A fast method for linear waves based on geometrical optics

Stolk, C.C.

Published in:
SIAM journal on numerical analysis

DOI:
10.1137/070698919

SIAM J. NUMER. ANAL.
Vol. 47, No. 2, pp. 1168–1194

A FAST METHOD FOR LINEAR WAVES BASED ON GEOMETRICAL OPTICS*

CHRISTIAAN C. STOLK†

Abstract. We develop a fast method for solving the one-dimensional wave equation based on geometrical optics. From geometrical optics (e.g., Fourier integral operator theory or WKB approximation) it is known that high-frequency waves split into forward and backward propagating parts, each propagating with the wave speed, with amplitude that is slowly changing depending on the medium coefficients, under the assumption that the medium coefficients vary slowly compared to the wavelength. Based on this we construct a method of optimal, $O(N)$ complexity, with basically the following steps: 1. decouple the wavefield into an approximately forward and an approximately backward propagating part; 2. propagate each component explicitly along the characteristics over a time step that is small compared to the medium scale but can be large compared to the wavelength; 3. apply a correction to account for the errors in the explicit propagation; repeat steps 2 and 3 over the necessary amount of time steps; and 4. reconstruct the full field by adding forward and backward propagating components again. Due to step 3 the method accurately computes the full wavefield. A variant of the method was implemented and outperformed a standard order (4,4) finite difference method by a substantial factor. The general principle is applicable also in higher dimensions, but requires efficient implementations of Fourier integral operators which are still the subject of current research.

Key words. wave equation, numerical method, multiscale method, geometrical optics, integrating factor

AMS subject classifications. 65M25, 76Q05

DOI. 10.1137/070698919

1. Introduction.

Consider waves propagating in an inhomogeneous medium which varies slowly on the scale of the wavelength. In this case, waves behave much like in the constant coefficient case. For example, in one dimension an initial pulse approximately splits into a forward propagating pulse and a backward propagating pulse, each propagating with the wave speed, and with slowly varying amplitude. Indeed for small times, the wave "sees" only a small, approximately constant part of the medium. This can be made precise using WKB, or geometrical optics theory, or the more general and advanced theory of Fourier integral operators. One finds that the above picture is true in the limit for high-frequency waves; these have the just described relatively simple interaction with the medium. For the low-frequency part the interaction with the medium is of course more complicated; e.g., reflections occur.

Simulating high-frequency waves using finite differences or finite elements is notoriously expensive, especially in three dimensions. One reason for this is the large number of time steps that is generally needed, since in conventional methods the time step is bounded by the space discretization length. In one dimension this leads to cost at least $O(N^2)$ if N is the number of space discretization points. This on the one hand is quite understandable: The wavefield is computed over a finite part of the (x,t)-plane with resolution $1/N$ in both the x and the t direction. On the other hand,

*Received by the editors August 1, 2007; accepted for publication (in revised form) November 17, 2008; published electronically February 19, 2009. This research was supported by the Netherlands Organisation for Scientific Research through VIDI grant 639.032.509. This work was done while the author was employed at the University of Twente.
http://www.siam.org/journals/sinum/47-2/69891.html
†Korteweg-de Vries Institute for Mathematics, University of Amsterdam, 1018 TV Amsterdam, The Netherlands (C.C.Stolk@uva.nl).

if we are interested only in the map from initial to final values, one can argue that there is room for improvement: The high frequencies are well described by translation and scaling over quantities that follow from the smoothly varying medium. The low frequencies still need to be computed by some discretization, but with a coarse grid. In this paper we will show that in fact we can devise a scheme that follows this pattern and is of complexity $O(N)$, i.e., optimal.

The observation about the high cost of simulating high-frequency waves is not new, and various authors have sought to deal with this, e.g., [12] in one dimension, [2, 9] in higher dimensions. The paper [12] uses the observation that the matrix that describes the propagator $P(t)$ (the operator exponent e^{tM} in the notation below, that maps initial values at time 0 to values at a later time t, assuming time-independent coefficients) can be compressed by wavelet compression. High-frequency signals in the propagator are concentrated around the characteristics. Low-frequency signals are not. Due to the separation in space and scale that is obtained using wavelets, this leads to many small entries that, if omitted, give only a small error to the matrix. The matrix is compressed in this way, and it becomes possible to store it. The operator exponent is then first computed for a small time τ, and subsequently for longer times by repeated squaring $P(2\tau) = P(\tau)^2$, $P(4\tau) = P(2\tau)^2$, etc. Unlike our method this idea is restricted to time-independent coefficients. Curvelet frames [15, 4] have been proposed to extend this idea to multiple dimensions.

In this paper we introduce a new, different concept to reduce computational cost. We explicitly separate forward and backward propagating parts of the waves, as made possible by high-frequency asymptotic theory, and propagate these explicitly. No matrix compression is used. Roughly speaking the method involves the following steps, that are repeated over a number of time steps to obtain the final result:

1. Decouple the wavefield into a forward and a backward propagating part, like for the constant coefficient medium where we can find two functions F and B such that the solution is given by $U_1(x,t) = B(x+ct) + F(x-ct)$.
2. Propagate each component explicitly over a time step that is small compared to the medium scale but large compared to the wavelength.
3. Apply a correction to account for the errors in the explicit propagation.
4. Reconstruct the full field by adding forward and backward propagating components again.

For higher dimensions one could perhaps devise a similar scheme; however, at this point in time it is not clear how to efficiently compute the Fourier integral operators needed in step 2.

Two methods according to this outline will be described. First we derive a relatively straightforward method, that is implemented numerically and tested. The goal of this is to get a first impression of what kind of numerical results can be obtained. Compared with an order (4,4) finite difference method we find improvements in speed of factors up to 20, depending on the smoothness of the medium.

A second method is derived using several more innovations, in particular a new multiscale time-stepping method; see section 6 and thereafter. For this method we study error estimates and the complexity, and we show that it has *optimal* $O(N)$ *complexity*. The $O(N)$ complexity is better than that in [12], but we also have another improvement compared to the repeated squaring method, namely that our method is also applicable in media with time-dependent coefficients.

Let us discuss in more mathematical terms the ideas behind the method. We consider the one-dimensional acoustic wave equation

$$(\partial_t \circ a(x,t)\partial_t - \partial_x \circ b(x,t)\partial_x)U_1(x,t) = 0, \tag{1.1}$$

with domain given by a circle Ω of integer length L. It will be convenient to write this as a first-order system; let

$$U_2 = a\partial_t U_1, \qquad U = \begin{pmatrix} U_1 \\ U_2 \end{pmatrix}, \qquad M = \begin{pmatrix} 0 & a^{-1} \\ \partial_x \circ b\partial_x & 0 \end{pmatrix}. \tag{1.2}$$

Then (1.1) becomes

$$\frac{d}{dt}U = MU. \tag{1.3}$$

We view this as an ODE with values in a function space, which explains the notation $\frac{d}{dt}$ in this equation.

We are interested in the initial value problem where $U(t_0) = U_0$ is given and $U(t_1)$ is to be determined. The natural space to consider the equation is $U(t) \in H^1 \times L^2$, where $H^s = H^s(\Omega)$ denotes the Sobolev space of order s. With coefficients that are $C^{k,1}$ in space, and with time derivative that is also $C^{k,1}$ in space, there is existence, uniqueness, and stable dependence on initial values for $U_0 \in H^{s+1} \times H^s$, with

$$U(t) \in C([t_0, t_1], H^{s+1} \times H^s),$$

for $-k-1 \le s \le k$ [14, 16].

Let us consider now where there is room for improvement in standard finite difference or finite element methods. Suppose U_1, U_2 are discretized on Ω by finite differences, using a regular grid with grid distance h and $N = L/h$ grid points. Then the operator M is discretized, and the time evolution is computed with some time-stepping procedure. The operator M behaves like a first-order operator, mapping $H^{s+1} \times H^s$ to $H^s \times H^{s-1}$. Its norm is proportional to h^{-1}. Accuracy and stability of a discrete approximation now require that the time step is of order h, $\Delta t \lesssim h/c(x,t)$, with $c = \sqrt{b/a}$ the velocity (the Courant–Friedrichs–Lewy condition). The cost for given N is therefore at least $O((\#\text{ of time steps}) \cdot N) = O(N^2)$.

To have lower cost, we will attack the number of time steps, by using larger time steps. An idea that has been used for this purpose is operator splitting with an *integrating factor method*. Suppose M is of the form

$$M = A + B. \tag{1.4}$$

Operator splitting is the idea that the matrix exponential $e^{\Delta t(A+B)}$ is approximated by products of factors $e^{\Delta t_j A}$ and $e^{\Delta t_k B}$. One way to derive an operator splitting method is the integrating factor method. Let $E(t,t_0)$ be a solution operator for $U' = AU$, i.e., an operator that maps $U(t_0)$ to the solution $U(t)$ of $U' = AU$. For the time-independent case $E(t,t_0) = e^{(t-t_0)A}$. Then we can define

$$V = E(t,t_0)^{-1}U. \tag{1.5}$$

The term $E(t,t_0)^{-1}$ is then an integrating factor. Differentiating the equivalent equation $E(t,t_0)V = U$ gives that

$$(A+B)U = \frac{dU}{dt} = AE(t,t_0)V + E(t,t_0)\frac{dV}{dt}.$$

Therefore, solving for $\frac{dV}{dt}$,

$$\frac{dV}{dt} = E(t,t_0)^{-1} B E(t,t_0) V. \tag{1.6}$$

To apply this usefully, the operator on the right-hand side must have smaller norm than the original operator M, so that time-stepping can be performed with larger time steps. This is applied in some nonlinear equations with a diffusive part, for which the time evolution can be computed efficiently in the Fourier domain [20]. Because of this use of an integrating factor, we call our method a geometrical optics integrating factor method.

A similar idea is used in the Egorov theorem of microlocal analysis. In this theory, a Fourier integral operator (FIO) $E(t,t_0)$ is constructed [11, 10, 19, 21], such that the field $V(t) = E(t,t_0)^{-1}U(t)$ satisfies

$$\frac{\partial}{\partial t} V(t) = R(t,t_0)V(t), \tag{1.7}$$

where the operator R is smoothing, in the sense that it maps $H^{s+1} \times H^s \to H^{s+1+K} \times H^{s+K}$ for any K desired (the order K depends on the amount of terms in the asymptotic series for the amplitude in the FIO $E(t,t_0)$). The fact that R is bounded means that a properly discretized version can be bounded independent of h. By the above reasoning the stepsize requirement would become independent of h (of course an estimate of the time discretization error is needed to establish this). For small h, as the number of time steps would become large due to the CFL condition, one might expect to have a gain in computation speed for the transformed differential equation (1.7).

Continuing this line of reasoning, the time step could become independent of the number of space discretization points N, assuming the desired accuracy stays fixed. For example, having initial conditions double in frequency, with the same medium and accuracy, one can conjecture that the time step could stay the same.

While Fourier integral operator theory has been developed for any space dimension, for dimension 2 or higher it is not clear how to efficiently obtain numerical approximations of Fourier integral operators (see for work in this direction, e.g., the recent paper [3]). Here we therefore treat the one-dimensional case.

