{v_i^{0,2}} \Delta t \\ \frac{2}{3}\left(\frac{\partial \mathbf{F}(\mathbf{U_i}^{0,2})}{\partial \mathbf{U_i}^{0,2}}\right)^t \mathbf{v_i^{1}} \Delta t \\ \vdots \\ \left(\frac{\partial \mathbf{F}(\mathbf{U_i}^n)}{\partial \mathbf{U_i}^n}\right)^t \mathbf{v_i^{n,1}} \Delta t \\ \frac{1}{4}\left(\frac{\partial \mathbf{F}(\mathbf{U_i}^{n,1})}{\partial \mathbf{U_i}^{n,1}}\right)^t \mathbf{v_i^{n,2}} \Delta t \\ \frac{2}{3}\left(\frac{\partial \mathbf{F}(\mathbf{U_i}^{n+2})}{\partial \mathbf{U_i}^{n+2}}\right)^t \mathbf{v_i^{n+1}} \Delta t \\ \vdots \\ 0 \end{bmatrix} - \mathbf{g} = 0. \tag{22}$$

Hence, it is possible to convert the linearization of the forward code to an adjoint code by

1. setting the initial condition to zero and substracting the vector $\mathbf{g}$ defined in (21) to the system,

2. replacing the derivative of the right hand side by its transpose,
3. transposing the SSP33 Runge-Kutta coefficient describing the time discretization.

A^T being an upper triangular matrix, the solution at any time iteration depends only on the solutions computed at the latter steps, and the equations can be solved by back substitution which corresponds to a backward integration in time. For this reason, it is necessary when computing the base solution with a forward run to store the value of the degrees of freedom after each Runge-Kutta stage. For large problems, it can result in massive disk usage forcing the model to be disk bound, especially in the case of an explicit time-stepping where the time-step is relatively small due to the CFL condition. Although not used in this work, a checkpointing mechanism can be used to reduce the disk storage requirements, at the expense of additional recomputations [35].

The construction of the right-hand side for the linearized equations involves several products by matrices, corresponding to different discrete operators (mass matrices, differentiation matrices, constraint to handle the sphere, combinations of variables). Each of those operators need to be transposed. Further, the order in which the products are performed need to be inverted when the transposition is done. If the linearized right-hand side is obtained by n_p products of matrices $\mathbf{P}_j$:

$$\frac{\partial \mathbf{F}({\mathbf{U}_\mathbf{i}}^{a,b})}{\partial {\mathbf{U}_\mathbf{i}}^{a,b}} = \prod_{j=1}^{n_p} \mathbf{P}_j \ {\mathbf{U}_\mathbf{i}}^{a,b}, \tag{23}$$

the one to be used for the adjoint problem will be

$$\left(\frac{\partial \mathbf{F}({\mathbf{U}_\mathbf{i}}^{a,b})}{\partial {\mathbf{U}_\mathbf{i}}^{a,b}}\right)^T = \prod_{j=1}^{n_p} \mathbf{P}^T_{n_p-j+1} \ {\mathbf{U}_\mathbf{i}}^{a,b}, \tag{24}$$

where the superscripts a, b correspond to the b-th Runge-Kutta stage of the a-th time iteration. In this work, a conforming mesh is considered, but not a conforming order: neighboring elements can have different orders, hence sharing faces with different numbers of nodes. In this case, additional projection operators appear to transfer the data between each side of the interfaces between elements, which need to be transposed correctly and applied at the right moment following (24). The manner to transpose and change the order of the operations being implementation-specific, no further details will be provided. However, the modification of the code has to be handled with care in order to preserve the equivalence of the dual form (10). This identity can be used to check the correctness of the adjoint implementation compared to the forward linearization at each step of the development of the code. The linearization itself can be verified by comparing with sensitivities obtained using a finite difference scheme (3).

4. Application to the 2011 tsunami in Japan

The model with adjoint capabilities has been applied to the simulation of the tsunami generated by the earthquake off the coast of Japan that occurred on Friday, 11 March 2011. As mentioned in section 2, the simulation process consists of two steps: the reconstruction of the initial condition using an adjoint-based source optimization (section 4.1) and a complete far-field simulation of the propagation of the tsunami (section 4.2). Those two steps share similar characteristics. The computational domain encompasses the global world ocean. Hence, there is no open boundary condition to enforce. The bathymetry of the ocean is derived from the ETOPO data [36] and varies from 0 to about 9000 m. The bottom stress is computed using the Manning formula with a coefficient $n = 0.03$ matching the values usually employed in the literature.

4.1. Source optimization

The mesh used in this first step is generated using the Gmsh[4] software [37] and focuses on the area surrounding the earthquake location (Figure 2). Away from the region of interest (i.e. outside of the area between the epicenter of the earthquake and the DART measurement buoys), the characteristic size of the elements increases strongly such that more than 80% of the elements (1850 over 2227) are located where the resolution of the flow is important. Further, the polynomial order of the elements away from the area of interest is set to its minimum value (4x4 nodes per element).

