Multilevel quasi-interpolation on a sparse grid with the Gaussian

Fuat Usta[1] and Jeremy Levesley[2]

[1] Department of Mathematics, Duzce University ,
Konuralp Campus, 81620, Duzce, Turkey,
fuatusta@duzce.edu.tr,
[2] Department of Mathematics, University of Leicester,
University Road, Leicester, LE1 7RH, United Kingdom,
j.levesley@le.ac.uk

Abstract. Motivated by the recent multilevel sparse kernel-based interpolation (MuSIK) algorithm proposed in [Georgoulis et. al. 2013], we introduce the new quasi-multilevel sparse interpolation with kernels (Q-MuSIK) via the combination technique. The Q-MuSIK scheme achieves better convergence and run time when compared with classical quasi-interpolation. Also, the Q-MuSIK algorithm is generally superior to the MuSIK methods in terms of run time in particular in high-dimensional interpolation problems, since there is no need to solve large algebraic systems. We subsequently propose a fast, low complexity, high-dimensional positive-weight quadrature formula based on Q-MuSIK approximation of the integrand. We present the results of numerical experimentation for both quasi-interpolation and quadrature in high dimensions.

1 Introduction

Over the last half century, numerical methods have obtained much attention, not only among mathematicians but also in the scientific and engineering communities.

High dimensionality usually causes some problems for mathematical modelling on gridded data. The main problem is known as the curse of dimensionality, a term due to Bellmann [4]. There is an exponential relationship between the computational cost of approximation with a given error bound ϵ and the dimension d of the space $\mathbb{R}^d$ for any given problem. For this reason, classical approximation techniques are limited to low dimensions. For example, the complexity of solving an approximation problem on gridded data over a bounded domain $\Omega \in \mathbb{R}^d$ is $\mathcal{O}(N^d)$, where d is the dimension of the input data. Radial basis function (RBF) approximation is one of the tools which is effective in high-dimensions when the data is scattered in its domain; see e.g. [17]. The potential

2010 *Mathematics Subject Classification*: 65F10, 65N55, 65N22, 65D32
Key Words : Quasi-interpolation, multilevel, sparse grids, hyperbolic crosses, quadrature, high dimension.

smoothness of the basis functions (such as Gaussians) means that RBF methods have also been successfully applied for the solution of smooth partial differential equations; see [12].

Another powerful tool for multidimensional problems is quasi-interpolation, which is widely used in scientific computation, mechanics and engineering. Theoretical errors estimates for RBF quasi-interpolation are available, for instance in [2, 20, 21]. A quasi-interpolation technique based on radial basis functions is discussed in the sequel.

Given a continuous function u on $\mathbb{R}^d$, the quasi-interpolant $Q_h u$, $h > 0$, is a superposition of dilated and shifted versions of a given generator function ν on $\mathbb{R}^d$ using as coefficients the samples of u on the lattice $h\mathbb{Z}^d$

$$Q_h u(\mathbf{x}) := \sum_{\mathbf{z} \in h\mathbb{Z}^d} u(h\mathbf{z}) \nu(h^{-1}\mathbf{x} - \mathbf{z}). \tag{1}$$

In this paper $\nu = \mu_c$ is the Gaussian function

$$\mu_c(\mathbf{x}) = \frac{\exp\left(-\|\mathbf{x}\|^2/(2c^2)\right)}{(c\sqrt{2\pi})^d}. \tag{2}$$

We will truncate the expansion (1) expansion so that u is evaluated only inside the unit cube; see Section 2. The kernel μ_c of the quasi-interpolant must have integral 1 in order that it be an approximate identity. This is what motivates the choice of normalisation constant in the above definition. We also wish to have the flexibility to chose a particular width of Gaussian. We will write $\mu = \mu_1$.

The main advantage of quasi-interpolation is that we do not need to solve a linear system as would have to do with interpolation. This means that the approximation process is much faster. We compare our results with the analagous interpolation method, MuSIK, described in [9] and see that they give more accurate results in the same amount of time.

We should also comment that we are aiming at approximation of smooth functions, so that we choose a smooth basis function. In high dimensional applications low dimensional non-smooth artefacts become more difficult to see, so that functions, from the point of view of an observer, smooth out. We should also observe that to test the uniform approximation of a method in ten dimensions on the unit cube would require 10^{10} points for very low resolution in each direction. Hence we can have no confidence that the results we give are appropriate. We explore this issue in the numerical examples. Thus, in high dimensions we observe the approximation of functionals of the solution, in this case, integrals.

Convergence for quasi-interpolation using the Gaussian has been discussed [14, 15], where they coin the phrase approximate approximation. This refers to the fact that low degree polynomials are not reproduced by standard quasi-interpolation with Gaussians. In [3] the authors generate a new kernel in (1) using a linear combination of Gaussians, and then balance the choice of width of the Gaussian and the parameter h in (1) in order to gain almost optimal error estimates. In Müller and Varnhorn [16], and Chen and Cao [5, 6] researchers study

convergence on a compact interval. Here we present our algorithm and numerical results, the analysis of the algorithm will be done in a series of subsequent papers, starting with [13].

Our algorithm is based on the sparse grid method introduced by Zenger [22] in 1991 as a solution for PDEs. These techniques have also been used for approximation and interpolation. It can be seen that these methods are similar to the hyperbolic cross notion of Babenko [1]. In 2012, Georgoulis, Levesley and Subhan [9] introduced a new sparse grid kernel-based interpolation technique which circumvents both computational complication and conditioning problems using. The same basic algorithm was implemented by Schreiber [18], but was not effective because Gaussians with a fixed width were used. Schemes which retain the same shape of basis function while the points get more dense are termed non-stationary, and non-stationary Gaussian interpolation is known to be ill-conditioned. The innovation in [9] was to use stationary interpolation with Gaussians (see the scheme in Section 2), mimicking the sparse-grid scheme with b-splines (this scheme is called SIK). However, SIK with Gaussians does not converge, in contract to the b-spline case. In order to gain convergence they used an iterative correction scheme, interpolating the residual from the previous level at the next level. This scheme is called MuSIK, and is observed in [9] to work very well. In this paper we use the same scheme with quasi-interpolation Q-SIK and Q-MuSIK respectively instead of SIK and MuSIK, and observe similar good results in high dimensions.

