SIAM J. SCI. COMPUT.
Vol. 24, No. 1, pp. 161–182

FAST CONVOLUTION FOR NONREFLECTING BOUNDARY CONDITIONS*

CHRISTIAN LUBICH† AND ACHIM SCHÄDLE†

Dedicated to Gerhard Wanner on the occasion of his 60th birthday

Abstract. Nonreflecting boundary conditions for problems of wave propagation are nonlocal in space and time. While the nonlocality in space can be efficiently handled by Fourier or spherical expansions in special geometries, the arising temporal convolutions still form a computational bottleneck. In the present article, a new algorithm for the evaluation of these convolution integrals is proposed. To compute a temporal convolution over N_t successive time steps, the algorithm requires $O(N_t \log N_t)$ operations and $O(\log N_t)$ memory. In the numerical examples, this algorithm is used to discretize the Neumann-to-Dirichlet operators arising from the formulation of nonreflecting boundary conditions in rectangular geometries for Schrödinger and wave equations.

Key words. transparent boundary conditions, radiation boundary conditions, fast convolution algorithm, wave equation, Schrödinger equation

AMS subject classifications. 65M99, 35L05, 35Q99, 44A10, 44A35

PII. S1064827501388741

1. Introduction.

Wave propagation is usually formulated in terms of partial differential equations on unbounded domains. For a computational treatment, however, the equations need to be restricted to a bounded domain in a neighborhood of the region of physical interest. This requires imposing *transparent boundary conditions* (or, synonymously, *nonreflecting* boundary conditions) which are ideally such that the solution of the equations incorporating these boundary conditions coincides with the restriction to the bounded domain of the solution to the whole-space equations. The derivation of such transparent boundary conditions is well understood; see, e.g., [3, 6] and also sections 3 and 4 below. However, these boundary conditions turn out to be nonlocal both in space and time. For special geometries, e.g., discs or balls or periodically extended slabs and cylinders, Fourier or spherical expansions decouple the boundary equations for the expansion coefficients. There still remain temporal convolutions of the type

$$\int_0^t f(t-\tau)\, g(\tau)\, d\tau\ , \qquad 0 \le t \le T. \tag{1.1}$$

For concreteness, consider the example of the wave equation $u_{tt} = u_{xx} + u_{yy} + u_{zz}$ on $\mathbb{R}^3$. Suppose the solution is 2π-periodic in y and z, a situation that occurs in modeling wave propagation near extended layers. Transparent boundary conditions are required at the planes $x = \pm a$. We consider the Fourier coefficients of the solution, $\hat{u}_k(x,t)$ with $k = (k_y, k_z) \in \mathbb{Z} \times \mathbb{Z}$. Their Laplace transforms $U_k(x,s)$ then satisfy the linear ordinary differential equation $s^2 U_k = \partial_{xx} U_k - |k|^2 U_k$ with $|k|^2 = k_y^2 + k_z^2$ for $\mathrm{Re}(s) > 0$. This equation can be solved analytically. Retaining only the solution decaying to zero as $|x| \to \infty$ gives $U_k(x,s) = \exp(-\sqrt{s^2 + |k|^2}\,|x \mp a|)\, U_k(\pm a, s)$. Differentiation of this equation at $x = \pm a$ yields a relation between the transforms of the

*Received by the editors May 2, 2001; accepted for publication (in revised form) November 19, 2001; published electronically May 20, 2002.
http://www.siam.org/journals/sisc/24-1/38874.html
†Mathematisches Institut, Universität Tübingen, Auf der Morgenstelle 10, D–72076 Tübingen, Germany (lubich@na.uni-tuebingen.de, schaedle@na.uni-tuebingen.de).

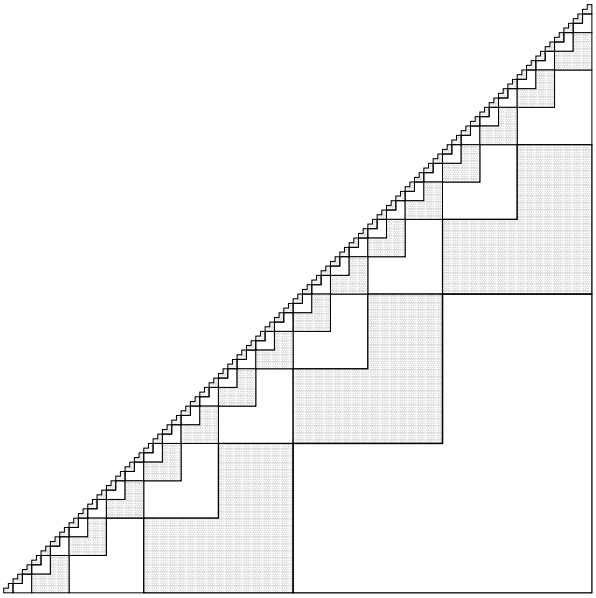

FIG. 1.1. *Tessellation for base* $B = 2$.

Dirichlet and Neumann data: $U_k(\pm a, s) = -(s^2 + |k|^2)^{-1/2} \partial_\nu U_k(\pm a, s)$. Applying the inverse Laplace transform to this equation gives the transparent boundary condition as the temporal convolution

$$\hat{u}_k(\pm a, t) = -\int_0^t f_k(t-\tau)\, \partial_\nu \hat{u}_k(\pm a, \tau)\, d\tau,$$

where $f_k(t)$ is the function with Laplace transform $F_k(s) = (s^2 + |k|^2)^{-1/2}$.

As in the above example, it is typically the Laplace transform $F(s)$ of $f(t)$ in (1.1), rather than the convolution kernel $f(t)$ itself, which is known a priori. The function $g(\tau)$ contains the expansion coefficients of Neumann or Dirichlet data of the solution on the boundary, which are not known beforehand but are computed as the algorithm proceeds from time step to time step. A naive implementation would require $O(N_t^2)$ operations and $O(N_t)$ memory per expansion coefficient for computing the temporal convolution over N_t time steps. In this paper we propose an algorithm that takes $O(N_t \log N_t)$ operations and $O(\log N_t)$ memory. The computational work and the memory requirements for accurately implementing the transparent boundary conditions thus become less than those for treating the interior domain and are asymptotically negligible as the spatial and temporal grid sizes tend to zero. The present paper describes the algorithmic aspects of this approach.

In section 2, after briefly reviewing existing approaches to the computational treatment of the temporal convolutions in transparent boundary conditions, we describe our algorithm. The basic idea is illustrated in Figure 1.1: each of the L-shaped regions in the triangle $\{(t, \tau) : 0 \le \tau \le t \le T\}$ corresponds to solving a set of scalar linear ordinary differential equations. On each of these regions, the memory requirements are therefore independent of the time step number, and along any vertical line in the triangle, i.e., for any fixed $t = n\Delta t$ with $n \le N_t$, there are at most $\log_2 N_t$ such regions. This leads to the logarithmic memory requirement mentioned above. The

scheme extends to decompositions of the triangle that correspond to other bases of the logarithm, yielding $\log_B N_t$, for example, with $B = 10$.

In section 3 we apply the algorithm to Schrödinger equations in conjunction with an implicit time discretization in the interior domain and in section 4 to wave equations with fully explicit time stepping. Numerical experiments illustrate the performance of the method in both cases. It should be noted that the algorithm is not restricted to Schrödinger and wave equations but applies equally to other problems of wave propagation, e.g., Maxwell's equations, linear elasticity equations, or damped wave equations.

2. The convolution algorithm. In this section we describe the algorithm for computing temporal convolutions which forms the basic ingredient in our treatment of nonreflecting boundary conditions.

2.1. Previously existing convolution algorithms. We consider the convolution (1.1) which is to be computed on the grid $t = 0, \Delta t, 2\Delta t, \ldots, T = N_t \Delta t$ with step size Δt. In (1.1) f and g assume different roles. We are interested in situations where the evaluation of $g(\tau)$ at $\tau = n\Delta t$ requires knowledge of the values of the convolution up to $(n-1)\Delta t$ so that the required values of $g(\tau)$ cannot be computed in advance. This is the situation in transparent boundary conditions or, more generally, in Volterra-type convolution equations. Typically, it is the Laplace transform $F(s)$ of the kernel $f(t)$, rather than the kernel itself, which is known a priori and can be evaluated easily. Therefore the algorithm should use evaluations only of $F(s)$.

If $F(s)$ is a rational function with a known partial fraction decomposition, say

$$F(s) = \sum_{j=1}^{m} \frac{c_j}{s - \lambda_j}\ , \qquad \text{then} \qquad f(t) = \sum_{j=1}^{m} c_j\, e^{\lambda_j t},$$

and it is a simple but very useful fact that the problem of computing the convolution (1.1) is in this case reduced to solving m linear differential equations:

$$\int_0^t e^{\lambda(t-\tau)}\, g(\tau)\, d\tau = y(t, 0, \lambda),$$

where $y(t, 0, \lambda)$ is the solution of the initial value problem

$$y' = \lambda y + g\ , \qquad y(0) = 0.$$

The computation of the convolution then requires $O(mN_t)$ operations and $O(m)$ active memory.

The situation of rational Laplace transforms $F(s)$ occurs in transparent boundary conditions on a sphere for the three-dimensional (3D) wave equation, and the above observation lies at the heart of the algorithm by Grote and Keller [4, 5]. Although this situation of rational $F(s)$ does arise in some situations of practical interest, $F(s)$ is *not* rational in many other cases arising from nonreflecting boundary conditions, be it for other equations (e.g., Schrödinger equations, damped wave equations) or other geometries. Moreover, even in cases of rational $F(s)$ the maximum degree m corresponds to the largest wave number considered and can therefore become large. The direct reduction to m differential equations is then not necessarily computationally efficient.