No dynamic adaptation of the mesh or polynomial order is considered during the source optimization. While it is possible to implement this feature in the model (see e.g. [38]), additional projection operators to transfer

[4]`http://www.geuz.org/gmsh`

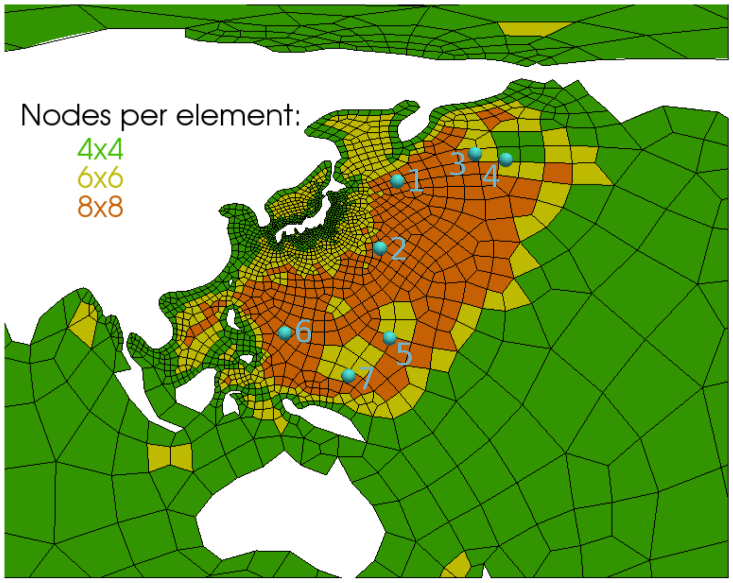

Figure 2: Computational mesh and number of nodes per element used for the first step of the simulation process: source optimization. The blue spheres correspond to the location of the DART buoys providing the free-surface elevation measurements to be used for the reconstruction of the initial condition. The Cartesian domain has been mapped to longitude/latitude coordinates.

the data when an element is modified need to be taken into account when constructing the adjoint. The different states of the mesh during the adjoint and forward runs need to match. Hence, it would be necessary to store at each time iteration the current mesh with the polynomial order used in each element, such that the adjoint model can load it to be coherent with the forward one. Given the relatively small domain of interest, using dynamic adaptation for the source optimization may not be worth the additional investment and cost. For similar reasons, a variable time resolution would require to keep track of the time step at each Runge-Kutta iteration. Since the time-step is limited by the celerity of the gravity waves, which is almost constant, there is no advantage to use a variable value; and it has been set to 4.5 seconds throughout.

To optimize the efficiency and stability of the simulation, the polynomial order varies spatially. Around coastal and shallow areas, small lower-order elements are used for stability: in unresolved areas, a high-order discretization is indeed susceptible to develop oscillations that may generate negative depths and make the model crash. Further, several small elements are needed to capture the geometry of the coastlines, which makes the use of a high-order discretization too costly. For large offshore elements, the polynomial order is increased to fully benefit from the high-order accuracy. To avoid a strong spatial change of resolution, the difference of the polynomial orders between two neighboring elements is constrained to 2.

The initial condition to be tuned is a combination of unit source functions defined by the NOAA center for tsunami research[5]. Those source functions are located along the known fault zones susceptible to generate a tsunami and are available for the entire Pacific Basin, Caribbean for the Atlantic region and Indian Ocean. A combination of eighteen unit sources, located around the epicenter of the earthquake, were chosen (Figure 3). Each unit source corresponds to a deformation due to an earthquake characterized by a fault length of 100 km, fault width of 50 km, and a slip value of 1 m [39]. The space of the design parameters in the source optimization is constituted by the coefficients multiplying each of those unit sources to compose the initial condition. Those coefficients are initially set to 0.1, and constrained in a range of physically acceptable values:

$$0 \leq \alpha_i \leq 0.33, \tag{25}$$

[5]`http://nctr.pmel.noaa.gov/propagation-database.html`

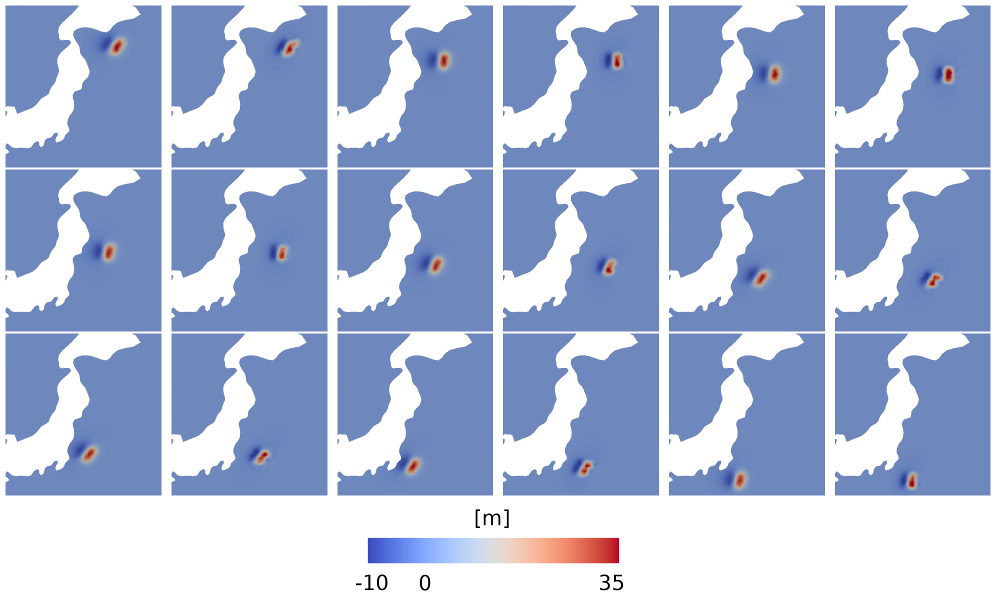

Figure 3: Unit tsunami source functions: elevation of the free-surface for each source. A linear combination of these sources is used to reconstruct the initial condition.

ensuring that the initial depth is always positive.

