Recent Results for Moving Least Squares Approximation

Gregory E. Fasshauer and Jack G. Zhang

Abstract. We describe two experiments recently conducted with the approximate moving least squares (MLS) approximation method. On the one hand, the NFFT library of Kunis, Potts, and Steidl is coupled with the approximate MLS method to obtain a fast and accurate multivariate approximation method. The second experiment uses approximate MLS approximation in combination with a multilevel approximation algorithm. This method can be used for data compression, or to obtain an approximation with radial functions that employs variable scales and non-uniform center locations.

§1. Introduction

In this paper we address two limitations of approximate moving least squares (MLS) approximation with radial weight functions encountered in our earlier work (see, e.g., [3, 5, 6]). The first problem is that, even though approximate MLS approximation reduces the computational work for multivariate radial function approximation to that of a mere summation problem (i.e., no linear systems need to be solved), this task is still rather expensive for weight functions with global support such as Gaussians. In the present paper we take advantage of the recent NFFT library by Kunis, Potts, and Steidl [9] to considerably speed up these summations. The NFFT library is an extension of the fast Fourier transform to the situation of non-uniform centers. By combining the approximate MLS approximation method with the NFFT library we are able to efficiently evaluate radial function approximations based on Gaussians and similar functions at millions of points in one, two and three space dimensions. Contrary to fast multipole-type algorithms, the NFFT software can be viewed as a general purpose approach that can handle many different radial functions without the need to develop different series expansions for each one of them.

xxx and xxx (eds.), pp. 1–4.

ISBN 0-9728482-x-x

The second difficulty associated with the approximate MLS approximation method is its application to non-uniform centers. While both theoretical [10] and computational work [6] on this topic exist, it still remains an open problem to find an approximate MLS approximation algorithm that works for arbitrarily scattered centers. By using a multilevel residual updating iteration coupled with a thresholding strategy we end up with a method that can be viewed as either a multivariate approximation method based on radial functions with variable scales and non-uniform centers, or as a data compression algorithm similar to those described in [1] and [8].

The remainder of the paper is organized as follows. In the next section we provide the reader with a brief review of the MLS, and approximate MLS methods. In Sect. 3 we explain how the NFFT library can be adapted for our purposes, and present some numerical experiments. In Sect. 4 we discuss our experiments with the multilevel approximation algorithm for data compression, and close with a summary and outlook to future work in Sect. 5.

§2. Review of MLS Approximation

Given a set of data, $\{(\boldsymbol{x}_i, f(\boldsymbol{x}_i)) : \boldsymbol{x}_i \in \mathbb{R}^s,\ i = 1, \ldots, N\}$, we seek a quasi-interpolant in the form

$$\mathcal{P}f(\boldsymbol{x}) = \sum_{i=1}^{N} f(\boldsymbol{x}_i)\Psi_i(\boldsymbol{x}), \qquad \boldsymbol{x} \in \mathbb{R}^s, \tag{1}$$

with appropriate generating functions Ψ_i.

It is well known that if the Ψ_i are cardinal, i.e., $\Psi_i(\boldsymbol{x}_j) = \delta_{ij}$ for $i, j = 1, \ldots, N$, then $\mathcal{P}f$ interpolates the given set of data. Moreover, if the cardinal functions are associated with positive definite radial functions then $\mathcal{P}f$ is norm-minimizing, i.e., the cardinal interpolant has minimal Chebyshev norm for all functions f in the native space associated with the Ψ_i (see, e.g., [15]). Therefore, we intend to find values of the generating functions $\Psi_i(\boldsymbol{x}) = \Psi(\boldsymbol{x}_i, \boldsymbol{x})$, which are as close to cardinal values as possible. Following this philosophy the *Backus-Gilbert* approach to moving least squares (MLS) approximation (see, e.g., [2]) is to minimize

$$\frac{1}{2}\sum_{i=1}^{N} \Psi_i^2(\boldsymbol{x})\frac{1}{W(\boldsymbol{x}, \boldsymbol{x}_i)} \tag{2}$$

subject to the polynomial reproduction constraints

$$\sum_{i=1}^{N} p(\boldsymbol{x}_i)\Psi_i(\boldsymbol{x}) = p(\boldsymbol{x}) \quad \text{for all } p \in \Pi_d^s, \tag{3}$$

where Π_d^s is the space of s-variate polynomials of total degree at most d with dimension $m = \binom{s+d}{d}$. The $W(\cdot, \boldsymbol{x}_i)$, $i = 1, \ldots, N$, form a set of positive weight functions whose value changes with the evaluation point $\boldsymbol{x}$. According to the standard moving least squares method, if $W(\cdot, \boldsymbol{x}_i)$ is strong at $\boldsymbol{x}_i$ and decreases rapidly away from $\boldsymbol{x}_i$, then the resulting approximation function nearly interpolates the given data. In fact, interpolation is achieved by using weights that are singular (infinite) at $\boldsymbol{x}_i$.

Besides the norm-minimization (2) the polynomial reproduction constraints (3) are used to obtain a desired approximation order (determined by the value of d). One can show that the polynomial reproduction constraints correspond to a set of *discrete moment conditions* for the generating function.

