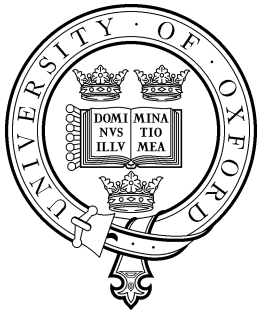
Numerical Linear Algebra of Approximation Involving Radial Basis Functions



Shengxin Zhu

St Anne's College

University of Oxford

A thesis submitted for

Doctor of Philosophy

Trinity Term 2014

To my parents and the Huangs'

Acknowledgements

First of all, I would like to thank my supervisor, Professor Andrew J. Wathen, for his trust, encouragement as well as constructive criticism, in particular, for his great patience and tolerance that he has shown when guiding and coaching me. His constant support and professional training helps me not only to make academic progress but also to shape a better personality.

I would like to express my sincere thanks to these people who helped me in my academic life. Thank Professor Nick L. Trefethen FRS for his *Problem Solving Squad*—the unique training for PhD students in the world, for his lectures (on *Scientific Computing for DPhil Students* and *Spectral Methods*), for his advice and for his *Hey Jude* musical box. Thank Professor Endre Süli and Dr Ian Sobey who acted as my confirmation and transfer examiners respectively and gave me detailed feedback and advice. Special thanks goes to Dr Colin B. Macdonald; he witnessed and examined the whole process of my PhD research, from the recruitment interview, transfer of status and confirmation of status to the final viva. He provided me a lot of advice ranging from research, daily life even to pronunciation. I should also thank Professor Holger Wendland for introducing me *Meshfree Methods*. I am also in debt to Prof. Rosemary Renaut who introduced me the Picard condition.

Thank Professor Alan Goriely and Professor Helen Byrne for their help and support they provided during my DPhil study. Thank Dr Chris Breward for his frequent knocking at doors and reminding us for tea and coffee break. I am appreciated my following colleague friends who helped me with proofreading. They are Dr Laura Kimpton, Dr Sander Rhebergen, Ingrid von Glehn, Georgina Lang, Alexiandex Lewis, Stephen O'Keeffe and Paul Robert. Thanks should also go to all my colleagues from Oxford Centre for Collaborative and Applied Mathematics (OCCAM) and Numerical Analysis (NA) group, from whom I not only obtained peer pressure but also learned a lot. Thanks for the daily help that the support teams in OCCAM, NA and the department have provided.

I should also express my thanks to all of these my Chinese friends at Oxford, in particular, these colleagues in the same groups: Dr K. Xu, Dr W. Xiang, Y. Chen, S. Fang, C. Wang and K. Wei. It is easy and comfortable to stay with them and to

share my joys and sadness. Human is a social animals and likes to stay with peers. Without these friendships, I can not imagine that how hollow and empty I would be, and whether it is possible for me to go through my DPhil.

In the vice-chancellor's words on matriculation, '*your life (at Oxford), your experience, and every people you meet will become parts of your education*'. Every person that I meet here has a unique personality which deserves me to learn from. It will take a tremendously long list if I list all the names. For these names that I list above, it is impossible to list all of what I should thank for.

Thanks for the life, the experience and the people I have met.

Thank Oxford for encouraging people thinking freely, deeply and independently.

Last but not least, I would like to thank Oxford Centre for Collaborative and Applied Mathematics (OCCAM) for the generous OCCAM international studentship. Without the studentship, I cannot carry out my research and finish this thesis. At the same time, I would like to thank the Industry Mathematics Knowledge Transfer Network who supplied me a chance of collaborating with industry. The collaboration was funded by Engineering Physical Science Research Council (EPSRC-IM100852), VSN International Ltd., and the University of Oxford. Thanks for them to let me realize the value of my research and skills.

Shengxin Zhu (朱圣鑫)

Contents

Acknowledgements i

List of Tables v

List of Figures vii

List Algorithms and Programs ix

List of Symbols x

Abstract xiv

I Introduction 1

1 Scattered data interpolation 2

1.1 Problem description 2

1.2 Motivation 2

1.3 Applications 4

1.4 Stance 9

1.5 Outline 11

1.6 Summary of contributions 13

1.7 Notation 14

2 Characterization and construction 15

2.1 Characterization of radial basis functions 15

2.2 Construction of compactly supported RBFs 20

2.3 Relationship between compactly and globally supported RBFs . . . 28

2.4 Notes 30

II Theory **31**

3 Spectral distribution and smoothness mathcing **32**

3.1 Eigenvalue distribution 32

3.2 Smoothness matching 47

3.3 Discussion 55

4 Convexity and solvability of CSRBFs with different shapes **57**

4.1 Accuracy and scalability 57

4.2 Results on convexity 61

4.3 Results on solvability 63

4.4 Convex interval and eigenvalue bound 69

4.5 Discussion 71

III Algorithms **73**

5 Assembly of interpolation matrices **74**

5.1 Check Theorem 4.2 74

5.2 Check Theorem 4.3 76

5.3 Complexity and numerical verification 76

5.4 Discussion 79

6 Fast solvers **84**

6.1 Sparse direct methods 84

6.2 Krylov subspace iterative methods 87

6.3 Near points preconditioning 93

6.4 Methods for dense linear systems 102

6.5 Discussion 106

7 Adaptive Algorithms **108**

7.1 Hierarchical interpolation with CSRBFs: illustration 108

7.2 Heterogeneous hierarchical interpolation on regular mesh 110

7.3 Hierarchical data sets construction 113

7.4 Heterogeneous hierarchical interpolation on scattered data set . . . 119
7.5 Adaptive shape and smoothing parameter selection 122
7.6 Discussion 123

8 Sparse kernel summation on Cartesian grids 125
8.1 Motivation 125
8.2 Data structure and algorithm design 127
8.3 Performance analysis and numerical verification 131
8.4 Discussion 133

IV Applications 136

9 Scattered spatial data analysis 137
9.1 3D object modelling by implicit surface reconstruction 137
9.2 Terrain modelling 139
9.3 Interpolating meteorological data on the earth 143
9.4 Pollution sources identification 144
9.5 Discussion 147

10 Discussion and future work 154
10.1 Error estimate for the adaptive multilevel approach 154
10.2 Smoothness matching and further implementation in 3D 155
10.3 CSRBFs and Strang-Fix condition on sphere 155
10.4 Sparse approximation 155
10.5 Data manipulation 156

Appendix 157

List of Tables

2.1 Some Wendland functions 24
2.2 Some missing Wendland functions 27

3.1 The order of eigenvalues of tensor product Kernel in R^2 44

4.1 Estimations of the convex interval for Wendland functions 69

5.1 Information of test problems from Program 5.4 80
5.2 Information of test problems from Program 5.6 80
5.3 Timing results for checking diagonal dominance condition 81

6.1 Timing results for sparse factorization methods with `amd` ordering . 86
6.2 Timing results for Krylov subspace iterative methods 93
6.3 Iteration numbers for Iterative methods without preconditioning . . 93
6.4 Comparison of the scalability of direct and iterative methods 94
6.5 The decay order of coefficients of cardinal functions of some RBFs . 97
6.6 Storage and timing results for preconditioner ilun(k) for problem D1 100
6.7 Timing results for preconditioned iterative methods with ilun(k) . . 101
6.8 Iteration numbers of iterative methods with preconditioner ilun(k) . 101
6.9 Timing results for preconditioned iterative methods 101
6.10 Storage and construction time for ilun(7) and ilu(0) preconditioners 101

7.1 Comparison of supports between stationary and adaptive approach 111
7.2 A list of test functions 113

8.1 Timing results for sparse kernel summation algorithm 134
8.2 Complexity estimation of fast summation algorithm 134

9.1 Observation information of temperature 144
9.2 Observations of air pollutants in Wuhan, June 14, 2014 148
9.3 Observations of air pollutants in Chongqing, June 14, 2014 148

10.1 Commonly used radial basis functions 157
10.2 Methods and possible future work for commonly used RBFs 157

List of Figures

2.1 Illustration of construction of compactly supported basis functions . 21
2.2 Illustration of construction of the Wendland functions 22
2.3 Comparison between Wendland and missing Wendland functions I . 26
2.4 Comparison between Wendland and missing Wendland functions II 27

3.1 Eigenvalue decay order of Wendland function and Gaussian 41
3.2 Distribution of eigenvalues and interpolation points 44
3.3 Error due to conditioning contaminates approximation quality . . . 48
3.4 Condition number and approximation quality 48
3.5 Four eigenfunctions of an interpolation matrix 52
3.6 The Picard plot 53

4.1 Illustration of of a scaled neighbourhood 64
4.2 Illustration of different supports for non-uniform grid 65
4.3 Two examples satisfying the diagonally dominant condition 67
4.4 Estimation of the convex interval 70

5.1 Data sets, matrix structure, and eigenvalues for the `TriDisc` problems 78

6.1 The fill-reducing ordering and its effects 85
6.2 The ordering effect for sparse factorization method 86
6.3 Illustration of local cardinal function selection 97
6.4 Convergence history related to sparse approximate inverse 98
6.5 Convergence history of (preconditioned) iterative methods 100

7.1 Illustration of frequency information of multilevel interpolation . . . 109
7.2 The radii information of CSRBFs in two multilevel approaches . . . 112
7.3 Convergence rate for multilevel methods on test functions F1 to F4 114
7.4 Convergence rate for multilevel methods on test functions F5 to F8 115

7.5 Convergence rate for multilevel methods on test functions F9 to F12 116
7.6 Hierarchical data sets of cities in Colorado,USA. 120
7.7 Applications and scablability of `hscatter2grid` 120
7.8 Multilevel stationary interpolation on scattered data sets 122
7.9 Cost functions for choosing best shape parameters 124

8.1 A gridbox data structure 127
8.2 Timing results for sparse kernel summation algorithm 135

9.1 Illustration of reconstruction of a heart curve with CSRBFs 138
9.2 Reconstruction of the Stanford bunny from point clouds 140
9.3 Terrain of Colorado, USA 141
9.4 Hierarchical data sets of cites in mainland of China. 142
9.5 Terrain of mainland of China. 142
9.6 The temperature maps for UK. 143
9.7 Temperature and pressure maps for the world on June 22, 2014 . . 145
9.8 Density distribution of air pollutants in Wuhan, June 14, 2014. . . . 149
9.9 Density distribution of air pollutants in Chongqing, June 14, 2014 . 150
9.10 Density distribution of PM2.5 and PM10 in China I 151
9.11 Density distribution of PM2.5 and PM10 in China II 152
9.12 Density distribution of heavy metals in soil of a city 153

List Algorithms and Programs

2.1 A Maple program to compute the Wendland functions 25
2.2 A Maple program to compute the missing Wendland functions . . . 28
5.1 A Matlab program to check Theorem 4.2 75
5.2 An algorithm to check Theorem 4.2 75
5.3 Adaptive algorithm to check diagonal dominance conditions 77
5.4 A Matlab function to generate the interpolation matrix 77
5.5 A C program for Algorithm 5.2 82
5.6 A C program for Algorithm 5.3 83
6.1 GMRES 89
6.2 The Lanczos biorthogonalization procedure 91
6.3 Biconjugate Gradient Methods 91
6.4 CGS 92
6.5 BiCGSTAB 92
6.6 Sparse approximate inverse by local cardinal function approximation 98
6.7 ilun(k):local incomplete LU factorization 100
6.8 Iterative method based on approximate Lagrange functions 104
7.1 An algorithm for multilevel interpolation 110
7.2 HHRBF: heterogeneous hierarchical interpolation with CSRBFs . . 111
7.3 An algorithm for constructing hierarchical Cartesian grids 117
7.4 `Hcgrid`: a Matlab program to construct hierarchical Cartesian grids 118
7.5 Constructing hierarchical scattered data sets by grid mapping . . . 119
7.6 `hscatter2grid`: hierarchical scattered data set construction 121
8.1 The subbox evaluation algorithm 130
8.2 The gridbox evaluation algorithm 130
8.3 The subbox modification algorithm 131
8.4 The gridbox modification algorithm 131
9.1 A framework for 3D surface reconstruction with CSRBFs 139

Notation

Numbers and Sets

$\mathbb{Z}$	The set of integers
$\mathbb{Z}^d$	$\mathbb{Z}^d := \{(z_1, \ldots, z_d) : z_i \in \mathbb{Z} \text{ for } 1 \leq i \leq d\}$
i	An integer or the imaginary unit
j, k, m, n, ℓ	Index integers
$\mathbf{k}, \mathbf{j}$	$\mathbf{k}, \mathbf{j} \in \mathbb{Z}^d$
α, β, γ	Real or complex numbers
$\bar{\alpha}$	The conjugate of a complex number
ϵ	A real number, usually for error or noise

Points, Vectors, Fields and Vector Spaces

$\mathbb{R}$	The real field
$\mathbb{R}^d$	The d dimensional real space
$\mathbb{R}^N$	The space of n-dimensional real vectors
$\mathbb{C}$	The complex field
$\mathbb{C}^N$	The space of n-dimensional complex vectors
x, y	Points in $\mathbb{R}$
$\mathbf{x}, \mathbf{y}$	Points in $\mathbb{R}^d$, $\mathbf{x} = (x_1, x_2, \ldots, x_d)$
$\mathbf{x}_i, \mathbf{x}_j, \mathbf{x}_k$	Numbered points in $\mathbb{R}^d$
$\mathbf{n}_i$	Normal direction at point $\mathbf{x}_i$
$\mathbf{x}^{\pm}$	Points outside $(+)$ or in $(-)$ an isosurface
$\underline{\alpha}$	$\underline{\alpha} = (\alpha_1, \alpha_2, \ldots, \alpha_N)^T \in \mathbb{R}^N$ or $\mathbb{C}^N$
$\underline{\alpha}^T$	The transpose of real vector $\underline{\alpha}$
$\underline{\alpha}^H$	The conjugate transpose of complex vector $\underline{\alpha}$

Point Sets and Related

$\mathcal{X}$	A set of distinct points $\mathcal{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\}$
$\mathcal{Y}$	A set of distinct points $\mathcal{Y} = \{\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_m\}$
$\mathcal{X}^{[-k]}$	$\mathcal{X}/\{\mathbf{x}_k\}$=$\{\mathbf{x}_1, \ldots, \mathbf{x}_{k-1}, \mathbf{x}_{k+1}, \ldots, \mathbf{x}_n\}$
$\mathcal{N}(\mathbf{x}_j, n_j)$	The n_j nearest neighbouring points to $\mathbf{x}_j$
r_j	$r_j = \min_{\mathbf{x} \in \mathcal{N}(\mathbf{x}_j, n_j)} \|\mathbf{x} - \mathbf{x}_j\|$
R_j	$R_j = \max_{\mathbf{x} \in \mathcal{N}(\mathbf{x}_j, n_j)} \|\mathbf{x} - \mathbf{x}_j\|$
q_j	$q_j = r_j / R_j$

m_j	$m_j = \sum_{\mathbf{x}_k \in \mathcal{N}(\mathbf{x}_j, n_j)} \|\mathbf{x}_k - \mathbf{x}_k\|/n_j R_j$
$\mathcal{B}(\mathbf{x}_j, \rho)$	A ball with center $\mathbf{x}_j$ and radius ρ
$h_{\mathcal{X},\Omega}$	The fill distance $h_{\mathcal{X},\Omega} = \sup_{\mathbf{x}\in\Omega} \min_{1\le j\le N} \|\mathbf{x} - \mathbf{x}_j\|_2$
$q_{\mathcal{X}}$	The separation distance $q_{\mathcal{X}} := \frac{1}{2}\min_{i\neq j} \|\mathbf{x}_i - \mathbf{x}_j\|$
Ω	A domain of in a field
$\partial\Omega$	Boundary of a domain Ω

Functions and Related

$s(\mathbf{x})$	An interpolant
ϕ, φ	Functions
$\phi(r)$	A univariate function on r
ϕ_ε	$\phi_\varepsilon(r) = \phi(\varepsilon r)$, a scaled function
$(\cdot)_+$	The truncation function $(\cdot)_+ = \max\{\cdot, 0\}$
$\lfloor\cdot\rfloor$	The floor function
$\lceil\cdot\rceil$	The ceiling function
$(1-r)_+$	The hat function
$\phi_\ell(r)$	The ℓth power of hat function $\phi_\ell(r) = (1-r)_+^\ell$
$\phi_{d,k}$	Wendland functions
$\mathcal{I}$	$\mathcal{I}\phi(r) = \int_r^\infty t\phi(t)dt$
$\mathcal{D}$	$\mathcal{D}(\phi) = -\frac{1}{r}\phi'(r)$
$K(\mathbf{x}, \mathbf{y})$	A kernel function
$\mathcal{K}$	An integral operator $\mathcal{K}\phi(\mathbf{x}) = \int_\Omega K(\mathbf{x}, \mathbf{y})\phi(\mathbf{y})d\mathbf{y}$
$\star$	The convolution operator
$T_k(x)$	The k-th Chebyshev polynomial
$\hat{u}_n$	The n-the Fourier coefficient of u
e^t	The exponential function
$\mathcal{O}$, o	The big O notation and small o notation
$C^k(\mathbb{R}^d)$	The set of k times continuously differentiable functions on Ω.
$L_1(\Omega)$	The space of integrable functions
$\|f\|_{L_p(\Omega)}$	$\|f\|_{L_p(\Omega)} := (\int_\Omega \|f(\mathbf{x})\|^p d\mathbf{x})^{1/p}$
$L_p(\Omega)$	$L_p(\Omega) = \{f : \|f\|_{L_p(\Omega)} < \infty\}$
$\mathcal{H}^s(\mathbb{R}^d)$	$\mathcal{H}^s(\mathbb{R}^d) = \{f \in L_2(\mathbb{R}^d) : \hat{f}(\cdot)(1 + \|\cdot\|^2)^{s/2} \in L_2(\mathbb{R}^d)\}$
$\pi_{k-1}(\mathbb{R}^d)$	The space of d-variate polynomials of degree at most $k-1$

$\mathcal{P}_k$	The set of conditional positive definite functions or order k
$\mathcal{CN}_k$	The set of conditional negative definite functions of order k
$\mathcal{CP}_k$	The set of conditionally positive definite functions

Matrices and Related

$\boldsymbol{\Delta}_{\mathcal{X},\mathcal{X}}$	The point-wise distance matrix for set $\mathcal{X}$, $\boldsymbol{\Delta} = (\|\mathbf{x}_i - \mathbf{x}_j\|_2)_{1\leq i,j\leq n}$
$\boldsymbol{\Delta}_{\mathcal{Y},\mathcal{X}}$	The distance matrix from set $\mathcal{Y}$ to $\mathcal{X}$, $\boldsymbol{\Delta} = (\|\mathbf{y}_i - \mathbf{x}_j\|_2)_{1\leq i\leq m,1\leq j\leq n}$
$\boldsymbol{\Delta}$	The distance matrix the same as $\boldsymbol{\Delta}_{\mathcal{X},\mathcal{X}}$
$\boldsymbol{\Delta}^k$	The element-wise power of a distance matrix $\boldsymbol{\Delta} = (\|\mathbf{x}_i - \mathbf{x}_j\|_2^k)_{1\leq i,j\leq n}$
$\phi(\boldsymbol{\Delta})$	Apply ϕ on each element of $\boldsymbol{\Delta}$, i.e. $\phi(\boldsymbol{\Delta}) = (\phi(\|\mathbf{x}_i - \mathbf{x}_j\|))_{1\leq i,j\leq n}$
$A_{\phi,\mathcal{X}}$	An interpolation matrix on $\mathcal{X}$ with radial function ϕ, stress ϕ
$A_{\mathcal{X},\mathcal{X}}$	An interpolation matrix on $\mathcal{X}$
$A_{\phi_{\varepsilon_j},\mathcal{X}}$	An interpolation matrix on $\mathcal{X}$ with radial function with various scales
$A_{\mathcal{Y},\mathcal{X}}$	An evaluation matrix on points $\mathcal{Y}$ with interpolation centres $\mathcal{X}$
A, B, C	General matrices
U, V	Orthogonal matrices
σ_i, λ_i	Singular values and eigenvalues
S, Λ	Diagonal matrices $S = \text{diag}\{\sigma_1, \ldots, \sigma_n\}$, $\Lambda = \text{diag}\{\lambda_1, \ldots, \lambda_n\}$
V^T	The transpose of V
$A = USV^T$	The singular value decomposition
$A = X\Lambda X^T$	The spectral decomposition for symmetric matrices
dim	The dimensional operator
ker	The kernel operator
$\mathcal{K}_m(A, \underline{\mathrm{r}})$	The Krylov space $\text{span}\{\underline{\mathrm{r}}, A\underline{\mathrm{r}}, \ldots A^{m-1}\underline{\mathrm{r}}\}$

Numerical Linear Algebra of Approximation Involving Radial Basis Functions

Shengxin Zhu

St Anne's College

Doctor of Philosophy

Trinity Term 2014

Abstract

This thesis aims to acquire, deepen and promote understanding of computing techniques for high dimensional scattered data approximation with radial basis functions. The main contributions of this thesis include sufficient conditions for the solvability of compactly supported radial basis functions with different shapes, near points preconditioning techniques for high dimensional interpolation systems with compactly supported radial basis functions, a heterogeneous hierarchical radial basis function interpolation scheme, which allows compactly supported radial basis functions of different shapes at the same level, an $\mathcal{O}(N)$ algorithm for constructing hierarchical scattered data set and an $\mathcal{O}(N)$ algorithm for sparse kernel summation on Cartesian grids. Besides the main contributions, we also investigate the eigenvalue distribution of interpolation matrices related to radial basis functions, and propose a concept of smoothness matching. We look at the problem from different perspectives, giving a systematic and concise description of other relevant theoretical results and numerical techniques. These results are interesting in themselves and become more interesting when placed in the context of the bigger picture. Finally, we solve several real-world problems. Presented applications include 3D implicit surface reconstruction, terrain modeling, high dimensional meteorological data approximation on the earth and scattered spatial environmental data approximation.

Part I

Introduction

1. Scattered data interpolation

1.1 Problem description

Given a set of points, $\mathcal{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N\}$, in $\mathbb{R}^d$, $d \geq 2$, and corresponding function values or observations, f_j, $1 \leq j \leq N$, finding a function, $s(\mathbf{x})$, which is a combination of simple basis functions of the form $s(x) = \sum_{j=1}^{N} \alpha_j \phi_j(\mathbf{x})$ to interpolate or approximate $f(\mathbf{x})$ is a fundamental problem. For simplicity, we shall first focus on the interpolation problem, the case when $s(\mathbf{x}_i) = f_i$, for $1 \leq i \leq N$. For a given set of basis functions, we can determine the weights, α_j, by solving the following linear system

$$\begin{pmatrix} \phi_1(\mathbf{x}_1) & \phi_2(\mathbf{x}_1) & \cdots & \phi_N(\mathbf{x}_1) \\ \phi_1(\mathbf{x}_2) & \phi_2(\mathbf{x}_2) & \cdots & \phi_N(\mathbf{x}_2) \\ \vdots & \vdots & \ddots & \vdots \\ \phi_1(\mathbf{x}_N) & \phi_2(\mathbf{x}_N) & \cdots & \phi_N(\mathbf{x}_N) \end{pmatrix} \begin{pmatrix} \alpha_1 \\ \alpha_2 \\ \vdots \\ \alpha_N \end{pmatrix} = \begin{pmatrix} f_1 \\ f_2 \\ \vdots \\ f_N \end{pmatrix}. \tag{1.1}$$

In short, we denote the linear system as $A\underline{\alpha} = \underline{\mathrm{f}}$, where $\underline{\alpha} = (\alpha_1, \ldots, \alpha_N)^T$ and $\underline{\mathrm{f}} = (f_1, \ldots, f_N)^T$. Once the weights, $\underline{\alpha}$, have been determined, an immediate question is how to economically predict or evaluate $s(\mathbf{x})$ on a new point. Such problems are fundamental and involve many applications as we shall soon see. As such, they deserve close study, as we seek to understand the properties of the underlying problems, to find potentially useful traits and to design efficient algorithms.

1.2 Motivation

One challenge of these problems lies in their high dimensionality, which not only becomes cumbersome, in terms of storage and computing, but also requires us to think from an unconventional perspective. For example, consider the basic problem of how to choose proper basis functions in $\mathbb{R}$ and $\mathbb{R}^d$, $d > 1$, respectively. In one dimensional space, commonly-used basis functions come from the polynomial space of degree at most $n-1$. We can, for example, choose $\phi_j(x) = x^{j-1}$, $j = 1, \ldots, n$. If

the n interpolation points are distinct, then the linear system (1.1) always has an unique solution, since it is a non-singular Vandermonde linear system. Whereas, in high dimensional spaces, the Mairhuber-Curtis theorem [208, p.19][141][142] illustrates that uniqueness of the solution to (1.1) cannot always be guaranteed for multi-variate polynomial interpolation. Such an uncertainty was possibly first noted and proved by Haar [102][141, p.610]. He pointed out that the linear system (1.1) can be singular, even for some distinct points in $\mathbb{R}^d$, when $d > 1$. His arguments are based on the following basic facts of linear algebra: (a) uniqueness of the solution to (1.1) is equivalent to the determinant of the interpolation matrix being non-zero; (b) the determinant of a matrix is a continuous function of its elements; and (c) switching two rows of a matrix will change the sign of its determinant. Based on these facts, one can find two points, say, $\mathbf{x}_1, \mathbf{x}_2$ and construct two distinct curves $\xi_1(t), \xi_2(t)$ which connect these two points such that $\xi_1(0) = \mathbf{x}_1, \xi_1(1) = \mathbf{x}_2$, $\xi_2(0) = \mathbf{x}_2, \xi_2(1) = \mathbf{x}_1$, where the two curves have no other common points and do not intersect with the remaining $n-2$ interpolation points. When $\mathbf{x}_1$ goes along $\xi_1(t)$ to $\mathbf{x}_2$ and $\mathbf{x}_2$ goes along $\xi_2(t)$ to $\mathbf{x}_1$, the first two rows in (1.1) change continuously, and finally when $t = 1$, $\mathbf{x}_1$ and $\mathbf{x}_2$ get switched. Therefore, the determinant of the matrix continuously changes and finally changes sign, and thus there must be some $t \in [0, 1]$ such that the determinant is zero. Such an uncertainty on uniqueness of multivariate polynomial interpolation in high dimensional space differs from that of univariate polynomial interpolation. It motivated people to find non-polynomial basis functions.

If we choose $\phi_j(\mathbf{x}) = \phi(\mathbf{x} - \mathbf{x}_j)$ for a function $\phi : \mathbb{R}^d \mapsto \mathbb{R}$, then when we switch two rows in the interpolation matrix, two columns (two basis functions) will also be switched. Therefore, the determinant of the interpolation matrix will keep the same sign. Such basis functions have the potential to avoid a singularity of the linear system (1.1). Perhaps the simplest such basis function is $\phi(\mathbf{x}) = \|\mathbf{x}\|_2 = \sqrt{x_1^2 + x_2^2 + \cdots + x_d^2}$, which has *radial symmetry*. In this case $\phi_j(\mathbf{x}) = \|\mathbf{x} - \mathbf{x}_j\|_2$ and the interpolation matrix is a distance matrix in $\mathbb{R}^d$. A distance matrix for n distinct points is always invertible, this result was proved by Schoenberg in [182]. Precisely, a distance matrix has one positive eigenvalue and $n-1$ negative eigenvalues, if the n points are distinct [183, p.792]. Such matrices with only one positive eigenvalue and the other eigenvalues negative are said to be *almost negative definite*. Schoenberg's

work was motivated by examining whether a given n numbers in $\mathbb{R}^d$ can serve as the lengths of edges of a simplex, a generalization of the problem of whether three real numbers can serve as the lengths of edges of a triangle in a plane (in $\mathbb{R}^2$ a simplex is a triangle). This seemingly abstract mathematical result before long became the starting point and a powerful tool for one of the most important papers on radial basis functions by Micchelli [148]. Micchelli's paper was motivated by proving a conjecture proposed by Franke who did pioneering work in the area of scattered data approximation. The conjecture can be interpreted as the invertibility of the interpolation matrix in (1.1) with $\phi_j(\mathbf{x}) = \sqrt{1 + \|\mathbf{x} - \mathbf{x}_j\|_2}$. The basis function named as *multiquadric* had been successfully used in practice for military map making and other scattered data approximation [91, 105, 106]. In the paper [148], Micchelli also proved that a class of *radial basis functions* can always guarantee invertible interpolation matrices in $\mathbb{R}^d$ provided that the n points are distinct. Such results, which avoid singular interpolation matrices for high dimensional scattered data approximation, are so encouraging that many high dimensional applications use radial basis functions as a promising tool.

1.3 Applications

Theoretical research on radial basis functions and various applications cross-fertilize each other. With the increasing demand for solving high dimensional problems and the advance of research on radial basis functions, applications which involve radial basis functions are soaring. Here we only describe some representative applications.

1.3.1 Surface reconstruction

An explicit surface is the *graph* of a function of two variables. The graph of a function $f : \Omega \mapsto \mathbb{R}$ is the subset of $\mathbb{R}^{d+1}$ defined by

$$\text{graph}\, f = \{(\mathbf{x}, f(\mathbf{x})) \in \mathbb{R}^{d+1} : \mathbf{x} \in \Omega \subset \mathbb{R}^d\}. \tag{1.2}$$

Here we call graph f a $d+1$ (dimension) graph. With a graph available, one can find its *level sets*. A level set $f^{-1}(\gamma)$, for each real number γ, is defined by

$$f^{-1}(\gamma) = \{\mathbf{x} \in \Omega : f(\mathbf{x}) = \gamma\}. \tag{1.3}$$

A level set of a function of two variables is a contour. Contours can be used to visualize terrain as a two dimensional map [46]. Contour maps are useful in evaluation and interpretation of geophysical data. Traditionally, 2D contour maps were constructed from surveying data, at a digital age, 3D terrain are digitized by reconstructing a 3D graph from existing 2D contour maps. A level set of a function of three variables is also called an implicit surface. Implicit surfaces can be used to visualize 3D objects. The essence of surface reconstruction from a set of observations is a multivariate interpolation problem.

Multiquadrics became popular in the area of terrain modelling and other irregular surface reconstruction in the 1970s [105]: a topography and other irregular surfaces are assumed to consist of multiple *local quadratic surfaces*, i.e., each ϕ_i in $s(\mathbf{x}) = \sum_{i=1}^{N} \alpha_i \phi_i(\mathbf{x})$ is a local *quadratic hyperboloid surface* at (x_i, y_j)

$$\phi_i(\mathbf{x}) = \sqrt{\|\mathbf{x} - \mathbf{x}_i\| + c^2} = \sqrt{(x - x_i)^2 + (y - y_i)^2 + c^2}. \tag{1.4}$$

Here we assume $\mathbf{x} = (x, y) \in \mathbb{R}^2$. Alternative quadratic surfaces were also considered as the local surface ϕ_i at the beginning, for example, the *quadratic cone surface* and the *quadratic paraboloid surface*

$$\phi_i(\mathbf{x}) = \sqrt{(x - x_i)^2 + (y - y_i)^2} \text{ and } \phi_i(\mathbf{x}) = \alpha_i \left[(x - x_i)^2 + (y - y_i)^2\right]. \tag{1.5}$$

They are not preferred because they bring in interpolation matrices with zero diagonal elements and the latter case can encounter numerically singular interpolation matrices in practice [131], we shall see this more clearly in Section 2.1. In modern usage, the term multiquadric is usually used for radial basis functions of the form $\sqrt{r^2 + c^2}$, where $r = \|\mathbf{x} - \cdot\|$.

The multiquadrics were proposed to achieve a high order approximation scheme for relatively high quality terrain modelling. As Kansa said in a review paper [107]

> *The method resulted from the frustration of trying to use various trigonometric and polynomial series to represent topography from sparse, scattered data, of the type collected and manually interpolated by skilled field topographers to produce contours.*

In an extended numerical study [90, p.81][91], Franke concludes that multiquadrics

are the most impressive method among the global methods used in [90, 91]. The spectral convergence of interpolation with multiquadrics was proved in [28, 140]. The success of multiquadric interpolation in surface reconstruction led to multitude applications, for example, in the study of vertical crustal movements [174], in estimations of areal rainfall [47, 131], and in the investigation of gradient estimation from scattered data [190]. It was observed in [190] that multiquadric performs well in gradient approximation, in particular, it performs very well in sharp gradient regions. A steep gradient region is of great interest in many applications, for example, steep gradients in a temperature field can be used to locate the position of cold and warm fronts where rain usually falls. Such quality of gradient approximation with multiquadrics led to a new method for solving partial differential equations.

1.3.2 Solving partial differential equations

1.3.2.1 Unsymmetrical collocation method

Inspired by the high quality surface reconstruction and gradient approximation with multiquadrics, Kansa in [119] applied an *enhanced* multiquadric scheme to a wave propagation problem; enhanced in the way that a *shear transform* was used to remap nearly correlated data points to improve the conditioning. Multiquadrics were applied to design a high order arbitrary Lagrangian-Eulerian (ALE) remapping scheme for a two dimensional problem. Monotonicity, convexity and high accuracy of the scheme were reported by Kansa. Such good properties of an ALE remapping scheme for a two dimensional problem at that time even now must receive a lot of attention. In a subsequent paper, Kansa proposed schemes for using multiquadrics to solve parabolic, hyperbolic and elliptic partial differential equations [120]. The proposed schemes were exemplified by an one dimensional linear advection-diffusion problem with small diffusion rate, a dynamic one-dimensional von Neuman blast wave problem and a two dimensional elliptic problem [120]. Their applicability to such challenging problems made the schemes attractive, whereas few people use the exact schemes that Kansa used in [120] and repeat his examples. For example, for the elliptic problem, Kansa used a combination of two multiquadrics, $\phi(\|\mathbf{x}-\mathbf{x}_j\|)-\phi(\|\mathbf{x}-\mathbf{x}_1\|)$, as the j-th basis function [120, eq.71b]. The later so called Kansa's unsymmetrical collocation method has been simplified and exemplified by

the following linear elliptic partial differential equation (PDE) of the form

$$\mathcal{L}u(\mathbf{x}) = f(\mathbf{x}), \mathbf{x} \in \Omega; \tag{1.6a}$$

$$u(\mathbf{x}) = g(\mathbf{x}), \mathbf{x} \in \partial\Omega. \tag{1.6b}$$

The solution to the above equations is approximated by a sum of translates of a radial basis function $\phi(\|\mathbf{x}\|)$ at scattered data set $\mathcal{X} \subset \Omega \bigcup \partial\Omega$:

$$\hat{u}(\mathbf{x}) = \sum_{i=1}^{N} \alpha_j \phi(\|\mathbf{x} - \mathbf{x}_j\|) + p(\mathbf{x}), \tag{1.7}$$

where $p(\mathbf{x})$ is a low order polynomial, usually of order 0, or a constant [212]. For simplicity, we ignore the low order polynomial term in the following discussion. The scattered data set $\mathcal{X}$ consists of two parts: $\mathcal{X} = \mathcal{X}_\Omega \bigcup \mathcal{X}_{\partial\Omega}$, where $\mathcal{X}_\Omega \subset \Omega$ and $\mathcal{X}_{\partial\Omega} \subset \Omega$. To solve the PDE above, the approximate solution $\hat{u}(\mathbf{x})$ is forced to satisfy (1.6a) and (1.6b). This leads to an unsymmetrical collocation matrix A, whose elements satisfy

$$a_{i,j} = \begin{cases} \mathcal{L}\phi_j(\|\mathbf{x}_i - \mathbf{x}_j\|), & \mathbf{x}_i \in \mathcal{X}_\Omega, \mathbf{x}_j \in \mathcal{X}; \\ \phi_j(\|\mathbf{x}_i - \mathbf{x}_j\|), & \mathbf{x}_i \in \mathcal{X}_{\partial\Omega}, \mathbf{x}_j \in \mathcal{X}. \end{cases} \tag{1.8}$$

The coefficients α_j in (1.7) can be determined by solving the linear system $A\underline{\alpha} = (\underline{\mathrm{f}}^T, \mathrm{g}^T)^T$. A symmetric collocation matrix for solving (1.6a) and (1.6b) can be obtained by adjusting the choice of basis functions.

1.3.2.2 Symmetric collocation method

The *symmetric collocation* method applies the following basis functions to approximate the solution

$$\hat{u}(\mathbf{x}) = \sum_{\mathbf{x}_j \in \mathcal{X}_\Omega} \alpha_j \mathcal{L}_1 \phi_j(\|\mathbf{x} - \mathbf{x}_j\|) + \sum_{\mathbf{x}_j \in \mathcal{X}_{\partial\Omega}} \alpha_j \mathcal{L}_2 \phi_j(\|\mathbf{x} - \mathbf{x}_j\|), \tag{1.9}$$

where $\mathcal{L}_i$, $i = 1, 2$ are certain differential operators. Forcing $\hat{u}(\mathbf{x})$ to satisfy the equation (1.6a) and (1.6b), we can obtain a collocation matrix

$$\begin{array}{c} \\ \mathbf{x}_i \in \mathcal{X}_\Omega \\ \mathbf{x}_i \in \mathcal{X}_{\partial\Omega} \end{array} \begin{array}{c} \begin{array}{cc} \mathbf{x}_j \in \mathcal{X}_\Omega & \mathbf{x}_j \in \mathcal{X}_{\partial\Omega} \end{array} \\ \begin{pmatrix} \mathcal{L}\mathcal{L}_1\phi_j(\|\mathbf{x}_i - \mathbf{x}_j\|) & \mathcal{L}\mathcal{L}_2\phi_j(\|\mathbf{x}_i - \mathbf{x}_j\|) \\ \mathcal{L}_1\phi_j(\|\mathbf{x}_i - \mathbf{x}_j\|) & \mathcal{L}_2\phi_j(\|\mathbf{x}_i - \mathbf{x}_j\|) \end{pmatrix} \end{array} = A. \tag{1.10}$$

If we choose $\mathcal{L}_1 = \mathcal{L}$ and $\mathcal{L}_2$ as the identity operator, for the Dirichlet Problem, then the collocation matrix is symmetric. Such a method is based on the Hermite-Birkhoff interpolation [212]. The Hermite-Birkhoff interpolation is a generalized Hermite interpolation problem of finding a function which interpolates n distinct pairs $(\mathcal{L}_j, x_j, f_j)$, where $\mathcal{L}_i$ is a functional. For example, a special case of Hermite-like interpolation with missing data where the derivatives information need not necessarily be consecutive.

Radial basis functions can also be used to design high order finite difference schemes and pseudospectral methods on scattered nodes [76, 82, 186, 211]. Many applications are based on methods for solving PDEs with RBFs, for example, computer-aided design, mesh repair [143], and image registration [89].

1.3.3 Probability density function estimation

Probability density function estimation is the process of constructing an estimator, $\tilde{f}(x)$, to $f(x)$, based on given observations. It is one fundamental problem in statistics. The unknown probability density function allows calculations of probabilities associated with a random variable, X, as follows

$$P(a < X < b) = \int_a^b f(x)dx, \text{for all } a < b. \tag{1.11}$$

Kernel estimators are commonly used estimators for probability density function estimation. They are approximated as follows. Suppose that a kernel function, K, satisfies

$$\int_{-\infty}^{\infty} K(x)dx = 1, \tag{1.12}$$

then the kernel estimator with the kernel K is defined by

$$\tilde{f}(x) = \frac{1}{nh} \sum_{i=1}^{n} K(\frac{x - X_i}{h}), \tag{1.13}$$

where h is referred to as a *(window) width* or *smoothing parameter* in statistics. Usually, but not necessarily, K will be a radial function, for example the Gaussian, $e^{-\alpha x^2}$ or the compactly supported Epanechnikov kernel [68]

$$K_e(x) = \begin{cases} \frac{3}{4\sqrt{5}} - \frac{3x^2}{20\sqrt{5}} & \text{for } |x| \leq \sqrt{5}, \\ 0 & \text{for } |x| > \sqrt{5}. \end{cases} \tag{1.14}$$

Such an approach, which makes no assumptions about the mean and variance of the probability density function, is referred to as the *non-parametric* approach; it is easy to generalise into higher dimensional spaces. We shall see strategies for choosing the window bandwidth for a relatively accurate density function estimation in the following chapters.

Besides geodesy, geography, hydrology, digital terrain modelling and mesh-free methods for solving PDEs, other application areas includes sampling [166], global optimisation, nonlinear regression, signal processing [1][150], machine learning [110][167], neural networks and artificial intelligence [93][149][155]. These applications are usually based on multivariate non-parametric regression or kernel learning techniques.

1.4 Stance

As the demand of conquering increasingly complicated and high dimensional problems increases with the advancement of computing technologies, any potential method which can deal with real-world problems in high dimensional spaces will receive more attention. The critical aspect of successful problem solving is to choose a proper method for the underlying problem. Proper use of RBFs can solve some complicated real-world problems better or more simply than certain other methods. RBFs are one of the potential tools to handle high dimensional scattered data.

Radial basis functions have received a lot of attention as an elegant approximation scheme for multidimensional scattered data and as an attractive way to solve

partial differential equations. Success stories related to RBFs have been reported from geography and digital terrain modelling, data assimilation in geodesy and meteorology, engineering design and mesh generation, neural networks and artificial intelligence, global optimization, implicit surface construction, sampling, signal processing and machine learning. Based on such a trend and solid mathematical theory, it seems reasonable that the application areas which involve radial basis functions will continuously expand. With the expanding applications, some problems become more common and even universal; it is worthy to acquire, deepen and promote understanding of these problems, customize special fast methods, and supply tools or suggestions when needed.

But we should be realistically optimistic rather than biased, overly optimistic or just blindly following a trend. There are negative results on using radial basis functions which should have received more attention. Firstly, we shall see that designing scalable and convergent interpolation schemes with RBFs is challenging according to the Strang-Fix condition; the Strang-Fix condition can guarantee local polynomial convergence results [192, Thm 2.3] for a scalable interpolation scheme. However, there is no positive definite compactly supported radial basis function which satisfies the Strang-Fix condition in a space of dimension higher than or equal to two [214, Thm 2.3]. Secondly, it is difficult to design a pleasant subdivision scheme for radial basis functions; pleasant in the way that a local modification on some basis functions does not impact on the weights of other basis functions far away. The consequence of these negative results is that radial functions are not particularly useful on grids, on which box splines and tensor splines enjoy a more pleasant subdivision scheme; the advantage of radial basis functions clearly lies on the side of truly scattered data problems, where the question of convenience becomes paramount.

Therefore, we don't argue that RBFs are proper tools for every problem, in fact, they may be inferior to other methods in some cases. They are suggested only when necessary. We focus on developing tools which are potentially useful for the cases when radial basis functions are the best tool for the underlying problems. Because there are monographs on the theoretical discussion of radial basis functions, see [32, 208], here we focus on discussing practical issues which one has to consider when solving a problem with radial basis functions. We prefer to ask questions

from realistic applications and then find solutions rather than to propose a series of assumptions which may diverge from most realistic situations and then derive theoretical results. Theoretical results are of course important and sometimes essential, but in most of places of this thesis, we develop or borrow theory for mathematical insights into the behavior of heuristic algorithms, for constructing algorithms directly and finally for problem solving.

1.5 Outline

This thesis consists of four parts. We first raise the underlying mathematical problems directly and provide the underlying motivations. This is followed by rigorous analysis and practical algorithms. Finally, we solve a couple of real-world applications. As the discussion proceeds, the mathematical flavour will fade away gradually, more practical engineering components will be added, and finally, we shall solve a couple of practical applications.

The first part consists of two introductory chapters. In Chapter 1, we introduce the underlying mathematical problems and glance at some representative applications which motivate ongoing research in this field. We focus on the positive definiteness and sparsity of the underlying interpolation matrices in Chapter 2. Both are desirable properties for large scale computation. A number of established theoretical results related to positive definiteness and sparsity are collected. The aim of this chapter is to include many brief results related to the positivity and sparsity of the underlying interpolation linear system so that new intuitions may be inspired by such connections and comparisons. These results are beautiful in the sense that the results themselves are concise and solve important problems, while sophisticated and advanced mathematics is involved in the detailed proofs. Because of the practical focus of this thesis, only results and their relationships are described. Occasionally, where some technical terminology is unavoidable, we try to be clear and focus on the big picture. Those interested in the detailed proofs are directed to the original references.

Part II consists of two theoretical chapters. We first focus on conditioning issues related to interpolation matrices with radial basis functions. The relationship between the eigenvalue distribution of an interpolation matrix and the smoothness of corresponding radial basis functions is discussed. The results we use are scat-

tered in several unconnected places, mostly in the field of linear integral equations. Theorem 3.9 on the eigenvalue distribution and the geometrical properties of the interpolation points did not appear directly elsewhere. Based on the relationship between the eigenvalue distribution of interpolation matrices and the smoothness of the underlying radial basis function, we propose a smoothness matching concept inspired by the Picard conditioning, which brings insight into several important issues.

In Chapter 4, we focus on a long-standing problem: how to use radial basis functions of different scales at the same time. This problem was proposed and discussed when radial basis functions were first used for scattered data interpolation in the 1970s when few theoretical results on radial basis function interpolation were established, whereas few practical results for large data sets were obtained. Solving the problem is one important step towards adaptivity; adaptivity plays a critical role in reducing the curse of high dimensionality. From this perspective, this is a significant problem for large scale scattered data approximation. In this chapter, we knock at one door of adaptivity and the door slides open. We prove that a class of compactly supported radial basis functions are convex on a certain region. Based on this local convexity and a local geometrical property of underlying interpolation points, we construct sufficient conditions which guarantee diagonally dominant interpolation matrices for compactly supported radial basis functions with various shapes. The proof only requires basic mathematics and well-known linear algebra results, the proof is constructive and can be used to design an algorithm directly. Results in this chapter are completely new.

Part III discusses practical algorithms and consists of 4 chapters. In Chapter 5, we discuss how to assemble interpolation matrices according to the methods described in Chapter 4. Chapter 6 examines fast methods to solve the interpolation linear systems. Standard sparse direct methods and sparse iterative methods are compared, a near point preconditioning technique is customized for Krylov subspace iterative methods. Also discussed in this chapter includes the RBF-QR method and the approximate Lagrangian methods for globally supported radial basis functions.

In Chapter 7, we develop a heterogeneous hierarchical method based on the approach of using compactly supported radial basis functions with different shapes. The proposed hierarchical method has a good scalability and convergence results

are observed. At the same time, we develop $\mathcal{O}(N)$ algorithms for constructing hierarchical scattered data sets, which are required for hierarchical methods on scattered data sets.

Chapter 8 focuses on post-processing and visualisation. We propose a simple and efficient algorithm for sparse kernel summation on Cartesian grids. The algorithm takes advantage of the sparsity of the underlying kernel function and the features of Cartesian grids. It is a scalable algorithm with a complexity of $\mathcal{O}(N)$. Special attention is paid to current computer architectures. Numerical examples for 3D surface reconstruction are presented to illustrated the efficiency of the algorithm.

Finally in Part IV, we demonstrate how to solve a couple of real-world applications in Chapter 9 and discuss potential opportunities for further research in Chapter 10.

1.6 Summary of contributions

The main contributions of this thesis can be summarized as follows:

- We provide sufficient conditions for the solvability of compactly supported radial basis functions with different shape parameters in Chapter 4. The result has been published in *Journal of Scientific Computing* [223].

- We have a good understanding on how the eigenvalue distribution of an interpolation matrix is related to the smoothness of the radial basis functions in Chapter 3. The results on the eigenvalue distribution have been accepted by *Numerical Algorithm* [205].

- We develop a simple fast algorithm for sparse kernel summation on Cartesian grids in Chapter 8. The algorithm is used for post-processing and visualisation. The result have been submitted for possible publication [222].

- We have a good understanding on how the decay of the elements of the inverse of an interpolation matrix is related to the smoothness of the radial basis functions in Chapter 6. This result sheds light on choosing the sparse pattern of a approximate sparse inverse. At the same time, a memory efficient incomplete factorization preconditioning technique is developed in Chapter 6. Results on preconditioning have not been published, we plan to contribute the results to a journal on numerical linear algebra.

- Based on using compactly supported radial basis functions with different shape parameters, we develop a heterogeneous hierarchical radial basis function interpolation scheme in Chapter 7. The scheme aims to achieve a trade-off between two hierarchical approaches in existence. For the moment, let us call them 'Method 1' and 'Method 2' (details will follow in Chapter 7). Method 1 has good scalability in term of storage for the interpolation matrix but does not converge; method 2 converges but it requires more storage. The proposed heterogeneous radial basis functions scheme requires less storage than method 1 and first order convergence in infinity norm and second order convergence in 2-norm are observed on hierarchical Cartesian grids. At the same time, we develop in Chapter 7 an $\mathcal{O}(N)$ algorithm with a high concurrency to construct nested hierarchical scattered data set, which is considerable advance when compared with previously $\mathcal{O}(N \log N)$ algorithm. We are still improving these results for a journal on scientific computing.

1.7 Notation

Frequently used notation is listed on pages xi to xiii. The following notation is reserved in the following chapters. Let $\mathbb{R}^d$ be d-dimensional Euclidean space. Points in $\mathbb{R}^d$ are represented by bold font letters like $\mathbf{x}$, $\mathbf{y}$, $\mathbf{x}_1$ and $\mathbf{x}_k$. Vectors are denoted by letters with an underline like $\underline{\alpha}$, $\underline{\beta}$ and $\underline{\mathrm{f}}$. An interpolation matrix is denote as A, $A_{\phi,\mathcal{X}}$, $A_{\mathcal{X},\mathcal{X}}$, or $A_{\mathcal{X}}$. The interpolation linear system is denoted as $A\underline{\alpha} = \underline{\mathrm{f}}$. By default $\|\cdot\|$ denotes the 2-norm. Other occasionally used notation will be defined locally in discussion.

The reproducing kernel Hilbert space and Sobolev space of order s are frequently refereed in the context of radial basis functions. They are defined as follows.

Let $\mathcal{F}$ be a real Hilbert space of functions $f : \Omega \to \mathbb{R}$. A function $\Phi : \Omega \times \Omega \to \mathbb{R}$ is called a reproducing kernel for $\mathcal{F}$ if (1) $\Phi(\cdot, \mathbf{y}) \in \mathcal{F}$ for all $\mathbf{y} \in \Omega$, (2) $f(\mathbf{y}) = (f, \Phi(\cdot, y))_{\mathcal{F}}$ for all $f \in \mathcal{F}$ and all $\mathbf{y} \in \Omega$. $\mathcal{F}$ is also called a reproducing kernel Hilbert space. The Sobolev space of order s is defined by the decay order of Fourier coefficients $\mathcal{H}^s(\mathbb{R}^d) = \{f \in L_2(\mathbb{R}^d) : \hat{f}(\cdot)(1 + \|\cdot\|^{s/2}) \in L_2(\mathbb{R}^d)\}$.

2. Characterization and construction

Let $\phi_i(\mathbf{x}) = \Phi(\mathbf{x} - \mathbf{x}_i)$ for $i = 1, \ldots, n$ in (1.1), then the interpolation matrix with $\{\phi_1, \ldots, \phi_n\}$ can be written as

$$A = \begin{pmatrix} \Phi(\mathbf{x}_1 - \mathbf{x}_1) & \Phi(\mathbf{x}_1 - \mathbf{x}_2) & \cdots & \Phi(\mathbf{x}_1 - \mathbf{x}_n) \\ \Phi(\mathbf{x}_2 - \mathbf{x}_1) & \Phi(\mathbf{x}_2 - \mathbf{x}_2) & \cdots & \Phi(\mathbf{x}_2 - \mathbf{x}_n) \\ \vdots & \vdots & \ddots & \vdots \\ \Phi(\mathbf{x}_n - \mathbf{x}_1) & \Phi(\mathbf{x}_n - \mathbf{x}_2) & \cdots & \Phi(\mathbf{x}_n - \mathbf{x}_n) \end{pmatrix}. \tag{2.1}$$

A function $\Phi : \mathbb{R}^d \mapsto \mathbb{R}$ is said to be radial, if there exist a function $\varphi : [0, \infty)$ such that $\Phi(x) = \varphi(\|\mathbf{x}\|_2)$ for all $\mathbf{x} \in \mathbb{R}^d$. We characterize a radial basis function in two ways: first whether the resulting interpolation matrix (2.1) is positive definite, and second whether the resulting linear system is sparse. Both are desirable properties for large scale computation. In this chapter, we first briefly introduce how to characterize a radial basis function according to the positivity of its Fourier transform, and then describe how to construct a compactly supported radial basis function with desirable properties, finally we discuss the relationship between some compactly supported functions and globally supported functions.

2.1 Characterization of radial basis functions

A radial basis function Φ is said to be *positive definite*, if the resulting interpolation matrix in (2.1) is positive definite for any n distinct points. Furthermore if Φ has an integrable Fourier transform $\hat{\Phi}$, then Φ is positive definite if the Fourier transform $\hat{\Phi}$ is positive. This can be shown in several lines by using a quadratic form and the Fourier inversion formula [208, p.67]. We outline the proof here:

According to the Fourier inversion formula, it follows that

$$\Phi(\mathbf{x}) = \frac{1}{\sqrt{(2\pi)^d}} \int_{\mathbb{R}^d} \hat{\Phi}(\omega) e^{i\mathbf{x}^T\omega} d\omega. \tag{2.2}$$

On the other hand, for any $\underline{\beta} \in \mathbb{C}^n$, consider the following quadratic form

$$\underline{\beta}^H A \underline{\beta} = \sum_{k=1}^{n}\sum_{j=1}^{n} \beta_j \bar{\beta}_k \Phi(\mathbf{x}_k - \mathbf{x}_j) = \frac{1}{\sqrt{(2\pi)^d}} \sum_{k=1}^{n}\sum_{j=1}^{n} \beta_j \bar{\beta}_k \int_{\mathbb{R}^d} \hat{\Phi}(\omega) e^{i(\mathbf{x}_k - \mathbf{x}_j)^T \omega} d\omega. \tag{2.3}$$

Separate $e^{i(\mathbf{x}_k - \mathbf{x}_j)^T \omega}$ as $e^{i\mathbf{x}_k^T \omega} e^{-i\mathbf{x}_j^T \omega}$, then

$$\sum_{k=1}^{n}\sum_{j=1}^{n} \beta_j \bar{\beta}_k e^{i(\mathbf{x}_k - \mathbf{x}_j)^T \omega} = \sum_{j=1}^{n} \beta_j e^{-i\mathbf{x}_j^T \omega} \sum_{k=1}^{n} \bar{\beta}_k e^{i\mathbf{x}_k^T \omega} = \left\| \sum_{j=1}^{n} \beta_j e^{-i\mathbf{x}_j^T \omega} \right\|^2. \tag{2.4}$$

Therefore, we have

$$\underline{\beta}^H A \underline{\beta} = \frac{1}{\sqrt{(2\pi)^d}} \int_{\mathbb{R}^d} \hat{\Phi}(\omega) \left\| \sum_{j=1}^{n} \beta_j e^{-i\mathbf{x}_j^T \omega} \right\|^2 d\omega. \tag{2.5}$$

If $\hat{\Phi} > 0$, the quadratic form in (2.5) is always positive definite and thus the interpolation matrix A is symmetric and positive definite. One of the most famous positive definite radial basis functions is the Gaussian $e^{-\epsilon^2 \|\mathbf{x}\|_2^2}$, with a positive Fourier transform $\frac{1}{(\sqrt{2}\epsilon)^d} e^{-\frac{\|\omega\|^2}{4\epsilon^2}}$.

Using the Fourier transform to characterize a positive definite function dates back to Mathias [145], Bochner [18][208, p.67], and followed by von Neumann, Schoenberg [203], and Iske [116] among others; and it provides a simple way to verify whether the linear system (1.1) is positive definite or not. We only need to operate on a univariate function to find the Fourier transform for a radial function, as shown in Schaback and Wu's work [181], whereas, generally speaking, finding a multi-variate Fourier transforms is not easy. An alternative way to examine whether a special class of radial basis functions are positive *semi-definite* is to calculate the derivatives of a related univariate function $\varphi(\sqrt{\cdot})$ and examine whether $\varphi(\sqrt{\cdot})$ is *completely monotone*.

A function $h(r)$ is said to be completely monotone on $(0, \infty)$, if $h(r) \in C^{\infty}_{(0,\infty)}$ and its derivatives satisfy $(-1)^{\ell} h^{(\ell)}(r) \geq 0$ for all positive integers ℓ. If further $h(r) \in C^{\infty}_{[0,\infty)}$, it is said to be completely monotone on $[0, \infty)$. Schoenberg pointed out in [184] that a function $h(r)$ is completely monotone on $[0, \infty)$ if and only if $h(r^2)$ is positive semi-definite on every $\mathbb{R}^d$, where $r = \|\mathbf{x}\|$. An example of a radial basis function of this type is the *inverse multiquadric* $(c^2 + r^2)^{-\beta}$. Let $h(r) = (c^2 + r^2)^{-\beta}$,

one can verify

$$(-1)^{\ell} h^{(\ell)}(r) = (-1)^{2\ell} \beta(\beta+1)(\beta+2)\cdots(\beta+\ell-1)(r^2+c^2)^{-\beta-\ell} \geq 0, \qquad (2.6)$$

for any positive integer ℓ [208, p.96].

A natural generalization of a positive definite function is a *conditional (semi-) positive definite function*. A conditional (semi-) positive definite function of order k is a continuous function which guarantees that the quadratic form $\underline{\alpha}^H A \underline{\alpha}$ is (non-negative) positive on a subspace, where $\underline{\alpha} \in \mathbb{C}^n/\{0\}$ satisfies $\sum_{j=1}^n \alpha_j p(\mathbf{x}_j) = 0$ for any distinct point set $\mathcal{X}$ and any polynomials $p(\mathbf{x})$ in $\pi_{k-1}^d(\mathbb{R}^d)$, the polynomial space of degree less than $k-1$. A conditional (semi-) negative definite function of order k can be defined similarly.

For conditional (semi-) positive/negative definite functions of order k, the approximation $s(\mathbf{x})$ is usually used as

$$s(\mathbf{x}) = \sum_{j=1}^{n} \alpha_j \phi_j(\mathbf{x}) + \sum_{k=1}^{Q} \gamma_k p_k(\mathbf{x}), \qquad (2.7)$$

where $\{p_k(\mathbf{x})\}_{k=1}^Q$ is one basis set of the polynomial space $\pi_{k-1}(\mathbb{R}^d)$. Together with the lower order polynomial constraints

$$\sum_{j=1}^{n} \alpha_j p_k(\mathbf{x}_j) = 0, \text{ for } 1 \leq k \leq Q, \qquad (2.8)$$

the approximation (2.7) results in a saddle point system

$$\begin{pmatrix} A & P \\ P^T & 0 \end{pmatrix} \begin{pmatrix} \underline{\alpha} \\ \underline{\gamma} \end{pmatrix} = \begin{pmatrix} \underline{\mathrm{f}} \\ 0 \end{pmatrix}. \qquad (2.9)$$

If P is full rank, and the quadratic form $\underline{\alpha}^H A \underline{\alpha}$ is positive on the null space of P^T, i.e. $\{\underline{\alpha} : P^T \underline{\alpha} = 0\}$, then the saddle point system (2.9) has a unique solution [208, p.117]. When $k = 1$ and the points in $\mathcal{X}$ are distinct, P is always full rank, therefore for conditional positive/negative definite functions of order 1, the saddle points system in (2.9) always has a unique solution. For higher order k, whether P is full rank depends on the geometric properties of the point set $\mathcal{X}$. Well known

conditional negative definite functions of order 1 includes the *multiquadric* radial basis functions $\varphi(r) = \sqrt{1+r^2}$ and the *power function* r^δ, $0 < \delta < 2$. We shall first examine a simple function which is conditional semi-negative function of order 1 by direct computations and then introduce an alternative way to examine a general function.

