Physics-motivated numerical solvers for partial differential equations

L. San Martin and Y. Oono
Department of Physics and Materials Research Laboratory, 1110 West Green Street, and University of Illinois at Urbana–Champaign, Urbana, Illinois 61801-3080

(Received 22 June 1995; revised manuscript received 25 August 1997)

Trying to capture the essential physics of a natural phenomenon directly on computers may lead us to useful numerical schemes to solve the partial differential equation describing the phenomenon. Here we try to capture the consequences of space-time translational symmetry such as advection in fluids or Huygens' principle in wave propagation. Efficient modeling of these phenomena becomes possible with the aid of Hermite polynomial interpolations to realize a continuum on discrete lattices. To illustrate these ideas, we present a new method to derive wave equation solvers that are high order but local (the computational cell or stencil includes nearest neighbors only), a clear advantage over standard high-order algorithms of the finite-difference or finite-element families. The purpose of the paper is to demonstrate our methodology. Therefore, in two- and three-spaces, details are given only for the lowest-order algorithms, a preview of a more optimal higher-order scheme is also included. [S1063-651X(98)01204-5]

PACS number(s): 02.70.Bf, 02.60.Cb

I. INTRODUCTION

Many nonequilibrium phenomena are spatially extended, and the most popular means to model them is the partial differential equation (PDE). Resultant PDE's are, however, often nonlinear, defying analytical approaches. Even when the PDE's are linear, the conventional ''mathematical physics'' is not useful in many cases due, e.g., to nontrivial geometry of their domains. In this paper we wish to demonstrate that the problem of devising numerical solvers for (physically meaningful) PDE's may be considered as modeling problems or at least motivated as such in physics. As an example of our approach, we present a wave equation solver based on Huygens' principle [1].

The new algorithms have several attractive features. Important among them is the realization of high-order accuracy without increasing the size of the computational cell, i.e., the computational stencil includes nearest neighbors only. We also demonstrate that the fifth-order version of the algorithm can propagate sharp pulses over long distances with little distortion. This feature facilitates the development of higher-order schemes that can be applied to practical situations that can include inhomogeneities and geometrically complicated boundaries. However, the main purpose of the paper is to explain our methodology. We evaluate the quality of the algorithm in one-space and introduce the lowest order scheme for three-space. Results of two variants of the lowest-order algorithm applied to a two-space test problem are given. For comparison we include a result obtained with a higher-order version of the algorithm. A detailed evaluation and applications of the higher-order scheme are planned to be given in a subsequent paper [2].

Our motivation came from the successful examples of solvers for the Cahn-Hilliard equation [3] and the Fisher-KPP equation [4]. In fact these solvers were not conceived originally as PDE solvers. Modeling the phase transition kinetics in terms of space-time discrete cell-dynamical systems (CDS's) [5] was initially conceived independent of the existing PDE models more or less in the spirit of the lattice gauge theory [6]. The basic idea of the hyperbolic solver implemented in this paper was an outcome of *ab initio* discrete modeling of fluid dynamics (i.e., independent of the Navier-Stokes equation) within CDS's [7]. There, an interpolation-resampling strategy was proposed as a means to capture spatial translational symmetry on a discrete lattice. A natural extension of this idea to wave propagation leads to the use of interpolation to implement Huygens' principle on a lattice.

Our approach may be interpreted as a general method to devise numerical schemes for PDE's in terms of Hermite interpolation. However, the details of how to use the Hermite interpolation and the convenience of using a high- or a low-order implementation are conditioned by the type of PDE to be solved. We believe that the methods described below are ideal for PDE's with a finite domain of dependence (crudely speaking, PDE's in which disturbances propagate with finite speed).

The use of interpolation is somewhat reminiscent of the finite-element strategy. However, the similarity is only superficial. The new algorithm is based on the implementation of Huygens' principle as a solution map for the wave equation. The inclusion of additional equations for derivatives make our algorithm explicit, unlike the traditional finite-element algorithms which are implicit. The new wave solver is explicit and local as the second-order finite-difference algorithms.

In Sec. II, we explain the interpolation-resampling approach based on the Hermite polynomial interpolation. In Sec. III, preliminary considerations on wave equation solvers are given, and the algorithm with which we compare our results is presented. In Sec. IV, the general idea for the wave equation solver is presented and illustrated in one-space. Two versions of the algorithm, based on third- and fifth-order implementations of Huygens' principle, are described and evaluated. The robustness of the fifth-order version is tested by computing the solution of the wave equation with a position-dependent velocity. In Sec. V, a solver for the three-space wave equation based on a third-order interpolation is presented. Two versions of the algorithm are evaluated in a two-space problem. In Sec. VI, an evaluation of the efficiency of the optimal three-space solver is given. Section VII is devoted to a summary and final remarks.

II. INTERPOLATION-RESAMPLING STRATEGY

The use of interpolation for the computation of advection is an old idea. Krishnamurty [8], more than 30 years ago, proposed the use of interpolation in an implicit scheme for computing advection in the context of weather prediction. Several improvements since then have produced the so called semi-Lagrangian advection schemes of wide acceptance in the numerical weather prediction (NWP) community [9]. These methods are now standard in medium range forecasts in many countries [10]. In weather prediction incompressibility is normally assumed, and the complications associated with shocks or contact discontinuities are not present.

Even though highly popular among the NWP community, the method was not applied outside this community until the eighties. Benque *et al.* [11] used a semi-Lagrangian approach to compute the solution of the Navier-Stokes equation in two-space. Malevsky and co-workers used a parallel implementation of a semi-Lagrangian scheme with third-order splines in the context of advection diffusion, and the Navier-Stokes equations in two- and three-spaces [12,13]. Yabe and co-workers [14] proposed a similar approach based on Hermite polynomial interpolation as a general hyperbolic equation solver. Yabe and coworkers obtained excellent results with their semi-Lagrangian scheme in several compressible hydrodynamic tests. They showed that shocks and contact discontinuities can be captured with as few as two grid points [14]. Several other highly nontrivial applications are given in Refs. [14,15].

To explain the interpolation-resampling strategy, we consider a numerical scheme for the elementary first-order linear hyperbolic PDE:

$$(\partial_t + c\,\partial_x) f(x,t) = 0, \tag{2.1}$$

where c is a constant, $\partial_x \equiv \partial/\partial x$, and $\partial_t \equiv \partial/\partial t$. The ordinary hyperbolic conservation law is close to this equation locally in space. It is currently the trend in the physics and astrophysics communities to use algorithms in which conservation is explicitly built into the schemes. Although such algorithms may be less prone to develop unphysical solutions, needless to say, conservation alone cannot guarantee the accuracy of the result. Conservation laws are often due to symmetry (and variational principles). Therefore, we consider the symmetry of the system first. The crucial symmetry for hyperbolic transport systems is the (at least local) translational symmetry of space where the disturbance propagates at a finite speed. Equation (2.1) implies

$$f(x,t+\delta t) = f(x - c\,\delta t, t), \tag{2.2}$$

so that evolving Eq. (2.1) for δt is equivalent to estimating f accurately at position $x - c\,\delta t$ at time t. If we could describe the translational symmetry of continuum space precisely, this estimation is trivial.

If we may assume that the grid mesh is fine enough [16], then we must try to preserve the information already captured by the values of f sampled at the grid points. The basic idea to implement this information preservation is illustrated in Fig. 1. In short, we reconstruct the continuum with the aid of an appropriate interpolation method, translate it spatially, and then resample the values of the translated interpolant at the grid points. Thus the usefulness of this interpolation-resampling strategy depends on the availability of good interpolation schemes, which may be problem dependent.

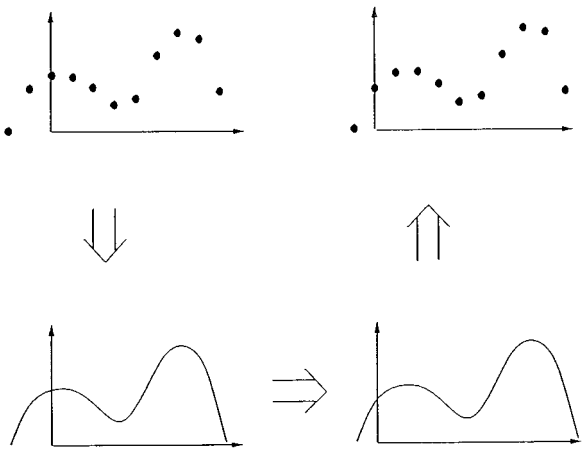

FIG. 1. Interpolation-resampling strategy. With the data at the grid points of the function f, a continuous interpolant is generated and shifted by $v\,\delta t$. Then, the interpolant is resampled to obtain new values of f at the grid points.

There are many different interpolation schemes. These can be classified into two types: local interpolation schemes and global interpolation schemes. The most popular global interpolation schemes are spectral methods which utilize (generalized) Fourier expansion. We immediately recognize two limitations of these methods. One is the requirement of relatively simple geometry of the domain of the problem in order to be able to find the eigenmodes of the structure, and the other is the need of a large number of modes to capture rapid localized changes such as shock fronts.

With a local interpolation these problems are considerably diminished. A characteristic feature of hyperbolic equations is the finite speed of propagation of disturbances, which implies the localization of the domain of dependence of the solution. This feature is naturally implemented with a local interpolation scheme. In this respect the Hermite interpolation we have chosen is optimal because it involves only nearest neighbors. Also, if we expect to use parallel computational environments, it is generally more advantageous to use a local interpolation scheme than a global one. Among the local interpolation schemes, splines [12,13], give interesting alternative possibilities. High-order splines, however, require an extended stencil which makes the treatment of inhomogeneities and localized boundaries highly nontrivial.

We now describe the Hermite polynomial interpolation [17] used in the way proposed by Yabe and co-workers [14]. As we show below, with the introduction of additional variables for derivatives, we obtain a high-order Hermite polynomial interpolation that involves only nearest neighbors and whose coefficients can be solved explicitly. The practically useful lowest-order interpolation is the third order. The one-space case is discussed here in detail for the sake of illustration. Let $\{x_i\}$ be the coordinates of grid points on the x axis ordered as $\{i\}$. We use the notations $f[i] = f(x_i)$, $f'[i] = f'(x_i)$. The third-order Hermite interpolant $F_i(x)$ of $f(x)$ for $x \in [x_i, x_{i+1}]$ is defined in terms of $f[i]$, $f[i+1]$, $f'[i]$, and $f'[i+1]$ as

$$F_i(x) \equiv C_3[i]X^3 + C_2[i]X^2 + C_1[i]X + C_0[i], \quad (2.3)$$

where $X = x - x_i$ and the coefficients $C_j[i]$ $(j = 0, \ldots, 3)$ are determined by the condition that the interpolant and its first derivative are continuous at the end points of the interval $[x_i, x_{i+1}]$. Thus we obtain $C_0[i] = f[i]$, $C_1[i] = f'[i]$,

$$C_2[i] = \frac{3(f[i+1] - f[i])}{\Delta x^2} - \frac{2f'[i] + f'[i+1]}{\Delta x}, \quad (2.4)$$

and

$$C_3[i] = \frac{f'[i] + f'[i+1]}{\Delta x^2} - \frac{2(f[i+1] - f[i])}{\Delta x^3}, \quad (2.5)$$

where $\Delta x \equiv x_{i+1} - x_i$.