For each fixed $\boldsymbol{x}$ (evaluation point), satisfying (2) and (3) involves the solution of a (small) linear system obtained via the standard Lagrange multiplier approach (see, e.g., [14] or [7]). To avoid doing this, we proposed earlier [3, 5, 6] to use the idea of *approximate approximation* (see, e.g., [10]) first suggested by Maz'ya in the late 1980s. That is, our goal is to replace the discrete moment conditions by continuous ones as described in the following theorem.

Theorem 1. *(Maz'ya and Schmidt [10]) Let $f \in C^{d+1}(\mathbb{R}^s)$, $\{\boldsymbol{x}_{\boldsymbol{\nu}} = h\boldsymbol{\nu} : \boldsymbol{\nu} \in \mathbb{Z}^s\} \subset \mathbb{R}^s$ and Ψ be a continuous generating function which satisfies the* continuous moment conditions

$$\int_{\mathbb{R}^s} \boldsymbol{x}^{\boldsymbol{\alpha}} \Psi(\boldsymbol{x}) d\boldsymbol{x} = \delta_{\boldsymbol{\alpha}\boldsymbol{0}}, \quad 0 \le |\boldsymbol{\alpha}| \le d, \tag{4}$$

along with a mild decay requirement. Then

$$\mathcal{M}_h f(\boldsymbol{x}) = \mathcal{D}^{-1/2} \sum_{\boldsymbol{\nu} \in \mathbb{Z}^s} f(\boldsymbol{x}_{\boldsymbol{\nu}}) \Psi\left(\frac{\boldsymbol{x} - \boldsymbol{x}_{\boldsymbol{\nu}}}{\sqrt{\mathcal{D}} h}\right) \tag{5}$$

leads to

$$||\mathcal{M}_h f - f||_\infty = \mathcal{O}(h^{d+1} + \epsilon_0(\Psi, \mathcal{D})).$$

This theorem tells us that we can approximate (or quasi-interpolate) the given data with numerical approximation order $\mathcal{O}(h^{d+1})$ until a certain so-called *saturation error* ϵ_0 takes over. However, this saturation error can be controlled by appropriately scaling the generating functions (with the right choice of $\mathcal{D}$). In other words, we can limit the saturation error within the range of the machine accuracy on any particular computer. We point out that we will be using regularly spaced data sites $\boldsymbol{x}_{\boldsymbol{\nu}}$ in all of our applications below. However, another version of the theorem also covers irregularly spaced centers (see [10]), and initial numerical experiments for this more general situation have been reported in [6].

Now, we have come to the essential part for the practical application of this procedure, i.e., to find/construct appropriate generating functions. To make a connection to radial basis function (RBF) approximation, we assume that

$$\Psi(\boldsymbol{x}) = q(||\boldsymbol{x}^2||)\phi(||\boldsymbol{x}||^2),$$

where q is a polynomial which will give us the desired approximation order, and ϕ is a sufficiently decaying univariate function to be chosen by the user. Hence, all we need is to find q by satisfying (cf. (4))

$$\int_{\mathbb{R}^s} ||\boldsymbol{x}||^{2k} q(||\boldsymbol{x}||^2)\phi(||\boldsymbol{x}||^2)d\boldsymbol{x} = \delta_{k0}, \quad 0 \le k \le d.$$

If we use s-dimensional spherical coordinates and the change of variables $r = \|\boldsymbol{x}\|$ and $y = r^2$, we have instead the 1D conditions

$$\frac{\pi^{s/2}}{\Gamma(s/2)} \int_0^\infty y^{k-1} q(y)\phi(y) y^{s/2} dy = \delta_{k0}, \quad 0 \le k \le d.$$

This is equivalent to finding a univariate polynomial $q = q(y)$ orthogonal with respect to the weight $\phi(y)y^{s/2}$ on the interval $[0, \infty)$. Thus, the guiding principle is to pick an arbitrary univariate weight function $\phi(y)y^{s/2}$ (actually just ϕ), then look for a polynomial q of a certain order d (depending on the desired approximation order).

For example, starting with $\phi(r) = e^{-r}$, the following table lists the generating functions Ψ for $s = 1, 2, 3$ with approximation orders 2, 4, 6.

s	$O(h^2)$	$O(h^4)$	$O(h^6)$
1	$\frac{1}{\sqrt{\pi}} e^{-\|x\|^2}$	$\frac{1}{\sqrt{\pi}} \left(\frac{3}{2} - \|x\|^2\right) e^{-\|x\|^2}$	$\frac{1}{\sqrt{\pi}} \left(\frac{15}{8} - \frac{5}{2}\|x\|^2 + \frac{1}{2}\|x\|^4\right) e^{-\|x\|^2}$
2	$\frac{1}{\pi} e^{-\|\boldsymbol{x}\|^2}$	$\frac{1}{\pi} \left(2 - \|\boldsymbol{x}\|^2\right) e^{-\|\boldsymbol{x}\|^2}$	$\frac{1}{\pi} \left(3 - 3\|\boldsymbol{x}\|^2 + \frac{1}{2}\|\boldsymbol{x}\|^4\right) e^{-\|\boldsymbol{x}\|^2}$
3	$\frac{1}{\pi^{3/2}} e^{-\|\boldsymbol{x}\|^2}$	$\frac{1}{\pi^{3/2}} \left(\frac{5}{2} - \|\boldsymbol{x}\|^2\right) e^{-\|\boldsymbol{x}\|^2}$	$\frac{1}{\pi^{3/2}} \left(\frac{35}{8} - \frac{7}{2}\|\boldsymbol{x}\|^2 + \frac{1}{2}\|\boldsymbol{x}\|^4\right) e^{-\|\boldsymbol{x}\|^2}$

Tab. 1. Generating functions associated with $\phi(r) = e^{-r}$.

