SIAM J. SCI. COMPUT.
Vol. 31, No. 5, pp. 3356–3386

A TWO–SCALE SOLUTION ALGORITHM FOR THE ELASTIC WAVE EQUATION*

TETYANA VDOVINA†, SUSAN E. MINKOFF‡, AND SEAN M.L. GRIFFITH§

Abstract. Operator-based upscaling is a two-scale algorithm that speeds up the solution of the wave equation by producing a coarse grid solution which incorporates much of the local fine-scale solution information. We present the first implementation of operator upscaling for the elastic wave equation. By using the velocity-displacement formulation of the three-dimensional elastic wave equation, basis functions that are linear in all three directions, and applying mass lumping, the subgrid solve (first stage of the two-step algorithm) reduces to solving explicit difference equations. At the second stage of the algorithm, we upscale both velocity and displacement by using local subgrid information to formulate the coarse-grid problem. The coarse-grid system matrix is independent of time, sparse, and banded. This paper explores both serial and parallel implementations of the method. The main simplifying assumption of the method (special zero boundary conditions imposed on coarse blocks in the first stage of the algorithm) leads to an easily parallelizable algorithm because very little communication is required between processors. In fact, for this upscaling implementation calculation of the load vector for the coarse solve dominates the cost of a time step. We show that for a homogeneous medium convergence is second-order in space and time so long as both the coarse and fine grids are simultaneously refined. A series of heterogeneous-medium numerical experiments demonstrate that the upscaled solution captures the fine-scale fluctuations in the input parameters accurately. Most notable for use in a seismic inversion algorithm, the upscaling algorithm accurately locates the depth of reflectors (interface changes).

Key words. upscaling, elastic wave propagation, multiscale methods, seismology

AMS subject classifications. 35L05, 86-08, 65M60, 86A15

DOI. 10.1137/080714877

1. Introduction.

For scientists to be able to produce oil and gas, to predict earthquakes and other tectonic events such as tsunamis, to safely remediate contaminants, and to bury excess greenhouse gases underground, they must first be able to image Earth's subsurface. Rock layers, fluids, and faults need to be mapped and their depths and lateral extent understood. To create an image of the subsurface, energy is sent into the ground which generates a wave. The heterogeneous nature of the subsurface causes a portion of these waves to be sent back to the surface where seismometers (microphones) record the waves as they pass. From these signals scientists

*Received by the editors February 1, 2008; accepted for publication (in revised form) April 22, 2009; published electronically August 14, 2009. This work was partially supported by funding from the Collaborative Math-Geoscience Program at NSF (grant EAR-0222181).
http://www.siam.org/journals/sisc/31-5/71487.html
†Department of Mathematics and Statistics, University of Maryland, Baltimore County, 1000 Hilltop Circle, Baltimore, MD 21250. Current address: Department of Computational and Applied Mathematics, Rice University, 6100 Main St., MS 134, Houston, TX 77025 (vdovina@rice.edu). The work of this author was partially supported by NASA Goddard's Earth Sciences and Technology Center (GEST), National Science Foundation grants DMS-0512673 and DMS-0714193, AFOSR grant FA9550-05-1-0473, the Rice Inversion Project, and the State of Texas funding through the Texas Enterprise Fund.
‡Department of Mathematics and Statistics, University of Maryland, Baltimore County, 1000 Hilltop Circle, Baltimore, MD 21250 (sminkoff@umbc.edu).
§Department of Mathematics and Statistics, University of Maryland, Baltimore County, 1000 Hilltop Circle, Baltimore, MD 21250. Current address: The Center for Inherited Disease Research, The Johns Hopkins University, Triad 2000, 4940 Eastern Avenue, Baltimore, MD 21224 (SML. Griffith@gmail.com). The work of this author was supported by the National Science Foundation through UMBC's ADVANCE grant (SBE-0244880).

try to infer the structure of the subsurface (solve an inverse minimization problem between measured data and data predicted by a wave equation model). This inference is enormously complicated by both the very complex mechanical nature of rock and the vast amounts of data which are often collected in seismic studies. Solving the inverse problem involves repeated solution of the forward problem (wave equation) and is prohibitively expensive for elastic wave propagation in three dimensions.

In this paper we explore ways to speed up the solution of the forward problem. Seeking more efficient elastic algorithms is crucial for geoscience applications, since solving the elastic wave equation on fine grids by standard methods is overwhelming even for modern-age technology. In computational seismology, the domains of interest may be as large as 100 wavelengths in each direction. Finite-difference methods require 7 to 10 grid points per wavelength to minimize dispersion, so a typical three-dimensional (3D) problem may include 10^9 grid points. Approximately 100 flops are required to compute a finite-difference solution for the isotropic elastic wave equation at each grid point [13]. Therefore, the number of flops per simulation consisting of 10^2–10^4 time steps can be as high as 10^{15}. A typical survey in computational seismology involves 10^4–10^6 simulations or 10^{19} flops and could take thirty years to complete on a Gflop machine.

To speed up solution of the wave equation we have adapted upscaling techniques originally developed in the context of elliptic flow problems to the wave equation. Examples of multiscale methods include the multiscale finite element method [17, 2] and its mixed version [10], the variational multiscale method [18], operator-based upscaling [4, 5, 7], mortar upscaling [28], and the heterogeneous multiscale method [14], among others.

The multiscale approach to wave propagation problems was first introduced and exploited in the framework of homogenization theory [9]. Assuming that a typical wavelength is small and on the order of the period of the medium, one can explicitly construct the effective or homogenized partial differential operator with simple coefficients that describe the macroscopic properties of the underlying medium. Recently Ohwadi et al. [26, 27] developed a homogenization procedure that does not assume scale separation or ergodicity (two assumptions typically required by standard homogenization theory). The method (metric-based upscaling) is developed in the context of finite elements and makes use of oscillating test functions previously discussed and implemented in papers by Hou et al. [17, 10]. The main difference between metric-based upscaling and other methods based on oscillating test functions is that the test functions are constructed using global harmonic coordinates rather than being the solution of a local cell problem. This approach allows the authors to avoid the resonance problem and establish convergence in the presence of a continuum of scales.

Operator-based upscaling uses a decomposition of the unknowns into the coarse- and fine-scale components and solves the problem in two stages. (See papers by Arbogast et al. for a thorough treatment of the method in the context of elliptic flow problems [4, 5, 6, 3, 7].) The method proceeds as follows: At the first stage, the problem is solved locally for fine-scale information interior to coarse blocks. At the second stage, the impact of this subgrid information is incorporated into the global problem solved on the coarse grid [4].

In the first paper to appear on operator upscaling for the wave equation [31], we adapted the method to the constant density, variable sound velocity acoustic wave equation in two dimensions. Operator upscaling was originally developed in the context of mixed finite elements [4]. As a result, the upscaling algorithm introduces

mass matrices on the fine and coarse scales and appears to be more expensive than standard staggered finite-difference approaches [23, 34, 35] in terms of both memory and speed. One way to reduce the storage and to avoid solving the linear system is to use mass-lumping techniques [13] which result in diagonal mass matrices on both the fine and coarse scales [22]. Since one must solve for most of the original fine-grid unknowns at the subgrid stage of the algorithm, the majority of the computational work occurs at this stage. The good news is that homogeneous boundary conditions imposed along coarse block edges for the subgrid solve lead to a natural parallelization strategy (i.e., there is no communication between coarse blocks at the first stage of the algorithm so each processor can take ownership of a subset of coarse blocks without need of input from other processors). In Vdovina et al. [31], we describe our parallel algorithm for using operator upscaling to solve the acoustic wave equation in detail and demonstrate that it provides near optimal speedup without the communication overhead required by standard data parallelism. The numerical accuracy of the method is studied for several geophysically realistic experiments that show that the upscaled solution captures the fine-scale fluctuations of the medium accurately. In two related papers ([22, 32]), we study the physical meaning of the upscaled equations and rigorously analyze convergence of the method.

Motivated by the efficiency and accuracy we observed applying operator upscaling to the acoustic problem, we turn in this paper to the much more computationally expensive problem of 3D elastic wave propagation. As in the case of acoustics, the upscaling algorithm relies on a two-scale decomposition of unknowns that is best described in the context of finite elements. However, we would like to avoid storing and solving large linear systems typically associated with finite element methods. The velocity-displacement formulation of the elastic wave equation (see Komatitsch et al. [19, 21]) and mass lumping result in the spectral finite element method which we will use as our basis for the upscaling algorithm described in this paper. In this upscaling implementation we upscale both displacement and velocity (six unknowns in three dimensions) which differs from the acoustic implementation which upscaled velocity only (pressure was not upscaled).

For the elastic implementation described here, the interactions between basis functions are more complicated than for acoustics. As a result, mass lumping does not produce a diagonal mass matrix for the coarse problem, but the matrix does have a sparse and banded structure, and it can be assembled once before the start of the time-step loop. In contrast to our implementation of the upscaling algorithm for acoustics, the most expensive part of the elastic upscaling algorithm is assembling the coarse-grid load vectors and (in some cases) solution of the coarse-grid system matrix. The coarse-grid load vectors are constructed from inner products that depend on input parameters defined on the fine grid. One of the advantages of operator-based upscaling is that it does not require explicit averaging of input data. Instead, we approximate the coarse-grid inner products by fine-grid quadrature rules. In the case of elasticity, this procedure involves approximating nine inner products for each of the three velocity equations and two inner products for each of the three displacements.

The first implementations of operator upscaling for elliptic problems used numerical Green's functions to treat the subgrid and coarse components independently [6, 3]. Unfortunately, numerical Green's functions become prohibitive for the elastic wave equation. The upscaling algorithm described in this paper relies on the augmented solution (sum of coarse and fine solutions). In Vdovina and Minkoff [32] we show that for the acoustic upscaling algorithm, the augmented solution approach leads to an algorithm that is twice as fast as the algorithm based on numerical Green's functions.

In sections 2 and 3, we derive the weak form of the velocity-displacement formulation for the 3D elastic wave equation and describe the two-scale finite element method used as a basis for upscaling. Sections 4–6 contain the description of the elastic upscaling algorithm. In section 4, we introduce the augmented solution and establish the connection between this solution and the standard numerical Green's function technique. The implementation of the sugbgrid and coarse problems is discussed in sections 5 and 6, respectively. In section 7, we study the numerical convergence and accuracy of the elastic upscaling algorithm. Our experiments show that the upscaling algorithm performs well even in those challenging cases where the heterogeneity of the input model is on a scale smaller than a single coarse block. Finally, in section 8 we describe parallelization of the elastic upscaling method and performance of the algorithm.

2. Model problem and weak form. We derive the velocity-displacement formulation from the equation of balance of linear momentum:

$$\rho(\mathbf{x})\frac{\partial^2 \mathbf{u}(t,\mathbf{x})}{\partial t^2} = \nabla \cdot \boldsymbol{\sigma}(\mathbf{x}) + \mathbf{f}(t,\mathbf{x}) \text{ for } t \in [0,T] \text{ and } \mathbf{x} \in \Omega \subset \mathbb{R}^3, \tag{2.1}$$

where $\mathbf{u}(t,\mathbf{x})$ is the displacement vector, $\rho(\mathbf{x})$ is the density of the material, $\boldsymbol{\sigma}(\mathbf{x})$ is the stress tensor, and $\mathbf{f}(t,\mathbf{x})$ is a body force [1, 8]. We convert the second-order equation (2.1) into a first-order system by introducing velocity as the time derivative of displacement:

$$\rho(\mathbf{x})\frac{\partial \mathbf{v}}{\partial t}(t,\mathbf{x}) = \nabla \cdot \boldsymbol{\sigma}(\mathbf{x}) + \mathbf{f}(t,\mathbf{x}) \quad \text{in } \Omega, \tag{2.2}$$

$$\rho(\mathbf{x})\frac{\partial \mathbf{u}}{\partial t}(t,\mathbf{x}) = \rho(\mathbf{x})\mathbf{v}(t,\mathbf{x}) \quad \text{in } \Omega. \tag{2.3}$$

To this set of equations we add the following boundary and initial conditions:

$$\mathbf{u}(0,\mathbf{x}) = \mathbf{u}_0(\mathbf{x}), \tag{2.4}$$

$$\mathbf{v}(0,\mathbf{x}) = \mathbf{v}_0(\mathbf{x}), \tag{2.5}$$

$$\boldsymbol{\sigma} \cdot \boldsymbol{\nu} = \mathbf{0} \quad \text{on } \partial\Omega, \tag{2.6}$$

where $\boldsymbol{\nu}$ is the unit outward normal to the boundary $\partial\Omega$. The operator upscaling method can be applied to a problem with arbitrary boundary conditions. The zero-traction boundary condition (2.6) was chosen because it models the interface of a solid with the air. In particular, this condition is used in geoscience applications to model the surface of Earth [13, 19]. To formulate the variational problem, we choose W to be the set of functions $\mathbf{w}(\mathbf{x}) \in H^1(\Omega)$ such that $\mathbf{w}(\mathbf{x}) = \mathbf{0}$ on $\partial\Omega$. Multiplying (2.2)–(2.3) by $\mathbf{w} \in W$, integrating over Ω, and applying the divergence theorem to (2.2), we obtain the following variational problem: For all $\mathbf{w}(\mathbf{x})$ in W and $t \in [0,T]$ find $\mathbf{v}(t,\mathbf{x})$ and $\mathbf{u}(t,\mathbf{x})$ in W such that

$$\left(\rho\frac{\partial \mathbf{v}}{\partial t}, \mathbf{w}\right) = -\left(\boldsymbol{\sigma}, \nabla \mathbf{w}\right) + \left(\mathbf{f}, \mathbf{w}\right), \tag{2.7}$$

$$\left(\rho\frac{\partial \mathbf{u}}{\partial t}, \mathbf{w}\right) = \left(\rho\mathbf{v}, \mathbf{w}\right), \tag{2.8}$$

where $(\cdot,\cdot)$ denotes the L^2 inner product over Ω. The boundary term $\int_\Gamma \boldsymbol{\sigma} \cdot \boldsymbol{\nu}\mathbf{w}d\Gamma$ vanishes due to boundary condition (2.6).

We use the following relation to eliminate stress variables from (2.7)–(2.8):

$$\sigma_{ij} = \lambda \sum_{k=1}^{3} \frac{\partial u_k}{\partial x_k} \delta_{ij} + \mu \left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} \right), \tag{2.9}$$

where we use (x_1, x_2, x_3) to refer to the components of the vector $\mathbf{x}$. (See references [1, 8] for details.) To minimize the use of subscripts, in the remainder of the paper we denote the components of the vector $\mathbf{x}$ by (x, y, z).

