Scientific Computing with Radial Basis Functions

Edited by
C.S. Chen
University of Southern Mississippi, USA

Y.C. Hon
City University of Hong Kong, China

R.A. Schaback
University of Göttingen, Germany

Contents

page v
Preface vi

1 Introduction 1
1.1 Radial Basis Functions 1
1.2 Multivariate Interpolation and Positive Definiteness 2
1.3 Stability and Scaling 5
1.4 Solving Partial Differential Equations 6
1.5 Comparison of Strong and Weak Problems 7
1.6 Collocation Techniques 9
1.7 Method of Fundamental Solutions 12
1.8 Method of Particular Solutions 13
1.9 Time–dependent Problems 14
1.10 Lists of Radial Basis Functions 14

2 Basic Techniques for Function Recovery 17
2.1 Interpolation of Lagrange Data 17
2.2 Interpolation of Mixed Data 19
2.3 Error Behavior 21
2.4 Stability 22
2.5 Regularization 26
2.6 Scaling 33
2.7 Practical Rules 37
2.8 Large Systems: Computational Complexity 38
2.9 Sensitivity to Noise 41
2.10 Time–dependent Functions 44

3 Collocation Techniques 46
3.1 High Dimensional Problems 47
3.2 Irregular Boundary Shape Problems 50

3.3 Transport Problems 55
3.4 Free Boundary Value Problems 59
3.5 Moving Boundary Value Problems 68

4 The Method of Fundamental Solutions 78
4.1 Introduction 78
4.2 Theoretical Background 80
4.3 Fundamental Solutions 85
4.4 Static Implementation 86
4.5 Interlude: Stability Issues 100
4.6 Dynamic Implementation 116
4.7 Modified Implementation 121
4.8 Special Problems 123
4.9 Inverse Problems 137

5 The Method of Particular Solutions 150
5.1 Introduction 150
5.2 The Dual Reciprocity Method (DRM) 151
5.3 The Method of Particular Solutions (MPS) 152
5.4 Approximate Particular Solutions 153
5.5 Particular Solutions for the Laplace operator 158
5.6 Particular Solutions for Biharmonic Equations 176
5.7 Particular Solutions for Helmholtz Operators 177
5.8 Multiscale Techniques 204
5.9 Solving Large Problems on Circular Points 226

6 Time–Dependent Problems 240
6.1 Method of Laplace Transforms 240
6.2 Time–stepping Methods 245
6.3 Operator-Splitting Methods 257
6.4 Wave Equation 260

References 265
Index 280

This book is dedicated to our families,

as our constant sources of joy and support.

Preface

In recent decades **radial basis functions** have proven to be very useful in solving problems in Scientific Computing which arise in application areas like

- Computational Mechanics, including elasticity and stress analysis,
- Fluid dynamics, including shallow water equations, reaction-diffusion and convection-advection problems,
- Computer Graphics and image analysis, including shape modeling and animated deformations, and
- Economics, including pricing of options.

These seemingly unrelated applications are linked via certain common mathematical concepts and problems like

- Recovery of functions from scattered data,
- Meshless methods solving partial differential equations (PDEs),
- Ill–posed and inverse problems,
- Neural networks, and
- Learning algorithms

which can be handled easily and successfully by radial basis functions. The mathematical reasons for this are covered by books of M.D. Buhmann [Buh03] and H. Wendland [Wen05] providing the necessary theoretical background, while a new book of G.E. Fasshauer [Fas07] additionally covers MATLAB implementations of algorithms.

In contrast to these, this text focuses almost entirely on applications, with the goal of helping scientists and engineers apply radial basis functions successfully. This book is intended to meet this need. We do not assume that readers are familiar with the mathematical peculiarities of radial basis functions. We do, however, assume some knowledge of the partial differential equations arising in applications, and we include many computational examples without dealing with implementation issues.

