Journal of Computational Physics 229 (2010) 1311–1326

Contents lists available at ScienceDirect

Journal of Computational Physics

journal homepage: www.elsevier.com/locate/jcp

The uselessness of the Fast Gauss Transform for summing Gaussian radial basis function series

John P. Boyd *

Department of Atmospheric, Oceanic and Space Science, University of Michigan, 2455, Hayward Avenue, Ann Arbor, MI 48109-2143, United States

ARTICLE INFO

Article history:
Received 8 February 2009
Received in revised form 4 June 2009
Accepted 19 October 2009
Available online 3 November 2009

Keywords:
Fast Gauss Transform
Radial basis functions
Fast multipole
Treecode

ABSTRACT

The Fast Gauss Transform is an algorithm for summing a series of Gaussians which is sometimes much faster than direct summation. Gaussian series in d dimensions are of the form $\sum_{j=1}^{N} \lambda_j \exp(-[\alpha/h]^2 \ \|\mathbf{x}-\mathbf{x}_j\|^2)$ where the $\mathbf{x}_j$ are the centers, h is the average separation between centers and α is the relative inverse width parameter. We show that the speed-up of the Fast Gauss Transform is bounded by a factor $\Omega(\alpha)$. When $\alpha \ll 1$, Ω can be large. However, when applied to Gaussian radial basis function interpolation, it is difficult to apply the Gaussian basis in this parameter range because the interpolation matrix is exponentially ill-conditioned: the condition number $\kappa \sim (1/2)\exp\left(\frac{\pi^2}{4\alpha^2}\right)$ for a uniform, one-dimensional grid, and larger still in two dimensions or when the grid is irregular. Furthermore, the Gaussian RBF interpolant is ill-conditioned for most series in the sense that the interpolant is the small difference of terms with exponentially large coefficients. Fornberg and Piret developed a "QR-basis" that ameliorates this difficulty for approximations on the surface of a sphere, but because the recombined basis functions are perturbed spherical harmonics, not Gaussians, the Fast Gauss Transform is no longer applicable. The solution of the interpolation matrix system by a preconditioned iteration is less sensitive to condition numbers than direct methods because iterations are self-correcting and also because the preconditioning reduces the spread of the eigenvalues. However, each iteration requires a matrix–vector multiply which is fast only if this operation can be performed by some species of Fast Summation. When $\alpha \sim O(1)$, alas, we show that Ω is not large and the Fast Gauss Transform is not accelerative. Gaussian RBFs are unusual among RBF species through the absence of long-range interactions due to the exponential decay of the Gaussians with distance from their centers; many other RBF species do have long-range interactions, and it is well-established that these can be accelerated by fast multipole and treecode algorithms. We offer a less rigorous scale analysis argument to explain why the underlying difficulty in accelerating short-range interactions is not peculiar to the Gaussian RBF basis or to the Fast Gauss Transform, but rather is likely to be a generic difficulty in accelerating the short-range interactions of almost any RBF basis with almost any Fast Summation.

1. Introduction

Radial basis functions (RBFs) have been successfully applied to solve many species of partial differential equations. However, most of these applications have been restricted to rather small N where N is the total number of degrees of freedom. For $N \leqslant 4000$, it does not matter that the RBF interpolation, differentiation and summation matrices are dense $N \times N$ matrices

* Tel.: +1 734 764 3338; fax: +1 734 764 5137.
E-mail address: jpboyd@umich.edu

0021-9991/$ - see front matter © 2009 Elsevier Inc. All rights reserved.
doi:10.1016/j.jcp.2009.10.032

Table 1
List of symbols.

a_n	nth Chebyshev coefficient
d	Spatial dimension
FGT	Fast Gauss Transform
h	Average grid spacing
IFGT	Improved (Taylor series-based) Fast Gauss Transform
m	Radius from an RBF center in units of h
N_B	Number of RBFs with centers in a given block
p	Degree of Hermite series or Chebyshev series
$\tilde{p}$	$p-1$, one less than the degree of a Taylor series
P	One less than the degree of Taylor series
r	$(\equiv \epsilon s/\sqrt{2})$ length of a side of a block relative to the RBF e-folding scale
$\mathcal{R}$	Parameter in Taylor series-based IFGT
q	Exponent of Generalized Multiquadric
s	Length of one side of the block enclosing a cluster of RBFs
x	Coordinate (one dimension)
$\mathbf{x}$	Coordinate (two or more dimensions)
TGT	Truncated Gauss Transform
α	$(=\epsilon h)$ inverse width relative to the grid spacing
ϵ	Absolute inverse width of the RBFs
λ_j	Coefficient of jth term in RBF series
$\phi(r)$	RBF function
ρ	$(\equiv r\sqrt{\exp(1)/p})$ parameter that must be small for small Hermite series error

because these can still be inverted or multiplied in a few minutes on a workstation. Because of its ease of programming, RBFs are a great "small-N" method. Applications to "high-N" problems will be prohibitively costly, however, unless some analogue to the Fast Fourier Transform can be found.

Truncation, short-range interactions and long-range interactions offer what military strategists call "lines of attack". If the RBFs are spatially localized, as true of Gaussians, then great savings can be realized in summation at a point $\mathbf{x}$ by discarding the very tiny contributions from RBFs whose peaks are sufficiently distant from $\mathbf{x}$. This gives the "Truncated Gauss Transform" for Gaussians, and similarly for other RBFs that decay exponentially away from their peaks such as sech's.

Fast Summation methods, which we shall use as a shorthand for a broad family of algorithms that include the Fast Multipole Method, Fast Gauss Transform, and treecodes, are designed to evaluate a summation of N terms at each of L points where $L \geqslant O(N)$. Instead of evaluating the sum at a given point independently of all the other points at a cost of $O(N)$ operations per point, Fast Summations share work between the sums at different points so as to greatly reduce the cost. Many species of Fast Summations including the Fast Gauss Transform replace interactions among particles, vortices or radial basis functions by a *proxy* series, a sort of mathematical stunt double. Some Fast Summations attack only long-range interactions, some only short-range interactions, and some both kinds of interactions.

For example, the earliest Fast Summation method, the "Fast Multipole Method", was invented to accelerate computation of the long-range gravitational interactions among N stars. The stars — RBFs in our application — are grouped into clusters containing N_B elements; the combined influence of all the stars in the group is approximated by a series of negative powers of the distance ("multipole expansion") from a group of stars to the evaluation point. If the proxy series can be truncated to only M terms where $M \ll N_B$, then the summation is accelerated by a factor of $\Omega \equiv M/N_B$. (A full list of symbols is given in Table 1.) Treecodes are similar but use a Taylor series as a proxy [33].

Because both Fast Summations and RBFs have bloomed into a bewildering variety of genera and species, we shall focus primarily on two summation methods for one type of RBFs: the Fast Gauss Transform (FGT) and Truncated Gauss Transform (TGT) for Gaussian RBFs. The FGT and TGT are illuminating to analyze because both are particularly tailored to Gaussians. Gaussian RBFs are useful examples because (i) they are popular [45,51,52,54,12,13,30,34,37,49,59,3],[1] widely used and converge exponentially fast for smooth functions $f(x)$ and (ii) have *only* short-range interactions, allowing us to focus on that part of summing RBFs where Fast Summations have difficulties. (As we shall explain later, long-range interactions, for those RBF species that have them, are well accelerated by some Fast Summations.)

In d dimensions where d is an arbitrary positive integer, a Gaussian RBF approximation to a function $f(x)$ has the form:

$$f(\mathbf{x}) = \sum_{j=1}^{N} \lambda_j \exp(-(\alpha/h)^2 \ \|\mathbf{x}-\mathbf{x}_j\|_2^2) \tag{1}$$

where $\mathbf{x}$ is a point in R^d, the λ_j are the RBF coefficients and the $\mathbf{x}_j$ are the "centers", α is a user-chosen constant ("inverse width parameter") and h is the *average* grid spacing; one of the charms of RBFs is that the grid can be *irregular*. Because the basis functions are not orthogonal, the RBF coefficients λ_j are almost always found by interpolation at a set of interpolation points

[1] The *Web of Science* returned 197 articles for the keyword "Gaussian radial basis function". We do not cite them all, but it should be noted that Gaussian RBFs are *very* popular in neural networks.

that usually (though not necessarily) coincide with the "centers". (A cruder alternative is quasi-interpolation, discussed in Section 3.)

As noted earlier, Fast Summations have proved very effective in accelerating long-range interactions for those RBF species that have them. By "long-range" interactions we mean contributions to the sum at a point $\mathbf{x}$ which come from RBFs whose centers are sufficiently far from $\mathbf{x}$. The success of Fast Summations for long-range interactions is not impeded by the RBF width parameter α: the Fast Multipole method was devised for gravitational interaction of stars, which correspond to Inverse Quadratic RBFs with $\alpha \to \infty$.

For Gaussians, though, the long-range contributions are exponentially small and therefore ignorable. For Gaussian RBFs, summation can be accelerated (beyond the speed-up of omitting long-range interactions) only by successfully finding a good proxy for the *short-range* contributions to the sum.

