

## A HIERARCHICAL 3-D DIRECT HELMHOLTZ SOLVER BY DOMAIN DECOMPOSITION AND MODIFIED FOURIER METHOD\*

E. BRAVERMAN<sup>†</sup>, M. ISRAELI<sup>‡</sup>, AND A. AVERBUCH<sup>§</sup>

**Abstract.** The paper contains a noniterative solver for the Helmholtz and the modified Helmholtz equations in a hexahedron. The solver is based on domain decomposition. The solution domain is divided into mostly parallelepiped subdomains. In each subdomain a particular solution of the nonhomogeneous Helmholtz equation is first computed by a fast spectral 3-D method which was developed in our earlier papers (see, for example, *SIAM J. Sci. Comput.*, 20 (1999), pp. 2237–2260). This method is based on the application of the discrete Fourier transform accompanied by a subtraction technique. For high accuracy the subdomain boundary conditions must be compatible with the specified inhomogeneous right-hand side at the edges of all the interfaces. In the following steps the partial solutions are hierarchically matched. At each step pairs of adjacent subdomains are merged into larger units. The paper describes in detail the matching algorithm for two boxes which is a basis for the domain decomposition scheme. The hierarchical approach is convenient for parallelization and can minimize the global communication. The algorithm requires  $O(N^3 \log N)$  operations, where  $N$  is the number of grid points in each direction.

**Key words.** fast three-dimensional solver, Helmholtz equation, Fourier method, domain decomposition

**AMS subject classifications.** 65N35, 65T50, 65N55

**DOI.** 10.1137/S1064827502417039

**1. Introduction.** Elliptic equations arise in the determination of the pressure field for incompressible Navier–Stokes equations, in the implicit solution of viscous and heat transfer problems, in the solution of the Maxwell equations for lithographic exposure, in the solution of reaction-diffusion equations for baking and dissolution processes in semiconductor manufacture, and in many other applications. Fast and accurate solution of such equations is an important step towards resolution of problems which appear in computational physics or fluid dynamics. For example, the semi-implicit discretization in time (see [12]) of the incompressible three-dimensional (3-D) Navier–Stokes equations gives rise to one Poisson equation for the pressure and three modified Helmholtz equations for the momentum equations.

Application of high-order spectral and pseudospectral methods, which are based on global expansions into orthogonal polynomials (Chebyshev or Legendre polynomials), to the solution of elliptic equations results in full matrix problems. The cost of inverting full  $P \times P$  matrices without using special properties is  $O(P^3)$  operations.

We present a noniterative domain decomposition based algorithm for a high-order spectral solution of the 3-D Helmholtz equation. Most Poisson and Helmholtz solvers were initially developed for the two-dimensional (2-D) case, such as the fast multipole

---

\*Received by the editors November 26, 2002; accepted for publication (in revised form) July 28, 2004; published electronically April 19, 2005.

<http://www.siam.org/journals/sisc/26-5/41703.html>

<sup>†</sup>Department of Mathematics and Statistics, University of Calgary, 2500 University Drive NW, Calgary, Alberta T2N 1N4, Canada (maelena@math.ucalgary.ca). The research of this author was supported by University of Calgary Research grant F103998350.

<sup>‡</sup>Department of Computer Science, Technion-Israel Institute of Technology, Haifa 32000, Israel (israeli@cs.technion.ac.il). The research of this author was supported by the VPR fund for promotion of research at the Technion.

<sup>§</sup>School of Mathematical Sciences, Tel Aviv University, Tel Aviv 69978, Israel (amir@math.tau.ac.il).

method (FMM) in [10], domain decomposition and other preconditioning strategies, the boundary integral method in [14], and the modified Fourier method in [3, 4]. An adaptive algorithm for a fast solution of the 2-D Poisson equation by decomposition of the domain into square domains and the subsequent matching of these solutions by the FFM was developed in [9].

We solve the modified Helmholtz

$$(1.1) \quad \Delta u - \lambda^2 u = u_{xx} + u_{yy} + u_{zz} - \lambda^2 u = f(x, y, z)$$

or the Helmholtz equation

$$(1.2) \quad \Delta u + k^2 u = u_{xx} + u_{yy} + u_{zz} + k^2 u = f(x, y, z)$$

in a 3-D domain  $\Omega$  with Dirichlet

$$(1.3) \quad u = \Phi(x, y) \quad \text{on} \quad \partial\Omega$$

boundary conditions by domain decomposition (DD) methods.

The present 3-D algorithm is based on the fast spectral Helmholtz solver which was developed in [6]. It incorporates the application of the FFT with a preliminary subtraction technique. In this paper we generalize the algorithm of [2] to the 3-D case. The efficiency of the algorithm is especially vital for 3-D problems since such applications usually require heavy computations. The 3-D method enjoys the properties of the 2-D algorithm: fast convergence (i.e., small  $N$  is necessary to achieve the prescribed accuracy) and comparatively small number of operations per point ( $O(\log N)$ ).

The application of the Fourier method has the following advantages when solving the Helmholtz equation:

1. Differential operators are represented in the Fourier basis by diagonal matrices. This fact reduces the operator inversion to a simple division of the Fourier coefficients by the corresponding wave numbers. The cost of this step is  $O(N^3 \log N)$ , where  $N$  is the number of grid points in each of the three directions. ( $N$  is also the number of Fourier harmonics in the related series representation.)
2. If the function is infinitely differentiable and periodic, then a Fourier series approximation of  $f$  converges to  $f$  spectrally, i.e., more rapidly than any finite power of  $1/N$ .

Multidimensional Fourier representations can be considered for Cartesian geometries (rectangles in 2-D and parallelepipeds in 3-D). However, rapid convergence of the series representation requires that the periodic extension of the solution has a certain number of continuous derivatives. The periodic extension of a nonperiodic function is discontinuous and the corresponding Fourier series converges only as  $1/N$ , which is not better than a first-order finite-difference scheme. The slow convergence is caused by the so-called Gibbs phenomenon. Below we describe two steps of the algorithm developed in [6] for the 3-D Helmholtz equation and characterize the methods used to avoid the Gibbs phenomenon. The first step addresses the Gibbs phenomenon in the particular solution via extension; the second step uses subtraction to improve the accuracy of the inhomogeneous solver.

1. The function  $f$  in the right-hand side of (1.1) or (1.2) is extended to a larger domain and it is replaced by a new function which coincides with  $f$  in the original domain but the periodic extension of the larger domain has a certain

number of continuous derivatives [16, 4]. The extension procedure is based on the local Fourier basis method [11, 17], which employs folding functions as described and tested in [6].

2. An auxiliary boundary value problem for the Helmholtz equation is solved to satisfy the original boundary conditions. (In the present work we consider principally the Dirichlet boundary conditions.) To reduce the effect of the Gibbs phenomenon, the subtraction technique used in [6] for the 3-D Laplace equation is used. In particular, if after the subtraction procedure the function is periodic together with all its derivatives up to the third order, then the error converges as  $1/N^4$  [7]. This is exactly the convergence of the algorithm that was developed in [6].

Our matching procedures also extensively use the Fourier representations. A subtraction technique is used in each case to provide that at least  $O(1/N^4)$  convergence holds at all the stages of the algorithm.

In the process of the domain decomposition we introduce some boundary conditions on the interfaces. Certainly these conditions do not coincide with the final values on the interfaces, so another matching procedure is needed. If we are looking for a unique solution of the boundary value problem for (1.1) or (1.2), we assume that the (modified) Helmholtz equation is satisfied everywhere. If this condition does not hold, the solution is not unique. Therefore, in order not to introduce an additional singularity, which degrades the accuracy of the algorithm, we assume arbitrary boundary conditions on the interfaces subject to unique condition: they should match the right hand side along the edges of the interfaces where the Laplacian can be computed exactly. It remains to discuss the case when in certain places on the edges of the original boundary, the boundary conditions (the Laplacian can be computed exactly) does not match the right-hand side producing the singularity, which significantly degrades the accuracy of the finite difference or any other numerical methods. This problem in 2-D cases was treated in [15, 3]. The subtraction technique in the singular case is an appropriate solution for the problem [3]: we subtract a function, which satisfies the homogeneous equation and matches the singularity. This technique is straightforward in the 2-D case (the singularity can appear in the corners only) and it is much more complicated in three dimensions. See [5] for some examples on how singularities in the Poisson equation are treated.

**2. Algorithmic steps and operations count.** The proposed algorithm consists of the following steps:

*Step 1.* **Solution of the nonhomogeneous equation in each subdomain.**

Each subdomain is covered by a  $N \times N \times N$  grid. The right-hand side of the nonhomogeneous (modified) Helmholtz equation is evaluated at grid points of the extended subdomain  $((N + \varepsilon) \times (N + \varepsilon) \times (N + \varepsilon))$ , where  $\varepsilon$  is the number of the extension points in each direction. The Dirichlet boundary conditions are introduced on the interfaces. This is done to avoid the introduction of singularities on corners and edges. These boundary conditions in each subdomain match adjacent subdomains and match the right-hand side of the edges where the Laplacian of the solution can be computed from the boundary conditions. The solution of the nonhomogeneous equation is found in each subdomain using the highly accurate local Fourier bases method that was developed in [6]. The complexity of this step is  $O((N + \varepsilon)^3 \log(N + \varepsilon))$  operations for each subdomain.