Multilevel techniques have been suggested by a number of researchers. The first to consider the multilevel approximation were Floater and Iske [8]. They combined a thinning algorithm and compactly supported radial basis function interpolation. Furthermore, multilevel interpolation techniques enable us to combine the benefits of stationary and non-stationary standard RBFs interpolation, such that this leads to an accelerated convergence. Other researchers have since contributed the multilevel interpolation literature [10, 11, 19].

MuSIK has been extended in [7] both theoretically and numerically. More detailed this study showed that SIK and accordingly MuSIK scheme are interpolatory for the special case of scaled Gaussian kernels. In addition to this a numerical integration algorithm is also proposed in [7], based on interpolating the (high-dimensional) integrand. A series of numerical examples are presented, highlighting the practical applicability of the proposed algorithm for both interpolation and quadrature for up to 10-dimensions. We compare our results with these.

2 Sparse quasi-interpolation with kernels

Let $\Omega := [0,1]^d$, $d \geq 2$, and let $u : \Omega \to \mathbb{R}$. In order to describe the sparse grid construction, we need to introduce some multi-index notation. Throughout this and the next sections we will use $\mathbf{l} := (l_1, \ldots, l_d) \in \mathbb{N}^d$ and $\mathbf{i} := (i_1, \ldots, i_d) \in \mathbb{N}^d$ as a spatial position. For the above multi-indices, the family of directionally uniform grids $\Upsilon_{\mathbf{l}}$ on Ω can be described with mesh size $h_{\mathbf{l}}; = 2^{-\mathbf{l}} = (2^{-l_1}, \ldots, 2^{-l_d})$.

That is, the family of grids $\Upsilon_{\mathbf{l}}$ consists of the points

$$\mathbf{x}_{\mathbf{l},\mathbf{i}} := (x_{l_1,i_1}, \ldots, x_{l_d,i_d}), \tag{3}$$

where $x_{l_p,i_p} := i_p \cdot h_{l_p} = i_p \cdot 2^{-l_p}$ and $i_p \in \{0, 1, \ldots, 2^{l_p}\}$. These grids may have different mesh sizes for each coordinate direction. In addition to this, one can calculate the number of nodes $\mathcal{P}_{\mathbf{l}}$ in $\Upsilon_{\mathbf{l}}$ by using the formula

$$\mathcal{P}_{\mathbf{l}} := \prod_{p=1}^{d} (2^{l_p} + 1). \tag{4}$$

For instance, should one choose $h_{l,p} = 2^{-n}$, for all $p = 1, \ldots, d$, the Cartesian full grid converts to a uniform full grid denoted by $\Upsilon^{n,d}$ with $\mathcal{P} = (2^n + 1)^d$, where n is the grid level. We also consider the following subset of $\Upsilon^{n,d}$,

$$\Upsilon_{n,d} := \bigcup_{|\mathbf{l}|_1 = n+d-1} \Upsilon_{\mathbf{l}} \tag{5}$$

with $|\mathbf{l}|_1 := l_1 + l_2 + \cdots + l_d$, which will be referred to as the sparse grid of level n in d dimensions. We refer to Figure 1 for a visual representation of (5) for $n = 4$ and $d = 2$. Notice that there is some redundancy in this definition as some grid points are included in more than one sub-grid.

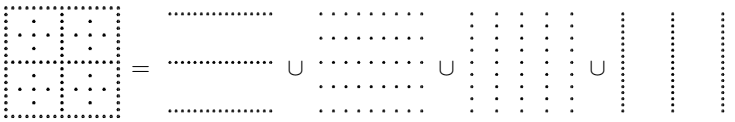


Fig. 1: Sparse grid $\Upsilon_{4,2}$ via (5).

As can easily be seen, the sparse grid treatment itself already requires the use of anisotropic basis functions since interpolation of data with anisotropic distribution of data sites in the domain requires special consideration. For this reason, we will use anisotropic Gaussian functions for Q-SIK. Let $A_{\mathbf{l}} = \operatorname{diag}(2^{l_1}, 2^{l_2}, \cdots, 2^{l_d})$. Then define

$$\mu_{c,\mathbf{l}}(\mathbf{x}) = \mu_c(A_{\mathbf{l}}\mathbf{x}).$$

For each $\mathbf{l}$ we construct a quasi-interpolant

$$Q_{\mathbf{l}}u(\mathbf{x}) = \sum_{\mathbf{z} \in \Upsilon_{\mathbf{l}}} u(\mathbf{z}) \mu_{c,\mathbf{l}}(\mathbf{x} - \mathbf{z}), \quad \mathbf{x} \in \mathbb{R}^n. \tag{6}$$

Then we obtain the Q-SIK approximation by the combination technique:

$$Q_{n,d}u(\mathbf{x}) = \sum_{q=0}^{d-1} (-1)^q \binom{d-1}{q} \sum_{|\mathbf{l}|_1 = n+(d-1)-q} Q_{\mathbf{l}}u(\mathbf{x}). \tag{7}$$

For example, in 2-D the Q-SIK approximation is

$$Q_{n,2}u(\mathbf{x}) = \sum_{|\mathbf{l}|_1=n+1} Q_{\mathbf{l}}u(\mathbf{x}) - \sum_{|\mathbf{l}|_1=n} Q_{\mathbf{l}}u(\mathbf{x}). \tag{8}$$

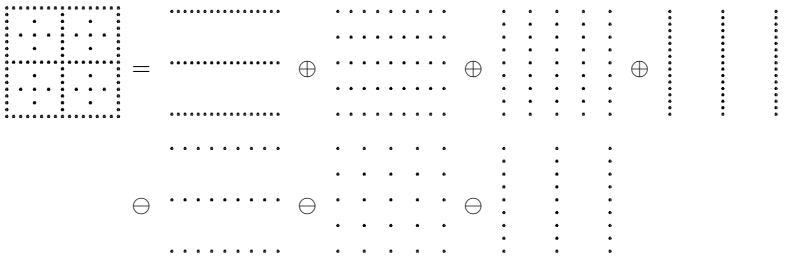


Fig. 2: The construction of $\Upsilon_{4,2}$ as an algebraic sum.

