SIAM J. SCI. COMPUT.
Vol. 40, No. 3, pp. A1274–A1300

AN ADAPTIVE FAST GAUSS TRANSFORM IN TWO DIMENSIONS*

JUN WANG† AND LESLIE GREENGARD‡

Abstract. A variety of problems in computational physics and engineering require the convolution of the heat kernel (a Gaussian) with either discrete sources, densities supported on boundaries, or continuous volume distributions. We present a unified fast Gauss transform for this purpose in two dimensions, making use of an adaptive quad-tree discretization on a unit square which is assumed to contain all sources. Our implementation permits either free-space or periodic boundary conditions to be imposed, and is efficient for any choice of variance in the Gaussian.

Key words. fast Gauss transform, heat equation, adaptive mesh refinement

AMS subject classifications. 31A10, 35K10, 65R10, 65Y20

DOI. 10.1137/17M1159865

1. Introduction.

A variety of problems in applied physics and engineering involve the solution of the heat equation

$$\begin{aligned} u_t(\mathbf{x},t) &= \Delta u(\mathbf{x},t) + F(\mathbf{x},t), \\ u(\mathbf{x},0) &= f(\mathbf{x}) \end{aligned} \tag{1.1}$$

for $t > 0$, in an interior or exterior domain Ω, subject to suitable conditions on its boundary $\Gamma = \partial\Omega$. For simplicity, we will assume that these take the form of either Neumann conditions,

$$\frac{\partial u}{\partial n}(\mathbf{x},t) = g(\mathbf{x},t) \quad \text{for} \quad \mathbf{x} \in \Gamma, \tag{1.2}$$

Dirichlet conditions,

$$u(\mathbf{x},t) = h(\mathbf{x},t) \quad \text{for} \quad \mathbf{x} \in \Gamma, \tag{1.3}$$

or periodic boundary conditions, with Ω the unit square.

In the absence of physical boundaries, the equations (1.1) are well-posed in free space (under mild conditions on the behavior of u, f, and F at infinity) without auxiliary conditions. Moreover, assuming $F(\mathbf{x},t)$ and $f(\mathbf{x})$ are compactly supported in the region Ω, the solution to (1.1) can be expressed at the next time step, $t = \Delta t$, in closed form as

$$u(\mathbf{x},\Delta t) = J[f](\mathbf{x},\Delta t) + V[F](\mathbf{x},\Delta t) \tag{1.4}$$

*Submitted to the journal's Methods and Algorithms for Scientific Computing section December 5, 2017; accepted for publication (in revised form) February 20, 2018; published electronically May 3, 2018.

http://www.siam.org/journals/sisc/40-3/M115986.html

Funding: The second author's work was supported in part by the Applied Mathematical Sciences Program of the U.S. Department of Energy under contract DEFGO288ER25053 and by the RiskEcon Lab for Decision Metrics, Courant Institute.

†Courant Institute of Mathematical Sciences, New York University, New York, NY 10012. Current address: Flatiron Institute, Simons Foundation, New York, NY 10010 (junwang@flatironinstitute.org).

‡Courant Institute of Mathematical Sciences, New York University, New York, NY 10012, and Flatiron Institute, Simons Foundation, New York, NY 10010 (greengard@cims.nyu.edu).

with

$$J[f](\mathbf{x},\Delta t) = \int_\Omega G(\mathbf{x}-\mathbf{y},\Delta t) f(\mathbf{y})\, d\mathbf{y}, \tag{1.5}$$

$$V[F](\mathbf{x},\Delta t) = \int_0^{\Delta t}\int_\Omega G(\mathbf{x}-\mathbf{y},\Delta t-\tau) F(\mathbf{y},\tau)\, d\mathbf{y} d\tau\,. \tag{1.6}$$

Here,

$$G(\mathbf{x},t) = \frac{e^{-\|\mathbf{x}\|^2/4t}}{(4\pi t)^{d/2}}$$

is the fundamental solution of the heat equation in d dimensions. The functions $J[f]$ and $V[F]$ are referred to as *initial* (heat) potentials and *domain* (heat) potentials, respectively. In the remainder of this paper, we assume $d = 2$.

For the Neumann problem (1.1), (1.2), the classical representation [17, 23] takes the form

$$u(\mathbf{x},\Delta t) = J[f](\mathbf{x},\Delta t) + V[F](\mathbf{x},\Delta t) + S[\sigma](\mathbf{x},\Delta t), \tag{1.7}$$

where

$$S[\sigma](\mathbf{x},\Delta t) = \int_0^{\Delta t}\int_\Gamma G(\mathbf{x}-\mathbf{y},\Delta t-\tau)\sigma(\mathbf{y},\tau)\; ds_{\mathbf{y}} d\tau \tag{1.8}$$

is a *single layer* (heat) potential. For the Dirichlet problem (1.1), (1.3), the classical representation takes the form

$$u(\mathbf{x},\Delta t) = J[f](\mathbf{x},\Delta t) + V[F](\mathbf{x},\Delta t) + D[\mu](\mathbf{x},\Delta t), \tag{1.9}$$

where

$$D[\mu](\mathbf{x},\Delta t) = \int_0^{\Delta t}\int_\Gamma \frac{\partial G}{\partial n_{\mathbf{y}}}(\mathbf{x}-\mathbf{y},\Delta t-\tau)\mu(\mathbf{y},\tau)\; ds_{\mathbf{y}} d\tau \tag{1.10}$$

is a *double layer* (heat) potential. Here, $\frac{\partial}{\partial n_{\mathbf{y}}}$ denotes the derivative in the outward normal direction at the boundary point $\mathbf{y}$. The only unknowns in the representations (1.7), (1.9) are the scalar densities σ and μ supported on Γ. These are obtained by solving integral equations to enforce the desired boundary conditions [17, 23]. Once σ or μ is known, (1.7), (1.9) can be used to evaluate the solution at time $t = \Delta t$. This yields a one-step marching method for the heat equation that is both stable and robust (see, for example, [1, 5, 7, 8, 13, 18, 21, 30, 31]).

For our present purposes, we assume that σ and μ are given. We assume also that a suitable M-stage quadrature has been applied to $V[F](\mathbf{x},\Delta t)$, $S[\sigma](\mathbf{x},\Delta t)$, and $D[\sigma](\mathbf{x},\Delta t)$ with respect to the time variable, yielding

$$\begin{aligned}
V[F](\mathbf{x},\Delta t) &\approx \sum_{j=1}^M w_{V,j}\int_\Omega G(\mathbf{x}-\mathbf{y},\Delta t-\tau_j)F(\mathbf{y},\tau_j)\, d\mathbf{y}\,,\\
S[\sigma](\mathbf{x},\Delta t) &\approx \sum_{j=1}^M w_{S,j}\int_\Gamma G(\mathbf{x}-\mathbf{y},\Delta t-\tau_j)\sigma(\mathbf{y},\tau_j)\, ds_{\mathbf{y}}\,,\\
D[\mu](\mathbf{x},\Delta t) &\approx \sum_{j=1}^M w_{D,j}\int_\Gamma \frac{\partial G}{\partial n_{\mathbf{y}}}(\mathbf{x}-\mathbf{y},\Delta t-\tau_j)\mu(\mathbf{y},\tau_j)\, ds_{\mathbf{y}}\,,
\end{aligned} \tag{1.11}$$

where $w_{V,j}$, $w_{S,j}$, $w_{D,j}$ are known quadrature weights.

Thus, the computational burden of time-marching (that is, evaluating the various heat potentials) is dominated by the volume integrals

$$\mathcal{V}[f](\mathbf{x}) = \int_\Omega e^{-\frac{|\mathbf{x}-\mathbf{y}|^2}{\delta}} \tilde{f}(\mathbf{y})\, d\mathbf{y} \tag{1.12}$$

and the boundary integrals

$$\begin{aligned} \mathcal{S}[\sigma](\mathbf{x}) &= \int_\Gamma e^{-\frac{|\mathbf{x}-\mathbf{y}(s)|^2}{\delta}} \tilde{\sigma}(\mathbf{y}(s))\, ds_{\mathbf{y}}\,, \\ \mathcal{D}[\mu](\mathbf{x}) &= \int_\Gamma \frac{\partial}{\partial n_{\mathbf{y}(s)}} e^{-\frac{|\mathbf{x}-\mathbf{y}(s)|^2}{\delta}} \tilde{\mu}(\mathbf{y}(s))\, ds_{\mathbf{y}} \end{aligned} \tag{1.13}$$

for various values of δ and given functions $\tilde{f}, \tilde{\sigma}, \tilde{\mu}$. Evaluating these integrals accurately and efficiently is the focus of the present paper.

DEFINITION 1.1. *The integrals* (1.12) *and* (1.13) *will be referred to as volume and boundary Gauss transforms, respectively.*

DEFINITION 1.2. *By the* discrete Gauss transform *(DGT), we mean the evaluation of the Gaussian "potential" at M points $\{\mathbf{x}_i\}$ due to N sources located at $\{\mathbf{y}_j\}$ of strength $\{q_j\}$:*

$$F(\mathbf{x}_i) = \sum_{j=1}^{N} q_j \cdot e^{-\frac{|\mathbf{x}_i-\mathbf{y}_j|^2}{\delta}} \quad \textit{for } i = 1, \ldots, M. \tag{1.14}$$

A variety of algorithms have been developed for the rapid evaluation of sums of the form (1.14), such as the fast Gauss transform (FGT) [15] (see also [16, 24, 25, 28]). While the naive DGT requires $O(MN)$ work, the FGT permits the evaluation of the values $\{F(\mathbf{x}_i)\}$ using only $O(M + N)$ work, independent of δ. High-dimensional versions of the FGT are of interest in statistical and machine learning applications (see, for example, [9]), but we are concerned here with physical modeling, where the ambient dimension is generally less than or equal to three.

Here, we seek to develop a robust version of the FGT that is fully adaptive, insensitive to δ, and able to compute transforms with discrete sources, volume sources, and densities supported on boundaries (see Figure 1.1). Some notable prior work on continuous (volume) fast transforms includes [27], which describes a triangulation-based adaptive refinement method, and [31], which makes use of a high-order, adaptive, quad-tree–based discretization. As in [31], our approach relies on an adaptive quad-tree with high-order Chebyshev grids on leaf nodes, but we carry out a modified version of the FGT on the quad-tree itself. This requires a somewhat more complicated implementation, following that of the hierarchical fast multipole method (FMM) [14]. We will assume that the adaptive quad-tree satisfies the following level-restriction condition: while variable levels of refinement are allowed in distinct subregions of the computational domain, two leaf nodes which share a boundary point must be no more than one refinement level apart (see Figure 2.1). Trees satisfying this condition are sometimes called "balanced," but we will use the term "level-restricted" since it is more standard in the FMM/fast algorithm literature [2, 10, 12, 22].

Our hierarchical FGT permits the inclusion of boundary Gauss transforms and discrete sources at the same time. We should note that adaptive FGT variants using an FMM data structure have been constructed previously, such as in [20], but for the

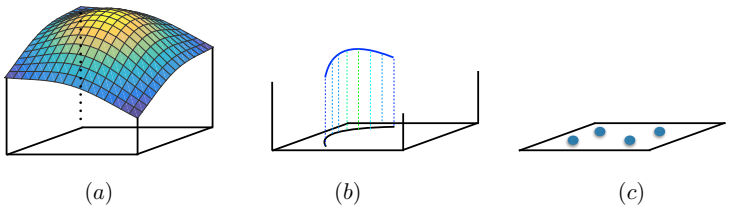

FIG. 1.1. *We seek a version of the FGT that is able to handle volume sources* (a)*, densities supported on boundaries* (b)*, and point sources* (c).

discrete setting only, where small values of δ pose no additional quadrature challenges. We also introduce new error estimates that are relevant for the hierarchical processing.