**Step 2. Matching discontinuities.**

The solution that is obtained with the prescribed boundary conditions has discontinuities in the normal derivatives along the interfaces. To remove these discontinuities, the difference between the first derivatives at the two sides of the interfaces is computed. Then, harmonic functions are added to both sides of each interface. In the next steps, correction functions will be needed only on the interfaces. The accumulation of all the corrections determines the final local boundary conditions. The complexity of this step is  $O(k^2 N^2 \log N)$ , where  $k$  is the number of boxes that are simultaneously matched,  $N$  is the number of collocation points in each direction in each box.

**Step 3. Solution of the homogeneous equations in subdomains.**

The homogeneous (modified) Helmholtz equation is solved in each subdomain with the boundary conditions which were determined in the previous step. Combining the result with the solution of the nonhomogeneous equation leads to a smooth global solution. The complexity of this step is  $O(N^3 \log N)$ . For the Poisson equation the matching of discontinuities corresponds to the introduction of single layers (double layers in the case of the Neumann boundary conditions). In the (modified) Helmholtz equation case, the matching operation also changes the values of the solution on the global boundary. After this procedure the resulting solution does not satisfy the original boundary conditions any more.

**Step 4. Global solution.**

The smooth global solution, which was computed in the previous step, does not match the prescribed values on the boundaries due to the matching procedure. A global homogeneous (modified) Helmholtz equation is solved to satisfy the prescribed boundary conditions of the global domain. The complexity of this step is  $O(N^3 \log N)$ .

We showed in our previous works [5, 6] that the solution in a 3-D box (i.e., in each subdomain) can be computed with high-order accuracy (corresponding to the second- and fourth-order accuracies, etc.), depending on the order of our subtraction procedure.

The removal of interface jump can be inexpensive if initially (and later in each step) only adjacent boxes are matched. The present hierarchical approach, which matches only two adjacent boxes at each level, requires only local corrections at the boundaries of these boxes, such that only the solutions in the adjacent subdomains are coupled in each matching step, then these joint subdomains are matched, etc. If originally we had  $k^2$  subdomains, after  $2 \log k$  steps we obtain a smooth global solution.

In this paper we present the basic algorithm which is the solution of the Helmholtz equation (or the modified Helmholtz equation) and a matching of the solution in two adjacent boxes. For illustration we consider a box with eight subdomains. Then, the matching of a 3-D box, which is divided into eight subdomains (see Figure 2.1), can be completed in three steps: (1) Box 1 is matched with box 2, box 3 with 4, 5 with 6, 7 with 8. (2) The merged box 1,2 is matched with 3,4 and 5,6 with 7,8. (3) The whole slice 1,2,3,4 is patched to 5,6,7,8.

Similar ideas can be applied to a linear domain decomposition.

The hierarchical matching for  $m$  boxes in one line (see Figure 2.2) takes  $\log m$  steps. Each original or merged subdomain is matched to an adjacent subdomain. When nonhierarchical matching was performed, the influence of each derivative jump

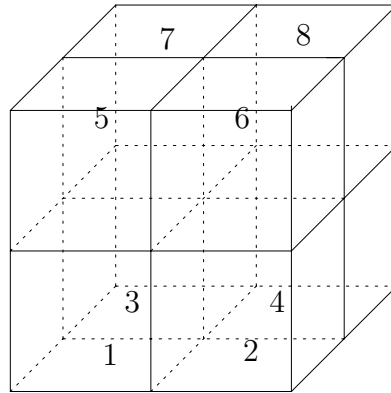


FIG. 2.1. *The domain is decomposed into eight subdomains.*

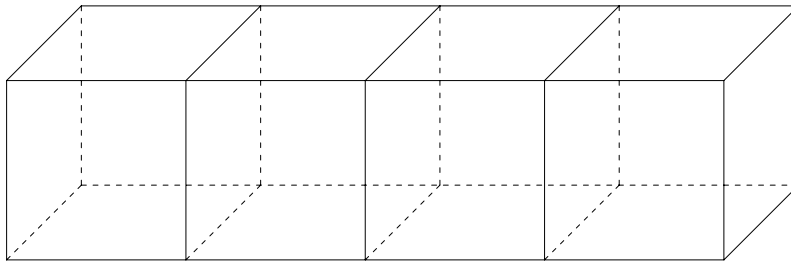


FIG. 2.2. *The domain is decomposed into  $m$  subdomains (here  $m = 4$ ).*

has to be computed in each interface.

The present scheme is part of a domain decomposition algorithm for the Helmholtz equation or the modified Helmholtz equation in a 3-D subdomain of a generally curvilinear geometry. The domain is decomposed into nonoverlapping subdomains, mainly of cubic or parallelepiped shape (see Figure 2.3). A smaller part of boundary subdomains are curvilinear. Particular solutions are found in the subdomains. The computation of the solution and matching is not expensive for parallelepiped subdomains. Irregular subdomains can be treated similar to 3-D Poisson solvers in general domains [8, 13, 14]. The idea of the matching procedure is just to add different harmonic functions from both sides of the interface to cancel the jump in the function or its first derivative. For curvilinear boundaries these functions are evaluated on the surfaces and then the homogeneous (modified) Helmholtz equation is solved in a curvilinear 3-D domain. To reduce the number of computations, most of the domain should be covered by subdomains of a parallelepiped shape. If we solve a free boundary problem where it is not necessary to satisfy the specified boundary conditions but the (modified) Helmholtz equation only, then the same algorithm can be used in a curvilinear domain as well. (The idea is similar to [9].) The right-hand side is smoothly extended in such a way that it is specified in an assembly of parallelepiped subdomains covering the original domain. First the modified Fourier method is used to solve an equation in each subdomain, then the matching functions at the interfaces are evaluated, and, finally, the homogeneous (modified) Helmholtz equation is solved in each subdomain. The sum of two solutions is computed for each of the points of the original (not extended) domain. Thus, the nonhomogeneous Helmholtz equation

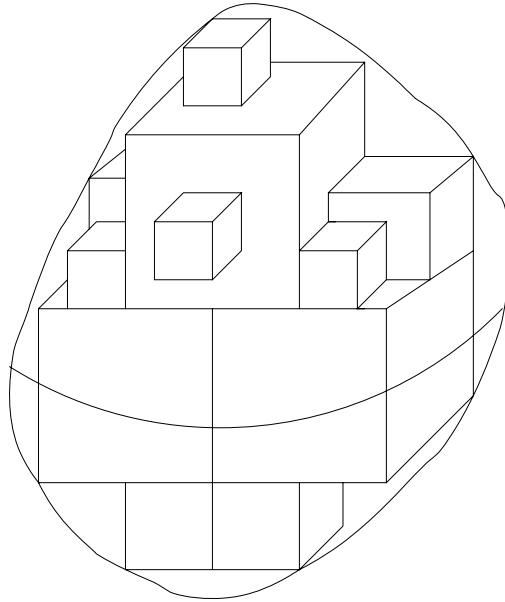


FIG. 2.3. A domain of a general geometry is divided into mostly parallelepiped subdomains. The nonhomogeneous equation is solved in each subdomain and then the solutions are patched. Matching functions are evaluated on both plane and curvilinear boundaries.

or the modified Helmholtz equation can be solved in a curvilinear domain without prescribed boundary conditions in  $O(N^3 \log N)$  operations.

When we have to satisfy the original boundary conditions, the computation of the global solution in the curvilinear global domain can become a time-consuming step in the algorithm. In section 6.3 we demonstrate how the global solution can be computed using partial grid which nearly does not degrade the accuracy. The results are presented for the parallelepiped domain; however, the idea of a sparse grid for the solution of the homogeneous equation can be applied for curvilinear domains as well. As explained in section 6.2, the domain decomposition technique can be incorporated with adaptive grids, which improves the efficiency of the algorithm.

**3. The matching procedure.** In this section we describe the matching procedure for two adjacent boxes, which is a basic step of the algorithm.

In the first step, compatible boundary conditions are introduced on the interface ABCD (see Figure 3.1) to avoid a solution's discontinuity. After we derived the solution of the homogeneous equation with the prescribed boundary conditions, the continuity on the interface is achieved. Nevertheless, the jump in the first derivative in the  $x$ -direction remains. In the second step we add certain solutions of the homogeneous equation to compensate for the jump, which completes the matching procedure for two boxes.