The key idea here is that we view the sparse grid in Figure 1 as an algebraic sum of grids, as in Figure 2. We can summarize the algorithm of sparse kernel-based interpolation as follows.

Algorithm 1 Sparse quasi-interpolation with kernels

Require: Sparse grid data $\{(\mathbf{x}_i, u_i), u_i = u(\mathbf{x}_i), i = 1, \cdots N\}$

1: Create the sub-grids for each level $\mathbf{l} \in \mathbb{N}^d$, $|\mathbf{l}|_1 = n, \ldots, n + (d-1)$ and $\Upsilon_{\mathbf{l}}$ with mesh size $h_{\mathbf{l}}$.
2: Solve the anisotropic sub-grid quasi-interpolation problems $Q_{\mathbf{l}}u(\mathbf{x}) = \sum_{\mathbf{z}\in\Upsilon_{\mathbf{l}}} u(\mathbf{z})\mu_{\mathbf{l}}(\mathbf{x}-\mathbf{z})$.
3: Combine the all sub-grid quasi-interpolation problems obtained above $Q_{n,d}u(\mathbf{x}) = \sum_{q=0}^{d-1}(-1)^q \binom{d-1}{q} \sum_{|\mathbf{l}|_1=n+(d-1)-q} Q_{\mathbf{l}}u(\mathbf{x})$.
4: **return** The Q-SIK approximation $Q_{n,d}u(\mathbf{x})$

As can be seen from the above description, the Q-SIK method is very amenable to parallel computation since each sub-approximation problem can be solved independently. Therefore this method can be applied in a computational cluster or distributed across workstations. Thus, the Q-SIK method provides us with computationally cheap approximation, especially for multidimensional problems.

3 Multilevel sparse quasi-interpolation with kernels

We will see in Section 5 (and the same is observed in SIK), Q-SIK does not converge, even for the function $u \equiv 1$. In the case of the infinite grid in one

dimension, this can be seen easily. For a $1/n$ spaced grid the quasi-interpolant is

$$Q_n(x) = \sum_{l=-\infty}^{\infty} \mu(nx - l).$$

This function is $1/n$-periodic and even and so has a Fourier series

$$Q_n(x) = \sum_{k=0}^{\infty} a_k \cos(2\pi knx),$$

where, for $k \geq 1$,

$$\begin{aligned} a_k &= n \int_0^{1/n} Q_n(x) \cos(2\pi knx) dx \\ &= n \int_0^{1/n} \left(\sum_{l=-\infty}^{\infty} \mu(nx - l) \right) \cos(2\pi knx) dx \\ &= \int_0^{1} \left(\sum_{l=-\infty}^{\infty} \mu(y - l) \right) \cos(2\pi ky) dy \\ &= \int_{-\infty}^{\infty} \mu(y) \cos(2\pi ky) \\ &= 2 \exp(-2\pi^2 k^2), \end{aligned}$$

the Fourier coefficient of the normal distribution. Additionally $a_0 = 1$. Hence, for any n

$$Q_n(0) = 1 + 2 \sum_{k=0}^{\infty} \exp(-2\pi^2 k^2),$$

so that there is no convergence as $n \to \infty$.

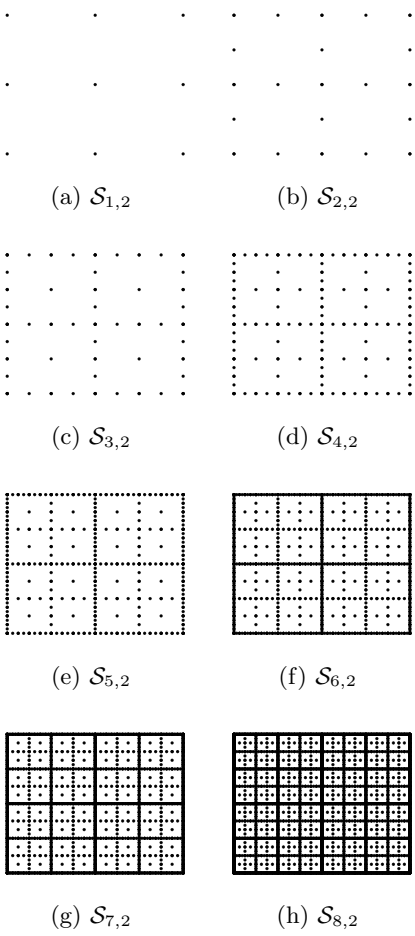
Thus we adopt a multilevel refinement strategy, which we call Q-MuSIK. The only difference between MuSIK and Q-MuSIK is that we will use the Q-SIK method instead of the SIK method for each sub-grid approximation problem.

In contrast to the single level Q-SIK method discussed in the previous section, we will now use nested sparse grids which are sorted from the lower level to the higher level. In other words, sparse grids with increasingly greater data densities provide us with a hierarchical decomposition of the sparse grid data,

$$\Upsilon_{i,d} \subset \Upsilon_{i+1,d}, \quad i = 1, 2, \cdots .$$

The hierarchical decomposition in two dimensions can be seen in Figure 3.

After obtaining the sparse grid decomposition, the sparse grid interpolation $Q_{n_0,d}$ needs to be evaluated for level 1. In other words, the coarsest level approximation is $S_{0,d}u = Q_{n_0,d}u$. Then Δ_j needs to be calculated for each



(a) $\mathcal{S}_{1,2}$ (b) $\mathcal{S}_{2,2}$

(c) $\mathcal{S}_{3,2}$ (d) $\mathcal{S}_{4,2}$

(e) $\mathcal{S}_{5,2}$ (f) $\mathcal{S}_{6,2}$

(g) $\mathcal{S}_{7,2}$ (h) $\mathcal{S}_{8,2}$

Fig. 3: Sparse grid decomposition for levels 1 to 8 in 2 dimensions.

level using Q-SIK on the residual $\Delta_j = Q_{n_0+j,d}(u - S_{j-1,d}u)$ on $\Upsilon_{n_0+j,d}$, and $S_{j,d}u = S_{j-1,d}u + \Delta_j$, for $1 \leq j \leq n$. This multilevel iterative refinement algorithm is called Q-MuSIK.