In this case, it is convenient not to work with the field U, or with V in (1.7) directly, but instead work with forward and backward propagating components. These will be denoted by u_1 and u_2. An operator Q and its inverse will be constructed such that $u = (u_1, u_2)^T = Q^{-1}U$ (this gives step 1 and 3). We will show that in terms of these variables the differential equation (1.3) becomes

$$\frac{d}{dt} u = (T + R)u \tag{1.8}$$

with

$$T = \begin{pmatrix} \sqrt{b/a}\partial_x + f_1 & 0 \\ 0 & -\sqrt{b/a}\partial_x + f_2 \end{pmatrix},$$

f_1, f_2 functions given below, and R a remainder operator, that is explicitly derived and is continuous $H^{s+1} \times H^{s+1} \to H^{s+2} \times H^{s+2}$ (for time-independent coefficients $f_1 = f_2 = 0$). Versions of R with off-diagonal terms that are even more smoothing can also be constructed; see further on in the paper.

Equation (1.8) will be used for operator splitting. The equation $u' = Tu$ corresponds to two transport equations (step 2 in the outline above). These are solved

using the method of characteristics. This yields a geometrical optics approximation of the propagator. The term R then yields the correction mentioned in step 3 of the four points above.

Computing with the characteristics is cheaper than computing directly on the wavefield, e.g., using a discretization of the transport equation. The explanation for this is that the time steps in an ODE solver needed for solving for the characteristics depend on the medium smoothness, and not on the smoothness of the wavefield, and can therefore be longer than the time steps in a discretization of the transport equation. Similarly, it is not necessary to compute a characteristic for each grid point because interpolation can be used. After computing the characteristics, applying the flow along the characteristics becomes a standard interpolation problem.

The computation of flow along characteristics is related to the use of moving grids in scalar conservation laws. Originally the reason to have the grid moving with the singularities of solutions was that an adapted (locally refined) grid would stay adapted to the singularities. But it was also observed that this could lead to larger time steps [13].

As mentioned we have both numerical and theoretical results. First we derive a relatively simple method following the above ideas. This method has been implemented and compared with a standard order (4,4) finite-difference method described in [6]. Factors of order 10 to 20 of improvement in the computation speed were obtained in examples.

In the second part of the paper we study error estimates and complexity. It turns out that the method described in sections 2 to 4 does not yet have the best possible complexity. With several enhancements we construct a method (or a class of methods) with optimal complexity $O(N)$ to solve the initial value problem. These additional features are the use of higher-order decoupling, and of a multiscale decomposition where each scale has its own time step (multiscale time-stepping). They will be further introduced in section 6.

The remainder of the paper will be organized as follows. In section 2 we describe the separation of the forward and backward propagating parts of the wavefield (decoupling). The differential equation is then transformed into one to which operator splitting and the integrating factor method can be applied. This is discussed in section 3. We then describe a simple space discretization and the resulting algorithm in section 4. Section 5 contains some numerical results. Section 6 introduces the main additional ideas behind the method for which we establish $O(N)$ complexity. These are further worked out and proved in sections 7, 8, and 9. We end with a short discussion of the results.

2. Decoupling the equation. The splitting in (1.4)–(1.6) is not directly applied to M; first the equation is transformed to new variables as announced in (1.8). We define new variables by

$$U(t) = Q(t)u(t),$$

with Q an invertible matrix operator. The operator Q is independent of t if M is independent of t, and may otherwise depend on t. The equation for u is then ($'$ denoting time differentiation)

$$u' = (Q^{-1}MQ - Q^{-1}Q')u. \tag{2.1}$$

The purpose of this section is to find a suitable operator Q, such that the resulting differential equation is of the form

$$\frac{d}{dt}\begin{pmatrix} u_1 \\ u_2 \end{pmatrix} = \begin{pmatrix} \sqrt{b/a}\partial_x + f_1 & 0 \\ 0 & -\sqrt{b/a}\partial_x + f_2 \end{pmatrix}\begin{pmatrix} u_1 \\ u_2 \end{pmatrix} + R\begin{pmatrix} u_1 \\ u_2 \end{pmatrix}, \tag{2.2}$$

with Q, f_1, and f_2 and the remainder operator R to be determined. In fact, we will derive an explicit expression for

$$QR = \underbrace{MQ}_{A} - \underbrace{Q\begin{pmatrix} \sqrt{b/a}\partial_x + f_1 & 0 \\ 0 & -\sqrt{b/a}\partial_x + f_2 \end{pmatrix}}_{B} - \underbrace{Q'}_{C}. \tag{2.3}$$

The notations A, B, C will be used below in evaluating the product. Note that R is not given directly, but has to be computed as the product of Q^{-1} and QR which are given; the reason for this is that we want to minimize the use of inverse differential operators, and here the only place where those occur is in Q^{-1}. We will find that the operator R belongs to a class of pseudodifferential operators of order -1.

In the remainder of the section the actual computation is done. We treat separately the cases where a, b are time-independent, resp., the general case with time-dependent a, b. For convenience we collect the results in the following lemma.

LEMMA 2.1. *For the time-independent case, with Q given by* (2.5)*, and $f_1 = f_2 = 0$, QR is given by* (2.6) *and* (2.7)*. For the time-dependent case, with Q, f_1, f_2 given by* (2.8)*,* (2.9)*, and* (2.12)*, QR is given by* (2.10)*,* (2.11)*,* (2.13)*, and* (2.14)*.*

Computation for the time-independent case. In this case we will take Q independent of t, so that $C = 0$, and such that f_1 and f_2 vanish. Consider first the following choice for Q:

$$Q^{(0)} = \begin{pmatrix} 1 & 1 \\ \sqrt{ab}\partial_x & -\sqrt{ab}\partial_x \end{pmatrix}.$$

A quick computation shows that

$$Q^{(0)}R^{(0)} = MQ^{(0)} - Q^{(0)}\begin{pmatrix} \sqrt{b/a}\partial_x & 0 \\ 0 & -\sqrt{b/a}\partial_x \end{pmatrix} = \begin{pmatrix} \text{order}(0) & \text{order}(0) \\ \text{order}(1) & \text{order}(1) \end{pmatrix}, \tag{2.4}$$

so to highest order this is a good choice.

Next we modify Q so that (1) it is invertible, and (2) the components of QR vanish to one order lower. The operator Q becomes invertible when the derivative is replaced by a regularized derivative, which will be denoted by $\tilde{\partial}_x$, defined in the Fourier domain by multiplication with $ik + \frac{\alpha}{\beta k^2+1}$, with α, β suitable positive, real constants that remain to be chosen. To eliminate the order 0 and order 1 terms in (2.4), the columns of Q will be normalized by a weight function; we will try

$$Q = \begin{pmatrix} f(x) & f(x) \\ f(x)\sqrt{ab}\tilde{\partial}_x & -f(x)\sqrt{ab}\tilde{\partial}_x \end{pmatrix}, \qquad Q^{-1} = \tfrac{1}{2}\begin{pmatrix} f^{-1} & \tilde{\partial}_x^{-1} f^{-1}\frac{1}{\sqrt{ab}} \\ f^{-1} & -\tilde{\partial}_x^{-1} f^{-1}\frac{1}{\sqrt{ab}} \end{pmatrix}, \tag{2.5}$$

with f given by $f = a^{-1/4}b^{-1/4}$.

For contribution A we then find

$$A_{11} = -A_{12} = f(x)\sqrt{\frac{b}{a}}\partial_x + f(x)\sqrt{\frac{b}{a}}(\tilde{\partial}_x - \partial_x) = a^{-3/4}b^{1/4}\partial_x + a^{-3/4}b^{1/4}(\tilde{\partial}_x - \partial_x)$$

and

$$\begin{aligned} A_{21} = A_{22} &= \partial_x b \partial_x f \\ &= a^{1/4}b^{1/4}\partial_x a^{-1/2}b^{1/2}\partial_x + R_1 \end{aligned}$$

with

$$\begin{aligned} R_1 = a^{-1/4}b^{3/4}\big[&\left(\tfrac{1}{4}\partial_x \log a - \tfrac{3}{4}\partial_x \log b\right)\left(\tfrac{1}{4}\partial_x \log a + \tfrac{1}{4}\partial_x \log b\right) \\ &- \left(\tfrac{1}{4}\partial_x^2 \log a + \tfrac{1}{4}\partial_x^2 \log b\right)\big]. \end{aligned}$$

Contribution B is given by

$$B_{11} = -B_{12} = a^{-3/4}b^{1/4}\partial_x$$

and

$$B_{21} = B_{22} = a^{1/4}b^{1/4}\partial_x a^{-1/2}b^{1/2}\partial_x + a^{1/4}b^{1/4}(\tilde{\partial}_x - \partial_x)a^{-1/2}b^{1/2}\partial_x.$$

We thus find the following for QR:

$$(QR)_{11} = -(QR)_{12} = a^{-3/4}b^{1/4}(\tilde{\partial}_x - \partial_x) \tag{2.6}$$

and

$$(QR)_{21} = (QR)_{22} = R_1 - a^{1/4}b^{1/4}(\tilde{\partial}_x - \partial_x)a^{-1/2}b^{1/2}\partial_x. \tag{2.7}$$

The time-dependent case. In this case we try

$$Q = \begin{pmatrix} f(x) & f(x) \\ f(x)\sqrt{ab}\tilde{\partial}_x + c_1 & -f(x)\sqrt{ab}\tilde{\partial}_x + c_2 \end{pmatrix}, \tag{2.8}$$

with f as above, and f_1, f_2, c_1, c_2 to be determined. The inverse of Q will be discussed below. We find

$$\begin{aligned} A_{11} &= a^{-3/4}b^{1/4}\partial_x + a^{-3/4}b^{1/4}(\tilde{\partial}_x - \partial_x) + a^{-1}c_1, \\ A_{12} &= -a^{-3/4}b^{1/4}\partial_x - a^{-3/4}b^{1/4}(\tilde{\partial}_x - \partial_x) + a^{-1}c_2; \end{aligned}$$

A_{21} and A_{22} remain unchanged. For the coefficients of the matrix operator B we find

$$\begin{aligned} B_{11} &= a^{-3/4}b^{1/4}\partial_x + (ab)^{-1/4}f_1, \\ B_{12} &= -a^{-3/4}b^{1/4}\partial_x + (ab)^{-1/4}f_2, \\ B_{21} &= a^{1/4}b^{1/4}\partial_x a^{-1/2}b^{1/2}\partial_x + a^{1/4}b^{1/4}(\tilde{\partial}_x - \partial_x)a^{-1/2}b^{1/2}\partial_x \\ &\quad + c_1\sqrt{b/a}\partial x + (ab)^{1/4}\tilde{\partial}_x \circ f_1 + c_1 f_1, \\ B_{22} &= a^{1/4}b^{1/4}\partial_x a^{-1/2}b^{1/2}\partial_x + a^{1/4}b^{1/4}(\tilde{\partial}_x - \partial_x)a^{-1/2}b^{1/2}\partial_x \\ &\quad - c_2\sqrt{b/a}\partial x - (ab)^{1/4}\tilde{\partial}_x \circ f_2 + c_2 f_2. \end{aligned}$$