The paper is organized as follows. In sections 2 and 3, we review our adaptive discretization strategy and the analytic machinery on which the FGT is based. In section 4, we describe the new FGT itself, focusing primarily on the volume integral case (1.12), with a brief discussion of the modifications needed for (1.14) and (1.13). We also discuss the incorporation of periodic boundary conditions. Section 5 illustrates the performance of the algorithm with several numerical examples, and section 6 contains some concluding remarks.

2. Data structure. In the classical FGT [15], aimed at the computation of (1.14), where the sources are discrete, a uniform grid is superimposed on the computational domain, with a box size of dimension $(r\sqrt{\delta})^d$, where $r \approx 1$ (Figure 2.1, left). Because of the exponential decay of the Gaussian, it is easy to see that only a finite range of nearby boxes needs to be considered to achieve any desired precision. That is, the effect due to sources in B at targets that are at least m boxes away is of the order $O(e^{-m^2r^2})$. Since the field due to sources in any box B is efficiently represented by a suitable Hermite expansion (see section 3), it is straightforward to develop an algorithm of complexity $O(N + M)$, where N is the number of discrete sources and M is the number of targets. The FGT is easily modified to allow for adaptivity. One simply needs to sort the source and target points on the uniform grid while ignoring empty boxes and keeping track of the relevant neighbors for each box. The total storage is then of the order $O(N + M)$ as well. This can be accomplished, for example, with an adaptive quad-tree that is refined uniformly to a level where the box size is approximately $(r\sqrt{\delta})^d$, pruning empty boxes on the way.

Such a strategy fails for volume integrals of the form (1.12), since there are no empty boxes. Instead, we will assume that the right-hand side (the function f in (1.12)) is specified on a level-restricted quad-tree. These data structures have been shown to be extremely effective for elliptic volume integrals [2, 6, 10, 19, 12, 22]. For the sake of simplicity, we assume that the source distribution f in (1.12) is supported in the unit box D, centered at the origin. Following the discussion of [10], we assume that superimposed on D is a hierarchy of refinements (a quad-tree). Grid level 0 is defined to be D itself, with grid level $l + 1$ obtained recursively by subdividing each box at level l into four equal parts. If B is a fixed box at level l, the four boxes at level $l + 1$ obtained by its subdivision will be referred to as its children. In a level-restricted adaptive tree, we do not assume that the same number of levels is used in all subdomains of D. We do, however, require that two leaf nodes which share a boundary point must be no more than one refinement level apart (Figure 2.1, right).

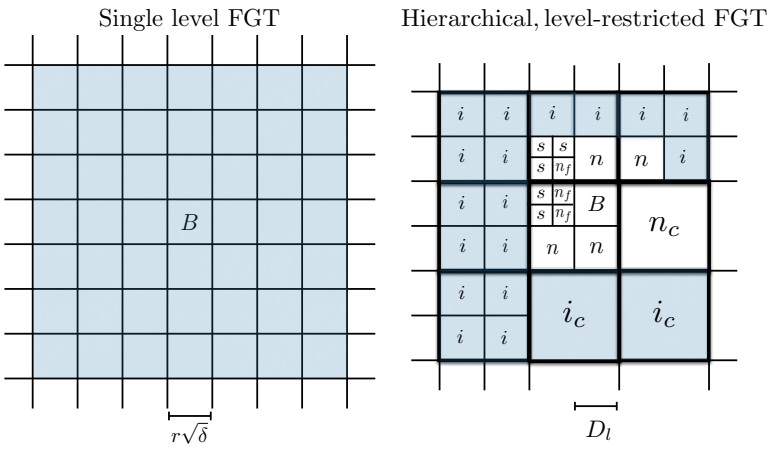

FIG. 2.1. *The original FGT data structure (left) and a level-restricted quad-tree data structure (right). In the original FGT, the interaction region (shaded blue-gray) consists of boxes which are close enough to B so that the Gaussian field induced by the sources in B is significant. (Outside the shaded region, the field is exponentially small and can be ignored for any fixed precision.) In the quad-tree, multiple types of interactions must be accounted for, and are described in detail in section* 4. *The interaction list for a typical node B consists of the boxes labeled i, while the near neighbors at the same refinement level are labeled n. In a level-restricted tree, for a leaf node B, there can be near neighbors at one coarser level, labeled n_c, or at one finer level, labeled n_f as well. The boxes labeled s are separated from B but at a finer level (see Definition* 4.2*), while the boxes labeled i_c are separated from B but at a coarser level (see Definitions* 4.2 *and* 4.3*).*

On each leaf node B, we assume that we are given f on a $k \times k$ tensor product grid—either regularly spaced or at Chebyshev nodes. We then construct a $k-1$-degree polynomial approximation to f on B of the form

$$f_B(y_1, y_2) \approx \sum_{j=1}^{N_k} c_B(j)\, b_j(y_1, y_2), \tag{2.1}$$

where $N_k = \frac{k(k+1)}{2}$ is the number of basis functions needed for kth order accuracy, and the basis functions b_j are assumed to be scaled to the relevant box size and centered on the box center. The coefficient vector is defined to be $\vec{c}_B = (c_B(1), \ldots, c_B(N_k))$.

For fourth or sixth order accuracy, polynomial approximation using a regular grid is fairly well-conditioned, and we assume that f is given on a regular grid. We use as basis functions

$$\{y_1^l y_2^m |\ l, m \geq 0, l + m \leq k - 1\}.$$

This corresponds to keeping polynomials of *total degree* less than or equal to $k - 1$. It is easy to see that this leads to a kth order accurate approximation as the mesh is refined. Other options, such as the full tensor product basis ($\max(l, m) \leq k-1$) or the Euclidean truncation ($l^2+m^2 \leq (k-1)^2$), also lead to kth order accurate schemes. For an interesting discussion of the effect of this choice as k and the ambient dimension increase, see Trefethen's recent paper [29].

If we let $\vec{f_B} \in \mathbf{R}^{k^2}$ denote the given function values (in standard ordering), then the coefficient vector $\vec{c_B}$ can be computed as the solution of a least squares problem (interpolating the desired data $\vec{f_B}$ at the corresponding points). The solution operator

for this least squares task is denoted by $\mathcal{P} \in \mathbf{R}^{N_k \times k^2}$, so that

$$\vec{c}_B = \mathcal{P}\, \vec{f}_B.$$

$\mathcal{P}$ can be precomputed and stored (say, using QR factorization). For eighth (or higher) order accuracy, polynomial interpolation at equispaced nodes becomes increasingly unstable (the so-called Runge phenomenon). In such cases, we assume that f is given on a $k \times k$ tensor product Chebyshev grid and use as basis functions

$$\{T_l(y_1)T_m(y_2)|\ l, m \geq 0, l + m \leq k - 1\},$$

where $T_l(x)$ denotes the (suitably scaled) Chebyshev polynomial of degree l. The coefficients of the tensor product Chebyshev expansion can be computed efficiently using the fast cosine transform [4].

Remark 1. We have chosen above to use polynomials which satisfy the *total degree condition*: $l + m \leq k - 1$. It is easy to see that this leads to a kth order accurate approximation as the mesh is refined. Other options, such as the full tensor product basis ($\max(l, m) \leq k - 1$) or the Euclidean truncation ($l^2 + m^2 \leq (k - 1)^2$), also lead to kth order accurate schemes. In an interesting, recent paper [29], Trefethen discusses these options in detail and proves that Euclidean truncation is, in some sense, optimal—an effect whose advantage becomes more and more pronounced as k and the ambient dimension increase.

In order to develop a fast algorithm for the various kinds of source distributions shown in Figure 1.1, we will make use of efficient far field and local representations of the induced field.

3. Analytical apparatus. Following the discussion in [15], we define the Hermite functions $h_n(x)$ by

$$h_n(x) = (-1)^n D^n e^{-x^2}, \quad x \in \mathbb{R},$$

where $D = d/dx$. They satisfy the relation

$$e^{-(x-y)^2/\delta} = \sum_{n=0}^{\infty} \frac{1}{n!} \left(\frac{y - c}{\sqrt{\delta}} \right)^n h_n \left(\frac{x - c}{\sqrt{\delta}} \right), \tag{3.1}$$

where $y_0 \in \mathbb{R}$ and $\delta > 0$. This formula can be interpreted in two ways: (1) as a Hermite expansion of the Gaussian at the target point x centered at c due to a source at y, and (2) as a Taylor series for the Gaussian at the target location y centered at c due to a source at x.

It will be convenient to use multi-index notation. In two dimensions, a multi-index is a pair of nonnegative integers $\alpha = (\alpha_1, \alpha_2)$ with which, for any $\mathbf{x} = (x_1, x_2) \in \mathbf{R}^2$, we define

$$|\alpha| = \alpha_1 + \alpha_2, \quad \alpha! = \alpha_1!\alpha_2!, \quad \mathbf{x}^\alpha = x_1^{\alpha_1} x_2^{\alpha_2}, \quad D^\alpha = \partial_{x_1}^{\alpha_1} \partial_{x_2}^{\alpha_2}.$$

If p is an integer, we say $\alpha \geq p$ if $\alpha_1, \alpha_2 \geq p$. Multidimensional Hermite functions are defined by

$$h_\alpha(\mathbf{x}) = h_{\alpha_1}(x_1) h_{\alpha_2}(x_2),$$

and the analogue of (3.1) is

$$e^{-|\mathbf{x}-\mathbf{y}|^2/\delta} = \frac{1}{\alpha!} \sum_{\alpha \geq 0} \left(\frac{\mathbf{y} - \mathbf{c}}{\sqrt{\delta}} \right)^\alpha h_\alpha \left(\frac{\mathbf{x} - \mathbf{c}}{\sqrt{\delta}} \right) \tag{3.2}$$

for a center $\mathbf{c}$.

3.1. Hermite expansions and translation operators. We turn now to the analytical apparatus needed in the FGT algorithm. The first lemma describes how to transform the field due to a volume source distribution and a collection of discrete Gaussians into a Hermite expansion about the center of box B in which they are supported.

LEMMA 3.1. *Let B be a box with center $\mathbf{s}_B$ and side length $r\sqrt{\delta}$, and let the Gaussian field $\phi(\mathbf{x})$ be defined by*

$$\phi(\mathbf{x}) = \int_B e^{-\frac{|\mathbf{x}-\mathbf{y}|^2}{\delta}} f(\mathbf{y})\, d\mathbf{y} \; + \sum_{j=1}^{N_s} q_j e^{-\frac{|\mathbf{x}-\mathbf{y}_j|^2}{\delta}}, \tag{3.3}$$

where the $\{\mathbf{y}_j\}$ lie in B. Then,

$$\phi(\mathbf{x}) = \sum_{\alpha \geq 0} A_\alpha h_\alpha \left(\frac{\mathbf{x}-\mathbf{s}_B}{\sqrt{\delta}} \right), \tag{3.4}$$

where

$$A_\alpha = \frac{1}{\alpha!} \left(\int_B \left(\frac{\mathbf{y}-\mathbf{s}_B}{\sqrt{\delta}} \right)^\alpha f(\mathbf{y})\, d\mathbf{y} + \sum_{j=1}^{N_s} \left(\frac{\mathbf{y}_j-\mathbf{s}_B}{\sqrt{\delta}} \right)^\alpha q_j \right). \tag{3.5}$$

The error in truncating the Hermite expansion with p^2 terms is given by

$$|E_H(p)| = \left| \sum_{\alpha \geq p} A_\alpha h_\alpha \left(\frac{\mathbf{x}-\mathbf{s}_B}{\sqrt{\delta}} \right) \right| \leq K^2 Q_B (2S_r(p) + T_r(p)) T_r(p), \tag{3.6}$$

where

$$Q_B = \int_B |f(\mathbf{y})| d\mathbf{y} + \sum_{j=1}^{N_s} |q_j|, \tag{3.7}$$

$$S_r(p) = \sum_{n=0}^{p} \frac{r^n}{\sqrt{n!}}, \quad T_r(p) = \sum_{n=p}^{\infty} \frac{r^n}{\sqrt{n!}}, \tag{3.8}$$

and $K < 1.09$.