Algorithm 2 Quasi multilevel sparse interpolation with kernels

Require: Sparse grid data decomposition and function u

1: Initialize the first interpolation value at zero with Q-SIK, that is $S_{0,d}u = 0$.
2: Construct the nested sparse grids as $\Upsilon_{n_0,d} \subset \Upsilon_{n_0+1,d} \subset \cdots \subset \Upsilon_{n_0+n,d}$.
3: For every value of $j = 1, 2, \cdots, n$,

- Solve $\Delta_j = Q_{n_0+j,d}(u - S_{j-1,d}u)$ on $\Upsilon_{n_0+j,d}$
- Update $S_{j,d}u = S_{j-1,d}u + \Delta_j$

4: **return** Sequence of progressive approximations, $S_{1,d}u, S_{2,d}u, \ldots, S_{n,d}u$ to u.

4 Q-MuSIK Quadrature

Now, we introduce a new quadrature formula by integrating the Q-MuSIK approximation over the unit cube:

$$I_{\mathbf{l}}u(\mathbf{x}) = \sum_{\mathbf{z}\in\Upsilon_{\mathbf{l}}} u(\mathbf{z}) \int_{[0,1]^d} \mu_{\mathbf{l}}(\mathbf{x}-\mathbf{z})\, d\mathbf{x}.$$

In the case when the Gaussians are tensor products, namely, $\mu_{\mathbf{l}}(\mathbf{x}-\mathbf{z}) = \mu_{l_1}(x_1 - z_{l_1}) \times \mu_{l_2}(x_2 - z_{l_2}) \times \cdots \times \mu_{l_d}(x_d - z_{l_d})$, we can straightforwardly calculate the weights as follows:

$$\int_{[0,1]^d} \mu_{\mathbf{l}}(\mathbf{x}-\mathbf{z})\, d\mathbf{x} = \prod_{i=1}^{d} \int_0^1 \mu_{l_i}(x_i - z_{l_i}) dx_i.$$

The integral of a univariate Gaussian is of the form

$$\phi_l(z_l) = \int_0^1 \mu_l(x - z_l)dx = \int_0^1 (2\pi)^{-1/2} \exp(-(A_l(x - z_l))^2)dx,$$

for some A_l, $m_l \in \mathbb{R}$, which, upon the change of variables $\xi = A_l(x - z_l)$, gives

$$\phi_l(z_l) = \frac{1}{\sqrt{\pi}} \int_{-A_l z_l}^{A_l(1-z_l)} \exp(-\xi^2)d\xi = \frac{1}{2\sqrt{2}}[\mathrm{erf}(A_l(1 - z_l)) - \mathrm{erf}(-A_l z_l)],$$

with the error function erf defined by

$$\mathrm{erf}(x) = \frac{2}{\sqrt{\pi}} \int_0^x e^{-t^2} dt. \tag{9}$$

Here the error function can be calculated to arbitrary precision.

In a similar way, all the sub-grids for each level can be constructed in a similar manner to the method discussed in previous sections. In other words, we will use the grid set $\Upsilon_{\mathbf{l}}$ with mesh size $h_{\mathbf{l}}$ constructed via sparse grids. In order to then solve the each sub-grid quadrature problem, we need to use the

anisotropic quasi-quadrature technique, which uses sparse grids. In detail, by using the quadrature construction from the previous section, the anisotropic quasi-quadrature formula of u at the grids $\Upsilon_{\mathbf{l}}$ can be defined by

$$I_{\mathbf{l}}u(\mathbf{x}) = \sum_{\mathbf{z}\in\Upsilon_{\mathbf{l}}} u(\mathbf{z}) \prod_{i=1}^{d} \phi_{l_i}(z_{l_i}).$$

where

$$\phi_{l_i}(z_{l_i}) = \frac{1}{2\sqrt{2}}[\mathrm{erf}(A_{l_i}(1 - z_{l_i})) - \mathrm{erf}(-A_{l_i} z_{l_i})]$$

We then obtain the Q-SIK quadrature formula by combining all sub quadrature problems with the combination technique, that is, the Q-SIK quadrature is defined by

$$I_{n,d}u(\mathbf{x}) = \sum_{q=0}^{d-1} (-1)^q \binom{d-1}{q} \sum_{|\mathbf{l}|_1 = n+(d-1)-q} I_{\mathbf{l}}u(\mathbf{x}).$$

The multilevel quadrature scheme is achieved by iterative refinement exactly as described in the previous section.

5 Numerical Experiments

In this section, we present a collection of numerical experiments for approximation for $d = 2, 3, 4$, where the implementation of Q-SIK and Q-MuSIK algorithms is assessed and compared against both the standard quasi interpolation (QI) on uniform full grids and its standard multilevel version (MLQI). It is worth remarking that in higher dimensions it becomes harder to calculate approximation errors. In all examples we use the level basis function $\mu_{0.4}$. Thus, for higher dimensions we move to assessing the errors in quadrature, which is the subject of the final examples, where we consider $d = 4, 5, 10$. We compare Q-MuSIK with MuSIK for these examples.

5.1 2-*D* Example

The Q-SIK and Q-MuSIK algorithm with Gaussian basis functions is applied to the test function $P^{2d}(\mathbf{x}) : [0, 1]^2 \to \mathbb{R}$, with

$$P^{2D}(\mathbf{x}) := \frac{1.25 + \cos(5.4x_2)}{6 + 6(3x_1 - 1)^2} \tag{10}$$

Here SGs stands for the number of sparse grid centers used, NoVs represents the total number of nodes visited in the Q-SIK and Q-MuSIK, Maxerror and RMS-error are the maximum norm and root-mean-square (L^2-norm) errors,

SGs	NoVs	Q-SIK		Q-MuSIK	
		Max Error	Rms Error	Max Error	Rms Error
9	**9**	**1.48e-1**	**4.63e-2**	**1.48e-1**	**4.63e-2**
21	**39**	**8.60e-2**	**2.08e-2**	**4.37e-2**	**1.43e-2**
49	**109**	**4.92e-2**	**9.56e-3**	**1.61e-2**	**4.28e-3**
113	**271**	**3.32e-2**	**6.26e-3**	**7.66e-3**	**1.31e-3**
257	**641**	**2.75e-2**	**5.60e-3**	**3.26e-3**	**4.09e-4**
577	**1475**	**2.63e-2**	**5.48e-3**	**1.33e-3**	**1.27e-4**
1281	**3333**	**2.59e-2**	**5.45e-3**	**5.57e-4**	**3.73e-5**
2817	**7431**	**2.66e-2**	**5.43e-3**	**1.77e-4**	**1.01e-5**
6145	**16393**	**2.61e-2**	**5.42e-3**	**4.77e-5**	**2.88e-6**

Table 1: P^{2D}:Q-SIK and Q-MuSIK error on an equally spaced 160×160 evaluation grid in 2 dimensions.