The construction of many other generating functions is also possible. For instance, in [3] compactly supported generating functions based on the weight $\phi(r) = (1 - \sqrt{r})_+^4 (4\sqrt{r} + 1)$, $r = \|\boldsymbol{x}\|$, were constructed and compared to the well-known radial basis function $(1 - \|\boldsymbol{x}\|)_+^4 (4\|\boldsymbol{x}\| + 1)$ of Wendland [13].

Later in this paper we will use the set of generating functions listed in Table 1 to present applications of the fast evaluation via non-uniform fast Fourier transform (NFFT) and data compression based on a multilevel approximation algorithm.

§3. Fast Evaluation via NFFT

Assume that for equally spaced data sites $\boldsymbol{x}_\nu \in [0,1)^s$ we are to evaluate

$$\mathcal{M}_h f(\boldsymbol{x}) = \mathcal{D}^{-s/2} \sum_{\boldsymbol{\nu}\in[1,N]^s} f(\boldsymbol{x}_{\boldsymbol{\nu}})\Psi\left(\frac{\boldsymbol{x}-\boldsymbol{x}_{\boldsymbol{\nu}}}{\sqrt{\mathcal{D}}h}\right) \tag{6}$$

at non-equally spaced evaluation points $\boldsymbol{x} = \boldsymbol{y}_j \in [0,1)^s$ for $j = 1, \ldots, M$.

Direct evaluation of (1) or (6) at M points requires $\mathcal{O}(N^s M)$ operations. It is therefore desirable to use a fast method with an acceptable loss of accuracy to obtain a better computational speed. The *fast Fourier transform for non-equally spaced points* (NFFT) ideally provides this capability. While various papers on this subject have appeared in recent years (see, e.g., some of the references in [9]), public domain software and a user's guide are now available in form of the NFFT package [9].

We would like to emphasize that with the NFFT software neither the M evaluation points nor the N^s data centers need to be uniformly spaced as in the standard FFT. Moreover M and N^s do not have to be equal nor does the number of centers have to be an integer power of N. However, for this discussion on approximate MLS approximation we restrict our attention to equispaced centers $\boldsymbol{x}_{\boldsymbol{\nu}}$.

In order to perform the NFFT algorithm, let us present some necessary background material. First, in order to use the NFFT software package for our computational experiments some scaling and shifting of the variables are needed. Specifically, we use two basic functions (provided in the NFFT library), `nfft_(`s`)D_trafo` and `nfft_(`s`)D_transpose` (with $s = 1, 2$ or 3), which correspondingly compute

$$\varphi(\boldsymbol{x}_j) = \sum_{\boldsymbol{k}\in[-\frac{n}{2},\frac{n}{2})^s} c_{\boldsymbol{k}} e^{\pm 2\pi i \boldsymbol{k}\boldsymbol{x}_j} \quad \text{for } \boldsymbol{x}_j \in [-\frac{1}{2},\frac{1}{2})^s \tag{7}$$

$$c_{\boldsymbol{k}} = \sum_{\boldsymbol{x}_j\in[-\frac{1}{2},\frac{1}{2})^s} \varphi(\boldsymbol{x}_j) e^{\pm 2\pi i \boldsymbol{k}\boldsymbol{x}_j} \quad \text{for } \boldsymbol{k} \in [-\frac{n}{2},\frac{n}{2})^s, \tag{8}$$

where the plus-minus signs indicate that the NFFT-functions can be used either in forward or backward mode (e.g., a "+" in (7) and a "-" in (8) correspond to the forward NFFT). Note that both (7) and (8) require a domain that is symmetric about the origin but (6) is evaluated in $[0,1)^s$.

Denote the generating functions

$$\Phi(\boldsymbol{x}) = \Psi\left(\frac{\boldsymbol{x}}{\sqrt{\mathcal{D}}h}\right) \quad \text{for } \boldsymbol{x} \in [-1,1)^s. \tag{9}$$

Hence (6) becomes

$$\mathcal{M}_h f(\boldsymbol{x}) = \mathcal{D}^{-s/2} \sum_{\boldsymbol{\nu}\in[0,N)^s} f(\boldsymbol{x}_{\boldsymbol{\nu}})\Phi\left(\boldsymbol{x}-\boldsymbol{x}_{\boldsymbol{\nu}}\right). \tag{10}$$

Our goal is to eventually replace the right-hand side of (10) (or (6)) by a pure Fourier transform. To this end, we first express $\Phi(\boldsymbol{x})$ by a Fourier transform approximation. This is possible since Φ is infinitely smooth.