In component form, the stress-free variational formulation of the elastic wave propagation problem is given by the following equations:

$$\begin{aligned} \left(\rho \frac{\partial v_1}{\partial t}, w\right) = &- \left((\lambda + 2\mu)\frac{\partial u_1}{\partial x} + \lambda \frac{\partial u_2}{\partial y} + \lambda \frac{\partial u_3}{\partial z}, \frac{\partial w}{\partial x}\right) \\ &- \left(\mu \frac{\partial u_1}{\partial y} + \mu \frac{\partial u_2}{\partial x}, \frac{\partial w}{\partial y}\right) \\ &- \left(\mu \frac{\partial u_1}{\partial z} + \mu \frac{\partial u_3}{\partial x}, \frac{\partial w}{\partial z}\right) + (f_1, w), \end{aligned} \tag{2.10}$$

$$\begin{aligned} \left(\rho \frac{\partial v_2}{\partial t}, w\right) = &- \left(\mu \frac{\partial u_1}{\partial y} + \mu \frac{\partial u_2}{\partial x}, \frac{\partial w}{\partial x}\right) \\ &- \left(\lambda \frac{\partial u_1}{\partial x} + (\lambda + 2\mu)\frac{\partial u_2}{\partial y} + \lambda \frac{\partial u_3}{\partial z}, \frac{\partial w}{\partial y}\right) \\ &- \left(\mu \frac{\partial u_2}{\partial z} + \mu \frac{\partial u_3}{\partial y}, \frac{\partial w}{\partial z}\right) + (f_2, w), \end{aligned} \tag{2.11}$$

$$\begin{aligned} \left(\rho \frac{\partial v_3}{\partial t}, w\right) = &- \left(\mu \frac{\partial u_1}{\partial z} + \mu \frac{\partial u_3}{\partial x}, \frac{\partial w}{\partial x}\right) \\ &- \left(\mu \frac{\partial u_3}{\partial y} + \mu \frac{\partial u_2}{\partial z}, \frac{\partial w}{\partial y}\right) \\ &- \left(\lambda \frac{\partial u_1}{\partial x} + \lambda \frac{\partial u_2}{\partial y} + (\lambda + 2\mu)\frac{\partial u_3}{\partial z}, \frac{\partial w}{\partial z}\right) + (f_3, w), \end{aligned} \tag{2.12}$$

$$\left(\rho \frac{\partial u_i}{\partial t}, w\right) = (\rho v_i, w) \quad \text{for} \quad i = 1, 2, 3 \tag{2.13}$$

and $w(\mathbf{x})$ in W, $t \in [0, T]$.

Various formulations of the elastic wave equation exist. Specifically, the velocity-stress formulation of the elastic wave equation is widely used by the seismic community. Typically these equations are solved via finite differences [34, 23, 35]. As an alternative, the stress-free formulation forms the basis for the spectral element method (see Komatitsch et al. [19, 20, 21]). This formulation leads to reduced memory requirements and less expensive linear systems to solve. Thus we focus on this formulation for the upscaling algorithm described in this paper. Other formulations may very well be amenable to upscaling. Cohen and Fauqueux in [12] modified the velocity stress formulation by introducing new vector-valued unknowns. They use the resulting nonclassical decomposition of the elasticity system to develop a mixed version of the spectral method.

3. Finite element method. We begin by constructing a 3D two-scale mesh. We decompose the domain Ω into a coarse mesh $T_H(\Omega)$, and then subdivide each

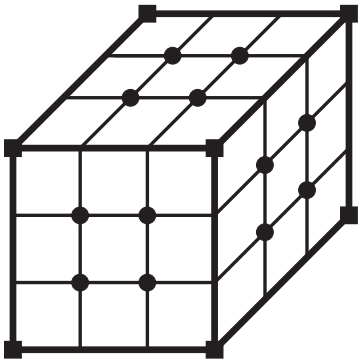


FIG. 1. *A single coarse grid block subdivided into* $3 \times 3 \times 3$ *fine-grid blocks. (Note that the computational domain would contain multiple coarse-grid blocks. However, for simplicity we show only one here.) The squares represent the location of both coarse velocity and coarse displacement unknowns* $\mathbf{v}^c$ *and* $\mathbf{u}^c$*. The circles give the location of the subgrid velocity and displacement unknowns* $\delta\mathbf{v}$ *and* $\delta\mathbf{u}$*.*

coarse block E_c into subgrid blocks. The finite element space over the composite fine mesh is given by

$$W_{H,h} = W_H \oplus \delta W_h, \tag{3.1}$$

where coarse space W_H is defined on $T_H(\Omega)$ and subgrid space δW_h is defined for each coarse block E_c. Both spaces consist of vector functions with components linear in all three spatial directions. The nodes for the subgrid and coarse spaces are located at the corners of the subgrid and coarse blocks, respectively (see Figure 1). Therefore, each coarse and subgrid block has eight nodes and eight basis functions corresponding to these nodes. We construct the subgrid and coarse basis functions as products of 1D linear Lagrange polynomials determined in terms of control points x_q so that $l_p(x_q) = \delta_{pq}$ [13].

In this work, we upscale both velocity and displacement. Using decomposition (3.1), we separate the displacement and velocity unknowns into coarse and subgrid parts. We approximate the derivatives with respect to time by second-order centered finite differences and obtain the finite element formulation of the problem represented by (2.10)–(2.13). Specifically, we want to find $\mathbf{u} = \mathbf{u}^c + \delta\mathbf{u}$ and $\mathbf{v} = \mathbf{v}^c + \delta\mathbf{v}$ in $W_{H,h}$ so that for all w in $W_{H,h}$

(3.2)

$$\begin{aligned}
&\left(\rho(v_1^c + \delta v_1)^{n+1}, w\right) = \left(\rho(v_1^c + \delta v_1)^n, w\right) \\
&-\Delta t \left[\left((\lambda + 2\mu) \frac{\partial}{\partial x} (u_1^c + \delta u_1)^{n+\frac{1}{2}} + \lambda \frac{\partial}{\partial y} (u_2^c + \delta u_2)^{n+\frac{1}{2}} + \lambda \frac{\partial}{\partial z} (u_3^c + \delta u_3)^{n+\frac{1}{2}}, \frac{\partial}{\partial x} w \right) \right. \\
&+ \left(\mu \frac{\partial}{\partial y} (u_1^c + \delta u_1)^{n+\frac{1}{2}} + \mu \frac{\partial}{\partial x} (u_2^c + \delta u_2)^{n+\frac{1}{2}}, \frac{\partial}{\partial y} w \right) \\
&\left. + \left(\mu \frac{\partial}{\partial z} (u_1^c + \delta u_1)^{n+\frac{1}{2}} + \mu \frac{\partial}{\partial x} (u_3^c + \delta u_3)^{n+\frac{1}{2}}, \frac{\partial}{\partial z} w \right) - \left(f_1^{n+\frac{1}{2}}, w \right) \right],
\end{aligned}$$

(3.3)
$$\begin{aligned}
&\left(\rho(v_2^c+\delta v_2)^{n+1},w\right)=\left(\rho(v_2^c+\delta v_2)^{n},w\right)\\
&-\Delta t\left[\left(\mu\frac{\partial}{\partial y}(u_1^c+\delta u_1)^{n+\frac12}+\mu\frac{\partial}{\partial x}(u_2^c+\delta u_2)^{n+\frac12},\frac{\partial}{\partial x}w\right)\right.\\
&+\left(\lambda\frac{\partial}{\partial x}(u_1^c+\delta u_1)^{n+\frac12}+(\lambda+2\mu)\frac{\partial}{\partial y}(u_2^c+\delta u_2)^{n+\frac12}+\lambda\frac{\partial}{\partial z}(u_3^c+\delta u_3)^{n+\frac12},\frac{\partial}{\partial y}w\right)\\
&\left.+\left(\mu\frac{\partial}{\partial z}(u_2^c+\delta u_2)^{n+\frac12}+\mu\frac{\partial}{\partial y}(u_3^c+\delta u_3)^{n+\frac12},\frac{\partial}{\partial z}w\right)-\left(f_2^{n+\frac12},w\right)\right],
\end{aligned}$$

(3.4)
$$\begin{aligned}
&\left(\rho(v_3^c+\delta v_3)^{n+1},w\right)=\left(\rho(v_3^c+\delta v_3)^{n},w\right)\\
&-\Delta t\left[\left(\mu\frac{\partial}{\partial z}(u_1^c+\delta u_1)^{n+\frac12}+\mu\frac{\partial}{\partial x}(u_3^c+\delta u_3)^{n+\frac12},\frac{\partial}{\partial x}w\right)\right.\\
&+\left(\mu\frac{\partial}{\partial y}(u_3^c+\delta u_3)^{n+\frac12}+\mu\frac{\partial}{\partial z}(u_2^c+\delta u_2)^{n+\frac12},\frac{\partial}{\partial y}w\right)\\
&+\left(\lambda\frac{\partial}{\partial x}(u_1^c+\delta u_1)^{n+\frac12}+\lambda\frac{\partial}{\partial y}(u_2^c+\delta u_2)^{n+\frac12}+(\lambda+2\mu)\frac{\partial}{\partial z}(u_3^c+\delta u_3)^{n+\frac12},\frac{\partial}{\partial z}w\right)\\
&\left.-\left(f_3^{n+\frac12},w\right)\right],
\end{aligned}$$

(3.5)
$$\left(\rho\left(u_q^c+\delta u_q\right)^{n+3/2},w\right)=\left(\rho\left(u_q^c+\delta u_q\right)^{n+\frac12},w\right)+\Delta t\left(\rho\left(v_q^c+\delta v_q\right)^{n+1},w\right),$$

where n and Δt denote time level and time step, respectively.

4. Operator upscaling method. We do not solve (3.2)–(3.5) directly. Instead, we localize the subgrid information by imposing homogeneous boundary conditions on coarse blocks and apply the upscaling algorithm. The upscaling process consists of two steps and produces a solution on the coarse grid. First, we restrict to subgrid test functions δw in (3.2)–(3.5) and obtain a series of subgrid problems, one for each coarse block E_c. In the second step, we use the subgrid solutions to augment the problem solved on the coarse grid. The coarse-grid problem comes from (3.2)–(3.5), where now the coarse basis functions w^c are used as test functions.

In order to obtain local subgrid information, we need to solve (3.2)–(3.5) with subgrid basis functions used as test functions. However, these equations involve coarse velocity and displacement solutions $\mathbf{v}^c$ and $\mathbf{u}^c$ that are unknown at the subgrid stage of the algorithm. One way to obtain the subgrid information independently of the coarse solutions is to use the numerical Green's function technique [6, 3]. The numerical Green's function technique allows one to eliminate the unknown coarse components from the subgrid equations by breaking the original problem into a set of simple subproblems (essentially analogous to applying unit forcing functions at each node of the coarse block and then using superposition to construct a total solution from the sum of these solutions and the solution to the problem with the physical source as right-hand side). After the coarse problem is solved, the subgrid solutions have to be constructed as a linear combination of the influence functions and coarse coefficients. This construction is extremely expensive, since it depends on several matrix-vector products. In particular, implementation of numerical Green's functions for the 3D elastic wave equation involves eight matrix-vector products for each of the six unknowns at every time step. We propose to replace the numerical Green's function technique by a much less expensive approach.

Our approach is based on the observation that the inner products in (3.2)–(3.5) depend on the sum of the subgrid and coarse solutions rather than on the unknown coarse solution alone. In order to solve (3.2)–(3.5) numerically, we approximate these inner products using fine-grid quadrature rules. Our analysis shows that the sum of the subgrid and coarse solutions evaluated at nodes internal to a coarse block is completely independent of the unknown coarse information. In Lemma 4.1, we introduce the augmented solution as a sum of the coarse and subgrid components and explain the connection between this solution and the influence function solution that comes from the numerical Green's function technique.

LEMMA 4.1. *Let $\delta\mathbf{v}$, $\delta\mathbf{u}$ and $\mathbf{v}^c$, $\mathbf{u}^c$ be the subgrid and coarse components of the velocity and displacement solutions. The augmented solutions $\mathbf{v}$ and $\mathbf{u}$ are defined on the fine grid as follows:*

$$\mathbf{v} = \delta\mathbf{v} + \mathbf{v}^c, \qquad \mathbf{u} = \delta\mathbf{u} + \mathbf{u}^c. \tag{4.1}$$

We will use $\delta\mathbf{v}^0$ and $\delta\mathbf{u}^0$ to denote the solution of (3.2)–(3.5) *with subgrid basis functions used as test functions and $\mathbf{v}^c$ and $\mathbf{u}^c$ set to zero on time level $n+1$.*

1. *At nodes internal to coarse block E_c, the augmented solution reduces to*

$$\mathbf{v}_{i,j,k} = \delta\mathbf{v}^0_{i,j,k}, \qquad \mathbf{u}_{i,j,k} = \delta\mathbf{u}^0_{i,j,k}, \tag{4.2}$$

   where local indices i, j, k range over the subgrid nodes internal to coarse block E_c.
2. *At nodes located on coarse block faces, the augmented velocity and displacement solutions reduce to the coarse solution evaluated at these nodes.*

For the proof of this result we refer the reader to [33]. More detail on the numerical Green's function technique applied to upscaling of the acoustic wave equation can be found in [33, 32, 31]. The practical significance of this lemma follows from the fact that it provides a computationally inexpensive way to obtain the subgrid information without having to construct the subgrid solution via numerous matrix-vector products at the end of each time step. Timing studies performed for the acoustic problem demonstrate that the upscaling algorithm based on the augmented solution is two times faster than the version of the algorithm based on numerical Green's functions [32]. This result implies that employing the augmented solution is crucial for efficiency and performance of the elastic upscaling algorithm. Lemma 4.1 indicates that the coarse grid computation requires only the component of the subgrid solution that represents the influence of the source. In the following section, we discuss the structure of the linear system that corresponds to the subgrid equations.

5. Subgrid problem. We obtain the subgrid equations by using subgrid basis functions as test functions in (3.2)–(3.5). The structure of the linear system that corresponds to each of the resulting subgrid equations is determined by the basis functions and quadrature rules that we use to approximate the inner products. As mentioned in section 3, the basis functions are constructed as products of linear Lagrange polynomials. We show below that this choice of basis functions leads to a diagonal mass matrix. The process of replacing the mass matrix with a diagonal matrix is known as mass lumping and relies on using a quadrature rule which evaluates the function to be integrated at the node points where the basis functions have a value of one [11], [13]. In the derivation that follows, we lump the subgrid mass and stiffness matrices by applying a quadrature rule on the subgrid scale. Therefore, in this derivation we assume that input parameters ρ, λ, and μ vary smoothly within

a fine-grid cell. No further smoothness is required of the parameters. Application of the mass-lumping technique allows us to write subgrid linear systems as explicit difference equations. Since the difference form of the elastic system in three dimensions is long, we keep the presentation simple here by describing the procedure leading to the difference equations. The complete equations are given in Appendix A.