In preparing this text, we soon realized that it was impossible to cover all of the interesting techniques and applications, as summarized in a recent Acta Numerica article [SW06]. We decided to leave out neural networks and kernel-based learning algorithms completely, since the latter currently supersede the former and are covered in several books including applications [CST00, SS02, STC04]. Instead we focused on meshless methods

- for reconstruction of multivariate functions from scattered data, and
- for solving partial differential equations.

Even within these seemingly small areas we had to confine ourselves to a few core techniques which include Kansa's method, the method of fundamental solutions, the method of particular solutions, etc. These techniques allowed us to extend the radial basis functions to numerically solving a large class of partial differential equations without mesh generation. In particular, we devoted a great deal of our effort to the derivation of particular solutions using radial basis functions for certain differential operators. Furthermore, this book should enable the reader to follow the references to other methods not covered here and to keep up with the pace of new developments in the area. To this end, we included some unpublished material at various places.

This manuscript has been used as lecture notes to teach at the graduate special topic courses in scientific computing at the University of Nevada, Las Vegas (UNLV) during 2004–2005 and the University of Southern Mississippi (USM) during 2006–2007. During the course of teaching these classes, we have been fortunate that our students have enthusiastically given us a great amount of feedback and have allowed us to constantly make revision of the content of the book. It is worth mentioning that, due to these courses, two Master's theses in meshless methods were produced at UNLV and two potential Ph.D. theses in a similar topic at USM are currently being done. We are pleased to see some of our students were able to adopt new concepts they learn from the book, and had turned them into research projects, and presenting their results in conferences and conference proceedings. As such, we believe the book is suitable for a one year graduate or post-graduate course in the area of scientific computing.

We would like to acknowledge the editorial help from Drs. John Dudley and Joseph Kolibal at USM. We also appreciate Dr. Michael Golberg, and Professor Andreas Karageorghis and Professor Greame Fairweather for providing important references in the method of fundamental solutions. The comments and suggestions from Professor Karageorghis to improve our book is especially helpful. Part of this monograph was completed during the course of Dr. Chen's summer visit in 2004 to Professor Satya Atluri at the University of California, Irvine. The kind support of Professor Atluri is greatly appreciated. The institutional support from the University of Southern Mississippi, City University of Hong Kong, and the University of Göttingen for providing resources and facilities to complete this project is also critical and we would like to express our gratitude for this support.

November 24, 2014 C.S. Chen, Y.C. Hon, and R.A. Schaback

1
Introduction

1.1 Radial Basis Functions

Scientific Computing with Radial Basis Functions focuses on the *recovery* of *unknown functions* from *known data*. The functions are multivariate in general, and they may be solutions of partial differential equations satisfying certain additional conditions. However, the reconstruction of multivariate functions from data may cause problems if the space furnishing the "trial" functions is not fixed in advance, but is data–dependent [Mai56]. **Finite elements** (see e.g. [Bra01, BS02]) provide such data–dependent spaces. They are defined as piecewise polynomial functions on regular triangularizations.

To avoid triangularizations, re-meshing and other geometric programming efforts, **meshless methods** have been suggested [BKO⁺96]. This book focuses on a special class of meshless techniques for generating data–dependent spaces of multivariate functions. The spaces are spanned by shifted and scaled instances of **radial basis functions** (**RBF**) like the **multiquadric** [Har71]

$$\mathbf{x} \mapsto \Phi(\mathbf{x}) := \sqrt{1 + \|\mathbf{x}\|_2^2},\ \mathbf{x} \in I\!\!R^d$$

or the **Gaussian**

$$\mathbf{x} \mapsto \Phi(\mathbf{x}) := \exp(-\|\mathbf{x}\|_2^2),\ \mathbf{x} \in I\!\!R^d.$$

These functions are multivariate, but reduce to a scalar function of the Euclidean norm $\|\mathbf{x}\|_2$ of their vector argument $\mathbf{x}$, i.e. they are **radial** in the sense

$$\Phi(\mathbf{x}) = \phi(\|\mathbf{x}\|_2) = \phi(r),\ \mathbf{x} \in I\!\!R^d$$

for the "radius" $r = \|\mathbf{x}\|_2$ with a scalar function $\phi\ :\ I\!\!R \to I\!\!R$. This makes their use for high–dimensional reconstruction problems very efficient, and it induces invariance under orthogonal transformations.