Thus, for a sufficiently large n, if we define coefficients $b_\ell \in \mathbb{C}^s$ by

$$b_\ell = \frac{1}{n^s}\sum_{\boldsymbol{k}\in I_n} \Phi\left(\frac{2\boldsymbol{k}}{n}\right) e^{2\pi i \boldsymbol{k}\frac{\boldsymbol{\ell}}{n}} \quad \text{for all } \ell \in I_n = [\frac{-n}{2},\frac{n}{2})^s, \tag{11}$$

then for $\boldsymbol{j} \in [-n,n)^s$

$$\begin{aligned}
\sum_{\boldsymbol{\ell}\in I_n} b_{\boldsymbol{\ell}} e^{-2\pi i \boldsymbol{\ell}\frac{\boldsymbol{j}}{2n}} &= \frac{1}{n^s}\sum_{\boldsymbol{\ell}\in I_n}\sum_{\boldsymbol{k}\in I_n} \Phi\left(\frac{2\boldsymbol{k}}{n}\right) e^{2\pi i \boldsymbol{k}\frac{\boldsymbol{\ell}}{n}} e^{-2\pi i \boldsymbol{\ell}\frac{\boldsymbol{j}}{2n}} \\
&= \frac{1}{n^s}\sum_{\boldsymbol{k}\in I_n} \Phi\left(\frac{2\boldsymbol{k}}{n}\right) \sum_{\boldsymbol{\ell}\in I_n} e^{2\pi i \boldsymbol{\ell}(\frac{\boldsymbol{k}}{n}-\frac{\boldsymbol{j}}{2n})} \\
&= \frac{1}{n^s}\sum_{\boldsymbol{k}\in I_n} \Phi\left(\frac{2\frac{\boldsymbol{j}}{2}}{n}\right) \\
&= \Phi\left(\frac{\boldsymbol{j}}{n}\right).
\end{aligned} \tag{12}$$

The third equality in the above simplification follows since for (multi-)integer $\boldsymbol{j}$, $\boldsymbol{k}$, $\boldsymbol{\ell}$, and n

$$\sum_{\boldsymbol{\ell}\in I_n} e^{-2\pi i \boldsymbol{\ell}\frac{\boldsymbol{j}-\boldsymbol{k}}{n}}$$

is n when $\boldsymbol{j}-\boldsymbol{k}$ is a multiple of n (zero is the only possibility in our case) and zero otherwise.

Result (12) simply implies that its left-hand side interpolates Φ at a set of n^s equally spaced points in $[-1,1)^s$. Hence, we have

$$\Phi(\boldsymbol{x}) \approx \sum_{\boldsymbol{\ell}\in I_n} b_{\boldsymbol{\ell}} e^{-2\pi i \boldsymbol{\ell}\frac{\boldsymbol{x}}{2}} \quad \text{for } \boldsymbol{x} \in [-1,1)^s, \tag{13}$$

where the Fourier transform coefficients b_ℓ are determined by (11). Note that $\boldsymbol{x}$, $\boldsymbol{x}_{\boldsymbol{\nu}} \in [0,1)^s$. Hence Φ is evaluated in $[-1,1)^s$ and therefore must be approximated in this domain as well.

Now we replace Φ in (10) by (13). Hence, we have for $\boldsymbol{x} \in [-1,1)^s$

$$\mathcal{M}_h f(\boldsymbol{x}) \approx \mathcal{D}^{-s/2} \sum_{\boldsymbol{x}_{\boldsymbol{\nu}}\in[0,1)^s} f(\boldsymbol{x}_{\boldsymbol{\nu}}) \sum_{\boldsymbol{\ell}\in I_n} b_{\boldsymbol{\ell}} e^{-2\pi i \boldsymbol{\ell}\frac{\boldsymbol{x}-\boldsymbol{x}_{\boldsymbol{\nu}}}{2}}. \tag{14}$$

Evaluating (14) at $\boldsymbol{x} = \boldsymbol{y}_j \in [0,1)^s$ for $j = 1, \ldots, M$

$$\begin{aligned} \mathcal{M}_h f(\boldsymbol{y}_j) &\approx \mathcal{D}^{-s/2} \sum_{\boldsymbol{x_\nu} \in [0,1)^s} f(\boldsymbol{x_\nu}) \sum_{\boldsymbol{\ell} \in I_n} b_{\boldsymbol{\ell}} e^{-2\pi i \boldsymbol{\ell} \frac{\boldsymbol{y}_j - \boldsymbol{x_\nu}}{2}} \\ &= \mathcal{D}^{-s/2} \sum_{\boldsymbol{\ell} \in I_n} b_{\boldsymbol{\ell}} \sum_{\boldsymbol{x}_\nu \in [0,1)^2} f(\boldsymbol{x}_\nu) e^{2\pi i \boldsymbol{\ell} \frac{\boldsymbol{x}_\nu}{2}} e^{-2\pi i \boldsymbol{\ell} \frac{\boldsymbol{y}_j}{2}}. \qquad (15) \end{aligned}$$

