Sparse Grid Approximation with Gaussians

Thesis submitted for the degree of
Doctor of Philosophy
at the University of Leicester

by

Fuat Usta

Department of Mathematics
University of Leicester
England, United Kingdom

June 2015

Abstract

Motivated by the recent multilevel sparse kernel-based interpolation (MuSIK) algorithm proposed in [Georgoulis, Levesley and Subhan, SIAM J. Sci. Comput., 35(2), pp. A815-A831, 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 in comparison with classical quasi-interpolation; namely, 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 quadrature formula based on Q-MuSIK interpolation of the integrand. We present the results of numerical experimentation for both interpolation and quadrature in $\mathbb{R}^d$, for $d = 2$, $d = 3$ and $d = 4$.

In this work we also consider the convergence rates for multilevel quasi-interpolation of periodic functions using Gaussians on a grid. Initially, we have given the single level quasi-interpolation error by using the shifting properties of Gaussian kernel, and have then found an estimate for the multilevel error using the multilevel algorithm for unit function.

Acknowledgements

Foremost, I would like to express my sincere gratitude to my advisor, Prof. Jeremy Levesley, for his continuous support of my Ph.D studies and research, for his patience, motivation, enthusiasm, and immense knowledge. His guidance has helped me during all my research and the writing of this thesis. I could not have imagined having a better advisor and mentor for my Ph.D.

I am grateful to all the honourable staff members of the mathematics department for their help. I wish to thank Dr. Steve Hales, Dr. Emmanuil Georgoulis and Peter Dong for their valuable ideas, suggestions and discussions throughout my studies in Leicester

I sincerely thank the Turkish Ministry of National Education for providing me with the scholarship to carry out this research.

I reserve my deepest thanks to my dearest wife, Asuman, who was always with me through my PhD journey, and for giving me a lovely daughter. This thesis would not have been possible unless she had continuously encouraged me with her sincere optimism and enthusiasm for my work. I would like to show gratitude to my father and my mother, as well as to my brother and sisters. I love you all, and I am grateful for the bounty of a loving and supportive family.

Last but not least, I am indebted to many of my friends from Leicester for their support. I will always remember the enjoyable time spent together with Gül Aktas, Dr. Sıtkı Aktas, Dr. Ugur Sarı, Yadigar Fırat, İshak Fırat, Burhan Alveroğlu. I would finally like to thank many of my friends from Turkey for their encouragement: Prof. Hasan Nuri Yaşar, Adv. Serap Yaşar, Mehmet Gürsültür, Serkan Tahir Yücel and İsmail Uluçam.

To my beloved wife Asuman and lovely daughter Ela Nisa...

Contents

Abstract i

Acknowledgements ii

1 Introduction 1

1.1 Motivation and Objectives 5

1.2 Quasi-interpolation 8

1.2.1 Quasi-interpolation using Gaussian Kernels 8

1.2.2 Anisotropic Quasi-interpolation 11

1.3 Radial Basis Functions Interpolation 13

1.3.1 Radial Functions 13

1.3.2 Radial Basis Functions Interpolation 17

1.3.3 Anisotropic Radial Basis Functions Interpolation . . . 22

1.4 Main Achievements 24

1.5 Outline 25

2 Hyperbolic Cross Product Spaces and Sparse Grid Techniques 27

2.1 General Idea of Interpolation on a Full Grid 28

2.2 Multi-stage Subspace Decomposition 29

2.2.1 One-dimensional Hierarchical Basis Functions 31

2.2.2 Multidimensional Construction with Tensor Product . 34

2.3 Sparse Grids Technique 37

2.3.1 Approximation Properties of Sparse Grid Spaces 39
2.3.2 Combination Technique 41

3 Multilevel Sparse Kernel Based Interpolation 44
3.1 Sparse interpolation with Kernels 45
3.2 Tensor product kernels give interpolatory schemes 46
3.3 Multilevel sparse interpolation with kernels 54
3.4 Tensor product of univariate cardinal functions 55

4 Quasi Sparse Interpolation with Kernels 57
4.1 Sparse Grid Construction 58
4.2 Quasi sparse interpolation with kernels 59
4.3 Numerical Experiments in 2-D 62
4.3.1 Experiments using Gaussian Tensor Products 65
4.4 Discussion 72

5 Quasi Multilevel Sparse Interpolation with Kernels 74
5.1 Multilevel Quasi interpolation 75
5.2 Quasi multilevel sparse interpolation with kernels 77
5.3 Numerical Experiments in 2-D 80
5.3.1 Experiments using Gaussian Tensor Products 81
5.4 Numerical Experiments in 3-D 87
5.5 Numerical Experiments in 4-D 93
5.6 Discussion 96

6 Error estimation of Multilevel Quasi-Interpolation for Periodic Functions 98
6.1 Error Bounds of Quasi-interpolation for Periodic Functions . . 100

6.2 Pointwise Error Estimation 103
6.3 Error analysis of Multilevel Quasi-interpolation 107

7 A Q-MuSIK-based Multidimensional Quadrature Formula 111
7.1 Quasi-Quadrature Method 114
7.2 Q-SIK Quadrature Formula 115
7.3 Q-MuSIK Quadrature Method 117
7.4 Numerical Experiments in 2-D 118
7.5 Numerical Experiments in 3-D 121
7.6 Numerical Experiments in High Dimensions 121
7.7 Discussion 122

8 Conclusions and Future Work 125
8.1 Conclusions 125
8.2 Future Work 128

Bibliography 129

List of Tables

1.1 One-Dimensional Basis Functions and their Fourier Transform. 8

1.2 Multidimensional Basis Functions and their Fourier Transform. 9

1.3 Piecewise Smooth Radial Basis Functions. 15

1.4 Infinitely Smooth Radial Basis Functions. 16

4.1 Quasi-interpolation results using Gaussian basis functions with shape parameter $d = 0.4$, test function $F_1^{2d}(x, y)$, on an equally spaced 160×160 evaluation grid. 65

4.2 Q-SIK results using Gaussian basis functions with shape parameter $d = 0.4$, test function $F_1^{2d}(x, y)$, on an equally spaced 160×160 evaluation grid. 66

4.3 Quasi-interpolation results using Gaussian basis functions with shape parameter $d = 0.4$, test function $F_2^{2d}(x, y)$, on an equally spaced 160×160 evaluation grid. 68

4.4 Q-SIK results using Gaussian basis functions with shape parameter $d = 0.4$, test function $F_2^{2d}(x, y)$, on an equally spaced 160×160 evaluation grid. 68

4.5 Q-SIK results using Gaussian basis functions with shape parameter $d = 0.4$, test function $F_3^{2d}(x, y)$, on an equally spaced 160×160 evaluation grid. 69

5.1 Q-MuSIK results using Gaussian basis functions with shape parameter $d = 0.4$, test function $F_1^{2d}(x, y)$, on an equally spaced 160×160 evaluation grid. 82
5.2 Q-MuSIK results using Gaussian basis functions with shape parameter $d = 0.4$, test function $F_4^{2d}(x, y)$, on an equally spaced 160×160 evaluation grid. 83
5.3 Q-MuSIK results using Gaussian basis functions with shape parameter $d = 0.4$, test function $F_6^{2d}(x, y)$, on an equally spaced 160×160 evaluation grid. 87
5.4 Q-MuSIK results using Gaussian basis functions with shape parameter $d = 0.4$, test function $F_1^{3d}(x, y, z)$, on an equally spaced $50 \times 50 \times 50$ evaluation grid. 88
5.5 Q-MuSIK results using Gaussian basis functions with shape parameter $d = 0.4$, test function $F_4^{3d}(x, y, z)$, on an equally spaced $50 \times 50 \times 50$ evaluation grid. 89
5.6 Q-SIK results using Gaussian basis functions with shape parameter $d = 0.4$, test function $F_1^{4d}(x, y, z, t)$, on an equally spaced $21 \times 21 \times 21 \times 21$ evaluation grid. 94

7.1 The exact value of test functions on the domain $[0, 1]^2$ 118
7.2 Q-MuSIK quadrature results using Gaussian basis functions with shape parameter $d = 0.4$, test function $F_1^{2d}(x, y)$, on the domain $[0, 1]^2$. 119
7.3 The exact values of test functions on the domain $[0, 1]^3$ 121

List of Figures

1.1 Isotrophic Gaussian basis function (left) and Anisotropic Gaussian basis function (right) 13

1.2 Gaussian basis function with $\varepsilon = 0.5$ (left) and $\varepsilon = 4$ (right) centered at the origin on $\mathbb{R}^d$ 16

2.1 Linear combination of triangle basis functions (solid, blue), one dimensional piecewise linear interpolation $\mathfrak{f}(x)$ (dashed, red) and original function $f(x)$ (solid, green) 29

2.2 The nodal point basis for the first four levels 33

2.3 The piecewise linear hierarchical basis for the first four levels . 34

2.4 Piecewise bilinear basis function $\triangle_{\mathbf{l},\mathbf{i}}$ on $\Upsilon_{\mathbf{l}}$ 37

2.5 The two-dimensional subspaces $\omega_{\mathbf{l}}$ up to $l = 4$ in each dimension 38

2.6 Uniform full grid and sparse grid in two (left) and three (right) dimensions for level $n = 4$ 41

2.7 Sparse Grid $\mathcal{S}_{4,2}$ as a combination of coarser full grids 42

4.1 Test functions in two dimensions 64

4.2 RMS error versus N(Quasi-interpolation) or SG (Q-SIK) nodes using Gaussian basis functions with $\rho = 0.4$: Quasi-interpolation (red) and Q-SIK (blue) on a 160×160 uniform grid. 67

4.3 RMS error versus CPU time using Gaussian basis functions with $\rho = 0.4$: Quasi-interpolation (red) and Q-SIK (blue) on a 160×160 uniform grid. 70

4.4 CPU time versus N(Quasi-interpolation) or SG (Q-SIK) nodes using Gaussian basis functions with $\rho = 0.4$ and $c = 0.45$: Quasi-interpolation, Q-SIK (red) and RBF interpolation, SIK (blue) on a 160×160 uniform grid. 71

5.1 Sparse grid decomposition, $\mathcal{S}_{1,2} \subset \mathcal{S}_{2,2} \subset \mathcal{S}_{3,2} \subset \mathcal{S}_{4,2} \subset \mathcal{S}_{5,2} \subset \mathcal{S}_{6,2} \subset \mathcal{S}_{7,2} \subset \mathcal{S}_{8,2}$. 79

5.2 RMS error versus N(Quasi-interpolation, Multilevel Quasi-interpolation) or SG (Q-SIK, Q-MuSIK) nodes using Gaussian basis functions with $\rho = 0.4$: Quasi-interpolation (black), Q-SIK (green), Multilevel Quasi-interpolation (blue) and Q-MuSIK (red) on a 160×160 uniform grid. 84

5.3 RMS error versus computational time using Gaussian basis functions with $\rho = 0.4$: Quasi-interpolation (black), Q-SIK (green), Multilevel Quasi-interpolation (blue) and Q-MuSIK (red) on a 160×160 uniform grid. 85

5.4 RMS error versus computational time using Gaussian basis functions with $\rho = 0.4$ and $c = 0.45$: MuSIK (blue) and Q-MuSIK (red) on a 160×160 uniform grid. 86

5.5 RMS error versus N(Quasi-interpolation, Multilevel Quasi-interpolation) or SG (Q-SIK, Q-MuSIK) using Gaussian basis functions with $\rho = 0.4$: Quasi-interpolation (black), Q-SIK (green), Multilevel Quasi-interpolation (blue) and Q-MuSIK (red) on a $50 \times 50 \times 50$ uniform grid. 90

5.6 RMS error versus computational time using Gaussian basis functions with $\rho = 0.4$: Quasi-interpolation (black), Q-SIK (green), Multilevel Quasi-interpolation (blue) and Q-MuSIK (red) on a $50 \times 50 \times 50$ uniform grid. 91

5.7 RMS error versus computational time using Gaussian basis functions with $\rho = 0.4$ and $c = 0.45$: MuSIK (blue) and Q-MuSIK (red) on a $50 \times 50 \times 50$ uniform grid. 92

5.8 RMS error versus N(Quasi-interpolation, Multilevel Quasi-interpolation) or SG (Q-SIK, Q-MuSIK) using Gaussian basis functions with $\rho = 0.4$: Quasi-interpolation (black), Q-SIK (green), Multilevel Quasi-interpolation (blue) and Q-MuSIK (red) on a $21 \times 21 \times 21 \times 21$ uniform grid. 95

7.1 Absolute error versus N(Quasi-quadrature, Multilevel Quasi-quadrature) or SG (Q-SIK quadrature, Q-MuSIK quadrature) nodes using Gaussian basis functions with $\rho = 0.4$: Quasi-quadrature (black), Q-SIK quadrature (green), Multilevel Quasi-quadrature (blue) and Q-MuSIK quadrature (red) on the domain $[0, 1]^2$. 120

7.2 Absolute error versus N(Quasi-quadrature, Multilevel Quasi-quadrature) or SG (Q-SIK quadrature, Q-MuSIK quadrature) nodes using Gaussian basis functions with $\rho = 0.4$: Quasi-quadrature (black), Q-SIK quadrature (green), Multilevel Quasi-quadrature (blue) and Q-MuSIK quadrature (red) on the domain $[0,1]^3$. 123

7.3 Absolute error versus SG and CPU Time(Q-MuSIK quadrature, Q-MuSIK quadrature) nodes using Gaussian basis functions with $\rho = 0.4$ and $c = 0.45$: MuSIK quadrature (blue) and Q-MuSIK quadrature (red) on the domain $[0,1]^{4,5,10}$. . . . 124

Chapter 1

Introduction

Over the last half century, numerical methods have gained much attention, not only among mathematicians but also in the scientific and engineering communities. Therefore a number of works concerned with multivariate scattered data interpolation and approximation in $\mathbb{R}^d \times \mathbb{R}$ are considered to be interdisciplinary studies, namely between mathematics and several application areas such as solutions of partial differential equations [35], [34], [64], [65], non-uniform sampling in medicine [21], [94], mapping problems in geophysics, geodesy and meteorology [55], [56], [57], [58], Lyapunov function for a dynamical system [47], [48], fitting of potential energy surfaces in chemistry, coupling of engineering models with sets of inconsonant parameters, derivative pricing in financial mathematics [81], [60], [74], learning theory, neural networks, data mining [6], [49], computer graphics [3], [27], numerical integration [28], [66] and optimization.

For the above scientific areas, researchers are generally confronted with a significant computational problem since there are a number of parameters which need to be handled. However, working in high dimensions needs considerable computer memory and evaluation time which, even given current computational resources, are still mostly impossible. For this reason, researchers have embarked on the quest to find new methods which are able to cope with the interpolation problem in high dimensions.

Radial Basis Function (RBF) interpolation is one of the tools which is effective in approximating and interpolating high-dimensional functions when the data is scattered in its domain. The historical background of the radial basis function approximation method goes back to 1968 with Hardy, who was the first person to use multiquadric RBFs in academia [58]. Thereafter, the use of RBFs has become increasingly popular as an approximation method since it obtains delicate and accurate consequences without using a mesh. This is true not only in approximation or interpolation of data sets [82] but also in solving partial differential equations; see [65], for example of applications of the RBF method.

Of course, there are some problems with the RBF method, such as computational cost and stability, and a number of feasible approaches have been suggested to deal with these problems [43]. In detail, due to the fact that data density enlarges, the interpolation system is, most of time, ill-conditioned [36]. There are some precautions one can take to avoid this instability, such as using compactly supported and sharp-pointed basis functions. However, this is only efficient when the degree of freedom is a moderate magnitude. In

addition to this, a scaling parameter, named the shape parameter, plays a significant role for some RBF methods. For instance, a small shape parameter obtains good accuracy when one interpolates a given data set. However, in these circumstances, we obtain enormous interpolation coefficients since the interpolation matrix is ill-conditioned. This can lead to cancellation of terms when the interpolation coefficients are associated to obtain the interpolant, which might result in a consequential loss of computational precision; see [10].

Another problem is that solving the RBF system, in practice, collapses because of computational difficulties. For example, solving a non-customized RBF system with Gaussian elimination demands $\mathcal{O}(N^3)$ flops and $\mathcal{O}(N^2)$ storage [89]. Furthermore, a direct RBF interpolation needs to evaluate $\mathcal{O}(N)$ operations. It can be seen that there is extensive consideration given to computational difficulties in the RBF literature. Although, in reality, we need to use hundreds of thousands of data sites to model a given problem, classical RBF methods allow us to use only a few thousands data sites. All in all, it can be said that instabilities and the complexity issue are two major problems for the RBF method.

In order to overcome and design a better method, a lot of techniques have been suggested, such as compactly supported radial basis functions (CSRBFs) [84], domain decomposition [32] , multilevel interpolation techniques [61], domain decomposition associated with approximate cardinal functions [13], preconditioning strategies and least square approximate cardinal functions, etc. Among these methods, [32] and [11] can be said to be faster and more

effective techniques for RBFs interpolation.

Another powerful tool for multidimensional problems is quasi-interpolation, which is comprehensively used in scientific computations, mechanics and engineering, and from which a number of successful results have been gained [9], [93], [92]. A quasi-interpolation technique based on radial basis functions is discussed in the sequel. One of the advantages of quasi-interpolation is relevant to features of the generating functions themselves, such as smoothness, simplicity, good shape properties and their exponential decay behaviour at infinity. In addition to these, quasi-interpolation can yield a solution directly since there is no need to solve any large-scale system of equations. Therefore this method can approximate the function in a reduced computational time, even in high dimension, in comparison with other meshless techniques such as RBF interpolation. Quasi-interpolation has been successfully applied to scattered data approximation and interpolation, numerical solutions of partial differential equations and quadrature.

In the literature of the quasi-interpolation, some methods having convergence rates have been also discussed [71, 72]. Although quasi-interpolation operators have principally used functions defined all over the real line, some applications require functions defined on a specific compact interval to allow for efficient approximation procedures, such as boundary integral equations and treatment of partial differential equations. Müller and Varnhorn [77] applied quasi-interpolation operators to such functions. In contrast to all-space functions, a truncation error has to be controlled for these kind of functions; in addition to these pointwise error estimates and L_1 error estimates have

also to be given explicitly [77].

Another study of quasi-interpolation has been made by Chen and Cao [24]. In this study, the convergence analysis in the supremum norm has been presented by modifying the quasi-interpolation operator. Then, in [25], further investigation of quasi-interpolation has been made on the compact interval.

1.1 Motivation and Objectives

High dimensions usually cause some problems for mathematical modelling on gridded data. The main problem is known as the curse of dimensionality, a term due to Bellmann [12]. There is a 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 a gridded data over a bounded domain $\Omega \in \mathbb{R}^d$ is $\mathcal{O}(N^d)$, where N is the dimension of the input data.

In order to cope with this problem, there are a number of remedies proposed in academia. The first of these is hyperbolic cross spaces, which were mentioned by Babenko [4] and Smolyak [87] in the structure of numerical integration in early 1960's. In these papers, the hyperbolic cross product was proposed in order to construct a quadrature rule for multidimensional functions via additional smoothness assumptions. Then, sparse grid methods

were introduced by Zenger [95] 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 hyperbolic cross product hypothesis. The *sparse grid method* arises from a sparse tensor product construction (hyperbolic cross product) and hierarchical basis. Additionally, sparse grid techniques have also been used as solutions of PDEs for finite difference and finite volume methods.

In 2012, Georgoulis, Levesley and Subhan [46] introduced a new kernel-based interpolation technique which circumvents both computational complication and conditioning problems. They obtained more reliable and faster results in higher dimension RBF interpolation problems by means of this technique, which can be seen as an improved version of hyperbolic cross functions. The basic principle of this method is the use of anisotropic radial basis function interpolation. All in all, sparse interpolation with kernels, called SIK, allows for an enormous reduction in the amount of computing resources required to solve interpolation problems at a given level of accuracy.

In order to take the advantages of the SIK method a step further, we have introduced the quasi-sparse interpolation with kernels, called Q-SIK. The main motivation for the proposed method stems from the idea of the SIK method. The principal point of the proposed algorithm is to use anisotropic Gaussian interpolation for all directions associated with the sparse grid combination technique. The combination technique [53] is a sparse grid representation, where partial solutions are computed on a certain sequence of coarser grids; one then gets the solution by linearly combining all partial solutions. This

is the main idea behind the proposed algorithm.

In order to obtain both accelerated convergence and more accurate consequences for the interpolation, multilevel techniques have been suggested by a number of researchers. The first to consider the multilevel approximation were Floater and Iske [40]. 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 [54].

The proposed scheme, Q-SIK, can be extended to a multilevel variant, quasi-multilevel sparse interpolation with kernels (Q-MuSIK). The Q-MuSIK technique is derived from the Q-SIK method. The main considerations behind the approach to multilevel techniques can be divided into two steps: the first is to interpolate the data sets at the coarsest level, and the second is to update the interpolation of the residuals on gradually finer data sets by means of properly scaled basis functions.

Principally, construction of a sparse grid in Q-SIK techniques is similar to multilevel methods. When we interpolate given data sets using the Q-MuSIK method, we need to use sparse grids which are nested, and from lower to higher levels, i.e., $\mathcal{S}_{n,d} \subset \mathcal{S}_{n+1,d}$ where d is dimension and $n \in \mathbb{N}$. In addition to this, properly scaled anisotropic quasi-interpolation should be used for each level. All in all, it can be said that the Q-MuSIK method does not adversely affect the complexity features of SIK because of the geometric

progression in the problem dimension in the nested-ness of sparse grids.

1.2 Quasi-interpolation

1.2.1 Quasi-interpolation using Gaussian Kernels

Quasi-interpolation is one of the methods which generates an approximation defined over the whole $\mathbb{R}^n$. In order to construct quasi-interpolation, we need a function μ, called a basis function, which is, in general, a finite linear combination of translates of some basic function φ. For instance, $\mu(x) = \varphi(|x|^2)$ with $\varphi(t) = t \log t$ for thin plate splines, or μ may be a B-spline and φ a truncated power function. In general, these basis functions are assumed to be Fourier transformable, even and continuous. The following table lists some basis functions on $\mathbb{R}$ and their Fourier transform in one dimension.

Basis Functions $\mu(x)$	Fourier Transform $\mathcal{F}(\xi)$
$\pi^{-\frac{1}{2}} e^{-x^2}$	$e^{-\pi^2\xi^2}$
$\pi^{-1} \cosh^{-1} x$	$\cosh^{-1} \pi^2\xi^{-1}$
$2x\pi^{-2} \sinh^{-1} x$	$\cosh^{-2} \pi^2\xi^{-1}$
$2\pi^{-1}(1+x^2)^{-2}$	$(1+2\pi\|\xi\|)e^{-2\pi\|\xi\|}$
$\pi^{-\frac{1}{2}} e^{-x^2+\frac{1}{2}} \cos\sqrt{2}x$	$e^{-\pi^2\xi^2} \cosh\sqrt{2}\pi\xi$

Table 1.1: One-Dimensional Basis Functions and their Fourier Transform.

For a multivariate function (n-dimensional case), one can take the tensor

product of the above one-dimensional functions. Table 1.2 shows some basis functions and their Fourier transform in the multidimensional case.

Basis Functions $\mu(x)$	Fourier Transform $\mathcal{F}(\xi)$
$\pi^{-\frac{n}{2}} e^{-\lvert\mathbf{x}\rvert^2}$	$e^{-\pi^2\lvert\boldsymbol{\xi}\rvert^2}$
$\operatorname{sech}(\lvert\mathbf{x}\rvert)$	$\dfrac{\pi^2 \tanh(\pi^2\lvert\boldsymbol{\xi}\rvert)}{\lvert\boldsymbol{\xi}\rvert \cosh(\pi^2\lvert\boldsymbol{\xi}\rvert)}$
$\dfrac{4\Gamma\left(\frac{n+5}{2}\right)}{3\pi^{n+1/2}(1+\lvert\mathbf{x}\rvert^2)^{n+5/2}}$	$(1+2\pi\lvert\boldsymbol{\xi}\rvert+\frac{4}{3}\pi^2\lvert\boldsymbol{\xi}\rvert^2)e^{-2\pi\lvert\boldsymbol{\xi}\rvert}$

Table 1.2: Multidimensional Basis Functions and their Fourier Transform.

The principal advantage of quasi interpolation is that there is no need to solve for a large algebraic system. In other words, quasi-interpolation can yield an approximant directly and does not require computation of any linear system. This property makes quasi-interpolation considerably faster compared to other methods. In addition to this quasi interpolation is desirable interpolation technique because of the properties of the basis functions themselves, such as smoothness, simplicity and exponential decay behaviour at infinity.

Bernstein's approximation might be seen as the oldest type of quasi-interpolation. In this approximation, Bernstein polynomials

$$b_\nu^n(x) = \binom{n}{\nu} x^\nu (1-x)^{n-\nu}, \quad \nu = 0, 1, 2, \ldots, n. \tag{1.1}$$

are used to construct a quasi-interpolation of an univariate function f on

$[0,1]$, which is

$$\sum_{\nu=0}^{n} f\left(\frac{\nu}{n}\right) b_{\nu}^{n}(x), \quad x \in [0,1]. \tag{1.2}$$

Although Bernstein's approximation is a fundamental form of approximation theory, it has a wide range of application areas such as Computer Aided Geometric Design.

The standard form of quasi-interpolation takes the values of an n-dimensional function $f(\mathbf{k}h)$, $\mathbf{k} \in \mathbb{Z}^n$ on a uniform grid with mesh size h and a collection of basis functions $\mu(\cdot)$ to build an approximant of f using linear combination. In this thesis, we will use the Schoenberg's model [83]

$$f(\mathbf{x}) \sim \sum_{\mathbf{k}\in\mathbb{Z}^n} f(\mathbf{k}h)\mu\left(\frac{\mathbf{x}}{h} - \mathbf{k}\right), \quad \mathbf{x} \in \mathbb{R}^n, \quad h > 0. \tag{1.3}$$

where μ is a basis function type such as univariate splines, multivariate splines and radial basis functions on $\mathbb{R}^n$.

We will use the Gaussian

$$\mu(\mathbf{x}) = \frac{1}{(\pi\rho)^{n/2}} e^{-\frac{\|\mathbf{x}\|^2}{\rho}}, \tag{1.4}$$

as a basis function through this and the following sections.

So, one defines the quasi-interpolants $Q_h f$ with the appropriate function $f \in C(\mathbb{R}^n)$ by using the Gaussian kernel such that

$$Q_h f : \mathbb{R}^n \to \mathbb{R}, \qquad Q_{\rho,h} f(\mathbf{x}) := \frac{1}{(\pi\rho)^{n/2}} \sum_{\mathbf{k}\in\mathbb{Z}^n} f(\mathbf{k}h) e^{-\frac{\|\mathbf{x}-\mathbf{k}h\|^2}{\rho h^2}}. \tag{1.5}$$

These types of operator are known as *quasi-interpolants* and we take a close

interest in their behaviour as h goes to zero. If one applies Poisson summation formula (see [73]) to the positive, smooth and bounded function

$$\Phi : \mathbb{R} \to \mathbb{R}, \qquad \Phi(x,\rho) := \sum_{k\in\mathbb{Z}} \varphi(x,\rho) = \frac{1}{\sqrt{\pi\rho}} \sum_{k=-\infty}^{\infty} e^{-\frac{\|x-k\|^2}{\rho}}, \tag{1.6}$$

these equivalent representations are obtained

$$\Phi(x,\rho) - 1 = 2\sum_{j=1}^{\infty} e^{-\rho\pi^2 j^2} \cos 2\pi j x. \tag{1.7}$$

Thus we deduce that

$$|\Phi(x,\rho) - 1| \leq \frac{2e^{-\rho\pi^2}}{1 - e^{-\rho\pi^2}}. \tag{1.8}$$

Then the integer shift of the Gaussian is an approximate identity and approaches the Dirac δ as $h \to 0$ in a distributional sense for $\rho > 0$. In other words, we can say that the system of Gaussian kernels $\phi(k) : k \in \mathbb{Z}$ is an approximation of the unit function on $\mathbb{R}$ when ρ is positive. We refer to the book [73], for the detailed explanation of the above calculations.

1.2.2 Anisotropic Quasi-interpolation

Most of the basis functions described above are isotropic, which implies that the behaviour of the function in any direction is the same. However, in general, data sets show variety along different directions, such that variation in one direction is much faster or larger than the other dimensions. Therefore, it can be easily said that distribution of the data sets in the domain shows

anisotropic behaviour.

In this study we will focus on the anisotropic behaviour of Gaussian functions. A significant generalization of the Gaussian can be given by the exponential function as

$$e^{-\langle A\mathbf{x},\mathbf{x}\rangle}, \quad \mathbf{x} \in \mathbb{R}^n, \tag{1.9}$$

where A is a $n \times n$ matrix which is non-singular. The key point here is that the inverse matrix of A must belong to the same set of matrices. That is, it is an invertible matrix. In our study, we will use anisotropic Gaussian functions in two different interpolation methods, radial basis functions and quasi-interpolation.

Definition 1.1. (Anisotropic Gaussian) Let $A \in \mathbb{R}^{n\times n}$ be any properly selected invertible matrix and $\mu(x)$ be a Gaussian basis function. The anisotropic Gaussian function (AG) μ_A can then be described as

$$\mu_A(\mathbf{x}) := \frac{e^{-\langle A^{-1}\mathbf{x},\mathbf{x}\rangle}}{(\pi)^{n/2}|A|^{1/2}}, \tag{1.10}$$

and generates the quasi-interpolant and its Fourier transform

$$\mathcal{F}(\mu_A(\boldsymbol{\xi})) := e^{-\pi^2\langle A\boldsymbol{\xi},\boldsymbol{\xi}\rangle}. \tag{1.11}$$

In order to aid understanding, isotropic and anisotropic Gaussian basis functions are shown in Figure 1.1 for $A = \begin{bmatrix} 2^4 & 0 \\ 0 & 2 \end{bmatrix}$.

Now, we can define the anisotropic quasi-interpolation (AQI) as

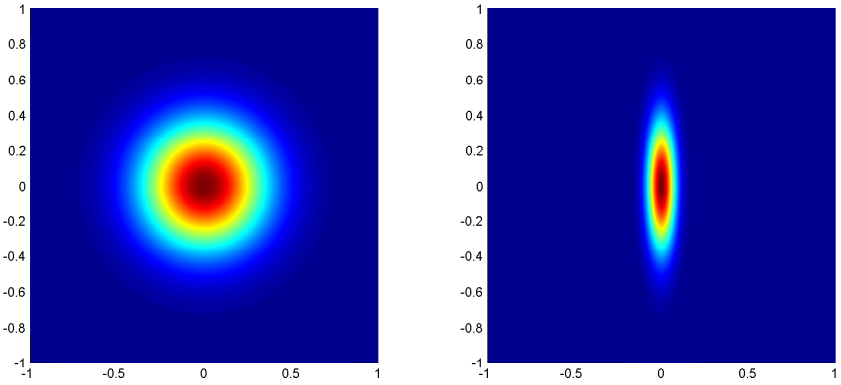


Figure 1.1: Isotrophic Gaussian basis function (left) and Anisotropic Gaussian basis function (right)

$$Q_f^{\mathbf{Ani.}}(\mathbf{x}) = \sum_{\mathbf{k}\in\mathbb{Z}^n} f(\mathbf{k}h)\mu_A\left(\frac{\mathbf{x}}{h} - \mathbf{k}\right), \quad \mathbf{x}\in\mathbb{R}^n, \quad h > 0. \tag{1.12}$$

We shall use the anisotropic quasi interpolation in next chapters when introduce the proposed algorithm.

1.3 Radial Basis Functions Interpolation

1.3.1 Radial Functions

In order to explain radial basis functions clearly, we need to give some related definitions.

Definition 1.2. (Vector $p-$norm) Let $p \geq 1$ be a real number. The $p-$norm of $\boldsymbol{\nu} \in \mathbb{R}^n$ is defined as $\|\boldsymbol{\nu}\|_p = \{\sum_{k=1}^n |\nu_k|^p\}^{\frac{1}{p}}$. In general, some

commonly used $p-$norm are:

1. $\|\boldsymbol{\nu}\|_1 = \sum_{k=1}^{n} |\nu|$ (the grid norm)
2. $\|\boldsymbol{\nu}\|_2 = \{\sum_{k=1}^{n} |\nu|^2\}^{\frac{1}{2}}$ (the Euclidean norm)
3. $\|\boldsymbol{\nu}\|_\infty = \lim_{p\to\infty} \|\boldsymbol{\nu}\|_p = \lim_{p\to\infty} \{\sum_{k=1}^{n} |\nu_k|^p\}^{\frac{1}{p}} = \max_i |\nu_i|$ (the max norm)

Definition 1.3. (Matrix $p-$norm) Let $A : \mathbb{R}^n \to \mathbb{R}^m$ and $\boldsymbol{\nu} \in \mathbb{R}^n$. The matrix norm of A induced by the vector norm is $\|A\|_p = \max_{\boldsymbol{\nu}\in\mathbb{R}^n-\{0\}} \frac{\|A\boldsymbol{\nu}\|_p}{\|\boldsymbol{\nu}\|_p}$, where $\|\cdot\|$ is the p-norm. Some of the commonly used matrix norm are:

1. $\|A\|_1 = \max_{1\le j\le n} \sum_{k=1}^{n} |A_{kj}|$
2. $\|A\|_2 = \sqrt{\sigma(A)(A^T A)}$ where $\sigma(A)$ is the spectral radius of matrix A.
3. $\|A\|_\infty = \max_{1\le k\le n} \sum_{j=1}^{n} |A_{kj}|$

Definition 1.4. A radial function is a function $\boldsymbol{\Psi} : \mathbb{R}^d \to \mathbb{R}$ satisfying $\boldsymbol{\Psi(\cdot - x_k)} = \psi(r)$, where ϕ is a univariate function such that $\psi : [0,\infty) \to \mathbb{R}$ and $r = \|\cdot - x_k\|$. Here $\|\cdot\|$ denotes some norm on $\mathbb{R}^d$ - in general, the standard Euclidean distance.