Proof. The error estimate relies on Cramer's inequality, which takes the form

$$\frac{1}{\alpha!} |h_\alpha(\mathbf{x})| \leq K^2 2^{|\alpha|/2} \frac{1}{\sqrt{\alpha!}} e^{-|\mathbf{x}|^2/2} \tag{3.9}$$

in two dimensions, where $K < 1.09$, and on the fact that

$$\left| \frac{\mathbf{y}-\mathbf{s}_B}{\sqrt{\delta}} \right| \leq \frac{\sqrt{2}}{2} r \tag{3.10}$$

for $\mathbf{y}$ in B. The desired result follows from integration over the domain B and summation over the discrete sources. ∎

Note that the Hermite expansion converges extremely rapidly for $r < 1$. For larger r, it still converges but requires larger values of p. (See [3, 20, 26, 32] for further discussion of error estimates.) In the original FGT, setting $r \approx 1$ is a sensible choice, since a modest value of p is sufficient and the number of boxes within the interaction region (where the Gaussian field is not vanishingly small) is modest as well. The interaction region for a box B is the shaded area on the left in Figure 2.1. Also notice that the estimate is *uniform* with respect to the target. In the original FGT, this is necessary since the Hermite expansion is evaluated at all relevant locations. In the hierarchical FGT, however, the Gaussian field due to a Hermite expansion is evaluated only for boxes that are "well-separated" (the boxes labeled by i on the right-hand side of Figure 2.1). Moreover, we will compute such interactions at every level of the quad-tree, so that the boxes can be of arbitrary size. Fortunately, once boxes are separated by a distance $R\sqrt{\delta}$, their interactions can be ignored with an error of the order $O(e^{-R^2})$, limiting the size of the expansions (see Figure 3.1).

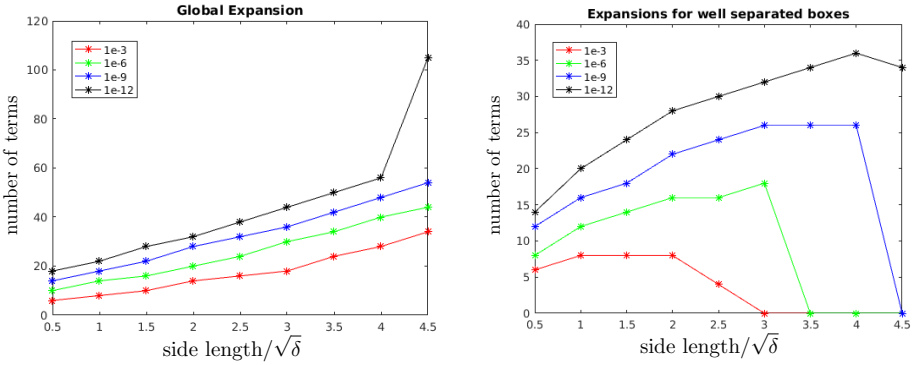

FIG. 3.1. *(Left) plot of the number of terms p needed in a Hermite expansion as a function of the box size, when used uniformly in the plane, including the near field. (Right) plot of the number of terms needed when used only in the far field.*

In the full algorithm, a more refined estimate that makes use of the separation criterion will be useful. We have the following lemma.

LEMMA 3.2. *Let B be a box with center $\mathbf{s}_B$ and side length $r\sqrt{\delta}$, and let C be a box with center $\mathbf{t}_C$ and side length $r\sqrt{\delta}$ with $x \in C$. Assuming the distance between B and C is at least $r\sqrt{\delta}$, the Gaussian field defined by* (3.3) *and its Hermite expansion* (3.4) *satisfy the error bound*

$$|E_H(p)| = \left| \sum_{\alpha \geq p} A_\alpha h_\alpha \left(\frac{\mathbf{x} - \mathbf{s}_B}{\sqrt{\delta}} \right) \right| = K^2 Q_B e^{-\frac{9}{8} r^2} (2S_r(p) + T_r(p)) T_r(p), \tag{3.11}$$

where Q_B is given by (3.7)*, and $S_r(p)$, $T_r(p)$ are given by* (3.8).

Proof. The proof follows the same outline as that of Lemma 3.1. In this case, after applying Cramer's inequality (3.9), we make the additional observation that $|\frac{\mathbf{x}-\mathbf{s}_B}{\sqrt{\delta}}| \geq \frac{3}{2} r$, which contributes to the exponential decay in r. ∎

The next lemma describes the conversion of a Hermite expansion about $\mathbf{s}_B$ into a Taylor expansion about $\mathbf{t}_C$.

LEMMA 3.3. *Let*

$$\phi(\mathbf{x}) = \sum_{\alpha \geq 0} A_\alpha h_\alpha \left(\frac{\mathbf{x} - \mathbf{s}_B}{\sqrt{\delta}} \right) \tag{3.12}$$

denote the Hermite expansion of a Gaussian field induced by a source distribution in a box B with center $\mathbf{s}_B$ and side length $r\sqrt{\delta}$. Then $\phi(\mathbf{x})$ has the following Taylor expansion about the center $\mathbf{t}_C$ of box C with side length $r\sqrt{\delta}$:

$$\phi(\mathbf{x}) = \sum_{\beta \geq 0} B_\beta \left(\frac{\mathbf{x} - \mathbf{t}_C}{\sqrt{\delta}} \right)^\beta. \tag{3.13}$$

The coefficients are given by

$$B_\beta = \frac{(-1)^{|\beta|}}{\beta!} \sum_{\alpha \geq 0} A_\alpha h_{\alpha+\beta} \left(\frac{\mathbf{s}_B - \mathbf{t}_C}{\sqrt{\delta}} \right). \tag{3.14}$$

Assuming that the distance between boxes B and C is at least $r\sqrt{\delta}$, the error $E_T(p)$ in truncating the Taylor series after p^2 terms satisfies

$$|E_T(p)| = \left| \sum_{\alpha \geq p} A_\alpha h_\alpha \left(\frac{\mathbf{x} - \mathbf{s}_B}{\sqrt{\delta}} \right) \right| \leq K^2 Q_B e^{-\frac{9}{8} r^2} (2S_r(p) + T_r(p)) T_r(p), \tag{3.15}$$

where Q_B is given by (3.7)*, and $S_r(p)$, $T_r(p)$ are given by* (3.8)*.*

Proof. This result follows, again, from the standard error estimate in [3, 26, 32], with one modification: the exponential term in Cramer's inequality can be bounded by $e^{-\frac{9}{8} r^2}$ instead of 1. ∎

In practice, we need a variant of Lemma 3.3, in which the Hermite expansion is truncated before being converted to a Taylor expansion.

LEMMA 3.4. *Let*

$$\phi(\mathbf{x}) = \sum_{\alpha \leq p} A_\alpha h_\alpha \left(\frac{\mathbf{x} - \mathbf{s}_B}{\sqrt{\delta}} \right) \tag{3.16}$$

denote a truncated Hermite expansion corresponding to the Gaussian field induced by a source distribution in a box B with center $\mathbf{s}_B$ and side length $r\sqrt{\delta}$. The induced Taylor series in a box C with center $\mathbf{t}_C$ and side length $r\sqrt{\delta}$ is given by

$$\phi(\mathbf{x}) = \sum_{\beta \geq 0} C_\beta \left(\frac{\mathbf{x} - \mathbf{t}_C}{\sqrt{\delta}} \right)^\beta, \tag{3.17}$$

with coefficients

$$C_\beta = \frac{(-1)^{|\beta|}}{\beta!} \sum_{\alpha \leq p} A_\alpha h_{\alpha+\beta} \left(\frac{\mathbf{s}_B - \mathbf{t}_C}{\sqrt{\delta}} \right). \tag{3.18}$$

Assuming that the distance between boxes B and C is at least $r\sqrt{\delta}$, the error $E_{HT}(p)$ in truncating the Taylor series after p^2 terms satisfies the bound

$$|E_{HT}(p)| = \left| \sum_{\alpha \geq p} A_\alpha h_\alpha \left(\frac{\mathbf{x} - \mathbf{s}_B}{\sqrt{\delta}} \right) \right| \leq K^2 Q_B e^{-2r^2} (2S_r(p) + T_r(p)) T_r(p) S_r^2(p), \tag{3.19}$$

where Q_B is given by (3.7)*, and $S_r(p)$, $T_r(p)$ are given by* (3.8)*.*

Proof. The result is a straightforward application of the triangle inequality and Lemma 3.3. ∎

Note that the total error in using both a Hermite and a local expansion consists of two contributions: the first comes from truncating the Hermite expansion, given by (3.6), while the second comes from truncating the local expansion according to (3.19).

For the hierarchical FGT, we will also need to propagate Hermite and Taylor expansions between levels of the quad-tree. The following two lemmas provide the needed analytical tools. Lemma 3.5 describes a formula for shifting the center of a Hermite expansion, and Lemma 3.6 describes one for shifting the center of a Taylor expansion. The derivation is straightforward [20].

LEMMA 3.5. *Let a Gaussian field be given by the Hermite expansion*

$$\phi(\mathbf{x}) = \sum_{\alpha \geq 0} A_\alpha h_\alpha \left(\frac{\mathbf{x} - \mathbf{s}_B}{\sqrt{\delta}} \right), \tag{3.20}$$

about a center $\mathbf{s}_B$*, and let* $\mathbf{s}_C$ *denoted a shifted expansion center. Then,*

$$\phi(\mathbf{x}) = \sum_{\beta \geq 0} B_\beta h_\beta \left(\frac{\mathbf{x} - \mathbf{s}_C}{\sqrt{\delta}} \right), \tag{3.21}$$

where the coefficients are given by

$$B_\beta = \sum_{\alpha \leq \beta} \frac{\alpha!}{\beta!} \binom{\beta}{\alpha} \left(\frac{\mathbf{s}_B - \mathbf{s}_C}{\sqrt{\delta}} \right)^{\beta - \alpha} A_\alpha. \tag{3.22}$$

LEMMA 3.6. *Let* $\mathbf{t}_B \in \mathbb{R}^2$*, and let* $\{C_\alpha\}$ *denote the expansion coefficients for a truncated Taylor series with* p^2 *terms. Letting* $\mathbf{t}_C \in \mathbb{R}^2$ *be a shifted expansion center, we have*

$$\sum_{\alpha \leq p} C_\alpha \left(\frac{\mathbf{x} - \mathbf{t}_B}{\sqrt{\delta}} \right)^\alpha = \sum_{\beta \leq p} C'_\beta \left(\frac{\mathbf{x} - \mathbf{t}_C}{\sqrt{\delta}} \right)^\beta, \tag{3.23}$$

where

$$C'_\beta = \sum_{\beta \leq \alpha \leq p} C_\alpha \binom{\alpha}{\beta} \left(\frac{\mathbf{t}_C - \mathbf{t}_B}{\sqrt{\delta}} \right)^{\alpha - \beta}. \tag{3.24}$$

3.2. Local interactions. In the previous section, we summarized the analytical machinery needed for the fast evaluation of Gaussian fields with well-separated sources and targets. Before providing a formal description of the full algorithm in the next section, it remains to consider the computation of local interactions between neighboring boxes at the level of leaf nodes. For point sources, this is done by direct evaluation. We concentrate in this section on domain integrals and defer a discussion of densities supported on boundaries to section 4.2.