In order to use these explicit formulas for the coefficients we need $f[i]$ and $f'[i]$ at every time step. One way proposed in Ref. [14] (actually the general idea was proposed by Tusheva *et al.* in 1975 according to Ref. [18]) is to introduce an additional equation for f'. This approach gives excellent results when applied to compressible hydrodynamics and to many other cases [14,15]. Thus the interpolation algorithm for Eq. (2.1) reads as follows with this Yabe proposal. Let us denote the value of f (f') at the nth time step at the grid point x_i as $f_n[i]$ $(f'_n[i])$. The updating algorithm with the third-order Hermite interpolation is given by

$$f_{n+1}[i] = F_i(x_i - c\,\delta t) = f_n[i] + \{(C_3[i]\xi + C_2[i])\xi + f'_n[i]\}\xi, \quad (2.6)$$

$$f'_{n+1}[i] = \partial_x F_i(x_i - c\,\delta t) = f'_n[i] + (3C_3[i]\xi + 2C_2[i])\xi, \quad (2.7)$$

where $\xi = -c\,\delta t$. This expression is for $c < 0$. For $c \geq 0$ the equivalent expression is obtained by replacing Δx with $-\Delta x$ and $i+1$ with $i-1$ in Eqs. (2.4) and (2.5). Yabe and coworkers [14] computed C_j and then used Eqs. (2.6) and (2.7) to update the system. We can, however, combine these two steps into one as a map from the time n to the time $n+1$ as

$$f_{n+1}[i] = f_n[i]A_1[i] + f_n[i - \mathrm{sgn}(c)]A_2[i] - c\,\delta t\{f'_n[i]A_3[i] + f'_n[i - \mathrm{sgn}(c)]A_4[i]\}, \quad (2.8)$$

$$f'_{n+1}[i] = -\mathrm{sgn}(c)\{f_n[i]A_5[i] + f_n[i - \mathrm{sgn}(c)]A_6[i]\}/\Delta x + f'_n[i]A_7[i] + f'_n[i - \mathrm{sgn}(c)]A_8[i], \quad (2.9)$$

where the coefficients are functions of the Courant-Friedrichs-Lewy (CFL) number, $\kappa \equiv c\,\delta t/\Delta x$, as $A_1 = 1 + 2\kappa^3 - 3\kappa^2$, $A_2 = -2\kappa^3 + 3\kappa^2$, $A_3 = 1 + \kappa^2 - 2\kappa$, $A_4 = \kappa^2 - \kappa$, $A_5 = 6\kappa^2 - 6\kappa$, $A_6 = -6\kappa^2 + 6\kappa$, $A_7 = 1 + 3\kappa^2 - 4\kappa$, and $A_8 = 3\kappa^2 - 2\kappa$. The map defined by Eqs. (2.8) and (2.9) has only one parameter, κ, as can easily be seen by multiplying Eq. (2.9) by $c\,\delta t$; the map consisting of Eq. (2.8) and $c\,\delta t \times$ Eq. (2.9), for the variables f and $\tilde{f}' \equiv c\,\delta t f'$, is explicitly dependent only on κ. In the one-space case shown in Eqs. (2.8) and (2.9), the stability of the scheme requires the CFL number to be in [0,1] [19].

Higher-order interpolation-resampling schemes can be constructed with the aid of higher-order derivatives such as $\partial_x^2 f$. For each newly added derivative, the degree of the interpolation polynomial increases by two. For example, the fifth-order Hermite scheme uses the continuity conditions for f, $\partial_x f$, and $\partial_x^2 f$ to determine the interpolation coefficients. The formulas for the fifth-order Hermite interpolant in one-space are given in Appendix A. In the following we demonstrate the quality of the fifth-order interpolation for the pure advection problem [20,21].

We must know the improvements that can be accomplished by increasing the order of interpolation schemes. To this end the third- and fifth-order schemes are compared through solving Eq. (2.1) with $c = 1$ for two types of initial conditions: sharp pulses and smooth sinusoidal curves. For the sharp pulses we use a grid of 200 points with periodic boundary conditions and $\Delta x = 0.005$ and $\delta t = 0.2\Delta x$ (i.e., $\kappa = 0.2$). Recall that the map depends only on the ratio $\kappa = c\,\delta t/\Delta x$, so that the effect of the choice of Δx is only through κ. We propagate the following pulses.

Pulse A:

$$u_0[i] = \begin{cases} 1 & \text{for } \forall i \in \{90, \ldots, 110\} \\ 0 & \text{for } \forall i \in \{1, \ldots, 89\} \cup \{111, \ldots, 200\}. \end{cases} \quad (2.10)$$

Pulse B:

$$u_0[i] = \begin{cases} 1 & \text{for } \forall i \in \{92, \ldots, 108\} \\ 0 & \text{for } \forall i \in \{1, \ldots, 88\} \cup \{112, \ldots, 200\} \\ 0.25(i - 88) & \text{for } \forall i \in \{89, \ldots, 91\} \\ -0.25(i - 112) & \text{for } \forall i \in \{109, \ldots, 111\}. \end{cases} \quad (2.11)$$

The results of pulse A after 1000 time steps are shown in Fig. 2(a), and those after 20 000 steps in Fig. 2(b). Numerical diffusion is greatly reduced in the fifth-order scheme. Small overshootings at the edges of the pulse are due to the interpolation [14]. The results of other commonly used algorithms for the solution of pulse A are given in Yabe and Takei [14]. Figure 2(c) contains the results for the initial pulse B after 1000 time steps; those after 20 000 time steps are shown in Fig. 2(d). For this pulse the fifth-order scheme produces almost no overshooting.

As a smooth initial condition we use a cosine function. Pulse C:

$$u_0[i] = \cos[k(i - N/2)\Delta x] \quad \text{for} \quad \forall i \in \{1, \ldots, N\}, \quad (2.12)$$

where $k = 2\pi/(L\Delta x)$, and L is the number of points per wavelength (ppw). To be consistent with the periodic boundary conditions, we use wavelengths commensurate with the computational grid. For the third-order scheme we use a grid of $N = 1000$ points with $\Delta x = 0.005$, $\delta t = 0.2\Delta x$, and the following values of L: 31.25, 62.5, 125, 250, 500, and 1000. For the fifth-order scheme we use a grid of $N = 250$ points

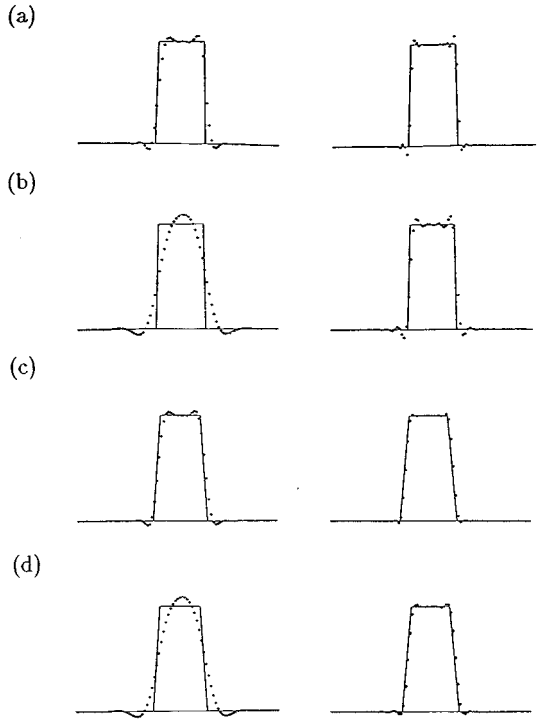

FIG. 2. Comparison of third- (left) and fifth- (right) order interpolation-resampling schemes. Propagation of the initial condition A after (a) 1000 time steps and (b) 20 000 time steps. Propagation of the initial condition B (three points in the transition region) after (c) 1000 time steps and (d) 20 000 time steps.

with $\Delta x=0.02$, $\delta t=0.2\Delta x$, $c=1$, and the values of L: 3.90625, 7.8125, 15.625, 31.25, 62.5, 125, and 250.

Due to the periodic boundary conditions, after a certain number of steps, the propagated sinusoidal curves return to their initial positions. At such instants we measure the error in the l_1 norm ($\|x-y\|\equiv\Sigma|x_i-y_i|$) between the computed and exact solutions. Figure 3(a) shows the growth of the normalized l_1-norm error for the third order with 62.5 ppw, and the fifth order with 15.625 ppw. We normalize the error dividing the l_1-norm error by the l_1 norm of the initial condition. Figure 3(b) shows the amplitude attenuation rates per iteration as a function of the number of ppw. For the wave equation numerical dispersion is the largest source of error, so the amplitude attenuation is not very significant.

Figure 3(b) demonstrates that the fifth-order scheme with one-fourth as many grid points per wavelength produces a smaller error than the third-order scheme. The number of operations is six (10) additions and eight (12) multiplications per update for the third- (fifth-) order scheme. The fifth order is superior [22].

The extension of the above interpolation scheme to higher dimensions is straightforward. For convenience, the third-order Hermite interpolation formulas for two- and three-spaces, which will be used in Sec. V, are given in Appendix B.

Due to the inclusion of higher-order derivatives, the memory requirement increases with the order of the scheme. For example, in three-space a continuous function has three first, six second, ten third, etc. independent derivatives [23]. Correspondingly, the number of operations increases. If, however, Δx can be made large enough to offset the difference in number of operations and memory requirement, then the higher-order method is more efficient, as we have seen in the comparison of the third- and fifth-order schemes.

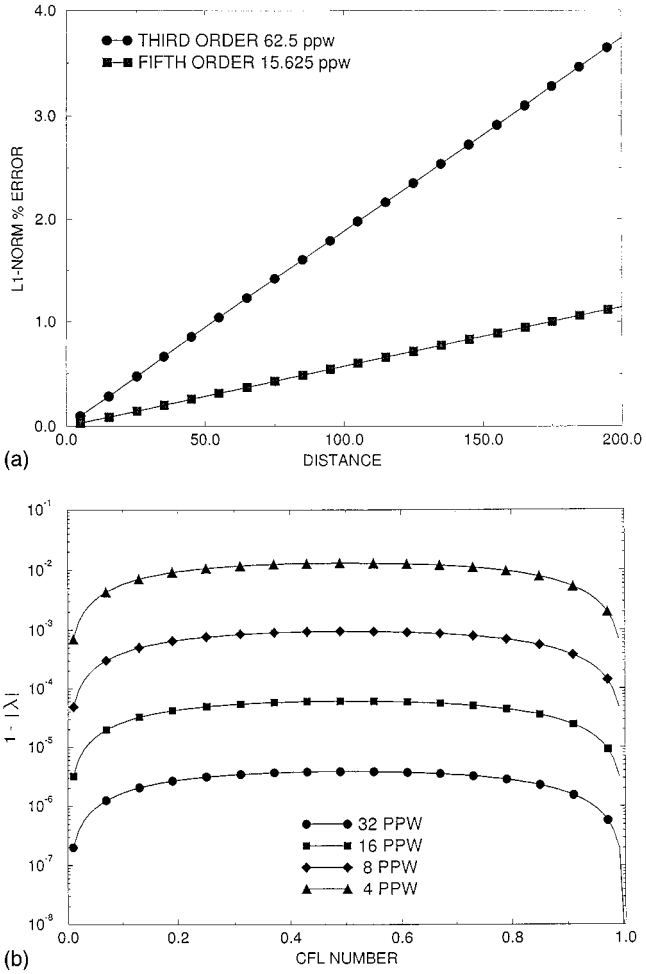

FIG. 3. Comparison of errors of third- and fifth-order interpolation-resampling schemes for test C. (a) The growth of the l_1-norm error. (b) The amplitude attenuation rates per iteration as a function of ppw for various CFL number.

The Hermite polynomial interpolation we used for Eq. (2.1) satisfies the global conservation law that the sum over space of f is preserved. This can be shown easily for any higher-order schemes, provided the initial spatial derivatives are zero (we can impose such initial conditions, as pointed out by Yabe and co-workers [24]). However, whether or not conservation holds exactly is not very relevant in wave propagation, because the main errors are due to numerical dispersion.

III. WAVE EQUATION

We wish to devise a solver for the linear wave equation

$$(\partial_t^2-c^2\Delta)u(\mathbf{x},t)=0. \tag{3.1}$$

This is a first step toward deriving algorithms for more complicated wave phenomena.

Wave equation solvers can be classified into two major types: frequency-domain solvers and space-time (ST) domain solvers. We are interested in the computation of tran-

sient behavior in structures of complicated geometry with or without inhomogeneity. For this type of problems space-time methods are more appropriate. We do not compare with frequency-domain methods because they are useful only when the eigenmodes of the structure of interest can be found easily. In such cases the frequency-domain solution is computed first, and by taking an inverse Fourier transform the time domain solution is obtained [25]. The procedure may be prohibitively expensive when the propagating wave is composed of sharp pulses because of the wide band required.