Radial functions are a particular category of functions. According to the definition, $\boldsymbol{\Psi}$ at any point at a certain distance scale from a fixed point $x_k \in \mathbb{R}^d$, called the center, or the origin, is constant; therefore, Ψ is radially symmetric around its center. Although there are some suggestions about the choice of centers x_k, mainstream thought chooses the centers to coincides with the data sites. In addition to this, one of the most significant properties

of radial functions is that the distance from a center point is increasing (or decreasing) gradually due to its shape.

Moreover, another powerful property of radial functions is that the interpolation problem for multivariate approximation is not affected by the dimension d of the space in which the data sites lie. Hence, we can use the univariate function ψ for any dimension d whenever we use a multivariate function $\mathbf{\Psi}$, since the complexity is directly proportional to the dimensionality.

Classical radial basis functions can be classified into two group: piecewise smooth RBFs and infinitely smooth RBFs, which are given in Table 1.3 and Table 1.4.

Radial Basis Functions	$\psi(r)$	Parameters	m (Order)
Piecewise Polynomial (R_n)	r^n	$n > 0$, $n \notin 2\mathbb{N}$	$m \geq \lceil \frac{n}{2} \rceil$
Thin Plate Spline (TPS_n)	$r^{2n} \ln(r)$	$n \in \mathbb{N}$	$m \geq n + 1$
Polyharmonic Spline (PS_n)	$(-1)^{n+1} r^{2n} \ln(r)$	$n \in \mathbb{N}$	$m \geq n + 1$

Table 1.3: Piecewise Smooth Radial Basis Functions.

In these tables, m represents the order of RBFs, $K_n(r)$ represents a modified Bessel function of order $n > 0$ and $\lceil r \rceil$ and $\lfloor r \rfloor$ stand for closest integers that are greater than, and less than, r, respectively. The rate of convergence using infinitely smooth RBFs is higher than when using the piecewise smooth RBFs, which have an algebraical rate of convergence [14].

In some cases, such as the basis functions in Table 1.4, $\psi(r)$ is replaced with

Radial Basis Functions	$\psi(r,\varepsilon)$	Parameters	m (Order)
Exponentials (EXPs)	$e^{-\varepsilon r}$	$\varepsilon > 0$	$m \geq 0$
Gaussians (GAs)	$e^{-(\varepsilon r)^2}$	$\varepsilon > 0$	$m \geq 0$
Multiquadrics (MQs)	$(1+\varepsilon^2 r^2)^{\frac{n}{2}}$	$n, \varepsilon > 0$, $n \notin 2\mathbb{N}$	$m \geq \lceil \frac{n}{2} \rceil$
Inverse Quadrics (IQs)	$(1+\varepsilon^2 r^2)^n$	$n < 0$, $\varepsilon > 0$	$m \geq 0$
Inverse Multiquadrics ($IMQs$)	$(1+\varepsilon^2 r^2)^{\frac{n}{2}}$	$n < 0$, $\varepsilon > 0$	$m \geq 0$
Matern (MAs)	$\frac{2^{1-n}}{\Gamma(n)} r^n K_n(r)$	$n > 0$	$m \geq 0$

Table 1.4: Infinitely Smooth Radial Basis Functions.

$\psi(r,\varepsilon)$, where ε is called the shape parameter. As its name suggest, this parameter modifies and controls the shape of these functions. For instance, while one can obtain flatter RBFs with large values in the shape parameter, a more peaked RBF can be obtained with a small ε. Figure 1.2 shows the property of shape dependence and radial symmetry of Gaussian basis functions. We assume that the shape parameter is some fixed and non-zero real value throughout this and the next section.

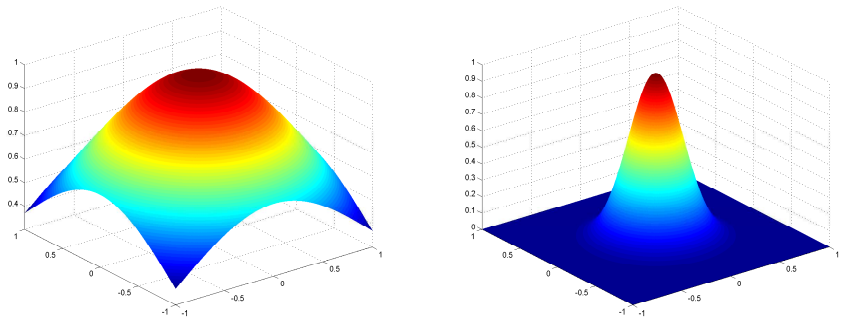


Figure 1.2: Gaussian basis function with $\varepsilon = 0.5$ (left) and $\varepsilon = 4$ (right) centered at the origin on $\mathbb{R}^d$

1.3.2 Radial Basis Functions Interpolation

The primary RBF technique is described as follows:

Definition 1.5. Consider a given data set $\mathbf{f} = (f_1, ..., f_N)^T \in \mathbb{R}^N$ of function values, taken from an unknown function $\mathbf{f} : \mathbb{R}^d \to \mathbb{R}$ at scattered data points $\mathbf{x}_j \in \mathbb{R}^d$, $j = 1, ..., N$ such that $\mathbf{f}_j = \mathbf{f}(x_j)$ and $d \geq 1$. The RBF interpolation is given by

$$I_f(\mathbf{x}) = \sum_{j=1}^{N} \alpha_j \psi(\|\mathbf{x} - \mathbf{x_j}\|), \tag{1.13}$$

where $\psi(\cdot)$ is a radial function and $\|\cdot\|$ is the Euclidean distance. The coefficient α_j can be determined from interpolation requirements $I_f(\mathbf{x}_j) = \mathbf{f}_j$ by solving the following symmetric linear system:

$$\Big[A\Big]_{N\times N} \Big[\alpha\Big]_{N\times 1} = \Big[f\Big]_{N\times 1}, \tag{1.14}$$

where the matrix $A_{(N\times N)}$ is constructed for a_{jk} such that $a_{jk} = \psi(\|x_j - x_k\|)$, $j, k = 1, \ldots, N$.

In this kind of RBF interpolation problem, the order of the basis function is 0, $(m = 0)$. In other words, the basis function ψ is a positive definite function, such as a Gaussian.

Let us focus on the existence of a unique solution to this kind of RBF interpolation problem. A sufficient condition is that the matrix A should be

non-singular in order that this system has a unique solution. Although there has been much research into how to guarantee the non-singularity of the technique for all other basis functions used, nobody has yet succeeded in characterizing a category of basis functions which produce a non-singular system. However, the situation is preferable if we use positive definite functions. In order to define positive definite functions, we need to make some definitions:

Definition 1.6. (Completely monotone functions) A function $\Im : [0, \infty) \to \mathbb{R}$ is called a completely monotone function if the following conditions hold:

1. $\Im \in C[0, \infty) \cap C^{\infty}(0, \infty)$
2. $(-1)^{\jmath}\Im^{(\jmath)}(r) \geq 0$ for $r > 0$ and $\jmath = 0, 1, 2, \ldots$.

Definition 1.7. (Positive definite matrix) An $n \times n$ matrix A is called positive semi-definite if

$$\boldsymbol{\alpha}^T A \boldsymbol{\alpha} = \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j A_{ij} \geq 0, \tag{1.15}$$

for $\boldsymbol{\alpha} = [\alpha_1, \ldots, \alpha_N]^T \in \mathbb{R}$. Then A is called positive definite if the above inequality is zero only for $\boldsymbol{\alpha} \equiv \mathbf{0}$.

Definition 1.8. (Positive definite functions) A function $\psi : \mathbb{R}^d \to \mathbb{R}$ is called positive definite function if

$$\sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j \psi(x_i - x_j) \geq 0, \tag{1.16}$$

for all possible finite system pairwise distinct points $x_1, \ldots, x_N \in \mathbb{R}^d$, and $\boldsymbol{\alpha} = [\alpha_1, \ldots, \alpha_N]^T \in \mathbb{R}^N$.

Theorem 1.9. *If the function $\psi : [0, \infty) \to \mathbb{R}$ is completely monotone but not constant, then for any set of distinct points $(x_1, \ldots, x_N)$ the matrix A, which consists of $a_{jk} = \psi(\|x_j - x_k\|)$, $j, k = 1, \ldots, N$ is positive definite.*

As a result, the matrix A will be positive definite since the basis function ψ is positive definite. For this reason the matrix A is non-singular because of its positive definite property. That is, there is only one solution for the symmetric linear system (1.14). In other words, this kind of RBF interpolation problem is well-posed for the case of $m = 0$.

However, we cannot solve the interpolation problem by using the above method when the order of the basis function is equal to or greater than 1. In this situation the basis function is conditionally positive definite of order $(m \geq 1)$. For instance, the above method cannot be applied to some types of basis functions such as a piecewise polynomial, because $\psi(r) >$, $\psi'(r) > 0$ for $r \geq 0$. In order to obtain sufficient conditions for a non-singular matrix system, Micchelli added some restrictions such as new requirements to (1.14). These restrictions lead to a technique called the augmented RBF technique. However, we need to give here some definitions in order to understand and explain it:

Definition 1.10. $\prod_m^d$ is the space of d-variate polynomials of total degree less than or equal to m. Additionally, M is the dimension of $\prod_m^d$ which is $M = \binom{m+d}{d}$.

Definition 1.11. (Conditionally positive definite functions) A function $\psi : \mathbb{R}^d \to \mathbb{R}$ is called a conditionally positive definite function of order m if

$$\sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_i \alpha_j \psi(x_i - x_j) > 0 \tag{1.17}$$

holds for all possible finite system pairwise distinct points $x_1, \ldots, x_N \in \mathbb{R}^d$, and $\boldsymbol{\alpha} = [\alpha_1, \ldots, \alpha_N]^T \in \mathbb{R}^N$, satisfying the vanishing moment conditions

$$\sum_{j=1}^{N} \alpha_j P(\mathbf{x_j}) = 0, \tag{1.18}$$

for all $P \in \prod_{m-1}^{d}$.

Definition 1.12. Consider a given data set $\mathbf{f} = (f_1, ..., f_N)^T \in \mathbb{R}^N$ of function values, taken from an unknown function $\mathbf{f} : \mathbb{R}^d \to \mathbb{R}$ at scattered data points $\mathbf{x}_j \in \mathbb{R}^d$, $j = 1, ..., N$, such that $\mathbf{f}_j = \mathbf{f}(x_j)$ and $d \geq 1$. The augmented RBF interpolation is given by

$$I_f(\mathbf{x}) = \sum_{j=1}^{N} \alpha_j \psi(\|\mathbf{x} - \mathbf{x_j}\|) + \sum_{k=1}^{M} \upsilon_k d_k(\mathbf{x}) = f(\mathbf{x}), \quad \text{for} \quad \mathbf{x} \in \mathbb{R}^d, \tag{1.19}$$

where $(d_1(\mathbf{x}), \ldots, d_P(\mathbf{x}))$ is a basis for $\prod_m^d$ and $\psi(r)$, $r \geq 0$, is any basis function. With additional terms there are $N + M$ unknown variables which are the coefficients $\boldsymbol{\alpha}$ and $\boldsymbol{\upsilon}$. In order to determine these coefficients, the following restrictions are imposed:

$$\sum_{j=1}^{N} \alpha_j d_k(\mathbf{x_j}) = 0, \quad \text{for} \quad k = 1, \ldots, M, \tag{1.20}$$

which cause the following symmetric linear system

$$\begin{bmatrix} A & P \\ P^T & \mathbf{O} \end{bmatrix}_{(N+M)\times(N+M)} \times \begin{bmatrix} \alpha \\ \upsilon \end{bmatrix}_{(N+M)\times 1} = \begin{bmatrix} f \\ \mathbf{0} \end{bmatrix}_{(N+M)\times 1}, \tag{1.21}$$

where $A_{(N\times N)}$ is the matrix in (1.14), $P_{(N\times M)}$ is the matrix with entries $d_k(\mathbf{x_j})$, $\mathbf{O}_{(N\times N)}$ is a null matrix and $\mathbf{0}_{(M\times 1)}$ is a null vector.

Theorem 1.13. *Let $\psi^{m+1}(r)$ for $m \geq 0$ be completely monotone but constant on $(0,\infty)$. The full $(N+M)\times(N+M)$ matrix in (1.21) is non-singular since for any set of $\boldsymbol{x}_j \in \mathbb{R}^d$, $j = 1, ..., N$ satisfy the condition of $rank(P) = M$, where P is a matrix in (1.18).*

As a result, whenever the above theorem holds, the first matrix in (1.21) guarantees the non-singularity, which means the well-posedness of the symmetric linear system. Hence, there is only one solution for this system.

The most frequently used RBFs in applications are Gaussians with $m = 0$, the multiquadric (MQ) with $m = 1$, the inverse multiquadric (IMQ) with $m = 0$ and thin plate splines (TPS_2), which are a special case of polyharmonic splines with $m = 2$. Although the MQ is CPD is of order $m = 1$, it has a unique solution without the additional constant polynomial whenever $\alpha = 0$ [75].

According to Schoënberg's result [85], the interpolation matrix A is always non-singular whenever the Euclidean norm is chosen as a distance. Although we have only focused on the well-posedness problem for Euclidean distance in

describing the interpolation problem, there has been some research conducted for other distance types. If the data points are located on vertices of a closed polygon, the grid norm (p=1) when $d = 2$ can yield a singular interpolation matrix [67]. In [7], the well-posedness for $p \in (0, 1)$ was proven. In addition to this, [7] showed that it is possible to construct points which produce a singular interpolation matrix whenever p and dimension are greater than 2.

Moreover, there are a few RBF interpolation algorithms with flatter basis functions. One of the methods which uses small shape parameters is the Contour-Pade of Fornberg and Wright [43], whilst another is the RBF-QR of Fornberg and Piret [42], which was improved for the case where nodes are scattered over the surface of a sphere. In addition to these, two different research groups - Fasshauer and McCourt [38], and Fornberg, Larsson and Flyer [41] - have introduced two stable algorithms in order to evaluate Gaussian RBF interpolation with a flat kernel. In [38], stable Gaussian RBF interpolation in $\mathbb{R}^4$ has been presented by the authors. The aim of this method is to overcome stability issues, though their solution remains limited to low dimensional problems because of curse of dimensionality. In [41], two-dimensional domains have been used in the RBF-QR method of [42].

1.3.3 Anisotropic Radial Basis Functions Interpolation

In order to approximate anisotropic data as discussed above, anisotropic radial basis functions have been both introduced and used effectively in practice. In [23] and [22], the numerical efficiency of anisotropic radial basis func-

tions for local fitting has been showed in various applications. On the other hand, in [8], the standard error estimation has been improved for anisotropic radial basis functions and its derivatives. Now we can define the anisotropic radial basis functions:

Definition 1.14. (Anisotropic radial basis function) Let $A \in \mathbb{R}^{d\times d}$ be any properly selected invertible matrix and $\psi(\|x - x_j\|)$ is any given radial basis function which centred at $x_j \in \mathbb{R}^d$. Then the anisotropic radial basis function $(ARBF)$ ψ_A is described as

$$\psi_A(\|x - x_j\|) = \psi(\|A(x - x_j)\|), \tag{1.22}$$

Clearly, if A is the $d \times d$ identity matrix, then, $\psi_A(\|x - x_j\|) = \psi(\|x - x_j\|)$. In the light of this information, one can modify the solution of interpolation problem by using ARBFs.

Consider a given data set $\mathbf{f} = (f_1, ..., f_N)^T \in \mathbb{R}^N$ of function values, taken from an unknown function $\mathbf{f} : \mathbb{R}^d \to \mathbb{R}$ at scattered data points $\mathbf{x}_j \in \mathbb{R}^d$, $j = 1, ..., N$, such that $\mathbf{f}_j = \mathbf{f}(x_j)$ and $d \geq 1$. Let $A \in \mathbb{R}^{d\times d}$ be a properly selected invertible matrix, and ψ be a CPD radial function of order m and, finally, $\{p_i\}_{i=1}^M$ be a basis of the polynomial space $\prod_{m-1}^d$. Now we can describe the anisotropic radial basis functions interpolant $I_f^{\mathbf{Ani.}}$

$$I_f^{\mathbf{Ani.}}(\mathbf{x}) = \sum_{j=1}^{N} \alpha_j \psi_A(\|\mathbf{x} - \mathbf{x_j}\|) + \sum_{k=1}^{M} \upsilon_k d_k(A\mathbf{x}) = f(\mathbf{x}), \quad \mathbf{x} \in \mathbb{R}^d. \tag{1.23}$$

With the following restrictions

$$\sum_{j=1}^{N} \alpha_j d_k(A\mathbf{x_j}) = 0, \quad k = 1, \ldots, M, \tag{1.24}$$

and describing the set of transformed data sites $\{\boldsymbol{\beta_i}\}_{i=1}^{N}$, where $\boldsymbol{\beta}_i = A\mathbf{x_i}$, we can obtain the following transformed symmetric linear system

$$\sum_{j=1}^{N} \alpha_j \psi_A(\|\boldsymbol{\beta_i} - \boldsymbol{\beta_j}\|) + \sum_{k=1}^{M} \upsilon_k d_k(\boldsymbol{\beta_i}) = \mathbf{f_i}, \quad i = 1, \ldots, N, \tag{1.25}$$

and

$$\sum_{j=1}^{N} \alpha_j d_k(\boldsymbol{\beta_j}) = 0, \quad k = 1, \ldots, M. \tag{1.26}$$

The final transformed symmetric linear system is well-posed because of the non-singular property of the transformation matrix A and the result of Theorem 1.13. So there exists only one solution for this transformed symmetric linear system provided that the transformation matrix A is chosen to be non-singular.

1.4 Main Achievements

In this thesis, Q-SIK and Q-MuSIK algorithms will be presented and applied to some test functions to confirm their performance. In order to show their superior properties, we have applied the proposed algorithms in numerical integration schemes. Then, finally, we have presented a theoretical analysis of the convergence properties of multilevel quasi-interpolation.

In order to show the behaviour of the proposed algorithm, we have performed some numerical experiments using six different test functions. We have observed from numerical results that the Q-MuSIK scheme is superior in terms of convergence, computational time, and complexity as compared to classical multilevel quasi-interpolation. In addition, we have compared Q-MuSIK with MuSIK. According to this comparison, the run time is less in the Q-MuSIK method because there is no need to solve any large algebraic systems. Therefore we might propose that the Q-MuSIK algorithm could be used for problems which need to be completed in a short time, in particular for high-dimensional problems.

We have presented convergence analysis of multilevel quasi-interpolation for periodic functions using a Gaussian kernel on a grid. Thus our numerical experiments have been confirmed theoretically. In addition to this, we have shown that the Q-MuSIK algorithm can be applied successfully to numerical integration.

1.5 Outline

This thesis is composed of 8 chapters and is organised as follows:

In Chapter 2, we discuss linear sparse grid spaces and introduce their indirect version called the combination technique.

In Chapter 3, we discuss the sparse interpolation with kernels method and its multilevel version on the nodal exactness.

In Chapter 4, we introduce the new quasi-sparse interpolation using Gaussian kernels. The Q-SIK method is constructed by evaluating anisotropic Gaussian kernels for each sub-interpolation problem. The solution will then obtained by combining these sub-interpolation problems using the combination technique. At the end of the this chapter, some numerical results will be presented and compared.

In Chapter 5, we present the multilevel version of the new quasi-sparse interpolation with Gaussian kernels. This scheme has provided more accurate results with less evaluation time in comparison with the other methods. Numerical experiments will be presented in this chapter.

In Chapter 6, we will present the convergence analysis of multilevel quasi-interpolation for periodic functions. Firstly, we will give the single level quasi-interpolation error by using the shifting properties of the Gaussian kernel. We will then find the multilevel error estimation using the multilevel algorithm for unit function.

In Chapter 7, we will apply the Q-SIK and Q-MuSIK algorithms to numerical integration. The proposed algorithms are expected to show their superior properties in the field of quadrature, with results to be confirmed by numerical experiments.

In Chapter 8, we will summarise a number of our conclusions regarding the new developments presented in this work. Finally, we will briefly discuss our future research and ideas on possible extension of the current results.

Chapter 2

Hyperbolic Cross Product Spaces and Sparse Grid Techniques

In this chapter we deal with hyperbolic cross product spaces and sparse grid techniques which depend on hierarchical bases of linear splines. Hyperbolic cross product spaces play a significant role in this context. The approximation of functions by polynomials from hyperbolic cross product spaces was introduced by Babenko [4] in 1960. Then, in 1963, Smolyak [87] focused on quadrature and interpolation formulas based on tensor products of low dimension operators because of the *curse of dimensionality*. For instance, evaluation of N functions or grid points in one dimension causes to N^d grid points in d-dimensions. So, the exponential behaviour of dimensionality leads to some strict restrictions for the approximations which can be handled. In 1991, the sparse grid scheme was first presented and applied to the finite difference and finite element methods - to find numerical solutions of par-

tial differential equations - by Zenger [95] and Griebel [50]. This technique made it possible to deal with the curse of dimensionality, at least for sufficiently smooth functions. The sparse grid technique is based upon using grids which contribute to the interpolation more than other grids, which by comparison contribute only a little to the interpolation. This allows us to construct larger multidimensional approximations and interpolations than were previously possible.

2.1 General Idea of Interpolation on a Full Grid

Let $f : \mathcal{C} \equiv [0,1]^d \to \mathbb{R}$ be a function which is given only computationally, that is, we only know the value of f at arbitrary points, and consider its piecewise multi linear interpolation.

One needs to divide $\mathcal{C}$ into regular grids which are equally spaced on grid points in order to interpolate function f. Thus, the grid points $\mathbf{x}_i$ with mesh size $h_n = 2^{-n}$ are obtained for some refinement level n. Then the interpolation function of f are described as follows, with a defined, suitable set of piecewise multi linear basis functions $\psi_j(\mathbf{x})$

$$f(\mathbf{x}) \approx \mathfrak{f}(\mathbf{x}) = \sum_j \gamma_j \psi_j(\mathbf{x}), \tag{2.1}$$

where γ_j are coefficients.

Using a full grid leads to the curse of dimensionality in this kind of interpolation method. One can prevent this problem by choosing suitable basis

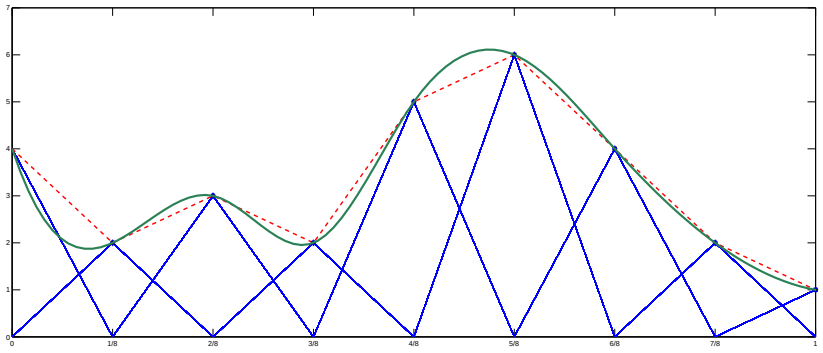


Figure 2.1: Linear combination of triangle basis functions (solid, blue), one dimensional piecewise linear interpolation $\mathfrak{f}(x)$ (dashed, red) and original function $f(x)$ (solid, green)

functions in the interpolation problem. In other words, some basis functions can be omitted since their contribution to the interpolation is less than others. Thus, the full grid can be reduced to a sparse grid, which provides us with the means to deal with multidimensional function interpolation. This kind of basis function can be chosen from a hierarchical basis function, which will be introduced in the following sections.

2.2 Multi-stage Subspace Decomposition

In order to discuss sparse grids in detail, we need some notation and definitions.

Definition 2.1. (Basic multi-index notations) Let $\mathcal{C} := [0,1]^d$ be the d-dimensional unit cube, $\mathbf{x} := (x_1, \ldots, x_d)$ and $f : \mathcal{C} \to \mathbb{R}$. Let $\mathbb{N}_0$ denote the set of non-negative integers. A d-dimensional multi-index is a d-tuple $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_d) \in \mathbb{N}_0^d$.

For $\mathbf{x} = (x_1, \ldots, x_d) \in \mathcal{C}$, the bounded weak mixed derivatives are

$$D^{\alpha} = \left(\frac{\partial}{\partial x_1}\right)^{\alpha_1}, \ldots, \left(\frac{\partial}{\partial x_d}\right)^{\alpha_d} = \frac{\partial^{|\alpha|}}{\partial x_1^{\alpha_1}, \ldots, \partial x_d^{\alpha_d}}. \tag{2.2}$$

Additionally, the discrete L_1-norm and L_∞-norm of $\boldsymbol{\alpha}$ are described respectively as follows:

$$|\boldsymbol{\alpha}|_1 = \sum_{i=1}^{d} \alpha_i \qquad \text{and} \qquad |\boldsymbol{\alpha}|_\infty = \max_{1 \le i \le d} \alpha_i. \tag{2.3}$$

We will use component-wise arithmetic operations which, given below throughout this and next sections, can be defined as:

$$\boldsymbol{\alpha} \cdot \boldsymbol{\beta} := (\alpha_1\beta_1, \ldots, \alpha_d\beta_d), \tag{2.4}$$

$$\xi \cdot \boldsymbol{\alpha} := (\xi\alpha_1, \ldots, \xi\alpha_d), \tag{2.5}$$

$$\delta^{\boldsymbol{\alpha}} := (\delta^{\alpha_1}, \ldots, \delta^{\alpha_d}), \tag{2.6}$$

$$\mathbf{0} := (0, \ldots, 0), \tag{2.7}$$

$$\mathbf{1} := (1, \ldots, 1), \tag{2.8}$$

and the corresponding element-wise relations are

$$\boldsymbol{\alpha} \le \boldsymbol{\beta} \Leftrightarrow \forall_{1 \le i \le d} \quad \alpha_i \le \beta_i, \tag{2.9}$$

$$\boldsymbol{\alpha} < \boldsymbol{\beta} \Leftrightarrow \boldsymbol{\alpha} < \boldsymbol{\beta} \quad \text{and} \quad \alpha \ne \beta. \tag{2.10}$$

For $p = 2$ and $p = \infty$, we describe the space

$$\chi^{p,s}(\mathcal{C}) := \{f : \mathcal{C} \to \mathbb{R} : D^{\boldsymbol{\alpha}} f \in L_p(\mathcal{C}), |\boldsymbol{\alpha}|_\infty \leq s\}. \tag{2.11}$$

Here, $\chi^{p,s}(\mathcal{C})$ represents the space of all functions of bounded mixed derivatives up to order s with respect to L_p-norm. One of the subspaces of $\chi^{p,s}(\mathcal{C})$ is $\chi_0^{p,s}(\mathcal{C})$, which consists of those functions f vanishing on the boundary $\partial\mathcal{C}$. In other words, the homogeneous counterpart of $\chi^{p,s}(\mathcal{C})$ can be defined as follows:

$$\chi_0^{p,s}(\mathcal{C}) := \{f \in \chi^{p,s}(\mathcal{C}) : f|_{\partial\mathcal{C}} = 0\}. \tag{2.12}$$

Within this context, we will limit ourselves to this subspace. In addition, the smoothness parameter s is 2 for the case of the piecewise linear approximation. Finally, the semi-norm of function $f \in \chi_0^{p,s}(\mathcal{C})$ can be described as follows:

$$|f|_2 := \|D^2 f\|_2 = \left(\int_{\mathcal{C}} |D^2 f|^2 dx\right)^{1/2} \quad \text{and} \quad |f|_\infty := \|D^2 f\|_\infty. \tag{2.13}$$

2.2.1 One-dimensional Hierarchical Basis Functions

One-dimensional multilevel basis functions play an important role in the sparse grid method since it will help the solution of the high dimensional problems. In the classical approach, hierarchical basis functions based upon one dimensional linear functions are considered. One can choose a triangular function (also known as a standard hat function) as a one-dimensional linear function. Let us define the triangular function:

Definition 2.2. (Triangular function) The function $\triangle : \mathbb{R} \to \mathbb{R}$ is called a triangular function, and is defined as

$$\triangle(t) = \max(1 - |t|, 0). \tag{2.14}$$

Now, let us consider a set of anisotropic 1-dimensional grids Υ_l of level l on the unit interval $\mathcal{C} = [0, 1]$ and mesh size 2^{-l}. In this notation, the index l represents the level of a grid or a space, whereas the multi-index i represents the location of a basis function or grid point $(x_{l,i})$ in Υ_l. So the set of equally spaced grids $x_{l,i}$ are given by

$$x_{l,i} := i \cdot h_l = i \cdot 2^{-l}, \qquad i \in [0, 2^l]. \tag{2.15}$$

Now, one can generate a scaled piecewise linear basis function $\triangle_{l,i}(x)$ with support $[(i-1)h_l, (i+1)h_l]$ by translation and dilation by using a triangular function, i.e.,

$$\triangle_{l,i}(x) := \triangle(2^l x - i). \tag{2.16}$$

This basis, in general, is called the nodal point basis; as an example, the nodal point basis for $l = 3$ is shown in Figure 2.2.

One can use these basis functions in order to define function spaces ϖ_l, which are constructed from piecewise linear functions. Here, we suppose that the function value in ϖ_l is vanishing on the boundary of $\mathcal{C}$. In order to deal with this restriction, one can add suitable boundary basis functions. So the

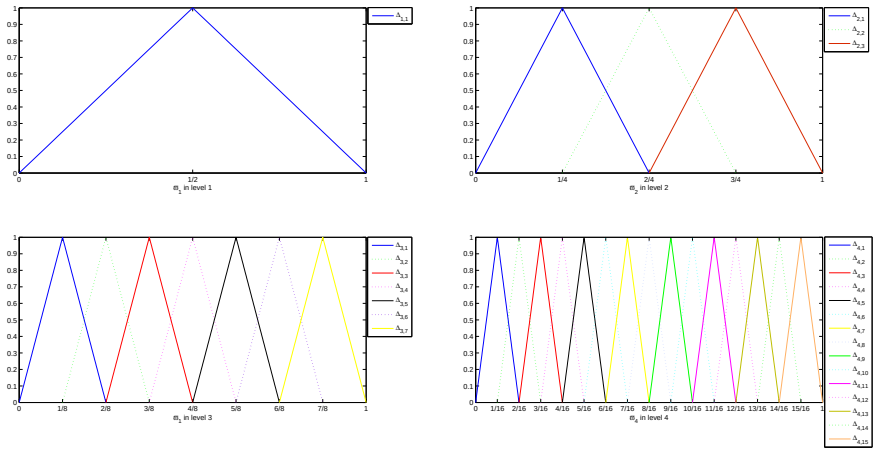


Figure 2.2: The nodal point basis for the first four levels

function space ϖ_l is

$$\varpi_l := \text{span}\{\triangle_{l,i} : i \in \mathbb{N}_0, \quad i \in [1, 2^l - 1]\}. \tag{2.17}$$

Then, we need to define the index set to construct the hierarchical increment space ω_l, that is

$$\omega_l := \text{span}\{\triangle_{l,i} : i \in \aleph_l\}, \tag{2.18}$$

with

$$\aleph_l = \{i \in \mathbb{N}_0 : i \in [1, 2^l - 1], \quad i\text{'s} \quad \text{are odd}\}. \tag{2.19}$$

Additionally, there is a relation between the function space ϖ_l and the hierarchical increment space ω_l, such that

$$\varpi_n = \bigoplus_{l \leq n} \omega_l, \tag{2.20}$$

And the basis related ω_l is called a hierarchical basis. For example, the

piecewise linear hierarchical basis functions for $l = 3$ are shown in Figure 2.3.

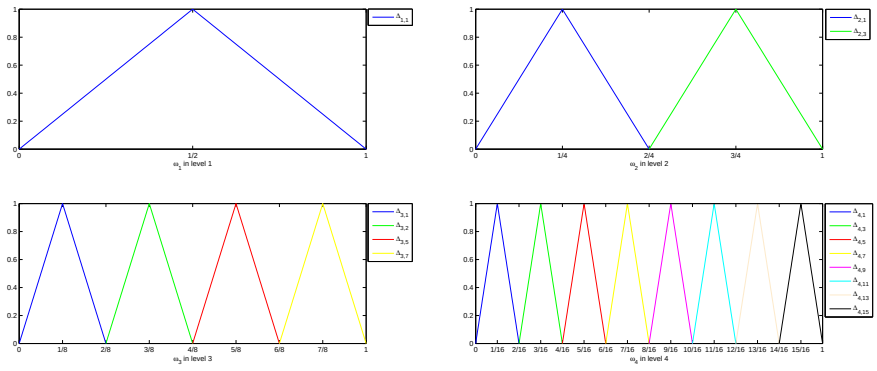


Figure 2.3: The piecewise linear hierarchical basis for the first four levels

In addition to this, one can represent any function $\mathfrak{f} \in \varpi_n$ as follows

$$\mathfrak{f}(x) = \sum_{l \leq n} \sum_{i \in \aleph_l} \lambda_{l,i} \cdot \triangle_{l,i}(x), \tag{2.21}$$

where $\lambda_{l,i} \in \mathbb{R}$ are coefficient values.

