Electronic Transactions on Numerical Analysis.
Volume 6, pp. 162-181, December 1997.
Copyright © 1997, Kent State University.
ISSN 1068-9613.

ETNA
Kent State University
etna@mcs.kent.edu

WAVE-RAY MULTIGRID METHOD FOR STANDING WAVE EQUATIONS[*]

A. BRANDT[†] AND I. LIVSHITS[†]

Abstract. Multigrid methods are known for their high efficiency in the solution of definite elliptic problems. However, difficulties that appear in highly indefinite problems, such as standing wave equations, cause a total loss of efficiency in the standard multigrid solver. The aim of this paper is to isolate these difficulties, analyze them, suggest how to deal with them, and then test the suggestions with numerical experiments. The modified multigrid methods introduced here exhibit the same high convergence rates as usually obtained for definite elliptic problems, for nearly the same cost. They also yield a very efficient treatment of the radiation boundary conditions.

Key words. Helmholtz equations, multigrid methods, wave-ray approach, radiation boundary conditions.

AMS subject classifications. 65N55, 65N06, 65N22, 65B99.

1. Introduction.

What are the properties of highly indefinite problems that make their solution by standard multigrid methods inefficient?

First, there is the well-known limitation on the coarsest-grid mesh-size, which should be sufficiently fine, in fact much finer than the wavelength of the oscillatory solution, to avoid large phase errors (cf. Sec. 6 below).

To solve these coarsest grid equations one has to apply the slowly converging Kaczmarz relaxation (or some similar procedure): the faster Gauss-Seidel relaxation causes smooth components of the error to diverge. As a result, the solution will cost $O(N^3)$ operations (in two-dimensional problems), where N is the number of coarse grid points.

Another basic difficulty is posed by the nonlocal character of the *radiation boundary conditions* (RBC) that usually accompany highly indefinite equations, making their discrete expression extremely costly.

The basic approach which has guided the present work was already stated in Sec. 3.2 of [2] and in [7]. It is based on the fact that the problematic error components (the ones which cannot be reduced by the standard multigrid process applied to the Helmholtz equation) can be factorized by representing it as the product of a certain high-frequency Fourier component and a smooth envelope function (a ray function). The idea is then to reduce this type of error by approximating these smooth envelope functions on the coarse grids.

However, a substantial number of important algorithmic aspects still had to be clarified or invented, especially with regard to the levels of transition between wave and ray representation (such as the use of rotated coordinates for the ray representation, increasing a number of ray functions on the finest ray grid, optimal scaling, optimal weighting, etc.).

We started our work with the model one-dimensional Helmholtz equation, with different types of coefficients: constant, smooth and discontinuous. For this problem we developed an efficient multigrid solver [6]. In the present paper we describe the next step of our research: A solver for the two-dimensional Helmholtz equations with constant coefficients. The solver is developed in the *Full Approximation Scheme* (FAS) multigrid version, and for full understanding the reader is well advised to acquire some familiarity with this version (see, e.g., [1] or [3]).

The presented approach can clearly be extended to higher-dimensional problems. The extension to variable coefficients is briefly discussed in Sec. 10. The approach developed here for obtaining a fast multigrid solver can also be used in a new type of setting where only geometrical optics (ray tracing) needs to be used throughout a very large problem domain,

[*] Received May 16, 1997. Accepted August 1, 1997. Communicated by J. Dendy.

[†] Weizmann Institute of Science, Rehovot 76100, Israel.

whereas the full, costly wave equations (as well as suitable intermediate levels, introduced herein) can be restricted to special small subdomains where the pure geometrical optics representation breaks down (see Sec. 10).

2. Characteristic Components. For simplicity, our whole discussion is restricted to a model problem – the scalar two-dimensional standing wave equation (Helmholtz equation)

$$\Delta u(x,y) + k^2 u(x,y) = f(x,y), \qquad (x,y) \in \Re^2, \tag{2.1}$$

with constant coefficient k, where $f(x,y)=0$ outside some compact set $\Omega_f \subset \Re^2$.

The purpose is to construct an efficient multigrid solver for discretized standing wave equations, e.g., with the second-order discretization

$$\frac{u^h_{i-1,j} - 2u^h_{i,j} + u^h_{i+1,j}}{h^2} + \frac{u^h_{i,j-1} - 2u^h_{i,j} + u^h_{i,j+1}}{h^2} + k^2 u^h_{i,j} = f_{i,j}, \tag{2.2}$$

where $u_{i,j} \approx u(ih,jh)$ and $f_{i,j} = f(ih,jh)$, with the radiation boundary conditions. The problem is considered in a computational domain Ω, whose exact size will be discussed later, but it clearly should contain Ω_f.

We focus here on the *highly* indefinite problem, meaning that the wavelength $\frac{2\pi}{k}$ of the solutions to the homogeneous equations is much smaller than the diameter d of Ω, i.e., $dk \gg 1$. Otherwise the problem can already be solved by simpler modifications to the standard multigrid solver (see [5]).

To develop an efficient solver, we need to satisfy the basic multigrid rule: Each Fourier error component needs an appropriate grid on which it is treated efficiently. This does not happen straightforwardly.

Let us first note that any Fourier component of the form $e^{i(k_1x+k_2y)}$, with $k_1^2+k_2^2 = k^2$, satisfies the homogeneous ($f \equiv 0$) Helmholtz equation (2.1). These components will be called here the *principal components*, and their frequencies $\mathbf{k} = (k_1,k_2)$ will be called *principal frequencies*. In the plane of frequencies, the circle of principal frequencies will be called the *principal circle*. To have an efficient discretization for principal components, and also for components with frequencies close to the principal circle (we call such components *characteristic*), we discretized the principal circle, e.g., by a uniform *lattice* of L *lattice frequencies*

$$\mathbf{k}^l = (k^l_1, k^l_2) = (k\cos[\theta(l-1)+\theta_0], k\sin[\theta(l-1)+\theta_0]), \quad l = 1,\ldots,L, \tag{2.3}$$

where $\theta = \frac{2\pi}{L}$ and $0 \le \theta_0 < \theta$ (see Sec. 8).

Let us consider a solution of the homogeneous equation (2.1) in the following form

$$u(x,y) = \int_0^{2\pi} U_t e^{ik(x\cos(t)+y\sin(t))} dt$$

or, equivalently,

$$u(x,y) = \sum_{l=1}^{L} \hat{u}_l(x,y) e^{i(k^l_1 x + k^l_2 y)}, \tag{2.4}$$

where the functions $\hat{u}_l(x,y)$ are not uniquely defined but always can be chosen so that they are a combination of Fourier components with frequencies smaller or comparable to $k\theta$, $\theta = \pi/L$. Note, that if we consider (2.4) with a larger L, θ becomes smaller, and the coefficient functions $\hat{u}_l(x,y)$ in the expansion (2.4) become smoother.

With the discretization (2.2) or any other discretization of (2.1), there is *no* grid on which characteristic components can be treated efficiently. On the fine grids, where they are accurately approximated by the discrete equations, they are invisible to any local relaxation since their errors can have very small residuals. Indeed, the size of such components is not determined locally, on the scale of the fine mesh-size, but on a much larger scale. On the other hand, on coarser grids such components cannot be approximated, because the grid does not resolve their oscillations. (In fact, to approximate such components by (2.2) the mesh-size h needs to be *much* smaller than the wavelength $\frac{2\pi}{k}$; to avoid large phase error accumulated over the domain diameter d $hk \ll O((dk)^{-1/2})$ must be satisfied.)

Thus, there is a need for an alternative approach for reducing characteristic error components. Ours is based on the fact that the error $v(x,y)$ that cannot be reduced by a usual multigrid cycle can be represented, similarly to (2.4), by

$$v(x,y) = \sum_{l=1}^{L} \hat{v}_l(x,y) e^{i(k_1^l x + k_2^l y)},$$

where, by choosing sufficiently large L, the $\hat{v}_l(x,y)$ are smooth enough to be approximated on coarse levels.

Turning to optical terminology, we call the functions $\hat{u}_l(x,y)$ and $\hat{v}_l(x,y)$ the *ray functions*, and the equations satisfied by them the *ray equations*. The grids on which the wave equations and the ray equations are treated are called the *wave grids* and the *ray grids*, respectively.

To reduce the error (2.4), in addition to the usual multigrid cycles on wave grids, we use ray cycles: They include recursive derivation of ray equations on increasingly coarser ray grids, having increasingly finer lattices (larger L). The equations on each grid are based on the residuals of the previous (next *finer*) grid, except that boundary conditions are defined and interpolated from the next *coarser* grid. The resulting ray equations are relaxed on each level; On the coarsest ray level they are solved and the radiation boundary conditions are imposed, facilitated by the nearly pure ray representation (very smooth $\hat{v}_l$, with large L) obtained on sufficiently coarse levels.