The pseudospectral methods, although very good for simple geometries [26–28], are not equally advantageous when the domain boundaries do not coincide with the nodal planes of the numerical lattice or when the geometrical details require smaller scales than $\lambda/2$ [29,30]. In such cases, to maintain the accuracy of the scheme, a special treatment of boundaries becomes necessary. The use of interpolation for this purpose is natural. It might be convenient to develop a sort of hybrid scheme combining an algorithm based on interpolation, as the one presented in this paper, with the ordinary procedures of the pseudospectral algorithms.

Among the ST methods there are two main types, the finite-element time domain (FETD) approach and the finite-difference time domain (FDTD) approach. Recently, there have been advances in explicit FETD methods for the wave equation [31]. The FETD methods so far proposed are, however, not better than the standard second-order FDTD method; even though FETD methods can be high order and have a smaller error, the computational cost increases so that the second-order FDTD with a finer mesh is still more efficient. We will not consider the standard FETD methods because they are implicit and, consequently, inefficient in comparison to the FDTD methods for the computation of transients. In the last few years, FDTD methods have become the preferred choice in many applications [32,33]. FDTD algorithms are local in the sense that the evolution at a given grid point depends only on the local information around that grid point. Consequently, they are ideal for parallel computation.

To contrast our results, we have chosen the second-order centered differences (2-2CD) solver of the wave equation. With the notation $u_n[ijk] \equiv u(x_{ijk}, n\delta t)$ for the scalar function u at position $x_{ijk} = (i\Delta s, j\Delta s, k\Delta s)$ and at time $t = n\delta t$, the 2-2CD algorithm for the three-space wave equation (3.1) is

$$u_{n+1}[ijk] = \left(\frac{c\delta t}{\Delta s}\right)^2 \{u_n[i+1jk] + u_n[i-1jk] + u_n[ij+1k] + u_n[ij-1k] + u_n[ijk+1] + u_n[ijk-1] - 6u_n[ijk]\} + 2u_n[ijk] - u_{n-1}[ijk]. \quad (3.2)$$

The main reasons for the success of the second-order finite difference algorithms are its efficiency and, in particular, its robustness. For example, Yee's algorithm in electromagnetism has been extensively applied to materials with inhomogeneities, anisotropies, memory effects, etc., [33]. This is a reason for our choice. Further justifications of this choice follow.

First of all we wish to point out that many practical FDTD codes use second-order differences [33]. This is the case even though it has been known for many years [34] that higher-order finite-difference algorithms (e.g., fourth order) are more efficient when applied to the propagation in homogeneous space. The development has been slow because straightforward generalizations of FDTD to higher orders lack robustness beyond the simple homogeneous space without complicated boundaries. More elaborate high-order FDTD for wave propagation is an active area of research [35,36]. We are, however, rather skeptical about the significant and swift progress in general purpose high-order accurate codes that are practically applicable to, e.g., elastodynamics or Maxwell's equations, especially when the boundaries between media cannot coincide with the planes of the numerical lattice. It is natural then to compare our method with the 2-2CD method which is the basis of most of the practical space-time codes; e.g., Yee's algorithm in electromagnetism can be reduced to Eq. (3.2) in homogeneous space.

The key problem is to develop a high-order FDTD algorithm that is stable and keeps its high-order accuracy in the presence of inhomogeneity and nontrivial boundaries. Higher-order schemes in the standard FDTD approach require larger stencils. When the media is inhomogeneous, e.g., in the staircase approximation, the stencil includes several cells with different material constants. To obtain a stable high-order accurate discrete implementation of the operators in the general case is a difficult problem. A fourth-order method in space and time for smooth inhomogeneity was proposed in Ref. [35], but, as pointed out in Ref. [37], there is a discrepancy between the order of convergence in homogeneous space and that with inhomogeneity. See Ref. [37] for an error analysis of several higher-order schemes for the wave equation in inhomogeneous media. For the two-space inhomogeneous problem, Ref. [38] claimed to have developed a method which needs 68 coefficients per node that need be stored in order to account for the inhomogeneity. Our method which can be applied to the inhomogeneous case as well, and requires 20 variables per node, in three-space, is an attractive alternative.

The high-order implementation of boundary conditions is also a nontrivial problem, with the additional complication that there are no real grid points beyond the physical boundaries. Only recently a one-space implementation of boundary conditions for high-order schemes that is stable in time was proposed [39]. In the case of interfaces between media that coincide with the planes of the grid mesh, a stable fourth-order algorithm was developed [40]. We are not aware of more complicated geometries treated with high-order methods.

Given the slow progress in the development of standard higher-order FDTD algorithms, we are proposing an alternative way to reap the benefit of higher orders, i.e., a sharp reduction of numerical dispersion that allows an overall increase in efficiency. Although the number of operations per node increases, the reduction in numerical dispersion implies that for a given error tolerance the higher-order methods require fewer points per wavelength, thus the higher-order methods are also efficient in their use of memory resources.

The main problem of the 2-2CD algorithm (3.2) is the dispersion of the computed modes. That is, the phase velocity of numerical modes in the 2-2CD scheme varies with modal wavelength, direction of propagation, and lattice discretization. Numerical dispersion can lead to nonphysical results such as pulse distortion and artificial anisotropy. Pseudorefraction will be produced whenever the grid size varies with position. The severity of these problems depends on the length in time and the relation between the smallest wavelength of the propagating wave and the size of the grid mesh. For typical simulations 10–20 points per wavelength are used.

The stability limit of the 2-2CD algorithm for the d-space problem is a CFL number less than or equal to $1/\sqrt{d}$. In one-space, when the algorithm is used with a CFL number equal to 1, the initial condition is propagated without distortion. This result does not extend to higher dimensions. The distortion introduced by 2-2CD in higher dimensions is similar to the distortion that occurs in the one-space problem when run with a CFL number less than 1. In the next section we will evaluate the one-space algorithm for a CFL number less than 1 in order to have an estimate of the performance in higher dimensions. These results are also relevant for the propagation of waves in inhomogeneous materials in one-space, because in those cases the CFL number cannot be the same everywhere. For more details about the 2-2CD algorithm for the wave equation, see Ref. [25].

IV. WAVE PROPAGATION IN ONE-SPACE

We wish to solve the initial value problem of Eq. (3.1) with the initial conditions, $u(\mathbf{x},t)$ and $\partial_t u(\mathbf{x},t)$ given at some initial time t in an infinite domain $\mathbb{R}^d$. The evolution from t to $t+\delta t$ can be written as

$$\begin{pmatrix} u(\mathbf{x},t+\delta t) \\ \partial_t u(\mathbf{x},t+\delta t) \end{pmatrix} = \begin{pmatrix} K & L \\ c^2 LD & K \end{pmatrix} \begin{pmatrix} u(\mathbf{x},t) \\ \partial_t u(\mathbf{x},t) \end{pmatrix}, \tag{4.1}$$

where K, D, and L are linear operators depending on spatial dimensionality.

The generalized Huygens principle can be cast into the form of Eq. (4.1). In three-space for example, thanks to the strong form of Huygens' principle, all the terms in the map are integrals with a spherical domain. Historically, Huygens' principle is older than the PDE called the wave equation due to d'Alembert. Recently, efficient algorithms for the computation of scattering (elliptic equations) based on Huygens' equivalence principle were developed [41]. In our algorithm we implement Eq. (4.1) with the aid of interpolants of $u(x,t)$ and $\partial_t u(x,t)$.

In a bounded domain, map (4.1) should be used for the grid points in the interior of the domain. The boundary grid points, defined as those grid points whose distance to the boundary is less than $c\,\delta t$, require a different map whose details depend on the particular shape of the nearby boundary. In this paper we show how to construct a map for the interior grid points. The use of interpolation is advantageous for the treatment of boundary conditions. Perhaps the simplest way to deal with a boundary of arbitrary orientation is by redrawing the grid locally, in conformity with the shape of the boundary, and then using the interpolation to transfer values between grids. This procedure for the modeling of boundaries was already developed in conjunction with second-order schemes in electromagnetics [42]. In our algorithm this approach can easily be incorporated because the interpolation is already available as an integral part of the algorithm.

We now restrict our attention to the one-space case to illustrate the basic scheme and evaluate the numerical errors. Map (4.1) in one-space takes the form:

$$u(x,t+\delta t) = \tfrac{1}{2}\{u(x-c\,\delta t,t)+u(x+c\,\delta t,t)\} + \frac{1}{2c}\int_{x-c\,\delta t}^{x+c\,\delta t} d\sigma\ \partial_t u(\sigma,t), \tag{4.2}$$

$$\partial_t u(x,t+\delta t) = \frac{c}{2}\{-\partial_x u(x-c\,\delta t,t)+\partial_x u(x+c\,\delta t,t)\} + \tfrac{1}{2}\{\partial_t u(x+c\,\delta t,t)+\partial_t u(x-c\,\delta t,t)\}. \tag{4.3}$$

The use of third-order interpolation requires additional equations for the spatial derivatives of $u(x,t)$ and $\partial_t u(x,t)$:

$$\partial_x u(x,t+\delta t) = \tfrac{1}{2}\{\partial_x u(x-c\,\delta t,t)+\partial_x u(x+c\,\delta t,t)\} + \frac{1}{2c}\{\partial_t u(x+c\,\delta t,t)-\partial_t u(x-c\,\delta t,t)\}. \tag{4.4}$$

$$\partial_x\partial_t u(x,t+\delta t) = \frac{c}{2}\{-\partial_x^2 u(x-c\,\delta t,t)+\partial_x^2 u(x+c\,\delta t,t)\} + \tfrac{1}{2}\{\partial_x\partial_t u(x+c\,\delta t,t)+\partial_x\partial_t u(x-c\,\delta t,t)\}. \tag{4.5}$$

Note that all but the integral term in Eq. (4.2) are given by simple shifts in space. We use the notation $u_n[i] = u(x_i, n\,\delta t)$ ($\partial_t u_n[i] = \partial_t u(x_i, n\,\delta t)$, etc.) to denote the value of u ($\partial_t u$, etc.) at time $t = n\,\delta t$ and position $x = i\Delta x$. We proceed as follows. With given values of $u_n[i]$, $\partial_x u_n[i]$, $\partial_t u_n[i]$, and $\partial_x\partial_t u_n[i]$, third-order interpolants for the functions $u(x,t)$ and $\partial_t u(x,t)$ are generated. By resampling the interpolants at a distance $c\,\delta t$ to the left or right of the grid points, the shift terms in Eqs. (4.2)–(4.5) are computed. The integral in Eq. (4.2) is evaluated by integrating the third-order interpolant of $\partial_t u(x,t)$. We will call this algorithm HUY3.

The use of fifth-order interpolation requires additional equations for $\partial_x^2 u(x,t)$ and $\partial_x^2\partial_t u(x,t)$. Spatial differentiation of Eqs. (4.4) and (4.5) gives

$$\partial_x^2 u(x,t+\delta t) = \tfrac{1}{2}\{\partial_x^2 u(x-c\,\delta t,t)+\partial_x^2 u(x+c\,\delta t,t)\} + \frac{1}{2c}\{\partial_x\partial_t u(x+c\,\delta t,t)-\partial_x\partial_t u(x-c\,\delta t,t)\} \tag{4.6}$$

and

$$\partial_x^2\partial_t u(x,t+\delta t)=\frac{c}{2}\{-\partial_x^3 u(x-c\,\delta t,t)+\partial_x^3 u(x+c\,\delta t,t)\}$$
$$+\tfrac{1}{2}\{\partial_x^2\partial_t u(x+c\,\delta t,t)+\partial_x^2\partial_t u(x-c\,\delta t,t)\}. \tag{4.7}$$

Knowing $u_n[i]$, $\partial_x u_n[i]$, $\partial_x^2 u_n[i]$, $\partial_t u_n[i]$, $\partial_x\partial_t u_n[i]$, and $\partial_x^2\partial_t u_n[i]$, we can compute all the terms in Eqs. (4.2)–(4.7) from the fifth-order interpolants of $u(x,t)$ and $\partial_t u(x,t)$ (Appendix A). We will call this algorithm HUY5.

