High-Frequency Nonlinear Earthquake Simulations on Petascale Heterogeneous Supercomputers

Daniel Roten[1], Yifeng Cui[2], Kim B. Olsen[1], Steven M. Day[1], Kyle Withers[1,2], William H. Savran[1,2], Peng Wang[3] and Dawei Mu[2]
[1]San Diego State University, [2]University of California, San Diego, [3]NVIDIA, Inc.
[1](droten,kbolsen,sday@mail.sdsu.edu), [2](yfcui,kbwithers,wsavran,damu@ucsd.edu), [3](penwang@nvidia.com)

***Abstract*—The omission of nonlinear effects in large-scale 3D ground motion estimation, which are particularly challenging due to memory and scalability issues, can result in costly misguidance for structural design in earthquake-prone regions. We have implemented nonlinearity using a Drucker-Prager yield condition in AWP-ODC and further optimized the CUDA kernels to more efficiently utilize the GPU's memory bandwidth. The application has resulted in a significant increase in the model region and accuracy for state-of-the-art earthquake simulations in a realistic earth structure, which are now able to resolve the wavefield at frequencies relevant for the most vulnerable buildings ($>$ 1 Hz) while maintaining the scalability and efficiency of the method. We successfully run the code on 4,200 Kepler K20X GPUs on NCSA Blue Waters and OLCF Titan to simulate a M 7.7 earthquake on the southern San Andreas fault with a spatial resolution of 25 m for frequencies up to 4 Hz.**

***Keywords*—SCEC, earthquake ground motion, fault zone plasticity, nonlinear soil behavior, GPU**

I. Introduction

The growth of urban environments in seismically active regions has resulted in increasing economic exposure to earthquake hazards, a tendency that is reflected in the list of recent devastating earthquakes. Measures to improve the resilience of society towards earthquakes include a better understanding and prediction of effects that control the variability and severity of shaking, such as source directivity effects [1], wave guide effects [2] or amplification by soft sedimentary deposits [3]. In regions where detailed representations of the geologic structure and seismic fault lines are available, these effects can be predicted using three-dimensional numerical simulations of the dynamic rupture process and the resulting wave propagation [4].

For example, repeated simulation efforts have been carried out to quantify the ground motions that must be expected during a future large earthquake rupturing the southern segment of the San Andreas fault (SAF) [2, 5–7]. Early terascale simulations (Terashake) [2], performed in 2005 with the AWP-ODC finite difference code, predicted strong long-period ground motions in the Los Angeles basin if the southern SAF were to rupture from southeast to northwest. These amplifications were attributed to coupling between source directivity and amplification by the soft sediments inside a string of sedimentary basin, which form an effective waveguide that channels surface waves from the SAF into the densely populated downtown Los Angeles region.

In 2008, simulations conducted at the Southern California Earthquake Center (SCEC) involving a similar M 7.8 earthquake scenario were adopted by the USGS as the basis for the first Great Southern California ShakeOut - an earthquake emergency response and preparedness exercise that is now repeated yearly in California and a growing number of other regions [8]. Two years later, a M8 8 "wall-to-wall" event on the southern SAF was successfully simulated using 223,074 CPU cores on the OLCF (Oak Ridge Leadership Computational Facility) system *Jaguar*, in a SCEC milestone calculation marking one of the largest and most detailed earthquake simulations performed to date.

Improvements in the resolution of these earthquake simulations have been closely coupled to the development of leadership-class computational facilities. For example, the introduction of the highly optimized GPU-based version of the AWP-ODC code [9, 10] has allowed seismologists to benefit from the heterogeneous architecture in the current generation of supercomputers. This GPU code played a critical role in the CyberShake project, which generated the first physics-based seismic hazard maps of the Los Angeles region [10, 11]. These maps have identified several sites where the combination of rupture directivity and basin response leads to an elevated hazard level (with respect to conventional seismic hazard maps), because they explicitly incorporated deterministic source and wave propagation effects in the seismic hazard calculations through the use of 3D ground motion simulations.

The seismic hazard maps produced by CyberShake, as well as previous TeraShake or ShakeOut simulations, were limited to frequencies below 1 Hz, and thus only relevant for structures vulnerable to long-period ground motions - such as highrise buildings, long-span bridges or oil tanks. However, the deployment of GPU-based supercomputers has also resulted in a significant improvement in in the scale and detail of wave propagation simulations, which are now able to resolve the wavefield for large ($M > 7$) earthquakes at frequencies of up to 10 Hz [10, 12, 13]. This development offers the prospect to predict ground motions from scenario simulations at the frequencies relevant for more common structures, including moderately tall buildings or single-story homes, and ultimately produce physics-based seismic hazard maps for the entire

SC16; Salt Lake City, Utah, USA; November 2016
978-1-4673-8815-3/16/$31.00

frequency range of engineering interest (0 – 10 Hz).

However, the physics of wave propagation at these high frequencies imposes several new computational challenges to deterministic ground motion prediction. It is now increasingly accepted that anelastic attenuation becomes frequency-dependent at frequencies above ∼1 Hz [14], which necessitates a departure from the traditional assumption of frequency-independent quality factors Q used in previous low-frequency simulations [15]. Additionally, small-scale heterogeneities must be included in the velocity mesh to properly account for scattering of seismic waves at higher frequencies. Simulations [13, 16] also suggest that fault roughness (i.e., the departure of natural fault surfaces from planarity) is required for realistic high-frequency ground motion.

Nonlinear response in rocks and soils, which tends to become especially important at higher frequencies, represents another difficulty. Numerical studies show that the high stresses near the rupture front exceed the strength of crustal rock, requiring a nonlinear treatment of the deformation [17–21]. Soft soils near the surface may also exhibit nonlinear response during strong shaking, especially at higher frequencies [3]. This nonlinear soil behavior is typically treated using 1-D nonlinear or equivalent nonlinear simulations in site-specific seismic hazard assessment. Observations made on vertical arrays (i.e., strong-motion stations equipped with a borehole accelerometers) in recent years tend to confirm the importance of this phenomenon [22–25].

Theoretical studies [26–28] also predict that the large strains induced by long-period surface waves emitted from the SAF may give rise to nonlinear behavior of the shallow sedimentary rock underlying the basins. Indeed recent simulations of the ShakeOut scenario for an elasto-plastic medium predict long-period ground motions that are 30–70% lower compared to viscoelastic solutions in the LAB [29].

In order to predict such nonlinear effects in high-frequency earthquake simulations, we have implemented nonlinearity based on the Drucker-Prager yield condition in the AWP-ODC FD code. Section II of this this paper provides a brief summary of the linear AWP-ODC algorithm. In section III we describe technical details of the plasticity implementation, discuss optimizations that preserve the scalability and efficiency of the code, and show weak scaling results. Section IV shows how this code is used to simulate a M 7.7 earthquake on the southern segment of SAF for frequencies up to 4 Hz with and without nonlinear material response.

II. Linear AWP-ODC Algorithm and Implementation

AWP-ODC was originally developed as a personal research code by Kim Olsen at the University of Utah [30]. The code solves a coupled system of partial differential equations,

$$\begin{aligned} \partial_t \mathbf{v} &= \frac{1}{\rho}\nabla\cdot\sigma \\ \partial_t \sigma &= \lambda(\nabla\cdot\mathbf{v})I + \mu(\nabla\mathbf{v} + \nabla\mathbf{v}^T), \end{aligned} \tag{1}$$

where λ and μ are the Lamé coefficients, ρ is the density and $\mathbf{v}$ and σ are the particle velocity and symmetric stress tensor, respectively. Decomposing eq. 1 component-wise leads to three scalar-valued equations for the velocity stress component and six scalar-valued equations for the stress component.

A. Staggered-grid Finite Difference Equations

These nine scalar equations are approximated using an explicit finite differences scheme on a grid staggered in both time and space. Time derivatives are approximated by the following central difference equations:

$$\begin{aligned} \partial_t \mathbf{v}(t) &\approx \frac{\mathbf{v}(t+\frac{\Delta t}{2}) - \mathbf{v}(t-\frac{\Delta t}{2})}{\Delta t} \\ \partial_t \sigma\Big(t+\frac{\Delta t}{2}\Big) &\approx \frac{\sigma(t+\Delta t) - \sigma(t)}{\Delta t}. \end{aligned} \tag{2}$$

For the spatial derivatives, let Φ denote a generic velocity or stress component, and Δh the equidistant mesh size. The FD approximation to $\partial_x \Phi$ at grid point (i, j, k) is

$$\partial_x \Phi_{i,j,k} \approx D_x^4(\Phi)_{i,j,k} = \frac{c_1\big(\Phi_{i+\frac{1}{2},j,k} - \Phi_{i-\frac{1}{2},j,k}\big) + c_2\big(\Phi_{i+\frac{3}{2},j,k} - \Phi_{i-\frac{3}{2},j,k}\big)}{\Delta h}, \tag{3}$$

where $c_1 = \frac{9}{8}$ and $c_2 = -\frac{1}{24}$. This equation is used to approximate each spatial derivative for each velocity and stress component. Thus, the approximation is 2^{nd} order accurate in time and 4^{th} order accurate in space.

B. Boundary Conditions

In the CPU version of AWP-ODC (thereafter called AWP-ODC CPU), three kind of absorbing boundary conditions are implemented: sponge layers [31], perfectly matched layers (PML) [32] and multi-axial PMLs, called M-PMLs [33]. The GPU version of the code (AWP-ODC GPU) only supports sponge layers. At the free surface zero stress boundary conditions are applied [34].