Inspired by the success of Fast Summations in other context, Torres and Barba [56], who developed an inner–outer iteration for computing Gaussian RBF interpolants, write "we have not at this time implemented a fast summation method; ...with a Fast Gauss Transform implementation, the evaluation of the summations is $O(N)$." Here, we show that this optimism is unfounded: despite the claims of Livne and Wright [38] and Fasshauer [19], the Fast Gauss Transform does not produce acceleration except for unusably small α.

We must make a clear distinction between long-range and short-range interactions because Fast Summations are in fact accelerative for long-range RBF interactions as discussed more below. It is the short-range interactions that are the bother. However, because of their exponential decay with $\|\mathbf{x} - \mathbf{x}_j\|$, Gaussian RBFs have *only* short-range interactions, and for the Fast Gauss Transform to fail for them is to fail completely.

We must note immediately an important caveat: the Fast Gauss Transform has been a brilliant success in many applications outside RBFs, and nothing in this article argues to the contrary. The *width* α of the Gaussians turns out to be extremely important: when the Gaussians are wide compared to the average grid spacing (small α), the Fast Gauss Transform is highly effective. In many applications, α is intrinsically small and the coefficients λ_j are derived from the Green's function of the heat equation or some other source which does not require interpolation. Unfortunately, as we shall discuss in detail below, the interpolation matrix for RBFs is exponentially ill-conditioned as α decreases, and therefore Gaussian RBFs cannot be used in *Gaussian form* for α sufficiently small so that the Fast Gauss Transform is accelerative.

2. The Fast Gauss Transform

Greengard and Strain [25] developed the Fast Gauss Transform specifically to sum series of Gaussians. Strain [55], Baxter and Roussos [2], Kobayashi [32] and Wan and Karniadakis [57] have made improvements. The classical form of the Fast Gauss Transform used multiple Hermite function series of low degree as a global proxy for the Gaussian series; in other words, Hermite series were the proxy for computing all interactions. In contrast, some other species in the multipole/tree-code family use different proxies for long-range and short-range interactions.

Greengard and Sun [26] replaced the Hermite proxy by plane waves. However, this was published in an obscure journal that is not archived by the Web of Science and we have not been able to find later applications, so we shall not discuss this variant here.

Yang et al. [60,61,50] have developed an Improved Fast Gauss Transform (IFGT) which replaces the Hermite expansions of the FGT by Taylor series. Although also aimed at high dimensions, it works well for $d \leqslant 3$. Mittelman and Miller [43] developed better error bounds for the IFGT. We shall analyze the efficiency of both the Greengard–Strain FGT and IFGT in detail in later sections.

Mittelman and Miller [42] have developed a new Fast Gauss Transform that employs the Singular Value Decomposition (SVD), primarily targeted at high dimension d. (In two dimensions, their method is equivalent to replacing a matrix–vector multiplication by a series of M vector inner products that are the truncation of the SVD matrix to its M rank-one components of largest singular values.) However, they abandoned it in favor of their enhanced IFGT, so we shall not discuss their SVD Gauss Transform further.

All flavors of Fast Summation have several stages; in particular, there is a non-trivial cost to compute the coefficients of the proxy series. To bore in on the fundamental difficulty, we shall analyze just a single issue: the *size* of the proxy series relative to the *size* of the original, Gaussian series. Obviously, if the proxy summation is longer than the original, then it is time for the "fast" summation to run up the white flag.

3. RBF background: condition number of the interpolation matrix and restrictions on α

3.1. Anti-Gaussian growth of the condition number as $\alpha \to 0$

The elements of the Gaussian interpolation matrix $\mathbf{G}$ are

$$G_{jk} = \exp(-(\alpha/h)^2 \ \|\mathbf{x}_j - \mathbf{x}_k\|^2) \tag{2}$$

For expository simplicity, specialize to $d = 1$, and the uniform grid defined by

$$x_j \equiv hj \tag{3}$$

where h is the grid spacing. The elements of the interpolation matrix are

$$G_{jk} = \exp(-\alpha^2\ |i-j|^2) \tag{4}$$

Note that the matrix is independent of h and depends only on the relative inverse width α. As $\alpha \to 0$, the RBFs become flatter and flatter. The condition number grows because all $G_{jk} \to 1$, forcing the matrix to be singular in the limit.

Boyd and Gildersleeve [9] show that for a uniform grid, the condition number κ of the interpolation matrix is *independent* of N for large N. κ asymptotes exponentially fast as N increases to

$$\kappa_1(\alpha, N) \sim (1/2)\exp\left(\frac{\pi^2}{4\alpha^2}\right) \quad \text{[Condition Number : 1D Uniform Grid]} \tag{5}$$

$$\kappa_2(\alpha, N) \sim (\kappa_1(\alpha, N))^2 = (1/4)\exp\left(\frac{\pi^2}{2\alpha^2}\right) \quad \text{[Cond. Number : 2D Uniform Grid]} \tag{6}$$

These numerical asymptotics confirm the *tight* upper bounds proved (in any number of dimensions) for a uniform grid by Baxter [1].

The logarithm of the condition number estimates for *irregular* grids, reviewed in [58,15], are much larger than for uniform grids. Boyd and Gildersleeve's numerical results show that grid irregularity does indeed have a lamentable effect on condition number. Baxter's tight bounds on κ for uniform grids should be regarded as lower bounds for κ on irregular grids.

The two-dimensional uniform grid condition number is larger than the reciprocal of standard single precision arithmetic ("machine epsilon", 2.2×10^{-16}), for all $\alpha < 0.36$. The analogous "danger point" in one space dimension is at $\alpha = 0.26$.

3.2. Ill-conditioning of the coefficients as $\alpha \to 0$

The situation is actually even worse than the condition number implies because generically the Gaussian RBF coefficients λ_j become exponentially large compared to the norm of $f(x)$ as $\alpha \to 0$. (A review is given in [11]). It is then necessary to compute the RBF coefficients to very high accuracy because the sum of the RBF series is the tiny difference of exponentially large terms. Even if the coefficients are computed without inverting the interpolation matrix, such as by using the coefficient integrals that are available for a *uniform* grid only [15], the Fast Summation must be *extremely* accurate.

In one dimension with the centers on a uniform grid, it is possible to bypass the Gaussian series entirely. On an infinite interval, one can use the approximate RBF cardinal function [10,41]. On a finite interval, Platte and Driscoll showed that the Gaussian RBF interpolation can be converted by a change of coordinate into a polynomial interpolation problem on a transformed, nonuniform grid [45]. These devices make the FGT unnecessary, but unfortunately do not generalize to higher dimensions or to nonuniform grids in any dimension.

3.3. Quasi-interpolation

Beale and Majda showed that Gaussians, or Gaussians multiplied by polynomials of low degree, could yield quasi-interpolants of arbitrary order [4,5]. Developments are reviewed in [41,19]. In quasi-interpolation, it is unnecessary to solve a linear system with the interpolation matrix: the RBF coefficients are

$$\lambda_j^{QI} = \frac{\alpha}{\sqrt{\pi}}\, f(x_j) \tag{7}$$

Unlike interpolation, the coefficients in quasi-interpolation are always $O(\|f\|_\infty)$.

However, there is a penalty for quasi-interpolation: for the case shown in Fig. 1, classical interpolation with just thirty points is more accurate than even sixth order quasi-interpolation with 500. Furthermore, the errors rise steeply as α decreases. Thus, the small-α regime is undesirable for quasi-interpolation just as for interpolation.

3.4. Remedies for large condition number

Strategies for defeating high condition number are numerous as reviewed by [28,19]. However, all have baggage.

One option is to replace the usual Cholesky factorization by the Singular Value Decomposition (SVD) [44]. For sufficiently small α, even SVD-filtering fails and more complicated variations have been tried [18]. A grave disadvantage is that an SVD is typically thirty times more expensive than a Cholesky factorization.

Another option is to split the problem into multiple domains. Function-adaptive grids can reduce N; smaller matrices are better conditioned than large matrices. However, subdomains destroy the simplicity of global RBFs: if one has to cope with the bookkeeping of many subdomains, why not use the well-established finite element or high order spectral element methods?

Because RBF interpolation matrices are often dense matrices, the cost of direct solution of the interpolation matrix problem is $O(N^3)$ where N is the total number of grid points. This is okay for small to moderate N, but prohibitively costly when N is large. This has inspired great interest in solving the interpolation problem *iteratively* [19,14,56,20,27,29,36,35,39,54].

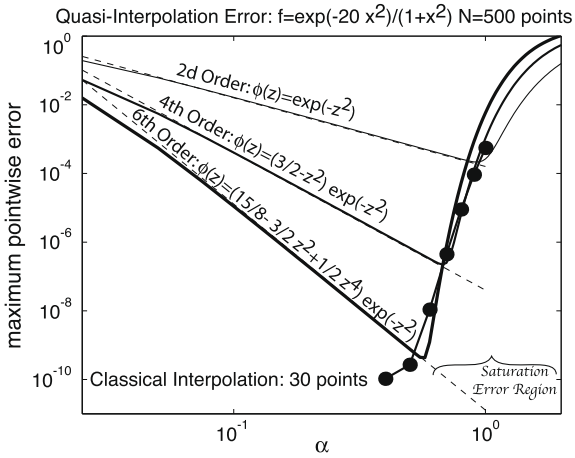

Fig. 1. Gaussian RBF quasi-interpolation of $f(x) \equiv \exp(-20x^2)/(1+x^2)$ using 500 uniformly spaced points: errors in the L_∞ norm plotted versus α. For comparison, the thick curve with large disks shows the errors in standard RBF interpolation, also on an evenly-spaced grid, but using just thirty grid points. For $\alpha > 3/5$ or so, the errors of both interpolation and quasi-interpolation are roughly equal to the infinite interval saturation error, $E_S(\alpha) \sim 4\exp(-\pi^2/\alpha^2)$, discussed later in this article. The error in quasi-interpolation rises inversely with the order of quasi-interpolation as $\alpha \to 0$. The dashed lines are power-law guidelines which are almost indistinguishable from the quasi-interpolation errors they approximate: $40/(N^2\alpha^2)$, $2500/(N^4\alpha^4)$ and $200,000/(N^6\alpha^6)$ from top to bottom.