For C we have

$$C_{11} = \partial_t (ab)^{-1/4},$$

$$C_{12} = \partial_t (ab)^{-1/4},$$

$$C_{21} = \partial_t (ab)^{1/4} \tilde{\partial}_x + \partial_t c_1,$$

$$C_{22} = -\partial_t (ab)^{1/4} \tilde{\partial}_x + \partial_t c_2.$$

Adding all the contributions we find that

$$(QR)_{11} = +a^{-3/4} b^{1/4} (\tilde{\partial}_x - \partial_x) + a^{-1} c_1 - (ab)^{-1/4} f_1 - \partial_t (ab)^{-1/4}$$

and

$$(QR)_{21} = R_1 - a^{1/4} b^{1/4} (\tilde{\partial}_x - \partial_x) a^{-1/2} b^{1/2} \partial_x - c_1 \sqrt{b/a} \partial x - (ab)^{1/4} \tilde{\partial}_x f_1$$
$$- c_1 f_1 - \partial_t (ab)^{1/4} \tilde{\partial}_x - \partial_t c_1.$$

The lower-order terms vanish if

$$\begin{aligned} c_1 &= -a^{3/4} b^{-1/4} ((ab)^{-1/4} \partial_t (ab)^{1/4}), \\ f_1 &= 0. \end{aligned} \tag{2.9}$$

What results is

$$(QR)_{11} = a^{-3/4} b^{1/4} (\tilde{\partial}_x - \partial_x) \tag{2.10}$$

and

$$\begin{aligned} (QR)_{21} = R_1 &- a^{1/4} b^{1/4} (\tilde{\partial}_x - \partial_x) a^{-1/2} b^{1/2} \partial_x - \partial_t (ab)^{1/4} (\tilde{\partial}_x - \partial_x) \\ &+ \partial_t (\sqrt{a/b} \partial_t (ab)^{1/4}). \end{aligned} \tag{2.11}$$

Similarly we have for the 12 and 22 components

$$(QR)_{12} = -a^{-3/4} b^{1/4} (\tilde{\partial}_x - \partial_x) + a^{-1} c_2 - (ab)^{-1/4} f_2 - \partial_t (ab)^{-1/4},$$

$$(QR)_{22} = R_1 - a^{1/4} b^{1/4} (\tilde{\partial}_x - \partial_x) a^{-1/2} b^{1/2} \partial_x + c_2 \sqrt{b/a} \partial x + (ab)^{1/4} \tilde{\partial}_x f_2$$
$$- c_2 f_2 + \partial_t (ab)^{1/4} \tilde{\partial}_x - \partial_t c_2,$$

with lower-order terms vanishing if

$$\begin{aligned} c_2 &= -a^{3/4} b^{-1/4} ((ab)^{-1/4} \partial_t (ab)^{1/4}), \\ f_2 &= 0. \end{aligned} \tag{2.12}$$

The result for $(QR)_{12}$ and $(QR)_{22}$ are

$$(QR)_{12} = -a^{-3/4} b^{1/4} (\tilde{\partial}_x - \partial_x), \tag{2.13}$$

$$(QR)_{22} = R_1 - a^{1/4} b^{1/4} (\tilde{\partial}_x - \partial_x) a^{-1/2} b^{1/2} \partial_x + \partial_t (ab)^{1/4} (\tilde{\partial}_x - \partial_x)$$
$$+ \partial_t (\sqrt{a/b} \partial_t (ab)^{1/4}). \tag{2.14}$$

This completes the time-dependent case, except for the inverse of Q.

For the inversion, rewrite Q as

$$Q = \begin{pmatrix} f(x) & f(x) \\ f(x)\sqrt{ab}(\tilde{\partial}_x + \bar{c}_1) & -f(x)\sqrt{ab}(\tilde{\partial}_x - \bar{c}_2) \end{pmatrix},$$

with $\bar{c}_j = \frac{c_j}{f\sqrt{ab}}$. It turns out that Q can be inverted, according to the following explicit formula:

$$Q^{-1} = \frac{1}{2}\begin{pmatrix} \tilde{\partial}^{-1}(\tilde{\partial}_x - \bar{c}_2)f^{-1} & \tilde{\partial}^{-1}\frac{1}{\sqrt{ab}}f^{-1} \\ \tilde{\partial}^{-1}(\tilde{\partial}_x + \bar{c}_1)f^{-1} & -\tilde{\partial}^{-1}\frac{1}{\sqrt{ab}}f^{-1} \end{pmatrix}. \tag{2.15}$$

This is basically due to the fact that $\bar{c}_1 = \bar{c}_2$.

3. Operator splitting and time-stepping. The equation for the decoupled wavefields u is now

$$\frac{d}{dt}u = (T + R)u \tag{3.1}$$

with R as derived in the previous section and T given by

$$T = \begin{pmatrix} \sqrt{b/a}\partial_x & 0 \\ 0 & -\sqrt{b/a}\partial_x \end{pmatrix}.$$

The integrating factor will be $E(t, t_0)^{-1}$, where $E(t, t_0)$ solves $\frac{d}{dt}E(t, t_0) = TE(t, t_0)$, $E(t_0, t_0) = \mathrm{Id}$, and we will define a field v by

$$v(t, t_0) = E(t, t_0)^{-1}u(t),$$

which satisfies the differential equation

$$\frac{dv}{dt} = E(t, t_0)^{-1}RE(t, t_0)v. \tag{3.2}$$

Applying Euler forward time-stepping for this equation gives

$$v(t + \Delta t, t) \approx (1 + \Delta t\, E(t + \Delta t, t)^{-1}RE(t + \Delta t, t))u(t),$$

using that $v(t, t) = u(t)$. Hence

$$u(t + \Delta t) \approx (1 + \Delta t\, R)E(t + \Delta t, t)u(t).$$

A symmetric form of splitting (cf. Strang splitting [17]) leads to the following time-stepping, expressed in time-stepping for u:

$$u(t + \Delta t) \approx (1 + \tfrac{1}{2}\Delta t\, R)E(t + \Delta t, t)(1 + \tfrac{1}{2}\Delta t\, R)u(t). \tag{3.3}$$

Let us now explain in more detail the computation of $E(t, t_0)$. This is a diagonal 2×2 matrix operator. We take the forward propagating component (the $E_{2,2}$ component, which acts on the u_2 field); the backward propagating component is done similarly. The characteristic equation is

$$\frac{dx}{dt} = c(x, t). \tag{3.4}$$

For the time-independent case, we can solve this ODE for $x(t)$ with initial value $x(t_0) = x_0$ by separating the variables, which yields the equation $\int_{x_0}^{x} c(\xi)^{-1}\, d\xi = t - t_0$, so the computation can be done from a primitive $\int c(x)\, dx$. For the time-dependent case (3.4) is solved directly. Let $X(x_0, t, t_0)$ denote the solution $x(t)$ with initial values $x(t_0) = x_0$. Then we have

$$E_{2,2}(t, t_0)u_2 = u_2(t_0, X(x, t_0, t)) \tag{3.5}$$

(the characteristic is computed backward). If $\Phi_2(t, t_0)$ denotes the characteristic flow mapping x_0 to $X(x_0, t, t_0)$, this equals the pull back $E_{2,2}(t, t_0)u_2(t_0) = \Phi_2(t_0, t)^* u_2(t_0)$.

4. Numerical implementation. For a numerical implementation, it remains to perform the space discretization. We chose to work with finite differences, which are easy to implement. The following operators were discretized:

1. ∂_x. This operator was discretized using central differences.
2. $\tilde{\partial}_x$, $\tilde{\partial}_x^{-1}$, $\tilde{\partial}_x - \partial_x$. These are applied in the Fourier domain, with a regularized version of central differences. There computation involves an FFT and an inverse FFT, which, due to the $O(N \log N)$ cost of this operation, will form the bulk of the computations.
3. Multiplications with coefficients and derivatives of coefficients. Derivatives of coefficients are computed again using central differences.
4. The *translation operator* $E(t, t_0)$ is computed for the time-independent case using the primitive $\int c(x)^{-1}\, dx$, mentioned above, and using a Runge–Kutta ODE solver otherwise. Then third-order Lagrange interpolation is applied. For the time-independent case a sparse matrix is precomputed, that performs the translation over a given time step Δt.

In this way a simple numerical implementation of the method given by (3.3) was made.

5. Numerical results. In the numerical results we concentrate on the method for the time-independent case. For this case comparisons of computation time were made. For the time-dependent case it was observed that solutions are well approximated. But we feel the results for the time-independent case give sufficient indication of the effectiveness of the method.

For this method, with the assumption of medium smoothness it is of course an important question *just how smooth the medium coefficients need to be* in order that the method demonstrates an improvement compared to more conventional methods. Therefore numerical results were computed for media with increasing smoothness. The media were chosen parameterized by B-splines of order 3; the coefficients a of the media were randomly chosen, uniformly distributed between 0.4 and 1.6. The increasing smoothness was obtained by increasing the node distance, for which we took the values 1, 2, 4, and 8. The b coefficient was chosen equal to 1. The initial value for U_1 was a pulse of approximately unit width; the initial value for U_2 was chosen equal to zero. In Figure 5.1 one such medium is displayed. In Figure 5.2 the initial value for U_1 is plotted. The propagation was over approximately 100 wavelengths.

The results were compared with the result of an order (4,4) finite difference method; see [6]. Both methods were implemented in MATLAB. For our method the main cost was in the Fourier transform used for computing $\tilde{\partial}_x$ and its inverse. In the standard finite difference methods, for each time step a sparse matrix was applied, and this constituted almost 100% of the cost.

The first check was that the method actually approximates the solutions well. This was indeed the case. In Table 5.1 some numerical results are given, where

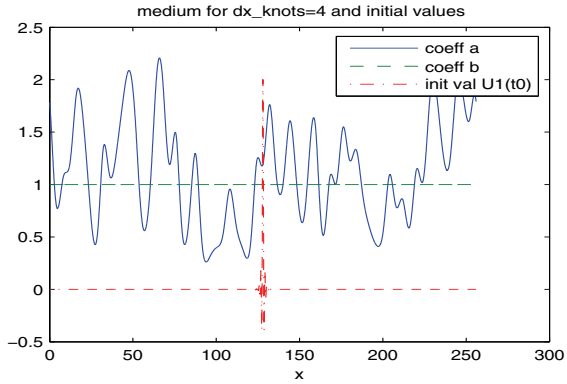

FIG. 5.1. *Medium coefficients with random B-splines with knot distance* 4.

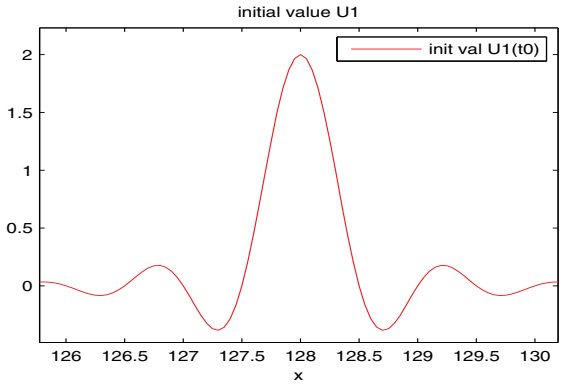

FIG. 5.2. *Initial value for* U_1 *used in the numerical tests.*

TABLE 5.1

Comparison of the cost of the method of section 4 *with an order* (4,4) *finite difference method.* h_{FD} *is the space stepsize taken in the finite difference scheme for which the comparison is made.*

Medium scale	h_{FD}	$\frac{\text{Cost FD}}{\text{Cost GOIF}}$
1	0.05	0.37
	0.025	0.30
2	0.05	3.3
	0.025	5.4
4	0.05	9.0
	0.025	15.7
8	0.05	17.4
	0.025	25.3

computation time is compared. For the new method we required the error to be smaller in both the supremum and the L^2 sense, or at most 10% larger in one of the two, but better when both are taken into account. As can be seen, knot distance 1 is not sufficient to obtain any gain, but from knot distance 2 considerable gain is obtained, up to a factor of about 20 for very smooth media.