If we introduce another set of coefficients $a_\ell \in \mathbb{C}^s$ as

$$a_{\boldsymbol{\ell}} = \sum_{\boldsymbol{x_\nu} \in [0,1)^2} f(\boldsymbol{x_\nu}) e^{2\pi i \boldsymbol{\ell} \frac{\boldsymbol{x_\nu}}{2}} \quad \text{for } \boldsymbol{\ell} \in [-\frac{n}{2}, \frac{n}{2})^s, \qquad (16)$$

and utilize the abbreviation $d_{\boldsymbol{\ell}} = a_{\boldsymbol{\ell}} b_{\boldsymbol{\ell}}$, then (10) can be rewritten (via (15)) as

$$\begin{aligned} M_h f(\boldsymbol{y}_j) &= \mathcal{D}^{-s/2} \sum_{\boldsymbol{\nu} \in [0,N]^s} f(\boldsymbol{x_\nu}) \Phi\left(\boldsymbol{y}_j - \boldsymbol{x_\nu}\right) \\ &\approx \mathcal{D}^{-s/2} \sum_{\boldsymbol{\ell} \in I_n} d_{\boldsymbol{\ell}} e^{-2\pi i \boldsymbol{\ell} \frac{\boldsymbol{y}_j}{2}} \quad \text{for } j = 1, \ldots, M. \qquad (17) \end{aligned}$$

Therefore, the entire fast computation algorithm is completed by three basic NFFT computations. They are,

- b_ℓ given by (11) which is computed by (7);
- a_ℓ given by (16) which is computed by (8); and
- (17) which is again computed by (7).

Thus, if n is chosen to be constantly proportional to N, then the computational complexity becomes $\mathcal{O}(\log(N)(N^s + M))$. Compared to $\mathcal{O}(N^s M)$, the evaluation time is significantly reduced when M is very large.

It is important to note that the key to this efficiency improvement is the value for n, which decides the accuracy loss due to the FFT approximation (13) of the generating functions, and also the accuracy of the computational algorithm for the NFFT itself. The latter is mathematically and experimentally analyzed in [9]. Although it is difficult to give the explicit value for this error (caused by the NFFT algorithm itself), we have seen that it is relatively small enough to be ignored. Hence, only the error introduced by the representation (13) warrants a concern.

Let

$$\Phi_n(\boldsymbol{x}) = \sum_{\boldsymbol{\ell} \in I_n} b_{\boldsymbol{\ell}} e^{-2\pi i \boldsymbol{\ell} \frac{\boldsymbol{x}}{2}} \quad \text{for } \boldsymbol{x} \in [-1,1)^s$$

as in (13). Therefore, as shown earlier, for each n

$$\Phi(\boldsymbol{x}) - \Phi_n(\boldsymbol{x}) = 0 \quad \text{at } \boldsymbol{x} = \frac{\boldsymbol{k}}{n} \quad \text{for all } \boldsymbol{k} \in I_n. \qquad (18)$$

Since Φ is infinitely smooth in $[-1,1)^s$, observing (11), $b_{\boldsymbol{\ell}}$ is clearly bounded for all n and all $\boldsymbol{\ell}$. Hence, standard Fourier theory tells us that Φ_n is continuous and differentiable in $[-1,1)^s$ for all n. Also, for all n the gradient of Phi_n is pointwise bounded for any $x \in [-1,1)^s$. Therefore, Φ_n is uniformly convergent to Φ as n goes to ∞.

The test data for this experiment is taken from the following modified version g of "Franke's function" f given by

$$\begin{aligned}
f(x_1,x_2,x_3) &= \frac{3}{4}\Bigg[\exp(-\frac{(9x_1-2)^2}{4}-\frac{(9x_2-2)^2}{4}-\frac{(9x_3-2)^2}{4}) \\
&\quad +\exp(-\frac{(9x_1+1)^2}{49}-\frac{(9x_2+1)^2}{10}-\frac{(9x_3+1)^2}{29})\Bigg] \\
&\quad +\frac{1}{2}\exp(-\frac{(9x_1-7)^2}{4}-(9x_2-3)^2-\frac{(9x_3-5)^2}{2}) \\
&\quad -\frac{1}{5}\exp(-(9x_1-4)^2-(9x_2-7)^2-(9x_3-5)^2) \\
g(x_1,x_2,x_3) &= 15f(x_1,x_2,x_3)\prod_{i=1}^{3}\exp(\frac{-1}{1-4(x_i-1/2)^2}),
\end{aligned}$$

where $x_1,\ x_2,\ x_3$ are used according to the space dimension s.

In our experiments we use $n = 4N$ in (11) for all computations except for the very last experiments in 2D and 3D. We do not have an automated strategy for choosing n. However, the values just mentioned yield satisfactory results and go along with the values suggested by Theorems 3.1 and 3.4 of [11]. In all experiments displayed in Figures 1–3, the parameter $\mathcal{D}$ in (6) is taken to be 3.0. Figure 4 shows the effect of different choices of $\mathcal{D}$ on the saturation error.