C. Frequency-dependent Attenuation

The attenuation of seismic waves due to energy-loss by a variety of mechanisms is modeled by the seismic quality factor, Q. This parameter can have a significant impact on the ground motion even at several wavelengths from the source. As simulations extend to higher frequencies, energy losses from anelasticity become progressively more important as there are more wavelengths of propagation within a modeled domain. At low frequencies (<1 Hz), Q has typically been modeled as a constant, independent of frequency [e.g., 15]. Observations have shown that Q increases above ∼1 Hz, and thus anelastic attenuation drops off at higher frequencies. AWP uses a computationally efficient memory-variable approach to a frequency-dependent Q model that is constant below, and a power-law above, a chosen transition frequency in the form

$$Q(f) = Q_0 \cdot \left(\frac{f}{f_0}\right)^{\gamma}, \tag{4}$$

where f_0 is a reference frequency with Q_0 and γ a constant that varies with the region and geology [14]. The implementation is an extension of the coarse-grained approach [15, 35]

that fits arbitrary Q models by solving for the coefficients that scale the weights at each relaxation time across a bandwidth of over 2 decades. The $Q(f)$ model has larger ground motion at frequencies greater than 1 Hz, better matching the decay in spectral acceleration and cumulative energy as a function of distance from the fault, with the effect becoming more important as distance from the source increases. For example, the constant-Q model predicts amplitudes that are deficient by a factor of 3 to 5, relative to both the data and the $\gamma = 0.8$ power-law model of the M_w 5.4 Chino Hills earthquake, at distances up to 60 km and up to 4 Hz [14].

D. Seismic Source

The seismic source in physics-based earthquake simulations can be specified using a kinematic or dynamic representation. In the kinematic approach the spatial and temporal evolution of slip is prescribed *a priori* on each grid point on the fault (subfault). In a dynamic source, only the friction properties and initial stresses are prescribed on the fault, and the time history of fault slip evolves spontaneously [36] as part of the elasto-dynamic solution. At present dynamic mode (spontaneous rupture) is only implemented in the CPU version of AWP-ODC. Both the CPU and GPU versions are able to run in wave propagation mode, i.e. using a kinematic source.

Simulation of dynamic rupture follows the staggered-grid split-node (SGSN) method [37]. This approach and its implementation in AWP-ODC CPU has been verified against other finite element and finite difference codes in the framework of the SCEC/USGS dynamic rupture code verification project [36, 38, 39].

E. Multi-CPU AWP Implementation

AWP-ODC CPU uses data parallelism incorporating a 3D domain decomposition [40] and communicates through the Message Passing Interface (MPI). Each MPI process is responsible for performing stress and velocity calculations within its own subdomain of the simulation grid. Ghost cells, which manage the most recently updated wavefield parameters exchanged with adjacent subgrids, occupy a two-cell padding layer in the CPU code.

The code was heavily modified and optimized for massive scalability in preparation of large-scale milestone simulations, such as the M8 run which sustained 220 Tflop/s for 24 hours on OLCF *Jaguar* [7]. Examples of single-CPU optimizations include cache blocking techniques or the use of reciprocal forms of Lamé parameters. Multi-CPU optimizations involved asynchronous MPI communication calls to overlap computation with communication and I/O optimizations (described below). A complete description of AWP-ODC CPU is beyond the scope of this paper, and the reader is referred to the explanations given in the SC10 article by Cui *et al.* [7].

F. Multi-GPU AWP Implementation

The multi-GPU implementation of AWP-ODC was designed by Zhou *et al.* [41] on the multi-CPU version with maximum throughput for heterogeneous computing in mind. AWP-ODC GPU performs a two-layer 3D domain decomposition: First the computational domain is decomposed along the two horizontal directions, with each subdomain mapped to a single CPU core which controls an associated GPU. The sub-domain inside each GPU is then further sub-partitioned for streaming multi-processors (SM) in the Y and Z directions. Two computation kernels, one for velocity and one for stress, are carried out in sequence on the GPU.

To reduce the number of communications between different GPUs, the ghost cell regions were increased from 2 layers (in the CPU version) to 4 layers in the GPU code [9]. This strategy eliminates the need to exchange the stresses (represented by six independent tensor elements) between neighboring subdomains, and only velocities (consisting of three vector components) are exchanged between GPUs. An efficient computing / communication schedule was designed to hide the communication latency. Velocities are first computed along the domain boundaries and copied to the CPU. From there, they are sent to other nodes using non-blocking `MPI_Isend` calls, while the GPU proceeds computing velocities in the center of the domain. This order is reversed for the computation of stresses, such that velocities in the ghost cell regions have been received by the time stresses are computed along the domain boundaries [9]. This overlap between computation and communication results in an excellent scalability of the code.

G. Input and Output

Both the CPU and GPU version of AWP-ODC use MPI-IO to write velocity output concurrently to a single file. Efficient MPI-IO performance is obtained by defining indexed data types at the initialization stage and setting logical file views for each sub-process participating in the output [7]. A parallel approach is also employed for reading in the mesh file, which often reaches a size of several Terabytes. To minimize fragmentation and scattering when dealing with hundreds of thousands of processors, the mesh is only read by a portion of the processors, which employ MPI-IO to read highly contiguous big chunks of the mesh data and then redistribute it to the destination cores using point-to-point communication [7].

The subroutines involved in the input of the mesh and the output of velocities are now part of a dedicated modular library, called SEISM-IO library [42]. This library is available to developers of wave propagation and dynamic rupture applications that are based on structured grids, with the aim to minimize duplicate investments in the handling of IO for each application. SEISM-IO implements optimized high-performance I/O operations in a software layer between the high-level application and the file system, and provides both a C and Fortran interface to structured mesh applications. It aims to simplify the programming of parallel I/O for code developers by hiding some of the more complicated aspects from the user (i.e., dealing with output on a partitioned grid or buffering of outputs for improved performance) [43]. The library also supports high-level file choices, such as HDF5 [44], ADIOS [45], and parallel netCDF [46].

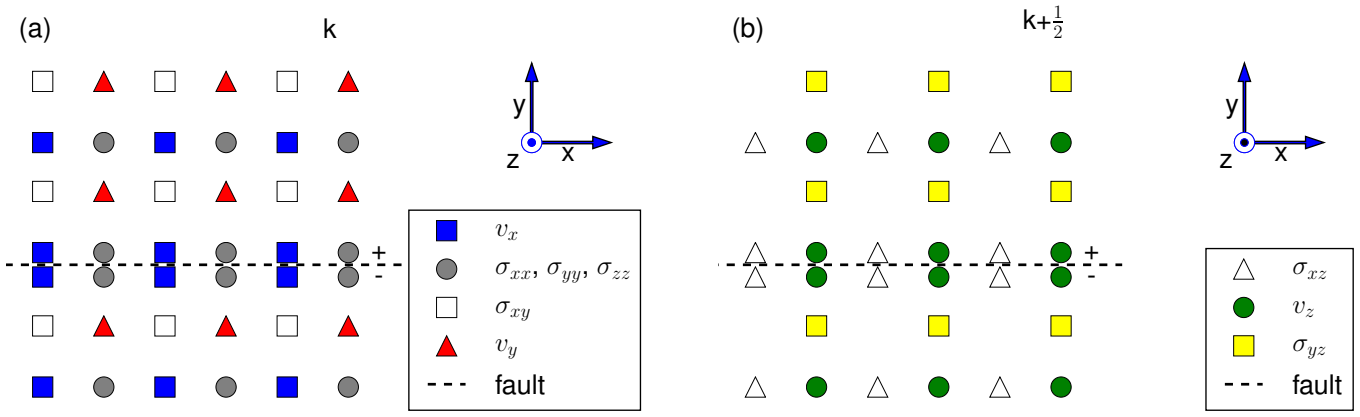

Figure 1. staggered-grid split-node (sgsn) [37] layout around the fault in a cross-section through the xy-plane at a depth of *(a)* k and *(b)* $k+\frac{1}{2}$. (note that σ_{yy} is not discontinuous across the fault plane.)

III. Implementation of Plasticity in AWP-ODC

In the plastic version of AWP-ODC, an additional kernel function is called after the stresses are updated from velocities. This function checks if the yield stress has been exceeded, and reduces the stress deviator if this is the case.

A. Drucker-Prager Equations

The implementation to Drucker-Prager (DP) plasticity follows return map algorithm [47, 48], in accordance with guidelines specified in the SCEC/USGS code verification benchmark TPV27 [39]. The material is described using non-associative Drucker-Prager plasticity with yielding in shear. The DP yield stress $Y(\sigma)$ is defined as

$$Y(\sigma) = \max\left(0,\ c\ \cos\varphi - (\sigma_m + P_f)\sin\varphi\right), \tag{5}$$

where c is the cohesion, φ the friction angle, P_f the fluid pressure and σ_m the mean stress:

$$\sigma_m = \frac{1}{3}\left(\sigma_{xx} + \sigma_{yy} + \sigma_{zz}\right) = \frac{I_1}{3} \tag{6}$$

The Drucker-Prager yield function $F(\sigma)$ is defined as

$$F(\sigma) = \sqrt{J_2(\sigma)} - Y(\sigma). \tag{7}$$

J_2 is the second invariant of the stress deviator,

$$s_{ij} = \sigma_{ij} - \sigma_m \delta_{ij} \tag{8}$$

(with Kronecker delta δ) and defined as

$$J_2 = \frac{1}{2}\sum_{i,j} s_{ij} s_{ji}. \tag{9}$$

In the return map algorithm, the stress tensor is updated using the same kernel as in an elastic or viscoelastic simulation, and the Drucker-Prager yield function (7) is computed from this trial stress. If $F(\sigma^{\text{trial}}) < 0$, the trial stress is taken as stress before the iteration continues. If $F(\sigma^{\text{trial}}) \geq 0$, the yield stress has been exceeded. In this case, the factor r is computed as

$$r = \frac{Y(\sigma^{\text{trial}})}{\sqrt{J_2(\sigma^{\text{trial}})}} + \left(1 - \frac{Y(\sigma^{\text{trial}})}{\sqrt{J_2(\sigma^{\text{trial}})}}\right) e^{\frac{-\Delta t}{t_v}}, \tag{10}$$

where Δt is the temporal discretization in the FD simulation and t_v is the viscoplastic relaxation time. In a pure elasto-plastic simulation ($t_v = 0$), the term to the right of the plus sign in (10) vanishes. In a viscoplastic simulation, the material responds elastically to a sudden increase in stress, and the stress tensor decays exponentially with characteristic time t_v. Two-dimensional dynamic rupture simulations with a finite difference method [17] show that time-dependent relaxation stabilizes the solution if t_v is set to at least the time for an S wave to propagate one grid spacing.