The mass matrices for the six velocity and displacement components are all computed similarly. To simplify notation, we omit the time level index $n+1$, use v to refer to the first component of the velocity solution, and illustrate the mass matrix computation for this component of velocity only. The inner product on the left-hand side of (3.2) is the only inner product that contributes to the mass matrix. The computation is based on the augmented solution and results of Lemma 4.1. We begin by using the fact that the sum of subgrid and coarse solutions can be replaced by the augmented solution v:

$$(\rho(\mathbf{x})(v^c(\mathbf{x})+\delta v(\mathbf{x})),\delta w_{i,j,k}(\mathbf{x}))_{E_c} = (\rho(\mathbf{x})v(\mathbf{x}),\delta w_{i,j,k}(\mathbf{x}))_{E_c}. \tag{5.1}$$

To determine the entries of the mass matrix, we integrate over the support of basis function $\delta w_{i,j,k}(\mathbf{x}) = \delta w_i(x)\delta w_j(y)\delta w_k(z)$ and use the trapezoid rule to approximate the integrals. The support of basis function $\delta w_{i,j,k}$ is given by a tensor product of 1D intervals $[x_{i-1},x_{i+1}]\times[y_{j-1},y_{j+1}]\times[z_{k-1},z_{k+1}]$:

$$\begin{aligned}(\rho(\mathbf{x})v(\mathbf{x}),\delta w_{i,j,k}(\mathbf{x}))_{E_c}&\\ = \int_{z_{k-1}}^{z_{k+1}}\delta w_k(z)\int_{y_{j-1}}^{y_{j+1}}\delta w_j(y)\int_{x_{i-1}}^{x_{i+1}}\rho(\mathbf{x})v(\mathbf{x})\delta w_i(x)\,d\mathbf{x}.&\end{aligned} \tag{5.2}$$

Applying the trapezoid rule on each of the intervals $[z_{k-1},z_k]$, $[z_k,z_{k+1}]$, $[y_{j-1},y_j]$, $[y_j,y_{j+1}]$, $[x_{i-1},x_i]$, and $[x_i,x_{i+1}]$ and using the fact that each linear basis function is nonzero at one grid node only, we obtain

$$(\rho(\mathbf{x})v(\mathbf{x}),\delta w_{i,j,k}(\mathbf{x}))_{E_c} \approx \Delta x\Delta y\Delta z\rho_{i,j,k}v(\mathbf{x}_{i,j,k}) = \Delta x\Delta y\Delta z\rho_{i,j,k}\delta v^0_{i,j,k}, \tag{5.3}$$

where the last equality follows from statement 1 of Lemma 4.1. Equation (5.3) implies that the mass matrix is diagonal with entries $\Delta x\Delta y\Delta z\rho_{i,j,k}$, where Δx, Δy, and Δz denote the fine-grid space steps.

The inner products on the right-hand side of (3.2)–(3.5) form the entries of the load vectors. As in the case of the mass matrices, we approximate integrals directly on the interval of support of the basis functions and analytically obtain an explicit difference scheme that corresponds to (3.2)–(3.5). To illustrate the procedure, we approximate the inner product that involves the derivatives of u_1 and δw with respect to x (see the second inner product on the right-hand side of (3.2)). As before, we simplify notation by omitting subscript 1 and time level superscript n. The rest of the inner products can be computed in a similar way. The inner product over the coarse block E_c reduces to the integral over the support of basis function $\delta w_{i,j,k}$:

$$\begin{aligned}\left((\lambda+2\mu)(\mathbf{x})\frac{\partial}{\partial x}u(\mathbf{x}),\frac{\partial}{\partial x}\delta w_{i,j,k}(\mathbf{x})\right)_{E_c}&\\ = \int_{z_{k-1}}^{z_{k+1}}\int_{y_{j-1}}^{y_{j+1}}\int_{x_{i-1}}^{x_{i+1}}(\lambda+2\mu)(\mathbf{x})\frac{\partial}{\partial x}u(\mathbf{x})\frac{\partial}{\partial x}\delta w(\mathbf{x})d\mathbf{x}.&\end{aligned} \tag{5.4}$$

Next, we replace $u(\mathbf{x})$ by its finite element expansion reduced to coarse block E_c:

$$\begin{aligned}(5.5)\quad &\int_{z_{k-1}}^{z_{k+1}}\int_{y_{j-1}}^{y_{j+1}}\int_{x_{i-1}}^{x_{i+1}}(\lambda+2\mu)(\mathbf{x})\frac{\partial}{\partial x}u(\mathbf{x})\frac{\partial}{\partial x}\delta w(\mathbf{x})d\mathbf{x}\\ &=\int_{z_{k-1}}^{z_{k+1}}\int_{y_{j-1}}^{y_{j+1}}\int_{x_{i-1}}^{x_{i+1}}(\lambda+2\mu)(\mathbf{x})\frac{\partial}{\partial x}\left(\sum_{i',j',k'}\hat{u}_{i',j',k'}\delta w_{i',j',k'}(\mathbf{x})\right)\frac{\partial}{\partial x}\delta w(\mathbf{x})d\mathbf{x},\end{aligned}$$

where by $\hat{u}_{i',j',k'}$ we denote the coefficients in the finite element expansion of u and indices i', j', and k' range over all subgrid nodes internal to coarse block E_c. Rearranging the terms in the last equation, we obtain

$$(5.6)\quad \sum_{i',j',k'}\hat{u}_{i',j',k'}\int_{z_{k-1}}^{z_{k+1}}\delta w_{k'}\delta w_k\int_{y_{j-1}}^{y_{j+1}}\delta w_{j'}\delta w_j\int_{x_{i-1}}^{x_{i+1}}(\lambda+2\mu)\frac{\partial}{\partial x}\delta w_{i'}\frac{\partial}{\partial x}\delta w_i d\mathbf{x}.$$

As in the case of the mass matrix, we apply the trapezoid rule on each of the intervals $[z_{k-1},z_k]$, $[z_k,z_{k+1}]$, and $[y_{j-1},y_j]$, $[y_j,y_{j+1}]$ and use the fact that basis functions are nonzero at a single grid point:

$$\begin{aligned}(5.7)\quad &\left((\lambda+2\mu)(\mathbf{x})\frac{\partial}{\partial x}u(\mathbf{x}),\frac{\partial}{\partial x}\delta w_{i,j,k}(\mathbf{x})\right)_{E_c}\\ &\quad\approx\Delta y\Delta z\sum_{i'}\hat{u}_{i',j,k}\int_{x_{i-1}}^{x_{i+1}}(\lambda+2\mu)(z_k,y_j,x)\frac{\partial}{\partial x}\delta w_{i'}(x)\frac{\partial}{\partial x}\delta w_i(x)dx.\end{aligned}$$

Basis function δw_i has zero interaction with all the basis functions except for δw_{i-1} on $[x_{i-1},x_i]$, δw_{i+1} on $[x_i,x_{i+1}]$, and itself. Therefore, (5.7) becomes

$$\begin{aligned}(5.8)\quad &\Delta y\Delta z\left[\hat{u}_{i-1,j,k}\int_{x_{i-1}}^{x_i}(\lambda+2\mu)(z_k,y_j,x)\frac{\partial}{\partial x}\delta w_{i-1}(x)\frac{\partial}{\partial x}\delta w_i(x)dx\right.\\ &\quad+\hat{u}_{i,j,k}\int_{x_{i-1}}^{x_{i+1}}(\lambda+2\mu)(z_k,y_j,x)\left(\frac{\partial}{\partial x}\delta w_i(x)\right)^2dx\\ &\quad\left.+\hat{u}_{i+1,j,k}\int_{x_i}^{x_{i+1}}(\lambda+2\mu)(z_k,y_j,x)\frac{\partial}{\partial x}\delta w_{i+1}(x)\frac{\partial}{\partial x}\delta w_i(x)dx\right].\end{aligned}$$

The hat basis function $\delta w_i(x)$ is continuous on each of the intervals $[x_{i-1},x_i]$ and $[x_i,x_{i+1}]$ with derivatives given by $1/\Delta x$ on $[x_{i-1},x_i]$ and $-1/\Delta x$ on $[x_i,x_{i+1}]$. Using this fact and applying the trapezoidal rule to approximate each of the integrals, we obtain

$$\begin{aligned}(5.9)\quad &\frac{\Delta x\Delta y\Delta z}{2}\left[-\hat{u}_{i-1,j,k}\frac{(\lambda+2\mu)_{i-1,j,k}+(\lambda+2\mu)_{i,j,k}}{\Delta x^2}\right.\\ &\quad+\hat{u}_{i,j,k}\frac{(\lambda+2\mu)_{i-1,j,k}+2(\lambda+2\mu)_{i,j,k}+(\lambda+2\mu)_{i+1,j,k}}{\Delta x^2}\\ &\quad\left.-\hat{u}_{i+1,j,k}\frac{(\lambda+2\mu)_{i,j,k}+(\lambda+2\mu)_{i+1,j,k}}{\Delta x^2}\right].\end{aligned}$$

Rearranging the terms gives

$$\begin{aligned}\left((\lambda+2\mu)(\mathbf{x})\frac{\partial}{\partial x}u(\mathbf{x}),\frac{\partial}{\partial x}\delta w(\mathbf{x})\right)_{E_c}&\\ \approx -\frac{\Delta x\Delta y\Delta z}{2}\Bigg[&(\lambda+2\mu)_{i-1,j,k}\frac{\hat{u}_{i-1,j,k}-\hat{u}_{i,j,k}}{\Delta x^2}\\ &+(\lambda+2\mu)_{i,j,k}\frac{\hat{u}_{i-1,j,k}-2\hat{u}_{i,j,k}+\hat{u}_{i+1,j,k}}{\Delta x^2}\\ &+(\lambda+2\mu)_{i+1,j,k}\frac{\hat{u}_{i+1,j,k}-\hat{u}_{i,j,k}}{\Delta x^2}\Bigg].\end{aligned}\tag{5.10}$$

Recall that indices i, j, and k range over subgrid nodes internal to coarse block E_c. Therefore, according to Lemma 4.1 augmented coefficients $\hat{u}_{i,j,k}$ that correspond to these nodes reduce to the subgrid solutions $\delta u^0_{i,j,k}$. Coefficients $\hat{u}_{i-1,j,k}$ and $\hat{u}_{i-1,j,k}$ may be located on coarse block faces, in which case they become coarse solutions evaluated at the face nodes. Notice also that in the case of homogeneous Lamé parameters, expression (5.10) reduces to a second order finite-difference scheme:

$$-\Delta x\Delta y\Delta z(\lambda+2\mu)\frac{u_{i-1,j,k}-2u_{i,j,k}+u_{i+1,j,k}}{\Delta x^2}.\tag{5.11}$$

In fact, in the case of a homogeneous medium, the explicit difference scheme that corresponds to the subgrid problem becomes a standard second order in space and time finite-difference scheme as studied by Cohen in [13].

6. Coarse problem. In this section, we will show that the coarse-grid mass matrix has a sparse and banded structure and describe the procedure for assembling the nonzero diagonals of the global mass matrix from the element matrices.

Each of equations (3.2)–(3.5) with coarse-grid basis functions used as test functions generates a linear system of equations for the coefficients in the finite element representation of the coarse velocity and displacement solutions. As in the subgrid problem, we apply fine-grid quadrature to approximate the inner products in (3.2)–(3.5), since these equations involve the density and Lamé parameters defined on the fine grid, and we do not wish to average these input quantities. Although the resulting coarse-grid mass matrices are not diagonal, they have a sparse and banded structure. Moreover, we will show that we can use the same mass matrix for all six equations. The banded structure of the mass matrix follows from the fact that each coarse basis function has zero interaction with all basis functions except for itself and its immediate neighbors (resulting in 27 nonzero diagonals). For example, in the case of a domain discretized into $20\times20\times20$ coarse grid blocks, the dense mass matrix consists of $2^3\cdot10^3\times2^3\cdot10^3=2^6\cdot10^6$ entries for each of the six equations. We need to store only $27\cdot2^3\cdot10^3$ nonzero entries (less than 1% of all entries). The nonzero diagonals are assembled from the element mass matrices. As in the case of the subgrid problem, the procedure for computing the element mass matrices is based on the augmented solution and results of Lemma 4.1. As before, we note that the sum of the subgrid and coarse solutions is the augmented solution $\mathbf{v}$ and rewrite the inner product over the whole domain as a sum of the inner products over the coarse blocks:

$$\left(\rho(\mathbf{x})\left(\mathbf{v}^c(\mathbf{x})+\delta\mathbf{v}(\mathbf{x})\right),\mathbf{w}^c(\mathbf{x})\right)_\Omega=\sum_{E_c\in\Omega}\left(\rho(\mathbf{x})\mathbf{v}(\mathbf{x}),\mathbf{w}^c(\mathbf{x})\right)_{E_c}\tag{6.1}$$

$$=\sum_{E_c\in\Omega}\sum_{\delta E\in E_c}\int_{\delta E}\rho(\mathbf{x})\mathbf{v}(\mathbf{x})\mathbf{w}^c(\mathbf{x})d\mathbf{x}.\tag{6.2}$$

Here we omit the time-level index n and further rewrite (6.1) as a sum of the inner products over the subgrid blocks located in coarse block E_c. Setting $\mathbf{w}^c(\mathbf{x})$ equal to each of the eight coarse basis functions $\mathbf{w}_l^c$ for $l = 1, \ldots, 8$ located at the corners of coarse block E_c and applying the trapezoid rule on each block δE, we obtain

$$\sum_{\delta E \in E_c} \int_{\delta E} \rho(\mathbf{x}) \mathbf{v}(\mathbf{x}) \mathbf{w}_l^c(\mathbf{x}) d\mathbf{x} \approx \sum_{\delta E \in E_c} \frac{\Delta x \Delta y \Delta z}{8} \sum_{i,j,k=1}^{2} \rho(\mathbf{x}_{i,j,k}) \mathbf{v}(\mathbf{x}_{i,j,k}) \mathbf{w}_l^c(\mathbf{x}_{i,j,k}), \tag{6.3}$$

where $\mathbf{x}_{i,j,k}$ for i, j, $k = 1, 2$ are the quadrature nodes that coincide with the subgrid nodes located at the corners of subgrid blocks δE. Lemma 4.1 states that the value of the augmented solution at a particular grid node depends on the location of this node. Therefore, we have to distinguish between the nodes located inside coarse block E_c and the nodes located on its faces. We denote the set of nodes internal to coarse block E_c by I, and nodes located on the boundary of the coarse block are contained in set B. Using this notation, we rewrite the expression on the right-hand side of (6.3) as a sum over all the subgrid nodes. Using Lemma 4.1, we replace the augmented solution by the subgrid solution due to the source (denoted by superscript 0) at the internal nodes and by the coarse solution at the boundary nodes:

$$\frac{\Delta x \Delta y \Delta z}{8} \left(8 \sum_{\mathbf{x}_{i,j,k} \in I} \rho(\mathbf{x}_{i,j,k}) \delta \mathbf{v}_{i,j,k}^0 \mathbf{w}_l^c(\mathbf{x}_{i,j,k}) + \sum_{\mathbf{x}_{i,j,k} \in B} a(\mathbf{x}_{i,j,k}) \rho(\mathbf{x}_{i,j,k}) \mathbf{v}^c(\mathbf{x}_{i,j,k}) \mathbf{w}_l^c(\mathbf{x}_{i,j,k}) \right), \tag{6.4}$$

where we gain a factor of 8 for the summation over the internal nodes because each internal node is shared by eight neighboring blocks. To complete the computation, we evaluate the coarse solution $\mathbf{v}^c$ at subgrid nodes $\mathbf{x}_{i,j,k}$ using a finite element expansion. The finite element expansion of the coarse solution evaluated at $\mathbf{x}_{i,j,k}$ is a linear combination of eight coefficients $\mathbf{v}_{l'}^c$ and basis functions $\mathbf{w}_{l'}^c$:

$$\frac{\Delta x \Delta y \Delta z}{8} \left(8 \sum_{\mathbf{x}_{i,j,k} \in I} \rho(\mathbf{x}_{i,j,k}) \delta \mathbf{v}_{i,j,k}^0 \mathbf{w}_l^c(\mathbf{x}_{i,j,k}) + \sum_{\mathbf{x}_{i,j,k} \in B} a(\mathbf{x}_{i,j,k}) \rho(\mathbf{x}_{i,j,k}) \sum_{l'=1}^{8} \mathbf{v}_{l'}^c \mathbf{w}_{l'}^c(\mathbf{x}_{i,j,k}) \mathbf{w}_l^c(\mathbf{x}_{i,j,k}) \right). \tag{6.5}$$

The second term in (6.5) is the only term that involves the unknown coarse coefficients and gives input into the local mass matrix. Based on relation (6.5) we come to the following conclusions about the coarse-grid mass matrix. First, the subgrid solutions do not provide input into the mass matrices. Therefore, we can use the same mass matrix for all the coarse equations. Second, since coarse basis functions and density are functions of spatial location only, we compute the local mass matrices and assemble the nonzero diagonals of the global matrix outside the time step loop. Finally, relation (6.5) shows that the entries of the mass matrix are constructed from

the weighted density values taken along coarse block faces with weights given by coarse basis functions. This result is similar to the result obtained by Korostyshevskaya and Minkoff [22] for the acoustic wave equation. Namely, this result shows that to a certain degree the algorithm compensates for the homogeneous boundary condition imposed on coarse blocks at the subgrid stage.