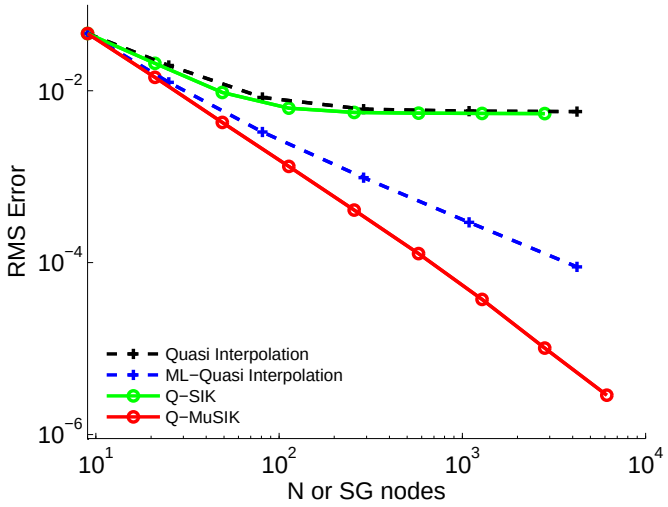


Fig. 4: P^{2D}:RMS error versus N (QI, MLQI) or SG (Q-SIK, Q-MuSIK) with QI (black), Q-SIK (green), MLQI (blue) and Q-MuSIK (red) on a 160×160 uniform grid in 2 dimensions.

respectively, evaluated on a 160×160 uniform grid. We see in Table 1 that, as expected, Q-SIK does not converge. In other words, the error in the Q-SIK method remains stable after a few levels. However we see in the same table that Q-MuSIK converges.

In Figure 4 we graph the results of quasi interpolation, multilevel quasi interpolation, Q-SIK and Q-MuSIK. Obviously, quasi interpolation and Q-SIK do

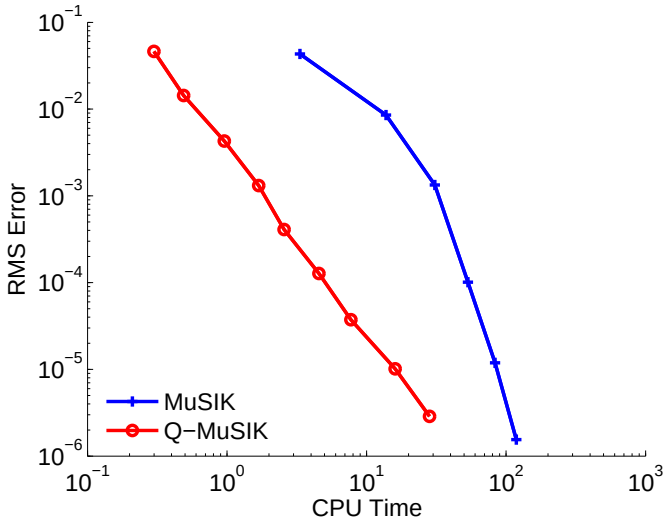


Fig. 5: P^{2D}:RMS error versus computational time for MuSIK (blue) and Q-MuSIK (red) on a 160×160 uniform grid in 2 dimensions.

not converge to the target functions. However their multilevel versions shows good convergence behaviour.

In Figure 5 we compare the MuSIK and Q-MuSIK results with regard to computational time. According to this figure Q-MuSIK reaches the same error level as MuSIK in a shorter time. This is because quasi interpolation does not require the solution of any large algebraic system as does an interpolatory scheme such as MuSIK.

5.2 3-*D* Example

In this section we approximate two functions, the first smooth and the second with derivative singularities.

Let

$$G^{3D}(\mathbf{x}) := \frac{18}{\pi} e^{-(x_1^2+81x_2^2+x_3^2)}. \tag{11}$$

We test the Q-MuSIK approximation to this function on an equally spaced $50 \times 50 \times 50$ evaluation grid. The corresponding results to those in 2 dimensions can be seen in Figures 6 and 7. Again we see that Q-MuSIK outperforms MuSIK.

Now consider

$$H^{3D}(\mathbf{x}) = \prod_{i=1}^{3} (x_i - 1/2)_+,$$

where $(x)_+ = x$ if x is positive and is zero otherwise. In Figure 8 we see that the error appears to behave better than for the smoother function G^{3D}. The

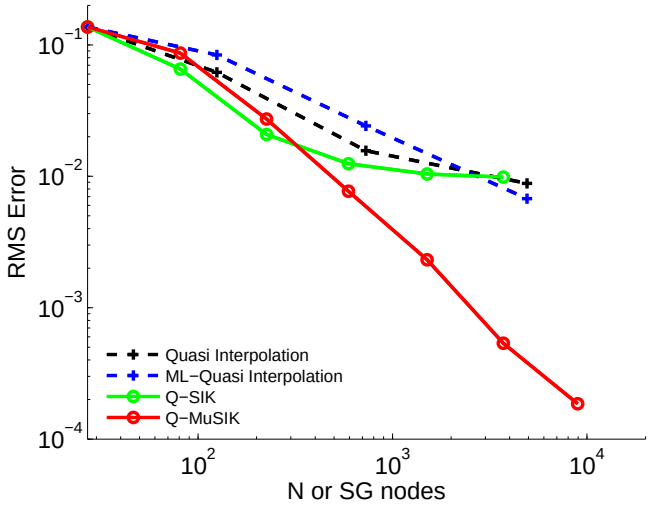


Fig. 6: G^{3D}:RMS error versus N (QI, MLQI) or SG (Q-SIK, Q-MuSIK), QI (black), Q-SIK (green), MLQI (blue) and Q-MuSIK (red) on a $50 \times 50 \times 50$ uniform grid in 3 dimensions.