In a viscoelastic time step the trial stress deviator is multiplied with the yield factor r from (10) and used to compute the adjusted stress:

$$\sigma_{ij} = \sigma_m^{\text{trial}} \delta_{ij} + r s_{ij}^{\text{trial}}. \tag{11}$$

This results in the material yielding in shear.

B. Parameters for Nonlinear Simulations

Linear wave propagation simulations only compute the stress change from an initial state, where the initial stresses and forces are assumed equal to zero. In a nonlinear simulations, however, the total value of the stress tensor needs to be known to compute the DP yield stress (5). Simulations with the total value of the stress tensor require an initial stress state that is in static equilibrium, which may be obtained from a static calculation [e.g. 49], as well as explicit gravity and boundary tractions [39].

A simpler method, which does not require a static simulation, consists in adding the initial stress σ^o to the stress change just before the DP yield stress is computed (5), and subtracting it again after the stresses have been adjusted (11). We use the latter approach in our nonlinear simulations, which requires an extra 3-D array containing the six elements of the initial stress tensor. Additional 3-D arrays are also needed to store the fluid pressure P_f and the nonlinear material parameters φ and c. In plastic simulations we also compute and store the quantity η, which represents the accumulated inelastic strain due to yielding [18].

C. Implementation of DP-Plasticity on a Staggered FD Grid

The staggered finite difference grid, introduced in 1976 by Madariaga [50], has become a preferred approach in seismic wave propagation problems. For the implementation of plasticity, however, the staggered grid is not optimal, because all the six independent components of the stress tensor must be known to calculate equation 7 at any location where a stress component is defined (Fig. 1). For example, to obtain the shear stresses σ_{xz} at position $(i+\frac{1}{2}, j, k)$, where the normal stresses are defined, the following interpolation has to be performed:

$$\sigma_{xz}^{(i+\frac{1}{2},j,k)} = \frac{1}{4}\Big(\sigma_{xz}^{(i,j,k-\frac{1}{2})} + \sigma_{xz}^{(i,j,k+\frac{1}{2})} + \sigma_{xz}^{(i+1,j,k-\frac{1}{2})} + \sigma_{xz}^{(i+1,j,k+\frac{1}{2})}\Big). \tag{12}$$

Similar expressions are required for σ_{xy} and σ_{yz}. Conversely, to compute the DP yield function at the location of any shear stress component, the other two shear stresses need to be interpolated as well as the three normal stresses. Interpolations are also required for nonlinear material parameters and the six components of the initial stress, which are defined at the same position as other constants, at $(i, j+\frac{1}{2}, k+\frac{1}{2})$. All these interpolations add up to significant computational cost. We compared two different ways of implementing plasticity to optimize efficiency.

1) Individual Interpolation (EP1): In the first implementation, the yield condition is evaluated at every location where shear and normal stresses are defined. Because equation (7) must be computed from the unmodified trial stress for all components, the arrays containing trial stresses are first copied into separate arrays before stresses can be modified. The procedure is as follows:

(a) Exchange stresses in ghost cell regions between processes.
(b) Copy trial stresses into new stress arrays.
(c) Evaluate DP equations (5 – 11) for σ_{xx}, σ_{yy} and σ_{zz}, interpolating σ_{xz}, σ_{yz}, σ_{xy}, the material properties c and φ, fluid pressure P_f and all six components of σ^o.
(d) Evaluate DP equations for σ_{xz}, interpolating σ_{xx}, σ_{yy}, σ_{zz}, σ_{yz}, σ_{xy} c, φ, P_f and σ^o.
(e) Evaluate DP equations for σ_{yz}, interpolating σ_{xx}, σ_{yy}, σ_{zz}, σ_{xz}, σ_{xy}, c, φ, P_f and σ^o.
(f) Evaluate DP equations for σ_{xy}, interpolating σ_{xx}, σ_{yy}, σ_{zz}, σ_{xz}, σ_{yz}, c, φ, P_f and σ^o.
(g) Exchange stresses in ghost cell regions between processes.

With this implementation of DP-plasticity, a nonlinear simulation takes almost 4 times as long as a linear simulation (Table I).

2) Yield Factor Interpolation (EP2): In the second implementation, we take advantage of the fact that the mean stress (6) is only required at the locations where the normal stresses are defined. The DP yield function is only evaluated at the location of the normal stresses, where we define the yield factor r (10). If the DP yield stress has not been exceeded, r is set to one. At the location where shear stress components are defined, only the yield factor r (10) is interpolated, and shear stresses are reduced using (10) if the interpolated value of r is less than one. EP2 involves the following procedure:

(a) Exchange stresses in ghost cell regions between processes
(b) Compute r at location of normal stresses (5 – 10), interpolating σ_{xz}, σ_{yz}, σ_{xy}, the material properties c and φ, the fluid pressure P_f and the six components of the initial stress tensor, σ^o.
(c) Exchange yield factor r in ghost cell regions between processes.
(d) Adjust normal stresses using r (11).
(e) Adjust σ_{xz}, interpolating r and σ_{xz}^o if $r < 1$.
(f) Adjust σ_{yz}, interpolating r and σ_{yz}^o if $r < 1$.
(g) Adjust σ_{xy}, interpolating r and σ_{xy}^o if $r < 1$.
(h) Exchange stresses in ghost cell regions between processes.

This approach eliminates the need to create copies of the stress tensor before the DP routines are evaluated; only one extra 3-D variable is needed for r. Because the number of interpolations is greatly reduced with respect to EP1, implementation EP2 is much faster (Table I). The solution for benchmark TPV27 [39], which includes plasticity, requires only about 65% more CPU time than the linear benchmark TPV26. Although formulation EP1 is not identical to EP2, both methods yield practically identical ground motion results in wave propagation simulations, as we show in the verification paragraph below.

Table I
COMPUTATIONAL COST FOR TPV26/27 [39] IN A LINEAR SIMULATION AND USING TWO DIFFERENT IMPLEMENTATIONS OF DRUCKER-PRAGER ELASTOPLASTICITY.

Method	CPU Time per iteration (s)	Normalized
Elastic	0.176	100%
EP1	0.676	384%
EP2	0.290	165%

D. Implementation of Plasticity in SGSN Dynamic Rupture Mode

In dynamic rupture simulations, the vertical, planar fault is intersecting normal stresses, and split nodes are defined for σ_{xx}, σ_{zz}, v_x, v_z and σ_{xz} on the opposing sides of the fault [37] (Fig. 1). In the SGSN method, the accuracy of FD equations is reduced to 2^{nd}-order for grid points less than two spatial increments from the fault plane. Spatial derivatives of velocity and shear stress components on the fault plane are computed by one-sided FD operators conforming to the traction continuity conditions.

As the cost of computing DP plasticity within the fault zone is relatively small, we always use implementation EP1 within two grid points of the fault. The DP equations (5 - 11) are computed separately on both sides of the fault. To compute the DP yield function (7) at the location of normal stresses on split nodes, the values of σ_{yz} and σ_{xy} are interpolated from the shear traction T_x and T_z on the fault, defined at the grid positions of the velocities v_x and v_z (Fig. 1). No stress tensor components are interpolated across the fault.

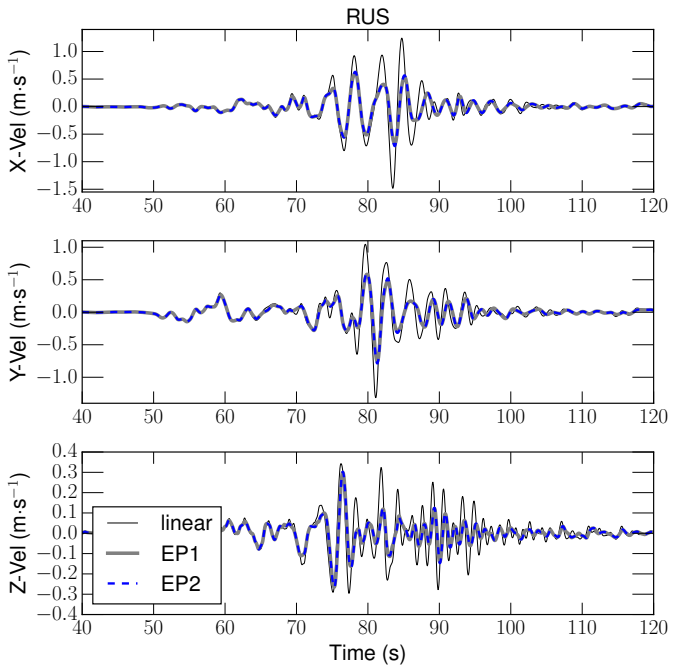

Figure 2. Seismograms at station **rus** in the Whittier narrows corridor (Fig. 5) , obtained from linear simulation and nonlinear simulations using implementations EP1 and EP2.