The source optimization procedure is driven by the L-BFGS-B[6] software [30; 40], a quasi-Newton code for bound-constrained optimization. The value of the objective function J and its derivatives $\frac{\mathrm{d}J}{\mathrm{d}\alpha_i}$ with respect to the design parameters α_i are obtained from the forward and adjoint runs and provided to L-BFGS-B. The latter computes a projected gradient, taking into account the set of variables that will be held at the bounds [41]. A new set of design parameters values, computed to minimize this gradient, is generated by L-BFGS-B for a new iteration of forward and adjoint runs. After each iteration, the initial condition converges towards a shape minimizing the objective function and its projected gradient (Figure 4). Table 1 shows that a total of 16 iterations are needed towards convergence. However, the criterion used for convergence (projected gradient $< 10^{-5}$ while J and α_i are of order 1) is rather severe and may be relaxed. After 8 iterations, the value of the objective function is almost definitive, and the error is negligible for the considered application. The average computational cost associated with each

[6]`http://users.eecs.northwestern.edu/~nocedal/lbfgsb.html`

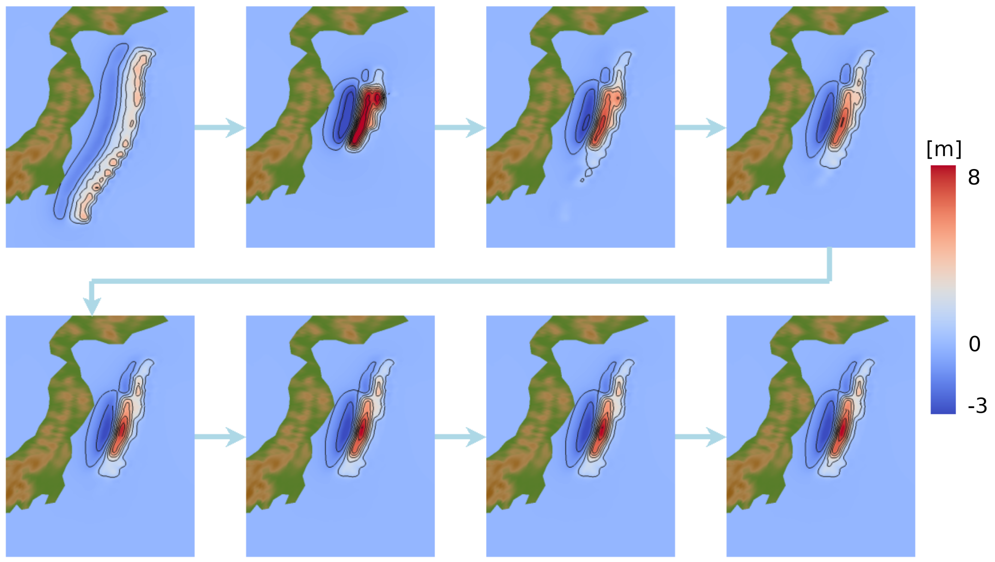

Figure 4: Initial condition before the 8 first forward/adjoint iterations of the source optimization. Contour line every 1.3 m.

iteration on 96 AMD 2.2 GHz Opteron processors of a Cray XT5m is 125 seconds for the forward run and 128 seconds for the adjoint run. Hence, a sequence of 8 iterations lasts around 33 minutes 44 seconds.

According to Table 2, several parameters are stuck at the zero lower bound, indicating that the associated unit sources cannot reduce the error. As the set of unit sources has been chosen simply based on a proximity criteria, some of them do not correspond to the actual tsunami source and should not be activated. Hence, the source optimization process automatically selects the sources that, by being able to reduce the error, are part of the actual tsunami initial condition.

The computed sensitivities have been compared with the ones obtained using the finite differentiation technique and the linearization of the equations described in section 1 to validate the correct implementation of the adjoint. This comparison was performed for the first iteration of the source optimization procedure, for each of the eighteen design parameters and is shown on Table 3. The parameter ϵ from (3) is chosen to be 10^{-7}, which is large enough compared with the machine precision, but small enough to limit the errors due to nonlinear effects. For most of the parameters, the relative error when comparing with finite differences is small and rather close

Iteration	$J(\boldsymbol{\alpha}) \cdot 10^{-3}$ [m^{2}s$\cdot 10^{-3}$]	projected gradient [m^{2}s$\cdot 10^{-3}$]
1	1.1056267738	2.300e-01
2	0.4903519590	3.300e-01
3	0.3532528041	2.057e-01
4	0.3300028885	6.775e-02
5	0.3232013498	4.172e-02
6	0.3209625334	3.452e-02
7	0.3206245630	1.149e-02
8	0.3205895815	6.812e-03
9	0.3205772213	4.823e-03
10	0.3205703566	3.096e-03
11	0.3205656472	9.224e-04
12	0.3205653295	2.670e-04
13	0.3205652873	1.317e-04
14	0.3205652795	7.179e-05
15	0.3205652769	2.350e-05
16	0.3205652766	2.309e-06

Table 1: Values of the objective function $J(\boldsymbol{\alpha})$ (scaled to be around 1) and its projected gradient for each forward/adjoint iteration of the source optimization.

i	α_i	α_i (double res)
1	0.0000	0.0000
2	0.0589	0.0391
3	0.0000	0.0000
4	0.1112	0.0787
5	0.2003	0.1850
6	0.0489	0.0499
7	0.2651	0.2705
8	0.0000	0.0000
9	0.2272	0.2092
10	0.0000	0.0000
11	0.0735	0.0663
12	0.0336	0.0180
13	0.0000	0.0000
14	0.0000	0.0088
15	0.0000	0.0000
16	0.0000	0.0000
17	0.0000	0.0000
18	0.0000	0.0000

Table 2: Values of the design parameters α_i at convergence, associated with the 18 unit sources for two optimization runs of different resolutions.