2.2.2 Multidimensional Construction with Tensor Product

Up to this point, we have dealt with 1-D hierarchical basis functions on the interval $\mathcal{C} = [0, 1]$. In the light of this information, one can obtain a high-dimensional basis function on the interval $\mathcal{C} = [0, 1]^d$ from 1-D hierarchical basis functions by using a tensor product approach.

Now, let us consider a set of anisotropic grids $\Upsilon_{\mathbf{l}}$ of level $\mathbf{l} \in \mathbb{N}_0^d$ on the unit

interval $\mathcal{C} = [0,1]^d$ and mesh size $2^{-\mathbf{l}}$. Here, the multi-index $\mathbf{l}$ represents the level of grid where $\mathbf{l} = (l_1, \ldots, l_d) \in \mathbb{N}_0^d$ and mesh size $h_\mathbf{l} = (2^{-l_1}, \ldots, 2^{-l_d})$. In other words, although the points of $\Upsilon_\mathbf{l}$ are placed equidistantly in each coordinate direction, the mesh sizes are different along different coordinate directions. So the s-dimensional anisotropic grids $\Upsilon_\mathbf{l}$ are formed from the following points:

$$x_{\mathbf{l},\mathbf{i}} := \mathbf{i} \cdot h_\mathbf{l} = \mathbf{i} \cdot 2^\mathbf{l}, \qquad \mathbf{i} \in [\mathbf{1}, 2^\mathbf{l} - \mathbf{1}]. \tag{2.22}$$

Now, one can define the piecewise d-linear basis function $\triangle_{\mathbf{l},\mathbf{i}}$ for each grid point via a tensor product of one-dimensional basis functions $\triangle_{l,i}$ in each direction, that is

$$\triangle_{\mathbf{l},\mathbf{i}}(\mathbf{x}) := \prod_{m=1}^{d} \triangle_{l_m,i_m}(x_m). \tag{2.23}$$

In addition to this, the arbitrary dimensional case notations are formulated by using the one-dimensional situations, e.g., the function space $\varpi_\mathbf{l}$ constructed from piecewise multilinear functions which are vanishing on the boundary of $\mathcal{C}$ is

$$\varpi_\mathbf{l} := \operatorname{span}\{\triangle_{\mathbf{l},\mathbf{i}} : i \in \mathbb{N}_0^d \quad \wedge \quad i \in [\mathbf{1}, 2^\mathbf{l} - \mathbf{1}]\}. \tag{2.24}$$

Similarly, the subspaces $\omega_\mathbf{l}$ are described as follows

$$\omega_\mathbf{l} := \operatorname{span}\{\triangle_{\mathbf{l},\mathbf{i}} : \mathbf{i} \in \aleph_\mathbf{l}\}, \tag{2.25}$$

with the index set $\aleph_{\mathbf{l}}$

$$\aleph_{\mathbf{l}} = \{\mathbf{i} \in \mathbb{N}_0^d : \mathbf{i} \in [\mathbf{1}, 2^{\mathbf{l}} - \mathbf{1}], \ \wedge \ i_j \text{ odd for all } j \in [1, d]\}. \tag{2.26}$$

Now, $\varpi_{\mathbf{l}}$ is the space of piecewise multi linear basis functions related to the equally located grids of level n, which we represent by $\varpi_{n,d}$. Thus

$$\varpi_{n,d} = \bigoplus_{|\mathbf{l}|_\infty \leq n} \omega_{\mathbf{l}} = \bigoplus_{l_1=0}^{n}, \ldots, \bigoplus_{l_d=0}^{n} \omega_{\mathbf{l}}, \tag{2.27}$$

where $\mathbf{l} = (l_1, \ldots, l_d)$. Thus, the hierarchical basis function space is

$$\{\triangle_{\mathbf{l},\mathbf{i}} : \mathbf{i} \in \aleph_{\mathbf{l}}, |\mathbf{l}|_\infty \leq n\}, \tag{2.28}$$

which leads to the full grid with $(2^n - 1)^d$ grid points. Additionally, the interpolation function $\mathfrak{f}(\mathbf{x}) \in \varpi_{n,d}$ is as follows

$$\mathfrak{f}(\mathbf{x}) = \sum_{|\mathbf{l}|_\infty \leq n} \sum_{\mathbf{i} \in \aleph_{\mathbf{l}}} \lambda_{\mathbf{l},\mathbf{i}} \cdot \triangle_{\mathbf{l},\mathbf{i}}(\mathbf{x}), \tag{2.29}$$

where $\lambda_{\mathbf{l},\mathbf{i}} \in \mathbb{R}^d$ are coefficient values. For instance, in two dimensions, the basis functions of subspace $\omega_{\mathbf{l}}$ are shown in Figure 2.4, which correspond to an anisotropic subgrid.

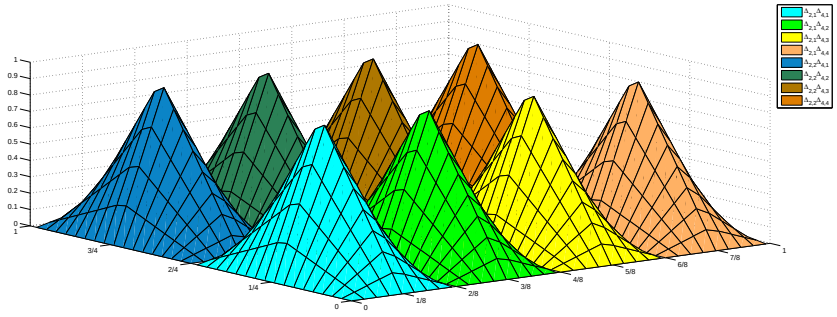


Figure 2.4: Piecewise bilinear basis function $\triangle_{\mathbf{l},\mathbf{i}}$ on $\Upsilon_{\mathbf{l}}$

2.3 Sparse Grids Technique

Now we can choose the subspaces which best provide us with a greater contribution to the solution of the full grid approximation than others by using the hierarchical representation described above. In order to achieve this, we restrict ourselves to the functions f which belong to the Sobolev space $\chi^{2,2}(\mathcal{C})$ defined in (2.11) and have bounded mixed second derivatives $D^{\alpha}f$ for $|\alpha|_{\infty} \leq 2$. Thus, the functions f satisfy the smoothness requirement which is required for sparse grid method. Then, according to a significant result taken from [15], for functions $f \in \chi^{2,2}(\mathcal{C})$, the hierarchical coefficients $\lambda_{\mathbf{l},\mathbf{i}}$ are diminishing as

$$|\lambda_{\mathbf{l},\mathbf{i}}| = \mathcal{O}(2^{-2|\mathbf{l}|_1}). \tag{2.30}$$

Additionally, the number of degrees of freedom (i.e., the size of the subspace of $\omega_{\mathbf{l}}$) is given by

$$|\omega_{\mathbf{l},\mathbf{i}}| = \mathcal{O}(2^{|\mathbf{l}|_1}). \tag{2.31}$$

Therefore, the relationship between the resulting approximation accuracy and the number of degrees of freedom (i.e., the number of grids) directly causes the sparse grid space denoted by $\hat{\varpi}_{n,d}$:

$$\hat{\varpi}_{n,d} = \bigoplus_{|\mathbf{l}|_1 \leq n+d-1} \omega_{\mathbf{l}}, \tag{2.32}$$

omitting those subspaces from the uniform full grid space $\varpi_{n,d}$ with a number of basis functions of little contribution. The selection of subspaces and sparse grids space $\hat{\varpi}_{n,d}$ is shown in Figure 2.5. According to Figure 2.5, black points represent the optimal selection of subspaces, that is, sparse grids. The combination of all grid points (the black and red points) represents the full grids.

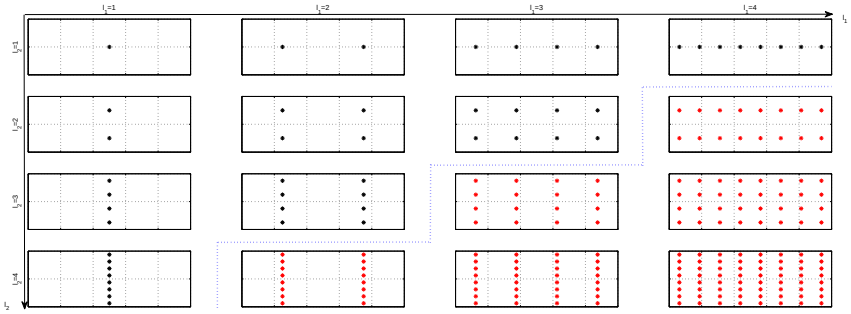


Figure 2.5: The two-dimensional subspaces $\omega_{\mathbf{l}}$ up to $l = 4$ in each dimension

Additionally, the basis of the sparse grid space is given by $\{\triangle_{\mathbf{l},\mathbf{i}} : \mathbf{i} \in \aleph_{\mathbf{l}}, |\mathbf{l}|_1 \leq n\}$ and these subspaces satisfy $\hat{\varpi}_{n,d} \subset \varpi_{n,d}$. The choosing criteria for subspaces are directly related to the norm. For instance, the above result is preferable for both the L_2 and maximum norms. Throughout this and the next sections, the sparse grid of level n in d dimensions which is constructed

from the approximation space $\hat{\varpi}_{n,d}$ is denoted by $\mathcal{S}_{n,d}$. Additionally, as might be expected, the sparse grid in one dimension coincides with the one dimensional uniform full grid.

2.3.1 Approximation Properties of Sparse Grid Spaces

One of the most significant properties of this method is that the sparse grid considerably reduces the computational cost to a reasonable level for the multidimensional approximation problems. In [15], Bungartz and Griebel have presented some important results for approximation order and size of sparse grid spaces. According to this study, the dimension of $\bar{\varpi}_{n,d}$, in other words the number of grid points or degrees of freedom, is given by

$$\begin{aligned} |\bar{\varpi}_{n,d}| &= \sum_{j=0}^{n-1} 2^j \cdot C_{d-1}^{d-1+j} \\ &= (-1)^d + 2^n \cdot \sum_{j=0}^{d-1} C_j^{n+d-1} \cdot (-2)^{d-1-j} \\ &= 2^n \cdot \left(\frac{n^{d-1}}{(d-1)!} + \mathcal{O}(n^{d-2}) \right) \\ &= \mathcal{O}(2^n n^{d-1}) \\ &= \mathcal{O}(h_n^{-1} \cdot (\log h_n^{-1})^{d-1}), \end{aligned} \tag{2.33}$$

where C_n^d is binomial coefficients, $\binom{d}{n}$, and $h_n = 2^{-n}$. When we compare the number of degrees of freedom of the full grid space, which is $|\varpi_{n,d}| = \mathcal{O}(2^{nd}) = \mathcal{O}(h_n^{-d})$, it can be easily seen that the computational and storage requirements

of the sparse grid method are cheaper than for classical methods. In addition to this, the approximation accuracy of sparse grid spaces for function $f \in \chi^{2,2}(\mathcal{C})$ is given by

$$\begin{aligned} \|f - \mathfrak{f}\|_p &= \mathcal{O}(2^{-2n} n^{d-1}), \\ &= \mathcal{O}(h_n^2 \cdot (\log h_n^{-1})^{d-1}) \end{aligned} \tag{2.34}$$

in the L_p-norms. The same sensibility for uniform full grid spaces is given by

$$\begin{aligned} \|f - \mathfrak{f}\|_p &= \mathcal{O}(2^{-2n}). \\ &= \mathcal{O}(h_n^2) \end{aligned} \tag{2.35}$$

In both comparisons, the dimension affects the order notation as a logarithmic. In other words, sparse grid spaces $\bar{\varpi}_{n,d}$ provide us considerable positive contribution vis-à-vis the uniform full grid spaces because there is a remarkable amount of difference between the number of grid points of sparse grid spaces and full grid spaces in the multi dimension approximation. Therefore, the computation and storage requirements are decreased with a smaller number of grids. However, there is only a little positive influence from sparse grid spaces for approximation accuracy in comparison with the full grid spaces. Thus the curse of dimensionality can be dealt with in this manner. Figure 2.6 shows uniform full grids and sparse grids in two and three dimensions for level $n = 4$ each.

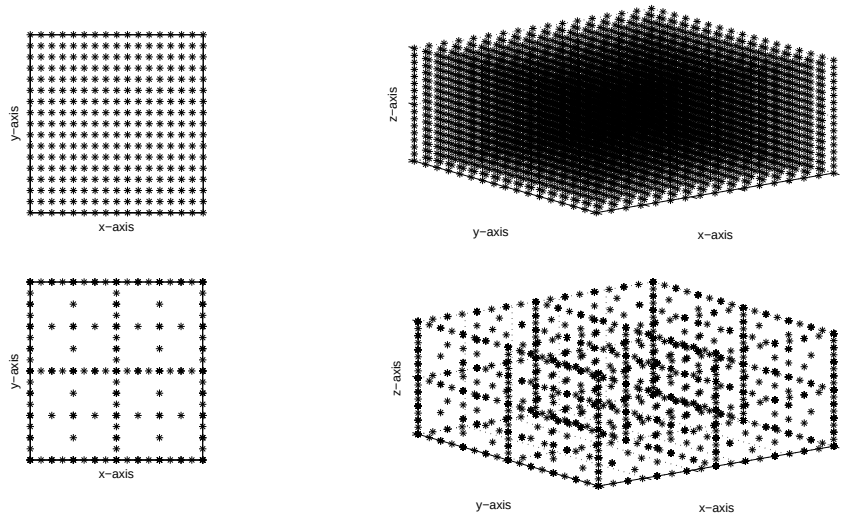


Figure 2.6: Uniform full grid and sparse grid in two (left) and three (right) dimensions for level $n = 4$

2.3.2 Combination Technique

The sparse grid combination technique, which was first introduced by Griebel, Schneider and Zenger in 1992 [53], has been discussed in a number of studies [17], [18] and [19]. This technique is derived from the previously introduced sparse grid method in order to increase the amenable properties it provides us. According to the sparse grid combination technique, sparse grid interpolation for some functions can be computed from the linear combination of interpolation $f_{\mathbf{l}}$ which is calculated on the respective coarse grids, since sparse grid $\mathcal{S}_{n,d}$ can be stated as a superposition of a number of (much coarser) full grids $\Upsilon_{\mathbf{l}}$. This combination is shown in Figure 2.7. Thus, this technique provides us with an acceptable approximation accuracy and low memory requirements by combining a number of smaller approximations. In other words, combination technique provides better performance by com-

puting several sub interpolation problems with using small amount of grid points. For this reason, it is a highly effective method for both RBF interpolation and quasi-interpolation.

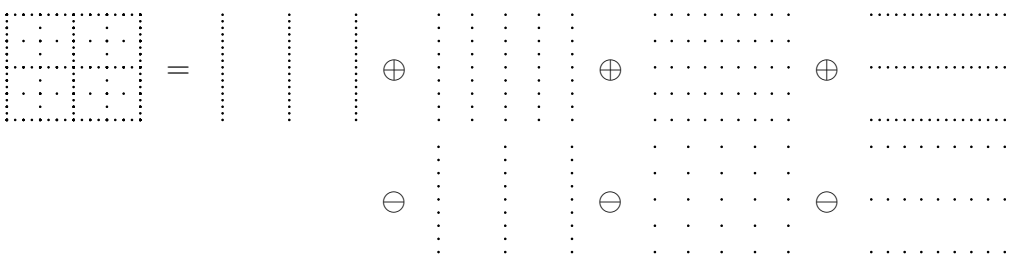


Figure 2.7: Sparse Grid $\mathcal{S}_{4,2}$ as a combination of coarser full grids

Let $f_{\mathbf{l}}$ be a partial interpolant of function f_n on a definite sequence of anisotropic grids $\Upsilon_{\mathbf{l}}$, $\mathbf{l} = (l_1, \ldots, l_d)$. These grids are placed in each coordinate direction with different but uniform mesh sizes. So, the interpolation of function f_n by using these partial interpolants $f_{\mathbf{l}}$ is given by the following combination formula [30], [70]:

$$f_n(\mathbf{x}) = \sum_{q=0}^{d-1} (-1)^q C_q^{d-1} \sum_{|\mathbf{l}|_1 = n+(d-1)-q} f_{\mathbf{l}}(\mathbf{x}), \tag{2.36}$$

where

$$f_{\mathbf{l}} = \sum_{i_1=0}^{2^{l_1}}, \ldots, \sum_{i_d=0}^{2^{l_d}} \lambda_{\mathbf{l},\mathbf{i}} \cdot \triangle_{\mathbf{l},\mathbf{i}}(\mathbf{x}). \tag{2.37}$$

For example, the 2-D case is given by

$$f_n(\mathbf{x}) = \sum_{|\mathbf{l}|_1 = n+1} f_{\mathbf{l}}(\mathbf{x}) - \sum_{|\mathbf{l}|_1 = n} f_{\mathbf{l}}(\mathbf{x}), \tag{2.38}$$

and the 3-D combination formula is given by

$$f_n(\mathbf{x}) = \sum_{|\mathbf{l}|_1=n+2} f_{\mathbf{l}}(\mathbf{x}) - 2 \sum_{|\mathbf{l}|_1=n+1} f_{\mathbf{l}}(\mathbf{x}) + \sum_{|\mathbf{l}|_1=n} f_{\mathbf{l}}(\mathbf{x}). \tag{2.39}$$

There are two significant characteristic properties of the sparse grid combination technique which leads it to be superior to the direct discretization on sparse grids. First of all, one can obtain acceptable solutions for complicated problems with this technique by using existing codes. Secondly, easy computability of different subproblems via parallel computation makes this technique perfectly suited to modern high performance computers [51], [52], [63].

Although the sparse grid combination technique uses some points more than once, it still needs only a small amount of memory for computation and it provides good results in high dimensions in comparison with the full grids. In [45] and [59], some recent studies of the sparse grid combination technique are presented in the literature.

Chapter 3

Multilevel Sparse Kernel Based Interpolation

As discussed in the previous chapter, the sparse grid technique provides us with a remarkable decrease in computational cost since it restricts the data size for linear interpolation. Thus, for this reason, one has obtained logarithmic loss of convergence rate for this method when applied to multidimensional cases in comparison with classical techniques. For instance, in [86], a direct sparse grid method has been applied by means of tensor products of one-dimensional RBF. In this study, although there is an error analysis which has been developed by using the tensor product behaviour, computational evaluation has not given by the author. In addition to these, the new method of sparse kernel-based interpolation (SIK) has been introduced by Georgoulis, Levesley and Subhan [46] in 2013. This method uses anisotropic radial basis function interpolation on partial grids and then linearly combines

them with the combination technique. In other words, SIK can be obtained by solving many sub-interpolation problems with properly selected subgrids and then linearly combining them. This technique provides us with a considerable advantage in the interpolation of huge amounts of data in multiple dimensions.

3.1 Sparse interpolation with Kernels

Let $u : \Omega \to \mathbb{R}$, such that $\Omega := [0,1]^d$ and $u(\mathbf{x}) \in \mathbb{R}$ for $\mathbf{x} = \{x_1, \ldots, x_d\} \in \Omega$. Let $\{(\mathbf{x}_i, u_i), u_i = u(\mathbf{x}_i), i = 1, \cdots, N\}$, be data-sampled from an unknown function u at a finite point set $X = \{\mathbf{x}_1, \cdots, \mathbf{x}_N\} \subset \Omega$. The main goal of this kind of interpolation problem is to find a suitable function, interpolant $I : \mathbb{R}^d \to \mathbb{R}$, which holds for the interpolation condition

$$I(\mathbf{x}_j) = u(\mathbf{x}_j), \quad 1 \leq j \leq N. \tag{3.1}$$

The key idea is to use the anisotropic RBF discussed in Chapter 1 because of the anisotropic behaviour of each subgrid $\Upsilon_{\mathbf{l}}$. For this reason we need a transformation coefficient matrix $A = A_{\mathbf{l}} = \mathrm{diag}(2^{l_1}, \cdots, 2^{l_d})$, which provides us anisotropic RBFs $\psi_{A_l}(\|x - x_j\|)$ for dealing with this undesirable property. Thus the well-posedness problem has been solved as well.

All in all, the anisotropic RBF interpolation $I_u^{\mathbf{Ani.}}(\mathbf{x})$ of u at the grids $\Upsilon_{\mathbf{l}}$ is

then defined by

$$I_{u_{\mathbf{l}}}^{\mathbf{Ani.}}(\mathbf{x}) = \sum_{j=1}^{\mathcal{P}_{\mathbf{l}}} \alpha_j \psi_{A_{\mathbf{l}}}(\|\mathbf{x} - \mathbf{x_j}\|) + \sum_{k=1}^{M} \upsilon_k d_k(A_{\mathbf{l}}\mathbf{x}), \quad \mathbf{x} \in \Omega. \tag{3.2}$$

where $\mathcal{P}_{\mathbf{l}}$ is the number of sparse grids and α_j needs to be chosen such that the interpolation requirements

$$\left[I_{u_{\mathbf{l}}}^{\mathbf{Ani.}} = u\right]_{\Upsilon_{\mathbf{l}}}, \tag{3.3}$$

must be satisfied. The SIK method which has been introduced in [46] is then defined by

$$I_n(\mathbf{x}) = \sum_{q=0}^{d-1} (-1)^q C_q^{d-1} \sum_{|\mathbf{l}|_1 = n+(d-1)-q} I_{u_{\mathbf{l}}}^{\mathbf{Ani.}}(\mathbf{x}). \tag{3.4}$$

The above formula was first presented in [30] for Lagrange interpolation and then in [53] for the numerical solution of elliptic PDE using the FEM on sparse grids by using the combination method. For instance, in 2-D, equation (3.4) becomes

$$I_n(\mathbf{x}) = \sum_{|\mathbf{l}|_1 = n+1} I_{u_{\mathbf{l}}}^{\mathbf{Ani.}}(\mathbf{x}) - \sum_{|\mathbf{l}|_1 = n} I_{u_{\mathbf{l}}}^{\mathbf{Ani.}}(\mathbf{x}). \tag{3.5}$$

3.2 Tensor product kernels give interpolatory schemes

We shall show that the combination formula (3.4) is, indeed, an interpolant. To highlight the key ideas of the proof, we first consider an example in three

dimensions. A two dimensional example is too straightforward, and a four dimensional one is already too complicated.

Setting, for instance, $d = 3$ and $n = 7$, we seek to compute the value of the sparse kernel-based interpolant I_7 to the function f at the point $a = (1/4, 1/8, 1/16) = (2^{-2}, 2^{-3}, 2^{-4})$, which first appears in the grid with multi-index $\mathbf{l} = (2, 3, 4)$. From (3.4), we see that for $d = 3$, the sub-grids of the previous two levels are linearly combined to give I_7. Hence, the set of multi-indices which index the approximation are $\{\alpha : |\alpha| = 7, 8, 9\}$.

We decompose these sets of multi-indices into two groups: The first sets of grids, let us call them Group 1, have two of the three components of the multi-index less than the corresponding components for the multi-index for the point, e.g., the grids represented by the multi-indices $\{\alpha = (k, 1, 1) : k = 5, 6, 7\}$. The remaining grids, let us call them Group 2, have only one component less than the corresponding component for the point; for instance the family of grids $\{\alpha = (k, l, 2) : k + l \leq 7,\ k \geq 2,\ l \geq 3\}$.

We shall now study the values of the function f on a typical set of grids from Group 1, $\{(k, 1, 1) : k = 5, 6, 7\}$. To this end, we consider the cardinal functions from points on these grids, and their values at the point a. Let $b = (r_1 2^{-k}, r_2/2, r_3/2)$, for some $r_i \in \mathbb{N}_0$, $i = 1, 2, 3$, where $0 \leq r_1 \leq 2^k$, $0 \leq r_2, r_3 \leq 2$, with cardinal function $\chi_{(k,1,1),b}$. Then

$$\chi_{(k,1,1),b}(a) = \chi_{k,r_1 2^{-k}}(1/4)\chi_{1,r_2/2}(1/8)\chi_{1,r_3/2}(1/16) = 0,$$

if $r_1 2^{-k} \neq 1/4$, since $k \geq 2$. If $r_1 2^{-k} = 1/4$ then

$$\chi_{(k,1,1),b}(a) = \chi_{1,r_2/2}(1/8)\chi_{1,r_3/2}(1/16)$$

which is independent of k.

Thus $(1/4, r_2/2, r_3/2)$ is a node on all grids $\{\Upsilon_{(k,1,1)} : k = 5, 6, 7\}$, and the value of the cardinal functions $\chi_{(k,1,1),b}$ are identical at a. Hence, the contribution to the interpolant from $f(b)$, on these grids, is

$$\begin{aligned} & \sum_{q=0}^{2} (-1)^q \binom{2}{q} f(b)\chi_{(7-q,1,1),b}(a) \\ = \;& f(b)\chi_{1,r_2/2}(1/8)\chi_{1,r_3/2}(1/16) \sum_{q=0}^{2} (-1)^q \binom{2}{q} \\ = \;& 0. \end{aligned}$$

If we combine the results of the last two paragraphs we see that the contribution to the interpolant from all of the points on grids from Group 1 is 0.

We now turn to grids from Group 2. Let us consider points on the grids $\Upsilon_{(k,l,2)}$ for $k + l \leq 7$, $k \geq 2$, and $l \geq 3$, and their values at a. Let $b = (r_1 2^{-k}, r_2 2^{-l}, r_3/4)$, for some $0 \leq r_1 \leq 2^k$, $0 \leq r_2 \leq 2^l$, and $0 \leq r_3 \leq 4$ with cardinal function $\chi_{(k,l,2),b}$. Then,

$$\chi_{(k,l,2),b}(a) = \chi_{k,r_1 2^{-k}}(1/4)\chi_{l,r_2 2^{-l}}(1/8)\chi_{2,r_3/4}(1/16) = 0,$$

unless $r_1 2^{-k} = 1/4$ and $r_2 2^{-l} = 1/8$, since $k \geq 2$ and $l \geq 3$. If $r_1 2^{-k} = 1/4$

and $r_2 2^{-l} = 1/8$, then $\chi_{(k,l,2),b}(a) = \chi_{2,r_3/4}(1/16)$, which is independent of k and of l.

Thus, $(1/4, 1/8, r_3/4)$ is contained on all grids $\Upsilon_{(k,l,2)}$ for $k+l \leq 7$, $k \geq 2$, and $l \geq 3$, and the value of the cardinal functions $\chi_{(k,l,2),b}$, for all the permissible values of k are identical at a. Hence, the contribution to the interpolant from $f(b)$, on these grids, is

$$\begin{aligned}
&\sum_{q=0}^{2}(-1)^q\binom{2}{q}f(b)\sum_{k+l\leq 7-q,\ k\geq 2,\ l\geq 3}\chi_{(k,l,2),b}(a)\\
&= f(b)\chi_{2,r_3/4}(1/16)\sum_{q=0}^{2}(-1)^q\binom{2}{q}\operatorname{card}\{(k,l): k+l\leq 7-q, k\geq 2,\ l\geq 3\}\\
&= f(b)\chi_{2,r_3/4}(1/16)\sum_{q=0}^{2}(-1)^q\binom{2}{q}(3-q) \;=\; 0.
\end{aligned}$$

Therefore, the contribution to the interpolant from all grids in Group 2 is also zero.

Thus, the only grid contributing to the interpolant is $\Upsilon_{(2,3,4)}$, and the only cardinal function from that grid which takes non zero values at a is the cardinal function at a itself. Therefore, the SIK I_7 is, indeed, an interpolant at all points of $\Upsilon^{7,3}$. Hopefully, this example highlights clearly the key role that tensor-product nature of the kernel has in the proof of interpolation property. The results we shall present in this section are given extendedly in [31].

Equipped with the insight gained by the above example, we shall now consider the general case. To this end, we begin with the following counting

result.

Lemma 3.1. Let $\mathbf{k} \in \mathbb{N}^d$ and $|\mathbf{k}| < p \in \mathbb{N}$. Then,

$$\text{card}\,\{\mathbf{j} \in \mathbb{N}^d : \mathbf{j} \geq \mathbf{k},\ |\mathbf{j}| = p\} = \binom{p - |\mathbf{k}| + d - 1}{d - 1}.$$

Proof: For $d = 1$, the result is immediate, since for $k < p$,

$$\text{card}\,\{j \in \mathbb{N} : j \geq k,\ j = p\} = 1.$$

For $d \geq 2$, let us write $\mathbf{k} = (k_1, \tilde{\mathbf{k}})$ and consider the set $\{\tilde{\mathbf{j}} \in \mathbb{N}^{d-1} : |\tilde{\mathbf{j}}| = p - j_1\}$, for $j_1 = k_1, k_1 + 1, \cdots, p - |\tilde{k}|$. Then, by induction,

$$\text{card}\,\{\tilde{\mathbf{j}} \in \mathbb{N}^{d-1} : \tilde{\mathbf{j}} \geq \tilde{\mathbf{k}},\ |\tilde{\mathbf{j}}| = p - j_1\} \quad = \quad \binom{p - j_1 - |\tilde{\mathbf{k}}| + d - 1}{d - 1}.$$

Thus,

$$\begin{aligned}
\text{card}\,\{\mathbf{j} \in \mathbb{N}^d : \mathbf{j} \geq \mathbf{k},\ |\mathbf{j}| = p\} &= \sum_{j_1 = k_1}^{p - |\tilde{\mathbf{k}}|} \text{card}\,\{\tilde{\mathbf{j}} : \tilde{\mathbf{j}} \geq \tilde{\mathbf{k}},\ |\tilde{\mathbf{j}}| = p - j_1\} \\
&= \sum_{j_1 = k_1}^{p - |\tilde{\mathbf{k}}|} \binom{p - j_1 - |\tilde{\mathbf{k}}| + d - 1}{d - 1} \\
&= \sum_{j_1 = 1}^{p - |\mathbf{k}| + 1} \binom{j_1 + d - 2}{d - 1} \\
&= \binom{p - |\mathbf{k}| + d}{d},
\end{aligned}$$

using the well-known summation formula for binomial coefficients; e.g., [1].

□

We are now ready to state and prove the main result of this section.

Theorem 3.2. *Assuming that the interpolation kernel has the form*

$$\mu(\mathbf{y}) = \prod_{i=1}^{d} \varphi(y_i),$$

then sparse interpolation with this kernel is an interpolatory scheme.

Proof: We wish to compute the value of the sparse grid interpolant I_n to the function f at the point $\mathbf{a} = (a_1 2^{-l_1}, a_2 2^{-l_2}, \cdots, a_d 2^{-l_d})$, with $0 \leq a_j \leq 2^{l_j}$, $1 \leq j \leq d$, with at least one of the $a_j \in \mathbb{N}_0$ odd (this ensures that $\Upsilon_{\mathbf{l}}$ is the grid with the smallest index in which this points appears). From hypothesis, the kernel is of tensor-product form, which implies the cardinal functions for interpolation are of the form

$$\chi_{\mathbf{m},\mathbf{x}} = \prod_{j=1}^{d} \chi_{m_j, x_j 2^{-m_j}}. \tag{3.6}$$

In order to create a decomposition of the indices as in the example above, we need to introduce some notation. Let $d_1, d_2 \in \mathbb{N}$, with $d_1 + d_2 = d$. Let $\mathbf{n}_1 \in \mathbb{N}^{d_1}$ with $\mathbf{n}_1(j) \in \{1, 2, \cdots, d\}$, $j = 1, 2, \cdots, d_1$, and $\mathbf{n}_1(j) < \mathbf{n}_1(j+1)$, $j = 1, 2, \cdots, d_1 - 1$. Similarly, let $\mathbf{n}_2 \in \mathbb{N}^{d_2}$ with $\mathbf{n}_2(j) \in \{1, 2, \cdots, d\}$, $j = 1, 2, \cdots, d_2$, and $\mathbf{n}_2(j) < \mathbf{n}_2(j+1)$, $j = 1, 2, \cdots, d_2 - 1$. Additionally, $\mathbf{n}_1(j) \neq \mathbf{n}_2(k)$ for any j, k. In other words, the components of $\mathbf{n}_1$ and $\mathbf{n}_2$ exhaust the set $\{1, 2, \cdots, d\}$, and these numbers are ordered within the vectors $\mathbf{n}_1$ and $\mathbf{n}_2$. We should note that once $\mathbf{n}_1$ is specified, $\mathbf{n}_2$ is uniquely determined and vice versa.

Let $\mathbf{m}_i \in \mathbb{N}^{d_i}$, $i = 1, 2$. Then, let the multi-index $\mathbf{m} = (\mathbf{n}_1, \mathbf{m}_1, \mathbf{n}_2, \mathbf{m}_2) \in \mathbb{N}^d$ have components $\mathbf{m}(k) = \mathbf{m}_i(j)$ if $\mathbf{n}_i(j) = k$, $i = 1, 2$. So, for instance, if $\mathbf{n}_1(3) = 7$, then $\mathbf{m}(7) = \mathbf{m}_1(3)$, and if $\mathbf{n}_2(4) = 5$, then $\mathbf{m}(5) = \mathbf{m}_2(4)$. In this way, we break multi-indices into two pieces in a convenient fashion.