E. Verification of DP Implementation

We verified our implementation of DP plasticity in AWP-ODC by participating in benchmarks TPV26 and TPV27 of the SCEC/USGS dynamic rupture code verification project [36, 38, 39]. Time series obtained by AWP-ODC for a linear (TPV26) and nonlinear (TPV27) medium compare favorably with solutions from other spontaneous rupture simulation methods [39].

Figure 2 shows a comparison between plasticity implementations EP1 and EP2 for the realistic case of a San Andreas earthquake, where significant nonlinearity occurs in soft soils at larger distance from the fault. This test was done with a coarser version (Δh = 100 m) of the mesh used in the 4 Hz simulations described later. Time series at station **rus** are consistent between both plasticity implementations. Horizontal peak ground velocities are reduced from ∼1.5 m/s in the linear case to less than 0.75 m/s in the nonlinear case, as already shown in previous studies [29, 51].

F. Communication in Plastic Calculations

In the CPU version of the code, stress updates performed by the plasticity kernels require an additional communication of stresses in ghost cell regions between processes. In contrast to linear AWP-ODC, where only the stress components required for velocity computation are exchanged [7], all six stress components in the two ghost cell layers must be swapped in the nonlinear code to compute the DP yield stress. Implementation EP2 also requires exchange of the yield factor r.

In the GPU version of AWP-ODC we avoided introducing additional communication routines (for stresses and yield factors) that could adversely affect the scalability of the code. Instead, we extended the ghost cell region by an additional four layers on each side of the subdomain to a total of 8 layers. The additional computational time required by the plasticity kernels leaves more room to hide the communication of data in these increased ghost cell regions behind the computation.

Table II
COMPARISON OF GPU TIME AND MEMORY REQUIREMENTS FOR DIFFERENT MEMORY OPTIMIZATION STRATEGIES.

Version	Time per iteration	Normalized	CUDA Memory	Normalized
linear	141.5 ms	100.0%	3.04 Gb	100.0%
11 var	218.6 ms	154.5%	4.51 Gb	148.4%
5 var	219.5 ms	155.1%	3.71 Gb	122.0%
3 var	268.0 ms	189.4%	3.45 Gb	113.5%

G. Memory Optimization in AWP-ODC GPU

Plastic simulations need 11 new 3-D arrays in addition to the 21 arrays required for linear simulations [7]: 6 for the initial stress, one for the fluid pressure, two for φ and c, one for the yield factor r, and one for the accumulated plastic strain η. These extra 3-D arrays are exhausting the limited memory available on the accelerators (6 Gb on K20X GPUs). Although the total available memory may be increased by requesting a larger amount of GPUs, the lack of domain partitioning in the vertical direction may lead to a subdomain geometry that is stretched in the vertical direction and narrowed along the horizontal directions. As a consequence the volume of ghost cells becomes relatively large compared to the volume of the whole subdomain, decreasing the efficiency of the code.

We reduce the need for additional memory on the GPU during plastic simulations by re-computing some of the extra variables during each iteration in the plasticity kernel. Because the initial stress tensor is computed from the vertical stress in our San Andreas simulations, we only store the vertical initial stress, σ^o_{zz}, in a 3-D array, and recompute the remaining 5 components at each grid point during plasticity stress updates. The fluid pressure, which depends only on the depth if the ground water table is assumed to be constant throughout the computational domain, needs to be computed just once per CUDA thread. This approach reduces the number of extra 3-D variables to 5. Table II compares the requirements of GPU time and memory for a linear simulation and nonlinear simulations with different memory optimization strategies.

Reducing the number of 3-D variables from 11 to 5 does not significantly increase the time per iteration, which is about 55% longer than a linear simulation, but reduces the need for extra memory from 48% to 22%. Because the variables c and φ are calculated from initial stresses and material properties based on computationally expensive empirical equations (described in section IV-B), re-computing them during every call to the plasticity kernel significantly increases the time per iteration (version '3-var' in Table II). For our San Andreas scenario simulations, we used the version requiring 5 extra variables.

The cost of plasticity, both in terms of computational time and memory, could potentially be further reduced by only

performing plastic updates in regions where nonlinearity is expected (e.g., close to the surface and in the vicinity of the fault) without compromising physics. In the simulations shown in this paper, the DP yield condition is evaluated throughout the computational domain.

H. Improvements of Memory Throughput in Plasticity and Stress Kernels

As a typical stencil code, the finite difference kernels are memory bound. We optimized the efficiency of the plasticity and stress kernels by ensuring that they have a sufficient number of active threads (i.e., improving *occupancy*). Performance in this respect was quantified by measuring the achieved DRAM throughput (defined as the actual bytes accessed divided by kernel time) with the profiler. We also aimed to increase the access fraction, which gives the percentage of non-redundant access in the total DRAM traffic. In case of perfect caching, i.e. if each array is accessed from global memory only once, the access fraction will equal 1.

Occupancy is increased by reducing the resources used by the threads (mainly shared memory and registers). As a trade-off in increasing occupancy, more register spilling could potentially occur. We used `__launch_bounds__` to control the occupancy and tuned the launch bound parameters to find the best trade-off between occupancy and register usage. Redundant halo access was reduced by using texture cache combined with register queues to cache read-only earth model parameters, while shared memory was used to cache velocity variables that need both reading and writing.

After these optimizations, the performance of the plasticity kernel improved by ∼45% and the performance of the stress kernel improved by 20%. The plasticity kernel reached ∼75% in DRAM throughput and an access fraction of almost 100%, meaning there is very little room for further optimization of this kernel in both aspects. In the stress kernel, which accesses more arrays than the plasticity kernel, cache miss increases due to contention in texture arrays and L2 cache, through which halo data are loaded from global memory. The DRAM throughput of the stress kernel achieved 62% of peak throughput and the access fraction was 74%.

I. Scaling and Sustained Performance

Figure 3 shows the weak scaling for the linear and nonlinear case, based on a sub-domain mesh size of 280×280×512 grid points. A parallel efficiency of 99.2% between 4 and 8,192 nodes was measured on OLCF Titan in the linear case, in line with results reported by [52]. We observed some level of performance degration due to the additional four ghost-cell layers included in in the nonlinear case, with the parallel efficiency dropping to 93.1% on 4,096 nodes and 92.6% on 8,192 nodes. On NCSA Blue Waters, a benchmark using topology tuning (based on Cray's Topaware tool [52]) achieved a parallel efficiency of 98.8% on 3,680 nodes even in the nonlinear case. As a result of its higher computational intensity (FLOPS/Bytes=0.537), the nonlinear AWP-ODC achieves a better overall peak performance than the linear code. We

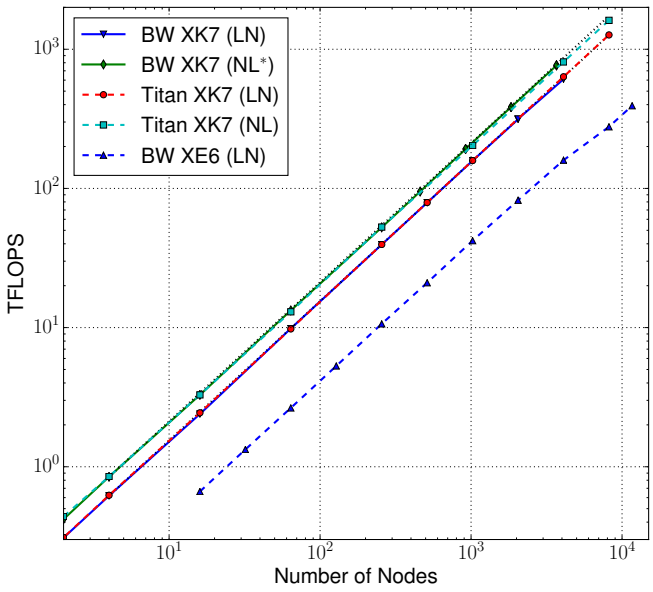

Figure 3. Weak scaling of AWP-ODC CPU and AWP-ODC GPU on NCSA Blue Waters (BW) and OLCF Titan. LN=linear, NL=nonlinear (*with Topology-aware scheduler).

recorded 1.61 PFLOPS running on 8,192 GPUs in the nonlinear case, compared to 1.27 PFLOPS in the linear case (Fig. 3).

J. Optimization of Source Input

Previous scenario simulations [7] with AWP-ODC GPU [e.g., 10] required the generation of a kinematic moment-rate file from results of a dynamic simulation by the *dSrcG* source generator. The *PetaSrcP* source partitioner [7] was then invoked to distribute the kinematic source data to associated processors, which also partitioned the data in the time domain to conserve memory in the case of large-sized dynamic sources. However, the huge amount of source input data required in our high-resolution nonlinear simulations (section IV-D) makes this approach impractical.

In the current version of AWP-ODC GPU, input time series for source nodes are read directly from the output of the previous dynamic simulation, eliminating two steps (the generation of the moment-rate file and the source partitioning) from the entire workflow. These new source input functions significantly reduce the time-to-solution, as pre-processing jobs often spend long times in the queue. Source data are read every few 100 time steps (`READ_STEP`) using collective MPI-IO read calls with explicit offset and stored in buffers on the CPU host. Due to memory constraints, the source buffers are kept much smaller on the GPU side, and source data are copied from the host to the device every few tens of time steps (`READ_STEP_GPU`).