**3.1. Boundary conditions on the interfaces.** Boundary conditions on the interfaces should satisfy the Helmholtz equation at the edges. Consider the case when two subdomains are matched and the boundary conditions are defined at  $x = 0, 2$ ,  $y = 0, 1$ ,  $z = 0, 1$  (see Figure 3.1). Thus we have to introduce the boundary conditions only on the interface  $x = 1$ :  $u(1, y, z) = \varphi(y, z)$ . Now we will demonstrate that  $\varphi$  can be described as a solution of either a 2-D Helmholtz or a modified Helmholtz equation

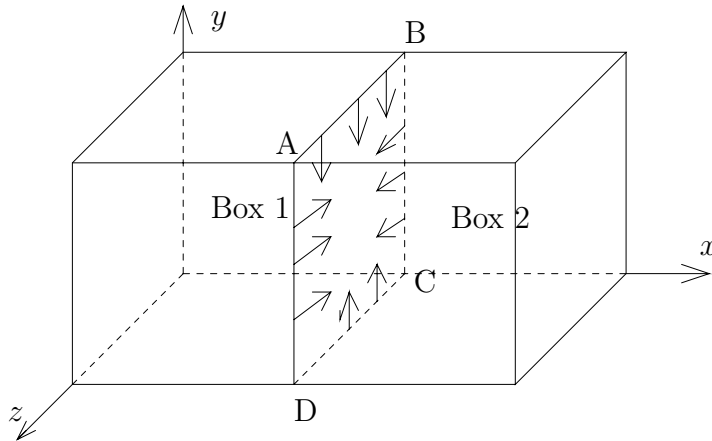


FIG. 3.1. We define the primary values on the interface  $x = 1$ ; the values and the second normal derivatives are known on the frame of the interface.

on the interface  $x = 1$ .

Initially, we derive  $\varphi$  for the modified Helmholtz equation (1.1). From the equality

$$(3.1) \quad \frac{\partial^2 u}{\partial y^2}(1, y, 0) + \frac{\partial^2 u}{\partial z^2}(1, y, 0) - \lambda^2 u(1, y, 0) = f(1, y, 0) - \frac{\partial^2 u}{\partial x^2}(1, y, 0) \equiv g_1(y, 0)$$

the function  $f(1, y, 0)$  is known, the second derivative in  $x$  along the edge can be found by the boundary condition on the interface  $z = 0$ , and thus  $g_1(y, 0)$  is easily computed. Similarly,

$$(3.2) \quad \frac{\partial^2 u}{\partial y^2}(1, y, 1) + \frac{\partial^2 u}{\partial z^2}(1, y, 1) - \lambda^2 u(1, y, 1) = f(1, y, 1) - \frac{\partial^2 u}{\partial x^2}(1, y, 1) \equiv g_1(y, 1),$$

$$(3.3) \quad \frac{\partial^2 u}{\partial y^2}(1, 0, z) + \frac{\partial^2 u}{\partial z^2}(1, 0, z) - \lambda^2 u(1, 0, z) = f(1, 0, z) - \frac{\partial^2 u}{\partial x^2}(1, 0, z) \equiv g_1(0, z),$$

$$(3.4) \quad \frac{\partial^2 u}{\partial y^2}(1, 1, z) + \frac{\partial^2 u}{\partial z^2}(1, 1, z) - \lambda^2 u(1, 1, z) = f(1, 1, z) - \frac{\partial^2 u}{\partial x^2}(1, 1, z) \equiv g_1(1, z).$$

Consequently, the 2-D function  $\varphi(y, z)$  satisfies the modified Helmholtz equation

$$(3.5) \quad \Delta\varphi(y, z) - \lambda^2\varphi(y, z) = g_1(y, z).$$

Now, let us specify an equation for  $\varphi$  for (1.2). We have

$$(3.6) \quad \frac{\partial^2 u}{\partial y^2}(1, y, 0) + \frac{\partial^2 u}{\partial z^2}(1, y, 0) + k^2 u(1, y, 0) = f(1, y, 0) - \frac{\partial^2 u}{\partial x^2}(1, y, 0) \equiv g_2(y, 0)$$

and the equalities, similar to (3.2)–(3.4) for the other edges of the wire frame ABCD, hold. Thus, the function  $\varphi(y, z)$  satisfies the 2-D Helmholtz equation

$$(3.7) \quad \Delta\varphi(y, z) + k^2\varphi(y, z) = g_2(y, z).$$

The function  $\varphi(y, z)$  is known on the wire frame ABCD together with its 2-D Laplacian. We should construct the function  $g_1$  ( $g_2$  for the Helmholtz equation (3.7)) inside the rectangle ABCD and assume  $\varphi(y, z)$  to be a solution of (3.5) with the above boundary conditions on the frame ABCD.

We can choose the functions  $g_1(x, y)$  and  $g_1(x, y)$  in an arbitrary form. (An incorrect choice will be later balanced by the proper matching procedure.) They should only have the specified values on the boundary frame.

Let us construct a smooth 2-D function  $g_1$ , which is known at the wire frame ABCD (see (3.1)–(3.4)). If the solution of the original boundary value problem is not smooth, this will arise at the matching step. However, if the original solution is smooth there is no reason to introduce additional singularities in the solution. Numerical examples include both smooth and steep solution cases.

We used in practical implementations the following scheme to extend the function  $g_1$  known on the wire frame only to the whole rectangle. Let

$$(3.8) \quad a = \frac{\partial^2 g_1}{\partial y^2}(0, 0) = -\frac{\partial^2 g_1}{\partial z^2}(0, 0) = -b$$

and the same for all the other corners. Further,  $g$  was presented as a sum of four known functions  $h_i$ , which subtract Laplacian in the corners and some harmonic function. For example, the function

$$h_1(y, z) = \frac{a + b}{2(\lambda_1^2 - \lambda_2^2)} \left[ \left( \frac{\sinh(\lambda_1(1 - y))}{\sinh \lambda_1} - \frac{\sinh(\lambda_2(1 - y))}{\sinh \lambda_2} \right) (1 - z) \right. \\ \left. + \left( \frac{\sinh(\lambda_1(1 - z))}{\sinh \lambda_1} - \frac{\sinh(\lambda_2(1 - z))}{\sinh \lambda_2} \right) (1 - y) \right]$$

has the same Laplacian at  $(0, 0)$  as  $g_1$ . Three other corners are similarly treated. Further,  $g_1 - h_1 - h_2 - h_3 - h_4$  was constructed as a harmonic function with boundary conditions obtained as a difference. The solution of the Laplace equation in a 2-D domain was based on the algorithm developed in [4]. The same construction was applied to  $g_2$ .

Once  $g_1$  ( $g_2$ ) is known, then  $\varphi(y, z) = u(1, y, z)$  is assumed to be a solution of the 2-D modified Helmholtz equation (3.5) (the 2-D Helmholtz equation (3.7)). We introduce a singularity unless the boundary conditions on the edge of the interface match the Laplacian described by the right-hand side and the boundary conditions. Such a singularity can essentially degrade the accuracy of the algorithm.

**3.2. Matching of boxes.** After we find a solution for the Helmholtz equation in each subdomain, there is a jump in the first derivative in the  $x$ -direction. We add symmetric functions in the following way to match the solutions. Let the jump in the first derivative be

$$(3.9) \quad \frac{\partial u}{\partial x}(x_0+, y, z) - \frac{\partial u}{\partial x}(x_0-, y, z) = \phi(y, z).$$

Since the first derivative in  $x$  is continuous on the frame  $x = x_0, y = 0, 1$  or  $z = 0, 1$ , then  $\phi(y, z)$  vanishes on the boundary. Thus, the second derivative of  $\phi$  in  $z$  is periodic on the boundaries  $z = 0, 1, 0 \leq y \leq 1$ , and so is the second derivative of  $\phi$  in  $y$  at the boundaries  $y = 0, 1, 0 \leq z \leq 1$ . We expand the second derivative into the sine series

$$(3.10) \quad \frac{\partial^2 \phi}{\partial z^2}(y, 0) = \sum_j a_j \sin(\pi j y).$$

For the *modified Helmholtz equation* (1.1) we add the following function to the solution for  $x \geq x_0$ :

$$(3.11) \quad \sum_j \frac{a_j}{2(\lambda_1^2 - \lambda_2^2)} \sin(\pi j y) \left[ \frac{\sin(\lambda_1(1-z)) \sinh(\mu_{1j}(L-x+x_0))}{\sin(\lambda_1) \mu_{1j} \cosh(\mu_{1j}L)} - \frac{\sin(\lambda_2(1-z)) \sinh(\mu_{2j}(L-x+x_0))}{\sin(\lambda_2) \mu_{2j} \cosh(\mu_{2j}L)} \right],$$

where

$$(3.12) \quad \mu_{1j} = \sqrt{\lambda_1^2 + \pi^2 j^2 + \lambda^2}, \mu_{2j} = \sqrt{\lambda_2^2 + \pi^2 j^2 + \lambda^2}.$$

A symmetric function with respect to the plane  $x = x_0$  (in the present example  $x_0 = 1$ )

$$(3.13) \quad \sum_j \frac{a_j}{2(\lambda_1^2 - \lambda_2^2)} \sin(\pi j y) \left[ \frac{\sin(\lambda_1(1-z)) \sinh(\mu_{1j}(L+x-x_0))}{\sin(\lambda_1) \mu_{1j} \cosh(\mu_{1j}L)} - \frac{\sin(\lambda_2(1-z)) \sinh(\mu_{2j}(L+x-x_0))}{\sin(\lambda_2) \mu_{2j} \cosh(\mu_{2j}L)} \right]$$

is added for  $x \leq x_0$ . Here  $L$  is chosen to be the maximal distance from the interface  $x = x_0$  to the boundary (here  $L = 1$ ). Similar functions are subtracted for the other three boundaries.