3. Ray Levels: Grids and Lattices. Any function on the n-th level of the ray cycle $(n = 1, \ldots, N)$ has the representation

$$u^n = \sum_{l=1}^{L^n} \hat{u}_l^n(x,y) e^{i(k_1^l x + k_2^l y)}, \tag{3.1}$$

where the (k_1^l, k_2^l) are given by (2.3) with $L = L^n$.

For each principal component $e^{i(k_1 x + k_2 y)}$, we define a rotated Cartesian coordinate system (ξ, η), such that the direction ξ is parallel to the vector (k_1, k_2). In these coordinates, this principal component is of the form $e^{ik\xi}$, i.e., ξ is its propagation direction. The ray function $\hat{u}_l^n(x,y)$ will be discretized on a correspondingly rotated *grid*, with mesh-size h_ξ^n in the propagation direction and mesh-size h_η^n in the perpendicular direction. The levels are enumerated so that $n = N$ is the level with the coarsest grid (largest $\mathbf{h}^n = (h_\xi^n, h_\eta^n)$ and finest lattice, i.e., largest L^n).

An advantage of working with the rotated grids is that identical and relatively simple procedures can be used for different ray functions.

Another, more basic advantage is the possibility of using the most economical ray grids. The actual mesh and lattice sizes are determined by requiring that at the highest level $(n = 1)$

we have $h_\xi^n = 2C/k$ and $h_\eta^n = C/k$ and $L^n = 8$, where typically $1.25 \lesssim C \lesssim 2.5$. This implies that $h_\xi^n = C(L^n)^2/32k$ and $h_\eta^n = CL^n/8k$.

For $n > 1$, the size of the lattice and the mesh-sizes are calculated as

$$L^n = \begin{cases} 2L^{n-1} & \text{if} \quad n \quad \text{is even,} \\ L^{n-1} & \text{if} \quad n \quad \text{is odd,} \end{cases}$$

$$h_\xi^n = 2h_\xi^{n-1},$$

$$h_\eta^n = \begin{cases} 2h_\eta^{n-1} & \text{if} \quad n \quad \text{is even,} \\ h_\eta^{n-1} & \text{if} \quad n \quad \text{is odd,} \end{cases}$$

or, in terms of L^n,

$$h_\xi^n = \begin{cases} C(L^n)^2/64k & \text{if} \quad n \quad \text{is even,} \\ C(L^n)^2/32k & \text{if} \quad n \quad \text{is odd,} \end{cases}$$

$$h_\eta^n = \frac{CL^n}{8k}.$$

When the number of lattice points is increased by factor two, the maximal coarsening of the mesh-sizes is chosen to satisfy $h_\eta^{-2} = O(h_\xi^{-1}k)$ (i.e, h_ξ becomes four times and h_η – two times coarser before the next lattice refinement), hence the two main terms of the ray equation (see (7) below) will have the same mesh coupling, making the equation "h-elliptic" (see [3]). This choice of coarsening corresponds to the fact that ray solutions $\hat{u}_l^n$ that we need are much smoother in the propagation direction than perpendicularly, in a way which is exactly exploited by having $h_\eta^2 \approx h_\xi/k$. Also, and most important, these mesh-sizes enable a representation of a given ring of frequencies (specifically: a ring of width $O(h_\xi^{-1})$ around the principal circle) using a *minimal number* of ray components (minimal L^n), i.e., using a maximal distance between lattice points. This type of coarsening is also important for an effective and accurate *separation* of the ray components from each other (see Sec. 4).

The rectangular domain covered by the grid on which $\hat{u}_l^n$ is defined, as well as the grid itself, will be denoted Ω_l^n; it is formed from a basic rectangle $\Omega^n = \{(x,y) : \|x| \leq d_1^n/2, |y| \leq d_2^n/2\}$ by rotating it $\frac{2\pi l}{L^n}$ radians. At the highest level, the sizes d_1^1 and d_2^1 are chosen so that Ω_l^1 all contain the wave computational domain on level M_r, which will be designated as the "wave-to-ray switching level"; its exact choice will be discussed in Sec. 4. At each lower ($n > 1$) level the d_1^n and d_2^n are defined so that each Ω_l^n completely includes the associated higher-level rectangles ($\Omega_{l/2}^{n-1}$ if l is even and both $\Omega_{(l-1)/2}^{n-1}$ and $\Omega_{(l+1)/2}^{n-1}$ if l is odd, where $l \pm 1$ is taken modulo L^{n-1}), with at least four additional mesh-sizes in each direction.

4. Ray Equations and Separation. In this section our description will be given in terms of a *Correction Scheme* (CS) multigrid version which will however change in Sec. 5 below. This means that at each lower ray level n, the represented ray functions are designed in turn to approximate the *correction* needed at the next higher level $n-1$, while the functions computed on the highest ray level 1 are designed to approximate the correction needed for the solution on the wave level $m = M_r$ of the "wave multigrid cycle".

The error component that needs to be reduced on a ray grid has the form

$$v(x,y) = \hat{v}(x,y)e^{i(k_1 x + k_2 y)}, \tag{4.1}$$

where $\hat{v}(x,y)$ is smooth. A right-hand-side, that corresponds to each such error component, can similarly be represented as

$$r(x,y) = \hat{r}(x,y)e^{i(k_1x+k_2y)}, \tag{4.2}$$

and it, as we will see, actually approximates corresponding *residuals* on the next finer level.

Substitution of (4.1) and (4.2) into Eq. (2.1) (with v being the solution and r the right-hand-side) gives us an equation for the ray function $\hat{v}(x,y)$.

$$\hat{L}\hat{v}(x,y) = \hat{v}_{xx}(x,y) + \hat{v}_{yy}(x,y) + 2ik_1\hat{v}_x(x,y) + 2ik_2\hat{v}_y(x,y) = \hat{r}(x,y).$$

In the rotated coordinates it simplifies to

$$\hat{L}\hat{v}(\xi,\eta) = \hat{v}_{\xi\xi}(\xi,\eta) + \hat{v}_{\eta\eta}(\xi,\eta) + 2ik\hat{v}_\xi(\xi,\eta) = \hat{r}(\xi,\eta). \tag{4.3}$$

The operator $\hat{L}$ will be approximated by the following second-order finite-difference stencil:

$$\hat{L}^h = \begin{pmatrix} & \frac{1}{2h_\eta{}^2} & \frac{1}{2h_\eta{}^2} & \\ \frac{1}{2h_\xi{}^2} & (-\frac{1}{h_\eta{}^2} - \frac{2ik}{h_\xi} - \frac{1}{2h_\xi{}^2}) & (-\frac{1}{h_\eta{}^2} + \frac{2ik}{h_\xi} - \frac{1}{2h_\xi{}^2})^* & \frac{1}{2h_\xi{}^2} \\ & \frac{1}{2h_\eta{}^2} & \frac{1}{2h_\eta{}^2} & \end{pmatrix}, \tag{4.4}$$

which is centered at the mid-point (ξ,η); for orientation, $*$ marks the coefficient at the grid-point $(\xi + h_\xi/2, \eta)$. This discretization has several advantages: First, the symbol of the operator defined by (4.4) proves to be close to the symbol of the finest-grid discrete wave operator for the desired characteristic components, i.e, the ray operators (4.4) provide an excellent approximation to (2.2) within a minimal number of lattice points, meaning a minimal number of ray functions. Another important property of (4.4), which makes this discretization attractive, is that its symbol is bounded away from zero (i.e., it is stable) for all other components.

The ray equation (4.3) is almost first-order in ξ since the term $\hat{v}_{\xi\xi}$ is small compared to $k\hat{v}_\xi$ for any function visible on the grid. Hence, the ray equations (4.4) can almost be solved by one sweep of a suitable line relaxation, costing only a number of operations proportional to the number of grid points. Thus, unlike regular multigrid cycles, in the ray cycles (in two dimensions) there is no need to visit the coarsest possible levels in order to save computational work — the cost of solving ray equations on any ray grid is comparable to the cost of one relaxation sweep there. The coarsest ray level is thus mainly determined by other considerations, discussed later.

One purpose of the ray cycles is to approximate the smooth ray functions that correspond to those residuals left unreduced after the wave cycle. This approximation can be done if each ray grid obtains *its own* appropriate part of the residuals, i.e., a residual function of the form (4.2), where $\hat{r}(x,y)$ is sufficiently smooth to be well approximated on that grid. Hence, we need a procedure that for such a wave residual function $r(x,y)$ in a characteristic ring calculates smooth (on scale $\mathbf{h}^n$) functions $\hat{r}_l^n(x,y)$ so that

$$r(x,y) = \sum_{l=1}^{L^n} \hat{r}_l^n(x,y)e_l^n(x,y), \quad e_l^n(x,y) = e^{i(k_1^l x + k_2^l y)},$$

where the (k_1^l, k_2^l) are defined as in (2.3), with $L = L^n$. We call such a procedure *separation*, and we first describe it here in general terms.