We further optimized the efficiency of the method by offloading source data read operations to an idle CPU core, effectively hiding source input latency behind the computation. Two MPI processes are launched on each node, with the first process interacting with the GPU as in the previous code

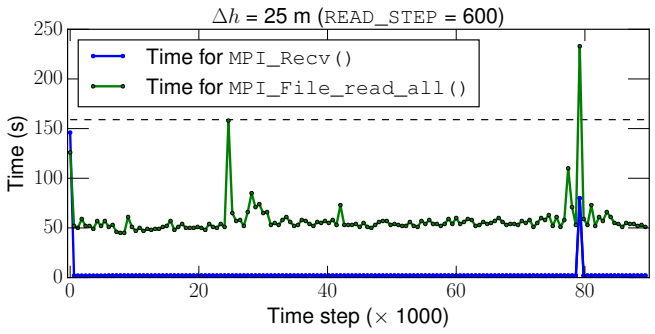

Figure 4. Overlap of source input operation with GPU computation during 4 Hz SAF simulation on NCSA Blue Waters. The blue line shows the time the first process was waiting to receive data from second process. The green line shows time required by the MPI collective read operation on second process. The dashed line shows the approximate time for computing 600 time steps on the GPU.

version, and the second process performing the read operations for the sources in the subdomain assigned to that GPU. Round-robin rank ordering is used to preserve the ranks of processes invoking the GPU and to keep code changes to a minimum.

No overlap occurs while source data are being read for the first `READ_STEP` time steps during initialization, when the first process on the node needs to wait to receive data from the second process. The second process then proceeds reading data for the next `READ_STEP` time steps, while the first process launches the kernels and coordinates communication with other nodes. In our scenario simulations, we used `READ_STEP` = 600, and most source read operation completed within 50 to 80 seconds, well within the ∼160 seconds it took to compute 600 time steps (Fig. 4). This resulted in a full overlap between read operations and computations, except for one read operation after 79,200 time steps, which was delayed due to temporary slow filesystem response.

IV. Nonlinear AWP-ODC San Andreas Simulations

No major earthquake has ruptured the portion of the San Andreas fault south of Parkfield (Fig. 5) since the 1857 Fort Tejon earthquake. The southernmost segment, between Cajon Creek and Bombay Beach, has been locked even longer, since ca. 1680. A large earthquake that would release the seismic slip deficit which has accumulated on this dangerous fault (5–6 m) [53] has been the subject of many previous milestone simulations, such as *TeraShake* (M 7.7)[2], *ShakeOut* (M 7.8) [5] or even *M8* [7]. The scope of our simulations was to take these previous milestone scenarios to higher frequencies, to cover a wider part of the spectrum relevant for civil engineering. We also aim to account for effects that are relevant at higher frequencies (nonlinearity, frequency-dependent attenuation and scattering), taking advantage of the new features implemented in the code.

A. Computational Domain and Fault Geometry

We generated an earthquake scenario similar to the dynamic M 7.8 *ShakeOut-D* simulations. ShakeOut was chosen because it predicts significant ground motions in the San Bernardino and Los Angeles basin, and the level of shaking has been shown to be sensitive to nonlinear effects by our previous low-frequency simulations [29]. Because back-propagation studies [54] suggest that the southernmost portion of the San Andreas fault (south of the bend near Indio) would not contribute significantly to the shaking in the LAB, we approximated the 300 km long fault fault trace from ShakeOut using a shorter, 250 km long segment (Fig. 5).

The SGSN mode in AWP-ODC CPU was used to create a dynamic rupture model for the scenario earthquake. Because the CPU code in dynamic rupture mode is less efficient than the GPU code, a two-step method was employed. In the first step, spontaneous rupture on the planar, vertical fault segment was modeled within a smaller mesh representing the velocity structure around the fault (Fig 5). In the second step, the wave propagation resulting from this dynamic source was simulated within a larger mesh containing the urban areas of the Los Angeles and San Bernardino basin (Fig 5). Both steps of the procedure were performed for a nonlinear and nonlinear medium using a resolution of Δh = 25 m.

B. Geophysical Mesh and Random Heterogeneities

As frequencies in the earthquake source increase, the waves generated are sensitive to smaller and and smaller features along their paths [55–57]. State-of-the-art 3D Community Velocity models, such as the CVM-SI 4.26 [58, 59] used here, accommodate large-scale scattering (scale-length ∼ kilometers), but fail to predict the ground motion complexity due to smaller-scale heterogeneities (scale length ∼ tens of meters to kilometers). For this reason, we superimpose a statistical (von Karman) autocorrelation function onto the CVM-SI 4.26. The parameters describing the statistical distribution, a Hurst exponent of ν = 0.1, autocorrelation lengths of a_z = 100 m and a_h = 500 m in the vertical and horizontal directions, respectively, and a standard deviation of 5% compared to the background media values, is constrained by borehole sonic logs in the Los Angeles basin [60, 61] and generated according to Klimes [62]. The minimum shear-wave velocity was set to

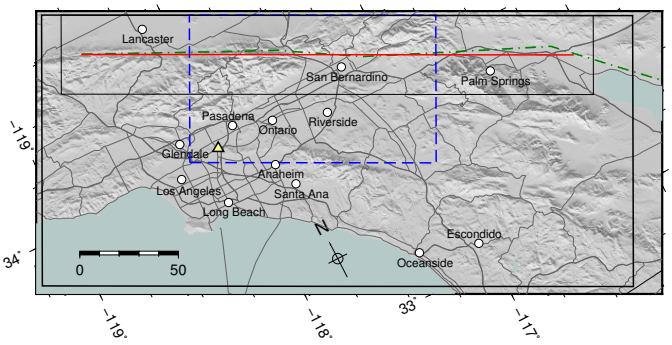

Figure 5. Computational domains used in the dynamic (inner black rectangle) and kinematic (outer black rectangle) simulation of a M 7.7 earthquake on the southern San Andreas fault. The solid red line shows the surface trace of the fault used in this work, the dash-dotted green line the trace used in the ShakeOut-D simulations [6]. The yellow triangle shows the location of site **rus**. The blue dashed rectangle outlines the excerpt shown in Figure 7.

500 m/s, yielding a maximum frequency of 4 Hz with at least 5 grid points per wavelength.

Quality factors at the reference frequency were calculated from the S-wave velocity using the approximate empirical relationship [7] $Q_s = 50 \cdot v_s$ (in km/s) and $Q_p = 2\ Q_s$. We used an exponent of $\gamma = 0.6$ in the power law (4), as this value of has been shown to fit observational data in Southern California and ground motion prediction equations (designed to fit observed events) better than a constant-Q model [63].

For the nonlinear simulations, we defined realistic values of the cohesion c and the friction angle φ using the Hoek-Brown criterion for fractured rock masses [64]. Hoek-Brown parameters were chosen to represent an average quality sandstone, with c and φ computed from known properties at every point on the grid [51].

C. Dynamic Rupture Simulations

For the dynamic rupture simulations we used a mesh of 11,400×1,600×2,048 grid points including the region within 20 km from the fault trace and within 35 km from the bottom of the fault. The Y-axis of our mesh is rotated from north by 27.5°. Initial shear and normal stresses on the fault were chosen to be consistent with the regional stress field. We assumed that the initial principal stress, σ_1, points to 7°W [65, 66], and took the vertical stress (computed from the lithostatic load) as the intermediate principal stress. The ground water table was assumed to be at the surface. We further assumed that $\sigma_1 = \frac{4}{3}\sigma_2$ and = $\sigma_3 = \frac{2}{3}\sigma_2$.

Similar to the increasing sensitivity to small-scale heterogeneities in the medium surrounding the fault, the accuracy of the ground motions at higher frequencies are dependent on realistic complexity in the rupture propagation at larger detail. We introduced such complexity through fault roughness emulated by applying a 2-D random field to the static and dynamic friction coefficients, μ_s and μ_d, respectively [51]. The random field was computed using a von Karman autocorrelation function, [6, 67–70], with autocorrelation lengths computed according to established empirical equations [71]. We used an average friction coefficient of $\overline{\mu_s} = 0.40$ and $\overline{\mu_d} = 0.23$. The average static stress drop is 7.5 MPa, consistent with the high-stress drop assumed for ShakeOut [72].

Dynamic rupture was simulated using 750 nodes (24,000 CPU cores) on NCSA Blue Waters. 90,000 iterations were completed with a temporal resolution of Δt = 1 ms, resulting in a simulation length of 90 s. The linear dynamic simulation completed in 24.5 hours, while the nonlinear dynamic simulation required almost 37 hours.

D. Generation of Kinematic Seismic Source

In previous scenario simulations that used a two-step method [e.g., 6, 7, 68], the sliprates from the dynamic simulations were formulated as moment-rates time histories and transferred to subfaults as a kinematic source in the wave propagation simulation. In a linear simulation, this moment-rate transfer (MRT) preserves the fault slip obtained during the dynamic rupture, and ground motions at near-fault sites

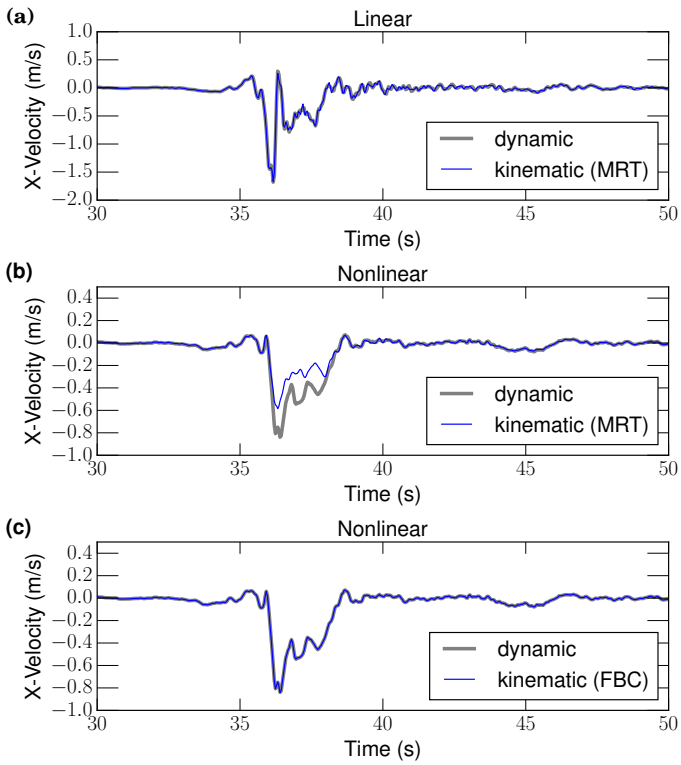

Figure 6. Fault-parallel velocities from dynamic and kinematic simulation at a position 3 grid points from the fault, tested for different source input methods using Δh = 50 m. *(a)* Linear case. MRT = moment rate transfer. *(b)* Nonlinear case. *(c)* Nonlinear case. FBC = fault boundary condition.

are consistent for the two steps (Fig. 6a). If the medium is nonlinear, however, a part of the seismic moment is absorbed by plastic deformation in the fault zone, and not transferred to the kinematic simulation if moment-rates are computed from elastic on-fault slip (Fig. 6b).