Recovery of functions from meshless data is then made by **trial functions** u which are linear combinations

$$u(\mathbf{x}) := \sum_{k=1}^{n} \alpha_k \phi(\|\mathbf{x} - \mathbf{y}_k\|_2) \tag{1.1.1}$$

of translates $\phi(\|\mathbf{x} - \mathbf{y}_k\|_2)$ of a single radial basis function. The translations are specified by vectors $\mathbf{y}_1, \ldots, \mathbf{y}_n$ of $I\!R^d$, sometimes called **centers** or **trial points**, without any special assumptions on their number or geometric position. This is why the methods of this book are truly "meshless." In certain cases one has to add multivariate polynomials in $\mathbf{x}$ to the linear combinations in (1.1.1), but we postpone these details.

Our main goal is to show how useful radial basis functions are in applications, in particular for solving partial differential equations (**PDE**) of science and engineering. Therefore we keep the theoretical background to a minimum, referring to recent books [Buh03, Wen05, Fas07] on radial basis functions whenever possible. Furthermore, we have to ignore generalizations of radial basis functions to **kernels**. These arise in many places, including probability and learning theory, and they are surveyed in [SW06]. The rest of this chapter gives an overview of the applications we cover in this book.

1.2 Multivariate Interpolation and Positive Definiteness

The simplest case of reconstruction of a d–variate unknown function u^* from data occurs when only a finite number of data in the form of values $u^*(\mathbf{x}_1), \ldots, u^*(\mathbf{x}_m)$ at arbitrary locations $\mathbf{x}_1, \ldots, \mathbf{x}_m$ in $I\!R^d$ forming a set $X := \{\mathbf{x}_1, \ldots, \mathbf{x}_m\}$ are known. In contrast to the n **trial centers** $\mathbf{y}_1, \ldots, \mathbf{y}_n$ of (1.1.1), the m data locations $\mathbf{x}_1, \ldots, \mathbf{x}_m$ are called **test points** or **collocation points** in later applications. To calculate a trial function u of the form (1.1.1) which reproduces the data $u^*(\mathbf{x}_1), \ldots, u^*(\mathbf{x}_m)$ well, we have to solve the $m \times n$ linear system

$$\sum_{k=1}^{n} \alpha_k \phi(\|\mathbf{x}_i - \mathbf{y}_k\|_2) \approx u^*(\mathbf{x}_i),\ 1 \le i \le m \tag{1.2.1}$$

for the n coefficients $\alpha_1, \ldots, \alpha_n$. Matrices with entries $\phi(\|\mathbf{x}_i - \mathbf{y}_k\|_2)$ will occur at many places in the book, and they are called **kernel matrices** in **machine learning**.

Of course, users will usually make sure that $m \ge n$ holds by picking at least as many test points as trial centers, but the easiest case will occur when the centers $\mathbf{y}_k$ of trial functions (1.1.1) are chosen to be *identical* to the data locations $\mathbf{x}_j$ for $1 \le j \le m = n$. If there is no noise in the data, it then makes sense to reconstruct u^* by a function u of the form (1.1.1) by enforcing the exact interpolation conditions

$$u^*(\mathbf{x}_j) = \sum_{k=1}^{n} \alpha_j \phi(\|\mathbf{x}_j - \mathbf{x}_k\|_2),\ 1 \le j \le m = n. \tag{1.2.2}$$