Thus, suppose B is a leaf node—that is, a box at level l of the tree hierarchy on which a $k \times k$ tensor product grid of function values has been specified. Let r_l denote the side length of B, so that its area is $r_l \times r_l$. Consider now a target point $\mathbf{t}$, which lies in either B, a neighboring box of B at the same refinement level, or a coarse

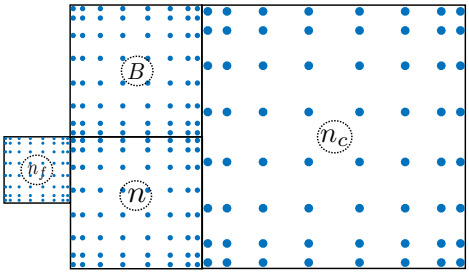

FIG. 3.2. *For a leaf node B, there are three types of possible local interactions: interaction with a colleague (see Definition 4.2), and interactions with fine and coarse neighbors—one level finer or one level coarser, respectively. The grid shown corresponds to an eighth order accurate tensor product Chebyshev discretization.*

or fine neighbor for B, which can be at most one refinement level apart (see Figure 3.2). Because of the translation invariance of the kernel, a simple counting argument shows that there are at most $k \times k \times 9$ possible targets at the same level and at most $k \times k \times 12$ possible targets in neighbors at either a coarser or a finer level. Recalling that the source distribution f on B is given by (2.1), the Gaussian field induced at $\mathbf{t}$ by f_B can be approximated by

$$\psi_B(\mathbf{t}) = \sum_{n=1}^{N_k} c_B(n) G(\mathbf{t}, n), \tag{3.25}$$

with

$$G(\mathbf{t}, n) = \int_B e^{-\frac{(t_1-y_1)^2}{\delta}} e^{-\frac{(t_2-y_2)^2}{\delta}} b_n(y_1, y_2)\, dy_1 dy_2, \tag{3.26}$$

where (t_1, t_2), (y_1, y_2) denote the coordinates of the target $\mathbf{t}$ and source with respect to the center of box B. Once the values $\{G(\mathbf{t}, n)\}$ have been tabulated for all possible target locations and all basis functions, *all* local interactions can be computed directly from (3.25), with some careful bookkeeping.

Assuming $b_n(y_1, y_2) = p_{n_1}(y_1) p_{n_2}(y_2)$, and using the fact that the Gaussian itself allows for separation of variables, the formula for $G(\mathbf{t}, n)$ can be written in the form

$$G(\mathbf{t}, n) = \int_{-r_l/2}^{r_l/2} e^{-\frac{(t_1-y_1)^2}{\delta}} p_{n_1}(y_1) dy_1 \cdot \int_{-r_l/2}^{r_l/2} e^{-\frac{(t_2-y_2)^2}{\delta}} p_{n_2}(y_2) dy_2. \tag{3.27}$$

Thus, for colleagues (see Definition 4.2), it is straightforward to check that there are at most $3k$ possible relative target locations $(t_i - y_i)$ and at most k basis functions $p_{n_i}(y_i)$, which are either monomials or scaled Chebyshev polynomials. These $3k^2$ numbers can be computed in milliseconds on a single core. For coarse or fine neighbors, there are at most $4k$ possible relative target locations $(t_i - y_i)$, so that these tables involving $4k^2$ numbers can be generated in milliseconds as well. Finally, we note that such tables must be generated for each refinement level that contains a leaf node.

4. Fast Gauss transform algorithm. We now describe a version of the FGT that uses a level-restricted quad-tree, closely following the discussion in [10]. Since the Gaussian kernel $e^{-\|\mathbf{x}-\mathbf{y}\|^2/\delta}$ is rapidly decaying, we will ignore interactions beyond a distance where they can be considered negligible, according to a user-defined precision

ϵ. That is, we define a cut-off parameter r_c so that $e^{-|\mathbf{x}-\mathbf{y}|^2/\delta} \leq \epsilon$, when $\|\mathbf{x}-\mathbf{y}\| \geq r_c\sqrt{\delta}$. Clearly, if a source box has side length greater than or equal to $r_c\sqrt{\delta}$, its contribution to well-separated boxes is negligible. We will also make use of the following definitions.

DEFINITION 4.1 (cut-off level). *Given a quad-tree with levels $l = 0, 1, \ldots, L$, the cut-off level is defined to be the coarsest level of the tree at which the box size is less than or equal to $r_c\sqrt{\delta}$. We denote this by l_{cut}. If the box size is greater than $r_c\sqrt{\delta}$ even at the finest level (level L), we let $l_{cut} = L + 1$.*

DEFINITION 4.2 (neighbors). *Leaf nodes at the same level as B which share a boundary point, including B itself, are referred to as* colleagues. *Leaf nodes at the level of B's parent which share a boundary point with B are referred to as the* coarse neighbors *of B. Leaf nodes one level finer than B which share a boundary point with B are referred to as* fine neighbors. *The union of the colleagues, coarse neighbors, and fine neighbors of B is referred to as B's* neighbors. *The* s-list *of a box B consists of those children of B's colleagues which are not fine neighbors of B (Figure* 2.1*).*

DEFINITION 4.3 (interaction lists). *The* interaction region *for B consists of the area covered by the neighbors of B's parent, excluding the area covered by B's colleagues and coarse neighbors. The* interaction list *for B consists of those boxes in the interaction region which are at the same refinement level (marked as i in Figure* 2.1*) and is denoted by $\mathcal{I}(B)$. Boxes at coarser levels will be referred to as the* coarse interaction list*, denoted by $\mathcal{I}_c(B)$ (marked as i_c in Figure* 2.1*).*

DEFINITION 4.4 (expansions). *We denote by $B_{l,k}$ the* k*th box at refinement level l and by $\Phi_{l,k}$ the Hermite expansion describing the far field due to the source distribution supported inside $B_{l,k}$. We denote by $\Psi_{l,k}$ the local expansion describing the field due to the source distribution outside the neighbors of $B_{l,k}$, and denote by $\tilde{\Psi}_{l,k}$ the local expansion describing the field due to the source distribution outside the neighbors of the* parent *of $B_{l,k}$. When the context is clear, we will sometimes use the notation $\Phi(B)$, $\Psi(B)$, $\tilde{\Psi}(B)$ to describe the expansions associated with a box B.*

Remark 2. Let $B = B_{l,k}$ be a box in the quad-tree hierarchy with children C_1, C_2, C_3, C_4. Then, according to Lemma 3.5, there is a linear operator $\mathcal{T}_{HH}$ for which

$$\Phi_{l,k} = \Phi(B) = \mathcal{T}_{HH}[\Phi(C_1), \Phi(C_2), \Phi(C_3), \Phi(C_4)]. \tag{4.1}$$

The operator $\mathcal{T}_{HH}$ is responsible for merging the expansions of four children into a single expansion for the parent. Likewise, according to Lemma 3.6 there is a linear operator $\mathcal{T}_{LL}$ for which

$$[\tilde{\Psi}(C_1), \tilde{\Psi}(C_2), \tilde{\Psi}(C_3), \tilde{\Psi}(C_4)] = \mathcal{T}_{LL}\,\Psi_{l,k} = \mathcal{T}_{LL}\,\Psi(B). \tag{4.2}$$

$\mathcal{T}_{LL}$ is responsible for shifting the incoming data (the local expansion) from a parent box to its children. Finally, according to Lemma 3.4, for any source box $B_{l',k'}$ in the interaction list $\mathcal{I}(B)$ of box $B_{l,k}$, there is a linear operator $\mathcal{T}_{HL}$ for which the induced field in $B_{l,k}$ is given by $\Psi = T_{HL}\Phi_{l',k'}$. Clearly,

$$\Psi_{l,k} = \tilde{\Psi}_{l,k} + \sum_{i \in \mathcal{I}(B)} T_{HL}\Phi_i. \tag{4.3}$$

Since our algorithm is operating on an adaptive tree, the leaf nodes may need to handle interactions between nonneighboring boxes at different levels. More precisely,

viewing each such leaf node B as a "target box," we need to incorporate the influence of the *s-list* and the coarse interaction list (see Figure 2.1). For every box in the *s-list*, its Hermite expansion is rapidly convergent in B and its influence can be computed by direct evaluation of the series. We also need to compute the dual interaction, namely the influence of a leaf node B on a box B' in the *s-list*. Rather than evaluate the Hermite expansion of B at all targets in B', or shift to a local expansion in B', we can directly expand the influence of the polynomial source distribution in B, given by the coefficients $\vec{c}_B$, as a local expansion in B'. Thus, incorporating all far field interactions into (4.3), we have

$$\Psi_{l,k} = \tilde{\Psi}_{l,k} + \sum_{i \in \mathcal{I}} T_{HL} \Phi_i, + \sum_{i \in \mathcal{I}_c} T_{direct}\, \vec{c}_i. \tag{4.4}$$

The operator T_{direct}, which maps the coefficients of a polynomial approximation of the density in B' (a coarse interaction list box) onto the p^2 coefficients of the local expansion in B, can be precomputed and stored for each level in the quad-tree hierarchy. Inspection of Figure 2.1, the translation invariance of the kernel, and a simple counting argument show that this requires $O(kpL)$ work and storage, where k is the degree of polynomial approximation, p is the order of the local expansion, and L is the number of levels. More precisely, let $b_n(y_1, y_2) = p_{n_1}(y_1)\, p_{n_2}(y_2)$ be a basis function for the polynomial approximation in box B', and let $\alpha = (\alpha_1, \alpha_2)$ denote the multi-index of a term in the induced local expansion in B. Then

$$T_{direct}(\alpha, n) = T_1(\alpha_1, n_1)\, T_2(\alpha_2, n_2), \tag{4.5}$$

where

$$T_i(\alpha_i, n_i) = \frac{1}{\alpha_i!} \int_{D_{l-1}/2}^{D_{l-1}/2} h_{\alpha_i} \left(\frac{y_i - s_i}{\sqrt{\delta}} \right) p_{n_i}(y_i)\, dy_i\,,$$

(s_1, s_2) denotes the center of B, and D_{l-1} denotes the side length of box B' at level $l-1$.

4.1. Pseudocode for the fast Gauss transform. We assume we are given a square domain $B_{0,0}$, on which is superimposed an adaptive hierarchical quad-tree with l_{max} refinement levels. We let l_{cut} denote the cut-off level. If $l_{cut} \leq l_{max}$, for each level l that satisfies the condition $l_{cut} \leq l \leq l_{\max}$, determine the number of terms needed in the Hermite expansions $N_h(l)$ and the number of terms needed in the local expansion $N_t(l)$ according to the box size, the parameter δ that defines the variance of the Gaussian, the user-defined precision ϵ, and the estimates (3.11), (3.15).

We denote the leaf nodes by $B_i,\ i = 1, \ldots, M$, where M is the total number of leaf nodes across all levels. We assume that the source distribution on each B_i is given by a collection of point sources, as well as a smooth function f, sampled on a $k \times k$ grid. The number of grid points is denoted by $N = Mk^2$ and the number of discrete sources is denoted by N_s. We assume the output is desired at the N grid points as well as the N_s source locations.