We resolved this problem by imposing near-fault particle velocities obtained during the dynamic simulation as a fault boundary condition (FBC) in the kinematic simulation [e.g., 73]. Velocities were saved in a volume of 10,560×5×1000 grid points surrounding the fault. A fault-perpendicular spacing of two grid points from the fault was necessary because the FD operators are 2$^{\text{nd}}$ order accurate in space. Using this method, we obtain consistent near-fault ground motions in the dynamic and kinematic simulation, both for linear and nonlinear media (Fig. 6c).

A drawback of the FBC is that the volume of data that needs to be transferred from the dynamic simulation is much larger than with MRT. The FBC involves the transfer of 3 velocity components inside a volume, while the MRT requires only two components on a surface. In our 4 Hz San Andreas simulations, velocity time series inside the volume amounted to 52 Tb (for comparison, the moment-rate file used for M8 was only 2.1 Tb [7]). Therefore, optimizations targeting input operations for the source, in particular the overlap of read operations with computation on the GPU (section III-J), represent essential code developments for efficient nonlinear wave propagation simulations.

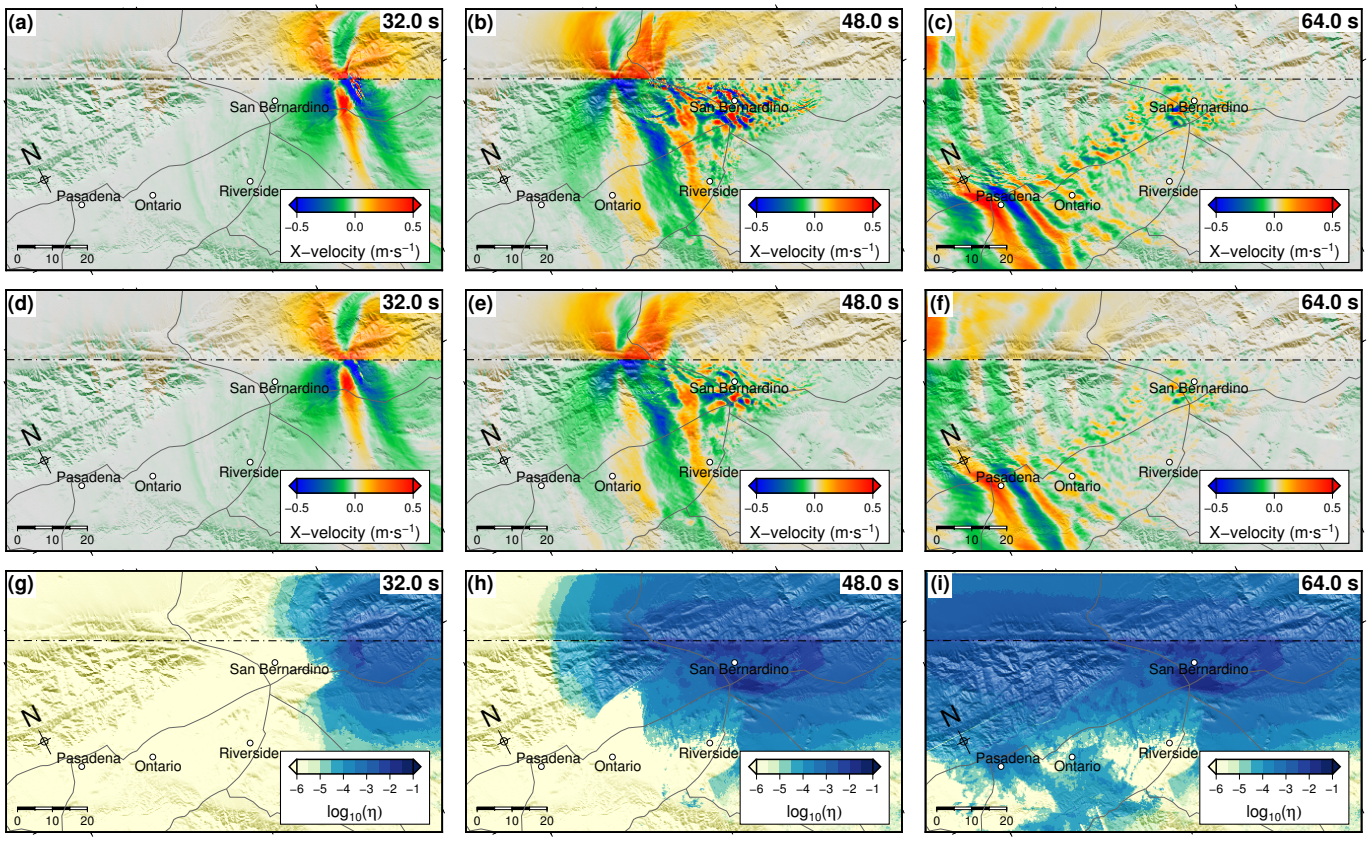

Figure 7. Snapshots from the 4 Hz San Andreas simulation inside the blue rectangle in Fig. 5. *(a-c)* and *(d-f)* show fault-parallel velocity for the linear and nonlinear cases, respectively, and *(g-i)* depict the evolution of permanent plastic strain at the surface obtained from the nonlinear simulation. The dashed line shows the fault trace. An animated version of these snapshots is available in the supplementary data to this article.

E. Wave Propagation Simulations

The wave propagation from the dynamic source was modeled inside a mesh with a dimension of 12,000×5,488×2,048 grid points (Fig. 5) using a resolution of Δh = 25 m and a duration of 120 s (120,000 time steps). The nonlinear simulation, executed on OLCF Titan using 4,200 Cray XK7 nodes, required a wall-clock time of 12 hours, while the linear simulation finished in 8 hours. We saved the output velocity time series and the permanent plastic deformation on the surface of the computational domain.

Snapshots of the near-fault velocity wavefield from the linear simulation (Fig. 7 a-c) are characterized by high-amplitude (> 0.5 m/s) short-period waves reverberating in the San Bernardino basin. The wavefield obtained from the nonlinear simulation (Fig. 7 d-f) contains significantly less high-frequency signals, as they are converted to permanent plastic deformation (Fig. 7 g-h). A comprehensive evaluation of these results in the context of strong motion prediction (which will be published elsewhere) will depend on more simulations, with different realizations of random parameters (such as small-scale heterogeneities and fault friction parameters) and different friction angles and cohesions to analyze the sensitivity of ground motions to the strength of rocks and soils [e.g. 29, 51].

V. Conclusions

We have implemented plasticity based on the Drucker-Prager yield condition in both the CPU and GPU versions of the highly scalable AWP-ODC FD code. An efficient implementation has been introduced which addresses the challenge of computing the yield condition on the staggered FD grid, limiting the cost of accounting for nonlinearity to ∼55% compared to linear simulations. Further optimizations in the plasticity kernel of AWP-ODC GPU have resulted in optimal DRAM throughput while limiting the number of 3-D arrays storing plasticity variables on the GPU. The additional computations performed for nonlinearity have resulted in an overall improved sustained performance of the code.

Simulations of a M 7.7 earthquake on the southern San Andreas fault, performed using 4,200 GPUs on NCSA Blue Waters and OLCF Titan, demonstrate the feasibility of performing large-scale earthquake scenario simulations that account for nonlinear behavior in crustal rocks and soft sediments. We have streamlined the workflow by efficiently overlapping computation with read operations for the kinematic source, which must be prescribed as a fault boundary condition in nonlinear simulations using the two-step modeling procedure for AWP-ODC. Results of our 4 Hz scenario simulations confirm the importance of plasticity for accurate ground motion prediction at the frequencies relevant for engineering design and seismic

hazard analysis.

Given the almost perfect scaling of AWP-ODC GPU, such nonlinear wave propagation simulations can be carried out at even higher frequencies. With the introduction of a discontinuous FD grid, currently in development for AWP-ODC [74], physics-based ground motion simulations will be possible for realistic, nonlinear media for the entire frequency range of engineering interest (0–10 Hz).

ACKNOWLEDGMENT

We thank Carl Pearson and Wen-Mei Hwu of UIUC for their contributions to velocity kernel performance optimization on Kepler, Robert Fiedler of Cray and Gregory Bauer of NCSA for their topology related optimization on Blue Waters, Sharif Islam, JaeHyuk Kwack, Brett Bode, Jing Li and Andriy Kot of NCSA, Matt Belhorn, William Renaud and Judith Hill of ORNL, and Tom Gibbs and Steven Rohm of NVIDIA for their technical support. The authors acknowledge the Office of Science of the U.S. Department of Energy (DOE) for providing HPC resources that have contributed to the research results reported within this paper through an Innovative and Novel Computational Impact on Theory and Experiment (INCITE) program allocation award, and NCSA for providing resources through PRAC program (Petascale Computing Resource Allocation). Computations were performed on Titan, which is part of the Oak Ridge Leadership Facility at the Oak Ridge National Laboratory supported by DOE Contract No. DE-AC05-00OR22725, and Blue Waters at NCSA. Research funding was provided through NCSA Blue Waters PAID program under NSF award OCI-0832698, NSF SI2-SSI (OCI-114849 and OCI-1450451), NSF FESD (EAR-1135455), NSF Geoinformatics (EAR-1349180), NVIDIA Corporate Responsibility GIA award, and XSEDEs Extended Collaborative Support Service (ECSS) program. This research was supported by SCEC which is funded by NSF Cooperative Agreement EAR-0529922 and USGS Cooperative Agreement 07HQAG0008.