As this is only a first implementation we feel this is strong encouragement to further analyze geometrical optics based methods.

6. An optimal complexity method: Overview. For the method introduced above there were no rigorous error estimates given. The complexity is, however, at least $O(N \log N)$ since the regularized derivative $\tilde{\partial}_x$ and its inverse were computed in the Fourier domain and needed to be computed for each time step. In this section we present a more elaborate algorithm, for which we establish that the complexity is $O(N)$, where N denotes the number of grid points in the space discretization.

So the task in the remaining sections is on the one hand to control the error in a numerical method and on the other hand control the cost. The discretization will be done for the differential equation

$$\frac{dv}{dt} = E(t, t_0)^{-1} R E(t, t_0) v, \tag{6.1}$$

that resulted from (3.1) after applying the integrating factor. It follows from the results in section 8 below that the transformation from the original equation (1.3) to this form and back can be done at cost $O(N)$ and with error satisfying bounds that are sufficient.

We will provide precise error estimates of classical type; i.e., we assume the input has a certain amount of additional regularity, we consider the discretization error in the result given that the input has to be approximated in an N-dimensional space of (spline-) functions, and we then show that the total error in the output is of the same order in N as the discretization error. Evolution according to (6.1) maps initial values $v(t_0) = v_0$ in $H^1 \times H^1$ to final values $v(t_1)$ that are also in $H^1 \times H^1$. We will assume that v_0 is in $H^{1+\alpha} \times H^{1+\alpha}$, i.e., has α additional orders of regularity. The discretization error that results from putting v_0 in an N-dimensional spline space can then be estimated by $CN^{-\alpha}$. We will show that, for a method with cost that can be bounded by CN, the final result satisfies an estimate of the type

$$\|v_{\text{approx}}(t_1) - v(t_1)\|_{H^1 \times H^1} \leq CN^{-\alpha}$$

(the letter C may mean a different constant in different equations).

A naive approach would be to simply take the differential equation (6.1), first apply a discretization in space, and then subsequently apply discretization in time. The time discretization should preferably be of higher order. There are two main problems with this approach, which will lead to additional special features of our method. These new features are the following:

1. *Higher order decoupling.* Control of the time discretization error in higher-order time-stepping, say of order K, requires bounds on the time derivatives of the operator $E(t, t_0)^{-1} R E(t, t_0)$ occurring on the right-hand side of (6.1). The first time derivative contains a commutator $[R, T]$ (which is of order 0 and hence bounded), but higher time derivatives contain higher-order commutators, that are of positive order, and hence do not satisfy the required bounds. To address this issue we will introduce higher-order decoupling. In section 7 we will construct a new operator R, with off-diagonal terms that are smoothing operators of order K, and show that its time derivatives of order $0, \ldots, K$ are bounded on a sufficiently large range of Sobolev spaces. The higher-order decoupling is obtained by adapting an argument of Taylor [19, Chapter 9] or [18].
2. *Multiscale time-stepping.* The second problem that needs to be addressed is that in our complexity estimates, with increasing N, the error must decrease. This in turn means that the time step must decrease, which would lead to

superlinear complexity. To address this issue we introduce *multiscale time-stepping*. The idea is that the coarse scales are propagated with a small time step. The coarse scales are parameterized with relatively few coefficients but contain most of the energy. It is therefore affordable to use a smaller time step, and at the same time this leads to a big improvement in the error. For the fine scales, that contain relatively little energy, larger time steps are used. Incidentally this is very much in agreement with the philosophy of asymptotic methods, where the high frequencies are well approximated. Each time step amounts to a correction to the purely asymptotic approximation, so few are needed for the high frequencies. The idea of multiscale time-stepping is new to our knowledge.

Because of the multiscale time-stepping, we assume the use of a wavelet based multiscale discretization in space. We will use [5] as our main reference for wavelet discretization; see also [7].

In the next three sections we will work out the above issues in detail and prove the $O(N)$ complexity result. Section 7 concerns the higher-order decoupling. Discretization and operator approximation will be discussed in section 8. Section 9 will contain the ideas on multiscale time-stepping and the final parts of the proof that combine all the intermediate results.

7. Higher-order decoupling. By the transformation $u = Q^{-1}U$ in section 2, the original system (1.3) was transformed to

$$u' = (T + R)u,$$

where $T + R = Q^{-1}MQ - Q^{-1}Q'$. We had

$$T = \begin{pmatrix} \sqrt{b/a}\partial_x & 0 \\ 0 & -\sqrt{b/a}\partial_x \end{pmatrix}.$$

The operator R is a matrix pseudodifferential operator, with components that are of order

$$R = \begin{pmatrix} \text{order}(-1) & \text{order}(-1) \\ \text{order}(-1) & \text{order}(-1) \end{pmatrix}. \tag{7.1}$$

Here by order(-1) we mean that it is bounded $H^s \to H^{s+1}$ for a suitable range of s.

In this section we explain how to construct Q such that R has the property that

$$\frac{d^j}{dt^j}(E(t,t_0)^{-1}RE(t,t_0)) \text{ is bounded on } H^1 \times H^1 \text{ for } j = 0, 1, \ldots, K, \tag{7.2}$$

with K a positive integer indicating, as mentioned, the order of the time-stepping that is going to be used.

We first argue that property (7.1) is not sufficient if $K > 1$. Take for example the first time derivative of $E(t,t_0)^{-1}RE(t,t_0)$:

$$\frac{d}{dt}\left(E(t,t_0)RE(t,t_0)\right) = E(t,t_0)^{-1}\left([R,T] + \frac{dR}{dt}\right)E(t,t_0). \tag{7.3}$$

Consider the commutator $[R,T]$ occurring inside the brackets:

$$[R,T] = \begin{pmatrix} [R_{1,1},T_{1,1}] & R_{1,2}T_{2,2} - T_{1,1}R_{1,2} \\ R_{2,1}T_{1,1} - T_{2,2}R_{2,1} & [R_{2,2},T_{2,2}] \end{pmatrix}. \tag{7.4}$$

To get the idea assume that the coefficients a and b are C^∞, so that R and T have smooth symbols. What we see from this expression is the following:

- The diagonal terms $[R,T]_{1,1}$ and $[R,T]_{2,2}$ are commutators of scalar pseudodifferential operators, and their order equals the order of $R_{1,1}$, resp., $R_{2,2}$.
- For the off-diagonal terms $[R,T]_{1,2}$ and $[R,T]_{2,1}$ this is not true; their order is increased by 1 compared to $R_{1,2}$, resp., $R_{2,1}$.

This has nothing to do with the specific form of R; if R is replaced by a different matrix pseudodifferential operator, these two statements remain true. So consider the second-order time derivative of $E(t,t_0)RE(t,t_0)$. This contains the higher-order commutator $[[R,T],T]$. Assuming (7.1) and using (7.4) twice, it follows that the off-diagonal terms $[[R,T],T]_{1,2}$ and $[[R,T],T]_{2,1}$ are (a priori) of order 1, implying that (7.2) is violated.

To address this problem we will construct a modified operator Q, such that

$$R = \begin{pmatrix} \text{order}(-1) & \text{order}(-K) \\ \text{order}(-K) & \text{order}(-1) \end{pmatrix}. \tag{7.5}$$

The old operators Q and R will be referred to as $Q^{(-1)}$ and $R^{(-1)}$, because of (7.1). The new operators will be referred to as $Q^{(-K)}$ and $R^{(-K)}$. This way, we can handle K time derivatives, each of which can increase the order of the off-diagonal term by 1.

We write $\tilde\partial_x = \partial_x + \Psi$, where from now on we assume that Ψ is smoothing in the sense that it is continuous $H^s \to H^{s+K}$, $1-K \le s \le 1$. The reason is that then any term that is a product of Ψ and other operators, none of which is of positive order, automatically is of order$(-K)$ and is hence "safe" (see (7.5)). For Ψ, we could use for example

$$\Psi = \frac{\alpha}{\beta(-\partial_x^2)^{\lceil K/2\rceil}+1}$$

with symbol $\frac{\alpha}{\beta k^{2\lceil K/2\rceil}+1}$. This is a modification with respect to the original definition of $\tilde\partial$ in section 2. However, it does not affect equations like (2.6), (2.7), (2.10), (2.11), (2.13), and (2.14), because the specific form of $\tilde\partial_x - \partial_x$ is not used in their derivation.

The main result of this section is captured in the following theorem, a short explanation of which is given after its formulation.

THEOREM 7.1. *Assume a, b are at least $C^{2K+1,1}$. There exists an operator $Q^{(-K)}$ of the form*

$$Q^{(-K)} = Q^{(-1)} \begin{pmatrix} 1 & E \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ F & 1 \end{pmatrix}$$

such that the operator $R^{(-K)}$ satisfies (7.2). *The operators E, F can be chosen of the form*

$$\sum_{j=2}^{K} c_{E^{(-j)}}(x,t)\tilde\partial_x^{-j}, \qquad \sum_{j=2}^{K} c_{F^{(-j)}}(x,t)\tilde\partial_x^{-j},$$

where the $c_{E^{(-j)}}(x,t)$, $c_{F^{(-j)}}(x,t)$ are (x,t) dependent coefficients that depend on $a(x,t)$, $b(x,t)$ and derivatives of order up to j of a, b. The operators that form the matrix elements of $R^{(-K)}$ are sums of products of the following basic operators: operator Ψ, operators $\tilde\partial^{-k}$ for $k \ge 0$, and multiplication by coefficients that are functions

of a, b and derivatives of order at most $K+1$ of a and b. This can be done such that all the terms for the off-diagonal elements of $R^{(-K)}$ are explicitly of order $-K$ in the sense that they contain a factor of Ψ or at least K powers of $\tilde{\partial}_x^{-1}$.

The description as a sum of products of basic operators is such that the operators involved can be numerically approximated with the techniques described in section 8. We note in particular that there are no cancellations between terms of $R^{(-K)}$ of order $> -K$. This is important, to avoid the situation where $R^{(-K)}$ consists of several contributions whose highest-order parts cancel analytically but not numerically due to the errors made in the numerical approximation. In the proof we will also describe a calculational scheme to compute the $c_{E^{(-j)}}(x,t)$, $c_{F^{(-j)}}(x,t)$. (We have not calculated any case $K > 1$ explicitly.)

Proof. We write temporarily

$$T + R^{(-1)} = \begin{pmatrix} A & B \\ C & D \end{pmatrix}.$$

We will first assume that a, b are C^∞, so that all pseudodifferential operators involved have smooth symbols; later we will investigate how much smoothness for the coefficients is needed. Using a transformation with a matrix pseudodifferential operator of the form $\left(\begin{smallmatrix} 1 & E \\ 0 & 1 \end{smallmatrix}\right)$ the operator B will be removed to the highest $K-1$ orders.