For the *Helmholtz equation* (1.2)  $\lambda_1, \lambda_2$  can be chosen to exceed  $k$ , and then the same functions (3.11) and (3.13) are added to the right and to the left of the interface, respectively, with

$$(3.14) \quad \mu_{1j} = \sqrt{\lambda_1^2 + \pi^2 j^2 - k^2}, \mu_{2j} = \sqrt{\lambda_2^2 + \pi^2 j^2 - k^2}.$$

After the completion of this operation, the remaining jump in the derivative  $\phi_1(y, z)$  vanishes along the boundary together with its second derivatives in  $y$  and  $z$ . Thus, the function  $\phi_1(x, y)$  can be accurately expanded into a 2-D sine series

$$(3.15) \quad \phi_1(x, y) = \sum_i \sum_j a_{ij} \sin(\pi i x) \sin(\pi j y).$$

Then, after the function

$$(3.16) \quad \sum_i \sum_j \frac{1}{2} a_{ij} \sin(\pi i y) \sin(\pi j z) \frac{\sinh(\mu_{ij}(L-x+x_0))}{\mu_{ij} \cosh(\mu_{ij}L)},$$

with  $\mu_{ij} = \sqrt{\pi^2(i^2 + j^2) + \lambda^2}$ , is added for  $x \geq x_0$  and a symmetric function with respect to the plane  $x = x_0$  for  $x \leq x_0$ , we obtain that the first derivative in  $x$  is matched across the interface  $x = x_0$ . Similarly, the derivative is matched on the other interfaces.

The analogue of (3.16) for the Helmholtz equation (1.2) is

$$(3.17) \quad \sum_{\pi^2(i^2+j^2) < k^2} \frac{1}{2} a_{ij} \sin(\pi iy) \sin(\pi jz) \frac{\sin(\lambda_{ij}(L-x+x_0))}{\lambda_{ij} \cos(\lambda_{ij}L)} + \sum_{\pi^2(i^2+j^2) > k^2} \frac{1}{2} a_{ij} \sin(\pi iy) \sin(\pi jz) \frac{\sinh(\mu_{ij}(L-x+x_0))}{\mu_{ij} \cosh(\mu_{ij}L)},$$

where

$$(3.18) \quad \lambda_{ij} = \sqrt{k^2 - \pi^2(i^2 + j^2)}, \quad \mu_{ij} = \sqrt{\pi^2(i^2 + j^2) - k^2}.$$

Certainly, we do not evaluate all the add-on functions in all the domains but only on the boundary and on other interfaces. Afterward, the sum of all these functions is computed and the homogeneous Helmholtz equation with the corresponding boundary conditions is solved. The addition of this solution to the previous ones in each subdomain matches the first derivative of the solution on the interfaces; i.e., we obtain a smooth solution.

**4. Numerical results.** Assume that  $u$  is the exact solution and  $u'$  is the computed solution. Let  $u_i$  and  $u'_i$  be the values of  $u$  and  $u'$  at the collocation points, respectively. In the following examples we will use the following measures to estimate the errors:

$$\begin{aligned} \varepsilon_{MAX} &= \max \|u'_i - u_i\| \\ \varepsilon_{MSQ} &= \sqrt{\frac{\sum_{i=1}^N (u'_i - u_i)^2}{n}} \\ \varepsilon_{\mathcal{L}^2} &= \sqrt{\frac{\sum_{i=1}^N (u'_i - u_i)^2}{\sum_{i=1}^N u_i^2}}. \end{aligned}$$

**4.1. Modified Helmholtz equation.**

*Example 4.1.* We solve the modified Helmholtz equation in  $[0, 1] \times [0, 1] \times [0, 2]$  with  $\lambda = 1$ . The right-hand side and the boundary conditions correspond to the exact solution

$$(4.1) \quad u(x, y, z) = \cos(x - 0.4) \cos(y - 0.5) \cos(z - 1.0).$$

Numerical results are presented in Table 1.

TABLE 1

*MAX, MSQ, and  $\mathcal{L}^2$  errors for the solution of the modified Helmholtz equation with  $\lambda = 1$  by matching two boxes for the exact solution (4.1).*

$N_x \times N_y \times N_z$	$\varepsilon_{MAX}$	$\varepsilon_{MSQ}$	$\varepsilon_{\mathcal{L}^2}$
$8 \times 8 \times 8$	3.9e-5	7.4e-6	3.0e-5
$16 \times 16 \times 16$	2.1e-6	3.8e-7	1.5e-6
$32 \times 32 \times 32$	1.5e-7	3.1e-8	1.2e-7
$64 \times 64 \times 64$	1.1e-8	2.2e-9	8.5e-9

*Example 4.2.* We solve the modified Helmholtz equation in  $[0, 1] \times [0, 1] \times [0, 2]$  with  $\lambda = 1$ . The right-hand side and the boundary conditions correspond to the exact solution

$$(4.2) \quad u(x, y, z) = \exp \left\{ -\alpha \left( (x - 0.4)^2 + (y - 0.5)^2 + z^2 \right) \right\}.$$

TABLE 2

*MAX*, *MSQ*, and  $\mathcal{L}^2$  errors for the solution of the modified Helmholtz equation by matching two boxes for the exact solution (4.2).  $\alpha = 1$ .

$N_x \times N_y \times N_z$	$\varepsilon_{MAX}$	$\varepsilon_{MSQ}$	$\varepsilon_{\mathcal{L}^2}$
$8 \times 8 \times 8$	4.7e-4	7.9e-5	1.7e-4
$16 \times 16 \times 16$	3.2e-5	5.6e-6	1.2e-5
$32 \times 32 \times 32$	2.2e-6	4.0e-7	8.4e-7
$64 \times 64 \times 64$	1.4e-7	2.7e-8	5.7e-8

In Table 2  $\alpha = 1$ .

We observe that in Examples 4.1 and 4.2 the rate of convergence is  $O(h^4)$ . (The error is 16 times less when the number of points in each direction is doubled.) This matches the theoretical results of [7]. See also [5] for the 3-D case.

Tables 3 and 4 present the dependence of the errors on the steepness  $\alpha$  of the Gaussian and the equation parameter  $\lambda$ . The number of points in each direction is 16. The number of extension points is 4.

TABLE 3

*MAX*, *MSQ*, and  $\mathcal{L}^2$  errors for the solution of the modified Helmholtz equation by matching two boxes for the exact solution (4.2) and  $16 \times 16 \times 16$  points in each subdomain with  $\lambda = 1$  and varying  $\alpha$ .

$\alpha$	$\varepsilon_{MAX}$	$\varepsilon_{MSQ}$	$\varepsilon_{\mathcal{L}^2}$
0.1	6.4e-7	1.2e-7	1.4e-7
0.2	1.8e-6	3.4e-7	4.4e-7
0.5	9.9e-6	1.8e-6	3.0e-6
0.8	2.3e-5	4.1e-6	8.0e-6
1.0	3.2e-5	5.6e-6	1.2e-5
2.0	3.5e-5	5.3e-6	1.5e-5
5.0	3.3e-5	5.9e-6	3.0e-5
8.0	7.5e-5	1.1e-5	7.5e-5
10.0	7.3e-5	9.9e-6	8.0e-5
12.0	4.6e-5	5.9e-6	5.5e-5
15.0	1.6e-5	1.9e-6	2.1e-5
20.0	1.7e-6	1.9e-7	2.5e-6

TABLE 4

*MAX*, *MSQ*, and  $\mathcal{L}^2$  errors for the solution of the modified Helmholtz equation by matching two boxes for the exact solution (4.2) and  $16 \times 16 \times 16$  points in each subdomain with varying  $\lambda$ .