REFERENCES

[1] A. Ben-Menahem, "Radiation of seismic surface-waves from finite moving sources", *Bull. seism. Soc. Am.*, vol. 51, no. 3, pp. 401–435, 1961.

[2] K. B. Olsen, S. M. Day, J. B. Minster, Y. Cui, A. Chourasia, M. Faerman, R. Moore, P. Maechling, and T. Jordan, "TeraShake: Strong shaking in Los Angeles expected from southern San Andreas earthquake", *Seism. Res. Lett.*, vol. 77, pp. 281–282, 2006.

[3] S. L. Kramer, *Geotechnical Earthquake Engineering*. Prentice Hall, New Jersey, 1996, p. 653.

[4] R. Taborda and D. Roten, "Physics-based ground-motion simulation", *Encyclopedia of Earthquake Engineering, Springer, Berlin, Heidelberg*, pp. 1–33, 2015.

[5] R. W. Graves, B. T. Aagaard, K. W. Hudnut, L. M. Star, J. P. Stewart, and T. H. Jordan, "Broadband simulations for M_w 7.8 southern San Andreas earthquakes: Ground motion sensitivity to rupture speed", *Geophys. Res. Lett.*, vol. 35, p. L22302, 2008.

[6] K. B. Olsen, S. M. Day, L. A. Dalguer, J. Mayhew, Y. Cui, J. Zhu, V. Cruz-Atienza, D. Roten, P. Maechling, T. Jordan, D. Okaya, and A. Chourasia, "Shakeout-D: ground motion estimates using an ensemble of large earthquakes on the southern San Andreas fault with spontaneous rupture propagation", *Geophys. Res. Lett.*, vol. 36, p. L04303, 2009.

[7] Y. Cui, K. B. Olsen, T. H. Jordan, K. Lee, J. Zhou, P. Small, D. Roten, G. Ely, D. K. Panda, A Chourasia, J. Levesque, S. M. Day, and P. Maechling, "Scalable earthquake simulation on petascale supercomputers", in *Proceedings of SC10, November 13-19, New Orleans, LA*, 2010.

[8] Southern California Earthquake Center. (2016). Shakeout, [Online]. Available: http://www.shakeout.org.

[9] J. Zhou, Y. Cui, E. Poyraz, D. J. Choi, and C. C. Guest, "Multi-GPU implementation of a 3D finite difference time domain earthquake code on heterogeneous supercomputers", *Procedia Computer Science*, vol. 18, pp. 1255–1264, 2013.

[10] Y. Cui, E. Poyraz, K. B. Olsen, J. Zhou, K. Withers, S. Callaghan, J. Larkin, C. Guest, D Choi, A. Chourasia, *et al.*, "Physics-based seismic hazard analysis on petascale heterogeneous supercomputers", in *Proceedings of SC13: International Conference for High Performance Computing, Networking, Storage and Analysis*, ACM, 2013, p. 70.

[11] R. Graves, T. H. Jordan, S. Callaghan, E. Deelman, E. Field, G. Juve, C. Kesselman, P. Maechling, G. Mehta, K. Milner, D. Okaya, P. Small, and K. Vahi, "CyberShake: A physics-based seismic hazard model for southern California", *Pure Appl. Geophys.*, vol. 168, no. 3-4, pp. 367–381, 2011.

[12] K. Withers, K. B. Olsen, and S. M. Day, "Deterministic high-frequency ground motion using dynamic rupture along rough faults, small-scale media heterogeneities, and frequency-dependent attenuation", in *AGU Fall Meeting Abstracts*, vol. 1, 2013, p. 08.

[13] Z. Shi and S. M. Day, "Rupture dynamics and ground motion from 3-D rough-fault simulations", *J. Geophys. Res*, 2013.

[14] K. B. Withers, K. B. Olsen, and S. M. Day, "Memory-efficient simulation of frequency-dependent Q", *Bull. Seism. Soc. Am.*, 2015.

[15] S. M. Day and C. R. Bradley, "Memory-efficient simulation of anelastic wave propagation", *Bull. seism. Soc. Am.*, vol. 91, no. 3, p. 520, 2001.

[16] E. M. Dunham, D. Belanger, L. Cong, and J. E. Kozdon, "Earthquake ruptures with strongly rate-weakening friction and off-fault plasticity, Part 2: Nonplanar faults", *Bull. Seism. Soc. Am.*, vol. 101, no. 5, pp. 2308–2322, 2011.

[17] D. Andrews, "Rupture dynamics with energy loss outside the slip zone", *J. Geophys. Res*, vol. 110, no. B1, B01307, 2005.

[18] S. Ma and D. J. Andrews, "Inelastic off-fault response and three-dimensional dynamics of earthquake rupture on a strike-slip fault", *Journal of Geophysical Research: Solid Earth (1978–2012)*, vol. 115, no. b4, 2010.

[19] B. Duan and S. M. Day, "Inelastic strain distribution and seismic radiation from rupture of a fault kink", *J. Geophys. Res*, vol. 113, no. B12, 2008.

[20] E. M. Dunham, D. Belanger, L. Cong, and J. E. Kozdon, "Earthquake ruptures with strongly rate-weakening friction and off-fault plasticity, Part 1: planar faults", *Bull. Seism. Soc. Am.*, vol. 101, no. 5, pp. 2296–2307, 2011.

[21] A.-A. Gabriel, J.-P. Ampuero, L. A. Dalguer, and P. M. Mai, "Source properties of dynamic rupture pulses with off-fault plasticity", *J. Geophys. Res.*, vol. 118, no. 8, pp. 4117–4126, 2013.

[22] L. F. Bonilla, K. Tsuda, N. Pulido, J. Régnier, and A. Laurendeau, "Nonlinear site response evidence of K-NET and KiK-net records from the 2011 off the Pacific coast of Tohoku Earthquake", *Earth Planets and Space*, vol. 63, no. 7, p. 785, 2011.

[23] J. Regnier, L. F. Bonilla, E. Betrand, and J. F. Semblat, "Empirical evidence of nonlinear site response at several KiK-net stations", in *4th IASPEI/IAEE International Symposium: Effects of Surface Geology on Seismic Motion, August 23-26, 2011, University of Santa Barbara*, 2011.

[24] S. Noguchi, H. Sato, and T. Sasatani, "Characterization of nonlinear site response based on strong motion records at K-NET and KiK-net stations in the east of Japan", in *Proceedings of the 15th WCEE, Sep 24–28 2012, Lisbon, Portugal*, 2012.

[25] D. Roten, D. Fäh, and L. F. Bonilla, "High-frequency ground motion amplification during the 2011 Tohoku earthquake explained by soil dilatancy", *Geophys. J. Int.*, vol. in press, 2013.

[26] W. B. Joyner, "Strong Motion from Surface Waves in Deep Sedimentary Basins", *Bull. seism. Soc. Am.*, vol. 90, no. 6B, S95–112, 2000.

[27] N. H. Sleep, "Nonlinear behavior of strong surface waves trapped in sedimentary basins", *Bull. Seism. Soc. Am.*, vol. 100, no. 2, pp. 826–832, 2010.

[28] N. H. Sleep and B. A. Erickson, "Nonlinear attenuation of S-waves and Love waves within ambient rock", *Geochemistry, Geophysics, Geosystems*, vol. 15, no. 4, pp. 1419–1440, 2014.

[29] D. Roten, K. B. Olsen, S. M. Day, Y. Cui, and D. Fäh, "Expected seismic shaking in Los Angeles reduced by San Andreas fault zone

plasticity", *Geophysical Research Letters*, vol. 41, no. 8, pp. 2769–2777, 2014.

[30] K. B. Olsen, "Simulation of three-dimensional wave propagation in the Salt Lake basin", PhD thesis, University of Utah, Salt Lake City, Utah, 1994, p. 157.

[31] C. Cerjan, D. Kosloff, and M. Reshef, "A nonreflecting boundary condition for discrete acoustic and elastic wave equations", *Geophysics*, vol. 50, pp. 705–708, 1985.

[32] C. Marcinkovich and K. Olsen, "On the implementation of perfectly matched layers in a three-dimensional fourth-order velocity-stress finite difference scheme", *J. geophys. Res*, vol. 108, p. 2276, 2003.

[33] K. C. Meza-Fajardo and A. S. Papageorgiou, "A nonconvolutional, split-field, perfectly matched layer for wave propagation in isotropic and anisotropic elastic media: stability analysis", *Bull. seism. Soc. Am.*, vol. 98, no. 4, p. 1811, 2008, ISSN: 0037-1106.

[34] E. Gottschämmer and K. Olsen, "Accuracy of the explicit planar free-surface boundary condition implemented in a fourth-order staggered-grid velocity-stress finite-difference scheme", *Bull. seism. Soc. Am.*, vol. 91, no. 3, p. 617, 2001.

[35] S. M. Day, "Efficient simulation of constant Q using coarse-grained memory variables", *Bull. seism. Soc. Am.*, vol. 88, no. 4, p. 1051, 1998.