A basic tool for the separation is a simple one-dimensional three-point weighting (convolution) operator $W = (w_0, w_1, w_0)$. Consider a one-dimensional function $g(\alpha)$ on a grid with a mesh-size h having the form

$$g(\alpha) = a_1(\alpha)e^{-ip\alpha} + a_2(\alpha) + a_3(\alpha)e^{ip\alpha}, \tag{4.5}$$

where the $a_j(\alpha)$ are smooth, compared to $e^{\pm ip\alpha}$, functions and $\pi/2 \lesssim ph \leq \pi$. The weighting coefficients w_0 and w_1 are chosen so that the result of applying the convolution weighting operator to g

$$(Wg)(\alpha) = w_0 g(\alpha - h) + w_1 g(\alpha) + w_0 g(\alpha + h)$$

is an approximation to the function $a_2(\alpha)$; Specifically,

$$Wg \equiv g, \quad \text{if} \quad g \equiv const,$$

$$Wg \equiv 0, \quad \text{if} \quad g(\alpha) = e^{\pm ip\alpha}.$$

It is easy to find ω_0 and ω_1 that satisfy these conditions.

A *two-dimensional* weighting operator can be constructed as a tensor product of two one-dimensional operators: It approximates the functions which are smooth in both directions and nearly annihilates some high-frequency components.

More precisely, consider a function $g(\xi, \eta)$ which can be represented as

$$g(\xi, \eta) = \sum_l \hat{g}_l(\xi, \eta) e^{i(p_1^l \xi + p_2^l \eta)}, \tag{4.6}$$

defined in the (ξ, η) coordinate system, where the $\mathbf{p}^l = (p_1^l, p_2^l)$ are some given frequencies, sufficiently remote from each other, and $\hat{g}_l(\xi, \eta)$ are smooth functions. We assume that g is defined on a grid Ω_0 with mesh-size $\mathbf{h} = (h_\xi, h_\eta)$, and our purpose is to approximate $\hat{g}_l$ on some grid Ω_J with a coarser mesh-size in the (ξ_l, η_l) coordinate system. We also suppose that a sequence of grids in the (ξ_l, η_l) coordinates is given

$$\Omega_1 \rightarrow \Omega_2 \rightarrow \ldots \rightarrow \Omega_J,$$

where the vector mesh-size of Ω_1 is approximately $\mathbf{h}$ and the relation $\Omega_{j-1} \rightarrow \Omega_j$ means that Ω_j coincides with Ω_{j-1} or that it is obtained from Ω_{j-1} by doubling the mesh-size in at least one direction. An approximation to $\hat{g}_l$ is then evaluated as follows:

$$\hat{g}_l \approx W_{J-1}^J(W_{J-2}^{J-1}(\ldots(W_1^2(I_{(\xi,\eta)}^{(\xi_l,\eta_l)}[g(\xi,\eta)e^{-i(p_1^l\xi + p_2^l\eta)}]))\ldots)), \tag{4.7}$$

where $I_{(\xi,\eta)}^{(\xi_l,\eta_l)}$ is an interpolation operator from grid Ω_0 to grid Ω_1, and each $W_{j-1}^j : \Omega_{j-1} \rightarrow \Omega_j$ is a two-dimensional weighting operator, i.e., a tensor product of two one-dimensional weighting operators. Each of the latter can be applied in either the ξ_l, or η_l, or one of the two diagonal directions (diagonal means employing either the left-lower and the right-upper, or the left-upper and the right-lower diagonally neighboring grid-points).

What are the actual weighting directions we chose in our algorithm? Assume that on the current fine grid Ω_{j-1}, a function g to which a weighting is applied can be represented as

$$g(\alpha, \beta) = g_s(\alpha, \beta) + e^{ip\alpha} g_h(\alpha, \beta),$$

where α is either ξ_l, η_l or one of the diagonal directions; β is perpendicular to α; g_s is a smooth function of α on the fine-grid scale and should be represented on the next coarser grid, while $e^{ip\alpha}g_h(\alpha,\beta)$ is a high-frequency function of α and should not be transferred to the next coarser grid (because it is either too high-frequency there and aliases with smooth components, or it "belongs" to another neighboring ray function). If all of the above is true, the weighting needs to be applied in the α direction.

To construct the CS right-hand-side $\hat{r}_l^1$ on the *finest* ray grid, the residual function on a *wave* grid with mesh-size $O(k^{-1})$ serves as g; the wave level used for this purpose is denoted M_r. (See details of calculating $\hat{r}_l^1$ in the Appendix A.)

To approximate a right-hand side $\hat{r}_l^n$ on coarser ($n > 1$) ray levels for the l-th problem, g is taken from the next finer $(n-1)$ ray level: Specifically,

$$g(\xi,\eta) = \hat{R}_{l/2}^{n-1}, \quad \text{if} \quad l \quad \text{is even}$$

and

$$g(\xi,\eta) = \left[e_{(l-1)/2}^{n-1}\hat{R}_{(l-1)/2}^{n-1} + e_{(l+1)/2}^{n-1}\hat{R}_{(l+1)/2}^{n-1} \right] / e_l^n, \quad \text{if} \quad l \quad \text{is odd},$$

where $\hat{R}_j^{n-1}$ is the residual function calculated on the $(n-1)$ level for the j-th ray problem. (See further details in Sec. 8.)

5. FAS and RBC.

The discussion so far has been in terms of the CS multigrid version. Namely, each level has represented a *correction* to a finer level. However, the need to introduce the radiation boundary conditions (RBC) on the coarsest ray grids, and the need to use larger domains for coarser levels, imply the use of the *Full Approximation Scheme* (FAS). As in previous works (see, e.g., [1] or [3]) the difference between the CS and the FAS is that instead of the *correction* $v^m(x,y)$ (the function which is eventually interpolated to a finer (higher) level and corrects its current approximation), the function for which the coarse-grid equations are directly written is $u^m(x,y) = v^m(x,y) + \overline{u}^m(x,y)$, where $\overline{u}^m(x,y)$ represents some known fixed approximation to the *current solution*, so that $u^m(x,y)$ is actually the intended *full solution*.

On a wave grid, as in previous works, $\overline{u}^m$ is taken to be $I_{m+1}^m(u^{m+1})$ in regions where the latter can be defined; here I_{m+1}^m is some fine-to-coarse transfer (injection or averaging) and u^{m+1} is the current solution on the next finer level. In other regions (outside the domain where u^{m+1} is defined) $\overline{u}^m$ is taken to be the latest approximation u^m obtained during the last "visit" to level m. In either case, this $\overline{u}^m$ is indeed fixed throughout the current "visit" to level m. (By a *visit to level* m we mean all the processing on level m *and on lower levels* (coarser grids) which takes place between a switching from level $m+1$ to level m and the first following switching back from level m to level $m+1$. The purpose of such a visit is the calculation of the correction v^m.)

On a ray level n, however, it is more appropriate to take the initial approximation to the ray function $\overline{\hat{u}}_l^n(\xi,\eta)$ to be everywhere the value of $\hat{u}_l^n(\xi,\eta)$ at the *previous visit* to this very level. This ensures that the boundary conditions (which are brought here from *lower* levels) remain satisfied, and it saves us from the need to separate the solution approximation. Thus, instead of the CS ray correction function $\hat{v}_l^n(\xi,\eta)$, the solution function $\hat{u}_l^n(\xi,\eta)$ actually calculated and stored on Ω_l^n is the sum

$$\hat{u}_l^n(\xi,\eta) = \hat{v}_l^n(\xi,\eta) + \overline{\hat{u}}_l^n(\xi,\eta).$$

Therefore, instead of an equation of the type (4.3), the *FAS equations*, actually employed in the calculations, are

$$\hat{L}_l^n \hat{u}_l^n = \begin{cases} \hat{r}_l^n + \hat{L}_l^n \overline{\hat{u}}_l^n & \text{wherever} \quad \hat{r}_l^n \quad \text{can be defined} \\ \hat{L}_l^n \overline{\hat{u}}_l^n & \text{elsewhere,} \end{cases} \tag{5.1}$$

where $\hat{L}_l^n$ coincides with the $\hat{L}$ of (4.3) upon taking ξ and η to be the Cartesian coordinates of Ω_l^n; i.e., ξ is the propagation direction of e_l^n. As described above $\hat{r}_l^n$ is defined, by a local separation procedure, from the next-finer-level residuals. Note that due to using FAS, an equation for $\hat{u}_l^n$ is well defined also in regions of the coarse grid not covered by the fine grid (that is, where $\hat{r}_l^n$ cannot be defined from the fine-grid residuals).