i	Adjoint	Finite differences	(rel. error)	Linearization	(rel. error)
1	545.89369124140035	545.89458668401562	1.6403e-06	545.89369124147311	1.3329e-13
2	309.51194848571919	309.50488105459510	2.2834e-05	309.51194848574244	7.5115e-14
3	714.45754849775346	714.45000486805009	1.0559e-05	714.45754849772027	4.6464e-14
4	51.94654524408983	51.94625631194072	5.5621e-06	51.94654524388099	4.0203e-12
5	-309.06529204770328	-309.06111984924303	-1.3499e-05	-309.06529204803832	-1.0840e-12
6	-98.21564376933510	-98.19515231935128	-2.0864e-04	-98.21564376972265	-3.9459e-12
7	-1183.39213001641292	-1183.40001953512751	-6.6669e-06	-1183.39213001618600	-1.9175e-13
8	1.41714986533523	1.41012219242753	4.9590e-03	1.41714986578511	3.1746e-10
9	-500.02168037345700	-500.02065758192714	-2.0455e-06	-500.02168037239215	-2.1296e-12
10	944.07435894297760	944.08967279994272	1.6221e-05	944.07435894337311	4.1895e-13
11	961.68910696649993	961.70494593797866	1.6470e-05	961.68910696695843	4.7676e-13
12	742.96681771204578	742.97355183111665	9.0638e-06	742.96681771172109	4.3702e-13
13	930.68312253188617	930.70583198560939	2.4401e-05	930.68312253242266	5.7645e-13
14	1213.62922647885603	1213.59977680192765	2.4266e-05	1213.62922647852179	2.7540e-13
15	857.26930802852178	857.30989401754346	4.7343e-05	857.26930802920856	8.0113e-13
16	2054.87664900041182	2054.89822982379656	1.0502e-05	2054.87664900030768	5.0678e-14
17	1772.98399749335999	1772.98019365494542	2.1454e-06	1772.98399749452165	6.5520e-13
18	1543.15813970857198	1543.14799526789693	6.5738e-06	1543.15813970817885	2.5476e-13

Table 3: Values of $\frac{\mathrm{d}J}{\mathrm{d}\alpha_i}$ [m^{2}s] at the first iteration for the 18 unit sources, computed using the adjoint method, finite differentiation and the forward linearization. The relative error is computed by taking the adjoint computation as reference.

to ϵ. The eighth parameter is an exception, which is not caused by a higher absolute error, but the normalization performed using a smaller sensitivity value: the relative error is higher because resulting from a division by a reference value closer to zero. The linearization, meanwhile, matches the adjoint model with an error close to machine precision, confirming a correct implementation.

4.2. Forward run

Once the initial condition is reconstructed, a complete forward run can be performed to simulate the propagation of the tsunami across the ocean. For this step, the mesh has a similar resolution around the epicenter of the earthquake than the mesh used for the optimization (Figure 5). However, since the domain of interest includes the whole globe, the resolution is increased along the global ocean coasts to represent more accurately the coastlines. The resulting mesh is made up of 6840 elements.

For a global simulation, dynamic adaptation is much more interesting, and its efficiency for tsunami simulations has been highlighted by [14]. We use a similar adaptation strategy but the behaviour has been slightly modified for better efficiency and stability in this application. In shallow and coastal areas, the polynomial order is limited to a maximum value that de-

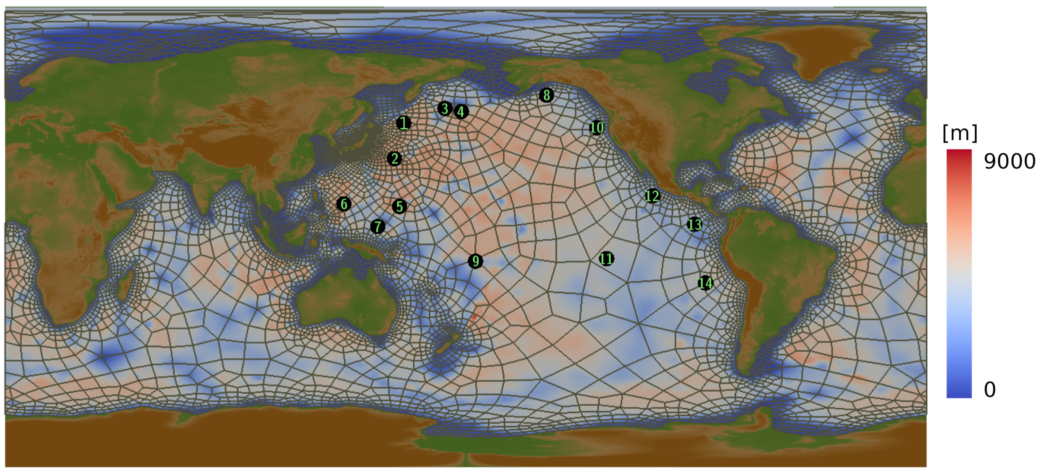

Figure 5: Sea depth at rest and initial computational mesh used for the forward run. The circles indicate the location of the DART boys used for the comparison of the model with measurements. The Cartesian domain has been mapped to longitude/latitude coordinates.

pends on the distance to the coasts and depth. Hence, the resolution in those regions can be increased only by mesh refinement. In the rest of the domain, polynomial refinement is preffered, and mesh refinement occurs when the polynomial order reaches its maximum. To avoid refinement where the resolution is initially sufficient, *hp*-refinement can only occur if the resolution is above a given threshold, which is about 45 km. Strong spatial changes of resolution are avoided by constraining the difference of the polynomial orders and refinement levels between two neighboring elements to be respectively 2 and 1.