Replacing Q by $Q\left(\begin{smallmatrix} 1 & E \\ 0 & 1 \end{smallmatrix}\right)$ yields the following for the new operator R; see (2.1):

$$\begin{pmatrix} 1 & E \\ 0 & 1 \end{pmatrix}^{-1} \begin{pmatrix} A & B \\ C & D \end{pmatrix} \begin{pmatrix} 1 & E \\ 0 & 1 \end{pmatrix} - \begin{pmatrix} 1 & E \\ 0 & 1 \end{pmatrix}^{-1} \begin{pmatrix} 0 & E' \\ 0 & 0 \end{pmatrix} = \begin{pmatrix} A - EC & B + AE - ED - E' \\ C & D + CE, \end{pmatrix}, \tag{7.6}$$

where we used the explicit inverse $\left(\begin{smallmatrix} 1 & E \\ 0 & 1 \end{smallmatrix}\right)^{-1} = \left(\begin{smallmatrix} 1 & -E \\ 0 & 1 \end{smallmatrix}\right)$. The first problem is to find E such that $B+AE-ED-E'$ is of the desired lower order. Next we do a transformation with a matrix $\left(\begin{smallmatrix} 1 & 0 \\ F & 1 \end{smallmatrix}\right)$ of the matrix in (7.6). After this second transformation, the new operator R becomes

$$\begin{pmatrix} 1 & 0 \\ F & 1 \end{pmatrix}^{-1} \begin{pmatrix} A - EC & B + AE - ED - E' \\ C & D + CE \end{pmatrix} \begin{pmatrix} 1 & 0 \\ F & 1 \end{pmatrix} - \begin{pmatrix} 1 & 0 \\ F & 1 \end{pmatrix}^{-1} \begin{pmatrix} 0 & 0 \\ F' & 0 \end{pmatrix}$$

$$= \begin{pmatrix} A - EC + (B + AE - ED - E')F & B + AE - ED - E' \\ C + (D + CE)F - F(A - EC) - F' & D + CE - F(B + AE - ED - E') \end{pmatrix}.$$

Just like E we must then choose F, such that $C + (D + CE)F - F(A - EC) - F'$ is of the desired lower order. The new Q is then $Q\left(\begin{smallmatrix} 1 & E \\ 0 & 1 \end{smallmatrix}\right)\left(\begin{smallmatrix} 1 & 0 \\ F & 1 \end{smallmatrix}\right)$ (using the factor $\left(\begin{smallmatrix} 1 & E \\ 0 & 1 \end{smallmatrix}\right)\left(\begin{smallmatrix} 1 & 0 \\ F & 1 \end{smallmatrix}\right)$ is convenient compared to $\left(\begin{smallmatrix} 1 & E \\ F & 1 \end{smallmatrix}\right)$ because it has an explicit inverse, easy numerically).

Let us consider the construction of E. This follows a standard pattern in pseudodifferential operator theory, choosing E order by order. We let

$$E = E^{(-2)} + E^{(-3)} + \cdots + E^{(-K)},$$

and set

$$B^{(-2)} = B^{(-1)} + AE^{(-2)} - E^{(-2)}D - E^{(-2)\prime},$$

$$B^{(-3)} = B^{(-2)} + AE^{(-3)} - E^{(-3)}D - E^{(-3)\prime},$$

$$\text{etc., until } B^{(-K)} = B + AE - ED - E'.$$

The principal symbol of $B^{(-k)}$ is of the form $c_{B^{(-k)}(x)}(i\xi)^{-k}$, while those of A and $-D$ are both equal to $\sqrt{b/a}(i\xi)$. Hence if we choose the principal symbol of $E^{(-k-1)}$ equal to $-\frac{c_{B^{(-k)}(x)}}{2\sqrt{b/a}}(i\xi)^{-k-1}$, then the principal symbol of $B^{(-k-1)}$ vanishes, with as a result that $B^{(-k-1)}$ becomes an operator of order $-k-1$ as desired. So we set

$$c_{E^{(-k-1)}} = -\frac{c_{B^{(-k)}}}{2\sqrt{b/a}} \qquad \text{and} \qquad E^{(-k-1)} = c_{E^{(-k-1)}}\tilde{\partial}_x^{-k-1}.$$

The operators $E^{(-k)}$ follow from this scheme. The coefficients $c_{B^{(-k)}}$ and $c_{E^{(-k)}}$ are determined inductively. This can be done on the symbol level using pseudodifferential operator calculus, or directly, as we will demonstrate now.

We further investigate this construction of the $c_{B^{(-k)}}$ and $c_{E^{(-k)}}$ and of the remainders $R^{(-k)}$. It is convenient to just take the matrix $R^{(-1)}$, which is the starting point of the induction, and apply a few steps of the recipe. Doing this, the key properties that allow the successful construction will become clear, without becoming overly formal.

The matrix $R^{(-1)}$ follows in the time-independent case from (2.6), (2.7), and (2.5). Omitting anything involving $\tilde{\partial}_x - \partial_x$ (which is smoothing by definition), we have the following terms relevant for the higher-order decoupling:

$$R^{(-1)} = \begin{pmatrix} \tilde{\partial}_x^{-1}a^{-1/4}b^{-1/4}R_1 & \tilde{\partial}_x^{-1}a^{-1/4}b^{-1/4}R_1 \\ -\tilde{\partial}_x^{-1}a^{-1/4}b^{-1/4}R_1 & -\tilde{\partial}_x^{-1}a^{-1/4}b^{-1/4}R_1 \end{pmatrix} + \text{order}(-K).$$

So we set, following the above scheme,

$$E^{(-2)} = -\frac{a^{-1/4}b^{-1/4}R_1}{2\sqrt{b/a}}\tilde{\partial}_x^{-2}.$$

We then find

$$\begin{aligned} B^{(-2)} &= B^{(-1)} + \left(\sqrt{b/a}\partial_x + \tilde{\partial}_x a^{-1/4}b^{-1/4}R_1\right)E - E\left(-\sqrt{b/a}\partial_x - \tilde{\partial}_x a^{-1/4}b^{-1/4}R_1\right) \\ &\quad - E' \\ &= \tilde{\partial}_x^{-1}a^{-1/4}b^{-1/4}R_1 - \sqrt{b/a}\partial_x\frac{a^{-1/4}b^{-1/4}R_1}{2\sqrt{b/a}}\tilde{\partial}_x^{-2} - \frac{a^{-1/4}b^{-1/4}R_1}{2\sqrt{b/a}}\tilde{\partial}_x^{-2}\sqrt{b/a}\partial_x \\ &\quad + \text{order}(-3). \end{aligned} \tag{7.7}$$

In the first term we need to commute $\tilde{\partial}_x^{-1}$ to the right, in the second term we need to commute ∂_x to the right, and in the third term we need to commute $\tilde{\partial}_x^{-2}$ to the right.

To continue an understanding of the commutator of $\tilde{\partial}_x^{-1}$ with (multiplication by) some function $g(x)$ is needed. Such a commutator yields the following:

$$\begin{aligned}[\tilde{\partial}_x^{-1}, g] =& -\tilde{\partial}_x^{-1}[\partial_x + S, g]\tilde{\partial}_x^{-1} \\ =& -\tilde{\partial}_x^{-1}(\partial_x g)\tilde{\partial}_x^{-1} - \tilde{\partial}_x^{-1}(Sg - gS)\tilde{\partial}_x^{-1}.\end{aligned}$$

The first term in this expression for the commutator is of order -2 and contains a coefficient with one more derivative. The second term is of order less than $-K$ and is hence to be disregarded.

After the commutations the highest-order terms in $B^{(-2)}$ cancel, and what remains are commutator terms and other lower-order terms.

Several more remarks are in order. First the general form, involving as basic operations the $\tilde{\partial}_x^j$, the operator $\Psi = \tilde{\partial}_x - \partial_x$, and multiplications with coefficients and derivatives and powers of coefficients, remains conserved in each step.

Concerning the order of derivatives of the coefficients that occur, in $B^{(-1)}$ and $E^{(-2)}$ we have at most second-order derivatives, in $B^{(-2)}$ and $E^{(-3)}$ at most third order, and inductively we find that in $B^{(-j)}$ and $E^{(-j-1)}$ we have at most $j+1$ order of derivatives. One of the assumptions is that the coefficients are $C^{2K+1,1}$, which implies that in $R^{(-K)}$ the coefficients are still $C^{K,1}$.

Does this also hold for the time derivatives; i.e., do we have (7.2)? We must then carefully study (7.3) and (7.4). It turns out that each time derivative leads to a loss of at most one derivative in the regularity of the coefficients of a coefficient multiplication operator. With K time derivatives, we need $C^{0,1}$ smoothness to have a bounded map on $H^1 \times H^1$ (L^∞ would be enough if the operator was considered on $L^2 \times L^2$). Therefore $C^{K,1}$ in the coefficients occurring in $R^{(-K)}$ is sufficient and (7.2) follows.

The operator F can be determined in a similar fashion. This completes the proof of Theorem 7.1. ∎

8. Discretization and operator approximation. The multiscale discretization will be done using wavelets. We follow the book of Cohen [5], which gives an excellent description of one-dimensional wavelet discretization theory; see also [7]. In a wavelet discretization functions in $L^2(\Omega)$ and $H^s(\Omega)$ are approximated by elements of increasingly large finite-dimensional subspaces of $L^2(\Omega)$ given by a multiresolution analysis V_j, $j = 0, 1, 2, \ldots$. The spaces V_j are spanned by translates and scalings of the scaling function ϕ:

$$\phi_{j,k} = 2^{j/2}\phi(2^j \cdot -k), \quad k \in \mathbb{Z}/(2^j L\mathbb{Z}).$$

The V_j are assumed to form an increasing sequence $V_j \subset V_{j+1}$, $\bigcup_{j=0}^{\infty} V_j = L^2(\Omega)$.

In our case, where the domain is a circle of integer length L, the space V_j has $L2^j$ elements. We denote by J the final level of discretization, so that $N = L2^J$. Typically we will denote by f_j an approximation of a function f in V_j, and by A_j the approximation of an operator A on V_j.

The multiscale decomposition is obtained from the wavelet spaces. The wavelet space W_j is such that $V_{j+1} = V_j \oplus W_j$. It is spanned by the translates and scalings $\psi_{j,k}$ of a mother wavelet ψ. This leads to the multiscale decomposition

$$V_j = V_0 \oplus W_0 \oplus \cdots \oplus W_{j-1}.$$

The scaling function can be chosen with compact support, and with any order C^k smoothness. Together with the V_j, a dual multiresolution analysis $\tilde{V}_j$ can be

constructed, spanned by translates and scalings of a dual scaling function $\tilde{\phi}$, such that the basis functions satisfy the biorthogonality property $\langle \tilde{\phi}_{j,k}, \phi_{j,k'} \rangle = \delta_{k,k'}$. One of $\phi, \tilde{\phi}$ can be chosen as a compactly supported spline, we assume ϕ is a spline, and V is a spline space of a certain order. The space V_j can be made to satisfy $V_j \subset H^s$ for any s by choosing wavelets of sufficiently high order of smoothness. Throughout the analysis we will assume sufficient smoothness of the wavelets, without specifying this precisely.