In view of the benefits of a rational Laplace transform $F(s)$, it is not a far-fetched idea to approximate a general $F(s)$ by a rational function and hence to approximate $f(t)$ by a finite sum of exponentials. This approach is excellently reviewed by

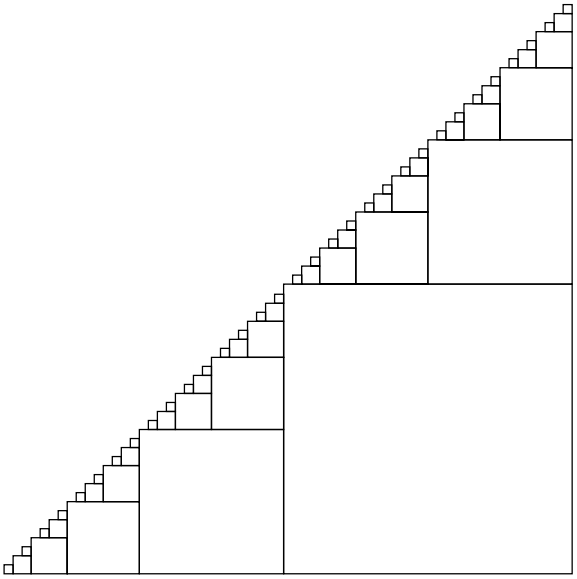

FIG. 2.1. *Tessellation for fast Fourier transforms* [7].

Hagstrom [6], where the reader finds many more references. In particular, Alpert, Greengard, and Hagstrom [1] give theoretical bounds on the degree m required for a uniform approximation of $f(t)$ on $[0,T]$ by a sum of m complex exponentials for the particular kernels $f(t)$ arising from nonreflecting boundary conditions of the wave equation in spherical and cylindrical geometries. The bounds grow only logarithmically with the wave number bound and with T. Alpert, Greengard, and Hagstrom [1] also propose a nonlinear least squares algorithm for the numerical construction of the exponents λ_j and coefficients c_j, which is, however, not free from computational difficulties.

A different approach to a fast convolution algorithm is given by Hairer, Lubich, and Schlichte [7]. That algorithm applies after replacing the continuous convolution with a discrete convolution

$$\Delta t \sum_{j=0}^{n} \omega_{n-j}\, g(j\Delta t),$$

where the convolution quadrature weights ω_n $(n \geq 0)$ can be constructed from $F(s)$ by the method of [8, 9]. The fast discrete convolution algorithm of [7] is based on decomposing the triangle $0 \leq \tau \leq t \leq T$ of the (t,τ)-plane into squares on which partial convolutions, composed of the values $\omega_{n-j}\, g(j\Delta t)$ for grid points $(n\Delta t, j\Delta t)$ of the square, can be computed by fast Fourier transforms. In Figure 2.1 each square corresponds to an FFT of a length proportional to the length of a side of the square. The computational complexity is then $O(N_t(\log N_t)^2)$ operations and $O(N_t)$ memory. A problem with this algorithm in the context of nonreflecting boundary conditions is the fact that all past boundary values and all quadrature weights need to be kept in active memory throughout the computation, which may not be feasible for 3D problems over longer time intervals.

2.2. Local approximation by discretized contour integrals. The algorithm to be described in the following approximates the kernel $f(t)$ by sums of exponentials *locally* on a sequence of fast growing time intervals I_ℓ covering $[\Delta t, T]$:

$$I_\ell = [B^{\ell-1}\Delta t, (2B^\ell - 1)\Delta t], \tag{2.1}$$

where the base $B > 1$ is an integer. For example, $B = 10$ was found to be a good choice in our numerical experiments. The approximation of $f(t)$ on I_ℓ results from applying the trapezoidal rule to a parametrization of the contour integral for the inverse Laplace transform,

$$f(t) = \frac{1}{2\pi i}\int_{\Gamma_\ell} F(\lambda)\, e^{t\lambda}\, d\lambda \approx \sum_{j=-N}^{N} w_j^{(\ell)}\, F(\lambda_j^{(\ell)})\, e^{t\lambda_j^{(\ell)}}, \qquad t \in I_\ell, \tag{2.2}$$

with a suitably chosen complex contour Γ_ℓ to be described in detail below. The number of quadrature points on Γ_ℓ, $2N + 1$ is chosen independent of ℓ. It is much smaller than what would be required for a uniform approximation of the contour integral on $[0, T]$.

To make such a contour integral approximation computationally efficient, $F(s)$ must have a bounded analytic extension to a complex domain whose boundary is given by contours that have only pieces of short total length in and near the right half-plane and tend to infinity with an acute angle to the negative real axis so that exponentials decay rapidly along such contours. It is a remarkable fact that the Laplace transforms $F(s)$ arising from Fourier or spherical expansions in nonreflecting boundary conditions on planar, cylindrical, and spherical geometries for the Schrödinger equation, undamped and damped wave equations, and Maxwell equations all have this property, in contrast to the resolvents of the corresponding differential operators (which are *not* sectorial operators). In the sections below, we will encounter

$$F(s) = \frac{1}{\sqrt{is+\alpha}} \quad \text{(S)} \qquad \text{and} \qquad F(s) = \frac{1}{\sqrt{s^2+\alpha^2}} \quad \text{(W)}$$

for the Schrödinger and the wave equation, respectively. We will also consider approximations to these functions with similar behavior, which result from a regular space discretization on the exterior domain. All these functions satisfy the above requirements uniformly in the wave number parameter $\alpha \in \mathbb{R}$, though with different contours shifted by $\pm i\alpha$.

The numerical integration in (2.2) is done by applying the trapezoidal rule with equidistant steps to a parametrization of a *Talbot contour* [12, 10], which is given by

$$(-\pi, \pi) \to \Gamma \quad \theta \mapsto \gamma(\theta) = \sigma + \mu\,(\theta\cot(\theta) + i\nu\theta)\,, \tag{2.3}$$

where the parameters μ, ν, and σ are such that the singularities of $F(s)$ lie to the left of the contour; see Figure 2.2. We may also use two shifted Talbot contours with $\sigma = \pm i\alpha$ if necessary, as in (W) above with large α. The parameter μ will depend on ℓ via the right end-point of I_ℓ, which yields a Talbot contour Γ_ℓ depending on the approximation interval I_ℓ.

For completeness we note that the weights and quadrature points in (2.2) are given by (omitting ℓ in the notation)

$$w_j = -\frac{j}{2(N+1)}\,\gamma'(\theta_j)\,, \quad \lambda_j = \gamma(\theta_j) \qquad \text{with} \quad \theta_j = \frac{j\pi}{N+1}.$$

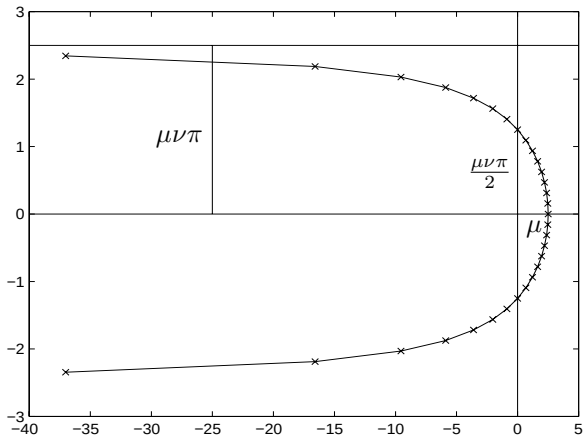

FIG. 2.2. *Talbot contour.*

The following choices of parameters were found to give good results in our numerical experiments. First consider the Schrödinger case (S). A relative accuracy of $1 \cdot 10^{-3}$ on the interval I_ℓ with right end-point T_ℓ is obtained with $B = 10$, $N = 10$, $\mu = \mu_0/T_\ell$ with $\mu_0 = 8$, $\nu = 0.6$, $\sigma = i\alpha$. For a relative approximation error of $2 \cdot 10^{-5}$, take $B = 5$, $N = 15$, and the other parameters as before. See Figure 2.3 for plots of the relative errors $|f_{\text{approx}}(t) - f(t)|/|f(t)|$ for $N = 10$ and $B = 10$ on three fast growing intervals $[10^{-3}, 2 \cdot 10^{-2}]$, $[10^{-2}, 2 \cdot 10^{-1}]$, and $[0.1, 2]$, and for $N = 15$ and $B = 5$ and for four fast growing intervals $[10^{-3}, 9 \cdot 10^{-3}]$, $[5 \cdot 10^{-3}, 5 \cdot 10^{-2}]$, $[2.5 \cdot 10^{-2}, 2.5 \cdot 10^{-1}]$, and $[0.125, 1.25]$. In fact, since $|f(t)| = (\pi t)^{-1/2}$, the same figures are obtained on any interval $[a, 20a]$ for $B = 10$ and $[a, 10a]$ for $B = 5$ with arbitrary $a > 0$.

In the case (W) for the wave equation we use only one contour as long as $\alpha \le \alpha^* = \frac{1}{2}\pi\mu\nu_0$. For a relative approximation error 10^{-3}, we choose $B = 10$, $N = 10$, $\mu = \mu_0/T_\ell$ with $\mu_0 = 8$, $\nu = \nu_0(1 + \alpha/\alpha^*)$ with $\nu_0 = 0.6$, $\sigma = 0$. For an accuracy of 10^{-5}, we choose $B = 5$, $N = 15$, and μ_0, ν_0, σ as before. For $\alpha > \alpha^*$ we take two Talbot contours with $\sigma = \pm i\alpha$ and the other parameters as in case (S) above. Using the symmetry with respect to the real axis approximately halves the computational work.