The model has been run in parallel on 96 cores using domain decomposition with dynamic load-balancing (see [14] for details). While it takes about 22 hours for the tsunami to cross the Pacific Ocean, the numerical simulation is obtained in less than 31 minutes. The propagation of the tsunami can be observed in Figure 6, as well as the mesh with the number of nodes per element. As long as the tsunami travels through the Pacific Ocean, the order of interpolation and the mesh are adapted to precisely track the waves. The computational power is used effectively by concentrating the load at the front of the wave, where it is needed to accurately resolve the propagation and best match the arrival times.

The elevation of the free-surface has been compared with the DART data at fourteen different stations (Figure 5), including those used during the op-

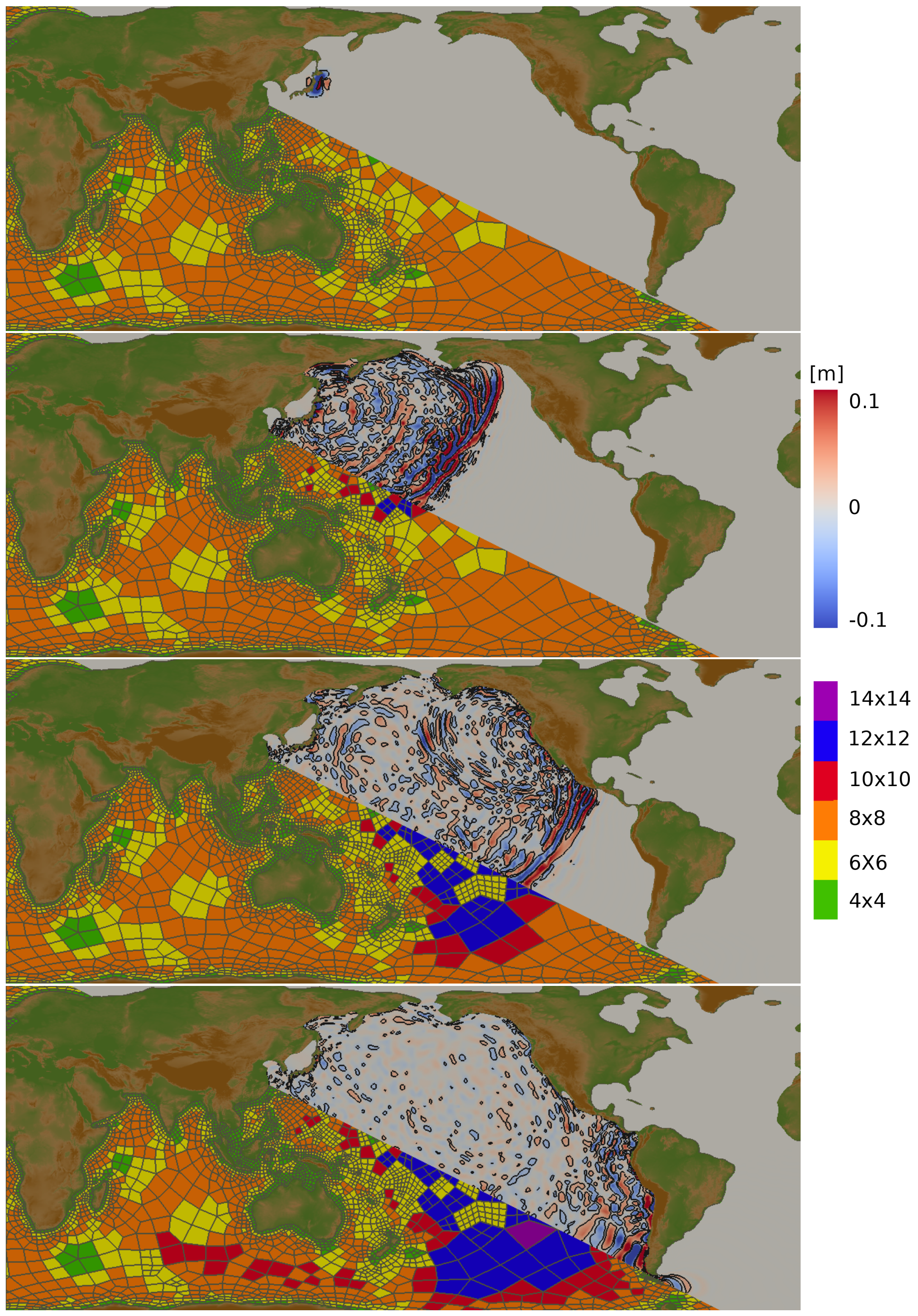

Figure 6: Propagation of the wave after 0h, 7.5h, 15h and 22.5h. UPPER PART: Free-surface elevation. The isocontour lines have been set to ±1 cm. LOWER PART: State of the mesh with number of nodes per element. The Cartesian domain has been mapped to longitude/latitude coordinates.