On the other hand, for $\mathbf{m} \in \mathbb{N}^d$, let $\mathbf{m}_{\mathbf{n}_1} \in \mathbb{N}^{d_1}$ with $\mathbf{m}_{\mathbf{n}_1}(i) = \mathbf{m}(\mathbf{n}_1(i))$, $i = 1, 2, \cdots, d_1$ and $\mathbf{m}_{\mathbf{n}_2} \in \mathbb{N}^{d_2}$ with $\mathbf{m}_{\mathbf{n}_2}(i) = \mathbf{m}(\mathbf{n}_2(i))$, $i = 1, 2, \cdots, d_2$. Then, we have the identity

$$\mathbf{m} = (\mathbf{n}_1, \mathbf{m}_{\mathbf{n}_1}, \mathbf{n}_2, \mathbf{m}_{\mathbf{n}_2}).$$

Now, for each $d_1 = 1, 2, \cdots, d-1$, and $\mathbf{n}_1 \in \mathbb{N}^{d_1}$, let

$$I(\mathbf{l}, d_1, \mathbf{n}_1) = \{\mathbf{m}_1 : \mathbf{m}_1 < l_{\mathbf{n}_1}\}.$$

For each $\mathbf{m}_1 \in I(\mathbf{l}, d_1, \mathbf{n}_1)$, define

$$J(\mathbf{l}, d_1, \mathbf{n}_1, \mathbf{m}_1) = \{(\mathbf{n}_1, \mathbf{m}_1, \mathbf{n}_2, \mathbf{m}_2) : \mathbf{m}_2 \in \mathbb{N}^{d_2},\ \mathbf{m}_2 \geq \mathbf{l}_{\mathbf{n}_2}\}.$$

The sets $J(\mathbf{l}, d_1, \mathbf{n}_1, \mathbf{m}_1)$ partition all multi-indices into sets with a fixed set of components of the multi-index less than those of $\mathbf{l}$ and the remaining components greater than or equal to those of $\mathbf{l}$. The only multi-index missing from this set is $\mathbf{l}$ itself.

We compute the cardinality of the subsets of $J(\mathbf{l}, d_1, \mathbf{n}_1, \mathbf{m}_1)$, given by

$$J(\mathbf{l}, d_1, \mathbf{n}_1, \mathbf{m}_1, q) = \{\mathbf{m} \in J(\mathbf{l}, d_1, \mathbf{n}_1, \mathbf{m}_1) : |\mathbf{m}| = n + d - 1 - q\},$$

for $q = 0, 1, \cdots, d-1$, which, using Lemma 3.1, is given by

$$\operatorname{card} J(\mathbf{l}, d_1, \mathbf{n}_1, \mathbf{m}_1, q) \;=\; \binom{n + d + d_1 - 2 - |\mathbf{l}_{\mathbf{n}_2}| - q}{d_1 - 1}. \tag{3.7}$$

We now consider the contribution to the interpolant from a typical point on one of the grids indexed by elements of $J(\mathbf{l}, d_1, \mathbf{n}_1, \mathbf{m}_1)$.

Let $\mathbf{m} = (\mathbf{n}_1, \mathbf{m}_1, \mathbf{n}_2, \mathbf{m}_2) \in J(\mathbf{l}, d_1, \mathbf{n}_1, \mathbf{m}_1)$. Then, by definition, $\mathbf{l}_{\mathbf{n}_2} \geq \mathbf{m}_2$. Let $\mathbf{x} = (x_1 2^{-m_1}, x_2 2^{-m_2}, \cdots x_d 2^{-m_d})$, for some $0 \leq x_j \leq 2^{m_j}$, $j = 1, 2, \cdots, d$, with cardinal function $\chi_{\mathbf{m},\mathbf{x}}$. Then,

$$\chi_{\mathbf{m},\mathbf{x}}(\mathbf{a}) = \prod_{i=1}^{d} \chi_{m_i, x_i 2^{-m_i}}(a_i 2^{-l_i}).$$

Hence, $\chi_{\mathbf{m},\mathbf{x}}(\mathbf{a}) \neq 0$ only if $x_i 2^{-m_i} = a_i 2^{-l_i}$, $i = \mathbf{n}_1(j)$, $j = 1, 2, \cdots, d_2$. In this case,

$$\chi_{\mathbf{m},\mathbf{x}}(\mathbf{a}) = \prod_{j=1}^{d_1} \chi_{m_{\mathbf{n}_1(j)}, x_{\mathbf{n}_1(j)} 2^{-m_{\mathbf{n}_1(j)}}}(a_{\mathbf{n}_1(j)} 2^{-l_{\mathbf{n}_1(j)}}),$$

which is independent of $\mathbf{m}_2$.

So we have, for fixed $\mathbf{n}_1$ and $\mathbf{m}_1$, the same contribution at $\mathbf{a}$ from any individual point which appears in a grid in $J(\mathbf{l}, d_1, \mathbf{n}_1, \mathbf{m}_1)$. Thus, the contribution

to from $\mathbf{x}$ at point $\mathbf{a}$ is

$$\begin{aligned}
&\sum_{q=0}^{d-1}(-1)^q\binom{d-1}{q}f(\mathbf{x})\sum_{\mathbf{m}\in J(\mathbf{l},d_1,\mathbf{n}_1,\mathbf{m}_1,q)}\chi_{\mathbf{m},\mathbf{x}}(\mathbf{a})\\
&\quad= f(\mathbf{x})\prod_{j=1}^{d_1}\chi_{m_{\mathbf{n}_1(j)},x_{\mathbf{n}_1(j)}2^{-m_{\mathbf{n}_1(j)}}}(a_{\mathbf{n}_1(j)}2^{-l_{\mathbf{n}_1(j)}})\\
&\qquad\qquad\times\sum_{q=0}^{d-1}(-1)^q\binom{d-1}{q}\operatorname{card}J(\mathbf{l},d_1,\mathbf{n}_1,\mathbf{m}_1,q)\\
&\quad= f(\mathbf{x})\prod_{j=1}^{d_1}\chi_{m_{\mathbf{n}_1(j)},x_{\mathbf{n}_1(j)}2^{-m_{\mathbf{n}_1(j)}}}(a_{\mathbf{n}_1(j)}2^{-l_{\mathbf{n}_1(j)}})\\
&\qquad\qquad\times\sum_{q=0}^{d-1}(-1)^q\binom{d-1}{q}\binom{n+d+d_1-2-|\mathbf{l}_{\mathbf{n}_2}|-q}{d_1-1}\\
&\quad= 0,
\end{aligned}$$

since the binomial coefficient is a polynomial in q of degree less than $d-1$, which is thus annihilated by the difference operator.

Thus, we see that all contributions from points in other grids other than $X_{\mathbf{l}}$ are zero at points in $X_{\mathbf{l}}$. Clearly, the only contribution at $\mathbf{a}\in X_{\mathbf{l}}$ from points in $X_{\mathbf{l}}$ will be from $\mathbf{a}$ itself. □

3.3 Multilevel sparse interpolation with kernels

In 2013, Georgoulis, Levesley and Subhan [46] introduced the Multilevel Sparse Interpolation with Kernels (MuSIK, for short) method (previously referred to as MLSKI). The main motivation of MuSIK is using the residual interpolation with the SIK method for each level.

Indeed, the MuSIK algorithm uses the same principle as the classical multilevel RBF algorithm. One can commence the MuSIK algorithm by computing the sparse interpolation with kernels I_{n_0} on the coarsest sparse grid $\mathcal{S}_{n_0,d}$ and set $\Delta_0 := I_{n_0}$. Then, for each remaining sparse grid ($j = 1, \ldots, n$), Δ_j is the sparse grid interpolant to the residual

$$f - \sum_{i=1}^{j-1} \Delta_i,$$

on $\mathcal{S}_{j,d}$. After all these calculations, multilevel sparse interpolation with kernels can be obtained, which is

$$I_n^{MuSIK} := \sum_{i=1}^{n} \Delta_i. \tag{3.8}$$

3.4 Tensor product of univariate cardinal functions

Gaussian kernels on $\mathbb{R}^d$ can be viewed as tensor products of univariate Gaussians. Of course this is also valid for anisotropic version of Gaussian kernels. This property will be used below result.

Corollary 3.3. Multilevel versions of sparse interpolations with Gaussian kernels are interpolatory on the kernel centres.

Proof: According to Theorem 3.2 the sparse interpolation with Gaussian kernels is an interpolatory scheme. By using this result one can say that

$u - \sum_{j=0}^{k-1} \Delta_j$ is interpolatory on $\mathcal{S}_{k,d}$, for each k. This implies that the multilevel versions of the sparse interpolations with Gaussian kernels $I_n^{MuSIK} := \sum_{j=0}^{n} \Delta_j$ is also interpolatory. $\square$

Hence, it is possible to compute the cardinal functions for multivariate approximation by computing *ab initio* (to arbitrarily high precision, e.g., by using symbolic calculators) the cardinal functions for univariate approximation up to (for instance) $5, 9, 17, \cdots, 129$ equally spaced points, and store these.

Chapter 4

Quasi Sparse Interpolation with Kernels

One of the most powerful aspects of quasi-interpolation is that it can be generalized easily to the multidimensional case because of its simple structure. In previous chapters, we have seen some basis functions which have a number of nice properties for quasi-interpolation on a uniform grid in $\mathbb{R}^n$ such as simplicity, smoothness and exponential decay behaviour at infinity. Thus, this kind of basis function makes multidimensional approximation possible with basic analytical representations.

Another advantage of quasi-interpolation is that there is no need to solve large algebraic systems. For example, for some approximation techniques, such as radial basis functions interpolation, one needs to solve large number of matrix systems to find the best approximation and require significant stor-

age for univariate problems solutions. However, due to the nature of quasi-interpolation, we just need to know function value at specific points. Thus, this makes quasi-interpolation useful in practice not only for low dimensional approximations but also in multidimensional cases. Also integration using quasi interpolation methods provides us positive weights because of it nature.

Now, in order to provide more benefits from nice properties of quasi-interpolation, we will propose to apply this new technique to quasi-interpolation, which is called the quasi-sparse kernel-based interpolation (Q-SIK) method. Thus, we similarly expect to obtain results with lower computational cost, especially in the interpolation of huge amount of data in multiple dimensions.

4.1 Sparse Grid Construction

In order to clarify sparse grid construction, we need to introduce some multi-index notation. Throughout this and 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. The key point here is that the multi-index $\mathbf{l}$ consists of non zero components. In other words, non-zero elements are an undesired condition for solving interpolation problems.

For the above multi-indices, the cartesian (discrete) full 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

$$\mathbf{x}_{\mathbf{l},\mathbf{i}} := (x_{l_1,i_1}, \ldots, x_{l_d,i_d}), \tag{4.1}$$

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 $\Sigma_{\mathbf{l}}$ in $\Upsilon_{\mathbf{l}}$ by using the formula

$$\mathcal{P}_{\mathbf{l}} := \prod_{p=1}^{d} (2^{l_p} + 1). \tag{4.2}$$

For instance, should one choose constant $\mathbf{l}$, which is equal to n for all $p = 1, \ldots, d$, the cartesian full grid converts to a uniform full grid denoted by $\Gamma^{n,d}$ with $\mathcal{P} = (2^n + 1)^d$, where n is the grid level. In addition to this, the sparse grid $\mathcal{S}_{n,d}$, discussed in the previous chapter, can be obtained by the union of a lot of (much coarser) uniform full grids $\Upsilon_{\mathbf{l}}$.

As can easily be seen, the sparse grid treatment itself already requires the use of anisotropic basis functions. For this reason, we will use anisotropic Gaussian functions for both SIK and Q-SIK.

4.2 Quasi sparse interpolation with kernels

In the previous sections we have discussed the advantageous of both the quasi-interpolation method and the sparse grid combination technique for multidimension interpolation. Both tools provide superior features such as decreasing computational complexity. For example, the SIK method has been introduced, as given in previous chapter, in order to benefit the sparse grid technique.

In this section, we improve on this via a new quasi-SIK interpolation formula

on a sparse grid for a function of several variables. In accordance with this purpose, we solve a number of sub-anisotropic quasi-interpolation problems on well constructed subgrids and then obtain the quasi-SIK interpolant by combining the resultant sub-quasi interpolants linearly.

In a similar way, all the subgrids for each level can be constructed as per the previous section. That is, we have $\Upsilon_{\mathbf{l}}$ with a mesh size $h_{\mathbf{l}}$. Then, in order to compute each subgrid interpolation problem, we need to use the anisotropic quasi-interpolation method; the anisotropic quasi-interpolation $Q_f^{\mathbf{Ani.}}(\mathbf{x})$ of u at the grids $\Upsilon_{\mathbf{l}}$ is defined by

$$Q_{u_{\mathbf{l}}}^{\mathbf{Ani.}}(\mathbf{x}) = \sum_{\mathbf{k}\in\Upsilon_{\mathbf{l}}} u(\mathbf{k}h)\mu_{A_{\mathbf{l}}}\left(\frac{\mathbf{x}}{h} - \mathbf{k}\right), \quad \mathbf{x}\in\mathbb{R}^n, \quad h>0, \tag{4.3}$$

where μ_A is the anisotropic Gaussian basis function. Then we obtain the quasi-SKI by combining the all sub-quasi-interpolation problems with the combination technique. In other words, the quasi-SKI is defined by

$$Q_n(\mathbf{x}) = \sum_{q=0}^{d-1}(-1)^q C_q^{d-1} \sum_{|\mathbf{l}|_1=n+(d-1)-q} Q_{u_{\mathbf{l}}}^{\mathbf{Ani.}}(\mathbf{x}). \tag{4.4}$$

For example, in 3-D the quasi-SIK is

$$Q_n(\mathbf{x}) = \sum_{|\mathbf{l}|_1=n+2} Q_{u_{\mathbf{l}}}^{\mathbf{Ani.}}(\mathbf{x}) - 2\sum_{|\mathbf{l}|_1=n+1} Q_{u_{\mathbf{l}}}^{\mathbf{Ani.}}(\mathbf{x}) + \sum_{|\mathbf{l}|_1=n} Q_{u_{\mathbf{l}}}^{\mathbf{Ani.}}(\mathbf{x}). \tag{4.5}$$

The key idea here is subtracting the redundant grids visited on several occasions. That is, the middle term on the right hand side of equation (4.5) serves to remove any undesired grids used more than once, which comes from

the first and third term on the right hand side of equation (4.5). We can summarize the algorithm of sparse kernel-based interpolation in the following flow chart.

This technique can be perceived as a modified version of the idea of the sparse kernel-based interpolation. For this reason, the quasi-SIK algorithm can be implemented with only a small modification of existing SIK code. Of course, the quasi-SIK method will help us when constructing the multilevel algorithm.

Algorithm 1: Q-SIK method

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

Result: The sparse kernel-based interpolation $I_n(\mathbf{x})$

1: Create the subgrids 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: Compute the anisotropic sub-interpolation problems

$Q_u^{\mathbf{Ani.}}(\mathbf{x}) = \sum_{\mathbf{k} \in \Upsilon_{\mathbf{l}}} u(\mathbf{k}h) \mu_A \left(\frac{\mathbf{x}}{h} - \mathbf{k} \right)$

3: Combine the all sub-interpolation problems obtained above

$Q_n(\mathbf{x}) = \sum_{q=0}^{d-1} (-1)^q C_q^{d-1} \sum_{|\mathbf{l}|_1 = n+(d-1)-q} Q_{u_{\mathbf{l}}}^{\mathbf{Ani.}}(\mathbf{x})$

As can be seen in the above explanations, the SIK method is very amenable to parallel computation since each sub-interpolation problem can be solved independently. A significant property of this technique is that no return is needed from long stages of computation, i.e., each sub-interpolation problem is absolutely independent of any other. Therefore this method can be applied in a computational cluster or distributed across workstations. Thus, the

approximation of the entire solution can be obtained after collection of each sub-interpolation problem. For this reason, the SIK method provides us with computationally cheap approximation, especially for multidimensional problems.

4.3 Numerical Experiments in 2-D

In this section, in order to test and compare the SIK and Q-SIK methods, we will present some interpolation results. In accordance with this purpose we have used some test functions, which are given below. Of course, we also will provide classical RBF and quasi-interpolation results to show superior properties of sparse interpolations.

- $F_1^{2d}(x,y)$ (Franke's test function)

$$\begin{aligned} F_1^{2d}(x,y) &= \frac{3}{4}e^{-((9x-2)^2+(9y-2)^2)/4} + \frac{3}{4}e^{-(9x+1)^2/49-(9y+1)^2/10} \\ &+ \frac{1}{2}e^{-((9x-7)^2+(9y-3)^2)/4} - \frac{1}{5}e^{-((9x-4)^2+(9y-7)^2)}. \end{aligned}$$

- $F_2^{2d}(x,y) = 4^2x(1-x)y(1-y)$.
- $F_3^{2d}(x,y) = \sqrt{\frac{18}{\pi}}e^{-(x^2+81y^2)}$.
- $F_4^{2d}(x,y) = \frac{1.25+\cos(5.4y)}{6+6(3x-1)^2}$.
- $F_5^{2d}(x,y) = \sqrt{64-81((x-0.5)^2+(y-0.5)^2)}/9 - 0.5$.
- $F_6^{2d}(x,y) = \max(x-\frac{1}{2})\max(y-\frac{1}{2})$.

The Franke's test function is widely used in RBF literature and can be found in [44]. The others can be found in [36], [8] and [23], respectively. These test functions are illustrated in Figure 4.1.

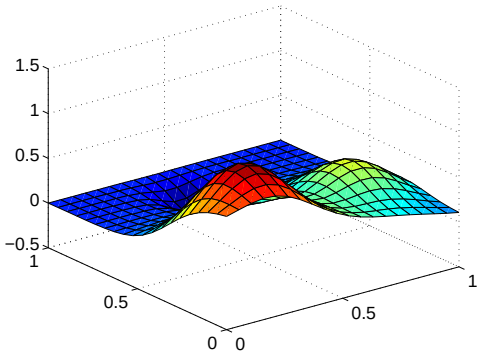
All these numerical experiments have been done using a 160×160 uniform grid in the domain $[0,1]^2 \subset \mathbb{R}^2$. In the numerical experiments, *Max-Error* represents the maximum modulus error, i.e., $\|f-g\|_\infty$ and *Rms-Error* represents the standard root mean squared error, i.e.

$$\sqrt{\frac{\sum_{i=1}^{Neval} |f_i - g_i|^2}{Neval}}, \tag{4.6}$$

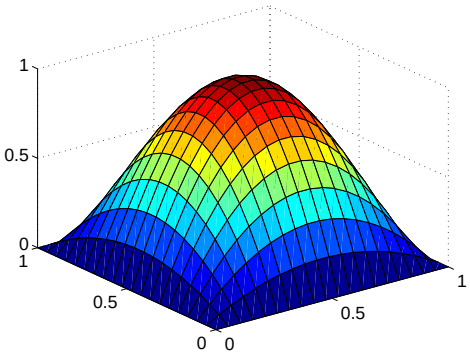
where f is the exact solution, g is the approximate solution, and *Neval* is the number of the test points.

In addition to these, N represents the number of data points for both standard RBF interpolation and quasi-interpolation. In sparse interpolation with kernels, *SGnode* represents the number of nodes in the SIK and Q-SIK methods. Similarly, *DOFs(SG)* represents the total amount of nodes visited in the SIK and Q-SIK methods. Of course, *DOFs(SG)* is much larger than *SGnode* since some nodes are visited several times in every sub-interpolation problem. *Time* stands for the CPU time consumed in every numerical experiment.

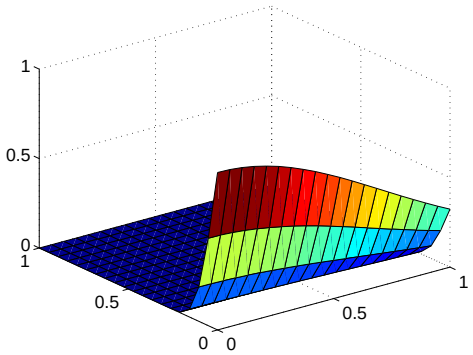
All these experiments have been performed in *MATLAB* since it is suitable for our interpolation algorithm. For the low levels, we have used a standard desktop computer provided by the University of Leicester. However, for high levels, we have used SPECTRE, which has allowed us to run sev-



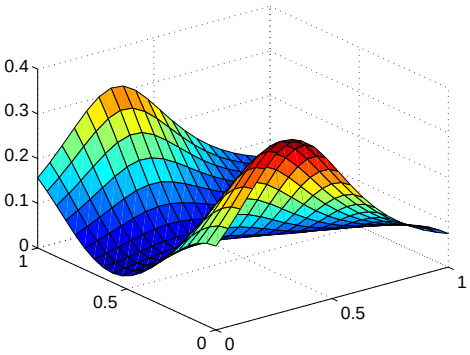
(a) $F_1^{2d}(x, y)$



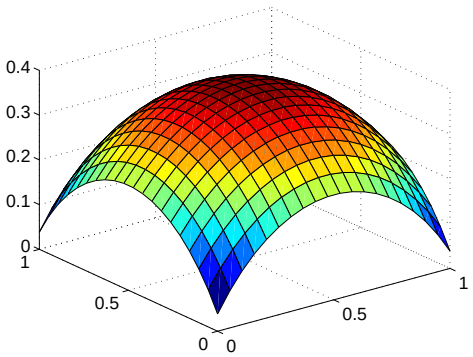
(b) $F_2^{2d}(x, y)$



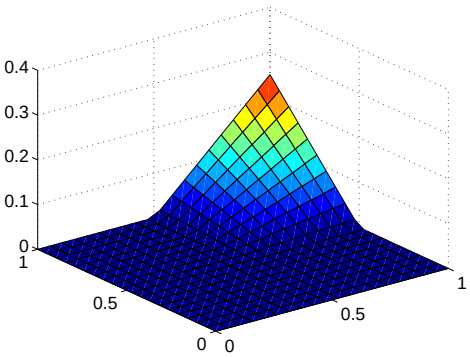
(c) $F_3^{2d}(x, y)$



(d) $F_4^{2d}(x, y)$



(e) $F_5^{2d}(x, y)$



(f) $F_6^{2d}(x, y)$

Figure 4.1: Test functions in two dimensions

eral experiments simultaneously via access a large number of computational nodes. Thus SPECTRE provide us the computational capacity which otherwise would have required a large number of standard computers.

4.3.1 Experiments using Gaussian Tensor Products

In our numerical experiments, we have used the Gaussian basis functions on sparse grids as discussed previous chapters. The tensor product nature of Gaussian basis functions play a key role in our numerical implementation. More specifically, one can write a d-dimensional Gaussian basis function

$$e^{\|A(\boldsymbol{\xi}_i-\boldsymbol{\xi}_j)\|^2} = e^{A|\xi_1^i-\xi_1^j|^2} e^{A|\xi_2^i-\xi_2^j|^2} \cdots e^{A|\xi_d^i-\xi_d^j|^2}, \tag{4.7}$$

where $\boldsymbol{\xi}_i = (\xi_1^i, \xi_2^i, \cdots, \xi_d^i) \in \mathbb{R}^d$. By using this motivation, we first implement quasi-sparse interpolation with Gaussian basis functions.

N	Maximum Error	RMS Error
9	6.402506e-01	1.923844e-01
25	2.617628e-01	6.068551e-02
81	1.707982e-01	2.753333e-02
289	1.028995e-01	1.641467e-02
1089	7.900508e-02	1.490842e-02
4225	7.914526e-02	1.469706e-02

Table 4.1: Quasi-interpolation results using Gaussian basis functions with shape parameter $d = 0.4$, test function $F_1^{2d}(x, y)$, on an equally spaced 160×160 evaluation grid.

Numerical results from some of our numerical experiments have been presented in Table 4.1 and 4.2 for quasi-sparse interpolation with a Gaussian

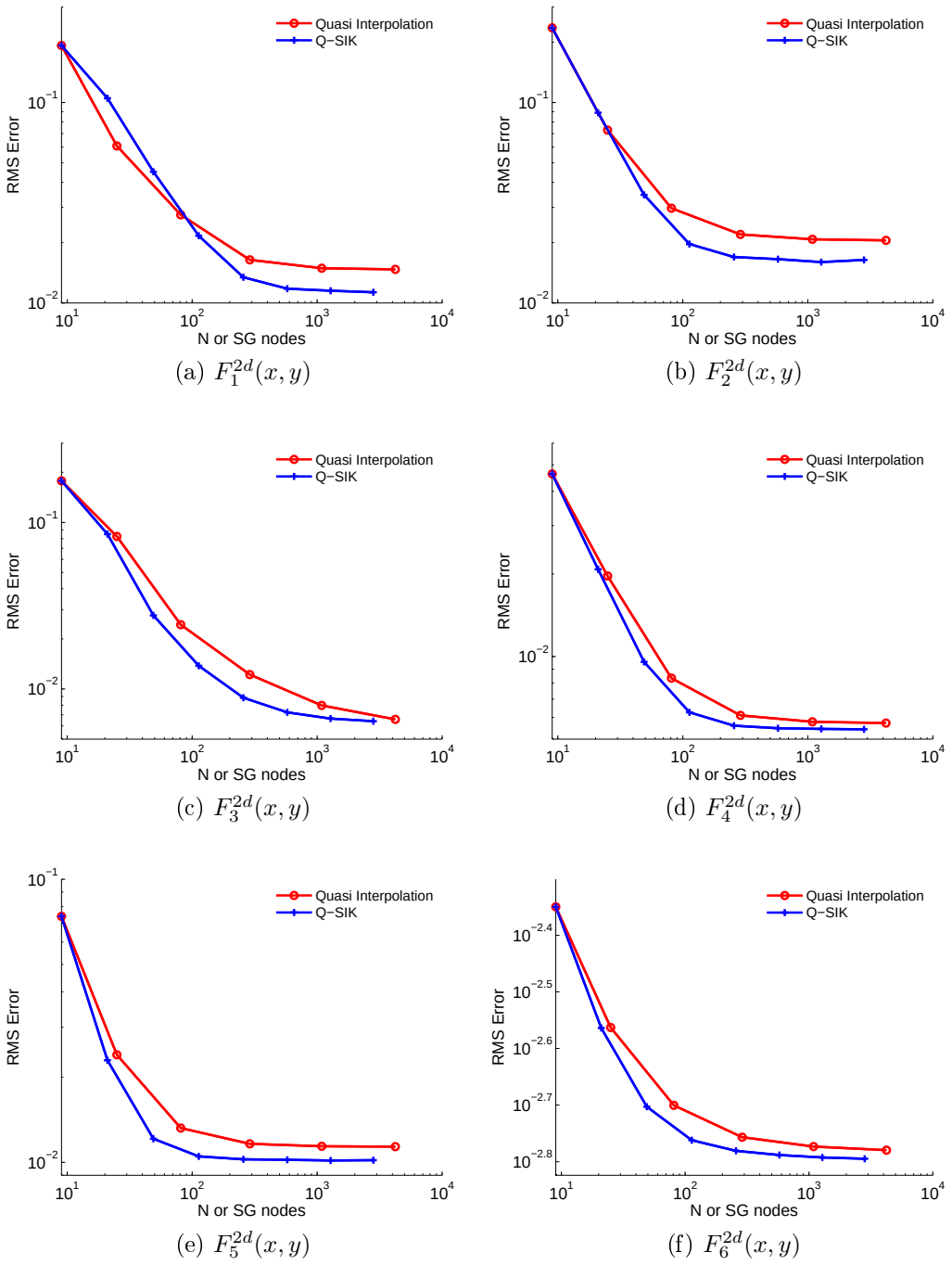
SGnode	DOFs(SG)	Maximum Error	RMS Error
9	9	6.402506e-01	1.923844e-01
21	39	4.071659e-01	1.050494e-01
49	109	2.005483e-01	4.512258e-02
113	271	1.202167e-01	2.163062e-02
257	641	7.342570e-02	1.342539e-02
577	1475	6.952113e-02	1.178005e-02
1281	3333	6.339311e-02	1.150994e-02

Table 4.2: Q-SIK results using Gaussian basis functions with shape parameter $d = 0.4$, test function $F_1^{2d}(x, y)$, on an equally spaced 160×160 evaluation grid.

kernel. These figures permitted a more detailed understanding of these results. These experiments have been made on an equally spaced 160×160 evaluation grid as test points.

In these results, Q-SIK has reached the stable maximum and root mean square errors at a low number of points when compared with quasi-interpolation. Thus, it can be said that Q-SIK obtains interpolation values faster in comparison to classical quasi-interpolation since evaluation time increases in direct proportion to nodes used. Additionally, since the errors remain stable when the number of used nodes increases, Q-SIK accords with the multilevel algorithm which will be examined in detailed in subsequent chapters. In other words, we will apply the multilevel algorithm because the Q-SIK method does not converge.

We present RMS error in terms of the degrees of freedom. In Figures 4.2 (a)-(f), RMS error is plotted versus the number of nodes used for six different test functions. According to this graph, the Q-SIK algorithm provides us



(a) $F_1^{2d}(x, y)$ (b) $F_2^{2d}(x, y)$

(c) $F_3^{2d}(x, y)$ (d) $F_4^{2d}(x, y)$

(e) $F_5^{2d}(x, y)$ (f) $F_6^{2d}(x, y)$

Figure 4.2: RMS error versus N(Quasi-interpolation) or SG (Q-SIK) nodes using Gaussian basis functions with $\rho = 0.4$: Quasi-interpolation (red) and Q-SIK (blue) on a 160×160 uniform grid.

N	Maximum Error	RMS Error
9	3.841200e-01	2.571824e-01
25	1.503965e-01	7.870755e-02
81	6.140495e-02	2.351856e-02
289	3.556470e-02	1.024405e-02
1089	3.066932e-02	8.037256e-03
4225	2.846629e-02	7.709436e-03

Table 4.3: Quasi-interpolation results using Gaussian basis functions with shape parameter $d = 0.4$, test function $F_2^{2d}(x, y)$, on an equally spaced 160×160 evaluation grid.

SGnode	DOFs(SG)	Maximum Error	RMS Error
9	9	3.841200e-01	2.571824e-01
21	39	1.731361e-01	1.009403e-01
49	109	7.123960e-02	3.631736e-02
113	271	3.746303e-02	1.363285e-02
257	641	2.736527e-02	7.253871e-03
577	1475	2.253495e-02	6.036340e-03
1281	3333	2.610709e-02	5.620270e-03

Table 4.4: Q-SIK results using Gaussian basis functions with shape parameter $d = 0.4$, test function $F_2^{2d}(x, y)$, on an equally spaced 160×160 evaluation grid.

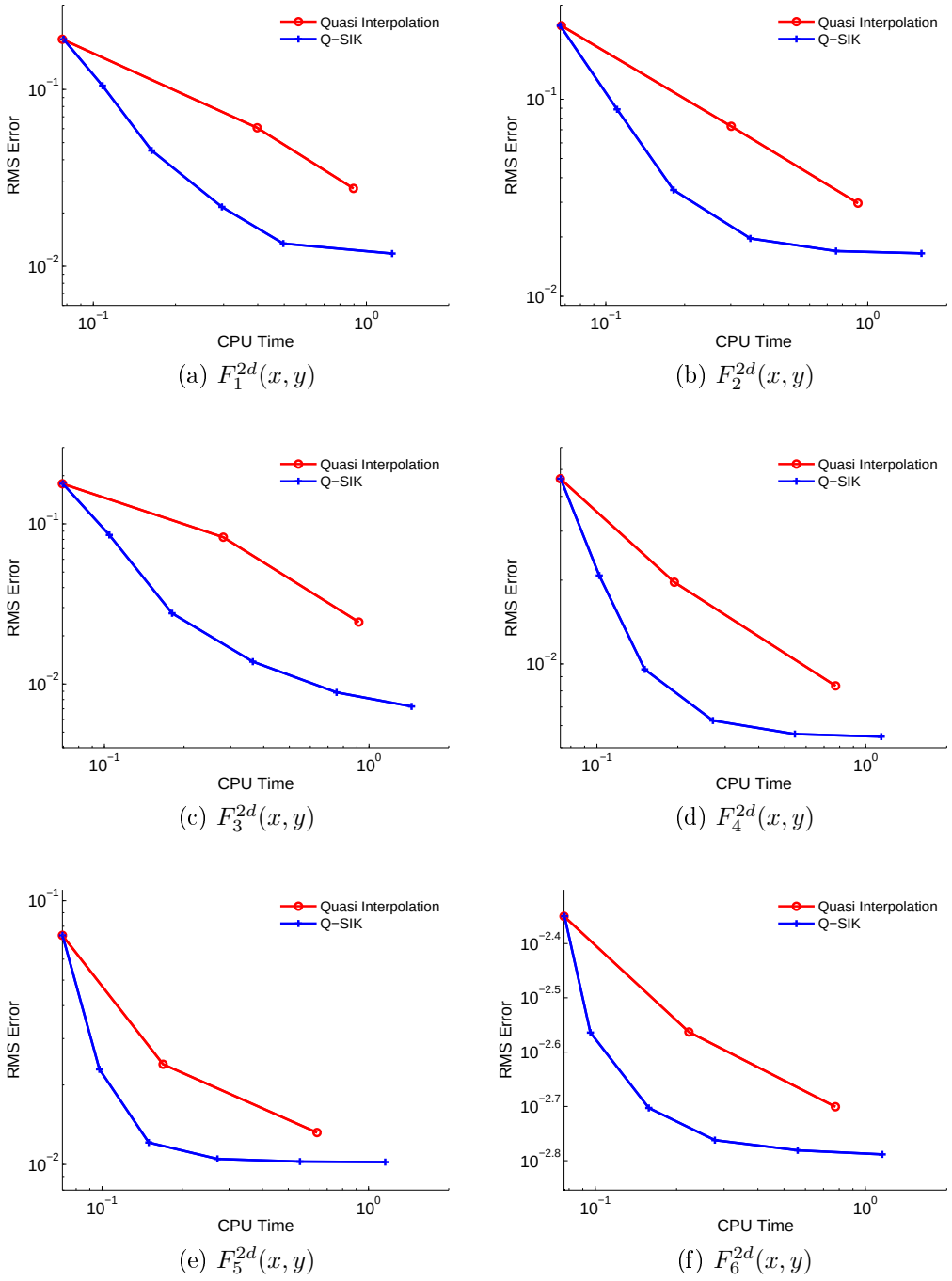
with a high performance interpolation technique. Similarly, in Figures 4.3 (a)-(f), RMS error is plotted verses the computational time for the same test functions. These figures show that the Q-SIK method gives a lower degree of error in comparison with the classical quasi-interpolation within an identical time frame. In either case, these figures confirm that Q-SIK is superior to classical quasi-interpolation. In addition to these, Q-SIK can deal with high dimension problems easily because of its nature. Classical full grid interpolation techniques are limited to solving large problem on high

performance computers. However, the Q-SIK method can overcome these problems by using sparse grids.