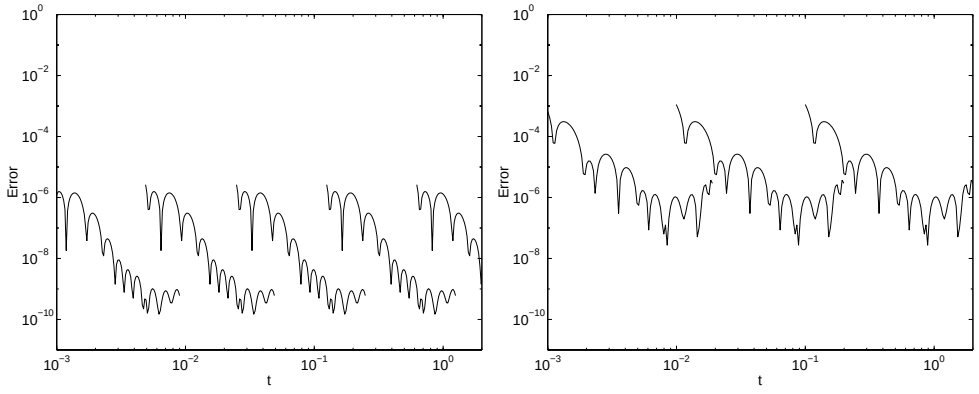

FIG. 2.3. *Quadrature errors versus time for $N = 15$ and $B = 5$ (left figure) and $N = 10$ and $B = 10$ (right figure).*

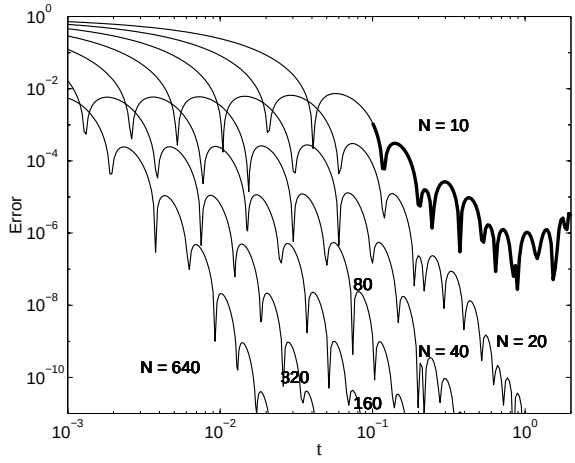

FIG. 2.4. *Quadrature error versus time.*

Figure 2.4 plots the relative errors for (S) on the interval $[1 \cdot 10^{-3}, 2]$ (or any interval $[a, 2000a]$ with $a > 0$) for $N = 10, 20, 40, 80, 160, 320, 640$. The thick part of the line for $N = 10$ is identical to that of the right figure in Figure 2.3. This error behavior clearly demonstrates the advantage of using local approximations with a rather small B. With $B = 10$, we need three approximation intervals to cover the interval $[1 \cdot 10^{-3}, 2]$ so that for a work of $3 \cdot N$ with $N = 10$ we obtain better accuracy than with $N = 640$ over the whole interval. The errors also behave similarly for (W).

The error of the quadrature approximation is known to decrease exponentially with N [12]. The constants in the error bound depend on the distance of the singularities of the analytic function F to the contour and on bounds of F. This error behavior would suggest to choose μ large, but this must be counterbalanced with the increased sensitivity to perturbations in terms multiplied with $e^{\mu t}$. For the above functions $F(s)$, which have inverse square root bounds near the singularities and at infinity, it can be shown that the required number N to obtain an error bounded by $\epsilon/\sqrt{t}$ on $(0, \infty)$ decomposed as in (2.1) is bounded by

$$N \le CB \log \frac{1}{\epsilon}, \tag{2.4}$$

with a moderate constant C, uniformly in ℓ and $B \ge 2$.

2.3. Reduction to ordinary differential equations. For general boundary points $a < b$ in the integral we have

$$\begin{aligned} \int_a^b f(t-\tau)\, g(\tau)\, d\tau &= \int_a^b \frac{1}{2\pi i} \int_\Gamma F(\lambda)\, e^{(t-\tau)\lambda}\, d\lambda\; g(\tau)\, d\tau \\ &= \frac{1}{2\pi i} \int_\Gamma F(\lambda)\, e^{(t-b)\lambda} \int_a^b e^{(b-\tau)\lambda}\, g(\tau)\, d\tau\; d\lambda, \end{aligned}$$

where the inner integral, henceforth denoted by $y(b, a, \lambda)$, is recognized as the solution at time b of the scalar linear initial value problem

$$y' = \lambda y + g\ , \qquad y(a) = 0. \tag{2.5}$$

If $[t-b, t-a] \subset I_\ell$, then the contour integral over the Talbot contour $\Gamma = \Gamma_\ell$ is replaced with its trapezoidal rule approximation (2.2), which gives (omitting the superscripts ℓ for notational simplicity)

$$\begin{aligned}\int_a^b f(t-\tau)\, g(\tau)\, d\tau &\approx \int_a^b \sum_{j=-N}^{N} w_j\, F(\lambda_j)\, e^{(t-\tau)\lambda_j}\, g(\tau)\, d\tau \\ &= \sum_{j=-N}^{N} w_j\, F(\lambda_j)\, e^{(t-b)\lambda_j}\, y(b,a,\lambda_j). \qquad (2.6)\end{aligned}$$

The $2N+1$ differential equations (2.5) with $\lambda = \lambda_j$ are solved approximately by replacing the function g with its piecewise linear approximation and then solving exactly. Setting $g_n = g(a+n\Delta t)$, we get approximations $y_n \approx y(a+n\Delta t)$ recursively via

$$\begin{aligned} y_{n+1} &= e^{\Delta t\lambda}\, y_n + h \int_0^1 e^{(1-\theta)\Delta t\lambda} \big(\theta g_{n+1} + (1-\theta) g_n\big)\, d\theta \\ &= y_n + \frac{e^{\Delta t\lambda}-1}{\Delta t\lambda} \left(\Delta t\lambda y_n + \Delta t g_n + \Delta t\, \frac{g_{n+1}-g_n}{\Delta t\lambda} \right) - \Delta t\, \frac{g_{n+1}-g_n}{\Delta t\lambda}. \qquad (2.7)\end{aligned}$$

To estimate the error, note that in total we approximate

$$\int_a^b f(t-\tau)\, g(\tau)\, d\tau \approx \int_a^b \widetilde{f}(t-\tau)\, \widetilde{g}(\tau)\, d\tau,$$

where $\widetilde{f}$ is the quadrature approximation to f constructed in the previous subsection, whose error is well under control, and $\widetilde{g}$ is the piecewise linear interpolant of g. (Higher-order interpolants of g might also be used, but we have not implemented such an extension.)

2.4. The convolution algorithm with base 2. The approximations of sections 2.2 and 2.3 can be combined into a fast convolution algorithm that requires $O(N_t \log N_t)$ arithmetical operations and $O(\log N_t)$ memory. This algorithm is best explained by describing the first few steps for base $B = 2$. (Here B refers to (2.1).)

First step. We compute the convolution integral at $t = \Delta t$ by approximating $g(\tau)$ linearly:

$$\int_0^{\Delta t} f(\Delta t-\tau)\, g(\tau)\, d\tau \approx \int_0^{\Delta t} f(\Delta t-\tau)\, d\tau\; g(0) + \int_0^{\Delta t} f(\Delta t-\tau)\, \tau\, d\tau\; \frac{g(\Delta t)-g(0)}{\Delta t}.$$

The remaining integrals are approximated as the inverse Laplace transforms of $F(s)/s$ and $F(s)/s^2$, respectively:

$$\begin{aligned} \phi_1 &= \int_0^{\Delta t} f(\Delta t-\tau)\, d\tau \approx \sum_{j=-N}^{N} w_j\, F(\lambda_j)/\lambda_j\; e^{\Delta t\lambda_j}, \\ \phi_2 &= \int_0^{\Delta t} f(\Delta t-\tau)\, \tau\, d\tau \approx \sum_{j=-N}^{N} w_j\, F(\lambda_j)/\lambda_j^2\; e^{\Delta t\lambda_j}, \qquad (2.8)\end{aligned}$$

where the weights w_j and nodes λ_j correspond to a Talbot contour with the parameter μ chosen for $t = \Delta t$ (e.g., $\mu = 8/\Delta t$).

Second step. The integral over $[0, 2\Delta t]$ is split into integrals over $[0, \Delta t]$ and $[\Delta t, 2\Delta t]$. The latter integral is approximated as in the first step:

$$\int_{\Delta t}^{2\Delta t} f(2\Delta t - \tau)\, g(\tau)\, d\tau \approx \phi_1\, g(\Delta t) + \phi_2\, \frac{g(2\Delta t) - g(\Delta t)}{\Delta t}.$$

The integral over $[0, \Delta t]$ is approximated according to (2.6):

$$\int_0^{\Delta t} f(2\Delta t - \tau)\, g(\tau)\, d\tau \approx \sum_{j=-N}^{N} w_j^{(1)}\, F\left(\lambda_j^{(1)}\right)\, e^{\Delta t \lambda_j^{(1)}}\, y\left(\Delta t, 0, \lambda_j^{(1)}\right),$$

where $w_j^{(1)}$, $\lambda_j^{(1)}$ correspond to the Talbot contour Γ_1 for the interval $I_1 = [\Delta t, 3\Delta t]$. This requires solving $2N + 1$ differential equations (2.5) with $\lambda = \lambda_j^{(1)} \in \Gamma_1$ by one step of (2.7).