Having calculated a solution $\hat{u}_l^n$ to Eq. (5.1) (see the cycle description below), it is of course the *difference* $\hat{u}_l^n - \overline{\hat{u}}_l^n$ which actually represents the correction $\hat{v}_l^n$ and which is therefore interpolated to the next finer level and *added to* the appropriate ray function(s) on level $(n-1)$; except that *at boundaries* the full values of $\hat{u}_l^n$ are directly interpolated and replace the boundary values of the finer ray function(s). See algorithmic details in Sec. 7 below.

On fine enough wave grids ($m \geq M_r$), too, the solution near the boundary is interpolated directly from increasingly coarser grids, eventually from the ray grids. Indeed, near a boundary distant from Ω_f the wave solution behaves like a combination of principal components which are smooth on the scales of the fine wave levels.

On each of the grids Ω_l^N at the lowest ray level, the RBC are imposed: Each entering ray is represented on the grid (or divided between the two grids) with the closest propagation direction, on which its boundary values are indeed very smooth. Note that the components represented on Ω_l^N may actually have propagation direction deviating by up to $\theta = O(\pi/L^N)$ from the corresponding ξ direction (the propagation direction of e_l^N). Hence, if

$$\Omega_l^N = \{(\xi,\eta) : |\xi| \leq d_1^N, |\eta| \leq d_2^N\}$$

and

$$\Omega_f \subset \{(\xi,\eta) : |\xi| \leq d_1^N, |\eta| \leq d_f\},$$

for some d_f and for d_1^N, d_2^N, defined as in Sec. 3., and if $d_1^N \geq d_f + d_2^N \tan\theta$, then all the *exiting-only* rays (rays which do not enter the domain but are created in Ω_f) can actually exit only through the "exit boundary" $\{(\xi,\eta) : |\xi| \leq d_1, \eta = d_2\}$. Hence on all other boundaries of Ω_l^N we can impose as boundary conditions the incoming rays (or zero, if no incoming rays are assumed). No boundary conditions are needed at the exit boundary of Ω_l^N, since the discrete equations (4.4) and the order in which we relax them (from entrance to exit) ensure that information propagation in the negative ξ direction is effectively prohibited.

6. Phase Errors.

In the previous sections we mentioned that the finest wave mesh-size should satisfy a certain condition so that the discrete solution that can be produced by the solver is an accurate approximation to the differential solution.
The relative error for the wave discretization (2.2) for a component $e^{ik_1x+k_2y}$ with $k_1^2 + k_2^2 = k^2$ is given by

$$E^h(k_1,k_2) \approx \frac{k^4h^2/\gamma}{2\pi k/d},$$

where $24 \leq \gamma \leq 48$. Therefore, in order to have relative errors smaller than ϵ throughout the computational domain, it is required that

$$dk^3h^2/(2\pi\gamma) < \epsilon. \tag{6.1}$$

For fast convergence, the discrete solution in the ray representation needs to have a small phase error as well, so that it can efficiently approximate the characteristic error left by the wave grids.

In this context the choice of the lattice frequencies proves to be an important issue. Obviously, the ray operators provide much better approximation to those characteristic components which are close to the lattice components, and they are much less accurate for those farther away. If the same principal frequencies are always picked as the lattice ones, then the same frequencies always have the worst approximation. An easy way to improve the situation is thus to vary the set of lattice points. One possibility is to use two sets: regular

$$\mathbf{k}_l^{n,r} = k\Big(\cos((l-1)\theta, \sin((l-1)\theta)\Big), \tag{6.2}$$

and staggered

$$\mathbf{k}_l^{n,s} = k\Big(\cos((l-1/2)\theta), \sin((l-1/2)\theta)\Big), \tag{6.3}$$

where in each case $l = 1, \dots L^n$ and $\theta = 2\pi/L^n$.
The relative error for the ray discretization (4.4) for a component $e^{i(\theta_\xi \xi + \theta_\eta \eta)}$ is thus given by

$$\hat{E}^{\mathbf{h}}(\theta_\xi, \theta_\eta) \approx \frac{\frac{h_\eta^2}{12}\theta_\eta^4 + \frac{h_\xi^2}{24}\theta_\eta^3 k + \frac{h_\xi^2}{8}\theta_\xi^2\theta_\eta^2}{2\pi k/d},$$

with the choice of the ray parameters given in Sec. 8, where d is the size of the computational domain. Analysis of the ray phase error shows then that in order to have ray relative errors to be smaller than ϵ, the number L^N of lattice points on the coarsest ray grid N, should satisfy the following condition:

$$\frac{kd}{5(L^N)^2\beta^4} < \epsilon, \tag{6.4}$$

where β is equal to either 1 or 2 depending on whether only regular or both regular and staggered lattices are employed at that level.
A heuristic explanation of the choice of β follows: The maximal relative error appears when $(\theta_\xi + k, \theta_\eta)$ is a principal component which is maximally distant from the lattice points where the ray operator indeed provides an excellent approximation to the wave finest grid operator. The values of $(\theta_\xi, \theta_\eta)$ can be thus estimated as

$$\theta_\xi \approx k\theta^2/2, \qquad \theta_\eta \approx k\theta,$$

where $\theta = \pi/L^N$. If not one but two different sets of lattice points are employed, then the worst approximation that could be provided by using two ray cycles will show up not for the same “distant” component (since the most “distant” component of the first ray cycle becomes a lattice one of the next ray cycle) but rather for a “middle” component with $\theta = \pi/2L^N$ which has an “average” approximation in both types of the ray cycles.

If both (6.1) and (6.4) are satisfied with $R = 1/\epsilon$ then the ray approximation is good enough to provide an average convergence rate R per ray cycle for all characteristic components.

If, however, the finest wave mesh-size does not satisfy (6.1) with $\epsilon = 1/R$, in order to provide the fast convergence of the algorithm, the ray operator should be modified by adding some additional terms so that it approximates the finest discrete wave operator, rather than the differential one. This addition is shown to be important in various numerical experiments, although it has been omitted in those reported below, since the accuracy ϵ there is sufficiently small. To derive the correction terms, one has to calculate the First Differential Approximation (FDA) to the discrete operator (2.4) applied to the wave function in the ray form $v(x,y) = \hat{v}(x,y)e^{i(k_1x+k_2y)}$:

$$FDA(L^h v(x,y)) = e^{i(k_1x+k_2y)}((L^r + h^2L^c)\hat{v}),$$

with $L^r\hat{v} = \Delta\hat{v} + 2ik_1\hat{v}_x + 2ik_2\hat{v}_y$ (the ray differential operator (4.3) in the (x,y) coordinate system) and

$$L^c\hat{v} = \frac{(k_1^4+k_2^4)}{12}\hat{v} - \frac{i}{3}(k_1^3\hat{v}_x + k_2^3\hat{v}_y) - \frac{1}{2}(k_1^2\hat{v}_{xx} + k_2^2\hat{v}_{yy}) + \frac{i}{3}(k_1\hat{v}_{xxx} + k_2\hat{v}_{yyy}) + \frac{1}{12}(\hat{v}_{xxxx} + \hat{v}_{yyyy}).$$

Here h is the finest wave mesh-size, and (k_1,k_2) is the lattice frequency for which the ray equations are written. The modified ray operator in the (x,y) coordinates is then $L^r + h^2L^c$.

7. Wave Cycle.

The outer part of the algorithm is a regular FAS V *wave cycle*, hosting two *ray cycles* at a certain stage. To describe it, the wave grids (levels) are numbered $1, 2, \ldots, M$, where 1 is the coarsest grid, with uniform mesh-size h_1 (usually in the range $5k^{-1} \lesssim h_1 \lesssim 10k^{-1}$), and each subsequent grid has the uniform mesh-size $h_m = h_{m-1}/2$. Grid M is the target level, where the target equations (2.2) are given, with $h = h_M$. The grids are all aligned (coarser grid lines are obtained by taking every other line of the next finer level). The domain covered by grid Ω_m is $\{(x,y) : |x| \le a_m, |y| \le a_m\}$, where $a_M = d/2$, and $a_m = a_{m+1} + K_mh_m$, i.e., each coarser grid m is widened by K_m meshseizes in each direction. (In our model algorithm we put $K_m = 4$ if $m \ge M_r$, and $K_m = 0$ otherwise).

On each wave level m, FAS equations are given by (2.2), with $u = u^m$, $h = h_m$ and with a right-hand-side f^m given by

$$f^m_{i,j} = \begin{cases} f_{i,j} & \text{if} \quad m = M, \\ r^m_{i,j} + (L^m\overline{u}^m)_{i,j} & \text{if} \quad m < M, \quad \text{and} \quad r^m_{i,j} \quad \text{can be defined}, \\ (L^m\overline{u}^m)_{i,j} & \text{if} \quad m < M, \quad \text{and} \quad r^m_{i,j} \quad \text{cannot be defined}, \end{cases}$$

where $r^m = I^m_{m+1}(f^{m+1} - L^{m+1}\tilde{u}^{m+1})$, which can be defined only at points interior to Ω_{m+1}; I^m_{m+1} is an adjoint interpolation operator from the finer level $m+1$ to level m; $\tilde{u}^{m+1}$ is the latest solution approximation on level $m+1$, L^m and L^{m+1} are the wave operators of type (2.2) on levels m and $m+1$, respectively, and $\overline{u}^m$ is some fixed approximation to the solution on level m, whose choice has been discussed in Sec. 5.