Modern iterative methods are almost always implemented with the aid of a preconditioning matrix because an unpreconditioned iteration converges very slowly. The rate of convergence is proportional to the ratio of the largest eigenvalue to the smallest eigenvalue of the matrix; for a symmetric matrix, this ratio is the condition number. The preconditioner is chosen so that the product of the inverse of the preconditioner with the original matrix (here the interpolation matrix) has a much smaller condition number. Unfortunately, when the interpolation matrix is ill-conditioned, its preconditioner must be also ill-conditioned in order to cluster the eigenvalues of the matrix product. Thus, reducing condition number through preconditioning is a hope that "two wrongs make a right". The self-correcting nature of an iteration makes this true, at least to some extent. However, there has not been a careful study of how much α can be safely reduced by replacing direct methods with preconditioned iterations. In the absence of such, and in view of the ill-conditioning of the Gaussian RBF sum itself, we assert that small α is still risky.

Fornberg and Wright [23] developed the contour-Padé algorithm. This bypasses both the RBF sum and interpolation matrix and computes the RBF interpolant accurately even for very small α. Unfortunately, ill-conditioning and cost limit it to $N < 100$. In the words of Platte and Driscoll, p. 762 of [45], ""Fornberg and Wright recently presented a contour-integral approach that allows numerically stable computations of RBF interpolants for all values of the free parameter, but this technique is expensive and has been applied only for experimental purposes." The developers themselves acknowledge this in a later paper: "Although this [Contour-Padé] algorithm formed a very successful tool for discovering and further exploring several key features of RBF approximations, it was limited to a relatively low number of data points ($N < 200$ in 2D) [22]." For such small N the Fast Gauss Transform is irrelevant.

Fornberg and Piret [22] have more recently developed a "QR-basis", which is successful for N up to several thousand. (This is so far restricted to the surface of a sphere although there seems no reason why it cannot be generalized to other geometries.) However, the new basis functions are linear combinations of RBFs which are constructed to be perturbed spherical harmonics. Because the recombinant basis is no longer a set of Gaussians, the Fast Gauss Transform is *inapplicable*.

Therefore, we must be precise in our language. It is premature to conclude that Gaussian RBFs are useless for small α. In the rest of this article, we assert that Gaussian RBFs are useless for small α in *Gaussian form*, that is, when the RBF approximation is written in a form appropriate for the Fast Gauss Transform. To be more precise, Gaussian RBFs in Gaussian form will never be applied for $N > 100$ for $\alpha < 1/4$ in one dimension, $\alpha < 1/3$ in two dimensions and probably only for still larger α in three dimensions. This constraint on α is very important in understanding the limits of the Fast Gauss Transform as applied to Gaussian RBFs.

3.5. Error saturation

The Truncated Gauss Transform, described below, becomes cheaper and cheaper as α becomes large. Unfortunately, it is not possible to use large α. In the limit $N \to \infty$ for fixed α, the error of the RBF approximation does not tend to zero but rather to a non-zero quantity $E_S(\alpha)$ which is called the "saturation error".

For Gaussians, the saturation error is [40,11,8]:

$$E_S(\alpha) \sim \begin{cases} O(4\exp(-\pi^2/\alpha^2)\|f\|_\infty) & \text{[Uniform Infinite Grid]} \\ O(\exp(-\pi^2/[4\alpha^2])\|f\|_\infty) & \text{[Nonuniform Grid]} \\ O(\exp(-0.47/\alpha^2)\|f\|_\infty) & \text{[Uniform Finite Grid]} \end{cases} \tag{8}$$

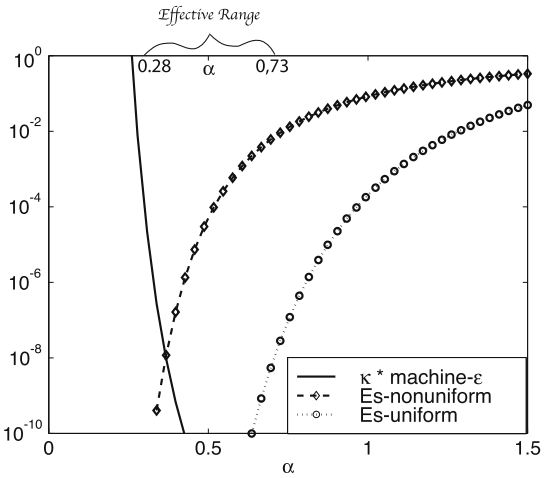

Fig. 2. The solid curve is the condition number of the Gaussian RBF interpolation matrix, scaled by multiplication by "machine epsilon", $\epsilon_{machine} = 2.2 \times 10^{-16}$. The upper and lower curves with diamonds and circles show the saturation error on a nonuniform grid and a uniform grid, respectively. If we define the "effective range" as that where simultaneously $\kappa(\alpha) \times \epsilon_{machine} > 0.01$ and $E_S^{nonuniform}(\alpha) < 0.01$, α must lie on $\alpha \in [0.28, 0.73]$ as marked.

This inverse relationship between condition number (growing as $\alpha \to 0$) and saturation error (growing as $\alpha \to \infty$) is illustrated in Fig. 2.

4. Other fast summations for Gaussian RBFs

Roussos and Baxter [53] use the Fast Gauss Transform as an auxiliary technology to rapidly sum *non-Gaussian* RBF species such as multiquadrics and inverse multiquadrics. Because their article excludes Gaussian RBF summations as a goal, it is not directly relevant here.

Potts et al. have developed a Fast Summation for Gaussians (and other RBFs) [47,46]. This employs the Nonuniform Fourier Transform (NUFFT) [48] devised by Boyd [6] and later Dutt and Rohklin[16,17]. The key idea is to replace the Gaussian RBF (called the "kernel" $\mathcal{K}$) by its Fourier series, interchange the order of summation, compute the inner sum for the coefficients a_m by a Nonuniform FFT, and then sum the remaining Fourier series by another NUFFT:

$$f_{RBF}(y_k) = \sum_{j=1}^{N} \lambda_j \mathcal{K}(|y_k - x_j|) \tag{9}$$

$$= \sum_{j=1}^{N} \lambda_j \left\{ \sum_{m=-N_F/2}^{N_F/2-1} b_m \exp(2\pi i m\ (y_k - x_j)) \right\} \tag{10}$$

$$= \sum_{m=-N_F/2}^{N_F/2-1} b_m \left\{ \sum_{j=1}^{N} \lambda_j \exp(2\pi i m\ (-x_j)) \right\} \exp(2\pi i m\ y_k)) \tag{11}$$

$$= \sum_{m=-N_F/2}^{N_F/2-1} b_m a_m\ \exp(2\pi i m\ y_k)) \tag{12}$$

$$= \sum_{m=-N_F/2}^{N_F/2-1} d_m\ \exp(2\pi i m\ y_k)) \tag{13}$$

For expository simplicity, we have written the series in one-dimensional form, but every line applies in multiple dimensions if x, y, m and j are interpreted as vectors.

In one dimension, the periodized approximation is a Jacobi theta function of period P.

$$\mathcal{K} \approx \sum_{-\infty}^{\infty} \exp\left(-\frac{\alpha^2}{h^2}[x - mP]^2\right) \tag{14}$$

$$= \frac{\sqrt{\pi}h}{P\alpha}\theta_3\left(\frac{\pi}{P}x; q = \exp\left(-\frac{\pi^2 h^2}{P^2\alpha^2}\right)\right) \tag{15}$$

$$= \frac{\sqrt{\pi}h}{P\alpha} \sum_{n=-\infty}^{\infty} \exp\left(-\frac{\pi^2\ h^2 n^2}{P^2\alpha^2}\right) \exp\left(i\frac{2\pi}{P}nx\right)$$

If the interpolation points and centers lie on $[-1,1]$, then $\max|y_k - x_j| \leqslant 2$, which requires P slightly greater than 4 so that the theta function is not evaluated too close to the edges of the periodicity interval at $[-P/2, P/2]$. Recalling that $h \approx 2/N$ for a uniform grid of length 2,

$$b_{N_F/2} \approx \exp\left(-\frac{\pi^2}{16\alpha^2}\frac{N_F^2}{N^2}\right) \tag{16}$$

It follows that the truncation error of the Fourier approximation is less than or equal to the saturation error only if N_F is the same order-of-magnitude as N. However, smaller α increases the rate of convergence of the theta function Fourier series and allows a smaller Fourier truncation N_F.