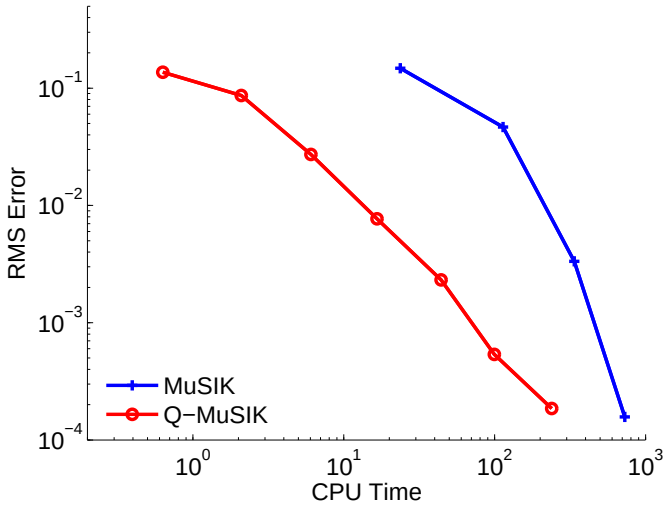


Fig. 7: G^{3D}:RMS error versus computational time for MuSIK (blue) and Q-MuSIK (red) on a $50 \times 50 \times 50$ uniform grid in 3 dimensions.

error is tested on a $50 \times 50 \times 50$ uniform grid. If we test on a $100 \times 100 \times 100$ uniform grid we see different behaviour for the maximum error (and similarly

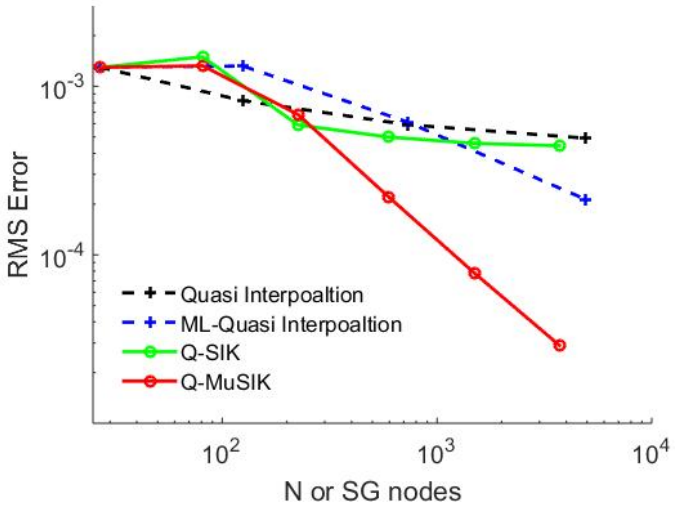


Fig. 8: H^{3D}: RMS error versus N (QI, MLQI) or SG (Q-SIK, Q-MuSIK), QI (black), Q-SIK (green), MLQI (blue) and Q-MuSIK (red) on a $50 \times 50 \times 50$ uniform grid.

for RMS): On the finer grid with 10^6 test points we see that the error in the less

SGnode	G^{3D}	H^{3D}
27	1.25e+00	9.88e-01
81	8.27e-01	5.11e-01
225	2.21e-01	4.54e-01
593	6.39e-02	9.01e-02
1505	3.02e-02	6.46e-02
3713	1.45e-02	4.11e-02

Table 2: Q-MuSIK maximum error on $100 \times 100 \times 100$ for G^{3D} and H^{3D}.

smooth H^{2D} is greater. This emphasises the point we made in the introduction that error testing in high dimensions becomes more and more difficult, and that singularities in high dimensional functions are hard to locate.

5.3 4-*D* Example

Let us now consider a 4 dimensional non-tensor product test function

$$H^{4D}(\mathbf{x}) = \sin(x_1^2 x_2^2 x_3^2 x_4^2). \tag{12}$$

In Figure 9 the root mean-square error for QI, MLQI, Q-SIK and Q-MuSIK, respectively, for $H^{4D}(\mathbf{x})$, are plotted against the number of data sites. We see similar performance as in 2 and 3 dimensions.

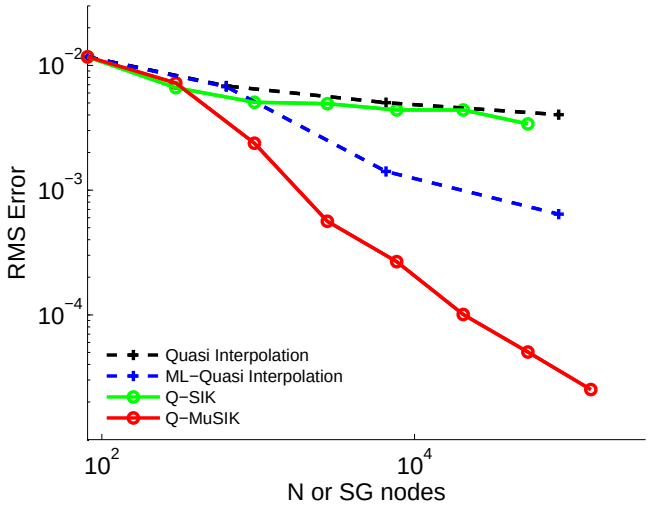


Fig. 9: H^{4D}:RMS error versus N (QI, MLQI) or SG (Q-SIK, Q-MuSIK): QI (black), Q-SIK (green), MLQI (blue) and Q-MuSIK (red) on a $21 \times 21 \times 21 \times 21$ uniform grid.

5.4 Quadrature Examples

In this section we give a number of examples for quadrature. We use two tensor product examples, respectively in 5 and 10 dimensions, and two non-tensor product examples. We consider the following non-tensor product example

$$M^{3D}(\mathbf{x}) = \sin(x_1 x_2 x_3). \tag{13}$$

In Figure 10 we give the QI, MLQI, Q-SIK and Q-MuSIK quadrature results for $M^{3D}(\mathbf{x})$ with regard to the corresponding exact values of the integral of $M^{3D}(\mathbf{x})$ on the domain $[0,1]^3$, which is (with 16 digits accuracy) 0.122434028745371.