$\lambda^2$	$\varepsilon_{MAX}$	$\varepsilon_{MSQ}$	$\varepsilon_{\mathcal{L}^2}$
0.01	2.8e-5	5.1e-6	1.1e-5
0.09	2.9e-5	5.1e-6	1.1e-5
0.49	3.0e-5	5.3e-6	1.1e-5
1.0	3.2e-5	5.6e-6	1.2e-5
4.0	4.3e-5	7.2e-6	1.5e-5
16.0	1.0e-4	1.5e-5	3.3e-5
25.0	1.6e-4	2.3e-5	4.8e-5
49.0	3.6e-4	4.6e-5	9.7e-5
100.0	9.2e-4	1.0e-4	2.2e-4
225.0	2.7e-3	2.7e-4	5.7e-4

*Example 4.3.* We solve the modified Helmholtz equation in  $[0, 1] \times [0, 1] \times [0, 2]$  with  $\lambda = 1$ . The right-hand side and the boundary conditions correspond to the exact

solution which is a sum of random Gaussian functions

$$(4.3) \quad u(x, y, z) = \sum_{i=1}^{12} \exp \{ -\alpha_i ((x - x_i)^2 + (y - y_i)^2 + (z - z_i)^2) \},$$

where  $(x_i, y_i, z_i)$  and  $\alpha_i$  are

$$(0.23, 0.54, 0.32), (0.20, 0.65, 0.30), (0.03, 0.78, 0.02), (0.74, 0.06, 0.24),$$

$$(0.12, 0.26, 0.28), (0.28, 0.83, 0.09), (0.69, 0.19, 0.31), (0.86, 0.37, 0.11),$$

$$(0.37, 0.55, 0.33), (0.99, 0.71, 0.12), (0.17, 0.34, 0.18), (0.46, 0.96, 0.37),$$

and

$$\alpha_i = 2.0, 8.0, 0.04, 4.5, 9.7, 12.3, 6.9, 8.2, 11.5, 7.2, 10.7, 9.6,$$

respectively.

TABLE 5

*MAX, MSQ, and  $\mathcal{L}^2$  errors for the solution of the modified Helmholtz equation by matching two boxes for the exact solution (4.3), which is a sum of random Gaussian functions.*

$N_x \times N_y \times N_z$	$\varepsilon_{MAX}$	$\varepsilon_{MSQ}$	$\varepsilon_{\mathcal{L}^2}$
$8 \times 8 \times 8$	3.8e-2	5.8e-3	2.8e-3
$16 \times 16 \times 16$	1.1e-3	1.8e-4	8.5e-5
$32 \times 32 \times 32$	2.9e-5	4.4e-6	2.1e-6
$64 \times 64 \times 64$	9.5e-7	1.0e-7	4.8e-8

In Table 5 we get low accuracy for low resolutions; nevertheless, the convergence is even faster than what the theory predicts: the maximal error decays 30 to 38 times when the number of points is doubled.

Numerical results for exact solutions (4.4)–(4.6) are presented in Tables 6–8.

**4.2. Helmholtz equation.**

*Example 4.4.* We solve the Helmholtz equation in  $[0, 1] \times [0, 1] \times [0, 2]$  with  $k = 1$ . The right-hand side and the boundary conditions correspond to the exact solution

$$(4.4) \quad u(x, y, z) = \cos(x - 0.4) \cos(y - 0.5) \cos(z - 1.0).$$

TABLE 6

*MAX, MSQ, and  $\mathcal{L}^2$  errors for the solution of the Helmholtz equation for the exact solution (4.4) with  $k = 1$  by matching two boxes.*

$N_x \times N_y \times N_z$	$\varepsilon_{MAX}$	$\varepsilon_{MSQ}$	$\varepsilon_{\mathcal{L}^2}$
$8 \times 8 \times 8$	1.4e-5	2.8e-6	1.1e-5
$16 \times 16 \times 16$	6.1e-7	1.2e-7	4.7e-7
$32 \times 32 \times 32$	4.5e-8	9.5e-9	3.7e-8
$64 \times 64 \times 64$	3.1e-9	6.8e-10	2.6e-9

*Example 4.5.* We solve the Helmholtz equation in  $[0, 1] \times [0, 1] \times [0, 2]$  with  $k = 1$ . The right-hand side and the boundary conditions correspond to the exact solution

$$(4.5) \quad u(x, y, z) = \frac{1}{\sqrt{(x + 0.3)^2 + (y + 0.4)^2 + (z + 0.5)^2}}.$$

TABLE 7

*MAX*, *MSQ*, and  $\mathcal{L}^2$  errors for the solution of the Helmholtz equation for the exact solution (4.5) with  $k = 1$  by matching two boxes.

$N_x \times N_y \times N_z$	$\varepsilon_{MAX}$	$\varepsilon_{MSQ}$	$\varepsilon_{\mathcal{L}^2}$
$8 \times 8 \times 8$	2.1e-4	2.3e-5	4.0e-5
$16 \times 16 \times 16$	3.2e-6	3.5e-7	6.2e-7
$32 \times 32 \times 32$	2.8e-7	1.8e-8	3.3e-8
$64 \times 64 \times 64$	1.1e-8	1.1e-9	2.0e-9

*Example 4.6.* We solve the Helmholtz equation in  $[0, 1] \times [0, 1] \times [0, 2]$  with  $k = 1$ . The right-hand side and the boundary conditions correspond to the exact solution

$$(4.6) \quad u(x, y, z) = \exp \left\{ -\alpha \left( (x - 0.4)^2 + (y - 0.5)^2 + z^2 \right) \right\}.$$

TABLE 8

*MAX*, *MSQ*, and  $\mathcal{L}^2$  errors for the solution of the Helmholtz equation for the exact solution (4.6) with  $k = 1$  and  $\alpha = 1$  by matching two boxes.

$N_x \times N_y \times N_z$	$\varepsilon_{MAX}$	$\varepsilon_{MSQ}$	$\varepsilon_{\mathcal{L}^2}$
$8 \times 8 \times 8$	4.0e-4	6.8e-5	1.5e-4
$16 \times 16 \times 16$	2.5e-5	4.6e-6	9.7e-6
$32 \times 32 \times 32$	1.6e-6	3.1e-7	6.6e-7
$64 \times 64 \times 64$	1.1e-7	2.1e-8	4.4e-8

We observe that in Examples 4.4–4.6 the rate of convergence is  $O(h^4)$  (the error is 16 times less when the number of points in each direction is doubled), similar to the case of the modified Helmholtz equation.

Tables 9 and 10 present the dependence of the errors on the steepness  $\alpha$  of the Gaussian and the parameter  $k$ . The number of points in each direction is 16 and the number of extension points is 4.

TABLE 9

*MAX*, *MSQ*, and  $\mathcal{L}^2$  errors for the solution of the Helmholtz equation for the exact solution (4.6) by matching two boxes with  $16 \times 16 \times 16$  points in each subdomain with  $k = 1$  and varying  $\alpha$ .

$\alpha$	$\varepsilon_{MAX}$	$\varepsilon_{MSQ}$	$\varepsilon_{\mathcal{L}^2}$
0.1	3.3e-5	3.2e-6	3.7e-6
0.2	6.3e-6	6.4e-7	8.3e-7
0.5	6.3e-6	1.2e-6	2.0e-6
0.8	1.7e-5	3.2e-6	6.2e-6
1.0	2.5e-5	4.6e-6	9.7e-6
2.0	2.7e-5	4.2e-6	1.2e-5
5.0	3.7e-5	6.6e-6	3.3e-5
8.0	7.9e-5	1.2e-5	8.0e-5
10.0	7.6e-5	1.1e-5	8.6e-5
12.0	4.8e-5	6.4e-6	5.9e-5
15.0	1.7e-5	2.1e-6	2.2e-5
20.0	1.8e-6	2.0e-7	2.7e-6

**5. Matching of arbitrary number of boxes.** The above procedure for matching two adjacent subdomains is a part of a more general algorithm.

Assume the domain is divided into  $2^m$  segments in each direction, resulting in  $2^{3m}$  subdomains (see Figure 5.1). Thus, the matching can be implemented successively (hierarchically): the first eight adjacent subdomains are matched (see Figure 2.1).

TABLE 10

$MAX$ ,  $MSQ$  and  $\mathcal{L}^2$  errors for the solution of the modified Helmholtz equation for the exact solution (4.6) by matching two boxes with  $16 \times 16 \times 16$  points in each subdomain with varying  $k$ .

$k^2$	$\varepsilon_{MAX}$	$\varepsilon_{MSQ}$	$\varepsilon_{\mathcal{L}^2}$
0.01	2.8e-5	5.1e-6	1.1e-5
0.09	2.8e-5	5.0e-6	1.1e-5
0.49	2.7e-5	4.8e-6	1.0e-5
1.0	2.5e-5	4.6e-6	9.7e-6
4.0	1.7e-5	3.1e-6	6.6e-6
16.0	2.5e-5	3.7e-6	7.9e-6
49.0	3.6e-4	4.6e-5	9.7e-5
100.0	9.7e-4	2.5e-4	5.3e-4

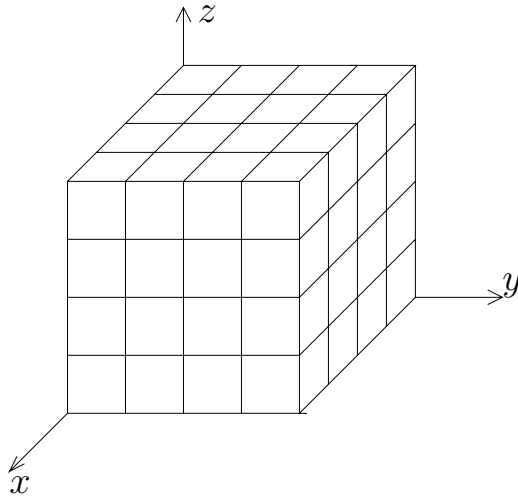


FIG. 5.1. The domain is divided into  $2^{3m}$  subdomains which are hierarchically matched. It takes  $3m$  steps; in each step two adjacent subdomains are merged.