The simplest conditional semi-negative function of order 1 would be the power function r^2, where $r = \|\cdot\|$, and the elements of the matrix A in (2.1) are $a_{ij} = \|\mathbf{x}_i - \mathbf{x}_j\|^2$, $1 \le i, j \le n$. For any $\underline{\alpha}$ such that $\sum_{i=1}^n \alpha_i = 0$ (choose $p(\mathbf{x}) = 1$, the constant polynomial), it follows that

$$\begin{aligned}
\underline{\alpha}^H A \underline{\alpha} &= \sum_{i=1}^n \sum_{j=1}^n \alpha_j \bar{\alpha}_i \|\mathbf{x}_i - \mathbf{x}_j\|_2^2 = \sum_{i=1}^n \sum_{j=1}^n \bar{\alpha}_i \alpha_j \left(\|\mathbf{x}_i\|_2^2 + \|\mathbf{x}_j\|_2^2 - 2\mathbf{x}_i^T \mathbf{x}_j \right) \\
&= \sum_{i=1}^n \bar{\alpha}_i \|\mathbf{x}_i\|_2^2 \sum_{j=1}^n \alpha_j + \sum_{i=1}^n \bar{\alpha}_i \sum_{j=1}^n \alpha_j \|\mathbf{x}_j\|_2^2 - 2 \sum_{i=1}^n \sum_{j=1}^n \bar{\alpha}_i \alpha_j \mathbf{x}_i^T \mathbf{x}_j \\
&= 0 + 0 - 2 \| \sum_{i=1}^n \alpha_i \mathbf{x}_i \|^2 \le 0.
\end{aligned}$$

Therefore r^2 is a conditional semi-negative definite function of order 1. In this case there is still some possibility that the interpolation matrix A is singular. For example, let $\mathbf{x}_1 = (0,0)^T$, $\mathbf{x}_2 = (0,1)^T$, $\mathbf{x}_3 = (1,1)^T$, $\mathbf{x}_4 = (1,0)^T$, $\underline{\alpha} = (1,-1,1,-1)^T$, then $\sum_{i=1}^4 \alpha_i = 0$, $\sum_{i=1}^4 \alpha_i \mathbf{x}_i = 0$, and

$$A = \begin{pmatrix} 0 & 1 & 2 & 1 \\ 1 & 0 & 1 & 2 \\ 2 & 1 & 0 & 1 \\ 1 & 2 & 1 & 0 \end{pmatrix}$$

is a circulant matrix and its eigenvalues are $-2, -2, 0, 4$. For the case $\varphi(r) = r^\delta$, $0 < \delta < 2$, Schoenberg proved that the corresponding matrix A has $n-1$ negative eigenvalues and 1 positive eigenvalue [183]. An alternative and easy way to examine whether a general radial basis function is conditionally positive/negative is based on a generalization of characterization of a positive definite function, a *completely monotone function* [208, p.85].

Let us denote the class of the conditional positive definite functions of order

k over any $\mathbb{R}^d$ as $\mathcal{P}_k = \bigcap_{d>0} \mathcal{P}_k(\mathbb{R}^d)$. Micchelli in [148, Theorem 2.1] proves that $F(r^2) \in \mathcal{P}_k$ *whenever* $F(r)$ *is continuous on* $[0, \infty)$ *and* $(-1)^k F^{(k)}(r)$ *is completely monotone on* $(0, \infty)$. A positive definite function can be viewed as a conditional positive definite function of order 0 (with zero constraint). For the case of general multiquadrics $\varphi(r) = (-1)^{\lceil \delta \rceil}(c^2 + r^2)^\delta$, $\delta, c > 0$ and $\delta \notin \mathbb{Z}$, let $F_\delta(r) = (-1)^{\lceil \delta \rceil}(c^2 + r)^\delta$, then $F_\delta^{(k)}(r) = (-1)^{\lceil \delta \rceil}\delta(\delta - 1)\cdots(\delta - k + 1)(c^2 + r)^{\delta - k}$. One can verify that

$$(-1)^{\lceil \delta \rceil} F_\delta^{\lceil \delta \rceil}(r) = \delta(\delta - 1)\cdots(\delta - \lceil \delta \rceil + 1)(c^2 + r)^{\delta - \lceil \delta \rceil}$$

is reduced to be an inverse multiquadric, which has been shown to be completely monotone on $(0, \infty)$ in (2.6). Therefore, the *generalized multiquadric* defined above is conditionally positive definite of order $\lceil \delta \rceil$ on $\mathbb{R}^d$. Similarly the power functions $(-1)^{\lceil \delta \rceil} r^\delta$ for $\delta > 0$, $\delta \notin 2\mathbb{Z}$ are conditionally positive definite of order $\lceil \delta/2 \rceil$ on every $\mathbb{R}^d$, and the *thin-plate spline* or *surface splines* $(-1)^{k+1} r^{2k} \log r$ are conditionally positive definite of order $k + 1$ on every $\mathbb{R}^d$. The reader is directed to [208, p.116] for details.

Special attention should be paid to the following radial basis functions, the cubic spline r^3 in $\mathbb{R}$, $r^2(\ln r - 1)$ in $\mathbb{R}^2$, r in $\mathbb{R}^3$, $\ln r$ in $\mathbb{R}^4$, and r^{4-d} in $\mathbb{R}^d$, $d \geq 5$, because they are Green functions of the biharmonic operators [173, p.140, Table 1]. In one and two dimensions, it has been shown that the collocation of such functions at given points has minimum curvature if and only if the underlying basis function is the Green function of the biharmonic operator [23]. For the case in $\mathbb{R}^3$, the multiquadratic is used to avoid numerical singularity [108]. In higher dimensional spaces, the Green functions of biharmonic operators are unbounded at the origin and thus are not very useful.

So far, the radial basis functions that we have mentioned: the Gaussian radial basis functions, multiquadrics, inverse multiquadrics, the power functions, the thin-plate splines and the Green functions of biharmonic operators are referred to as globally supported radial basis functions because they result in dense interpolation matrices. Dense interpolation matrices suffer from poor scalability. For large scale problems, sparsity is a desirable property. The next section will discuss how to construct compactly supported radial basis functions (CSRBFs) which result in sparse interpolation matrices.

2.2 Construction of compactly supported RBFs

It is not difficult to construct compactly supported functions if there are no other requirements such as smoothness and positive definiteness. For example the *truncated power functions*, which are also called *Askey's power functions* [4], given by

$$\phi_\ell(\mathbf{x}) = (1 - \|\mathbf{x}\|_2)_+^\ell = \begin{cases} (1 - \|\mathbf{x}\|_2)^\ell & \text{for } 1 - \|\mathbf{x}\|_2 \le 0; \\ 0 & \text{for } 1 - \|\mathbf{x}\|_2 \ge 0, \end{cases} \tag{2.10}$$

have a compact support in the ball $\|\mathbf{x}\|_2 \le 1$. With dedicated choice of the power ℓ, the truncated power function can be a positive definite function [39][132][208, p.80].

Proposition 2.1. *Askey's truncated power function is positive definite on $\mathbb{R}^d$ if ℓ is an integer and $\ell \ge \lfloor d/2 \rfloor + 1$, where $\lfloor \cdot \rfloor$ is the floor function.*

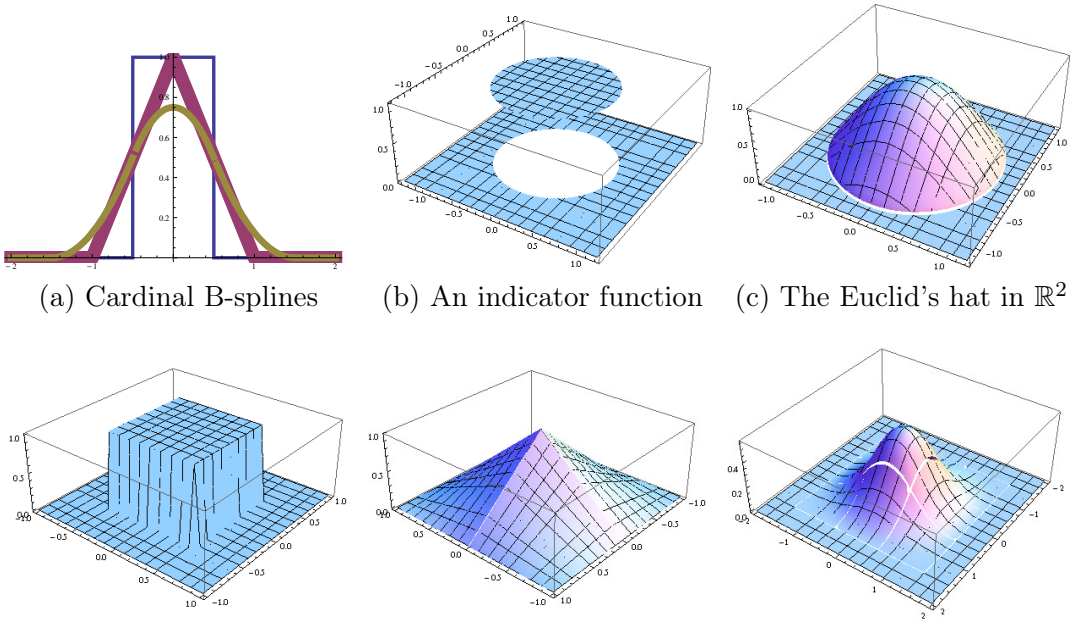
However, the Askey power functions do not have continuous derivatives at $\|\mathbf{x}\|_2 = 0$, even when ℓ is large, see Figure 2.2(a) for illustration. Thus the next question is how to increase the smoothness.

2.2.1 Increasing smoothness

Smoother CSRBFs can be constructed by convolution. The well-known cardinal B-splines [185], box-splines [51] and the *Euclidean hat* functions [44, p.81][98][206] in $\mathbb{R}^d$ are constructed in this way, see Figure 2.1 for illustration. The Euclidean hat function is the self-convolution of the indicator function of a Euclidean ball with diameter 1 in $\mathbb{R}^d$. Smoother CSRBFs than the Euclidean hat can be constructed recursively like the cardinal B-splines. However it turns out that computing convolution in high dimensional space $\mathbb{R}^d$ is not trivial. Wolfram Mathematica shows that the box-spline generated by self-convolution in Figure 2.1(f) is a piecewise polynomial in $\mathbb{R}^2$ (with 17 different polynomials). A simpler method than recursive convolution is needed.

Let us start with a simpler operation than convolution. Consider the Askey power function on $\mathbb{R}$, $\varphi(t) = (1 - |t|)_+ \in C^0$, then

$$\int_{-1}^{x} \varphi(t)\, dt \in C^1 \text{ and } \int_{-1}^{x}\int_{-1}^{s} \varphi(t)\, dtds \in C^2. \tag{2.11}$$



(a) Cardinal B-splines (b) An indicator function (c) The Euclid's hat in $\mathbb{R}^2$

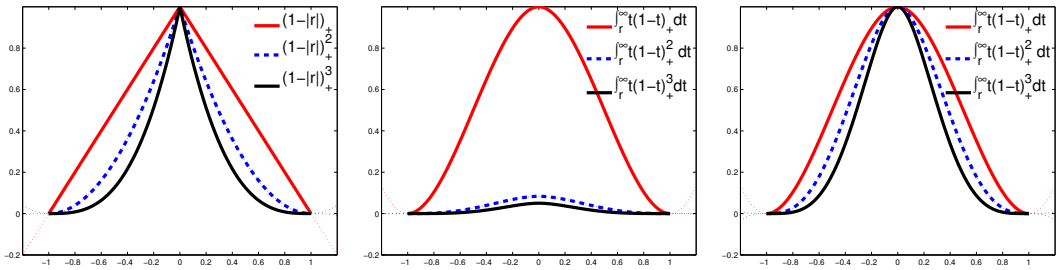
(d) An indicator function (e) A box-spline (f) A box-spline

Figure 2.1: Constructing smoother compactly supported functions by self-convolution. (a) shows the first 3 cardinal B-spline B_k, $k = 0, 1, 2$. B_0 is the indicator function of $[-\frac{1}{2}, \frac{1}{2}]$, which is discontinuous. B_k are defined recursively as the convolution product $B_k := B_0 \star B_{k-1}$, $k = 1, 2, \ldots$. B_k has a compact support on $[-\frac{k+1}{2}, \frac{k+1}{2}]$ and $B_k \in C^{k-1}$. Figures (b) and (d) are indicator functions on a disc and a unit square respectively. Figures (c) and (e) are the self-convolutions of the indicator functions in (b) and (d) respectively. Figure (f) is the convolution of the functions in (d) and (e).

It can be shown that [48, p.6]

$$\int_{-1}^{x}\int_{-1}^{s} \varphi(t)\, dtds = \int_{-1}^{x} x\varphi(t)\, dt - \int_{-1}^{x} t\varphi(t)\, dt. \tag{2.12}$$

Therefore both $\int_a^x x\varphi(t)\, dt$ and $\int_a^x t\varphi(t)\, dt$ have second derivatives and can be viewed twice smoother than $\varphi(t)$. It turns out that a similar integral operator to $\int_{-1}^{x} t\varphi(t)\, dt$ simplifies computations of constructing CSRBFs in higher-dimensional space via *dimension walk*.



(a) ϕ_ℓ where $\ell = 1, 2, 3$. (b) $\mathcal{I}(\phi_\ell), \ell = 1, 2, 3$. (c) scaled $\mathcal{I}(\phi_\ell), \ell = 1, 2, 3$.

Figure 2.2: Smoothing functions using the operator $\mathcal{I}$. Functions in (b) and (c) are even extensions of $\mathcal{I}(\phi_\ell)$.

2.2.2 Dimension walk and Wu's construction

The following integral operators were employed by Schaback and Wu in the context of constructing CSRBFs [181, 213]:

$$(\mathcal{I}\phi)(r) := \int_r^\infty t\phi(t)dt, \text{ for } r \geq 0 \text{ and } \phi(t)t \in L_1[0, \infty); \tag{2.13}$$

$$(\mathcal{D}\phi)(r) := -\frac{1}{r}\phi'(r), \text{ for } r > 0 \text{ and } \phi \in C^2(\mathbb{R}). \tag{2.14}$$

$\mathcal{I}\phi$ is smoother than ϕ and ϕ is smoother than $\mathcal{D}\phi$. Furthermore, if ϕ is compactly supported on $[0, 1]$, so are $\mathcal{I}\phi$ and $\mathcal{D}\phi$; $\mathcal{D}\mathcal{I}\phi = \phi$ for $t\phi \in L_1[0, \infty)$ and $\mathcal{I}\mathcal{D}\phi = \phi$ for $\phi \in C^2(\mathbb{R})$. The most attractive property of the two operators is the *dimension walk* property [208, p.121][181], which can reduce computations (Fourier transform of a radial function) on $\mathbb{R}^d$ to computations on $\mathbb{R}$.

Proposition 2.2. *Let ϕ be a continuous function satisfying* (2.13) *and* (2.14) *respectively, then the radial function $\phi(r)$ with $r = \|\mathbf{x}\|_2$ is positive definite on $\mathbb{R}^d$ if and only if*

1. *$\mathcal{I}\phi(r)$ is positive definite on $\mathbb{R}^{d-2}$ for $d > 3$;*
2. *$\mathcal{D}\phi(r)$ is positive definite on $\mathbb{R}^{d+2}$.*

Wu constructs CSRBFs by using the dimension walk property of the operator $\mathcal{D}$. He starts with a very smooth positive CSRBF in $\mathbb{R}$ $w_\ell(r) := \phi_\ell(r^2) \star \phi_\ell(r^2)$, where $\phi_\ell(\cdot)$ is Askey's power function defined in (2.10) and $\star$ denotes for the convolution

operator. According to Proposition 2.2, $\mathcal{D}^k w_\ell(2r)$ is a positive definite CSRBFs on $\mathbb{R}^{2k+1}$. However, $\mathcal{D}^k w_\ell(2r)$ in $\mathbb{R}^{2k+1}$ is less smooth than w_ℓ in $\mathbb{R}$. To get a smooth CSRBF in $\mathbb{R}^{2k+1}$, one has to start with much smoother CSRBFs in $\mathbb{R}$ which correspond to higher degree polynomials $w_\ell(r)$. Thus, at the end of his paper [213], Wu poses the question: what is the lowest degree of a positive definite CSRBF with a given smoothness in $\mathbb{R}^d$? Wendland answers Wu's question by constructing his CSRBFs of minimal degree [207].

2.2.3 Construction of the Wendland functions

Wendland functions are constructed via the integral operator $\mathcal{I}$ in (2.13). By repeatedly applying $\mathcal{I}$ to Askey's truncated power functions $\phi_\ell(r) = (1-r)_+^\ell$ k times, Wendland obtains the following functions

$$\phi_{d,k}(r) = \mathcal{I}^k \phi_\ell, \text{ where } \ell = \lfloor d/2 \rfloor + k + 1, \text{ and } \phi_\ell = (1-r)_+^\ell. \tag{2.15}$$

Because $\lfloor d/2 \rfloor + k + 1 \ \geq \lfloor (d+2k)/2 \rfloor + 1$, according to the property of Askey's power function in Proposition 2.1, ϕ_ℓ with $\ell = \lfloor d/2 \rfloor + k + 1$ is positive definite on $\mathbb{R}^{d+2k}$ for a non-negative integer k. According to Proposition 2.2, $\phi_{d,k}$ is positive definite on $\mathbb{R}^d$. Since ℓ defined in (2.15) is the smallest integer such that ϕ_ℓ is positive definite on $\mathbb{R}^{d+2k}$, and thus ℓ is also the smallest integer such that $\phi_{d,k}$ is positive definite on $\mathbb{R}^d$. In this sense Wendland functions are also called CSRBFs of minimal degree.

$\phi_{d,k}(r)$ can be easily computed with the help of mathematical software. Table 2.1 is computed by a short Maple program provided in Program 2.1. As in Table 2.1, Wendland functions can be represented as follows:

$$\phi_{d,k}(r) = \mathcal{I}^k \phi_\ell = \phi_{\ell+k} p_{k,\ell}(r) = (1-r)_+^{(\lfloor d/2 \rfloor + k + 1) + k} p_{k,\ell}(r), \tag{2.16}$$

where $p_{k,\ell}(r)$ is a polynomial of degree k whose coefficients depend on ℓ and $p_{0,\ell} := 1$.

It can be shown that Wendland functions $\phi_{d,k}(r)$, $r = \|\mathbf{x}\|_2$ defined in (2.15) are piecewise polynomials of degree $\lfloor d/2 \rfloor + 3k + 1$ on $\mathbb{R}^d$ with respect to r and positive definite on $\mathbb{R}^d$ with a compact support in $\|\mathbf{x}\|_2 \leq 1$. For each k, $\phi_{d,k}(r)$ possesses $2k$ continuous derivatives around zero and $k + \ell - 1 = 2k + \lfloor d/2 \rfloor$ continuous derivatives around $\|\mathbf{x}\|_2 = 1$ [207][208, p.128, Theorem 9.13,]. It also has the following so-called

Table 2.1: Wendland compactly supported radial basis functions of minimal degree, computed by the Maple program in Program 2.1.

d	Wendland function $\phi_{d,k}(r)$,$r = \|\mathbf{x}\|_2$	Smoothness
$d=1$	$\phi_{1,0}(r) = (1-r)_+$	C^0
	$\phi_{1,1}(r) = (1-r)_+^3(1+3r)/12$	C^2
	$\phi_{1,2}(r) = (1-r)_+^5(3+15r+24r^2)/840$	C^4
	$\phi_{1,3}(r) = (1-r)_+^7(15+105r+285r^2+315r^3)/151200$	C^6
	$\phi_{1,4}(r) = (1-r)_+^9(105+945r+3555r^2+6795r^3+5760r^4)/51891840$	C^8
$d\leq 3$	$\phi_{3,0}(r) = (1-r)_+^2$	C^0
	$\phi_{3,1}(r) = (1-r)_+^4(1+4r)/20$	C^2
	$\phi_{3,2}(r) = (1-r)_+^6(3+18r+35r^2)/1680$	C^4
	$\phi_{3,3}(r) = (1-r)_+^8(15+120r+375r^2+480r^3)/332640$	C^6
	$\phi_{3,4}(r) = (1-r)_+^{10}(105+1050r+4410r^2+9450r^3+9009r^4)/121080960$	C^8
$d\leq 5$	$\phi_{5,0}(r) = (1-r)_+^3$	C^0
	$\phi_{5,1}(r) = (1-r)_+^5(1+5r)/30$	C^2
	$\phi_{5,2}(r) = (1-r)_+^7(3+21r+48r^2)/3024$	C^4
	$\phi_{5,3}(r) = (1-r)_+^9(15+135r+477r^2+693r^3)/665280$	C^6
	$\phi_{5,4}(r) = (1-r)_+^{11}(105+1155r+5355r^2+12705r^3+13440r^4)/259459200$	C^8
$d\leq 7$	$\phi_{7,0}(r) = (1-r)_+^4$	C^0
	$\phi_{7,1}(r) = (1-r)_+^6(1+6r)/42$	C^2
	$\phi_{7,2}(r) = (1-r)_+^8(3+24r+63r^2)/5040$	C^4
	$\phi_{7,3}(r) = (1-r)_+^{10}(15+150r+591r^2+960r^3+591r^2+960r^3)/1235520$	C^6
	$\phi_{7,4}(r) = (1-r)_+^{12}(105+1260r+6390r^2+16620r^3+19305r^4)/518918400$	C^8

reproducing kernel property [208, p.160, Theorem 10.35]:

Proposition 2.3. *For $d \geq 3$, and a non-negative integer, k, $\phi_{d,k}$ is a reproducing kernel in a Hilbert space which is norm-equivalent to the Sobolev space $\mathcal{H}^{d/2+k+1/2}(\mathbb{R}^d)$.*

Proposition 2.3 suggests that there must be some CSRBFs missing in perhaps the interesting case $d = 2$, namely, in $\mathbb{R}^2$. Such functions have been found recently and named as the missing Wendland functions [179], which can reproduce the Sobolev space $\mathcal{H}^{d/2+k+1/2}(\mathbb{R}^d)$ for even d and half-integer k.

Program 2.1 A Maple program to compute the Wendland functions.

```
wd := proc (d, k, r)
local wd, kk;
wd := (1-r)^(floor((1/2)*d)+k+1);
for kk to k do wd := int(t*subs(r = t, wd), t = r .. 1)
end do;
return factor(wd)
end proc
```

2.2.4 Construction of the missing Wendland Functions

The missing Wendland functions are constructed by using a more general integral operator $\mathcal{I}_\alpha$, as mentioned above:

$$\mathcal{I}_\alpha(f)(t) := \int_t^\infty f(s)\frac{(s-t)^{\alpha-1}}{\Gamma(\alpha)}ds, \tag{2.17}$$

where α can be a half-integer and $\Gamma(z) = \int_0^\infty t^{z-1}e^{-t}\ dt$ is the Gamma function. The operator $\mathcal{I}_\alpha$ is a scaled integral operator which is closely related to fractional derivatives, and was used to simplify the multivariate Fourier transform for radial functions [181]. Applying this operator to a modified version of the truncated power function, $a_\mu(s) := (1-\sqrt{2s})_+^\mu$, we get

$$\mathcal{I}_\alpha(a_\mu)(t) = \int_t^\infty (1-\sqrt{2s})_+^\mu \frac{(s-t)^{\alpha-1}}{\Gamma(\alpha)}ds,\ t \geq 0. \tag{2.18}$$

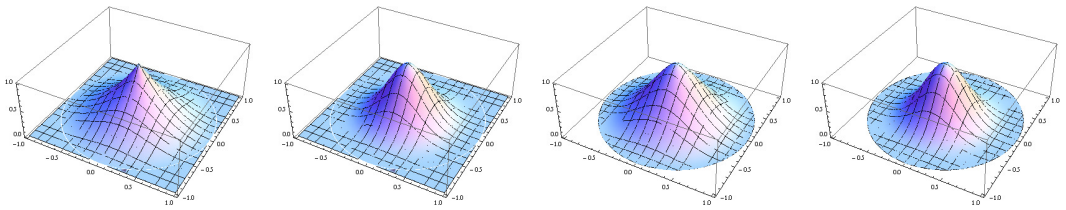
Defining $\Psi_{\mu,\alpha}(r) := \mathcal{I}_\alpha(a_\mu)(r^2/2)$, then we have

$$\Psi_{\mu,\alpha}(r) = \int_{r^2/2}^\infty (1-\sqrt{2s})_+^\mu \frac{(s-r^2/2)^{\alpha-1}}{\Gamma(\alpha)}ds = \int_r^1 t(1-t)^\mu \frac{(t^2-r^2)^{\alpha-1}}{\Gamma(\alpha)2^{\alpha-1}}dt,\ r \geq 0. \tag{2.19}$$

In particular, when $\alpha = 1$,

$$\Psi_{\mu,1}(r) = \int_r^1 t(1-t)^\mu dt = \int_r^\infty t(1-t)_+^\mu dt = \mathcal{I}(\phi_\mu)(r). \tag{2.20}$$

It turns out that $\Psi_{\ell,1}(r)$ is simply the operator $\mathcal{I}$ defined in (2.13) acting on the truncated power functions $\phi_\ell(t)$. Furthermore, it can be shown that the following



(a) $\phi_{2,0}$ (b) $\phi_{2,1}$ (c) $\Psi_{2,1/2}$ (d) $\Psi_{3,3/2}$

Figure 2.3: Scaled Wendland functions and missing Wendland functions in $\mathbb{R}^2$.

relationship between the Wendland function $\phi_{d,k}$ and the function defined in (2.19) holds for non-negative integer k:

$$\phi_{d,k} = \Psi_{\lfloor d/2 \rfloor + k + 1, k}. \tag{2.21}$$

The above relationship shows that $\Psi_{\mu,\alpha}$ are a larger class of CSRBFs which includes Wendland functions. It can also be shown that $\Psi_{\mu,\alpha}$ is positive definite when μ and α satisfy some constraint [179][181].

Proposition 2.4. *For all non-negative integers $\mu \in \mathbb{N}$ and all half-integer $\alpha = n + 1/2, n \in \mathbb{N}$, the generalized Wendland function defined in* (2.19) *is positive definite on $\mathbb{R}^d$, if $\mu \geq \lfloor d/2 + \alpha \rfloor + 1$.*

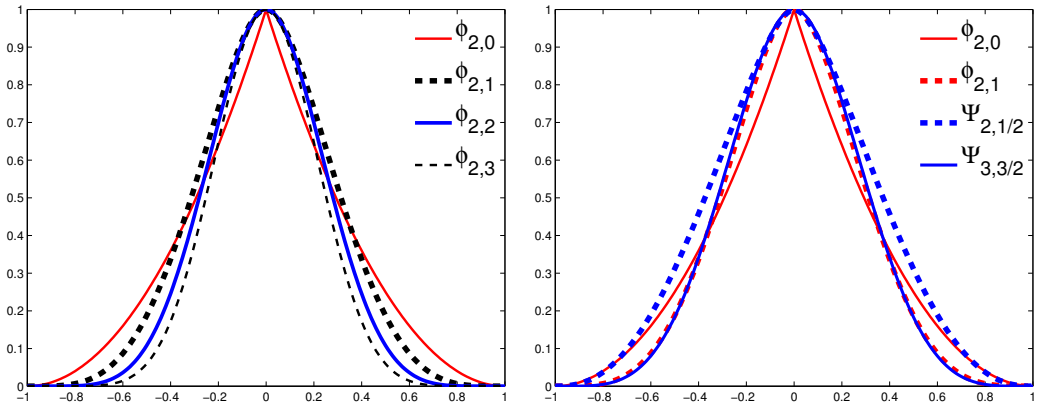
Schaback also proves that the function $\Psi_{\mu,\alpha}$ has a similar reproducing property as the Wendland function in Proposition 2.3 in even-dimensional space $\mathbb{R}^d$ [179, p.75 Collollary 1]:

Proposition 2.5. *For integers $m \geq 1, n \geq 0, d = 2m$, $\Psi_{\mu,n+1/2}$ reproduce a Hilbert space which is isomorphic to the Sobolev space $\mathcal{H}^{m+n+1}(\mathbb{R}^d) = \mathcal{H}^{d/2+\alpha+1/2}(\mathbb{R}^d)$, where $\alpha = n + 1/2$.*

For such functions $\Psi_{\mu,\alpha}$, where μ is an integer and $\alpha = n + 1/2$ is a half integer, they are the so-called missing Wendland functions.

The generalized Wendland functions can be computed by a 6-line Maple program in Program 2.2. It turns out that the missing Wendland functions $\Psi_{\mu,\alpha}$ involve two non-polynomial terms, and can be written as

$$\Psi_{\mu,\alpha}(r) = \mathcal{P}_{\mu,\alpha} \log(\frac{r}{1 + \sqrt{1 - r^2}}) + \mathcal{Q}_{\mu,\alpha} \sqrt{1 - r^2}, \tag{2.22}$$



(a) Wendland functions in $\mathbb{R}^2$ (b) Missing Wendland functions in $\mathbb{R}^2$

Figure 2.4: Comparison between Wendland functions and missing Wendland functions on $\mathbb{R}^2$. The missing Wendland function $\Psi_{3,3/2}$ is very similar to and overlaps the Wendland function $\phi_{2,1}$.

where $\mathcal{P}_{\mu,\alpha}$ and $\mathcal{Q}_{\mu,\alpha}$ are polynomials in r^2. For a detailed derivation and property of $\mathcal{P}_{\mu,\alpha}$ and $\mathcal{Q}_{\mu,\alpha}$, the reader is directed to [179]. For more details on the Wendland and the missing Wendland functions, one can refer to a recent paper [114]. Several missing Wendland functions are listed in Table 2.2.

Table 2.2: Missing Wendland functions computed by Maple program in Program 2.2.

$\Psi_{\mu,\alpha}$	Function
$\Psi_{2,1/2}$	$\frac{\sqrt{2}}{3\Gamma(1/2)}\left(3r^2\mathcal{L}+(2r^2+1)\mathcal{S}\right)$
$\Psi_{3,3/2}$	$\frac{-\sqrt{2}}{480\Gamma(3/2)}\left((15r^6+90r^4)\mathcal{L}+(81r^4+28r^2-4)\mathcal{S}\right)$
$\Psi_{4,5/2}$	$\frac{\sqrt{2}}{40320\Gamma(5/2)}\left((945r^8+2520r^6)\mathcal{L}+(256r^8+2639r^6+690r^4-136r^2+16)\mathcal{S}\right)$
$\Psi_{5,7/2}$	$\frac{-\sqrt{2}}{5677056\Gamma(7/2)}\left(\mathcal{P}_{5,7/2}\mathcal{L}+\mathcal{Q}_{5,7/2}\mathcal{S}\right)$
	$\mathcal{P}_{5,7/2}=3465r^{12}+83160r^{10}+13860r^8$
	$\mathcal{Q}_{5,7/2}=37495r^{10}+160290r^8+33488r^6-724r^4+1344r^2-128$
	$\mathcal{L}(r)=\log(\frac{r}{1+\sqrt{1-r^2}})\quad \mathcal{S}(r):=\sqrt{1-r^2}$

Program 2.2 A Maple program to compute the missing Wendland functions [179].

```
mswd := proc (mu, alpha, r)
local mswd;
mswd:=t*(1-t)^mu*(t^2-r^2)^(alpha-1)/(GAMMA(alpha)*2^(alpha-1));
mswd := int(mswd, t = r .. 1);
return combine(simplify(mswd), ln)
end proc
```

2.2.5 Unification

Both the Wendland functions and the missing Wendland functions can be written as

$$\psi_{d,k} := \begin{cases} \frac{1}{\Gamma(k)2^{k-1}} \int_r^1 t(1-t)^\ell (t^2-r^2)^{k-1} dt & \text{for } 0 \le r \le 1, \\ 0 & \text{for } r > 1, \end{cases} \tag{2.23}$$

where $\ell = \lfloor \frac{d}{2} + k \rfloor + 1$. When k is a positive integer, this defines the original Wendland functions; when k is a positive half-integer, this defines the missing Wendland functions.

The closed formula of $\psi_{d,k}$ is related to the *associated Legendre function of the first kind* and the Gamma function [114, Theorem 3.1]. Interestingly, the missing Wendland functions can be associated to the Chebyshev polynomials of the first and second kind [114, Corollary 4.6]. Further it can be shown that [42, Lemma 3.1]

$$\psi_{d,k}(0) = \frac{\Gamma(\ell+1)\Gamma(2k)}{2^{k-1}\Gamma(k)\Gamma(\ell+2k+1)}. \tag{2.24}$$

Applying this formula, one can show the relationship between the Gaussian and the compactly supported radial basis functions $\psi_{d,k}$ with $k \to \infty$.

2.3 Relationship between compactly and globally supported RBFs

As we can see from Figure 2.4, the compactly supported functions $\psi_{d,k}$ have a bell shape which are similar to the Gaussian. One might imagine that after certain scaling, the function $\psi_{d,k}$ would converge to a Gaussian. Such an intuition can be supported by other similar results on cardinal B-splines and the Euclidean hat

functions.

Consider the cardinal B-spline mentioned above, i.e. $B_0 = \chi_{[-1/2,1/2]}$ is the indicator function on $[-1/2, 1/2]$, and B_m is defined recursively by convolution $B_m = B_0 \star B_{m-1}$. It is known that B_m has a support $[-\frac{m+1}{2}, \frac{m+1}{2}]$, and as k increases, the normalised cardinal B-spline tends to the Gaussian function [196]

$$\lim_{m\to\infty} \left\{ \left(\frac{m+1}{12}\right) B_m(\sqrt{\frac{m+1}{12}} \cdot r) \right\} = \frac{1}{\sqrt{2\pi}} \exp(-\frac{r^2}{2}). \tag{2.25}$$

The limit may be taken in the point-wise sense or in the $L_p(\mathbb{R})$. Further, for $m \geq k+1$, the k-th derivatives, $B_m^{(k)}$, converges to the k-th derivative of the Gaussian function [24]

$$\lim_{m\to\infty} \left\{ \left(\frac{m+1}{12}\right)^{\frac{k+1}{2}} B_m^{(k)}(\sqrt{\frac{m+1}{12}} \cdot r) \right\} = \frac{1}{\sqrt{2\pi}} \frac{d^k \exp(-\frac{r^2}{2})}{dr^k}. \tag{2.26}$$

For the case $k = 2$, the right hand-side, $\frac{d^2 \exp(-r^2/2)}{dr^2}$, is the *Mexican hat wavelet* $(r^2-1)\exp(-r^2/2)$.

Euclidean hat functions are defined by the following formulae [98][206].

$$h_1(r) = (1-r)_+, \quad h_2(r) = \begin{cases} \frac{2}{\pi}(\arccos r - r(1-r^2)^{1/2}), & 0 \leq r \leq 1; \\ 0, & r > 1, \end{cases} \tag{2.27}$$

$$h_n(r) = h_{n-2}(r) - \frac{\Gamma(n/2)}{\pi^{1/2}\Gamma((n+1)/2)} r(1-r^2)_+^{(n-1)/2}, n \geq 3. \tag{2.28}$$

The asymptotic behaviour of the scaled h_n is related to the special function erfc [98, p.93], specifically,

$$\lim_{n\to} h_n(\frac{r}{\sqrt{(n-1)/2}}) = \mathrm{erfc}(r) = \frac{2}{\sqrt{\pi}} \int_r^\infty \exp(-v^2) dv. \tag{2.29}$$

For the Wendland function and the missing Wendland function, $\psi_{d,k}$ defined in (2.23), it has been proven recently that the scaled Wendland function and missing Wendland function converge to Gaussian as the smoothness increases [42].

$$\lim_{k\to\infty} \frac{1}{\psi_{d,k}(0)} \psi_{d,k}\left(\frac{2\varepsilon r \Gamma(k+1)}{(\ell+2k+1)\Gamma(k+\frac{1}{2})}\right) = \exp(-\varepsilon^2 r^2), \tag{2.30}$$

where $\psi_{d,k}(0)$ is given in (2.24), $\ell = \lfloor \frac{d}{2} + k \rfloor + 1$.

On the other hand, Schaback proved in [178] that interpolation by Gaussian radial basis functions with *flat limit* [60]—the case when ε tends to be 0—converges to the de Boor-Ron least polynomial interpolant [53, 54, 55].

2.4 Notes

The title of this chapter comes from Schaback and Wendland's publication [180] and partial of the content is illuminated by their discussion in [180]. Several new results and other interesting results are included here. It is very easy to loose mathematical depth and it is impossible to include all related significant results in such a dense chapter. Discussions of the construction of compactly supported radial basis function can also be found in [30, 31, 99].

The author realised the potential application of approximating a Gaussian by a compactly supported covariance function for statistical simulation, and noticed the illuminating results (2.25), (2.26) and (2.28). The author wanted to derive the limiting result, and thus prove convergence to the Gaussian, but while working on this problem the author was informed that the problem had just been solved in a paper in review by Chernih et al. [42].

Part II

Theory

3. Spectral distribution and smoothness matching

This chapter examines the spectral distribution of interpolation matrices arising from radial basis function interpolation and principles of choosing proper basis functions. In contrast with the condition number, the spectral distribution can reveal more details on eigenvalues, for example how many small eigenvalues there are, and thus can further characterize the conditioning of the underlying problem. We shall see that the spectral distribution is closely related to error due to round-off and the Picard condition. Then we conclude that the behavior of the solution depends on whether the smoothness of the chosen basis function matches with the smoothness of the target function. Finally we introduce a principle on how to choose a basis function or how to choose a proper shape parameter to match the smoothness of the target function which perhaps is unknown or at least not well-understood in the RBF community.

3.1 Eigenvalue distribution

The main idea described here is to relate the RBF interpolation scheme to a discrete integral equation of the first kind. Instead of considering the finite-dimensional linear algebra problem directly, we investigate the corresponding classical infinite-dimensional linear operator problem. Once put in a proper perspective, powerful tools are already available and many results can be applied to explore the underlying problem. Combining several scattered results, we obtain results on how the decay of eigenvalues of kernel matrices of RBFs is closely related to the smoothness of the underlying RBFs (Theorem 3.4 and Theorem 3.5), and how many small eigenvalues there are for analytic kernels (Theorem 3.9). These results are not strikingly new, however we have not seen the results elsewhere. Several established results are well known in the community of linear integral equations. The difference and connections between these theorems are discussed.

Consider the following integral equation of the first kind

$$\int_{\Omega} K(\mathbf{x},\mathbf{y})\alpha(\mathbf{y})d\mathbf{y} = f(\mathbf{x}) \tag{3.1}$$

where $\mathbf{x},\mathbf{y} \in \mathbb{R}^d$, $\Omega \subset \mathbb{R}^d$. We call $K(\mathbf{x},\mathbf{y})$ a *kernel function* in $\mathbb{R}^d$. Further, if $K(\mathbf{x},\mathbf{y})$ satisfies $\int_{\Omega} K(\mathbf{x},\mathbf{y})\alpha(\mathbf{y})\alpha(\mathbf{x})\ d\mathbf{x}d\mathbf{y} > 0$ for any non-zero function $\alpha(\mathbf{x})$, then $K(\mathbf{x},\mathbf{y})$ is positive definite. If a simple quadrature rule is applied to sample $\alpha(\mathbf{y})$, then we get a discrete equation

$$\sum_{j=1}^{N} K(\mathbf{x},\mathbf{y}_j)w_j\alpha(\mathbf{y}_j) = f(\mathbf{x}). \tag{3.2}$$

Further collocating at $\mathbf{x}_1,\mathbf{x}_2,\ldots,\mathbf{x}_N$ gives

$$\sum_{j=1}^{N} K(\mathbf{x}_i,\mathbf{y}_j)w_j\alpha(\mathbf{y}_j) = f_i, i = 1,2,\ldots,N, \tag{3.3}$$

where $f_i = f(\mathbf{x}_i)$. Let $K(\mathbf{x},\mathbf{y}) = \phi(\|\mathbf{x}-\mathbf{y}\|)$, then we can write these equations as a linear system

$$\begin{pmatrix} \phi(\|\mathbf{x}_1-\mathbf{y}_1\|) & \phi(\|\mathbf{x}_1-\mathbf{y}_2\|) & \cdots & \phi(\|\mathbf{x}_1-\mathbf{y}_N\|) \\ \phi(\|\mathbf{x}_2-\mathbf{y}_1\|) & \phi(\|\mathbf{x}_2-\mathbf{y}_2\|) & \cdots & \phi(\|\mathbf{x}_2-\mathbf{y}_N\|) \\ \vdots & \vdots & \ddots & \vdots \\ \phi(\|\mathbf{x}_N-\mathbf{y}_1\|) & \phi(\|\mathbf{x}_N-\mathbf{y}_2\|) & \cdots & \phi(\|\mathbf{x}_N-\mathbf{y}_N\|) \end{pmatrix} \begin{pmatrix} \alpha_1 \\ \alpha_2 \\ \vdots \\ \alpha_N \end{pmatrix} = \begin{pmatrix} f_1 \\ f_2 \\ \vdots \\ f_N \end{pmatrix}, \tag{3.4}$$

where $\alpha_j = w_j\alpha(\mathbf{y}_j)$, $j = 1,2,\ldots,N$. We denote the matrix in (3.4) by $A_{\phi,\mathcal{X}}$, where $\mathcal{X}$ denotes the set $\{\mathbf{x}_1,\mathbf{x}_2,\ldots,\mathbf{x}_N\}$. When $\mathbf{x}_i = \mathbf{y}_i$, $1 \le i \le N$, the linear system as in (3.4) is the same as that obtained for interpolation matrices with the radial basis function ϕ at the points $\mathbf{x}_1,\mathbf{x}_2,\ldots,\mathbf{x}_N$. We call the matrix in (3.4) a *kernel matrix*.

On the other hand, consider the interpolation scheme with a sum of translates of a radial basis function

$$s(\mathbf{x}) = \sum_{j=1}^{N} \alpha_j\phi(\|\mathbf{x}-\mathbf{y}_j\|). \tag{3.5}$$

If $s(\mathbf{x})$ interpolates an unknown function, $f(\mathbf{x})$, on the data set $\mathcal{X}$, and further $\mathbf{x}_j = \mathbf{y}_j$, $1 \leq j \leq N$, it results in the same linear system (3.4). We investigate the spectral distribution of kernel matrices (or interpolation matrices) by studying the infinite dimensional linear operator

$$\mathcal{K}\Psi(\mathbf{x}) = \int_{\Omega} K(\mathbf{x}, \mathbf{y})\Psi(\mathbf{y})d\mathbf{y}, \tag{3.6}$$

where $K(\mathbf{x}, \mathbf{y}) = \phi(\|\mathbf{x} - \mathbf{y}\|)$ is a radial function ϕ. To investigate the spectral distribution of the operator $\mathcal{K}$ in (3.6), consider the following well known Weyl-Courant minimax principle.

Lemma 3.1 (**The Weyl-Courant minimax principle [136, p.133]**). *If $\mathcal{K}$ is any compact symmetric operator on a Hilbert space $\mathcal{H}$ with eigenvalues*

$$|\lambda_0| \geq |\lambda_1| \geq |\lambda_2| \geq \cdots \geq |\lambda_n| \geq \cdots,$$

and if $\mathcal{S}$ is any operator of rank $\leq n$, then $\|\mathcal{K} - \mathcal{S}\| \geq |\lambda_n|$.

In [136], Little and Reade apply the Weyl-Courant minimax principle to tails of the Chebyshev expansion for analytic kernels, and conclude that the eigenvalues of an analytic kernel on a finite interval go to zero at least as fast as ρ^{-n} for some fixed $\rho > 1$. The proof depends on the estimation of Chebyshev coefficients of analytic functions. Some results used in their proof in fact date back to Bernstein [17] and relate to the so-called *Bernstein's ellipse*, denoted as $\mathcal{E}_\rho$ [195, p.56]. The ellipse, $\mathcal{E}_\rho$, has foci at ± 1, and the sum of the semi-axes equals $\rho > 1$.

Lemma 3.2 ([17][195, p.57]). *Let a function, f, be analytic in $[-1, 1]$ and analytically continuable to the open Bernstein ellipse $\mathcal{E}_\rho$, where it satisfies $|f(z)| \leq M$ for some M. Then its Chebyshev coefficients, a_n, satisfy $a_0 \leq M$ and*

$$|a_n| \leq 2M\rho^{-n}, n \geq 1. \tag{3.7}$$

Its Chebyshev truncations (partial sums of the Chebyshev expansion) satisfy

$$\|f - S_n\| \leq \frac{2M\rho^{-n}}{\rho - 1} \tag{3.8}$$

and its Chebyshev interpolants satisfy

$$\|f - p_n\| \leq \frac{4M\rho^{-n}}{\rho - 1}, \tag{3.9}$$

where $S_n(x) = \sum_{k=0}^{n} a_k T_k(x)$*, and* $p_n(x)$ *is the polynomial obtained by interpolation in Chebyshev points.*

Formula (3.7) is due to Bernstein, and the second part of the lemma can be found in [195, p.57, thm. 8.2].

Observation 1. For an analytic function, its Chebyshev coefficients in the Chebyshev expansion decay geometrically.

A function, $f(x)$, of *bounded variation* on $\mathbb{R}$ is a real-valued function, it is an integrable function, say, $f(x) \in L(\mathbb{R})$, and, V, the supremum of $\int f(x)\, dg(x)$ over all $g(x) \in C^1(\mathbb{R})$ with $|g(x)| < 1$ is finite. If $f(x)$ is continuous, then the supremum of $\sum_{j=1}^{N} |f(x_j) - f(x_{j-1})|$ is bounded over all finite samples $x_0, x_1, \ldots, x_N$. For finitely differentiable functions with the highest derivatives of bounded variations, we consider the following result.

Lemma 3.3 ([195, p.52-p.53])**.** *For any integer* $\nu \geq 0$*, let* u *and its derivatives* $u', \ldots, u^{(\nu-1)}$ *be absolutely continuous on* $[-1, 1]$*, and* $u^{(\nu)}$ *be of bounded variation* V*, then for any* $n \geq \nu + 1$*, the Chebyshev coefficients of* u *satisfy*

$$|a_n| \leq \frac{2V}{\pi(n-\nu)^{\nu+1}}, \quad n > \nu. \tag{3.10}$$

Its Chebyshev truncations S_n *satisfy*

$$\|u - S_n\| \leq \frac{2V}{\pi\nu(n-\nu)^{\nu}}, \quad n > \nu \tag{3.11}$$

and its Chebyshev interpolants satisfy

$$\|f - p_n\| \leq \frac{4V}{\pi\nu(n-\nu)^{\nu}}, \quad n > \nu. \tag{3.12}$$

Observation 2. The Chebyshev coefficients of smoother functions decay faster.

Besides Chebyshev truncations, we also consider the Fourier series truncations. Such an approximation is valid for a kernel function which is Lebesgue integrable,

not necessarily analytic, and it can be generalized to higher dimensional space. Here, we demonstrate the idea with the simplest one dimensional case. For a Lebesgue integrable function u on Ω, say, $u \in L^1(\Omega)$, the n-th Fourier coefficient of u is defined by

$$\hat{u}_n := \hat{u}(n) = \frac{1}{T}\int_{-T/2}^{T/2} u(t)e^{-int}\,dt \tag{3.13}$$

for a T-periodic function, u, on $\Omega = [-T/2, T/2]$, where T could be finite or infinite. If there is no confusion we use $\hat{u}_n$ for short, otherwise we use $\hat{u}(n)$. The *Fourier series* $S[u]$ of a function $u \in L^1(\Omega)$ is the trigonometric series

$$S[u] \sim \sum_{n=-\infty}^{\infty} \hat{u}_n e^{int}. \tag{3.14}$$

The n-th Fourier series truncation is denoted as $S_n[u] = \sum_{k=-n}^{n} \hat{u}_n e^{int}$. Remembering Euler's formula $e^{it} = \cos t + i \sin t$, the Fourier series of a function $u \in L^1(\Omega)$ is equal to

$$\frac{A_0}{2} + \sum_{n=1}^{\infty}(A_n \cos nt + B_n \sin nt) = \frac{A_0}{2} + \sum_{n=1}^{\infty} \sqrt{A_n^2 + B_n^2}\sin(nt + \theta_n), \tag{3.15}$$

where $A_n = \hat{u}_n + \hat{u}_{-n}$, $B_n = i\{\hat{u}_n - \hat{u}_{-n}\}$ and $\arcsin\theta_n = B_n/\sqrt{A_n^2 + B_n^2}$. In particular, when u is an even function, $\hat{u}_n = \hat{u}_{-n}$,

$$S_n = \hat{u}(0) + 2\sum_{k=1}^{n} \hat{u}_n \cos nt. \tag{3.16}$$

As discussed above, it is clear that *the Fourier series truncation S_n is a function of finite rank $n+1$.*

By standard Fourier analysis, see [123] for example, we have:

Lemma 3.4 (**Fourier coefficients of differentiable functions).** *Let u be a square integral function on Ω with Fourier transform $\hat{u}$ and Fourier coefficients $\hat{u}_n$.*

1. *If u has $\nu - 1$ continuous derivatives in $L^2(\Omega)$ for some $\nu \geq 0$, and the ν^{th} derivative is of bounded variation, then $\hat{u}_n = \mathcal{O}(|n|^{-\nu-1})$ as $|n| \to \infty$.*

2. *If $u \in L^2(\Omega)$ and $u \in C^\infty$, then $\hat{u}_n = \mathcal{O}(|n|^{-\nu})$ as $|n| \to \infty$ for every $\nu \geq 0$.*

3. *If u has up to ν times continuous derivatives and $u^{(\nu)} \in L^2(\Omega)$, then $\hat{u}_n$ decays on the order of $o(n^{-\nu-1/2})$.*

Observation 3. The Fourier coefficients of smoother functions decay faster.

3.1.1 Main results

The eigenvalue problem of the linear integral operator (3.6) has been well-studied for many years since the work of Fredholm in 1903 [92]. There are many insightful results in the literature, see [45, 56, 92, 109, 125, 136, 157, 158, 159, 160, 161, 162, 163, 164, 165, 188, 210] for example. Here we only consider the most relevant and brief results, provide alternative simple proofs, and develop new results.

Theorem 3.1 (**Weyl** [210, p.449-450])**.** *If the kernel function $K(x,y) = K(y,x)$ and $\frac{\partial K^\nu(x,y)}{\partial^\nu x}$ exists and is continuous, then the magnitude of its eigenvalues decays at the order $|\lambda_n| = o(n^{-\nu-1/2})$.*

The original result of Weyl only states this for the case $\nu = 1$, and states it as $\lim_{n\to\infty} n^{3/2}\lambda_n = 0$ [210, p.449]. If the kernel function is positive definite, sharper results hold [38, 101, 163].

Theorem 3.2 (**Reade-Ha**)**.** *If a kernel function $K(x,y)$ is positive definite, 2π-periodic in x and y, and ν times continuously differentiable, then*

1. *$\lambda_n = \mathcal{O}(n^{-\nu-1})$;*
2. *for even ν, the sharper result $\sum_{n=1}^{\infty} n^\nu \lambda_n < \infty$ holds.*

Remark 3.1. This theorem was first proved for $\nu = 1$ in [159] and then for the general case in [160]. The second part of the theorem was initially believed to be true for all ν [101], but Reade [163] constructs a counterexample for odd ν. The eigenvalues of positive kernels have also been considered in [26, 27, 77] with additional constraints.

It has been noted that both Theorem 3.1 and Theorem 3.2 are sharp [162]. Without the positive definite constraint, similar results can be obtained according to the following result which states the essential connection between the Fourier coefficients of a 2π-periodic kernel function and the eigenvalues of its corresponding integral operator.

Theorem 3.3 (**Hille-Tamarkin** [109, p.10]). *If $K(x,y) = k(x-y)$ and $k(x) \in L^2[-\pi,\pi]$ is a periodic kernel, then $\lambda_n = 2\pi\hat{k}_n$, where $\hat{k}_n = \frac{1}{2\pi}\int_{-\pi}^{\pi} k(x)e^{inx}\,dx$ is the Fourier coefficient of the kernel function.*

Applying Lemma 3.4 and Theorem 3.3, it is easy to show the following results.

Theorem 3.4. *Let $K(x,y) = k(x-y)$, where $k(x)$ is a 2π periodic and square integrable function on $[-\pi,\pi]$. Further let $|\lambda_n|$ be the n^{th} largest eigenvalue in magnitude of its corresponding integral operator, then*

1. *if $k(x) \in C^\nu$, for some natural number $\nu > 0$ and its ν-th derivative is of bounded variation, then $|\lambda_n| = \mathcal{O}(n^{-\nu-1})$ as $n \to \infty$;*
2. *if $k(x) \in C^\infty$, then $|\lambda_n| = \mathcal{O}(n^{-\nu})$ as $n \to 0$ for any $\nu > 0$;*
3. *if $k(x)$ has up to ν times continuous derivatives and the ν-th derivative is square integrable, then $|\lambda_n| = o(n^{-\nu-1/2})$.*

Now we consider methods for more general cases.

3.1.2 First method: truncated Fourier series Approximation

For simplicity, we consider the compactly supported radial basis functions; such functions are even and can be extended to be periodic on the real line. Let $k(x)$ be a square integrable even function in $\mathbb{R}$. Its Fourier series truncation has the form in (3.16). Furthermore,

$$\lim_{n\to\infty} \|k(x) - S_n\|_{L_2} = 0. \tag{3.17}$$

This implies $\sum_n^\infty |\hat{k}(n)|^2$ is convergent. According to the translation property of the Fourier coefficient,

$$k(x-y) \sim \hat{k}(0) + \sum_{n=1}^{\infty} \hat{k}(n)e^{-iny}\cos nx, \tag{3.18}$$

since $|\hat{k}_y(n)|^2 = |\hat{k}(n)e^{-iny}|^2 = |\hat{k}(n)|^2$, and thus

$$\|k(x-y) - S_n[k_y]\|_{L^2}^2 = \sum_{m=n+1}^{\infty} |\hat{k}_y(m)|^2 = \sum_{m=n+1}^{\infty} |\hat{k}(m)|^2. \tag{3.19}$$

According to the Weyl-Courant minimax principle, the $n+1$-th eigenvalue of the kernel function $k(x,y)$ satisfies

$$|\lambda_{n+1}|^2 \leq \|k(x-y) - S_n[k_y]\|_{L^2}^2 = \sum_{m=n+1}^{\infty} |\hat{k}(m)|^2. \tag{3.20}$$

Suppose $k(x)$ satisfies Lemma 3.4.1, then we get

$$|\lambda_{n+1}|^2 \leq \sum_{m=n+1}^{\infty} |\hat{k}(m)|^2 \leq \sum_{m=n+1}^{\infty} |Cm^{-2\nu-2}| \tag{3.21}$$

$$\leq \int_{n+1}^{\infty} \frac{C}{t^{2\nu+2}} dt = \frac{C}{(2\nu+1)} \frac{1}{(n+1)^{2\nu+1}}. \tag{3.22}$$

Thus, $|\lambda_n| = \mathcal{O}(n^{-\nu-1/2})$. Similarly for $k^{(\nu)} \in L^2(\Omega)$, we have $|\lambda_n| = o(n^{-\nu})$.

Theorem 3.5. *Let ϕ be a radial basis function in $L^2(\Omega)$, then*

1. *if ϕ has $\nu-1$ continuous derivatives for some $\nu > 0$ and the ν^{th} derivative is of bounded variation, then its eigenvalues decay at least in the order $\mathcal{O}(n^{-\nu-1/2})$.*

2. *if ϕ has $\nu - 1$ continuous derivatives and $\phi^{(\nu)} \in L^2(\Omega)$, then the eigenvalues of the corresponding linear operator decay in the order $o(n^{-\nu})$.*

By the Fourier dilation formula and a few additional computations, we obtain the following results for scaled radial basis functions.

Corollary 3.6. *Let $\phi_\varepsilon(x) = \phi(\varepsilon x)$, $\varepsilon > 0$ be a scaled radial basis function in $L^2(\Omega)$, then*

1. *if ϕ has $\nu-1$ continuous derivatives for some $\nu > 0$ and the ν^{th} derivative is of bounded variation, then its eigenvalues decay at least in the order $\mathcal{O}\left((\frac{\varepsilon}{n})^{\nu+1/2}\right)$.*

2. *ϕ has ν continuous derivatives and $\phi^{(\nu)} \in L^2(\Omega)$, then the eigenvalues of the corresponding linear operator decay in the order $o\left((\frac{\varepsilon}{n})^{\nu}\right)$.*

Remark 3.2. For the compactly supported Wendland function, $\phi_{d,k} \in C^{2k}$, the lower bound of the smallest eigenvalues is $\lambda_{\min} \geq Cq^{2k+1}$ [208, p.214] for some constant C, where q is the so-called separation distance. For equal space sampling in the

interval $[-1,1]$ with the sampling points arranged as $-1 = x_0 < x_1 < \cdots < x_n = 1$, $q = \frac{1}{2}\min_{i=j}|x_i - x_j| = \frac{1}{n}$, so that $\lambda_{\min} \geq Cn^{-2k-1}$. The proof of Theorem 3.5 shows that $\lambda_{\min} = o(n^{-2k})$. Therefore, we obtain both a lower and upper bound on the smallest eigenvalue for large n, say, $C_1 n^{-2k-1} \leq \lambda_{min} \leq C_2 n^{-2k}$.

The following method employing Chebyshev truncation brings sharper results.

3.1.3 Second method: truncated Chebyshev series approximation

Using truncated Chebyshev series to approximate an analytic kernel function has been studied in [136]. Two key points in their poof are the Weyl-Courant minimax principle and Bernsten's results (3.7). A slightly different formula to (3.8) is needed. The Chebyshev expansion of a kernel function is

$$K(x,y) = \frac{1}{2}a_0(y) + \sum_{\ell=1}^{\infty} a_\ell(y)T_\ell(x) \tag{3.23}$$

Thus the estimation of $a_\ell(y)$ is needed. For details see [136].

Theorem 3.7 (Little-Reade [136]). *If $K(x,y) = K(y,x) \in C[-1,1]^2$, and for each $y \in [-1,1]$ there is an analytic continuation to $K(z,y)$ for z inside the ellipse $\mathcal{E}_\rho$, which is uniformly bounded in z, y in this range, then the eigenvalues corresponding to the kernel function $K(x,y)$ decay in the order $|\lambda_{n+1}| = \mathcal{O}(\rho^{-n})$.*

Proof. See [136], or one can directly apply Lemma 3.2. □

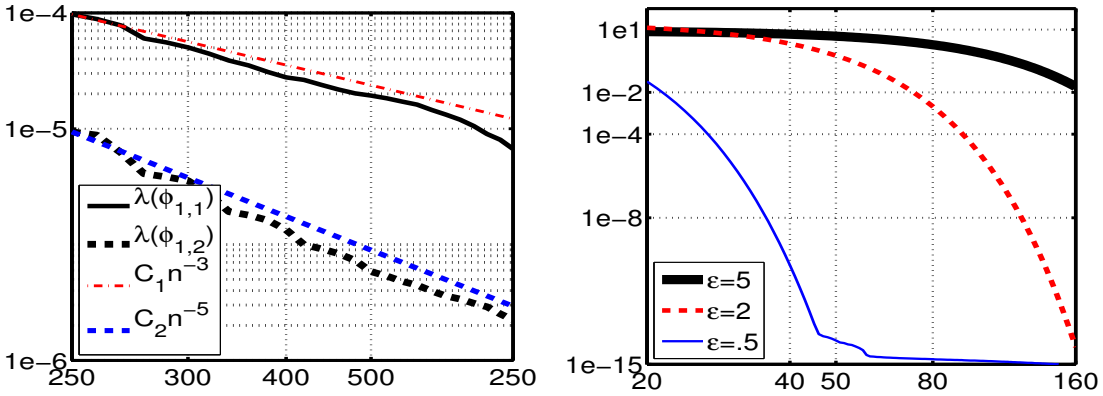
Now consider the scaled kernel function $K(\varepsilon x, \varepsilon y)$ in the unit square,

$$\mathcal{K}f(\varepsilon x) = \int_{|\varepsilon t|\leq 1} K(\varepsilon x, \varepsilon t)f(\varepsilon t)d(\varepsilon t). \tag{3.24}$$

Then the operator (3.24) is equivalent to the following

$$\mathcal{K}f(X) = \int_{-1/\varepsilon}^{1/\varepsilon} K(X,T)f(T)d(T). \tag{3.25}$$

Suppose $\mathcal{E}_{\rho_\varepsilon}$ denotes the ellipse with foci at $\pm\frac{1}{\varepsilon}$ and semi-axis sum ρ_ε, then $\rho_\varepsilon = \frac{\rho}{\varepsilon}$. If $\varepsilon < 1$ and the kernel function $K(x,y)$ is analytic in $\mathcal{E}_{\rho_\varepsilon}$, we have the following result.