This is a system of m linear equations in $n = m$ unknowns $\alpha_1, \ldots, \alpha_n$ with a symmetric kernel matrix

$$\mathbf{A}_X := (\phi(\|\mathbf{x}_j - \mathbf{x}_k\|_2))_{1 \le j,k \le m} \tag{1.2.3}$$

In general, solvability of such a system is a serious problem, but one of the central features of kernels and radial basis functions is to make this problem obsolete via

Definition 1.2.4 *A radial basis function ϕ on $[0, \infty)$ is* **positive definite** *on $I\!R^d$, if for all choices of sets $X := \{\mathbf{x}_1, \ldots, \mathbf{x}_m\}$ of finitely many points $\mathbf{x}_1, \ldots, \mathbf{x}_m \in I\!R^d$ and arbitrary m the symmetric $m \times m$ matrices $\mathbf{A}_X$ of* (1.2.3) *are positive definite.*

Consequently, solvability of the system (1.2.2) is guaranteed if ϕ satisfies the above definition. This holds for several standard radial basis functions provided in Table 1.1, but users must be aware that problems may occur when using other scalar functions such as $\exp(-r)$. A more detailed list of radial basis functions will follow later on page 15.

Name	$\phi(r)$
Gaussian	$\exp(-r^2)$
Inverse multiquadrics	$(1 + r^2)^{\beta/2},\ \beta < 0$
Matern/Sobolev	$K_\nu(r) r^\nu,\ \nu > 0$

Table 1.1. *Positive definite radial basis functions*

But there are some very useful radial basis functions which fail to be positive definite. In such cases one has to add polynomials of a certain maximal degree to the trial functions of (1.1.1). Let P^d_{Q-1} denote the space spanned by all d-variate polynomials of degree up to $Q - 1$, and pick a basis $p_1, \ldots, p_q$ of this space. The dimension q then comes out to be $q = \binom{Q-1+d}{d}$, and the trial functions of (1.1.1) are augmented to

$$u(\mathbf{x}) := \sum_{k=1}^{n} \alpha_k \phi(\|\mathbf{x} - \mathbf{y}_k\|_2) + \sum_{\ell=1}^{q} \beta_\ell p_\ell(\mathbf{x}). \tag{1.2.5}$$

Now there are q additional degrees of freedom, but these are removed by q additional homogeneous equations

$$\sum_{k=1}^{n} \alpha_k p_\ell(\mathbf{x}_k) = 0,\ 1 \le \ell \le q \tag{1.2.6}$$

restricting the coefficients $\alpha_1, \dots, \alpha_n$ in (1.2.5). Unique solvability of the extended system

$$\begin{array}{rcll} \displaystyle\sum_{k=1}^{n} \alpha_k \phi(\|\mathbf{x}_j - \mathbf{y}_k\|_2) + \sum_{\ell=1}^{q} \beta_\ell p_\ell(\mathbf{x}_j) & = & u(\mathbf{x}_j), & 1 \le j \le n \\ \displaystyle\sum_{k=1}^{n} \alpha_k p_\ell(\mathbf{x}_k) & = & 0, & 1 \le \ell \le q \end{array} \tag{1.2.7}$$

is assured if

$$p(\mathbf{x}_k) = 0 \text{ for all } 1 \le k \le n \text{ and } p \in P^d_{Q-1} \text{ implies } p = 0. \tag{1.2.8}$$

This is the proper setting for **conditionally positive definite** radial basis functions of **order** Q, and in case $Q = 0$ it will coincide with what we had before, since then $q = 0$ holds, (1.2.6) is empty, and (1.2.5) reduces to (1.1.1). We leave details of this to the next chapter, but we want the reader to be aware of the necessity of adding polynomials in certain cases. Table 1.2 provides a selection of the most useful conditionally positive definite functions, and again we refer to Table 1.3 on page 15 for other radial basis functions.