The V cycle has two subsequent parts, or “legs”. In the first leg the algorithm proceeds sequentially from the finest wave level $(m = M)$ to the coarsest one $(m = 1)$. On each level m, several relaxation sweeps are performed, then (if $m > 1$) the residuals and the approximate solution are transferred to the next coarser grid to define its right-hand side (f^{m-1}).

In the second leg of the V cycle the algorithm proceeds sequentially from the coarsest level $(m = 1)$ to the finest $(m = M)$. On each level m several relaxation sweeps are performed, then in the particular case that $m = M_0$ *two ray cycles* (see Sec. 8) are performed, and then the correction to the solution is interpolated to the interior points of the next finer grid, i.e.,

$$u^{m+1} = \tilde{u}^{m+1} + I^{m+1}_m(u^m - \overline{u}^m). \tag{7.1}$$

Here I_m^{m+1} is an interpolation operator and $\tilde{u}^{m+1}$ is the former latest approximation to the solution on level $m+1$ (formed just before the visit to level m and used in calculating r^m, as part of defining f^m). On the fine levels $(m \geq M_r)$ the values *on the boundary* are interpolated directly as

$$u^{m+1} = I_m^{m+1}(u^m). \tag{7.2}$$

On coarse wave grids ($m < M_r$), the boundary conditions are injected from the finer grids in the fine-to-coarse leg of the wave cycle, and no boundary corrections are introduced by these levels in the coarse-to-fine leg of the cycle (i.e., $u^{m+1} = \tilde{u}^{m+1}$ on the boundary).

Let us carefully follow the behavior of different error components through this multigrid cycle on the wave grids (excluding the ray cycles). For the majority of components, on the grids with $kh \lesssim 1$, the relaxation properties for (2.2) are similar to those for definite elliptic operators, which means an efficient reduction of high-frequency components. The smoothing factors (i.e., the convergence factors per sweep for the high-frequency components on each scale) are presented in Table 1. One can see that on the levels with $kh \lesssim 1$ and $kh \gtrsim 4$, the smoothing factors of the Gauss-Seidel relaxation are quite high, and only a few relaxation sweeps on each level suffice to reduce high-frequency components by an order of magnitude. However, the fast convergence is not always desirable, since it might be damaging when obtained for *erroneous components*, i.e., those which have large relative errors on the relaxed grids.

kh	0.125	0.5	1.0	1.25	1.5	2.0	3.0	4.0	6.0	8.0
μ_1	0.50	0.52	0.65	0.80	div	div	0.66	0.20	0.07	0.04
μ_2	0.80	0.82	0.92	0.98	1.02	1.04	0.93	0.40	0.13	0.07

Table 1. Here μ_1 and μ_2 are the smoothing factors for Gauss-Seidel and Kaczmarz relaxations, respectively, in lexicographic ordering on different grids. The precise meaning of the "divergence" that appears for $kh \approx 2$ is given in Table 3.

A Fourier component $e^{i(\omega_1 x + \omega_2 y)}$ is erroneously approximated on the grid h if its relative error $E^h(\omega_1, \omega_2)$ satisfies

$$E^h(\omega_1, \omega_2) > 1. \tag{7.3}$$

For equations (2.2) erroneous components happen to be close to the principal circle, i.e., their frequencies satisfy

$$\sqrt{{\omega_1}^2 + \omega_2^2} = k(1+\delta), \quad |\delta| \ll 1.$$

The symbol of (2.1) for $e^{i(\omega_1 x + \omega_2 y)}$ is given by $-\omega^2 + k^2$, while the symbol of the difference operator (2.2) is $k^2 + 2h^{-2}(\cos(\omega_1 h) + \cos(\omega_2 h) - 2)$. Hence, the relative error for such a component is

$$E^h(\omega_1, \omega_2) = \left| \frac{2h^{-2}(\cos(\omega_1 h) + \cos(\omega_2 h) - 2) + \omega^2}{-\omega^2 + k^2} \right|.$$

For any frequency component (ω_1, ω_2) which satisfies (7.3) holds

$$k(1-\delta_-) \leq \sqrt{{\omega_1}^2 + \omega_2^2} \leq k(1+\delta_+) \tag{7.4}$$

with δ_- and δ_+ presented in Table 2.

kh	0.25	0.5	1.0	1.25	1.5	2.0	3.0	4.0
δ_+	.003	.011	.05	.08	.13	any	any	any
δ_-	.003	.01	.04	.06	.07	.11	.16	.21

Table 2. Limits of poorly approximated components. On the grids with $kh \geq 5$ all components have a small relative error – characteristic components are too oscillatory to be visible there. Relaxation on such grids is not damaging for those components. By "any", we mean that *all* components with $\sqrt{{\omega_1}^2 + \omega_2^2} > k$ which are visible on the grid have a bad approximation there.

In Table 3 the amplification factors of Gauss-Seidel and Kaczmarz relaxation for components in the range (7.4) are shown. It is clear from the Table that on grids with $kh \leq 0.5$, the influence of the Gauss-Seidel relaxation on the erroneous components is negligible, and therefore it can be applied without significant damage; by Table 1, 3–4 sweeps would suffice to reduce high-frequency components on these grids.

kh	0.25	0.5	1.0	1.25	1.5	2.0	3.0	4.0
μ_0	1.04	1.14	2.00	div	div	div	1.00	1.00
μ_1	0.99	0.99	0.96	0.90	0.71	div	0.00	0.00
μ_2	1.00	0.99	0.99	0.97	0.90	0.20	0.05	0.01

Table 3. Here μ_0 is the maximal divergence factor over all components per Gauss-Seidel relaxation sweep; μ_1 are μ_2 are the strongest (i.e., the smallest) convergence factors in the range (7.4) for a Gauss-Seidel and a Kaczmarz relaxation sweep, respectively. For $kh \approx 2$, the Gauss-Seidel relaxation has a bad divergence, since the "diagonal" coefficient of (2.2) becomes very small, compared to the other coefficients.

Actually, on a sufficiently fine grids (with $kh \lesssim 0.125$), the fast red-black Gauss-Seidel, with the smoothing factor $.25$, can be applied. However, its use on the coarser levels is not advisable, since its divergence qualities are even stronger than the ones of the lexicographic Gauss-Seidel (μ_0 in Table 2).

When h is close to $1/k$, the Gauss-Seidel divergence becomes too strong for some smooth components. Hence, we choose to use the slower but always converging Kaczmarz relaxation on such grids. Relaxation on all levels with $h \lesssim 1/k$ would then provide an efficient reduction of all error components $e^{i(\omega_1 x + \omega_2 y)}$ in the range

(7.5) $$\sqrt{{\omega_1}^2 + \omega_2^2} \geq \frac{\pi k}{\alpha_\star}, \qquad 1 \lesssim \alpha_\star \lesssim 2.$$

On levels with $1/k < h < 5/k$, both Gauss-Seidel and Kaczmarz relaxation schemes strongly change components (7.4), introducing their erroneous approximations. Therefore, any relaxation on these grids should be avoided.

On a level with mesh-size $h \geq 5/k$, *no* characteristic components are visible, and the convergence factor for the Gauss-Seidel relaxation there is very good (small) for *all* error components which are represented there, i.e., the ones with

(7.6) $$\sqrt{{\omega_1}^2 + \omega_2^2} \leq \frac{\pi k}{\beta_\star}, \qquad 5 \lesssim \beta_\star \lesssim 10.$$

Indeed, such a level can be chosen as the coarsest wave level, since one Gauss-Seidel relaxation sweep on it sufficiently converges *all* visible components, including those smooth components which are also visible on much coarser levels.

In the next Table we specify the number and the type of relaxation sweeps actually applied on the wave levels in the experiments reported below. As one can see, many Kaczmarz sweeps are applied on the level with $kh = 1$ in order to reduce the width of the characteristic ring. An alternative here is to employ one additional *ray* level, finer than the finest used by our current algorithm, and, by this, approximate a wider range of characteristic components. We found it simpler to use the wave approach. The number of sweeps is bounded, i.e., it does not depend on the size of the domain. Also, this level is much coarser than the finest wave grid, so the cost of even 30 sweeps is small.

kh	$1/16$	$1/8$	$1/4$	$1/2$	1	2	4	8
N	2	2	4	4	30	0	0	2
Type	G-S(r-b)	G-S(r-b)	G-S(l)	G-S(l)	K(l)	none	none	G-S(l)

Table 4. Here h is the wave mesh-size, N is the number of relaxation sweeps applied on grid h, and "Type" stands for the type of relaxation: "G-S" meaning Gauss-Seidel and "K" – Kaczmarz relaxations; "l" and "r-b" in parenthesis stands for the relaxation ordering: lexicographic or red-black, respectively.