To evaluate the quality of these two algorithms, we solve Eq. (3.1) with sharp and smooth pulses as initial conditions. This test is very important because, as we mentioned before, the errors in two- and three-spaces are of the same type. In two- and three-spaces there is the additional issue of numerical anisotropy which is important, and can also be largely diminished by the HUY schemes, as the tests in Sec. V show. We now proceed to the description of the tests.

For the propagation of sharp pulses we use a grid of 200 points with periodic boundary conditions, $\Delta x=0.005$, $\delta t=0.2\Delta x$, and $c=1.0$. The initial conditions are as follows. For pulse D:

$$u_0[i]=\begin{cases}1 & \text{for } \forall i\in\{90,\ldots,110\}\\ 0 & \text{for } \forall i\in\{1,\ldots,89\}\cup\{111,\ldots,200\},\end{cases} \tag{4.8}$$

$$\partial_t u_0[i]=0 \quad \text{for } \forall i\in\{1,\ldots,200\}. \tag{4.9}$$

For pulse E:

$$u_0[i]$$
$$=\begin{cases}1 & \text{for } \forall i\in\{92,\ldots,108\}\\ 0 & \text{for } \forall i\in\{1,\ldots,88\}\cup\{112,\ldots,200\}\\ 0.25(i-88) & \text{for } \forall i\in\{89,\ldots,91\}\\ -0.25(i-112) & \text{for } \forall i\in\{109,\ldots,111\},\end{cases} \tag{4.10}$$

$$\partial_t u_0[i]=0 \quad \text{for } \forall i\in\{1,\ldots,200\}. \tag{4.11}$$

Due to the periodic boundary conditions, the waves go back to the initial position every 1000 time steps. Figure 4(a) exhibits pulse (D) after 1000 time steps and Fig. 4(b) after 20 000 time steps. The pulse propagated by HUY3 suffers a rather large deformation. Numerical dispersion is drastically diminished with HUY5. Figure 4(c) exhibits pulse E after 1000 time steps, and Fig. 4(d) after 20 000 time steps. We can say that the error is small in HUY5 even after 20 000 time steps. Large numerical dispersion errors completely distort the pulses propagated with 2-2CD. We only show one representative figure in this case. Figure 5 shows pulse D propagated with 2-2CD after 1000 time steps.

Figures 5(a)–5(d) exhibit the results for the same tests with the 2-2CD algorithm. Needless to say, for the propagation of sharp pulses the 2-2CD algorithm is out of practical question because of the large dispersion errors.

As a smooth initial condition we use a sinusoidal wave packet with a Gaussian envelope for pulse F:

$$u_0[i]=\cos[k(i-N/2)\Delta x]\exp\{-[(i-N/2)\Delta x]^2/\sigma^2\}$$

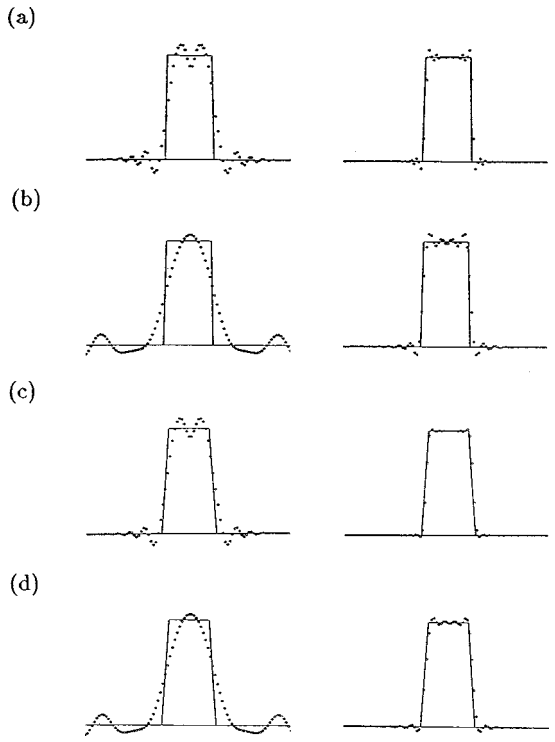

FIG. 4. Sharp pulse propagation with HUY3 (left) and HUY5 (right). Propagation of the initial condition D after: (a) 1000 time steps and (b) 20 000 time steps; propagation of the initial condition E after: (c) 1000 time steps and (d) 20 000 time steps. A significant reduction of the dispersion is clearly demonstrated.

$$\text{for } \forall i\in\{1,\ldots,N\}, \tag{4.12}$$

$$\partial_t u_0[i]=0 \quad \text{for } \forall i\in\{1,\ldots,N\}. \tag{4.13}$$

For HUY3 we use a grid of $N=1000$ points with periodic boundary conditions, $\Delta x=0.005$, $\delta t=0.2\Delta x$, $c=1.0$, $\sigma=0.3$, and $k=6.25\pi$, which is equivalent to 64 ppw. For HUY5 we use a grid of $N=250$ points with $\Delta x=0.02$, $\delta t=0.2\Delta x$, $c=1.0$, $\sigma=0.3$, and $k=6.25\pi$, which is equivalent to 16 ppw.

For the initial condition F the growth of the l_1-norm error [43] of the 2-2CD algorithm with 64 ppw is given in Fig. 6(a). Figure 6(b) shows the growth of the error for HUY3 with 64 ppw. Figure 6(c) [Fig. 6(d)] shows the same results for HUY5 with 16 ppw (6.4 ppw). HUY5 is superior to the 2-2CD

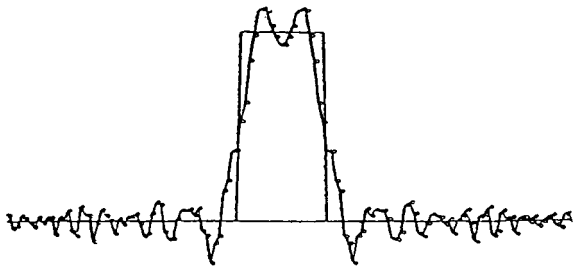

FIG. 5. Sharp pulse propagation with 2-2CD algorithm. Propagation of the initial condition D after 1000. For the propagation of sharp pulses this scheme is practically out of the question.

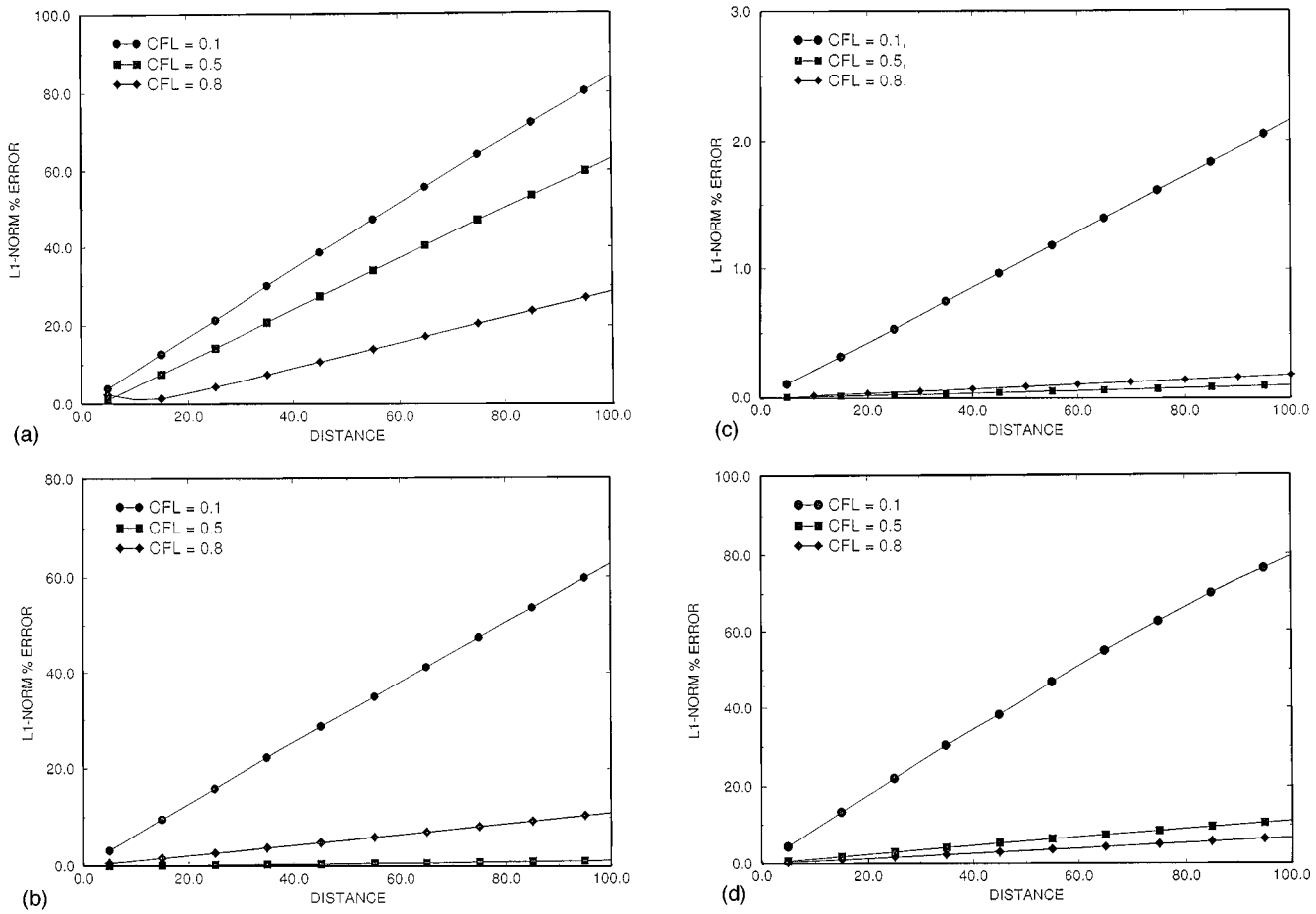

FIG. 6. Growth of the l_1-norm errors for HUY3, HUY5, and 2-2CD for the initial condition F, with a CFL number equal to 0.1, 0.5, and 0.8, are shown. (a) 2-2CD algorithm with 64 ppw, $\Delta x = 0.005$. (b) HUY3 with 64 ppw. $\Delta x = 0.005$. (c) HUY5 with 16 ppw, $\Delta x = 0.02$. (d) HUY5 with 6.4 ppw, $\Delta x = 0.05$. Note that HUY5 with $\frac{1}{10}$ as many grid points per wavelength produces an error that is comparable (still smaller) to the error due to 2-2CD.

scheme for all types of initial conditions even with one-tenth as many ppw, as demonstrated in Fig. 6(d). Comparison of CPU times for one-space cases are not important. In Sec. VI, a detailed evaluation of the efficiency in the important three-space case is given.

The errors shown in Figs. 6(b)–6(d) are much larger than expected from pure attenuation of the sinusoidal waves, which can be estimated from Fig. 3(b). Notice that the discrepancy measured in l_1 norm between a sine wave and its shifted version with a small phase shift $\delta(\ll \pi)$ is linear in δ. We conclude that in HUY3 and HUY5 the main source of error is numerical dispersion. The high quality of HUY5 in comparison to 2-2CD and HUY3 is a consequence of a large reduction of numerical dispersion.

The stability of HUY3 and HUY5 has been studied numerically. HUY3 is unstable for a CFL number that is an element of [0.91, 1.0]. We restrict the value of the CFL number to the interval [0.0, 0.7]. For CFL numbers larger than 0.7, numerical dispersion becomes too large and the shape of an initially sharp pulse deteriorates rapidly. HUY5 is unstable for a CFL number that is an element of [0.81, 1.0], but again due to numerical dispersion we restrict the CFL number to interval [0.0, 0.75].