SGnode	DOFs(SG)	Maximum Error	RMS Error
9	9	5.464474e-01	1.862580e-01
21	39	3.315317e-01	8.431328e-02
49	109	1.219863e-01	2.253456e-02
113	271	9.849219e-02	1.202155e-02
257	641	7.950969e-02	9.890267e-03
577	1475	7.083609e-02	9.483027e-03
1281	3333	6.564442e-02	9.369861e-03

Table 4.5: Q-SIK results using Gaussian basis functions with shape parameter $d = 0.4$, test function $F_3^{2d}(x, y)$, on an equally spaced 160×160 evaluation grid.

Furthermore, the Q-SIK method shows the same outstanding properties in comparison with the SIK method, which is based on the classical RBF technique. In general, quasi-interpolation is one of the fast approximation methods because of simplicity, smoothness and rapid exponential decay at infinity. Additionally, there is no need to solve large algebraic systems like Radial Basis Function interpolation. For these reasons, one can obtain faster results with the Q-SIK technique, which combines the superior properties of quasi-interpolation with those of the sparse grid method. In Figure 4.4, evaluation time is plotted versus the number of degrees of freedom. According to these figures, quasi-interpolation shows better performance with respect to classical RBF interpolation. Correspondingly, performance of the Q-SIK method is superior to the SIK method for all the above reasons.

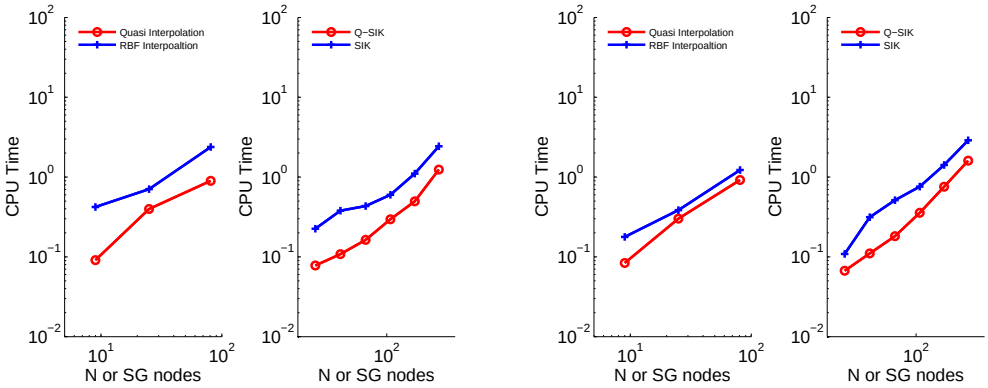


(a) $F_1^{2d}(x, y)$ (b) $F_2^{2d}(x, y)$

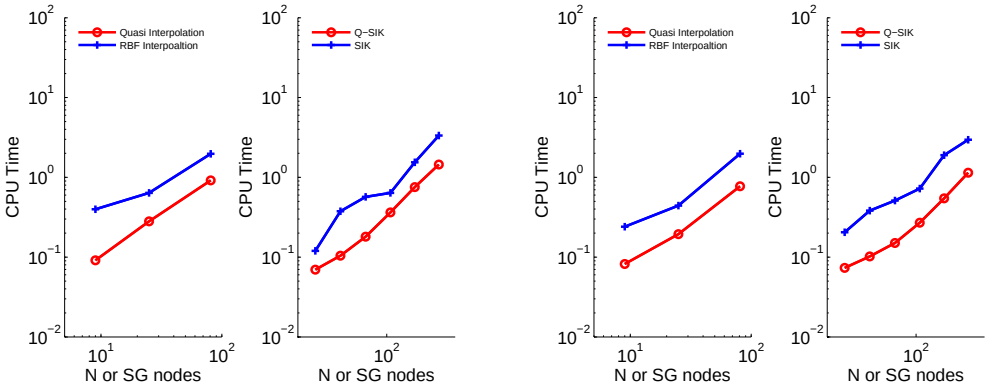
(c) $F_3^{2d}(x, y)$ (d) $F_4^{2d}(x, y)$

(e) $F_5^{2d}(x, y)$ (f) $F_6^{2d}(x, y)$

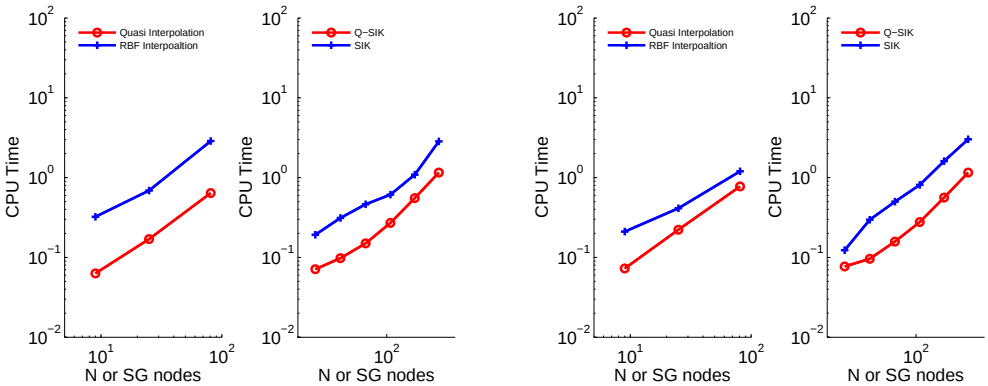
Figure 4.3: RMS error versus CPU time using Gaussian basis functions with $\rho = 0.4$: Quasi-interpolation (red) and Q-SIK (blue) on a 160×160 uniform grid.



(a) $F_1^{2d}(x, y)$ (b) $F_2^{2d}(x, y)$



(c) $F_3^{2d}(x, y)$ (d) $F_4^{2d}(x, y)$



(e) $F_5^{2d}(x, y)$ (f) $F_6^{2d}(x, y)$

Figure 4.4: CPU time versus N(Quasi-interpolation) or SG (Q-SIK) nodes using Gaussian basis functions with $\rho = 0.4$ and $c = 0.45$: Quasi-interpolation, Q-SIK (red) and RBF interpolation, SIK (blue) on a 160×160 uniform grid.

4.4 Discussion

The recommended algorithm, Quasi-Sparse Interpolation with Kernels, is advantageous in dealing with multidimensional problems on a short timescale. This property of the Q-SIK method has been verified by our numerical experiments, shown in the preceeding tables and figures. All these experiments have been done in MATLAB in series.

The main motivation for proposing the Q-SIK method stemmed from the desirable features of the SIK method, as introduced by Georgoulis, Levesley and Subhan [46] in 2013. The SIK method shows outstanding features over classical RBF interpolation. In order to take a step forward, the Q-SIK method has been proposed. One of the most significant properties of quasi-interpolation is that there is no need to solve a large algebraic system. In other words, it can yield straight solutions. This property of quasi-interpolation plays a crucial role in decreasing evaluation time. Furthermore, quasi-interpolation has a number of desirable features, such as good shape properties, locality, and simple computation and evaluation. The proposed algorithm combines all these desirable properties with the advantages of the sparse grid. Thus, Q-SIK provides fast results in comparison with the SIK method.

In the Q-SIK method, we have used the combination technique which causes repeated and frequent visits to certain nodes. Therefore, the Q-SIK method needs more storage space than classical sparse grid algorithms. However, it still takes advantage of the reduced input data requirements of classical sparse grid methods. As a consequence, the Q-SIK technique needs less

memory in comparison to the full grid interpolation method, especially in high dimensions.

Chapter 5

Quasi Multilevel Sparse Interpolation with Kernels

Multilevel methods are one of the most effective contemporary techniques for large scale algebraic systems arising out of both the approximation problems and the discretization of PDEs, which can be found in [78], [80], [68], [39] and [35], respectively. The main idea of this method may be identified as a treatment where one works on a number of levels (on coarse levels), at every turn recovering the residuals which come from the previous level (finer levels). These residuals then need to be added to previous level fits, culminating in a better approximation. Thus the computational level remains stable, which leads to a restrainable error bound of approximation.

The multilevel interpolation algorithm was first proposed by Floater and Iske [40] in 1996 by using compactly supported RBFs. Theoretical investiga-

tions of the multilevel approximation method were subsequently presented by Narcowich, Schaback and Ward [78] and Fasshauer and Jeremo [37] in 1999. The simple point-wise error bounds on the multilevel approximation using polyharmonic splines was been presented by Hales and Levesley [54] in 2002. Additionally, a univariate and multivariate multilevel scheme has been applied to Wu-Schaback's quasi-interpolation formula [93], which was proposed via polynomial reproduction with modifications at the endpoints, by Ling [68] in 2004 and [69] in 2005 respectively.

In this chapter, we provide a brief overview of multilevel quasi-interpolation and multilevel sparse kernal-based interpolation. Then, by using a multilevel version of the Q-SIK method, we will introduce quasi-multilevel sparse interpolation with kernels.

5.1 Multilevel Quasi interpolation

For the multivariate scattered data interpolation problem, one needs to find a continuous unknown function $u : \mathbb{R}^d \to \mathbb{R}$ such that $u(\mathbf{x}_i) = f(\mathbf{x}_i)$ for $i = 1, 2, \ldots, N$ by recovering an unknown function $f : \mathbb{R}^d \to \mathbb{R}$, which comes from $\{f(x_1), f(x_2), \ldots, f(x_N)\} \in \mathbb{R}$ sampled at a finite set $\mathcal{X} = \{x_1, x_2, \ldots, x_N\} \subset \mathbb{R}^d$ of pairwise distinct data sites. Multilevel approximation method brings leads to successful results for this kind of problem, especially in large N and with disorganized, distributed points.

In order to construct multilevel quasi-interpolation, we firstly need to de-

scribe a nested sequence of sets, that is

$$\mathcal{X}_1 \subset \mathcal{X}_2 \subset \cdots \mathcal{X}_K = \mathcal{X} \subset \mathbb{R}^d, \tag{5.1}$$

where $\mathcal{X}$ are pairwise distinct data sites of progressively greater data density. The principal notion of multilevel interpolation is to interpolate data sets at the coarsest level and then interpolate the residuals to progressively finer levels using properly adjusted basis functions. Although this method is usually used with radial basis function interpolation, we will describe it here in the content of quasi-interpolation. Namely we will use the multilevel algorithm into Q-SIK method since it does not converge which is given in previous chapter.

We can borrow the idea of formulating multilevel quasi-interpolation in [40],[61] and [80]. The letter l will denote the "level index", where $l = 1, 2, \ldots, L$, throughout this and the following sections. Set

$$\begin{cases} u_0 \equiv 0, \\ u_l = u_{l-1} + \delta_l f_{l-1}, \quad l = 1, 2, \ldots, L, \end{cases}$$

where u_l denotes the multilevel quasi-interpolant of the point set $\mathcal{X}_l$ and $\delta_l f_{l-1}$ denotes the quasi-interpolant to the residual $f - u_{l-1}$ at the l-th level. Note that u_l for each l matches f on the subset $\mathcal{X}_l$, i.e.,

$$u_l \mid_{\mathcal{X}_l} = f \mid_{\mathcal{X}_l} \quad \text{for all} \quad l = 1, 2, \ldots, L. \tag{5.2}$$

In other words, first of all one needs to compute the quasi-interpolant on the

data sets for the first level. Then the residual on all data sets needs to be calculated. Finally, quasi-interpolation of the residual is computed in order to recover the corresponding level of the quasi-interpolant.

Since the hierarchy (5.1) of the given sets plays a significant role in multilevel interpolation, the algorithm for this hierarchy has been presented in [62]. Adaptively constructed hierarchical sets have been given in [61]. In addition to these, Ling in [69] has showed that the multilevel scheme also conserves convexity and monotonicity providing the node placings are good enough in the content of multilevel quasi-interpolation using the dimension-splitting MQ (DSMQ) basis.

5.2 Quasi multilevel sparse interpolation with kernels

In this section, we will introduce the quasi-multilevel sparse interpolation with kernels (Q-MuSIK, for short) method by combining a multilevel algorithm and sparse interpolation with kernels. Indeed, the main motivation is the same as for MuSIK, as discussed above; the only difference is that we will use the Q-SIK method instead of the SIK method for each subgrid interpolation problem.

In contrast to the single level quasi-sparse interpolation with kernels method discussed in the previous chapter, we will now use the nested sparse grids which sort from the lower level to the higher level. In other words, sparse

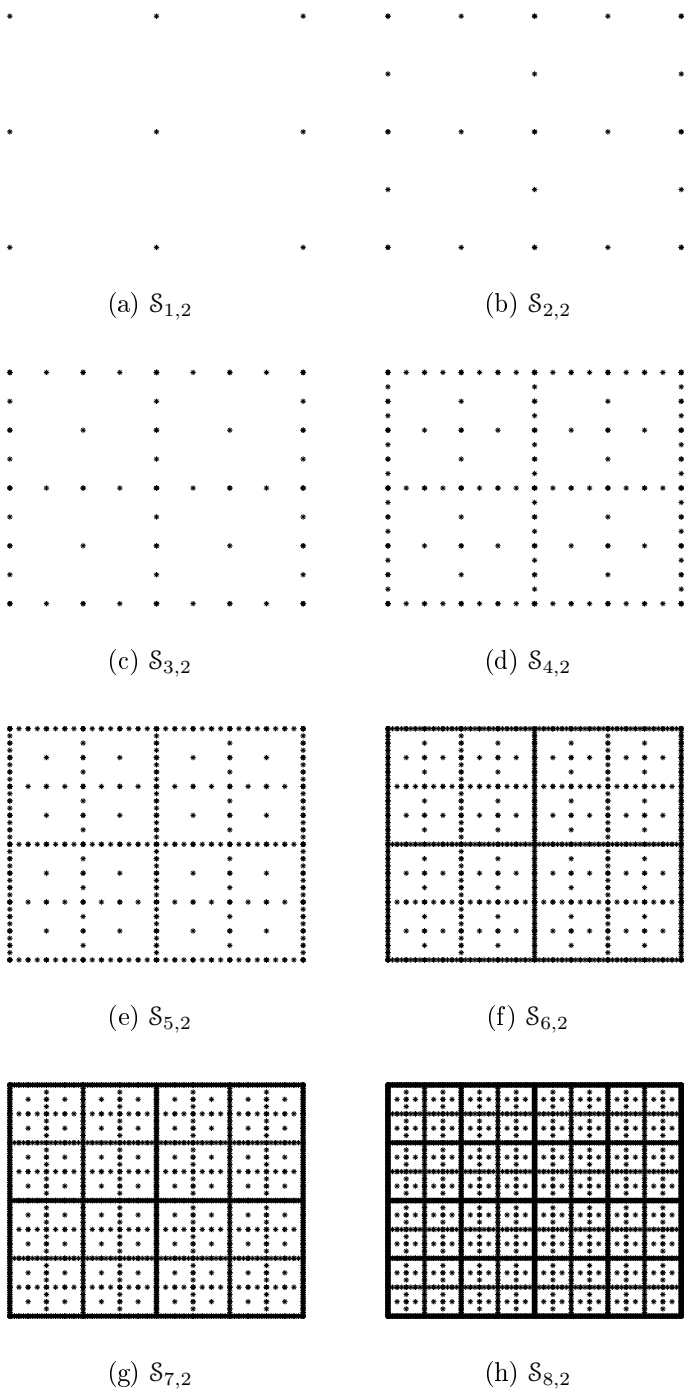
grids with increasingly greater data densities provide us with a hierarchical decomposition of the sparse grid data, which is

$$\mathcal{S}_{1,d} \subset \mathcal{S}_{2,d} \subset \mathcal{S}_{3,d} \subset \mathcal{S}_{4,d} \subset \mathcal{S}_{5,d} \subset \mathcal{S}_{6,d} \subset \mathcal{S}_{7,d} \subset \mathcal{S}_{8,d},$$

where d is the dimension. It must be noted that although the data density increases from level 1 to level n, the value of h, which is mesh size in quasi-interpolation, decreases. Thus, one expects to increase interpolation accuracy with the nested-ness of the sparse grid. The hierarchical decomposition in two dimensions can be seen in Figure 5.1.

After obtaining the sparse grid decomposition, the sparse grid interpolation Q_{n_0} needs to be evaluated for level 1. In other words, the coarsest level interpolation is set as a first residual, that is $\Delta_0 = Q_{n_0}$. Then Δ_j needs to be calculated for each level by interpolating the residual $f - Q_{n_{j-1}}$ on $\mathcal{S}_{j,d}$, for $2 \leqslant j \leqslant n$. Here, the level j residual comes from the level $j-1$ interpolant on the level j sparse grid. Finally, the quasi-multilevel sparse interpolation with kernel in level n is found as $Q_n := Q_{n-1} + \Delta_{n_0}$ where $Q_0 = 0$.

An algorithm for Q-MuSIK is as follows:



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

Figure 5.1: Sparse grid decomposition, $\mathcal{S}_{1,2} \subset \mathcal{S}_{2,2} \subset \mathcal{S}_{3,2} \subset \mathcal{S}_{4,2} \subset \mathcal{S}_{5,2} \subset \mathcal{S}_{6,2} \subset \mathcal{S}_{7,2} \subset \mathcal{S}_{8,2}$.

Algorithm 2: Q-MuSIK method

Data: Sparse grid data decomposition

Result: Interpolation value

1. Initialize the first interpolation value at zero, that is $Q_0(x) = 0$.
2. Construct the nested sparse grids as $\mathcal{S}_{1,d} \subset \mathcal{S}_{2,d} \subset \cdots \subset \mathcal{S}_{n,d}$.
3. For every value of $j = \{1, 2, ..., n\}$,
   3.a. Solve $\Delta_j(x) = f(x) - Q_{n_{j-1}}(x)$ on $\mathcal{S}_{j,d}$
   3.b. Update $Q_j(x) = Q_{j-1}(x) + \Delta_j(x)$.

Because of the nature of the Q-SIK method for each single level, the Q-MuSIK method has linear complexity, which means it needs a lower amount of memory and run time. In addition to this, both MuSIK and Q-MuSIK techniques are amenable to parallel programming because of combination technique. Thus these methods provide us with a more effective implementation, especially in high dimensions, when using the current generation of computing systems.

5.3 Numerical Experiments in 2-D

In this section, we will perform some numerical experiments in 2-D in order to verify the theoretical basis of our technique and demonstrate its advantages with applications. In order to make an objective comparison, we will use the same test functions, $F_1^{2d}(x, y)$,$F_2^{2d}(x, y)$, $F_3^{2d}(x, y)$, $F_4^{2d}(x, y)$, $F_5^{2d}(x, y)$ and $F_6^{2d}(x, y)$, which were presented in the previous chapter.

In addition to these, similarly, N represents the number of data points for both standard quasi- and multilevel quasi-interpolation. In sparse interpolation with kernels, *SGnode* represents the number of nodes in the MuSIK and Q-MuSIK methods. Similarly, *DOFs(SG)* represents the total number of nodes visited in the MuSIK and Q-MuSIK methods. Of course, *DOFs(SG)* is much more than *SGnode* since some nodes are visited several times in every sub-interpolation problem. *Time* stands for the CPU time consumed per numerical experiment. All these numerical experiments have been done using a 160×160 uniform grid in the domain $[0,1]^2 \subset \mathbb{R}^2$. The root mean squared error has been calculated for quasi-interpolation, multilevel quasi-interpolation, SIK, Q-SIK, MuSIK and Q-MuSIK for comparison.

5.3.1 Experiments using Gaussian Tensor Products

We present numerical results which were computed with anisotropic Gaussian basis functions for both MuSIK and Q-MuSIK. Of course, because of the nature of the multilevel algorithm, Q-MuSIK needs more time in comparison with the Q-SIK method. However, this might be considered to be reasonable since the multilevel algorithm has linear complexity with time. This will be verified by the following experiments.

Numerical results from some of our Q-MuSIK numerical experiments have been presented in Table 5.1, Table 5.2 and Table 5.3 for $F_1^{2d}(x,y)$,$F_4^{2d}(x,y)$ and $F_6^{2d}(x,y)$ respectively. Of course, figures support the findings and give an opportunity to compare them.

SGnode	DOFs(SG)	Maximum Error	RMS Error
9	9	6.402506e-01	1.923844e-01
21	39	3.846635e-01	9.884292e-02
49	109	1.328952e-01	3.775068e-02
113	271	4.951348e-02	1.223503e-02
257	641	1.771163e-02	3.717801e-03
577	1475	7.715302e-03	1.256865e-03
1281	3333	2.573988e-03	3.750473e-04
2817	7431	6.926234e-04	1.016338e-04
6145	16393	1.973276e-04	2.645588e-05

Table 5.1: Q-MuSIK results using Gaussian basis functions with shape parameter $d = 0.4$, test function $F_1^{2d}(x, y)$, on an equally spaced 160×160 evaluation grid.

In Figure 5.2, RMS errors are plotted verses the number of nodes used for six different test functions. From the graph below we can see that the Q-MuSIK method provides us with a better approximation than the Q-SIK method. Of course, the multilevel algorithm plays a significant role in this situation. Both the Q-SIK and the Q-MuSIK methods present better performance than the direct multilevel quasi-interpolation.

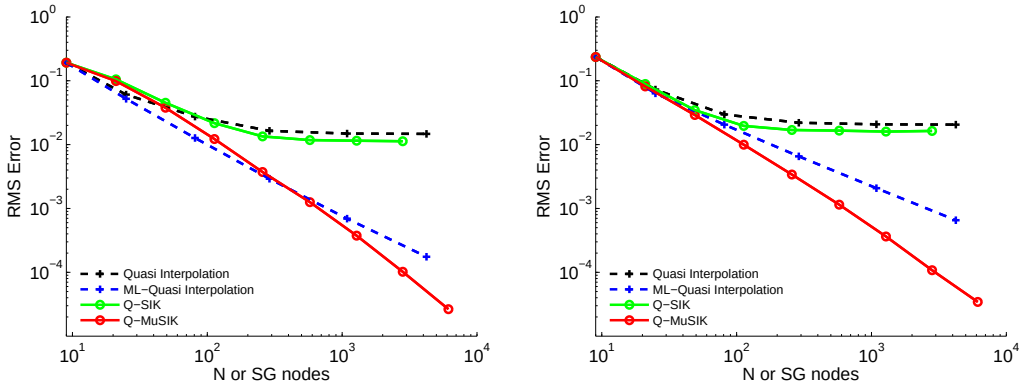
In addition to this, from Figure 5.3 we can see that the Q-MuSIK method is faster than multilevel quasi-interpolation. The main reason of this result is that Q-MuSIK technique uses a smaller number of points in comparison with the full grid interpolation. As a result this makes the Q-MuSIK method faster than the classical multilevel quasi interpolation.

Lastly, the differences between MuSIK and Q-MuSIK are compared in Figure 5.4. In this figure, RMS errors versus computation time are plotted for six test functions. It can be said that Q-MuSIK is superior when compared

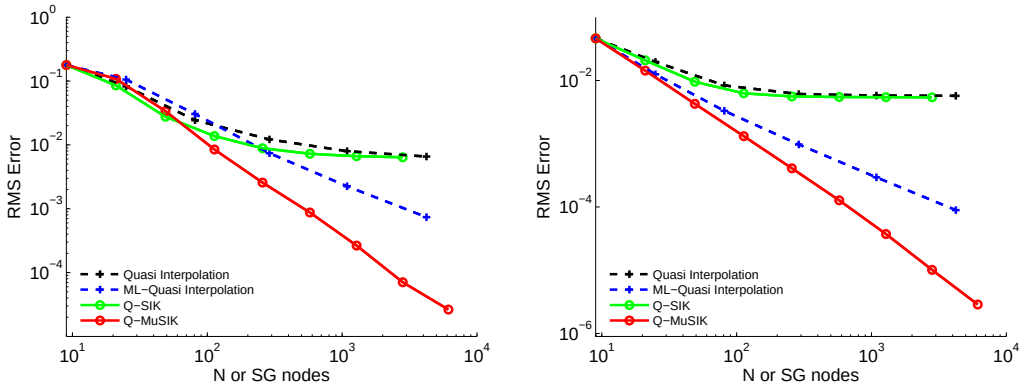
SGnode	DOFs(SG)	Maximum Error	RMS Error
9	9	1.476663e-01	4.633068e-02
21	39	4.366024e-02	1.432694e-02
49	109	1.605678e-02	4.281684e-03
113	271	7.655495e-03	1.314830e-03
257	641	3.261574e-03	4.088407e-04
577	1475	1.326272e-03	1.273487e-04
1281	3333	5.571766e-04	3.734527e-05
2817	7431	1.772635e-04	1.014380e-05
6145	16393	4.767200e-05	2.875345e-06

Table 5.2: Q-MuSIK results using Gaussian basis functions with shape parameter $d = 0.4$, test function $F_4^{2d}(x, y)$, on an equally spaced 160×160 evaluation grid.

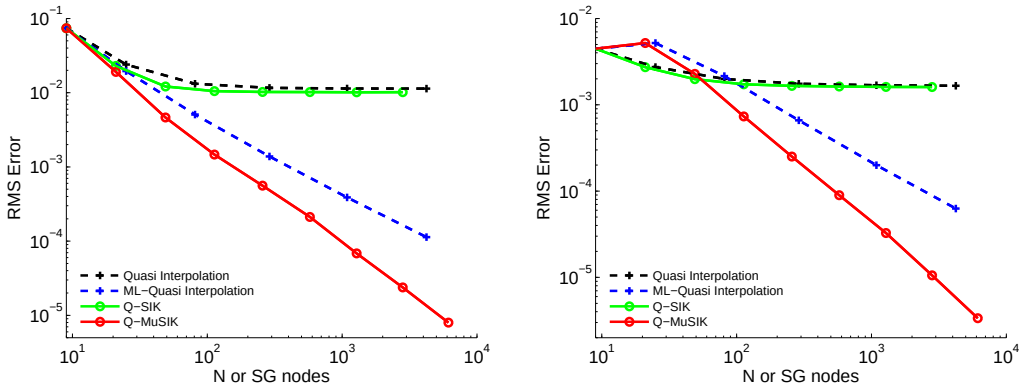
with the MuSIK method since Q-MuSIK is matrix free. As is known, the MuSIK method needs to solve a large algebraic system in order to compute interpolation coefficients. However, Q-MuSIK requires just the function value for interpolation. Although the MuSIK method convergence is better in comparison with Q-MuSIK method at high levels, the Q-MuSIK method is actually superior at low levels since it has the same convergence properties in less time.



(a) $F_1^{2d}(x, y)$ (b) $F_2^{2d}(x, y)$

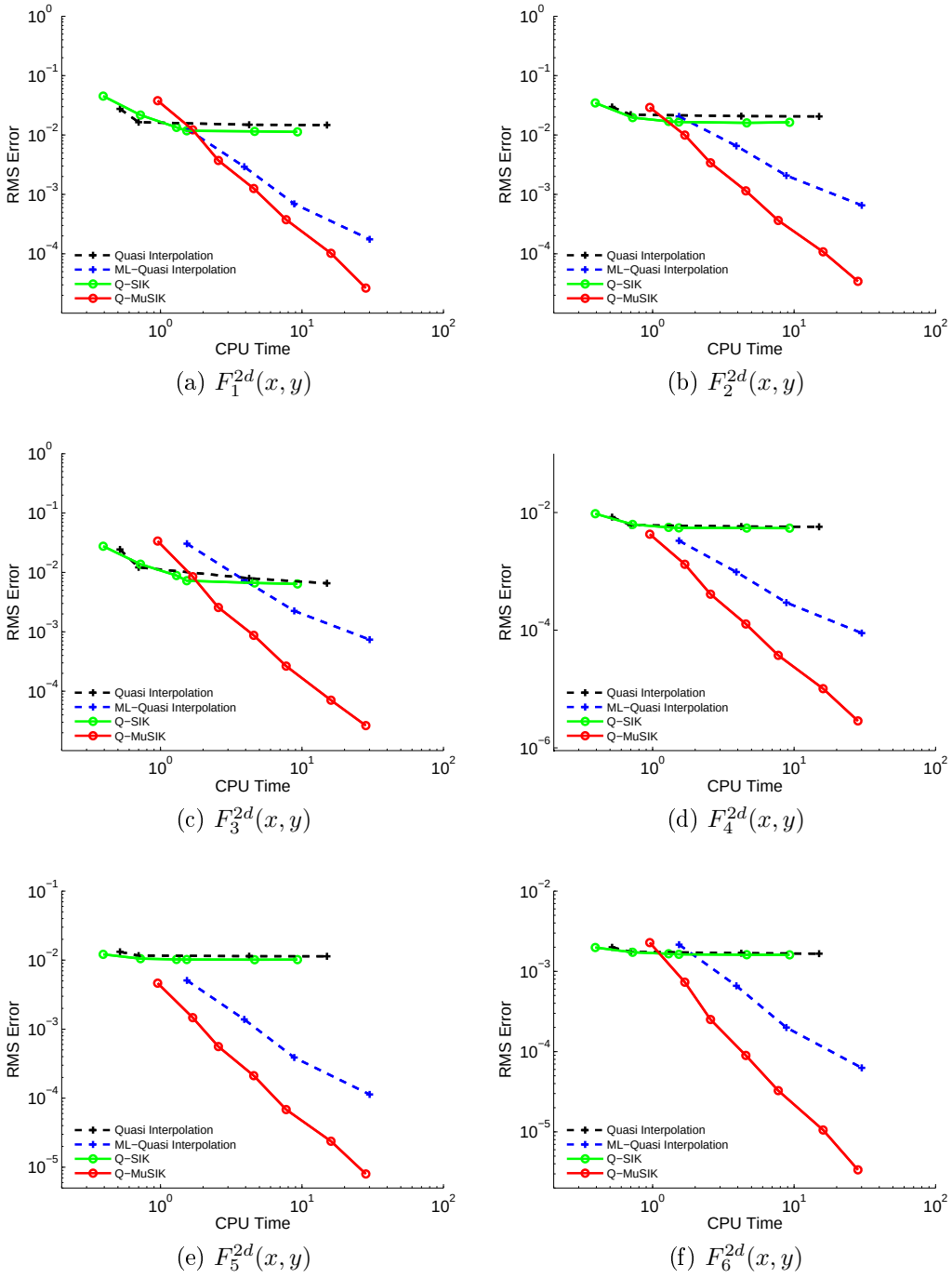


(c) $F_3^{2d}(x, y)$ (d) $F_4^{2d}(x, y)$



(e) $F_5^{2d}(x, y)$ (f) $F_6^{2d}(x, y)$

Figure 5.2: RMS error versus N(Quasi-interpolation, Multilevel Quasi-interpolation) or SG (Q-SIK, Q-MuSIK) nodes using Gaussian basis functions with $\rho = 0.4$: Quasi-interpolation (black), Q-SIK (green), Multilevel Quasi-interpolation (blue) and Q-MuSIK (red) on a 160×160 uniform grid.