In the numerical experiments reported below we have used *cubic* interpolation operators. Applying instead linear interpolation leads to a mild drop in the convergence rates, but still provides a good multigrid efficiency. However, it is advisable to use the higher-order (cubic) interpolation at least on relatively coarse grids where it is inexpensive.

From (7.5) and (7.6) we see that the characteristic components $e^{i(\omega_1 x+\omega_2 y)}$ which cannot efficiently be reduced on the wave grids lie roughly in the range

$$\frac{\pi k}{\beta_\star} \le \sqrt{\omega_1^2+\omega_2^2} \le \frac{\pi k}{\alpha_\star}. \tag{7.7}$$

These are the components that should be reduced by the ray cycles, discussed next.

8. Ray Cycles.

The residuals for the ray cycles are calculated not directly on the "wave-to-ray switching" level M_r, ($k\mathbf{h}_{M_r} \approx (1,1)$) but on a sufficiently fine wave grid (level M_0 with $k\mathbf{h}_{M_0} \ll (1,1)$), so that they still yield accurate approximation to the target (finest) wave equation. That level (M_0) should be fine enough to yield with the discretization (2.2) a good point-wise approximation to the characteristic error components. In practice the residuals of that wave level are transferred, when they are needed, through intermediate levels (by applying full-weighting operators) to the wave level M_r, where the ray separation starts (see Sec. 4).

The ray cycle can be regarded as a modified V cycle. As a usual V cycle it consists of two legs and employs several grids (levels) $1, 2, \dots N$, where N is the coarsest and 1 is the finest level. On the finest ray level L^1 ray problems are represented: Each ray problem l is defined in the appropriate rotated coordinates (ξ, η) (ξ being in propagation direction of e_l^1) on a grid with mesh-size $\mathbf{h}_1 = (h_\xi^1, h_\eta^1)$, with $2.5k^{-1} \le h_\xi^1 \le 5k^{-1}$ and $1.25k^{-1} \le h_\eta^1 \le 2.5k^{-1}$.

There is *no* need to choose the finest ray grid to be finer than that, since on this scale all characteristic components which concerns in the ray cycles still have an accurate resolution.

The coarsening of the ray grids is performed according to the smoothness of the ray functions, needed to be represent on these grids. Each coarser grid n then has mesh-sizes $h_\xi^n = 2h_\xi^{n-1}$ and

$$h_\eta^n = \begin{cases} 2h_\eta^{n-1} & \text{if} \quad n \quad \text{is even,} \\ h_\eta^{n-1} & \text{if} \quad n \quad \text{is odd,} \end{cases}$$

and the number of ray problems represented there is equal to

$$L^n = \begin{cases} 2L^{n-1} & \text{if} \quad n \quad \text{is even,} \\ L^{n-1} & \text{if} \quad n \quad \text{is odd.} \end{cases}$$

In the first leg of the ray cycle (i.e., when the algorithm proceeds sequentially from finer to coarser grids) several relaxation sweeps (two Kaczmarz sweeps in our model algorithm) are done on each ray level, except for the coarsest one.

Following relaxation, a switch is made to a coarser level, and the following transfers are made. If n is even, meaning that the number of ray functions remains the same on the coarser grid, the residual transfer is done as in a regular V-cycle with coarsening only in the propagation direction (no separation is applied). Otherwise, the next coarser level employs twice as many ray functions; half of them correspond to the same principal lattice components that already appeared on the coarser lattice (corresponding to the *finer* ray level), and another half are represented only on the finer lattice (*coarser* ray level). If $n-1$ and n are the finer and the coarser levels, correspondingly, then the coarse-grid residuals $\hat{r}^n$ are evaluated by the following two formulae:

$$\hat{r}^n_{2l+1} = W^n_{n-1} R^{n-1}_l, \tag{8.1}$$

$$\hat{r}^n_{2l} = W^n_{n-1}[e^{n-1}_l I^{2l}_{2l-1}(R^{n-1}_l) + e^{n-1}_{l+1} I^{2l}_{2l+1}(R^{n-1}_{l+1})]/e^n_{2l}, \tag{8.2}$$

where W^n_{n-1} is a successive combination of simple weighting separation operators, constructed as described in Sec. 4. and in the Appendix A, that acts from the grid on level $n-1$ to the grid on level n; R^{n-1}_j are the ray residuals, calculated on the ray level $n-1$; and $I^{2l}_{2l\pm1}$ are interpolation operators (cubic, in our experiments) which transfer the residuals from the grid in coordinates $(\xi^n_{l\pm1}, \eta^n_{l\pm1})$ to the grid with the same mesh-sizes $(h^{n-1}_\xi, h^{n-1}_\eta)$ but in the coordinates which coincide with $(\xi^n_{2l}, \eta^n_{2l})$. Finally, the FAS right-hand-side is calculated as in (5.1).

On the coarsest ray level *all* ray problems represented there are solved almost directly, by one sweep of line relaxation (see Sec. 4) which includes imposing the RBC as described in Sec. 5.

In the second leg of the ray cycle (which means proceeding from coarser to finer levels), the correction $\hat{v}^n_l$,

$$\hat{v}^n_l = (\hat{u}^n_l - \overline{\hat{u}}^n_l), \tag{8.3}$$

calculated on a coarse ray grid n ($\hat{u}^n_l$ being the current approximation and $\overline{\hat{u}}^n_l$ – the initial one), is interpolated to the next finer grid to improve the interior values of one or two finer ray functions. Namely, if n is odd

$$\hat{u}^{n-1}_l = \tilde{\hat{u}}^{n-1}_l + I^{n-1}_n \hat{v}^n_l, \tag{8.4}$$

and, if n is even

$$\hat{u}^{n-1}_l = \tilde{\hat{u}}^{n-1}_l + I^{n-1}_n \hat{v}^n_{2l} + [e^n_{2l-1} I^{n-1}_n(\hat{v}^n_{2l-1}) + e^n_{2l+1} I^{n-1}_n(\hat{v}^n_{2l+1})]/(2e^{n-1}_l) \tag{8.5}$$

where $2l \pm 1$ is taken modulo L^n; I^{n-1}_n is an interpolation operator to Ω^{n-1}_l from Ω^n_{2l} or from Ω^n_{2l-1} or from Ω^n_{2l+1} as needed; and $\tilde{\hat{u}}^{n-1}_l$ is the former approximate solution on Ω^{n-1}_l (the latest approximation there, from which the residuals R^{n-1}_l were calculated in forming (8.1)–(8.2)). On the boundaries of the fine domains not the correction (8.3), but rather the coarse-grid solutions $\hat{u}^n_l$ themselves, using procedures similar to (8.4)–(8.5), directly replace the former boundary values.

For $n = 1$, the corrections to the ray functions serve to improve the interior values of the wave function u^m,

$$u^m = \tilde{u}^m + \sum_{l=1}^{L^n} I_{l,n}^m(\hat{v}_l^n) e_l^n, \tag{8.6}$$

while on the boundaries of Ω_m the values of u^m are given by

$$u^m = \sum_{l=1}^{L^n} I_{l,n}^m(\hat{u}_l^n) e_l^n, \tag{8.7}$$

where $m = M_r$ and $I_{l,n}^m$ is an interpolation (cubic, in our experiments) operator from the ray grid Ω_l^n (defined in the proper rotated coordinate system) to the wave grid Ω_m (defined in the (x, y) coordinate system) and $\tilde{u}^m$ is the approximation on the wave level M_r just before the ray cycle (i.e., the approximation from which the finest ray residuals $\hat{r}_l^1$ are calculated). Then the algorithm proceeds (with several relaxation sweeps on each wave level) to the wave level $m = M_0$ as described in (7.1)–(7.2).

No relaxation can be performed in the second leg of the ray cycle, since the ray coarse-grid correction is distributed between the fine-grid ray functions not according to the weights of residuals obtained from them, but with equal weights. Hence, after the coarse-grid correction the right-hand-side and the approximate solutions on the finer grids are not compatible.

Each wave cycle employs two ray cycles. The residuals for the first ray cycle are calculated when the second leg of the wave cycle reaches the wave level M_0; then these residuals are transferred to the level M_r, etc. After the first ray cycle has accomplished its work, the algorithm returns back to the wave representation, proceeding from the wave level M_r to the finer wave levels, as described in Sec. 7, and reaches the wave level M_0, where the wave residuals for the second ray cycle are calculated and transferred back to the level M_r.