Third step. We compute

$$\int_{2\Delta t}^{3\Delta t} f(3\Delta t - \tau)\, g(\tau)\, d\tau \approx \phi_1\, g(2\Delta t) + \phi_2\, \frac{g(3\Delta t) - g(2\Delta t)}{\Delta t}$$

and

$$\int_0^{2\Delta t} f(3\Delta t - \tau)\, g(\tau)\, d\tau \approx \sum_{j=-N}^{N} w_j^{(1)}\, F(\lambda_j^{(1)})\, e^{\Delta t \lambda_j^{(1)}}\, y\left(2\Delta t, 0, \lambda_j^{(1)}\right),$$

which requires advancing the solutions of the differential equations for Γ_1 from Δt to $2\Delta t$.

Fourth step. A new situation appears at $t = 4\Delta t$. Continuing as in the two previous steps would involve approximations of $f(t - \tau)$ for $t - \tau \in [\Delta t, 4\Delta t] \not\subset I_1$. As is indicated in Figure 2.5 by the different textures of the column for $t = 4\Delta t$, the integral from 0 to $4\Delta t$ is therefore split into integrals over the intervals $[0, 2\Delta t]$, $[2\Delta t, 3\Delta t]$, and $[3\Delta t, 4\Delta t]$, which correspond to different distance classes from the diagonal $t - \tau = 0$. The integral over $[3\Delta t, 4\Delta t]$, which is next to the diagonal, is approximated as in the above formulas with ϕ_1 and ϕ_2, with the arguments of g advanced by Δt. The integral over the interval $[2\Delta t, 3\Delta t]$, whose points τ have $t - \tau \in I_1$, is approximated by

$$\int_{2\Delta t}^{3\Delta t} f(4\Delta t - \tau)\, g(\tau)\, d\tau \approx \sum_{j=-N}^{N} w_j^{(1)}\, F(\lambda_j^{(1)})\, e^{\Delta t \lambda_j^{(1)}}\, y\left(3\Delta t, 2\Delta t, \lambda_j^{(1)}\right),$$

which uses the result of one step of method (2.7) for differential equations for Γ_1. The approximation of the integral over $[0, 2\Delta t]$, which is farthest from the diagonal, involves the weights and nodes for the Talbot contour Γ_2 that corresponds to the approximation interval $I_2 = [2\Delta t, 7\Delta t]$:

$$\int_0^{2\Delta t} f(4\Delta t - \tau)\, g(\tau)\, d\tau \approx \sum_{j=-N}^{N} w_j^{(2)}\, F(\lambda_j^{(2)})\, e^{2\Delta t \lambda_j^{(2)}}\, y\left(2\Delta t, 0, \lambda_j^{(2)}\right).$$

This requires the solution of another set of $2N + 1$ differential equations (2.5) with $\lambda = \lambda_j^{(2)} \in \Gamma_2$. The different textures in Figure 2.5 thus correspond to different sets of differential equations (2.5), which correspond to different Talbot contours

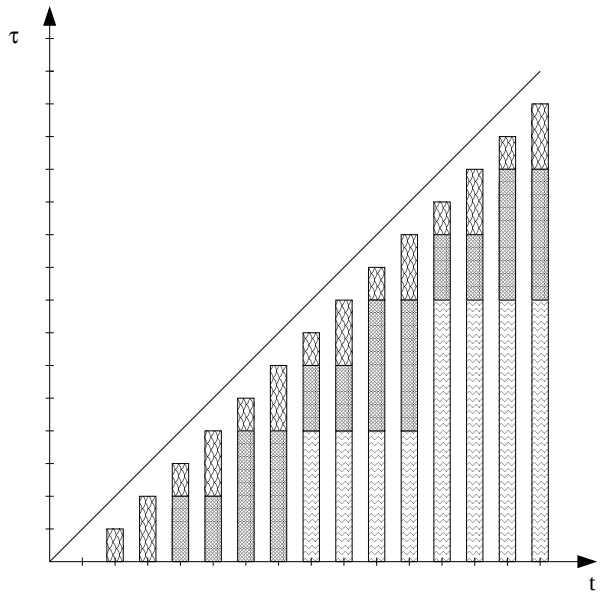

FIG. 2.5. *Decomposition for* $B = 2$.

Γ_ℓ, which in turn correspond to different approximation intervals I_ℓ. Identical textures in different columns of Figure 2.5 correspond to identical sets of differential equations/contours/approximation intervals/distance classes from the diagonal.

Computing $y(2\Delta t, 0, \lambda_j^{(2)})$ only at this moment would require the values $g(0)$, $g(\Delta t)$, $g(2\Delta t)$, which would thus have to be kept in memory. However, we can reorganize the computations so that in the second step above we not only compute $y(\Delta t, 0, \lambda_j^{(1)})$ but also $y(\Delta t, 0, \lambda_j^{(2)})$ for the second contour and in fact $y(\Delta t, 0, \lambda_j^{(\ell)})$ for all further contours Γ_ℓ that are required to cover the interval $[\Delta t, T]$ by the approximation intervals I_ℓ of (2.1) with $B = 2$. The third step then advances the solutions $y(t, 0, \lambda_j^{(\ell)})$ from Δt to $2\Delta t$ for all $\ell \geq 1$. The fourth step advances $y(t, 0, \lambda_j^{(\ell)})$ from $2\Delta t$ to $3\Delta t$ for all $\ell \geq 2$, whereas $y(t, 2\Delta t, \lambda_j^{(\ell)})$ is advanced for $\ell = 1$, in accordance with Figure 2.5. In this way one proceeds stepwise from bottom up in the triangle, rather than from left to right.

It is now clear how the algorithm continues in the next steps. The *fifth step* uses $y(4\Delta t, 2\Delta t, \lambda_j^{(1)})$ and again $y(2\Delta t, 0, \lambda_j^{(2)})$, and the *sixth step* uses $y(5\Delta t, 4\Delta t, \lambda_j^{(1)})$ and $y(4\Delta t, 0, \lambda_j^{(2)})$. A third set of differential equations, for the contour Γ_3, enters in the eighth step, where $y(4\Delta t, 0, \lambda_j^{(3)})$ is used. Carrying Figure 2.5 further to more steps finally gives the tessellation of the (t, τ)-triangle shown in Figure 1.1.

The algorithm is organized such that all the differential equations for all integration contours required for the given interval $[0, T]$ are advanced by one time step in every step $t \to t + \Delta t$ of the algorithm. In this way, past values of $g(t)$ need not be kept in memory but instead the present value and possibly one past value of the solutions $y(t, 0, \lambda_j^{(\ell)})$ for all j and ℓ. (For example, $y(4\Delta t, 0, \lambda_j^{(3)})$ must be stored in steps 4 to 11, while the differential equation with $\lambda_j^{(3)}$ is advanced by Δt in each of these steps.)

The number of operations is proportional to $N_t\, L_t\, N$, where N_t is the number of time steps, $L_t \le \log_2 N_t$ is the number of different contours, and $2N+1$ is the number of quadrature points on each contour, which depends logarithmically on the quadrature error tolerance; see (2.4). The required memory is approximately $5\, L_t N$. This includes storing the quadrature points $\lambda_j^{(\ell)}$, their exponentials $\exp(\Delta t \lambda_j^{(\ell)})$, and the weighted function values $w_j^{(\ell)} F(\lambda_j^{(\ell)})$. We also note that the algorithm is highly parallelizable.

2.5. The convolution algorithm for general bases. The algorithm extends to the base $B = 4$ in (2.1) if the contours in the above base-2 algorithm are chosen such that $\Gamma_1 = \Gamma_2$ is the joint Talbot contour for the union of the base-2 approximation intervals $I_1 \cup I_2 = [\Delta t, 7\Delta t]$, $\Gamma_3 = \Gamma_4$ is the contour for $I_3 \cup I_4 = [4\Delta t, 31\Delta t]$, $\Gamma_5 = \Gamma_6$ for $[16\Delta t, 127\Delta t]$, and so on. In this case, the differential equations associated with the quadrature points $\lambda_j^{(1)} = \lambda_j^{(2)}$, etc. are the same, which leads to substantial savings, since half of the sets of differential equations can be omitted. On the other hand, N must be chosen slightly larger so that there are now more quadrature points on each contour and hence more differential equations for each contour. As Figure 2.3 indicates, this base-4 algorithm is nevertheless more efficient, except possibly for extreme accuracy requirements below 10^{-10}. Figure 2.6 shows the decomposition of the (t, τ)-triangle corresponding to the base-4 algorithm, which is obtained from Figure 1.1 by uniting some of the tiles. Each of the differently colored regions corresponds to one set of differential equations.

Similarly, a base-8 algorithm is obtained by taking instead $\Gamma_1 = \Gamma_2 = \Gamma_3$ as the Talbot contour for $I_1 \cup I_2 \cup I_3 = [\Delta t, 15\Delta t]$, $\Gamma_4 = \Gamma_5 = \Gamma_6$ the contour for $[8\Delta t, 127\Delta t]$, etc. The base-2^m algorithm is obtained by choosing every m consecutive base-2 contours identical.