(a) Eigenvalues of Wendland functions (b) Eigenvalues of Gaussian kernels

Figure 3.1: Eigenvalue decay order of positive definite kernels.

Theorem 3.8. *The eigenvalues of the scaled kernel $K(\varepsilon x, \varepsilon y)$, $\varepsilon < 1$, decay in the order $|\lambda_{n+1}| = \mathcal{O}(\rho_\varepsilon^{-n}) = \mathcal{O}(\varepsilon^n \rho^{-n})$.*

When $\varepsilon \geq 1$, the factor ε^n does not work, because the ellipse $\mathcal{E}_{\rho_\varepsilon} \subset \mathcal{E}_\rho$ can not give a tighter estimation of a_k in (3.7). The case $\varepsilon \to 0$ corresponds to the basis functions tending to flatness; the smallest eigenvalue of the kernel matrix will become smaller.

Lemma 3.3 can be used to obtain similar results as Theorem 3.5 according to Chebyshev series truncation. The proof involves additional complex analysis results and the reader is directed to [136, 195] for details.

Observation 4. The eigenvalues of smoother kernels decay faster.

3.1.4 Separable kernels in $\mathbb{R}^d$

A separable kernel in $\mathbb{R}^d$ can be expressed as the product of multiple kernels, say

$$K(\mathbf{x}, \mathbf{y}) = K_1(x_1, y_1) K_2(x_2, y_2) \cdots K_d(x_d, y_d), \mathbf{x}, \mathbf{y} \in \mathbb{R}^d, x_i, y_i \in \mathbb{R}.$$

Such kernels exist and have been considered for a long time, see [188] for example. The famous example is the Gaussian radial basis function $e^{-\|\mathbf{x}-\mathbf{y}\|^2}$.

Consider the simplest separable kernel in the case $d = 2$, if there are analytical

eigenfunction expansions for each K_i, $i = 1, 2$,

$$K_1(x_1, y_1) = \sum_{m=1}^{\infty} \lambda_m \varphi_m(x_1) \varphi_m^*(y_1) \text{ and } K_2(x_2, y_2) = \sum_{n=1}^{\infty} \mu_n \psi_n(x_2) \psi_n^*(y_2), \quad (3.26)$$

and further $\{\lambda_k\}_{k=1}^{\infty}$ and $\{\mu_k\}_{k=1}^{\infty}$ are absolutely convergent, then according to the Mercer theorem [156, p.96]

$$\begin{aligned}
&\int\int K_1(x_1, y_1) K_2(x_2, y_2) \varphi_p(y_1) \psi_q(y_2) dy_1 dy_2 \\
=&\int\int \varphi_p(y_1) \left(\sum_{m=1}^{\infty} \lambda_m \varphi_m(x_1) \varphi_m^*(y_1) \right) \psi_q(y_2) \left(\sum_{n=1}^{\infty} \mu_n \psi_n(x_2) \psi_n^*(y_2) \right) dy_1 dy_2 \\
=&\lambda_p \mu_q \varphi_p(x_1) \psi_q(x_2).
\end{aligned}$$

When the kernel is symmetric, then $\varphi_m^* = \varphi_m$ and $\psi_n^* = \psi_n$. In this way, one can show that if λ is an eigenvalue of $K_1(x_1, y_1)$ and μ is an eigenvalue of $K_2(x_2, y_2)$, then $\lambda\mu$ is an eigenvalue of $K(\mathbf{x}, \mathbf{y})$.

Suppose the eigenvalues of $K_i(x_i, y_i)$, $i = 1, 2$ are in the order $\mathcal{O}(1)$, $\mathcal{O}(R^{-1})$, $\mathcal{O}(R^{-2})$,..., for some $R > 1$, then the eigenvalues of an interpolation matrix with $K(\mathbf{x}, \mathbf{y})$ are expected in the following order: 1 of order $\mathcal{O}(1)$, 2 of order $\mathcal{O}(R^{-1})$, 3 of order $\mathcal{O}(R^{-2})$ and $\binom{m+1}{1}$ of order $\mathcal{O}(R^{-m})$. See Table 3.1 for illustration. Such a case can happen, for example, with a separable kernel on a square with an equally spaced $N \times N$ regular mesh. If $K_1(x_1, y_1)$ is the kernel in the horizontal direction, and $K_2(x_2, y_2)$ is the kernel in the vertical direction, then their corresponding discrete kernel matrices should have the same eigenvalues because their relative distances in each direction are identical. Similarly, one can show in $\mathbb{R}^d$ that, on a $d \times d$ cube with a regular equally spaced mesh, the discrete kernel matrix of a separable kernel is supposed to have $\binom{k+d}{d} = \frac{(k+d)!}{k!(d)!}$ eigenvalues in similar order. This number is equal to the number of terms in the expansions of

$$(x_1 + x_2 + \ldots + x_d)^k \quad (3.27)$$

and will become more clear as our discussion proceeds. We first use the analytic and numerical results on Gaussian radial basis functions to verify the result.

Example 3.1. Consider a weighted inner product defined by

$$(u(x), v(x)) = \int u(x)v(x)w(x)dx,$$

where $w(x) > 0$ is a weight function, then the eigenvalues of a kernel are defined by

$$\int K(x, y)w(y)\psi(y)dy = \lambda\psi(x).$$

With the weight function $w(x) = \frac{\alpha}{\sqrt{\pi}}e^{-\alpha^2x^2}$, $\alpha > 0$, the analytic eigenvalues of the Gaussian kernel $e^{-\varepsilon^2(x-y)^2}$ in $\mathbb{R}$ are given by

$$\lambda_n = \sqrt{\frac{\alpha^2}{\alpha^2 + \delta^2 + \varepsilon^2}}\left(\frac{1}{1 + \delta^2/\varepsilon^2 + \alpha^2/\varepsilon^2}\right)^{n-1},$$

where

$$\delta^2 = \frac{\alpha^2}{2}\left(\sqrt{1 + (\frac{2\varepsilon}{\alpha})^2} - 1\right), \text{for } n = 1, 2, 3, \cdots.$$

See [220][156, p.97] for details. Clearly, $1 + \delta^2/\varepsilon^2 + \alpha^2/\varepsilon^2 > 1$ is the parameter ρ in theorem 3.7. For the multivariate Gaussian kernel

$$K(\mathbf{x}, \mathbf{y}) = e^{-\varepsilon_1^2(x_1-y_1)^2-\cdots-\varepsilon_d^2(x_d-y_d)^2},$$

we only consider the case $\varepsilon_1 = \cdots = \varepsilon_d = \varepsilon$; according to formula (3.6a) in [72, p.A742], the eigenvalues of a multivariate Gaussian radial basis function under the weighted inner product with the above weight function can be written as

$$\lambda_{\mathbf{n}} = \prod_{j=1}^{d}\lambda_{n_j} = \left(\frac{\alpha^2}{\alpha^2 + \delta^2 + \varepsilon^2}\right)^d\left(\frac{1}{1 + \delta^2/\varepsilon^2 + \alpha^2/\varepsilon^2}\right)^{\sum_{j=1}^d n_j - d}. \tag{3.28}$$

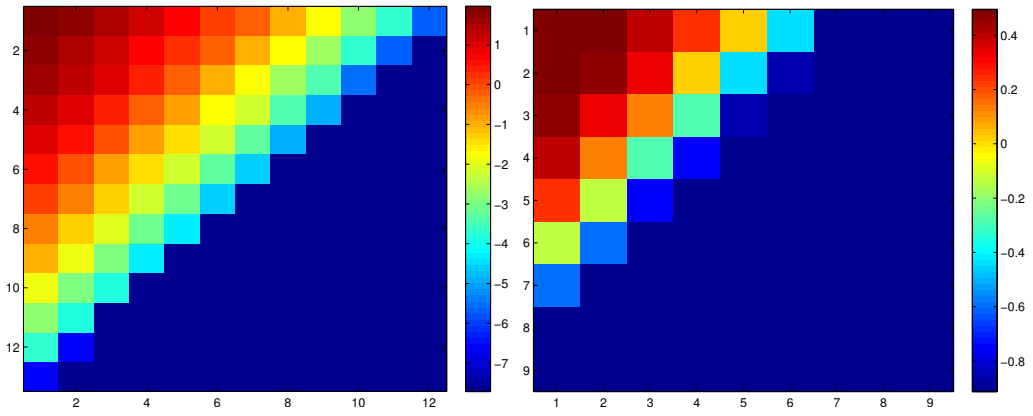
Then if $\sum_{j=1}^d n_j = k$, there are $\binom{k+d}{d} = \frac{(k+d)!}{k!d!}$ possible combinations of $n_1, \cdots, n_d$.

3.1.5 Infinite smooth kernels in $\mathbb{R}^d$ with flat limit

The discussion above is primarily based on the properties of the continuous integral operator and without consideration of the location of the underlying interpolation points. This section concentrates on the discrete form, in particular, on those cases where the shape parameter of the underlying radial basis function tends

	1	R^{-1}	R^{-2}	R^{-3}	R^{-4}	R^{-5}	$\dots$
1	1	R^{-1}	R^{-2}	R^{-3}	R^{-4}	R^{-5}	$\dots$
R^{-1}	R^{-1}	R^{-2}	R^{-3}	R^{-4}	R^{-5}	R^{-6}	$\ddots$
R^{-2}	R^{-2}	R^{-3}	R^{-4}	R^{-5}	R^{-6}	$\ddots$	$\ddots$
R^{-3}	R^{-3}	R^{-4}	R^{-5}	R^{-6}	$\ddots$	$\ddots$	$\ddots$
R^{-4}	R^{-4}	R^{-5}	R^{-6}	$\ddots$	$\ddots$	$\ddots$	$\ddots$
R^{-5}	R^{-5}	R^{-6}	$\ddots$	$\ddots$	$\ddots$	$\ddots$	$\ddots$
$\vdots$	$\ddots$	$\ddots$	$\ddots$	$\ddots$	$\ddots$	$\ddots$	$\ddots$

Table 3.1: The order of eigenvalues of tensor product Kernel in R^2



(a) a 9×9 mesh on unit square (b) 27 equal spaced points on a circle

Figure 3.2: Eigenvalues of kernel matrices of Gaussian radial basis function on different meshes. The shape parameter $\varepsilon = 5$. We first sort the eigenvalues in descending order, then map the 1 dimensional array to a 2 dimensional diagram by the inverting Cantor pairing function (http://mathworld.wolfram.com/PairingFunction.html). The figure shows the log scale of the eigenvalues by the Matlab function `imagesc`. The dark blue area in the bottom-right corners is empty with `NaN`. Panel (a) demonstrates that, on the regular mesh, the eigenvalues in the same order are grouped by $1, 2, 3, 4, \dots$; while panel (b) shows the eigenvalues are grouped by $1, 2, 2, 2, \dots$.

to 0. These radial basis functions with shape parameter $\varepsilon \to 0$ are called radial basis functions with flat limit, and have received a lot of attention in recent years [85, 86, 87]. The flatness brings the advantage of asymptotic techniques in ε.

Another technique which can be used to investigate the spectral distribution of high dimensional kernels with flatness limit is due to Schaback. In [178], Schaback

proves that a Gaussian with flat limit interpolation tends to de Boor-Ron's least polynomial interpolant [53, 54, 55]. In [178, Theorem 6, p.307], the author focused on geometric properties of the interpolation points. The techniques used there can be used to prove the eigenvalue distribution. The basic idea is to construct a sequence of nested subspaces, and investigate quadratic forms corresponding to the interpolation matrix (3.4). Combining these techniques with other linear algebra results, one can show the spectral distribution of scaled infinitely smooth radial functions, where the radial functions are not necessarily separable. To prove the results, we introduce the following notation and lemma.

Let $\{p_1, p_2, \ldots, p_Q\}$ be a basis set for $\pi_{k-1}(\mathbb{R}^d)$, the multivariate polynomial space of degree at most $k-1$, and denote $P_k \in \mathbb{R}^{N\times Q}$ as the matrix with entries $\left(p_j\left(\mathbf{x}_i\right)\right)_{1\le i\le N, 1\le j\le Q}$, then for any given data set $\mathcal{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N\} \subset \Omega \subset \mathbb{R}^d$, we have $\ker(P_{k+1}^T) \subseteq \ker(P_k^T)$ for $k = 1, 2, 3, \ldots$. Further denote $\ker(P_0^T) = \mathbb{R}^N$, then for any finite set $\mathcal{X}$, there is a positive integer $\mu(\mathcal{X})$ such that

$$\emptyset = \ker(P_{\mu(\mathcal{X})}^T) \subseteq \cdots \subseteq \ker(P_{k+1}^T) \subseteq \ker(P_k^T) \subseteq \cdots \subseteq \ker(P_1^T) \subset \ker(P_0^T). \quad (3.29)$$

$\mu(\mathcal{X})$ can be viewed as a geometric property of the data set $\mathcal{X}$, it can be interpreted as there existence of a subset of $\mathcal{X}$ on which there exists a unique polynomial interpolation in $\pi_{\mu(\mathcal{X})}(\mathbb{R}^d)$.

Let $\mathbf{\Delta}$ be the Eucidean distance matrix with entries $(\|\mathbf{x}_i - \mathbf{x}_j\|)_{1\le i,j\le N}$ and $\mathbf{\Delta}^k = (\|\mathbf{x}_i - \mathbf{x}_j\|^k)_{1\le i,j\le N}$ denote the element-wise power of $\mathbf{\Delta}$. There is a well known result on this element-wise power of distance matrices.

Lemma 3.5 (Micchelli [148, p.15, Lemma 3.1]). *If $\sum_{i=1}^n \alpha_i p(\mathbf{x}_i) = 0$ for all $p \in \pi_{k-1}(\mathbb{R}^d)$, then $(-1)^k \underline{\alpha}^T \mathbf{\Delta}^{2k} \underline{\alpha} \ge 0$, where the equality holds if and only if*

$$\sum_{i=1}^n \underline{\alpha}_i p(x_i) = 0,$$

for all $p \in \pi_k(\mathbb{R}^d)$.

For convenience, we denote the interpolation matrix (3.4) corresponding to the scaled radial basis function $\phi(\varepsilon r) = g(\varepsilon^2 r^2)$ by $A_{\phi_\varepsilon,\mathcal{X}} = \phi(\varepsilon\mathbf{\Delta}) = g(\varepsilon^2\mathbf{\Delta}^2)$. The function g is taken to be infinitely smooth and has a Taylor expansion at the origin.

Theorem 3.9. *Let $A_{\phi_\varepsilon,\mathcal{X}}$ be an interpolation matrix corresponding to an infinitely smooth radial basis function $\phi_\varepsilon(r) = \phi(\varepsilon r) = g((\varepsilon r)^2)$. If g has a convergent Taylor expansion near the origin on the real line, then, when $\varepsilon \to 0$, there are exactly $\dim\left(\ker(P_k^T)\right) - \dim\left(\ker(P_{k+1}^T)\right)$ eigenvalues on the order of ε^{2k}, for $0 \leq k \leq \mu(\mathcal{X}) - 1$, where P_k^T and $\mu(\mathcal{X})$ are defined in (3.29). Furthermore, if each P_k is full rank, then there are exactly $\binom{k+d-1}{d-1} = \frac{(k+d-1)!}{k!(d-1)!}$ eigenvalues on the order of ε^{2k}.*

Proof. Since the function g has a convergent Taylor expansion near the origin, we can write the entries of the interpolation matrix as a sum of the element powers of the distance matrix $\mathbf{\Delta}$,

$$A_{\phi_\varepsilon,\mathcal{X}} = g((\varepsilon\mathbf{\Delta})^2) = \sum_{\ell=0}^{\infty} \frac{g^{(\ell)}(0)}{\ell!}(\varepsilon\mathbf{\Delta})^{2\ell}. \tag{3.30}$$

Let $m = \mu(\mathcal{X}) - 1$, $\underline{\alpha} \in \ker(P_m^T)$, then by Lemma 3.5 $\underline{\alpha}^T\mathbf{\Delta}^{2\ell}\underline{\alpha} = 0$, for $\ell \leq m$. Therefore we have

$$\underline{\alpha}^T A_{\phi_\varepsilon,\mathcal{X}}\boldsymbol{\alpha} = \varepsilon^{2m}\frac{f^{(m)}(0)}{m!}\underline{\alpha}^T\mathbf{\Delta}^{2m}\underline{\alpha} + \sum_{\ell=\mu(X)}^{\infty} \varepsilon^{2\ell}\frac{f^{(\ell)}(0)}{\ell!}\underline{\alpha}^T\mathbf{\Delta}^{2\ell}\underline{\alpha} \tag{3.31}$$

which decays like ε^{2m} as $\varepsilon \to 0$. According to the Courant-Fischer's minimum-maximum principle [112, p.179], $A_{\phi_\varepsilon,\mathcal{X}}$ has at least $\dim(\ker(P_m^T))$ eigenvalues which decay at least as fast as ε^{2m}.

Further denote the space $\mathcal{M}_k \subseteq \ker(P_k^T)$ and $\mathcal{M}_k \perp \ker(P_{k+1}^T)$, then

$$\dim(\mathcal{M}_k) = \dim(\ker(P_k^T)) - \dim(\ker(P_{k+1}^T)). \tag{3.32}$$

Applying the same argument on the space $\mathcal{M}_k$, we can find at least $\dim(\mathcal{M}_k)$ eigenvalues which decay at least as ε^{2k} in the subspace $\mathcal{M}_k$. Since $\sum_{k=1}^{m}\dim(\mathcal{M}_k) = N$, there are exactly $\dim(\mathcal{M}_k)$ on the order of ε^{2k} when $\varepsilon \to 0$.

If every P_k is full rank, then $\operatorname{rank}(P_k) = \dim(\pi_{k-1}(\mathbb{R}^d)) = \binom{k-1+d}{d}$, and

$$\dim(\mathcal{M}_k) = \dim(\pi_k(\mathbb{R}^d)) - \dim(\pi_{k-1}(\mathbb{R}^d)) = \frac{(k+d-1)!}{k!(d-1)!}.$$

□

Note that the number $\binom{k-1+d}{d}$ is the number of terms in the expansion of $(x_1 + \cdots + x_d)^{k-1}$, which is consistent with the discussion above, whereas the term $\dim(\pi_k(\mathbb{R}^d)) - \dim(\pi_{k-1}(\mathbb{R}^d))$ depends on the geometric property, $\mu(\mathcal{X})$, of the interpolation data points. The geometric property is not easy to identify in general. For example, when the interpolation points lie on a circle with radial 1, then $\dim(\pi_k(\mathbb{R}^2)) - \dim(\pi_{k-1}(\mathbb{R}^2)) = 2$ for $k = 1, 2, 3$, because $\{1, x, y\}$ are independent, while for $\{x^2, xy, y^2\}$, we have $y^2 = 1 - x^2$, for $\{x^3, y^3, x^2y, y^2x\}$, we have $x^2y = y - y^3, xy^2 = x - x^2$. It was observed that when the eigenvalues of kernel matrices for Gaussian radial basis function on a circle are grouped in different orders, the numbers of each group in descending order are $1, 2, 2, 2, 2, \cdots$. The reader is directed to [88, p.389] for more scenarios. Figure 3.2(b) illustrates the eigenvalues of a kernel matrix for a Gaussian radial basis function on a circle.

3.2 Smoothness matching

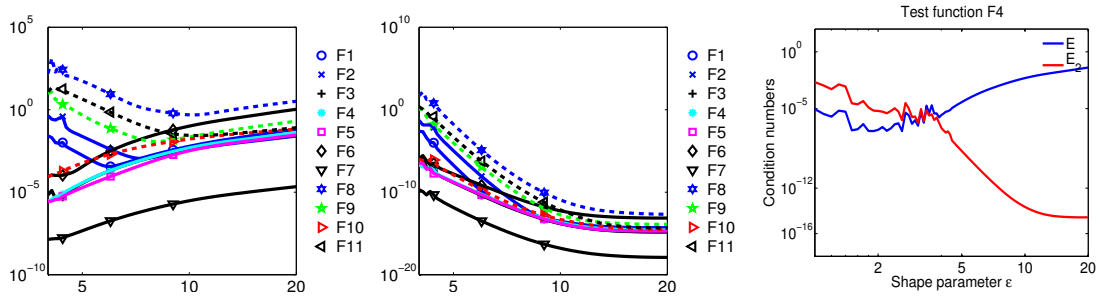
The previous section examined how fast the eigenvalues of an interpolation matrix decay given a radial basis function, showing that the eigenvalues of interpolation matrices with smoother basis functions decay faster, and in $\mathbb{R}^d$, there are many small eigenvalues. As is known, the sensitiveness of the interpolant to the input data, $\underline{f}$, depends on the smallest (in magnitude) eigenvalue of the interpolation matrix. In this section, we first examine the danger of many small eigenvalues, and then discuss how a good solution which can be well represented in a computer is closely related to the smoothness of the underlying kernel function and target function. Finally we introduce a principle to choose a basis function or to adjust the shape parameter for a basis function.

3.2.1 Error sources analysis

Without carefully dealing with the ill-conditioning, the expected approximation quality may be contaminated. Consider the following decomposition

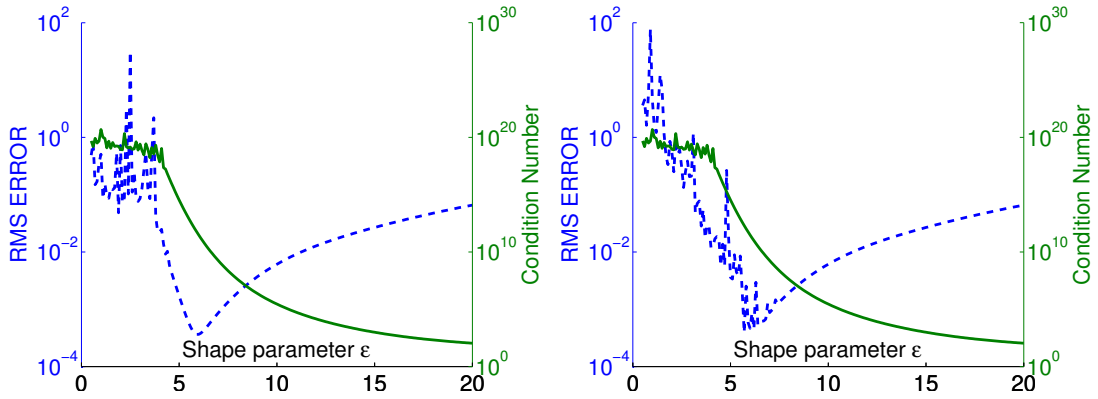
$$\|f(\mathbf{x}) - \tilde{s}(\mathbf{x})\| \leq \underbrace{\|f(\mathbf{x}) - s(\mathbf{x})\|}_{E_1} + \underbrace{\|s(\mathbf{x}) - \tilde{s}(\mathbf{x})\|}_{E_2}. \tag{3.33}$$

$s(\mathbf{x})$ interpolates $f(\mathbf{x})$ on the data set $\mathcal{X}$; $\tilde{s}(\mathbf{x}) = \sum_{i=1}^{N}(\alpha_i + \delta\alpha_i)\phi_j(\mathbf{x})$, where $\underline{\alpha}$ is the exact solution to the interpolation problem, and $\underline{\tilde{\alpha}} = \underline{\alpha} + \delta\underline{\alpha}$ is the solution due



(a) E (b) E_2 (c) E and E_2

Figure 3.3: Compare the overall evaluation error (E) and the error due to ill-conditioning (E_2). Interpolation 11 test functions on 289 Halton points with Gaussian radial basis functions $e^{-(\varepsilon r)^2}$.



(b) Double precision (c) Single precision

Figure 3.4: Condition number (green solid lines) and the overall root-mean-square error (blue dash lines). Interpolation the `franke` test function in Matlab on 289 Halton points with Gaussian radial basis function $e^{-(\varepsilon r)^2}$. The RMS errors are evaluated on 50×50 grid points.

to round-off or ill-conditioning. Approximation theory focuses on the estimation of the term, E_1. A reasonable scheme should prevent the term E_2 from contaminating the term E_1. Now we examine the term E_2.

Let the evaluation matrix on a finer point set $\mathcal{Y}$ with the interpolant $s(\mathbf{x})$ on $\mathbf{x}$ be $A_{\mathcal{Y},\mathcal{X}}$; its perturbation in finite floating point is $\tilde{A}_{\mathcal{Y},\mathcal{X}}$ and we assume that

$$\|A_{\mathcal{Y},\mathcal{X}} - \tilde{A}_{\mathcal{Y},\mathcal{X}}\| \leq \|A_{\mathcal{Y},\mathcal{X}}\| \epsilon_{\text{machine}}, \tag{3.34}$$

where $\epsilon_{\text{machine}}$ is the machine precision. Correspondingly, the interpolation matrix

and its perturbation are denoted as $A_{\mathcal{X},\mathcal{X}}$ and $\tilde{A}_{\mathcal{X},\mathcal{X}}$ respectively. Suppose the singular value decomposition of $A_{\mathcal{X},\mathcal{X}}$ is $A_{\mathcal{X},\mathcal{X}} = U\Sigma V^T$, where $\Sigma = \text{diag}\{\sigma_1, \ldots, \sigma_N\}$ is the diagonal matrix with positive singular values and U and V are orthogonal matrices. We assume that $\tilde{A}_{\mathcal{X},\mathcal{X}} = U\tilde{\Sigma}V^T$ so that only the singular values and not the singular vectors are perturbed. Such a simplification is not a rigorous mathematical model, it is only used for qualitative analysis. We assume the right hand side $\underline{\mathrm{f}}$ can be represented exactly in a computer and that the solution to $A_{\mathcal{X},\mathcal{X}}\underline{\alpha} = \underline{\mathrm{f}}$ and the solution to $\tilde{A}_{\mathcal{X},\mathcal{X}}\underline{\tilde{\alpha}} = \underline{\mathrm{f}}$ are

$$\underline{\alpha} = V\Sigma^{-1}U^T\underline{\mathrm{f}} = \sum_{i=1}^{N} \frac{\beta_i}{\sigma_i}\underline{\mathrm{v}}_i, \text{ and } \underline{\tilde{\alpha}} = \sum_{i=1}^{N} \frac{\beta_i}{\tilde{\sigma}_i}\underline{\mathrm{v}}_i, \tag{3.35}$$

where $\beta_i = \underline{\mathrm{u}}_i^T\underline{\mathrm{f}}$ and $\underline{\mathrm{u}}_i$ and $\underline{\mathrm{v}}_i$ are the i-th columns of U and V respectively (the left and right singular vectors).

For the term E_2, we have

$$E_2 = \|A_{\mathcal{Y},\mathcal{X}}\underline{\alpha} - A_{\mathcal{Y},\mathcal{X}}\underline{\tilde{\alpha}}\| = \|(A_{\mathcal{Y},\mathcal{X}} - \tilde{A}_{\mathcal{Y},\mathcal{X}})\underline{\tilde{\alpha}} - A_{\mathcal{Y},\mathcal{X}}(\underline{\alpha} - \underline{\tilde{\alpha}})\| \tag{3.36}$$

$$\leq \|A_{\mathcal{Y},\mathcal{X}}\|\|\underline{\tilde{\alpha}}\|\epsilon_{\text{machine}} + \|A_{\mathcal{Y},\mathcal{X}}\|\|\underline{\alpha} - \underline{\tilde{\alpha}}\|, \tag{3.37}$$

by using (3.34). Substituting (3.35) into (3.37), it follows that

$$E_2 \leq \|A_{\mathcal{Y},\mathcal{X}}\| \left(\epsilon_{\text{machine}} \sqrt{\sum_{i=1}^{N} (\frac{\beta_i}{\tilde{\sigma}_i})^2} + \sqrt{\sum_{i=1}^{N} \epsilon_i^2 (\frac{\beta_i}{\tilde{\sigma}_i})^2} \right), \tag{3.38}$$

where $\epsilon_i := \frac{\sigma_i - \tilde{\sigma}_i}{\sigma_i}$ is the relative perturbation or error of the i-th singular value. If each ϵ_i could achieve machine precision then

$$E_2 \leq 2\epsilon_{\text{machine}}\|A_{\mathcal{Y},\mathcal{X}}\| \sqrt{\sum_{i=1}^{N} (\frac{\beta_i}{\tilde{\sigma}_i})^2} = 2\epsilon_{\text{machine}}\|A_{\mathcal{Y},\mathcal{X}}\|\|\underline{\tilde{\alpha}}\|. \tag{3.39}$$

In practice, we use $\|A_{\mathcal{X},\mathcal{X}}\|$ rather than $\|A_{\mathcal{Y},\mathcal{X}}\|$ to estimate the upper bound of the term E_2.

Conclusion 3.1. Let $\underline{\tilde{\alpha}}$ be the numerical solution to $\tilde{A}_{\mathcal{X},\mathcal{X}}\underline{\tilde{\alpha}} = \underline{\mathrm{f}}$. The upper bound of the term E_2 can be estimated by $2\epsilon_{\text{machine}}\|A_{\mathcal{X},\mathcal{X}}\|\|\underline{\tilde{\alpha}}\|$.

If there are many small singular values $\tilde{\sigma}_i$ of $\tilde{A}_{\mathcal{X},\mathcal{X}}$, then $\|\underline{\tilde{\alpha}}\|$ will be big, and so will E_2. Figure 3.3 illustrates the overall evaluation error and the term E_2 for 11 test functions, where the functions are interpolated and evaluated on the same data sets. It is clearly seen that the term E_2 also depends on the underlying problem. Figure 3.4 demonstrates that there would be an "optimal" shape parameter ε^* to balance the term E_1 and E_2. We shall look at the problem from another perspective in § 3.2.4 on p. 54.

3.2.2 Picard conditioning

Another danger of many small eigenvalues or singular values is that the solution is more likely to loose the information in some frequencies. Suppose the singular value vector $\underline{\mathrm{v}}_i$ in (3.35) is normalised so that $\|\underline{\mathrm{v}}_i\|_\infty < 1$. If the range of values $\{\frac{\beta_i}{\sigma_i}\}_{i=1}^n$ spans too many scales, then it is likely that some information is unable to be captured due to finite precision representation. For example, consider only the first and the last singular value pairs; if

$$\log_{10}|\frac{\beta_n}{\sigma_n}| - \log_{10}|\frac{\beta_1}{\sigma_1}| = \log_{10}(\frac{\sigma_1}{\sigma_n}\frac{|\beta_n|}{|\beta_1|}) = \log_{10}\kappa(A) + \log_{10}(\frac{|\beta_n|}{|\beta_1|}) > 16, \qquad (3.40)$$

where $\kappa(A) = \frac{\sigma_1}{\sigma_n}$ is the condition number of A, then IEEE double precision is not enough to represent the sum of $\frac{\beta_1}{\sigma_1} + \frac{\beta_n}{\sigma_n}$. The sum can be represented when $\kappa(A) \leq 10^{16-\log_{10}\frac{|\beta_n|}{|\beta_1|}}$. It is too complicated to analyse the round-off in the solution (3.35) when considering all the singular vectors $\underline{\mathrm{v}}_i$. But so far, it is not a surprise that the approximation quality is not contaminated by other factors when $\kappa(A) \leq 10^{16-\gamma}$, for some $\gamma > 0$ in Figure 3.4.

One possible indicator for a good solution which can be well represented is that the $\underline{\alpha}$ in (3.35) should not blow up. The commonly used *Picard condition* in the context of solving ill-conditioned integral equations is used to described such an indicator. The Picard Condition, proved first by Picard in 1910 [49, p.160], is a necessary and sufficient condition for the existence of a square integrable solution to the Fredholm integral equation of the first kind (3.1). Suppose the kernel function in (3.1) has a *singular value expansion* (SVE),

$$K(\mathbf{x},\mathbf{y}) = \sum_{k=1}^{n} \sigma_k u_k(\mathbf{x}) v_k(\mathbf{y}), \qquad (3.41)$$

then the continuous solution $\alpha(\mathbf{y})$ to (3.1) can be represented as

$$\alpha(\mathbf{y}) = \sum_{l=1}^{\infty} \langle v_k(\mathbf{y}), \alpha(\mathbf{y}) \rangle v_k(\mathbf{y}) = \sum_{l=1}^{\infty} \frac{\langle u_k(\mathbf{x}), f(\mathbf{x}) \rangle}{\sigma_k} v_k(\mathbf{y}). \tag{3.42}$$

The Picard condition reads as follows.

Condition 1 (Picard Condition [49, III.§11][104, p.12],[125, p.279])**.** *An integral equation of the first kind* (3.1) *has an unique square integrable solution* $\alpha(\mathbf{y})$ *if and only if the series* (3.42) *satisfies*

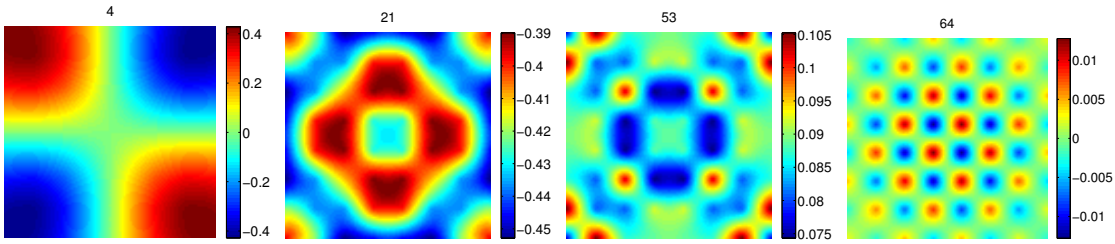
$$\|\alpha(\mathbf{y})\|^2 = \langle \alpha(\mathbf{y}), \alpha(\mathbf{y}) \rangle = \sum_{k=1}^{\infty} \frac{\langle u_k(\mathbf{x}), f(\mathbf{x}) \rangle^2}{\sigma_k^2} < \infty. \tag{3.43}$$

The SVE is a generalization of an eigenfunction or Fourier expansion of a symmetric kernel function [188]. When K is symmetric positive definite, $u_k = v_k$, the SVE (3.41) is reduced to the eigenfunction expansion, and the absolute and uniform convergence of (3.41) is guaranteed by Mercer's theorem [147]. For finite dimensional operators (matrices), the SVE is the SVD.

Furthermore, $\langle u_k(\mathbf{x}), f(\mathbf{x}) \rangle$ can be viewed as the generalized Fourier coefficients of f with respect to the orthogonal system $\{u_k\}_{k=1}^{\infty}$ [49, p.160]. Large singular values correspond to low frequency singular functions and small singular values correspond to high frequency singular functions, see Figure 3.5 for illustration. In this sense, the magnitudes $|\langle u_k(\mathbf{x}), f(\mathbf{x}) \rangle|$ determine the *smoothness* of f with respect to the basis $\{u_k(\mathbf{x})\}_k^{\infty}$; On the other hand, as we have seen in Section 3.1, the decay of the eigenvalues (and thus singular values for symmetric kernels) is closely related to the *smoothness* of the kernel functions.

As seen in the Hill-Tamarkin Theorem 3.3, the decay order of the eigenvalues of a symmetric kernel has a close relationship with the decay order of the Fourier coefficients. Furthermore Lemma 3.4 says the decay order of the Fourier coefficients of a function depends on the smoothness of the underlying function. Both are related to the decay order of the Fourier transform of the function. The decay order of the Fourier transform can also be used to characterize the *Sobolev space* [208, p.141]. The *Sobolev space of order* s, $s > d/2$, $\mathcal{H}^s(\mathbb{R}^d)$ is defined as

$$\mathcal{H}^s(\mathbb{R}^d) := \{f \in L_2(\mathbb{R}^d) \cap C(\mathbb{R}^d) : \hat{f}(\omega)(1 + \|\omega\|^2)^{s/2} \in L_2(\mathbb{R}^d)\}. \tag{3.44}$$



(a) $k = 4$ (b) $k = 21$ (c) $k = 53$ (d) $k = 64$

Figure 3.5: The k-th eigenfunction of an interpolation matrix with Gaussian radial basis function $e^{-(8r)^2}$ on a 8×8 regular mesh on the unit square. Smaller eigenvalues correspond to higher frequency eigenfunctions.

Let the kernel function $K \in \mathcal{H}^{s_1}$ and $f \in \mathcal{H}^{s_2}$. If we use this decay order as that of σ_k and $\langle u_k(\mathbf{y}), f(\mathbf{y})\rangle$ in (3.43), then the Picard condition essentially says a well-conditioned solution should satisfy $s_2 > s_1$, and $\mathcal{H}^{s_2} \subset \mathcal{H}^{s_1}$. Because $\mathcal{H}^s$ is equivalent to the space W_2^s, the set of square integrable functions which have weak derivatives up to order s. Therefore larger s implies that functions in $\mathcal{H}^s$ are smoother. Here we arrive at the *smoothness matching condition.*

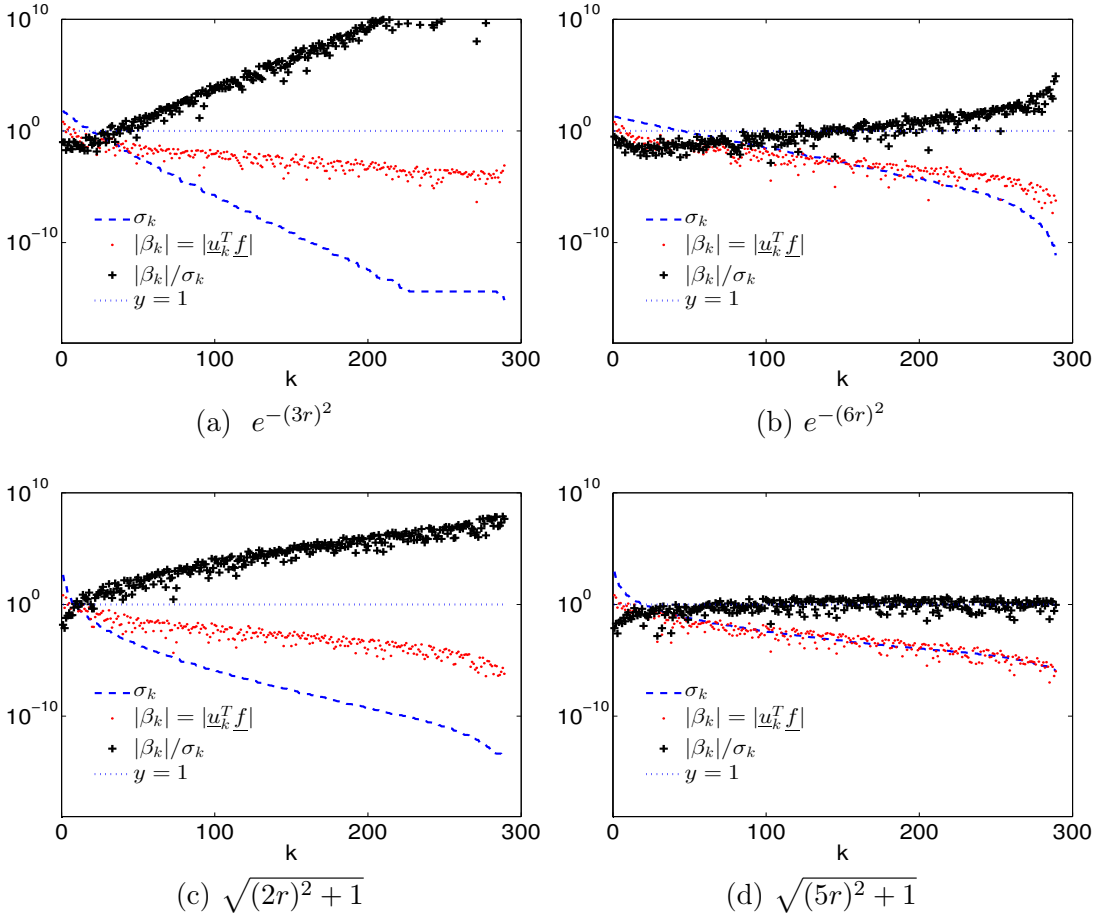
Condition 2. *A well conditioned solution to the Fredholm integral equation of the first kind* (3.1) *or the interpolation problem* (1.1) *requires that the smoothness of the target function f should be no less than the smoothness of the kernel function or basis functions.*

If the properties of the target function are known in advance, and there is room for us to choose the basis function, condition 2 is a useful criteria for us to choose a proper basis function. Otherwise, we have to use an alternative way to avoid bad solutions.

3.2.3 The Discrete Picard Condition

In practice, the singular value decomposition (SVD) rather than SVE is employed to investigate the ill-posedness of the discrete problem (1.1). Recall that the solution $\underline{\alpha}$ can be represented as a combination of the singular vectors as in (3.35) and thus

$$\|\underline{\alpha}\|_2^2 = \sum_{k=1}^{N} \frac{(\underline{\mathrm{u}}_k^T \underline{\mathrm{f}})^2}{\sigma_k^2}, \tag{3.45}$$



(a) $e^{-(3r)^2}$ (b) $e^{-(6r)^2}$

(c) $\sqrt{(2r)^2+1}$ (d) $\sqrt{(5r)^2+1}$

Figure 3.6: The Picard plot of interpolation the test `franke` function with Gaussian radial basis function and multiquadrics on a 17×17 mesh on the unit square. Where σ_k and $\underline{u}_k$ are the k-th singular value and left singular vector of the interpolation matrix $A_{\phi_\varepsilon,\mathcal{X}}$ or $A_{\mathcal{X},\mathcal{X}}$. Where $A_{\mathcal{X},\mathcal{X}}\underline{\alpha} = \underline{f}$, and $\|\underline{\alpha}\| = \sum_{k=1}^{N} \beta_k^2/\sigma_k^2$. When the cases (a) and (c) happen, $\|\underline{\alpha}\|$ will blow up, a truncated SVD or other regularizations are usually used. Here we can adjust the shape parameter so that the decay order of singular values σ_k of the kernel matrix (the smoothness of the kernel functions) matches the decay order of $|\beta_k|$ (the smoothness of the target function).

Analogously to the continuous case, Hansen [103] describes the Picard condition for the discrete case.

Condition 3 (Discrete Picard Condition [104, p.37])**.** *Suppose that there exist r and τ such that numerically $\sigma_k = O(\tau), \forall\, k > r$. Then the discrete Picard condition (DPC), the restatement of the continuous Picard condition for the square integrability of the solution α, is said to be satisfied if, for all singular values which are larger than τ, the corresponding coefficients $|\underline{u}_k^T \underline{f}|$, on average, decay faster than σ_k.*

The DPC can be illustrated by the Picard plot, see Figure 3.6. As illustrated in Figure 3.6, in the context of radial basis function interpolation, there is room to choose a kernel function by adjusting the shape parameter, so that the smoothness of the kernel function can match that of the target function. As such, a natural question immediately arises: how to choose the right shape parameter? Looking at this problem from a statistical perspective can bring easier solutions.

3.2.4 Bias-variance trade-off principle

Recall the one dimensional kernel based density function estimation formula in (1.13):

$$\tilde{f}(x) = \frac{1}{nh}\sum_{i=1}^{n} K(\frac{x - X_i}{h}). \tag{3.46}$$

Choosing a proper shape parameter is equivalent to choosing a proper window width or smoothing parameter h. The *bias-variance trade-off principle* is widely used to choose a proper window width.

The *bias* of a kernel estimator is the difference between the mean of the estimator and the true mean of the function:

$$\text{bias}_h = b_h(x) = E(\tilde{f}(x)) - f(x)). \tag{3.47}$$

The variance of the estimator is given by

$$\text{var}(\tilde{f}(x)) = E((\tilde{f}(x) - E\tilde{f}(x))^2). \tag{3.48}$$

The *mean integrated square error* (MISE) which is also called *risk* in statistics, is used to measure the global accuracy of the kernel estimator. MISE can be decomposed into two parts:

$$\text{MISE}_x(\tilde{f}) = \text{Risk}(f, \tilde{f}) = E\int \{\tilde{f}(x) - f(x)\}^2 dx = \int\ b_h(x)^2 dx + \int \text{var}\,\tilde{f}(x) dx. \tag{3.49}$$

To get an optimal accuracy, we have to minimize the MISE or risk. Under mild assumption of the kernel function and f, the bias term satisfies [204, p.314]

$$\int b_h(x)^2 dx \approx \frac{1}{4}h^4 \int f''(x)dx \left(\int xk(x)dx\right)^2, \tag{3.50}$$

and the variance term satisfies

$$\int \text{var}\, \tilde{f}(x)dx \approx \frac{\int K^2(x)dx}{nh}. \tag{3.51}$$

An optimal smoothing parameter or window width is obtained by balancing the bias and variance. Such a balance between the bias term and variance term is called the *bias-variance trade-off principle*.

For example, the optimal window width for this one dimensional kernel estimator is given by [204, p. 314]

$$h^* = \frac{c_1^{-2/5} c_2^{1/5} c_3^{-1/5}}{n^{1/5}}, \tag{3.52}$$

where $c_1 = \int x^2 K(x)dx$, $c_2 = \int K(x)^2 dx$ and $c_3 = \int (f''(x))^2 dx$.

For high dimensional density function estimation, there are quite a few techniques to get a proper window width or smoothing parameter. The idea is to define a proper risk function and minimize the risk. We shall see how to use the cross-validation methods to minimize a risk function so that a optimal shape parameter for radial basis functions is obtained.

3.3 Discussion

There are a number of pioneering results on estimating the smallest (in magnitude, hereafter) eigenvalue. There are two fundamental motivations. First, to understand the solvability of a high-dimensional interpolation problem with certain radial basis functions, one has to prove that the related matrices do not have any zero eigenvalues. For example, Ball proves [5] that the lower bound of the smallest eigenvalue of a distance matrix is bounded away from zero. And Micchelli proves that a class of conditional negative definite radial basis functions leads to non-singular interpolation matrices [148]. Second, to understand the stability and sensitivity of the interpolation system, one has to prove that the norm of the inverse of the interpolation matrix is bounded. Such an upper bound depends on the lower bound of the smallest eigenvalue of the underlying interpolation matrix. There are quite a few papers on estimation the lower bound of the smallest eigenvalues, for example see [8, 152, 153, 154, 176, 177]. There is also one paper which discusses an upper bound on the smallest eigenvalues of kernel matrices [6]: it gives the lower bound of the norm of the inverse of the interpolation matrix on a regular mesh, and

also shows how ill-conditioned the regular mesh is. The proofs of these results are complicated.

The Picard condition is usually used to evaluate the degree of ill-posedness of a problem, see [56, 104, 125, 199, 200] for example. The well-posedness of solutions to (3.1) is analysed via the singular value expansion [188].

Here we start with a concrete linear algebra problem. We prove the eigenvalues' distribution of the interpolation matrix with limited use of the Fourier transform. With the help of the SVD and the Picard condition, we derive a condition to guarantee a solution which can be well represented on a computer, identifying that the smoothness of the basis functions should match with the smoothness of the target function. Form this point of view, if the problem is ill-conditioned, then it is possibly because the target function, f, is in a larger space than that the basis functions can recover. In such a case, one can adjust the shape parameter of the basis function to make those two spaces closer. The risk function used in the bias-variance principle seems to be one way to measure such distance between the target function and the space which the kernel function can recover.

4. Convexity and solvability of CSRBFs with different shapes

As seen in Section 1.2, radial basis functions are promising candidates for high dimensional scattered data approximation. To reduce the curse of high dimensionality, it has taken a number of mathematicians a long time to seek and construct compactly supported radial basis functions (CSRBFs). When CSRBFs have been found, it is natural to ask how well the CSRBFs can reconstruct a function, how to use CSRBFs adaptively to further reduce high dimensionality and whether there are some scalable algorithms for truly large scale scattered data. Negative results were soon obtained from classical finite element results. It is more difficult for CSRBFs to balance with accuracy and scalability than other methods, but remedies exist. A positive and new result is that using CSRBFs with different shapes at the same time is possible.

4.1 Accuracy and scalability

Let ϕ be a basis function (not necessarily radial). Consider the following approximation in $\mathcal{H}^s$ by the expansion of translates of ϕ:

$$\sum_{\mathbf{j}\in\mathbb{Z}^d} \alpha_{\mathbf{j}}^h \phi_{\mathbf{j}}^h(\mathbf{x}), \text{ with } \phi_{\mathbf{j}}^h = \frac{1}{h_w^{d/2}} \phi\left(\frac{\mathbf{x}-\mathbf{j}h}{h_w}\right), \tag{4.1}$$

where h_w is referred as the window width, and h is the grid length. By default, $h_w = h$.

Definition 4.1 (Strang-Fix)**.** *The basis function ϕ satisfies the Strang-Fix condition of order m if its Fourier transform $\hat{\phi}$ is such that*

$$\hat{\phi}(0) \neq 0 \tag{4.2}$$

and $\hat{\phi}$ *has zeros of order at least* $m+1$ *at points* $\xi = 2\pi\mathbf{j}$ *for* $\mathbf{j} \in \mathbb{Z}^d \backslash \{0\}$, *i.e.*

$$D^{\mathbf{k}}\hat{\phi}(\xi) = 0 \text{ for } |\mathbf{k}| < m, \tag{4.3}$$

where

$$D^{\mathbf{k}}\phi = (\frac{\partial}{\partial x_1})^{k_1} \cdots (\frac{\partial}{\partial x_d})^{k_d}\phi. \tag{4.4}$$

This condition comes from condition (i) in [192, Thm I] (the simplest case when the "trial space" is generated by only one basis function). Such a Strang-Fix condition is a necessary and sufficient condition for (local) polynomial reproduction of order m for ϕ, i.e. all polynomials in $\pi_m(\mathbb{R}^d)$ can be written as linear combinations of ϕ and its translates [192, p.808]. Local polynomial reproduction plays a central role in local error estimation for radial basis function estimations, see [208, Chapter 3]. For any p, $1 \leq p \leq \infty$, the approximant in (4.1) provides a local L_p-approximation of order m *if and only if* ϕ satisfies the Strang-Fix condition of order m [52][118, Thm 3]. Since the Fourier transform of any positive definite function is positive, therefore positive definite kernels alone can not reproduce polynomials of any order. Thus L_p convergence can not be guaranteed. Also there is no positive definite compactly supported radial basis function which satisfies the Strang-Fix condition in a space of dimension greater or equal to 2 [214, Thm 2.3]. In particular, Buhmann in fact proves that the positive definite kernel, the Gaussian, does not recover a constant function [29, Thm. 23].

We should be very careful to interpret such a result when comparing with available convergence results for positive definite kernels. The Strang-Fix condition applies to the cases when the scaling parameter h_w in (4.1) is proportional to the length of the grids h, such an approach is referred to as the *stationary* interpolation. The stationary interpolation with positive definite kernels does not guarantee convergence. Available convergence results for compactly supported positive definite kernels (for example [208, Thm 11.17]) in the author's interpretation are for the *nonstationary* approach, where the support of each basis function is kept as a constant, when the data set is refined.

An interesting case is the interpolation with Gaussian with flat limit, $\phi(r) = e^{-(\varepsilon r)^2}$ with $\varepsilon \to 0$. Its Fourier transform, $\varepsilon^{-d}e^{-\frac{|\omega|^2}{\varepsilon^2}}$, tends to the delta function which does not satisfy the Strang-Fix condition of any order, but does asymptoti-

cally satisfies the Strang-Fix condition of arbitrary order in the sense that $D^{\mathbf{k}}\hat{\phi}(\xi)$ can arbitrarily tends to 0 for any $\xi \in \mathbb{Z}^d \backslash \{0\}$. In this case, the Gaussian tends to reproduce arbitrarily order polynomials, which agrees with Schaback's conclusion that Gaussian interpolation with flat limit tends to the de Boor-Ron least polynomial interpolate [53, 54, 55]. There is no surprise that the flat limit case receives a lot of attention. So far, we would like to go back to compare the results in Figure 3.4; the double precision results and the single precision results make no significant difference on the trend of the approximation error, so there must be an essential mathematical problem besides ill-conditioning. The reason lies in that, even for a non-stationary approach, interpolation with the Gaussian does not guarantee uniform convergence, but can achieve high order approximation up to some *saturation error*; by adjusting the shape parameter the saturation error can be made negligible in numerical applications [146]. This could explain in Figure 3.4 on page 48 why there is no essential difference on the trends of the evaluation error.

For globally supported radial basis functions, there is no difference on the storage requirements between the stationary approach and the nonstationary approach, whereas for CSRBFs, there is a significant disparity.

For the stationary approach, as each basis function has a window width proportional to h, thus the non-zero elements of the interpolation matrix for each basis function does not increase. The storage for the interpolation matrix is just $\mathcal{O}(N)$, where N is the number of interpolation points, while for the non-stationary approach, the number of interpolation points which are covered by each basis function increase proportionally to N, and thus the storage increases as $\mathcal{O}(N^2)$. One can slightly decrease the support as the number of interpolation points increases, and this will lead to a storage increase only at the order of $\mathcal{O}(N^{2-\delta})$, $0 \leq \delta < 1$.

Observation 5. The storage for stationary interpolation with radial basis function increases at the order of $\mathcal{O}(N)$ as the number of points $N \to \infty$; the storage for non-stationary interpolation with radial basis function increases at the order of $\mathcal{O}(N^{2-\delta})$ for some $0 \leq \delta < 1$.

Here comes the dilemma, the stationary interpolation is scalable and thus potentially useful for large data sets but it doesn't converge. Once we figure out why stationary interpolation does not converge, there might be some remedies.

The first way is to add low order polynomials like those conditionally positive/negative definite basis function like multiquadrics as in (2.7). For only recovering the constant function, there is no problem. If the order of the polynomial is high, we are not sure whether the interpolation matrix (2.9) is invertible on the scattered data set.

The second way is based on the *partition of unity* approach to construct functions which at least can recover constants. For example, one can construct function like

$$\Psi(\mathbf{x}) = \frac{\phi(\|\mathbf{x} - \mathbf{x}_i\|/\delta)}{\sum_{j\in\{j:\|\mathbf{x}-\mathbf{x}_j\|\leq\delta\}} \phi(\|\mathbf{x} - \mathbf{x}_j\|/\delta)}. \tag{4.5}$$

A generalization of this approach leads to the *moving least square* approach [208, Chap.4]. Such a scheme satisfies the Strang-Fix condition [133, 215] and is often referred as the *moving least square reproducing kernel method* (MLSRK) in engineering [138].

For the third approach, we attempt to explain the failure of stationary interpolation according to the *smoothness matching condition*. When $h \to 0$ in (4.1), the combinations of the basis functions can only capture the high frequency information and the low frequency information is filtered out. It is impossible to recover a target function with limited frequency information. From this perspective, the low frequency information is easy to obtain by using a subset of the interpolation points with larger support. This leads to the multi-step or multilevel methods [79, 151]. In a coarse data set, a wider basis function is use to capture the low frequency information, and in a fine data set, the support becomes smaller to capture high frequency information. In this case the approximation spaces are generated by basis functions with multi-scales. Recent analysis shows that multilevel stationary interpolation can converge [209].

As such remedies are available to balance the accuracy and scalability, it is natural to ask how to choose the scale of the basis function for truly scattered data and is it possible to use various scales at the same level.

As shown in Section 2.1, interpolation with radial basis functions of the same scale can guarantee a non-singular interpolation matrix, whereas, if the interpolation points and the basis function centres are different, or the basis functions are of different scales or are different kinds of functions, the formula (2.4) does not hold

and there is no such guarantee. An example of the latter case, using different kinds of radial basis functions, is the original unsymmetrical collocation method for solving partial differential equations (PDEs) with multiquadrics [120], where Kansa uses basis functions of different scales or different combinations at each centre.

Here, we focus on compactly supported radial basis functions [208, 213]. Sufficient conditions to guarantee a unique solution to the RBF interpolation problem with different shapes are supplied. The conditions only depend on a *local convexity* of the underlying basis function, and on a *local geometric property* of a neighbourhood of each centre. Information on both aspects are easy to obtain.

4.2 Results on convexity

A function is said to be convex on $[a, b]$, if $\forall\, x_1, x_2 \in [a, b]$ and $\forall\, t \in [0, 1]$: $f(tx_1 + (1-t)x_2) \leq tf(x_1) + (1-t)f(x_2)$. Let $f(r) \in C^2[a, b]$, if $f'(r) < 0$ and $f''(r) \geq 0$ for all $r \in (a, b)$, then f is convex on (a, b). This section focuses on one popular class of compactly supported radial basis functions—Wendland functions.

Recall that the Wendland function $\phi_{d,k}$ is defined as

$$\phi_{d,k} = \mathcal{I}^k \phi_{\lfloor d/2 \rfloor + k + 1}, \tag{4.6}$$

where $\phi_\ell(r) = \max\{(1-r)^\ell, 0\}$, $\lfloor \cdot \rfloor$ is the floor operator, and $\mathcal{I}$ is the integral operator defined as

$$\mathcal{I}(\phi)(r) := \int_r^\infty t\phi(t)dt. \tag{4.7}$$

Here, $\phi_{d,k}(\|\mathbf{x}\|)$ is a compactly supported radial basis function and $\phi_{d,k} \in C^{2k}(\mathbb{R}^d)$. Another operator is defined as

$$\mathcal{D}\phi(r) = -\frac{1}{r}\phi'(r), \text{ for } \phi \in C^2(\mathbb{R}). \tag{4.8}$$

It is known that $\mathcal{D}\mathcal{I}\phi = \phi$ and $\mathcal{I}\mathcal{D}\phi = \phi$ [208, p.121]. Applying the formula (4.8), we can get the following properties of functions defined by (4.7).

Lemma 4.1. *If $\phi \in L_1[0, \infty)$ is a non-negative function, then,*

1. $\frac{d}{dr}\mathcal{I}^k\phi(r) = -r\mathcal{I}^{k-1}\phi(r) \leq 0$, *for* $k \geq 1$;
2. $\frac{d^2}{dr^2}\mathcal{I}^k\phi(r) = r^2\mathcal{I}^{k-2}\phi(r) - \mathcal{I}^{k-1}\phi(r)$, *for* $k \geq 2$.

Proof. Because ϕ is non-negative, by the definition of the operator $\mathcal{I}$ in (4.7), $\mathcal{I}^k\phi(r)$ is non-negative. Employing the formula (4.8), we can get

$$\frac{d}{dr}\mathcal{I}^k\phi(r) = -r\mathcal{D}\mathcal{I}^k\phi(r) = -r\mathcal{I}^{k-1}\phi(r) \leq 0.$$

The second part holds because

$$\frac{d^2}{dr^2}\mathcal{I}^k = \frac{d}{dr}(-r\mathcal{I}^{k-1}\phi(r)) = r^2\mathcal{I}^{k-2}\phi(r) - \mathcal{I}^{k-1}\phi(r).$$

□

The first part of Lemma 4.1 demonstrates that Wendland functions are non-increasing on $[0, 1]$, and the second part gives a recurrence formula to compute the second derivative of a Wendland function.

Lemma 4.2. *If $\phi(r)$ is a continuous and monotonically decreasing function on $[0, 1]$, which satisfies $\phi(0) > 0$ and $\phi(1) = 0$, then for the function*

$$f(r) = r^2\phi(r) - \int_r^1 t\phi(t)dt,$$

there exists a positive number $\gamma \in (0, 1)$ such that $f(r) \geq 0$ on $[\gamma, 1]$.