In our algorithm, on the finest ray level (with $n = 1$ and $L^n = 8$) we use two sets of lattice points: Regular

$$\mathbf{k}_l^{1,r} = k\left(\cos((l-1)\frac{\pi}{4}), \sin((l-1)\frac{\pi}{4})\right),$$

and staggered

$$\mathbf{k}_l^{1,s} = k\left(\cos((l-1)\frac{\pi}{4} - \frac{\pi}{8}), \sin((l-1)\frac{\pi}{4} - \frac{\pi}{8})\right),$$

where in each case $l = 1, \dots 8$.

At lower ray levels ($n > 1$), if there are some, however, only regular sets (with $\theta_0 = 0$ in (2.3)) are used in both cycles.

Let us now estimate what are the values of kd for which the algorithm still can employ only *one* ray level. The values of L and R in (6.4) we are interested in are $L = 8$ and $R = \sqrt{16}$, meaning $R^2 = 16$ as a convergence factor for two ray cycles. (16 is the smoothing factor of two red-black Gauss-Seidel relaxation sweeps, performed on the finest wave level. We, indeed, would like to see the same factor for all error components, including the characteristic ones treated by the ray cycles.) The following estimates can be obtained:

$$kd \lesssim 1280, \tag{8.8}$$

if two types of the lattices (regular *and* staggered) are employed, and

$$kd \lesssim 80, \tag{8.9}$$

if only one lattice type is used in the algorithm.

9. Numerical Results. The model problem chosen for tests of the wave-ray multigrid algorithm is

$$\Delta u(x,y) + k^2 u(x,y) = f(x,y),\ (x,y) \in \Re^2, \tag{9.1}$$

where f is randomly defined if $\sqrt{x^2+y^2} < R_0$, or otherwise it is set to zero .

The computational work is measured in *work units* – the unit being the number of arithmetic operations required in the relaxation of the equations (2.2) on the finest wave grid. One cycle of the algorithm can be shown to cost about 7 work units (about 4 of them on the finest grid), invested in relaxation (2 sweeps on the finest level), residual calculation, separation, and interpolation.

We have tested the algorithm described above for different parameters of the model problem and of the algorithm itself.

Since we are interested in showing an efficient solver for the highly indefinite problems, in our experiments we always choose the wave number k and the computational domain diameter d so that

$$kd \gg 1. \tag{9.2}$$

The results of computations for different values of d with fixed $k = 1$ and $R_0 = 5$ are presented in Table 5. The algorithm employs one ray level with 8 lattice points, alternating using the regular lattice and the staggered one (see Sec. 8).

kd	**C2**	**C3**	**C4**	**C5**	**C6**	**C7**	**C8**	**C9**
10	1.8e-02	5.4e-04	2.0e-05	9.2e-07	4.8e-08	2.6e-09	1.5e-10	8.7e-12
20	7.0e-03	2.1e-04	8.7e-06	4.4e-07	2.4e-08	1.3e-09	7.5e-11	4.4e-12
40	3.7e-03	1.2e-04	4.7e-06	2.3e-07	1.2e-08	6.9e-10	4.0e-11	2.3e-12
80	2.1e-03	8.5e-05	3.7e-06	1.6e-07	7.6e-09	3.8e-10	2.1e-11	1.2e-12
160	1.2e-03	8.2e-05	5.5e-06	3.2e-07	1.7e-08	9.1e-10	5.0e-11	3.1e-12

Table 5. This Table shows the L_2 norm of the residual function, calculated on the finest wave grid before each cycle. The results are calculated for (9.1) with $k = 1$ and $R_0 = 5$ for different sizes of the computational domain d. $C\#$ stands for the number of the cycle before which the residual norm is calculated. The finest wave level ($M = 8$) has a mesh-size $h_8 = 0.0625$, the residuals for the ray cycle are calculated on level $M_0 = 6$, whose mesh-size is $h_6 = 0.25$; the separation process starts on level $M_r = 4$ with mesh-size $h_4 = 1.0$. The number and the type of relaxation sweeps on the wave levels is specified in Table 4.

Clearly, the algorithm exhibits excellent convergence even for large values of kd, and this is indeed achieved by a relatively cheap procedure.

In Table 6 we present the results for the algorithm which employs one ray level and uses *only a regular* lattice for both ray cycles, and the results for the algorithm that employs *two* ray levels at each ray cycle (with two types of lattices being used on the first ray level, but not on the second) and compare them with the results from the previous Table.

	$kd = 10$	$kd = 20$	$kd = 40$	$kd = 80$	$kd = 160$
R_m	18.0	17.8	17.7	19.3	18.0
R_r	17.7	11.9	6.7	5.7	3.1
R_t	17.8	18.6	13.3	11.5	11.0

Table 6. R_m is the asymptotic convergence factor per cycle for the L_2 norm of the residual function for (9.1)performed by the algorithms that employs one ray level and uses both regular and staggered lattice ($L = 8$ and $\beta = 2$ in (6.4)); R_r is the asymptotic convergence factor performed by the algorithms that

ETNA
Kent State University
etna@mcs.kent.edu

employs only a regular lattice ($L = 8$ and $\beta = 1$); R_t is the asymptotic convergence factor performed by the "two-ray levels" algorithm ($L = 16$ and $\beta = 1$).

The algorithm which uses only one lattice type (R_r) slows down when the value of kd is roughly close to the estimate (8.8), and the slowdown in R_t can similarly be explained by (6.4). We expect R_m to slow down when kd exceeds the estimate (8.9).

In the previous Tables we presented the results for a fixed value of k. Table 7 shows how changes in k influence the performance when for all values of k we run the wave-ray cycle with the same parameters as chosen above for $k = 1$.

	$k = 0.66$	$k = 0.7$	$k = 0.8$	$k = 1.0$	$k = 1.2$	$k = 1.3$	$k = 1.33$
R	17.5	18.7	18.7	17.7	18.9	14.0	11.9

Table 7. Here R is the asymptotic convergence factor per cycle for the L_2 norm of the residual function for (9.1) performed by the algorithms that employs one ray level and uses both regular and staggered lattice for different values of the wave number k. Parameters of the algorithm are as in Table 5, $d = 40$.

We see that the algorithm shows a good convergence anywhere in the range $.66 \lesssim k \lesssim 1.33$. The convergence factor decreases for k outside this range. This can easily be corrected just by choosing different algorithmic parameters, so that they again satisfy in terms of kh the relations which hold in the above model algorithm for $k \in [.66, 1.33]$. For example, for $k \in [1.33, 2.66]$ all the wave-to-ray transfer mesh-sizes (those of M_0, M_r and Ω_n) should be chosen twice finer than for $k \in [0.66, 1.33]$, and for $k \in [.33, .66]$ – twice coarser, etc. We emphasize that with a proper scaling this algorithm works equally well for *any* value of k.

10. Remarks and Future Work. In the work presented here, we have developed a multigrid wave-ray algorithm for solving two-dimensional standing wave equations. Our solvers exhibit high efficiency, permit a natural introduction of radiation boundary conditions, eliminate constraints on the size of the domains considered and the mesh-size of the grids, and use fast relaxation schemes on the finest, i.e., the most expensive, levels.

The present article has focused on the fast convergence of the multigrid cycles, not on the *accuracy* of the obtained solution. A series of experiments has shown that $O(h^2)$ accuracy is indeed achieved, provided each of the coarse-level (wave and ray) domains is large enough – its diameter increasing as the finest mesh-size h decreases. The rate of the needed increase is mild, however, keeping the total number of grid points on all levels still $O(h^{-2})$. Detailed derivation of the required domain sizes, together with numerical results of accuracy and of corresponding FMG algorithms, will be given elsewhere.

The method developed here needs to be and can be extended in a number of ways. The extension to *three dimensions* is relatively straightforward, although the circular lattices of the present work will have to be replaced by spherical ones, requiring a tessellation of the unit sphere.

Also relatively simple is the introduction of boundary conditions for the wave equations, such as Dirichlet or Neumann or mixed, along some given curves (or surfaces), in addition to the exterior RBC. This will imply certain relations between the various ray functions of a ray level along such boundaries, corresponding to the reflection relations in geometrical optics.

The extension to the case of *variable coefficient* $k = k(x, y)$ is more complicated. It has been studied in [6] for the one-dimensional case, but the higher dimensional algorithms are substantially different, as amply shown by the present article. In case $k(x, y)$ varies *slowly* (meaning that its changes over a distance of one wavelength are small compared to itself), there should be no difficulty in applying the separation and other local procedures. On each ray level of the algorithm, each of the L grids will correspond not to a fixed lattice point, but to a fixed solution of the eikonal equation, the grid direction ξ following the continuously carrying propagation direction. In case of *abrupt* changes in $k(x, y)$, relations between different rays, corresponding to reflection and refraction will necessarily enter.