The error estimates and assumptions on the smoothness of initial values are formulated in terms of regularity in L^2 based Sobolev spaces. That is natural and convenient for wave equations (where physical energy conservation holds). It is also easy to handle in wavelet discretizations, because of norm equivalences. The Sobolev norms $\|\cdot\|_{H^s}$ are equivalent to weighted norms of the wavelet coefficients. If

$$f = \sum_{k=0}^{K-1} c_{-1,k}\phi_{0,k} + \sum_{j=0}^{\infty}\sum_{k=0}^{2^j L} c_{j,k}\psi_{j,k},$$

and the wavelets are sufficiently smooth, then there is the norm equivalence

$$\|f\|^2_{H^\alpha(\Omega)} \sim \sum_{j=-1}^{\infty}\sum_{k} |2^{\alpha j} c_{j,k}|^2.$$

From these norm equivalences one can easily derive an important approximation result. Assume that f is in H^α; then the projections $\Pi_{V_j} f$ of f to the V_j satisfy

$$\|f - \Pi_{V_j} f\|_{L^2(\Omega)} \le C 2^{-\alpha j} \|f\|_{H^\alpha(\Omega)}.$$

In our application we typically deal with products of operators that are applied after each other, in discrete form, to a discretized function. We first derive a criterion, that we call *order k approximation operator*, for each of the operators to satisfy, such that such products converge. After this we will argue that the operators in our application can be approximated such that the approximation indeed satisfies the approximation property.

Suppose A is some operator $H^{s_1} \to H^{s_2}$, and A_j is a discrete approximation to A. As pointed out, convergence estimates are done using additional regularity, say k additional orders of regularity. For our operator A from $H^{s_1} \to H^{s_2}$ we therefore assume its argument, say f is in H^{s_1+k}. The result Af may be the argument of another operator, so we will require $Af \in H^{s_2+k}$; in other words we will assume

$$A \text{ is continuous } H^{s_1+s} \to H^{s_2+s} \text{ for } 0 \le s \le k.$$

Next we discuss a property that ensures that $A_j f_j$ approximates Af if f_j approximates f.

Definition. Let A be as just described; then we say A and A_j satisfy the *order k approximation property* if

$$\|A - A_j\|_{H^{s_1+k} \to H^{s_2}} \le C 2^{-jk}.$$

This also implies that A_j is continuous $H^{s_1+s} \to H^{s_2+s}$ for $0 \le s \le k$. This implies that if a function $f \in H^{s_1+k}$ is approximated in H^{s_1} by functions f_j, with

the convergence as expected from the additional regularity, i.e., $\|f - f_j\|_{H^{s_1}} \leq C2^{-kj} \|f\|_{H^{s_1+k}}$, then $A_j f_j$ approximates Af in the same way, since

$$\begin{aligned}\|Af - A_j f_j\|_{H^{s_2}} &\leq \|A_j(f - f_j)\|_{H^{s_2}} + \|(A - A_j)f\|_{H^{s_2}} \\ &\leq C2^{-jk}\|f\|_{H^{s_1+k}}.\end{aligned}$$

We will assume that k is an integer, although this does not seem essential, and that $k \geq 1$.

The basic operators needed here are partial differential operators, the operator $(-\partial_x^2 + 1)^{-1}$ or inverses of higher-order elliptic operators for the approximation of the operator S of section 7, and the pull back along the characteristic flow (which is a smooth coordinate transformation). Here we discuss partial differential operators and constant coefficient inverse partial differential operators; the pull back will be discussed in the last part of this section. We state the result on the approximation of $R^{(-K)}$, $Q^{(-K)}$ as a lemma.

LEMMA 8.1. *Assume the coefficients a, b are $C^{k+K+1,1}$. Then numerical approximations to the operators $R^{(-K)}$ on $H^1 \times H^1$, $Q^{(-K)}$ from $H^1 \times H^1 \to H^1 \times L^2$ and $(Q^{(-K)})^{-1}$ from $H^1 \times L^2 \to H^1 \times H^1$ can be constructed that satisfy the order k approximation property.*

Proof. Multiplication by polynomials and differentiation operators can be discretized using results of [8]; see that reference or section 2.5 of [5]. They can be discretized at cost $O(N)$, in such a way that the above order k approximation property is satisfied. For multiplication operators with functions other than polynomials, the coefficient is locally approximated by polynomials. As for the regularity requirement on the coefficients, for an approximate multiplication operator on H^{s_1} to have the order k approximation property, it is sufficient to have $C^{k+s_1-1,1}$ coefficients, since a $C^{k-1,1}$ function can be approximated to error 2^{-jk} by polynomials on regions of size order 2^{-j}.

In the case of the approximation of $R^{(-K)}$ on $H^1 \times H^1$, the coefficients in the remainder term need to be $C^{k,1}$. It follows that the coefficients a and b must be in $C^{k+K+1,1}$.

The operator $(-\partial_x^2 + 1)^{-1}$ can be computed in $O(N)$ cost using a multigrid algorithm [1]; a wavelet variant of this algorithm was given in [5]. To show that the approximation property holds, a slight change in the argument about multilevel preconditioning in example 4 in section 3.11 of [5] is needed; namely, n_j is chosen such that $\rho^{n_j} \leq 2^{-t'j}$, with $t' > t$. Similar arguments work for the higher-order inverse elliptic operator Ψ. This concludes the proof. ∎

Next we will show a similar result for $E(t, t_0)$. This operator was diagonal with $E_{2,2}$ given by (see (3.5))

$$E_{2,2}(t, t_0)u_2(x) = u_2(t_0, X(x, t_0, t)). \tag{8.1}$$

The 1,1 component of $E(t, t_0)$ is given by a similar formula.

We will first discuss the approximation of $X(x, t, t_0)$; then the next lemma will contain the result on $E(t, t_0)$.

Let $X_j(x, t, t_0)$ denote a numerical approximation used at level j. This must be computed for a set of points x. We require increasing accuracy as j increases, with error bounded by $C2^{-j(k+1)}$. It is allowed that, as j increases, the computational cost increases as 2^j. We find that for the time-independent case C^{k+1} smoothness of

the coefficients is sufficient, while for the time-dependent case C^{2k+2} smoothness is sufficient for this computation, as we will now show.

For the time-independent case, the evaluation of (8.1) can be done by solving $X = X(x, t, t_0)$ from

$$\int_x^X c(\xi)^{-1}\, d\xi = t - t_0. \tag{8.2}$$

First the primitive $\int_0^x c(\xi)^{-1}\, d\xi$ is computed for all x in the periodic grid with grid distance 2^{-j}. Assuming that c is C^{k+1}, this can be done at cost $O(2^j)$, with error $\leq C2^{-j(k+1)}$. Next the solution of (8.2) can be done for a set of 2^j points x using interpolation, which conserves the order of error, i.e., with error still bounded by $C2^{-j(k+1)}$.

For the time-dependent case we solve for the characteristics using a Runge–Kutta method of order $2k+2$. We require C^{2k+2} smoothness of c; then we can take order $2^{j/2}$ points with distance between them of $2^{-j/2}$ and solve with time steps of order $2^{-j/2}$. The total error is then bounded by $C2^{-j(k+1)}$.

Next we discuss how (8.1) can be computed numerically such that the order k approximation property is satisfied.

LEMMA 8.2. *Assume the coefficients a, b are C^{k+1} for the time-independent case or C^{2k+2} in the time-dependent case, and the wavelets are order $k+1$ splines. Then a numerical approximation to the operator $E(t, t_0)$ on $H^1 \times H^1$ can be constructed that satisfies the order k approximation property.*

Proof. We consider the approximation at level J of $E_{2,2}(t, t_0)f$, with f an element of V_J. We have that $E_{2,2}(t, t_0)f(x) = f(X(x, t_0, t))$. For brevity we will write $X(x)$ instead of $X(x, t_0, t)$. We will write $h(x) = f(X(x))$. We want to compute $c_{J,\tilde{k}} = \langle \tilde{\phi}_{J,\tilde{k}}, h\rangle$. The computation of matrix elements of polynomials, i.e., $\langle \tilde{\phi}_{J,\tilde{k}}, p\rangle$, when p is a polynomial, is basically exact; see the method of section 2.5 of [5]. To compute matrix elements of other smooth functions, it is common to approximate these locally by polynomials, and we will also use this in this argument. So to compute the approximate coefficient of the scaling function $\phi_{J,\tilde{k}}$, the function h is approximated around the support of $\phi_{J,\tilde{k}}$ by a polynomial p. The approximate value of the coefficient is then $\tilde{c}_{J,\tilde{k}} = \langle \tilde{\phi}_{J,\tilde{k}}, p\rangle$ and is obtained according to the mentioned section of [5].

Thus we must define how to approximate h locally by a polynomial. This can simply be done by polynomial interpolation with an order k polynomial. A function h in $C^{k,1}$ can be approximated by interpolation on a grid of size 2^{-J} up to an error bounded by

$$\sup_{x \in S_{J,\tilde{k}}} |h(x) - p(x)| \leq C2^{-(k+1)J} \|h\|_{C^{k,1}(S_{J,\tilde{k}})}.$$

We will apply this to a wavelet, $f = \psi_{j,\hat{k}}$. We assume that the wavelet ψ is $C^{k,1}$ and use that X is also $C^{k,1}$. The function $h(x) = \psi_{j,\hat{k}}(X(x))$ satisfies $\|\psi_{j,\hat{k}}(X(\cdot))\|_{C^{k,1}(S_{J,\tilde{k}})} \leq C2^{j(k+3/2)}$. Thus the error with p an exact interpolating polynomial is given by

$$|c_{J,\tilde{k}} - \tilde{c}_{J,\tilde{k}}| \leq \|\tilde{\phi}_{J,\tilde{k}}\|_{L^1} \sup_{x \in S_{J,\tilde{k}}} |h(x) - p(x)| \leq C2^{J(-k-3/2)+j(k+3/2)} \leq C2^{(k+1)(j-J)}.$$

Here we used that $\|\tilde{\phi}_{J,\tilde{k}}\|_{L^1}$ can be bounded by $C2^{-J/2}$ (which has to do with the normalization; the L^2 norm of $\tilde{\phi}_{J,\tilde{k}}$ is normalized to unity). Thus we find that the

map from f to the error $\sum_{\tilde{k}}(c_{J,\tilde{k}} - \tilde{c}_{J,\tilde{k}})\phi_{J,\tilde{k}}$ is bounded by $C2^{-(k+1)J}$ from H^{k+1} to L^2, and hence by $C2^{-kJ}$ from H^{k+1} to H^1.

A second source of error is that $X_J(x)$ is used instead of the exact value $X(x)$. For these errors we have

$$\psi_{j,\hat{k}}(X_J(x)) - \psi_{j,\hat{k}}(X(x)) = \int_{X(x)}^{X_J(x)} \frac{d\psi_{j,\hat{k}}}{dx}(s)ds.$$

Since $\frac{d\psi_{j,\hat{k}}}{dx}$ is bounded by $C2^{3j/2}$, and $|X_J(x) - X(x)| < C2^{-J(k+1)}$, these errors satisfy

$$|\psi_{j,\hat{k}}(X_J(x)) - \psi_{j,\hat{k}}(X(x))| \leq C2^{3j/2-J(k+1)}.$$

From this a bound $C2^{-J(k+1)}$ follows for the map from input to this error, considered in spaces $H^{3/2} \to L^2$, and a bound $C2^{-JK}$ from $H^{3/2} \to H^1$, which is better than or equal to the bound for the interpolation error, since $k > 1/2$. ∎

9. Multiscale time-stepping and proof of the theorem. In this section multiscale time-stepping is introduced to finally obtain an $O(N)$ algorithm. The results of section 7 enable the use of higher-order time-stepping methods and lead to estimates for the time discretization errors. The results of section 8 allow us to estimate the errors due to space discretization. Here we will combine space and time discretization, choose parameters, like the order of space and time discretization, and establish the complexity of the algorithm by estimating error and cost of the algorithm.