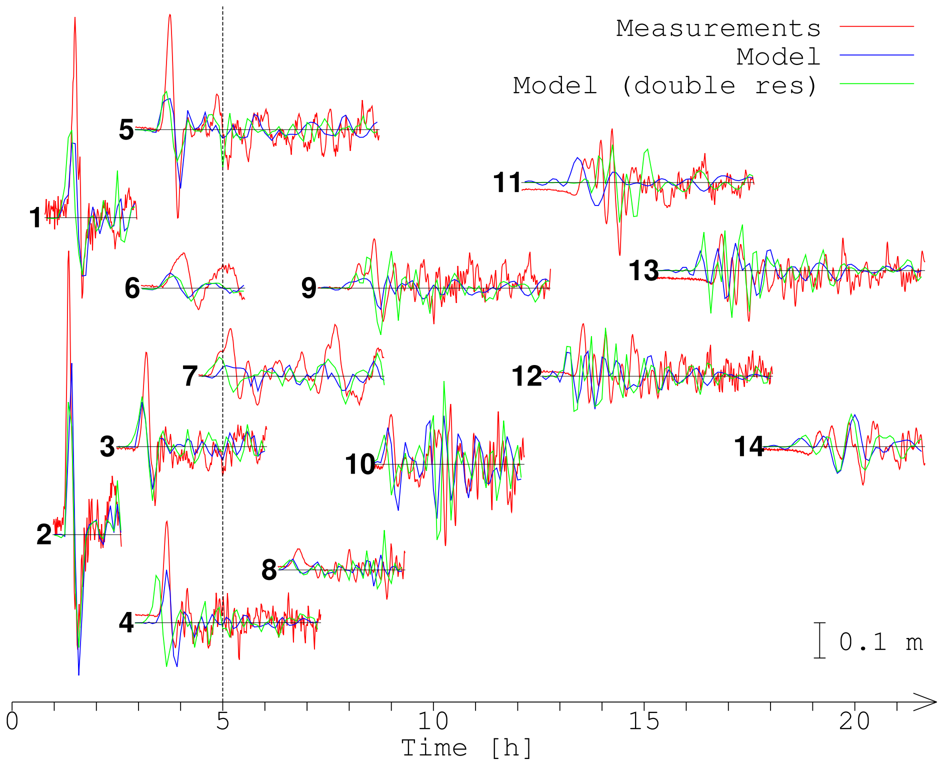

Figure 7: Comparison of the free-surface elevation provided by the model at two different resolutions with DART data. The measurements at the left of the dashed line have been used in the source optimization procedure as the reference to match.

timization. It is seen that the model estimates correctly the time at which the tsunami reaches the different stations, with an error up to 34 minutes at station 14 (Figure 7). The amplitude of the waves is also well predicted. The variations of the elevation at the left of the dashed line is better approximated, which is coherent since those measurements correspond to the reference to match in the source optimization. For farther stations, the model is not always able to reproduce the high frequency oscillations, which is probably due to numerical diffusion.

To verify this assumption, the model, including the source reconstruction, has been run on a refined mesh in which each element of the original mesh has been initially split into four children elements to double the resolution. For the closest measurements used as reference for the optimization, there is no significant improvement (Figure 7). The optimizer set stronger values of the design parameters (hence the initial condition) for the lower resolution run (Table 2), which has the effect of counterbalancing the additional numerical diffusion related to the coarser grid. Hence, close measurements can be accurately resolved with the coarser grid. However, far-field forecasts are more accurate for the high-resolution run which is better to approximate higher frequency oscillations (this is especially visible for stations 9 and 11, which are located offshore). The model error at some stations close to the earthquake is not significantly improved by the increase of resolution (e.g. stations 6 and 7). This is probably due to the fact that this error is not caused by the discretization, but rather by a parameter space that may be too poor to match exactly the actual initial condition. For stations 12, 13 and 14 the shift in the tsunami arrival time is not improved by the increase of resolution. This is likely due to a insufficiently accurate or wrong bathymetry which is the most important factor impacting wave propagation times. This shift may also result from an incorrect assumption about the tsunami source, considered to have no spatial dimension. Each unit source corresponds to a displacement of the sea-surface occuring at time $t = 0$ which may not match the exact moment at which the ground moved. Additional design parameters could be added to specify the time at which each unit source is activated. Apart from numerical errors, additional potential sources of error should be investigated, such as the choice of the mathematical model (e.g. approximations associated with the shallow water equations), and reliability of data (e.g. bathymetry, tsunameter measurement and tidal filtering procedures).

The value of the objective function for the adaptive run is shown in Table 4, for which the initial condition is obtained from the source optimization, but using different numbers of optimization iterations. A comparison with table 1 indicates that the forward runs with and without dynamic adaptation do not share exactly the same point of convergence; which is obvious since different meshes and methods are used. However, they converge towards a similar value, suggesting that using different space discretizations for the source optimization and the forward run is acceptable as soon as their resolution is similar. Table 4 also confirms that no more than 8 iterations are needed, since the objective function is not sensibly improved with more iterations.

Conclusions

The present article describes an application of a real-time tsunami model, using several numerical tools such as the high-order discontinuous Galerkin

Iterations	$J_{\mathrm{adapt}}(\boldsymbol{\alpha}) \cdot 10^{-3}$
0	0.5391000503
1	0.6295562139
2	0.3883790240
3	0.3430059138
4	0.3568695893
5	0.3662980102
6	0.3621919843
7	0.3668633141
8	0.3660865801
9	0.3649512666
10	0.3662966313
11	0.3660944804
12	0.3661192840
13	0.3688849627
14	0.3661253032
16	0.3661250145

Table 4: Values of the objective function for the adaptive forward run $J_{\mathrm{adapt}}(\boldsymbol{\alpha})$ (scaled to be around 1), using the initial condition from the source optimization after different numbers of iterations.

method, *hp*-refinement, parallel dynamic load balancing and adjoint-based data assimilation. While the physical propagation time of the tsunami across the Pacific Ocean is around 22 hours, accurate estimations of the tsunami propagation time and its wave amplitude can be obtained in about 6 hours on 96 processors (5 hours to collect the measurements, 34 minutes of source optimization and 31 minutes of forward run), allowing for far-field early forecasts.