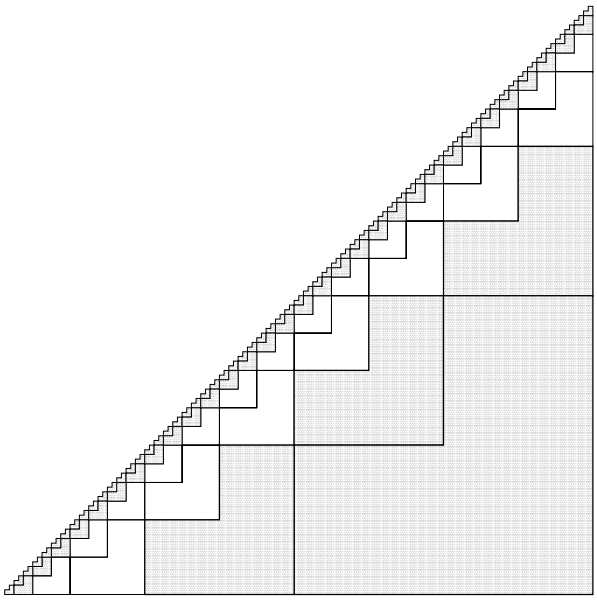

FIG. 2.6. *Tessellation for* $B = 4$.

More formally, the general base-B algorithm approximates the convolution as follows: in the nth step ($n = 1, \dots, N_t$), let $t = n\Delta t$ and approximate

$$\int_{t-\Delta t}^{t} f(t-\tau)\, g(\tau)\, d\tau \approx \phi_1\, g(t-\Delta t) + \phi_2\, \frac{g(t) - g(t-\Delta t)}{\Delta t} \tag{2.9}$$

with ϕ_1 and ϕ_2 of (2.8). Let L be the smallest integer for which $t < 2B^L \Delta t$. For $\ell = 1, 2, \dots, L-1$ determine the integer $q_\ell \geq 1$ such that

$$\tau_\ell = q_\ell B^\ell \Delta t \quad \text{ satisfies } \quad t - \tau_\ell \in [B^\ell \Delta t, (2B^\ell - 1)\Delta t].$$

Note that q_ℓ is augmented by 1 every B^ℓ steps and $t - \Delta t > \tau_1 > \cdots > \tau_{L-1} > 0$. Set $\tau_0 = t - \Delta t$ and $\tau_L = 0$. Then approximate

$$\int_0^{t-\Delta t} f(t-\tau)\, g(\tau)\, d\tau \approx \sum_{\ell=1}^{L} \sum_{j=-N}^{N} w_j^{(\ell)}\, F(\lambda_j^{(\ell)})\, e^{(t-\tau_{\ell-1})\lambda_j^{(\ell)}}\, y(\tau_{\ell-1}, \tau_\ell, \lambda_j^{(\ell)}), \tag{2.10}$$

where $w_j^{(\ell)}$ and $\lambda_j^{(\ell)}$ are the weights and quadrature points for the Talbot contour Γ_ℓ that corresponds to the base-B approximation interval $I_\ell = [B^{\ell-1}\Delta t, (2B^\ell - 1)\Delta t]$ of (2.1). Note that $[t - \tau_{\ell-1}, t - \tau_\ell] \subset I_\ell$ for all ℓ. As described for the base-2 algorithm, the differential equations determining $y(t, \tau, \lambda)$ are advanced by one step of (2.7) for all required values λ on all Talbot contours in every time step $t \to t + \Delta t$. The operation counts and memory requirements are still proportional (with the same factors as in the base-2 case) to $N_t\, L_t\, N$ and $L_t\, N$, respectively, where the number of integration contours is now bounded by $L_t \leq \log_B N_t$.

3. Schrödinger equations. In this section we illustrate the use of the convolution algorithm on Schrödinger equations (or Fresnel equations, as they are called in applications to fiber optics) on unbounded domains. We consider transparent boundary conditions for both the spatially continuous and the spatially discretized problem. The convolution algorithm for the transparent boundary conditions is combined with an implicit time discretization of the partial differential equation in the interior domain. We discuss the one-dimensional (1D) case in some detail because it shows the basics of the approach with a minimum of notational effort. We then turn to a two-dimensional (2D) problem which is periodic in one space direction and requires transparent boundary conditions in the orthogonal direction.

3.1. Transparent boundary conditions for a 1D Schrödinger equation. We consider the time-dependent Schrödinger equation for $u = u(x, t)$,

$$\frac{i}{c}\, \partial_t u = \partial_{xx} u - b(x, t, u) - \alpha^2 u\ , \qquad x \in \mathbb{R},\ t > 0, \tag{3.1}$$

where $b(x, t, u) = 0$ for $|x| \geq a > 0$ and all t and u and α and c are real constants. The differential equation is complemented with the initial condition $u(x, 0) = u_0(x)$ for $x \in \mathbb{R}$, where we also assume $u_0(x) = 0$ for $|x| \geq a$. Further, there is a radiation condition for $|x| \to \infty$, which ensures that $u(x, t) \to 0$ for $|x| \to \infty$ uniformly on bounded time intervals.

The derivation of transparent boundary conditions at $x = \pm a$ is well known (see, e.g., [6]) and proceeds formally as follows: the Laplace transform $U(x, s)$ satisfies, for $\operatorname{Re} s > 0$,

$$\frac{is}{c}\, U = \partial_{xx} U - \alpha^2 U, \qquad |x| \geq a. \tag{3.2}$$

For given Neumann data $\partial_\nu U(\pm a, s) = \pm \partial_x U(\pm a, s)$ and with the condition $U(x, s) \to 0$ for $|x| \to \infty$ this equation can be solved to give

$$U(x,s) = -\frac{1}{\sqrt{is/c + \alpha^2}} \exp\left(-|x \mp a|\sqrt{is/c + \alpha^2}\right) \partial_\nu U(\pm a, s)$$

for $x \geq a$ and $x \leq -a$, respectively. At $x = \pm a$ this yields

$$U(\pm a, s) = -\frac{1}{\sqrt{is/c + \alpha^2}} U_\nu(\pm a, s) \quad \text{or} \quad \partial_\nu U(\pm a, s) = -\frac{is/c + \alpha^2}{\sqrt{is/c + \alpha^2}} U(\pm a, s).$$

Taking the inverse Laplace transforms and denoting by $f(t)$ the function with Laplace transform $F(s) = 1/\sqrt{is/c + \alpha^2}$, we obtain the Neumann-to-Dirichlet map

$$u(\pm a, t) = -\int_0^t f(t - \tau)\, \partial_\nu u(\pm a, \tau)\, d\tau \tag{3.3}$$

and the Dirichlet-to-Neumann map

$$\partial_\nu u(\pm a, t) = -\Big(\frac{i}{c}\partial_t + \alpha^2\Big) \int_0^t f(t - \tau)\, u(\pm a, \tau)\, d\tau. \tag{3.4}$$

Either (3.3) or (3.4) forms the transparent boundary conditions for (3.1) restricted to the interior $|x| \leq a$. Numerical methods can be based on either of them, but in the following we work with (3.3) or spatially discrete variants thereof.

3.2. Transparent boundary conditions for a spatially discretized 1D Schrödinger equation. Spurious reflections introduced by the spatial discretization are avoided if the transparent boundary conditions are derived directly for the spatially discretized equation; cf. [2] and [11]. Let $2a = N_x \Delta x$ and let $\partial_{xx} u$ be discretized by the standard finite difference quotient

$$\delta_{xx} u(x,t) = \frac{u(x + \Delta x, t) - 2u(x,t) + u(x - \Delta x, t)}{\Delta x^2}$$

for $x = m\Delta x$, m an integer, so that (3.1) is replaced with

$$\frac{i}{c}\partial_t u = \delta_{xx} u - b(x,t,u) - \alpha^2 u\ , \qquad x = m\Delta x,\ t > 0. \tag{3.5}$$

Then (3.2) becomes the difference equation

$$\frac{is}{c} U = \delta_{xx} U - \alpha^2 U \qquad \text{for } \ x = \pm(a + m\Delta x),\ m \geq 0.$$

The characteristic roots of this equation are determined from

$$z^2 - \big(2 + (is/c + \alpha^2)\Delta x^2\big) z + 1 = 0. \tag{3.6}$$

Choosing $z(s)$ as the root of modulus greater than 1 for $\operatorname{Re} s > 0$, we obtain for the decaying solution of the difference equation

$$U(\pm a, s) = -\frac{\Delta x}{z(s) - 1}\, \delta_\nu U(\pm a, s),$$

where

$$\delta_\nu U(\pm a, s) = \frac{U(\pm a, s) - U(\pm(a - \Delta x), s)}{\Delta x}.$$

Transforming back gives the "discrete Neumann-to-Dirichlet map," which establishes a relation between the solution values at $\pm a$ and at $\pm(a - \Delta x)$:

$$u(\pm a, t) = -\int_0^t f(t - \tau)\, \delta_\nu u(\pm a, \tau)\, d\tau, \tag{3.7}$$

where $f(t)$ is now the function with Laplace transform

$$F(s) = \frac{\Delta x}{z(s) - 1}, \tag{3.8}$$

which is $O(\Delta x)$ close to $1/\sqrt{is/c + \alpha^2}$ uniformly for s/c in a bounded complex domain away from $i\alpha^2$ and behaves as $1/(is/c + \alpha^2)$ for $|is/c + \alpha^2| \to \infty$. Equation (3.7) gives a transparent boundary condition for the spatial semi-discretization (3.5) restricted to $|x| \le a$. (This would remain unchanged if a different, adaptive discretization were used in the interior domain away from the boundary.) The expression for $F(s)$ can be rewritten as

$$F(s) = -\frac{2}{\Delta x \sqrt{is/c + \alpha^2}\left(\sqrt{is/c + \alpha^2} + \sqrt{is/c + \alpha^2 + 4/\Delta x^2}\right)}. \tag{3.9}$$

This shows that $F(s)$ has a singularity not only at $i\alpha^2 c$, as in the continuous case, but additionally at $i(\alpha^2 + 4/\Delta x^2)$, which tends to infinity as $\Delta x \to 0$. Therefore, one would need two Talbot contours Γ_0 and Γ_1 to take account of these two singularities; see Figure 3.1 for $\alpha = 0$. However, the contribution from Γ_1 is small: Figure 3.2 and an analytic study show that it is of size $\Delta x^2/t^{3/2}$. For the analytic study we refer to the Ph.D. thesis of Achim Schädle (in preparation). Therefore, Γ_1 may be omitted in the algorithm, at least when the time step is considerably larger than Δx^2, as is typical when using an implicit time discretization scheme. For small time steps and fixed Δx, the Talbot contours are such that they enclose all singularities by construction.