We solve the equivalent of differential equation (3.2) with higher-order decoupling, after the application of the integrating factor; i.e., we solve

$$\frac{dv}{dt}(t) = S(t,t_0)v(t), \tag{9.1}$$

with

$$S(t,t_0) = E(t,t_0)^{-1}R^{(-K)}E(t,t_0),$$

where $R^{(-K)}$ is as constructed in section 7. We will approximate the solution $v(t_1)$ starting from t_0. The approximation is done in $H^1 \times H^1$. The initial values $v_0 = u_0$ also must be in $H^1 \times H^1$. We assume they have α additional orders of regularity; i.e., they are in fact in $H^{1+\alpha} \times H^{1+\alpha}$. It follows from the results of sections 7 and 8 that we can transform the values $U(t)$ of the original system (1.3) to those of the transformed system (9.1) and back with complexity $O(N)$.

Operators will be approximated with the order k approximation property, with $k > \alpha$. A minimum value for k is derived below. Regularity assumptions follow from these assumptions according to the previous sections. Note that this is different from the previous section, where the order k corresponded to the order of additional regularity of functions that operators acted on, while here $k > \alpha$. By S_j we denote an approximation of S in $V_j \times V_j$ with the order k approximation property, according to the methods of section 8. (Note that $S_j \neq \Pi_{V_j} S \Pi_{V_j}$.)

In general in an integrating factor method it is common to frequently reset t_0, so that $E(t,t_0)$ propagates only over small time intervals. We will refrain from doing so, as this is not needed in this context, and the frequent application of $E(t,t_0)$ to the full

signal (i.e., not only the addition made during a small time interval by a Runge–Kutta time step) may cause additional errors.

As motivated in section 6, we will make a multiscale decomposition of the signal and do time-stepping separately for each scale. The initial values are decomposed as follows:

$$u_0 = \sum_{j=0}^{J} w_{0,j},$$

with $w_{0,0} = \Pi_{V_0} u_0$, and $w_{0,j} = \Pi_{W_{j-1}} u_0$, for $j = 1, \ldots, J$. Here Π_{V_j}, Π_{W_j} denote the projection on $V_j \times V_j$ and $W_j \times W_j$, respectively. The field $v(t)$ will also be decomposed. The jth component, corresponding to initial values in $W_{j-1} \times W_{j-1}$, will not be approximated in $V_j \times V_j$, however (nor in $W_{j-1} \times W_{j-1}$), but in a space $V_{l(j)} \times V_{l(j)}$, $j \le l(j) \le J$. To indicate this we write the components of the sum as $v_{j,l(j)}$. We will show that $v(t)$ can be approximated like

$$v(t) \approx \sum_{j=0}^{J} v_{j,l(j)}.$$

The motivation for doing this is simple: Large errors would result in the time propagation in $V_j \times V_j$ of the $w_{0,j}$, while large cost would result if we would work in the full space $V_J \times V_J$. By working in an intermediate space both cost and errors can be controlled.

The final numerical approximation will be a sum of components $w_{j,l(j),\Delta t_j}$. The terms describe the discrete time propagation with time step Δt_j, using the space discretized operators $S_{l(j)}(t)$, applied to the initial values $w_{0,j}$.

For purposes of error estimation we consider two sets of fields in addition to $w_{j,l(j),\Delta t_j}$. We assume the fields $v_{j,l(j)}$ introduced above describe the continuous time propagation of the operator $\Pi_{V_{l(j)}} S \Pi_{V_{l(j)}}$, and the field $v_{j,l(j),\Delta t_j}$ will describe the discrete time propagation of $\Pi_{V_{l(j)}} S \Pi_{V_{l(j)}}$.

We first establish that $v_J(t_1)$ can be approximated like

$$v_J(t_1) \approx \sum_{k=0}^{J} v_{j,l(j)}(t_1).$$

LEMMA 9.1. *Suppose $l(j)$ is such that*

$$k(l(j) - j) = \alpha(J - j). \tag{9.2}$$

Then

$$\left\| \sum_{j=0}^{J} v_{j,l(j)}(t_1) - v(t_1) \right\|_{H^1 \times H^1} \le C 2^{-\alpha J} \|u_0\|_{H^{1+\alpha} \times H^{1+\alpha}}. \tag{9.3}$$

Proof. Let $v_{j,\infty}$ denote the solution of the exact differential equation with initial value $w_{0,j}$. It satisfies $\frac{dv_{j,\infty}}{dt} = S v_{j,\infty}$. As S is bounded on $H^{1+s} \times H^{1+s}$, $0 \le s \le k$, it follows that $v_{j,\infty}(t)$ satisfies the bound

$$\|v_{j,\infty}(t)\|_{H^{1+s} \times H^{1+s}} \le C \|w_{0,j}\|_{H^{1+s} \times H^{1+s}}$$

for $0 \le s \le k$, $t_0 \le t \le t_1$.

We have $\frac{dv_{j,l(j)}}{dt} = \Pi_{V_{l(j)}} S \Pi_{V_{l(j)}} v_{j,l(j)}$, so the difference $v_{j,l(j)} - v_{j,\infty}$ satisfies

$$\frac{dv_{j,l(j)} - v_{j,\infty}}{dt} = \Pi_{V_{l(j)}} S \Pi_{V_{l(j)}} (v_{j,l(j)} - v_{j,\infty}) + (\Pi_{V_{l(j)}} S \Pi_{V_{l(j)}} - S) v_{j,\infty}. \tag{9.4}$$

By standard estimates for ODEs we have that

$$\|v_{j,l(j)}(t) - v_{j,\infty}(t)\|_{H^{1+s}\times H^{1+s}} \le C_1 \|v_{j,l(j)}(t_0) - v_{j,\infty}(t_0)\|_{H^{1+s}\times H^{1+s}} + C_2 \int_{t_0}^{t} \|(\Pi_{V_{l(j)}} S \Pi_{V_{l(j)}} - S) v_{j,\infty}(s)\|_{H^{1+s}\times H^{1+s}}\, ds.$$

The first term on the right-hand side is zero. For the second term we use that by the regularity assumptions we have

$$\|\Pi_{V_{l(j)}} S \Pi_{V_{l(j)}} - S\|_{H^{1+k}\times H^{1+k} \to H^1 \times H^1} \le C 2^{-kl(j)}. \tag{9.5}$$

The components of the initial values $w_{0,j}$ are bounded according to

$$\|w_{0,j}\|_{H^{1+k}\times H^{1+k}} \le C 2^{j(k-\alpha)} \|w_{0,j}\|_{H^{1+\alpha}}, \tag{9.6}$$

and the same is true for $v_{j,\infty}(t)$ for $t_0 < t < t_1$. The inhomogeneous term in (9.4) can therefore be bounded by

$$\begin{aligned} \left\|(\Pi_{V_{l(j)}} S \Pi_{V_{l(j)}} - S) v_{j,\infty}(t)\right\|_{H^1\times H^1} &\le C 2^{-kl(j)+j(k-\alpha)} \|w_{0,j}\|_{H^{1+\alpha}\times H^{1+\alpha}} \\ &= C 2^{-\alpha J} \|w_{0,j}\|_{H^{1+\alpha}\times H^{1+\alpha}}. \end{aligned}$$

The error $v_{j,l(j)}(t_1) - v_{j,\infty}(t_1)$ therefore satisfies the bound

$$\|v_{j,\infty}(t_1) - v_{j,l(j)}(t_1)\|_{H^1\times H^1} \le C 2^{-\alpha J} \|w_{0,j}\|_{H^{1+\alpha}\times H^{1+\alpha}}. \tag{9.7}$$

Adding the estimates for each j results in (9.3). ∎

The second step in the estimation of the error is to estimate the time discretization error for the field $v_{j,l(j)}$. We will argue that the fields $v_{j,l(j)}$ can be sufficiently accurately approximated using Runge–Kutta time discretization. By $v_{j,l(j),\Delta t_j}$ we denote the time-discretized fields. We assume the use of an order K Runge–Kutta method for the time-stepping.

LEMMA 9.2. *Suppose that the time step Δt_j satisfies the inequality*

$$\Delta t_j \le C 2^{-\alpha(J-j)/K}, \tag{9.8}$$

and that the coefficients a, b are at least $C^{2K+1,1}$; then we have

$$\left\| \sum_{j=0}^{J} v_{j,l(j),\Delta t_j}(t_1) - \sum_{j=0}^{J} v_{j,l(j)}(t_1) \right\|_{H^1\times H^1} \le C 2^{-\alpha J} \|u_0\|_{H^{1+\alpha}\times H^{1+\alpha}}. \tag{9.9}$$

Proof. The error per time step in

$$\|v_{j,l(j)} - v_{j,l(j),\Delta t_j}\|_{H^1\times H^1}$$

is bounded by

$$(\Delta t_j)^{K+1} \sup_{\tau\in[t,t+\Delta t_j]} \left\| \frac{d^{K+1} v_{j,l(j)}(\tau)}{dt^{K+1}} \right\|_{H^1\times H^1}.$$

Using the differential equation, the higher-order time derivative $\frac{d^{K+1}v_j(\tau)}{dt^{K+1}}$ can be expanded as a sum of terms that are each given by a product of factors $\frac{d^\gamma}{dt^\gamma}\Pi_{V_{l(j)}} S \Pi_{V_{l(j)}}$ (total sum of the γ's is $\le K$) acting on $v_{j,l(j)}(\tau)$. In section 7 it was shown that with the given smoothness assumption on a, b, the time derivatives $\frac{d^j S}{dt^j}$ were bounded operators on $H^1 \times H^1$ for $j = 0, \dots, K$. The same is true for $\frac{d^j}{dt^j}\Pi_{V_{l(j)}} S \Pi_{V_{l(j)}}$. It follows that the error per time step is bounded by

$$C(\Delta t_j)^{K+1} \sup_{\tau\in[t,t+\Delta t_j]} \left\| v_{j,l(j)}(\tau) \right\|_{H^1\times H^1}.$$

Using standard arguments to go from local to global error, we find that the error at time t_1 can be estimated by

$$\|v_{j,l(j)}(t_1) - v_{j,l(j),\Delta t_j}(t_1)\|_{H^1\times H^1} \le C(\Delta t_j)^K \|w_{0,j}\|_{H^1\times H^1}.$$

We have that

$$\sum_{j=0}^{J} \left(2^{\alpha j}\|w_{0,j}\|_{H^1\times H^1}\right)^2$$

is bounded. We therefore require that

$$(\Delta t_j)^K \le C 2^{\alpha j} 2^{-\alpha J}; \tag{9.10}$$

then (9.9) follows. The conditions (9.8) and (9.10) are of course equivalent. ∎

For the estimate of the time discretization error it turned out to be convenient to work with $\Pi_{V_{l(j)}} S \Pi_{V_{l(j)}}$, an exact discretization that is not practical to compute, instead of $S_{l(j)}$, the approximate discretization discussed in section 8. The reason is that the errors made in $S_{l(j)}$ are not differentiable. So the next step is to take into account the difference between $S_{l(j)}$ and $\Pi_{V_{l(j)}} S \Pi_{V_{l(j)}}$.