Then adjacent cubic subdomains are matched, etc. The only problem is how to define the primary boundary conditions on the interfaces. Following the ideas in section 3, this can be hierarchically implemented beginning with the largest interface. In Figure 3.1, for example, first the boundary conditions between the merged boxes 1-4 and 5-8 are introduced, then using the same algorithm the interfaces between 1-2 and 3-4, 5-6 and 7-8 are described. Finally, for the smallest interfaces between boxes 1 and 2, 3 and 4, 5 and 6, 7 and 8, the boundary conditions are computed. This is implemented as a preprocessing step, before the first matching procedure began.

However, it is possible to define the initial boundary conditions on the interfaces locally. Consider the case when the domain is divided into parallelepiped subdomains. We can set arbitrary values for the solution  $u$  on the horizontal interfaces.

If  $u$  is assumed to vanish on the horizontal interfaces, then the derivatives of  $u$  in  $x$  and  $y$  vanish and therefore the second derivative in  $z$  can be evaluated. Let us define  $f_1$  and  $f_2$  as in the 2-D case: they are functions of  $z$  that at the ends of the vertical edges vanish and have the second derivatives, which were computed above. The values at the faces parallel to  $XZ$  plane (see Figure 5.2) are defined as in section 3. Thus, the compatible boundary conditions are determined on the cylindrical envelope.

The operations count is described below for the general algorithm (see Figure 5.1).

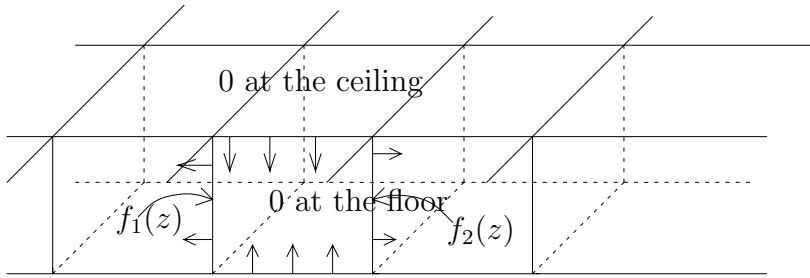


FIG. 5.2. In the first step we define consistent boundary conditions on all the interfaces.

We assume that there are  $N \times N \times N$  collocation points in the smallest subdomains and the number of smallest subdomains is  $2^{3m}$ . Thus we have  $m$  successive matching steps. The basic number of grid points in each direction is doubled in each next step.

**5.1. Operations count.**

**Step 1. Solution of the nonhomogeneous equation in each subdomain.**

This step is local, so its complexity is  $O(2^{3m}(N + \varepsilon)^3 \log(N + \varepsilon))$ , where  $\varepsilon$  is the number of extension points.

**Step 2. Matching of discontinuities.**

The complexity of this step is  $C =$

$$O(2^{3(m-1)}N^2 \log N) + O(2^{3(m-2)}8N^2 \log(2N)) + \dots + O(2^{3m}N^2 \log(2^m N)),$$

which satisfies

$$O(2^{3(m-1)}N^2m \log N) < C < O(2^{3(m-1)}N^2m(\log N + m)).$$

**Step 3. Solution of the homogeneous equations in subdomains.**

For the smallest subdomains the complexity of this step is

$$O(2^{3m}N^3 \log N).$$

**Step 4. Global solution.**

For the smallest subdomains the complexity of this step is

$$O(2^{3(m-1)}N^3 \log N).$$

The total number of operations for all the matching steps is

$$O(2^{3(m-1)}N^3 \log N) + O(2^{3(m-1)}N^3 \log(2N)) + \dots + O(2^{3(m-1)}N^3 \log(2^m N)) > O(m2^{3(m-1)}N^3 \log N),$$

which (if immediately implemented) is the most expensive part of the algorithm.

Ways to optimize the above algorithm are discussed in section 6.

Let us compare between the time needed to implement a domain decomposition algorithm and the algorithm when the geometry of the domain is linear (see Figure 2.2) and the domain are matched in a naive way. In the matching step all the matched functions are evaluated on all the interfaces. Figure 5.3 presents the number of operations (divided by  $10^7$ ) in both algorithms.

We observe that the number of operations for the implementation of the domain decomposition algorithm is insignificantly greater than the local Fourier bases method in the global domain. The difference increases when the number of subdomains becomes larger (and so the matching step proportionally becomes more expensive). On

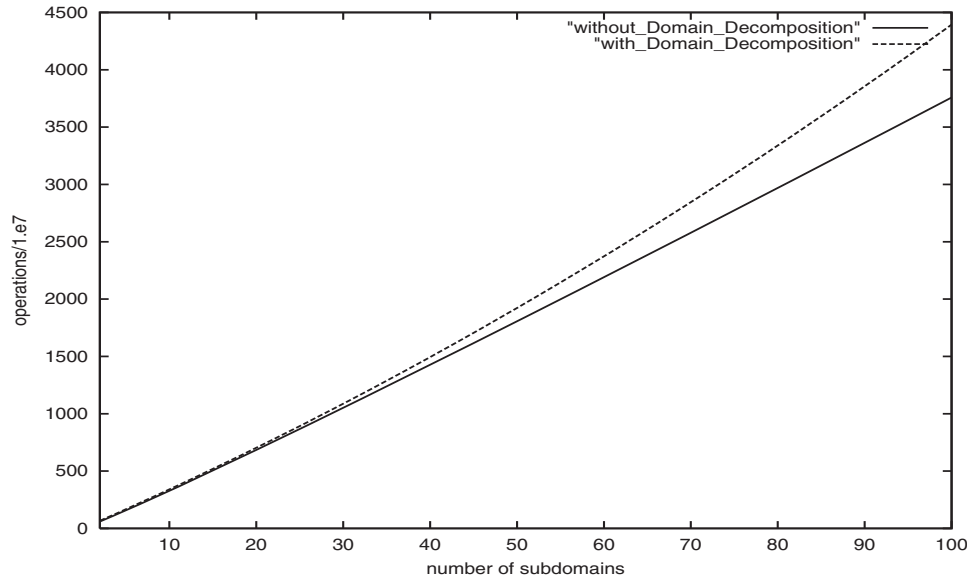


FIG. 5.3. The number of operations (divided by  $10^7$ ) that are needed to implement the domain decomposition algorithm. It is compared to the local Fourier bases algorithm; the number of points in each subdomain is  $64 \times 64 \times 64$ ; the number of extension points is 8 points from each side of the domain.

one hand, the number of operations for the application of the FFT is  $O(N^3 \log N)$ , which is  $\log N$  per point and so the domain decomposition leads to a more efficient algorithm. On the other hand, the domain decomposition algorithm includes some additional steps, like having twice a local solution of the homogeneous Helmholtz equation in the subdomains and the matching procedure, which degrades the efficiency of the algorithm. It is to be noted that a similar 2-D domain decomposition algorithm becomes competitive (compared to the local Fourier bases method) when the number of subdomains is  $m \geq 56$  [2].

The domain decomposition algorithm has the following advantages when compared to the method without domain decomposition:

1. The domain decomposition algorithm has an intrinsic parallel structure and allows a parallel implementation with reduced communication.
2. The choice of adaptive grids in subdomains can contribute to both efficiency and accuracy of the algorithm.
3. The domain decomposition algorithm can be applied to solve the equation in the greater part of a curvilinear domain of a curvilinear domain (see Figure 2.3) than the algorithm for a parallelepiped subdomain. Thus it becomes an essential part of the domain decomposition solution of the (modified) Helmholtz equation in a curvilinear subdomain.

**6. Optimization of the algorithm.** In this section we investigate the ways to optimize the above algorithm and we present relevant numerical results.