7. Numerical experiments. In this section we discuss the numerical convergence and accuracy of the upscaling algorithm. To our knowledge, no dispersion or stability analysis presently exists for the spectral finite element method. In Cohen [13], the author provides stability analysis for the second order in space and time finite-difference scheme that corresponds to the elastic wave equation with homogeneous input data. Recall from section 5 that in the case of a homogeneous medium, explicit difference equations obtained at the subgrid stage of our algorithm reduce to the standard finite-difference scheme discussed in [13]. In this work our rule of thumb is therefore a 3D version of the stability condition provided in [13]:

$$\Delta t \sqrt{\frac{1}{\Delta x^2} + \frac{1}{\Delta y^2} + \frac{1}{\Delta z^2}} \leq \frac{1}{\sqrt{V_p^2 + V_s^2}}, \tag{7.1}$$

where $V_p = \sqrt{(\lambda + 2\mu)/\rho}$ and $V_s = \sqrt{\mu/\rho}$ are maximal compressional and shear wave velocities respectively. To ensure that the stability condition is satisfied on both the coarse and subgrids, we use the fine-grid space step in (7.1), since it produces a more restrictive condition than the coarse space step. To minimize the effect of dispersion, we follow the heuristic rule of thumb discussed in the papers by Komatitsch et al. [20, 21]. We ensure that roughly five coarse-grid nodes are taken per minimum wavelength in each spatial direction. The dispersion condition on the subgrid is then automatically satisfied, since each coarse block is subdivided into a number of subgrid blocks.

7.1. Numerical convergence. In this section, we present numerical experiments that illustrate the convergence properties of the upscaling algorithm. As is standard practice for evaluating the rate of convergence, we consider here waves propagating in a homogeneous medium. The domain is a cube of size 1 m $\times$ 1 m $\times$ 1 m, and the source function is chosen to produce the closed form velocity and displacement solutions:

$$V_i = 2t \sin(2\pi x) \sin(2\pi y) \sin(2\pi z), \tag{7.2}$$

$$U_i = (t^2 - 0.25\Delta t^2) \sin(2\pi x) \sin(2\pi y) \sin(2\pi z) \tag{7.3}$$

for $i = 1, 2, 3$.

In experiment 1, we ran the upscaling code using one coarse block for the whole domain. In this case, the coarse component of the solution is equal to zero due to the zero-traction boundary condition (2.6), and the upscaling algorithm produces a solution equivalent to a standard full finite element solution. Since our basis functions are piecewise linear interpolating polynomials, we expect to see second-order convergence. Table 7.1 summarizes the convergence results for Experiment 1. The first two columns show the number of fine- and coarse-grid blocks in the domain. Column three gives the number of time steps. Column four contains relative errors at the final time between the first component of the augmented velocity solution and analytical solution in the infinity norm. Finally, the last column displays the rate of convergence. Each row of Table 7.1 presents a different subtest. When the grid

TABLE 7.1

Relative errors for the first component of augmented velocity when the domain contains a single coarse block (i.e., no upscaling is done).

Number of fine blocks	Number of coarse blocks	Number of time steps	$\frac{\|\|V_1 - v_1\|\|_\infty}{\|\|V_1\|\|_\infty}$	Rate
$50 \times 50 \times 50$	$1 \times 1 \times 1$	50	1.6664e−03	–
$100 \times 100 \times 100$	$1 \times 1 \times 1$	100	4.1485e−04	2.0
$200 \times 200 \times 200$	$1 \times 1 \times 1$	200	1.0377e−04	1.9

TABLE 7.2

Relative errors for the first component of augmented velocity for Experiment 2 *in which both the fine and coarse grids are refined.*

Number of fine blocks	Number of coarse blocks	Number of time steps	$\frac{\|\|V_1 - v_1\|\|_\infty}{\|\|V_1\|\|_\infty}$	Rate
$50 \times 50 \times 50$	$5 \times 5 \times 5$	50	2.4554e−01	–
$100 \times 100 \times 100$	$10 \times 10 \times 10$	100	5.5976e−02	2.1
$200 \times 200 \times 200$	$20 \times 20 \times 20$	200	1.5578e−02	1.9
$400 \times 400 \times 400$	$40 \times 40 \times 40$	400	3.8776e−03	2.0

TABLE 7.3

Relative errors for the first component of augmented velocity for Experiment 3 *in which the fine grid is refined but the coarse grid is fixed.*

Number of fine blocks	Number of coarse blocks	Number of time steps	$\frac{\|\|V_1 - v_1\|\|_\infty}{\|\|V_1\|\|_\infty}$	Rate
$50 \times 50 \times 50$	$10 \times 10 \times 10$	50	5.5967e−02	–
$100 \times 100 \times 100$	$10 \times 10 \times 10$	100	5.7067e−02	–
$200 \times 200 \times 200$	$10 \times 10 \times 10$	200	5.6439e−02	–
$400 \times 400 \times 400$	$10 \times 10 \times 10$	400	5.7506e−02	–

spacing is reduced by a factor of 2, the relative error goes down by a factor of 4. Thus the last two columns of Table 7.1 demonstrate that for a homogeneous medium with a single coarse block, the algorithm gives quadratic convergence.

In experiment 2, we show that if both the fine and coarse grids are refined, the upscaling algorithm preserves the second-order convergence rate. In each subtest, we reduce the grid spacing by half for both fine and coarse grids. The last two columns of Table 7.2 indicate that the method in this case exhibits quadratic convergence. One notes that although the input parameter fields (compressional and shear wave velocities and density) are homogeneous for these experiments, we might still expect to see less than quadratic convergence as the upscaling algorithm imposes zero boundary conditions on coarse block edges at the subgrid stage of the algorithm.

Our goal in Experiment 3 is to demonstrate that convergence cannot be expected if the coarse grid is fixed. In this experiment, we fix the coarse step size and reduce the fine-grid space step (and time step) by a factor of 2 in each subtest. We see from Table 7.3 that the error is constant. We conclude that the augmented solution error in the infinity norm is dominated by the the coarse solution error. Therefore, if the coarse grid is not refined, we cannot expect convergence.

In Experiment 4, we fix the fine grid and refine the coarse grid. We see second-order convergence of the augmented solution to the true solution in this case (see Table 7.4). As in Experiment 3, the coarse solution error dominates the augmented error.

TABLE 7.4
Relative errors for the first component of augmented velocity for Experiment 4 in which the coarse grid is refined but the fine grid is held fixed.

Number of fine blocks	Number of coarse blocks	Number of time steps	$\frac{\|\|V_1 - v_1\|\|_\infty}{\|\|V_1\|\|_\infty}$	Rate
$200 \times 200 \times 200$	$5 \times 5 \times 5$	200	2.4770e−01	–
$200 \times 200 \times 200$	$10 \times 10 \times 20$	200	5.6439e−02	2.1
$200 \times 200 \times 200$	$20 \times 20 \times 20$	200	1.5578e−02	1.9
$200 \times 200 \times 200$	$40 \times 40 \times 40$	200	3.7272e−03	2.1

7.2. Accuracy. To study the accuracy of the upscaling method applied to the elastic wave equation, we compare the upscaled and full finite element solutions for three sets of heterogeneous medium experiments. As we mentioned in the previous section, the full finite element solution is obtained by running the upscaling algorithm with a single coarse block. In the first experiment the medium is a layered earth model. In the second experiment we consider a periodic checkerboard medium corresponding to two alternating materials. The final experiment we discuss in this section is run on a larger domain, and the input velocity and density fields are truly heterogeneous. In this case we have chosen a single realization of a stochastic von Karman distribution of a two-component material mixture [16, 15].

In computational seismology, a fundamental elastic wave source is a point force. The source, located at position $\mathbf{x}_s$, is given by the following equation:

$$\mathbf{f}(t, \mathbf{x}) = Ah(t)g\left(|\mathbf{x} - \mathbf{x}_s|^2\right)\mathbf{a}, \tag{7.4}$$

where A is the amplitude, $\mathbf{a}$ describes the orientation of the force, and $h(t)$ and $g(|\mathbf{x} - \mathbf{x}_s|)$ define the waveform [13, 12, 21, 29].

Classical results on approximation of differential equations assume regularity of the source function. Since operator-based upscaling is based on a standard finite element method, it does not impose any additional restrictions on the smoothness of the source functions beyond those required by finite element or finite-difference methods. As in the case with standard methods, the straightforward representation of seismic sources by discrete delta functions produces solutions which, while reasonably accurate, converge only weakly.

In all the experiments, we set $\mathbf{a}$ equal to $\mathbf{x} - \mathbf{x}_s$ so that the force is uniformly distributed in all three spatial directions. We use the 3D version of a Gaussian for the spatial component g:

$$g(\mathbf{x}) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left\{\frac{|\mathbf{x} - \mathbf{x}_s|}{2\sigma^2}\right\}, \tag{7.5}$$

where σ controls the spatial support of the source. For the time-dependent component of the source, we use a Ricker wavelet (a standard seismic source which is the second derivative of a Gaussian; see [30]):

$$h(t) = -2\pi^2 f_0 \exp^{-(\pi f_0 t)^2}\left(1 - 2(\pi f_0 t)^2\right), \tag{7.6}$$

where the central frequency f_0 is set to 1.7 Hz (see Figure 2).

In the layered medium and checkerboard experiments, we will consider a cubical domain of size $12 \times 12 \times 12$ km discretized into $120 \times 120 \times 120$ fine-grid blocks. The wavelength is approximately 1 km, and therefore, the domain is 12 wavelengths in each

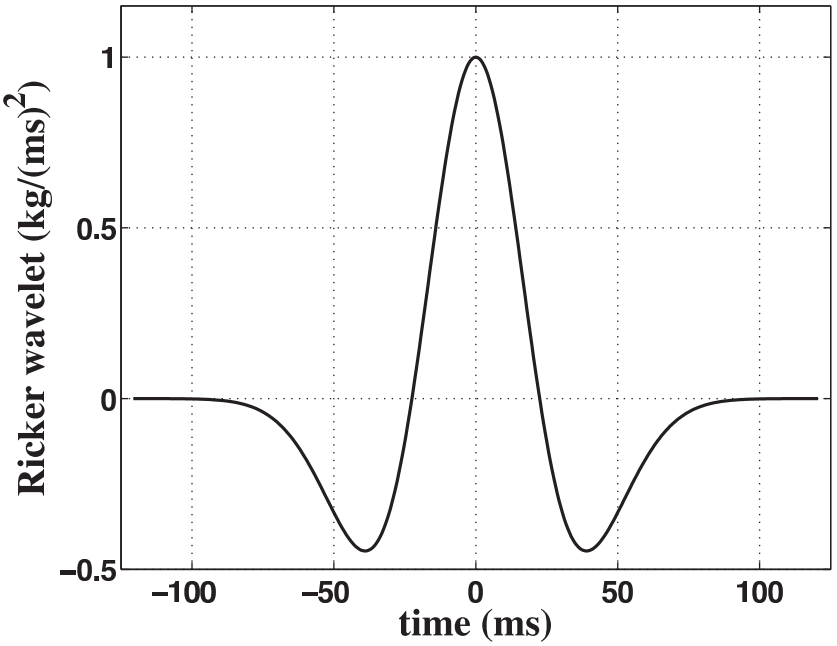


FIG. 2. *Plot of Ricker wavelet h(t) given by* (7.6).

TABLE 7.5

Compressional and shear wave velocities of the layered medium for Experiment 1. *Density is assumed constant at* $\rho = 2 \cdot 10^{12}$ *kg/km*^{3}.

Layer depth (km)	V_p (km/s)	V_s (km/s)
0–1.6	2.5	1.5
1.7–3.8	1.9	1.3
3.9–4.1	3.7	2.0
4.2–5.8	2.7	1.6
5.9–8.1	3.0	1.75
8.2–8.4	1.9	1.3
8.5–10.1	3.3	1.9
10.2–12.0	3.7	2.0

spatial direction. Note that to avoid boundary reflections our source was placed in the center of the domain for these experiments. The time step $\Delta t = 1.56 \cdot 10^{-2}$ seconds is computed according to condition (7.1) with $V_p = 3.7$ km/s and $V_s = 2.0$ km/s, which are the maximum values of wave velocities in these experiments, and the simulations were run for 100 time steps.