The left graphs in Figures 1–3 show the maximum error versus the number of centers N on a logarithmic scale for the three types of generating functions of Table 1. This illustrates that the approximation does converge well (almost reaching the rates predicted by the theory) as the data locations get finer. The errors are computed at M evaluation points randomly distributed in the unit cube with $M = 32,768$ for $s = 1$, $M = 262,144$ for $s = 2$, and $M = 2,146,689$ for $s = 3$. The presence of the saturation error is clearly visible in Figure 1. The graphs on the right compare the cost of direct summation versus the NFFT summation, and show that the efficiency is greatly improved by the use of the NFFT. Due to their long duration the computational times for the last two steps of the direct summation in the right graph of Figure 3 were only extrapolated (and therefore plotted with dashed lines).

We include Figure 4 to show the effect of the scaling parameter $\mathcal{D}$ in (6) on the saturation error. As the value of $\mathcal{D}$ increases, the saturation

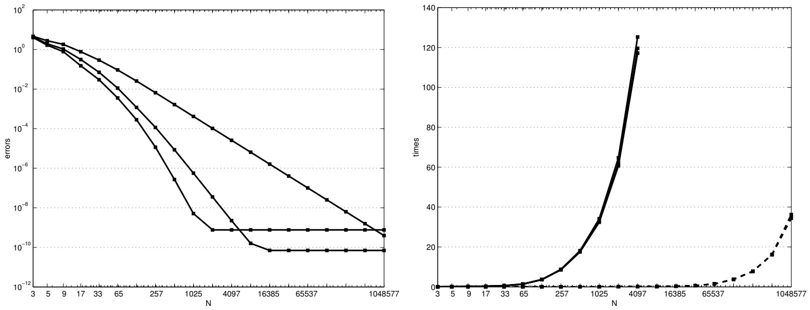

Fig. 1. Convergence and execution times for 1D example (Gaussian, linear and quadratic Gauss-Laguerre generating functions).

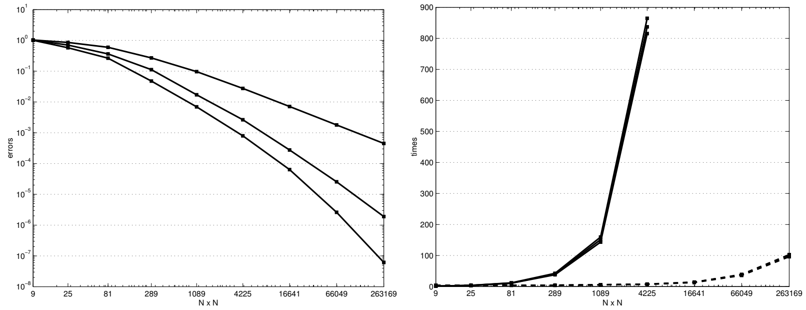

Fig. 2. Convergence and execution times for 2D example (Gaussian, linear and quadratic Gauss-Laguerre generating functions).

occurs at smaller and smaller errors.

§4. Data Compression via Multilevel Residual Iteration

In data compression applications (e.g., using wavelets) one often employs a multi-scale representation of the given data. The following experiments are motivated by the data compression algorithms described in [1] and [8]. In those papers univariate and bivariate splines were used to obtain the desired multi-scale representation. Since we are interested in multidimensional applications we use radial functions instead. However, instead of working in the standard radial basis function interpolation framework, we use the idea of approximate MLS approximation with radial weights described earlier. This has the advantage that we do not have to solve any systems of linear equations.

In order to create a representation of the given data with multiple levels of resolution we apply the following multilevel approximation algorithm.

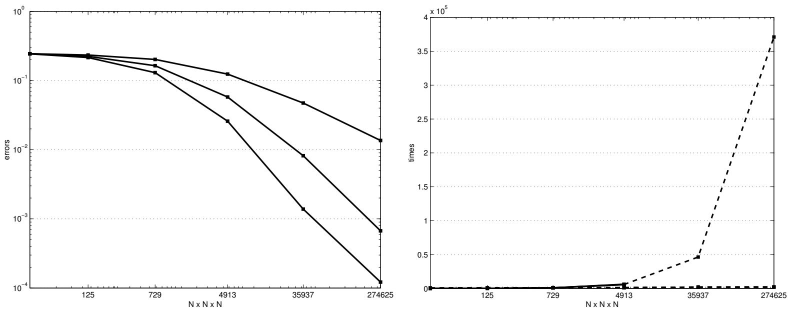

Fig. 3. Convergence and (predicted) execution times for 3D example (Gaussian, linear and quadratic Gauss-Laguerre generating functions).

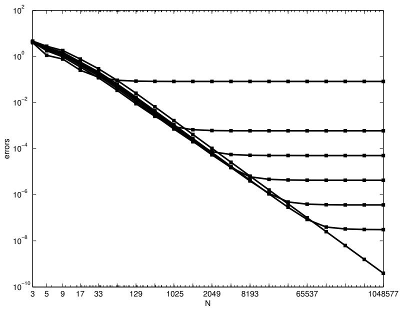

Fig. 4. Comparison of convergence for $\mathcal{D} = 0.5,\ 1.0,\ 1.25,\ 1.5, 1.75,\ 2.0,$ and 3.0 for 1D Gaussian generating functions.