Thus, the Fourier proxy for the RBF series does not much reduce the length of the sums. The primary advantage of the Potts et al. procedure is that transformed sums can be performed by using the NUFFT at a cost of $O(L\log_2(N))$ operations instead of $O(LN)$. Although the NUFFT has a big proportionality constant compared to the FFT, the NUFFT procedure is worthy of further study.

Keiner et al. have developed a Fast Summation for Gaussians (and other RBF species) on the sphere [31], but because of their geometry and the similarity to the Potts et al. work, we shall not discuss this article further.

Livne and Wright [38] claim that their multilevel algorithm is much faster than direct summation for many species of RBFs. One key idea is the same as in the scheme of Potts et al. [47]: expand ϕ in a series, reverse the order of summations, and collect terms. However, Livne and Wright expand $\phi(y_k - x_j)$ in just the second coordinate (centers) x_j as a series of ϕ on a *coarser* grid with points X_J, truncating the coarse sums to just the p nearest neighbors of x_j (the set σ_j):

$$f_{RBF}(y_k) = \sum_{j=1}^{N}\lambda_j\phi((\alpha/h)|y_k - x_j|) \tag{17}$$

$$= \sum_{j=1}^{N}\lambda_j\left\{\sum_{J\in\sigma_j}\omega_{jJ}\phi((\alpha/h)|y_k - X_J|)\right\} \tag{18}$$

$$= \sum_{J\in\sigma_j}\phi((\alpha/h)|y_k - X_J|)\left\{\sum_{j=1}^{N}\lambda_j\omega_{jJ}\right\} \tag{19}$$

$$= \sum_{J=0}^{N_{coaarse}}\phi((\alpha/h)|y_k - X_J|)\left\{\sum_{j\ :\ J\in\sigma_j}\lambda_j\omega_{jJ}\right\} \tag{20}$$

$$= \sum_{J=0}^{N_{coaarse}}\Lambda_J\phi((\alpha/h)|y_k - X_J|) \tag{21}$$

(The dependence on y_k is handled similarly, but we defer the details to [38].)

The smoother ϕ — the smaller α — the more rapidly the coarse grid expansion will converge. Thus, like the Fast Gauss Transform, their algorithm is strongly accelerative for $\alpha \ll 1$. In their numerical experiments, they set $\alpha = (1/4)/\sqrt{N}$; for a uniform grid in the unit square, $h = 1/N^{1/d}$. In their Table 6.2, the largest value of α is only 0.044. Therefore, the condition number κ of the interpolation matrix, using Baxter's bound as the asymptotic approximation that it is shown to be in [9], is no smaller than 10^{550}! It is quite impossible to work with Gaussian RBF series in Gaussian form for such small α.

Table 35.1 of Fasshauer's very readable book shows timing tests for the Fast Gauss Transform for $\alpha = 1/2$ for quasi-interpolating Gaussian series using software written by Adam Florence. The speed-ups are impressive. However, the comparison is between *untruncated* summation, which is very bad and an FGT routine that implements truncation [21]. Our analysis below shows that for α so large, Hermite condensation fails; all the speed-up is coming from the Truncated Gauss Transform described in the next section.

5. Truncated Gauss transform (TGT)

It is very common in the literature to compare the FGT to direct, *untruncated* summation. For Gaussians, such comparisons are foolish because no one but a blockhead would actually use untruncated summation because the TGT is very simple and, for moderate and large α, vastly faster.

Suppose that the user chooses to retain all terms greater than or equal to a user-specified tolerance δ. To sum the RBF series at a point $\mathbf{x}$, we must then include all terms whose centers lie sufficiently close to the target point that $\exp(-[\alpha/h]^2\|\mathbf{x} - \mathbf{x}_j\|^2) \geqslant \delta$. Let m denote the maximum of $\|\mathbf{x} - \mathbf{x}_j\|/h$, the distance to $\mathbf{x}$ in *units of the grid spacing* h, which is included in the sum. Then in any number of dimensions d,

$$\delta = \exp(-\alpha^2 m^2) \ \leftrightarrow\ m = \frac{\sqrt{-\log(\delta)}}{\alpha} \tag{22}$$

Noting that each term in the truncated sum incurs a cost of one multiplication and one addition, and using the familiar formulas for the area of a disk (πm^2) and volume of a sphere ($(4/3)\pi m^3$), we obtain the cost estimates in Table 2.

Table 2
Cost in operations of summing terms greater than or equal to a tolerance δ in Gaussian RBF sums.

Dimension d	General	$\alpha = 1/2, \delta = 2.2 \times 10^{-16}$	FFT
1	$4\sqrt{-\log(\delta)}/\alpha$	48	66
2	$2\pi(-\log(\delta))/\alpha^2$	900	132
3	$(8/3)\pi(-\log(\delta)^{3/2})/\alpha^3$	1450	198

For comparison, we have also included cost estimates for the Fast Fourier Transform, which is the Fast Summation employed with both Fourier series and Chebyshev polynomials. The floating point operation count of a one-dimensional FFT of a real-valued vector of length N asymptotes to $2.5N\ \log_2(N)$; a complex-valued argument doubles the cost. In applications to differential equations, a mix of real and complex FFTs are required; to make the comparison as favorable as possible for RBFs, we assume that all FFTs are complex-valued. Fig. 3 shows that FFT cost tends to its asymptote rather slowly, so again to be fair to RBFs, we empirically substituted 3.3 (dashed-dot line) for the asymptotic constant, 2.5.

The logarithmic factor in the FFT cost is a little awkward. However, $\log_2(N) \leqslant 10$ for all $N < 1000$ and grows very slowly for larger N. Multidimensional transforms are performed as a sequence of one-dimensional transforms, first in one coordinate, then in each of the others. Thus, the cost of a real-valued FFT is less than $33N$ for all $N \leqslant 1000$ in one dimension, less than $66N$ for all $N \leqslant 1,000,000$ in two dimensions and less than $99N$ for all $N \leqslant 10^9$ in three dimensions, and double that for the FFT of a complex-valued vector. These bounds are used in the table.

Truncating Gaussian sums is vastly more efficient than untruncated summation: with $N = 10^6$ in three dimensions, the $O(2N^2)$ cost of the untruncated series is more costly than the truncated sum for $\alpha = 1/2$ by a factor of 1300!

It should be noted that Fast Gauss Transform routines frequently truncate before computing proxies for the untruncated remainder. Some of the observed acceleration reported for large α relative to untruncated summation is really due to truncation rather than to any merit of the proxy expansion, as true of Table 35.2 of [19].

In one dimension, the Truncated Gauss Transform compares favorably with the FFT, at least for $\alpha \approx 1/2$. Unfortunately, in higher dimensions, the comparison is very unfavorable.

Nevertheless, it would be premature to conclude that Gaussian RBF methods are hopelessly inferior to classical spectral algorithms. Chebyshev and Fourier methods can be made adaptive by a change-of-coordinate, but only when the domain is simple rather than convoluted, and only when the high gradient features (shocks, fronts, combustion zones) have a simple topology. RBFs are meshless, and in principle can be applied in very complicated geometry, and also can cluster grid points in high gradient regions even if these have a complicated topology. What we can say is that the adaptivity of Gaussian RBFs must be *very* good — not a little better, but very much better — in order to match the efficiency of a less-flexible-but-much-faster-for-a-given-N Fourier or Chebyshev method.

Alternatively, of course, Gaussian RBF methods can close the "Fast Summation" gap by using algorithms better than the Truncated Gauss Transform. In the remainder of this article, we show that this is unfortunately not possible.

6. Analysis of the Fast Gauss Transform of Greengard and Strain

6.1. Hermite condensation

The first part of the Fast Gauss Transform is the "Hermite condensation": a group of N_B Gaussians of the form $\exp(-[\alpha/h]^2\ \|\mathbf{x} - \mathbf{x}_j\|^2)$ in a box whose sides are of length s is approximated by a Hermite series with $(p+1)^d$ terms as shown in Fig. 4. This description is a little misleading because the *absolute* width of the Gaussians and of the boxes is *irrelevant* to the Hermite series error and therefore to the success or failure of the Fast Gauss Transform. Greengard and Strain [25] therefore, introduced a *relative* box size r defined by

$$r \equiv \alpha \frac{s}{h\ \sqrt{2}} \tag{23}$$

This parameter is the length of the side of a box *relative* to the decay scale of the Gaussians. The smaller r is, the slower the variation of the RBFs across the width of a box and therefore the greater the accuracy of the Hermite series.