Finally, we will test the robustness of our higher-order scheme HUY5 solving a spatially inhomogeneous model problem

$$[\partial_t^2 - c^2(x)\partial_x^2]u(x,t) = 0. \tag{4.14}$$

The initial Gaussian pulse and the position-dependent velocity $c(x)$ are depicted in Fig. 7(a). The maximum (minimum) value for the velocity is $c = 1.0$ ($c = 0.5$). We use periodic boundary conditions. In the following, d_1 and d_2 denote the widths of the transition regions of the velocity of propagation measured in number of grid points, and w_1 the width of the initial Gaussian pulse. In Fig. 7(b) the result of 2-2CD with 1000 grid points ($d_1 = 20$, $d_2 = 100$, and $w_1 = 100$) is given. This result can be considered as the correct solution; higher resolution of space does not appreciably change the result. The result given by 2-2CD with 700 grid points ($d_1 = 14$, $d_2 = 70$, and $w_1 = 70$) is displayed in Fig. 7(c). This result is slightly different from that in Fig. 7(b). Figure 7(d) contains the result of HUY5 with 200 grid points ($d_1 = 4$, $d_2 = 20$, and $w_1 = 20$). The fine structure in Fig. 7(d) is as good as that in Fig. 7(c). Note that sharp pulses are generated during the propagation. For example, the right most pulse in Fig. 7(d) contains only seven grid points. The reduction in the number of grid points with HUY5 has not been optimized. A more detailed evaluation of propagation in inhomogeneous media will be given elsewhere.

As a conclusion of this section we can say that our best algorithm for wave propagation in one-space is HUY5. Even

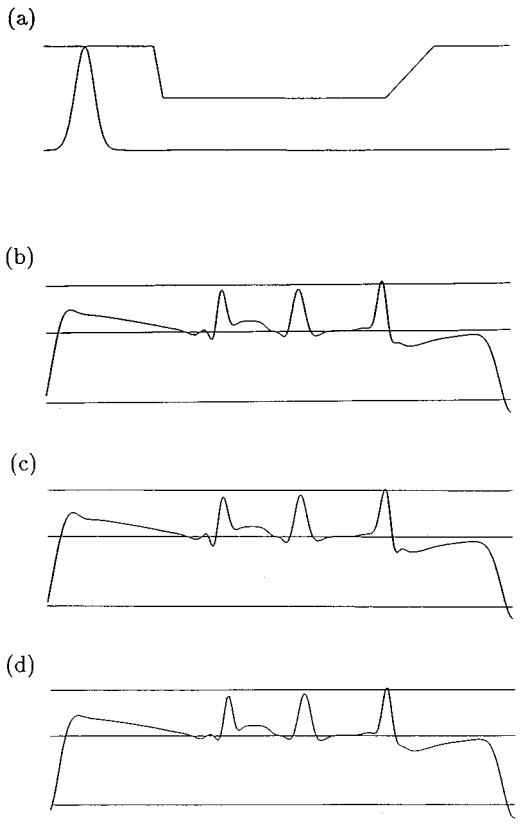

FIG. 7. Demonstration of the robustness of HUY5. Here Eq. (4.13) is solved with HUY5 and 2-2CD. (a) The Gaussian wave packet on the left is the initial condition. The function $c(x)$ is also plotted in this figure. Its upper (lower) level is 1 (0.5). (b) The result of 2-2CD with 1000 grid points and $\Delta x=0.005$ and $\delta t=0.005$. This result is regarded as the accurate solution. (c) 2-2CD with 700 grid points $\Delta x=0.007\,14$ and $\delta t=0.007\,14$. Already there are some errors compared to (b). (d) HUY5 with 200 grid points and $\Delta x=0.025$ and $\delta t=\Delta x/2$. The result is comparable to that of 2-2CD with 700 grid points (c). The line in (d) is slightly broken due to the small number of grid points. However, the details of the amplitude are correctly captured. Note that the right most pulse in (d) contains only seven points.

with one-tenth as many grid points per wavelength, HUY5 produces a smaller error than 2-2CD. The advantages of the new scheme become clear in higher dimensional spaces, where, as shown in Sec. VI, it requires 1–2 orders of magnitude fewer operations for solutions with the same error tolerance.

V. WAVE PROPAGATION IN TWO- AND THREE-SPACES

We demonstrate the effectiveness of our methodology using the lowest-order scheme. With this lowest-order demonstration we can already show stability and reduction numerical dispersion and anisotropy. A more optimal version based on a fifth-order interpolation on a staggered grid, an extension of HUY5, is planned to be evaluated in detail and applied to more challenging problems in a separate paper [2]. (However, see, Sec. VI.)

In three-space, map (4.1) can be written as spherical averages around each point in space [45]:

$$u(\mathbf{x},t+\delta t)=\frac{1}{4\pi c^2(\delta t)^2}\int_{|x-y|=c\,\delta t} d^3y[\,\delta t\,\partial_t u(\mathbf{y},t)+u(\mathbf{y},t)+\partial_\alpha u(\mathbf{y},t)cn_\alpha\delta t], \tag{5.1}$$

$$\partial_t u(\mathbf{x},t+\delta t)=\frac{1}{4\pi c^2(\delta t)^2}\int_{|x-y|=c\,\delta t} d^3y[\,\partial_t u(\mathbf{y},t)+\partial_\alpha\partial_t u(\mathbf{y},t)c\,\delta t n_\alpha+\partial_\alpha u(\mathbf{y},t)2cn_\alpha+\partial_\alpha\partial_\beta u(\mathbf{y},t)c^2n_\alpha n_\beta\delta t]. \tag{5.2}$$

Greek suffices denote spatial coordinates, and $\mathbf{n}$ is the outward normal to the surface of integration. The summation convention is implied. In this case the strong form of Huygens' principle holds. Map (5.1) is also known as the Green's representation formula for the wave equation.

To generate the third-order interpolants of $u_n(\mathbf{x},t)$ and $\partial_t u_n(\mathbf{x},t)$, the values of $u_n(\mathbf{x},t)$, $\partial_\alpha u_n(\mathbf{x},t)$, $\partial_t u(\mathbf{x},t)$, and $\partial_\alpha\partial_t u(\mathbf{x},t)$ are needed. The maps for $u(\mathbf{x},t)$, $\partial_\alpha u_n(\mathbf{x},t)$, $\partial_t u(\mathbf{x},t)$ and $\partial_\alpha\partial_t u(\mathbf{x},t)$, with the integrals scaled to the unit sphere S^2, are (σ is the surface element)

$$u(\mathbf{x},t+\delta t)=\frac{1}{4\pi}\int_{S^2} d\sigma[\,\delta t\,\partial_t u(\mathbf{x}+c\,\delta t\mathbf{n},t)+u(\mathbf{x}+c\,\delta t\mathbf{n},t)+n_\alpha\partial_\alpha u(\mathbf{x}+c\,\delta t\mathbf{n},t)c\,\delta t], \tag{5.3}$$

$$\partial_\alpha u(\mathbf{x},t+\delta t)=\frac{1}{4\pi}\int_{S^2} d\sigma[\,\delta t\,\partial_\alpha\partial_t u(\mathbf{x}+c\,\delta t\mathbf{n},t)+\partial_\alpha u(\mathbf{x}+c\,\delta t\mathbf{n},t)+\partial_\alpha n_\beta\partial_\beta u(\mathbf{x}+c\,\delta t\mathbf{n},t)c\,\delta t], \tag{5.4}$$

$$\partial_t u(\mathbf{x},t+\delta t)=\frac{1}{4\pi}\int_{S^2} d\sigma[\,\partial_t u(\mathbf{x}+c\,\delta t\mathbf{n},t)+n_\alpha\partial_\alpha\partial_t u(\mathbf{x}+c\,\delta t\mathbf{n},t)c\,\delta t+2cn_\alpha\partial_\alpha u(\mathbf{x}+c\,\delta t\mathbf{n},t)+n_\alpha n_\beta\partial_\alpha\partial_\beta u(\mathbf{x}+c\,\delta t\mathbf{n},t)c^2\delta t], \tag{5.5}$$

and

$$\partial_\alpha\partial_t u(\mathbf{x},t+\delta t)=\frac{1}{4\pi}\int_{S^2} d\sigma[\,\partial_\alpha\partial_t u(\mathbf{x}+c\,\delta t\mathbf{n},t)+\partial_\alpha n_\beta\partial_\beta\partial_t u(\mathbf{x}+c\,\delta t\mathbf{n},t)c\,\delta t+\partial_\alpha\partial_\beta u(\mathbf{x}+c\,\delta t\mathbf{n},t)2cn_\beta+\partial_\alpha\partial_\beta\partial_\gamma u(\mathbf{x}+c\,\delta t\mathbf{n},t)n_\beta n_\gamma c^2\delta t]. \tag{5.6}$$

We use the notations $u_n[ijk]=u(\mathbf{x}_{ijk},n\,\delta t)$, $\partial_\alpha u_n[ijk]=\partial_\alpha u(\mathbf{x}_{ijk},n\,\delta t)$ ($\partial_t u_n[ijk]=\partial_t u(\mathbf{x}_{ijk},n\,\delta t)$, $\partial_\alpha\partial_t u_n[ijk]=\partial_\alpha\partial_t u(\mathbf{x}_{ijk},n\,\delta t)$) for the values at time $t=n\,\delta t$ at the grid points $\mathbf{x}_{ijk}$. We proceed as follows. Assuming that we know

$u_n[ijk]$, $\partial_\alpha u_n[ijk]$, $\partial_t u_n[ijk]$, and $\partial_\alpha \partial_t u_n[ijk]$, we generate the interpolants for the functions $u_n(\mathbf{x},t)$ and $\partial_t u_n(\mathbf{x},t)$, which are then used to evaluate formulas (5.3)–(5.6). If we implement the algorithm on a single grid, then the update at a given point, done by evaluating Eqs. (5.3)–(5.6), requires in three-space the generation of interpolants in the interior of the eight cells that surround that point. If we use instead a staggered grid approach, with subgrids A and B, the nodes of subgrid A being at the center of the cells of subgrid B, and vice versa, then the evaluation of Eqs. (5.3)–(5.6) at a node of subgrid A, requires the evaluation of the interpolants in the interior of a single cell of subgrid B. In both cases it is possible to derive explicit maps of the forms

$$u_{n+1}[ijk]=HU_l[ijk]LU_l[ijk]+HUT_l[ijk]LUT_l[ijk], \tag{5.7}$$

$$\partial_{x_j} u_{n+1}[ijk]=KU_l[ijk]LU_l[ijk]+KUT_l[ijk]LUT_l[ijk], \tag{5.8}$$

$$\partial_t u_{n+1}[ijk]=MU_l[ijk]LU_l[ijk]+MUT_l[ijk]LUT_l[ijk], \tag{5.9}$$

$$\partial_{tx_j} u_{n+1}[ijk]=NU_l[ijk]LU_l[ijk]+NUT_l[ijk]LUT_l[ijk]. \tag{5.10}$$

Here summation convention is implied, and $LU_l[ijk]$ ($LUT_l[ijk]$) is a linear combination of $u_n[ijk]$ and $\partial_\alpha u_n[ijk]$, ($\partial_t u_n[ijk]$ and $\partial_\alpha \partial_t u_n[ijk]$). The coefficients HU_l, KU_l, MU_l, NU_l, HUT_l, KUT_l, MUT_l, and NUT_l depend only on the properties of the medium. The map for the single grid implementation involves nearest, next-, and next-next-nearest neighbors. In the staggered grid approach the map involves only nearest neighbors.

Applications to wave propagation in anisotropic media should be possible after numerically evaluating the appropriate Green's functions with, e.g., the method described in Ref. [44]. Our formulation is suitable for parallel implementations, because the solution at a given point depends only on the values at its surrounding points. We can obtain the two-space version of this algorithm assuming translational symmetry along one of the coordinate axes, i.e., the method of descent [45].

A possible strategy for applications to inhomogeneous media is to treat the terms including spatial derivatives of the parameters of the media as dependent sources, and then solve the resultant integral equation inside each cell. An alternative is to compute the local Green's function or use some approximation to it. This procedure might seem too costly at first sight, but the local problems are much simpler than the full problem. Complex geometries can be built through patching together simple pieces. We can expect that, because of the reduction to simpler problems, some analytic results may be incorporated into the algorithm. Further research is needed to determine the best way to model inhomogeneous media in the framework of the HUY algorithms.