In Experiment 1, we consider a layered medium with compressional and shear wave velocities varying as described in Table 7.5. (Figure 3 also shows a plot of compressional wave velocity.) Density in this experiment is assumed to be constant at $2 \cdot 10^{12}$ kg/km^3, and compressional and shear velocities vary only in depth z. Therefore, the wave field is homogeneous in the xy plane and identical in the xz and yz planes. We first show a comparison of full finite element and augmented solutions for the velocity when an experiment is run using a Gaussian source. While a Gaussian may not be a typical seismic source, this experiment clearly illustrates the impact of the input layers on the solution. In this experiment there are 120 fine blocks and

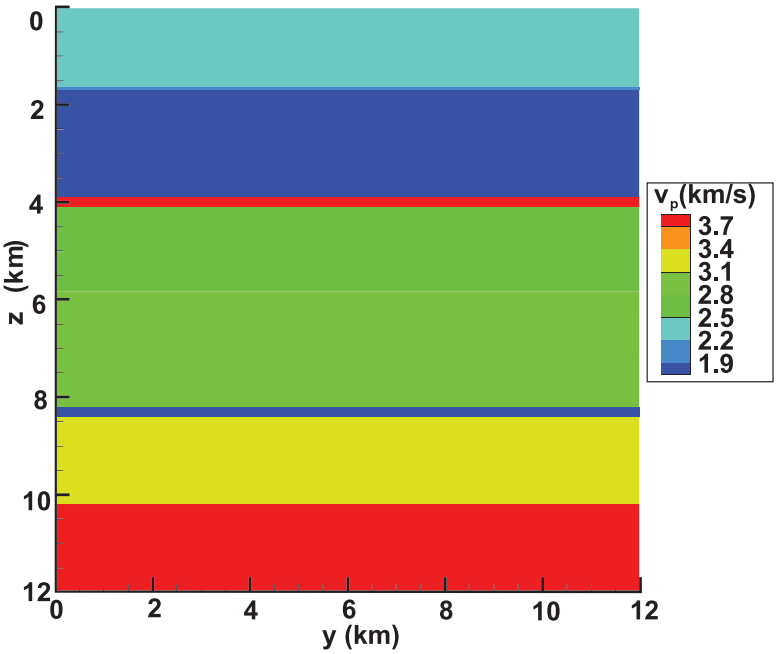
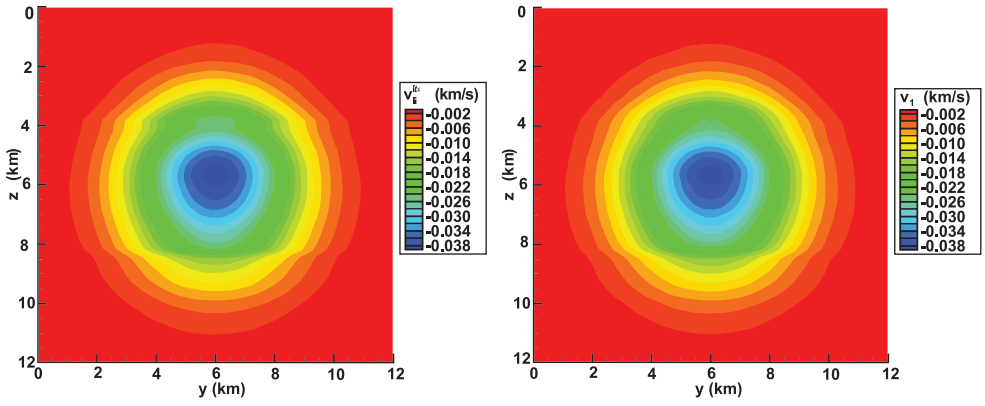


FIG. 3. *The yz plane slice of the compressional wave velocity for experiment* 1. *The domain is discretized into* $24 \times 24 \times 24$ *coarse blocks with* $5 \times 5 \times 5$ *fine blocks per coarse block. The thin velocity layers are approximately half as wide as a coarse grid block.*



(a) (b)

FIG. 4. *Comparison of yz plane slices for Experiment* 1 *with the Gaussian source function. The slice is taken at* 3.7 *km in the x-direction. The numerical grids for upscaling contain* $120 \times 120 \times 120$ *fine-grid blocks and* $24 \times 24 \times 24$ *coarse-grid blocks. The velocity field is shown in Figure* 3. (a) *First component of full finite element velocity solution (km/s).* (b) *First component of the augmented velocity solution (km/s).*

24 coarse blocks in each of the three directions. Figure 4 compares slices of the full finite element velocity solution and the augmented solution for the Gaussian source experiment. In Figures 5 and 6, we compare slices of the full finite element velocity solution and the augmented and coarse solutions for the more realistic Ricker source. We show only the slices in the yz plane.

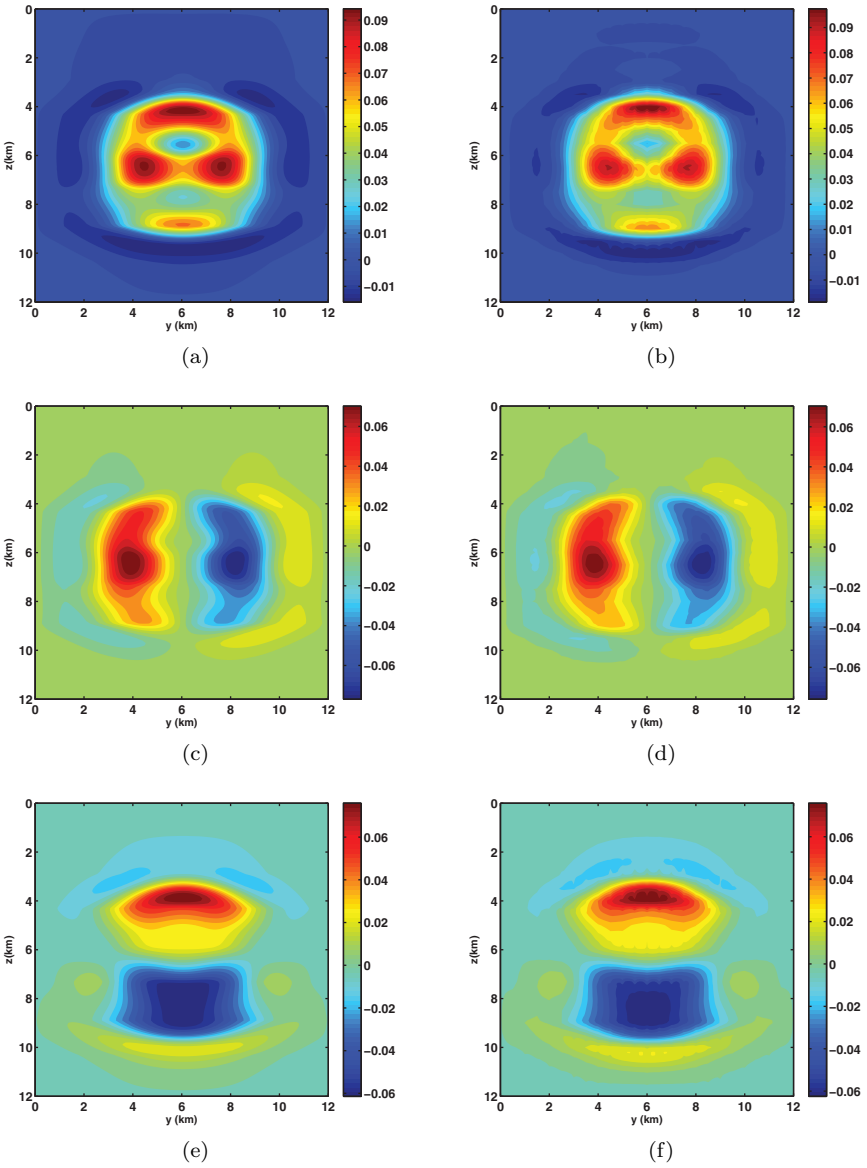


FIG. 5. *Comparison of yz plane slices for Experiment* 1 *with the Ricker wavelet source function. The slice is taken at* 3.7 *km in the x-direction. The numerical grids for upscaling contain* $120 \times 120 \times 120$ *fine grid blocks and* $24 \times 24 \times 24$ *coarse grid blocks. The velocity field is shown in Figure* 3. (a) *First component of full finite element velocity solution (km/s);* (b) *first component of the augmented velocity solution (km/s);* (c) *second component of full finite element velocity solution (km/s);* (d) *second component of the augmented velocity solution (km/s);* (e) *third component of full finite element velocity solution (km/s);* (f) *third component of the augmented velocity solution (km/s).*

We see from Figures 4 and 5 that the augmented solution appears to be in remarkable agreement with the full finite element solution. The coarse solution has a more homogeneous structure than the full finite element solutions (see Figure 6) but still captures the essential fluctuations of the input velocity fields. Note that the

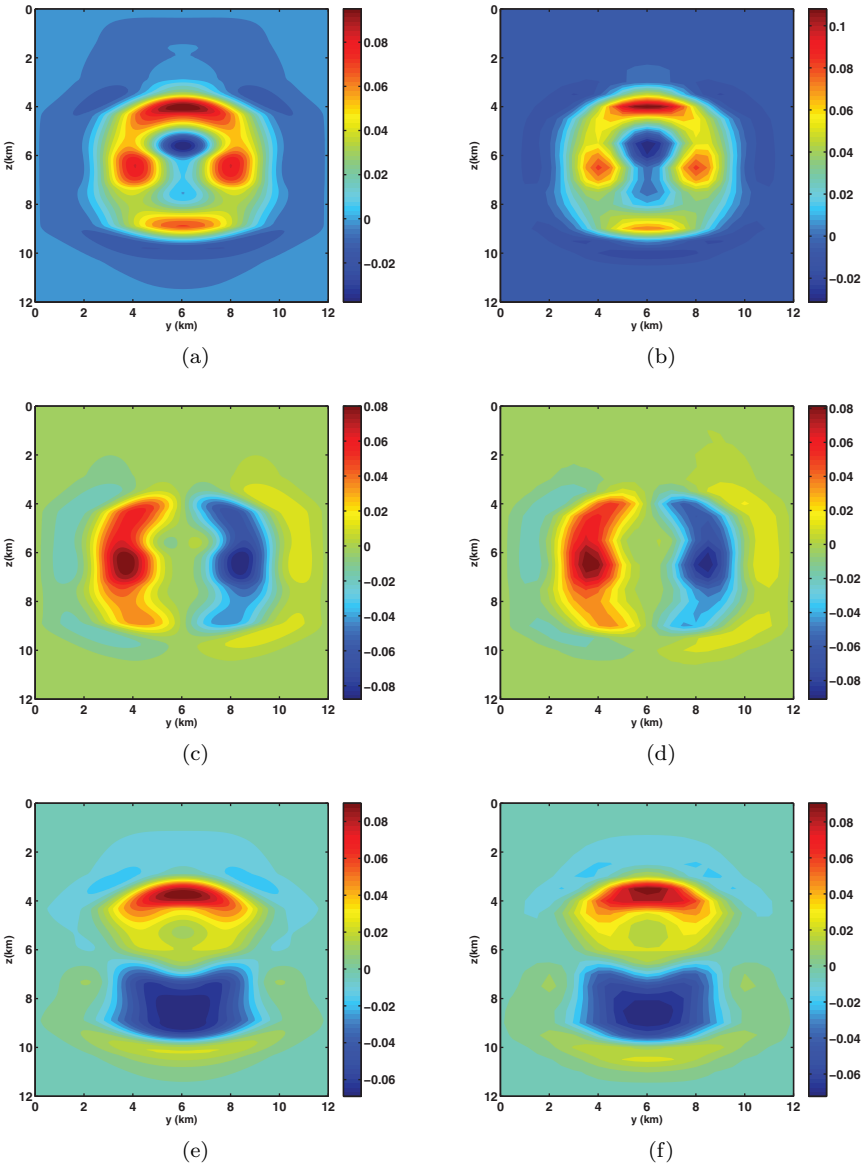


FIG. 6. *Comparison of yz plane slices for Experiment* 1 *with the Ricker wavelet source function. The slice is taken at* 4 *km in the x-direction. The numerical grids for upscaling contain* $120\times120\times120$ *fine-grid blocks and* $24\times24\times24$ *coarse-grid blocks. The velocity field is shown in Figure* 3. (a) *First component of full finite element velocity solution (km/s);* (b) *first component of the coarse velocity solution (km/s);* (c) *second component of full finite element velocity solution (km/s);* (d) *second component of the coarse velocity solution (km/s);* (e) *third component of full finite element velocity solution (km/s);* (f) *third component of the coarse velocity solution (km/s).*

comparisons of coarse and full finite element solutions and augmented and full finite element solutions are shown for different slice locations. We chose to plot the coarse solution on the coarse grid while the augmented solution is plotted on the fine grid.

Figures 4–6 show that the wave front does not have a spherical shape, as it would if the wave propagated in a homogeneous medium. The deformation is especially

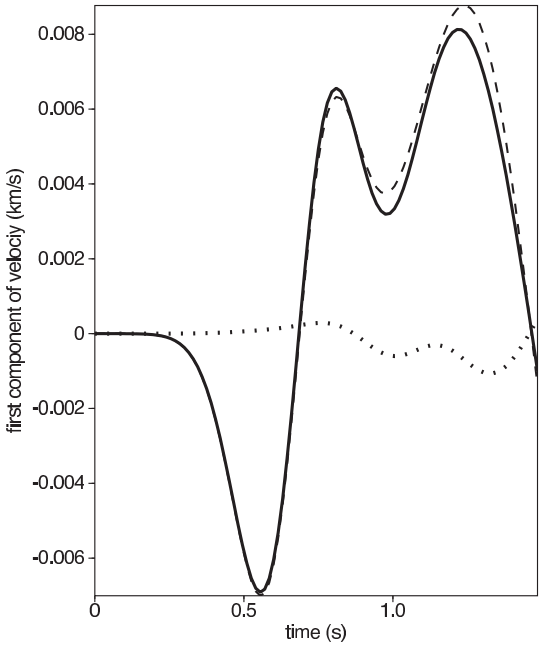


FIG. 7. *Comparison of traces for the first component of velocity for Experiment* 1. *Solid curve is the full finite element solution. Dashed curve is the coarse solution. Dotted curve is the difference between the full finite element and coarse solutions.*

obvious in the lower part of the domain where the wave propagates faster than in the rest of the domain due to the overall gradual increase of the compressional and shear wave velocities with depth (see Figure 4). The wave propagates faster between 3.9 and 4.1 km in the z-direction due to the high-velocity layer at this location in the input velocity field (see Table 7.5 and Figure 3). Although the thickness of the layer (0.3 km) is smaller than the size of the coarse block in the z-direction (0.5 km), both the coarse and augmented solutions capture this variation in the velocity field. The region of low amplitude located in the lower part of the domain reflects the decrease in the velocity field that occurs between 8.2 and 8.4 km in the z-direction.

Figure 7 compares time traces at the arbitrary receiver location $(x, y, z) = (8$ km, 4 km, 6 km) for the first component of velocity. The solid and dashed curves represent the full finite element and coarse solutions, respectively. The dotted curve shows the difference (error) between these two solutions. Figure 7 shows that, as expected, the shape of the Ricker pulse is distorted by the reflections from material interfaces. Comparing the solid and dashed curves, we conclude that the coarse solution is in good agreement with the full finite element solution. In fact, the relative error between the full finite element and coarse solutions is about 1%.