Name	$\phi(r)$	Q	condition
multiquadric	$(-1)^{\lceil\beta/2\rceil}(1+r^2)^{\beta/2}$	$\lceil\beta/2\rceil$	$\beta > 0,\ \beta \notin 2I\!N$
polyharmonic	$(-1)^{\lceil\beta/2\rceil}r^\beta$	$\lceil\beta/2\rceil$	$\beta > 0,\ \beta \notin 2I\!N$
polyharmonic	$(-1)^{1+\beta/2}r^\beta \log r$	$1+\beta/2$	$\beta > 0,\ \beta \in 2I\!N$
thin–plate spline	$r^2 \log r$	2	

Table 1.2. *Conditionally positive definite radial basis functions*

1.3 Stability and Scaling

Solving the system (1.2.2) is easy to program, and it is always possible if ϕ is a positive definite radial basis function. But it also can cause practical problems, since it may be badly conditioned and is non–sparse in case of globally non–vanishing radial basis functions. To handle bad conditions of moderately large systems, one can rescale the radial basis function used, or one can calculate an approximate solution by solving a properly chosen subsystem. Certain decomposition and preconditioning techniques are also possible, but details will be postponed to the next chapter.

In absence of noise, systems of the form (1.2.2) or (1.2.7) will in most cases have a very good **approximate solution**, because the unknown function u providing the right-hand side data can usually be well approximated by the trial functions used in (1.1.1) or (1.2.5). This means that even for high condition numbers there is a good reproduction of the right-hand side by a linear combination of the columns of the matrix. The coefficients are in many cases not very interesting, since users want to have a good trial function recovering the data well, whatever the coefficients are. Thus users can apply specific numerical techniques like **singular value decomposition** or **optimization algorithms** to get useful results in spite of bad conditions. We shall supply details in the next chapter, but we advise users not to use primitive solution methods for their linear systems.

For extremely large systems, different techniques are necessary. Even if a solution can be calculated, the evaluation of $u(\mathbf{x})$ in (1.1.1) at a single point $\mathbf{x}$ has $\mathcal{O}(n)$ complexity, which is not tolerable in general. This is why some **localization** is necessary, cutting the evaluation complexity at $\mathbf{x}$ down to $\mathcal{O}(1)$. At the same time, such a localization will make the system matrix sparse, and efficient solution techniques like preconditioned conjugate gradients become available. Finite elements achieve this by using a localized basis, and the same method also works for radial basis functions, if scaled functions with compact support are used. Fortunately, positive definite radial functions with compact support exist for all space dimensions and smoothness requirements [Wu95, Wen95, Buh98]. The most useful example is Wendland's function

$$\phi(r) = \begin{cases} (1-r)^4(1+4r), & 0 \leq r \leq 1, \\ 0, & r \geq 1, \end{cases}$$

which is positive definite in $I\!R^d$ for $d \leq 3$ and twice differentiable in $\mathbf{x}$

when $r = \|\mathbf{x}\|_2$ (see Table 1.1 and other cases in Table 1.3 on page 15). Other localization techniques use fast **multipole methods** [BGP96, BG97] or a **partition of unity** [Wen02]. This technique originated from **finite elements** [MB96, BM97], where it served to patch local finite element systems together. It superimposes local systems in general, using smooth weight functions, and thus it also works well if the local systems are made up using radial basis functions.

However, all localization techniques require some additional geometric information, e.g. a list of centers $\mathbf{y}_k$ which are close to any given point $\mathbf{x}$. Thus the elimination of triangulations will, in case of huge systems, bring problems of Computational Geometry through the back door.

A particularly local interpolation technique, which does not solve any system of equations but can be efficiently used for any local function reconstruction process, is the method of **moving least squares** [LS81, Lev98, Wen01]. We have to ignore it here. Chapter 2 will deal with radial basis function methods for interpolation and approximation in quite some detail, including methods for solving large systems in Section 2.8.