To demonstrate the reduction in numerical dispersion and numerical anisotropy, we run a simple two-space test. On a grid of 200×200 points with $\Delta x=\Delta y=1.0$, $\delta t=0.2\Delta x$, and $c=1.0$, we excite the center point at position $x=100$ and $y=100$ with a Gaussian excitation in the following way:

$$\partial_t u_{(k+1)}[100,100]=\partial_t u_k[100,100]+2e^{-[(k-80)\delta t)^2/6]}, \tag{5.11}$$

where k is an integer number that indicates the number of time steps.

In Fig. 8(a) we present a contour plot of the solution with nonstaggered HUY3 after 500 time steps with a CFL number equal to 0.2. In Fig. 8(b) the contour plot of the result of staggered HUY3 after 500 time steps and a CFL number equal to 0.2 is given. Figure 8(c) contains the contour plot of the result of the equivalent problem solved with 2-2CD after 200 time steps and a CFL number equal to 0.5. We use a larger time step with 2-2CD because it is known that 2-2CD works better with δt nearer the stability limit. The contour lines should be perfect circles, but the contours produced by 2-2CD are clearly distorted. The distortion is a consequence of numerical anisotropy, which is one of the problems of 2-2CD, as we mentioned in Sec. III. No anisotropic distortions can be seen in the results obtained with both versions of HUY3.

For the same input (5.11), we compare the results of the three schemes with the exact solution which can be obtained from the convolution of Eq. (5.11) and the two-space impulse response. The comparison is done at a cross section parallel to the x axis with $y=100$. In Fig. 9(a) the comparison between the nonstaggered HUY3 and the exact solution is presented. It is seen that the solution computed with the nonstaggered HUY3 deviates from the exact solution near the propagating front. This error is due to overshooting in the interpolation [14]. In Fig. 9(b) we show the comparison between the exact solution and staggered HUY3. There is an undulation on the internal side of the front caused by numerical dispersion. For comparison the same test with 2-2CD is shown in Fig. 9(c). The distortion due to numerical dispersion is very clear. The best results are obtained with staggered HUY3; however, this algorithm is not superior to the 2-2CD, because, in agreement with the one-space analysis, the improvement with HUY3 is not enough to compensate for the additional cost due to the larger number of operations. How the staggered version of staggered HUY5 can solve the problem may be glimpsed from Sec. VI. A full description of HUY5 is planned to be presented in a separate paper.

We also computed the solution for a point source in front of an interface that separates two media with different propagation velocities in the case when the amplitude and its normal derivative are continuous across the interface. The exact solution for this problem can be obtained with, e.g., the Cagniard–De Hoop method [46].

The solution with HUY3 was obtained by implementing Huygens' principle on both sides of the interface. This is equivalent to assuming that the interface is located between two nodal planes of the grid. The treatment of the interface is approximate because, even though the solution is computed correctly on both sides of the interface, the interpolant generated across the interface only approximately represents the effect of the interface. Further research to improve beyond this approximation is being undertaken.

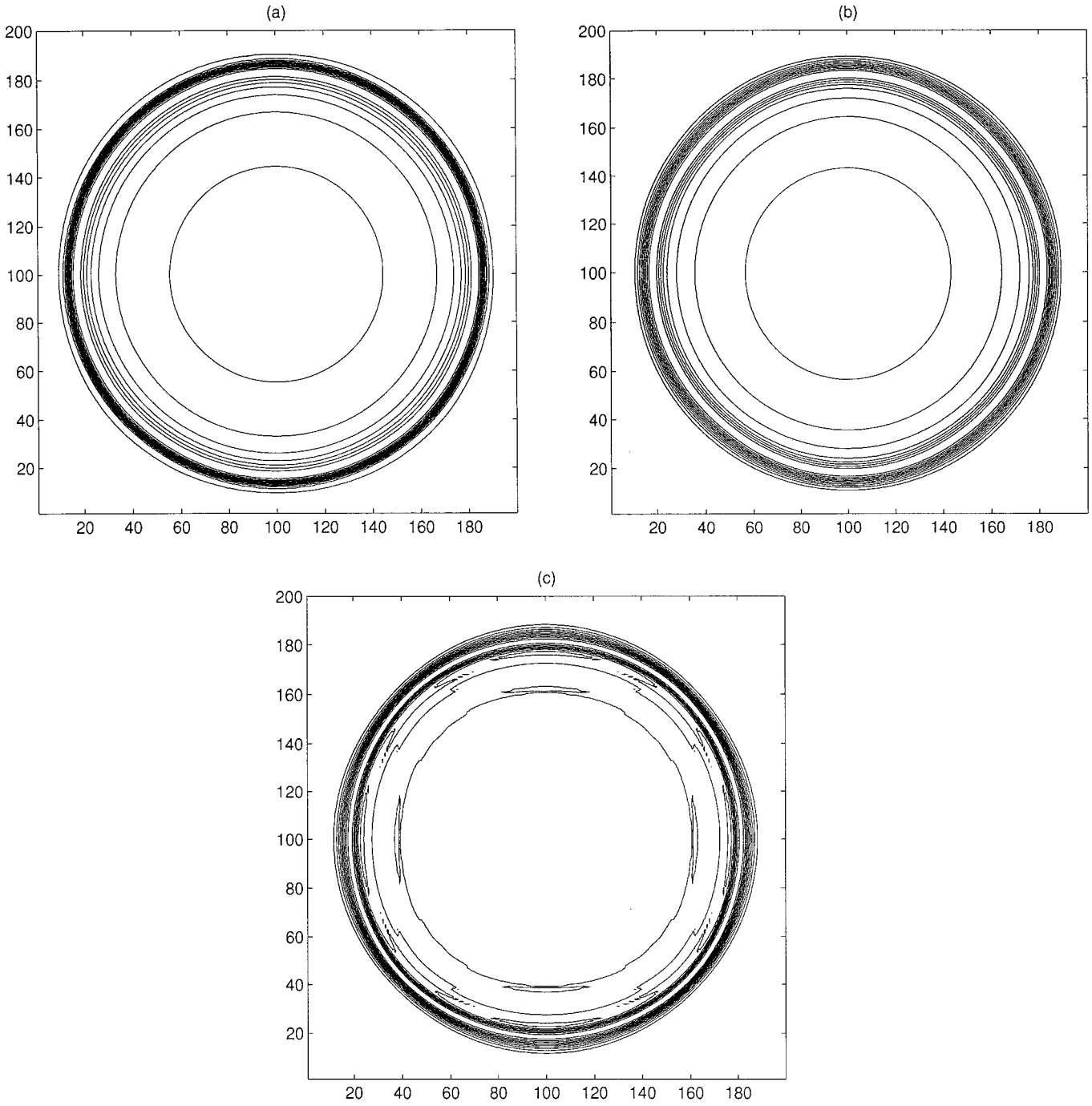

FIG. 8. Numerical anisotropy. A comparison between HUY3 and 2-2CD. Ten level contour plots of the results of the propagation with (a) the nonstaggered HUY3 after 500 time steps with a CFL number equal to 0.2, (b) the staggered HUY3 after 500 time steps with a CFL number equal to 0.2, and (c) 2-2CD after 200 time steps with a CFL number equal to 0.5.

We use a grid of 200×200 points with the interface located at position $y=100.25$, and a Gaussian excitation located at $x=100$ and $y=110$, as

$$\partial_t u_{(k+1)}[100,110]=\partial_t u_k[100,110]+0.5e^{-[((k-80)\delta t)^2/24]}. \tag{5.12}$$

In Fig. 10(a) the result is compared with the exact solution at positions $x=100$ and $y=120$. The same problem solved with the 2-2CD algorithm is shown in Fig. 10(b). The improvement with respect to 2-2CD is not significant for the same reasons stated above.

In our tests we have found that the three-space (two-space) version of the nonstaggered HUY3 is stable for a CFL number less than or equal to 0.11 (less than or equal to 0.2). Also from numerical tests, the stability limit of staggered HUY3 has been found to be a CFL number equal to 0.5, independent of dimensionality.

The idea of implementing Huygens' principle with the aid of interpolation is not limited to Cartesian grids. The algo-

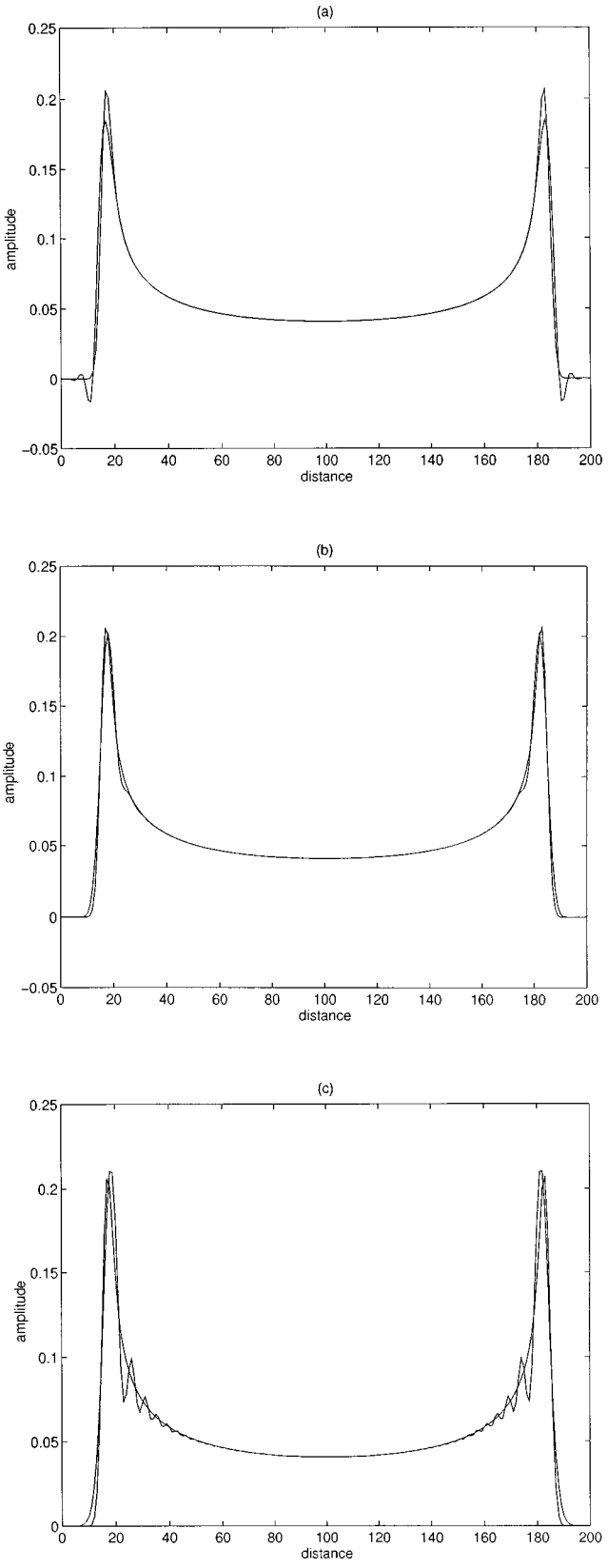

FIG. 9. Numerical dispersion. A comparison between HUY3 and 2-2CD. The signal is measured along the line $y=100$. Results obtained with (a) the nonstaggered HUY3 with a CFL number equal to 0.2, (b) staggered HUY3 with a CFL number equal to 0.2, and (c) 2-2CD with a CFL number equal to 0.5.