A different case is that of a *small-scale disturbance*, i.e., a small region, comparable in diameter to the wavelength, over which either $k(x, y)$ or the direction of the normal to the boundary or to a curve of discontinuity in k significantly changes. Such a region, together with a several-wavelength neighborhood around it, should be well resolved by a grid, on which the wave equation is discretized. Due to its small size, the fast multigrid solver does not need on that region by itself a ray-level acceleration, hence the precise ray discretization there is immaterial. In the neighborhood, however, the ray discretization is crucial: it would supply on one hand the rays entering into the region, and on the other obtain from it the exiting rays. This can be achieved provided the FAS cycle includes, at least on exit regions, a separation process not only for the residuals but also for the solution, i.e., the $\overline{\hat{u}}_l^n$ are built there not as in Sec. 6 above, but by separating $\hat{u}_l^{n+1}$ (or by separating the wave solution u^{M_r}, in case $n = 1$). Thus, in such a situation, these are the wave levels that would mediate the relation between incoming and outgoing rays.

This procedure can be regarded as a numerical extension of the WKB method, providing a general tool for calculating diffraction effects only around arbitrary obstacles and other disturbances. Typically, most of the problem domain will be treated by *geometrical optics*, i.e., very coarse *ray* levels, with finer discretization reaching into wave levels being introduced around regions of small-scale disturbances, where geometrical optics by itself would break down.

11. Appendix A.

Here we present a more detailed description of the separation process used in our model algorithm to approximate the ray residual functions $\hat{r}_l^1$, $l = 1, \dots, 8$, on the finest ray level 1 with mesh-size $(4/k_0, 2/k_0)$, where $k_0 = 2^j$, and $j \in Z$ is chosen so that $\overline{k} = k/k_0 \in [0.66, 1.33]$.

The separation starts with the wave function r, defined on the wave level M_r with mesh-size $(1/k_0, 1/k_0)$ as follows

$$r = I_{M_r+1}^{M_r}[\dots[I_{M_0}^{M_0-1}R^{M_0}]\dots],$$

where I_j^{j-1}, $j = M_0, \dots, M_r + 1$ are full-weighting operators; $R^{M_0} = f^{M_0} - L^{M_0}u^{M_0}$ is the wave residual function; u^{M_0} is the current solution approximation; f^{M_0} is the FAS right-hand-side and L^{M_0} is the operator (2.2) on the wave level $m = M_0$. This function is than interpolated to the rotated coordinates (ξ, η) and multiplied by $e^{-ik\xi}$, giving as a result the function $r_0(\xi, \eta)$.

The first separation operator W_0 is applied to $r_0(\xi, \eta)$. W_0 is a tensor product of two perpendicular "diagonal" one-dimensional weighting operators with the frequency parameter (see Sec. 4) taken equal to 2. The resulting function $r_1(\xi, \eta)$, defined on the grid with mesh-size $(h_\xi, h_\eta) = (2/k_0, 2/k_0)$, is given by

$$r_1(\xi, \eta) = \min(1, \overline{k}) \times [W_0(r_0(\xi, \eta)].$$

The next separation operator W_1, applied to $r_1(\xi, \eta)$ is a tensor product of a weighting operator in the ξ direction defined by the frequency parameter $\max(1, \overline{k})$, and a weighting operator in the η direction defined by the frequency parameter $.85 \max(1, \overline{k})$.

The resulting function $r_2(\xi, \eta)$ is defined on the grid with mesh-size $(4/k_0, 2/k_0)$ and is given by

$$r_2(\xi, \eta) = W_1(r_1(\xi, \eta)).$$

Finally, a weighting operator in the η direction defined by the frequency parameter $.75 \max(1, \overline{k}^2)$ is applied to r_2, yielding the target function $\hat{r}(\xi, \eta)$ on the same grid with $(h_\xi, h_\eta) = (4/k_0, 2/k_0)$.

12. Appendix B.

The numerical results shown in Table 5 where obtained by applying the algorithm briefly described below in the form of a flowchart. We hope that this flowchart will help the reader to go through some computational details. The algorithm employs eight wave and one ray (with eight ray functions) levels.

```
Wave–Ray Cycle
    First Leg of Wave Cycle: from finest level $M = 8$ ($h_8 = 0.0625$)
    to coarsest level 1 ($h_1 = 8$)
    Second Leg of Wave Cycle: from level 1 to level $M_0 = 6$ ($h_6 = 0.25$)
    (No changes of boundary values)
    On level $m = M_0$ calculate wave residuals $r^m = f^m - L^m u^m$
    Transfer $r^m$ by full-weighting procedure to level $M_r = 4$ ($h_4 = 1.0$):
        for ($m = M_0 - 1; m \geq M_r; m = m - 1$) $r^m$ = FullWeighting($r^{m+1}$)
    Employ Ray Cycle with Regular Lattice Set with $r^{M_r}$ as input
    and corrected solution approximation $u^{M_r}$ as output.
    Second Leg of Wave Cycle: from level $M_r$ to level $M_0$
    (On boundaries: $u^m = I^m_{m-1} u^{m-1}, m = M_0 + 1, \ldots M_r$)
    On level $m = M_0$ calculate wave residuals $r^m = f^m - L^m$
    Transfer $r^m$ by full-weighting procedure to level $M_r = 4$
        for ($m = M_0 - 1; m \geq M_r; m = m - 1$) $r^m$ = FullWeighting($r^{m+1}$)
    Employ Ray Cycle with Staggered Lattice Set with $r^{M_r}$ as input
    and corrected solution approximation $u^{M_r}$ as output.
    Second Leg of Wave Cycle: from level $M_r$ to level $M$
    (On boundaries: $u^m = I^m_{m-1} u^{m-1}, m = M_r + 1, \ldots M$)
End of Wave Ray Cycle
Ray Cycle
    for ($l = 1; l \leq 8; l = l + 1$) do
    Interpolate input residual $r$ defined in (x,y) coordinates on grid
    with mesh $(1, 1)$ to grid in $(\xi_l, \eta_l)$ coordinates with mesh $(1, 1)$ and
    "divide" it by l-th lattice Fourier component $e_l$: $r_0 = (I^{(\xi_l,\eta_l)}_{(x,y)} r)/e_l$
    Residual separation:
        From grid with mesh $(1, 1)$ to grid with mesh (2,2) $r1 = W_0(r_0)$
        From grid with mesh $(2, 2)$ to grid with mesh (4,2) $r_2 = W_1(r_1)$
        From grid with mesh $(4, 2)$ to grid with mesh (4,2) $\hat{r}_l = W_2(r2)$
    On level with mesh-size $(4, 2)$:
        Use previous values of $\hat{u}_l$ as initial approximation $\overline{\hat{u}}_l$
        Calculate FAS right-hand-side $\hat{f}_l = \hat{r}_l + \hat{L}\overline{\hat{u}}_l$
        Introduce RBC and make one line relaxation, resulting $\hat{u}_l$
        Calculate correction $\hat{u}_l - \overline{\hat{u}}_l$ and interpolate it to grid in $(x, y)$ coor-
        dinates with mesh $(1, 1)$, resulting $\delta_l$
        Correct interior values of wave approximate solution $u$: $u = u + e_l \delta_l$
    end for
    (Boundary values of $u$: $u = \sum \hat{u}_l e^l$)
End Ray Cycle
```

REFERENCES

[1] A. BRANDT, *Multi-level adaptive solutions to boundary value problems*, Math. Comp., 31(1977), pp. 333–390.

[2] ———, *Stages in Developing Multigrid Solutions*, in Proc. 2nd Int. Congr. on Numerical methods for Engineers, Dunod, Paris, E. Absi, R. Glowinsky, P. Lascaux, and M. Veysserge, eds., 1980, pp.23–43.

[3] ———, *Guide to multigrid development*, in Multigrid Methods, Lecture Notes in Math, 960, W. Hackbusch W. and U. Trottenberg, eds., Springer-Verlag, 1982, pp. 220–312.

[4] ———, *Multigrid Techniques: 1984 Guide with applications to fluid dynamics*, Monograph, GMD-Studie 85, GMD-FIT, Postfach 1240, D-5205, St. Augustin 1, W. Germany, 1985 (unpublished). (Also available from Secretary, Department of Mathematics, University of Colorado at Denver, Colorado 80204-5300.)

[5] A. BRANDT AND S.TA'ASAN, *Multigrid Methods for Nearly Singular and Slightly Indefinite Problems*, In Multigrid Methods II, Proceedings, Cologne, 1985, Lecture Notes in Mathematics 1228, W. Hackbusch and U. Trottenberg, eds, Springer-Verlag, pp. 100-122.

[6] I. LIVSHITS, *Multigrid Solvers for Wave Equations*, Ph. D. Thesis, Bar-Ilan University, Ramat-Gan, Israel, 1996.

[7] S. TA'ASAN, *Multigrid Methods for Highly Oscillatory Problems*, Ph. D. Thesis, The Weizmann Institute of Science, Rehovot, Israel, 1984.