6.2. Compression factor

The Hermite condensation step replaces the N_B Gaussian basis functions by a total of $(p+1)^d$ Hermite functions:

$$u_B(x) \equiv \sum_{j=1}^{N_B}\ \lambda_j \exp(-\epsilon^2\ \|\mathbf{x} - \mathbf{x}_j\|^2) \approx \sum_{m=0}^{p}\sum_{n=0}^{p} a_{mn} \exp(-x^2 - y^2)\ H_n(x) H_n(y) \tag{24}$$

where $u_B(x)$ denotes that part of $u(x)$ which is given by the sum of the Gaussians in the cluster, the origin is at the center of the cluster, and $H_n(x)$ denotes the usual Hermite polynomial of degree n. For simplicity, we displayed the $d = 2$ sum, but with multinomial notation, the Fast Gauss Transform can be applied in an arbitrary number of dimensions. Ignoring the opera-

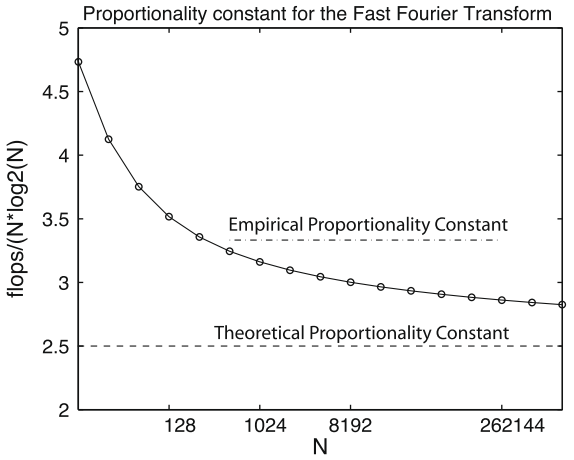

Fig. 3. The proportionality constant for the Fast Fourier Transform of a real-valued vector of length N as measured in Matlab 5.2. (Later versions of Matlab have improved FFT codes, but can no longer count floating point operations.) The long, lower dashed line is the theoretical asymptote as $N \to \infty$.

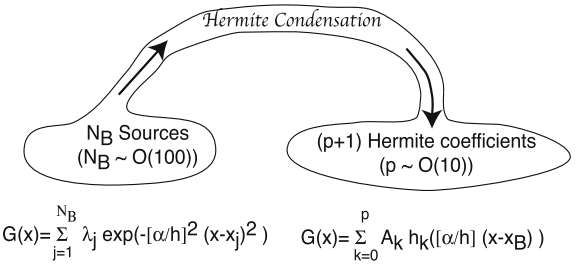

Fig. 4. Schematic of the first stage of the Fast Gauss transform. The "Hermite condensation" step is performed separately for each box or cluster.

tions required to compute the Hermite coefficients implies that the speed-up possible in this step is bounded by the "compression factor" Ω:

$$\Omega \equiv \frac{N_B}{(p+1)^d} \quad \text{[Compression Factor]} \tag{25}$$

Our goal is to make Ω as large as possible while still keeping the error in the Hermite series below some specified error tolerance δ.

6.3. Error parameter

The error bounds derived by Greengard and Strain [25] and improved by Wan and Karniadakis [57] and Baxter and Roussos [2] depend on a parameter

$$\rho \equiv r\ \sqrt{\exp(1)/p} \tag{26}$$

To express this in terms of the compression factor Ω, we need a bit of algebra.

The grid spacing h in dimension d is

$$h = s/(N_B^{1/d} - 1) \ \leftrightarrow\ s = h(N_B^{1/d} - 1) \tag{27}$$

Substituting this expression for the box side length s into the definition of the relative side length gives

$$r = \alpha\ (N_B^{1/d} - 1)/\sqrt{2} \tag{28}$$

It is convenient to replace N_B by Ω by means of

$$N_B = (p+1)^d\ \Omega \tag{29}$$

Then $r = \alpha\ ((p+1)\Omega^{1/d} - 1)/\sqrt{2}$ and the error parameter is

$$\rho = \alpha\ \sqrt{\exp(1)/2}\Big((p+1)\Omega^{1/d} - 1\Big)/\sqrt{p} \tag{30}$$

6.4. Error bound

Wan and Karniadakis prove that the relative error in approximating the Gaussian series [57] is bounded by

$$E(p,\alpha,\Omega) = \sum_{k=1}^{d} \frac{d!}{(d-k)!k!} \left(Kp^{-1/4}\frac{\rho}{1-\rho}\right)^k \tag{31}$$

where $K = 0.69$. The terms for $k > 1$ are rather small, so there is no harm in dropping them: the only consequence is to make the Fast Gauss Transform look a little better. We shall therefore, assume that the accuracy is satisfactory if E_d is less than δ where

$$E_d(p,\alpha,\Omega) \equiv \; d\; Kp^{-1/4}\frac{\rho}{1-\rho} \tag{32}$$

Fig. 5 shows how E_d varies with Hermite truncation p and the compression factor Ω for dimensions one and two and two different values of α. The spatial dimension has only a modest effect; the error is controlled primarily by the RBF inverse width α. Only when $\alpha = 1/8$ are there small regions in the Ω–p plane where the relative error bound is less than 1%. However, these regions are restricted to $\Omega < 2$. For $\alpha = 1/4$, which is the lower limit of what one would use in practice, the Hermite condensation fails utterly to simultaneously provide accuracy and compression.

In this analysis, we have ignored later steps in the Greengard–Strain transform, such as that in which the Hermite series from various boxes are condensed into a single Taylor series that approximates the entire RBF sum within a limited domain in x. However, if the Hermite step produces no compression, then it would be more logical to apply a Taylor series directly to the Gaussian sum itself. This is precisely what is done in the Fast Gauss Transform of Yang et al. which is analyzed in the next section.

7. Analysis of the Improved Fast Gauss Transform of Yang et al.

Yang et al. [60,61,50] developed an Improved Fast Gauss Transform (IFGT) which replaces the Hermite expansions of the FGT by Taylor series:

$$u_B(x) \equiv \sum_{j=1}^{N_B} \lambda_j \exp(-(\alpha/h)^2\; \|\mathbf{x}-\mathbf{x}_j\|^2) \;\approx \sum_{m=0}^{\tilde{p}} c_m \left(\frac{\alpha}{h}(x-x_*)\right)^n \tag{33}$$

where x_* is the center of the evaluation box and m is interpreted as a multinomial index and $\tilde{p}$ is one less than the degree (in one dimension) or the total degree when the dimension $d > 1$. This is a slight notational change from $p =$ degree used in earlier sections. Their error bound is

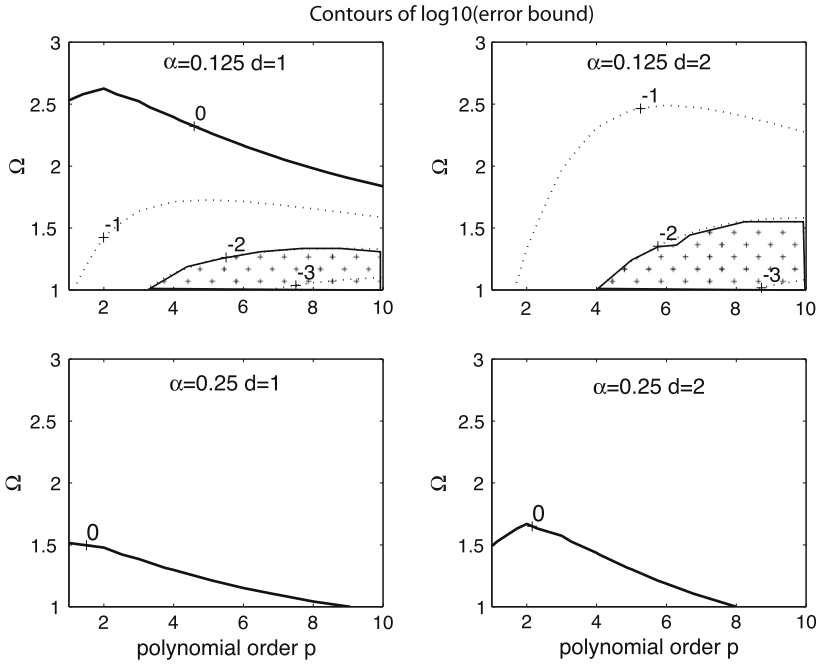

Fig. 5. Isolines of the base-10 logarithm of the error bound for two different values of the RBF inverse width parameter α and two different spatial dimensions, plotted versus the truncation p of the Hermite series (horizontal axes) and the compression factor Ω (vertical axes).

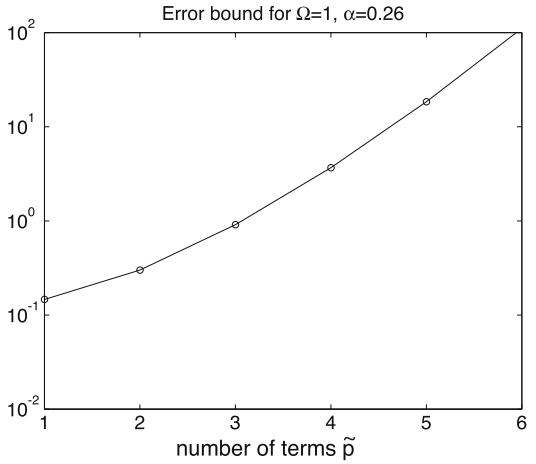

Fig. 6. A plot of the error bound $\mathcal{G}(\alpha, p)$ for the Improved Fast Gauss Transform for $\alpha = 0.26$. The error increases with $\Omega > 1$ and also for larger α.