In Figure 11 we compare the MuSIK quadrature and Q-MuSIK quadrature in terms of absolute error versus degree of freedom and absolute error versus evaluation time. The results in MuSIK were produced using the initial Gaussian basis function with standard deviation 0.45. We note that choice of this parameter is restricted in MuSIK by the conditioning of the interpolation problem. No such restriction is present for Q-MuSIK as there are no linear systems to invert. As

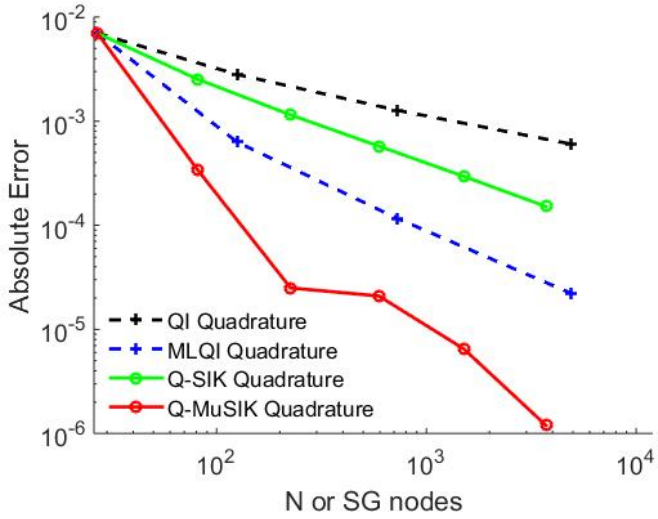


Fig. 10: Absolute error versus N (QI, MLQI) or SG (Q-SIK, Q-MuSIK) nodes on the domain $[0,1]^3$.

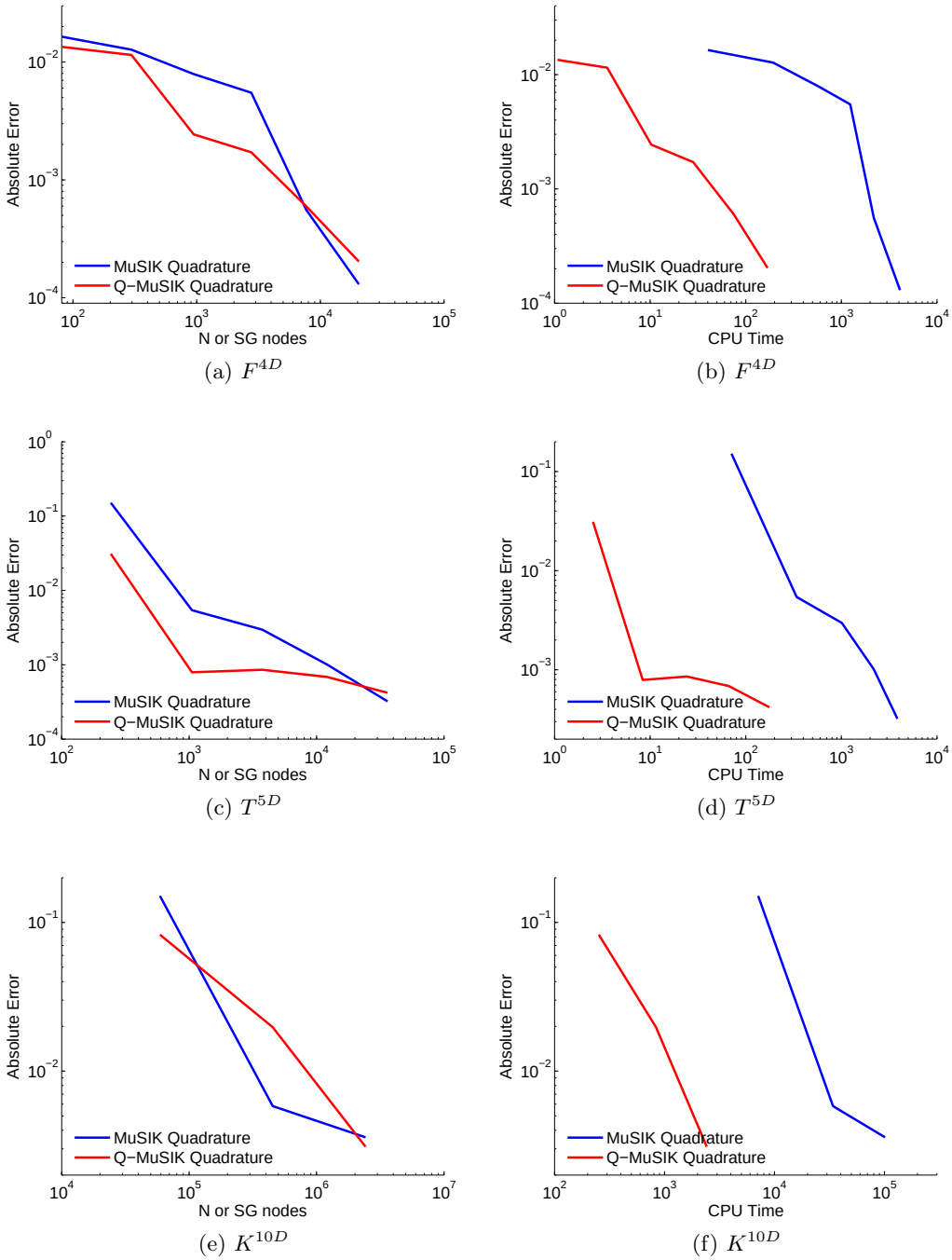
discussed in the previous examples Q-MuSIK has better performance especially for small n. In these comparisons we have used four, five and ten dimensional test functions which are given below:

$$\begin{aligned} F^{4D}(\mathbf{x}) = &\frac{3}{4}e^{-(9x_1-2)^2-(9x_2-2)^2-(9x_3-2)^2/4-(9x_1-2)^2/8} \\ &+ \frac{3}{4}e^{-(9x_1+1)^2/49-(9x_2+1)^2/10-(9x_3+1)^2/29-(9x_1+1)^2/39} \\ &+ \frac{1}{2}e^{-(9x_1-7)^2/4-(9x_2-3)^2-(9x_3-5)^2/2-(9x_1-5)^2/4} \\ &- \frac{1}{5}e^{-(9x_1-4)^2/4-(9x_2-7)^2-(9x_3-5)^2-(9x_1-5)^2}, \end{aligned}$$

$$T^{5D}(\mathbf{x}) = \prod_{i=1}^{4} \max(x_i - 1/2),$$

$$K^{10D}(\mathbf{x}) = \prod_{i=1}^{10} e^{-x_i(1-x_i)},$$

and the corresponding exact integral values of these functions on the domain $[0,1]^4$, $[0,1]^5$ and $[0,1]^{10}$ with 8 digits accuracy are 0.07766696, 0.001953125 and 0.19427907 respectively. These results confirm that the Q-MuSIK method provide us more rapid results for the same amount of error as the compared methods.



(a) F^{4D} (b) F^{4D}

(c) T^{5D} (d) T^{5D}

(e) K^{10D} (f) K^{10D}

Fig. 11: Absolute error versus SG and CPU Time(Q-MuSIK quadrature, Q-MuSIK quadrature) nodes using Gaussian basis functions: MuSIK quadrature (blue) and Q-MuSIK quadrature (red) on the domain $[0,1]^{4,5,10}$.

6 Concluding remarks

A new quasi multilevel sparse interpolation with Gaussian kernel, Q-MuSIK, is proposed and tested. One of the most significant positive aspects of quasi-interpolation is that it can yield a solution directly with no need to solve large algebraic systems, and the associated quadrature rule has positive weights. Numerical experiments suggest that Q-MuSIK is faster than the analagous interpolation algorithm, MuSIK, to give the same level of approximation accuracy. The algorithm is targeted at high dimensional approximation, where functions are typically observed as smooth. This motivates the choice of the Gaussian kernel as a basis function.

References

1. K. I. Babenko, *Approximation by trigonometric polynomials in a certain class of periodic functions of several variables*, Soviet Mathematics Doklady, 1 (1960), pp. 672–675.
2. R. Beatson and M. Powell, *Univariate multiquadric approximation: quasi-interpolation to scattered data*, Constructive Approximation, 8 (1992), p. 275288.
3. R. K. Beatson and W. A. Light, *Quasi-interpolation in the absence of polynomial reproduction*, in Numerical Methods of Approximation Theory, Vol. 9, D. Braess and L. L. Schumaker, eds., Basel, 1992, Birkhauser, pp. 21–39.
4. R. Belmann, *Adaptive Control process: a guide tour*, Princeton, 1961.
5. Z. Chen and F. Cao, *Global errors for approximate approximations with gaussian kernels on compact intervals*, Applied Mathematics and Computation, 217 (2010), p. 725734.
6. Z. Chen, F. Cao, and J. Hu, *Error estimates of quasi-interpolation and its derivatives*, Journal of Computational and Applied Mathematics, 236 (2012), pp. 3137–3146.
7. P. Dong, E. H. Georgoulis, J. Levesley, and F. Usta, *On nodal exactness of sparse grid interpolation in the absence of nested subspaces and application to high dimensional quadrature*. preprint.
8. M. S. Floater and A. Iske, *Multistep scattered data interpolation using compactly supported radial basis functions*, Journel of Computational Applied Mathematics, 73(1-2) (1996), pp. 65–78.
9. E. H. Georgoulis, J. Levesley, and F. Subhan, *Multilevel sparse kernel-based interpolation*, SIAM Journal of Scientific Computing, 35 (2013), pp. 815–832.
10. Q. T. L. Gia, I. H. Sloan, and H. Wendland, *Multiscale analysis in sobolev spaces on the sphere*, SIAM Journal on Numerical Analysis, 48 (2010), pp. 2065–2090.
11. S. J. Hales and J. Levesley, *Error estimates for multilevel approximation using polyharmonic splines*, Numerical Algorithms, 30(1) (2002), pp. 1–10.
12. E. J. Kansa, *Multiquadrics — a scattered data approximation scheme with applications to computational fluid-dynamics. ii. solutions to parabolic, hyperbolic and elliptic partial differential equations*, Computers and Mathematics with Applications, 19(8-9) (1990), pp. 147–161.
13. J. Levesley and S. Hubbert, *Multilevel quasi-interpolation on a sparse grid with the gaussian*, http://arxiv.org/abs/1609.02457, 2017.

14. V. Mazya and G. Schmidt, *On approximate approximations using gaussian kernels*, IMA Jornal of Numerical Analaysis, 16 (1996), p. 1329.
15. ——, *On quasi-interpolation with non-uniformly distributed centers on domains and manifolds*, Jornal of Approximation Theory, 110 (2001), p. 125145.
16. F. Muller and W. Varnhorn, *Error estimates for approximate approximations with gaussian kernels on compact intervals*, Jornal of Approximation Theory, 145 (2007), p. 171181.
17. M. J. D. Powell, *The theory of radial basis function approximation in 1990*, no. II, Oxford University Press, New York, 1992.
18. A. Schreiber, *Smolyak's method for multivariate interpolation*, PhD thesis, Department of Mathematics and Informatics, University of G ottingen, 2001.
19. H. Wendland, *Multiscale analysis in sobolev spaces on bounded domains*, Numerische Mathematik, 116 (2010), pp. 493–517.
20. Z. Wu and J. Liu, *Generalized strang-fix condition for scattered data quasi-interpolation*, Advances in Computational Mathematics, 23 (2005), pp. 201–214.
21. Z. Wu and R. Schaback, *Shape preserving properties and convergence of univariate multiquadric quasi- interpolation*, Acta Mathematicae Applicatae Sinica, 10(1) (1994), pp. 441–446.
22. C. Zenger, *Sparse grids, in Parallel Algorithms for Partial Differential Equations (Kiel, 1990)*, Notes on Numerical Fluid Mechanics, 31 (1991), pp. 241–251.