Proof. Since ϕ is a monotonically decreasing function on $[0, 1]$, then for any $t \in (r, 1)$, $0 < \phi(t) < \phi(r)$. By the Cauchy-Schwarz inequality,

$$\begin{aligned}\int_r^1 t\phi(t)dt &\leq \left(\int_r^1 t^2dt\right)^{1/2}\left(\int_r^1 \phi^2(t)dt\right)^{1/2} \\ &\leq \sqrt{\frac{1-r^3}{3}}\left(\phi(r)\sqrt{1-r}\right) \leq \phi(r)\sqrt{\frac{1-r^3}{3}}.\end{aligned}$$

Therefore

$$f(r) \geq \left(r^2 - \sqrt{\frac{1-r^3}{3}}\right)\phi(r) = h(r)\phi(r). \tag{4.9}$$

Compute $h'(r) = 2r + \frac{\sqrt{3}r^2}{2\sqrt{1-r^3}} > 0$ on $(0, 1)$. On the other hand, $h(0) = -\sqrt{1/3}$ and $h(1) = 1 > 0$. Therefore, according to the intermediate value theorem for continuous functions, there must exists a positive number $\gamma \in (0, 1)$, such that $h(\gamma) = 0$ and $h(r) > 0$ on $(\gamma, 1)$. Thus $f(r) \geq 0$ on $[\gamma, 1]$. □

Theorem 4.1. *For any Wendland function $\phi_{d,k}$ defined above, there exists a positive real number $q \in [0,1)$, such that $\phi_{d,k}$ is convex on $[q,1]$.*

Proof. If $k=0$, $\phi_{d,0} = (1-r)^{\lfloor d/2 \rfloor + 1}$. $\phi'_{d,0}(r) = -1(\lfloor d/2 \rfloor + 1)(1-r)^{\lfloor d/2 \rfloor} < 0$ on $(0,1)$. If $d=1$ $\phi''_{d,0}(r) = 0$, otherwise $\phi''_{d,0}(r) = (-1)^2(\lfloor d/2 \rfloor + 1)(\lfloor d/2 \rfloor)(1-r)^{\lfloor d/2 \rfloor - 1} \geq 0$. Thus $\phi_{d,0}(r)$ is convex on $[0,1]$.

For $k=1$, $\phi_{d,1}(r) = \int_r^1 t(1-t)^{\lfloor d/2 \rfloor + 1 + 1} dt$, $\phi'_{d,1}(r) = -r(1-r)^{\lfloor d/2 \rfloor + 2} \leq 0$ in $(0,1)$,

$$\phi''_{d,1}(r) = (1-r)^{\lfloor d/2 \rfloor + 1} \left((\lfloor d/2 \rfloor + 3)r - 1 \right).$$

Let $q = \frac{1}{\lfloor d/2 \rfloor + 3}$, then $\phi''_{d,1}(r) \geq 0$ on $[q,1]$.

For $k \geq 2$, by Lemma 4.1, $\phi'_{d,k}(r) \leq 0$ and

$$\phi''_{d,k}(r) = r^2 \mathcal{I}^{k-2} \phi_\ell(r) - \mathcal{I}^{k-1} \phi_\ell(r)$$

where $\ell = \lfloor d/2 \rfloor + k + 1$ and $\phi_\ell(r) = (1-r)_+^\ell$. Define $\varphi(r) = \mathcal{I}^{k-2}\phi_\ell(r)$, then $\varphi(r)$ is a decreasing and non-negative function on $[0,1]$. Note that $\mathcal{I}^{k-1}\phi_\ell(r) = \int_r^1 t\varphi(t)dt$. Employing Lemma 4.2, there exists a positive $0 < \gamma < 1$, such that $\phi''_{d,k}(r)$ is positive on $[\gamma, 1]$. Let $q = \gamma$, which finishes the proof. □

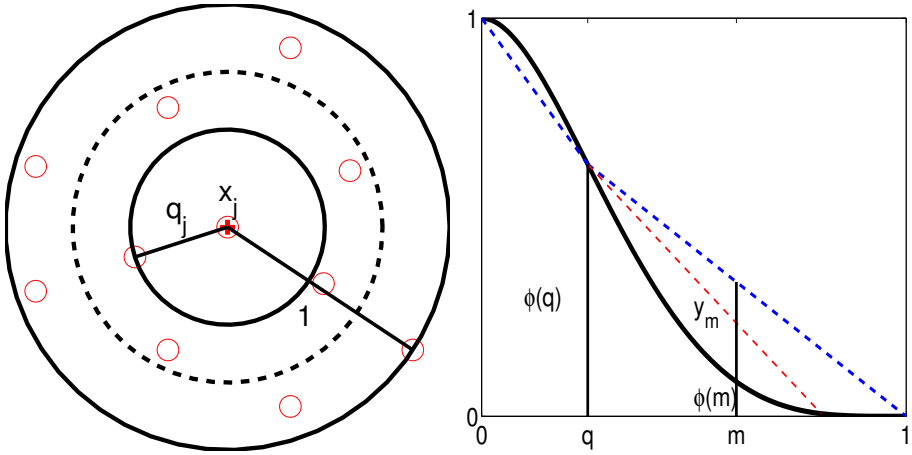
Lemma 4.3. *For any Wendland function $\phi_{d,k}(r)$, there exist a positive number m and c, such that for $r \in (\delta_1, 1)$, $\phi_{d,k}(r) \leq c(1-r)^m$.*

Proof. Because $\phi_{d,k}$ vanishes at 1, thus $\phi_{d,k} = (1-r)^m p(r)$ on $[0,1]$ (see (2.16)), where $p(r)$ is a polynomial with non-zeros on $[0,1]$. Let $c = \max_{r \in [0,1]} p(r)$, since $\phi_{d,k}$ is positive on $[0,1)$, thus $c > 0$. □

4.3 Results on solvability

Let $\mathcal{N}(\mathbf{x}_j, n_j) \subset \mathcal{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N\} \subset \mathbb{R}^d$, be the set of the n_j nearest neighbour points to $\mathbf{x}_j$, where $\mathbf{x}_j \notin \mathcal{N}(\mathbf{x}_j, n_j)$. For each $\mathcal{N}(\mathbf{x}_j, n_j)$, define

$$r_j = \min_{\mathbf{x}_k \in \mathcal{N}(\mathbf{x}_j, n_j)} \|\mathbf{x}_k - \mathbf{x}_j\| \quad \text{and} \quad R_j = \max_{\mathbf{x}_k \in \mathcal{N}(\mathbf{x}_j, n_j)} \|\mathbf{x}_k - \mathbf{x}_j\|. \tag{4.10}$$



(a) Scaled neighbourhood of $\mathbf{x}_j$. (b) Wendland function $\phi_{3,1}$.

Figure 4.1: (a) illustrates a *scaled* neighbourhood of $\mathbf{x}_j$ with 10 neighbour points. (b) illustrates a Wendland function which is convex in $(q, 1)$. One can see that $\frac{y_m}{\phi(q)} = \frac{1-m}{1-q}$ holds, which is used to prove Theorem 4.2.

It is obvious that $\mathcal{N}(\mathbf{x}_j, n_j) \subset \mathcal{B}(\mathbf{x}_j, R_j) \backslash \mathcal{B}(\mathbf{x}_j, r_j)$, where $\mathcal{B}(\mathbf{x}, R)$ is a Euclidean ball with center $\mathbf{x}$ and radius R. Further, define

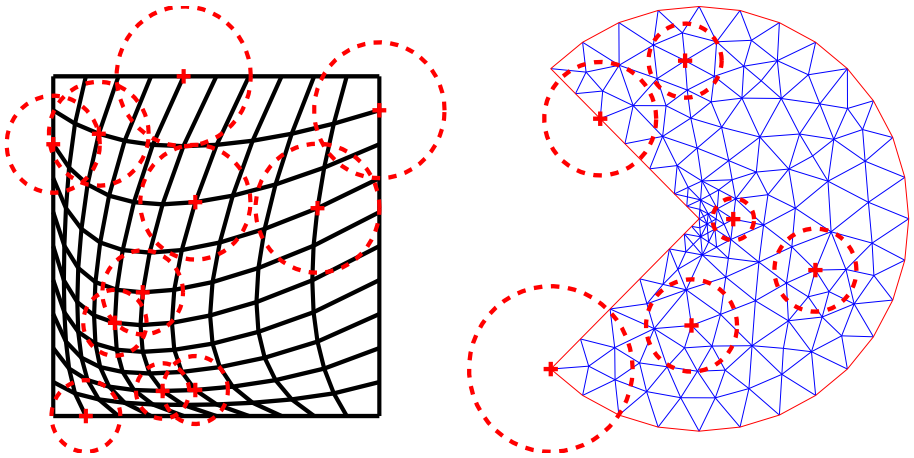
$$q_j = \frac{r_j}{R_j}, \text{ and } m_j = \frac{1}{n_j R_j} \sum_{\mathbf{x}_k \in \mathcal{N}(\mathbf{x}_j, n_j)} \|\mathbf{x}_k - \mathbf{x}_j\|. \tag{4.11}$$

It is easy to find that $0 < q_j \leq m_j \leq 1$, the equalities holding only when all of the points in $\mathcal{N}(\mathbf{x}_j, n_j)$ are located on the same sphere with $\mathbf{x}_j$ as center (including the special case $n_j = 1$). The quantities q_j and m_j are associated with the particular geometric property of $\mathcal{N}(\mathbf{x}_j, n_j)$: the points in $\mathcal{N}(\mathbf{x}_j, n_j)$ are located in $\mathcal{B}(\mathbf{x}_j, R_j) \backslash \mathcal{B}(\mathbf{x}_j, r_j)$, and can be viewed as distributed on the sphere $\|\mathbf{x}_j - \mathbf{x}\|_2 = m_j R_j$ on average. Such a local geometric property only depends on the relative distances between $\mathbf{x}_j$ and points in $\mathcal{N}(\mathbf{x}_j, n_j)$.

Denote the radial basis functions $\varphi(\|\mathbf{x}\|)$ with different scales, ϵ_j, as $\phi_{\epsilon_j}(\mathbf{x}) = \varphi(\epsilon_j \|\mathbf{x}\|)$, for $1 \leq j \leq n$. The interpolant is written as

$$s(\mathbf{x}) = \sum_{j=1}^{N} \alpha_j \phi_{\epsilon_j}(\mathbf{x}) \tag{4.12}$$

and the corresponding interpolation matrix on the data set $\mathcal{X}$ with $\{\phi_{\epsilon_j}\}_{j=1}^{n}$ is



(a) $\mathcal{N}(\mathbf{x}_j, 6)$ for a distort grid grid (b) $\mathcal{N}(\mathbf{x}_j, 7)$ for a FEM mesh

Figure 4.2: A demonstration of radial basis functions with different supports for non-uniform grid.

denoted as $A_{\phi_{\epsilon_j},\mathcal{X}}$.

Theorem 4.2. *Let $\phi(r)$ be a univariate compactly supported radial basis function on $[0,1]$ which is convex on $[q,1]$ and satisfies $\phi(0) = 1$. $\mathcal{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N\}$ is a scattered data set in $\mathbb{R}^d$. For a set $\mathcal{N}(\mathbf{x}_j, n_j)$ of n_j nearest neighbour points. R_j is defined by (4.10) and q_j and m_j are defined by (4.11). If for every $\mathbf{x}_j \in \mathcal{X}$, there is a set $\mathcal{N}(\mathbf{x}_j, n_j)$ of n_j nearest neighbouring points which satisfies $q_j \geq q$ and $m_j > 1 - \frac{1-q}{\phi(q)n_j}$, then the interpolation matrix $A_{\phi_{\epsilon_j},\mathcal{X}} = \phi(\epsilon_j \|\mathbf{x}_i - \mathbf{x}_j\|)_{1\leq i,j\leq N}$ is non-singular when $\epsilon_j \geq 1/R_j$.*

Proof. For $\mathbf{x}_k \in \mathcal{N}(\mathbf{x}_j, n_j)$, define $d_{kj} = \frac{\|\mathbf{x}_k - \mathbf{x}_j\|}{R_j}$. Since $q_j = \frac{r_j}{R_j} \geq q$, then for each k and j, $d_{kj} \geq q_j \geq q$. Suppose now y_{kj} is defined so that the points (d_{kj}, y_{kj}) are located on the straight line which passes through $(q, \phi(q))$, and $(1, 0)$, then $y_{kj} = \frac{1-d_{kj}}{1-q}\phi(q)$. Since $d_{kj} \in (q_j, 1) \subset [q, 1]$ and ϕ is convex on $[q, 1]$, then $y_{kj} \geq \phi(d_{kj})$. It follows that

$$\sum_{\mathbf{x}_k \in \mathcal{N}(\mathbf{x}_j, n_j)} y_{kj} = \frac{\phi(q)}{1-q} \sum_{\mathbf{x}_k \in \mathcal{N}(\mathbf{x}_j, n_j)} (1 - d_{kj}) = \frac{n_j \phi(q)}{1-q}(1 - m_j) \leq 1, \tag{4.13}$$

under the stated assumption that $m_j > 1 - \frac{1-q}{\phi(q)n_j}$. This proves that when the scale

parameter ϵ_j is taken as $\epsilon_j = 1/R_j$ then for each j we must have

$$1 - \sum_{\mathbf{x}_k \in \mathcal{N}(\mathbf{x}_j, n_j)} \phi(d_{kj}) = \phi(0) - \sum_{\mathbf{x}_k \in \mathcal{N}(\mathbf{x}_j, n_j)} \phi(\epsilon_j \|\mathbf{x}_k - \mathbf{x}_j\|_2) > 0.$$

That is: the interpolation matrix $A_{\phi_{\epsilon_j}, \mathcal{X}}$ is diagonally dominant and thus non-singular. For the case $\epsilon_j \geq 1/R_j$, this follows because

$$g(\epsilon) := \phi(0) - \sum_{\mathbf{x}_k \in \mathcal{N}(\mathbf{x}_j, n_j)} \phi(\epsilon \|\mathbf{x}_k - \mathbf{x}_j\|_2) \tag{4.14}$$

is a decreasing function on ϵ. □

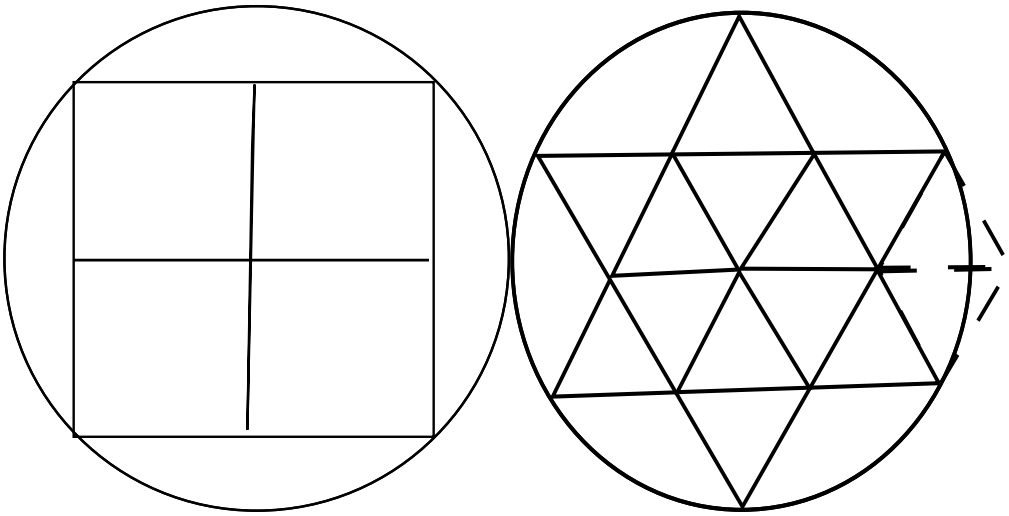
Remark 4.1. If $n_j = 1$, then $q_j = m_j = 1 > 1 - \frac{1-q}{\phi(q) n_j}$, Theorem 4.2 always holds. In this case the interpolation matrix is a diagonal matrix, and of course non-singular. However, this case is not of interest. For $n_j = 2$, then the non-zero off-diagonal elements are always smaller than 1 if $\epsilon_j = 1/R_j$, even if $\mathcal{N}(\mathbf{x}_j, 2)$ does not satisfy the conditions of Theorem 4.2. When $q_j \leq q$, the nearest point is too close to $\mathbf{x}_j$ and the two basis function are more likely to correlate to each other.

Example 4.1. Consider the Wendland function $\phi_{3,1}(r) = (1-r)_+^4(4r+1)$, its convex interval is $[0.25, 1]$. Consider a point $\mathbf{x}_j$ on the equally spaced mesh in $\mathbb{R}^2$; see Figure 4.3(a) for illustration. Let $R_j = 1$ for $\mathcal{N}(\mathbf{x}_j, n_j)$, $n_j = 5, 6, 7, 8$. $q_j = \frac{\sqrt{2}}{2} > 0.25$, $m_j = \frac{\sqrt{2}}{4} + \frac{1}{2} \approx 0.8536$ for $n_j = 8$.

$$1 - \frac{1 - 0.25}{8\phi_{3,1}(0.25)} \approx 0.8519 < m_j.$$

Therefore on the equally spaced mesh in $\mathbb{R}^2$, $\mathcal{N}(\mathbf{x}_j, n_j)$ for $n_j \leq 8$ satisfy Theorem 4.2. This also illustrates that on a equally spaced mesh, the radii for a compact support basis function which satisfies Theorem 4.2 is proportional to the underlying grid size.

Theorem 4.3. *Let ϕ be a compactly supported Wendland radial basis function. For any set of distinct point $\mathcal{X} = \{\mathbf{x}_k \in \mathbb{R}^d, k = 1, 2, \ldots, n.\}$, there exist scale parameters ϵ_j, such that each column of the interpolation matrix $A_{\phi_{\epsilon_j}, \mathcal{X}} = \phi(\epsilon_j \|\mathbf{x}_i - \mathbf{x}_j\|_2)_{1 \leq i,j \leq n}$ is strictly column diagonally dominant and each column has at least 3 non-zero elements.*



(a) On an equally spaced regular mesh (b) On a mesh filled by isosceles triangles

Figure 4.3: Two examples for the supports of basis functions which satisfy a diagonally dominant condition.

Proof. For given $\mathbf{x}_j$, let

$$\rho_1 = \min_{i \neq j} \|\mathbf{x}_i - \mathbf{x}_j\|, \rho_{m+1} = \min_{\|\mathbf{x}_k - \mathbf{x}_j\| > \rho_m} \|\mathbf{x}_k - \mathbf{x}_j\|, \text{ for } m > 1,$$

and

$$N(\rho) := \#\{\mathbf{x}_k : 0 < \|\mathbf{x}_k - \mathbf{x}_j\| < \rho\}.$$

be the number of points whose distance to $\mathbf{x}_j$ smaller than ρ, then for any $\delta > 0$ such that $\rho_1 + \delta < \rho_2$, we have $N(\rho_1) = N(\rho_1 + \delta)$. According to Lemma 4.3, we have

$$\sum_{\mathbf{x}_k \in \mathcal{N}(\mathbf{x}_j, N(\rho))} \phi(\frac{\|\mathbf{x}_k - \mathbf{x}_j\|}{\rho}) \leq cN(\rho)(1 - \frac{\rho_1}{\rho})^m. \tag{4.15}$$

If $N(\rho_1) \geq 2$, let $\rho = \delta + \rho_1 \leq \rho_2$, and force the right hand side in (4.15) less than one, we get

$$\delta \leq \frac{\rho_1 T}{1 - T} = \delta^*, \text{ where } T = (\frac{1}{cN(\rho_1)})^{1/m}.$$

Then for any $0 < \delta < \min\{\delta^*, \rho_2 - \rho_1\}$, the shape parameter $\epsilon_j = \frac{1}{\rho_1 + \delta}$ can make the j-th column have $N(\rho_1) + 1$ non-zero elements and be column diagonally dominant.

For the case $N(\rho_1) = 1$, let $\rho := \rho_2 + \delta < \rho_3$, then

$$\begin{aligned}\sum_{\mathbf{x}_k \in \mathcal{N}(\mathbf{x}_j, N(\rho))} \phi(\frac{\|\mathbf{x}_k - \mathbf{x}_j\|}{\rho}) &\leq \phi(\frac{\rho_1}{\rho_1 + \delta}) + c(N(\rho_2) - 1)\phi(\frac{\rho_2}{\rho_2 + \delta}) \\ &\leq \phi(\frac{\rho_1}{\rho_1 + \delta}) + c(N(\rho_2) - 1)(\frac{\delta}{\rho_2 + \delta})^m \\ &\leq \phi(\frac{\rho_1}{\rho_1 + \delta}) + c(N(\rho_2) - 1)(\frac{\delta}{\rho_2 + \delta}) := f(\delta),\end{aligned}$$

$f(0) = \phi(\frac{\rho_1}{\rho_2}) < 1$ and $f(\delta)$ is a monotonically increasing function. Suppose $f(\delta) \leq 1$ has solutions on $(0, \delta^*)$ for some δ^*. Then for any $0 < \delta \min\{\delta^*, \rho_3 - \rho_2\}$, the choice $\epsilon_j = \frac{1}{\rho_2 + \delta}$ makes the column have $N(\rho_2) + 1$ non-zero elements whilst keeping the diagonal dominance condition . □

Remark 4.2. Theorem 4.3 indicates that a point set $\mathcal{X}$ satisfying the Theorem is a sufficient but not necessary condition to guarantee a non-singular interpolation matrix with RBFs of various shapes.

Example 4.2. Consider a mesh which consists of isosceles triangles as illustrated in Figure 4.3(b). Since $N(\rho_1) = 6$, $\mathcal{N}(\mathbf{x}_j, n_j)$ for $n_j \leq 6$ satisfy a diagonal dominant condition but they result interpolation matrices with only one non-zero element in column j. For $N(\rho_2) = 12$, let $\rho_2 = R_j = 1$, where R_j corresponds to $\mathcal{N}(\mathbf{x}_j, n_j)$ for $n_j = 7$ to 12, then $q_j = \frac{\sqrt{3}}{3} > 0.25$ and $1 - 6 * \phi_{3,1}(q_j) \approx 0.3664 > 0$. Therefore, when using $\phi_{3,1}$ with radius as ρ_2, a diagonal dominant condition can also be guaranteed. The largest n_j such that $\mathcal{N}(\mathbf{x}_j, n_j)$ satisfies Theorem 4.2 is 8.

Remark 4.3. Both Example 1 and Example 2 illustrate the radii of the basis functions which can guarantee diagonal dominant interpolation matrix is a relative value to the mesh size.

Finally, we conclude with a useful practical observation that interpolation matrices constructed from Theorem 4.2 and 4.3 have an nice property which can guarantee (incomplete) LU factorizations. Matrices with such a property are called H-matrices. An H-matrix is a generalization of an M-matrix. A real matrix $A = (a_{i,j})_{1 \leq, i,j \leq n}$ with $a_{i,j} \leq 0$ for all $i \neq j$ is an M-matrix if A is non-singular and A^{-1} is a non-negative matrix. A matrix A is an H-matrix if its comparison matrix $\mathcal{M}(A)$ is an M-matrix, where the comparison matrix $\mathcal{M}(A) = (\alpha_{i,j})_{1 \leq i,j, \leq n}$

Table 4.1: Estimations of the convex interval for Wendland functions

Smoothness	Function	convex interval
C^0	$\phi_{1,0} = (1-r)_+$	$[0, 1]$
C^2	$\phi_{1,1} = (1-r)_+^3(3r+1)$	$[\frac{1}{3}, 1]$
C^4	$\phi_{1,2} = (1-r)_+^5(8r^2+5r+1)$	$[0.2760, 1]$
C^0	$\phi_{3,0} = (1-r)_+^2$	$[0, 1]$
C^2	$\phi_{3,1} = (1-r)_+^4(4r+1)$	$[\frac{1}{4}, 1]$
C^4	$\phi_{3,2} = (1-r)_+^6(35r^2+18r+3)$	$[0.2356, 1]$
C^0	$\phi_{5,0} = (1-r)_+^3$	$[0, 1]$
C^2	$\phi_{5,1} = (1-r)_+^5(5r+1)$	$[\frac{1}{5}, 1]$
C^4	$\phi_{5,2} = (1-r)_+^7(16r^2+7r+1)$	$[0.2056, 1]$

is defined by

$$\alpha_{i,i} := |a_{i,i}|, \text{ and } \alpha_{i,j} := -|a_{i,j}| \text{ for } i \neq j, (1 \leq i, j \leq n).$$

H-matrices can guarantee stable incomplete factorization pre-conditioners [202].

Theorem 4.4. *The interpolation matrices in Theorem 4.2 and 4.3 are H-matrices.*

Proof. Theorem 4.2 and Theorem 4.3 guarantee that the underlying interpolation matrix is strictly diagonally dominant. The result follows due to a established result: if a matrix is either a strictly diagonally dominant or an irreducibly diagonally dominant matrix, then A is an H-matrix [201, p.92, Theorem 3.27]. □

4.4 Convex interval and eigenvalue bound

4.4.1 Estimation of the lower bound of the convex interval

A more accurate estimation on the threshold q for the convex interval $[q, 1]$ can supply a tighter lower bound on m_j. For the Wendland function $\phi_{d,0}$, we can take $q = 0$; for $\phi_{d,1}$, $q = \frac{1}{\lfloor d/2 \rfloor + 3}$. In both cases q is minimal. It is easy to find out that the lower bound of the convex interval $\phi_{d,1}$ becomes smaller as d increases. This is also true for $\phi_{d,2}$ when $d = 1, 3, 5$. See Figure 4.4. Though neither Theorem 4.2 nor Lemma 4.2 indicates that there is only one root for $\phi''_{d,k}(r) = 0$ for $k \geq 2$, this is

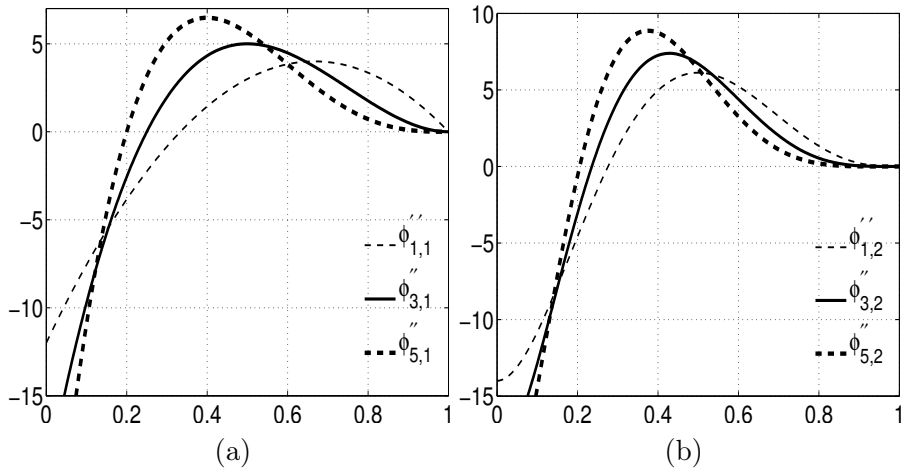


Figure 4.4: The second derivatives of Wendland function $\phi_{d,k}(r)$ for $d = 1, 3, 5$ and $k = 1, 2$. The root of $\phi''_{d,k}(r)$ in (0,1) is the exactly tight bound of q.

true for the case $k = 2$ and $d = 1, 3, 5$. It is possible that $\phi''_{d,k}$ at most has only one zero in $(0, 1)$. The estimations of the threshold q for several Wendland function in Table 4.1 are computed by Mathematica.

4.4.2 Sharper bound for m_j and eigenvalue distribution

From Figure 4.1(b), as one might predict, the lower bound on m_j in Theorem 4.2 can be far from tight in most cases. For a tighter bound on m_j, one can use $\frac{1-q_j}{n_j q_j}$ in (4.13). One can even verify the diagonal dominance by "brute force"—evaluate $\phi(\epsilon_j \|\mathbf{x}_k - \mathbf{x}_j\|)$ directly, but this involves more computations, every time changing the support or the size of the neighbourhood means that one has to re-evaluate the function. However, even if only some of the centres are checked according to Theorem 4.2, it can save a lot of computations when adaptively choosing the support.

In either case, one can guarantee that the smallest eigenvalue of the interpolation matrix is bounded significantly away from the origin. The diagonal dominance indicates that $\mu = \|A_{\phi_{\epsilon_j},\mathcal{X}}\|_1 < 2$. According to the classical Gerschgorin disc theorem [201, p.16], all the eigenvalues of the interpolation matrix are located in a disc with center $(1, 0)$ and radius $\mu - 1$. The upper bound on the ratio of the largest eigenvalue to the smallest length of the eigenvalue is $\frac{\mu}{2-\mu}$.

4.5 Discussion

The negative results [214] according to the Strang-Fix condition are seldom mentioned in popular publications. Wendland uses local polynomial reproduction properties explicitly in monograph [208, Chap 3], but does not mention the Strang-Fix condition. Chapter 5 of [208] discusses the moving least square approach, in fact such a method satisfies the Strang-Fix condition [133, 215]. Buhmann [32] uses the cardinal function approach to derive the local polynomial reproduction properties, where he discussed that the cardinal function approach is consistent with the Strang-Fix condition for conditional positive/negative functions, where he calls such functions *admissible of order* m. He proved that it is impossible for *pseudoadmissible* basis functions to recover constants [29, Thm 23], and pointed out that the Gaussian belongs to the pseudoadmissible functions. For non-stationary Gaussian radial basis function approximation, which conflicts with the Strang-Fix condition, it converges to some *saturation error*. It is unknown whether similar situations happens for other positive definite kernels. Fewer comparisons on the approximation quality between radial basis functions and other methods are reported in the literature. In [29], Buhmann did not mention positive definite CSRBFs because positive definite CSRBFs were constructed later. It is easy to prove that positive definite CSRBFs are pseudoadmissible. In his book [32], he did not mention that CSRBFs do not satisfy the Strang-Fix condition, but pointed out that estimation of the *saturation error* for some functions is an open problem.

That stationary interpolation with positive definite CSRBFs does not converge has been observed for a long time. It was believed that such a behavior had not been fully understood [70, p.99]. Here, we have easily proved that stationary interpolation with positive definite CSRBFs does not converge according to the Strang-Fix condition.

According to the Strang-Fix condition, the multilevel approach can converge, if there is a function $\Psi \in \text{span}\{\phi_{\delta_1}, \phi_{\delta_2}, \ldots, \phi_{\delta_m}\}$ which satisfy the Strang-Fix condition. If there is, it is not clear how to adjust such a scale δ_j such that only a limited m is enough. It seems very hard to apply to the multilevel approach to truly scattered data with many levels. The method presented here uses CSRBFs of different shapes which increases the variability of the scales in the same level, we

shall further integrate such a scheme into a multilevel approach in Chapter 7.

Interpolation with radial basis functions with different shapes are motivated by applications. For complicated real applications, it is helpful to use an adaptive refinement scheme, arranging more data where necessary to reduce the total computational complexity. In this case, one also wants to use radial basis functions with different scales for locally clustered points; if two basis function centres are too close to each other, the corresponding two columns of the system matrix A are nearly linearly dependent, and thus the matrix can be ill-conditioned. Whereas if such two basis functions employ different scales or are different, then the conditioning may be improved [88]. The idea of using radial basis functions with different scales dates back to the 1970s [174], whereas few practical results for large data sets were obtained. There are papers which discuss how to choose various shape parameters both numerically [22, 57, 81, 117, 121, 175, 219] and theoretically [20, 22]. In particular in [20], Bozzini et al. use matrix perturbation arguments, concluding that if the scales do not vary a lot across the domain, then interpolation with various scales can result in a non-singular interpolation matrix. The threshold of variation of the scales depends on the estimation of the smallest eigenvalue of an interpolation matrix with the same scale [20, Theorem 2]; such a dependence limits the utility of the method. Recently these results were improved [21]. Solvability for other interesting cases remains to be established for general radial basis functions.

A further open question arises: since positive definite compactly supported kernels do not satisfy the Strang-Fix condition, and since there are constructive ways to guarantee the diagonal dominance condition, are there non positive definite compactly supported functions which have a better local polynomial reproducing property? For example, compactly supported radial basis function with degrees less than that of Wendland function?

Part III

Algorithms

5. Assembly of interpolation matrices

The aim of this chapter is to design and implement efficient algorithms to check the diagonal dominance conditions discussed in Chapter 4, and assemble the underlying interpolation matrix with radial basis functions of different shapes.

5.1 Check Theorem 4.2

We assume that the k nearest neighbouring points for each center have been sorted out, and these points are arranged in the ascending order according to corresponding distance values. This can be achieved by a k nearest neighbour search procedure, for example, the `knnsearch` function in Matlab. The nearest neighbour search algorithm usually employs the kd-tree structure which can be constructed in $\mathcal{O}(dN\log(N))$ time [208, p.240], and finding the k nearest neighbouring points can be done in $\mathcal{O}((\log N+k)N)$ time [208, p.248] based on the kd-tree or bd-tree search. A faster approximate nearest neighbourhood search algorithm is also available [144].

Once the initial k nearest neighbouring points are obtained for each center, we only need to check the following two conditions:

$$\text{Condition A}: \qquad q_j > q, \tag{5.1}$$

$$\text{Condition B}: \qquad m_j > 1 - \frac{1-q}{\phi(q)n_j} := \text{RHS}, \tag{5.2}$$

where q_j, m_j, q, n_j are defined in Theorem 4.2. Suppose that the queried nearest points including the center itself are stored in an array `dst`. Then a Matlab code in Program 5.1 illustrates one possible way to find the largest number n_j (`mnj` in the code) which satisfies Theorem 4.2.

Techniques like using `cumsum` and `./` increase the efficiency of the Matlab code by vectorizing. This code works very efficiently up to a certain scale (depending on hardware), but is not suitable for large scale point clouds because it is not memory efficient and there are redundant computations. Memory efficiency plays an important role in designing scalable algorithms for large scale point clouds. A memory efficient procedure is described in Algorithm 5.2. The idea of the procedure

Program 5.1 A Matlab program to check Theorem 4.2

```
function [dsto,mnj,G]=checkt(dst,q,rbf)
%% A Matlab function to find a neighbourhood satisfying theorem
%IN:  dst:distance of k nearest neighbouring points in ^ order
%      rbf: a function handle
%        q: the lower bound of the convex interval of rbf
%OUT: dsto: elements of the interpolation matrix within current
    center
%      mnj: the largest number of nj satisfying theorem
%        G: G=[r_j, R_j, q_j, mj, lower bounds of eigenvalue]
k=length(dst)-1;                % the number of neighbours
dst2=dst(2:end);                % remove the center itself;
cs=cumsum(dst2);                % cumsum
n1j=[1:k]';                     % column vector of nj
mj=cs./(n1j.*dst2);             % mj as a vector of k
qj=dst2(1)./dst2;               % qj as a vector of k
RHS=1-(1-qj)./(n1j.*rbf(qj));
mask=find(mj>RHS &qj>q);        % satisfy mj> RHS and qj>q
if ~isempty(mask)
      mnj=max(mask);            % find the maximum nj
else
      mnj=2;                    % otherwise mnj=2
end
dsto=[1; rbf(dst2(1:mnj)/dst2(mnj))];
G=[dst2(1), dst2(idx),qj(mnj), mj(mnj), 1-sum(dsto(2:end))];
```

Algorithm 5.2 An algorithm to check Theorem 4.2

1: Start with $n_j = 0$, $m_j = q_j = 1$, RHS=0.
2: **while** $m_j >$ RHS and $n_j < k$. **do**
3: Increase n_j, increase R_j.
4: Update q_j.
5: **if** $q_j > q$ **then**
6: Update m_j and RHS.
7: **else if** $n_j == 1$ **then**
8: $R_j = (\rho_1 + \rho_2)/2$.
9: **end if**
10: **end while**
11: Set n_j and R_j.

is: start with the nearest neighbouring point and then increase the number of neighbouring points until either condition fails. If one condition fails when there

are only 2 nearest neighbouring points, the radius is forced to be $R_j = (\rho_2 + \rho_3)/2$, where ρ_2 and ρ_3 are defined as in Theorem 4.3, i.e.

$$\rho_1 = \min_{i \neq j} \|\mathbf{x}_1 - \mathbf{x}_j\| \quad \text{and} \quad \rho_{m+1} = \min_{\|\mathbf{x}_i - \mathbf{x}_j\| > \rho_m} \|\mathbf{x}_i - \mathbf{x}_j\|, \text{ for } m = 1, 2, \ldots. \quad (5.3)$$

Program 5.5 at the end of this chapter is one possible C/C++ implementation for Algorithm 5.2. Compared with Program 5.1, it only needs 4 numbers: `cs,qj,mj` and `rhs`.

5.2 Check Theorem 4.3

Algorithm 5.2 can be improved so that a larger n_j can be selected, and the condition in Theorem 4.3 that each column of the interpolation matrix contains at least 3 non-zero elements is guaranteed. For this, the sharper bound on m_j (discussed in § 4.4.2)

$$\text{Condition C}: \qquad m_j > 1 - \frac{1 - q_j}{\phi(q_j) n_j}. \quad (5.4)$$

is used after the condition B in (5.2) fails. At the beginning, a bold step to increase the neighbours according to condition B in (5.2) can be used. As soon as condition B fails, the condition C will be employed. The procedure is described in Algorithm 5.3, and a C/C++ implementation of the algorithm is described in Program 5.6 at the end of this chapter.

5.3 Complexity and numerical verification

Once the support of each center has been determined, it is easy to generate the interpolation matrix. An algorithm to assemble the interpolation matrix is illustrated in the Matlab code in Program 5.4, which is based on Program 5.1. Now we derive the computational complexity of Program 5.4. In Program 5.1, the `cumsum` takes k operations, calculating `mj` and `qj` takes $3k$ and k operations respectively, calculating `RHS` takes $4k$ operations plus k evaluations; the `find` step only need $2k$ comparisons and k logical operations. Suppose k is a number less than 100, then the total operation count is at most $\mathcal{O}(k^2)$. If k neighbouring points are supplied for each point, then the total operations to calculate entries for the interpolation

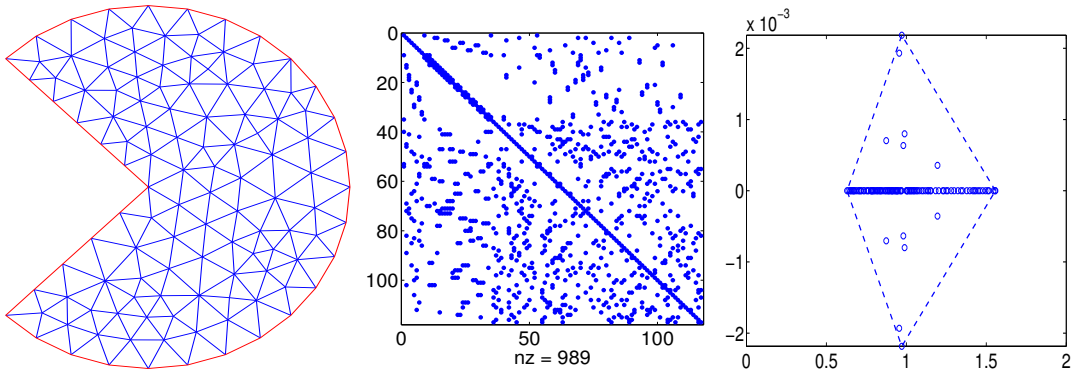
Algorithm 5.3 Adaptive algorithm to check diagonal dominance conditions

Start with $n_j = 0, q_j = 1, m_j = 1$, RHS $= 0$
while $q_j > q$ and m_j satisfies the condition B in (5.2)and $n_j < k - 3$ **do**
    $n_j \leftarrow n_j + 2$
    Update m_j and q_j.
end while
if $(nj = 2)$ **then**
    Set $n_j = 3$ and adjust R_j to satisfy diagonal dominance condition
else
    Step back n_j and update m_j, q_j, and RHS according to Condition C in (5.4).
    while n_j satisfies the condition C in (5.4). **do**
        $n_j \leftarrow n_j + 1$ // use small step
        Update m_j, q_j and RHS according to condition C in (5.4).
    end while
    Set n_j and R_j.
end if

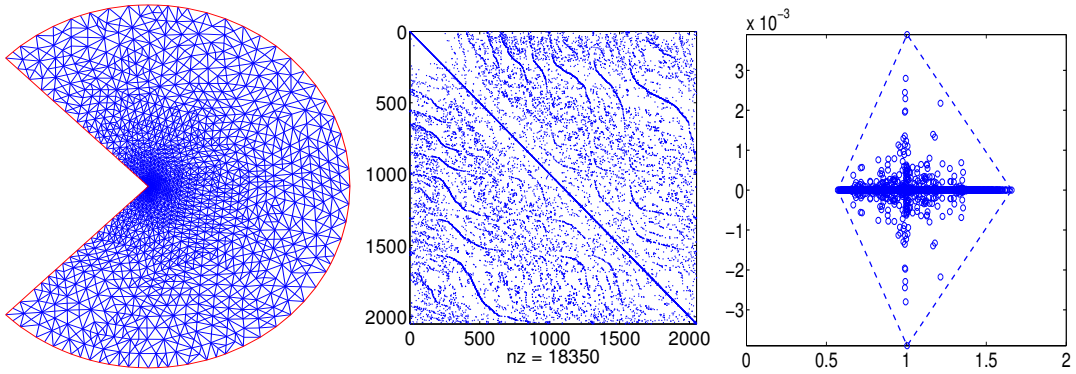
Program 5.4 A Matlab function to generate the interpolation matrix

```
function [A,R,mnj]=mkA(dsites,rbf,npts,q)
% IN     :    dsites: the interpolation points
%             rbf: the radial basis functions
%             npts: the initial number of neighbouring points
%             q: the lower bound of the convex interval of rbf
% OUT    :    A:  the interpolation matrix
%             R:  The radials of each basis functions
%             nj: the numbers of neighbouring points of the j-th
    basis
%                    function
% Requires package: statistics toolbox:knnsearch
NN=length(dsites);
[idx,dist]=knnsearch(dsites,dsites,'k',npts,'NSMethod','kdtree')
    ;
idx=idx';dist=dist';
 A=spalloc(NN,NN,NN*npts);s=zeros(NN,5);mnj=uint32(ones(NN,1));
 for k=1:NN
       [dst,mnj(k),G]=checkt(dist(:,k),q,rbf);
       R(k)=G(2);A(idx(1:mnj(k)+1,k),k)=dst;
 end

```



(a) T1 data set (b) T1 interpolation matrix (c) T1 eigenvalues



(d) T2 data set (e) T2 interpolation matrix (f) T2 eigenvalues

Figure 5.1: Data sets, interpolation matrices, and the eigenvalue distributions for the `TriDisc` problems. The blue circles in (c) and (f) are eigenvalues, and the blue dashed lines are the boundaries of the convex hull of all the eigenvalues.

matrix is at most $\mathcal{O}(Nk^2)$. Furthermore, for each point, this procedure can run independently and thus is convenient for parallel computing (by replacing the serial `for` loop in Algorithm 5.4).

Theorem 5.1. *Suppose the initial k nearest neighbouring points to each center are given, it takes most $\mathcal{O}(N)$ operations to assemble an interpolation matrix according to Algorithm 5.4 and Algorithm 5.3.*

Program 5.4 can also return the information n_j and R_j for each center and the corresponding lower bound on the smallest eigenvalue.

Ten scattered data sets are used to verify the presented algorithms. Our first 2 scattered data sets in Figure 5.1(a) and Figure 5.1(d) are generated by the following Matlab code

```
[~,p,e,t]=adaptmesh('cirsg','cirsb',1,0,0,'maxt',k,...
        'tripick','pdeadworst','ngen',inf);TriDisc1=p';
```

We refer to the problems as `TriDisc1` (T1), when `k` = 100 and `TriDisc2` (T2), when `k` = 3000. The other 8 problems come from our applications in Section 9.1. One for the 2D parametric heart curve (HC) reconstruction problem; four for the Stanford bunny model (B4, B3, B2, B1), and three for the Stanford dragon model (D3, D2, D1). In all cases, the Wendland function $\phi_{3,1}(r) = (1-r)_+^4(4r+1)$ is employed.

Figure 5.1(b) and Figure 5.1(e) illustrate the structure of the interpolation matrices generated by Program 5.4. Figure 5.1(c) and Figure 5.1(f) illustrate the eigenvalue distribution of the corresponding interpolation matrices.

A collection of results on n_j and R_j that satisfy Theorem 4.2 according to Program 5.4 are listed in Table 5.1. Similar results and timing based on Algorithm 5.3 are put in Table 5.2. We observe that the average number of neighbouring points of each center n_j tends to be a number around 10 for the tested data.

Table 5.3 reports the timing results for the C/C++ implementation of Algorithm 5.5, and compares these results with the time for the data querying part by `knnsearch` with the kd-tree method. The time for the checking procedure can almost be neglected. The last column in Table 5.3 is the time for the `sparse` function in Matlab to construct a sparse matrix, when all the vectors of indexes and the vector of elements are available. Such a procedure is not necessary if one only uses the C/C++ environment.

5.4 Discussion

We have demonstrated that using CSRBFs of different shapes as described takes little time when compared with the pre-processing part, the data querying part based on fast nearest neighbourhood search. Fast nearest neighbourhood search plays an important role in scattered data approximation and other applications based on a point cloud.

According to Table 5.1 and Table 5.2, the methods described in Chapter 4 result in well conditioned interpolation matrices. The results also show that interpolation with radial basis functions of different shapes in such a way requires limited memory. We shall examine the approximation quality later, and now we consider how to solve

these linear systems efficiently.

Table 5.1: Information of test problems from Program 5.4. The $\lambda_{\texttt{min}}$ and $\lambda_{\texttt{max}}$ are calculated by `eigs` in Matlab. μ is the 1-norm of the interpolation matrix as discussed in Section 4.4.2.

Prob	A		Eig Information				n_j			Scale Range (R_j)			
	size	nnz	$\frac{\lambda_{\max}}{\lvert\lambda\rvert_{\min}}$	$\lvert\lambda\rvert_{\min}$	$\frac{\mu}{2-\mu}$	$2-\mu$	min	max	mean	min	max	mean	max/min
HC	252	1704	2.23	0.63	4.0	0.45	6	8	6.7	0.1095	0.3568	0.2275	3.3
T1	117	989	2.47	0.63	7.6	0.23	6	14	8.5	0.2163	0.4	0.2824	1.8
T2	2050	18350	2.90	0.57	10.8	0.17	4	14	9.0	3.8e-4	0.1515	5.0e-2	391.9
B4	1351	9911	3.87	0.42	10.7	0.17	3	16	7.3	3.5e-3	1.6e-2	1.0e-2	4.6
B3	5643	42825	3.50	0.45	8.8	0.20	3	17	7.6	1.6e-3	8.7e-3	5.3e-3	5.6
B2	24425	183988	3.95	0.41	12.0	0.15	3	21	7.5	1.3e-4	4.2e-3	2.6e-3	32.2
B1	105615	740493	4.00	0.41	18.9	0.10	3	21	7.0	5.8e-6	2.9e-3	1.2e-3	510.3
D3	15563	117362	3.83	0.43	10.1	0.18	3	24	7.5	2.6e-4	5.8e-3	3.51-3	22.0
D2	68830	512376	4.02	0.41	14.1	0.13	3	19	7.4	1.5e-4	3.4e-3	1.7e-3	22.2
D1	300298	2136353	12.20	0.15	13.3	0.14	2	24	7.1	2.1e-5	2.3e-3	7.9e-4	109.8

Table 5.2: Information of test problems from Program 5.6. The $\lambda_{\texttt{min}}$ and $\lambda_{\texttt{max}}$ are calculated by `eigs` in Matlab. μ is the 1-norm of the interpolation matrix as discussed in Section 4.4.2.

Prob	A		Eig Information				n_j			Scale Range (R_j)			
	size	nnz	$\frac{\lambda_{\max}}{\lvert\lambda\rvert_{\min}}$	$\lvert\lambda\rvert_{\min}$	$\frac{\mu}{2-\mu}$	$2-\mu$	min	max	mean	min	max	mean	max/min
HT	252	1704	2.23	0.63	3.49	0.45	6	8	6.8	1.1e-01	3.6e-01	2.3e-01	3.3
T1	117	989	2.48	0.63	7.60	0.23	6	14	8.5	2.2e-01	4.0e-01	2.8e-01	1.9
T2	2050	18350	2.90	0.57	10.8	0.17	4	14	9.0	3.9e-04	1.5e-01	5.0e-02	391.9
B4	1351	14603	15.65	0.12	19.6	0.10	3	22	10.8	3.0e-03	2.1e-02	1.3e-02	7.0
B3	5643	63678	6.95	0.25	7.6	0.23	3	24	11.3	1.5e-03	1.1e-02	6.8e-03	7.7
B2	24425	275281	6.00	0.17	9.8	0.19	3	24	11.3	2.0e-04	5.3e-03	3.4e-03	26.3
B1	105615	1127137	4.24	0.40	19.8	0.10	3	27	10.7	5.8e-06	3.6e-03	1.6e-03	616.5
D3	15563	173323	28.22	0.07	53.4	0.04	3	27	11.1	4.0e-04	7.3e-03	4.4e-03	18.4
D2	68830	763863	36.34	0.05	197.4	0.01	3	27	11.1	1.5e-04	3.7e-03	2.2e-03	23.9
D1	300298	3167659	41.19	0.04	64.4	0.03	3	27	10.5	2.4e-05	2.8e-03	1.0e-03	116.1

Table 5.3: Timing results for the procedure of checking diagonal dominance condition. Time in seconds, C/C++ implementation for Algorithm 5.5

Prob	N	knn	check	sparse
B4	1351	0.0130	0.0002	0.0015
B3	5643	0.0543	0.0009	0.0068
B2	24425	0.2576	0.0040	0.0342
B1	105615	1.2357	0.0180	0.1546
D3	15563	0.1641	0.0027	0.0213
D2	68830	0.7776	0.0119	0.1023
D1	300298	3.5881	0.0521	0.5041

Program 5.5 A C program for Algorithm 5.2

```
 double rhsrbf(double r, int n1j)
// a function for the right handside $\delta$
{ return 1-(1-r)/(n1j*pow((1-r),4)*(4*r+1));}

void check1col(double *first,int k, int *nj,double *R)
{/*  input: *first, adrress of the first nearst neighbouring
    point
                q:    the lower bound of the convex interval
                k:    the N(x_j, k) nearst neighbouring points
                nj:   address to store nj
                R:    address to store R
      output: nj, R with selected values    */

    int nj1=0;  // starts with nj=0
    double cs=*first,qj=1.0,mj=1.0,rhs=0.0; // cs is current
    cusum
    while(mj>rhs && n1j<k)  // check diagonal dominance
    condition
    {   qj =first[0] /first[++(nj1)]; // increase nj and comput
    qj
        if(qj >=q)   // qj locates in the convex interval
        {   rhs= qj==1?0:rhsrbf(qj,n1j); // compute rhs,
            cs += first[n1j];            // accumulate cs
            mj =cs/(n1j*first[nj1]);     // calculate mj
        }
        else  //  if qj<=q, check diagonal dominance by brute
    force
        {   if(n1j==1)
            {   while(first[n1j]==first[n1j+1]) ++n1j;
                *R=.5*(first[n1j]+first[n1j+1]);   //
            }
            break;
        }
    }
    if(n1j!=1)  {   *nj=--n1j;   *R=first[n1j]; }
    else   {   *R=first[n1j]; *nj=n1j;}
}
```

Program 5.6 A C program for Algorithm 5.3

```
double rhsrbf(double r,int n1j)
{ return 1-(1-r)/(n1j*pow((1-r),4)*(4*r+1));}
void check1col2(double *first, int k, int *nj, double *R)
{   /*  input and out put the same as the algorithm check1col.*/
    int n1j=0; double cs=0.0,qj=1.0,mj=1.0,rhs=0.0;
    while(mj>rhs && qj >=q && n1j<k-3) //  Use coarse conditon
    {   //after while loop n1j =2 -> 4 ->6 ..... -> 14
        cs += first[n1j++];
        cs += first[n1j++];
        mj  = cs/(n1j*first[n1j]);
        qj  = first[0]/first[n1j];
        rhs = 1.0-q31/n1j;
    }
    if(n1j==2)
    {  // only 3 nearst points are slected, sepcial case deal
        if ( wd31(first[0]/first[2])+wd31(first[1]/first[2]) <1)
       {*nj=3;*R=first[2];}
        else  //first[2]=0.5*(first[1]+first[2]);
        {   *R=0.5*(first[1]+first[2]);
            while (wd31(first[0]/ *R)+wd31(first[1] / *R) >=1)
            {  *R = 0.5*(first[1] + *R);}
            *nj=3;
        }
    }
    else
    {  // n1j-2 satisfy Use sharper conditon (C)
        n1j--;
        cs -= first[n1j];
        mj  = cs/(n1j*first[n1j]);
        qj  = first[0]/first[n1j];
        rhs  =rhsrbf(qj,n1j);
        while (mj >rhs && qj>=q && n1j<k-2)
        {   cs += first[n1j++];
            mj  = cs/(n1j*first[n1j]);
            qj  = first[0]/first[n1j];
            rhs = rhsrbf(qj,n1j);
        }
        *nj=n1j; *R=first[n1j-1];
    }
}
```

6. Fast solvers

This chapter examines how to solve the resulting linear system by exploring features of the underlying problems. Linear systems arising from radial basis function interpolation can be divided into two categories: dense linear systems and sparse ones. We focus on discussing methods for the sparse linear systems, in particular, for linear systems obtained as in Chapter 5. Besides, we also discuss methods for dense radial basis function interpolation systems. Firstly, we examine the performance of sparse direct methods, followed by a discussion on sparse iterative methods, and then we try general preconditioning techniques and customize preconditioning techniques for the underlying linear systems. As the discussion proceeds, better solvers for the sparse linear systems will be selected and recommended. Finally we discuss some techniques for dense linear systems related to radial basis function interpolation. We shall see how to solve a 3D object modelling problem in 1 second with a preconditioned Krylov method and customized preconditioners. Finally, we recommend the BICGSTAB and conjugate gradient squared (CGS) methods with a preconditioning technique developed here for the linear systems resulted in Chapter 4.

6.1 Sparse direct methods

Many people would like to choose a direct method for a small or moderate size linear system. Modern sparse direct methods can solve large scale problems with dedicated sparse techniques. Compared with iterative methods, they are more robust but usually suffer from poor scalability. Sparse direct methods usually consist of three steps: analyse, factorize and solve. The analysis part plays an important role in the overall solution strategies for large sparse linear systems, see [50, 63]. We emphasize the analysis part so that this critical technique has been considered and included in implementations when comparing the performance between sparse direct and sparse iterative methods.

The analysis part usually consists of ordering and symbolic factorization. To increase numerical stability and to reduce *fill-in*, ordering is necessary. Numeri-

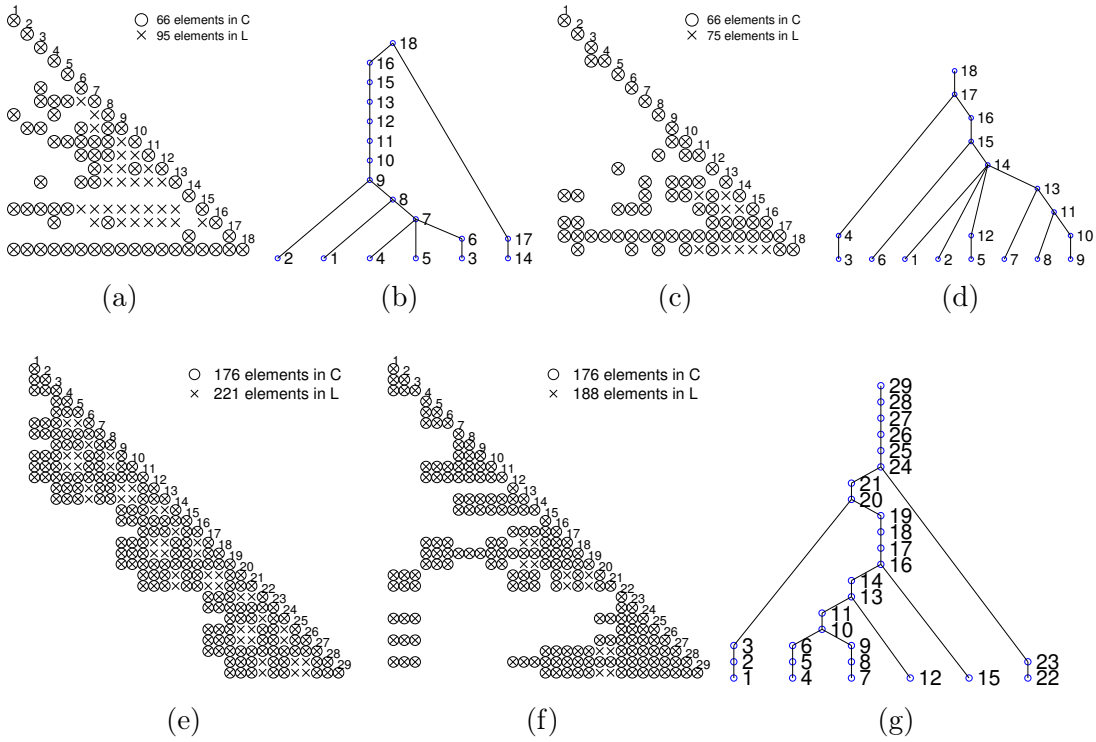


Figure 6.1: The fill-reducing ordering and its effects. Subfigure (a) demonstrates the sparse pattern of a matrix C and its LL^T factors L. Subfigure (b) shows the elimination tree for the LL^T factorization of the matrix in (a); the factorization starts with leaf nodes (column 1, 2, 3, 4, 5 and 14 in the matrix in (a)) and finishes at the root node (column 18 in the matrix in (a)). The factorization of a parent column depends on the data in its child columns. The height of the tree shows the sequential steps in the factorization. The width of the tree shows the maximum parallelism. Subfigure (c) illustrates the approximate minimum degree ordering of the matrix in Subfigure (a) and the LL^T factor corresponding to the amd ordering. After the `amd` ordering, there are only 9 fill-ins while there are 29 fill-ins in (a). Subfigure (d) demonstrates the elimination tree for the `amd` ordering, the tree is shorter and wider than that in (b). Subfigure (e) and (f) compare a Wathen matrix with its approximate minimum degree ordering. The matrix is generated by the Matlab function `C=gallery('wathen',2,3)`. (g) illustrates the elimination tree for the matrix in (f). The elimination tree for the matrix in (e) is a vertical line and thus is omitted here. Reordering can also shorten and broaden an elimination tree to reduce the sequential steps and increase parallelism.

cal stability can be improved by choosing proper *pivots*. Reducing fill-in plays an essential role in reducing storage and improving performance. The importance of fill-reducing ordering is illustrated in Figure 6.2. The fill-reducing ordering algorithm adopted here is the approximate minimum degree ordering (amd) [2, 3]; in Matlab this corresponds to the `amd` function.

Table 6.1: Timing results (in seconds) for sparse factorization with `amd` ordering.

Prob	N	`amd`	`lu`	all	`nnz(A)`	`nnz(L+U)`	$\frac{\texttt{nnz(L+U)}}{\texttt{nnz(A)}}$
B4	1351	0.0020	0.0335	0.0356	14603	78546	5.38
B3	5643	0.0085	0.1381	0.1466	63678	553534	8.69
B2	24425	0.0479	1.2763	1.3242	275281	3142223	11.41
B1	105615	0.3439	20.6308	20.9746	1127137	17003902	15.09
D3	15560	0.0274	0.4163	0.4437	173323	1572824	9.07
D2	68830	0.1969	3.5476	3.7445	763863	8567189	11.22
D1	300298	1.2526	42.5484	43.8010	3167659	30131788	9.51

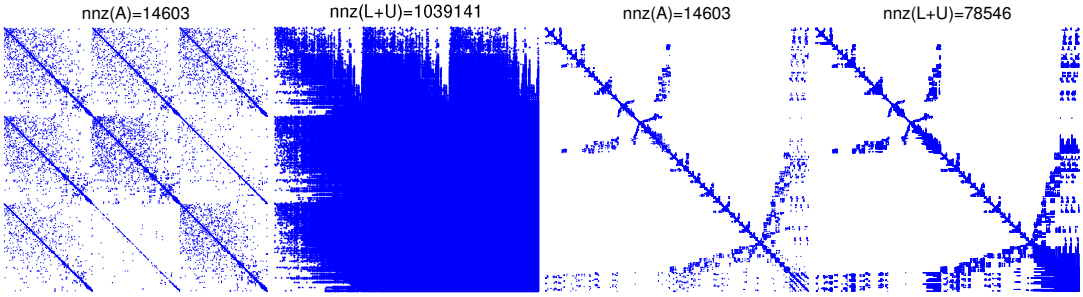


Figure 6.2: The ordering effect for sparse factorization methods. (a) an interpolation matrix for the Stanford bunny problem. (b) the non-zero pattern of the LU factors of matrix in (a). (c) the approximate minimum degree ordering of the matrix in (a). (d) the non-zero pattern of the LU factors of matrix in (c). To get the LU factorization of matrix in (a), it takes more than 1 second, while to get the LU factorization of matrix in (c), it only takes 0.003 seconds.