3.3. Numerical experiments with a full discretization. We use (3.7), discretized in time by the convolution algorithm of section 2, as a boundary condition for the trapezoidal rule time discretization of (3.5) for $|x| < a$, which reads (denoting the fully discrete approximation again by u)

$$\frac{i}{c}\,\frac{u(x,t) - u(x,t-\Delta t)}{\Delta t} = \frac{1}{2}\Big(\delta_{xx} u(x,t) - b(x,t,u(x,t)) - \alpha^2 u(x,t) \\ + \delta_{xx} u(x,t-\Delta t) - b(x,t-\Delta t,u(x,t-\Delta t)) - \alpha^2 u(x,t-\Delta t)\Big)$$

for $x = m\Delta x$ with integer m, $|x| < a$, and $t = n\Delta t$ with integer $n \ge 1$. This yields an implicit discretization scheme. The discretized boundary condition (3.7) gives, in view of (2.9), an equation of the form

$$\left(1 + \frac{\phi_2}{\Delta t \Delta x}\right) u(\pm a, t) - \frac{\phi_2}{\Delta t \Delta x}\, u(\pm(a - \Delta x), t) = \cdots,$$

where the dots represent known values, computed from (2.9) and (2.10).

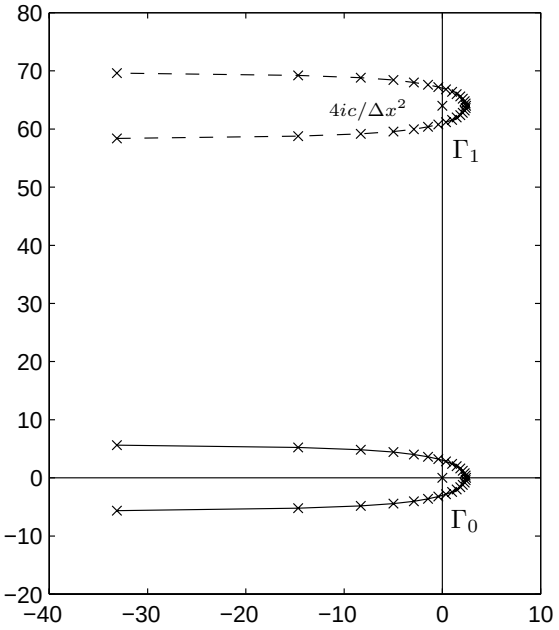

FIG. 3.1. *Two contours.*

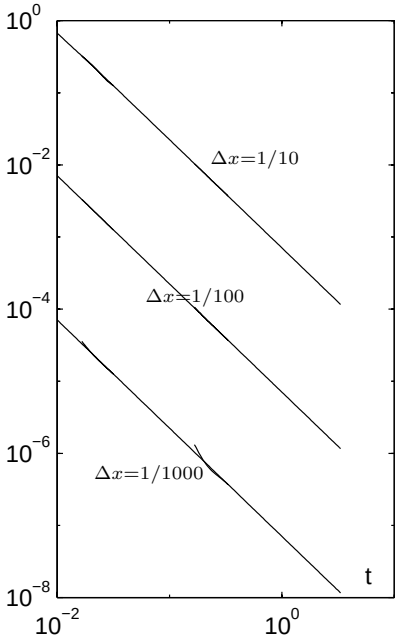

FIG. 3.2. *Contribution of* Γ_1.

For the parameter values $c = 1/14$, $a = 2$, $\alpha = 0$ and for $b \equiv 0$ we compute the evolution of a Gaussian beam shown in Figure 3.3 with starting value u_0 given by

$$u_0 = \exp\left(-4x^2 - i\frac{\tan(65^\circ)x}{2c}\right) \tag{3.10}$$

traveling left with an angle of 65°.

The decay of energy versus time, $E(t)/E(0)$ with

$$E(t) = \int_{-a}^{a} |u(x,t)|^2 dx,$$

is shown in Figure 3.4 for different convolution kernels. We used 160 grid points in x, 400 grid points in t, and we set $N = 10$ and $B = 10$ in the convolution algorithm. The parameters were chosen as in section 2.2: $\nu = 0.6$ and $\mu_0 = 8$. Figure 3.4 clearly shows that the convolution kernel for the spatially discrete equation, defined by (3.8), gives a far better approximation than the spatially continuous kernel in (3.3). It is also seen that the contribution of Γ_1 is not important and can therefore be omitted in the algorithm.

The algorithm is of order $\mathcal{O}(N_t \log(N_t))$ by construction. To verify this we plot cpu-time in seconds (Figure 3.5) and flops (Figure 3.6) as a function of N_t, indicated by stars. For these plots we have chosen $N = 10$ and $B = 10$. In each figure the two solid lines are of slope 1 and 2, respectively, and thus correspond to algorithms of order $\mathcal{O}(N_t)$ and $\mathcal{O}(N_t^2)$, and the dashed line is the function $C_1 N_t + C_2 N_t \log(N_t)$ for some constants C_1 and C_2. The implementation is in MATLAB and the cpu-time was obtained from an Intel PIII-Coppermine/600.

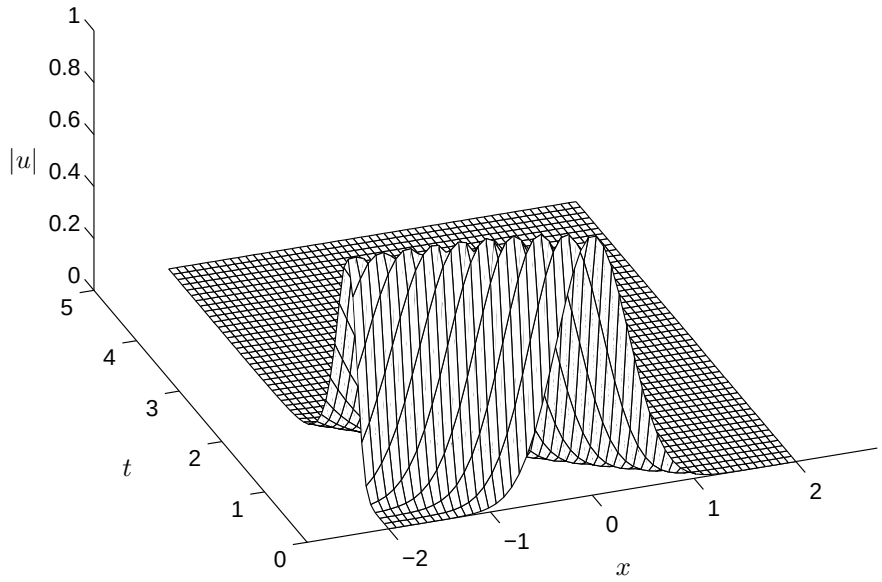

FIG. 3.3. *Evolution of a Gaussian beam.*

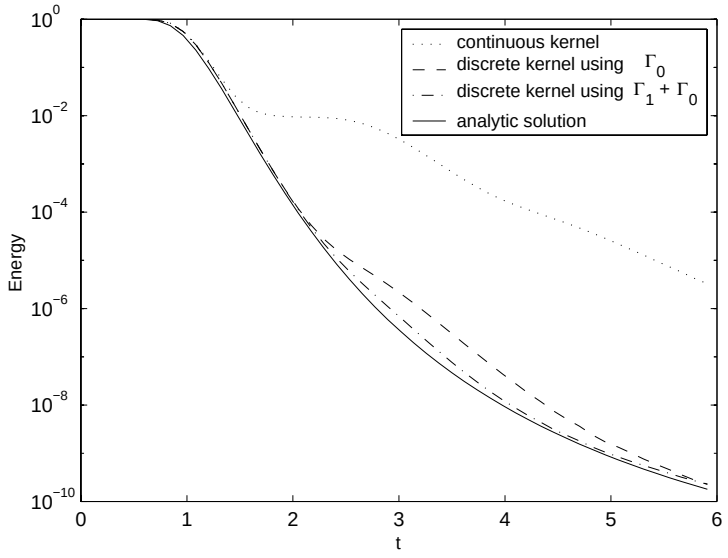

FIG. 3.4. *Decay of energy for different convolution kernels.*

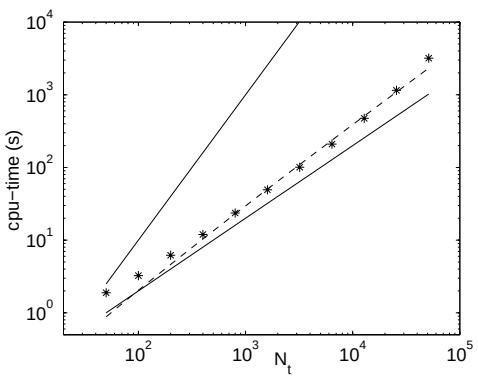

FIG. 3.5. *Cpu-time in seconds versus N_t.*

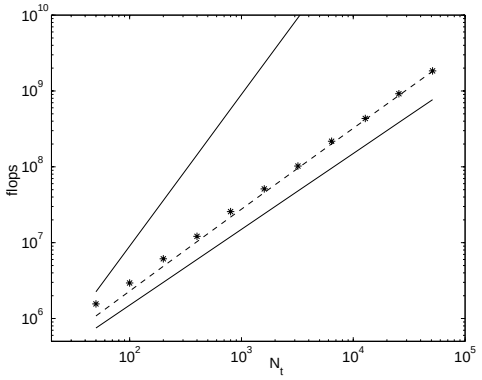

FIG. 3.6. *Flops versus N_t.*

3.4. Transparent boundary conditions for a spatially discretized 2D Schrödinger equation. We now consider the equation for $u = u(x, y, t)$,