### 6.1. Summary of methods.

1. *Adaptive grids resolutions for the solution of the nonhomogeneous equation (Step 1).* The behavior of the right-hand side can differ from subdomain to subdomain. Therefore, the choice of different adaptive grids in the subdo-

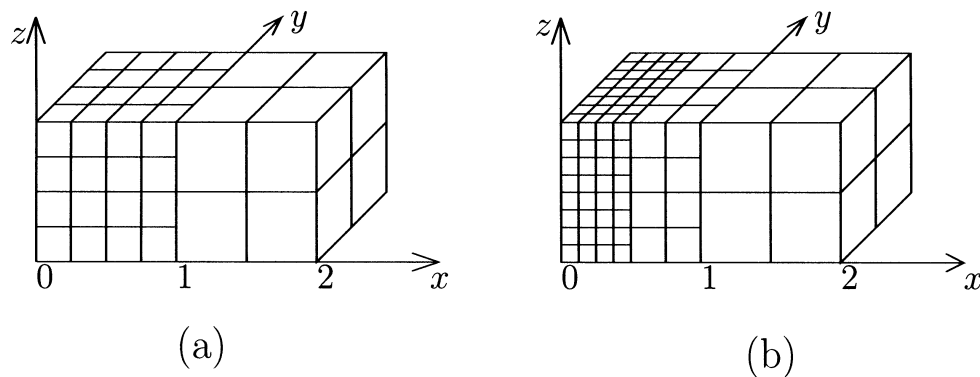


FIG. 6.1. Different grid resolutions can be chosen in adjacent subdomains: (a) the grid in the left box is twice denser than the right box; (b) three different grid resolutions can be chosen if the domain is divided into three subdomains.

mains can reduce the amount of work that is needed to implement Step 1. Thus, the number of collocation points can differ from subdomain to subdomain that have the same size (see Figure 6.1). For the 2-D case, this development was described and approved numerically in [1]. Obviously, the number of operations is significantly less if the right-hand side is mostly smooth in most subdomains while exhibiting steep and irregular behavior in a small number of subdomains. This obviously influences Step 3 as well.

2. *Adaptive matching of subdomains (Step 2)*. The idea is similar to Step 1. Only the resolution depends on the size of the box. When we compute the influence of the derivative jump on the interfaces we can take a constant number of points per interface, i.e., the number of computations does not grow when larger boxes are matched. For example, if in Figure 3.1 grids of  $8 \times 8$  collocation points at the interface are used to match box 1 with box 2, etc., then  $8 \times 8$  points are taken in the interface between the merged boxes 1,2 and 3,4 (points in the  $x$ -direction are twice less dense); finally,  $8 \times 8$  points in the interface are used to match the lower slice 1-4 with the upper slice 5-8. (The points are twice less dense both in the  $x$  and  $y$  directions.)
3. *Sparse global solution (Step 4)*. On one hand, the global solution of the Dirichlet problem is one of the most time-consuming steps of the algorithm (see [1] for the 2-D case) since the number of grid points is doubled when two adjacent subdomains are merged. On the other hand, the steepest parts of the solution, induced by charge distribution and the initial boundary conditions, are computed in Steps 1 and 3 (solution of the nonhomogeneous equation and local Dirichlet problems in the subdomains). Thus, if part of the solution, which is to be computed in Step 4, is smooth, then it does not need many grid points to get an accurate evaluation. Therefore, it is natural to take  $N \times N \times N$  points when merging two  $N \times N \times N$  boxes; in the  $x$ -direction (see Figure 3.1) we take every second point. This procedure is repeated when bigger boxes are merged. So in each step the global solution of the Dirichlet problem requires the same number of operations  $O(N^3 \log N)$ , where  $N$  is the number of collocation points in each direction in the smallest subdomain.

**6.2. Adaptive computation.** Let us consider the case when the right-hand side is smooth in some subdomains and very steep in the other parts. As a benchmark let us choose a steep Gaussian function

$$(6.1) \quad u(x, y, z) = \exp \left\{ -200 \left[ (x - x_0)^2 + (y - y_0)^2 + (z - z_0)^2 \right] \right\},$$

where  $x_0 = 0, y = 0.54, z = 0.43$ . A similar problem for the 2-D case was tested in [9].

**6.2.1. Adaptive computation for the modified Helmholtz equation.**

*Example 6.1.* We solve the modified Helmholtz equation (1.1) where  $\lambda = 1$  and the exact solution (6.1) using adaptive grids. The results are presented in Table 11.

TABLE 11

*MAX, MSQ, and  $\mathcal{L}^2$  errors for the adaptive solution of the modified Helmholtz equation (1.1);  $\lambda = 1$  and the exact solution is (6.1).  $N$  is the number of grid points in all the 3-D domain.*

$N$	$\varepsilon_{MAX}$	$\varepsilon_{MSQ}$	$\varepsilon_{\mathcal{L}^2}$
4608	9.8e-4	5.4e-5	4.0e-3
36864	3.9e-8	2.5e-9	1.6e-7
294912	6.3e-10	3.0e-11	2.0e-9
524288	1.0e-13	3.0e-15	2.1e-13

**6.2.2. Adaptive computation for the Helmholtz equation.**

*Example 6.2.* We solve the Helmholtz equation (1.2) where  $\lambda = 1$  and the exact solution (6.1) using adaptive grids. The results are presented in Table 12.

TABLE 12

*MAX, MSQ, and  $\mathcal{L}^2$  errors for the adaptive solution of the Helmholtz equation (1.2);  $\lambda = 1$  and the exact solution is (6.1).  $N$  is the number of grid points in all 3-D domain.*

$N$	$\varepsilon_{MAX}$	$\varepsilon_{MSQ}$	$\varepsilon_{\mathcal{L}^2}$
4608	1.8e-3	1.4e-4	5.8e-3
36864	3.8e-8	2.6e-9	1.6e-7
294912	6.4e-10	3.1e-11	2.1e-9
524288	1.0e-13	3.2e-15	2.3e-13

We can observe that in both cases (the Helmholtz and the modified Helmholtz equation) with an adaptive grid we achieve an accuracy of  $10^{-8}$  for  $4 \cdot 10^4$  points which corresponds to approximately 33 points in each direction for an equispaced grid and  $10^{-13}$  for  $5 \cdot 10^5$  points ( $\approx 80$  points in each direction).

**6.3. Sparse global solution.** Here we present a numerical illustration of the sparse global solution (item 3) as a novel element in the development of the algorithm which is crucial for its efficiency.

**6.3.1. Sparse global solution of the modified Helmholtz equation.**

*Example 6.3.* We solve the same modified Helmholtz equation as in Example 4.1 for both regular and sparse global solutions. The results are presented in Table 13.

*Example 6.4.* We solve the same modified Helmholtz equation as in Example 4.2 for both regular and sparse global solutions. The results are presented in Table 14.

**6.3.2. Sparse global solution for the Helmholtz equation.**

*Example 6.5.* We solve the same Helmholtz equation as in Example 4.5 for both regular and sparse global solutions. The results are presented in Table 15.

*Example 6.6.* We solve the same Helmholtz equation as in Example 4.6 for both regular and sparse global solutions. The results are presented in Table 16.

TABLE 13

Accuracy of the solution of the modified Helmholtz equation with  $\lambda = 1$  by matching two boxes. The exact solution is  $u(x, y, z) = \cos(x-0.4) \cos(y-0.5) \cos(z-1.0)$  (4.4). Two cases are considered: (1) all the points participate in the computation of the global solution of the Dirichlet problem (regular global solution); (2) only half the points in the  $x$ -direction participate in the computation (sparse global solution).

$N_x \times N_y \times N_z$	regular global solution			sparse global solution		
	$\varepsilon_{MAX}$	$\varepsilon_{MSQ}$	$\varepsilon_{\mathcal{L}^2}$	$\varepsilon_{MAX}$	$\varepsilon_{MSQ}$	$\varepsilon_{\mathcal{L}^2}$
$8 \times 8 \times 8$	3.9e-5	7.4e-6	3.0e-5	3.9e-5	7.7e-6	3.1e-5
$16 \times 16 \times 16$	2.1e-6	3.8e-7	1.5e-6	2.0e-6	3.9e-7	1.5e-6
$32 \times 32 \times 32$	1.5e-7	3.1e-8	1.2e-7	1.5e-7	3.1e-8	1.2e-7
$64 \times 64 \times 64$	1.1e-8	2.2e-9	8.5e-9	1.1e-8	2.2e-9	8.5e-9

TABLE 14

Accuracy of the solution of the modified Helmholtz equation with  $\lambda = 1$ , by matching two boxes. The exact solution is  $u(x, y, z) = \exp\{-((x-0.4)^2 + (y-0.5)^2 + z^2)\}$  ((4.2),  $\alpha = 1$ ). Two cases are considered: (1) all the points participate in the computation of the global solution of the Dirichlet problem (regular global solution); (2) only half the points in the  $x$ -direction participate in the computation (sparse global solution).

$N_x \times N_y \times N_z$	Regular global solution			Sparse global solution		
	$\varepsilon_{MAX}$	$\varepsilon_{MSQ}$	$\varepsilon_{\mathcal{L}^2}$	$\varepsilon_{MAX}$	$\varepsilon_{MSQ}$	$\varepsilon_{\mathcal{L}^2}$
$8 \times 8 \times 8$	4.7e-4	7.9e-5	1.7e-4	4.7e-4	8.6e-5	1.8e-4
$16 \times 16 \times 16$	3.2e-5	5.6e-6	1.2e-5	3.2e-5	5.7e-6	1.2e-5
$32 \times 32 \times 32$	2.2e-6	4.0e-7	8.4e-7	2.2e-6	4.0e-7	8.4e-7
$64 \times 64 \times 64$	1.4e-7	2.7e-8	5.7e-8	1.4e-7	2.7e-8	5.7e-8

TABLE 15

$MAX$ ,  $MSQ$ , and  $\mathcal{L}^2$  errors for the solution of the Helmholtz equation by matching two boxes. The exact solution is  $u(x, y, z) = 1/\sqrt{(x+0.3)^2 + (y+0.4)^2 + (z+0.5)^2}$  (4.5) with  $k = 1$ . Two cases are considered: (1) all the points participate in the computation of the global solution of the Dirichlet problem (regular global solution); (2) only half the points in the  $x$ -direction are taken (sparse global solution).