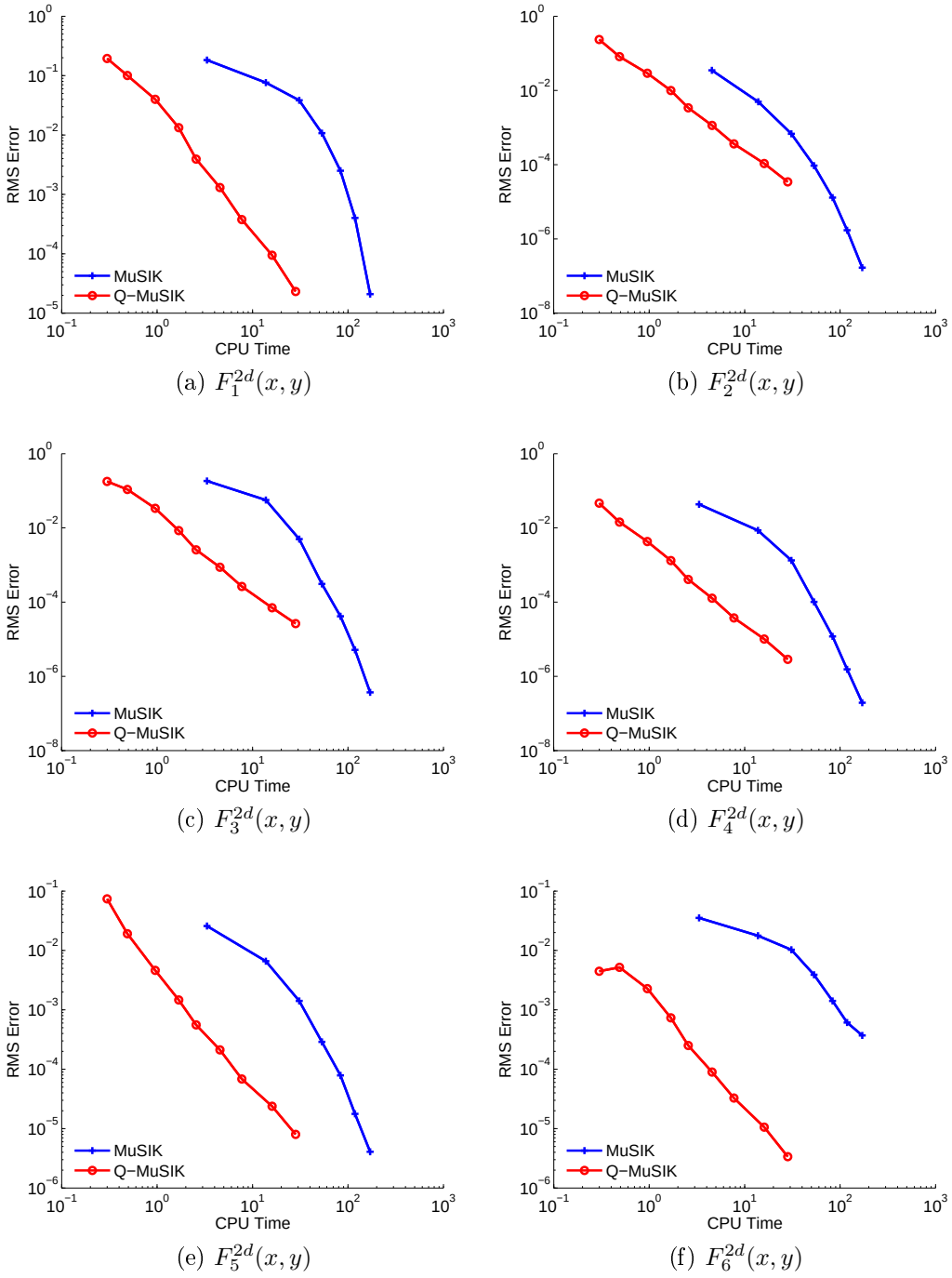


(a) $F_1^{2d}(x, y)$ (b) $F_2^{2d}(x, y)$

(c) $F_3^{2d}(x, y)$ (d) $F_4^{2d}(x, y)$

(e) $F_5^{2d}(x, y)$ (f) $F_6^{2d}(x, y)$

Figure 5.3: RMS error versus computational time using Gaussian basis functions with $\rho = 0.4$: Quasi-interpolation (black), Q-SIK (green), Multilevel Quasi-interpolation (blue) and Q-MuSIK (red) on a 160×160 uniform grid.



(a) $F_1^{2d}(x, y)$ (b) $F_2^{2d}(x, y)$

(c) $F_3^{2d}(x, y)$ (d) $F_4^{2d}(x, y)$

(e) $F_5^{2d}(x, y)$ (f) $F_6^{2d}(x, y)$

Figure 5.4: RMS error versus computational time using Gaussian basis functions with $\rho = 0.4$ and $c = 0.45$: MuSIK (blue) and Q-MuSIK (red) on a 160×160 uniform grid.

SGnode	DOFs(SG)	Maximum Error	RMS Error
9	9	5.105632e-02	4.468879e-03
21	39	3.063097e-02	5.204096e-03
49	109	1.936359e-02	2.278934e-03
113	271	9.304990e-03	7.331817e-04
257	641	4.261802e-03	2.510321e-04
577	1475	1.950718e-03	8.952410e-05
1281	3333	5.952518e-04	3.265428e-05
2817	7431	1.311157e-04	1.056364e-05
6145	16393	5.357728e-05	3.361347e-06

Table 5.3: Q-MuSIK results using Gaussian basis functions with shape parameter $d = 0.4$, test function $F_6^{2d}(x, y)$, on an equally spaced 160×160 evaluation grid.

5.4 Numerical Experiments in 3-D

In this section, we present some results which were performed with 3-D quasi-interpolation, multilevel quasi-interpolation, SIK, Q-SIK, MuSIK and Q-MuSIK. Namely, we have considered the following 3-D extension of the test functions used in the previous chapter.

- $F_1^{3d}(x, y, z)$ (Franke's test function)

$$\begin{aligned} F_1^{3d}(x,y,z) &= \frac{3}{4}e^{-((9x-2)^2+(9y-2)^2+(9z-2)^2)/4} + \frac{3}{4}e^{-(9x+1)^2/49-(9y+1)^2/10-(9z+1)^2/10} \\ &+ \frac{1}{2}e^{-((9x-7)^2+(9y-3)^2+(9z-5)^2)/4} - \frac{1}{5}e^{-((9x-4)^2+(9y-7)^2+(9z-5)^2)}. \end{aligned}$$

- $F_2^{3d}(x, y, z) = 4^3 x(1-x)y(1-y)z(1-z)$.
- $F_3^{3d}(x, y, z) = \sqrt{\frac{18}{\pi}} e^{-(x^2+81y^2+z^2)}$.

- $F_4^{3d}(x,y,z) = \cos(6z)\dfrac{1.25+\cos(5.4y)}{6+6(3x-1)^2}$.
- $F_5^{3d}(x,y,z) = \sqrt{64-81((x-0.5)^2+(y-0.5)^2+(z-0.5)^2)}/9 - 0.5$.
- $F_6^{3d}(x,y,z) = \max(x-\frac{1}{2})\max(y-\frac{1}{2})\max(z-\frac{1}{2})$.

These functions have been considered in a $[0,1]^3 \subset \mathbb{R}^3$. The 3-D equivalence of equation (4.4) is

$$I_n(\mathbf{x}) = \sum_{|\mathbf{l}|_1=n+2} I_u^{\mathbf{Ani.}}(\mathbf{x}) - 2\sum_{|\mathbf{l}|_1=n+1} I_u^{\mathbf{Ani.}}(\mathbf{x}) + \sum_{|\mathbf{l}|_1=n} I_u^{\mathbf{Ani.}}(\mathbf{x}). \tag{5.3}$$

We have performed our 3-D experiments on an equally spaced 125000 evaluation grid. The results of these experiments can be found in Tables 5.4 and Table 5.5.

SGnode	DOFs(SG)	Maximum Error	RMS Error
27	27	7.261597e-01	1.058522e-01
81	162	5.955357e-01	7.939472e-02
225	630	3.459408e-01	4.768733e-02
593	1997	1.767444e-01	2.067385e-02
1505	5687	9.231797e-02	8.454461e-03
3713	15188	4.212802e-02	3.357306e-03
8961	38868	1.367273e-02	1.277559e-03

Table 5.4: Q-MuSIK results using Gaussian basis functions with shape parameter $d = 0.4$, test function $F_1^{3d}(x,y,z)$, on an equally spaced $50\times50\times50$ evaluation grid.

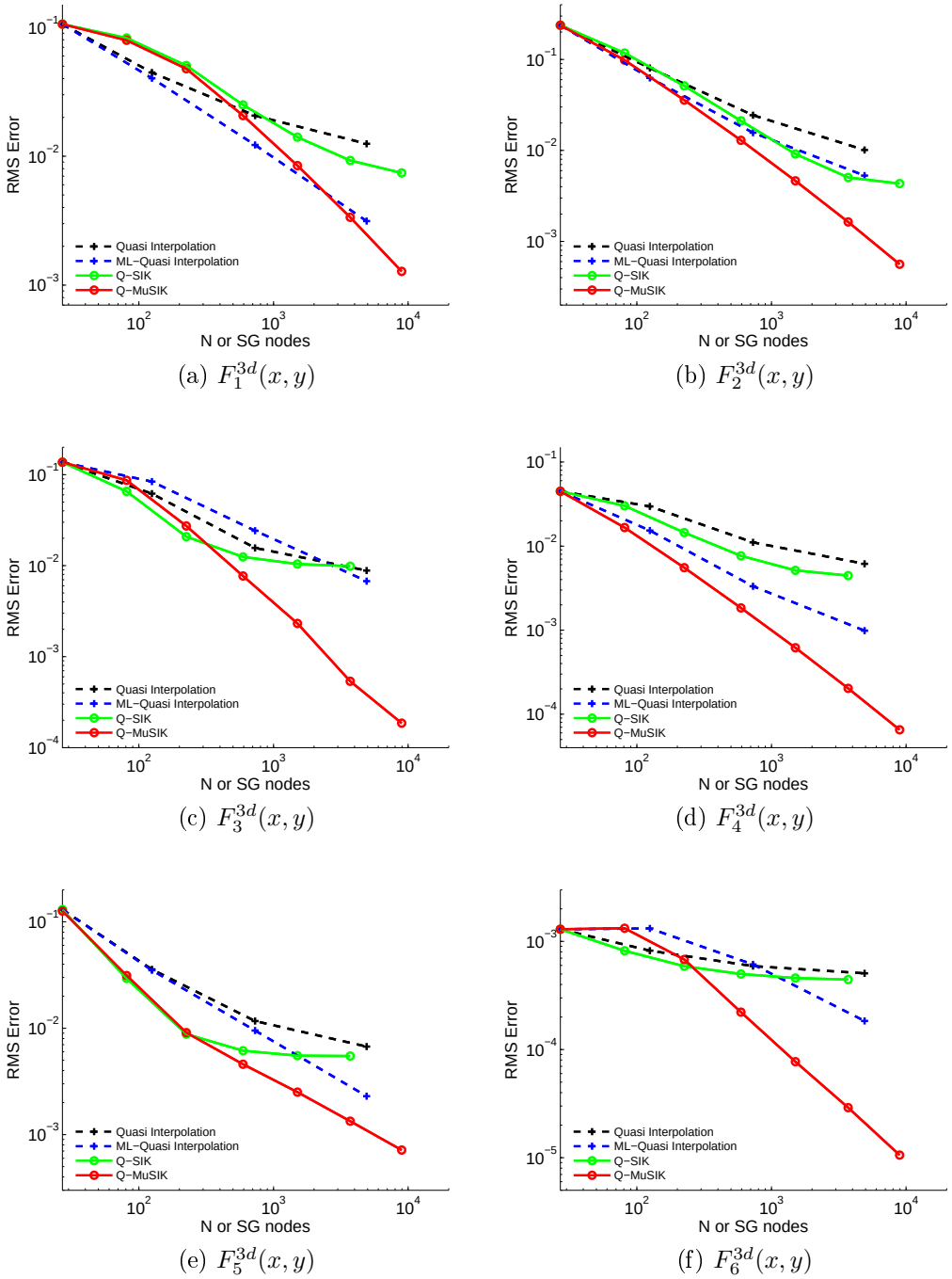
Similarly, Figure 5.5 shows that the Q-MuSIK method convergence is better than both the Q-SIK method and multilevel quasi-interpoaltion in 3-D.

SGnode	DOFs(SG)	Maximum Error	RMS Error
27	27	2.087103e-01	4.485704e-02
81	162	8.841902e-02	1.660209e-02
225	630	3.112325e-02	5.537253e-03
593	1997	1.070217e-02	1.836291e-03
1505	5687	5.403194e-03	6.165266e-04
3713	15188	2.104470e-03	2.026695e-04
8961	38868	8.392200e-04	6.528286e-05

Table 5.5: Q-MuSIK results using Gaussian basis functions with shape parameter $d = 0.4$, test function $F_4^{3d}(x, y, z)$, on an equally spaced $50 \times 50 \times 50$ evaluation grid.

In the numerical experiments, we find that the computational time of Q-SIK is less than the time required by classical quasi-interpolation, especially when the data density is large. An analogous relationship between evaluation time and the data density has been showed for Q-MuSIK and multilevel quasi-interpolation, which can be seen in Figure 5.6. For these reasons, Q-SIK and Q-MuSIK gain an advantage over classical quasi-interpolation and multilevel quasi-interpolation, respectively, in terms of computational time.

Compared with the MuSIK method, Q-MuSIK provides us with better results in terms of reduced computational time because of the reasons discussed in previous sections. According to Figure 5.7, Q-MuSIK presents better performance than MuSIK, particularly at low levels. Hence, it can be said that Q-MuSIK is a good alternative for problems which require a low level of interpolation.

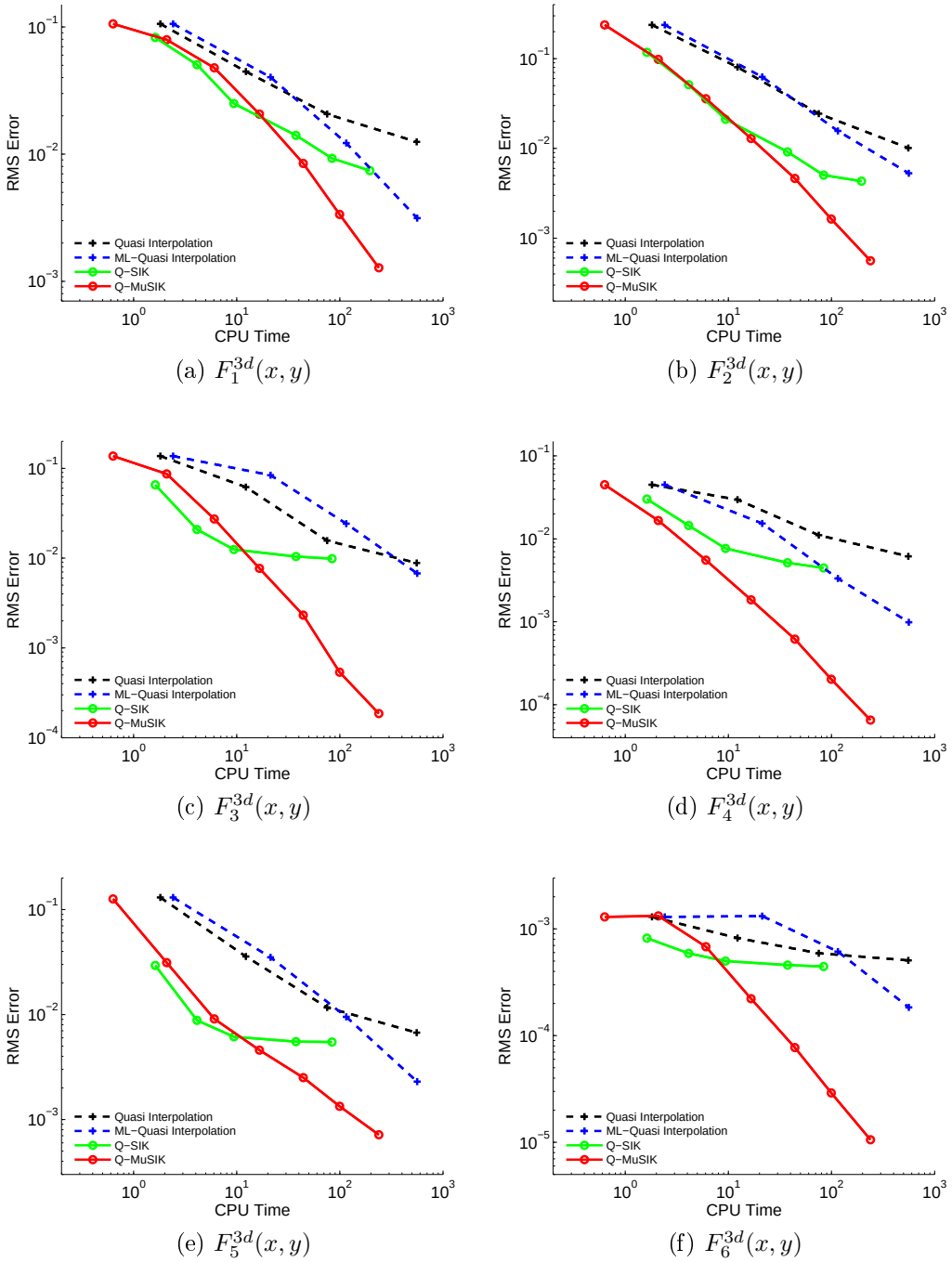


(a) $F_1^{3d}(x, y)$ (b) $F_2^{3d}(x, y)$

(c) $F_3^{3d}(x, y)$ (d) $F_4^{3d}(x, y)$

(e) $F_5^{3d}(x, y)$ (f) $F_6^{3d}(x, y)$

Figure 5.5: RMS error versus N(Quasi-interpolation, Multilevel Quasi-interpolation) or SG (Q-SIK, Q-MuSIK) using Gaussian basis functions with $\rho = 0.4$: Quasi-interpolation (black), Q-SIK (green), Multilevel Quasi-interpolation (blue) and Q-MuSIK (red) on a $50 \times 50 \times 50$ uniform grid.

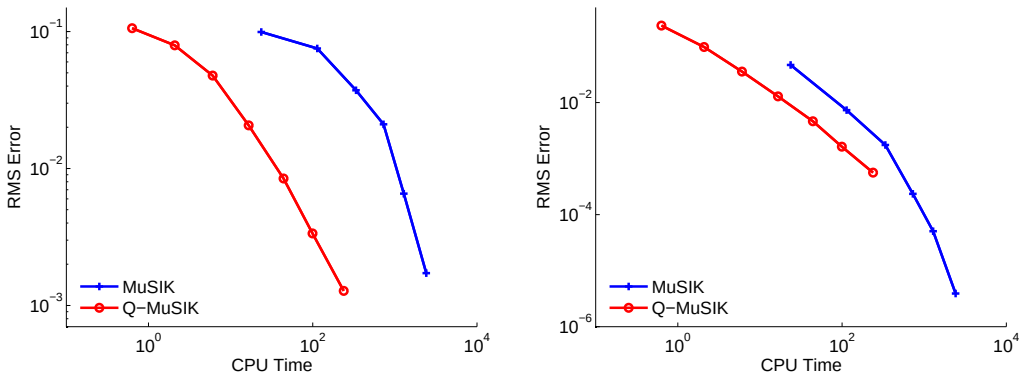


(a) $F_1^{3d}(x, y)$ (b) $F_2^{3d}(x, y)$

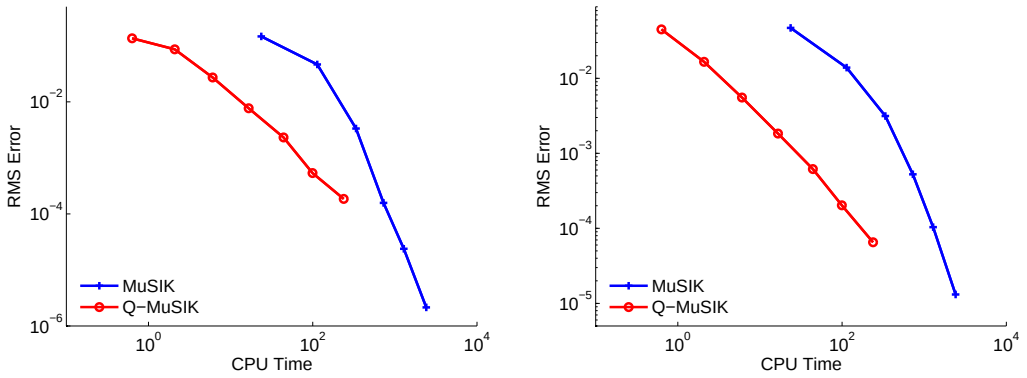
(c) $F_3^{3d}(x, y)$ (d) $F_4^{3d}(x, y)$

(e) $F_5^{3d}(x, y)$ (f) $F_6^{3d}(x, y)$

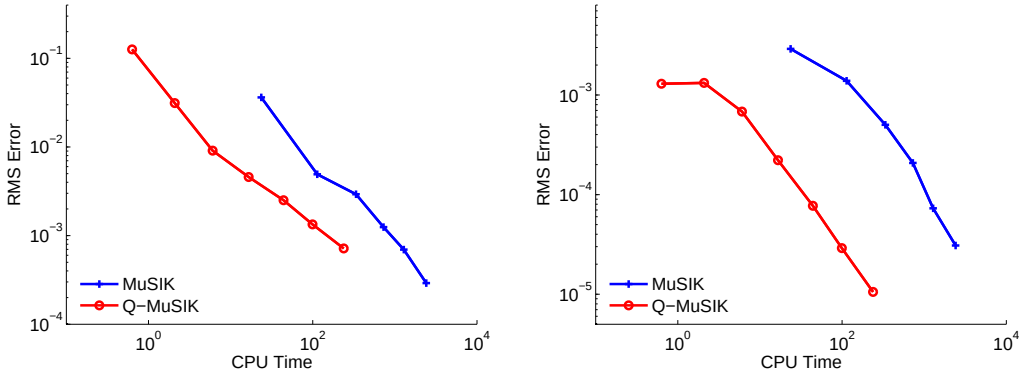
Figure 5.6: RMS error versus computational time using Gaussian basis functions with $\rho = 0.4$: Quasi-interpolation (black), Q-SIK (green), Multilevel Quasi-interpolation (blue) and Q-MuSIK (red) on a $50 \times 50 \times 50$ uniform grid.



(a) $F_1^{3d}(x, y)$ (b) $F_2^{3d}(x, y)$



(c) $F_3^{3d}(x, y)$ (d) $F_4^{3d}(x, y)$



(e) $F_5^{3d}(x, y)$ (f) $F_6^{3d}(x, y)$

Figure 5.7: RMS error versus computational time using Gaussian basis functions with $\rho = 0.4$ and $c = 0.45$: MuSIK (blue) and Q-MuSIK (red) on a $50 \times 50 \times 50$ uniform grid.

5.5 Numerical Experiments in 4-D

One of the most difficult problems in the field of approximation is approximation functions, or data in high dimensions. For example, classical full grid interpolations are limited to about 10000 grid points. For this reason, we need developed interpolation algorithms which require less data density. In this section, we present some results which were performed with 4-D quasi-interpolation, multilevel quasi-interpolation, SIK, Q-SIK, MuSIK and Q-MuSIK. The 4-D equivalence of equation (4.6) is

$$I_n(\mathbf{x}) = \sum_{|\mathbf{l}|_1=n+3} I_u^{\mathbf{Ani.}}(\mathbf{x}) - 3 \sum_{|\mathbf{l}|_1=n+2} I_u^{\mathbf{Ani.}}(\mathbf{x}) + 3 \sum_{|\mathbf{l}|_1=n+1} I_u^{\mathbf{Ani.}}(\mathbf{x}) + \sum_{|\mathbf{l}|_1=n} I_u^{\mathbf{Ani.}}(\mathbf{x}).$$

Namely, we have considered the following 4-D extension of the test functions used in previous chapters.

- $F_1^{4d}(x,y,z,t)$ (Franke's test function)

$$\begin{aligned} F_1^{4d}(x,y,z,t) &= \frac{3}{4} e^{(-(9x_1-2)^2-(9x_2-2)^2-(9x_3-2)^2)/4-(9x_4-2)^2)/8} \\ &+ \frac{3}{4} e^{(-(9x_1+1)^2)/49-((9x_2+1)^2)/10-((9x_3+1)^2)/29-((9x_4+1)^2)/39} \\ &+ \frac{1}{2} e^{(-(9x_1-7)^2)/4-(9x_2-3)^2-((9x_3-5)^2)/2-((9x_4-5)^2)/4} \\ &- \frac{1}{5} e^{(-(9x_1-4)^2)/4-(9x_2-7)^2-((9x_3-5)^2)-((9x_4-5)^2)}. \end{aligned}$$

- $F_2^{4d}(x,y,z,t) = 4^4 x(1-x)y(1-y)z(1-z)t(1-t)$.
- $F_3^{4d}(x,y,z,t) = \sqrt{\frac{18}{\pi}} e^{-(x^2+81y^2+z^2+81t^2)}$.

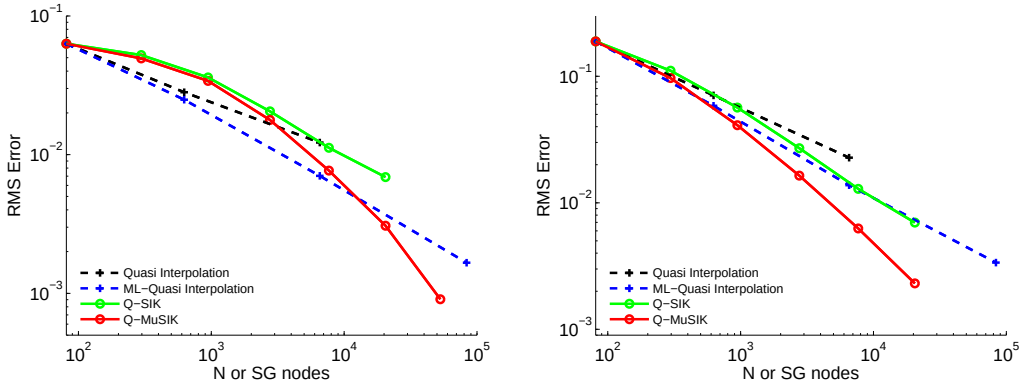
- $F_4^{4d}(x,y,z,t) = \cos(6z)\cos(6t)\dfrac{1.25+\cos(5.4y)}{6+6(3x-1)^2}$.
- $F_5^{4d}(x,y,z,t) = \sqrt{64-81((x-0.5)^2+(y-0.5)^2+(z-0.5)^2+(t-0.5)^2)}/9-0.5$.
- $F_6^{4d}(x,y,z,t) = \max(x-\frac{1}{2})\max(y-\frac{1}{2})\max(z-\frac{1}{2})\max(t-\frac{1}{2})$.

Table 5.6 shows some numerical results which were computed with the Q-SIK algorithm on an equally spaced 194481 evaluation grid. According to these results, Q-SIK and Q-MuSIK algorithms have been successfully implemented with the 4-D interpolation problems.

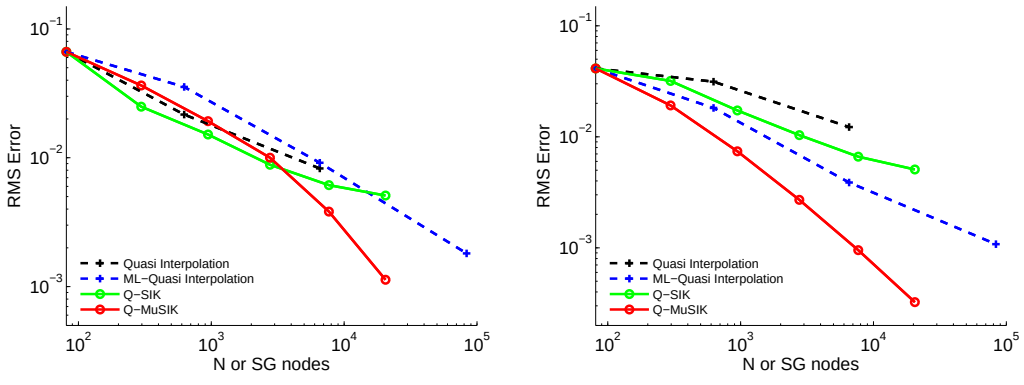
SGnode	DOFs(SG)	Maximum Error	RMS Error
81	81	7.235155e-01	6.302388e-02
297	621	6.550276e-01	5.220851e-02
945	2943	5.158184e-01	3.606199e-02
2769	11139	2.886416e-01	2.052927e-02
7681	36901	1.988700e-01	1.121024e-02
20481	112111	1.210933e-01	6.894223e-03

Table 5.6: Q-SIK results using Gaussian basis functions with shape parameter $d = 0.4$, test function $F_1^{4d}(x,y,z,t)$, on an equally spaced $21\times21\times21\times21$ evaluation grid.

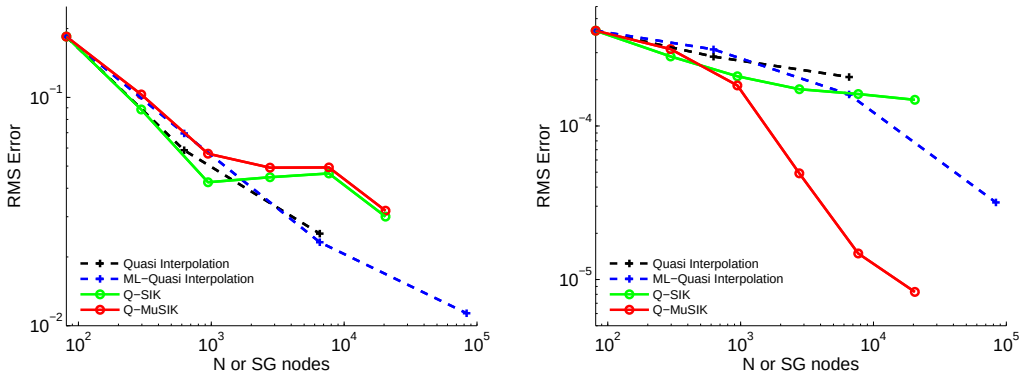
The numerical results obtained from our experiments are plotted in Figure 5.8. We confirm the same superiority of the Q-SIK and Q-MuSIK algorithms over classical quasi-interpolation and its multilevel variant with regards to complexity, convergence and computational time as already been seen in the case of lower dimensions. In other words, Q-MuSIK accelerates the convergence of Q-SIK.



(a) $F_1^{4d}(x, y)$ (b) $F_2^{4d}(x, y)$



(c) $F_3^{4d}(x, y)$ (d) $F_4^{4d}(x, y)$



(e) $F_5^{4d}(x, y)$ (f) $F_6^{4d}(x, y)$

Figure 5.8: RMS error versus N(Quasi-interpolation, Multilevel Quasi-interpolation) or SG (Q-SIK, Q-MuSIK) using Gaussian basis functions with $\rho = 0.4$: Quasi-interpolation (black), Q-SIK (green), Multilevel Quasi-interpolation (blue) and Q-MuSIK (red) on a $21 \times 21 \times 21 \times 21$ uniform grid.

5.6 Discussion

In this chapter, we have proposed the quasi-multilevel sparse interpolation with kernels and presented numerical experiments results. These results have verified the theoretical basis of our proposed method. According to the numerical results, we have observed that Q-MuSIK performs considerably better than classical multilevel quasi-interpolation with regards to computational time and convergence distinguishably.

Another advantage of the Q-MuSIK algorithm is that it has linear computational time and complexity since we have applied the Q-SIK algorithm for each step. For this reason, computational time requirements for the Q-MuSIK algorithm are remarkably low in comparison with the classical full grid multilevel interpolation. Q-MuSIK is superior in computational time and complexity to the classical multilevel quasi-interpolation not only in two dimensional problems, but also in high-dimensional interpolation problems, since it has almost one-dimensional complexity.

Furthermore, the Q-MuSIK algorithm is highly amenable to deployment in a parallel environment due to the inhernetly parallelized nature of the Q-SIK algorithm. Thus Q-MuSIK algorithm provides us with much better results in this regard. So, Q-MuSIK might be a good alternative technique for problems with large data density, especially when low computational time is particularly desirable.

We have also compared the Q-MuSIK method with MuSIK in terms of computational time. As might be expected, due to the nature of quasi-

interpolation, we expect that, of the two, the Q-MuSIK algorithm will have superior performance with regards to computational time. This expectation has been proved with our numerical experiments. In particular, Q-MuSIK could be an alternative technique for interpolation problems which require low level interpolation.

Chapter 6

Error estimation of Multilevel Quasi-Interpolation for Periodic Functions

In this chapter, we consider convergence rates for multilevel quasi-interpolation of periodic functions which can be represented as a combination of exponential functions using Gaussian basis functions on a grid. To date, unfortunately, studies of convergence rates for the multilevel approximation are limited because of difficulties in the multilevel algorithm, such as a changing approximation space from one level to the next.

In order to find a proper error bound for the multilevel approximation, in [78], Narcowich, Schaback and Ward provided a theoretical foundation in 1999. In this study, the authors found that the multilevel algorithm, which

is constructed by adding additional boundary conditions, convergences at least linearly. Then, in 2002, Hales and Levesley [54] showed at least linear convergence for the multilevel approximation for globally supported polyharmonic splines, such as radial powers and thin plate splines. In this study, Hales and Levesley present the linear convergence as a progress of the form

$$\|e_k\|_\infty = C\|e_{k-1}\|_\infty, \qquad k = 1, 2, \ldots, N. \tag{6.1}$$

where e_k denotes the error at level k and $C < 1$ is a level independent positive constant.

As mentioned above, the main problem facing convergence rate studies of the multilevel approximation is that there are different approximation spaces for each level. In other words, since the approximation spaces are not nested, the native space norms for each level are different from each other. In order to overcome this problem, Hales and Levesley proposed scaling the uniformly spaced grids instead of the basis function. Similarly, in [91], Wendland proved the same convergence rate for thin plate splines.