Algorithm: (Multilevel approximation)

```
Create nested sets $\mathcal{X}_1 \subset \cdots \subset \mathcal{X}_K = \mathcal{X} \subset \mathbb{R}^s$.
Initialize $s_0 = 0$.
For $k = 1, 2, \ldots, K$ do
    On $\mathcal{X}_k$ compute residuals
        $r_i^{(k)} = f(\boldsymbol{x}_i^{(k)}) - s_{k-1}(\boldsymbol{x}_i^{(k)}), \quad i = 1, \ldots, N.$
    Then
        $\mathcal{P}r^{(k)}(\boldsymbol{x}) = \sum_{i=1}^{N_k} r_i^{(k)} \Psi_i \left( \frac{\boldsymbol{x}}{\rho_k} \right).$
    Update $s_k = s_{k-1} + \mathcal{P}r^{(k)}$.
End
```

This algorithm updates the current approximation at level k by adding an approximate MLS approximation to the residual at the level $k-1$. At the end of the algorithm s_K approximates the given (data) function f on the set $\mathcal{X}$.

In our experiments we obtain the desired multi-scale representation of the data by using sets of uniformly distributed centers $\mathcal{X}_k$ with mesh-size h_k, so that the generating functions Ψ_i are scaled by the level-dependent factor $\rho_k = \sqrt{\mathcal{D}} h_k$.

In order to reduce the amount of data used to represent f we rely on a thresholding strategy, i.e., we ignore all terms in s_K for which

$$|r_i^{(k)}| < \delta_k = t\gamma^k, \quad k = 1, \ldots, K.$$

Here t is a basic threshold and $\gamma > 1$ is a ratio we use to allow a relaxed tolerance for finer grids. The end result is a multi-scale representation of the data using non-uniformly distributed centers.

It is well known (see any of the standard references on wavelet theory) that this strategy will be justified if $\|\mathcal{P}f\| \simeq \|f\|$, i.e., if our approximation scheme is stable. In other words, one needs to show that small expansion coefficients, $r_i^{(k)}$, indeed contribute only little to the approximation. This amounts to establishing frame bounds for the approximation. Initial steps in this direction are taken in [7].

In Figure 5 along with the corresponding Table 2 we illustrate the behavior of the multilevel approximation algorithm coupled with the thresholding strategy just described, where the generating functions are the Gaussians of Table 1. Table 3 contains analogous data for the first-order Laguerre-Gaussians. The sets $\mathcal{X}_k$ consist of $2^{k+1} \times 2^{k+1}$ uniformly spaced points in $[0,1)^2$, and $K = 5$.

The plots in Figure 5 match (left-to-right, top-to-bottom) the entries in Table 2.

	before	$\gamma = 2$, $t = 10^{-3}$	$\gamma = 4$, $t = 10^{-3}$	$\gamma = 4$, $t = 2.5 \times 10^{-4}$
ℓ_∞	.0446	.0443	.2218	.0585
ℓ_2	.6570	.7044	3.6820	1.4756
% used	100	65.2525	16.0943	36.0269

Tab. 2. Errors for compression based on Gaussian generating functions.

While the bottom-left plot in Figure 5 (corresponding to $\gamma = 4$ and $t = 10^{-3}$) clearly shows a loss of accuracy in the compression, the corresponding plots for the first-order Laguerre-Gaussians are virtually indistinguishable and therefore not included. This fact, along with the entries in the the 4th columns of Tables 2 and 3 clearly reveal the superior approximation properties of the higher-order generating functions. It should

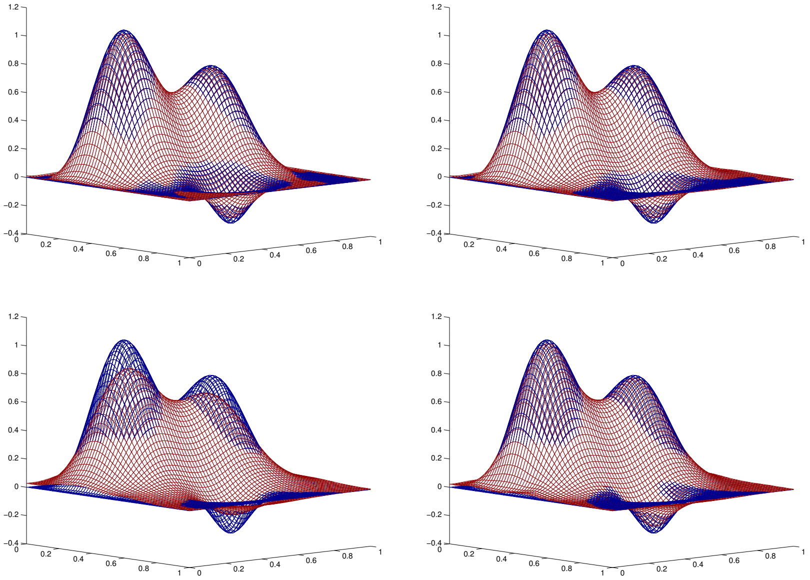

Fig. 5. Compression using Gaussian generating functions.

be emphasized that this significant improvement in approximation quality is achieved at virtually no additional cost.

	before	$\gamma = 2$, $t = 10^{-3}$	$\gamma = 4$, $t = 10^{-3}$	$\gamma = 4$, $t = 10^{-4}$
ℓ_∞	.0120	.0149	.0772	.0206
ℓ_2	.1246	.3374	1.6555	.5062
% used	100	37.7778	12.2559	31.4478

Tab. 3. Errors for compression based on first-order Laguerre-Gaussian generating functions.