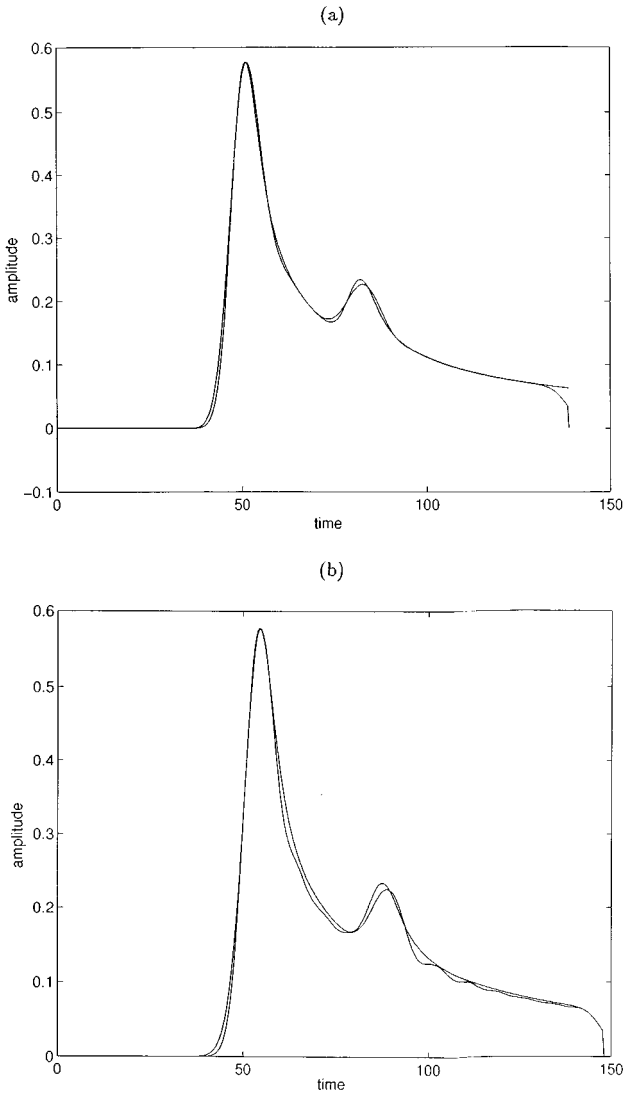

FIG. 10. Interface. Comparison with the exact solution. The signal is measured at positions $x=100$ and $y=120$. Results obtained with (a) the staggered HUY3 with a CFL number equal to 0.2 compared with the exact solution. (b) 2-2CD with a CFL number equal to 0.5 compared with the exact solution.

rithm should work equally well on grids of arbitrary shape as long as we have a good quality interpolation over which to implement Huygens' principle.

VI. EFFICIENCY OF HUY5 IN THREE-SPACE

In three-space the number of independent derivatives up to second order is 9. For the wave equation we need to keep track of two functions, the amplitude and its time derivative. Therefore, HUY5 (HUY3) in three-space requires 20 (eight) variables per grid point. If we straightforwardly extend the nonstaggered method discussed up to this point, the nonstaggered HUY5 would require 670 multiplications and 1600 additions, 2300 total. These values are to be compared with eight additions and three multiplications in the 2-2 CD algorithm. Notice that, even with this large difference in the number of operations per grid point, the nonstaggered HUY5 can be made more efficient than the 2-2 CD thanks to the

reduction in the number of grid points by a factor of 10, as shown in the one-space case. With this factor, the reduction in the number of grid points is 1000. Furthermore with a CFL number equal to 0.1, the larger δt gives an additional factor of 2 for a total of 2000, i.e., approximately ten times faster, with a memory requirement 100 times smaller than the 2-2 CD.

The use of a staggered grid considerably improves the efficiency of the HUY5. In the staggered mesh scheme the values at the grid points (i,j,k) are used to compute new values at positions on the staggered grid at $(i+\frac{1}{2},j+\frac{1}{2},k+\frac{1}{2})$, and vice versa. There are three main advantages. First, the unit cell for the staggered computation has only eight points, which permits a reduction in the number of operations per step by a factor of approximately 2. Second, the symmetry of the homogeneous space is better represented, because all the neighbors used in every update are equally weighted. Lastly, the stability limit is increased considerably from a CFL number equal to 0.1 to one equal to 0.4. The staggered HUY5 requires a total of about 1000 operations, less than one half the number required for the nonstaggered scheme. Notice that if we assume a reduction in the number of points by a factor of 4 in each direction, with a CFL number equal to 0.3, HUY5 and 2-2 CD have comparable CPU efficiencies, with HUY5 requiring about one-third the memory. With a reduction in the number of points by a factor of 6 (10) and a CFL number equal to 0.3, there is an efficiency gain factor of 7 (60) over the 2-2 CD algorithm. In addition, from the parallel computation point of view, HUY5 has more to gain, because a large part of the operations in each grid point can be done in parallel without any communication cost. Figure 11 shows a result of HUY5 for the same test of Sec. V. It is seen that the wave front can be captured with one cell with no visible distortion due to numerical dispersion. The continuous line corresponds to the exact solution obtained as a convolution of the impulse response and a Gaussian with standard deviation $\sigma=\Delta x/2$.

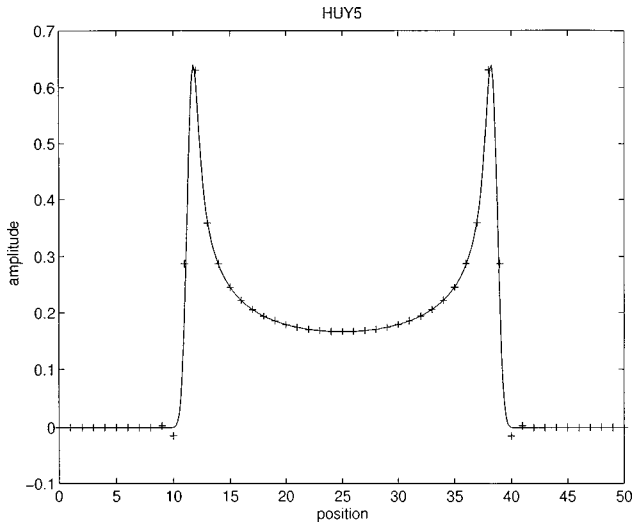

FIG. 11. HUY5 result, in a grid of 50×50 points, with the excitation at (25,25). The signal is measured along the line $y=25$. The exact result is given by the continuous line, and numerical result is indicated with (+).

VII. SUMMARY AND FINAL REMARKS

We have demonstrated the possibility of constructing efficient numerical solvers for hyperbolic PDE's by capturing the crucial physics, e.g., the symmetries of space-time implicit in Huygens' principle for the wave equation. A detailed evaluation of the one-space algorithms demonstrates a large reduction of numerical dispersion as the order of the interpolation increases. As a matter of fact, in one-space the algorithm HUY5, based on a fifth-order interpolation, is more accurate than the popular 2-2 CD algorithm even with one-tenth as many grid points per wavelength. In HUY5 the number of operations per node is higher, but the reduction in the number of grid points can make it significantly more efficient. In two- and three-spaces, HUY3, based on a third-order interpolation, has been explained in detail to demonstrate that our approach is not confined to one-space. A two-space test shows that numerical anisotropy is reduced. A comparison of the results of the two-space test with the exact solution, obtained as a convolution between the input and the two-space Green's function, demonstrates that the staggered algorithm is the most accurate. A preview of the result of staggered HUY5 has been added to show that with this algorithm sharp fronts can be captured with a single grid point. The improvement over standard algorithms is especially significant when the wave is a short pulse.

Efficiency gain for wave propagation in homogeneous space is usual with standard high-order FDTD algorithms. For conventional schemes, high order is achieved by enlarging the numerical stencil. In contradistinction, the numerical stencil used by the staggered HUY algorithms includes nearest neighbors only, independent of the order of the interpolation used.

The algorithm can also be applied to anisotropic media once the appropriate Green's functions have been computed. This can be accomplished with the aid of, e.g., the method described in Ref. [44]. The idea of using interpolation to implement Huygens' principle can be extended to nonstructured grids in a general geometry.

In one-space, we have used HUY5 to solve a simple wave propagation problem with a position dependent velocity. In the presence of sharp spatial variations in the medium, the solution remains stable and accurate. In two-space the problem of a point source in front of an interface between two media has been computed with the staggered HUY3. A possible strategy for the general case is to treat the terms with spatial derivatives of the parameters of the media as dependent sources, and then solve the resultant integral equation inside each cell. The solution under the assumption of local constant velocity should give a first approximation. It might be possible to improve beyond this approximation by, in the worse case, computing the local Green's functions. Further research has been undertaken to determine the most appropriate way to model material inhomogeneities within the framework of the HUY algorithms.

We consider the wave equation solvers presented in this paper as a first step toward the development of solvers for more complicated problems, e.g., pulse propagation in inhomogeneous or anisotropic materials with complicated boundary conditions or propagation of short pulses in nonlinear materials, both, situations in which space-time methods are superior to frequency-domain methods [47]. Also note that

the interpolation-resampling strategy is an ideal means to realize nonperturbative renormalization-group methods, allowing continuous scaling [48].

ACKNOWLEDGMENTS

The authors gratefully acknowledge very useful conversations with Andrei Malevsky (Refs. [12,13]). Mohan Ramamurthy kindly told the authors about the wide use of the semi-Lagrangian method in the NWP community. We thank Takashi Yabe for useful correspondences and for sending his work prior to publication. For useful conversations related to wave equation solvers and Refs. [26,29], the authors thank Weng Cho Chew. For Ref. [37] the authors thank Gerald Hedstrom. Nigel Goldenfeld informed us of Ref. [18]. Y.O. thanks the Institute Mittag-Leffler (Sweden) for its hospitality, and Nigel Goldenfeld for useful conversations there. The work was supported in part by National Science Foundation Grant No. NSF DMR 93-14938. L.S.M. was supported in part by National Science Foundation Grant No. NSF DMR 89-20538 which was administered through the University of Illinois Materials Research Laboratory and Air Force Office of Scientific Research (Grant No. F49620-96-10025).

APPENDIX A: FIFTH-ORDER HERMITE INTERPOLATION IN ONE-SPACE

The fifth-order interpolant $F_i(x)$ of $f(x)$, valid for $[x_i,x_{i+1}]$, is

$$F_i(x)=C_5[i]X^5+C_4[i]X^4+C_3[i]X^3+C_2[i]X^2+C_1[i]X+C_0[i], \tag{A1}$$

where $X=x-x_i$ and the coefficients $C_j[i]$ $(j=1,...,5)$ are determined by the conditions of continuity of the interpolant $F_i(x)$ and its first and second derivatives at the end points of the interval $[x_i,x_{i+1}]$. Thus we have $C_0[i]=f[i]$, $C_1[i]=f'[i]$, $C_2[i]=f''[i]/2$, and

$$C_5[i]=\frac{6(f[i+1]-f[i])}{(\Delta x)^5}-\frac{3(f'[i+1]+f'[i])}{(\Delta x)^4}+\frac{f''[i+1]-f''[i]}{2(\Delta x)^3}, \tag{A2}$$

$$C_4[i]=\frac{15(f[i]-f[i+1])}{(\Delta x)^4}+\frac{8f'[i]+7f'[i+1]}{(\Delta x)^3}+\frac{3f''[i]-2f''[i+1]}{2(\Delta x)^2}, \tag{A3}$$

$$C_3[i]=\frac{10(f[i+1]-f[i])}{(\Delta x)^3}-\frac{4f'[i+1]+6f'[i]}{(\Delta x)^2}+\frac{f''[i+1]-3f''[i]}{2\Delta x}. \tag{A4}$$

The resampling formulas analogous to formulas (2.6) and (2.7) are

$$f_{n+1}[i]=F_i(x_i-c\,\delta t)=f_n[i]+[[\{(C_5[i]\xi+C_4[i])\xi+C_3[i]\}\xi+f_n''[i]/2]\xi+f'[i]]\xi, \tag{A5}$$

$$f'_{n+1}[i]=\partial_x F_i(x_i-c\,\delta t)=f'_n[i]+[\{(5C_5[i]\xi+4C_4[i])\xi+3C_3[i]\}\xi+f''[i]]\xi, \tag{A6}$$

$$f''_{n+1}[i]=\partial_x^2 F_i(x_i-c\,\delta t)=f''_n[i]+\{(20C_5[i]\xi+12C_4[i])\xi+6C_3[i]\}\xi, \tag{A7}$$

where $\xi=-c\,\delta t$. This expression is valid for $c<0$. For $c\geqslant 0$ the equivalent expression is obtained by replacing Δx with $-\Delta x$, and $i+1$ with $i-1$ in Eqs. (A2), (A3), and (A4).