6.1 Error Bounds of Quasi-interpolation for Periodic Functions

The general quasi-interpolation formula given in Chapter 2 is of the form

$$Q_h f(x) = \sum_{k\in\mathbb{Z}} f(kh)\mu\left(\frac{x}{h} - k\right). \tag{6.2}$$

where $\mu(x) = \frac{1}{\sqrt{2\pi}}\exp(\frac{-x^2}{2})$, is a Gaussian basis function. In this context, we wish to analyse the multilevel algorithm for approximating a periodic function f with Fourier series

$$f(x) = \sum_{m\in\mathbb{Z}} a_m e^{2\pi imx}. \tag{6.3}$$

Although our attention will later be focused on these specialised cases, it is helpful to explain the multilevel algorithm in the general case. The single level error is defined by

$$r_0(x) = f(x) - Q_{2^0} f(x), \tag{6.4}$$

and

$$r_m(x) = r_{m-1}(x) - Q_{2^{-m}} r_{m-1}(x). \tag{6.5}$$

The above recursive summation gives us the approximations

$$S_m(x) = Q_{2^0} f(x) + \sum_{j=0}^{m-1} r_j(x). \tag{6.6}$$

In order to find an error bound for quasi-interpolation for periodic functions, we need to interpolate exponential function. $e_m(x)$ will denote the exponential function $\exp(2\pi imx)$ throughout this and the following sections. So, the quasi-interpolation of $e_m(x)$ for $n = 1, 2, 3, \cdots$ is

$$\begin{aligned} Q_{1/n}e_m(x) &= \sum_{k\in\mathbb{Z}} e_m(k/n)\mu(nx-k) \\ &= \sum_{j=0}^{n-1}\sum_{k\in\mathbb{Z}} e_m((nk+j)/n)\mu(nx-(nk-j)) \\ &= \sum_{j=0}^{n-1} e_m(j/n)\sum_{k\in\mathbb{Z}}\mu(n(x-k)-j) \\ &= \sum_{j=0}^{n-1} e_m(j/n)\sum_{l\in\mathbb{Z}} a_l e_l(x), \end{aligned} \tag{6.7}$$

where

$$\begin{aligned} a_l &= \int_{x\in[0,1]}\left[\sum_{k\in\mathbb{Z}}\mu(n(x-k)-j)\right]e_{-l}(x)dx \\ &= \frac{1}{n}\int_{x\in\mathbb{R}}\mu(x-j/n)e_{-l}(x/n)dx \\ &= \frac{1}{n}\int_{x\in\mathbb{R}}\mu(x)e_{-l}((x+j)/n)dx \\ &= \frac{1}{n}e_{-l}(j/n)\int_{x\in\mathbb{R}}\mu(x)e_{-l}(x/n)dx \\ &= \frac{1}{n}e_{-l}(j/n)\hat{\mu}(-l/n). \end{aligned}$$

Here $\hat{\mu}$ is the Fourier transform of the Gaussian

$$\begin{aligned} \hat{\mu}(x) &= \int_{x\in\mathbb{R}}\mu(y)e_x(y)dy \\ &= \exp(-2\pi^2x^2); \end{aligned}$$

see [73]. By substituting into (6.7), we obtain that

$$\begin{aligned} Q_{1/n}e_m(x) &= \sum_{j=0}^{n-1} e_m(j/n) \sum_{l\in\mathbb{Z}} \frac{1}{n} e_{-l}(j/n)\hat{\mu}(-l/n)e_l(x) \\ &= \sum_{l\in\mathbb{Z}} \hat{\mu}(-l/n)e_l(x) \left[\frac{1}{n}\sum_{j=0}^{n-1} e_{m-l}(j/n)\right] \\ &= \sum_{l\in\mathbb{Z}} \hat{\mu}(-l-m/n)e_{m+nl}(x), \end{aligned} \tag{6.8}$$

from the discrete orthogonality property of the complex exponentials. Then, using the above result, we have

$$\begin{aligned} f(x) - Q_{1/n}f(x) &= \sum_{m\in\mathbb{Z}} a_m e_m(x) - Q_{1/n}\left[\sum_{m\in\mathbb{Z}} a_m e_m(x)\right] \\ &= \sum_{m\in\mathbb{Z}} a_m e_m(x) - \sum_{m\in\mathbb{Z}} a_m Q_{1/n} e_m(x) \\ &= \sum_{m\in\mathbb{Z}} a_m e_m(x) - \sum_{m\in\mathbb{Z}} a_m \left[\sum_{l\in\mathbb{Z}} \hat{\mu}(-l-m/n)e_{m+nl}(x)\right] \\ &= \sum_{m\in\mathbb{Z}} a_m e_m(x) - \sum_{m\in\mathbb{Z}} \hat{\mu}(m/n) \sum_{l\in\mathbb{Z}} a_{m-nl} e_m(x) \\ &= \sum_{m\in\mathbb{Z}} a_m e_m(x) - \sum_{m\in\mathbb{Z}} a_m \hat{\mu}(m/n) e_m(x) \\ &+ \sum_{m\in\mathbb{Z}} a_m \hat{\mu}(m/n) e_m(x) - \sum_{m\in\mathbb{Z}} \hat{\mu}(m/n) \sum_{l\in\mathbb{Z}} a_{m-nl} e_m(x) \\ &= \sum_{m\in\mathbb{Z}} a_m (1-\hat{\mu}(m/n)) e_m(x) + \sum_{m\in\mathbb{Z}} \hat{\mu}(m/n) \sum_{l\neq 0} a_{m+nl} e_m(x) \\ &=: E_n f(x) + G_n(x). \end{aligned}$$

Lemma 6.1. Let

$$f(x) = \sum_{m\in\mathbb{Z}} a_m e^{2\pi i m x}.$$

be a 1-periodic function and let

$$Q_{1/n}f(x) = \sum_{k\in\mathbb{Z}} f(k/n)\mu(nx-k)$$

be the quasi-interpolation approximation to f on the spaced grid with mesh size $1/n$. Then

$$f(x) - Q_{1/n}f(x) = E_n f(x) + G_n(x),$$

where

$$\begin{aligned} E_n f(x) &= \sum_{m\in\mathbb{Z}} a_m(1-\hat{\mu}(m/n))e_m(x) \\ G_n(x) &= \sum_{m\in\mathbb{Z}} \hat{\mu}(m/n)\sum_{l\neq 0} a_{m+nl}e_m(x) \end{aligned}$$

for $m \in \mathbb{Z}$ and $n = 1, 2, 3, \ldots$.

6.2 Pointwise Error Estimation

In this section, we shall concentrate on pointwise error estimation. It is straightforward to adapt this analysis to the multilevel case. Therefore we need to introduce the native space for the approximation of the periodic functions with Gaussian. In order to define native space let us take a periodic function $\phi(x)$ which is

$$\phi(x) = \sum_{z\in\mathbb{Z}} \mu(x-z). \tag{6.9}$$

Then the native space $\mathcal{N}_\phi$, associated with ϕ, is the subspace of $L_2(\mathbb{R}^d)$ defined by

$$\mathcal{N}_\phi := \left\{ f : \|f\|_\mu = \left[\sum_{k\in\mathbb{Z}} \frac{\hat{f}_k^2}{\hat{\phi}_k} \right]^{1/2} < \infty \right\} \tag{6.10}$$

where $\hat{f}_k$ is the Fourier transform of the periodic function f and $\hat{\phi}_k$ is the Fourier coefficient of the $\phi(x)$. Thus the native space for the approximation of the periodic functions with Gaussian

$$\mathcal{N}_\phi = \left\{ f : \|f\|_\mu := \left[\sum_{k\in\mathbb{Z}} \exp(2\pi^2 k^2) |a_k(f)|^2 \right]^{1/2} < \infty \right\}, \tag{6.11}$$

with the associated norm as given. Then, by using the Cauchy-Schwarz inequality, we have

$$\begin{aligned}
\|E_n f(x)\|_\infty &\leqslant \sum_{k\in\mathbb{Z}} |a_k|(1-\hat{\mu}(k/n)) \\
&= \sum_{k\in\mathbb{Z}} |a_k| \exp(\pi^2 k^2) \exp(-\pi^2 k^2)(1-\hat{\mu}(k/n)) \\
&\leqslant \left[\sum_{k\in\mathbb{Z}} |a_k| \exp(2\pi^2 k^2) \right]^{1/2} \left[\sum_{k\in\mathbb{Z}} \exp(-2\pi^2 k^2)(1-\hat{\mu}(k/n))^2 \right]^{1/2} \\
&= \left[\sum_{k\in\mathbb{Z}} \exp(-2\pi^2 k^2)(1-\hat{\mu}(k/n))^2 \right]^{1/2} \|f\|_\mu.
\end{aligned}$$

Similarly

$$\begin{aligned}\|G_n(x)\|_\infty &\leqslant \sum_{m\in\mathbb{Z}} |\hat{\mu}(m/n) \sum_{l\neq 0} a_{m+nl}| \\ &= \sum_{m\in\mathbb{Z}} |c_m^n| \\ &= |c_0^n| + \sum_{j\neq 0} |c_{nj}^n| + \sum_{j\in\mathbb{Z}} \sum_{k=1}^{n-1} |c_{nj+k}^n| \end{aligned} \tag{6.12}$$

where $c_m^n = \sum_{m\in\mathbb{Z}} \hat{\mu}(m/n) \sum_{l\neq 0} a_{m+nl}$. Then

- if $m = 0$

$$\begin{aligned}|c_0^n| &= \left|\sum_{l\neq 0} a_{nl}\right| \\ &\leqslant \sum_{l\neq 0} \exp(-\pi^2(nl)^2) \exp(\pi^2(nl)^2)\, |a_{nl}| \\ &\leqslant \left[\sum_{l\neq 0} \exp(-2\pi^2(nl)^2)\right]^{1/2} \left[\sum_{l\neq 0} \exp(2\pi^2(nl)^2)|a_{nl}|^2\right]^{1/2} \\ &\leqslant 2\exp(-\pi^2 n^2)\|f\|_\mu. \end{aligned} \tag{6.13}$$

- if $m = nj$, $j \in \mathbb{Z}$

$$\begin{aligned}|c_{nj}^n| &= \left|\hat{\mu}(j) \sum_{l\neq 0} a_{nj+nl}\right| \\ &\leqslant \hat{\mu}(j) \sum_{l\in\mathbb{Z}} \exp(-\pi^2(n(j+l))^2) \exp(\pi^2(n(j+l))^2)\, |a_{nj+nl}| \\ &\leqslant \hat{\mu}(j) \left[\sum_{l\in\mathbb{Z}} \exp(-2\pi^2 l^2)\right]^{1/2} \left[\sum_{l\in\mathbb{Z}} \exp(2\pi^2 l^2)|a_l|^2\right]^{1/2} \\ &\leqslant 2\hat{\mu}(j)\|f\|_\mu. \end{aligned} \tag{6.14}$$

- if $m = nj + k$, $1 \leqslant n - 1$, $j \in \mathbb{Z}$

$$\begin{aligned}
|c^n_{nj+k}| &= \left| \hat{\mu}(nj + k/n) \sum_{l \neq 0} a_{nj+k+nl} \right| \\
&\leqslant \hat{\mu}(j) \sum_{l \in \mathbb{Z}} \exp(-\pi^2(n(j+l)+k)^2) \exp(\pi^2(n(j+l)+k)^2) \, |a_{nj+k+nl}| \\
&\leqslant \hat{\mu}(j) \exp(-\pi^2 k^2) \left[\sum_{l \in \mathbb{Z}} \exp(-2\pi^2 l^2) \right]^{1/2} \left[\sum_{l \in \mathbb{Z}} \exp(2\pi^2 l^2) |a_l|^2 \right]^{1/2} \\
&\leqslant 2\hat{\mu}(j) \exp(-\pi^2 k^2) \|f\|_\mu. \qquad (6.15)
\end{aligned}$$

Using (6.13), (6.14) and (6.15) in (6.12), we get

$$\begin{aligned}
\|G_n(x)\|_\infty &\leqslant |c^n_0| + \sum_{j \neq 0} |c^n_{nj}| + \sum_{j \in \mathbb{Z}} \sum_{k=1}^{n-1} |c^n_{nj+k}| \\
&\leqslant 2\|f\|_\mu \exp(-\pi^2 n^2) + 2\|f\|_\mu \sum_{j \neq 0} \hat{\mu}(j) + 2\|f\|_\mu \sum_{j \in \mathbb{Z}} \hat{\mu}(j) \sum_{k=1}^{n-1} \exp(-\pi^2 k^2) \\
&\leqslant 2\|f\|_\mu \left[\exp(-\pi^2 n^2) + 2\exp(-2\pi^2) + 2\exp(-\pi^2) \right] \\
&\leqslant 10\|f\|_\mu \exp(-\pi^2). \qquad (6.16)
\end{aligned}$$

Consequently, we gain the following result for the uniform norm of quasi-interpolation with Gaussian basis functions.

Theorem 6.2. *Let $Q_{1/n}f$ be the quasi-interpolant to $f \in \mathcal{N}_\mu$. Then*

$$\|f - Q_{1/n}f\|_\infty \leqslant \left[\left(\sum_{k \in \mathbb{Z}} \exp(-2\pi^2 k^2)(1 - \hat{\mu}(k/n))^2 \right)^{1/2} + 10\exp(-\pi^2) \right] \|f\|_\mu.$$

According to above theorem, the second term of right hand side of above equation does not affect when the n is increasing. In other words it is constant

for all levels.

6.3 Error analysis of Multilevel Quasi-interpolation

In this section, we shall concentrate on the error analysis for the multilevel domain decomposition algorithm using the above results. In line with this objective, we will use the unit function since it will help us when we find the error bound for all functions. The error on level one is defined by

$$\begin{aligned} r_1(x) &= f(x) - Q_1 f(x) \\ &= 1 - \sum_{k\in\mathbb{Z}} \mu(x-k) \\ &= 1 - \sum_{m\in\mathbb{Z}} \hat{\mu}(m) e_m(x). \end{aligned} \tag{6.17}$$

Let choose the machine precision $\epsilon = 10^{-10}$. The coefficients $\hat{\mu}(m)$, $m = 2, -2, 3, -3, \ldots$, can be very small, as seen from the relation $e^{-8\pi^2} \sim 10^{-34}$. In other words, we can ignore the terms which are less than the machine precision. Thus the error on level one is

$$r_1(x) = \hat{\mu}(1)(e_1(x) + e_{-1}(x)). \tag{6.18}$$

In order to find an error bound for the second level, we need to interpolate the first level error $r_1(x)$ with a half mesh size

$$\begin{aligned} Q_{1/2}r_1(x) &= Q_{1/2}\hat{\mu}(1)(e_1(x)+e_{-1}(x)) \\ &= \hat{\mu}(1)Q_{1/2}(e_1(x)+e_{-1}(x)). \end{aligned} \tag{6.19}$$

From (6.9) we get

$$\begin{aligned} Q_{1/2}e_1(x) &= \sum_{z\in\mathbb{Z}} \hat{\mu}(z+1/2)e_{1+2z}(x) = \hat{\mu}(-1/2)e_{-1}(x)+\hat{\mu}(1/2)e_1(x)+\hat{\mu}(3/2)e_3(x) \\ Q_{1/2}e_{-1}(x) &= \sum_{z\in\mathbb{Z}} \hat{\mu}(z-1/2)e_{-1+2z}(x) = \hat{\mu}(-3/2)e_{-3}(x)+\hat{\mu}(-1/2)e_{-1}(x)+\hat{\mu}(1/2)e_1(x). \end{aligned}$$

So (6.18) becomes

$$Q_{1/2}r_1(x) = \hat{\mu}(1)\left[\hat{\mu}(1/2)(e_1(x)+e_{-1}(x))+\hat{\mu}(3/2)(e_3(x)+e_{-3}(x))\right]. \tag{6.20}$$

Since $\hat{\mu}(1)\hat{\mu}(3/2) < \epsilon$, we have

$$\begin{aligned} r_2(x) &= r_1(x) - Q_{1/2}r_1(x) \\ &= \hat{\mu}(1)(e_1(x)+e_{-1}(x)) - \hat{\mu}(1)\hat{\mu}(1/2)(e_1(x)+e_{-1}(x)) \\ &= \hat{\mu}(1)(1-\hat{\mu}(1/2))(e_1(x)+e_{-1}(x)). \end{aligned}$$

Similarly

$$\begin{aligned} Q_{1/4}r_2(x) &= Q_{1/4}\hat{\mu}(1)(1-\hat{\mu}(1/2))(e_1(x)+e_{-1}(x)) \\ &= \hat{\mu}(1)(1-\hat{\mu}(1/2))Q_{1/4}(e_1(x)+e_{-1}(x)). \end{aligned} \tag{6.21}$$

From (6.9) we get

$$\begin{aligned} Q_{1/4}e_1(x) &= \sum_{z\in\mathbb{Z}} \hat{\mu}(z+1/4)e_{1+4z}(x) = \hat{\mu}(-3/4)e_{-3}(x)+\hat{\mu}(1/4)e_1(x)+\hat{\mu}(5/4)e_5(x) \\ Q_{1/4}e_{-1}(x) &= \sum_{z\in\mathbb{Z}} \hat{\mu}(z-1/4)e_{-1+4z}(x) = \hat{\mu}(-5/4)e_{-5}(x)+\hat{\mu}(-1/4)e_{-1}(x)+\hat{\mu}(3/4)e_3(x) \end{aligned}$$

So the equation (6.20) converts

$$\begin{aligned} Q_{1/4}r_2(x) &= \hat{\mu}(1)(1-\hat{\mu}(1/2))[\hat{\mu}(1/4)(e_1(x)+e_{-1}(x)) \\ &+ \hat{\mu}(3/4)(e_3(x)+e_{-3}(x))+\hat{\mu}(5/4)(e_5(x)+e_{-5}(x))]. \end{aligned}$$

Since $\hat{\mu}(1)\hat{\mu}(3/4) < \epsilon$ and $\hat{\mu}(1)\hat{\mu}(5/4) < \epsilon$, we have

$$\begin{aligned} r_3(x) &= r_2(x) - Q_{1/4}r_2(x) \\ &= \hat{\mu}(1)(1-\hat{\mu}(1/2))(e_1(x)+e_{-1}(x)) - \hat{\mu}(1)(1-\hat{\mu}(1/2))\hat{\mu}(1/4)(e_1(x)+e_{-1}(x)) \\ &= \hat{\mu}(1)(1-\hat{\mu}(1/2))(1-\hat{\mu}(1/4))(e_1(x)+e_{-1}(x)). \end{aligned}$$

As a result, the error of the n^{th} iteration is

$$r_n(x) = \hat{\mu}(1)(1-\hat{\mu}(1/2))(1-\hat{\mu}(1/4))\cdots(1-\hat{\mu}(1/2^{n-1}))(e_1(x)+e_{-1}(x)).$$

So the n^{th} level error goes to zero when n increases.

Conjecture 6.3. If $f \in \mathcal{N}_\phi$. Let r_n be the error at level n approximating the unit function by the multilevel quasi-interpolation using Gaussian basis functions. Then

$$\|r_n\|_\infty = C(1-\hat{\mu}(1/2))(1-\hat{\mu}(1/4))\cdots(1-\hat{\mu}(1/2^{n-1}))\|f\|_\phi \tag{6.22}$$

The error analysis of the unit function by the multilevel quasi interpolation using Gaussian will help us to make a general error bound in the future.

Chapter 7

A Q-MuSIK-based Multidimensional Quadrature Formula

In conjunction with approximation and interpolation of high-dimensional functions, numerical approximation of integrals, which is usually referred to as quadrature or numerical integration, have a number of applications in applied sciences, ranging from mathematics, statistics, physics and engineering, to economics and finance.

There are numerous quadrature methods, each with its own advantages and disadvantages. Indeed, the dimension plays a significant role in determining which method can be utilized with greatest efficiently. For instance, classical integration methods are able to address accurate and efficient results to

high accuracy [29], [88]. However, when one would like to find quadrature of the multivariate functions for high dimensions, such as 5 or 10 dimensions, classical approaches are challenged by the curse of dimensionality, which it is necessary to deal with. In other words, standard numerical integration techniques of a function $f : [0,1]^d \to \mathbb{R}$, generally need N^d function computation to deliver rates of convergence of order $N^{-\epsilon}$, where $\epsilon > 0$ is independent of d, on a d-dimensional grid with N points in each direction. Of course this leads to computationally expensive algorithms since the interpolation problem increases in size exponentially.

In order to overcome this problem, a number of techniques have been proposed in the computational mathematics community over the last five decades. For example, some of the these methods, based on randomness, are the Monte-Carlo, quasi-Monte-Carlo and hierarchical Monte-Carlo methods [20], [2], [33], [26], [76]. On the other hand, Smolyak, sparse grid, hyperbolic cross product and boolean interpolation-type constructions are other types of proposed integration techniques [87], [30], [90], [86], [16]. Monte-Carlo methods, of course, have some advantageous such as robustness, ease of implementation and convergence rate without the dimension d. However, these methods have slow convergence rates in a probabilistic sense [20], [2]. In other respects, sparse grid and hyperbolic cross products provide $N(\log N)^d$ complexity for the multivariate functions f. In addition to these, the "p-version" of sparse grids has been introduced for the computing multidimension integrals arising from the numerical approximation of elliptic boundary-value problems with random coefficients [5], [79]. More recently, a numerical integration algo-

rithm based on the interpolation of the multidimensional integrand has been introduced by Dong et al. by using the Multilevel Sparse Interpolation with Kernels (MuSIK) method [31]. In this study, numerical evidence shows that multilevel sparse Gaussian kernels integration for high-dimensional problems is able to deliver accurate and efficient results to high accuracy. On the other hand, this method provides more rapid results since undesired grids are removed from interpolation.

In this chapter, a new numerical integration algorithm has been proposed, based on the quasi-multilevel sparse interpolation with kernels (Q-MuSIK quadrature) approach for high-dimensional numerical integration. This method can be viewed as an alternative to MuSIK quadrature, by offering comparable computational time and convergence rate on a uniform sparse grid. As mentioned in previous chapters, Q-MuSIK is able to provide a powerful interpolation method by combining the superior properties of quasi-interpolation such as simplicity, smoothness and rapid exponential decay at infinity, and multilevel sparse grids in an advantageous manner. Thus the Q-MuSIK quadrature algorithm provides a good approximation for the high-dimensional integrations. In addition to this Q-MuSIK based quadrature formula provide us positive weights since there is no need to solve large algebraic systems. A broad range of numerical experiments highlight the fact that the proposed algorithm for quadrature for high-dimensional problems has practical applicability.

7.1 Quasi-Quadrature Method

We now give the quasi quadrature method in order to compare our new algorithm, which will be introduced over the next few sections. For this reason, we will use the quasi-interpolation method for integration over the unit cube. A quadrature rule will on d dimensions approximate the integral

$$\int_{[a,b]^d} f(x)dx \approx \sum_{\mathbf{m}\in\Omega} \lambda_{\mathbf{m}} f(\mathbf{m}), \tag{7.1}$$

where $m \subset [a,b]^d$ is a finite set of points and $\lambda_{\mathbf{m}}$'s are the weights. If we have the quasi-interpolation operator

$$Q_{h,\rho} f(\mathbf{x}) = \frac{1}{(\pi\rho)^{d/2}} \sum_{\mathbf{m}\in\mathbb{Z}^d} f(h\mathbf{m}) e^{-|\mathbf{x}-h\mathbf{m}|^2/\rho h^2}, \tag{7.2}$$

we get the quadrature formula

$$\int_{[a,b]^d} f(x)dx \approx \frac{1}{(\pi\rho)^{d/2}} \sum_{\mathbf{m}\in\mathbb{Z}^d} f(h\mathbf{m}) \int_{[a,b]^d} e^{-|\mathbf{x}-h\mathbf{m}|^2/\rho h^2} d\mathbf{x}, \tag{7.3}$$

where

$$\int_{[a,b]^d} e^{-|\mathbf{x}-h\mathbf{m}|/\rho h^2} d\mathbf{x} = \prod_{l=1}^{d} \int_a^b e^{-|x_l - h m_l|^2/\rho h^2} dx_l. \tag{7.4}$$

Here, the tensor product property of Gaussian basis functions is used to compute weights. As mentioned in previous chapters, since the Gaussian basis functions are straightforwardly computed, this algorithm is able to deliver rapid and highly accurate computation of the integration weights.

Then by changing the variables $\xi_l = \frac{|x_l - hm_l|}{\sqrt{\rho}h}$, we get

$$W^{Quasi} = \frac{h^d}{\pi^{d/2}} \sum_{\mathbf{m}\in\mathbb{Z}^d} f(h\mathbf{m}) \prod_{l=1}^{d} \varphi(m_l), \tag{7.5}$$

where

$$\varphi(m_l) = -\frac{\sqrt{\pi}}{2}\left[\text{erf}\left(\frac{a - hm_l}{\sqrt{\rho}h}\right) - \text{erf}\left(\frac{b - hm_l}{\sqrt{\rho}h}\right)\right] \quad l = 1, \ldots, d, \tag{7.6}$$

with the error function erf defined by

$$\text{erf}\,(x) = \frac{2}{\sqrt{\pi}} \int_0^x e^{-t^2} dt. \tag{7.7}$$

The error function can be computed to arbitrary precision.

7.2 Q-SIK Quadrature Formula

Now, we introduce a new Q-SIK quadrature formula on a sparse grid for a function of several variables. In accordance with this purpose, we need to compute a number of sub-anisotropic quasi-quadrature problems on well constructed subgrids, and thus obtain the Q-SIK quadrature by combining the resultant sub-quadratures linearly.

In a similar way, all the subgrids for each level can be constructed in a similar manner to the method discussed in Chapter 4. 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 compute each subgrid 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 f at the grids $\Upsilon_{\mathbf{l}}$ can be defined by

$$W_l = \frac{h^d}{\pi^{d/2}} \sum_{\mathbf{m}\in\Upsilon_{\mathbf{l}}} f(h\mathbf{m}) \prod_{l=1}^{d} \varphi_l(m_l), \tag{7.8}$$

where

$$\varphi(m_l) = -\frac{\sqrt{\pi}}{2}\left[\text{erf}\left(\frac{a-hm_l}{\sqrt{\rho}h}\right) - \text{erf}\left(\frac{b-hm_l}{\sqrt{\rho}h}\right)\right] \quad l = 1,\ldots,d, \tag{7.9}$$

with the error function erf defined by

$$\text{erf}\,(x) = \frac{2}{\sqrt{\pi}} \int_0^x e^{-t^2} dt. \tag{7.10}$$

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

$$W_n^{Q-SIK} = \sum_{q=0}^{d-1} (-1)^q C_q^{d-1} \sum_{|\mathbf{l}|_1 = n+(d-1)-q} W_l. \tag{7.11}$$

For example, in 3-D the quasi-SKI is given by

$$W_n^{Q-SIK}(\mathbf{x}) = \sum_{|\mathbf{l}|_1 = n+2} W_l - 2 \sum_{|\mathbf{l}|_1 = n+1} W_l + \sum_{|\mathbf{l}|_1 = n} W_l. \tag{7.12}$$

The error function can be computed to arbitrary precision. The key idea here is subtracting the redundant grids visited on several occasions. That is, the middle term on the right hand side of equation (7.12) serves to remove the undesired grids used more than once, which come from the first and third term on the right hand side of equation (7.12).

7.3 Q-MuSIK Quadrature Method

As we saw in previous chapters, significant progress for interpolation has been achieved in terms of computational time, complexity and convergence. In other words, using quasi-interpolation in sparse interpolation with kernels results in decreasing computational complexity because of the nature of quasi-interpolation.

In order to take advantage of the outstanding properties of multilevel quasi-sparse kernel interpolation with kernels, we will use it in numerical integration, in particular in high dimensions. Thus, this method provides us more accurate and faster integration values in comparison with the classical full grid technique.

Along similar lines to multilevel sparse interpolation, we will again use nested sparse grids which, sorted from lower level to the higher level, are given by

$$\mathcal{S}_{1,d} \subset \mathcal{S}_{2,d} \subset \ldots \subset \mathcal{S}_{n,d},$$

where d is the dimension and n is the quadrature level. Then, by using the

Test Function	Integration Value
$F_1^{2d}(x,y)$	2.452413044563417e-01
$F_2^{2d}(x,y)$	4.444444444441538e-01
$F_3^{2d}(x,y)$	4.462230950426736e-02
$F_4^{2d}(x,y)$	1.163806395814231e-01
$F_5^{2d}(x,y)$	2.865833317365560e-01
$F_6^{2d}(x,y)$	1.562500000000056e-02

Table 7.1: The exact value of test functions on the domain $[0,1]^2$

same steps with Algorithm 2, Q-MuSIK-based quadrature can be computed.

7.4 Numerical Experiments in 2-D

In this section we give a number of numerical results, which use the same test functions described in Chapter 4 and Chapter 5. The exact values of respective integrals on the domain $[0,1]^2$ with 16 digits are given in Table 7.1 The Q-MuSIK quadrature computation of the first test function, recording the number of nodes used, and the absolute and relative errors, are given in Table 7.2.

According to this table, the proposed Q-MuSIK method has been successfully applied to the numerical integration problems in $\mathbb{R}^2$ as shown by its encouraging performance, especially with regard to complexity.

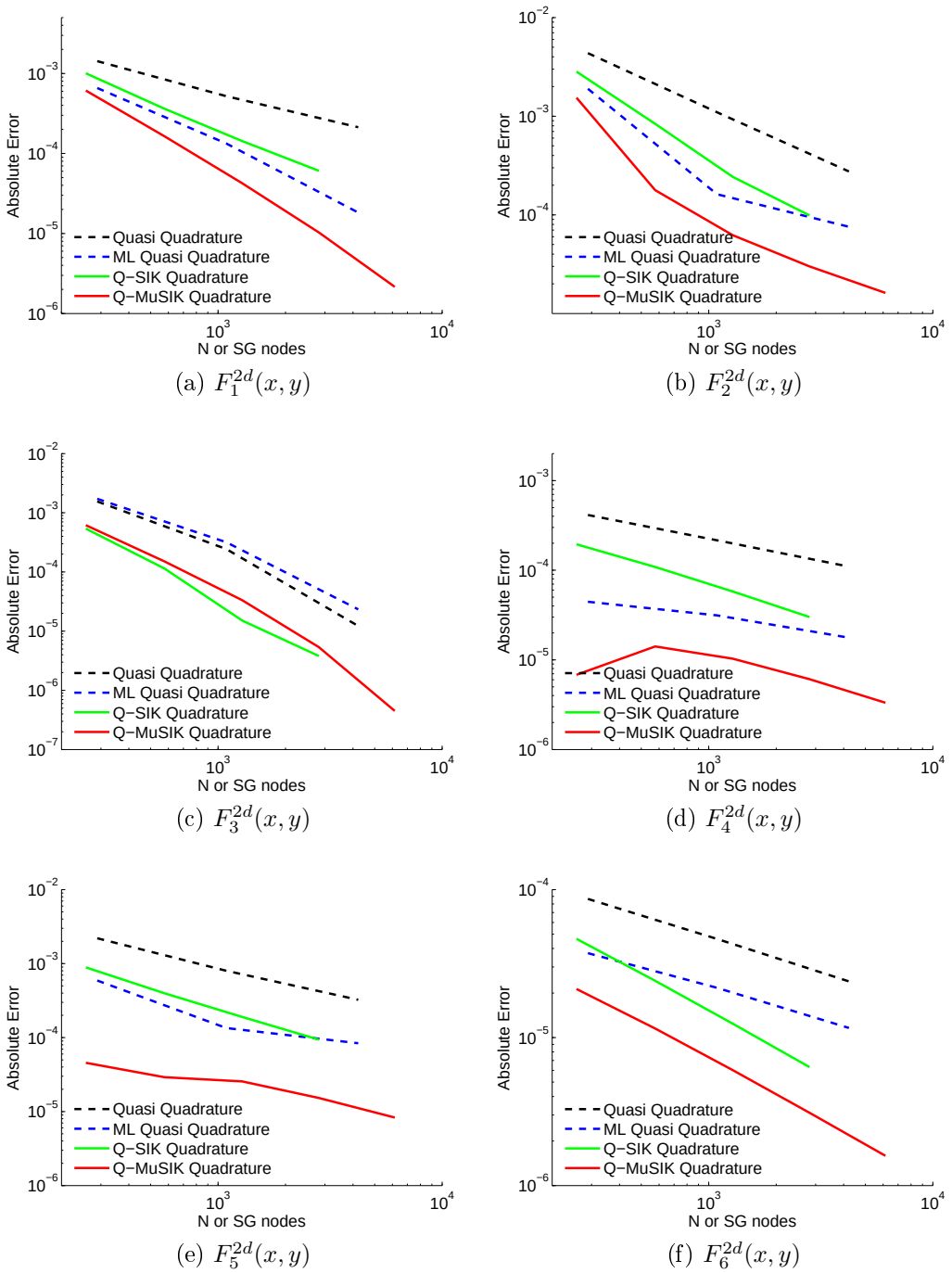
In order to demonstrate the outstanding features of the proposed technique, the results of the quasi quadrature, multilevel quasi quadrature, Q-SIK quadrature and Q-MuSIK quadrature can be compared in Figure 7.1. From the data

SGnode	DOFs(SG)	Absolute Error	Relative Error
9	9	8.663271e-02	3.532550e-01
21	39	4.363387e-02	1.779222e-01
49	109	6.613826e-03	2.696865e-02
113	271	2.128311e-03	8.678435e-03
257	641	6.120334e-04	2.495637e-03
577	1475	1.640092e-04	6.687668e-04
1281	3333	4.251719e-05	1.733688e-04
2817	7431	1.031525e-05	4.206165e-05
6145	16393	2.150008e-06	8.766909e-06

Table 7.2: Q-MuSIK quadrature results using Gaussian basis functions with shape parameter $d = 0.4$, test function $F_1^{2d}(x, y)$, on the domain $[0, 1]^2$.

in this figure, the Q-MuSIK quadrature method shows better performance than the Q-SIK quadrature. Herein, the multilevel algorithm plays a significant role. By combining the multilevel algorithm refinement feature and the quasi-sparse kernel-based quadrature method, we obtain a better convergence rate.

In addition to this, the Q-MuSIK quadrature formula performs better than the multilevel quasi-quadrature method. The superior performance of Q-MuSIK over the classical multilevel method is clearly observed. We obtain a smaller error for Q-MuSIK as compared to quasi-multilevel quadrature approaches.