$$\frac{i}{c}\,\partial_t u = \partial_{xx} u + \partial_{yy} u - b(x, y, t, u)\ , \qquad x, y \in \mathbb{R},\ t > 0, \tag{3.11}$$

where b is p-periodic in y and $b(x, y, t, u) = 0$ for $|x| \geq a$ and all y, t, u. Also, the initial data are assumed p-periodic in y and with support in $|x| \leq a$. The solution u is then also p-periodic in y. We consider the spatial semi-discretization

$$\frac{i}{c}\,\partial_t u = \delta_{xx} u + \delta_{yy} u - b(x, y, t, u)\ , \qquad x = m\Delta x,\ y = l\Delta y,\ t > 0, \tag{3.12}$$

for all integers m, l. We assume that the period p is a multiple of Δy: $p = N_y \Delta y$ with integer N_y. We take the discrete Fourier transform of length N_y with respect to y in (3.12) and denote the discrete Fourier coefficients by

$$\big(\widehat{u}_k(x,t)\big)_{k=-N_y/2}^{N_y/2-1} = \mathcal{F}_{N_y}\big(u(x, l\Delta y, t)\big)_{l=0}^{N_y-1}.$$

On the exterior $|x| \geq a$, (3.12) is thus transformed to

$$\frac{i}{c}\,\partial_t \widehat{u}_k = \delta_{xx}\widehat{u}_k - \alpha_k^2 \widehat{u}_k \quad \text{ with } \quad \alpha_k = \frac{2}{\Delta y}\,\sin(\pi k/N_y).$$

This decouples (3.12) on the exterior into N_y 1D problems (3.5) for the Fourier coefficients. The transparent boundary conditions at $x = \pm a$ are therefore, as in (3.7),

$$\widehat{u}_k(\pm a, t) = -\int_0^t f_k(t-\tau)\,\delta_\nu \widehat{u}_k(\pm a, \tau)\,d\tau, \tag{3.13}$$

where $f_k(t)$ is the function with Laplace transform

$$F_k(s) = \frac{\Delta x}{z_k(s) - 1}$$

and $z_k(s)$ is, for $\operatorname{Re} s > 0$, the root of modulus greater than 1 of (3.6) with $\alpha = \alpha_k$.

As in the 1D case, the convolution algorithm for (3.13) is combined with the implicit trapezoidal time discretization of (3.12) in the interior domain.

The numerical results are similar to the 1D case and therefore are not displayed.

4. Wave equations. As a further illustration of the use of the convolution algorithm, we consider transparent boundary conditions for spatially discretized 2D and 3D wave equations with periodicity in one and two space directions, respectively. The time discretization is now explicit, using the explicit leapfrog method in the interior of the domain and linear extrapolation in the convolution algorithm on the boundary.

4.1. Transparent boundary conditions for spatially discretized 2D and 3D wave equations. We consider the 2D wave equation for $u = u(x, y, t)$,

$$\frac{1}{c^2}\,\partial_{tt}u = \partial_{xx}u + \partial_{yy}u - b(x, y, t, u), \qquad x, y \in \mathbb{R},\ t > 0, \tag{4.1}$$

in a setting similar to section 3.4. We assume again that b is p-periodic in y and $b(x, y, t, u) = 0$ for $|x| \geq a$ and all y, t, u. Also the initial data u_0, $\dot{u}_0$ are assumed p-periodic in y and with support in $|x| \leq a$. The solution u is then also p-periodic in y. We discretize in space by finite differences,

$$\frac{1}{c^2}\,\partial_{tt}u = \delta_{xx}u + \delta_{yy}u - b(x, y, t, u), \qquad x = m\Delta x,\ y = l\Delta y,\ t > 0, \tag{4.2}$$

for all integers m, l, where we assume $p = N_y\Delta y$ with integer N_y. The discrete Fourier transform of length N_y with respect to y transforms (4.2) on the exterior $|x| \geq a$ to

$$\frac{1}{c^2}\,\partial_{tt}\widehat{u}_k = \delta_{xx}\widehat{u}_k - \alpha_k^2\widehat{u}_k \quad \text{ with } \quad \alpha_k = \frac{2}{\Delta y}\,\sin(\pi k/N_y).$$

The derivation of the transparent boundary conditions is now completely analogous to the Schrödinger case, formally replacing is/c by s^2/c^2 on every occurrence. The transparent boundary conditions for the Fourier coefficients at $x = \pm a$ thus become

$$\widehat{u}_k(\pm a, t) = -\int_0^t f_k(t - \tau)\,\delta_\nu\widehat{u}_k(\pm a, \tau)\,d\tau, \tag{4.3}$$

where $f_k(t)$ is the function with Laplace transform

$$F_k(s) = \frac{\Delta x}{z_k(s) - 1}$$

and $z_k(s)$ is, for $|s|$ large, the root of modulus greater than 1 of

$$z^2 - \big(2 + (s^2/c^2 + \alpha_k^2)\Delta x^2\big)z + 1 = 0. \tag{4.4}$$

The above extends straightforwardly to spatially discretized 3D wave equations with periodicity in the y- and z-directions,

$$\frac{1}{c^2}\,\partial_{tt}u = \delta_{xx}u + \delta_{yy}u + \delta_{zz}u - b(x, y, z, t, u), \tag{4.5}$$

for $x = m\Delta x$, $y = l\Delta y$, $z = j\Delta z$, $t > 0$. This now involves 2D discrete Fourier transforms in y and z, with Fourier coefficients $\widehat{u}_k(x, t)$ for $k = (k_y, k_z)$. The transparent boundary conditions are still of the above form, with

$$\alpha_k^2 = \left(\frac{2}{\Delta y}\,\sin(\pi k_y/N_y)\right)^2 + \left(\frac{2}{\Delta z}\,\sin(\pi k_z/N_z)\right)^2.$$

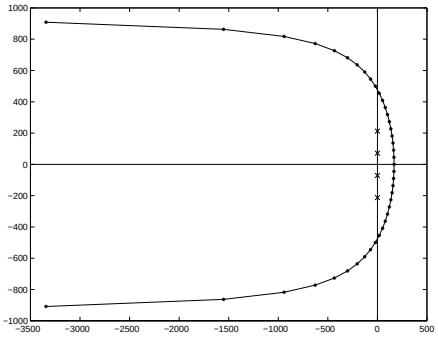

FIG. 4.1. *One contour enclosing all singularities.*

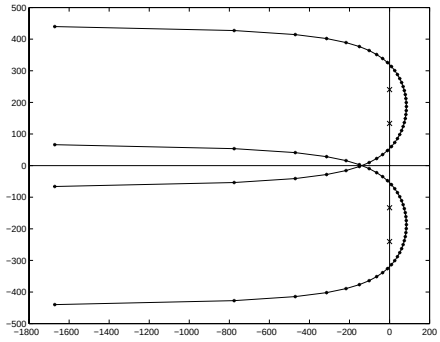

FIG. 4.2. *Two contours enclosing all singularities.*

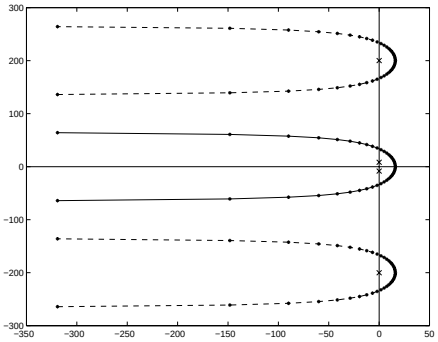

FIG. 4.3. *One contour enclosing "main" singularities.*

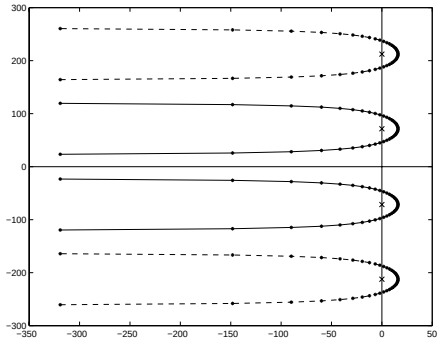

FIG. 4.4. *Two contours enclosing "main" singularities.*

We remark that this double-periodic geometry appears in studying wave propagation in the neighborhood of thin layers. (A corresponding problem for the Maxwell equations is currently under investigation.)