[36] R. Harris, M. Barall, R. Archuleta, E. Dunham, B. Aagaard, J. Ampuero, H. Bhat, V. Cruz-Atienza, L. Dalguer, P. Dawson, *et al.*, "The SCEC/USGS dynamic earthquake rupture code verification exercise", *Seismological Research Letters*, vol. 80, no. 1, pp. 119–126, 2009.

[37] L. A. Dalguer and S. M. Day, "Staggered-grid split-node method for spontaneous rupture simulation", *J. Geoph. Res.*, vol. 112, B02302, 2007.

[38] R. A. Harris, M. Barall, D. J. Andrews, B. Duan, S. Ma, E. M. Dunham, A.-A. Gabriel, Y. Kaneko, Y. Kase, B. T. Aagaard, J.-P. Oglesby, J. P. Ampuero, T. C. Hanks, and N. Abrahamson, "Verifying a computational method for predicting extreme ground motion", *Seism. Res. Lett.*, vol. 82, no. 5, pp. 638–644, 2011.

[39] Barall, M. (2016). The SCEC/USGS Spontaneous Rupture Code Verification Project, [Online]. Available: http://scecdata.usc.edu/cvws/.

[40] Y. Cui, R. Moore, K. Olsen, A. Chourasia, P. Maechling, B. Minster, S. Day, Y. Hu, J. Zhu, and T. Jordan, "Toward petascale earthquake simulations", *Acta Geotechnica*, vol. 4, no. 2, pp. 79–93, 2009.

[41] J. Zhou, D. Unat, D. J. Choi, C. C. Guest, and Y. Cui, "Hands-on performance tuning of 3D finite difference earthquake simulation on GPU Fermi chipset", *Procedia Computer Science*, vol. 9, pp. 976–985, 2012.

[42] J. Yu, D. Roten, and Y. Cui, "Development of Parallel IO Interface for High Performance SEISM-IO Library", in *SCEC 2015 Annual Meeting proceeding and abstracts*, 2015.

[43] H. Xu, E. Poyraz, D. Roten, and Y. Cui, "A High Level Parallel IO Library for High- Performance Seismic Applications", in *SCEC 2014 Annual Meeting proceeding and abstracts*, 2014.

[44] The HDF Group. (1997-2016). Hierarchical Data Format, version 5, [Online]. Available: http://www.hdfgroup.org/HDF5/.

[45] J. F. Lofstead, S. Klasky, K. Schwan, N. Podhorszki, and C. Jin, "Flexible IO and integration for scientific codes through the adaptable IO system (ADIOS)", in *Proceedings of the 6th international workshop on Challenges of large applications in distributed environments*, ACM, 2008, pp. 15–24.

[46] J. Li, W.-K. Liao, A. Choudhary, R. Ross, R. Thakur, W. Gropp, R. Latham, A. Siegel, B. Gallagher, and M. Zingale, "Parallel netCDF: a high-performance scientific I/O interface", in *Supercomputing, 2003 ACM/IEEE Conference*, IEEE, 2003, pp. 39–39.

[47] M. L. Wilkins, "Calculation of elastic-plastic flow", in *Methods of Computational Physics*, B. Alder, Ed. Academic Press, New York, 1964, vol. 3.

[48] J. C. Simo and T. J. R. Hughes, "Computational inelasticity", in *Interdisciplinary Applied Mathematics*, J. E. Marsden, L. Sirovich, and S. Wiggins, Eds. Springer-Verlag, Berlin, 1998, vol. 7.

[49] D. Andrews, T. C. Hanks, and J. W. Whitney, "Physical limits on ground motion at Yucca Mountain", *Bull. seism. Soc. Am.*, vol. 97, no. 6, pp. 1771–1792, 2007.

[50] R. Madariaga, "Dynamics of an expanding circular fault", *Bull. Seism. Soc. Am.*, vol. 66, no. 3, pp. 639–666, 1976.

[51] D. Roten, K. B. Olsen, Y. Cui, and S. M. Day, "Quantification of fault zone plasticity effects with spontaneous rupture simulations", in *Best Practices in Physics-based Fault Rupture Models for Seismic Hazard Assessment of Nuclear Installations Vienna, Austria, November 18-20, 2015*, 2015.

[52] Y. Cui, E. Poyraz, J. Zhou, S. Callaghan, P. Maechling, T. H. Jordan, L. Shih, and P. Chen, "Accelerating Cybershake calculations on the XE6/XK7 platform of Blue Waters", in *Extreme Scaling Workshop (XSW), 2013*, IEEE, 2013, pp. 8–17.

[53] R. Weldon, K. Scharer, T. Fumal, and G. Biasi, "Wrightwood and the earthquake cycle: what a long recurrence record tells us about how faults work", *Geol. Seism. Am. Today*, vol. 14, pp. 4–10, 2004.

[54] S. M. Day, D. Roten, and K. B. Olsen, "Adjoint analysis of the source and path sensitivities of basin-guided waves", *Geophys. J. Int.*, vol. 189, no. 2, pp. 1103–1124, 2012.

[55] S. Hartzell, S. Harmsen, and A. Frankel, "Effects of 3D random correlated velocity perturbations on predicted ground motions", *Bull. Seism. Soc. Am.*, vol. 100, no. 4, pp. 1415–1426, 2010.

[56] W. Imperatori and P. M. Mai, "Broad-band near-field ground motion simulations in 3-dimensional scattering media", *Geophysical Journal International*, vol. 192, no. 2, pp. 725–744, 2013.

[57] ——, "The role of topography and lateral velocity heterogeneities on near-source scattering and ground-motion variability", *Geophysical Journal International*, vol. 202, no. 3, pp. 2163–2181, 2015.

[58] H. Magistrale, S. M Day, R. W. Clayton, and R. Graves, "The SCEC Southern California Reference Three-Dimensional Seismic Velocity Model Version 2", *Bull. Seism. Soc. Am.*, vol. 90, no. 6B, S65–S76, 2000.

[59] E.-J. Lee, P. Chen, and T. H. Jordan, "Testing waveform predictions of 3D velocity models against two recent Los Angeles earthquakes", *Seismological Research Letters*, vol. 85, no. 6, pp. 1275–1284, 2014.

[60] N. Nakata and G. C. Beroza, "Stochastic characterization of mesoscale seismic velocity heterogeneity in Long Beach, California", *Geophysical Journal International*, vol. 203, no. 3, pp. 2049–2054, 2015.

[61] W. H. Savran and K. B. Olsen, "Model for small-scale crustal heterogeneity in Los Angeles basin based on inversion of sonic log data", *Geophysical Journal International*, vol. 205, no. 2, pp. 856–863, 2016.

[62] L. Klimeš, "Correlation functions of random media", *Pure and Applied Geophysics*, vol. 159, no. 7-8, pp. 1811–1831, 2002.

[63] K. Withers, K. B. Olsen, Z. Shi, and S. Day, "Broadband (0-8 Hz) Ground Motion Variability from Ensemble Simulations of Buried M_w 6.7 Thrust Earthquakes Including Rough Fault Descriptions and $Q(f)$", in *SCEC 2015 Annual Meeting proceeding and abstracts*, 2015.

[64] E. Hoek, C. Carranza-Torres, and B. Corkum, "Hoek-Brown failure criterion - 2002 edition", *Proceedings of NARMS-Tac*, vol. 1, pp. 267–273, 2002.

[65] L. M. Flesch, W. E. Holt, A. J. Haines, and B. Shen-Tu, "Dynamics of the Pacific-North American plate boundary in the western United States", *Science*, vol. 287, no. 5454, pp. 834–836, 2000.

[66] C. H. Scholz, "Evidence for a strong San Andreas fault", *Geology*, vol. 28, no. 2, 2002.

[67] L. A. Dalguer, S. M. Day, K. B. Olsen, and V. M. Cruz-Atienza, "Rupture models and ground motion for Shakeout and other southern San Andreas fault scenarios", in *Proceedings of 14th World Conference on Earthquake Engineering, Int. Assoc. for Earthquake Eng., Beijing, China*, 2008.

[68] D. Roten, K. B. Olsen, J. C. Pechmann, V. M. Cruz-Atienza, and H. Magistrale, "3D simulations of M 7 earthquakes on the Wasatch fault, Utah, Part I: Long-period (0-1 Hz) ground motions", *Bull. seism. Soc. Am.*, vol. 101, no. 5, pp. 2045–2063, 2011.

[69] L. A. Dalguer and P. M. Mai, "Prediction of near-source ground motion exceeding 1g at low frequencies ($<$ 2Hz) from Mw 6.5 deterministic and physics-based dynamic rupture simulations", in *Proceedings of 15th World Conference on Earthquake Engineering, Int. Assoc. for Earthquake Eng., Lisbon, Portugal*, 2012.

[70] C. Baumann and L. A. Dalguer, "Evaluating the compatibility of dynamic rupture-based synthetic ground motion with empirical ground-motion prediction equation", *Bull. Seism. Soc. Am.*, vol. 104, no. 2, 2014.

[71] P. M. Mai and G. C. Beroza, "A spatial random field model to characterize complexity in earthquake slip", *Journal of Geophysical Research (Solid Earth)*, vol. 107, pp. 2308–+, Nov. 2002.

[72] R. W. Graves, B. T. Aagaard, and K. W. Hudnut, "The Shakeout earthquake source and ground motion simulations", *Earthquake Spectra*, vol. 27, no. 2, pp. 273–291, 2011.

[73] D. Andrews, "Test of two methods for faulting in finite-difference calculations", *Bull. Seism. Soc. Am.*, vol. 89, no. 4, pp. 931–937, 1999.

[74] S. Nie, "Discontinuous staggered finite difference method for anelastic wave simulations", in *2015 AGU Fall Meeting*, AGU, 2015.