Step I: Upward pass

for $l = l_{\max}, \ldots, l_{cut}$
  for every box j on level l
    if j is childless **then**
      • Form Hermite expansion $\Phi_{l,j}$ using (3.5)
    else
      • Form Hermite expansion $\Phi_{l,j}$ by merging the expansions of its children using $\mathcal{T}_{HH}$ (see Lemma 3.5)
    endif
  end
end

Step II: Downward pass

for every box j on level l_{cut}
  • Set $\Psi_{l_{cut},j} = 0$
end
for $l = l_{cut} + 1, \ldots, l_{\max}$
  for every box j on level l:
    • Compute $\tilde{\Psi}_{l,j}$ from its parent's Ψ expansion using the operator $\mathcal{T}_{LL}$
    for every box m in j's interaction list:
      • Increment $\Psi_{l,j}$ by adding in the contributions from all boxes in j's interaction list, using (4.4).
    if j is childless **then**
      for every box m in j's *s-list:*
        • Evaluate the Hermite expansion $\Phi(m)$ at each target in box j.
      end
      for every box m in j's *s-list:*
        • Increment the local expansion $\Psi(m)$ from the smooth and point source distribution in j, using the precomputed operators (4.5) for the smooth source distribution and (3.2) for the point sources
      end
      • Evaluate the local expansion $\Psi_{l,j}$ at each target in box j (whether the target is a grid point or a point source location)
    endif
  end
end

Step III: Local interactions

for $l = 0, \ldots, l_{\max}$
  for every *leaf node* B_j on level l:
    • At each tensor product grid point in B_j, compute influence of the smooth source in colleagues, fine neighbors, and coarse neighbors using precomputed tables of coefficients (3.26)
    • For each point source location in B_j, use Chebyshev interpolation to obtain the Gaussian field due to smooth sources in colleagues, fine and coarse neighbors
    • For *all* targets in B_j, use direct computation to evaluate the Gaussian field due to point sources in colleagues, fine and coarse neighbors
  end
end

The cost of the adaptive FGT is easily estimated. Creating the tree and sorting sources into leaf nodes requires at most $O((N+N_s)l_{\max})$ work. Forming expansions on all leaf nodes requires $O((N+N_s)p^2$ work, for an expansion of order p. The remainder of the upward pass requires $O(N_b p^3)$ work to carry out the recursive merging of Hermite expansions, where N_b is the number of boxes in the quad-tree. The downward pass requires approximately $O(27N_b p^3)$ work to carry out the Hermite-local and local-local translations. Finally, the local work is of the order $O(N_s q)$ for the point sources (assuming the tree has been refined until there are $O(q)$ sources per leaf node). For the continuous source distribution, only approximately $13N\,\frac{k(k+1)}{2}+N_s k^2$ operations are required. The first term accounts for the cost of computing the Gaussian potential on the tensor product grids from the near neighbors, using precomputed tables, while the latter term is the interpolation cost at the point source locations. The factor 13 is a consequence of the observation that the maximum number of neighbors a box can have is 13 (12 fine neighbors and itself).

Remark 3. The preceding analysis assumes that the translation operators $\mathcal{T}_{HH}$, $\mathcal{T}_{LL}$, and $\mathcal{T}_{HL}$ have been computed according to the formulae (4.1), (4.2), and (4.3), taking advantage of the tensor product nature of the two-dimensional Hermite and local expansions to achieve $O(p^3)$ complexity, instead of the naive estimate $O(p^4)$. In the d-dimensional setting, the operation count is $O(dp^{d+1})$ instead of $O(p^{2d})$ [15, 16].

We have further accelerated the code by making use of diagonal translation operators, following the method described in [16] and [25]. Instead of Hermite expansions, it is straightforward to show that

$$\sum_{\alpha\geq 0} A_\alpha h_\alpha\left(\frac{\mathbf{x}-\mathbf{s}_B}{\sqrt{\delta}}\right)=\int_{\mathbb{R}^2} w(\mathbf{k})e^{-\frac{\|\mathbf{k}\|^2}{4}}e^{i\mathbf{k}\cdot(\mathbf{x}-\mathbf{s}^B)/\sqrt{\delta}}\,d\mathbf{k}\,, \tag{4.6}$$

where

$$w(\mathbf{k})=w(k_1,k_2)=\sum_{\alpha\geq 0} A_\alpha(-i)^{|\alpha|}k_1^{\alpha_1}k_2^{\alpha_2}. \tag{4.7}$$

This formula is derived from the Fourier relation

$$e^{-\|\mathbf{x}\|^2}=\left(\frac{1}{4\pi}\right)\int_{\mathbb{R}^2} e^{-\|\mathbf{k}\|^2}4e^{i\mathbf{k}\cdot\mathbf{x}}\,d\mathbf{k}. \tag{4.8}$$

In order to make practical use of (4.6), we need to discretize the integral, for which the trapezoidal rule is extremely efficient because of the smoothness and exponential decay of the integrand. The reason (4.6) is useful is because it provides a basis in which translation is diagonal. Assuming p_t denotes the number of trapezoidal quadrature points required, it is shown in [16] and [25] that the dominant cost of translating Hermite to local expansions, namely, the $O(27N_b p^3)$ term above, can be reduced to $O(3N_b p_t^2+N_b p p_t^2)$ work.

The principal difference between the methods in [16] and [25] and our hierarchical scheme is that p_t must be different on each level. Informally speaking, for a level where the linear box size is r_l, p_t must be sufficiently large so that the integrand $e^{i\mathbf{k}\cdot(\mathbf{x}-\mathbf{s}^B)/\sqrt{\delta}}$ is Nyquist-sampled for $(\mathbf{x}-\mathbf{s}^B)\leq 4r_l$, where $r_l<r_c\sqrt{\delta}$ and r_c is the cut-off parameter defined above. (It is easy to verify that $p_t=O(p)$ [16, 25].)

4.2. Boundary fast Gauss transform. We turn now to the evaluation of boundary Gauss transforms of the form (1.13) for targets both on and off the boundary

Γ. We assume that Γ itself is described as the union of M_b boundary segments:

$$\Gamma = \cup_{j=1}^{M_b} \Gamma_j,$$

with each boundary segment defined by a kth order Legendre series. That is,

(4.9)
$$\Gamma_j = \Gamma_j(s) = (x_j^1(s), x_j^2(s)) : \quad x_j^1(s) = \sum_{n=0}^{k-1} x_j^1(n) P_n(s),\ x_j^2(s) = \sum_{n=0}^{k-1} x_j^2(n) P_n(s),$$

with $-1 \le s \le 1$. We assume that the densities σ and μ in (1.13) are also given by corresponding piecewise Legendre series:

$$\sigma_j(s) = \sum_{n=0}^{k-1} \sigma_j(n) P_n(s), \quad \mu_j(s) = \sum_{n=0}^{k-1} \mu_j(n) P_n(s).$$

For the sake of simplicity, we assume that Γ has been discretized in a manner that is commensurate with the underlying adaptive data structure used above. That is, we assume the length of Γ_j, denoted by $|\Gamma_j|$, satisfies $|\Gamma_j| \approx r_l$, where r_l is the box size of the leaf node in the tree that contains the center point $\mathbf{c}_j$ of Γ_j.

Suppose now that we apply composite Gauss–Legendre quadrature to the integrals in (1.13). For the "single layer" type integral, we have

(4.10)
$$\mathcal{S}[\sigma](\mathbf{x}) = \int_\Gamma e^{-\frac{|\mathbf{x}-\mathbf{y}(s)|^2}{\delta}} \sigma(\mathbf{y}(s))\, ds_{\mathbf{y}} \approx \sum_{j=1}^{M_b} \sum_{i=0}^{k-1} e^{-\frac{|\mathbf{x}-\mathbf{y}_{ij}|^2}{\delta}} \sigma_{ij} w_{ij}\,,$$

where $\mathbf{y}_{ij} = (y_{ij}^1, y_{ij}^2)$ is the location of the ith scaled Gauss–Legendre node on Γ_j, σ_{ij} is the density value at that point, and $w_{ij} = w_i \sqrt{[dy_j^1/ds(s_i)]^2 + [dy_j^2/ds(s_i)]^2}$. Here, s_i and w_i denote the standard Gauss–Legendre nodes and weights on $[-1, 1]$.

Note that the quadrature weight w_{ij} involves both the standard weight w_i and the change of variables corresponding to an arc-length parametrization on each segment. The necessary derivatives can be computed from (4.9). Note also that the sum in (4.10) consists simply of point sources and is easily incorporated into the FGT above. As a precomputation step, we carry out a "tree sort" to assign point sources to boxes in the tree. After that the algorithm remains largely unchanged from the volume FGT, with the only exception that Hermite expansions are formed from point sources instead of volume integrals, according to Lemma 3.1. The point source FGT computes the sum (4.10) to a user-specified precision ϵ.

Now the only remaining issue has to do with the accuracy of the formula (4.10), since the smoothness of the integrand depends strongly on the parameter δ. Here, however, the rapid decay of the Gaussian makes the problem tractable for any δ. To see why, consider a boundary segment Γ_j, centered at $\mathbf{c}_j$ in a leaf box B of commensurate size (Figure 4.1, left). Suppose first that δ is sufficiently large that $C\sqrt{\delta} \ge |\Gamma_j|$, where $C = 2\sqrt{\ln(1/\epsilon)}$. Then the Gaussian is a smooth function on the scale of the boundary segment, and k-point Gauss–Legendre quadrature is *spectrally accurate*. Suppose, on the other hand, that $C\sqrt{\delta} < |\Gamma_j|$, and let $r_{corr} = |\Gamma_j|$. Then, for any target outside the circle of radius r_{corr} centered at $\mathbf{c}_j$, the integrand is bounded by $\epsilon\,|\Gamma_j|\,\|\sigma\|_\infty$. Thus, for each boundary segment with $C\sqrt{\delta} < |\Gamma_j|$, it remains only to correct the result obtained from the FGT within this circle D_j (Figure 4.1, right).

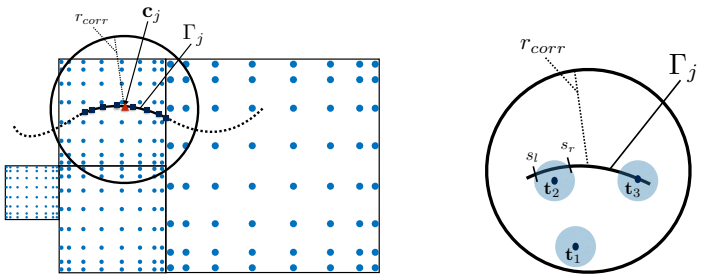

FIG. 4.1. *(Left) A boundary segment Γ_j with center point $\mathbf{c}_j$ lying in a leaf node B of side length r_l. Depending on the value of δ, a boundary integral of the form (1.13) is either resolved by its discretization using standard Gauss–Legendre quadrature with k nodes on Γ_j, or negligible outside the disk centered at $\mathbf{c}_j$ with radius r_{corr}, which we will denote by D_j. (Right) In the latter case (when δ is small and the Gaussian is sharply peaked), a simple interpolatory rule can be used to compute the correct contribution using $O(k^2)$ work per target point, either on or off the boundary. The shaded circles in the figure around the three target points $\mathbf{t}_i$ are meant to indicate the regions where the Gaussians centered at $\mathbf{t}_i$ are less than a user-prescribed tolerance ϵ. $\mathbf{t}_1$ is sufficiently far from Γ_j that it can be ignored. $\mathbf{t}_2$ and $\mathbf{t}_3$ are off and on the boundary, respectively. The relevant portion of Γ_j for $\mathbf{t}_2$ is marked in terms of the parameter s by s_l and s_r.*

This correction can be computed rapidly and accurately as follows. Taking the point of view of some target $\mathbf{t} \in D_j$, let us denote by $D(\mathbf{t}, r)$ the circle of radius r centered at $\mathbf{t}$, where $r = \sqrt{\delta \ln(1/\epsilon)}$ so that $e^{-\|\mathbf{t}-\mathbf{y}\|^2/\delta} < \epsilon$. If $D(\mathbf{t}, r)$ does not intersect Γ_j, the field is negligible and no correction is needed. Otherwise, we compute the intersection of Γ_j and $D(\mathbf{t}, r)$ and let the endpoints of the intersection be denoted by s_l and s_r (in terms of the underlying parametrization of Γ_j). We then interpolate the source distribution $\sigma(s)$ to k_c scaled Gauss–Legendre nodes on Γ_j^s for $s \in [s_l, s_r]$ and replace the original k-point quadrature on Γ_j by a k_c-point Gauss–Legendre rule on $[s_l, s_r]$. Setting k_c to 20 yields approximately 14 digits of accuracy assuming the density $\sigma(s)$ is locally smooth.