(a) $F_1^{2d}(x,y)$ (b) $F_2^{2d}(x,y)$

(c) $F_3^{2d}(x,y)$ (d) $F_4^{2d}(x,y)$

(e) $F_5^{2d}(x,y)$ (f) $F_6^{2d}(x,y)$

Figure 7.1: Absolute error versus N(Quasi-quadrature, Multilevel Quasi-quadrature) or SG (Q-SIK quadrature, Q-MuSIK quadrature) nodes using Gaussian basis functions with $\rho = 0.4$: Quasi-quadrature (black), Q-SIK quadrature (green), Multilevel Quasi-quadrature (blue) and Q-MuSIK quadrature (red) on the domain $[0,1]^2$.

Test Function	Integration Value
$F_1^{3d}(x,y)$	7.766696346045100e-02
$F_2^{3d}(x,y)$	9.999999999999993e-01
$F_3^{3d}(x,y)$	1.460689557604901e-02
$F_4^{3d}(x,y)$	-6.185674957636504e-01
$F_5^{3d}(x,y)$	3.885633084290729e-01
$F_6^{3d}(x,y)$	1.953124999999999e-03

Table 7.3: The exact values of test functions on the domain $[0,1]^3$

7.5 Numerical Experiments in 3-D

Analogically, we present numerical integration results for 3-D in this section. The exact values of the respective integrals on the domain $[0,1]^3$ with 16 digits are given in Table 7.2. The 3-D results of quasi-, multilevel quasi-, Q-SIK and Q-MuSIK quadrature can be compared in Figure 7.2. These results confirm that Q-MuSIK is superior to both the Q-SIK and multilevel quasi-quadrature methods. The numerical studies of this section suggest that Q-SIK and Q-MuSIK can be successfully applied in $\mathbb{R}^d$ for $d \geq 2$ and verifies its superior performance over the direct and multilevel quasi-quadrature approaches.

7.6 Numerical Experiments in High Dimensions

In Figure 7.3 we have compared the MuSIK quadrature and Q-MuSIK quadrature in terms of absolute error versus degree of freedom and absolute error versus evaluation time. In these comparison we have used four, five and ten

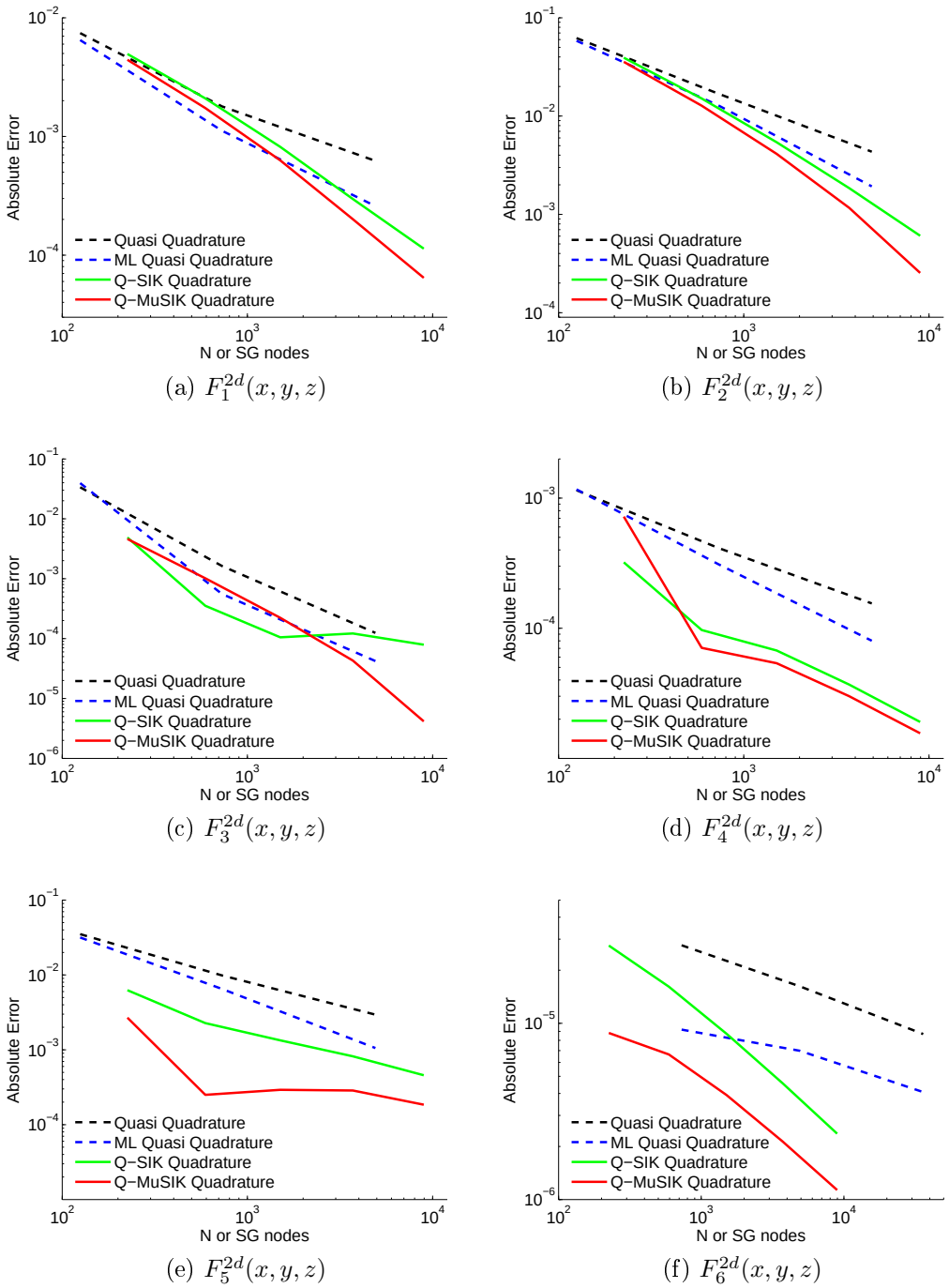
dimensional test functions. For the ten dimension our test function is

$$F_7^{10d} = \prod_{i=1}^{10} e^{-x_i(1-x_i)},$$

and the corresponding exact integral of this function on the domain $[0,1]^10$ with 16 digits accuracy is 0,194279067580947. These results confirms that Q-MuSIK method provide us more rapid results for the same amount of error.

7.7 Discussion

In this chapter we have applied the new quasi-multilevel sparse interpolation with kernels method to numerical integration. Similar to the interpolation performance of the proposed Q-MuSIK scheme, the new Q-MuSIK quadrature has achieved better convergence in high dimensions. The proposed technique provide us positive weights which is desired for numerical integration. The Q-MuSIK quadrature mostly outperforms both the Q-SIK and classical multilevel quasi-quadrature methods in terms of accuracy. Numerical results confirm the adequacy of the high dimensional numerical integration. In addition to this, we have compared the MuSIK and Q-MuSIK quadrature with different test function in high dimensions. This comparison shows that Q-MuSIK is superior than MuSIK method in terms of speed since the nature of quasi interpolation.

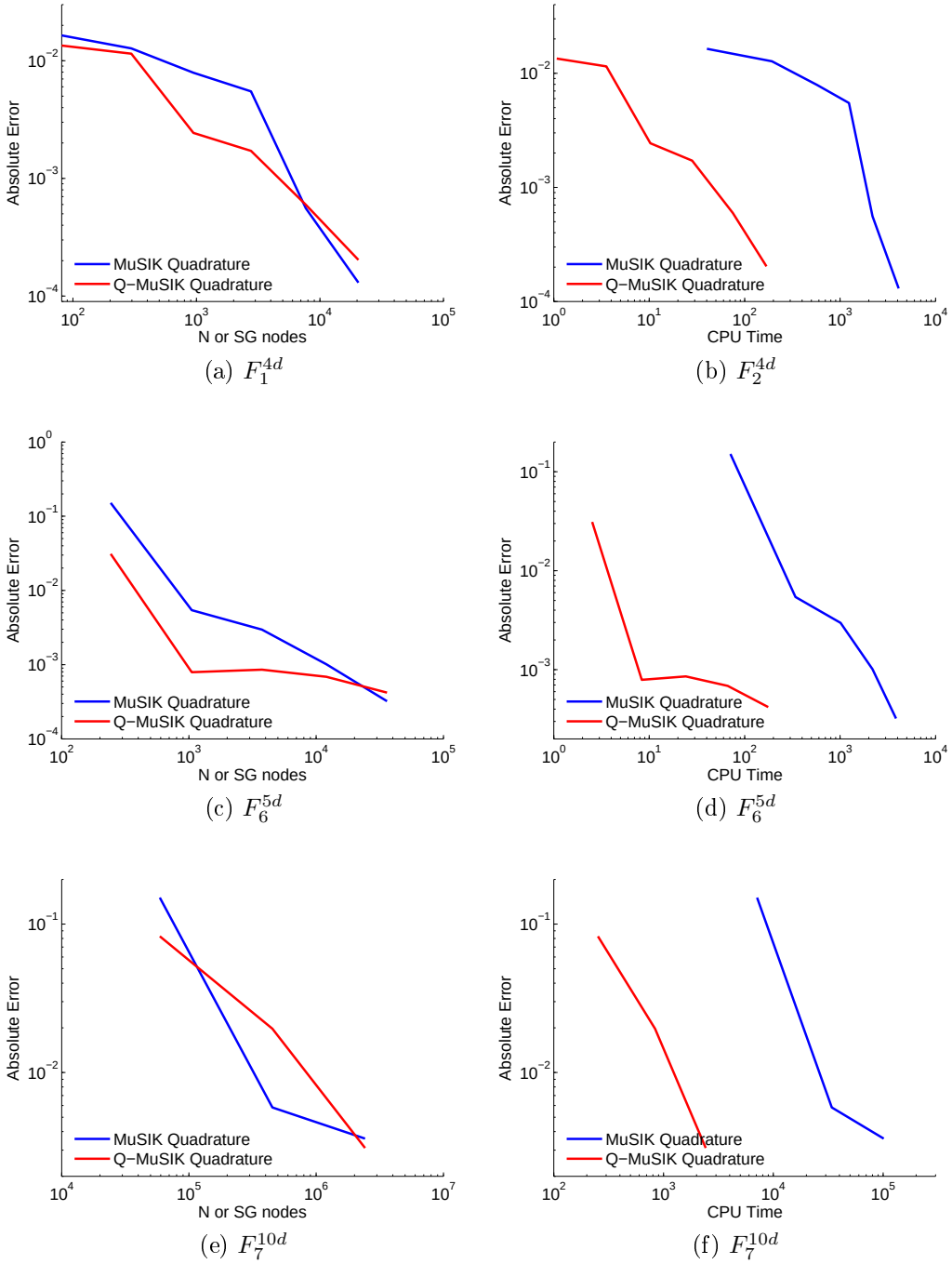


(a) $F_1^{2d}(x, y, z)$ (b) $F_2^{2d}(x, y, z)$

(c) $F_3^{2d}(x, y, z)$ (d) $F_4^{2d}(x, y, z)$

(e) $F_5^{2d}(x, y, z)$ (f) $F_6^{2d}(x, y, z)$

Figure 7.2: Absolute error versus N(Quasi-quadrature, Multilevel Quasi-quadrature) or SG (Q-SIK quadrature, Q-MuSIK quadrature) nodes using Gaussian basis functions with $\rho = 0.4$: Quasi-quadrature (black), Q-SIK quadrature (green), Multilevel Quasi-quadrature (blue) and Q-MuSIK quadrature (red) on the domain $[0, 1]^3$.



(a) F_1^{4d} (b) F_2^{4d}

(c) F_6^{5d} (d) F_6^{5d}

(e) F_7^{10d} (f) F_7^{10d}

Figure 7.3: Absolute error versus SG and CPU Time(Q-MuSIK quadrature, Q-MuSIK quadrature) nodes using Gaussian basis functions with $\rho = 0.4$ and $c = 0.45$: MuSIK quadrature (blue) and Q-MuSIK quadrature (red) on the domain $[0, 1]^{4,5,10}$.

Chapter 8

Conclusions and Future Work

8.1 Conclusions

In this thesis, we have presented and compared two algorithms, MuSIK and Q-MuSIK, as solutions to the multidimensional interpolation problem on well designed data sets. MuSIK is the multilevel version of the SIK method, which combines the positive features of radial basis function interpolation and the sparse grid technique. In other words, this scheme provides us with a more powerful interpolation method, in particular in high dimensions, since RBFs extends in its practicability in almost any dimension, whilst sparse grid methods can deal with the complexity issue in high dimensions. In addition to this we have presented the MuSIK method with the exactness of the sparse grid interpolation. However, in order to take the advantage of the MuSIK method, a further step had to be introduced via the Q-MuSIK

technique, which is the multilevel version of the Q-SIK method, based on quasi-interpolation.

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. In addition, it has a number of desirable features, such as good approximation behaviour, easy evaluation and computation, good shape properties, certain polynomial reproduction, and so on. The Q-SIK algorithm couples these good properties and the complexity of the sparse grid technique. Although Q-SIK visits some nodes more than once, it still uses a smaller number of nodes than the full grid scheme. Thus it still computes the interpolation with less time when comparised to the classical quasi-interpolation method. Then, in order to construct the Q-MuSIK scheme, we have used the SIK method at each level. Thus the Q-MuSIK algorithm computes the interpolation result on a shorter timescale when compared to the MuSIK method in problems with the same number of degrees of freedom because of the properties of quasi-interpolation listed above.

Additionally, a bivariate Q-MuSIK algorithm generally accelerates the convergence rate when compared to both the Q-SIK method and classical quasi-interpolation and its multilevel variant. The numerical experiments in Chapter 5 confirm that Q-MuSIK is usually superior to the other methods on gridded data. Hence the Q-MuSIK scheme gives generally better output in terms of complexity, accuracy, computational time and stability compared to the classical quasi-interpolation technique.

Numerical experiments show that the Q-MuSIK scheme is not only efficient

for bivariate problems, but is also able to solve large multidimensional interpolation problems. In Chapter 5, numerical results have been presented for the interpolation of d-variate functions in $\mathbb{R}$ for $d = 3$ and $d = 4$. In analogy to bivariate experiments, the results of high-dimensional interpolation problems support the idea that Q-MuSIK gives generally preferable output in proportion to classical quasi-interpolation methods. The similarly superior performance of the Q-MuSIK method in terms of computational time and accuracy over the Q-SIK method has been observed in high-dimensional problems. Additionally, the Q-MuSIK algorithm might be recommended as an alternative interpolation method to MuSIK at low degrees of freedom since it has better performance over any given time interval. Of course, although MuSIK is one of the best interpolation techniques in terms of accuracy, stability, etc., Q-MuSIK might be a good alternative, especially in high dimensions due to its nice properties.

The Q-MuSIK scheme also made a noteworthy improvement to numerical integration. Combining the sparse grid scheme and quasi-interpolation method, we have implemented the Q-MUSIK approach to find quadrature for d-variate functions in $\mathbb{R}$ for $d = 2$ and $d = 3$. The main advantage of using Q-MuSIK method for quadrature is that it produces the positive weights for numerical calculation. Numerical results, which were presented in Chapter 7, confirm the convergence features of the proposed approach in the field of numerical integration.

In order to support our numerical results theoretically, a convergence analysis for quasi-interpolation has been given in Chapter 6. In this chapter, we

considered the convergence rates for multilevel quasi-interpolation of periodic functions using Gaussian basis functions on a grid. In theory, quasi-interpolation uses shifts of the Gaussian kernel on a full infinite grid. Taking advantage of shifting features of Gaussian functions, we have calculated the single level error using the exponential function $\exp(2\pi imx)$. Then, by following the multilevel algorithm scheme of Floater and Iske [40], the error at level n approximating unit function by the multilevel quasi-interpolation using Gaussian functions has been presented at the end of the Chapter 7.

8.2 Future Work

The main goal of the current study was to develop the Q-MuSIK method by using the same scheme as MuSIK and presenting its superior performance with regards to the problems of interpolation, especially in high dimensions. Its success has been confirmed both by a number of numerical experiment results and theoretically.

A further study could assess the convergence analysis of multilevel interpolation for more general functions defined over the bounded domain. In order to make a generalisation along these line our findings will, of course, play a significant role.

Another area of interest is in applying Q-SIK to numerical methods for the solution of partial differential equations (PDE), in particular for high dimensions. This would help us for option pricing, which needs to solve the

Black-Scholes equation. In order to apply the Q-MuSIK method to derivative pricing problems, we initially need to show that it is able to approximate the pay-off function. In line with this purpose, we have used the test function f_6^{nd} in our experiments. So, for future work, we might be able to apply the proposed method to option pricing problems.

Bibliography

[1] A. Abramowitz and I. Stegun. *Handbook of Mathematical Functions*. 1964.

[2] J. Dick andF. Y. Kuo and I. H. Sloan. High-dimensional integration: The quasi-monte carlo way. *Acta Numerica*, 22:133–288, 5 2013.

[3] N. Arad, N. Dyn, and D. Reisfeld. Image warping by radial basis functions: applications to facial expressions. *CVGIP Graphical Models and Image Processing*, 56(2):161–172, 1994.

[4] K. I. Babenko. Approximation by trigonometric polynomials in a certain class of periodic functions of several variables. *Soviet Mathematics Doklady*, 1:672–675, 1960.

[5] I. Babuska, F. Nobile, and R. Tempone. A stochastic collocation method for el-liptic partial differential equations with random input data. *SIAM Jornal of Numerical Analysis*, 45:1005–1034, 2007.

[6] M. Bauer, O. Buchtala, H. Timo, K. Ralf, S. Bernhard, and W. Robert. Technical data mining with evolutionary radial basis function classifiers. *Applied Soft Computing*, 9:765–774, 2009.

[7] B. J. C. Baxter. Conditionally positive functions and p-norm distance matrices. *Constructive Approximation*, 7(4):427–440, 1991.

[8] R. Beatson, O. Davydov, and J. Levesley. Error bounds for anisotropic rbf interpolation. *Journal of Approximation Theory*, 162(3):512–527, 2010.

[9] R. Beatson and M. Powell. Univariate multiquadric approximation: quasi-interpolation to scattered data. *Constructive Approximation*, 8:275–288, 1992.

[10] R. K. Beatson, J. Levesley, and C. T. Mouat. Better bases for radial basis function interpolation problem. *Journal of Computational and Applied Mathematics*, 236(4):434–446, 2011.

[11] R. K Beatson, W. A. Light, and S. Billings. Fast solution of the radial basis function interpolation equations: domain decomposition methods. *SIAM J. Sci. Comput.*, 22(5):1717–1740, 2000. electronic.

[12] R. Belmann. *Adaptive Control process: a guide tour*. Princeton, 1961.

[13] D. Brown, L. Ling, E. Kansa, and J. Levesley. On approximate cardinal preconditioning methods for solving pdes with radial basis functions. *Engineering Analysis with Boundary Elements*, 29(4):343–353, 2005. Mesh Reduction Methods - Part III.

[14] M. Buhmann and N. Dyn. Spectral convergence of multiquadric interpolation. *Proceeding of the Edinburgh Mathematical Society*, 36(2):319–333, 1993.

[15] H. J. Bungartz and M. Griebel. A note on the complexity of solving poisson's equation for spaces of bounded mixed derivatives. *Journal of Complexity*, 15(2):167–199, 1999.

[16] H.-J. Bungartz and M. Griebel. Sparse grids. *Acta Numerica*, 13:147–269, 2004.

[17] H.-J. Bungartz, M. Griebel, D. Roschke, and C. Zenger. Pointwise convergence of the combination technique for the laplace equation. *East-West Journal of Numerical Mathematics*, 2:21–45, 1994.

[18] H.-J. Bungartz, M. Griebel, D. Roschke, and C. Zenger. Two proofs of convergence for the combination technique for the efficient solution of sparse grid problems, in *Domain Decomposition Methods in Scientific and Engineering Computing. Contemporary Mathematics*, 180:15–20, 1994.

[19] H.-J. Bungartz, M. Griebel, and U. Rude. Extrapolation, combination, and sparse grid techniques for elliptic boundary value problems. *Computer Methods in Applied Mechanics and Engineering*, 116(1-4):243–252, 1994.

[20] R. E. Caflisch. Monte carlo and quasi-monte carlo methods. *Acta numerica*, 7:1–49, 1998.

[21] J. C. Carr, W. R. Fright, and R. K. Beatson. Surface interpolation with radial basis functions for medical imaging. *IEEE Transactions on Medical Imaging*, 16:96–107, 1997.

[22] G. Casciola, D. Lazzaro, L. B. Montefusco, and S. Morigi. Shape preserving surface reconstruction using locally anisotropic radial basis function interpolants. *Computers and Mathematics with Applications*, 51(8):11185–1198, 2006.

[23] G. Casciola, L.B. Montefusco, and S. Morigi. The regularizing properties of anisotropic radial basis functions. *Applied Mathematics and Computation*, 190(2):1050–1062, 2007.

[24] Z.X. Chen and F.L. Cao. Global errors for approximate approximations with gaussian kernels on compact intervals. *Applied Mathematics and Computation*, 217:725–734, 2010.

[25] Z.X. Chen, F.L. Cao, and J. Hu. Error estimates of quasi-interpolation and its derivatives. *Journal of Computational and Applied Mathematics*, 236:3137–3146, 2012.

[26] K. A. Cliffe, M. B. Giles, R. Scheichl, and A. L. Teckentrup. Multilevel monte carlo methods and applications to elliptic pdes with random coeficients. *Computing and Visualization in Science*, 14:3–15, 2011.

[27] L. Condat and D. Van De Ville. Quasi-interpolating spline models for hexagonally-sampled data. *IEEE Transactions on Image Processing*, 16(5):1195–1206, 2007.

[28] C. Dagnino, V. Demichelis, and E. Santi. Numerical integration based on quasi-interpolating splines. *Computing*, pages 149–163, 1993.

[29] P. J. Davis. *Interpolation and approximation*. Dover Publications Inc., New York, 1975. Republication, with minor corrections, of the 1963 original, with a new preface and bibliography.

[30] F.-J. Delvos. d-variate boolean interpolation. *Journal of Approximation Theory*, 34:99–114, 1982.

[31] 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.

[32] M. R. Dubal. Domain decomposition and local refinement for multiquadric approximations. i. second-order equations in one-dimension. *Journal of Applied Mathematics and Computer Science*, 1(1):146–171, 1994.

[33] C. Schwab F. Y. Kuo and I. H. Sloan. Quasi-monte carlo methods for high-dimensional integration: the standard (weighted hilbert space) setting and beyond. *ANZIAM Journal*, 53:1–37, 2011.

[34] G. E. Fasshauer. Hermite interpolation with radial basis functions on spheres. *Advance Computational Mathematics*, 11(1):81–96, 1999.

[35] G. E. Fasshauer. Solving differential equations with radial basis functions: multilevel methods and smoothing. *Advance Computational Mathematics*, 11(2-3):139–159, 1999.

[36] G. E. Fasshauer. *Meshfree approximation methods with MATLAB, volume 6 of Interdisciplinary Mathematical Sciences*. World Scientific Publishing Co. Pte. Ltd., Hackensack, NJ, 2007.

[37] G. E. Fasshauer and J. W. Jerome. Multistep approximation algorithms: improved convergence rates through postconditioning with smoothing kernels. *Advance Computational Mathematics*, 10(1):1–27, 1999.

[38] G. E. Fasshauer and M. J. Mccourt. Stable evaluation of gaussian rbf interpolants. *SIAM Journal on Scientific Computing*, 34(2):737–762, 2012.

[39] G. E. Fasshauer and J. G. Zhang. Iterated approximate moving least squares approximation. *Advances in Meshfree Techniques*, page 221–240, 2007.

[40] M. S. Floater and A. Iske. Multistep scattered data interpolation using compactly supported radial basis functions. *Journel of Computational Applied Mathematics*, 73(1-2):65–78, 1996.

[41] B. Fornberg, E. Larsson, and N. Flayer. Stable computation with gaussain radial basis functions. *SIAM Journal on Scientific Computing*, 33(2):869–892, 2011.

[42] B. Fornberg and C. Piret. A stable algorithm for flat radial basis functions on a sphere. *SIAM Journal on Scientific Computing*, 30(1):60–80, 2007.

[43] B. Fornberg and G. Wright. Stable computation of multiquadric interpolants for all values of the shape parameter. *Computers and Mathematics with Applications*, 48(5-6):853–867, 2004.

[44] R. Franke. Scattered data interpolation: tests of some methods. *Mathematics of Computation*, 38(157):181–200, 1982.

[45] J. Garcke and M. Hegland. Fitting multidimensional data using gradient penalties and the sparse grid combination technique. *Computing*, 84(1-2):1–25, 2009.

[46] E. H. Georgoulis, J. Levesley, and F. Subhan. Multilevel sparse kernel-based interpolation. *SIAM Journal of Scientific Computing*, 35:815–832, 2013.

[47] P. Giesl. Construction of a global lyapunov function using radial basis functions with a single operator. *Discrete and Continuous Dynamical Systems Series B*, 7(1):101–124, 2007.

[48] P. Giesl. Construction of a local and global lyapunov function using radial basis functions. *Journal of Approximation Theory*, 153(2):184–211, 2008.

[49] F. Girosi. Some extensions of radial basis functions and their applications in artificial intelligence. *Computers and Mathematics with Applications*, 24(12):61–80, 1992.

[50] M. Griebel. A parallelizable and vectorizable multi-level algorithm on sparse grids, in *Parallel Algorithms for Partial Differential Equations. Notes on Numerical Fluid Mechanics*, 31:94–100, 1990.

[51] M. Griebel. The combination technique for the sparse grid solution of pdes on multiprocessor machines. *Parallel Processing Letters*, 2(1):61–70, 1992.

[52] M. Griebel. Sparse grid multilevel methods, their parallelization and their application to cfd, in *Proc. Parallel Computational Fluid Dynamics. Elsevier*, pages 161–174, 1993.

[53] M. Griebel, M. Schneider, and C. Zenger. A combination technique for the solution of sparse grid problems, in *Iterative Methods in Linear Algebra. Elsevier*, pages 263–281, 1992.

[54] S. J. Hales and J. Levesley. Error estimates for multilevel approximation using polyharmonic splines. *Numerical Algorithms*, 30(1):1–10, 2002.

[55] R. L. Hardy. Multiquadrics of topography and other irregular surface. *Journal of Geophysical Research*, 76:1905–1915, 1971.

[56] R. L. Hardy. Geodetic application of multiquadric analysis, avn allg. *Vermess Nachr.*, 79:389–406, 1972.

[57] R. L. Hardy. Research results in application equations to serveying and mapping problem. *Survg. mapp*, 35:321–332, 1975.

[58] R. L. Hardy. Theory and applications of the multiquadric-biharmonic method. 20 years of discovery 1968-1988. *Computers and Mathematics with Applications*, 19(8-9):163–208, 1990.

[59] M. Hegland, J. Garcke, and V. Challis. The combination technique and some generalisations. *Linear Algebra and its Applications*, 420(2-3):249–275, 2007.

[60] Y. C. Hon. A quasi-radial basis functions method for american options pricing. *Computers and Mathematics with Applications*, 43:513–524, 2002.

[61] A. Iske and J. Levesley. Multilevel scattered data approximation by adaptive domain decomposition. *Numerical Algorithms*, 39(1-3):187–198, 2005.

[62] Armin Iske. Hierarchical scattered data filtering for multilevel interpolation schemes. In *In Mathematical methods for curves and surfaces*, pages 211–221. Vanderbilt Univ. Press, 2000.

[63] M. Hegland J. Garcke and O. Nielsen. Parallelisation of sparse grids for large scale data analysis. *ANZIAM Journal*, 48(1):11–22, 2006.

[64] E. J. Kansa. Multiquadrics | a scattered data approximation scheme with applications to computational filuid-dynamics. i. surface approximations and partial derivative estimates. *Computers and Mathematics with Applications*, 19(8-9):127–145, 1990.

[65] E. J. Kansa. Multiquadrics | a scattered data approximation scheme with applications to computational filuid-dynamics. ii. solutions to parabolic, hyperbolic and elliptic partial differential equations. *Computers and Mathematics with Applications*, 19(8-9):147–161, 1990.

[66] C. Y. Li and C. G. Zhu. A multilevel univariate cubic spline quasi-interpolation and application to numerical integration. *Mathematical Methods in the Applied Sciences*, 33(13):1578–1586, 2010.

[67] W. A. Light, E. W. Cheney, and N. Dyn. Interpolation by piecewise-linear radial basis functions. *Journal of Approximation Theory*, 59(2):202–223, 1989.

[68] L. Ling. A univariate quasi multiquadric interpolation with better smoothness. *Computers and Mathematics with Applications*, 48(5-6):897–912, 2004.

[69] L. Ling. Multivariate quasi-interpolation schemes for dimension-splitting multiquadric. *Applied MAthematics and Computation*, 161(1):195–209, 2005.

[70] M. Schneider M. Griebel and C. Zenger. A combination technique for the solution of sparse grid problems, in *Iterative methods in linear algebra. Journal of Approximation Theory*, pages 263–281, 1992.

[71] V. Maźya and G. Schmidt. On approximate approximations using gaussian kernels. *IMA Jornal of Numerical Analaysis*, 16:13–29, 1996.

[72] V. Maźya and G. Schmidt. On quasi-interpolation with non-uniformly distributed centers on domains and manifolds. *Jornal of Approximation Theory*, 110:125–145, 2001.

[73] V. Mazya and G. Schmidt. *Approximate Approximations*. 2007. Providence.

[74] L. Mei and P. Cheng. Multivariate option pricing using quasi-interpolation based on radial basis functions. *Advanced Intelligent Computing Theories and Applications*, pages 620–627, 2008.

[75] C. A. Micchelli. Interpolation of scattered data: distance matrices and conditionally positive definite functions. *Constructive Approximation*, 2(1):11–22, 1986.

[76] G. N. Milstein and M. V. Tretyakov. *Multilevel Monte Carlo methods and applications to elliptic PDEs with random coeficients*. Scientific Computation,. Springer-Verlag, Berlin, 2004.

[77] F. Müller and W. Varnhorn. Error estimates for approximate approximations with gaussian kernels on compact intervals. *Jornal of Approximation Theory*, 145:171–181, 2007.

[78] F. J. Narcowich, R. Schaback, and J. D. Ward. Multilevel interpolation and approximation. *Applied and Computational Harmonic Analysis*, 7(3):243–261, 1999.

[79] F. Nobile, R. Tempone, and C. G. Webster. A sparse grid stochastic collo- cation method for partial differential equations with random input data. *SIAM Jornal of Numerical Analysis*, 46:2309–2345, 2008.

[80] Y. Ohtake, A. Belyaev, and H.-P. Seidel. 3d scattered data interpolation and approximation with multilevel compactly supported rbfs. *Graphical Models*, 67(3):155–165, 2005.

[81] U. Pettersson, E. Larsson, G. Marcusson, and J. Persson. Improved radial basis function methods for multi-dimensional option pricing. *Journal of Computational and Applied Mathematics*, 222(1):82–93, 2008.

[82] M. J. D. Powell. *The theory of radial basis function approximation in 1990.* Number II. Oxford University Press, New York, 1992.

[83] C. Rabut. An introduction to schoenberg's approximation. *Computers and Mathematics with Applications*, 24:149–175, 1992.

[84] R. Schaback. *Creating surfaces from scattered data using radial basis functions.* Vanderbilt University Press, Nashville, TN, 1995.

[85] I. J. Schoenberg. On certain metric spaces arising from euclidean spaces by a change of metric, and their imbedding in hilbert space. *Annals of Mathematics*, 38(4):787–793, 1937.

[86] A. Schreiber. *The method of Smolyak in multivariate interpolation.* Ph.D. Thesis, der Mathematisch-Naturwissenschaftlichen Fakultaten, der Georg-August-Universitat zu Gottingen, 2000.

[87] S. A. Smolyak. Quadrature and interpolation of formulas for tensor product of certian classes of functions. *Soviet Mathematics Doklady*, 4:240–243, 1963.

[88] S. L. Sobolev and V. L. Vaskevich. *The theory of cubature formulas*, volume 415 of *Mathematics and its Applications,*. Kluwer Academic Publishers Group, Dordrecht, 1997. Translated from the 1996 Russian original and with a foreword by S. S. Kutateladze, Revised by Vaskevich.

[89] F. Subhan. *Multilevel Sparse Kernel-Based Interpolation*. Ph.D. Thesis, University of Leicester, 2011.

[90] V. N. Temlyakov. *Approximation of functions with bounded mixed derivative*. 1989. AMS, 1989.

[91] H. Wendland. *Scattered data approximation*. Cambridge monographs on applied and computational mathematics. Cambridge University Press, Cambridge, 2005.

[92] Z.M. Wu and J.P. Liu. Generalized strang-fix condition for scattered data quasi-interpolation. *Advances in Computational Mathematics*, 23:201–214, 2005.

[93] Z.M. Wu and R. Schaback. Shape preserving properties and convergence of univariate multiquadric quasi- interpolation. *Acta Mathematicae Applicatae Sinica*, 10(1):441–446, 1994.

[94] C. A. Zala and I. Barrodale. Warping aerial photographs to orthomaps using thin plate splines. *Advances in Computational Mathematics*, 11(2-3):211–227, 1999.

[95] C. Zenger. Sparse grids, in *Parallel Algorithms for Partial Differential Equations (Kiel, 1990). Notes on Numerical Fluid Mechanics*, 31:241–251, 1991.