As in the Schrödinger case we rewrite the expression for $F_k(s)$

$$F_k(s) = -\frac{2}{\Delta x\sqrt{(s/c)^2+\alpha^2}\left(\sqrt{(s/c)^2+\alpha^2}+\sqrt{(s/c)^2+(\alpha^2+4/\Delta x^2)}\right)}, \tag{4.6}$$

which clearly shows that it has singularities not only in $\pm ic\alpha$ as the continuous kernel but also in $ic\sqrt{\alpha^2+4/\Delta x^2}$. Thus we have to develop a new strategy for choosing the Talbot contours with parameters μ, ν, and σ. Setting $\sigma = 0$, $\mu = 8/T_l$, $\nu_0 = 0.6$, and $\beta = c\pi\mu\nu_0/2$ we choose one contour if $\alpha < \beta$. If, in addition, $\beta < (\sqrt{\alpha^2+4/\Delta x^2} - \alpha)/2$ we set $\nu = \nu_0(1+\alpha/\beta)$, thus enclosing only the singularities $\pm ic\alpha$, or else we set $\nu = \nu_0(1+\sqrt{\alpha^2+4/\Delta x^2}/\beta)$, thus enclosing all singularities. If $\alpha > \beta$, we choose two contours. If, in addition, $\beta < (\sqrt{\alpha^2+4/\Delta x^2} - \alpha)/2$ we set $\nu = \nu_0$ enclosing only the singularities $\pm ic\alpha$, or else we set $\nu = \nu_0(1+(\sqrt{\alpha^2+4/\Delta x^2} - \alpha)/(2\beta))$. The situation is illustrated in Figures 4.1–4.4, where the solid lines correspond to Γ_0 and the dashed lines to Γ_1, similarly as in Figure 3.1.

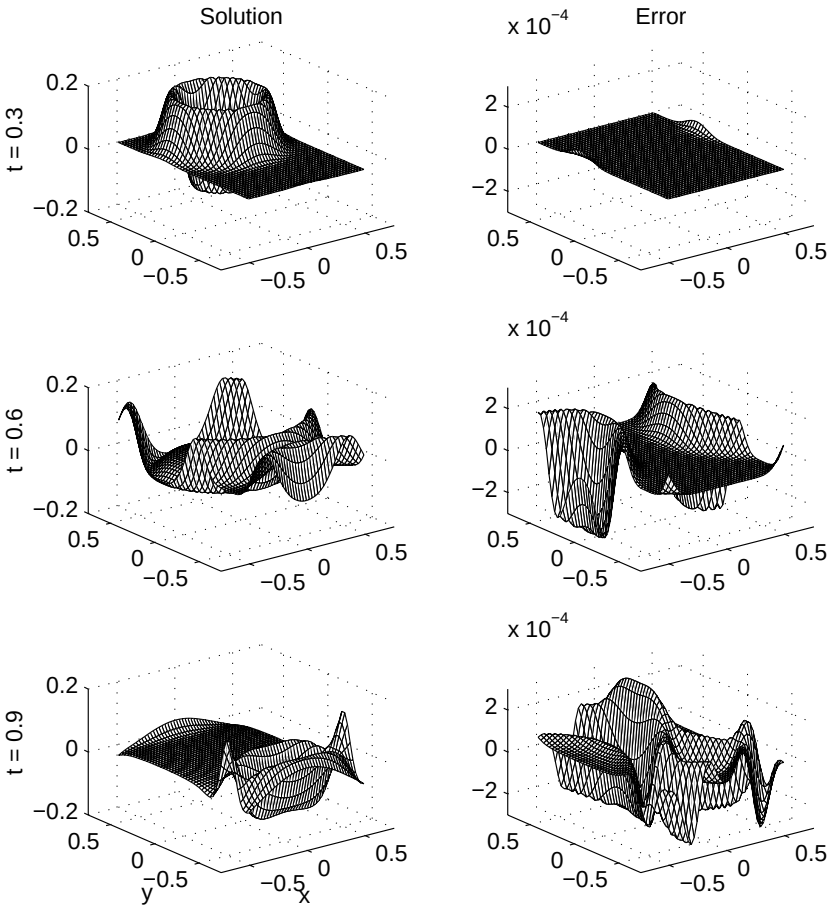

FIG. 4.5. *Solution and error.*

4.2. Numerical experiments with a full discretization. For time discretization in the interior domain we use the Störmer/leapfrog method

$$\frac{1}{c^2}\frac{u(x,y,t+\Delta t)-2u(x,y,t)+u(x,y,t-\Delta t)}{\Delta t^2} \\ = \delta_{xx}u(x,y,t)+\delta_{yy}u(x,y,t)-b(x,y,t,u(x,y,t))$$

for $x = m\Delta x$ with $|x| < a$, $y = l\Delta y$, and $t = n\Delta t$. The boundary values at $x = \pm a$ are obtained from the convolution algorithm applied to the transparent boundary conditions (4.3) where, in the approximation of $\widehat{u}_k(\pm a, t+\Delta t)$, we approximate $\delta_\nu \widehat{u}_k(\pm a, \tau)$ for $t \le \tau \le t+\Delta t$ by linear extrapolation from the known values at $t-\Delta t$ and t. This yields a fully explicit discretization scheme.

With $c = 1$, $a = 0.5$, period 1.5 in y, and with starting value

$$u(0) = \exp\left(-\frac{x^2+(y-0.25)^2}{0.01}\right), \quad \partial_t u(0) = 0,$$

we compute the movie of Figure 4.5. Here the error is given with respect to a reference solution computed on a grid with the same Δx, Δy, and Δt but on a domain whose

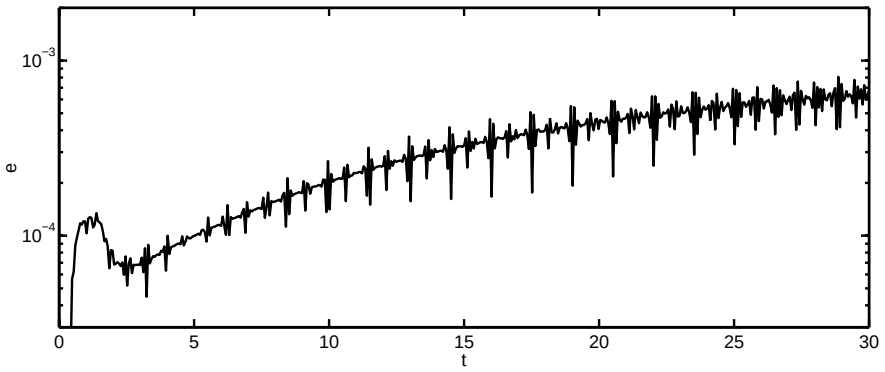

FIG. 4.6. *Evolution of the error.*

size in x is doubled. In the algorithm we are using Γ_0 and Γ_1 such that the only errors introduced are spurious reflections at the artificial boundary that emerge from using different time discretizations in the interior domain and in the boundary condition and from the quadrature approximations of the contour integrals. In our computation we have chosen $\Delta x = 1/81$, $\Delta y = 1/64$, $\Delta t = 1/200$, and the parameters in the convolution algorithm as $B = 5$, $N = 10$, $\nu_0 = 0.6$, and $\mu_0 = 8$. Almost identical figures were obtained with $N = 15$ and $B = 5$ or $B = 10$. Omitting the contributions of the dashed contours in Figures 4.3 and 4.4 gave errors about five times larger and made the algorithm more sensitive to changes of the parameters. We made experiments also with different time steps and different spatial grids and observed stability up to the stability limit of the leapfrog scheme of the interior discretization.

In Figure 4.6 we show the evolution of the error norm

$$e(t) = \left(\frac{1}{N_x N_y} \sum_{ij} (u_{ij}(t) - u_{ij}^{ref}(t))^2 \right)^{1/2}$$

over a long time interval $0 \le t \le 30$ for $B = 5$ and $N = 15$, and $\Delta x = 1/97$, $\Delta y = 1/85$, and $\Delta t = 1/200$. The computation used only Γ_0; using both Γ_0 and Γ_1 resulted in an almost identical error curve. Reducing the time step by powers of 2 gave a reduction of the error curves by approximately the same power of 4, as expected for a second order scheme.

We also implemented the double periodic 3D case and obtained similar results (at least on the shorter time intervals on which a computation of the error was still feasible).

Acknowledgment. We thank Gerhard Wanner for stimulating discussions and for helpful comments on an earlier draft of this paper.

REFERENCES

[1] B. ALPERT, L. GREENGARD, AND T. HAGSTROM, *Rapid evaluation of nonreflecting boundary kernels for time-domain wave propagation*, SIAM J. Numer. Anal., 37 (2000), pp. 1138–1164.

[2] A. ARNOLD, *Numerically absorbing boundary conditions for quantum evolution equations*, VLSI Design, 6 (1998), pp. 313–319.

[3] D. GIVOLI, *Numerical Methods for Problems in Infinite Domains*, Elsevier, Amsterdam, 1992.

[4] M. J. Grote and J. B. Keller, *Exact nonreflecting boundary conditions for the time dependent wave equation*, SIAM J. Appl. Math., 55 (1995), pp. 280–297.
[5] M. J. Grote and J. B. Keller, *On non-reflecting boundary conditions*, J. Comput. Phys., 122 (1995), pp. 231–243.
[6] T. Hagstrom, *Radiation boundary conditions for numerical simulation of waves*, Acta Numer., 8 (1999), pp 47–106.
[7] E. Hairer, Ch. Lubich, and M. Schlichte, *Fast numerical solution of nonlinear Volterra convolution equations*, SIAM J. Sci. Statist. Comput., 6 (1985), pp. 532–541.
[8] Ch. Lubich, *Convolution quadrature and discretized operational calculus* I, Numer. Math., 52 (1988), pp. 129–145.
[9] Ch. Lubich, *Convolution quadrature and discretized operational calculus* II, Numer. Math., 52 (1988), pp. 413–425.
[10] M. Rizzardi, *A modification of Talbot's method for the simultaneous approximation of several values of the inverse Laplace transform*, ACM Trans. Math. Software, 21 (1995), pp. 347–371.
[11] F. Schmidt and D. Yevick, *Discrete boundary conditions for Schrödinger-type equations*, J. Comput. Phys., 134 (1997), pp. 96–107.
[12] A. Talbot, *The accurate numerical inversion of Laplace transforms*, J. Inst. Math. Appl., 23 (1979), pp. 97–120.