APPENDIX B: THIRD-ORDER HERMITE INTERPOLATION IN TWO- AND THREE-SPACES

In two-space the third-order Hermite interpolant $F_{ij}(\mathbf{x})$ for $f(\mathbf{x})$ in the first quadrant $(x,y)\in[x_i,x_{i+1}]\times[y_j,y_{j+1}]$ is defined as

$$F_{ij}(x)=B_1[ij]X^3+B_2[ij]Y^3+B_3[ij]X^2Y+B_4[ij]XY^2+B_5[ij]X^2+B_6[ij]Y^2+B_7[ij]XY+\partial_x f[ij]X+\partial_y f[ij]Y+f[ij], \tag{B1}$$

where $X=x-x_i$ and $Y=y-y_i$. The seven constants B_i are solved from the conditions of continuity of f at $(i+1,j)$, $(i,j+1)$, and $(i+1,j+1)$ and continuity of $\partial_x f$ and $\partial_y f$ at $(i+1,j)$ and $(i,j+1)$. The interpolation for the other quadrants has the same form with different sets of constants. The conditions of continuity are imposed at the corners of the quadrants.

In hydrodynamic applications the direction of the velocity determines which of the surrounding neighbors are to be used to implement advection correctly. In two-space the signs of u_x and u_y determine one of the possible four quadrants; in three-space the signs of u_x, u_y, and u_z determine one of the possible eight octants. A special step in the algorithm has to be introduced to make this selection. For the wave equation, all the surrounding octants contribute, so that no selection step is needed to implement Huygens' principle.

In three-space the third-order interpolant $F_{ijk}(\mathbf{x})$ for $f(\mathbf{x})$ defined in the first octant $(x,y,z)\in[x_i,x_{i+1}]\times[y_j,y_{j+1}]\times[z_k,z_{k+1}]$ is

$$\begin{aligned}F_{ijk}(\mathbf{x})=&C_1[ijk]X^3+C_2[ijk]Y^3+C_3[ijk]Z^3\\&+C_4[ijk]X^2Y+C_5[ijk]XY^2+C_6[ijk]X^2Z\\&+C_7[ijk]XZ^2+C_8[ijk]Y^2Z+C_9[ijk]YZ^2\\&+C_{10}[ijk]XYZ+C_{11}[ijk]X^2+C_{12}[ijk]Y^2\\&+C_{13}[ijk]Z^2+C_{14}[ijk]XY+C_{15}[ijk]XZ\\&+C_{16}[ijk]YZ+\partial_x f[ijk]X+\partial_y f[ijk]Y\\&+\partial_z f[ijk]Z+f[ijk],\end{aligned} \tag{B2}$$

where $X=x-x_i$, $Y=y-y_i$ and $Z=z-z_i$. The 16

constants C_i are solved from the conditions of continuity of f at $(i+1,j,k)$, $(i,j+1,k)$, $(i,j,k+1)$, $(i+1,j+1,k)$, $(i+1,j,k,+1)$, $(i,j+1,k+1)$, and $(i+1,j+1,k+1)$, and continuity of $\partial_x f$, $\partial_y f$, and $\partial_x f$ at $(i+1,j,k)$, $(i,j+1,k)$, and $(i,j,k+1)$. The interpolation for other octants has the same form. The coefficients are solved from the conditions of continuity at the corners of those octants. For more details, see Ref. [14].

[1] We use Huygens' principle in the generalized sense, which applies to any dimension. This is not the strict Hadamard sense, sometimes also called strong Huygens' principle, which is valid only in odd dimensions larger than one.

[2] L. San Martin (unpublished).

[3] Y. Oono and S. Puri, Phys. Rev. Lett. **58**, 836 (1987).

[4] N. Parekh and S. Puri, J. Phys. A **23**, L1085 (1990).

[5] See, e.g., A. Shinozaki and Y. Oono, Phys. Rev. E **48**, 2622 (1993); M. Zapotocky, P. M. Goldbart, and N. Goldenfeld, *ibid.* **51**, 1216 (1995).

[6] K. G. Wilson, Phys. Rev. D **10**, 2445 (1974).

[7] Y. Oono and A. Shinozaki, Forma **4**, 75 (1989) (this is an early CDS review).

[8] N. T. Krishnamurty, J. Appl. Meteorol. **1**, 508 (1962).

[9] A. Staniforth and J. Cote, Mon. Weather Rev. **119**, 2206 (1991).

[10] H. Ritchie, C. Temperton, A. Simmons, M. Hortal, T. Davis, D. Dent, and M. Hamrud, Mon. Weather Rev. **123**, 489 (1995).

[11] P. L. Viollet, J. P. Benque, and J. Goussebaile, Nucl. Sci. Eng. **84**, 350 (1983).

[12] A. V. Malevsky and D. A. Yuen, Phys. Fluids A **3**, 2105 (1991).

[13] A. V. Malevsky (unpublished).

[14] T. Yabe and T. Takei, J. Phys. Soc. Jpn. **57**, 2598 (1988); T. Yabe and T. Aoki, Comput. Phys. Commun. **66**, 219 (1991); **66**, 232 (1991).

[15] T. Yabe and P.-Y. Wang, J. Phys. Soc. Jpn. **60**, 2105 (1991); T. Yabe, H. Hoshino, and T. Tsuchiya, Phys. Rev. A **44**, 2756 (1991); T. Yabe and F. Xiao, J. Phys. Soc. Jpn. **62**, 2537 (1993); T. Yabe, F. Xiao, and T. Mochizuki, Nucl. Eng. Des. **155**, No. 1-2, 45 (1995).

[16] In order to describe a continuous function f accurately, the grid mesh has to be fine enough to capture the details of f. For example, the sampling theorem tells us that the sampling interval in space should not be larger than one-half of the minimum wavelength of the Fourier components of f to avoid a loss of information (sort of a coarse graining of f).

[17] C. de Boor, *A Practical Guide to Splines* (Springer-Verlag, New York, 1978).

[18] A. I. Tolstykh, *High Accuracy Non-Centered Compact Difference Schemes for Fluid Dynamics Applications* (World Scientific, Singapore, 1994), p. 8.

[19] The interpolation-resampling strategy for translation, however, does not have any restriction in the value of the CFL number; it works correctly as long as we sample the correct point on the interpolant of f. In Eqs. (2.8) and (2.9), the CFL number has to be less than 1 because the interpolant is generated in the region between f_i and its nearest neighbors. In this paper we will work only with CFL numbers smaller than unity, because that is the case in the application to wave propagation. All the coefficients can be partially or completely precomputed to reduce the CPU time of simulation; with the use of look-up tables for the coefficients as functions of CFL the CPU time for each step in the one-space advection problem (2.1) can be reduced by approximately 50% without sacrificing accuracy. This one step implementation is clearly advantageous for parallel implementation.

[20] Higher order interpolations are often avoided because they can suffer from pathological oscillations of the interpolant. Although for our schemes up to ninth order applied to Eq. (2.1) the pathological effect (if any) is not important, in more complicated problems the pathological oscillation could be a serious inconvenience. In our scheme, however, the presence of this unwanted oscillation can be detected easily with the aid of Sturm's theorem [21], which allows us to count the number of zeros of the first derivative. If this number is larger than 2, we can rely on one of the following strategies: (1) to switch to a lower-order interpolation scheme, (2) to filter the original polynomial to remove the pathological behavior, or (3) to remesh the grid locally. The third strategy is a kind of adaptive grid scheme that can easily be incorporated thanks to the use of interpolation.

[21] Mathematical Society of Japan, *Encyclopedic Dictionary of Mathematics* (MIT Press, Boston, 1980), p. 39.

[22] Although we do not include the results, we also investigated further higher order interpolation schemes for this one-space pure translation problem. As we mentioned above, the growth rate of the error of the fifth-order scheme is more than two orders of magnitude smaller than that of the third-order scheme. Similar improvements were obtained when going from fifth- to seventh- and from seventh- to ninth-order interpolations. We could try to obtain better efficiency with the higher order schemes by reducing the number of grid points as we did with the fifth order interpolation for the pure advection problem. However, the maximum size of the step is often determined by nonadvective terms or spatial inhomogeneity of the system, so we feel (unless adaptive grid schemes are used) that practical schemes are based on the third- and fifth-order interpolations.

[23] The reader may question the legitimacy of considering higher order derivatives, because even the classical solution of Eq. (2.1) requires differentiability of the solution. Recall an obvious fact that no one can distinguish nondifferentiable functions and C^∞ functions from their values at the grid points only. Therefore we are free to mimic the points sampled from a (possibly) nondifferentiable function by those from a sufficiently differentiable but rapidly changing function.

[24] F. Xiao and T. Yabe, Shock Waves **4**, 101 (1994); T. Yabe and F. Xiao, Comput. Math. Appl. **29**, 15 (1995).

[25] W. C. Chew, *Waves and Fields in Inhomogeneous Media* (Van Nostrand Reinhold, New York, 1990).

[26] D. Kosloff and E. Baysal, Geophysics **47**, 1402 (1982).

[27] J. M. Carcione D. Kosloff, and R. Kosloff, Mech. Appl. Math. **41**, 319 (1988).

[28] B. Yang, D. Gottlieb, and J. S. Hesthaven, J. Comput. Phys. **134**, 216 (1997).
[29] Q. H. Liu, Microw. Opt. Technol. Lett. **15**, 158 (1997).
[30] Q. H. Liu, *IEEE Antennas and Propagation Society International Symposium* (IEEE Press, Piscataway, NJ, 1997), Vol. 1, p. 122.
[31] Curtis L. Card and Myron B. Allen, Numer. Methods. Par. Diff. Eq. **11**, 127 (1995); G. Cohen, P. Joly, and N. Tordjman, in *Second International Conference on Mathematical and Numerical Aspects of Wave Propagation*, edited by Ralph Kleinman (SIAM, Philadelphia, PA, 1993).
[32] R. S. Schechter, H. H. Chaskelis, R. B. Mignona, and P. P. Delsanto, Science **265**, 1188 (1994).
[33] K. Kunz and R. Luebbers, *The Finite Difference Time Domain Method for Electromagnetics* (CRC Press, Boca Raton, FL, 1993).
[34] M. A. Dablain, Geophysics **51**, 54 (1986).
[35] G. Cohen and P. Joly, Comput. Methods Appl. Mech. Eng. **80**, 397 (1990).
[36] C. W. Manry, S. L. Broschat, and J. B. Schneider, J. Appl. Comput. Electromagnetics Soc. **10**, 17 (1995).
[37] Alain Sei and W. W. Symes, Appl. Numer. Math. **15**, 465 (1994).
[38] Michel Barbiera and Gary Cohen, in *Second International Conference on Mathematical and Numerical Aspects of Wave Propagation* (Ref. [31]).
[39] Mark H. Carpenter, David Gottlieb, and Saul Abarbanel, J. Comput. Phys. **111**, 220 (1994).
[40] Alan R. Levander, Geophysics **53**, 1425 (1988).
[41] W. C. Chew and C-C. Lu, IEEE Trans. Antennas Propag. **41**, 897 (1993).
[42] K. S. Yee, J. S. Chen, and A. H. Chang, IEEE Trans. Antennas Propag. **40**, 1068 (1992).
[43] When we use the initial condition F, two wave packets propagate: one to the right, the other to the left. While they travel, both are distorted asymmetrically, but when they meet at the initial position due to the periodic boundary conditions, the asymmetric distortions are partially canceled. To measure the full error, we eliminate the wave packet traveling to the right, and measure the error between the left traveling wave packet and the initial condition divided by 2.
[44] C. Y. Wang and J. D. Achenbach, Wave Motion **18**, 273 (1993).
[45] F. John, *Partial Differential Equations* (Springer-Verlag, New York, 1982), Chap. 5.
[46] K. Aki and P. Richards, *Quantitative Seismology* (Freeman, San Francisco, 1980), p. 234.
[47] G. P. Agrawal, *Nonlinear Fiber Optics* (Academic, New York, 1995), pp. 54 and 55.
[48] N. Goldenfeld, O. Martin, and Y. Oono, J. Sci. Comput. **4**, 355 (1989); J. Bricmont, A. Kupiainen, and G. Lin, Commun. Pure Appl. Math. **47**, 893 (1994); L.-Y. Chen, N. Goldenfeld, and Y. Oono, Phys. Rev. E **54**, 376 (1996).