Together with reordering, a *symbolic analysis and factorization* can predict the structure of the *LU* factors before numerical factorization starts. The structure can be used to allocate proper working space in advance (statically) for the *LU* factors rather than dynamically. Such static and dynamic memory allocation can result in significantly different performance of implementations of the same algorithm. The symbolic factorization works on sparse patterns rather than numerical values, and thus it saves memory and can proceed without floating point arithmetic. The non-zero pattern of the *LU* factors can be analysed by the *elimination tree* [50, p.56][137], which is also used to analyse the data flow dependence; see illustration for symmetric cases in Figure 6.1.

Table 6.1 reports the timing results for sparse factorization methods with approximate minimum degree ordering (`amd`). Also reported are the number of non-

zero elements in A, the number of non-zero elements in $L + U$ and the ratio. It should be emphasized that without the fill-in reducing ordering, most of the problems in Table 6.1 take an extremely long time to get the LU factors or even fail due to insufficient memory.

For the largest problem, D1, it takes the sparse direct methods with dedicated ordering more than 40 seconds to solve the linear system. Our goal in this chapter is to reduce this time as much as possible, but being able to also solve systems of even larger dimension is our main reason for considering iterative methods.

6.2 Krylov subspace iterative methods

We first describe a general framework to derive Krylov subspace methods. Then we briefly discuss the derivation of two kinds of Krylov subspace methods for general non-symmetric matrices. The first kind of method is based on long recurrence orthogonalization, which can minimize the residual in a projected space in every iteration. The other kind of methods are based on biorthogonalization and are more memory efficient but do not have the minimal residual property. Finally we compare the performance of these methods when applied to our problems and select a solver.

6.2.1 General framework

A Krylov subspace of a matrix A with respect to a vector $\underline{\mathrm{v}}$ is a subspace of the form

$$\mathcal{K}_m(A, \underline{\mathrm{v}}) := \mathrm{span}\{\underline{\mathrm{v}}, A\underline{\mathrm{v}}, A\underline{\mathrm{v}}^2, \ldots, A^{m-1}\underline{\mathrm{v}}\}$$

which will be denoted simply by $\mathcal{K}_m$ if there is no confusion. A Krylov method approximates the solution to $A\underline{x} = \underline{\mathrm{f}}$ iteratively in a sequential affine subspace $\underline{x}_0 + \mathcal{K}_m$. The approximate solution $\underline{x}_m$ is determined according to the following constraint condition

$$\underline{\mathrm{f}} - A\underline{x}_m \perp \mathcal{L}_m,$$

where $\mathcal{L}_m$ is a subspace of dimension m. Different choices of the constraint space $\mathcal{L}_m$ result in different kinds of Krylov methods. Commonly used constraint spaces are $\mathcal{L}_m = \mathcal{K}_m$, $\mathcal{L}_m = A\mathcal{K}_m$ and $\mathcal{L}_m = \mathcal{K}_m(A^T, v)$.

If we write $\underline{x}_m = \underline{x}_0 + \delta\underline{x}$ and define $\underline{\mathrm{r}}_0 = f - A\underline{x}_0$, then $\underline{\mathrm{r}}_m = \underline{\mathrm{r}}_0 - A\delta\underline{x}$ and the

constraint condition is equivalent to $\underline{\mathrm{r}}_m \perp \mathcal{L}_m$. The approximate solution, $\underline{x}_m$, to the linear system can be updated if $\delta\underline{x}$ is determined; $\delta\underline{x}$ is determined by

$$\text{finding } \delta\underline{x} \in \mathcal{K}_m \text{ such that } (\underline{\mathrm{r}}_m, \underline{w}) = 0, \forall\, \underline{w} \in \mathcal{L}_m. \tag{6.1}$$

When A is symmetric positive definite and $\mathcal{L}_m = \mathcal{K}_m$, the solution to problem (6.1) satisfies [171, p.126]

$$\underline{x}_m = \arg \min_{\underline{x} \in \underline{x}_0 + \mathcal{K}_m} \|\underline{x}^* - \underline{x}\|_A, \tag{6.2}$$

where $A\underline{x}^* = \underline{\mathrm{f}}$ and $\|\underline{x}\|_A = \underline{x}^T A \underline{x}$. Such a choice of the $\mathcal{L}_m$ results in the conjugate gradient method for symmetric positive definite matrices. Here we focus on methods for general nonsymmetric matrices like the general minimal residual method.

6.2.2 GMRES

When $\mathcal{L}_m = A\mathcal{K}_m$ for an arbitrary square matrix, the solution to problem (6.1) satisfies [171, p.127]

$$\underline{x}_m = \arg \min_{\underline{x} \in \underline{x}_0 + \mathcal{K}_m} \|b - A\underline{x}\|_2. \tag{6.3}$$

Such a choice of the constraint space results in minimal residual methods. Here we focus on the generalized minimal residual method (GMRES) [172] for general non-symmetric matrices. The following lines derive the GMRES algorithm.

Since $\delta\underline{x} \in \mathcal{K}_m$, the approximate solution $\underline{x}_m$ can be written as $\underline{x}_m = \underline{x}_0 + \sum_{j=1}^{m} y_j \underline{\mathrm{v}}_j = \underline{x}_0 + V_m \underline{y}_m$, where $\underline{y}_m \in \mathbb{R}^m$, $V_m = [\underline{\mathrm{v}}_1, \underline{\mathrm{v}}_2, \ldots, \underline{\mathrm{v}}_m] \in \mathbb{R}^{n \times m}$, and $\{\underline{\mathrm{v}}_1 \ldots, \underline{\mathrm{v}}_m\}$ is a basis of the space $\mathcal{K}_m$. Ideally, $\{\underline{\mathrm{v}}_1, \ldots, \underline{\mathrm{v}}_m\}$ are orthogonal for numerical stability. V_m can be determined by the (modified) Gram-Schmidt algorithm (line 3 to line 15 in Algorithm 6.1). The Gram-Schmidt algorithm generates a Hessenberg matrix $\bar{H}_m$ of order $(m+1) \times m$. Such a process is often referred to as the Arnoldi procedure. The elements of $\bar{H}_m$ satisfy

$$A\underline{\mathrm{v}}_j = \sum_{i=1}^{j+1} h_{i,j} \underline{\mathrm{v}}_i, j = 1, 2, \ldots m. \tag{6.4}$$

The corresponding matrix equation form is

$$AV_m = V_{m+1}\bar{H}_m = V_m H_m + h_{m+1,m} \underline{\mathrm{v}}_{m+1} \underline{\mathrm{e}}_m^T, \tag{6.5}$$

Algorithm 6.1 GMRES [172]

1: Compute $\underline{\mathrm{r}}_0 = \underline{\mathrm{f}} - A\underline{x}_0$, $\beta = \|\underline{\mathrm{r}}_0\|_2$, and $\underline{\mathrm{v}}_1 = \underline{\mathrm{r}}_0/\beta$
2: Initialize a matrix $\bar{H}_m = (h_{ij})_{1\le i\le m+1, 1\le j\le j}$
3: **for** $i = 1, \ldots, m$ **do**
4: $\quad \underline{w}_j = A\underline{\mathrm{v}}_j$
5: $\quad$ **for** $i = 1, \ldots, j$ **do**
6: $\quad\quad h_{ij} = (\underline{w}_j, \underline{\mathrm{v}}_j)$
7: $\quad\quad \underline{w}_j = \underline{w}_j - h_{ij}\underline{\mathrm{v}}_i$
8: $\quad$ **end for**
9: $\quad h_{j+1,j} = \|\underline{w}_j\|_2$
10: $\quad$ **if** $h_{j+1,j} = 0$ **then**
11: $\quad\quad m = j$, go to
12: $\quad$ **else if**
13: $\quad\quad \underline{\mathrm{v}}_{j+1} = \underline{w}_j/h_{j+1,j}$ **then**
14: $\quad$ **end if**
15: **end for**
16: Compute $\underline{y}_m = \arg\min \|\beta\underline{e}_1 - \bar{H}_m\underline{y}\|_2$
17: Compute $\underline{x} = \underline{x}_0 + V_m\underline{y}_m$

where $\underline{\mathrm{e}}_m$ is the i-the column of the identity matrix.

$$\|\underline{\mathrm{r}}_m\|_2 = \|V_{m+1}\underline{\mathrm{r}}_m\|_2 = \|V_{m+1}(\underline{\mathrm{r}}_0 - AV_m\underline{y}_m)\|_2 = \|V_{m+1}(\underline{\mathrm{r}}_0 - V_{m+1}\bar{H}_m\underline{y}_m)\|_2. \quad (6.6)$$

Since $\underline{\mathrm{r}}_0 = \beta\underline{\mathrm{v}}_1$ (according to line 1 in Algorithm 6.1) and $\{\underline{\mathrm{v}}_1, \ldots, \underline{\mathrm{v}}_{m+1}\}$ are an orthogonal basis, the best solution $\underline{y}_m$ is the minimizer of $\|\underline{\mathrm{r}}_m\|_2$, i.e.

$$\underline{y}_m = \arg\min_{\underline{y}} \|\beta\underline{\mathrm{e}}_1 - \bar{H}_m\underline{y}\|. \quad (6.7)$$

The resulting algorithm is described in Algorithm 6.1. The GMRES algorithm only requires storage of m vectors and a Hessenberg matrix of order $(m+1)\times m$. Usually $m \ll N$, this can be achieved by restarting the orthogonalization procedure if the first m iterations do not generate a good solution. The main storage for GMRES methods is the m vectors and $\underline{\mathrm{v}}_i$, $1 \le i \le m$ and the Hessenberg matrix.

Further memory efficient methods can be derived via the *biorthogonalization* procedure, which uses short recurrence to generate a basis set $\{\underline{\mathrm{v}}_1, \ldots, \underline{\mathrm{v}}_m\}$. The basis set $\{\underline{\mathrm{v}}_1, \ldots, \underline{\mathrm{v}}_m\}$ itself is not orthogonal but biorthogonal to another set of vectors $\{\underline{w}_1, \ldots, \underline{w}_m\}$ in the sense that $(\underline{\mathrm{v}}_i, \underline{w}_j) = \delta_{ij}$, for $1 \le i, j, \le m$.

6.2.3 Biorthogonalization methods

The aim of biorthogonalization-based Krylov methods is to use a short recurrence formula to update the approximate solution $\underline{x}_m$ and residual $\underline{\mathrm{r}}_m$. When the constraint space $\mathcal{L}_m$ is chosen as $\mathcal{K}_m(A^T, \underline{\mathrm{r}}_0)$, a biorthogonal system which consists of $\{\underline{\mathrm{v}}_1, \ldots, \underline{\mathrm{v}}_m\}$ and $\{\underline{w}_1, \ldots, \underline{w}_m\}$ is found through the Lanzcos biorthogonalization procedure in Algorithm 6.2. If the biorthogonalization procedure does not break down, the basis sets satisfy $\mathrm{span}\{\underline{\mathrm{v}}_1, \ldots, \underline{\mathrm{v}}_m\} = \mathcal{K}_m(A, \underline{\mathrm{r}}_0)$, $\mathrm{span}\{\underline{w}_1, \ldots, \underline{w}_m\} = \mathcal{K}_m(A^T, \underline{\mathrm{r}}_0)$, and the following conditions:

$$AV_m = V_m T_m + \delta_m \underline{\mathrm{v}}_{m+1} \underline{\mathrm{e}}_m^T, \tag{6.8}$$

$$A^T W_m = W_m T_m^T + \beta_{m+1} \underline{w}_{m+1} \underline{\mathrm{e}}_m^T, \tag{6.9}$$

$$W_m^T A V_m = T_m, \tag{6.10}$$

where $V_m = [\underline{\mathrm{v}}_1, \ldots, \underline{\mathrm{v}}_m]$, $W_m = [\underline{w}_1, \ldots, \underline{w}_m]$, β_{m+1} and δ_{m+1} are generated in Algorithm 6.2 and

$$T_m = \begin{pmatrix} \alpha_1 & \beta_2 & & & \\ \delta_2 & \alpha_2 & \beta_3 & & \\ & \ddots & \ddots & \ddots & \\ & & \delta_{m-1} & \alpha_{m-1} & \beta_m \\ & & & \delta_m & \alpha_m \end{pmatrix}. \tag{6.11}$$

The approximate solution can be updated according to $\underline{x}_m = \underline{x}_0 + V_m \underline{y}_m$ where $\underline{y}_m$ is the solution to $W_m^T(\underline{\mathrm{r}}_0 - AV_m \underline{y}_m) = 0$, i.e. $y_m = T_m^{-1} W^T \underline{\mathrm{r}}_0 = T_m^{-1} \|\underline{\mathrm{r}}_0\| \underline{\mathrm{e}}_1$.

A typical method of this type is the *biconjugate gradient* (BiCG) method [78]. Based on the BiCG method, the *conjugate gradient squared* (CGS) method [189] and the *biconjugate gradient stabilized* (BiCGSTAB) method [197] are derived. For more details on these methods, one can refer to [67, 171, 198]. We only emphasize that these methods based on biorthogonalization are much more memory efficient, as shown from Algorithm 6.3 to Algorithm 6.5. The main storage for these methods are the sparse matrix itself and several vectors. Such memory efficiency results in better performance if these methods converge when compared to a long recurrence method. The drawback of such methods is that they do not have the minimal

Algorithm 6.2 The Lanczos biorthogonalization procedure [171, p.206]

Choose two vectors $\underline{\mathrm{v}}_1$, $\underline{w}_1$ such that $(\underline{\mathrm{v}}_1, \underline{w}_1) = 1$.
$\beta_1 = \delta_1 = 0; \underline{w}_0 = \underline{\mathrm{v}}_0 = \underline{0}$.
for $j = 1, 2, \ldots, m$ **do**
  $\alpha_j = (A\underline{\mathrm{v}}_j, \underline{w}_j)$
  $\underline{\mathrm{v}}_{j+1} = A\underline{\mathrm{v}}_j - \alpha_j \underline{\mathrm{v}}_j - \beta_j \underline{\mathrm{v}}_{j-1}$
  $\underline{w}_{j+1} = A^T \underline{w}_j - \alpha_j \underline{w}_j - \delta_j \underline{w}_{j-1}$
  $\gamma_j = (\underline{\mathrm{v}}_{j+1}, \underline{\mathrm{v}}_{j+1})$
  $\delta_j = \sqrt{\gamma_j}$
  if $\delta_{j+1} = 0$ **then**
    Stop
  end if
  $\beta_{j+1} = \gamma_j / \delta_{j+1}$
  $\underline{w}_{j+1} = \underline{w}_{j+1} / \beta_{j+1}$
  $\underline{\mathrm{v}}_{j+1} = \underline{\mathrm{v}}_{j+1} / \delta_{j+1}$
end for

Algorithm 6.3 BiCG [171, p.211]

$\underline{\mathrm{r}}_0 = \underline{\mathrm{f}} - A\underline{x}_0$. Choose $\underline{\mathrm{r}}_0^*$ such that $(\underline{\mathrm{r}}_0, \underline{\mathrm{r}}_0^*) \neq 0$.
for $j = 0, 1 \ldots,$ until convergence **do**
  $\underline{q}_j = A\underline{p}_j$
  $\alpha_j = (\underline{\mathrm{r}}_j, \underline{\mathrm{r}}_j^*)/(\underline{q}_j, \underline{p}_j^*)$
  $\underline{x}_{j+1} = \underline{x}_j + \alpha_j p_j$
  $\underline{\mathrm{r}}_{j+1} = r_j - \alpha_j q_j$
  $\underline{\mathrm{r}}_j^* = \underline{\mathrm{r}}_j^* - \alpha_j A^T \underline{p}_j^*$
  $\beta_j = (\underline{\mathrm{r}}_{j+1}, \underline{\mathrm{r}}_{j+1}^*)/(\underline{\mathrm{r}}_j, r_j^*)$
  $\underline{p}_{j+1} = \underline{\mathrm{r}}_{j+1} + \beta_j \underline{p}_j$
  $p_{j+1}^* = \underline{\mathrm{r}}_{j+1}^* + \beta_j p_j^*$
end for

residual property.

6.2.4 Comparison of performance of Krylov subspace methods

Here we compare the performance of seven Krylov subspace methods for general non-symmetric linear systems. The seven methods are: a restarted GMRES method—GMRES(30), the BiCG method, CGS method, BiCGSTAB method, the quasi-minimal residual method (QMR) [95], the transpose-free quasi-minimal residual algorithm (TFQMR) [94], and the BiCGSTAB(ℓ) method with $\ell = 2$ [187]. We

Algorithm 6.4 CGS [189]

Compute $\underline{\mathrm{r}}_0 = \underline{\mathrm{f}} - A\underline{x}_0$; $\underline{\mathrm{r}}_0^*$ arbitrary
$\underline{\mathrm{u}}_0 = \underline{p}_0 = \underline{\mathrm{r}}_0$
for $j = 0, 1, 2, \ldots,$ until convergence **do**
    $\underline{s}_j = A\underline{p}_j$; $\alpha_j = (\underline{\mathrm{r}}_j, \underline{\mathrm{r}}_0^*)/(\underline{s}_j, \underline{\mathrm{r}}_0^*)$
    $\underline{q}_j = \underline{\mathrm{u}}_j - \alpha_j \underline{s}_j$
    $\underline{t}_j = \underline{\mathrm{u}}_j + \underline{q}_j$
    $\underline{x}_{j+1} = \underline{x}_j + \alpha_j \underline{t}_j$
    $\underline{\mathrm{r}}_j = \underline{\mathrm{r}}_j - \alpha_j A\underline{t}_j$
    $\beta_j = (\underline{\mathrm{r}}_{j+1}, \underline{\mathrm{r}}_0^*)/(\underline{\mathrm{r}}_j, \underline{\mathrm{r}}_0^*)$
    $\underline{\mathrm{u}}_{j+1} = \underline{\mathrm{r}}_{j+1} + \beta_j \underline{q}_j$
    $\underline{p}_{j+1} = \underline{\mathrm{u}}_{j+1} + \beta_j(\underline{q}_j + \beta_u p_j)$
end for

Algorithm 6.5 BiCGSTAB [197]

Compute $\underline{\mathrm{r}}_0 = \underline{\mathrm{f}} - A\underline{x}_0$; $\underline{\mathrm{r}}_0^*$ arbitrary; $\underline{p}_0 := \underline{\mathrm{r}}_0$.
for $j = 0, 1, \ldots,$ until convergence **do**
    $\underline{q}_j = A\underline{p}_j$
    $\alpha_j = (\underline{\mathrm{r}}_j, \underline{\mathrm{r}}_0^*)/(\underline{q}_j, \underline{\mathrm{r}}_0^*)$
    $s_j = r_j - \alpha_j \underline{q}_j$
    $t_j = As_j$
    $\omega_j = (\underline{t}_j, \underline{s}_j)/(\underline{t}_j, \underline{t}_j)$
    $\underline{x}_{j+1} = \underline{x}_j + \alpha_j \underline{p}_j + \omega_j \underline{s}_j$
    $\underline{\mathrm{r}}_{j+1} = \underline{s}_j - \omega_j \underline{t}_j$
    $\beta_j = \frac{(\underline{\mathrm{r}}_{j+1}, \underline{\mathrm{r}}_0^*)}{(\underline{\mathrm{r}}_j, \underline{\mathrm{r}}_0^*)} \frac{\alpha_j}{\omega_j}$
    $\underline{p}_{j+1} = \underline{\mathrm{r}}_{j+1} + \beta_j(\underline{p}_j - \omega_j \underline{q}_j)$
end for

do not present the detailed algorithm for QMR, TFQMR and BICGSTAB(ℓ), because these methods have all been implemented in Matlab and we shall soon see these methods may be not of interest for our problems.

The power of Krylov subspace methods immediately appears when comparing Table 6.1 with Table 6.2. Compared with the long recurrence iterative methods GMRES(30), the short recurrence methods have better performance according to Table 6.2. In the short recurrence methods, the BiCGSTAB and CGS perform relatively better for our problems.

The performance of the 7 iterative methods are compared according to the following model $T = cN^{\gamma}$, where T is time and N is the number of unknowns, and c,

Table 6.2: Timing results for Krylov subspace iterative methods

Prob	gmres(30)	bicgstab	bicgstabl(2)	bicg	cgs	qmr	tfqmr
B4	0.0284	0.0103	0.0125	0.0128	0.0094	0.0144	0.0135
B3	0.0758	0.0229	0.0239	0.0296	0.0194	0.0303	0.0331
B2	0.4010	0.1051	0.1298	0.1443	0.1000	0.1859	0.1648
B1	1.4365	0.4041	0.5375	0.5769	0.3788	0.6774	0.5398
D3	0.4384	0.0996	0.1159	0.1224	0.0790	0.1624	0.1351
D2	2.3513	0.4949	0.6740	0.6658	0.4996	0.8967	0.7116
D1	6.4997	1.9680	2.7264	2.5520	1.7487	3.0599	2.5119

Table 6.3: Iteration numbers for Iterative methods without preconditioning

Prob	gmres	bicgstab	bicgstabl(2)	bicg	cgs	qmr	tfqmr
B4	42	49	49	43	28	43	55
B3	28	33	33	29	18	29	34
B2	32	37	35	32	21	32	41
B1	27	32	29	27	16	27	30
D4	57	62	62	49	30	49	59
D3	58	63	61	51	33	51	65
D1	44	54	53	43	27	43	54

γ are unknown constants to be approximated. An iterative method with the smallest γ (and then smallest c) is recommended for large scale problems after several trials on some training sets. Based on such a simple performance evaluation model and the results in Table 6.4, the BiCGSTAB and CGS methods are recommended.

One important and sometimes essential technique to improve performance of iterative methods is preconditioning [41]. We shall see how much benefit we can obtain from preconditioning for our problems. Designing an efficient preconditioner usually requires customization based on the features of the underlying problem. We shall examine some properties of the radial basis function interpolation problems and then discuss some possible preconditioning techniques.

6.3 Near points preconditioning

In this section, we develop a new preconditioning technique based on near points selection. It can be used both to construct a sparse approximate inverse and to construct local incomplete factorization preconditioners. The underlying intuition and theoretical support come from three sources. Firstly, intuitions which come

Table 6.4: Comparison of the scalability of direct and iterative methods. The timing results are assumed to be $T = cN^{\gamma}$, where N is the number of unknowns, and c and γ are fitted constants. The 95% confident intervals for both constants are supplied.

Method	$c \in$ 95% confident interval	$\gamma \in$ 95% confident interval
amd+lu	1.01e-06 ∈[1.29e-07, 9.33e-06]	1.39 ∈[1.18 1.56]
gmres	1.58e-05 ∈ [2.70e-06,9.24e-05]	1.02 ∈[0.85,1.20]
bicgstab	6.65e-06 ∈ [1.55e-06,2.85e-05]	0.98 ∈[0.84,1.12]
bicgstabl(2)	5.50e-06 ∈ [1.02e-06,2.96e-05]	1.02 ∈[0.86,1.19]
bicg	7.48e-06 ∈ [2.01e-06,2.79e-05]	1.00 ∈[0.87,1.12]
cgs	5.36e-06 ∈ [1.16e-06,2.48e-05]	0.99 ∈[0.84,1.14]
qmr	7.16e-06 ∈ [1.47e-06,3.48e-05]	1.02 ∈[0.87,1.17]
tfqmr	9.74e-06 ∈ [2.58e-06,3.67e-05]	0.98 ∈[0.85,1.11]

from approximation theory are based on theoretical results on *cardinal* and *local cardinal functions* for interpolation with radial basis functions. Secondly, intuitions which come from preconditioning techniques inspired by approximate sparse inverse, incomplete factorization and recent preconditioners based on strong components and strong subgraphs [61, 62]. Third, our recent practice with *querying* large scale scattered data shows that the proposed preconditioners are almost by-products of the process of generating interpolation matrices from queried data, see Table 5.3.

6.3.1 Local cardinal function

A *cardinal function*, Ψ_j, is a Lagrangian function in the sense that

$$\Psi_j(\mathbf{x}_i) = \delta_{ij} = \begin{cases} 1 & \text{for } i = j; \\ 0 & \text{for } i \neq j. \end{cases}$$

With the Lagrangian basis, the interpolant with cardinal functions is written as

$$s(\mathbf{x}) = \sum_{i=1}^{N} f_j \Psi_j(\mathbf{x}).$$

We approximate or interpolate the j-th cardinal function by $\{\varphi_i(\mathbf{x})\}_{i=1}^{N}$ on the point set $\mathcal{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$: $\Psi_j(\mathbf{x}) = \sum_{k=1}^{N} \alpha_{j_k} \varphi_k(\mathbf{x})$. It turns out, because of the cardinal property, that $\underline{\alpha}_j = (\alpha_{j_1}, \ldots, \alpha_{j_N})^N$ is the j-th column of the right inverse of the interpolation matrix A in (2.1) i.e. $\underline{\alpha}_j = A^{-1}\underline{\mathrm{e}}_j$, where $\underline{\mathrm{e}}_j$ is the j-th column

of the identity matrix.

Interpolation with Lagrangian functions leads to a well conditioned diagonal interpolation matrix. But transforming the original basis to the Lagrangian basis is equivalent to finding the inverse of the interpolation matrix A in (1.1); and the inverse of an *irreducible matrix* A is full [64, Thm6]. So this leads to a more difficult problem than solving the original linear system (1.1)! In practice, an approximation of the coefficients of a Lagrangian function is enough and they can be approximated by computing a *local cardinal function*.

A local cardinal function ψ_j is approximated by $\{\varphi_{j_k}(\mathbf{x})\}_{j_k=1}^{M}$ with the basis centres in $\mathcal{S}_{\mathcal{X}} = \{\mathbf{x}_{j_1}, \ldots, \mathbf{x}_{j_M}\}$, where $\mathcal{S}_{\mathcal{X}}$ is a subset of $\mathcal{X}$, usually $\mathbf{x}_j \in \mathcal{S}_{\mathcal{X}}$ and $M \ll N$. Denote the coefficients of $\psi_j(\mathbf{x}) = \sum_{j_k=1}^{M} \tilde{\beta}_{j_k} \varphi_{j_k}(\mathbf{x})$ as the components of $\underline{\tilde{\beta}}_j \in \mathbb{R}^M$, then prolong $\underline{\tilde{\beta}}_j$ to $\underline{\beta}_j \in \mathbb{R}^N$ according to the index of $\mathcal{S}_{\mathcal{X}}$ in $\mathcal{X}$; other elements of $\underline{\beta}^j$ are zeros (hereafter, $\underline{\tilde{\beta}}$ is always a vector in $\mathbb{R}^M$, and $\underline{\beta} \in \mathbb{R}^N$). Denote $P = [\underline{\beta}_1, \ldots, \underline{\beta}_N]$, it turns out P is an *approximate sparse inverse* of A, i.e. $AP \approx I$. The preconditioned linear system becomes $AP\underline{\gamma} = \underline{\mathrm{f}}$, and $\underline{\alpha} = P\underline{\gamma}$.

A common way to construct an approximate inverse is described as follows. Solve N independent problems in the form $\min \|\tilde{A}\underline{\tilde{\beta}}_j - \underline{\tilde{\mathrm{e}}}_j\|$ approximately, where $\tilde{A}$ is a nearby matrix of A usually approximated by a submatrix of A. One natural question of such a procedure is how to approximate $\tilde{A}$. The first way is to use a principle submatrix, $R^T A R$, of A as $\tilde{A}$, where R is a matrix of order $N \times M$ which consists of M different columns of the identity matrix I and solve the deflated small linear system $R^T A R \underline{\tilde{\beta}}_j = R^T \underline{\mathrm{e}}_j$. Such a way is equivalent to solving a small interpolation problem on a subset $\mathcal{S}_{\mathcal{X}} \subset \mathcal{X}$. Alternatively, one can solve a least square problem $\min \|AR\underline{\tilde{\beta}}_j - \underline{\mathrm{e}}_j\|$ and use the prolonged vector $R^T \underline{\tilde{\beta}}_j$ as the j-th column of the approximate inverse.

Theoretical results on the asymptotic coefficients of cardinal functions for general radial basis function shed light on the problem of selection of $\mathcal{S}_{\mathcal{X}}$ or the restriction matrix R. Beatson et al. [10] suggest that the closest point set of each center is a natural choice based on the decay of properties of cardinal function for the multiquadric functions. By applying a far field expansion, they find that for two dimensional multiquadrics, the corresponding cardinal function $\Psi(\mathbf{x})$ decays on the order $\mathcal{O}(\|\mathbf{x}\|^{-3})$ for large $\|\mathbf{x}\|$ (in fact we shall see that this order can be improved and depends on the dimension of the space $\mathbb{R}^d$). Baxter in [9] uses the asymptotic

coefficients of the cardinal function on an infinite lattice [7, 9].

Theoretical results on the asymptotic coefficients of cardinal functions for general radial functions have been established by Buhmann [29, 33] , Micchelli [34, 35] and many others. In particular, see the result on Multiquadrics in [9, 33, 35], Gaussian in [168] and recent investigation by Fornberg et al. [84].

Let $\mathbf{k} = (k_1 \ldots, k_d)$ be an index of an infinite regular lattice, $\mathbb{Z}^d$. The cardinal function for a radial function is written as

$$\Psi(\mathbf{x}) = \sum_{\mathbf{k}\in\mathbb{Z}^d} c_{\mathbf{k}}\phi(\|\mathbf{x}-\mathbf{k}\|) \text{ and } \Psi(\mathbf{x}_k) = \delta_{\mathbf{k},0}. \tag{6.12}$$

Let ϕ be *admissible of order* m. We use concrete examples in Table 6.5 to exemplify such a kind of function rather than to give the rigorous (somewhat long worded) definition; this will not prohibit the reader from obtaining information for constructing a preconditioner. Curious readers are redirected to [29, Def. 4, p.231] for a definition. Buhmann proves the following result [29, Thm. 10]:

Theorem 6.1. *Let φ be an admissible of order m on $\mathbb{R}^d$. Then the coefficients satisfy $|c_{\mathbf{k}}| = \mathcal{O}(\|\mathbf{k}\|^{-2d-m+\epsilon})$, as $\|\mathbf{k}\| \to \infty$ for some $\epsilon \in [0, 1)$.*

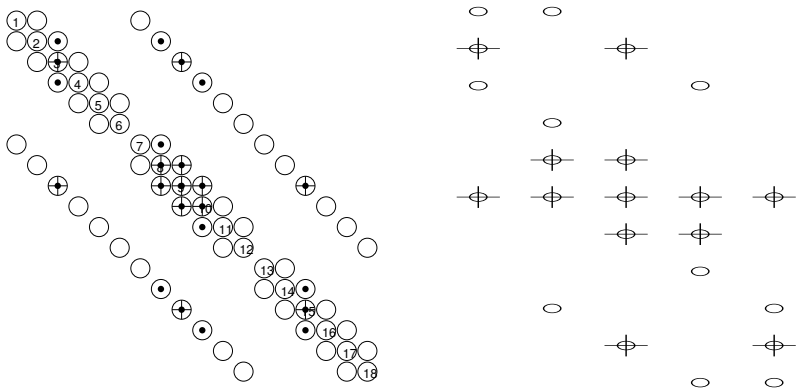
Theorem 6.1 states the decay order of the coefficients of the cardinal function not only depends on the underlying basis function but also the dimension of the space $\mathbb{R}^d$. For the same radial function from the underlying class of functions, the higher the dimension, the faster the decay of the coefficients $c_{\mathbf{k}}$ in the cardinal function in (6.12). In the corresponding finite dimensional case, this can be explained as the (i, j)-th element of A^{-1} decays as the distance $\|\mathbf{x}_i - \mathbf{x}_j\|$ grows. This brings a very encouraging strategy for choosing the sparse pattern of the approximate inverse: select the elements close to the center of the underlying cardinal function. Such a strategy is expected to work more effectively in higher dimensional space; only a small number of the nearest points should be sufficient.

6.3.2 Sparse approximate inverse

To describe an algorithm to compute the sparse approximate inverse based on local cardinal function approximation, let us introduce the following notation. Suppose each column of the underlying interpolation matrix has k_j non-zero elements, let $\mathcal{N}(\mathbf{x}_j, k)$ be the set of k nearest neighbour points, denote the index of $\mathcal{N}(\mathbf{x}_j, k)$

Table 6.5: The decay order of coefficients of cardinal functions of some radial basis functions

φ	$\mathbb{R}^d$	m	$c_{\mathbf{k}}$	reference
r^{2q+1}	d	$2q+1$	$\mathcal{O}(\Vert\mathbf{k}\Vert^{-2d-2q-1})$	[29, Ex.1][34, Ex.1]
$\sqrt{r^2+c^2}$	d	1	$\mathcal{O}(\Vert\mathbf{k}\Vert^{-2d-1})$	[29, Ex.2][34, Ex.1]
$r^{2q}\log r$	d	$2q$	$\mathcal{O}(\Vert\mathbf{k}\Vert^{-2d-2q-1})$	[29, Ex.3][34, Ex.1]
$(r^2+c^2)^{-1/2}$	d	-1	$\mathcal{O}(\Vert\mathbf{k}\Vert^{-2d+1})$	[29, Ex.4][34, Ex.1]
r^3	1		$=\frac{(-1)^k 3\sqrt{3}}{(2+\sqrt{3})^k}, k\geq 2$	[84, eq. 2.9][83]
r^3	2		$\approx -(\frac{5}{2\pi})^2\frac{1}{(k_1^2+k_2^2)^{7/2}}$	[84, eq. 5.1]
$\sqrt{1+(\varepsilon r)^2}$	1	$\approx$	$\frac{(-1)^k\varepsilon^2\sinh(\frac{\pi}{\varepsilon})}{2\pi\cosh(\frac{k\pi\varepsilon}{2})}-\frac{\tanh((\frac{\pi}{\varepsilon})^2)}{\pi^2}\frac{1}{k^2}$	[84, eq. 3.3]



(a) A (b) $A(\mathfrak{L}_9^5,\mathfrak{J}_9^5)$

Figure 6.3: Illustration of local cardinal function selection. (a) A matrix generated by the 5-point finite difference scheme on a regular mesh. $\mathfrak{J}_9^5=\{3,8,9,10,15\}$, $\mathfrak{L}_9^5=\{2,3,4,7,8,9,10,11,14,15,16\}$, $\mathfrak{L}_9^5\backslash\mathfrak{J}_9^5=\{2,4,7,11,14,16\}$, where the elements marked by dot-circle belong to $A(\mathfrak{L}_9^5,\mathfrak{J}_9^5)$, elements marked by dot-circle-cross belong to $A(\mathfrak{J}_9^5,\mathfrak{J}_9^5)$. (b) the submatrix $A(\mathfrak{L}_9^5,\mathfrak{J}_9^5)$, where the small circles are non-zero elements in $A(\mathfrak{L}_9^5\backslash\mathfrak{J}_9^5,\mathfrak{J}_9^5)$.

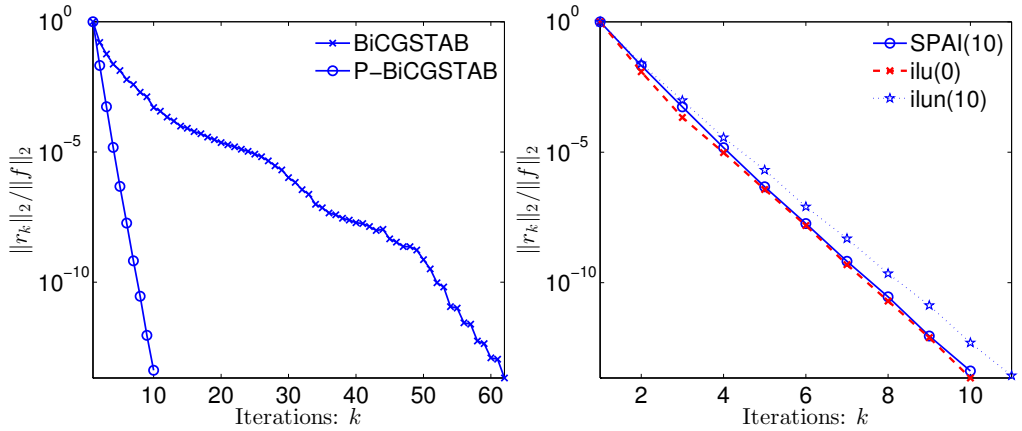
as $\mathfrak{J}_j^k=\{m:\mathbf{x}_m\in\mathcal{N}(\mathbf{x}_j,k)\text{ or }m=j\}$, and

$$\mathfrak{L}_j^k=\{i:\mathbf{x}_i\in\bigcup_{\mathbf{x}_m\in\mathfrak{J}_j^k}\mathcal{N}(\mathbf{x}_m,k_m)\}.$$

Using Matlab notation, define $A(\mathfrak{L}_j^k,\mathfrak{J}_j^k)$ and $R_{\mathfrak{L}_j^k}=I(:,\mathfrak{J}_j^k)$ as follows. $A(\mathfrak{L}_j^k,\mathfrak{J}_j^k)$ is the non-zero rows of matrix $A(:,\mathfrak{J}_j^k)$. Each local cardinal function is computed

Algorithm 6.6 SPAI: sparse approximate inverse by local cardinal function approximation

1: Initialization with queried data, supply k_0 and k_{max};
2: **for** $(j = 0;\ j < N\ ;++j)$ **do**
3: `set` $k \leftarrow k_0$, $\mathfrak{J} \leftarrow \mathfrak{J}_j^k$;// choose the k nearest neighbours to $\mathbf{x}_j$
4: `find row index:` $\mathfrak{L} \leftarrow \mathfrak{L}_j^k$; // keep non-zero rows of $A(:, \mathfrak{J}_j^k)$
5: `minimize` $\|A(\mathfrak{L}, \mathfrak{J})\underline{\beta}_j - R_{\mathfrak{L}}\underline{\mathrm{e}}_j\|$; // compute local cardinal function
6: **while** $\|A(\mathfrak{L}, \mathfrak{J})\underline{\beta}_j - R_{\mathfrak{L}}^T\underline{\mathrm{e}}_j\| \geq$ `tol` $\&k < k_{max}$ **do**
7: `increase` k `to` $k + m$, $m \geq 1$; // increase neighbours, default m=1;
8: `set` $\mathfrak{J} \leftarrow \mathfrak{J}_j^{k+m}$, $\mathfrak{L} \leftarrow \mathfrak{L}_j^{k+m}$;
9: `solve` $\|A(\mathfrak{L}, \mathfrak{J})\underline{\beta}_j - R_{\mathfrak{L}}\underline{\mathrm{e}}_j\|$ `by (QR) updating algorithm;`
10: **end while**
11: **end for**



(a) (b)

Figure 6.4: Convergence history related to sparse approximate inverse. (a) illustrates the convergence history of BiCGSTAB on problem B4 without (BiCGSTAB) and with a sparse approximate inverse preconditioner(P-BiCGSTAB). The sparse approximate inverse is constructed such that the local cardinal function involves at most 10 neighbour basis functions. (b) illustrates the convergence history of BiCGSTAB on problem B4 with different preconditioners. SPAI(10) stands for the sparse approximate inverse preconditioning with at most 10 basis functions to form a local cardinal function. ilu(0) stands for the standard incomplete factorization preconditioners with zero level fill-in. ilun(10) stands for the local incomplete factorization preconditioners.

by solving the following least square problem

$$\underline{\tilde{\beta}}_j^* = \arg\min_{\underline{\tilde{\beta}}_j} \|A(\mathfrak{L}_j^k, \mathfrak{J}_j^k)\underline{\tilde{\beta}}_j - R_{\mathfrak{L}_j^k}\underline{\mathrm{e}}_j\|. \tag{6.13}$$

The j-th column of the approximate inverse is $R_{\mathfrak{J}_j^k}\tilde{\underline{\beta}}_j^*$. Further denote the solution to the local cardinal interpolation problem as $\tilde{\underline{\beta}}_j^{**}$, i.e.,

$$A(\mathfrak{J}_j^k, \mathfrak{J}_j^k)\tilde{\underline{\beta}}_j^{**} = R_{\mathfrak{J}_j^k}^T \underline{\mathrm{e}}_j. \tag{6.14}$$

The solution $\tilde{\underline{\beta}}_j^*$ is better than $\tilde{\underline{\beta}}_j^{**}$ because

$$\|A(R_{\mathfrak{J}_j^k}\tilde{\underline{\beta}}_j^*) - \underline{\mathrm{e}}_j\| \leq \|A(\mathfrak{L}_j^k, \mathfrak{J}_j^k)\tilde{\underline{\beta}}_j^{**} - R_{\mathfrak{L}_j^k}\underline{\mathrm{e}}_j\| = \|A(\mathfrak{L}_j^k \backslash \mathfrak{J}_j^k, \mathfrak{J}_j^k)\tilde{\underline{\beta}}_j^{**}\|. \tag{6.15}$$

An algorithm based on the least square approach is described in Algorithm 6.6. Figure 6.4 illustrates the convergence history of the BiCGSTAB method with and without a sparse approximate inverse. At present the SPAI algorithm is implemented in Matlab, but it turns out it takes a long time to construct the approximate inverse even for the small problem B4. Now we consider a simple and more practical approach.

6.3.3 Local incomplete factorization

The local cardinal function approach is based on the fast decay property of the elements of the inverse. The elements of the interpolation matrix with CSRBFs also have a fast decay property. Employing this property, a local incomplete factorization preconditioner can be constructed; local incomplete refers to the incomplete factorization applied on a submatrix of of A. Each column of the submatrix only consists of at most k largest elements of A, this corresponds to the first k nearest points to each basis center. For memory efficiency, each column of the submatrix only keeps $\min\{k, n_j\}$ non-zero elements, where n_j is the number of non-zero elements in the j-th column of A. Precisely, split A into two parts $A = N(k) + E$, where

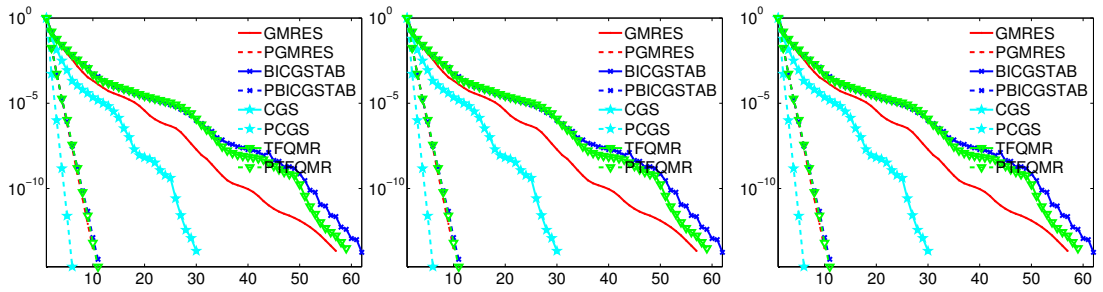
$$N_{ij} = \begin{cases} a_{ij} & \text{if}, i \in \mathfrak{J}_j^{\min\{k,n_j\}}, \\ 0 & \text{otherwise}, \end{cases} \tag{6.16}$$

and apply the incomplete factorization on the matrix $N(k)$. We refer to the method as *ilun(k)*.

Such a ilun(k) incomplete factorization is guaranteed to exist for the interpolation matrices obtained from Chapter 5, because A is an H-matrix as shown in

Algorithm 6.7 ilun(k): local incomplete LU factorization preconditioner based on k nearest points splitting.

1: Generating the matrix $N(k)$ according to (6.16).
2: Apply standard incomplete factorization algorithms to $N(k)$.



(a) ilunk(7) for D3 (b) ilunk(7) for D2 (c) ilunk(7) for D1

Figure 6.5: Convergence history of iterative methods with and without ilun(k) preconditioners.

k=4		k=7		k=10		k=13	
nnz(L+U)	time	nnz(L+U)	time	nnz(L+U)	time	nnz(L+U)	time
1200990	0.0579	2058621	0.1301	2712241	0.1669	3038710	0.2134

Table 6.6: Storage and timing for local incomplete factorization for problem D1.

Theorem 4.4 and stable incomplete factorization for H-matrix exists [202]. Discarding some small positive values in each column of A makes $N(k)$ more diagonally dominant.

One can also suggest admitting fill-in in the ilun(k) preconditioner by allowing $\max\{k, n_j\}$ non-zeros elements of the the j-th column of $L + U$. These non-zero elements lie in the first $\max\{k, n_j\}$ nearest neighbour points for each basis center. However, as k increases, the storage for the preconditioners will increase, as will the construction time. See Table 6.6 for illustration. Besides, a larger k also results in more sequential steps in the triangular solves, which may result in inefficiency in the preconditioning step. For our problems, numerical results show that choosing a larger k than the average number of non-zero elements per column is not necessary. Table 6.6 to Table 6.8 compare the effect of choosing different k.

By choosing a proper k, $ilun(k)$ works better than the standard incomplete factorization methods. The efficiency of ilun(k) is illustrated in Table 6.10, which

k	gmres	bicgstab	bicgstabl(2)	bicg	cgs	qmr	tfqmr
4	1.4688	0.8606	1.0623	1.8994	0.8015	2.0700	0.9284
7	1.3436	0.7929	1.0231	1.9600	0.8236	2.1870	0.9209
10	1.4732	0.8824	1.1721	2.3662	0.9240	2.5683	0.9436
13	1.4848	0.9424	1.2033	2.4396	0.9888	2.6955	1.0266

Table 6.7: Timing results for preconditioned iterative methods with local incomplete factorization based on k nearest neighbour points for problem D1.

k	gmres	bicgstab	bicgstabl(2)	bicg	cgs	qmr	tfqmr
4	12	13	13	12	7	12	13
7	10	10	10	10	6	10	10
10	10	10	10	10	6	10	10
13	10	10	10	10	6	10	10

Table 6.8: Iteration number for preconditioned iterative methods with k nearest neighbour points incomplete factorization for problem D1.

Prob	preconditioners	gmres	bicgstab	cgs
D1	ilun(7)	1.2743	0.7089	0.7165
D1	ilu(0)	1.3311	0.7854	0.7890

Table 6.9: Timing results for iterative methods with two different preconditioning techniques.

	ilun(7)		ilu(0)	
Prob	nnz(L+U)	time	nnz(L+U)	time
B4	9179	0.0003	14603	0.0007
B3	38977	0.0015	63678	0.0033
B2	169030	0.0072	275281	0.0165
B1	726220	0.0426	1127137	0.0823
D3	106621	0.0041	173323	0.0088
D2	474251	0.0233	763863	0.0499
D1	2058621	0.1277	3167659	0.2143

Table 6.10: Storage and construction time for ilun(7) and ilu(0) preconditioners.

compares the storage requirement and the construction time for ilun(k) and ilu(0). Table 6.9 shows the time for preconditioned Krylov methods to solve the largest problem D1.

6.4 Methods for dense linear systems

6.4.1 Iterative methods based on fast multipole methods

Applying iterative methods to dense interpolation linear systems from globally supported radial basis functions is possible. The main computational kernel is the dense matrix vector multiplication which usually takes $\mathcal{O}(N^2)$ operations. Fast multipole methods can be used to approximate the underlying matrix vector multiplication. Building up a hierarchical data structure for fast multipole methods takes $\mathcal{O}(N \log N)$ operations, and computing the matrix vector multiplication takes $\mathcal{O}(N \log N)$ operations [208, p.257]. Fast multipole methods based on a far field approximation are usually customized for the underlying basis function; for polyharmonic splines see [11, 12, 16], for multiquadrics [43], for Gaussians [19, 170, 191] and for Matérn kernel [40]. Kernel independent fast multipole methods [216, 217] are based on the SVD decomposition in $\mathbb{R}^2$ and the fast Fourier transform (FFT) in $\mathbb{R}^3$.

6.4.2 Discrete Laplacian preconditioner

Preconditioner(s) can be applied to iterative methods for dense linear systems. In fact, preconditioners play a crucial role in solving these linear systems from globally supported radial basis functions. Globally supported radial basis functions are usually smoother than compactly supported radial basis functions, and as shown in Section 3.1, the eigenvalues of smoother radial basis functions decay faster and thus the resulting linear system is more ill-conditioned.

For 2-dimensional interpolation with surface splines $\phi(r) = r^{(2m-1)} \log(r)$ and multiquadrics $\phi(r) = \sqrt{r^2 + c^2}$ on a lattice, Dyn and Levin propose a discrete Laplacian operator as a preconditioner for linear systems arising from integral equations and surface interpolation [65, 66] . Such a preconditioner is based on the fact that the Laplacian operator, Δ, satisfies

$$\lim_{r \to \infty} \Delta\phi(r) = 0 \text{ and } \lim_{r \to 0} \Delta\phi(r) \gg 1 \text{ or } \lim_{r \to 0} \Delta\phi(r) = \infty. \tag{6.17}$$

Therefore, the "diagonal dominance" of the preconditioned matrices is expected to be better than that of the original ones whose off-diagonal elements are bigger than

diagonal elements.

6.4.3 Iterative method based on approximate Lagrangian functions

Iterative methods can also be constructed based on the semi-norm of the native space of (conditionally) positive definite kernels. A commonly used iterative method for globally supported radial basis function is based on the approach of approximate Lagrange functions which dates back to Beatson and Powell [15]. The theoretical analysis can be found in [75] and a relatively recent description in [74]. The method is based on approximation of Lagrangian functions. Suppose the true interpolant is

$$s(\mathbf{x}) = \sum_{k=1}^{N} f_k \Psi_k(\mathbf{x}), \tag{6.18}$$

where $\Psi_k(\mathbf{x})$ are the cardinal functions on $\mathcal{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$. The idea of the approximate Lagrangian function approach is to replace the cardinal functions by local cardinal functions $\psi_j(\mathbf{x})$ which are more easily computed. The approximant can be written as

$$\tilde{s}(\mathbf{x}) = \sum_{k=1}^{N} f_k \psi_k(\mathbf{x}), \tag{6.19}$$

Each local cardinal function can be computed on a subset $\mathfrak{L}_k$ of $\mathcal{X}$ in advance and then $\tilde{s}(\mathbf{x})$ can be updated iteratively according to Algorithm 6.8. The data set $\mathcal{X}$ is divided into two parts $\mathfrak{L} = \{1, 2, \ldots, N-q\}$ and $\mathfrak{L}^* = \{N-q+1, \ldots, N\}$. Only $N-q$ cardinal functions are computed in advance. In the j-th iteration, the approximant is updated by $s_k^{(j)}(\mathbf{x}) = s_{k-1}^{(j)}(\mathbf{x}) + \theta_k^{(j)} \psi_k(\mathbf{x})$, $1 \leq k \leq N-q$, where

$$\theta_k^{(j)} = \frac{1}{\lambda_{kk}} \sum_{i \in \mathfrak{L}_k} (f_i - s_{k-1}^{(j)}(\mathbf{x}_i)) \lambda_{ki}, \tag{6.20}$$

and λ_{ki} are the coefficients which determine the local cardinal functions

$$\psi_k(\mathbf{x}) = \sum_{i \in \mathfrak{L}_k} \lambda_{ki} \phi(\|\mathbf{x} - \mathbf{x}_i\|) + p_k(\mathbf{x}). \tag{6.21}$$

$p_k(\mathbf{x})$ is a polynomial term for conditionally positive/negative definite basis functions. Such a choice guarantees that the error decreases [75][208, p.267].

Algorithm 6.8 Iterative method based on approximate Lagrange functions [208, p.268]

Set $s_0^0(\mathbf{x}) = 0$, $r_i = f_i$ for $i = 1, \ldots, N$, $\epsilon_i = \max_i |r_i|$, $j = 0$
while $\epsilon >$ `tol` **do**
    for $k = 1$ to $N - q$ **do**
        $s_k^{(j)}(\mathbf{x}) = s_{k-1}^{(j)}(\mathbf{x}) + \theta_k^{(j)} \psi_k(\mathbf{x})$
    end for
    Find an interpolant $\sigma^{(j)}(\mathbf{x})$, s.t. $\sigma^{(j)}(\mathbf{x}_i) = f_i - s_{N-q}^{(j)}(\mathbf{x}_i)$, $N-q+1 \le i \le N$.
    Update $s_0^{(j+1)} = s_{N-q}^{(j)} + \sigma^{(j)}(\mathbf{x})$.
    Update residual: $r_i^{(j)} = f_i - s_0^{(j+1)}(\mathbf{x}_i)$, $\epsilon = \max_i |r_i^{(j+1)}|$.
    Set $j \leftarrow j + 1$.
end while

Preconditioning schemes for Algorithm 6.8 have been developed in [100].

6.4.4 RBF-QR methods

For Gaussian radial basis functions, stable methods have been customized for Gaussian RBFs with flat limit. As we have seen in Chapter 3, radial basis functions with flat limit lead to ill-conditioned linear systems. The RBF-QR methods were proposed for interpolation on a sphere. They are based on an orthogonal spherical expansion [86, eq (3.3)]:

$$\phi(\varepsilon\|\mathbf{x} - \mathbf{x}_i\|) = \sum_{\mu=0}^{\infty} \sum_{\nu=-\mu}^{\mu} \{c_{\mu,\nu} \varepsilon^{2\mu} Y_\mu^\nu(\mathbf{x}_i)\} Y_\mu^\nu(\mathbf{x}), \tag{6.22}$$

where $\mathbf{x} = (x, y, z)$ on the sphere S^2 and Y_μ^ν is the spherical harmonic. Y_μ^ν is defined as [86, eq (3.1)]

$$Y_\mu^\nu(x, y, z) = \begin{cases} \sqrt{\frac{2\mu+1}{4\pi}} \sqrt{\frac{(\mu-\nu)!}{(\mu-\nu)!}} P_\mu^\nu(z) \cos(\nu \tan^{-1}(\frac{y}{x})), & \nu = 0, 1, \cdots, \mu, \\ \sqrt{\frac{2\mu+1}{4\pi}} \sqrt{\frac{(\mu+\nu)!}{(\mu-\nu)!}} P_\mu^{-\nu}(z) \sin(-\nu \tan^{-1} \frac{y}{x}), & \nu = -\mu, \cdots, -1, \end{cases} \tag{6.23}$$

where $P_\mu^\nu(z)$ are the *associated Legendre functions*. The spherical harmonics of order no more than k span the space of all independent polynomials in (x, y, z) of degree no more than k, i.e.

$$\dim\{Y_\mu^\nu : \nu = -\mu, -\mu+1, \cdots, \mu, \mu \le k\} = \dim\{\pi_k(\mathbb{R}^3) : x^2 + y^2 + z^2 = 1\}. \tag{6.24}$$

The number of these terms $c_{\mu,\nu}\varepsilon^{2\mu}$ in the order $\varepsilon^{2\mu}$ in (6.22) is exactly

$$\dim \pi_\mu(\mathbb{R}^3|_{S^2}) - \dim \pi_{\mu-1}(\mathbb{R}^3|_{S^2}), \tag{6.25}$$

where $\mathbb{R}^3|_{S^2}$ indicates the polynomials are restricted on the sphere. This coincides with the number of eigenvalues in the order of ε^{2k} in Theorem 3.9. In fact, this is just an orthogonal eigenvalue expansion on a sphere. Based on such an orthogonal eigenvalue expansion, the transform between the basis of translates of a radial basis function and the orthogonal spherical harmonics can be written as

$$\begin{pmatrix} \phi(\varepsilon\|\mathbf{x}-\mathbf{x}_1\|) \\ \phi(\varepsilon\|\mathbf{x}-\mathbf{x}_i\|) \\ \vdots \\ \phi(\varepsilon\|\mathbf{x}-\mathbf{x}_N\|) \end{pmatrix} = B \begin{pmatrix} Y_0^0(\mathbf{x}) \\ Y_0^{-1}(\mathbf{x}) \\ Y_1^0(\mathbf{x}) \\ Y_1^1(\mathbf{x}) \\ \vdots \end{pmatrix}, \tag{6.26}$$

where

$$B = \begin{pmatrix} c_{0,0}Y_0^0(\mathbf{x}_1) & \varepsilon^2 c_{1,0}Y_1^0(\mathbf{x}_1) & \varepsilon^2 c_{1,-1}Y_1^{-1}(\mathbf{x}_1) & \varepsilon^2 c_{1,1}Y_1^1(\mathbf{x}_1) & \cdots \\ c_{0,0}Y_0^0(\mathbf{x}_2) & \varepsilon^2 c_{1,0}Y_1^0\mathbf{x}_2 & \varepsilon^2 c_{1,-1}Y_1^1\mathbf{x}_2 & \varepsilon^2 c_{1,1}Y_1^1(\mathbf{x}_2) & \cdots \\ \vdots & \vdots & \vdots & \cdots & \\ c_{0,0}Y_0^0(\mathbf{x}_n) & \varepsilon^2 c_{1,-1}Y_1^{-1}(\mathbf{x}_n) & \varepsilon^2 c_{1,0}Y_1^0(\mathbf{x}_N) & \varepsilon^2 c_{1,1}Y_1^1(\mathbf{x}_N) & \cdots \end{pmatrix}. \tag{6.27}$$

If we write B as

$$B = \begin{pmatrix} Y_0^0(\mathbf{x}_1) & Y_1^{-1}(\mathbf{x}_1) & \cdots \\ Y_0^0(\mathbf{x}_2) & Y_1^{-1}(\mathbf{x}_2) & \cdots \\ \vdots & \vdots & \vdots \end{pmatrix} \begin{pmatrix} \varepsilon^0 & & \\ & \varepsilon^2 & \\ & & \ddots \end{pmatrix} \begin{pmatrix} c_{0,0} & & \\ & c_{1,-1} & \\ & & \ddots \end{pmatrix} = QRD_1D_2, \tag{6.28}$$

then the QR decomposition of the first matrix in (6.28) is completely independent of the kernel functions. The pattern of the D_1 on the sphere can be determined by the number of spherical harmonics in the order μ. The term D_2 depends on the underlying kernel. It has been reported that such an approach is more stable than using direct factorization methods to solve the underlying interpolation matrix with flat limit [85, 86, 126].

6.5 Discussion

In this chapter, we have reduced the time for solving the largest 3D object modelling problem (D1) from more than 40 seconds with a sparse direct methods with dedicated fill-reducing ordering to less than 1 second with a preconditioned Krylov method.

Compared with sparse direct methods, iterative methods require less storage: only for the sparse matrix, the preconditioner(s), and several vectors. The storage for preconditioners usually can be controlled. In the Krylov subspace methods, the short recurrence methods based on biorthogonalization performs better due to memory efficiency. Further we can show the realistic performance of sparse iterative methods and other computations with sparse features is limited by the *memory bandwidth* on a shared memory machine. On distributed computers, however, minimizing communication, especially minimizing synchronizations [221] plays a critical role in improving the performance of Krylov methods. It should be noticed that performance evaluation for iterative methods is receiving increasing attention with the emergence of the new metric, high performance conjugate gradient (HPCG), for ranking high performance computers [59]. Performance evaluation is becoming a standalone research subject because computing hardware is continuously changing and we want explore the full potential of the multitude of computer architectures in use. Such details go beyond our current discussion, but we shall see that realizing such practical issues can bring us insight into designing efficient algorithms in Section 8.1.

As seen in Chapter 3, the Picard condition indicates that the ill-conditioning arises because the smoothness of the target functions and the underlying basis functions does not match. The idea of of using radial basis function of different shapes in Chapter 4 is to adjust the shape of the basis functions so that the resulting interpolation matrices are well conditioned. The sparse linear systems which are generated in Chapter 5 are well conditioned due to the diagonal dominance conditions. Therefore, any standard Krylov subspace method for non-symmetric matrices works well without any preconditioning. On choosing a proper preconditioner, we recommend the local incomplete preconditioners ilun(k). To avoid preconditioning which is fast in converge rate but slow in overall performance, one should choose a proper k such

that a trade-off can be achieved. For the 7 iterative methods used in this chapter, the long recurrence method, GMRES, takes a longer time than the other methods. For our problems, we recommend a preconditioned BICGSTAB methods or conjugate squared with the ilun(k) preconditioners. The sparse approximate inverse (SPAI) approach based on approximate Lagrangian function requires more computations. Indeed, our current implementation has not shown any advantage than the ilun(k) preconditioners.