The velocity field used for the next set of experiments varies periodically on the fine scale. We consider a checkerboard medium with cells of size $4 \times 4 \times 4$ fine-grid blocks (see Figure 8). Each cell has a compressional wave velocity of 2.5 or 3.7 km/s and a shear wave velocity of 1.5 or 2.0 km/s. The coarse grid is discretized into $24 \times 24 \times 24$ grid blocks in experiment 2a and $15 \times 15 \times 15$ coarse-grid blocks in experiment 2b. Therefore, in experiment 2a the size of the coarse blocks is close to the period of the medium (the size of each checkerboard cell). In experiment 2b, each coarse block is two times larger than the fine-scale period. Figures 9 shows

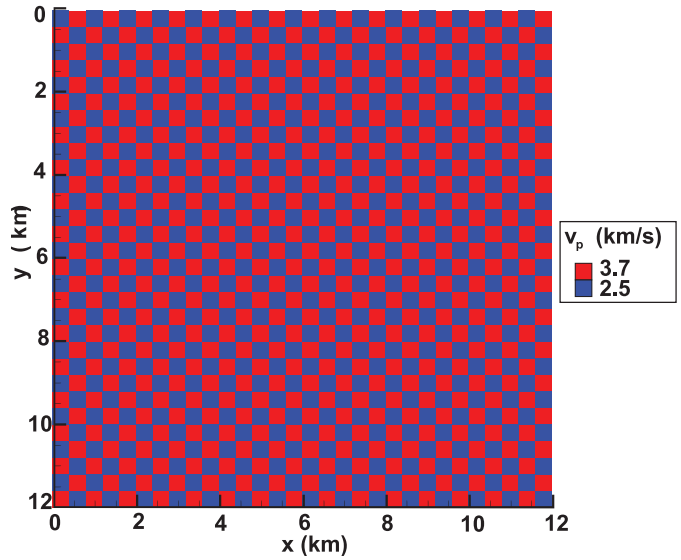
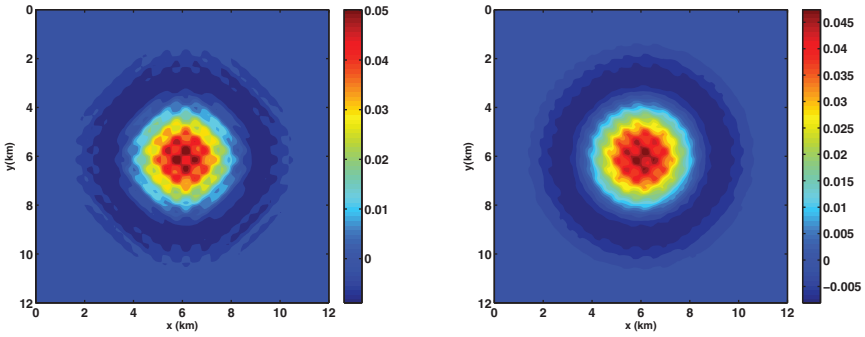


FIG. 8. *A yz slice of the compressional wave velocity for the checkerboard experiments. The domain is discretized into* $24 \times 24 \times 24$ *coarse blocks with* $5 \times 5 \times 5$ *fine blocks in each coarse block in Experiment* 2*a and* $15 \times 15 \times 15$ *coarse blocks with* $8 \times 8 \times 8$ *fine blocks in each coarse block in Experiment* 2*b.*

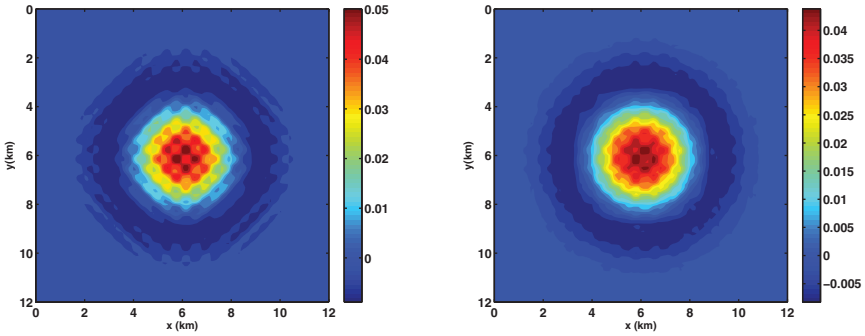


(a) (b)

FIG. 9. *Comparison of xy plane slices for Experiment* 2*a. The slice is taken at* 6.3 *km in the z-direction. The numerical grids for upscaling contain* $120 \times 120 \times 120$ *fine-grid blocks and* $24 \times 24 \times 24$ *coarse-grid blocks. The velocity field is shown in Figure* 8. (a) *Third component of full finite element velocity solution (km/s);* (b) *third component of the augmented velocity solution (km/s).*

the xy slices of the third component of the solutions from experiment 2a. Figure 10 compares the xy slices of the full finite element and augmented solutions from experiment 2b. Figures 9 and 10 show that the augmented upscaled solution captures the checkerboard variations of the wave velocity field.

The velocity field for the final experiment is a single realization from a stochastic distribution which models a two-component mixture of materials that vary based on input correlation lengths in x and y, a roughness parameter (or Hurst number), and a percentage distribution of the two materials. For this experiment, the materials are distributed evenly (a 50/50 mixture), and the correlation lengths were chosen to be 200



(a) (b)

FIG. 10. *Comparison of xy plane slices for Experiment 2b. The slice is taken at 6.3 km in the z-direction. The numerical grids for upscaling contain* $120 \times 120 \times 120$ *fine-grid blocks and* $15 \times 15 \times 15$ *coarse-grid blocks. The velocity field is shown in Figure 8.* (a) *Third component of full finite element velocity solution (km/s);* (b) *third component of the augmented velocity solution (km/s).*

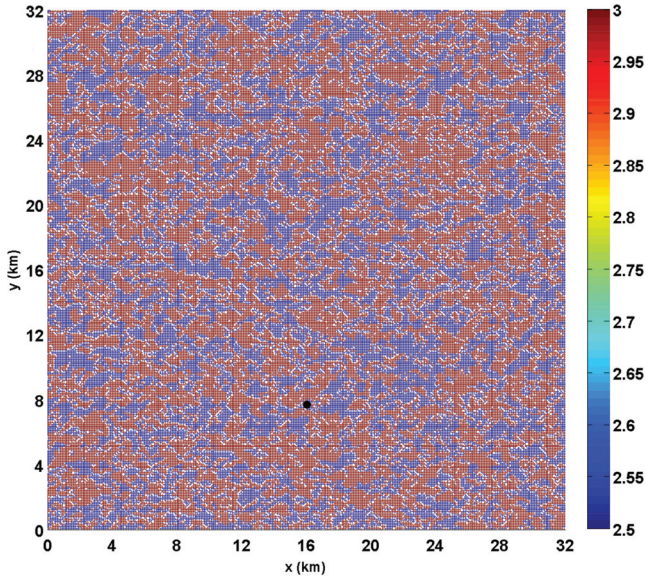
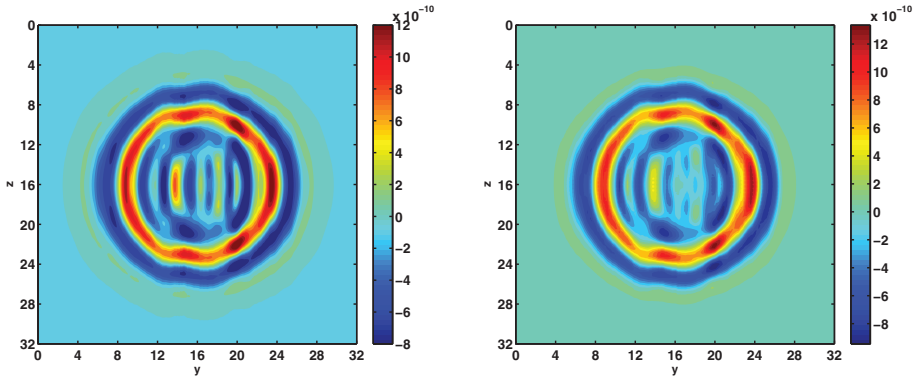


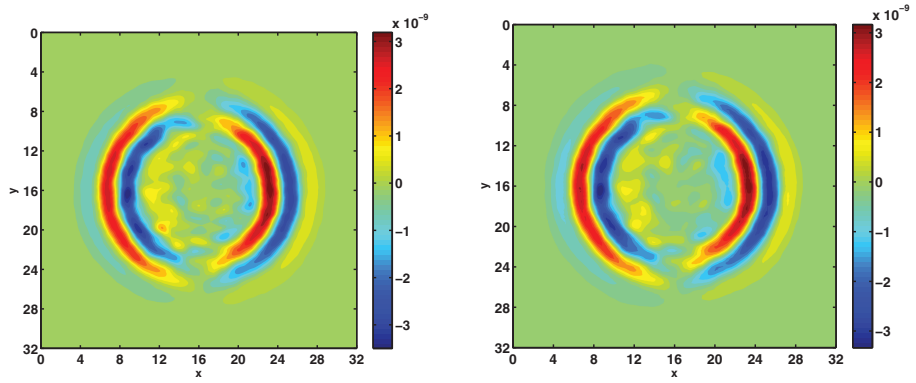
FIG. 11. *The xy plane slice of the compressional wave velocity for experiment 3. The domain is discretized into* $80 \times 80 \times 80$ *coarse blocks with* $4 \times 4 \times 4$ *fine blocks per coarse block. The black circle corresponds to the location of the receivers.*

m in the x-direction and 300 m in the y-direction for each material. The 2D stochastic model is extended to three dimensions by duplication of the layer for each value of z. These correlation lengths guarantee fine-scale heterogeneity below the coarse-grid block size of 0.4 km $\times$ 0.4 km $\times$ 0.4 km. The two materials have compressional velocity values of 2.5 km/s and 3.0 km/s (see Figure 11). The values for shear wave velocity for the two materials are 1.5 and 1.75 km/s, respectively, and the densities are $2.2 \cdot 10^{12}$ and $2.3 \cdot 10^{12}$ kg/km^3. In this experiment, we consider a much larger domain than in the previous experiments. The domain is 32 km $\times$ 32 km $\times$ 32 km, and we ran the simulation for 3.85 s with $\Delta t = 1.92 \cdot 10^{-2}$ s. As before, the wavelength is



(a) (b)

FIG. 12. *Comparison of yz plane slices for Experiment* 3. *The slice is taken at* 18.1 *km in the x-direction. The numerical grids for upscaling contain* $320 \times 320 \times 320$ *fine-grid blocks and* $80 \times 80 \times 80$ *coarse-grid blocks. The velocity field is shown in Figure* 11. (a) *First component of full finite element velocity solution (km/s);* (b) *first component of the augmented velocity solution (km/s).*



(a) (b)

FIG. 13. *Comparison of xy plane slices for Experiment* 3. *The slice is taken at* 18.1 *km in the z-direction. The numerical grids for upscaling contain* $320 \times 320 \times 320$ *fine-grid blocks and* $80 \times 80 \times 80$ *coarse-grid blocks. The velocity field is shown in Figure* 11. (a) *First component of full finite element velocity solution (km/s);* (b) *first component of the augmented velocity solution (km/s).*

approximately 1 km, so the total propagation distance is approximately 7 wavelengths if the source is placed in the center of the domain. As in the previous experiments, the source is a Ricker wavelet in time.

Figures 12 and 13 show slices of the full finite element and augmented solutions for the first component of velocity. We see that the upscaled solution captures many of the essential features of the heterogeneous medium.

Figure 14 compares time traces at receiver locations $(x, y, z) = (16$ km, 8 km, 12 km) and $(x, y, z) = (16$ km, 8 km, 18 km). The black circle in Figure 11 indicates the location of receivers in the xy plane. The solid and dashed curves show the full finite element and coarse solutions, respectively, corresponding to the random medium

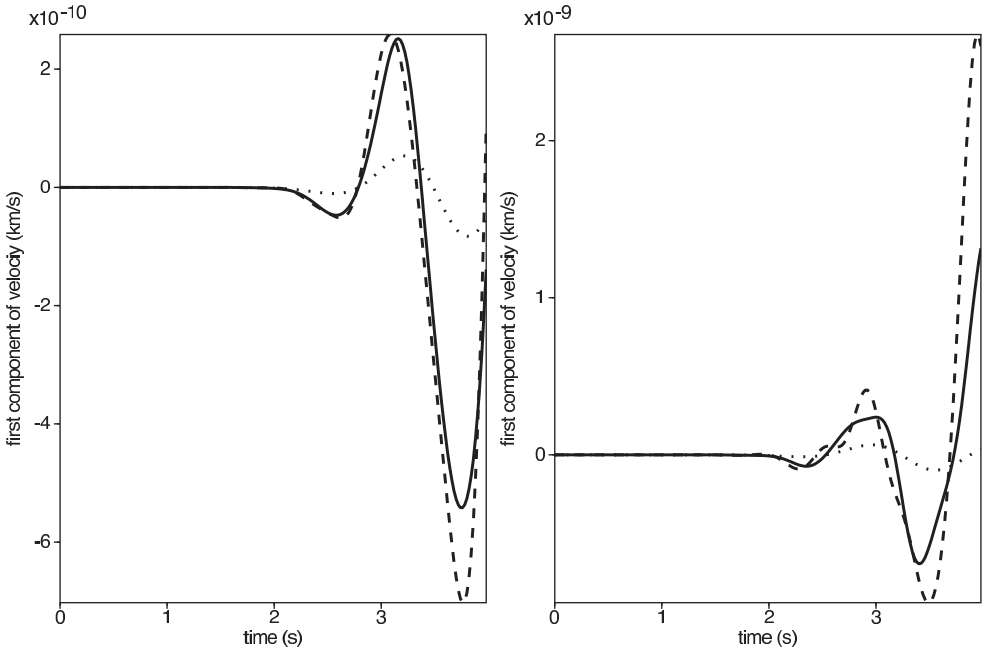


FIG. 14. *Comparison of time traces for the first component of velocity for Experiment* 3. *The solid curve is the full finite element solution. The dashed curve is the coarse solution. The dotted curve is the full finite element solution for a homogeneous medium with a single (average) value for each of the three input parameters.* (a) *Receiver location is* $(x, y, z) = (16\ km, 8\ km, 12\ km)$; (b) *receiver location is* $(x, y, z) = (16\ km, 8\ km, 18\ km)$.

shown in Figure 11. The dotted curve is the full finite element solution for an experiment with a single value for each of the three input parameters (determined by taking an average of the parameters for the two materials). Comparing traces in 14, we see that while the full finite element and coarse solutions may have different amplitudes, the coarse solution does a good job of locating interfaces between materials (events) and would therefore be of use as part of an inversion scheme.

8. Parallel implementation and performance. The parallel implementation of the elastic upscaling method is based on two facts about the upscaling algorithm: (1) the subgrid stage of the algorithm is embarrassingly parallel, and (2) the coarse load vector calculation and the subgrid solve dominate the time step. Table 8.1 gives timings for the time-step loop of the serial elastic upscaling code. In each row of the table a different test is performed. Specifically, the number of coarse-grid blocks is doubled in each spatial direction, while the fine grid remains fixed at $80 \times 80 \times 80$ blocks. The first column of Table 8.1 shows the number of coarse blocks in the domain. Column two gives the total time taken by the time-step loop and is the sum of the times shown in columns three (subgrid problems) and four (coarse problem). The time taken to solve the coarse problem is further decomposed into times for the linear solve and load vector calculation shown in columns five and six, respectively. As we see from Table 8.1, the time step is dominated by the coarse solve, with the time for the coarse solve being taken almost entirely by load vector calculations for this experiment.