Another way to interpret this experiment is that almost the same accuracy as for basic Gaussian quasi-interpolation (right-most entries in Table 2) can be achieved with the first-order Gauss-Laguerre expansion using close to one third the number of terms in the expansion (4th column in Table 3). Thus the Gauss-Laguerre generating functions allow roughly three times as much compression capability as the standard Gaussians.

As mentioned in the introduction it is also possible to view the end result of the multilevel residual iteration as a multivariate radial function approximation to the given data where the scales and centers of the radial functions are selected adaptively by the thresholding strategy.

§5. Closing Remarks

As already pointed out in earlier papers (see, e.g., [3, 5, 6]), the approximate approximation idea provides a matrix-free approach to MLS approximation with a saturation error that can be pushed to within machine accuracy. Moreover, computation with global generating functions can be done efficiently and in a rather general way with the help of the NFFT library. In this manner we are able to efficiently evaluate radial function expansions consisting of millions of terms at millions of points in one, two and three space dimensions.

The multilevel residual iteration based on quasi-interpolation provides an alternative data compression strategy for multivariate data. However, the theoretical foundation of this method (frame bounds) needs more work. Moreover, in order to improve the efficiency of the method it should be coupled with the NFFT algorithm. In any case, this application (along with the experiments reported in [6]) show that matrix-free meshfree approximation for nonuniform multivariate data is possible. However, more work for the case of arbitrarily scattered centers is required.

Some of the problems we plan to adapt our method to are approximation of Hermite data, and solution of PDEs (see [4] for some initial work in this direction).

Acknowledgments. Both authors acknowledge partial support from the National Science Foundation under grant DMS-0073636. We would also like to thank John Erickson for his help with the multilevel data compression experiments.

§6. References

1. D. Assaf, Data compression on the sphere using Faber decomposition, preprint, Vanderbilt University.
2. L. P. Bos and K. Šalkauskas, Moving least-squares are Backus-Gilbert optimal, J. Approx. Theory 59 (1989), 267–275.
3. G. E. Fasshauer, Approximate moving least-squares approximation with compactly supported weights, in *Lecture Notes in Computer Science and Engineering Vol.26: Meshfree Methods for Partial Differential Equations*, M. Griebel and M. A. Schweitzer (eds.), Springer Verlag, 2002, 105-116.
4. G. E. Fasshauer, Approximate moving least-squares approximation for time-dependent PDEs, in *WCCM V, Fifth World Congress on Computational Mechanics* (`http://wccm.tuwien.ac.at`), H. A. Mang, F. G. Rammerstorfer, and J. Eberhardsteiner (eds.), Vienna University of Technology, 2002.

5. G. E. Fasshauer, Approximate moving least-squares approximation: A fast and accurate multivariate approximation method, in *Curve and Surface Fitting: Saint-Malo 2002*, A. Cohen, J.-L. Merrien, and L. L. Schumaker (eds.), Nashboro Press, 2003, 139–148.
6. G. E. Fasshauer, Toward approximate moving least squares approximation with irregularly spaced centers, Computer Methods in Applied Mechanics & Engineering, to appear.
7. G. E. Fasshauer, Dual bases and discrete reproducing kernels: A unified framework for RBF and MLS approximation, submitted.
8. D. Hong and L. L. Schumaker, Surface compression using a space of C^1 cubic splines with a hierarchical basis, preprint, Vanderbilt University.
9. Kunis, S., D. Potts, and G. Steidl, Fast Fourier transforms at nonequispaced knots: A user's guide to a C-library, Universität Lübeck, `http://www.math.uni-luebeck.de/potts/nfft/`, 2002.
10. V. Maz'ya and G. Schmidt, On quasi-interpolation with non-uniformly distributed centers on domains and manifolds, J. Approx. Theory 110 (2001), 125–145.
11. A. Nieslony, D. Potts, and G. Steidl, Rapid evaluation of radial functions by fast Fourier transforms at nonequispaced knots, Universität Mannheim, preprint 2003.
12. D. Potts and G. Steidl, Fast summation at nonequispaced knots by NFFTs, SIAM J. Sci. Comput. 24 (2003), 2013–2037.
13. H. Wendland, Piecewise polynomial, positive definite and compactly supported radial functions of minimal degree, Advances in Comp. Math.4 (1995), 389–396.
14. H. Wendland, Local polynomial reproduction and moving least squares approximation, IMA J. Numer. Anal. 21 (2001), no. 1, 285–300.
15. H. Wendland, *Scattered Data Modelling by Radial and Related Functions*, Cambridge University Press, to appear.

Gregory E. Fasshauer and Jack G. Zhang
Illinois Institute of Technology
Chicago, IL 60616
`fasshauer@iit.edu and zhangou@iit.edu`