The ill-conditioning phenomena of interpolation with multiquadrics had been noticed soon after the multiquadrics collocation method appeared [122]. Baxter considered preconditioned conjugate gradient methods for multiquadric interpolation by exploring the Toeplitz structure of the interpolation matrices on grids [9]. Preconditioning based on structured Toeplitz matrices has also been considered in [37].

The approximate Lagrangian/cardinal function approach dates back to Beatson and Powell [15]. Preconditioning based on this approach is also referred to as the least square approach to preconditioning [10, 25, 73, 135]. Domain decomposition methods and overlapping additive Schwarz preconditioners have been considered in [13, 111, 134]. Preconditioning for spherical radial basis function has been considered in [127, 129, 194].

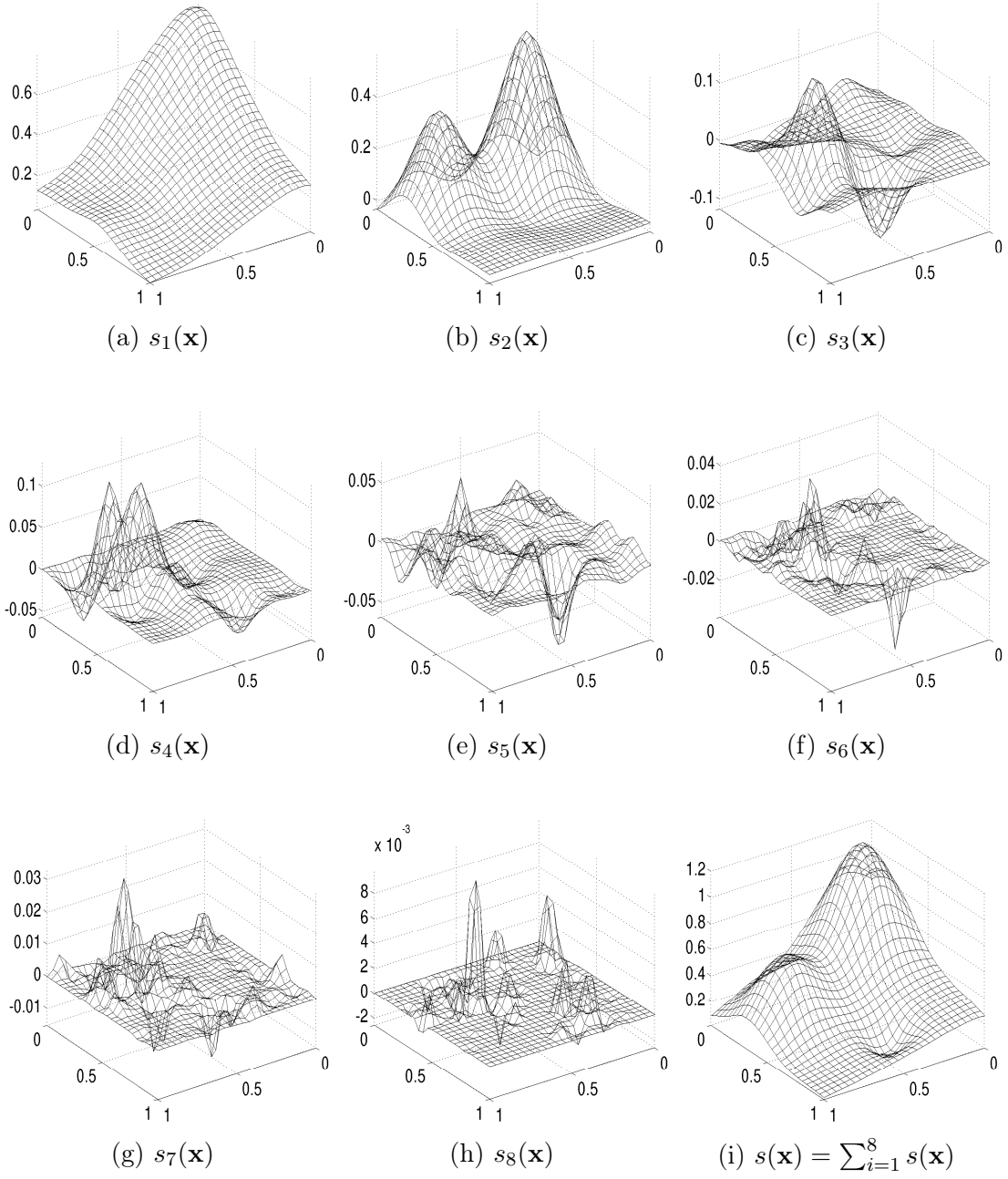
7. Adaptive Algorithms

This chapter examines a multilevel interpolation method. The method uses compactly supported radial basis functions of different shapes at each level according to the approach described in Chapter 4 and Chapter 5. Because it allows heterogeneity at each level, we call it the heterogeneous hierarchical approach. The proposed method has a good scalability and convergence of numerical results are observed. This differs from that of the so-called multilevel stationary interpolation approach, which is scalable but does not converge. At the same time, we design an $\mathcal{O}(N)$ algorithm to construct hierarchical scattered data sets, which is a necessary step before applying hierarchical methods to scattered data. Also discussed is the *'leave-one-out cross-validation'* method to find an optimal shape parameter for a global support radial basis function.

7.1 Hierarchical interpolation with CSRBFs: illustration

As mentioned in Chapter 4, stationary interpolation with positive definite compactly supported radial basis functions has a good scalability, but it does not converge because it violates the Strang-Fix condition. Non-stationary interpolation with positive definite compactly supported radial basis functions can converge, but the benefit of sparsity will be lost as the size of the data set increases. Our hierarchical approach gives a trade-off between the scalability and convergence.

Given some hierarchical data sets $\mathcal{X}_1 \subset \mathcal{X}_2 \ldots \subset \mathcal{X}_\ell = \mathcal{X}$, and observations, $\underline{\mathrm{f}} = (f_1, f_2, \ldots, f_n)^T$, on the data set, $\mathcal{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\}$, the idea of a multilevel approach can be described as in Algorithm 7.1, where the notation $\underline{\mathrm{f}}|_{\mathcal{X}_i}$, $i = 1, 2, \ldots \ell$, stands for the (restricted) observations on the subset $\mathcal{X}_i$. In such an approach, $s_i(\mathbf{x})$ interpolates the residual at level i, $\underline{\mathrm{r}}_{i-1}|_{\mathcal{X}_i}$ on data set $\mathcal{X}_i$. The number of zeros of $\underline{\mathrm{r}}_i$ increases as the level i increases. More zeros in $\underline{\mathrm{r}}_i$ result in an interpolant of higher frequency. In this sense, the multilevel approach can recover information on different frequencies at different levels. If we scale each interpolant $s_i(\mathbf{x})$ by its



(a) $s_1(\mathbf{x})$ (b) $s_2(\mathbf{x})$ (c) $s_3(\mathbf{x})$

(d) $s_4(\mathbf{x})$ (e) $s_5(\mathbf{x})$ (f) $s_6(\mathbf{x})$

(g) $s_7(\mathbf{x})$ (h) $s_8(\mathbf{x})$ (i) $s(\mathbf{x}) = \sum_{i=1}^{8} s(\mathbf{x})$

Figure 7.1: Illustration of frequency information of multilevel interpolation. Interpolate the `franke` function on 289 Halton points at 8 levels according to Algorithm 7.2. (a) to (h) illustrate the shape and magnitude of the interpolant $s_i(\mathbf{x})$, $i = 1, 2, \ldots, 8$. (i) illustrates the sum of the 8 interpolants. The 289 Halton points can be regenerated according to the following Matlab code (functions in statistics tool box) `p=haltonset(2,'Skip',1e2,'Leap',10); p=scramble(p,'RR2'); center=net(p,289);` and the hierarchical data sets are constructed according to Program 7.6.

Algorithm 7.1 An algorithm for multilevel interpolation

1: Input $\underline{\mathrm{f}}$ and $\mathcal{X}_1, \ldots, \mathcal{X}_\ell$, $\underline{\mathrm{r}}_0 = \underline{\mathrm{f}}$.
2: **for** $i = 1$ to ℓ **do**
3: Find an interpolant $s_i(\mathbf{x})$ s.t. $s_i|_{\mathcal{X}_i} = \underline{\mathrm{r}}_{i-1}|_{\mathcal{X}_i}$.
4: Update the residual $\underline{\mathrm{r}}_i = \underline{\mathrm{r}}_{i-1} - s_i|_{\mathcal{X}}$.
5: **end for**
6: $s(\mathbf{x}) = s_1(\mathbf{x}) + s_2(\mathbf{x}) + \ldots + s_\ell(\mathbf{x})$.

supremum norm on Ω, then

$$s(\mathbf{x}) = \sum_{i=1}^{\ell} \sup s_i(\mathbf{x}) \frac{s_i(\mathbf{x})}{\sup s_i(\mathbf{x})}. \tag{7.1}$$

We can relate the sum (7.1) to a generalised Fourier truncation in the sense that if the coefficients, $\sup s_i$, decay fast for smooth functions, then $\|f(\mathbf{x}) - s(\mathbf{x})\|_*$ is expected to tend to zero in a certain norm, $\|\cdot\|_*$, as ℓ goes to infinity. When ℓ is a finite number, the approximation quality, $\|f(\mathbf{x}) - s(\mathbf{x})\|_*$, should depend on the decay order of $\sup s_i(\mathbf{x})$, which itself depends on the decay order of $\underline{\mathrm{r}}_i$. Such an analogy may supply a direction to prove the convergence rate by a contraction technique.

To exemplify such an idea, we interpolate the `franke` function in Matlab on 289 Halton points according to a multilevel method described in Algorithm 7.2. The data set is partitioned into 8 levels; the 8 interpolants, $s_i(\mathbf{x})$, for $i = 1 \ldots, 8$, and their sum, $s(\mathbf{x})$, are illustrated in Figure 7.1. The hierarchical data sets are constructed according to Program 7.6.

7.2 Heterogeneous hierarchical interpolation on regular mesh

Unlike the commonly used multilevel approaches, which use radial basis functions of the same scale at one level, we allow heterogeneity at the same level and vary the supports at each level adaptively according to the approaches described in Chapter 4 and Chapter 5. We call this method *heterogeneous hierarchical interpolation*. The method is described in Algorithm 7.2. Such a method generates interpolation matrices with fixed bandwidth on average (under the natural order-

Algorithm 7.2 HHRBF: heterogeneous hierarchical interpolation with CSRBFs

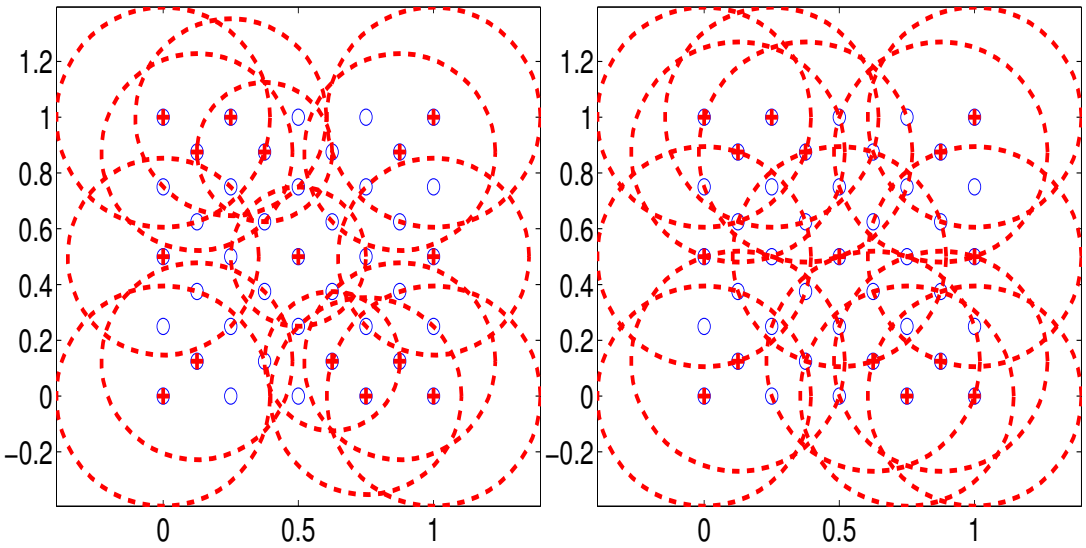
1: Input $\underline{\mathrm{f}}$ and $\mathcal{X}_1, \ldots, \mathcal{X}_\ell$.
2: $\underline{\mathrm{r}}_0 = \underline{\mathrm{f}}$.
3: **for** $i = 1$ to ℓ **do**
4: Check the diagonal dominance condition for $\mathcal{X}_i$ according to Algorithm 5.3.
5: Generate the interpolation matrix $A_{\mathcal{X}_i}$ according to Program 5.4.
6: Find an interpolant $s_i(\mathbf{x})$ by solving $A_{\mathcal{X}_i}\underline{\alpha}_i = \underline{\mathrm{r}}_{i-1}|_{\mathcal{X}_i}$.
7: Save the support information and the solution $\underline{\alpha}_i$.
8: **end for**

level	N	Using CSRBFs with different shapes						M1
		n_j			R_j			R_ℓ
		max	min	mean	max	min	mean	
1	9	8	6	6.7	1.1180	0.7071	1.0199	1.1180
2	13	8	6	6.8	0.7906	0.5000	0.7169	0.7906
3	25	8	7	7.5	0.5590	0.3536	0.4567	0.5590
4	41	8	6	7.2	0.3953	0.2500	0.3046	0.3953
5	81	8	7	7.9	0.2795	0.1768	0.2072	0.2795
6	145	8	6	7.6	0.1976	0.1250	0.1384	0.1976
7	289	8	7	8.0	0.1398	0.0884	0.0967	0.1398
8	545	8	6	7.8	0.0988	0.0625	0.0658	0.0988
9	1089	8	7	8.0	0.0699	0.0442	0.0464	0.0699

Table 7.1: Comparison of scaling information between stationary and adaptive approach

ing), when the points are uniformly distributed. This is similar to the multilevel stationary interpolation which covers a fixed stencil at each level for uniformly distributed data. On regular grids, the difference between the two approaches takes place at corners and along boundaries. Algorithm 7.2 results in basis functions with different supports: the basis functions at corners have the largest support, followed by those basis functions along boundaries, the basis functions in the middle have the smallest support. This procedure pays more attention to corners and boundaries adaptively, where large error usually occurs. See Figure 7.2 for illustrations. Such a difference at corners and along boundaries results in a significant disparity in convergence: the stationary multilevel interpolation does not converge, while the heterogeneous hierarchical approach converges. Such a disparity is observed on all of the 12 representative test functions in Table 7.2. See Figure 7.3 to Figure 7.5 for comparisons.

These test functions are interpolated on a 33×33 uniform mesh on the unit



(a) Using CSRBFs with different shapes (b) Radius for stationary interpolation

Figure 7.2: Comparisons of the radii of CSRBFs in two multilevel approaches.

square $[0,1] \times [0,1]$. The data set is partitioned into 9 levels. The nested data sets are constructed according to Algorithm 7.3. The errors between the interpolants, $\sum_{k=1}^{\ell} s_k(\mathbf{x})$, $\ell = 1, 2, \ldots, 9$, and each test function are evaluated on a 35×35 fine mesh. Two kinds of errors are recorded: the maximum error, $\|\cdot\|_\infty$, and the *root mean square error*, $\frac{\|\cdot\|_2}{\sqrt{N}}$, where N is the number of evaluation points.

Table 7.1 compares the radii information of basis functions at each level. For the heterogeneous hierarchical approach, we list the maximum, minimum and average numbers of neighbouring points, n_j, and corresponding radius, R_j, for each basis function. In the stationary approach, all the basis functions share a fixed radius R_ℓ at level ℓ. The difference of the supports on the same data set is illustrated in Figure 7.2. Figures 7.3 to Figure 7.5 compare the convergence history of the maximum error (maxerr) and the root mean square error (RMS) against the *fill distance*, h_ℓ. The fill distance is defined by

$$h_\ell := h_{\mathcal{X}_\ell,\Omega} = \sup_{\mathbf{x}\in\Omega} \min_{\mathbf{x}_i\in\mathcal{X}_\ell} \|\mathbf{x} - \mathbf{x}_j\|, \text{ where } \mathcal{X}_\ell \subset \Omega. \tag{7.2}$$

It is observed that the multilevel stationary interpolation approach provides only limited convergence [70, p.277]. In [70], only the root-mean-square error is reported using at most 7 levels. Here, we report both the maximum error and the root-mean-

Table 7.2: A list of test functions, where the R column stands for reference number.

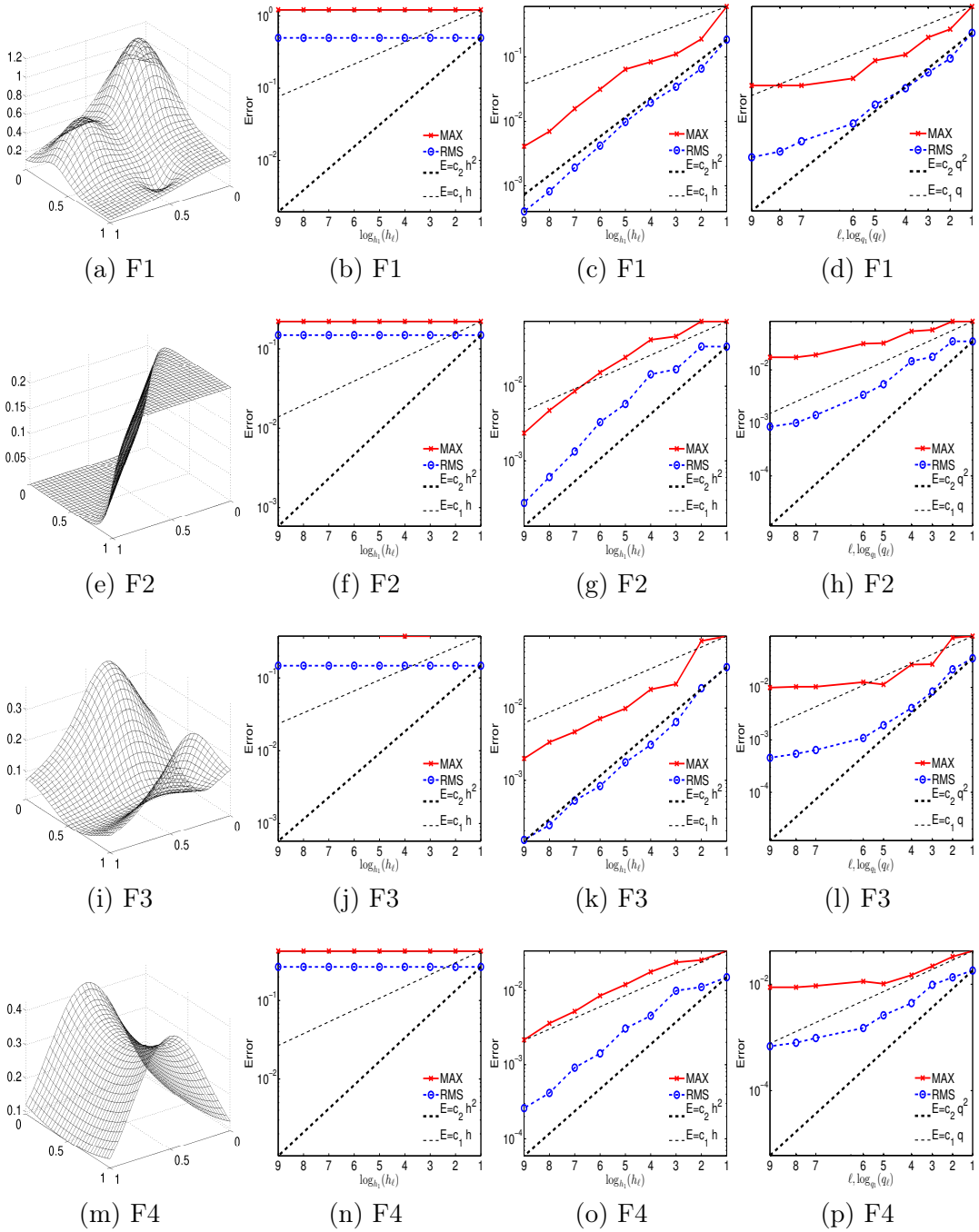
Name	R	formula
F1	[91]	`franke` function in Matlab
F2	[91]	$\tanh(9y - 9x) + 1/9$
F3	[91]	$(1.25 + \cos(5.4y))/(6(1 + (3x - 1)^2)$
F4	[91]	$\exp(\frac{81}{16}(x - 0.5)^2 + (y - 0.5)^2)/3$
F5	[91]	$\exp(\frac{81}{4}(x - 0.5)^2 + (y - 0.5)^2)/3$
F6		$\sqrt{64 - 81(x - 0.5)^2 + (y - 0.5)^2}$
F7	[169]	$1/\sqrt{1 + 2\exp(-3\sqrt{x^2 + y^2}) - 6.7}$
F8	[169]	$50\exp(-200(x - .3)^2 + (y - .3)^2) + \exp(-50((x - .5)^2 + (y - .5)^2))$
F9	[139]	$\tanh(-3(0.595576(y + 3.79761)^2 - x - 10)) + 1$
F10	[139]	$(1 - \frac{x}{2})^6(1 - \frac{y}{2})^6 + 1000(1 - x)^3x^3(1 - y)^3y^3 + y^6(1 - \frac{x}{2})^6 + x^6(1 - \frac{y}{2})^6$
F11	[169]	$\begin{cases} 1 & \text{if } y - \xi \geq 1/2, \text{ where } \xi = 2.1x - 0.1, \\ 2(y - \xi) & \text{if } 0 \leq y - \xi \leq 1/2, \\ \frac{\cos(4\pi\sqrt{(\xi - 1.5)^2 + (y - .5)}) + 1}{2} & \text{if } \sqrt{(\xi - 1.5)^2 + (y - .5)} \leq 1/4, \\ 0 & \text{otherwise.} \end{cases}$
F12		$\sin(2\pi x)\sin(\pi y)$

square errors. It turns out that the multilevel stationary interpolation approach does not converge in the infinity norm, whilst it provides limited convergence in the L_2 norm. In contrast, superlinear convergence in the infinity norm and quadratic convergence in the L_2 norm are observed for the heterogeneous hierarchical approach on hierarchical Cartesian sets generated according to Program 7.4.

The heterogeneous hierarchical approach converges on regular grids. It deserves further investigation on scattered data sets. Further investigation requires efficient algorithms to construct hierarchical scattered data sets.

7.3 Hierarchical data sets construction

Multi-step methods receive a lot of attentions recently. Hierarchical scattered sets are necessary for multi-step methods. Floater and Iske [79, 80] use *thinning algorithms*, which recursively remove points in $\mathcal{X}$ according to some criterion based on Delaunay triangulations. In general, the implementation of the thinning algorithm requires $\mathcal{O}(N^3)$ steps; planar points requires $\mathcal{O}(N^2)$ steps. The previously best achievable complexity is $\mathcal{O}(N \log N)$ according to [80, p.706]. Here we construct an $\mathcal{O}(N)$ algorithm to construct hierarchical scattered data. The method is implemented in $\mathbb{R}^2$. Generalization to $\mathbb{R}^d$ is possible but needs more tedious work.



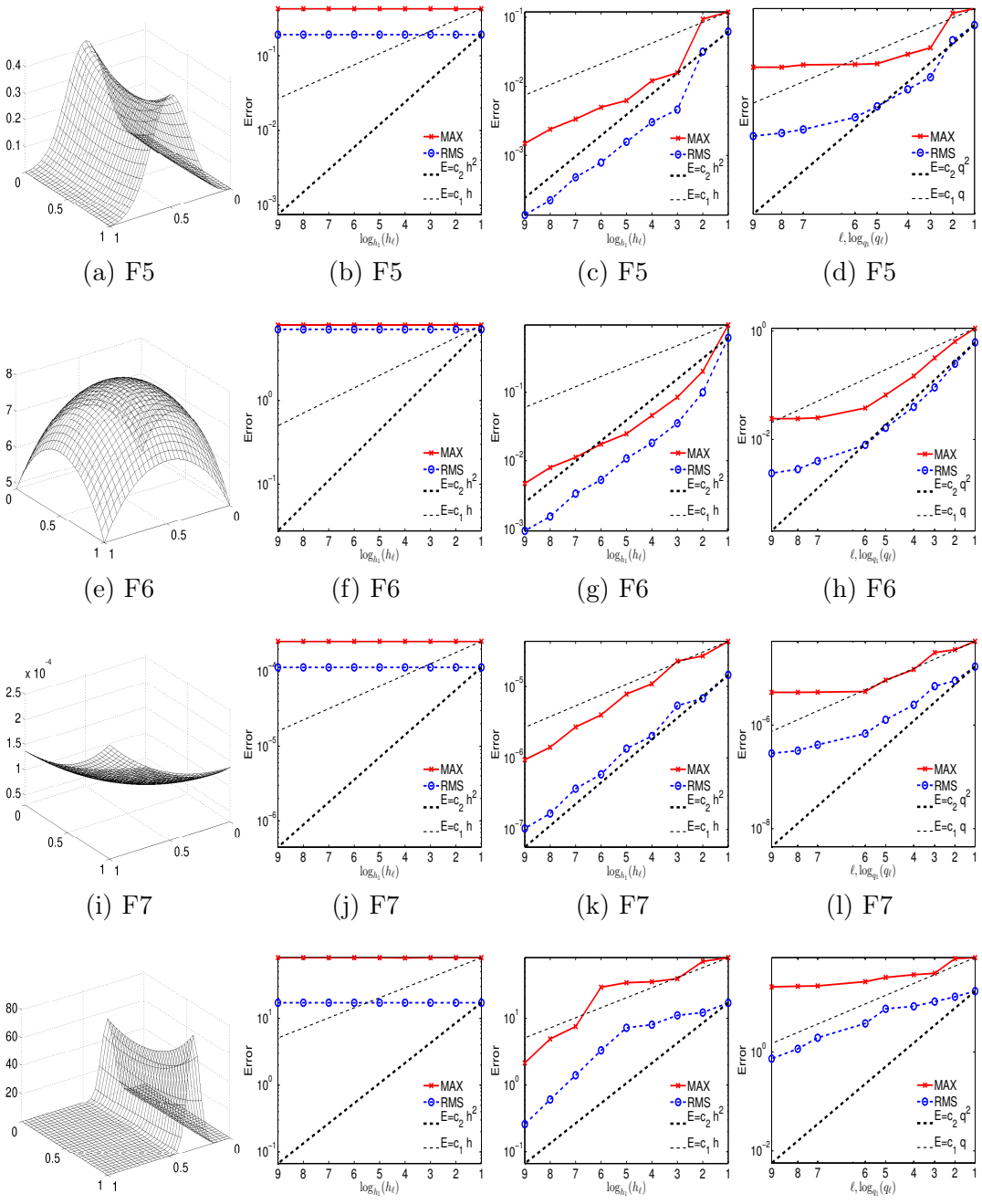
(a) F1 (b) F1 (c) F1 (d) F1

(e) F2 (f) F2 (g) F2 (h) F2

(i) F3 (j) F3 (k) F3 (l) F3

(m) F4 (n) F4 (o) F4 (p) F4

Figure 7.3: Convergence results of multilevel methods. The second column corresponds to the multilevel stationary method on 33×33 Cartesian grids. The third column corresponds to the heterogeneous hierarchical approach in Algorithm 7.2 on 33×33 Cartesian grids. The fourth column corresponds to Algorithm 7.2 on scattered data. The x-axis in the last column is the separation distance rather than the fill distance. The scattered data set can be regenerated in Matlab (statistics tool box) using `p=haltonset(2,'Skip',1e2,'Leap',10); p=scramble(p,'RR2'); center=net(p,1089);`.



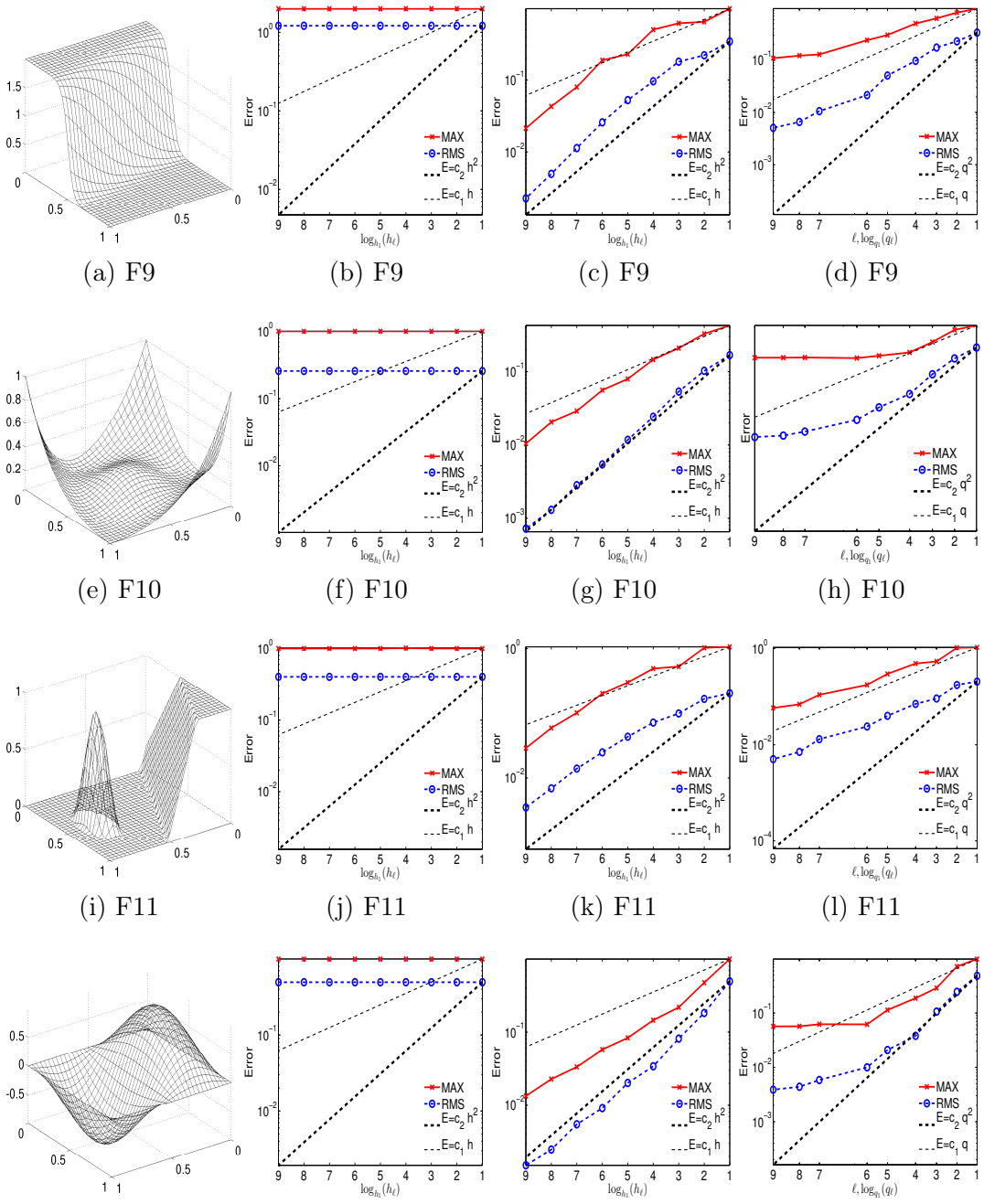
(a) F5 (b) F5 (c) F5 (d) F5

(e) F6 (f) F6 (g) F6 (h) F6

(i) F7 (j) F7 (k) F7 (l) F7

(m) F8 (n) F8 (o) F8 (p) F8

Figure 7.4: Convergence results of multilevel methods. The second column corresponds to the multilevel stationary method. The third column corresponds to the heterogeneous hierarchical approach in Algorithm 7.2. The fourth column corresponds to Algorithm 7.2 on scattered data. The x-axis in the last column is the separation distance rather than the fill distance. The scattered data set can be regenerated in Matlab (statistics tool box) using `p=haltonset(2,'Skip',1e2,'Leap',10); p=scramble(p,'RR2'); center=net(p,1089);`.



(a) F9 (b) F9 (c) F9 (d) F9

(e) F10 (f) F10 (g) F10 (h) F10

(i) F11 (j) F11 (k) F11 (l) F11

(m) F12 (n) F12 (o) F12 (p) F12

Figure 7.5: Convergence results of multilevel methods. The second column corresponds to the multilevel stationary method. The third column corresponds to the heterogeneous hierarchical approach in Algorithm 7.2. The fourth column corresponds to Algorithm 7.2 on scattered data. The x-axis in the last column is the separation distance rather than the fill distance. The scattered data set can be regenerated in Matlab (statistics tool box) using `p=haltonset(2,'Skip',1e2,'Leap',10); p=scramble(p,'RR2'); center=net(p,1089);`.

Algorithm 7.3 An algorithm for constructing hierarchical Cartesian grids

1: Given an $N \times N$ mesh gird, where $N = 2^k + 1$.
2: Start from 3×3 coarsest grid points as $\mathcal{X}_1$.
3: **for** $j = 2$ to k **do**
4: $\quad \mathcal{X}_{2(j-1)} = \mathcal{X}_{2(j-1)-1} \cup$ {face centres of $\mathcal{X}_{2(j-1)-1}$}.
5: $\quad \mathcal{X}_{2j-1} = \mathcal{X}_{2(j-1)} \cup$ {edge centres of $\mathcal{X}_{2(j-1)-1}$}.
6: **end for**

We propose a method based on grid mapping: mapping scattered points to grid points. Firstly, we construct hierarchical Cartesian grids. One possible algorithm for constructing hierarchical Cartesian data sets is described in Algorithm 7.3. The algorithm works for data sets in $\mathbb{R}^2$. In $\mathbb{R}^3$, cell centres are needed. For simplicity, we restrict our discussion to $\mathbb{R}^2$. Numerical computing for face centres and edge centres in Algorithms 7.3 can be avoided by exploring features of Cartesian grids. The algorithm can be done only by index mapping. Such an index mapping procedure is illustrated in Matlab Program 7.4. The Matlab Program only works for the special case on an $m \times m$ mesh with $m = (2^k + 1)^2$ for some positive integer k. This special case serves as the starting point for constructing hierarchical scattered data sets.

Hierarchical scattered data sets are obtained by mapping the scattered data set $\mathcal{X}$ to an $m \times m$ Cartesian mesh $\mathcal{Y}$, where $m = 2^k + 1$ for some k, i.e. find the nearest point in $\mathcal{Y}$ to a given point each in $\mathcal{X}$. Those points in $\mathcal{X}$ corresponding to $\mathcal{Y}_i \subset \mathcal{Y}$ go to the set $\mathcal{X}_i$. Those points in $\mathcal{X}$ but not mapped to any point in $\mathcal{Y}$ go to the finest data set in $\mathcal{X}$.

The mapping procedure can be done in parallel using the following formula

$$\texttt{idxreal} = (\mathbf{x} - \mathbf{y}_0 - \texttt{Stepsize})./\texttt{Stepsize}, \tag{7.3}$$

where $\mathbf{y}_0 = (\min(y_1), \min(y_2))$ for $\mathbf{y} = (y_1, y_2) \in \mathcal{Y}$, `Stepsize` is the step length in the Cartesian grid $\mathcal{Y}$, and `./` is the element-wise division as in Matlab. Rounding `idxreal` to the nearest integer array `idxint`

$$\texttt{idxreal} = \texttt{idxint} + \texttt{idxrem} \tag{7.4}$$

will map the j-th point in $\mathcal{X}$ to a grid point `(idxint(j,1),idxint(j,2))`. If

Program 7.4 `Hcgrid`: a Matlab program to construct hierarchical Cartesian grids

```
function [pidx]=Hcgrid(N)
if rem(log2(N-1),1)~=0
    error('N should be 2^k+1');
end
level=floor(log2(N));
G=reshape(1:N^2,N,N); % index of the grid.
pidx=cell(2*level-1,1);% cell array 4 the indices of points
p1=1:2^(level-1):N;Gp=G(p1,p1);
pidx{1}=Gp(:);          % indices of points in the first level
for k=2:level
    p=1:2^(level-k):N;% Gp(:) indices of
    pp=p(2:2:end);      % face points in X_{2(k-1)-1}
    Gp=G(pp,pp);        % matching face points
    pidx{2*(k-1)}=union(pidx{2*(k-1)-1},Gp(:)); % X_{2(k-1)}
    Gp=G(p,p);          % refine mesh as X_{2k-1}
    pidx{2*k-1}=Gp(:);% refine grid
end
```

multiple $\mathbf{x}_j$ are mapped to the same grid point, choose the nearest one by comparing the `idxrem` part. The other point(s) go(es) to the finest level of the data set. This step is similar to the thinning algorithm: first thin the data set from N points to m^2 points. Computing the array `idxreal` is highly scalable and can be done in one step by scaling vectors. Making the mapping one-to-one requires at most N logical and numerical comparisons. The procedure is described in Algorithm 7.5.

Theorem 7.1. *The complexity of the grid mapping algorithm in Algorithm 7.5 is* $\mathcal{O}(N)$.

One implementation of Algorithm 7.5 is Program 7.6, which takes advantage of the existing `knnsearch` algorithm in Matlab. The complexity of the algorithm is slightly higher, $\mathcal{O}(N \log(m^2))$, whilst in practice the linear scalability of the Matlab Program 7.6 is observed for scattered data sets; see Figure 7.7. With hierarchical scattered data sets available, we can investigate multilevel interpolation on scattered data sets.

Algorithm 7.5 Constructing hierarchical scattered data sets by grid mapping

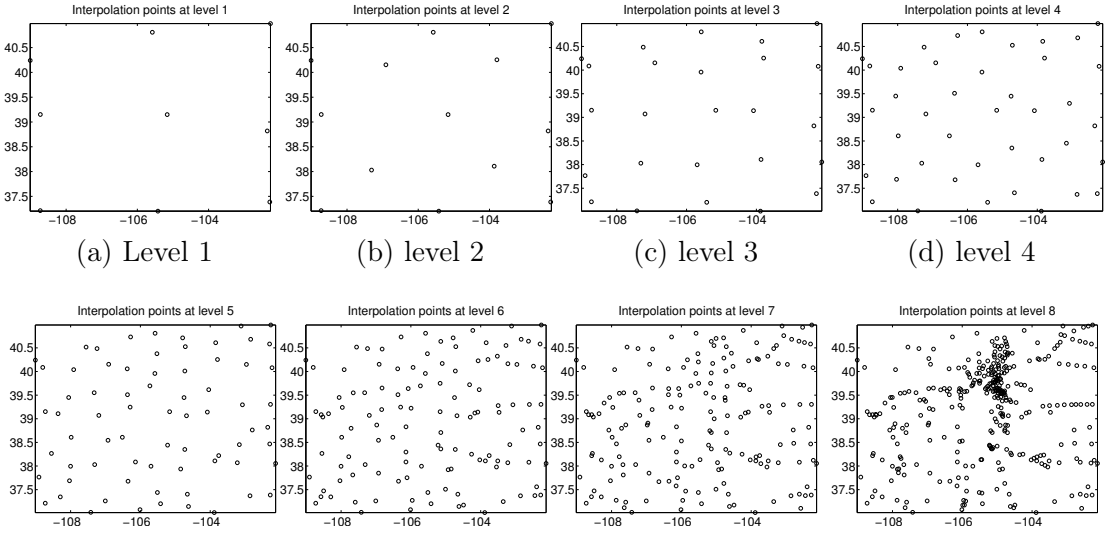
Given N scattered points $\mathcal{X}$, calculate $m = 2^{\lfloor \log_2 \sqrt{N} \rfloor} + 1$ // suppose $\mathbf{x} \in \mathbb{R}^2$.
Compute the left-bottom corner $\mathbf{x}_0$ and the up-right corner $\mathbf{x}_{N+1}$, `Stepsize` $= (\mathbf{x}_{N+1} - \mathbf{x}_0)./(m-1)$
Compute `idxreal` $= (\mathbf{x} - \mathbf{x}_0 -$ `Stepsize`$)./$ `Stepsize`
Compute `idxint` and `idxrem`.
Construct an empty $m \times m$ matrix , `G` to store the index of $\mathcal{X}$
for $j = 1$ to N **do**
    if `G(idxint(j,1), idxint(j,2))` is empty **then**
        `G(idxint(j,1),idxint(j,2))`$=j$
    else if
        then suppose `G(idxint(j,1),idxint(i,2))=p`.
        if `norm(idxrem(j,:),2)` $<$ `norm(idxrem(p,:), 2)` **then**
            `G(idxint(j,1),idxint(j,2))`$=j$.
        end if
    end if
end for
Apply Program 7.4, `pidx=hcgrid(m)`;
for `k=1` to `length(pidx)` **do**
    pidx{k}=G(pidx{k})
end for
if $N > m^2$ **then**
    `pidxk+1=[1:N]'`.
end if

7.4 Heterogeneous hierarchical interpolation on scattered data set

We interpolate the test functions on 1089 Halton points. These points are quasi-uniform, i.e. there exists a constant $c > 0$ such that

$$q_{\mathcal{X}} \le h_{\mathcal{X},\Omega} \le c q_{\mathcal{X}}, \text{ where } q_{\mathcal{X}} := \frac{1}{2} \min_{i \ne j} \|\mathbf{x}_i - \mathbf{x}_j\|_2. \tag{7.5}$$

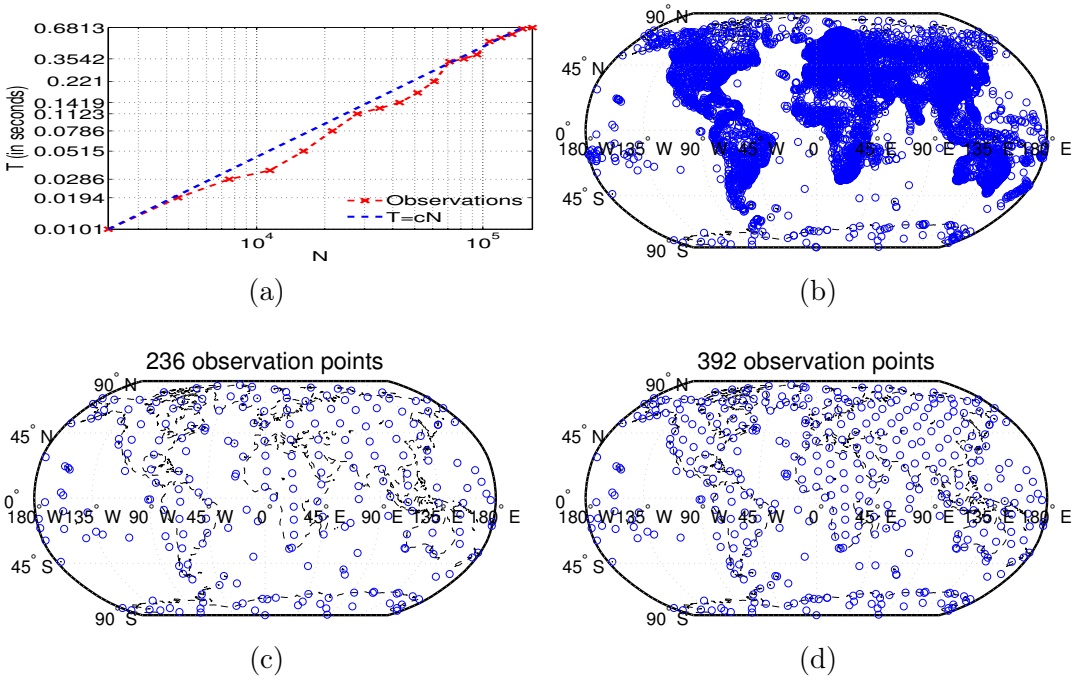
Since the fill distance is not easy to compute for scattered data sets, we use $q_{\mathcal{X}_i}$ instead of $h_{\mathcal{X}_i,\Omega}$ when we plot the convergence history. The convergence history for heterogeneous hierarchical interpolation is listed in the last column of Figure 7.3 to Figure 7.5. It seems that there might be some saturation for some problems on scattered data. But such seemingly saturation can also caused by the fact that



(a) Level 1 (b) level 2 (c) level 3 (d) level 4

(e) level 5 (f) level 6 (g) level 7 (h) level 8

Figure 7.6: Hierarchical data sets of cities in Colorado, USA.



(a) (b)

(c) (d)

Figure 7.7: Applications and scalability of `hscatter2grid`. (a) Scalability of `hscatter2grid`. Apply the Program 7.6 to scattered Halton points on the unit square with N points, where N range from 47^2 to 417^2. (b) 8564 observation stations belonging to the World Meteorological Organization. (c) and (d) are observation stations in level 7 and level 8 by `hscatter2grid`, where the longitude and latitude are used as coordinates.

Program 7.6 `hscatter2grid`:hierarchical scattered data sets construction by grid mapping

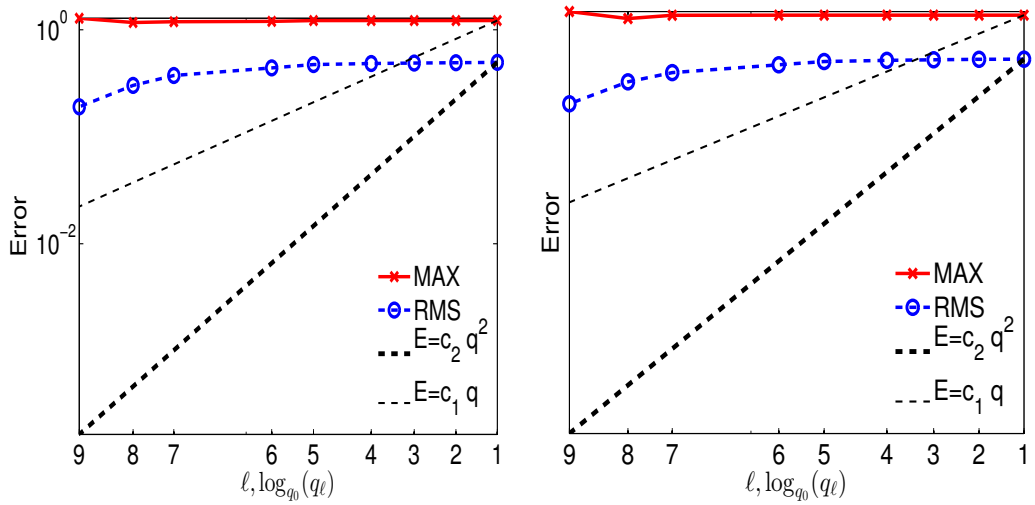
```
function [idxnet,xmin,xmax]=hscatter2grid(center)
n=length(center);
xmin=min(center);
xmax=max(center);
m=floor(sqrt(n));
level=floor(log2(m));
m=2^level+1;
[x,y]=meshgrid(linspace(xmin(1),xmax(1),m),linspace(xmin(2),xmax
    (2),m));
datae=[x(:) y(:)];
idxmap=knnsearch(center,datae,'NSMethod','kdtree','k',1);
pidx=hcgrid(m);
idxnet=pidx;
idxnet{1}=idxmap(pidx{1});
for k=2:length(pidx)
    idxnet{k}=idxmap(pidx{k});
end
idxnet=cellfun(@unique,idxnet,'UniformOutput',false);
if n>m^2
    idxnet{k+1}=[1:n]';
end

```

neither $q_{\mathcal{X}_i}$ nor $h_{\mathcal{X}_i}$ decreases smoothly as the case on regular grids; see the x-axis labels of Figure 7.3 to Figure 7.5. When the ratio $\frac{q_{\mathcal{X}_i}}{q_{\mathcal{X}_{i-1}}}$ is larger than the ratio of the background regular mesh, it leads to a smaller slope in the convergence curves. When the separation distance shrinks while the fill-in distance does not, the error won't be reduced as expected, because all the error estimates are related to the fill distance [208]. This is the possible reason for the poorer convergence history in column 4 of Figures 7.3 to Figure 7.4.

The multilevel stationary interpolation on scattered data does not converge at all. We only show the results for the `franke` function, F1, and F3 in Figure 7.8. Results for other functions are similar.

The difference between the heterogeneous hierarchical approach and the multilevel stationary interpolation is striking and needs further investigation. One reasonable explanation is that using CSRBFs of different shapes increases the va-



(a) F1 (b) F3

Figure 7.8: Multilevel stationary interpolation on scattered data. The scattered data points are generated in Matlab using `p=haltonset(2,'Skip',1e2,'Leap',10); p=scramble(p,'RR2'); center=net(p,1089);`

rieties of frequencies. There is less correlation between the basis functions in the middle of the underlying region and at points near the boundaries.

One particular example is the test function F8 in Figure 7.4, which has a narrow peak in the middle; in the first 5 levels, the support of each basis function is larger, and the method converges slowly. When the support becomes smaller than some threshold, it converges faster.

Multilevel interpolation with globally supported RBFs makes little sense, because it is difficult for globally supported radial basis functions to interpolate functions of high frequency. However we can still adjust the shape parameter and obtain a better approximation.

7.5 Adaptive shape and smoothing parameter selection

Rippa [169] uses the *leave-one-out cross validation* (LOOCV) to select a good shape parameter for radial basis functions. The idea of LOOCV is to minimize a *cost function* or a risk function. The cost function is a sum of residuals defined as follows: for each interpolation centre $\mathbf{x}_k$, define the set $\mathcal{X}^{[-k]} := \mathcal{X} \backslash \{\mathbf{x}_k\}$, and

find an interpolant $s^{[-k,\varepsilon]}(\mathbf{x})$ such that $s^{[-k,\varepsilon]}(\mathcal{X}^{[-k]}) = \underline{\mathrm{f}}|_{\mathcal{X}^{[-k]}}$, where ε is the shape parameter in the basis functions. For each interpolant $s^{[-k,\varepsilon]}$, we use f_k to validate the approximation:

$$\ell_k(\varepsilon) = f_k - s^{[-k,\varepsilon]}(\mathbf{x}_k). \tag{7.6}$$

Let $L(\varepsilon) = (\ell_1(\varepsilon), \ldots, \ell_N(\varepsilon))$, and define a certain norm of $L(\varepsilon)$ as the cost function. The best shape parameter is obtained by minimizing the cost function

$$\varepsilon^* = \arg\min_{\varepsilon} \|L(\varepsilon)\|_*. \tag{7.7}$$

Rippa uses the 1-norm in (7.7).

The element $\ell_k(\varepsilon)$ is given by the explicit formula

$$e_k(\varepsilon) = \frac{\alpha_k}{A^{-1}_{kk}}. \tag{7.8}$$

where $A\underline{\alpha} = \underline{\mathrm{f}}$ is the underlying interpolation system on $\mathcal{X}$ and A^{-1}_{kk} is the k-th diagonal element of A^{-1}. The formula in (7.8) is the most interesting part in LOOCV: it avoids solving N linear systems of order $(N-1)\times(N-1)$. Such a method is convenient for choosing a best shape parameter for problems of moderate size. One possible issue that should be taken into account is that the global minimum of the cost function might be missed by using some linear search procedure, see Figure 7.9 for illustration. This leads to a difficult global optimization problem, where radial basis functions can also serve as a useful tool. This method goes beyond our discussion.

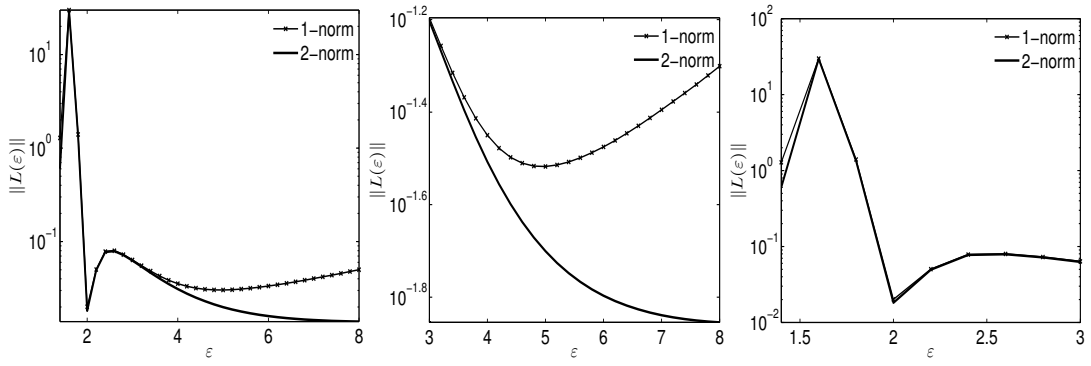
The same idea can be applied to choosing the smoothing parameter, μ, in the following linear system

$$(A + \mu I)\underline{\alpha} = \underline{\mathrm{f}}. \tag{7.9}$$

However when the problem size is large, evaluating the associated cost function is computationally intensive.

7.6 Discussion

The multilevel method was introduced by Floater and Iske [79]. The method has been further investigated in [71, 80, 97, 115]. Such methods and multiscale analysis have received much attention recently with the advance of error analysis [69, 128,



(a) (b) (c)

Figure 7.9: Cost functions for choosing best shape parameters. Interpolate the `franke` function on a 17×17 regular mesh on the unit square $[0,1]\times[0,1]^2$ with multiquadrics. The global minimum of the cost function $\|L(\varepsilon)\|_1$ is near 2. Using the linear search algorithm to find the minimum between $[1,8]$ may miss the global minimum, for example, the `fminbnd` function in Matlab finds a local minimum for the $\|L(\varepsilon)\|_1$ near 5 as shown in (b). The `fminbnd` can only find the global minimum when the initial search range containing the global minimum is narrow enough. For this problem, one can apply the `fminbnd` to several intervals, or use other global optimization techniques.

193, 209]. These convergence results are obtained by adjusting the scaling parameter and the bandwidth; these schemes usually result in sparse matrices with a wide bandwidth. In contrast, stationary interpolation with positive definite CSRBFs does not converge according to the Strang-Fix condition and this condition should have received more attention. The striking part of this chapter is that convergence is observed for heterogeneous hierarchical interpolation. Theoretical analysis of such a scheme needs to be investigated in future. It appears that the convergence rate observed is linear or superlinear in the maximum norm and quadratic in the L_2 norm. The method is not recommended for applications which require high resolution, whereas we shall see that there are a wide range of applications which do not require high resolution, and for which this method is applicable and very convenient.

The $\mathcal{O}(N)$ algorithms introduced here for constructing hierarchical scattered data are a considerable advance compared with the $\mathcal{O}(N\log N)$ thinning algorithms. The algorithm and implementation are restricted to special cases in $\mathbb{R}^2$ at the moment, but we are sure that it can be generalised to higher dimensional space.

8. Sparse kernel summation on Cartesian grids

In this chapter, we propose a fast algorithm for sparse kernel summations of the form $s(\mathbf{x}) = \sum_{i=1}^{M} \alpha_i \phi(\varepsilon_i \|\mathbf{x} - \mathbf{x}_i\|)$ on N points on arbitrary fine Cartesian grids in $\mathbb{R}^d$. The algorithm takes the advantages of the sparsity of ϕ and the feature of Cartesian grids. It is a scalable algorithm with a complexity of $\mathcal{O}(N)$. Efficiency can also be improved by looping over M bases instead of N points. The algorithm is used for post-processing for 3D object modelling with compactly supported radial basis functions. Numerical examples for 3D surface construction are presented to illustrated the efficiency of the algorithm.

8.1 Motivation

This chapter examines post processing of scattered data approximation. Post processing is usually needed for visualization. Post processing usually takes longer time than solving the underlying interpolation systems for 3D visualization because the number of evaluation of the summation of basis functions is usually far greater than the number of basis functions . As shown in Chapter 6, for our cases, the sparse linear system with M basis functions can be solved in $\mathcal{O}(M)$ time, where M is the number of interpolation points. Post processing needs to evaluate the sum of M basis functions on N points. Usually, $N \gg M$. In this chapter we propose an efficient method which can reduce the time for post processing.

Consider a summation of sparse kernels in the form

$$s(\mathbf{x}) = \sum_{i=1}^{M} \alpha_i K_i(\mathbf{x}, \mathbf{x}_i), \tag{8.1}$$

where $\underline{\alpha} = (\alpha_1, \alpha_2, \ldots, \alpha_M) \in \mathbb{R}^M$, $\mathbf{x} \in \mathbb{R}^d$ and $\mathbf{x}_i \in \mathbb{R}^d$ is the support centre for function K_i. The function K_i can be an arbitrary compactly supported function. Here, we focus on a compactly supported *radial basis function* with a support radius ρ_i, $K_i = \phi(\varepsilon_i \| \cdot \|)$. We are interested in the following problem: evaluating

the sum, $s(\mathbf{x})$, on arbitrarily fine Cartesian grids with N points (where $N \gg M$). The evaluation results are closely associated to visualizing an implicit surface [223**?**], an eigenmode [96], pattern evolution on a biological body [69] and similar post-processing (usually visualization) of results on Cartesian grids from mesh-free methods based on scattered data [70]. The Cartesian grids are required for some popular visualization functions like `meshgrid`, `patch`, `isosurface` in Matlab.

The sparse kernel summation problem arises frequently and is of increasing significance because of the increasing demand for vivid visualization. Similar problems also arise from various applications involving nonlinear regression, non-parametric density function estimation, and kernel based machine learning, see [130] for example. The essence of these problems share the same foundation: fast summation of sparse kernels. The challenges of these problems lie in the fact that because of the high dimensionality, it is easy to make the problem data-intensive and thus memory-intensive, and the sparsity usually results in a low ratio of computation to memory reference volume.

Both of these features contradict the traits of current and future design of chips (according to the current trend) which usually have relatively fast computation rates while low memory bandwidth and thus slow memory reference rate. Memory bandwidth usually limits the performance of such data/memory-intensive applications. In addition, here we allow the support, ρ_i, of each function K_i to vary, which increases the data heterogeneity.

For cases when the kernel functions are global, there are various available fast evaluation methods; for example, there are methods for polyharmonic splines [11, 12, 14, 16], multiquadrics [43], Gaussians [19, 170, 191] and Matérn kernels [40]. These methods are usually based on fast multipole like methods [216] or tree-code [57, 124]. To set up such efficient tree structures usually requires $\mathcal{O}(dN \log N)$ time and $\mathcal{O}(dN)$ storage [208, p240, p245]. In contrast to the global case, little has been mentioned for the compactly supported cases. Here, we contribute an efficient overhead-free algorithm which requires $\mathcal{O}(N)$ time and only $\mathcal{O}(N)$ storage, rather than $\mathcal{O}((d+1)N)$ storage, for fast summation of compactly supported kernel functions. The efficiency of the algorithm is obtained by taking advantage of sparsity, by removing the overhead for setting up a tree structure and by properly managing memory. Compared with recent $\mathcal{O}(N)$ algorithms for *truncated* Gaussian radial

basis functions [218] where numerical results are presented for the 2D `frank` test function which take 106 seconds to interpolate with 50 million data points and 1024 processors on a Blue Gene/L supercomputer, here we report numerical results for a 3D implicit surface which consists of 105,615 centres, and takes only 11 seconds to evaluate the $45,079,265$ points of a $389 \times 385 \times 301$ mesh on a low-end AMD chip without any multithreading techniques.

The remainder of this section is organized as follows. We first propose a simple and efficient data structure and algorithm design, followed by performance analysis of the algorithm and numerical verification. Finally, we give some remarks and discussion.

8.2 Data structure and algorithm design

8.2.1 Data structure

```
struct gridbox{
    uint8   d;
    double  *low_left;   // x_min
    int     *N;          // N=[N_1, ..., N_d]
    double  *steps[d];   // stepsize of each direction
}
```

Figure 8.1: A gridbox data structure to store $N = N_1 \times \cdots \times N_d$ points with only $\mathcal{O}(N_1 + \cdots + N_d)$ storage or even $\mathcal{O}(1)$ storage if the step size in each direction is a constant.