TABLE 8.1
Observed timings in seconds of a single time step of the elastic upscaling code for varying numbers of coarse blocks. The fine grid is $80 \times 80 \times 80$ *blocks.*

Number of coarse blocks	Time-step loop	Subgrid problems	Coarse problem	Linear solve	Load vector computation
1	5.644	1.066	4.578	0.000	4.578
2^3	5.962	1.018	4.943	0.000	4.943
4^3	6.654	1.013	5.642	0.000	5.641
5^3	7.711	1.043	6.218	0.001	6.217
10^3	9.989	1.105	8.885	0.017	8.867
20^3	17.92	1.329	16.591	0.431	16.16

In order to complete the load vector calculation at each time step, we must compute and assemble six load vectors, one for each of the three components of displacement and the three components of velocity. Computing a single entry of the load vector for any component of velocity requires the evaluation of ten inner products. Displacement components are slightly less expensive and require three inner products only. Still, each of these inner products is a triple integral. Thus, for a coarse grid that consists of 15 coarse blocks in each direction (or a $16 \times 16 \times 16$ coarse grid), we must evaluate $16^3 \cdot (3 \cdot 10 + 3 \cdot 3) = 159{,}744$ triple integrals at each time step. The great cost associated with the load vector computation suggests that if, in addition, to parallelizing subgrid problems we parallelize the coarse problem, the resulting algorithm will be more efficient. Therefore, we parallelize the computation of the coarse-grid load vector and use the sparse linear solver `SuperLU_DIST` [24] to solve the coarse-grid linear system in parallel.

Parallelizing the subgrid problems is straightforward. Because the subgrid problems are independent of one another, they can be parceled out to the available processors and solved separately without requiring communication. Computation of the load vectors is also essentially embarrassingly parallel. Calculating an entry of a load vector requires the evaluation of several inner products over the support of the coarse basis function associated with the load vector entry. Recall from section 6 that the support of each coarse basis function consists of eight coarse blocks. Parallelizing the load vector calculation relies on the observation that coarse blocks in the support of a basis function might be stored across multiple processors, but the inner products can be evaluated over the blocks stored on each processor separately and then summed. Therefore, the only communication in the load vector computation occurs when the pieces of the local vectors are assembled via `MPI_Allreduce`.

In addition to parallelizing the subgrid solve and load vector calculation, we also parallelize I/O. Because our parallel algorithm is based on calculations done over a single coarse block, our input algorithm splits coarse blocks as evenly as possible among processors. To avoid memory restrictions, we distribute the coarse blocks by reading one xy slice of the input fields at a time (see Minkoff [25]). The layer is then broadcast to each process, and each processor keeps the portion of the layer corresponding to its set of coarse blocks and discards the remainder. Output slices of the augmented and coarse solutions are handled similarly. In terms of memory usage, parallelizing I/O allows us to avoid storing the global fine-grid unknowns on a single process at any point of the algorithm.

Upscaling does not require storage of ghost cell information. Therefore, the subgrid stage of the upscaling algorithm consumes less memory than standard domain decomposition techniques. Unfortunately we do need to store a coarse-grid mass matrix and the coarse solution in addition to the subgrid solution. Although the

TABLE 8.2

Observed timings in seconds of a single time step of the parallel elastic upscaling code for different number of processes. The numerical grids consist of $320 \times 320 \times 320$ *fine blocks and* $16 \times 16 \times 16$ *coarse blocks.*

Number of processes	Time-step loop	Subgrid problems	Coarse problem	Linear solve	Load vector computation
1	179.36	15.25	163.70	0.43	163.27
2	89.23	7.27	81.79	0.26	81.53
4	45.98	3.64	42.25	0.20	42.02
8	22.59	1.81	20.74	0.15	20.59
16	11.32	0.91	10.40	0.14	10.26
32	5.72	0.46	5.25	0.12	5.13
64	2.98	0.23	2.74	0.12	2.62
128	1.54	0.12	1.41	0.12	1.29

TABLE 8.3

Observed timings in seconds of a single time step of the parallel elastic upscaling code for different number of processes. The numerical grids consist of $320 \times 320 \times 320$ *fine blocks and* $32 \times 32 \times 32$ *coarse blocks.*

Number of processes	Time-step loop	Subgrid problems	Coarse problem	Linear solve	Load vector computation
1	238.07	17.00	220.23	8.30	211.92
2	119.41	8.39	110.63	4.58	106.05
4	60.03	4.22	55.62	2.62	53.00
8	30.61	2.23	28.28	1.70	26.58
16	17.07	1.00	16.02	2.75	13.27
32	8.17	0.51	7.63	0.95	6.68
64	4.51	0.25	4.24	0.88	3.36
128	2.88	0.13	2.74	0.96	1.78

coarse-grid problem is significantly smaller than the full fine-scale problem, these additional memory requirements are very likely to make the upscaling algorithm as expensive in terms of memory as standard domain decomposition methods. However, if the upscaled coarse solution is used as the forward solution for an iterative inversion scheme, it is likely that less memory would be required to solve the inverse problem.

Tables 8.2 and 8.3 summarize the observed timing results (in seconds) for the parallel elastic upscaling code for a problem of size $64 \times 64 \times 64$ km discretized into $320 \times 320 \times 320$ fine grid blocks. The serial full finite element code takes 140.30 s to run a single time step of this problem. Table 8.2 shows fixed speedup timing results for the coarse grid of size $16 \times 16 \times 16$ blocks, and Table 8.3 shows timing results for the coarse grid of size $32 \times 32 \times 32$ blocks. Table 8.4 illustrates scaled speedup. Scaled speedup requires us to double the size of the problem every time the number of processes is doubled. Since the ratio between problem size and the number of processes remains constant, the optimal scaled speedup does not change.

We can see from Tables 8.2, 8.3, and 8.4 that even on a single process the cost of operator upscaling is comparable to the cost of the full finite element algorithm. As expected, the total time is dominated by the coarse problem. The time required to solve the subgrid problems and assemble the load vectors is cut in half when we double the number of processes. The time required to solve the linear system does not decrease after 32 processes. Better speedup of the linear system solve might be achievable with a different parallel solver. Luckily the linear solver is relatively cheap unless the number of coarse blocks is very large (i.e., not much upscaling is done).

TABLE 8.4

Scaled speedup timings in seconds of a single time step of the parallel elastic upscaling code for different number of processes. Each node solves a problem of size $80 \times 80 \times 80$ *fine grid blocks. The largest problem is size* $320 \times 320 \times 320$ *fine grid blocks. Each coarse block consists of* $4 \times 4 \times 4$ *fine grid blocks.*

Number of processes	Time-step loop	Subgrid problems	Coarse problem	Linear solve	Load vector computation
1	6.28	0.34	5.92	0.43	5.494
2	6.40	0.33	6.06	0.58	5.479
4	7.03	0.35	6.66	1.08	5.575
8	7.81	0.33	7.46	1.92	5.540
16	8.74	0.33	8.39	2.79	5.600
32	10.42	0.34	10.06	4.35	5.712
64	14.50	0.34	14.14	8.08	6.050

When the coarse problem is larger, as in the experiment presented in Table 8.4, the time required to solve the coarse-grid linear system becomes more significant. The second column of Table 8.4 shows that speedup decreases when the problem is solved on more than 16 processes. Columns three and six indicate that the time required to solve the subgrid problems and construct load vectors remains essentially the same for any number of processes, as expected for scaled speedup. The linear solver, on the other hand, takes more time whenever we increase the number of processes.

9. Conclusion. In this paper we present the first operator upscaling algorithm for the 3D elastic wave equation. We upscale all six unknowns (three velocities and three displacements). We have shown that both the coarse, and reconstructed fine-scale solutions capture some of the fluctuations of the input velocity fields even if these fluctuations occur on a scale smaller than a single coarse block. Most notable for inversion studies, the locations of "reflectors" (especially evident for the layered medium experiment) are well captured by the upscaled solution.

Experimental studies in a homogeneous medium indicate that the operator upscaling algorithm implemented using linear basis functions exhibits second-order convergence in both space and time. While one might expect this result for standard finite element methods, it is less intuitive in the case of upscaling due to the zero boundary conditions imposed on coarse block edges in the subgrid stage of the algorithm. For the algorithm to exhibit this convergence, the coarse grid must be refined. We describe both serial and parallel implementations of the method. The subgrid problems are embarrassingly parallel due to the choice of boundary conditions imposed on each coarse block in the subgrid solve. In addition, we have shown that the local subgrid equations are explicit. The coarse-grid system matrix has a sparse and banded structure and is independent of time and subgrid solutions. Therefore, the matrix can be assembled outside of the time-step loop and its setup is cheap in terms of both memory and computational effort. The load vectors, on the other hand, have to be updated at each time step. The computation of the load vectors requires approximation of multiple inner products with input parameters defined on the fine grid. The load vector calculation is therefore the most expensive part of the algorithm. Once again, however, parallelization of this calculation is easily distributed over the support of each coarse basis function without need for additional storage, and the resulting algorithm exhibits optimal speedup for both the load vector calculation and the subgrid solve.

Other implementations of the method for elasticity are possible. Specifically a mixed formulation of the spectral element method discussed by Cohen et al. in [13, 12]

would use H^1-L^2 variational spaces and the nonclassical transformation of the elastic wave equation into a first-order system for displacement and vector-valued variables γ. In the full fine-scale version of the method, the vector-valued variables γ have local support on each grid element and are discontinuous from one element to another, which creates a natural framework for upscaling. In addition, this mixed formulation of the method ensures conservation of energy on both the fine and coarse scales and allows for the implementation of perfectly matched layers for modeling unbounded domains.

Appendix A. In this appendix, we provide explicit difference equations that correspond to the subgrid problem. We approximate the inner products in the subgrid equations by using first-order Gauss–Lobatto quadrature. In section 5, we illustrate the procedure by evaluating a single inner product on the right-hand side of (3.2) for the first component of velocity. Applying the same approach to the rest of the inner products, we obtain the following system of equations:

$$
\begin{aligned}
\rho_{i,j,k}\frac{(v_1)^{n+1}_{i,j,k}-(v_1)^{n}_{i,j,k}}{\Delta t} = {} & \frac{1}{2\Delta x^2}\Big((\lambda+2\mu)_{i-1,j,k}\Big[(u_1)^{n+\frac12}_{i-1,j,k}-(u_1)^{n+\frac12}_{i,j,k}\Big] \\
& +(\lambda+2\mu)_{i,j,k}\Big[(u_1)^{n+\frac12}_{i-1,j,k}-2(u_1)^{n+\frac12}_{i,j,k}+(u_1)^{n+\frac12}_{i+1,j,k}\Big] \\
& +(\lambda+2\mu)_{i+1,j,k}\Big[(u_1)^{n+\frac12}_{i+1,j,k}-(u_1)^{n+\frac12}_{i,j,k}\Big]\Big) \\
& +\frac{1}{4\Delta x\Delta y}\Big(\lambda_{i-1,j,k}\Big[(u_2)^{n+\frac12}_{i-1,j-1,k}-(u_2)^{n+\frac12}_{i-1,j+1,k}\Big] \\
& +\lambda_{i+1,j,k}\Big[(u_2)^{n+\frac12}_{i+1,j+1,k}-(u_2)^{n+\frac12}_{i+1,j-1,k}\Big]\Big) \\
& +\frac{1}{4\Delta x\Delta z}\Big(\lambda_{i-1,j,k}\Big[(u_3)^{n+\frac12}_{i-1,j,k-1}-(u_3)^{n+\frac12}_{i-1,j,k+1}\Big] \\
& +\lambda_{i+1,j,k}\Big[(u_3)^{n+\frac12}_{i+1,j,k+1}-(u_3)^{n+\frac12}_{i+1,j,k-1}\Big]\Big) \\
& +\frac{1}{2\Delta y^2}\Big(\mu_{i,j-1,k}\Big[(u_1)^{n+\frac12}_{i,j-1,k}-(u_1)^{n+\frac12}_{i,j,k}\Big] \\
& +\mu_{i,j,k}\Big[(u_1)^{n+\frac12}_{i,j-1,k}-2(u_1)^{n+\frac12}_{i,j,k}+(u_1)^{n+\frac12}_{i,j+1,k}\Big] \\
& +\mu_{i,j+1,k}\Big[(u_1)^{n+\frac12}_{i,j+1,k}-(u_1)^{n+\frac12}_{i,j,k}\Big]\Big) \\
& +\frac{1}{4\Delta x\Delta y}\Big(\mu_{i,j-1,k}\Big[(u_2)^{n+\frac12}_{i-1,j-1,k}-(u_2)^{n+\frac12}_{i+1,j-1,k}\Big] \\
& +\mu_{i,j+1,k}\Big[(u_2)^{n+\frac12}_{i+1,j+1,k}-(u_2)^{n+\frac12}_{i-1,j+1,k}\Big]\Big) \\
& +\frac{1}{2\Delta z^2}\Big(\mu_{i,j,k-1}\Big[(u_1)^{n+\frac12}_{i,j,k-1}-(u_1)^{n+\frac12}_{i,j,k}\Big] \\
& +\mu_{i,j,k}\Big[(u_1)^{n+\frac12}_{i,j,k-1}-2(u_1)^{n+\frac12}_{i,j,k}+(u_1)^{n+\frac12}_{i,j,k+1}\Big] \\
& +\mu_{i,j,k+1}\Big[(u_1)^{n+\frac12}_{i,j,k+1}-(u_1)^{n+\frac12}_{i,j,k}\Big]\Big) \\
& +\frac{1}{4\Delta x\Delta z}\Big(\mu_{i,j,k-1}\Big[(u_3)^{n+\frac12}_{i-1,j,k-1}-(u_3)^{n+\frac12}_{i+1,j,k-1}\Big] \\
& +\mu_{ij,k+1}\Big[(u_3)^{n+\frac12}_{i+1,j,k+1}-(u_3)^{n+\frac12}_{i-1,j,k+1}\Big]\Big)+(f_1)^{n+\frac12}_{i,j,k},
\end{aligned}
\tag{A.1}
$$