$N_x \times N_y \times N_z$	regular global solution			sparse global solution		
	$\varepsilon_{MAX}$	$\varepsilon_{MSQ}$	$\varepsilon_{\mathcal{L}^2}$	$\varepsilon_{MAX}$	$\varepsilon_{MSQ}$	$\varepsilon_{\mathcal{L}^2}$
$8 \times 8 \times 8$	2.1e-4	2.3e-5	4.0e-5	1.4e-4	1.8e-5	3.3e-5
$16 \times 16 \times 16$	3.2e-6	3.5e-7	6.2e-7	3.0e-6	3.4e-7	6.0e-7
$32 \times 32 \times 32$	2.8e-7	1.8e-8	3.3e-8	2.7e-7	1.8e-8	3.2e-8
$64 \times 64 \times 64$	1.1e-8	1.1e-9	2.0e-9	7.5e-9	1.1e-9	1.9e-9

TABLE 16

$MAX$ ,  $MSQ$ , and  $\mathcal{L}^2$  errors for the solution of the Helmholtz equation, with  $k = 1$ , by matching two boxes. The exact solution is  $u(x, y, z) = \exp\{-((x-0.4)^2 + (y-0.5)^2 + z^2)\}$  ((4.6),  $\alpha = 1$ ). Two cases are considered: (1) all the points participate in the computation of the global solution of the Dirichlet problem (regular global solution); (2) only half the points in the  $x$ -direction are taken (sparse global solution).

$N_x \times N_y \times N_z$	regular global solution			sparse global solution		
	$\varepsilon_{MAX}$	$\varepsilon_{MSQ}$	$\varepsilon_{\mathcal{L}^2}$	$\varepsilon_{MAX}$	$\varepsilon_{MSQ}$	$\varepsilon_{\mathcal{L}^2}$
$8 \times 8 \times 8$	4.0e-4	6.8e-5	1.5e-4	4.0e-4	7.4e-5	1.6e-4
$16 \times 16 \times 16$	2.5e-5	4.6e-6	9.7e-6	2.5e-5	4.6e-6	9.8e-6
$32 \times 32 \times 32$	1.6e-6	3.1e-7	6.6e-7	1.6e-6	3.1e-7	6.6e-7
$64 \times 64 \times 64$	1.1e-7	2.1e-8	4.4e-8	1.1e-7	2.1e-8	4.3e-8

We observe that the sparse global solution reduces significantly the number of computations, while the accuracy of the algorithm is not degraded (see Tables 13, 14, 15, and 16).

We evaluate the gain of using the sparse global solution while comparing it to the usual (nonsparse) algorithm. In the simple case where eight boxes are matched, as in Figure 3.1, the number of operations in the global solution steps is

$$4O(2N^3 \log N) + 2O(4N^3 \log N) + O(8N^3 \log N) = 24O(N^3 \log N) = 24A,$$

while for the sparse global solution the cost is only

$$4O(N^3 \log N) + 2O(N^3 \log N) + O(N^3 \log N) = 7O(N^3 \log N) = 7A.$$

In general, if initially we have  $2^m$  subdivisions in each direction ( $2^{3m}$  subdomains), then the global matching step will take

$$2^{3(m-1)}24A + 2^{3(m-2)}8 \cdot 24A + \dots + 8^{m-1}24A = m2^{3m}3A$$

operations, where  $m$  is the number of matching steps for cubic boxes, compared to

$$2^{3(m-1)}7A + 2^{3(m-2)}7A + \dots + 7A = \frac{2^{3m}}{2^3 - 1}7A = 2^{3m}A,$$

which is  $3m$  times less than in the nonsparse case even when, as in the above computation, we neglect the  $\log N$  factor in the FFT.

**7. Summary and discussion.** The present algorithm can be treated as the main part of a domain decomposition based solution of the Helmholtz or the modified Helmholtz equation in a general 3-D subdomain. The domain is decomposed into nonoverlapping subdomains, mainly of cubic or parallelepiped shapes. Particular solutions are found in the subdomains. The computation of the solution and matching is not expensive for parallelepiped subdomains. Irregular subdomains can be treated similar to the Poisson solvers in general domains [8, 13, 14].

The main features of the algorithm are as follows:

1. The developed algorithms for the (modified) Helmholtz equation achieve high accuracy of  $10^{-7} - 10^{-8}$  for  $64 \times 64 \times 64$  points in the subdomains.
2. The algorithm takes  $O(N^3 \log N)$  operations, where  $N$  is the number of points in each direction. Similar to the 2-D case, the hierarchical matching procedure reduces the number of computations. Sparse computation of the global solution of the homogeneous equation significantly reduces the computation process.
3. The algorithm is expected to be applicable for parallel implementation as its previous 2-D version that was developed in [1, 2].

#### REFERENCES

- [1] A. AVERBUCH, E. BRAVERMAN, AND M. ISRAELI, *Parallel adaptive solution of a Poisson equation with multiwavelets*, SIAM J. Sci. Comput., 22 (2000), pp. 1053–1086.
- [2] A. AVERBUCH, E. BRAVERMAN, AND M. ISRAELI, *A new low communication parallel algorithm for elliptic partial differential equations*, in Parallel Computational Fluid Dynamics, North-Holland, Amsterdam, 2001, pp. 275–282.
- [3] A. AVERBUCH, M. ISRAELI, AND L. VOZOVoi, *On fast direct elliptic solver by modified Fourier method*, Numer. Algorithms, 15 (1997), pp. 287–313.

- [4] A. AVERBUCH, M. ISRAELI, AND L. VOZOVOI, *A fast Poisson solver of arbitrary order accuracy in rectangular regions*, SIAM J. Sci. Comput., 19 (1998), pp. 933–952.
- [5] E. BRAVERMAN, M. ISRAELI, A. AVERBUCH, AND L. VOZOVOI, *A fast 3-D Poisson solver of arbitrary order accuracy*, J. Comput. Phys., 144 (1998), pp. 109–136.
- [6] E. BRAVERMAN, M. ISRAELI, AND A. AVERBUCH, *A fast spectral solver for a 3D Helmholtz equation*, SIAM J. Sci. Comput., 20 (1999), pp. 2237–2260.
- [7] D. GOTTLIEB AND S. A. ORSZAG, *Numerical Analysis of Spectral Methods: Theory and Applications*, CBMS-NSF Regional Conf. Ser. in Appl. Math. 26, SIAM, Philadelphia, 1977.
- [8] A. GREENBAUM AND A. MAYO, *Rapid parallel evaluation of integrals in potential theory on general three-dimensional regions*, J. Comput. Phys., 145 (1998), pp. 731–742.
- [9] L. GREENGARD AND J.-Y. LEE, *A direct adaptive Poisson solver of arbitrary order accuracy*, J. Comput. Phys., 125 (1996), pp. 415–424.
- [10] L. GREENGARD AND V. ROKHLIN, *A fast algorithm for particle simulations*, J. Comput. Phys., 73 (1987), pp. 325–348.
- [11] M. ISRAELI, L. VOZOVOI, AND A. AVERBUCH, *Spectral multi-domain technique with Local Fourier basis*, J. Sci. Comput., 8 (1993), pp. 135–149.
- [12] E. G. KARNIADAKIS, M. ISRAELI, AND S. A. ORSZAG, *High-Order splitting methods for the incompressible Navier-Stokes equations*, J. Comput. Phys., 97 (1991), pp. 414–443.
- [13] A. MAYO AND A. GREENBAUM, *Fast parallel iterative solution of Poisson's and the biharmonic equations on irregular regions*, SIAM J. Sci. Statist. Comput., 13 (1992), pp. 101–118.
- [14] A. MCKENNEY, L. GREENGARD, AND A. MAYO, *A fast Poisson solver for complex geometries*, J. Comput. Phys., 118 (1995), pp. 348–355.
- [15] M. R. SCHUMACK, W. W. SCHULTZ, AND J. P. BOYD, *Spectral method solution of the Stokes equations on nonstaggered grids*, J. Comput. Phys., 94 (1991), pp. 30–58.
- [16] G. SKÖLERMO, *A Fourier method for numerical solution of Poisson's equation*, Math. Comput., 29 (1975), pp. 697–711.
- [17] L. VOZOVOI, M. ISRAELI, AND A. AVERBUCH, *Spectral multi-domain technique with local Fourier basis II: Decomposition into cells*, J. Sci. Comput., 9 (1994), pp. 311–326.