4.3. Periodic boundary conditions. It is straightforward to extend the FGT to handle periodic conditions on the unit square $D = [-0.5, 0.5]^2$. Conceptually speaking, this can be accomplished by tiling the entire plane $\mathbf{R}^2$ with copies of the source distribution f. For this, we let $\Lambda = \{\mathbf{j} = (j_1, j_2) | j_1, j_2 \in \mathbf{Z}\}$. The tile $\mathrm{T}_\mathbf{j}$ is a unit square centered at the lattice point $\mathbf{j} \in \Lambda$. The extended periodic source distribution will be denoted by $\tilde{f}$. From this, the solution to the periodic problem can be written as

$$\tilde{F}(\mathbf{x}) = \int_{\mathbf{R}^2} e^{-\frac{|\mathbf{x}-\mathbf{y}|^2}{\delta}} \tilde{f}(\mathbf{y})\, d\mathbf{y}, \quad \mathbf{x} \in D.$$

As in the FMM for the Poisson equation [14, 10], we can accommodate periodic boundary conditions with very little change to the data structure or processing. To see this, note that, if we carry out the upward pass of the FGT until the root node (level $l = 0$), we obtain a Hermite expansion describing the field due to all sources in D. Because of the translation invariance of the kernel, the coefficients of this expansion are the same for every tile $\mathrm{T}_\mathbf{j}$ covering the plane, expanded about the corresponding lattice point $\mathbf{j}$. We denote the expansion about $\mathbf{j}$ by

$$\phi(\mathbf{x}) = \sum_{\alpha \le p} A_\alpha h_\alpha \left(\frac{\mathbf{x} - \mathbf{j}}{\sqrt{\delta}} \right).$$

Let us now define the root node's colleagues at level 0 by

$$\mathcal{C} = \{(-1,-1),(-1,0),(-1,1),(0,-1),(0,0),(0,1),(1,-1),(1,0),(1,1)\}$$

and the punctured lattice by $\Lambda' = \Lambda - \mathcal{C}$. The tiles indexed by Λ' are clearly well-separated from D. From Lemma 3.4 and the linearity of the problem, it is clear that the contribution to the field in D from *all* tiles in Λ' can be represented by a local expansion

$$\tilde{F}_{far}(\mathbf{x}) = \sum_{\beta \geq 0} C_\beta \left(\frac{\mathbf{x}}{\sqrt{\delta}} \right)^\beta, \tag{4.11}$$

centered at the origin, with coefficients

$$C_\beta = \frac{(-1)^{|\beta|}}{\beta!} \sum_{\alpha \leq p} A_\alpha \mathcal{L}_{\alpha+\beta}, \tag{4.12}$$

where

$$\mathcal{L}_{\alpha+\beta} = \sum_{\mathbf{j} \in \Lambda'} h_{\alpha+\beta} \left(\frac{\mathbf{j}}{\sqrt{\delta}} \right). \tag{4.13}$$

Extending Definitions 4.2–4.4, we let $\Psi_{0,1}$ denote the local expansion for the root node D at level 0. When δ is so small that $l_{cut} \geq 0$, the far field $\tilde{F}_{far}(\mathbf{x})$ in the root node D is negligible (for given accuracy ϵ), so that we can initialize the coefficients of $\Psi_{0,1}$ to zero. Otherwise, we carry out the computation in (4.12) to obtain $\Psi_{0,1}$. This requires the evaluation of the lattice sums in (4.13). These are obtained rapidly from the Poisson summation formula:

$$\sum_{j_1=-\infty}^{\infty} \sum_{j_2=-\infty}^{\infty} h_\alpha \left(\frac{(j_1, j_2)}{\sqrt{\delta}} \right) = \pi^2 \delta^2 (-2\pi i \sqrt{\delta})^{\alpha_1+\alpha_2} \left(\sum_{m=-\infty}^{\infty} m^{\alpha_1} \cdot e^{-\pi^2 m^2 \delta} \right) \left(\sum_{n=-\infty}^{\infty} n^{\alpha_2} \cdot e^{-\pi^2 n^2 \delta} \right). \tag{4.14}$$

It is straightforward to verify that, when δ is large enough that $\tilde{F}_{far}(\mathbf{x})$ is non-negligible, only a few terms are required on the right-hand side of (4.14) and only milliseconds are needed for all $\mathcal{L}_{\alpha+\beta}$.

Only two other changes are needed in the FGT: the interaction list and near neighbor computations must be adjusted to account for periodic images. Having defined the colleagues of the root node above, this is handled automatically by the data structure. For large-scale problems with many levels of refinement, this involves a modest increase in work for boxes near the boundary of D and a negligible increase in the total work.

5. Numerical results. In this section, we illustrate the performance of both the volume and boundary FGT, implemented in Fortran, with experiments carried out on a single core 3.4GHz Intel Xeon processor.

DEFINITION 5.1. *In each example, we use the term* throughput *to mean the number of points processed per second. This is a useful benchmark for linear scaling algorithms, permitting simple estimation of the performance in terms of CPU time to any problem size.*

Our first example demonstrates the linear scaling of CPU time with the number of grid points. We compute the volume Gauss transform

$$G_\delta[f_k](\mathbf{x}) = \int_{\mathbf{R}^2} e^{-\frac{|\mathbf{x}-\mathbf{y}|^2}{\delta}} f(\mathbf{y}) d\mathbf{y}\,, \tag{5.1}$$

with the source distribution

$$f_k(\mathbf{x}) = \sin(2k\pi x_1)\cos(2k\pi x_2) \quad (k \in \mathbf{Z}), \tag{5.2}$$

imposing periodic boundary conditions.

In order for the numerical experiment to be nontrivial, we increase the complexity of the problem as we increase the number of degrees of freedom. More precisely, we consider four cases, with $k = 1, 2, 4, 8$ and $\delta = \frac{1}{k^2}$, requiring a finer and finer spatial mesh to resolve the data. For each choice of k, we create a level-restricted quad-tree, refined to a level where f_k is accurately represented with our piecewise eighth degree polynomial to 10 digits of accuracy. For the function described in (5.2), the refinement happens to be uniform, with $N = 256, 1024, 4096, 16384$ leaf nodes for the four cases, respectively. (We will see examples with inhomogeneous source distributions and adaptive data structures below.) Timings are provided in tabular and graphical form in Table 5.1 and Figure 5.1, respectively. In each case, the resulting error is less than the requested precision ϵ.

TABLE 5.1
Throughput on a single core for the volume FGT with periodic boundary conditions.

	k			
ϵ	1	2	4	8
10^{-3}	$3.0 \cdot 10^5$	$3.3 \cdot 10^5$	$6.1 \cdot 10^5$	$7.3 \cdot 10^5$
10^{-6}	$1.7 \cdot 10^5$	$1.8 \cdot 10^5$	$3.1 \cdot 10^5$	$4.3 \cdot 10^5$
10^{-9}	$0.9 \cdot 10^5$	$0.9 \cdot 10^5$	$1.1 \cdot 10^5$	$1.3 \cdot 10^5$

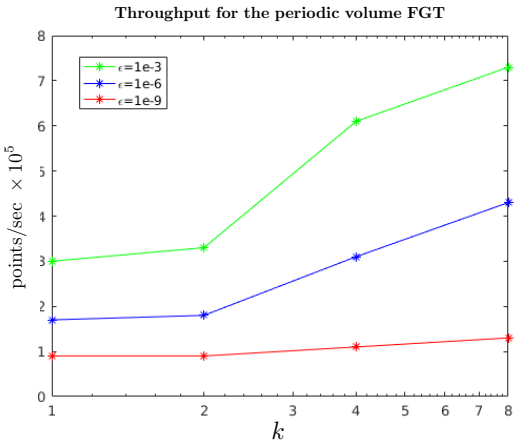

FIG. 5.1. *Throughput for the volume FGT with various precisions, plotted as a function of δ. (The data is the same as in Table 5.1.)*

While the cost appears to grow in a sublinear fashion with the number of grid points, this is simply because of the nontrivial cost of the precomputation. A more precise model for the CPU time takes the form

$$T(N,\epsilon) = A(\epsilon)N + B(\epsilon)\log N \tag{5.3}$$

for precision ϵ. The term $B(\epsilon)\log N$ is dominated by the building of tables for the local interactions, which is done once per level. In the present example, the sublinear part contributes about 30% of the cost for the smaller problem sizes and less than 10% for the largest N. If we subtract the time for precomputation/table building, and measure the time of the remainder of the FGT, we see a steady throughput for each fixed precision. This verifies the linear scaling (Table 5.2 and Figure 5.2). For the rest of the examples, the timings presented always include the precomputation.

TABLE 5.2
Throughput on a single core for the volume FGT with periodic boundary conditions.

	k			
ϵ	1	2	4	8
10^{-3}	$7.0 \cdot 10^5$	$7.2 \cdot 10^5$	$8.2 \cdot 10^5$	$8.1 \cdot 10^5$
10^{-6}	$3.8 \cdot 10^5$	$4.0 \cdot 10^5$	$4.3 \cdot 10^5$	$4.1 \cdot 10^5$
10^{-9}	$1.5 \cdot 10^5$	$1.5 \cdot 10^5$	$1.4 \cdot 10^5$	$1.5 \cdot 10^5$

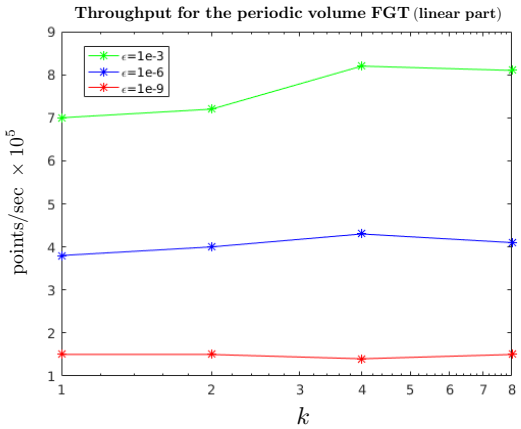

FIG. 5.2. *Throughput for the volume FGT with various precisions, plotted as a function of δ. (The data is the same as in Table 5.2.)*

For our second example, we again compute the volume FGT with periodic boundary condition where f_k is given by (5.2) with $k = 2$. The integral $G_\delta[f_k]$ is available analytically for this $f_k(\mathbf{x})$ from Fourier analysis. The source distribution is again resolved to 10 digits of accuracy, but we now compute the FGT with requested precisions of $\epsilon = 10^{-3}$, 10^{-6}, and 10^{-9}. For each choice of ϵ, we carry out the computation for a wide range of δ, from $\delta = 10^{-7}$ to $\delta = 10^{-1}$. Timings are provided in Table 5.3 and plotted in Figure 5.3. In each case, the resulting error is less than ϵ.

Our third example illustrates the performance of the FGT with an adaptive data structure. For this, we let B denote the unit box, with f given by

$$f(\mathbf{x}) = \sum_{i=1}^{5} e^{-\alpha_i |\mathbf{x}-\mathbf{x}_i|^2}, \tag{5.4}$$

TABLE 5.3
Throughput on a single core for the volume FGT with periodic boundary conditions.