Since Cartesian grids like those generated by `meshgrid` or `ndgrid` in Matlab are usually stored in multidimensional arrays, a d-dimensional array is usually used to store the evaluations of $s(\mathbf{x})$ on these grid points. To store the Cartesian grids, `meshgrid` or `ndgrid` in Matlab return d d-dimensional arrays. Such a multidimensional array data structure for regular Cartesian grids is inefficient and unnecessary for large scale high dimensional problems: suppose the number of Cartesian points is $N = N_1 \times \cdots \times N_d$, the d d-dimensional arrays will require $d \times N_1 \times \cdots \times N_d$ storage. The storage will soar as any N_i $(i = 1, \ldots, d)$ increases even a little; and in each multidimensional array, there are repeated, and therefore redundant values. Here we propose a simple data structure which can immediately reduce the storage for the N Cartesian points from $d \times N_1 \times \cdots \times N_d$ to $N_1 + \cdots + N_d$, or even a

constant in $\mathcal{O}(1)$; for the former case, it allows the step size in each direction to vary, for the latter case, the step size in each direction is a constant. We call the data structure `gridbox`; it is illustrated in Fig 8.1. When the step size is allowed to vary in each direction, the pointer `steps[i]` points to a vector of coordinates of the grids in the i-th direction, otherwise it only stores a constant step size in the that direction. The coordinates can be obtained by cheap computation at runtime rather than costly memory reference. Let us first consider the most memory efficient case when the step size in each direction is a constant.

8.2.2 Range search

The next question is how to find those points in a `gridbox` which are located in the support of a function $K_i = \phi(\varepsilon_i \| \cdot \|)$ with center $\mathbf{x}_i$ and radius ρ_i. Only those points located in the smallest box which contains the ball $\mathcal{B}(\mathbf{x}_i, \rho_i)$ can possibly be in the support. For some cases the box can shrink to a smaller one which covers most of the ball $\mathcal{B}(\mathbf{x}_i, \rho_i)$. The evaluation of the function K_i only takes place on this small box which will be referred to as a `subgridbox`. A `subgridbox` can be stored more economically than a `gridbox`, since only the coordinates of its lower-left and upper-right (think it as 2D) corners need to be stored. The relative coordinates in the `gridbox` of the lower-left corner and the upper-right corner can be computed as follows

$$\texttt{MyDownLeft} = \lceil \mathbf{x}_i - \mathbf{x}_0 - \rho_i \rceil ./ \texttt{steps} \text{ and } \texttt{MyUpRight} = \lfloor \mathbf{x}_i - \mathbf{x}_0 + \rho_i \rfloor ./ \texttt{steps}, \quad (8.2)$$

where $\mathbf{x}_0$ is the coordinate of the `low_left` corner in the `gridbox`, `steps`$\in \mathbb{R}^d$ is the step size in each direction and ./ is the element-wise division as used in Matlab; $\lceil \cdot \rceil$ and $\lfloor \cdot \rfloor$ are the ceiling and floor function respectively.

Special care has to be paid to any `subgridbox` near the boundary of the `gridbox`. Such cases can result in relative coordinates in (8.2) less than 0 in `MyDownLeft` or greater than $N_i - 1$ ($i = 1, \ldots, d$) in `MyUpRight`. In these cases, only the intersection of the `subgridbox` and the `gridbox` is used, the `subgridbox` is shrunk by adjusting `MyDownLeft` and `MyUpRight` as follows

$$\begin{aligned} \texttt{MyDownLeft} &= \max\{(0, \ldots, 0), \lceil \mathbf{x}_i - \mathbf{x}_0 - \rho_i \rceil ./ \texttt{Steps}\}, & (8.3) \\ \texttt{MyUpRight} &= \min\{(N_1 - 1, \ldots, N_d - 1), \lfloor \mathbf{x}_i - \mathbf{x}_0 + \rho_i \rfloor ./ \texttt{Steps}\}. & (8.4) \end{aligned}$$

Here we keep `MyDownLeft` and `MyUpRight` separately instead of saving them in a special `subgridbox` structure; this can increase the possibility of taking advantage of the register. In this small `subgridbox`, there are a small number of points near the corners which might be outside the support of current basis function. Such points are excluded at runtime as shown in Algorithm 8.1.

8.2.3 Algorithm

We consider the most common case, when $d = 3$. Generalization for other dimensions should be straightforward. First we consider how to evaluate the results of one compactly supported function $\alpha_i K_i(\mathbf{x}, \mathbf{x}_i)$ in its corresponding `subgridbox`; and then the same procedure can be applied to the other basis functions. The storage of the evaluation results is closely related to the implementation. The results are stored in an array `fval`. Since only those points near the surface in the `gridbox` are active points which can be selected during the process, only part of `fval` will be referred to during the evaluation. First, all the points in `fval` are stored in a special non-numerical format, for example, here, they are stored as `NaN`s (saving them as `NULL`s is not recommended, because `NULL` may display as 0 in some environments like Matlab, and an implicit surface usually is an `isosurface` with isovalue 0). When a point is selected as an active point for the first time, the evaluation result replaces the special format (`NaN` here) as shown in line 15 in Algorithm 8.1, otherwise the result is accumulated as in line 17 in Algorithm 8.1. Another issue needs to be mentioned in Algorithm 8.1: how the 3 dimensional $N = N_x \times N_y \times N_z$ grid maps to a one dimensional array. Here we suppose that `fval` is a 3D array in Matlab, it has N_z pages of 2 dimensional $N_x \times N_y$ arrays. If `fval` is viewed as a one dimensional array indexed from 0 to $N - 1$, then the index transform is described as follows.

$$\texttt{fval}(i, j, k) = \texttt{fval}(k * N_x * N_y + i * N_y + j). \tag{8.5}$$

Once the evaluation of one compactly supported function $\alpha_i K_i(\mathbf{x}, \mathbf{x}_i)$ is clear, a loop of Algorithm 8.1 makes the whole of Algorithm 8.2. In both pseudo-algorithms, C-like syntax and comments are used.

Algorithm 8.1 `subbox_eval(fval,`$\mathbf{x}_i$`,`ρ_i`,` α_i `, gridbox)`

```
Initialization
Compute MyLowLeft and MyUpRight
for (z_idx=MyLowLeft[2]; z_idx <= MyUpRight[2];++z_idx) do
   z=low_left[2]+z_idx*stepsize[2]; // This is current z coordinate
   idxz0=z_idx*Nx*Ny;  //current slice
   for (x_idx=MyLowLeft[0]; x_idx<=MyUpRight[0];++x_idx)  do
      x=low_left[0]+x_idx*stepsize[0]; // This is current x coordinate
      idxx0=x_idx*Ny;
      for (y_idx=MyLowLeft[1]; y_idx<=MyUpRight[1];++y_idx)  do
         y=low_left[1]+y_idx*stepsize[1]; // current y coordinate
         Dist2Center =‖x − x_i‖_2/ρ_i; // x = (x, y, z), x_i is the center in φ_i(ε_i‖x − x_i‖_2)
         if (Dist2Center <1.0) then
            f_idx=idxz0+idxx0+y_idx; // x ∈ B(x_i, ρ_i), compute index for fval
            if (isnan(fval[f_idx])) then
                fval[f_idx] = α_i K(x, x_i); //if first accumulation
            else
               fval[f_idx] +=α_i K(x, x_i); //if already accumulated
            end if
         end if
      end for
   end for
end for
```

Algorithm 8.2 `gridbox_eval(fval, centers,radials,weights,gridbox)`

```
for (i=0;i<M;++i) do
     subbox_eval(fval,centers[i],radials[i],weights[i],gridbox);
end for
```

8.2.4 For special $K_i(\mathbf{x}, \mathbf{x}_i)$ with tiny support

For some surfaces with fine features, the supports of $K_i(\mathbf{x}, \mathbf{x}_i)$ may vary significantly; the ratio of the largest support to the smallest support can be on the order of hundred, see Table 5.1 and Table 5.2. When such cases occur, to reduce the number of evaluation points, one may choose a `gridbox` whose `stepsize` is greater than the smallest radius of $K_i(\mathbf{x}, \mathbf{x}_i)$, $(i = 1, \ldots, M)$. In this case the support of these basis functions may located in a small cell of the grid, and thus these basis functions $K_i(\mathbf{x}, \mathbf{x}_i)$ should be treated specially. First, choose those points $\mathbf{x}_i$ *on* the

Algorithm 8.3 `modifysubbox_eval(fval, x_i, gridbox)`

```
Compute MyDownLeft and MyUpRight and modify them if necessary
for (z_idx=MyLowLeft[2]; z_idx <= MyUpRight[2];++z_idx) do
    idxz0=z_idx*Nx*Ny;  //current slice
    for (x_idx=MyLowLeft[0]; x_idx<=MyUpRight[0];++x_idx) do
        idxx0=x_idx*Ny;
        for (y_idx=MyLowLeft[1]; y_idx<=MyUpRight[1];++y_idx) do
            f_idx=idxz0+idxx0+y_idx;
            if (isnan(fval[f_idx])) then
                fval[f_idx] = 0; // only modify points related to x_i on surface
            end if
        end for
    end for
end for
```

Algorithm 8.4 `modify_eval`(centers,radials,gridbox)

```
for (i=0;i<P;++i) do
    if 2*radials[i] < min stepsizes then
        modifysubbox_eval(fval, x_i, gridbox) // Note i < P not M
    end if
end for
```

surface with radius ρ_i smaller than the `stepsize`, the ball $\mathcal{B}(\mathbf{x}_i, \rho_i)$ can be either in a cell of the `gridbox` or it could be in between two to eight cells (more in higher dimensions). In the former case, the coordinates `MyDownLeft` and `MyUpRight` defined in (8.2) should be swapped. In the latter cases if `MyDownLeft[i]` > `MyUpRight[i]`, they should be swapped (otherwise they would be equal).

8.3 Performance analysis and numerical verification

8.3.1 Performance analysis

First we consider how many points there are in the `subgridbox` corresponding to $K_i(\mathbf{x}, \mathbf{x}_i)$. Here we suppose that the height, length and width of the `gridbox` are H, L and W respectively. For simplicity, the step size in each direction of a `gridbox` are assumed to be h, and the radius of support is assumed to be ρ. Thus a `subgridbox` can consists of at most $\lfloor \frac{2\rho}{h} \rfloor^3$ active grids. Since $\lfloor \frac{2\rho}{h} \rfloor \leq \frac{W}{h} \frac{2\rho}{W}$ and

$\frac{LWH}{h^3} = N$, then

$$\left\lfloor \frac{2\rho}{h} \right\rfloor^3 \leq N \frac{8\rho^3}{LWH}. \tag{8.6}$$

When the radii of the supports of $K_i(\mathbf{x}, \mathbf{x}_i)$ are equal to ρ, $\frac{8\rho^3 M}{LWH}$ is a constant. Thus Algorithm 8.2 is an $\mathcal{O}(N)$ algorithm. In particular, when $\frac{8\rho^3 M}{LWH} \leq 1$, Algorithm 8.2 can be completed with at most N evaluations. In the case when ρ_i may vary, $\sum_{i=1}^{M} \frac{8\rho_i^3}{LWH} \leq M \frac{8 \max \rho_i^3}{LWH}$. Therefore we have the following result.

Theorem 8.1. *For given M basis functions, the sparse summation algorithm, Algorithm 8.2, is an $\mathcal{O}(N)$ algorithm.*

Remark 8.1. We do not make use of the seemingly more promising format $\mathcal{O}(\lfloor \frac{2\rho}{h} \rfloor^3 M)$, since the step size h is a variable and $N \propto \frac{1}{h^3}$. Since the support of most of the basis functions is a subset of the gridbox with dimension $L \times H \times W$, then $\frac{8 \max \rho_i^3}{LWH} \leq 1$, and Algorithm 8.2 at most takes $N \times M$ evaluations.

Remark 8.2. Looping over M basis functions can save overhead by avoiding unnecessary comparisons and branch statements. If we loop over N points instead of M basis functions, for each point, it requires at least M comparisons (to decide whether the current point locate in the support of each basis functions) and thus it results in at least $N \cdot M$ work.

8.3.2 Numerical verification

Consider the reconstruction of an implicit surface with compactly supported radial basis functions as described in [223]: a Wendland function [208, p.129], $\phi_{3,1}(r) = (1-r)_+^4(4r+1)$ is used as ϕ, where $(\cdot)_+ = \max\{\cdot, 0\}$ is the cut-off function and $r = \|\mathbf{x} - \mathbf{x}_i\|_2$. Various scales for support of the basis functions are chosen according to the procedure described in [223]. Four raw data sets of the Stanford bunny are used[1], namely `Bunny453`, `Bunny1889`, `Bunny8171` and `Bunny35947`, which result in the sum

$$s(\mathbf{x}) = \sum_{i=1}^{M} \phi(\varepsilon_i \|\mathbf{x} - \mathbf{x}_i\|), \tag{8.7}$$

with $M = 1351$, 5643, 24425 and 105615 respectively. Algorithm 8.2 is implemented in C/C++, with a Matlab `mex` gateway function. Numerical experiments are carried

[1] http://graphics.stanford.edu/data/3Dscanrep/

out on a relatively low end AMD Phenom II x4 925 CPU with frequency 2.8 GHz. No multi-threading technique is used. A high resolution timer is used to obtain accurate timing results. Timing results for these four data sets on different grids are presented in the first 6 columns in Table 8.1.

Two formulae are used to model the relationship between N and the timing results, T.

$$\text{Model I: } T = cN^{\beta_1} + \text{noise}, \text{and Model II: } T = cN + \beta_2 + \text{noise}. \tag{8.8}$$

Model I is used to confirm that β_1 is a constant near 1 for a given M. Once this is confirmed, Model II gives further explanation based on the assumption $\beta_1 = 1$. Experimental results and the almost coincident fitted lines from the two models are illustrated in Figure 8.2, which shows that both models fit the experimental data very well for given M. Table 8.2 presents the fitted parameters in both models and their 95% confidence intervals calculated according to [204, p.214]. It turns out that β_2 is a negative number. The ratio of $|\beta_2|$ over the timing results T, and $|\beta_2|/(TM)$ are listed in Table 8.1. One interesting phenomenon is that when this ratio $|\beta_2|/T$ is large, β_1 in model I is large, whereas when this number is scaled with M, $|\beta_2|/(TM)$ decrease as N increases. It seems that β_2 is related to operations outside the loops in Algorithm 8.1 or some overhead due to loading functions. When N is large, the average overhead is small. This is not a definite explanation of this subtle issue, which doesn't change the decisive conclusion: Algorithm 8.2 is a scalable $\mathcal{O}(N)$ algorithm.

8.4 Discussion

Visualizing an implicit surface constructed from scattered data points or visualizing numerical results on scattered data from mesh-free like methods is a problem which frequently occurs. Evaluating such results on suitably fine grids for desirable results usually consumes a lot of time and there is shortage of user-friendly software packages. The algorithms presented here take advantage of sparsity and the features of Cartesian grids. The `gridbox` data structure is essential to reduce the storage of the grids from $N_1 \times N_2 \cdots \times N_d$ to a constant, transforming a time-consuming data-intensive operation to a computationally intensive one. Another benefit is

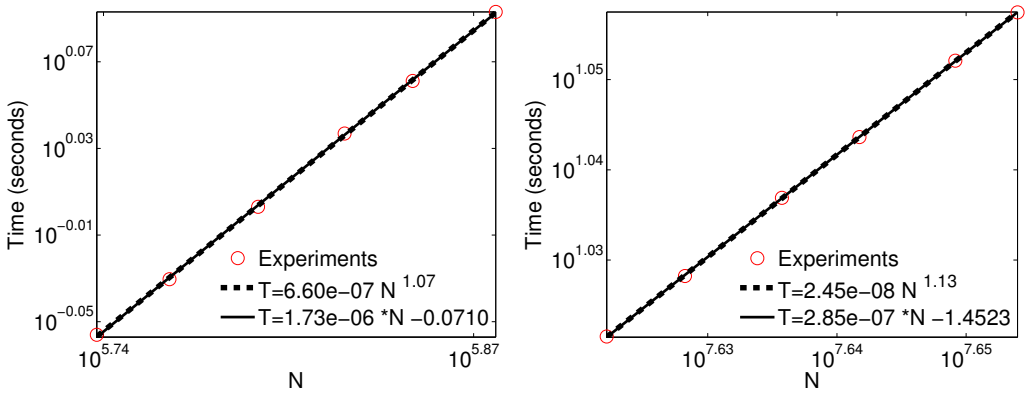
Table 8.1: Timing results (in seconds) for the summation $s(\mathbf{x})$ with M centres and N evaluation points, where β_2 is the fitted parameter in model II.

M	N_1	N_2	N_3	N	T	$\lvert\beta_2\rvert/T$	$\lvert\beta_2\rvert/(TM)$
1351	89	89	69	546549	0.8790	0.081	6.0e-05
1351	91	91	70	579670	0.9324	0.076	5.6e-05
1351	93	93	72	622728	1.0071	0.070	5.2e-05
1351	95	95	74	667850	1.0886	0.065	4.8e-05
1351	97	97	75	705675	1.1512	0.062	4.6e-05
1351	99	99	77	754677	1.2387	0.057	4.2e-05
5643	143	138	109	2151006	2.1361	0.087	1.5e-05
5643	145	140	111	2253300	2.2440	0.083	1.5e-05
5643	147	142	112	2337888	2.3374	0.080	1.4e-05
5643	149	144	114	2445984	2.4550	0.076	1.3e-05
5643	151	145	115	2517925	2.5315	0.074	1.3e-05
5643	153	147	117	2631447	2.6535	0.070	1.2e-05
24425	233	229	185	9871045	5.1615	0.108	4.4e-06
24425	235	231	187	10151295	5.3260	0.104	4.3e-06
24425	237	233	188	10381548	5.4584	0.102	4.2e-06
24425	239	235	190	10671350	5.6302	0.099	4.0e-06
24425	241	236	191	10863316	5.7369	0.097	4.0e-06
24425	243	238	193	11161962	5.9093	0.094	3.8e-06
105615	379	376	294	41896176	10.5082	0.138	1.3e-06
105615	381	378	295	42485310	10.6717	0.136	1.3e-06
105615	383	380	297	43225380	10.8867	0.133	1.3e-06
105615	385	382	298	43826860	11.0569	0.131	1.2e-06
105615	387	384	300	44582400	11.2744	0.129	1.2e-06
105615	389	385	301	45079265	11.4144	0.127	1.2e-06

Table 8.2: Parameter estimations and their 95% confidence intervals(CI) for the two models.

Model	M	c	95% CI	β_i	95% CI
I	1351	$6.60e \times 10^{-7}$	$[5.50e \times 10^{-7}, 7.92e \times 10^{-7}]$	1.07	$[1.05, 1.08]$
I	5643	$3.16e \times 10^{-7}$	$[2.86e \times 10^{-7}, 3.48e \times 10^{-7}]$	1.08	$[1.07, 1.09]$
I	24425	$1.03e \times 10^{-7}$	$[8.89e \times 10^{-8}, 1.19e \times 10^{-7}]$	1.10	$[1.09, 1.11]$
I	105615	$2.45e \times 10^{-8}$	$[2.22e \times 10^{-8}, 2.70e \times 10^{-8}]$	1.13	$[1.13, 1.14]$
II	1351	$1.73e \times 10^{-6}$	$[1.71e \times 10^{-6}, 1.76e \times 10^{-6}]$	-0.07	$[-0.09, -0.06]$
II	5643	$1.08e \times 10^{-6}$	$[1.07e \times 10^{-6}, 1.09e \times 10^{-6}]$	-0.19	$[-0.20, -0.17]$
II	24425	$5.79e \times 10^{-7}$	$[5.75e \times 10^{-7}, 5.83e \times 10^{-7}]$	-0.56	$[-0.60, -0.51]$
II	105615	$2.85e \times 10^{-7}$	$[2.84e \times 10^{-7}, 2.87e \times 10^{-7}]$	-1.45	$[-1.51, -1.39]$

that there is no overhead to set up a tree like data structure, such as those used in methods related to fast multipole methods. The data structure presented here



(a) $M = 1351$ (b) $M = 105615$

Figure 8.2: Timing results (red circle) and fitted line with model I and model II. The solid line for model II overlaps the dash line for model I.

can also be applied to functions which work together with `meshgrid` and `ndgrid` in Matlab, especially visualization functions for a 3D surface.

In stead of looping over the number of evaluation points, Algorithm 8.2 loops over the number of basis functions, M, which saves amount of unnecessary comparisons and branch statements. Both the analysis and numerical results show that the presented algorithms are scalable with $\mathcal{O}(N)$ complexity. Multithreading techniques could be applied to Algorithm 8.2 on the `for` loops with special care to avoid potential data race in `fval`. For multi-threading, one can only adjust the one loop function in Algorithm 8.2. More promising algorithms with $\mathcal{O}(M)$ complexity might be possible, if one allowed a more sophisticated data structure, however, the evaluation results may not be user-friendly for the visualization functions currently in Matlab.

Part IV

Applications

9. Scattered spatial data analysis

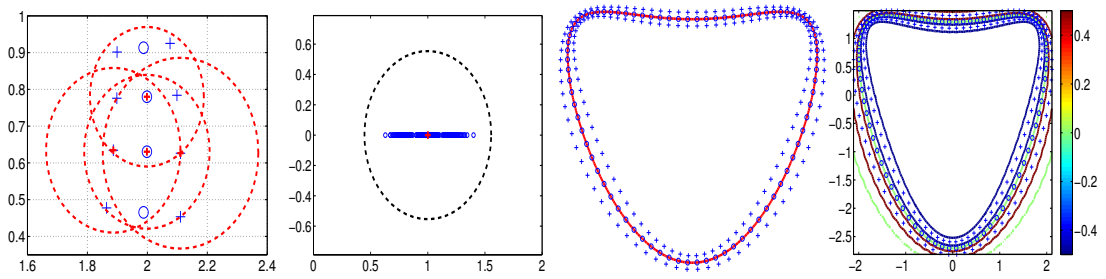
Designing different types of mesh-free methods for solving partial differential equations is one of the most important applications involving radial basis functions. Since there are lots of publications on such applications [70] and there are various other available techniques for solving partial differential equations, we shall avoid that application here. Instead we present other interesting applications. Firstly, we discuss how to use compactly supported radial basis functions to construct an implicit surface for modelling and visualizing a $3D$ object from point clouds. Secondly, we apply our methods to terrain modelling and meteorology. Finally, we apply our methods to scattered spatial environmental data analysis, for example, visualizing pollution in air, soil and water, identifying pollutant sources, and evaluation of environmental risk. We think that this is a relatively new application area to be explored in the light of increasing public concern about environmental problems.

9.1 3D object modelling by implicit surface reconstruction

As discussed on page 4, a planar implicit curve can be viewed as a contour of a 3D graph (see definition on page 4), a 3D object can be represented as an isosurface of a 4D graph. An implicit curve/surface can be constructed as a zero level set of a function $f(\mathbf{x})$, i.e. $f(\mathbf{x})$ satisfies

$$\begin{cases} f(\mathbf{x}_i) = 0 & \text{if } \mathbf{x} \text{ on the curve/surface;} \\ f(\mathbf{x}_i^+) = d_i^+ & \text{if } \mathbf{x}_i^+ \text{ off the curve/surface (outside);} \\ f(\mathbf{x}_i^-) = -d_i^- & \text{if } \mathbf{x}_i^- \text{ off the curve/surface (inside).} \end{cases}$$

Given a point $\mathbf{x}_i$ on the curve/surface and its unit normal $\mathbf{n}_i$, artificial off-surface points $\mathbf{x}^{\pm}$ can be constructed easily: $\mathbf{x}_i^{\pm} = \mathbf{x}_i \pm \delta_i \mathbf{n}_i$ for some δ_i. The $d_i^{\pm}$ can be arbitrary positive numbers. Carr et al. argue that the reconstruction result is insensitive to the choice of $d_i^{\pm}$ [36]. Here we choose $d_i = \delta_i$, and δ_i is proportional



(a) Basis centres (b) Eigenvalues (c) Reconstructed curve (d) Contours

Figure 9.1: Reconstruction of an implicit curve. The blue + points are artificial off-curve points constructed by related normal information. The red circles in (a) are the supports for the basis functions corresponding to red + centres (shown for only a subset of points, all the interpolation points are illustrated in (c)). (b) demonstrates that the eigenvalues of the interpolation matrix are located in the black circle. The red curve in (c) and the 0 level set in (d) are the reconstructed curve.

to the nearest distance from a point $\mathbf{x}_j$ on the surface to $\mathbf{x}_i$. An implicit curve construction problem is used to illustrate such a procedure.

9.1.1 Illustration in 2D

Consider the following heart-shaped curve

$$\begin{cases} x(t) = \sqrt{2}(\sin(t) - \cos(t)); \\ y(t) = -\sqrt{2}(\sin(t) + \cos(t)(1 + \sin(t))), \end{cases} \tag{9.1}$$

which is modified from an example in [70, p.257]. The 84 points on the curve are sampled by setting $\{t_i\}_{i=1}^{84}$ equally spaced in $[0, 2\pi]$. See the small blue circles in Figure 9.1(c). The off-curve artificial points are constructed according to $\mathbf{x}_i^{\pm} = \mathbf{x}_i \pm .75\delta_i\mathbf{n}_i$, where $\mathbf{n}_i$ is the normalised normal direction at $\mathbf{x}_i$ and δ_i is the distance between $\mathbf{x}_i$ and its nearest neighbour ($\mathbf{x}_m$ for some m) on the curve; the ratio .75 can be a scalar not far from 1 (the ratio is not sensitive according to Carr et al. [36]). We use CSRBFs with different shape as discussed in Chapter 4 to interpolate $f(\mathbf{x})$. The Wendland function $\phi_{3,1}(r) = (1-r)_+^4(4r+1)$ is employed. Figure 9.1(a) illustrates the support of each basis function, the supports are determined by Algorithm 5.1. Figure 9.1(b) demonstrates that the eigenvalues of the interpolation matrix are located in the black circle by the diagonal dominance condition. The red

Algorithm 9.1 A framework for 3D surface reconstruction with CSRBFs

1: Prepare the artificial data, normal information, and generating right hand side of the linear system.
2: Generating the interpolation matrix according to Algorithm 5.4.
3: Evaluating the interpolant on fine Cartesian grids according to Algorithm 8.2.
4: Visualising the object.

curve in Figure 9.1(c) and the zero level set in Figure 9.1(d) are the reconstructed result.

9.1.2 3D object modelling

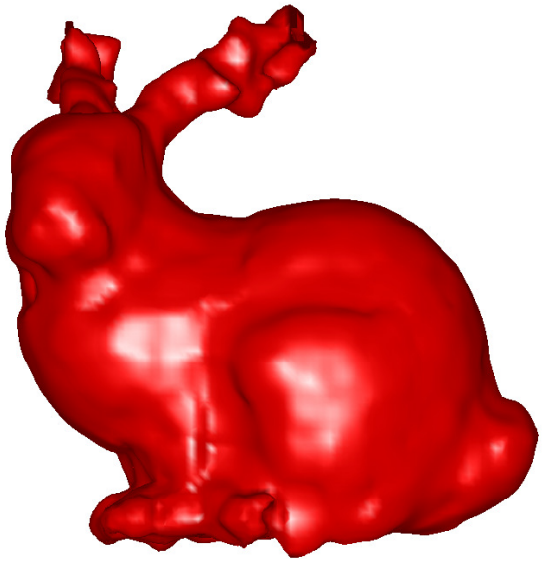
We consider the Stanford bunny model and the Stanford dragon model in the Stanford 3D scanning repository[1]. The Stanford bunny, `bunny.tar.gz` includes four data sets of 35947, 8171, 1889 and 453 points. The Stanford dragon model is larger; only three low resolution data sets in `dragon_recon.tar.gz` are used, the data sets with 100250, 22998 and 5205 vertices. The test problems are named as `Bunny453`(B4), `Bunny1889`(B3), `Bunny8171`(B2), `Bunny35947`(B1), `Dragon5205`(D3), `Dragon22998`(D2) and `Dragon-100250`(D1) respectively as previously. The normal information is computed by the program `normalsply` from the package `ply.tar.gz` provided by Greg Turk[2]. The artificial points are constructed according to $\mathbf{x}_i^{\pm} = \mathbf{x}_i \pm .5\delta_i \mathbf{n}_i$. A framework of 3D object modelling with radial basis functions is presented in Algorithm 9.1. Multilevel methods have not been applied here. Figure 9.2 shows the reconstruction results for the Stanford bunny model and the Stanford dragon model.

9.2 Terrain modelling

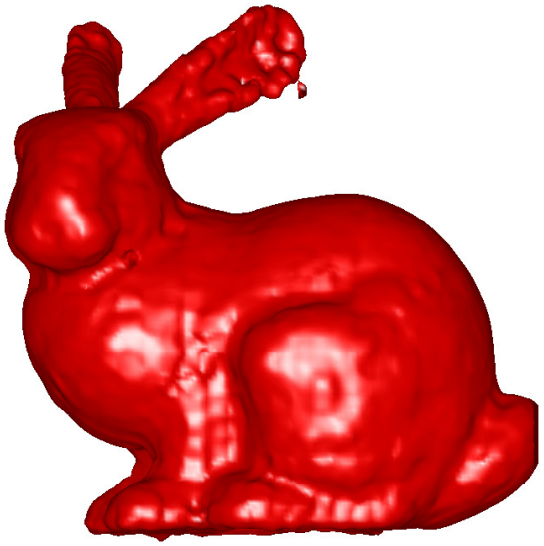
Terrain modelling is one of the forces which drives research on scattered data approximation. We use the terrain of Colorado as our first example because the region is roughly square on the earth and the true terrain is available from high resolution maps. The elevations and positions in longitude and latitude of 346 cites in Colorado are extracted from Wolfram Mathematica `CityData`.

```
{CityData[#, "Longitude"], CityData[#, "Latitude"],
```

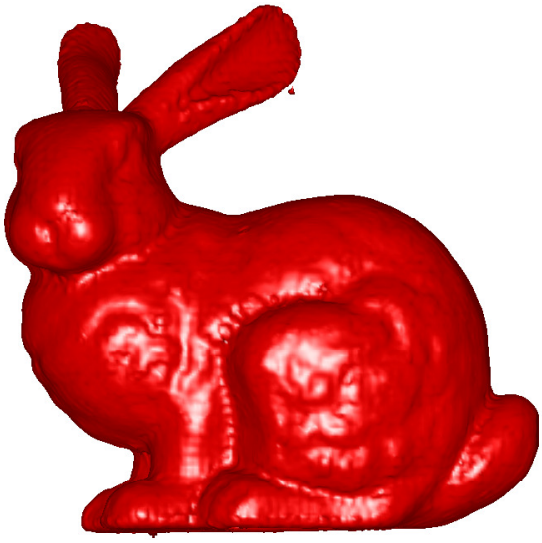
[1]http://graphics.stanford.edu/data/3Dscanrep/
[2]http://www.cc.gatech.edu/projects/large_models/ply.html



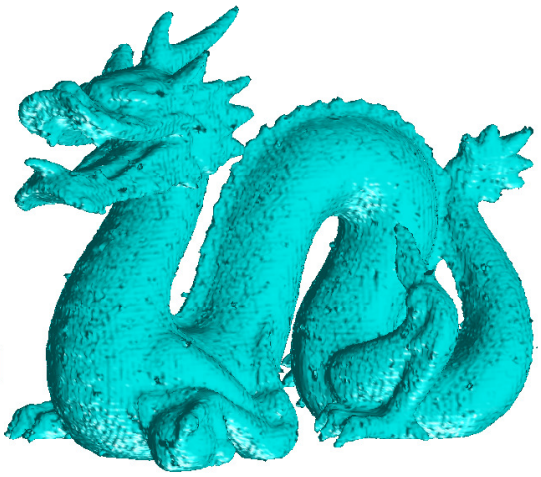
(a) $N = 89 \times 89 \times 69$



(b) $N = 143 \times 138 \times 109$

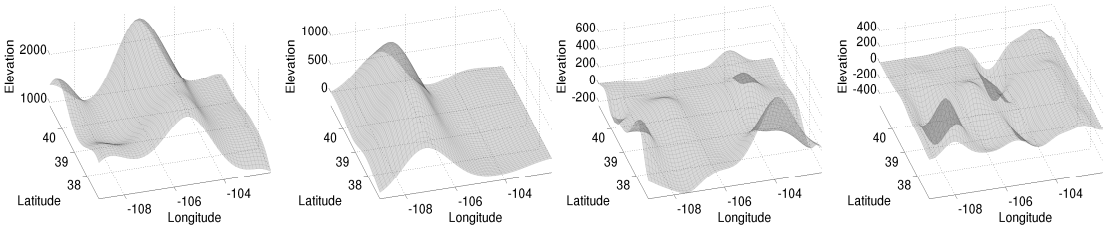


(c) $N = 233 \times 229 \times 185$

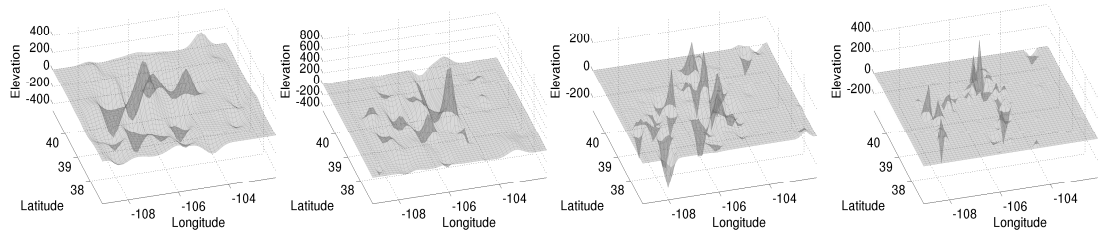


(d) Dragon5205

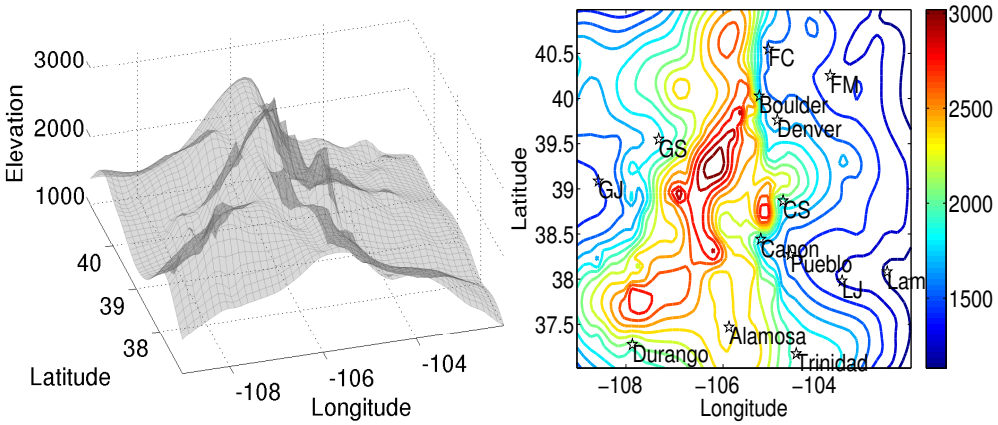
Figure 9.2: Reconstruction 3D surface from point clouds. (a) `Bunny453` reconstructed from 453 points with $M = 1351$ interpolation centres. (b) `Bunny1889` reconstructed from 1189 points with $M = 5643$ interpolation centres. (c) `Bunny8171` reconstructed from 8171 points with $M = 24425$ interpolation centres. (d) `Dragon5205` reconstructed from 5205 points with $M = 15563$ interpolation centres. N is the number of evaluation points for visualization.



(a) Level 1, $s_1(\mathbf{x})$ (b) level 2, $s_2(\mathbf{x})$ (c) level 3, $s_3(\mathbf{x})$ (d) level 4, $s_4(\mathbf{x})$



(e) level 5, $s_5(\mathbf{x})$ (f) level 6, $s_6(\mathbf{x})$ (g) level 7, $s_7(\mathbf{x})$ (h) level 8, $s_8(\mathbf{x})$



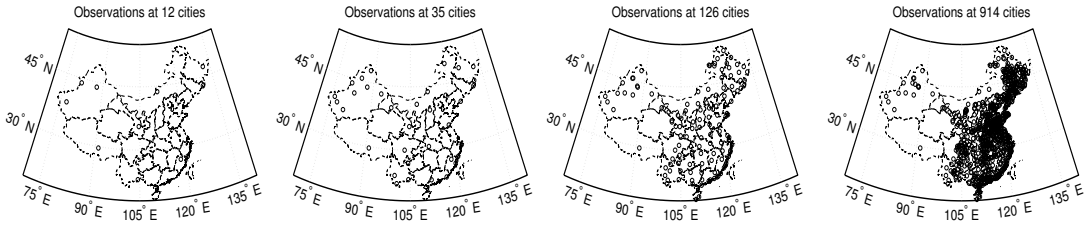
(i) Terrain of Colorado (j) Contour map of Colorado

Figure 9.3: Terrain of Colorado, USA. In (j), FC stands for Fort Collins, CS for Colorado Springs, LJ for La Junta, GS for Glenwood Spring, GJ for Grand Junction, and FM for Fort Morgan.

```
CityData[#, "Elevation"]} & /@ CityData[{All, "Colorado", "USA"}]
```

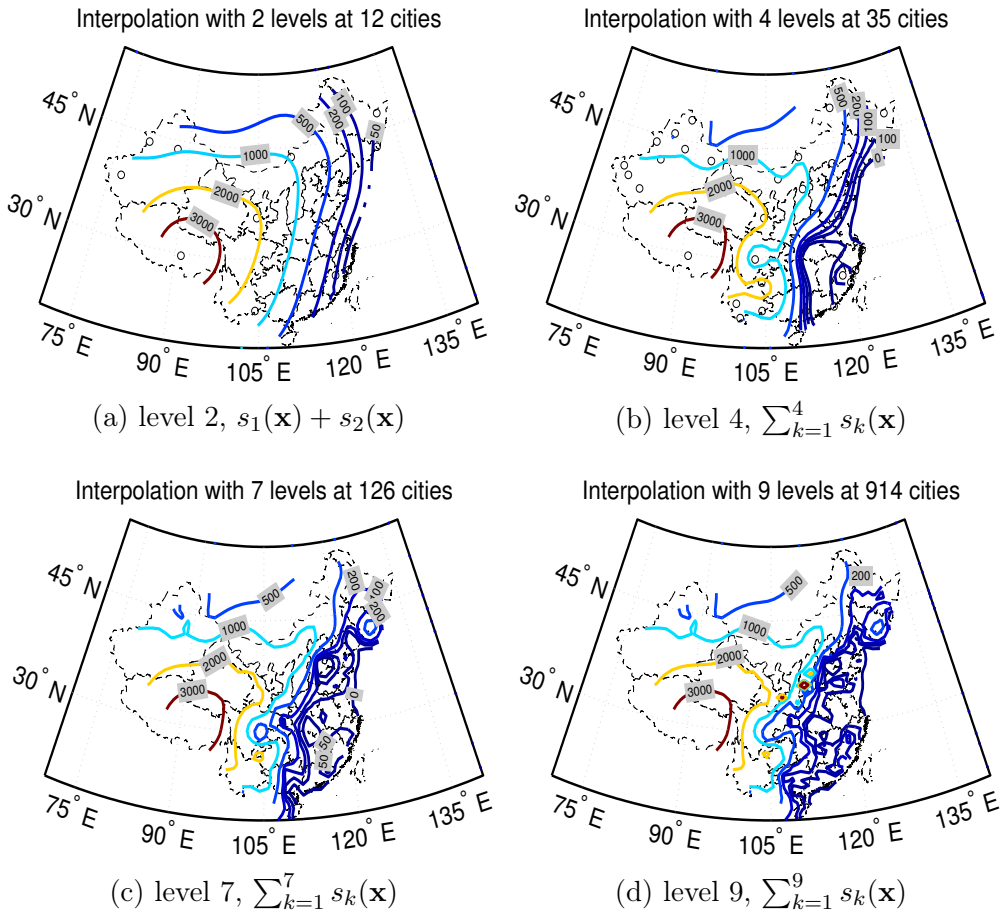
The cites can be divided into 8 hierarchical data sets in Figure 7.6. The interpolants in each level are illustrated in Figure 9.3. The contour map and several main cities are illustrated in Figure 9.3(j).

Our second example is the terrain of the mainland of China. The longitude, latitude and elevation of 914 cites are also obtained from Wolfram Mathematica



(a) Level 2 (b) Level 4 (c) Level 7 (d) Level 9

Figure 9.4: Hierarchical data sets of cites in mainland of China.



(a) level 2, $s_1(\mathbf{x}) + s_2(\mathbf{x})$ (b) level 4, $\sum_{k=1}^{4} s_k(\mathbf{x})$

(c) level 7, $\sum_{k=1}^{7} s_k(\mathbf{x})$ (d) level 9, $\sum_{k=1}^{9} s_k(\mathbf{x})$

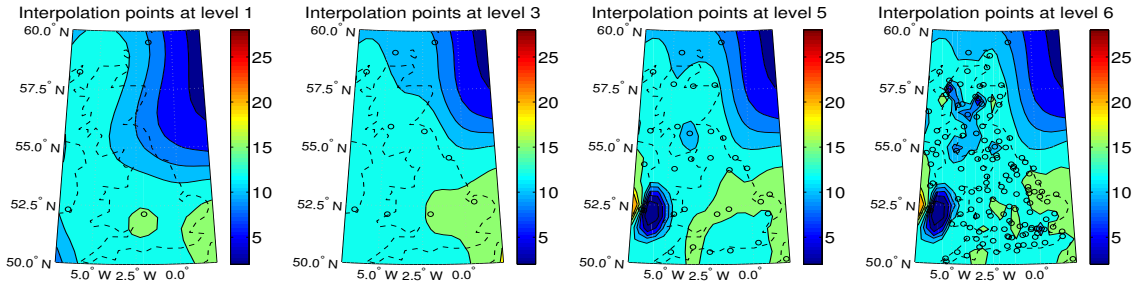
Figure 9.5: Terrain of mainland of China.

```
CityData
```

```
{CityData[#, "Longitude"], CityData[#, "Latitude"],
  CityData[#, "Elevation"]} & /@ CityData[{All, "China"}]
```

Hierarchical scattered data sets of the cites are illustrated in Figure 9.4. We use the planar coordinates, longitude and latitude, to calculate the distance. Geodesic distance could be considered for a more accurate model. The contour maps with different resolutions are illustrated in Figure 9.5.

9.3 Interpolating meteorological data on the earth



(a) (b) (c) (d)

Figure 9.6: The temperature maps for UK.

Meteorology is one important field which involves high dimensional scattered data approximation. Meteorological data are obtained from irregularly distributed observations stations. The observation stations which are associated to the world meteorology organization are scattered all over the world. Most of the stations are scattered in continents. See Figure 7.7 for illustration. Temperature, humidity, wind speed, wind direction, pressure and other data are collected at each station at a fixed time every day. An interpolation of these measurements onto certain mesh points can serve as the initial values for numerical weather forecasting programs. Weather forecasting and some climate modelling require four dimensional interpolation or approximation [58]. We consider a simpler 2D model. We only focus on how to interpolate a scalar field from these observations. The data come from Wolfram Mathematical `WeatherData`. Firstly we use the information from observation stations near the United Kingdom on June 3rd 2014, and construct the temperature distribution maps for the UK. The heterogeneous hierarchical approach with Wendland functions $\phi_{3,1}$ is used. The results are illustrated in Figure 9.6. Secondly, we extract all available temperature information on June 22, 2014, and draw the temperature maps of the world. We interpolate on the sphere in the following way: we first map the sphere to a $[-180, 180] \times [-90, 90]$ domain and then padding

Table 9.1: Observation information of temperature

	original observations	extended observations
temperature	8561	17076
pressure	6816	13608

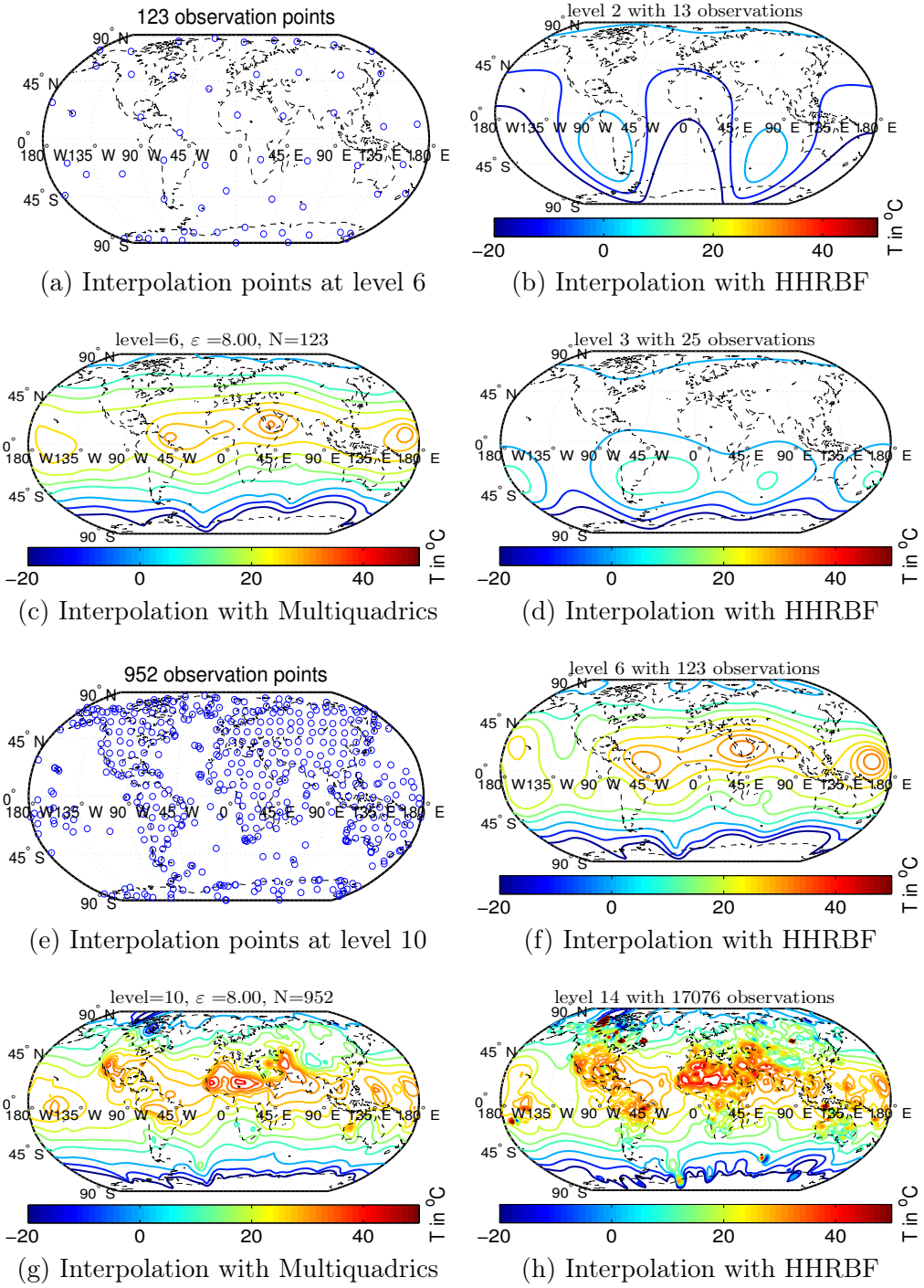
the domain along the longitude coordinate, the range of longitude are extended to -200 and 200 by assuming periodic conditions along the longitude coordinate so that the artificial interface is along the ± 180. The number of original observations and extended observations are listed in Table 9.1.

We use two ways to interpolate observations, firstly we use multiquadrics. Secondly we use the heterogeneous hierarchical approach. Figure 9.7 shows the interpolation results of the two methods.

9.4 Pollution sources identification

With the high-output of environmental data like concentrations of pollutants in air, soil and water, there is an increasing demand for visualizing spatial environmental data. Visualization brings us a vivid and direct impression of the concentration distribution of pollutants. This makes it easier for us to obtain intuitions and insights into potential reasons for the underlying pollution, where these pollutants come from, how they diffuse and finally how to control or clear them up. Such spatial environmental data are usually sparse, scattered and unstructured. For example, the small circles in Figure 9.8(e), Figure 9.8(f) and Figure 9.9(d) illustrate the positions of ground monitoring points for air pollutants in two cities; based on such observations, we want to approximate the concentration distributions of several air pollutants in the nearby region. The first column of Figure 9.10 and Figure 9.11 illustrate the distribution of cites which have ground air pollution monitoring system over a large part of China. Based on data from such sparse and scattered cities, we want to depict the density distribution of air pollutants over a larger scale, in a region of several provinces or the whole country.

Our second example is a soil pollution problem, the small white circles in Figure 9.12 are sample points of soil in a city. Concentrations of 8 heavy metals in the samples are tested. Based on such scattered measurements, we want to know the concentration distributions of these heavy metals in soil across the city. It is evident



(a) Interpolation points at level 6 (b) Interpolation with HHRBF

(c) Interpolation with Multiquadrics (d) Interpolation with HHRBF

(e) Interpolation points at level 10 (f) Interpolation with HHRBF

(g) Interpolation with Multiquadrics (h) Interpolation with HHRBF

Figure 9.7: Temperature and pressure maps for the world on June 22, 2014. The multiquadric interpolation formula is $s(\mathbf{x}) = \sum_{k=1}^{N} \alpha_k \sqrt{\varepsilon^2 \|\mathbf{x} - \mathbf{x}_k\|^2 + 1} + \alpha_0$. The results in the second column are obtained according to the heterogeneous hierarchical interpolation approach.

that scattered spatial environmental data are universal in environmental science.

For the air pollution problem, we focus on the density distribution of 3 air pollutants: SO2; fine particulate matter (PM2.5, diameter of 2.5 micrometers or less); and respirable suspended particle (RSP or PM10, particles with diameter of 10 micrometres or less). Table 9.2 and Table 9.3 list the positions of observation stations and the concentrations of the 3 air pollutants at these observation points. The real-time data are extracted from `www.pm25s.com`[3] on June 14, 2014. The position data are obtained from a GPS positioning website[4], which provides longitude and latitude information. We use the following formula to approximate such small observations:

$$s(\mathbf{x}) = \sum_{i=1}^{N} \alpha_i \phi_i(\varepsilon_i \|\mathbf{x} - \mathbf{x}_i\|) + \beta_0, \tag{9.2}$$

where β_0 is the average of the observation which can be calculated in advance. α_i and ε_i are determined by the algorithm described in Chapter 4. β_0 can contain background information, which can, for example, be obtained from remote sensing or a trend surface from a larger scale approximation. Figure 9.8 and Figure 9.9 illustrate the density distribution of the air pollutants. When using globally supported radial basis functions in (9.2), those region outside the convex hull of the interpolation points can be very large; while using compactly supported radial basis function without the constant β_0, the density would vanish at regions which are not covered by any basis function. The constant term β_0 for compactly supported radial basis functions is a trade-off which can reduces the large extrapolation error. See Figure 9.8 for comparison.

For larger scale approximation, we use the heterogeneous hierarchical approach described in Chapter 7. We extract the average concentrations of PM2.5 and PM10 for 190 cites in China on June 22, 2014. The data sets are divided into 6 regions. For each region we apply the heterogeneous hierarchical approach described in Chapter 7, and then draw the approximate density distribution of PM2.5 and PM10. The data comes from `www.pm25s.com`. The results are illustrated in Figure 9.10 and Figure 9.11.

For the soil pollution problem, the data comes from the China Undergraduate

[3] http://www.pm25s.com/chongqing.html and http://www.pm25s.com/wuhan.html
[4] http://www.gpsspg.com/latitude-and-longitude.htm

Mathematical Modelling Contest[5] (Problem A, 2011). Figure 9.12(a) illustrates the terrain of a city and some sample points. Concentrations of 8 heavy metals, As, Cd, Cr, Cu, Hg, Ni, Pb and Zn are tested at the sample points. The tolerable threshold of the concentration of a heavy metal depends on the function of the underlying area. A living area requires a higher standard than an industrial area. The functional areas of the city are illustrated in Figure 9.12(b), which consists of living area (C), industrial area (I), transportation (T), mountain (M) and parks (P). We use the multiquadric $\sqrt{(3r)^2+1}$ to approximate the concentration distributions of the 8 heavy metals. Figure 9.12 demonstrates the potential sources of the pollution and the distribution of the concentration.

9.5 Discussion

This chapter illustrates the utility and flexibility of radial basis functions for scattered spatial data analysis. All of the presented models do not require high accuracy; for example, predicting the temperature to 0.1^oC makes little sense in daily life. Flexibility and convenience of the underlying method is of great concern and even paramount. The 3D object modelling problems show that good scalability can be obtained when using CSRBFs with different shapes as described in Chapter 4. Even when the multilevel approach hasn't been applied, satisfactory results can be obtained.

For other applications, we want to show that scattered data approximation is common and becoming increasingly important in daily life. The air pollution problem in China, is a very recent problem which is receiving a great deal of public attention.

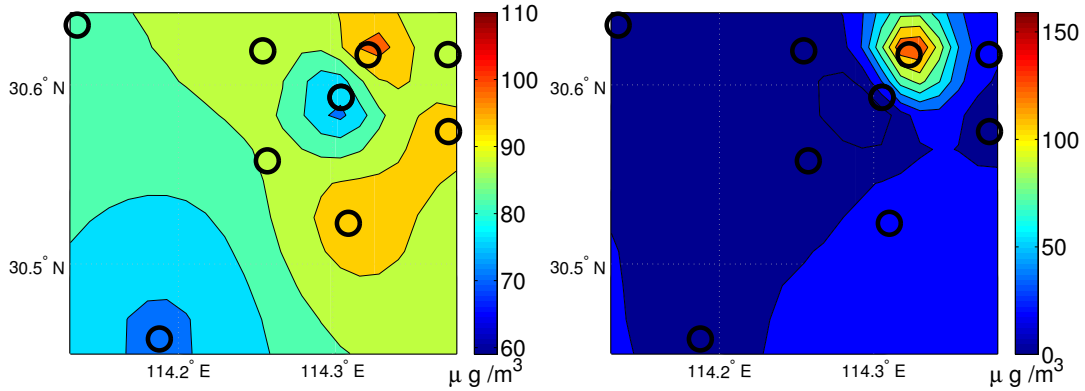
[5] http://www.mcm.edu.cn/html_cn/node/a1ffc4c5587c8a6f96eacefb8dbcc34e.html

Table 9.2: Observations of density distributions of PM2.5 and SO2 in Wuhan, China, Data were extracted from http://www.pm25s.com/wuhan.html on June 14, 2014. The position data are identified by a GPS positioning website http://www.gpsspg.com/latitude-and-longitude.html.

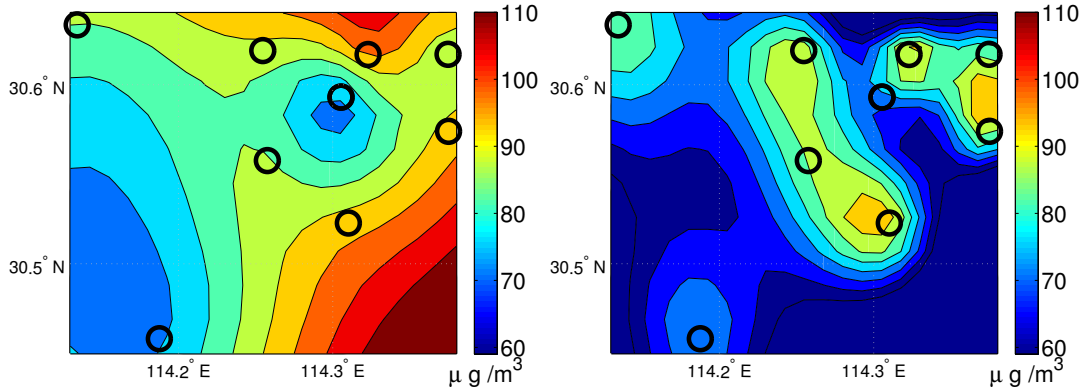
Position (latitude,longitude)	Density ($\mu g/m^3$) pm2.5	SO2
(30.574,114.375)	95	11
(30.593,114.305)	79	9
(30.633,114.134)	87	9
(30.523,114.310)	96	24
(30.617,114.323)	100	149
(30.619,114.255)	91	5
(30.558,114.258)	88	7
(30.593,114.305)	69	1
(30.458,114.188)	75	15
(30.617,114.375)	89	21

Table 9.3: Observations of density distributions of PM2.5, PM10 and SO2 in Chongqing China. Data were extracted from http://www.pm25s.com/chongqing.html on June 14, 2014. The position data are identified by a GPS positioning website http://www.gpsspg.com/latitude-and-longitude.htm.