$$|E| \leqslant \Lambda \left(\frac{2^{\tilde{p}}}{\tilde{p}!} \mathcal{R}^{2\tilde{p}} + \exp(-\mathcal{R}^2) \right) \tag{34}$$

where Λ is the sum of the magnitudes of the Gaussian RBF coefficients:

$$\Lambda = \sum_{j=1}^{N_B} |\lambda_j| \tag{35}$$

For simplicity we assumed the source and evaluation boxes are of the same size although their algorithm does not require this; $\mathcal{R}$ is the box size scaled by the absolute width of the Gaussian RBFs, which in our notation is

$$\mathcal{R} = \alpha \ (N_B^{1/d} - 1) \approx \alpha \ N_B^{1/d} \tag{36}$$

Let us drop the term $\exp(-\mathcal{R}^2)$, an approximation that only makes IFGT look better. Using $N_B = \Omega \tilde{p}^d$, the error bound becomes

$$|E|/\Lambda \ \leqslant \alpha^{2p} \ \Omega^{2\tilde{p}/d} \ \tilde{p}^{2\tilde{p}} \frac{2^{\tilde{p}}}{\tilde{p}!} \tag{37}$$

Since $\Omega > 1$, i. e., compression, merely decreases E/Λ, it follows that the error bound is smallest for $\Omega = 1$ (that is, no compression with the proxy series containing N_B terms, as many Gaussians as have centers in the source box). It follows that

$$|E|/\Lambda \leqslant \alpha^{2\tilde{p}} \ \tilde{p}^{2\tilde{p}} \frac{2^{\tilde{p}}}{\tilde{p}!} \equiv \mathcal{G}(\alpha, \tilde{p}) \tag{38}$$

As noted earlier, the smallest practical value for α is $\alpha \approx 0.26$ in one dimension and $\alpha < 0.36$ in two dimensions.

This function $\mathcal{G}(\alpha, \tilde{p})$ is plotted versus $\tilde{p}$ in Fig. 6 for $\alpha = 0.26$, the lower limit in α. The minimum is only 0.15, which means that even if compression is sacrificed, the best we can guarantee is a 15 % error; the error bound of the IFGT is essentially useless. Without an effective error bound, the IFGT is useless, too.

8. Long-range versus short-range interactions

The basic step common to multipole/treecode/FGT algorithms is to replace a group of N_B terms within a box of side s by a proxy series. However, the influence of a single cluster of RBF terms must be evaluated everywhere in the domain as illustrated in Fig. 7. It is not efficient to sum the RBF series on a point-by-point basis; rather, the evaluation points are grouped into boxes, too, two of which are shown by the solid boxes.

When the evaluation box and the source box are far apart, one has "long-range interactions". Treating these is trivial for Gaussians because the RBFs have decayed to negligible amplitude over the distance from source box to evaluation box and so Gaussian long-range actions can simply be ignored.

To accelerate long-range interactions for slowly-decaying or growing RBF, the proxy must be a good approximation far outside the box. For example, the long-range interactions of Inverse Quadratics can be accelerated by using

$$\frac{1}{1 + [\alpha/h]^2 \ (x - x_j)^2} \sim \frac{1}{[\alpha/h]^2 \ x^2} + \frac{2x_j}{[\alpha/h]^2 \ x^3} + \frac{3x_j^2 - [h/\alpha]^2}{[\alpha/h]^2 \ x^4} + O(x^{-5}) \tag{39}$$

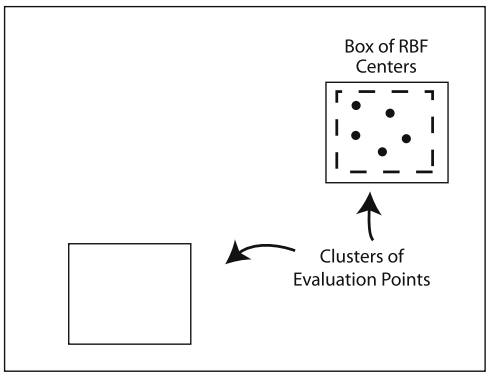

Fig. 7. A box of RBF centers (circles inside the dashed "source box") influences the sum of the RBF series over the entire domain (large square). Fast Summation methods group evaluation points into boxes, and apply proxy series to sum the RBF contributions from a cluster of RBF centers. When the evaluation points are far from the source box ("long-range interactions"), such as those in the lower left solid box, a series in inverse powers of distance converges rapidly. However, the series also includes short-range interactions, such as evaluating the sum of RBF terms in the same box that defines the RBF cluster (upper right solid box). Short-range interactions can always be evaluated by unaccelerated summation, but this is slow.

Note that the number of boxes far from a given collection of RBF centers is large compared to the number of boxes close to the source box. This implies that a scheme that only accelerates long-range interactions may greatly accelerate the overall summation even if all short-range interactions are summed without tricks or truncation.

The proxy expansions for the short-range interactions must be a good approximation *within the box itself*. Indeed, we can take this requirement as an implicit definition of what we mean by a "proxy for the short-range interactions" of the RBF terms. In the next two sections, we analyze in-the-same-box proxy expansions in greater generality.

9. Scale analysis for general RBFs

It is possible to show that the difficulty in creating successful proxies is not limited to Gaussian RBF by a non-rigorous — but illuminating — scale analysis argument. In the spirit of engineering and physics back-of-the-envelope calculations, the goal will be order-of-magnitude estimation rather than approximation.

First, assume that the spatial scale of the RBFs is

$$L \sim h/\alpha \tag{40}$$

where h is the average grid spacing of a grid that need not be uniform and α is the inverse width of the RBF functions relative to h.

As in previous sections, let p denote the degree of the proxy series so that the total number of terms in the proxy is $(p+1)^d$. This assumes a square truncation; if the proxy is truncated so that the *total* degree of each retained term is less than or equal to p, then the number of terms in such a circular truncation will be $(\pi/4)(p+1)^2$ in two dimensions and a spherical truncation will keep $(\pi/6)(p+1)^3$ terms in three dimensions. However, in scale analysis factors as small as $\pi/6$ are routinely ignored; for simplicity, we shall assume a square truncation of the proxy series in what follows.

Books on classical spectral methods such as Gottlieb and Orszag [24] and Boyd [7] offer Rules-of-Thumb for how many Chebyshev or Fourier polynomials are needed to at least crudely resolve a function of a given scale. Typically, these rules assert that for a wave-like disturbance of wavelength $2\pi L$ where L is the scale of variation of the wave, one needs ξ basis functions per wavelength. For a domain of size s, this implies

$$p \geqslant \xi \frac{s}{2\pi L} \tag{41}$$

The definitions of box side s, number of grid points N_B in the box, and average grid spacing h require that $s = h(N_B^{1/d} - 1)$; this is (27) above. The inequality becomes

$$p \geqslant \alpha \ \frac{\xi}{2\pi}(N_B^{1/d} - 1) \tag{42}$$

We see immediately that the larger α is, the larger the degree of the proxy series must be. Replacing N_B using the definition of the compression factor (29), $N_B = (p+1)^d \ \Omega$, yields

$$p \geqslant \alpha \ \frac{\xi}{2\pi}\left\{(p+1) \ \Omega^{1/d} - 1 \ \right\} \tag{43}$$

Unless p is small, it is a good approximation to neglect the ones in the above inequality; recall that compression requires that the compression parameter must be greater than one, hopefully *much* larger than one, yielding

$$1 \geqslant \alpha \frac{\xi}{2\pi} \Omega^{1/d} \tag{44}$$

or equivalently

$$\Omega \leqslant \frac{2^d \pi^d}{\xi^d} \frac{1}{\alpha^d} \tag{45}$$

The critical point is that the compression is bounded by a constant times $1/\alpha^d$: as the RBFs become wider and smoother (decreasing α), it becomes possible to find a proxy that contains fewer and fewer terms compared to the group of RBF terms with centers in the box. But how large is the proportionality constant, $(2\pi/\xi)^d$?

The most favorable case is for a Fourier basis; $\xi = 2$ in the limit of very large p. However, a nonperiodic basis such as Chebyshev polynomials is more appropriate. The theoretical minimum for ξ is then π, but Boyd [7] notes that this is only in the asymptotic limit $p \to \infty$; he recommended setting $\xi = 6 + 4(M-1)$ where M is the number of wavelengths on the interval.

Even this more pessimistic estimate for ξ leaves some hope that perhaps a fast summation could be usefully applied to short-range interactions for α sufficiently large to avoid ill-conditioning disaster. The scale analysis is too general to be sure. We therefore, analyze another case in more detail in the next section. What the scale analysis does establish is that there is clearly an inverse relationship between compression and α: the *wider* and *smoother* the basis functions, the more effectively one can accelerate RBF summation by using a proxy series for short-range interactions, that is, for summing a cluster of RBF terms within the same box spanned by their centers.

10. Chebyshev expansions of Generalized Multiquadrics

Many RBF species are subsumed under the class of Generalized Multiquadrics:

$$\phi(r;\alpha,h) \equiv \left(1 + [\alpha/h]^2 \, r^2\right)^q \tag{46}$$

where q is a constant. The special cases $q = 1/2$ ("Multiquadrics"), $q = -1/2$ ("Inverse Multiquadrics") and $q = -1$ ("Inverse Quadratics") all merit individual names and are widely used. If we use Chebyshev polynomials as a proxy series, we can borrow well-established theorems on the rate of convergence of Chebyshev series to estimate the degree p of the Chebyshev series required to reach a given error tolerance δ.