As mentioned in the introduction, the objective of this study is not a presentation of a full-fledged tsunami warning model, but rather a proof of concept showing that several advanced numerical methods can be used efficiently to solve geophysical problems of interest, such as the computation of fast and accurate tsunami warnings. Some improvements should be considered for an application in real life.

Real-time data gathering and filtering is needed to provide measurements of the free-surface usable by the code. This should not be a difficulty, as real-time raw data is available online, and studies have shown that the detiding can be done on the fly [31; 32]. In this work, the data is collected over a period of 5 hours, on which the model is fitted. Shorter (or larger) time intervals may be considered to identify the critical period over which the model results are reliable. The method can be improved by updating the available data during the source optimization. As long as the number of available measurements increases, they would be provided to the model to generate a series of increasingly accurate forecasts.

A wetting and drying algorithm should be added to the model, in order to simulate tsunami-generated inundations. While the implementation may not be straightfoward, mostly because of the need for a differentiable algorithm to allow for the linearization, the model should generate valuable information by taking inundations into account. It should also accelerate the data assimilation process. Collecting the measurements in coastal and dry areas closer to the eathquake epicenter will allow for a faster initial condition reconstruction and shorter range simulations, taking full advantage of the ability of the adjoint model to handle nonlinear effects.

If large computers are available, increasing the resolution with the number of processors can be used to improve the forecast. However, the bathymetry must be precise and the source functions chosen adequately to ensure the accuracy of the results.

Acknowledgements

During the progress of this study, Sébastien Blaise was a Postdoctoral Fellow with the Advanced Study Program of the US National Center for Atmospheric Research (NCAR) and a Postdoctoral Researcher with the Belgian Fund for Research (FNRS). The present study was carried out within the scope of the project "Collaborative Research: A multiscale unified simulation environment for geoscientific applications", which is funded by the National Science Foundation under Peta-Apps grant number 0904599. Support was provided by the Advanced Study Program and the Institute for Mathematics Applied to Geosciences of the National Center for Atmospheric Research (NCAR). Computing resources were provided by NCAR's Computational and Information Systems Laboratory (CISL). The National Center for Atmospheric Research is supported by the National Science Foundation.

References

[1] E. Deleersnijder, V. Legat, P. Lermusiaux, Multi-scale modelling of coastal, shelf and global ocean dynamics, Ocean Dynamics 60 (6) (2010) 1357–1359.

[2] V. Aizinger, C. Dawson, The local discontinuous Galerkin method for three-dimensional shallow water flow, Computer Methods in Applied Mechanics and Engineering 196 (4) (2007) 734–746.

[3] S. Blaise, R. Comblen, V. Legat, J. Remacle, E. Deleersnijder, J. Lambrechts, A discontinuous finite element baroclinic marine model on unstructured prismatic meshes, Ocean Dynamics 60 (6) (2010) 1371–1393.

[4] R. Comblen, S. Blaise, V. Legat, J. Remacle, E. Deleersnijder, J. Lambrechts, A discontinuous finite element baroclinic marine model on unstructured prismatic meshes, Ocean Dynamics 60 (6) (2010) 1395–1414.

[5] S. Blaise, B. de Brye, A. de Brauwere, E. Deleersnijder, E. J. Delhez, R. Comblen, Capturing the residence time boundary layer - application to the Scheldt Estuary, Ocean Dynamics 60 (3) (2010) 535–554.

[6] R. Nair, S. Thomas, R. Loft, A discontinuous Galerkin global shallow water model, Monthly Weather Review 133 (4) (2005) 876–888.

[7] F. Giraldo, M. Restelli, A study of spectral element and discontinuous Galerkin methods for mesoscale atmospheric modeling: equation sets and test cases, Journal of Computational Physics 227 (2007) 3849–3877.

[8] A. St-Cyr, D. Neckels, A fully implicit Jacobian-free high-order discontinuous Galerkin mesoscale flow solver, Computational Science-ICCS (2009) 243–252.

[9] A. St-Cyr, C. Jablonowski, J. Dennis, H. Tufo, S. Thomas, A Comparison of Two Shallow Water Models with Non-Conforming Adaptive Grids: classical tests, Monthly Weather Review 136 (2008) 1898–1922.

[10] C. Burstedde, O. Ghattas, M. Gurnis, T. Isaac, G. Stadler, T. Warburton, L. Wilcox, Extreme-Scale AMR, in: Proceedings of the 2010 ACM/IEEE International Conference for High Performance Computing, Networking, Storage and Analysis, SC '10, IEEE Computer Society, Washington, DC, USA, 2010, pp. 1–12.

[11] C. Chen, F. Xiao, X. Li, An adaptive multimoment global model on a cubed sphere, Monthly Weather Review 139 (2) (2011) 523–548.

[12] E. Kubatko, S. Bunya, C. Dawson, J. Westerink, Dynamic p-adaptive Runge-Kutta discontinuous Galerkin methods for the shallow water equations, Computer Methods in Applied Mechanics and Engineering 198 (21-26) (2009) 1766–1774.

[13] C. Eskilsson, An hp-adaptive discontinuous Galerkin method for shallow water flows, International Journal for Numerical Methods in Fluids 67 (11) (2011) 1605–1623.

[14] S. Blaise, A. St-Cyr, A dynamic *hp*-adaptive discontinuous Galerkin method for shallow water flows on the sphere with application to a global tsunami simulation., Monthly Weather Review 140 (2012) 978–996.

[15] L. Wang, D. Mavriplis, Adjoint-based h-p adaptive discontinuous Galerkin methods for the 2D compressible Euler equations, Journal of Computational Physics 228 (20) (2009) 7643–7661.