LEMMA 9.3. *Assume still* (9.2). *We have the estimate*

$$\left\| \sum_{j=0}^{J} w_{j,l(j),\Delta t_j}(t_1) - \sum_{j=0}^{J} v_{j,l(j),\Delta t_j}(t_1) \right\|_{H^1\times H^1} \le C 2^{-\alpha J} \|u_0\|_{H^{1+\alpha}\times H^{1+\alpha}}. \tag{9.11}$$

Proof. The difference $S_{l(j)} - \Pi_{V_{l(j)}} S \Pi_{V_{l(j)}}$ satisfies a similar estimate as the difference $\Pi_{V_{l(j)}} S \Pi_{V_{l(j)}} - S$, which was considered in the proof of Lemma 9.1. The proof of (9.11) therefore proceeds similarly as the proof of Lemma 9.1, except that difference equations are used instead of differential equations. The difference $w_{j,l(j),\Delta t_j} - v_{j,l(j),\Delta t_j}$ satisfies the linear inhomogeneous difference equation

$$\begin{aligned} &w_{j,l(j),\Delta t_j}(t+\Delta t) - v_{j,l(j),\Delta t_j}(t+\Delta t) \\ &\quad = \Delta t\, \mathrm{RKStep}(t, \Delta t, S_{l(j)})(w_{j,l(j),\Delta t_j}(t) - v_{j,l(j),\Delta t_j}(t)) \\ &\qquad + \Delta t(\mathrm{RKStep}(t, \Delta t, S_{l(j)}) - \mathrm{RKStep}(t, \Delta t, \Pi_{V_{l(j)}} S \Pi_{V_{l(j)}}))v_{j,l(j),\Delta t_j}(t), \end{aligned}$$

where $\Delta t\,\mathrm{RKStep}(t,\Delta t,A)y$ denotes the Runge–Kutta step for the equation $y'=Ay$, which is a linear map on y. It follows that

$$\|w_{j,l(j),\Delta t_j}(\hat{t})-v_{j,l(j),\Delta t_j}(\hat{t})\|_{H^1\times H^1}\le C\Delta t_j$$
$$\times\sum_{t\text{-values}<\hat{t}}\|(\mathrm{RKStep}(t,\Delta t,S_{l(j)})-\mathrm{RKStep}(t,\Delta t,\Pi_{V_{l(j)}}S\Pi_{V_{l(j)}}))v_{j,l(j),\Delta t_j}(t)\|_{H^1\times H^1}.$$

The difference $\mathrm{RKStep}(t,\Delta t,S_{l(j)})-\mathrm{RKStep}(t,\Delta t,\Pi_{V_{l(j)}}S\Pi_{V_{l(j)}})$ can be worked out. It is a product of $S_{l(j)}-\Pi_{V_{l(j)}}S\Pi_{V_{l(j)}}$ and of operators that are bounded on H^{1+s}, $0\le s\le k$. It follows that we have the estimate

$$\left\|\mathrm{RKStep}(t,\Delta t,S_{l(j)})-\mathrm{RKStep}(t,\Delta t,\Pi_{V_{l(j)}}S\Pi_{V_{l(j)}})\right\|_{H^{1+k}\times H^{1+k}\to H^1\times H^1}\le C2^{-kl(j)}.$$

Furthermore

$$\left\|v_{j,l(j),\Delta t_j}(t)\right\|_{H^{1+k}\times H^{1+k}}\le 2^{j(k-\alpha)}\left\|w_{0,j}\right\|_{H^{1+\alpha}}.$$

It follows that we can estimate

$$\begin{aligned}\left\|w_{j,l(j),\Delta t_j}(t_1)-v_{j,l(j),\Delta t_j}(t_1)\right\|&\le C2^{-kl(j)+j(k-\alpha)}\left\|w_{0,j}\right\|_{H^{1+\alpha}\times H^{1+\alpha}}\\&=C2^{-\alpha J}\left\|w_{0,j}\right\|_{H^{1+\alpha}\times H^{1+\alpha}}.\end{aligned}$$

The estimate (9.11) trivially follows from this. ∎

This ends our estimation of the error. The *cost* of this time-stepping is

$$\begin{aligned}C\sum_{j=0}^{J}(\Delta t_j)^{-1}\,2^{l(j)}&=C\sum_{j=0}^{J}2^{\alpha(J-j)/K+\alpha(J-j)/k+j}\\&=C2^J\sum_{j=0}^{J}2^{(-1+\alpha/K+\alpha/k)(J-j)}.\end{aligned}$$

The requirement is that the cost is bounded by CN, and hence that $-1+\alpha/K+\alpha/k<0$. If we allow logarithmic cost $O(N\log N)$, equality is also allowed. We hence have our final result.

THEOREM 9.4. *If a Kth-order Runge–Kutta scheme is used, if the operators S_j are approximated using the order k approximation property, with, in particular, order $k+1$ spline wavelets, if the initial data u_0 are in $H^{1+\alpha}\times H^{1+\alpha}$, if coefficient functions are at least $C^{K+1+\max(k,K),1}$, and if*

$$1/K+1/k<1/\alpha, \tag{9.12}$$

then the algorithm above with $N=L2^J$ degrees of freedom computes an approximation with error bound

$$\left\|\sum_{j=0}^{J}w_{j,l(j),\Delta t_j}(t_1)-v(t_1)\right\|_{H^1\times H^1}\le CN^{-\alpha}\|u_0\|_{H^{1+\alpha}\times H^{1+\alpha}} \tag{9.13}$$

at a cost $O(N)$. If

$$1/K+1/k=1/\alpha, \tag{9.14}$$

it satisfies the same error bound at cost $O(N \log N)$.

The requirement that u_0 is in $H^{1+\alpha} \times H^{1+\alpha}$ means that the initial values U_0 for the original system (1.3) must be in $H^{1+\alpha} \times H^{\alpha}$.

In (9.13) it may look like we are summing J functions of N sample points, with cost $O(JN) = O(N \log N)$. However, this is not the case. The terms $w_{j,l(j),\Delta t_j}(t_1)$ have $C2^{l(j)}$ sample points (being in $V_{l(j)}$). Using the wavelet spaces, and the fast wavelet transform (which is $O(N)$ for N sample points), the summation can be done at cost $C \sum_{j=0}^{J} 2^{l(j)} \leq C2^J = O(N)$.

10. Discussion. A numerical method for wave propagation in smooth media was developed. The numerical results in section 5 show that the method certainly has potential in applications with relatively smooth media. Further improvements might be possible to further improve computation speed or weaken the requirements of medium smoothness. One step that could possibly give an improvement is a coordinate change that makes the wave speed equal to unity. We refrained from doing this since it has no equivalent in higher dimensions, but it could reduce the error in the application of the operator T.

The material of sections 6 to 9 not only leads to the $O(N)$ complexity result but also suggests ways to possibly improve the method.

The main question for future research is in our view about the generalization to higher-dimensional cases. For the multidimensional case, curvelets form a redundant basis (frame) with respect to which the solution operator can be made sparse [4]. Potentially it could be used for computations. However, one needs to be able to implement operators that give the approximate effect of wave propagation, such as translation, rotation, and deformation, efficiently in a curvelet basis. Perhaps other fast implementations of Fourier integral operators could be used (cf. [3]) to compute the approximate wave propagation. In dimension 2 and higher the remainder operator R becomes, at least in the continuous setting, a pseudodifferential operator, which is more challenging to implement. But a priori there is no reason why the principle of combining an approximate solution operator with lower-order, exact "corrections" could not be extended to higher dimensions.

REFERENCES

[1] R. E. Bank and T. Dupont, *An optimal order process for solving finite element equations*, Math. Comp., 36 (1981), pp. 35–51.

[2] G. Beylkin and K. Sandberg, *Wave propagation using bases for bandlimited functions*, Wave Motion, 41 (2005), pp. 263–291.

[3] E. Candès, L. Demanet, and L. Ying, *Fast computation of Fourier integral operators*, SIAM J. Sci. Comput., 29 (2007), pp. 2464–2493.

[4] E. J. Candès and L. Demanet, *The curvelet representation of wave propagators is optimally sparse*, Comm. Pure Appl. Math., 58 (2005), pp. 1472–1528.

[5] A. Cohen, *Numerical Analysis of Wavelet Methods*, Stud. Math. Appl. 32, North–Holland, Amsterdam, 2003.

[6] G. C. Cohen, *Higher-Order Numerical Methods for Transient Wave Equations*, Sci. Comput., Springer-Verlag, Berlin, 2002.

[7] W. Dahmen, *Wavelet and multiscale methods for operator equations*, in Acta Numerica, 1997, Cambridge University Press, Cambridge, UK, 1997, pp. 55–228.

[8] W. Dahmen and C. A. Micchelli, *Using the refinement equation for evaluating integrals of wavelets*, SIAM J. Numer. Anal., 30 (1993), pp. 507–537.

[9] L. Demanet and L. Ying, *Wave atoms and time upscaling of wave equations*, Numer. Math., to appear.

[10] J. J. Duistermaat, *Fourier Integral Operators*, Birkhäuser, Boston, 1996.

[11] J. J. Duistermaat and L. Hörmander, *Fourier integral operators* II, Acta. Math., 128 (1972), pp. 183–269.
[12] B. Engquist, S. Osher, and S. Zhong, *Fast wavelet based algorithms for linear evolution equations*, SIAM J. Sci. Comput., 15 (1994), pp. 755–775.
[13] R. J. LeVeque, *Convergence of a large time step generalization of Godunov's method for conservation laws*, Comm. Pure Appl. Math., 37 (1984), pp. 463–477.
[14] J. L. Lions and E. Magenes, *Non-homogeneous Boundary Value Problems and Applications*, Vol. 1, Springer-Verlag, Berlin, 1972.
[15] H. F. Smith, *A Hardy space for Fourier integral operators*, J. Geom. Anal., 8 (1998), pp. 629–653.
[16] C. C. Stolk, *On the Modeling and Inversion of Seismic Data*, Ph.D. thesis, Utrecht University, Utrecht, The Netherlands, 2000.
[17] G. Strang, *On the construction and comparison of difference schemes*, SIAM J. Numer. Anal., 5 (1968), pp. 506–517.
[18] M. E. Taylor, *Reflection of singularities of solutions to systems of differential equations*, Comm. Pure Appl. Math., 28 (1975), pp. 457–478.
[19] M. E. Taylor, *Pseudodifferential Operators*, Princeton University Press, Princeton, NJ, 1981.
[20] L. N. Trefethen, *Spectral Methods in MATLAB*, Software Environ. Tools 10, SIAM, Philadelphia, 2000.
[21] F. Treves, *Introduction to Pseudodifferential and Fourier Integral Operators*, Vol. 2, Plenum Press, New York, 1980.