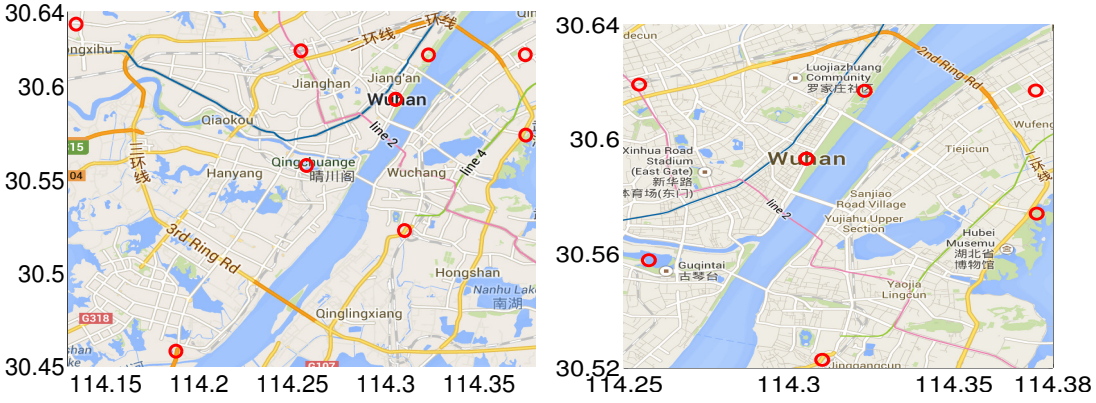
Position (latitude,longitude)	Density ($\mu g/m^3$) pm2.5	pm10	SO2
(29.714,106.623)	39	79	9
(29.534,106.565)	50	54	32
(29.425,106.579)	48	54	13
(29.600,106.642)	52	104	9
(29.818,106.419)	35	55	12
(29.475,106.472)	51	82	18
(29.511,106.510)	59	92	30
(29.490,106.363)	47	57	3
(29.670,106.484)	40	76	12
(29.756,106.652)	43	70	8
(29.841,106.395)	23	31	1
(29.743,106.485)	26	38	6
(29.600,106.307)	37	48	16
(29.560,106.573)	38	80	12
(29.580,106.459)	53	79	9
(29.383,106.519)	44	120	25



(a) PM2.5 (b) SO2

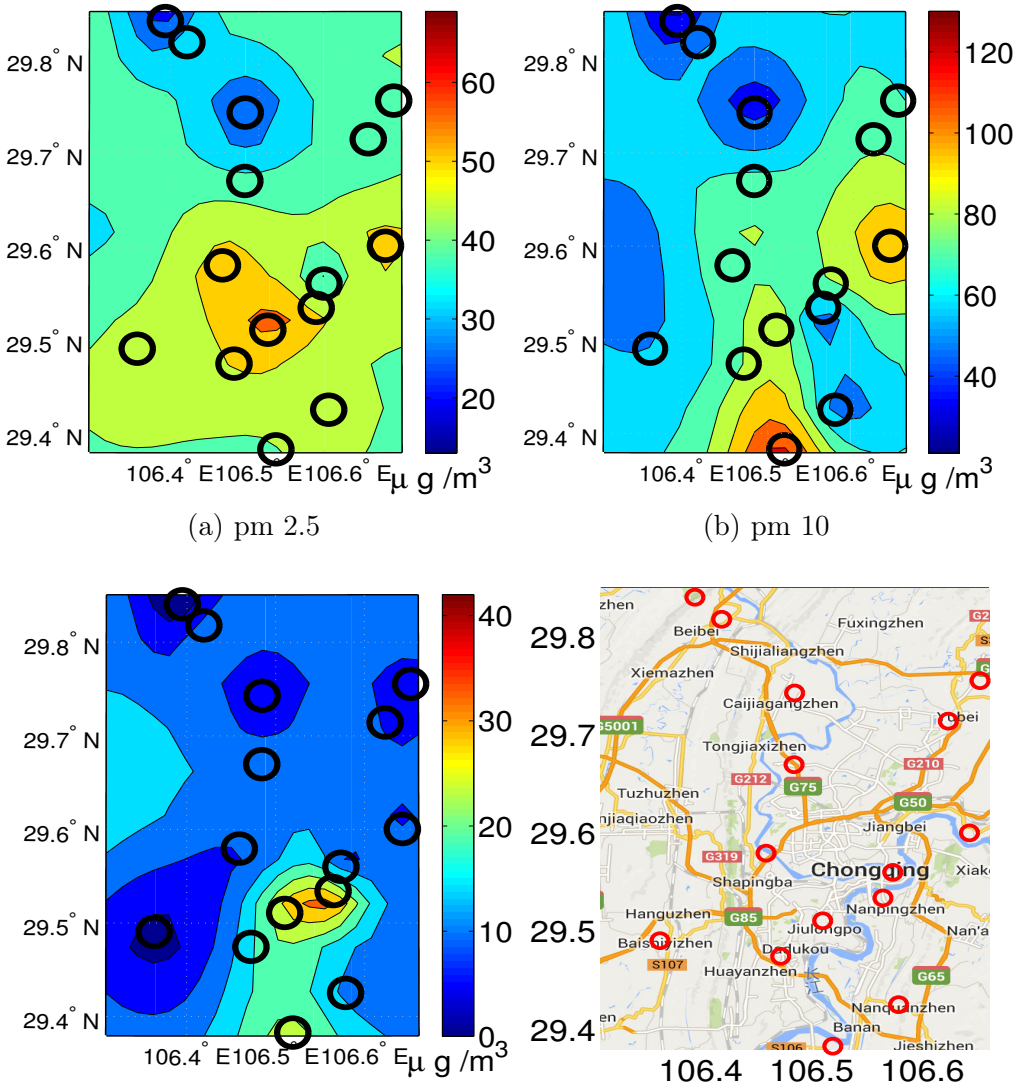


(c) PM2.5, $r^2(\log(r)-1)$ (d) PM2.5, $(1-r)_+^4(4r+1)$



(e) Map of Wuhan (f) Local map of Wuhan

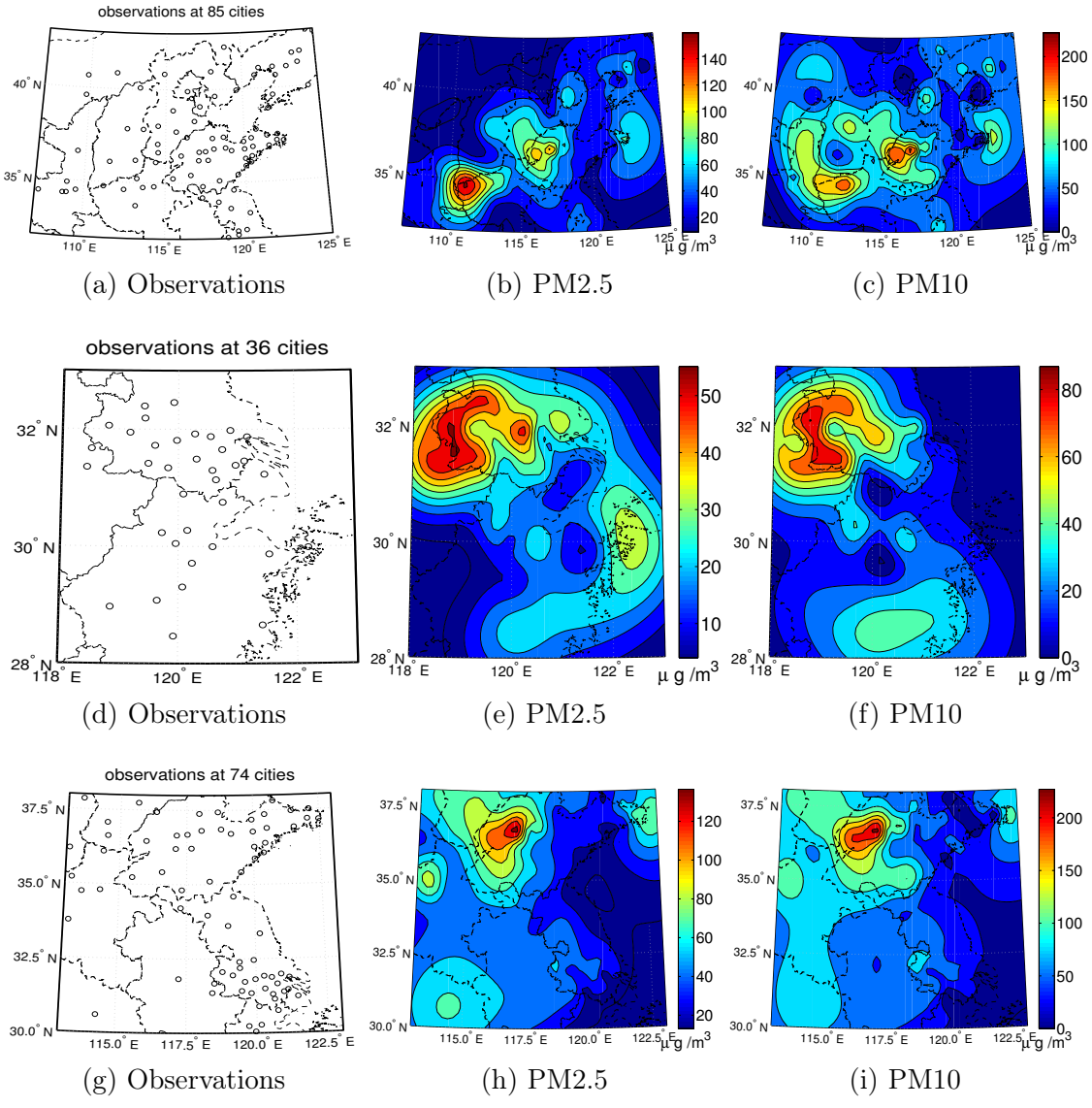
Figure 9.8: Density distribution of air pollutants in Wuhan, June 14, 2014. (a) and (b) use the method which first subtracts the average mean as the the background information, and finally add the background information to the interpolant. (c) uses the globally supported radial basis function $r^2 \log r$, it produces larger results at the bottom-right conner. (d) applies CSRBFs with different shapes directly on the observation data, it produces lower values at the bottom-right conner.



(a) pm 2.5 (b) pm 10

(c) SO2 (d) Map of Chongqing

Figure 9.9: Density distribution of air pollutant in Chongqing, June 14, 2014.

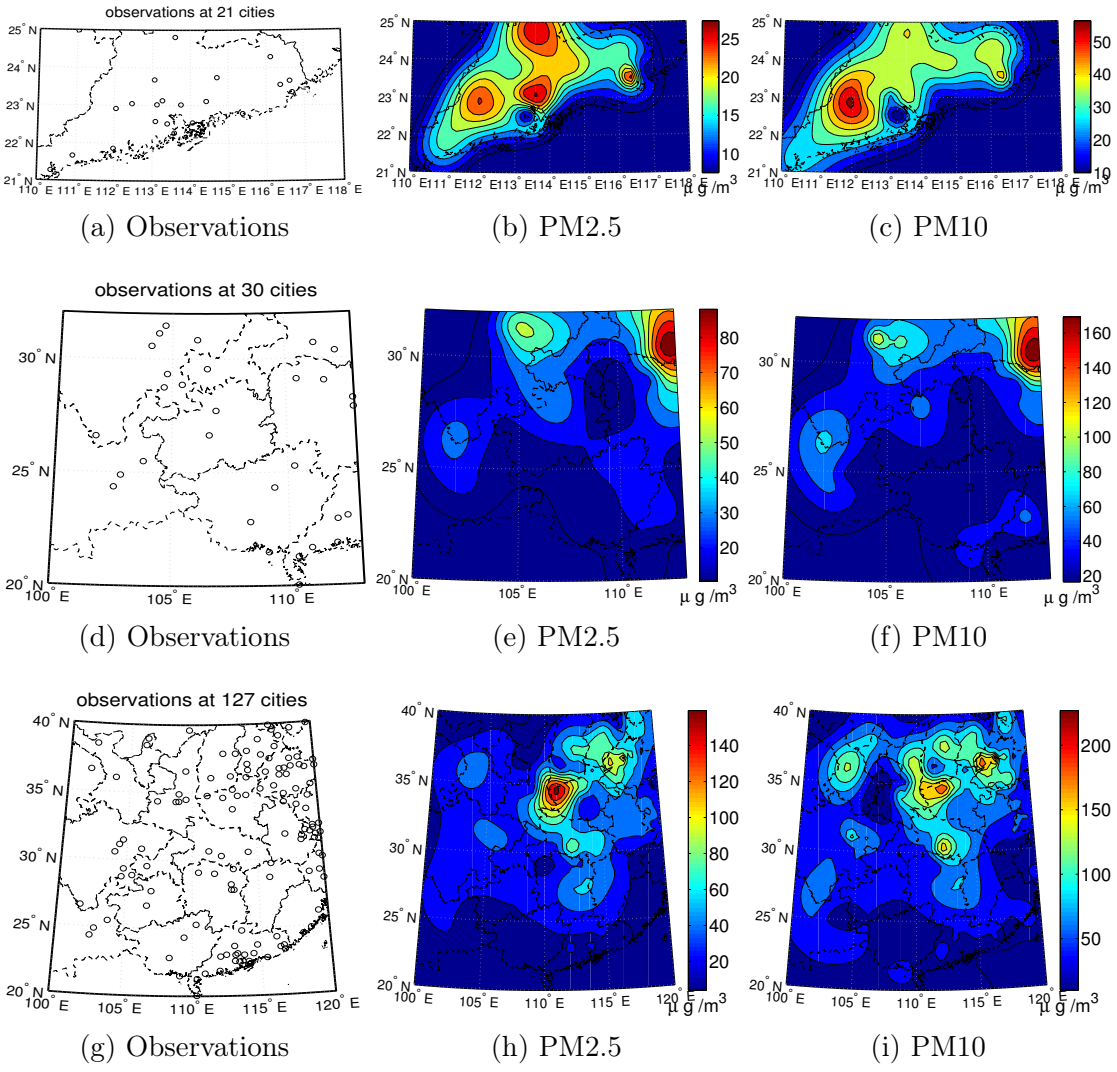


(a) Observations (b) PM2.5 (c) PM10

(d) Observations (e) PM2.5 (f) PM10

(g) Observations (h) PM2.5 (i) PM10

Figure 9.10: Density distribution of PM2.5 and PM10 in China I.



(a) Observations (b) PM2.5 (c) PM10

(d) Observations (e) PM2.5 (f) PM10

(g) Observations (h) PM2.5 (i) PM10

Figure 9.11: Density distribution of PM2.5 and PM10 in China II.

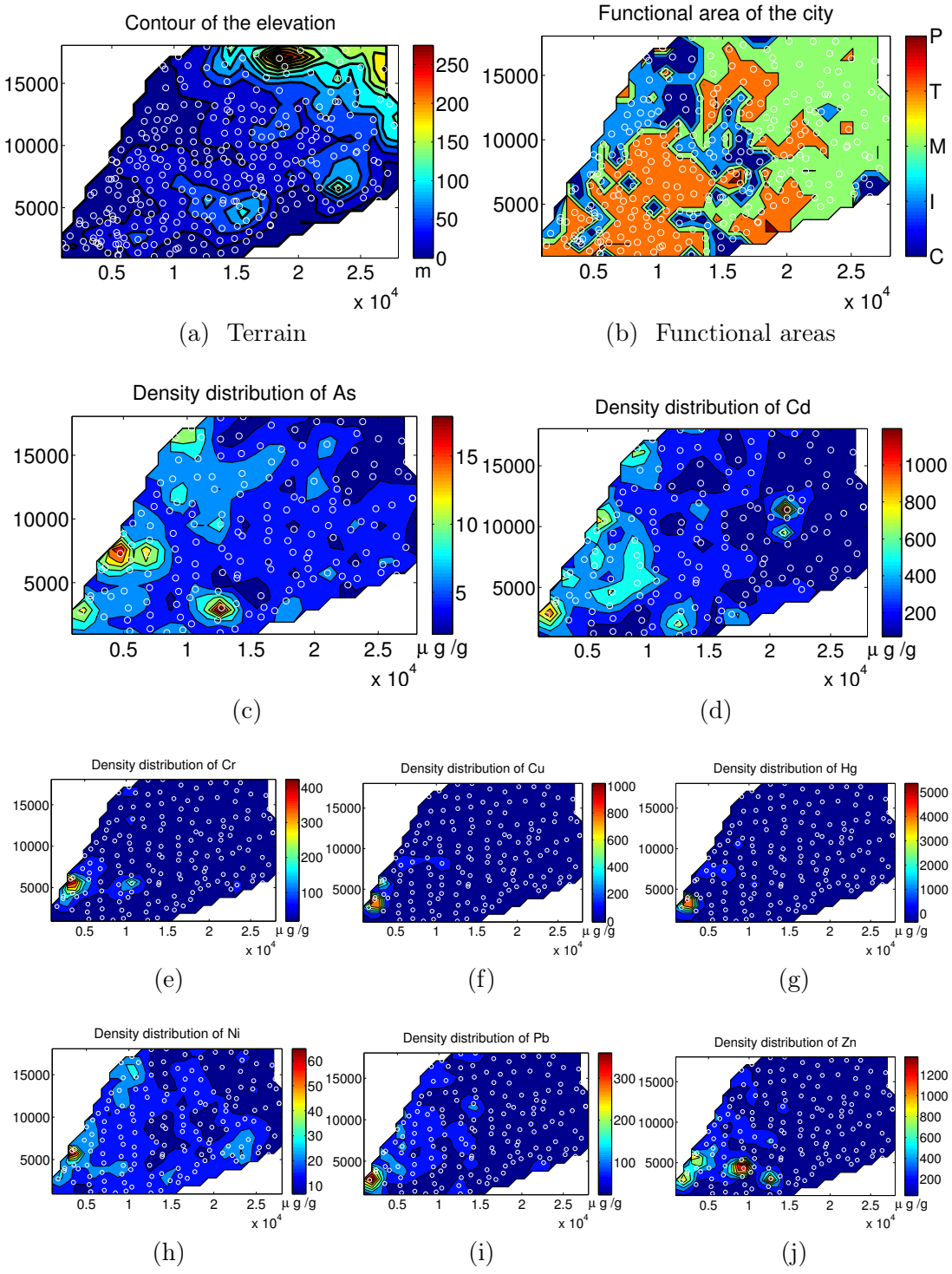


Figure 9.12: Density distribution of heavy metals in soil of a city. in ((b)), C stands for living area, I for industrial area, T for transportation area, M for mountains, and P for parks.

10. Discussion and future work

As surveyed in Chapter 2, there is a long history of seeking proper and convenient basis functions for scattered data approximation in high dimensional space in Chapter 2. Problems related to radial basis functions emerged one by one. Some of the problems are of great interest themselves, but most of the profound ones arose in response to some requirements of real applications. Engineers and scientists may wish to use multiquadrics or thin-plate splines to reconstruct irregular surfaces, whilst mathematicians are interested in why such methods work, how well such methods work and in seeking simper alternative methods. The derivation of positive definite compactly supported radial basis functions was a step forward. However, soon such basis functions were found to violate the Strang-Fix condition. Few people discuss such a negative result, but research on fixing this deficiency is continuing. To some degree, the Strang-Fix condition also suggests how to fix this: by using multi-scale generating functions. There is long way to go before for CSRBFs can compete with classical finite element, finite difference and finite volume method for solving many challenging real applications, but it is also difficult to find a simper method to deal with truly scattered data approximation like the applications presented in Chapter 9. The correct choice of tool is an essential step in problem solving.

In this final chapter of this thesis, we ask what problems which should be further considered and what opportunities exist in this field.

10.1 Error estimate for the adaptive multilevel approach

There are a multitude of reasons which motivated us to design the method of using CSRBFs with different shapes proposed in Chapter 4. We hope the reader is convinced that such a scheme is necessary for scattered data sets which are not quasi-uniform, like those in Figure 7.6, Figure 7.7, Figure 9.4, Figure 9.8 and Figure 9.9. Such a scheme is an important step towards to adaptivity, and adaptivity plays an important role in removing the curse of high dimensionality. One striking

result is that convergence is observed when employing such a scheme in a multilevel fashion and good scalability also remains. This is quite different for the multilevel stationary interpolation which does not converge. We have not understood why such a scheme works and how well such a scheme works. Further investigation is definitely needed.

10.2 Smoothness matching and further implementation in 3D

The perspective of looking at the spectral distribution of kernel matrices related to radial basis functions convinces us that the cause of the ill-conditioning lies in the observation that the smoothness does not match between the target function and the underlying kernel function. We have not made it clear how to match the smoothness, for instance, how to choose a proper basis function without any information on the regularity of the solution and the target function except some observations, and how to choose a proper window width for a compactly supported radial basis function. The LOOCV method provides some insight, but more profound results are needed. To find a specific matching condition, we have to find proper tools, possibly from nonlinear functional analysis and best approximation theory.

The multilevel method has not been applied to the 3D model; this is a further implementation need.

10.3 CSRBFs and Strang-Fix condition on sphere

Another approximation problem of great interest is how to find compactly supported zonal functions on the sphere. The problem is motivated by scattered data approximation or signal processing on the earth and other heavenly bodies. Some work in this direction has been carried out by Hubert [113] and Hubert is continuing working on this problem. We should also pay attention to whether there is a similar condition to the Strang-Fix condition on the sphere? The answers to such problems are unclear.

10.4 Sparse approximation

For many problems with enough observations, approximation rather than interpolation is used, particularly, as most observations are to some degree uncertainty.

A problem to be solved is how to adaptively choose interpolation centres and use limited basis functions to approximate many observations. This can avoid over-fitting. There are greedy algorithms based on recursive least squares. Recently introduced techniques like ℓ_1 and ℓ_0 optimization can serve as promising tools. Discussion in this direction seems rare.

10.5 Data manipulation

As we have seen, one of the challenging problem for scattered data approximation is data manipulation. Data sets may be "big", but this is not the most significant issue; the challenge lies in whether the data is structured or not. As we have seen, the preprocessing (data querying) and post-processing (evaluation and visualisation) time for scattered data approximation takes much longer time than solving the interpolation problems themselves. Many mathematicians mainly focus on designing approximation schemes, while few pay attention to manipulating data, such as data querying. Data manipulation on manifolds is common and more challenging—approximation problems arise in this context.

Scattered data approximation is a fast developing area and problems will continue to arise in many fields so that the relevance of approximation methods for such problems can only increase. There are many possible further research opportunities, we have only summarized a couple of them which are most relevant to this thesis.

Appendix: related work and possible directions

Table 10.1: Commonly used radial basis functions

Name(abbreviation)	Formula	Support
Multiquadrics (MQ)	$\sqrt{1+(\varepsilon r)^2}$	global
Inverse Multiquadrics (IMQ)	$\frac{1}{\sqrt{1+(\varepsilon r)^2}}$	global
Inverse quadrics (IQ)	$\frac{1}{1+(\varepsilon r)^2}$	global
Gaussian (GA)	$e^{-(\varepsilon r)^2}$	global
Thin-plate splines (TPS)	$(-1)^{k+1} r^{2k} \log r$	global
Biharmonic splines (BS)	r^3 in $\mathbb{R}$, $r^2(\ln r - 1)$, in $\mathbb{R}^2$	global
Power functions (PF)	r^γ, $\gamma > 0$	global
Askey's power (AP)	$(1-r)_+^\ell$	compact
Wendland functions (WD)	Table 2.1	compact
Missing Wendland functions (MS)	Table 2.2	compact
Wu functions (WF)	§ 2.2.2	compact

Table 10.2: Methods and possible future work for commonly used RBFs

Related to work in this thesis	Apply to RBFs	Possible directions
	Closed formula of WD, MWD	relationships of the derivatives between Gaussian and Wendland function increasing smoothness
Eigenvalue distribution (Chapter 3)	all RBFs	Smoothness matching, regularization techniques
Convexity of CSRBFs (Chapter 4)	WD, MWD, WF and other CSRBFs	derive explicit formula for the lower bound of a convex interval
Solvability of CSRBFs with different shapes (Chapter 4)	WD, MWD, WF	HHRBF
HHRBF (Chapter 7)	CSRBFs	approximation error and conditioning estimate, applications to 3D large data sets
Near point preconditioning (Chapter 6)	all RBFs	Preconditioned implicit surface reconstruction, memory efficient preconditioners

References

[1] F. Acosta. Radial basis function and related models: An overview. *Signal Processing*, 45(1):37 – 58, 1995.

[2] P. R Amestoy, T. A. Davis, and I. S. Duff. An approximate minimum degree ordering algorithm. *SIAM Journal on Matrix Analysis and Applications*, 17(4):886–905, 1996.

[3] P. R. Amestoy, T. A. Davis, and I. S. Duff. Algorithm 837: Amd, an approximate minimum degree ordering algorithm. *ACM Transactions on Mathematical Software (TOMS)*, 30(3):381–388, 2004.

[4] R. Askey. Radial characteristic functions. Technical Report 1262, Uni. of Wiscosin, 1973. MRC Technical Sum. Report.

[5] K. Ball. Eigenvalues of Euclidean distance matrices. *J. Approx. Theory*, 68(1):74–82, 1992.

[6] K. Ball, N. Sivakumar, and J. D. Ward. On the sensitivity of radial basis interpolation to minimal data separation distance. *Constr. Approx.*, 8(4):401–426, 1992.

[7] B. J. C. Baxter. The asymptotic cardinal function of the multiquadratic $\phi(r) = (r^2 + c^2)^{1/2}$ as $c \to \infty$. *Comput. Math. Appl.*, 24(12):1–6, 1992. Advances in the theory and applications of radial basis functions.

[8] B. J. C. Baxter. Norm estimates for inverses of Toeplitz distance matrices. *J. Approx. Theory*, 79(2):222–242, 1994.

[9] B. J. C. Baxter. Preconditioned conjugate gradients, radial basis functions, and Toeplitz matrices. *Comput. Math. Appl.*, 43(3-5):305–318, 2002. Radial basis functions and partial differential equations.

[10] R. K. Beatson, J. B. Cherrie, and C. T. Mouat. Fast fitting of radial basis functions: methods based on preconditioned GMRES iteration. *Adv. Comput. Math.*, 11(2-3):253–270, 1999. Radial basis functions and their applications.

[11] R. K. Beatson, J. B. Cherrie, and D. L. Ragozin. Fast evaluation of radial basis functions: methods for four-dimensional polyharmonic splines. *SIAM J. Math. Anal.*, 32(6):1272–1310 (electronic), 2001.

[12] R. K. Beatson and W. A. Light. Fast evaluation of radial basis functions: methods for two-dimensional polyharmonic splines. *IMA J. Numer. Anal.*, 17(3):343–372, 1997.

[13] 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 (electronic), 2000.

[14] R. K. Beatson, W. E. Ong, and I. Rychkov. Faster fast evaluation of thin plate splines in two dimensions. *J. Comput. Appl. Math.*, 261:201–212, 2014.

[15] R. K. Beatson and M. J. D. Powell. An iterative method for thin plate spline interpolation that employs approximations to Lagrange functions. In *Numerical analysis 1993 (Dundee, 1993)*, volume 303 of *Pitman Res. Notes Math. Ser.*, pages 17–39. Longman Sci. Tech., Harlow, 1994.

[16] R. K. Beatson, M. J. D. Powell, and A. M. Tan. Fast evaluation of polyharmonic splines in three dimensions. *IMA J. Numer. Anal.*, 27(3):427–450, 2007.

[17] S. Bernstein. Sur la valeur les recherches récentes relatives à la meilleure approximation des fonctions continues par des polynômes. In *Proc. 5th. Intern. Math. Congress, v. 1,*, volume 1, pages 256–266, 1912.

[18] S. Bochner. Monotone Funktionen, Stieltjessche Integrale und harmonische Analyse. *Math. Ann.*, 108(1):378–410, 1933.

[19] J. P. Boyd. The uselessness of the fast Gauss transform for summing Gaussian radial basis function series. *J. Comput. Phys.*, 229(4):1311–1326, 2010.

[20] M. Bozzini, L. Lenarduzzi, M. Rossini, and R. Schaback. Interpolation by basis functions of different scales and shapes. *Calcolo*, 41(2):77–87, 2004.

[21] M. Bozzini, L. Lenarduzzi, M. Rossini, and R. Schaback. Interpolation with variably scaled kernels. *IMA J. Numer. Anal.*, pages 1–21, 2014.

[22] M. Bozzini, L. Lenarduzzi, and R. Schaback. Adaptive interpolation by scaled multiquadrics. *Adv. Comput. Math.*, 16(4):375–387, 2002.

[23] I. C. Briggs. Machine contouring using minimum curvature. *Geophysics*, 39(1):39–48, 1974.

[24] R. Brinks. On the convergence of derivatives of B-splines to derivatives of the Gaussian function. *Comput. Appl. Math.*, 27(1):79–92, 2008.

[25] D. Brown, L. Ling, E. J. 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.

[26] J. Buescu and A. C. Paixão. Eigenvalue distribution of Mercer-like kernels. *Math. Nachr.*, 280(9-10):984–995, 2007.

[27] J. Buescu and A. C. Paixão. Eigenvalue distribution of positive definite kernels on unbounded domains. *Integral Equations Operator Theory*, 57(1):19–41, 2007.

[28] M. B. Buhmann and N. Dyn. Spectral convergence of multiquadric interpolation. *Proc. Edinburgh Math. Soc. (2)*, 36(2):319–333, 1993.

[29] M. D. Buhmann. Multivariate cardinal interpolation with radial-basis functions. *Constr. Approx.*, 6(3):225–255, 1990.

[30] M. D. Buhmann. Radial functions on compact support. *Proc. Edinburgh Math. Soc. (2)*, 41(1):33–46, 1998.

[31] M. D. Buhmann. A new class of radial basis functions with compact support. *Math. Comp.*, 70(233):307–318 (electronic), 2001.

[32] M. D. Buhmann. *Radial basis functions: theory and implementations*, volume 12 of *Cambridge Monographs on Applied and Computational Mathematics*. Cambridge University Press, Cambridge, 2003.

[33] M. D. Buhmann. Semi-infinite cardinal interpolation with multiquadrics and beyond. *Adv. Comput. Math.*, 24(1-4):57–80, 2006.

[34] M. D. Buhmann and C. A. Micchelli. Multiply monotone functions for cardinal interpolation. *Adv. in Appl. Math.*, 12(3):358–386, 1991.

[35] M. D. Buhmann and C. A. Micchelli. Multiquadric interpolation improved. *Comput. Math. Appl.*, 24(12):21–25, 1992. Advances in the theory and applications of radial basis functions.

[36] J.C. Carr, R.K. Beatson, J.B. Cherrie, T.J. Mitchell, W.R. Fright, B.C. McCallum, and T.R. Evans. Reconstruction and representation of 3D objects with radial basis functions. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, SIGGRAPH '01, pages 67–76, New York, NY, USA, 2001. ACM.

[37] R. Cavoretto, A. De Rossi, M. Donatelli, and S. Serra-Capizzano. Spectral analysis and preconditioning techniques for radial basis function collocation matrices. *Numer. Linear Algebra Appl.*, 19(1):31–52, 2012.

[38] C. H. Chang and C. W. Ha. On eigenvalues of differentiable positive definite kernels. *Integral Equations Operator Theory*, 33(1):1–7, 1999.

[39] A. Y. Chanysheva. Positive-definite functions of a special type. *Vestnik Moskov. Univ. Ser. I Mat. Mekh.*, (5):71–74, 1990.

[40] J. Chen, L. Wang, and M. Anitescu. A fast summation tree code for Matérn kernel. *SIAM J. Sci. Comput.*, 36(1):A289–A309, 2014.

[41] K. Chen. *Matrix preconditioning techniques and applications*. Number 19. Cambridge University Press, 2005.

[42] A. Chernih, I. H. Sloan, and R. S. Womersley. Wendland functions with increasing smoothness converge to a Gaussian. *Adv. Comput. Math.*, 40(1):185–200, 2014.

[43] J. B. Cherrie, R. K. Beatson, and G. N. Newsam. Fast evaluation of radial basis functions: methods for generalized multiquadrics in $\mathbb{R}^n$. *SIAM J. Sci. Comput.*, 23(5):1549–1571, 2002.

[44] J.-P. Chilès and P. Delfiner. *Geostatistics*. Wiley Series in Probability and Statistics. John Wiley & Sons Inc., Hoboken, NJ, second edition, 2012. Modeling spatial uncertainty.

[45] J. A. Cochran. *The analysis of linear integral equations*. McGraw-Hill Book Co., New York, 1972. McGraw-Hill Series in Modern Applied Mathematics.

[46] A. J. Cole. Algorithm for the production of contour maps from scattered data. *Nature*, pages 92–94, 1968.

[47] S. J. Cole and R. J. Moore. Hydrological modelling using raingauge- and radar-based estimators of areal rainfall. *Journal of Hydrology*, 358(3?4):159 – 181, 2008.

[48] P. J. Collins. *Differential and Integral Equations*. Oxford mathematics. Oxford University Press, 2006.

[49] R. Courant and D. Hilbert. *Methods of mathematical physics. Vol. I*. Interscience Publishers, Inc., New York, N.Y., 1953.

[50] Timothy A Davis. *Direct Methods for Sparse Linear Systems (Fundamentals of Algorithms 2)*. Society for Industrial and Applied Mathematics, 2006.

[51] C. de Boor, K. Höllig, and S. Riemenschneider. *Box splines*, volume 98 of *Applied Mathematical Sciences*. Springer-Verlag, New York, 1993.

[52] C. de Boor and R.-Q. Jia. Controlled approximation and a characterization of the local approximation order. *Proc. Amer. Math. Soc.*, 95(4):547–553, 1985.

[53] C. de Boor and A. Ron. On multivariate polynomial interpolation. *Constr. Approx.*, 6(3):287–302, 1990.

[54] C. de Boor and A. Ron. Computational aspects of polynomial interpolation in several variables. *Math. Comp.*, 58(198):705–727, 1992.

[55] C. de Boor and A. Ron. The least solution for the polynomial interpolation problem. *Math. Z.*, 210(3):347–378, 1992.

[56] F. R. de Hoog. Review of Fredholm equations of the first kind. In *Application and numerical solution of integral equations (Proc. Sem., Australian Nat. Univ., Canberra, 1978)*, volume 6 of *Monographs Textbooks Mech. Solids Fluids: Mech. Anal.*, pages 119–134. Sijthoff & Noordhoff, Alphen aan den Rijn, 1980.

[57] Q. Deng and T. A. Driscoll. A fast treecode for multiquadric interpolation with varying shape parameters. *SIAM J. Sci. Comput.*, 34(2):A1126–A1140, 2012.

[58] R. Dixon and E. Spackman. Towards a four-dimensional analysis of meteorological data. *Nature*, 226:131–133, 1970.

[59] J. Dongarra and M. A. Heroux. Toward a new metric for ranking high performance computing systems. Technical report, Sandia National Laboratories, June 2013. SAND2013-4744.

[60] T. A. Driscoll and B. Fornberg. Interpolation in the limit of increasingly flat radial basis functions. *Comput. Math. Appl.*, 43(3-5):413–422, 2002. Radial basis functions and partial differential equations.

[61] I. S. Duff and K. Kaya. Preconditioners based on strong components. In *Householder Symposium XVIII on Numerical Linear Algebra*, page 63, 2011.

[62] I. S. Duff and K. Kaya. Preconditioners based on strong subgraphs. *Electronic Transactions on Numerical Analysis*, 40:225–248, 2013.

[63] Iain S Duff, Albert Maurice Erisman, and John Ker Reid. *Direct methods for sparse matrices*. Clarendon Press Oxford, 1986.

[64] Iain S Duff, AM Erisman, CW Gear, and John K Reid. Sparsity structure and gaussian elimination. *ACM SIGNUM Newsletter*, 23(2):2–8, 1988.

[65] N. Dyn and D. Levin. Iterative solution of systems originating from integral equations and surface interpolation. *SIAM J. Numer. Anal.*, 20(2):377–390, 1983.

[66] N. Dyn, D. Levin, and S. Rippa. Numerical procedures for surface fitting of scattered data by radial functions. *SIAM J. Sci. Statist. Comput.*, 7(2):639–659, 1986.

[67] H. Elman, D. Silvester, and A. Wathen. *Finite elements and fast iterative solvers: with applications in incompressible fluid dynamics*. Oxford University Press, 2005.

[68] V. A. Epanechnikov. Non-parametric estimation of a multivariate probability density. *Theory of Probability and Its Applications*, 14(1):153–158, 1969.

[69] P. Farrell and H. Wendland. RBF multiscale collocation for second order elliptic boundary value problems. *SIAM J. Numer. Anal.*, 51(4):2403–2425, 2013.

[70] G. E. Fasshauer. *Meshfree approximation methods with MATLAB*, volume 6 of *Interdisciplinary Mathematical Sciences*. World Scientific Publishing Co. Pte. Ltd., Hackensack, NJ, 2007. With 1 CD-ROM (Windows, Macintosh and UNIX).

[71] G. E. Fasshauer and J. W. Jerome. Multistep approximation algorithms: improved convergence rates through postconditioning with smoothing kernels. *Adv. Comput. Math.*, 10(1):1–27, 1999.

[72] G. E. Fasshauer and M. J. McCourt. Stable evaluation of Gaussian radial basis function interpolants. *SIAM J. Sci. Comput.*, 34(2):A737–A762, 2012.

[73] G. E. Fasshauer and J. G. Zhang. Preconditioning of radial basis function interpolation systems via accelerated iterated approximate moving least squares approximation. In *Progress on Meshless Methods*, pages 57–75. Springer, 2009.

[74] A. C. Faul, G. Goodsell, and M. J. D. Powell. A Krylov subspace algorithm for multiquadric interpolation in many dimensions. *IMA J. Numer. Anal.*, 25(1):1–24, 2005.

[75] A. C. Faul and M. J. D. Powell. Krylov subspace methods for radial basis function interpolation. In *Numerical analysis 1999 (Dundee)*, volume 420 of *Chapman & Hall/CRC Res. Notes Math.*, pages 115–141. Chapman & Hall/CRC, Boca Raton, FL, 2000.

[76] A. J. M. Ferreira and G. E. Fasshauer. Analysis of natural frequencies of composite plates by an RBF-pseudospectral method. *Composite Structures*, 79(2):202–210, 2007.

[77] J. C. Ferreira and V. A. Menegatto. Eigenvalues of integral operators defined by smooth positive definite kernels. *Integral Equations Operator Theory*, 64(1):61–81, 2009.

[78] R. Fletcher. Conjugate gradient methods for indefinite systems. In *Numerical Analysis*, pages 73–89. Springer, 1976.

[79] M. S. Floater and A. Iske. Multistep scattered data interpolation using compactly supported radial basis functions. *J. Comput. Appl. Math.*, 73(1-2):65–78, 1996.

[80] M. S. Floater and A. Iske. Thinning algorithms for scattered data interpolation. *BIT*, 38(4):705–720, 1998.

[81] N. Flyer and E. Lehto. Rotational transport on a sphere: local node refinement with radial basis functions. *J. Comput. Phys.*, 229(6):1954–1969, 2010.

[82] N. Flyer, E. Lehto, S. Blaise, G. B. Wright, and A. St-Cyr. A guide to RBF-generated finite differences for nonlinear transport: Shallow water simulations on a sphere. *J. Comput. Phys.*, 231(11):4078–4095, 2012.

[83] B. Fornberg, T.A. Driscoll, G. Wright, and R. Charles. Observations on the behavior of radial basis function approximations near boundaries. *Comput. Math. Appl.*, 43(3):473 – 490, 2002.

[84] B. Fornberg, N. Flyer, S. Hovde, and C. Piret. Locality properties of radial basis function expansion coefficients for equispaced interpolation. *IMA J. Numer. Anal.*, 28(1):121–142, 2008.

[85] B. Fornberg, E. Larsson, and N. Flyer. Stable computations with Gaussian radial basis functions. *SIAM J. Sci. Comput.*, 33(2):869–892, 2011.

[86] B. Fornberg and C. Piret. A stable algorithm for flat radial basis functions on a sphere. *SIAM J. Sci. Comput.*, 30(1):60–80, 2007/08.

[87] B. Fornberg, G. Wright, and E. Larsson. Some observations regarding interpolants in the limit of flat radial basis functions. *Comput. Math. Appl.*, 47(1):37–55, 2004.

[88] B. Fornberg and J. Zuev. The Runge phenomenon and spatially variable shape parameters in RBF interpolation. *Comput. Math. Appl.*, 54(3):379–398, 2007.

[89] M. Fornefett, K. Rohr, and H. S. Stiehl. Elastic registration of medical images using radial basis functions with compact support. In *Computer Vision and Pattern Recognition, 1999. IEEE Computer Society Conference on.*, volume 1. IEEE, 1999.

[90] R. Franke. A critical comparison of some methods for interpolation of scattered data. Technical report, Naval Postgraduate School, 1979.

[91] R. Franke. Scattered data interpolation: tests of some methods. *Math. Comp.*, 38(157):181–200, 1982.

[92] I. Fredholm. Sur une classe d'équations fonctionnelles. *Acta Math.*, 27(1):365–390, 1903.

[93] J. A. Freeman and D. Saad. Learning and generalization in radial basis function networks. *Neural Computation*, 7:1000–1020, 1995.

[94] R. W. Freund. A transpose-free quasi-minimal residual algorithm for non-Hermitian linear systems. *SIAM J. Sci. Comput.*, 14(2):470–482, 1993.

[95] R. W. Freund and N. M. Nachtigal. QMR: a quasi-minimal residual method for non-Hermitian linear systems. *Numer. Math.*, 60(3):315–339, 1991.

[96] E. Fuselier and G. B. Wright. Scattered data interpolation on embedded submanifolds with restricted positive definite kernels: Sobolev error estimates. *SIAM J. Numer. Anal.*, 50(3):1753–1776, 2012.

[97] E. H. Georgoulis, J. Levesley, and F. Subhan. Multilevel sparse kernel-based interpolation. *SIAM J. Sci. Comput.*, 35(2):A815–A831, 2013.

[98] T. Gneiting. Radial positive definite functions generated by Euclid's hat. *J. Multivariate Anal.*, 69(1):88–119, 1999.

[99] T. Gneiting. Compactly supported correlation functions. *J. Multivariate Anal.*, 83(2):493–508, 2002.

[100] N. A. Gumerov and R. Duraiswami. Fast radial basis function interpolation via preconditioned Krylov iteration. *SIAM J. Sci. Comput.*, 29(5):1876–1899, 2007.

[101] C. W. Ha. Eigenvalues of differentiable positive definite kernels. *SIAM J. Math. Anal.*, 17(2):415–419, 1986.

[102] A. Haar. Die Minkowskische Geometrie und die Annäherung an stetige Funktionen. *Math. Ann.*, 78(1):294–311, 1917.

[103] P. C. Hansen. The discrete picard condition for discrete ill-posed problems. *BIT Numerical Mathematics*, 30:658–672, 1990. 10.1007/BF01933214.

[104] P. C. Hansen. *Discrete inverse problems:Insight and algorithms*, volume 7 of *Fundamentals of Algorithms*. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 2010.

[105] R. L. Hardy. Multiquadric equations of topography and other irregular surfaces. *Journal of Geophysical Research*, 76(8):1905–1915, 1971.

[106] R. L. Hardy. Geodetic applications of multiquadric analysis. *AVN Allg. Vermess. Nachr.*, 79:389–406, 1972.

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

[108] R. L. Hardy and S. A. Nelson. A multiquadric-biharmonic representation and approximation of disturbing potential. *Geophysical Research Letters*, 13(1):18–21, 1986.

[109] E. Hille and J. D. Tamarkin. On the characteristic values of linear integral equations. *Acta Math.*, 57(1):1–76, 1931.

[110] T. Hofmann, B. Schölkopf, and A. J. Smola. Kernel methods in machine learning. *Ann. Statist.*, 36(3):1171–1220, 2008.

[111] Y. C. Hon and Z. Wu. Additive schwarz domain decomposition with radial basis approximation. In *Int. J. Appl. Math.* Citeseer, 2002.

[112] R. A. Horn and C. R. Johnson. *Matrix analysis*. Cambridge University Press, Cambridge, 1990. Corrected reprint of the 1985 original.

[113] S. Hubbert. *Radial basis function interpolation on the sphere*. PhD thesis, University of London, 2002.

[114] S. Hubbert. Closed form representations for a class of compactly supported radial basis functions. *Adv. Comput. Math.*, 36(1):115–136, 2012.

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

[116] Armin Iske. Characterization of function spaces associated to conditionally positive definite functions. *Mathematical Methods for Curves and Surfaces*, pages 265–270, 1995.

[117] A. A. Jamshidi and M. J. Kirby. Skew-radial basis function expansions for empirical modeling. *SIAM J. Sci. Comput.*, 31(6):4715–4743, 2009/10.

[118] R. Q. Jia and J. Lei. A new version of the Strang-Fix conditions. *J. Approx. Theory*, 74(2):221–225, 1993.

[119] E. J. Kansa. Multiquadrics—a scattered data approximation scheme with applications to computational fluid-dynamics. I. Surface approximations and partial derivative estimates. *Comput. Math. Appl.*, 19(8-9):127–145, 1990.

[120] E. J. Kansa. Multiquadrics—a scattered data approximation scheme with applications to computational fluid-dynamics. II. Solutions to parabolic, hyperbolic and elliptic partial differential equations. *Comput. Math. Appl.*, 19(8-9):147–161, 1990.

[121] E. J. Kansa and R. E. Carlson. Improved accuracy of multiquadric interpolation using variable shape parameters. *Comput. Math. Appl.*, 24(12):99–120, 1992. Advances in the theory and applications of radial basis functions.

[122] E. J. Kansa and Y. C. Hon. Circumventing the ill-conditioning problem with multiquadric radial basis functions: applications to elliptic partial differential equations. *Comput. Math. Appl.*, 39(7-8):123–137, 2000.

[123] Y. Katznelson. *An introduction to harmonic analysis*. John Wiley & Sons Inc., New York, 1968.

[124] R. Krasny and L. Wang. Fast evaluation of multiquadric RBF sums by a Cartesian treecode. *SIAM J. Sci. Comput.*, 33(5):2341–2355, 2011.

[125] R. Kress. *Linear integral equations*, volume 82 of *Applied Mathematical Sciences*. Springer-Verlag, New York, second edition, 1999.

[126] E. Larsson, E. Lehto, A. Heryudono, and B. Fornberg. Stable computation of differentiation matrices and scattered node stencils based on Gaussian radial basis functions. *SIAM J. Sci. Comput.*, 35(4):A2096–A2119, 2013.

[127] Q. T. Le Gia, I. H. Sloan, and Andrew J. Wathen. Stability and preconditioning for a hybrid approximation on the sphere. *Numer. Math.*, 118(4):695–711, 2011.

[128] Q. T. Le Gia, I. H. Sloan, and H. Wendland. Multiscale analysis in Sobolev spaces on the sphere. *SIAM J. Numer. Anal.*, 48(6):2065–2090, 2010.

[129] Q. T. Le Gia and T. Tran. An overlapping additive Schwarz preconditioner for interpolation on the unit sphere with spherical radial basis functions. *J. Complexity*, 26(5):552–573, 2010.

[130] D. Lee, P. Sao, R. Vuduc, and A. G. Gray. A distributed kernel summation framework for general-dimension machine learning. *Statistical Analysis and Data Mining*, 7(1):1–13, 2014.

[131] P. S. Lee, P. P. Lynn, and E. M. Shaw. Comparison of multiquadric surfaces for the estimation of areal rainfall. *Hydrological Sciences Bulletin*, 19(3):303–317, 1974.

[132] G. Letac and Q. I. Rahman. A factorisation of the Askey's characteristic function $(1 - \|t\|_{2n+1})_+^{n+1}$. *Ann. Inst. H. Poincaré Probab. Statist.*, 22(2):169–174, 1986.

[133] S. Li and W. K. Liu. Moving least-square reproducing kernel method. II. Fourier analysis. *Comput. Methods Appl. Mech. Engrg.*, 139(1-4):159–193, 1996.

[134] L. Ling and E. J. Kansa. Preconditioning for radial basis functions with domain decomposition methods. *Math. Comput. Modelling*, 40(13):1413–1427 (2005), 2004.

[135] L. Ling and E. J. Kansa. A least-squares preconditioner for radial basis functions collocation methods. *Adv. Comput. Math.*, 23(1-2):31–54, 2005.

[136] G. Little and J. B. Reade. Eigenvalues of analytic kernels. *SIAM J. Math. Anal.*, 15(1):133–136, 1984.

[137] J. W. H Liu. The role of elimination trees in sparse factorization. *SIAM Journal on Matrix Analysis and Applications*, 11(1):134–172, 1990.

[138] W. K. Liu, S. F. Li, and T. Belytschko. Moving least-square reproducing kernel methods. I. Methodology and convergence. *Comput. Methods Appl. Mech. Engrg.*, 143(1-2):113–154, 1997.

[139] T. Lyche and K. Mørken. Knot removal for parametric B-spline curves and surfaces. *Comput. Aided Geom. Design*, 4(3):217–230, 1987.

[140] W. R. Madych and S. A. Nelson. Bounds on multivariate polynomials and exponential error estimates for multiquadric interpolation. *J. Approx. Theory*, 70(1):94–114, 1992.

[141] J. C. Mairhuber. On Haar's theorem concerning Chebychev approximation problems having unique solutions. *Proc. Amer. Math. Soc.*, 7:609–615, 1956.

[142] J. C. Mairhuber. *I. On Haar's theorem concerning Chebychev approximation. II. Boundary values of continuous analytic functions.* ProQuest LLC, Ann Arbor, MI, 1959. Thesis (Ph.D.)–University of Pennsylvania.

[143] E. Marchandise, C. Piret, and J. F. Remacle. CAD and mesh repair with radial basis functions. *J. Comput. Phys.*, 231(5):2376–2387, 2012.

[144] M. Marius and G. L. David. Fast approximate nearest neighbors with automatic algorithm configuration. In *International Conference on Computer Vision Theory and Application VISSAPP'09)*, pages 331–340. INSTICC Press, 2009.

[145] M. Mathias. Uber positive Fourier-Integrale. *Math. Z.*, 16(1):103–125, 1923.

[146] V. Maźya and G. Schmidt. On approximate approximations using Gaussian kernels. *IMA J. Numer. Anal.*, 16(1):13–29, 1996.

[147] J. Mercer. Functions of positive and negative type, and their connection with the theory of integral equations. *Philosophical Transactions of the Royal Society of London. Series A, Containing Papers of a Mathematical or Physical Character*, pages 415–446, 1909.

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

[149] C. A. Micchelli and M. Pontil. Learning the kernel function via regularization. *J. Mach. Learn. Res.*, 6:1099–1125 (electronic), 2005.

[150] B. Mulgrew. Applying radial basis functions. *IEEE Signal Processing Magazine*, 13:50–65, 1996.

[151] F. J. Narcowich, R. Schaback, and J. D. Ward. Multilevel interpolation and approximation. *Appl. Comput. Harmon. Anal.*, 7(3):243–261, 1999.

[152] F. J. Narcowich, N. Sivakumar, and J. D. Ward. On condition numbers associated with radial-function interpolation. *J. Math. Anal. Appl.*, 186(2):457–485, 1994.

[153] F. J. Narcowich and J. D. Ward. Norms of inverses for matrices associated with scattered data. In *Curves and surfaces (Chamonix-Mont-Blanc, 1990)*, pages 341–348. Academic Press, Boston, MA, 1991.

[154] F. J. Narcowich and J. D. Ward. Norm estimates for the inverses of a general class of scattered-data radial-function interpolation matrices. *J. Approx. Theory*, 69(1):84–109, 1992.

[155] T. Poggio and F. Girosi. Regularization algorithms for learning that are equivalent to multilayer networks. *Science*, 247(4945):978–982, 1990.

[156] C. E. Rasmussen and C. K. I. Williams. *Gaussian processes for machine learning*, volume 1. MIT press Cambridge, MA, 2006.

[157] J. B. Reade. Asymptotic behaviour of eigenvalues of certain integral equations. *Proc. Edinburgh Math. Soc. (2)*, 22(2):137–144, 1979.

[158] J. B. Reade. Eigenvalues of Lipschitz kernels. *Math. Proc. Cambridge Philos. Soc.*, 93(1):135–140, 1983.

[159] J. B. Reade. Eigenvalues of positive definite kernels. *SIAM J. Math. Anal.*, 14(1):152–157, 1983.

[160] J. B. Reade. Eigenvalues of positive definite kernels. II. *SIAM J. Math. Anal.*, 15(1):137–142, 1984.

[161] J. B. Reade. Eigenvalues of smooth kernels. *Math. Proc. Cambridge Philos. Soc.*, 95(1):135–140, 1984.

[162] J. B. Reade. On the sharpness of Weyl's estimate for eigenvalues of smooth kernels. *SIAM J. Math. Anal.*, 16(3):548–550, 1985.

[163] J. B. Reade. Positive definite C^p kernels. *SIAM J. Math. Anal.*, 17(2):420–421, 1986.

[164] J. B. Reade. On the sharpness of Weyl's estimates for eigenvalues of smooth kernels. II. *SIAM J. Math. Anal.*, 19(3):627–631, 1988.

[165] J. B. Reade. Eigenvalues of smooth positive definite kernels. *Proc. Edinburgh Math. Soc. (2)*, 35(1):41–45, 1992.

[166] C. Rieger. *Sampling Inequalities and Applications*. PhD thesis, University at Gottingen, 2008.

[167] C. Rieger and B. Zwicknagl. Sampling inequalities for infinitely smooth functions, with applications to interpolation and machine learning. *Adv. Comput. Math.*, 32(1):103–129, 2010.

[168] S. D. Riemenschneider and N. Sivakumar. On cardinal interpolation by Gaussian radial-basis functions: properties of fundamental functions and estimates for Lebesgue constants. *J. Anal. Math.*, 79:33–61, 1999.

[169] S. Rippa. An algorithm for selecting a good value for the parameter c in radial basis function interpolation. *Advances in Computational Mathematics*, 11:193–210, 1999.

[170] G. Roussos and Brad J. C. Baxter. Rapid evaluation of radial basis functions. *J. Comput. Appl. Math.*, 180(1):51–70, 2005.

[171] Y. Saad. *Iterative methods for sparse linear systems*. PWS Publishing Company, second edition, 1996.

[172] Y. Saad and M. H. Schultz. GMRES: a generalized minimal residual algorithm for solving nonsymmetric linear systems. *SIAM J. Sci. Statist. Comput.*, 7(3):856–869, 1986.

[173] D. T. Sandwell. Biharmonic spline interpolation of geos-3 and seasat altimeter data. *Geophysical Research Letters*, 14(2):139–142, 1987.

[174] R. H. Sanford and R. L. Hardy. Solvability and multiquadric analysis as applied to investigations of vertical crustal movements. *Tectonophysics*, 52(1?4):139 – 155, 1979. Recent Crustal Movements.

[175] S. A. Sarra and D. Sturgill. A random variable shape parameter strategy for radial basis function approximation methods. *Eng. Anal. Bound. Elem.*, 33(11):1239–1245, 2009.

[176] R. Schaback. Lower bounds for norms of inverses of interpolation matrices for radial basis functions. *J. Approx. Theory*, 79(2):287–306, 1994.

[177] R. Schaback. Error estimates and condition numbers for radial basis function interpolation. *Adv. Comput. Math.*, 3(3):251–264, 1995.

[178] R. Schaback. Multivariate interpolation by polynomials and radial basis functions. *Constr. Approx.*, 21(3):293–317, 2005.

[179] R. Schaback. The missing Wendland functions. *Adv. Comput. Math.*, 34(1):67–81, 2011.

[180] R. Schaback and H. Wendland. Characterization and construction of radial basis functions. In *Multivariate approximation and applications*, pages 1–24. Cambridge Univ. Press, Cambridge, 2001.

[181] R. Schaback and Z. Wu. Operators on radial functions. *J. Comput. Appl. Math.*, 73(1-2):257–270, 1996.

[182] I. J. Schoenberg. Remarks to Maurice Fréchet's article "Sur la définition axiomatique d'une classe d'espace distanciés vectoriellement applicable sur l'espace de Hilbert". *Ann. of Math. (2)*, 36(3):724–732, 1935.

[183] I. J. Schoenberg. On certain metric spaces arising from Euclidean spaces by a change of metric and their imbedding in Hilbert space. *Ann. of Math. (2)*, 38(4):787–793, 1937.

[184] I. J. Schoenberg. Metric spaces and completely monotone functions. *Ann. of Math. (2)*, 39(4):811–841, 1938.

[185] I. J. Schoenberg. *Cardinal spline interpolation*. Society for Industrial and Applied Mathematics, Philadelphia, Pa., 1973. Conference Board of the Mathematical Sciences Regional Conference Series in Applied Mathematics, No. 12.

[186] C. Shu, H. Ding, H. Q. Chen, and T. G. Wang. An upwind local RBF-DQ method for simulation of inviscid compressible flows. *Computer Methods in Applied Mechanics and Engineering*, 194(18):2001–2017, 2005.

[187] G. L. G. Sleijpen, H. A. van der Vorst, and D. R. Fokkema. BiCGstab(l) and other hybrid Bi-CG methods. *Numer. Algorithms*, 7(1):75–109, 1994.

[188] F. Smithies. The eigenvalue and singular values of integral equations. *Proc. London Math. Soc.*, 43:255–279, 1937.

[189] P. Sonneveld. CGS, a fast Lanczos-type solver for nonsymmetric linear systems. *SIAM J. Sci. Statist. Comput.*, 10(1):36–52, 1989.

[190] S. E. Stead. Estimation of gradients from scattered data. *Rocky Mountain Journal of Mathematics*, 14(1):265–280, 03 1984.

[191] J. Strain. The fast Gauss transform with variable scales. *SIAM J. Sci. Statist. Comput.*, 12(5):1131–1139, 1991.

[192] G. Strang and G. Fix. A fourier analysis of the finite element variational method. In *Constructive aspects of functional analysis*, pages 793–840. Springer, 1971.

[193] A. Townsend and H. Wendland. Multiscale analysis in Sobolev spaces on bounded domains with zero boundary values. *IMA J. Numer. Anal.*, 33(3):1095–1114, 2013.

[194] T. Tran, Q. T. Le Gia, I. H. Sloan, and E. P. Stephan. Preconditioners for pseudodifferential equations on the sphere with radial basis functions. *Numer. Math.*, 115(1):141–163, 2010.

[195] L. N. Trefethen. *Approximation Theory and Approximation Practice*. Applied Mathematics. Society for Industrial and Applied Mathematics, 2013.

[196] M. Unser, A. Aldroubi, and M. Eden. On the asymptotic convergence of B-spline wavelets to Gabor functions. *IEEE Trans. Inform. Theory*, 38(2, part 2):864–872, 1992.

[197] H. A. van der Vorst. Bi-CGSTAB: a fast and smoothly converging variant of Bi-CG for the solution of nonsymmetric linear systems. *SIAM J. Sci. Statist. Comput.*, 13(2):631–644, 1992.

[198] H. A. van der Vorst. *Iterative Krylov methods for large linear systems*, volume 13. Cambridge University Press, 2003.

[199] J. M. Varah. A practical examination of some numerical methods for linear discrete ill-posed problems. *SIAM Rev.*, 21(1):100–111, 1979.

[200] J. M. Varah. Pitfalls in the numerical solution of linear ill-posed problems. *SIAM J. Sci. Statist. Comput.*, 4(2):164–176, 1983.

[201] R. S. Varga. *Matrix iterative analysis*, volume 27 of *Springer Series in Computational Mathematics*. Springer-Verlag, Berlin, expanded edition, 2000.

[202] R. S. Varga, E. B. Saff, and V. Mehrmann. Incomplete factorizations of matrices and connections with H-matrices. *SIAM J. Numer. Anal.*, 17(6):787–793, 1980.

[203] J. von Neumann and I. J. Schoenberg. Fourier integrals and metric geometry. *Trans. Amer. Math. Soc.*, 50:226–251, 1941.

[204] L. Wasserman. *All of statistics*. Springer Texts in Statistics. Springer-Verlag, New York, 2004. A concise course in statistical inference.

[205] A.J. Wathen and S. Zhu. On spectral distribution of kernel matrices related to radial basis functions. *Numer. Algor.*, xx:xxx–xxx, 2015. doi:10.1007/s11075-015-9970-0.

[206] H. Wendland. Ein beitrag zur interpolation mit radialen basisfunktionen. Master's thesis, University of Göttingen, 1994.

[207] H. Wendland. Piecewise polynomial, positive definite and compactly supported radial functions of minimal degree. *Adv. Comput. Math.*, 4(4):389–396, 1995.

[208] H. Wendland. *Scattered data approximation*, volume 17 of *Cambridge Monographs on Applied and Computational Mathematics*. Cambridge University Press, Cambridge, 2005.

[209] H. Wendland. Multiscale analysis of Sobolev spaces on bounded domains. *Numer. Math.*, 116(3):493–517, 2010.

[210] H. Weyl. Das asymptotische verteilungsgesetz der eigenwerte linearer partieller differentialgleichungen (mit einer anwendung auf die theorie der hohlraumstrahlung). *Mathematische Annalen*, 71:441–479, 1912. 10.1007/BF01456804.

[211] Y. L. Wu and C. Shu. Development of RBF-DQ method for derivative approximation and its application to simulate natural convection in concentric annuli. *Computational Mechanics*, 29(6):477–485, 2002.

[212] Z. Wu. Hermite-Birkhoff interpolation of scattered data by radial basis functions. *Approx. Theory Appl.*, 8(2):1–10, 1992.

[213] Z. Wu. Compactly supported positive definite radial functions. *Adv. Comput. Math.*, 4(3):283–292, 1995.

[214] Z. Wu. Compactly supported radial functions and the strang-fix condition. *Applied mathematics and computation*, 84(2):115–124, 1997.

[215] Z. Wu and J. Liu. Generalized Strang-Fix condition for scattered data quasi-interpolation. *Adv. Comput. Math.*, 23(1-2):201–214, 2005.

[216] L. Ying. A kernel independent fast multipole algorithm for radial basis functions. *Journal of Computational Physics*, 213(2):451 – 457, 2006.

[217] L. Ying, G. Biros, and D. Zorin. A kernel-independent adaptive fast multipole algorithm in two and three dimensions. *J. Comput. Phys.*, 196(2):591–626, 2004.

[218] R. Yokota, L. A. Barba, and M. G. Knepley. PetRBF—a parallel $O(N)$ algorithm for radial basis function interpolation with Gaussians. *Comput. Methods Appl. Mech. Engrg.*, 199(25-28):1793–1804, 2010.

[219] F. Zhou, J. Zhang, X. Sheng, and G. Li. Shape variable radial basis function and its application in dual reciprocity boundary face method. *Eng. Anal. Bound. Elem.*, 35(2):244–252, 2011.

[220] H. Zhu, C. K. I. Williams, R. J. Rohwer, and M. Morciniec. Gaussian regression and optimal finite dimensional linear models. In C.M. Bishop, editor, *Neural Networks and Machine Learning*. Springer-Verlag / Berlin, 1998.

[221] S. Zhu, T. Gu, and X. Liu. Minimizing synchronizations in sparse iterative solvers for distributed supercomputers. *Comput. Math. Appl.*, 67(1):199–209, 2014.

[222] S. Zhu and A. J. Wathen. Fast sparse kernel summation on Cartesian grids. Technical report, The University of Oxford, March 2014.

[223] S. Zhu and A.J. Wathen. Convexity and solvability for compactly supported radial basis functions with different shapes. *J. Sci. Comput.*, pages 1–23, 2015. doi:10.1007/s10915-014-9919-9.