[16] S. Gopalakrishnan, D. Bacon, N. Ahmad, Z. Boybeyi, T. Dunn, M. Hall, Y. Jin, P. Lee, D. Mays, R. Madala, et al., An operational multiscale hurricane forecasting system, Monthly Weather Review 130 (7) (2002) 1830–1847.

[17] D. George, R. LeVeque, Finite volume methods and adaptive refinement for global tsunami propagation and local inundation, Science of Tsunami Hazards 24 (2006) 319–328.

[18] S. Nadarajah, A. Jameson, Studies of the continuous and discrete adjoint approaches to viscous automatic aerodynamic shape optimization, AIAA paper 2530 (2001) 1–25.

[19] D. Venditti, D. Darmofal, Grid adaptation for functional outputs: application to two-dimensional inviscid flows, Journal of Computational Physics 176 (1) (2002) 40–69.

[20] R. Balasubramanian, J. Newman, Adjoint-based error estimation and grid adaptation for functional outputs: Application to two-dimensional, inviscid, incompressible flows, Computers & Fluids 38 (2) (2009) 320–332.

[21] H. Bücker, J. Willkomm, S. Gross, O. Fortmeier, Discrete and continuous adjoint approaches to estimate boundary heat fluxes in falling films, Optimization Methods & Software 26 (1) (2011) 105–125.

[22] F. Rauser, J. Riehme, K. Leppkes, P. Korn, U. Naumann, On the use of discrete adjoints in goal error estimation for shallow water equations, Procedia Computer Science 1 (1) (2010) 107–115.

[23] R. Long, W. Thacker, Data assimilation into a numerical equatorial ocean model. I. The model and the assimilation algorithm, Dynamics of atmospheres and oceans 13 (3-4) (1989) 379–412.

[24] G. Carmichael, D. Daescu, A. Sandu, T. Chai, Computational aspects of chemical data assimilation into atmospheric models, Computational Science-ICCS (2003) 722–722.

[25] V. Titov, F. González, E. Bernard, M. Eble, H. Mofjeld, J. Newman, A. Venturato, Real-Time Tsunami Forecasting: Challenges and Solutions, in: E. Bernard (Ed.), Developing Tsunami-Resilient Communities, Springer Netherlands, 2005, pp. 41–58.

[26] H. Kanamori, J. Given, Use of long-period seismic waves for rapid evaluation of tsunami potential of large earthquakes, Tsunamis - Their Science and Engineering (1983) 37–49.

[27] V. V. Titov, F. I. Gonzales, Implementation and testing of the method of splitting tsunami (MOST) model, Tech. rep., NOAA Technical Memorandum ERL PMEL-112 (November 1997).

[28] M. B. Giles, N. A. Pierce, An Introduction to the Adjoint Approach to Design, Flow, Turbulence and Combustion 65 (2000) 393–415.

[29] L. Graña Drummond, B. Svaiter, A steepest descent method for vector optimization, Journal of computational and applied mathematics 175 (2) (2005) 395–414.

[30] R. H. Byrd, P. Lu, J. Nocedal, A Limited Memory Algorithm for Bound Constrained Optimization, SIAM Journal on Scientific and Statistical Computing 16 (1995) 1190–1208.

[31] M. Foreman, W. Crawford, R. Marsden, Quantitative Skill Assessment for Coastal Ocean Models, Vol. 47 of Coastal and Estuarine Studies, American Geophysical Union, Washington, DC, 1995, Ch. De-Tiding: Theory and Practice, pp. 203–239.

[32] E. Tolkova, Principal component analysis of tsunami buoy record: Tide prediction and removal, Dynamics of Atmospheres and Oceans 46 (1-4) (2009) 62–82.

[33] J. Côté, A Lagrange multiplier approach for the metric terms of semi-Lagrangian models on the sphere, Quarterly Journal of the Royal Meteorological Society 114 (1988) 1347–1352.

[34] D. L. Williamson, J. B. Drake, J. J. Hack, R. Jakob, P. N. Swarztrauber, A standard test set for numerical approximations to the shallow water equations in spherical geometry, Journal of Computational Physics 102 (1) (1992) 211–224.

[35] M. Alexe, A. Sandu, Forward and adjoint sensitivity analysis with continuous explicit Runge-Kutta schemes, Applied Mathematics and Computation 208 (2) (2009) 328–346.

[36] C. Amante, B. W. Eakins, ETOPO1 1 Arc-Minute Global Relief Model: Procedures, Data Sources and Analysis, Tech. rep., NOAA Technical Memorandum NESDIS NGDC-24 (March 2009).

[37] C. Geuzaine, J.-F. Remacle, Gmsh: a three-dimensional finite element mesh generator with built-in pre- and post-processing facilities, International Journal for Numerical Methods in Engineering 11 (2009) 1309–1331.

[38] M. Alexe, A. Sandu, Space-time adaptive solution of inverse problems with the discrete adjoint method, Tech. Rep. TR-10-14, Virginia Tech. (2010).

[39] E. Gica, M. C. Spillane, V. V. Titov, C. D. Chamberlin, J. C. Newman, Development of the forecast propagation database for NOOA's short-term inundation forecast for tsunamis (SIFT), Tech. Rep. OAR PMEL-139, NOAA, Seattle, WA (March 2008).

[40] C. Zhu, R. H. Byrd, J. Nocedal, L-BFGS-B: Algorithm 778: L-BFGS-B, FORTRAN routines for large scale bound constrained optimization, ACM Transactions on Mathematical Software 23 (4) (1997) 550–560.

[41] D. P. Bertsekas, Projected Newton methods for optimization problems with simple constraints, SIAM Journal on Control and Optimization 20 (2) (1981) 221–246.