The standard interval for Chebyshev polynomials is $x \in [-1,1]$, so a useful first step is to make a change of coordinate

$$X \equiv (2/s)x \;\leftrightarrow\; x = (s/2)X \tag{47}$$

Then the proxy problem is to expand

$$\left(1 + [s\alpha/(2h)]^2 \, X^2\right)^q \approx \sum_{n=0}^{p} a_n T_n(X) \tag{48}$$

Computing a full proxy for a cluster of Generalized Multiquadrics requires Chebyshev expansions for RBF functions with centers sprinkled over all of $X \in [-1,1]$. However, as explained in Chapter 2 of [7], the rate of convergence of a function with singularities a distance $\pm v$ from the real X-axis is *slowest* when the singularities are on the *imaginary axis*. In other words, the RBFs centered closest to the origin are the stiffest challenge. (Theory further shows that the rate of convergence is only slightly faster until the singularities are moved near the *endpoints* of the interval $X \in [-1,1]$.) Thus, we lose no generality by specializing to the RBF centered at $X = 0$.

The distance of the singularities of the RBF from the real axis is

$$v \equiv \frac{2h}{s\,\alpha} \tag{49}$$

It is shown on p. 57 of [7] that the error in truncating after the pth term is approximately

$$E_p \approx 2p^{-(1+q)} \, \exp(-pv) \tag{50}$$

for small v. Substituting the singularity distance v here gives

$$E_p \approx 2p^{-(1+q)} \, \exp\left(-p\frac{2h}{s\,\alpha}\right) \tag{51}$$

In one dimension, $s = h(N_B - 1) \approx hN_B$. The error bound becomes

$$E_p \approx 2p^{-(1+q)} \, \exp\left(-\frac{p}{N_B}\frac{2}{\alpha}\right) \tag{52}$$

$$\approx 2p^{-(1+q)} \, \exp\left(-\frac{1}{\Omega}\frac{2}{\alpha}\right) \tag{53}$$

The algebraic factor $2p^{-(1+q)}$ is something of an illusion: it is equal to 2 for Inverse Quadratics, and therefore irrelevant to achieving high accuracy in summation. It is as small as $p^{-3/2}$ for Multiquadrics, but this by itself can yield high accuracy only if p is *very* large. If the burden of matching an error tolerance δ is cast entirely upon the exponential, then $E_p < \delta$ becomes

$$\alpha < \frac{2}{\Omega}\frac{1}{(-\log(\delta))} \tag{54}$$

Even for a totally useless compression of $\Omega = 1$ — as many terms in the Chebyshev series as in the cluster of RBF terms — it is clear that the Chebyshev proxy requires $\alpha \ll 1$. Further, increasing accuracy in the Fast Summation demands that α must decrease further.

Boyd and Gildersleeve [9] have shown that for Inverse Quadratics

$$\kappa^{IQ}(\alpha) \sim (1/2)\exp(\pi/\alpha), \quad \alpha \to 0 \tag{55}$$

The product of κ^{IQ} with machine epsilon, taking this as 2.2×10^{-16}, is greater than 1/100 for $\alpha < 0.0978$. Thus, $\alpha = 1/10$ is likely to be lower limit of IQ RBF interpolation. If we pose the condition $\alpha \geqslant 1/10$, then

$$\Omega \leqslant \frac{20}{-\log(\delta)} \quad \forall \alpha > 1/10 \tag{56}$$

If $\delta > 2 \times 10^{-9}$, then in theory some compression is possible. In practice, of course, our analysis has ignored the costs of computing the Chebyshev coefficients. Furthermore, multipole/treecode methods to date have used Taylor expansions, which are easier to compute by recursion than Chebyshev series. However, Taylor expansions are highly nonuniform, so obtaining an accurate approximation over the whole of an interval requires more terms (and less compression) in the Taylor series than the Chebyshev series.

It follows that the prospects for finding a good proxy and Fast Summation for short-range effects of Generalized Multiquadratics do not look good. Some slight acceleration is perhaps possible, but not enough to expand the range of problems that can be attacked. In scientific computing, a speed-up by a factor of two (or so) is nice but never a "game-changer".

11. Summary

Our negative conclusions about the usefulness of the Fast Gauss Transform for RBF series must not be extended to other applications. When the Gaussian series is generated by the Green's function solution to the diffusion equation, for example, the RBF coefficients result from the *evaluation* of a quadrature formula rather than from *inversion* of a very ill-conditioned interpolation matrix. As the grid spacing in the quadrature formula is decreased, the absolute width of the Gaussians is unchanged, but the relative width α *decreases*. There is no ill-conditioning in the quadrature formula nor growth in the coefficients of the Gaussians; the only change is that as α decreases, the Fast Gauss Transform is more and more advantageous.

The difficulty in Gaussian radial basis function interpolation is that RBFs always require the solution of an interpolation matrix system which is exponentially ill-conditioned as $\alpha \to 0$ [9], and also the evaluation of a sum with huge coefficients and cancellations. This ill-conditioning forces one to either operate with α only slightly smaller than one — a regime where the Fast Gauss Transform is no faster than direct summation — or change to the Fornberg–Piret QR-basis, which generates *non-Gaussian* sums for which the Fast Gauss Transform is *inapplicable*. We cannot ignore the possibility that future breakthroughs, perhaps in the development of robust preconditioned iterations for the interpolation matrix, may someday remove the "small α" barrier for solving the interpolation matrix for Gaussians in Gaussian form. Because the *summation* of very wide Gaussians is intrinsically ill-conditioned, independent of the difficulties of the interpolation matrix, we think the "small α" prohibition will never completely disappear.

Likewise, quasi-interpolation does not require solving an interpolation matrix system and the RBF coefficients are always the same magnitude as $f(x)$. However, we have seen earlier that the quasi-interpolation error is (i) much worse than for interpolation and (ii) rises steeply as α^{-k} for kth order quasi-interpolation. Small α is bad for quasi-interpolation, too.

We have shown in the last two sections that this difficulty in summing short-range interactions by a compressed proxy is general, and not merely confined to Gaussian RBFs. However, we have only examined a subset of the many RBF species. Perhaps there is a kind of RBF such that a least a little acceleration of short-range interactions is possible. (By short-range interactions, we mean the summation of RBF terms at an evaluation point $\mathbf{x}$ which is close to the centers of the terms.)

Long-range interactions are a different story: the inverse power series

$$\frac{1}{1+[\alpha/h]^2\ r^2} \sim \sum_{n=1}^{\infty} \frac{(-1)^{n+1}}{[\alpha/h]^{2n}\ r^{2n}} \tag{57}$$

converges faster and faster as r increases. Multipole and treecode algorithms have triumphed using such approximations to accelerate the contributions of a group of terms when the centers are far from the evaluation point. Nothing in this article is intended to criticize or diminish this achievement.

However, long-range interactions in RBFs are an "own goal", as soccer players say about a ball accidentally knocked into the goal by a defender. Students of gravitational interactions in a star cluster cannot rewrite the law of gravity; the slow de-

cay of the inverse-square law implies that every star is attracting every other star. In employing RBFs purely as meshless basis, however, one has *choice*, and can always choose RBFs such as Gaussians which have only short-range interactions.

Acknowledgments

This work was supported by the National Science Foundation through Grants OCE998636, OCE 0451951 and ATM0723440. I thank the three reviewers for helpful comments. I also thank Nail Gumerov, Eric Miller and Adam Florence for helpful correspondence and reprints.