	δ						
ϵ	10^{-1}	10^{-2}	10^{-3}	10^{-4}	10^{-5}	10^{-6}	10^{-7}
10^{-3}	$3.4 \cdot 10^5$	$3.5 \cdot 10^5$	$3.3 \cdot 10^5$	$5.5 \cdot 10^5$	$6.2 \cdot 10^5$	$6.9 \cdot 10^5$	$7.3 \cdot 10^5$
10^{-6}	$1.7 \cdot 10^5$	$1.7 \cdot 10^5$	$1.7 \cdot 10^5$	$0.7 \cdot 10^5$	$3.0 \cdot 10^5$	$5.0 \cdot 10^5$	$5.4 \cdot 10^5$
10^{-9}	$0.8 \cdot 10^5$	$0.9 \cdot 10^5$	$0.6 \cdot 10^5$	$0.3 \cdot 10^5$	$3.5 \cdot 10^5$	$4.0 \cdot 10^5$	$4.4 \cdot 10^5$

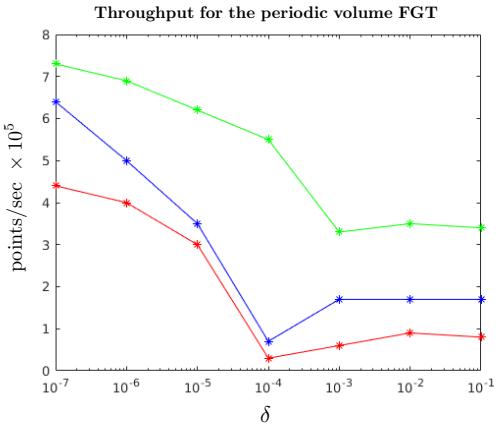

FIG. 5.3. *Throughput for the periodic volume FGT with various precisions, plotted as a function of δ. (The data is the same as in Table 5.3.)*

with

$$[\mathbf{x}_1, \ldots, \mathbf{x}_5] = [(0.20, 0.10), (0.31, 0.50), (0.68, 0.40), (0.41, 0.80), (0.12, 0.45)]$$

and

$$[\alpha_1, \ldots, \alpha_5] = (0.010, 0.005, 0.003, 0.002, 0.001).$$

The volume integral $G_\delta[f]$ is available analytically for this choice of $f(\mathbf{x})$. In Figure 5.4, we plot the source distribution along with a level-restricted quad-tree on which the source distribution is resolved to 10 digits of accuracy. We compute the volume FGT with requested precisions of $\epsilon = 10^{-3}$, 10^{-6}, and 10^{-9}. For each choice of ϵ, we carry out the computation for a wide range of δ, from $\delta = 10^{-7}$ to $\delta = 10^{-1}$. Timings are given in Table 5.4 and plotted in Figure 5.5. In each case, the resulting error is less than ϵ.

Note that, for a fixed ϵ, the performance of the volume FGT is relatively insensitive to the variance parameter δ. For large δ, the far field is nontrivial but very smooth. For sufficiently small δ, the interaction is entirely local and the FGT is particularly fast. The worst case performance is for $\delta \approx 10^{-4}$, where both the far field and the near field require significant effort.

Our fourth example illustrates the performance of the boundary FGT. We compute the integral

$$G_\delta[\sigma](\mathbf{x}) = \int_\Gamma e^{-\frac{|\mathbf{x}-\mathbf{y}|^2}{\delta}} \sigma(\mathbf{y}) ds_{\mathbf{y}}\,, \tag{5.5}$$

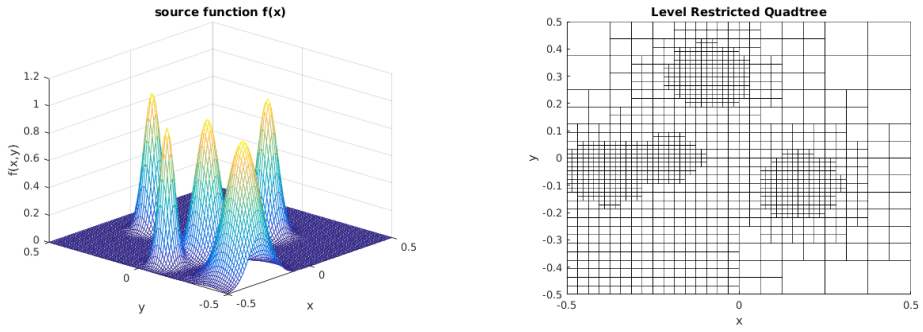

FIG. 5.4. *(Left) Plot of the volume source distribution, which is taken to be the sum of a few Gaussians. (Right) A level-restricted quad-tree on which the source distribution is resolved to* 10 *digits of accuracy as a piecewise polynomial of degree* 8.

TABLE 5.4
Throughput on a single core for the adaptive volume FGT (third example) with various precisions.

	δ						
ϵ	10^{-1}	10^{-2}	10^{-3}	10^{-4}	10^{-5}	10^{-6}	10^{-7}
10^{-3}	$2.8 \cdot 10^5$	$2.8 \cdot 10^5$	$3.4 \cdot 10^5$	$4.3 \cdot 10^5$	$5.0 \cdot 10^5$	$7.5 \cdot 10^5$	$8.0 \cdot 10^5$
10^{-6}	$1.6 \cdot 10^5$	$1.6 \cdot 10^5$	$1.5 \cdot 10^5$	$0.9 \cdot 10^5$	$3.1 \cdot 10^5$	$5.4 \cdot 10^5$	$6.0 \cdot 10^5$
10^{-9}	$0.8 \cdot 10^5$	$0.8 \cdot 10^5$	$0.5 \cdot 10^5$	$0.4 \cdot 10^5$	$3.6 \cdot 10^5$	$4.3 \cdot 10^5$	$4.8 \cdot 10^5$

where Γ is chosen to be the ellipse:

$$\begin{cases} y_1(\theta) = 0.45\cos(\theta), \\ y_2(\theta) = 0.25\sin(\theta), \end{cases} \quad (0 \le \theta \le 2\pi)\,. \tag{5.6}$$

We let

$$\sigma(\mathbf{y}) = \cos(2y_1) + \sin(y_2). \tag{5.7}$$

We discretize Γ adaptively as a function of θ with piecewise 16th order Gauss–Legendre panels until σ, y_1, and y_2 are all resolved to 12 digits of accuracy. We then create an adaptive quad-tree on the unit box, so that each leaf box of the tree contains no more than $O(1)$ boundary points, and then enforce the level-restricted condition, yielding the data structure shown in Figure 5.6. The leaf nodes (with 8×8 tensor product Chebyshev grids on each) define our volumetric targets. The boundary FGT is then evaluated at all volumetric grid points and all boundary points as well. Timings are given in Table 5.5 and plotted in Figure 5.7. In each case, the resulting error is less than than the requested tolerance ϵ.

Note that, again, the performance of the boundary FGT varies only modestly over a wide range of the parameter δ. For sufficiently small δ, the interaction is entirely local and no expansions are formed. For sufficiently large δ, a smooth quadrature rule is accurate enough to discretize the boundary integral, avoiding the need for local correction. The code is slowest for intermediate values of δ, where both local and far field contributions are significant (while still satisfying linear scaling with the number of source and target points).

Remark 4. As noted above, the boundary FGT relies on the FGT for discrete point sources, which carries out the rapid evaluation of sums of the form (1.14). The

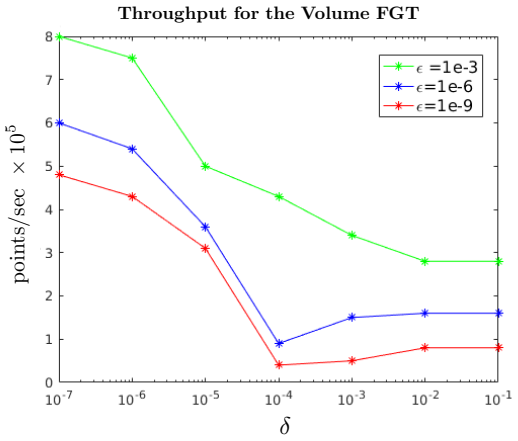

FIG. 5.5. *Throughput for the adaptive volume FGT with various precisions, plotted as a function of δ. (The data is the same as in Table 5.4.)*

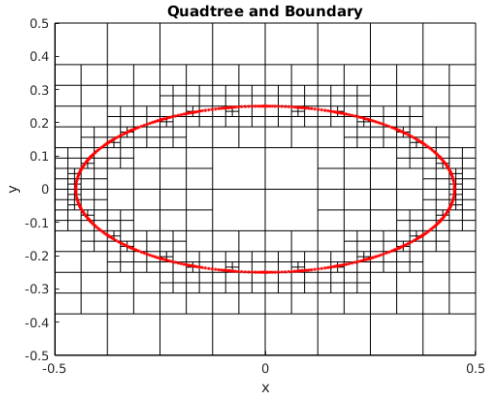

FIG. 5.6. *A level-restricted quad-tree determined by the discretization of the indicated ellipse.*

algorithm outline in section 4 describes all of the steps needed for the discrete FGT using the same level-restricted quad-tree as used for volume source distributions. We should point out, however, that the scheme of [16, 25] is slightly faster. To give an indication of relative performance, we consider the sum (1.14) with 100,000 random sources and targets located within the unit box $[-0.5, 0.5]^2$ with $\delta = 10^{-3}$. For nine digits of accuracy, the fast generalized Gauss transform [25] has a throughput of 1.8×10^6 points per second, while the hierarchical FGT yields 7.1×10^5 points per second. Since we are primarily interested in volume distributions, we have used only the hierarchical FGT in our experiments.

There are also a variety of kernel-independent or black-box FMMs which could be modified to handle Gaussian kernels and volume distributions, such as those in [11, 22, 34]. These typically do not come equipped with diagonal translation operators, and we have not attempted a direct comparison with any of these schemes.

TABLE 5.5
Throughput on a single core for the adaptive volume FGT (third example) with various precisions.

	δ						
ϵ	10^0	10^{-1}	10^{-2}	10^{-3}	10^{-4}	10^{-5}	10^{-6}
10^{-3}	$4.2 \cdot 10^5$	$3.9 \cdot 10^5$	$4.1 \cdot 10^5$	$3.6 \cdot 10^5$	$5.0 \cdot 10^5$	$2.8 \cdot 10^5$	$5.4 \cdot 10^5$
10^{-6}	$2.0 \cdot 10^5$	$1.8 \cdot 10^5$	$1.6 \cdot 10^5$	$1.6 \cdot 10^5$	$0.9 \cdot 10^5$	$1.8 \cdot 10^5$	$3.9 \cdot 10^5$
10^{-9}	$1.0 \cdot 10^5$	$0.9 \cdot 10^5$	$0.8 \cdot 10^5$	$0.4 \cdot 10^5$	$0.2 \cdot 10^5$	$1.1 \cdot 10^5$	$2.7 \cdot 10^5$

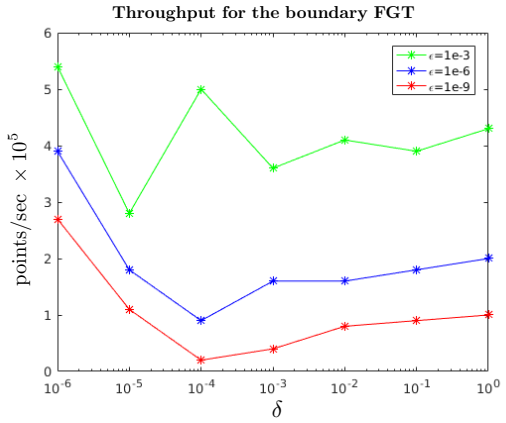

FIG. 5.7. *Throughput for the boundary FGT with boundary and volume targets for various precisions, plotted as a function of δ (fourth example).*

5.1. An initial value problem for the heat equation. As a final example, we consider the homogeneous heat equation,

$$u_t(\mathbf{x}, t) = \Delta u(\mathbf{x}, t), \qquad u(\mathbf{x}, 0) = f(\mathbf{x}) \tag{5.8}$$

for $\mathbf{x} \in D = [-0.5, 0.5]^2$, with periodic boundary conditions. The initial data is chosen to be a piecewise constant function,

$$f(\mathbf{x}) = C_i \quad \text{for } \mathbf{x} \in D_i, \tag{5.9}$$

where the unit box D is refined uniformly on a tree that is five levels deep, resulting in a 32-by-32 grid of leaf nodes D_i. On each leaf node, we let C_i take on a random value in the range $[0, 1]$. $f(x)$ is plotted in Figure 5.8.