$$
\begin{aligned}
\rho_{i,j,k}\frac{(v_2)^{n+1}_{i,j,k}-(v_2)^{n}_{i,j,k}}{\Delta t} &= \frac{1}{4\Delta x\Delta y}\Big(\mu_{i-1,j,k}\left[(u_1)^{n+\frac{1}{2}}_{i-1,j-1,k}-(u_1)^{n+\frac{1}{2}}_{i-1,j+1,k}\right]\\
&+\mu_{i+1,j,k}\left[(u_1)^{n+\frac{1}{2}}_{i+1,j+1,k}-(u_1)^{n+\frac{1}{2}}_{i+1,j-1,k}\right]\Big)\\
&+\frac{1}{2\Delta x^2}\Big(\mu_{i-1,j,k}\left[(u_2)^{n+\frac{1}{2}}_{i-1,j,k}-(u_2)^{n+\frac{1}{2}}_{i,j,k}\right]\\
&+\mu_{i,j,k}\left[(u_2)^{n+\frac{1}{2}}_{i-1,j,k}-2(u_2)^{n+\frac{1}{2}}_{i,j,k}+(u_2)^{n+\frac{1}{2}}_{i+1,j,k}\right]\\
&+\mu_{i+1,j,k}\left[(u_2)^{n+\frac{1}{2}}_{i+1,j,k}-(u_2)^{n+\frac{1}{2}}_{i,j,k}\right]\Big)\\
&+\frac{1}{4\Delta x\Delta y}\Big(\lambda_{i,j-1,k}\left[(u_1)^{n+\frac{1}{2}}_{i-1,j-1,k}-(u_1)^{n+\frac{1}{2}}_{i+1,j-1,k}\right]\\
&+\lambda_{i,j+1,k}\left[(u_1)^{n+\frac{1}{2}}_{i+1,j+1,k}-(u_1)^{n+\frac{1}{2}}_{i-1,j+1,k}\right]\Big)\\
&+\frac{1}{2\Delta y^2}\Big((\lambda+2\mu)_{i,j-1,k}\left[(u_2)^{n+\frac{1}{2}}_{i,j-1,k}-(u_2)^{n+\frac{1}{2}}_{i,j,k}\right]\\
&+(\lambda+2\mu)_{i,j,k}\left[(u_2)^{n+\frac{1}{2}}_{i,j-1,k}-2(u_2)^{n+\frac{1}{2}}_{i,j,k}+(u_2)^{n+\frac{1}{2}}_{i,j+1,k}\right]\\
&+(\lambda+2\mu)_{i,j+1,k}\left[(u_2)^{n+\frac{1}{2}}_{i,j+1,k}-(u_2)^{n+\frac{1}{2}}_{i,j,k}\right]\Big)\\
&+\frac{1}{4\Delta z\Delta y}\Big(\lambda_{i,j-1,k}\left[(u_3)^{n+\frac{1}{2}}_{i,j-1,k-1}-(u_3)^{n+\frac{1}{2}}_{i,j-1,k+1}\right]\\
&+\lambda_{i,j+1,k}\left[(u_3)^{n+\frac{1}{2}}_{i,j+1,k+1}-(u_3)^{n+\frac{1}{2}}_{i,j+1,k-1}\right]\Big)\\
&+\frac{1}{2\Delta z^2}\Big(\mu_{i,j,k-1}\left[(u_2)^{n+\frac{1}{2}}_{i,j,k-1}-(u_2)^{n+\frac{1}{2}}_{i,j,k}\right]\\
&+\mu_{i,j,k}\left[(u_2)^{n+\frac{1}{2}}_{i,j,k-1}-2(u_2)^{n+\frac{1}{2}}_{i,j,k}+(u_2)^{n+\frac{1}{2}}_{i,j,k+1}\right]\\
&+\mu_{i,j,k+1}\left[(u_2)^{n+\frac{1}{2}}_{i,j,k+1}-(u_2)^{n+\frac{1}{2}}_{i,j,k}\right]\Big)\\
&+\frac{1}{4\Delta z\Delta y}\Big(\mu_{i,j,k-1}\left[(u_3)^{n+\frac{1}{2}}_{i,j-1,k-1}-(u_3)^{n+\frac{1}{2}}_{i,j+1,k-1}\right]\\
&+\mu_{i,j,k+1}\left[(u_3)^{n+\frac{1}{2}}_{i,j+1,k+1}-(u_3)^{n+\frac{1}{2}}_{i,j-1,k+1}\right]\Big)+(f_2)^{n+\frac{1}{2}}_{i,j,k},\\
\rho_{i,j,k}\frac{(v_3)^{n+1}_{i,j,k}-(v_3)^{n}_{i,j,k}}{\Delta t} &= \frac{1}{4\Delta z\Delta x}\Big(\mu_{i-1,j,k}\left[(u_1)^{n+\frac{1}{2}}_{i-1,j,k-1}-(u_1)^{n+\frac{1}{2}}_{i-1,j,k+1}\right]\\
&+\mu_{i+1,j}\left[(u_1)^{n+\frac{1}{2}}_{i+1,j,k+1}-(u_1)^{n+\frac{1}{2}}_{i+1,j,k-1}\right]\Big)\\
&+\frac{1}{2\Delta x^2}\Big(\mu_{i-1,j,k}\left[(u_3)^{n+\frac{1}{2}}_{i-1,j,k}-(u_3)^{n+\frac{1}{2}}_{i,j,k}\right]\\
&+\mu_{i,j,k}\left[(u_3)^{n+\frac{1}{2}}_{i-1,j,k}-2(u_3)^{n+\frac{1}{2}}_{i,j,k}+(u_3)^{n+\frac{1}{2}}_{i+1,j,k}\right]\\
&+\mu_{i+1,j,k}\left[(u_3)^{n+\frac{1}{2}}_{i+1,j,k}-(u_3)^{n+\frac{1}{2}}_{i,j,k}\right]\Big)\\
&+\frac{1}{2\Delta y^2}\Big(\mu_{i,j-1,k}\left[(u_3)^{n+\frac{1}{2}}_{i,j-1,k}-(u_3)^{n+\frac{1}{2}}_{i,j,k}\right]\\
&+\mu_{i,j,k}\left[(u_3)^{n+\frac{1}{2}}_{i,j-1,k}-2(u_3)^{n+\frac{1}{2}}_{i,j,k}+(u_3)^{n+\frac{1}{2}}_{i,j+1,k}\right]\\
&+\mu_{i,j+1,k}\left[(u_3)^{n+\frac{1}{2}}_{i,j+1,k}-(u_3)^{n+\frac{1}{2}}_{i,j,k}\right]\Big)
\end{aligned}
\tag{A.2}
$$

$$
\begin{aligned}
&+\frac{1}{4\Delta z\Delta y}\left(\mu_{i,j-1,k}\left[(u_2)^{n+\frac{1}{2}}_{i,j-1,k-1}-(u_2)^{n+\frac{1}{2}}_{i,j-1,k+1}\right]\right.\\
&\left.+\,\mu_{i,j+1,k}\left[(u_2)^{n+\frac{1}{2}}_{i,j+1,k+1}-(u_2)^{n+\frac{1}{2}}_{i,j+1,k-1}\right]\right)\\
&+\frac{1}{4\Delta x\Delta z}\left(\lambda_{i,j,k-1}\left[(u_1)^{n+\frac{1}{2}}_{i-1,j,k-1}-(u_1)^{n+\frac{1}{2}}_{i+1,j,k-1}\right]\right.\\
&\left.+\,\lambda_{i,j,k+1}\left[(u_1)^{n+\frac{1}{2}}_{i+1,j,k+1}-(u_1)^{n+\frac{1}{2}}_{i-1,j,k+1}\right]\right)\\
&+\frac{1}{4\Delta y\Delta z}\left(\lambda_{i,j,k-1}\left[(u_2)^{n+\frac{1}{2}}_{i,j-1,k-1}-(u_2)^{n+\frac{1}{2}}_{i,j+1,k-1}\right]\right.\\
&\left.+\,\lambda_{i,j,k+1}\left[(u_2)^{n+\frac{1}{2}}_{i,j+1,k+1}-(u_2)^{n+\frac{1}{2}}_{i,j-1,k+1}\right]\right)\\
&+\frac{1}{2\Delta z^2}\left((\lambda+2\mu)_{i,j,k-1}\left[(u_3)^{n+\frac{1}{2}}_{i,j,k-1}-(u_3)^{n+\frac{1}{2}}_{i,j,k}\right]\right.\\
&+(\lambda+2\mu)_{i,j,k}\left[(u_3)^{n+\frac{1}{2}}_{i,j,k-1}-2(u_3)^{n+\frac{1}{2}}_{i,j,k}+(u_3)^{n+\frac{1}{2}}_{i,j,k+1}\right]\\
&\left.+\,(\lambda+2\mu)_{i,j,k+1}\left[(u_3)^{n+\frac{1}{2}}_{i,j,k+1}-(u_3)^{n+\frac{1}{2}}_{i,j,k}\right]\right)+(f_3)^{n+\frac{1}{2}}_{i,j,k},
\end{aligned}
\tag{A.3}
$$

$$
\rho_{i,j,k}\frac{(u_q)^{n+\frac{1}{2}}_{i,j,k}-(u_q)^{n-\frac{1}{2}}_{i,j,k}}{\Delta t}=\rho_{i,j,k}(v_q)^n_{i,j,k},
\tag{A.4}
$$

where $q = 1, 2, 3$ and i, j, k denote local indices that range over all the subgrid nodes.

Acknowledgments. We thank David Aldridge in the Geophysical Technology Department at Sandia National Labs for providing us with information on typical seismic sources used in elastic wave modeling. Conversations with Bill Symes were helpful, especially on wave propagation in heterogeneous media. We are grateful to John Zweck for reading an early draft of the manuscript. Finally, we thank the reviewers for many helpful comments which improved the paper.

REFERENCES

[1] J. Achenbach, *Wave Propagation in Elastic Solids*, North Holland, New York, 1984.
[2] G. Allaire and R. Brizzi, *A multiscale finite element method for numerical homogenization*, SIAM J. Multiscale Model. Simul., 4 (2005), pp. 790–812.
[3] T. Arbogast and S. Bryant, *A two-scale numerical subgrid technique for waterflood simulations*, SPE J., 27 (2002), pp. 446–457.
[4] T. Arbogast, S. Minkoff, and P. Keenan, *An operator-based approach to upscaling the pressure equation*, in Computational Methods in Water Resources XII, V. Burganos, G. Karatzas, A. Payatakes, C. Brebbia, W. Gray, and G. Pinder, eds., Computational Mechanics Publications, Southhampton, UK, 1998, pp. 405–412.
[5] T. Arbogast, *Numerical Subgrid Upscaling of Two-Phase Flow in Porous Media*, Lecture Notes in Physics 552, Springer-Verlag, Berlin, 2000.
[6] T. Arbogast, *Implementation of a locally conservative numerical subgrid upscaling scheme for two-phase flow*, Comput. Geosci., 6 (2002), pp. 453–48.
[7] T. Arbogast, *Analysis of a two-scale, locally conservative subgrid upscaling for elliptic problems*, SIAM J. Numer. Anal., 42 (2004), pp. 576–598.
[8] A. Bedford and D. Drumheller, *Introduction to Elastic Wave Propagation*, John Wiley & Sons Ltd., Chichester, England, 1994.
[9] A. Bensoussan, J. Lions, and G. Papanicolaou, *Asymptotic Analysis for Periodic Structure*, North Holland, New York, 1979.
[10] Z. Chen and T. Hou, *A mixed multiscale finite element method for elliptic problems with oscillating coefficients*, Math. Comp., 72 (2002), pp. 541–576.
[11] P. Ciarlet, *The Finite Element Method for Elliptic Problems*, SIAM, New York, 2002.
[12] G. Cohen and S. Fauqueux, *Mixed spectral finite elements for the linear elasticity system in unbound domains*, SIAM J. Sci. Comput., 26 (2005), pp. 864–884.

[13] G. Cohen, *Higher-Order Numerical Methods for Transient Wave Equations*, Springer-Verlag, New York, 2002.
[14] W. E, B. Engquist, and Z. Huang, *Heterogeneous multiscale method: A general methodology for multiscale modeling*, Phys. Rev., 67 (2003), pp. 092101–1–092101–4.
[15] J. Goff, K. Holliger, and A. Levander, *Modal fields: A new method for characterization of random seismic velocity heterogeneity*, Geophys. Res. Lett., 21 (1994), pp. 493–496.
[16] J. Goff and T. Jordan, *Stochastic modeling of seafloor morphology: Inversion of sea beam data from second-order statistics*, J. Geophys. Res., 93 (1988), pp. 13,589–13,608.
[17] T. Hou and X. Wu, *A multiscale finite element method for elliptic problems in composite materials and porous media*, J. Comput. Phys., 134 (1997), pp. 169–189.
[18] T. Hughes, *Multiscale phenomena: Green's functions, the Dirichlet-to-Neumann formulation, subgrid scale models, bubbles and the origins of stabilized methods*, Comput. Methods Appl. Mech. Engrg., 127 (1995), pp. 387–401.
[19] D. Komatitsch, R. Martin, J. Tromp, M. Taylor, and B. Wingate, *Wave propagation in 2-d elastic media using a spectral element method with triangles and quadrilangles*, J. Comput. Acoust., 9 (2001), pp. 703–718.
[20] D. Komatitsch, S. Tsuboi, and J. Tromp, *The Spectral-Element Method in Seismology*, vol. 157 of Geophysical Monograph Series, American Geophysical Union, Washington, DC, 2005.
[21] D. Komatitsch, J.-P. Vilotte, R. Vai, J. Castillo-Covarrubias, and F. Sanchez-Sesma, *The spectral-element method for elastic wave equations–application to 2-d and 3-d seismic problems*, Int. J. Numer. Methods Eng., 45 (1999), pp. 1139–1664.
[22] O. Korostyshevskaya and S. Minkoff, *A matrix analysis of operator-based upscaling for the wave equation*, SIAM J. Numer. Anal., 44 (2006), pp. 586–612.
[23] A. Levander, *Fourth-order finite-difference P-SV seismograms*, Geophys., 53 (1988), pp. 1425–1436.
[24] X. S. Li and J. W. Demmel, *SuperLU_Dist: A scalable distributed-memory sparse direct solver for unsymmetric linear systems*, ACM Trans. Math. Software, 29 (2003), pp. 110–140.
[25] S. Minkoff, *Spatial parallelism of a 3d finite difference, velocity-stress elastic wave propagation code*, SIAM J. Sci. Comput., 24 (2002), pp. 1–19.
[26] H. Owhadi and L. Zhang, *Metric based up-scaling*, accepted for publication in Communications on Pure and Applied Mathematics, available online from http://arxiv.org/abs/math/0505223 (2005).
[27] H. Owhadi and L. Zhang, *Homogenization of the acoustic wave equation with a continuum of scales*, available online from http://arxiv.org/abs/math/0604380 (2006).
[28] M. Peszynska, M. F. Wheeler, and I. Yotov, *Mortar upscaling for multiphase flow in multiblock domains*, Comput. Geosci., 6 (2002), pp. 315–332.
[29] G. Sleefe, D. Aldridge, G. Elbring, J. Claassen, H. Garbin, and R. Shear, *Advanced subterranean warfare (ASW) for deep underground structures*, Technical report, SAND98-0988, Sandia National Labs, Albuquerque, NM, 1998.
[30] W. W. Symes, I. Terentyev, and T. Vdovina, *Gridding requirements for accurate finite difference simulation*, in 78th Annual International SEG Meeting, SM 1.5, Society of Exploration Geophysicists, Las Vegas, NV, 2008.
[31] T. Vdovina, S. Minkoff, and O. Korostyshevskaya, *Operator upscaling for the acoustic wave equation*, SIAM J. Multiscale Model. Simul., 4 (2005), pp. 1305–1338.
[32] T. Vdovina and S. Minkoff, *An a priori error analysis of operator upscaling for the acoustic wave equation*, Internat. J. Numer. Anal. Modeling, 5 (2008), pp. 543–569.
[33] T. Vdovina, *Operator Upscaling for the Wave Equation*, Ph.D. thesis, University of Maryland Baltimore County, Baltimore, MD, 2006.
[34] J. Virieux, *SH-wave propagation in heterogeneous media: Velocity stress finite-difference method*, Geophys., 49 (1984), pp. 1933–1957.
[35] J. Virieux, *P-SV wave propagation in heterogeneous media: Velocity stress finite-difference method*, Geophys., 51 (1986), pp. 889–901.