References

[1] B.J.C. Baxter, Norm estimates for inverses of Toeplitz distance matrices, J. Approx. Theor. 70 (1994) 222–242.
[2] B.J.C. Baxter, G. Roussos, A new error estimate of the fast Gauss transform, SIAM J. Sci. Comput. 24 (2002) 257–259.
[3] B.J.C. Baxter, N. Sivakumar, On shifted cardinal interpolation by gaussians and multiquadrics, J. Approx. Theor. 87 (1996) 36–59.
[4] J. Beale, A. Majda, Vortex methods. I. Convergence in three dimensions, Math. Comput. 39 (1982) 1–27.
[5] J. Beale, A. Majda, Vortex methods. II. Higher order accuracy in two or three dimensions, Math. Comput. 39 (1982) 29–52.
[6] J.P. Boyd, A fast algorithm for Chebyshev and Fourier interpolation onto an irregular grid, J. Comput. Phys. 103 (1992) 243–257.
[7] J.P. Boyd, Chebyshev and Fourier Spectral Methods, second ed., Dover, Mineola, New York, 2001.
[8] J.P. Boyd, L. Bridge, Sensitivity of RBF interpolation on an otherwise uniform grid with a point omitted or slightly shifted, Appl. Numer. Math., submitted for publication.
[9] J.P. Boyd, K.W. Gildersleeve, Numerical experiments on the condition number of the interpolation matrices for radial basis functions, Appl. Numer. Math., submitted for publication.
[10] J.P. Boyd, L. Wang, An analytic approximation to the cardinal functions of Gaussian radial basis functions on a one-dimensional infinite uniform lattice, Appl. Math. Comput. 215 (2009) 2215–2223.
[11] J.P. Boyd, L. Wang, Asymptotic coefficients for Gaussian radial basis function interpolants, Appl. Math. Comput., submitted for publication.
[12] J.P. Boyd, L. Wang, Truncated gaussian rbf differences are always inferior to finite differences of the same stencil width, Commun. Comput. Phys. 5 (2009) 42–60.
[13] J.P. Boyd, C. Zhou, Kelvin waves in the nonlinear shallow water equations on the sphere: nonlinear travelling waves and the corner wave bifurcation, J. Fluid Mech. 617 (2008) 185–205.
[14] D. Brown, L. Ling, E. Kansa, J. Levesley, On approximate cardinal preconditioning methods for solving PDEs with radial basis functions, Eng. Anal. Boundary Elem. 29 (2005) 343–353.
[15] M.D. Buhmann, Radial basis functions: theory and implementations, Cambridge Monographs on Applied and Computational Mathematics, vol. 12, Cambridge University Press, 2003.
[16] A. Dutt, V. Rokhlin, Fast Fourier Transforms for nonequispaced data, SIAM J. Comput. 14 (1993) 1368–1393.
[17] A. Dutt, V. Rokhlin, Fast Fourier Transforms for nonequispaced data, II, Appl. Comput. Harmonic Anal. 2 (1995) 85–110.
[18] A. Emdadi, E.J. Kansa, N.A. Libre, M. Rahimian, M. Shekarchi, Stable PDE solution methods for large multiquadric shape parameters, CMES Comput. Model. Eng. Sci. 25 (2008) 23–41.
[19] G.F. Fasshauer, Meshfree Approximation Methods with MATLAB, Interdisciplinary Mathematical Sciences, World Scientific Publishing Company, Singapore, 2007.
[20] A.C. Faul, G. Goodsell, M.J.D. Powell, A Krylov subspace algorithm for multiquadric interpolation in many dimensions, IMA J. Numer. Anal. 25 (2005) 1–24.
[21] A.G. Florence, C.F. van Loan, A Kronecker product formulation of the Fast Gauss Transform, Cornell University Ithaca, New York, 2000, unpublished manuscript.
[22] B. Fornberg, C. Piret, A stable algorithm for flat radial basis functions on a sphere, SIAM J. Sci. Comput. 30 (2007) 60–80.
[23] B. Fornberg, G. Wright, Stable computation of multiquadric interpolants for all values of the shape parameter, Comput. Math. Appl. 48 (2004) 853–867.
[24] D. Gottlieb, S.A. Orszag, Numerical Analysis of Spectral Methods, SIAM, Philadelphia, PA, 1977. 200 pp..
[25] L. Greengard, J. Strain, The Fast Gauss Transform, SIAM J. Sci. Stat. Comput. 12 (1991) 79–94.
[26] L. Greengard, X. Sun, A new version of the fast Gauss transform, Documenta Mathematica Extra Volume ICM 1998 III (1991) 575–584.
[27] N.A. Gumerov, R. Duraiswami, Fast radial basis function interpolation via preconditioned Krylov iteration, SIAM J. Sci. Comput. 29 (2007) 1876–1899.
[28] E.J. Kansa, Y.C. Hon, Circumventing the ill-conditioning problem with multiquadrics radial basis functions: applications to elliptic partial differential equations, Comput. Math. Appl. 39 (2000) 123–137.
[29] E.J. Kansa, H. Power, G.E. Fasshauer, L. Ling, A volumetric integral radial basis function method for time-dependent partial differential equations: I. Formulation, J. Eng. Anal. Boundary Elem. 28 (2004) 1191–1206.
[30] M. Kayano, S. Konishi, Functional principal component analysis via regularized gaussian basis expansions and its application to unbalanced data, J. Stat. Plan. Inference 139 (2009) 2388–2398.
[31] J. Keiner, S. Kunis, D. Potts, Fast summation of radial functions on the sphere, Computing 78 (2006) 1–15.
[32] K. Kobayashi, A remark on the Fast Gauss Transform, Publications of the Research Institute for Mathematical Sciences, vol. 39, 2003, pp. 785–796.
[33] R. Krasny, L. Wang, Fast evaluation of multiquadric radial basis function approximations by a Cartesian Taylor series treecode, Appl. Numer. Math., submitted for publication.
[34] C.-F. Lee, L.V. Ling, R. Schaback, On convergent numerical algorithms for unsymmetric collocation, Adv. Comput. Math. 30 (2009) 339–354.
[35] L.V. Ling, E.J. Kansa, Preconditioning for radial basis functions with domain decomposition methods, Math. Comput. Model. 40 (2004) 1413–1427.
[36] L.V. Ling, E.J. Kansa, A least-squares preconditioner for radial basis functions collocation methods, Adv. Comput. Math. 23 (2005) 31–54.
[37] L. Liu, L.P. Chua, D.N. Ghista, Conforming radial point interpolation method for spatial shell structures on the stress-resultant shell theory, Arch. Appl. Mech. 75 (2006) 248–267.
[38] O.E. Livne, G.B. Wright, Fast multilevel evaluation of smooth radial basis function expansions, Electronic Trans. Numer. Anal. 23 (2006) 263–287.
[39] F. Magoules, L.A. Diago, I. Hagiwara, A two-level iterative method for image reconstruction with radial basis functions, JSME Int. J. Ser. C-Mech. Syst. Mach. Elem. Manufact. 48 (2005) 149–158.
[40] V. Maz'ya, G. Schmidt, On approximate approximation using Gaussian kernels, IMA J. Numer. Anal. 16 (1996) 13–29.
[41] V. Maz'ya, G. Schmidt, Approximate Approximations, American Mathematical Society, Providence, 2007. 349 pp..
[42] R. Mittelman, E.L. Miller, Fast Gauss Transforms based on a high order singular value decomposition for nonlinear filtering, in: 2007 IEEE/SP 14TH Workshop on Statistical Signal Processing, vols. 1 and 2, IEEE, New York, 2007, pp. 94–98.
[43] R. Mittelman, E.L. Miller, Nonlinear filtering using a new proposal distribution and the improved Fast Gauss Transform with tighter performance bounds, IEEE Trans. Signal Process. 56 (2008) 5746–5757.
[44] T.J. Moroney, I.W. Turner, A finite volume method based on radial basis functions for two-dimensional nonlinear diffusion equations, Appl. Math. Model. 30 (2006) 1118–1133.
[45] R.B. Platte, T.A. Driscoll, Polynomials and potential theory for Gaussian radial basis function interpolation, SIAM J. Numer. Anal. 43 (2005) 750–766.

[46] D. Potts, G. Steidl, Fast summation at nonequispaced knots by NFFT, SIAM J. Sci. Comput. 24 (2003) 2013–2037.
[47] D. Potts, G. Steidl, A. Nieslony, Fast convolution with radial kernels at nonequispaced knots, Numer. Math. 98 (2004) 329–351.
[48] D. Potts, G. Steidl, M. Tasche, Fast Fourier Transforms for nonequispaced data: a tutorial, in: J.J. Benedetto, P. Ferreira (Eds.), Modern Sampling Theory: Mathematics and Applications, Birkhauser, Boston, 2000, pp. 253–274.
[49] Y.F. Rashed, Transient dynamic boundary element analysis using Gaussian-based mass matrix, Eng. Anal. Boundary Elem. 26 (2002) 265–279.
[50] V.C. Raykar, C. Yang, R. Duraiswami, N.A. Gumerov, Improved Fast Gauss Transform, Tech. Rep. CS-TR-4767/UMIACS-TR-2005-69, University of Maryland, College Park, Maryland, 2003.
[51] S.D. Riemenschneider, N. Sivakumar, On cardinal interpolation by Gaussian radial-basis functions: properties of fundamental functions and estimates for Lebesgue constants, J. Analyse Math. 79 (1999) 33–61.
[52] J.L. Roberts, A method for calculating meshless finite difference weights, Int. J. Numer. Methods Eng. 74 (2008) 321–336.
[53] G. Roussos, B.J.C. Baxter, Rapid evaluation of radial basis functions, J. Comput. Appl. Math. 180 (2005) 51–70.
[54] R. Schaback, Multivariate interpolation by polynomials and radial basis functions, Construct. Approx. 21 (2005) 293–317.
[55] J. Strain, The Fast Gauss Transform with variable scales, SIAM J. Sci. Stat. Comput. 12 (1991) 1131–1139.
[56] C.E. Torres, L.A. Barba, Fast radial basis function interpolation with Gaussians by localization and iteration, J. Comput. Phys., submitted for publication.
[57] X. Wan, G.E. Karniadakis, A sharp error estimate for the fast Gauss transform, J. Comput. Phys. 219 (2006) 7–12.
[58] H. Wendland, Scattered Data Approximation, Cambridge University Press, 2005.
[59] T. Yamada, L.C. Wrobel, Properties of Gaussian radial basis functions in the dual reciprocity boundary element method, Zeitschrift fur Angewandte Math. Phys. 44 (1993) 1054–1067.
[60] C. Yang, R. Duraiswami, N.A. Gumerov, Improved Fast Gauss Transform, Tech. Rep. CS-TR-4495, University of Maryland, College Park, Maryland, 2003.
[61] C.J. Yang, R. Duraiswami, N.A. Gumerov, L. Davis, Improved Fast Gauss Transform and efficient kernel density estimation, in: Proceedings of Ninth IEEE International Conference on Computer Vision, vols. I and II, IEEE Computer Society, Washington, DC, 2003, pp. 464–471.