The exact solution of this problem is given by

$$u(\mathbf{x}, t) = \frac{1}{4\pi t} \int_{\mathbf{R}^2} e^{-\frac{|\mathbf{x}-\mathbf{y}|^2}{4t}} \tilde{f}(\mathbf{y}) d\mathbf{y}, \tag{5.10}$$

where $\tilde{f}$ is the periodic extension of f. This is precisely what is computed by the periodic version of the volume FGT, and the solution $u(\mathbf{x}, t)$ is plotted for various choices of t in Figure 5.8, with nine digits of precision in the FGT.

Remark 5. There is a subtle issue regarding the use of the FGT to compute (5.10), namely, that the error estimates for the FGT derived above are based on the Gaussian rather than the heat kernel, which includes the additional $1/(4\pi t)$ scaling

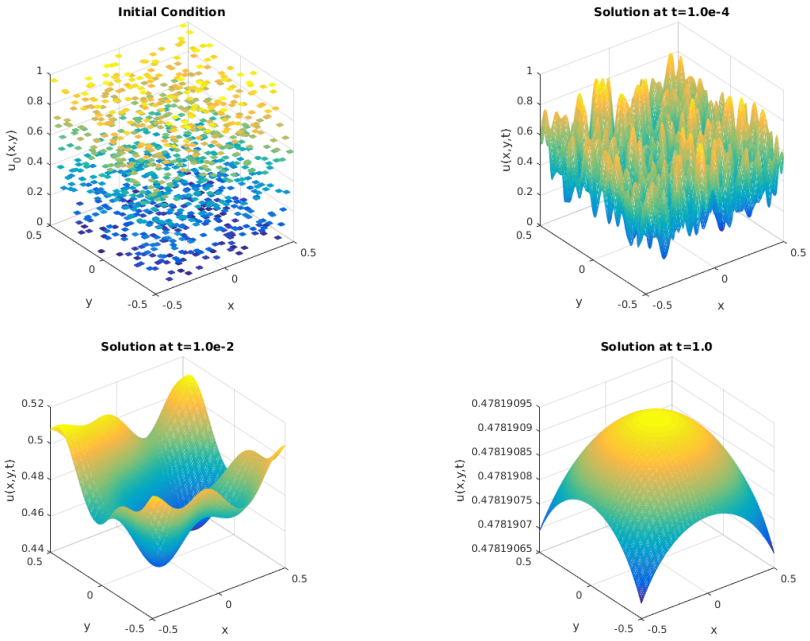

FIG. 5.8. *The solution to the heat equation on a box with periodic boundary conditions and piecewise constant initial data, plotted at various times.*

in two dimensions. To compute an accurate convolution requires that the local tables be built using the full heat kernel (whose support to a fixed precision ϵ is slightly greater than the support of the Gaussian alone). The far field and local expansions also require a few more terms. Without entering into a detailed analysis, we illustrate the difference when $t = 10^{-4}$ for leaf node boxes in the present example. For the FGT, Hermite expansions of order $p = 22$ are needed to achieve nine digits of precision. For the full heat kernel, it turns out that $p = 28$ is required to achieve the same accuracy.

6. Conclusions. We have presented a new adaptive version of the fast Gauss transform (FGT), which can be used for the evaluation of volume or boundary integrals with a Gaussian kernel as well as the field induced by discrete point sources. This is a standard and well-defined computational task in its own right, and serves as a key component in integral equation–based solvers for the heat equation in complex geometry [33]. The extension of the present method to three dimensions is straightforward and will be reported at a later date.

REFERENCES

[1] D. N. ARNOLD AND P. NOON, *Coercivity of the single layer heat potential*, J. Comput. Math., 7 (1989), pp. 100–104, http://www.jstor.org/stable/43692419.
[2] T. ASKHAM AND A. J. CERFON, *An adaptive fast multipole accelerated Poisson solver for complex geometries*, J. Comput. Phys., 344 (2017), pp. 1–22, https://doi.org/10.1016/j.jcp.2017.04.063.
[3] B. J. C. BAXTER AND G. ROUSSOS, *A new error estimate of the fast Gauss transform*, SIAM J. Sci. Comput., 24 (2002), pp. 257–259, https://doi.org/10.1137/S1064827501396920.
[4] J. P. BOYD, *Chebyshev and Fourier Spectral Methods*, Dover, New York, 2001.

[5] R. M. Brown, *The method of layer potentials for the heat equation in Lipschitz cylinders*, Amer. J. Math., 111 (1989), pp. 339–379, https://doi.org/10.2307/2374513.
[6] H. Cheng, J. Huang, and T. J. Leiterman, *An adaptive fast solver for the modified Helmholtz equation in two dimensions*, J. Comput. Phys., 211 (2006), pp. 616–637, https://doi.org/10.1016/j.jcp.2005.06.006.
[7] M. Costabel, *Time-dependent problems with the boundary integral equation method*, in Encyclopedia of Computational Mechanics, E. Stein, R. de Borst, and T. J. R. Hughes, eds., John Wiley & Sons, New York, 2004, pp. 703–721, https://doi.org/10.1002/0470091355.ecm022.
[8] G. F. Dargush and P. K. Banerjee, *Application of the boundary element method to transient heat conduction*, Int. J. Numer. Methods Engrg., 31 (1991), pp. 1231–1247, https://doi.org/10.1002/nme.1620310613.
[9] A. Elgammal, R. Duraiswami, and L. S. Davis, *Efficient kernel density estimation using the fast Gauss transform with applications to color modeling and tracking*, IEEE Trans. Pattern Anal. Mach. Intell., 25 (2003), pp. 1499–1504, https://doi.org/10.1109/TPAMI.2003.1240123.
[10] F. Ethridge and L. Greengard, *A new fast-multipole accelerated Poisson solver in two dimensions*, SIAM J. Sci. Comput., 23 (2001), pp. 741–760, https://doi.org/10.1137/S1064827500369967.
[11] W. Fong and E. Darve, *The black-box fast multipole method*, J. Comput. Phys., 228 (2009), pp. 8712–8725, https://doi.org/10.1016/j.jcp.2009.08.031.
[12] L. Greengard and J.-Y. Lee, *A direct adaptive Poisson solver of arbitrary order accuracy*, J. Comput. Phys., 125 (1996), pp. 415–424, https://doi.org/10.1006/jcph.1996.0103.
[13] L. Greengard and P. Lin, *Spectral approximation of the free-space heat kernel*, Appl. Comput. Harmon. Anal., 9 (2000), pp. 83–97, https://doi.org/10.1006/acha.2000.0310.
[14] L. Greengard and V. Rokhlin, *A fast algorithm for particle simulations*, J. Comput. Phys., 135 (1997), pp. 280–292, https://doi.org/10.1006/jcph.1997.5706.
[15] L. Greengard and J. Strain, *The fast Gauss transform*, SIAM J. Sci. Stat. Comput., 12 (1991), pp. 79–94, https://doi.org/10.1137/0912004.
[16] L. Greengard and X. Sun, *A new version of the fast Gauss transform*, Doc. Math., III (1998), pp. 575–584.
[17] R. B. Guenther and J. W. Lee, *Partial Differential Equations of Mathematical Physics and Integral Equations*, Prentice Hall, Englewood Cliffs, NJ, 1988.
[18] M. T. Ibáñez and H. Power, *An efficient direct BEM numerical scheme for phase change problems using Fourier series*, Comput. Methods Appl. Mech. Engrg., 191 (2002), pp. 2371–2402, https://doi.org/10.1016/S0045-7825(01)00416-9.
[19] M. H. Langston, L. Greengard, and D. Zorin, *A free-space adaptive FMM-based PDE solver in three dimensions*, Comm. Appl. Math. Comput. Sci., 6 (2011), pp. 79–122, https://doi.org/10.2140/camcos.2011.6.79.
[20] D. Lee, A. Gray, and A. Moore, *Dual-tree fast Gauss transforms*, Adv. Neural Inform. Process. Systems, 18 (2006), pp. 747–754.
[21] J.-R. Li and L. Greengard, *High order accurate methods for the evaluation of layer heat potentials*, SIAM J. Sci. Comput., 31 (2009), pp. 3847–3860, https://doi.org/10.1137/080732389.
[22] D. Malhotra and G. Biros, *Algorithm 967: A distributed-memory fast multipole method for volume potentials*, ACM Trans. Math. Softw., 43 (2016), 17, https://doi.org/10.1145/2898349.
[23] W. Pogorzelski, *Integral Equations and Their Applications*, Pergamon Press, Oxford, 1966.
[24] R. S. Sampath, H. Sundar, and S. Veerapaneni, *Parallel fast Gauss transform*, in Proceedings of the ACM/IEEE International Conference for High Performance Computing, Networking, Storage and Analysis (SC '10), New Orleans, LA, 2010, pp. 1–10.
[25] M. Spivak, S. Veerapaneni, and L. Greengard, *The fast generalized Gauss transform*, SIAM J. Sci. Comput., 32 (2010), pp. 3092–3107, https://doi.org/10.1137/100790744.
[26] J. Strain, *The fast Gauss transform with variable scales*, SIAM J. Sci. Stat. Comput., 12 (1991), pp. 1131–1139, https://doi.org/10.1137/0912059.
[27] J. Strain, *Fast adaptive methods for the free-space heat equation*, SIAM J. Sci. Comput., 15 (1994), pp. 185–206, https://doi.org/10.1137/0915013.
[28] J. Tausch and A. Weckiewicz, *Multidimensional fast Gauss transforms by Chebyshev expansions*, SIAM J. Sci. Comput., 31 (2009), pp. 3547–3565, https://doi.org/10.1137/080732729.
[29] L. N. Trefethen, *Multivariate polynomial approximation in the hypercube*, Proc. Amer. Math. Soc., 145 (2017), pp. 4837–4844, https://doi.org/10.1090/proc/13623.

[30] S. K. Veerapaneni and G. Biros, *A high-order solver for the heat equation in* $1D$ *domains with moving boundaries*, SIAM J. Sci. Comput., 29 (2007), pp. 2581–2606, https://doi.org/10.1137/060677896.

[31] S. K. Veerapaneni and G. Biros, *The Chebyshev fast Gauss and nonuniform fast Fourier transforms and their application to the evaluation of distributed heat potentials*, J. Comput. Phys., 227 (2008), pp. 7768–7790, https://doi.org/10.1016/j.jcp.2008.05.003.

[32] X. Wan and G. Karniadakis, *A sharp error estimate for the fast Gauss transform*, J. Comput. Phys., 219 (2006), pp. 7–12, https://doi.org/10.1016/j.jcp.2006.04.016.

[33] J. Wang, L. Greengard, S. Jiang, and S. K. Veerapaneni, *A high-order solver for the two-dimensional heat equation in moving domains*, in preparation, 2018.

[34] L. Ying, G. Biros, and D. Zorin, *A kernel-independent adaptive fast mulitpole method in two and three dimensions*, J. Comput. Phys., 196 (2004), pp. 591–626, https://doi.org/10.1016/j.jcp.2003.11.021.