1.4 Solving Partial Differential Equations

With some modifications, the above observations will carry over to solving partial differential equations. In this introduction, we confine ourselves to a **Poisson problem** on a bounded domain $\Omega \subset I\!R^3$ with a reasonably smooth boundary $\partial\Omega$. It serves as a model case for more general partial differential equations of science and engineering that we have in mind. If functions f^Ω on the domain Ω and f^Γ on the boundary $\Gamma := \partial\Omega$ are given, a function u on $\Omega \cup \Gamma$ with

$$\begin{array}{rcll} -\Delta u & = & f^\Omega & \text{in } \Omega \\ u & = & f^\Gamma & \text{in } \Gamma \end{array} \tag{1.4.1}$$

is to be constructed, where Δ is the **Laplace operator**

$$\Delta u = \frac{\partial^2 u}{\partial x_1^2} + \frac{\partial^2 u}{\partial x_2^2} + \frac{\partial^2 u}{\partial x_3^2}$$

in **Cartesian coordinates** $\mathbf{x} = (x_1, x_2, x_3)^T \in I\!R^3$. This way the problem is completely posed in terms of evaluations of functions and derivatives, without any integrations. However, this requires taking second derivatives of u, and a careful mathematical analysis shows that there are cases where this assumption is questionable. It holds only under certain additional assumptions, and this is why the above formulation is called a **strong form**. Except for the next section, we shall deal exclusively with methods for solving partial differential equations in strong form.

A **weak form** is obtained by multiplication of the differential equation by a smooth **test function** v with compact support within the domain Ω. Using Green's formula (a generalization of integration by parts), this converts to

$$-\int_\Omega v \cdot (\Delta u^*)dx = \underbrace{\int_\Omega v \cdot f^\Omega dx}_{=:(v,f^\Omega)_{L_2(\Omega)}} = \underbrace{\int_\Omega (\nabla v) \cdot (\nabla u^*)dx}_{=:a(v,u^*)}$$

or, in shorthand notation, to an infinite number of equations

$$a(v, u^*) = (v, f^\Omega)_{L_2(\Omega)} \text{ for all test functions } v$$

between two bilinear forms, involving two local integrations. This technique gets rid of the second derivative, at the cost of local integration, but with certain theoretical advantages we do not want to explain here.

1.5 Comparison of Strong and Weak Problems

Concerning the range of partial differential equation techniques we handle in this book, we restrict ourselves to cases we can solve **without integrations**, using radial basis functions as trial functions. This implies that we ignore boundary integral equation methods and finite elements as numerical techniques. For these, there are enough books on the market. Since there is no integration, we need no "background" or "integration" mesh, and some authors call such methods "truly meshless".

On the analytical side, we shall consider only problems in **strong form**, i.e. where all functions and their required derivatives can be evaluated pointwise. Some readers might argue that this rules out too many important problems. Therefore we want to provide some arguments in favor of our choice. Readers without a solid mathematical background should skip over these remarks.

First, we do not consider the additional **regularity** needed for a strong solution to be a serious drawback in practice. Useful error bounds and rapidly convergent methods will always need regularity assumptions on the problem and its solutions. Thus our techniques should be compared to **spectral methods** or the p–technique in finite elements. If a solution of a weak Poisson problem definitely is not a solution of a strong problem, the standard finite element methods will not converge with reasonable orders anyway, and we do not want to compete in such a situation.

Second, the problems to be expected from taking a strong form instead of a weak form can in many cases be eliminated. To this end, we look at those problems somewhat more closely.

The first case comes from domains with **incoming corners**. Even if the data functions f^Ω and f^Γ are smooth, there may be a singularity of u^* at the boundary. However, this singularity is a known function of the incoming corner angle, and by adding an appropriate function to the set of trial functions, the problem can be overcome.

The next problem source is induced by **non–smooth data** functions. Since these are fixed, the exceptional points are known in principle, and precautions can be taken by using nonsmooth trial functions with singularities located properly. For time–dependent problems with moving boundaries or discontinuities, meshless methods can be adapted very flexibly, but this is a research area which is beyond the scope of this book.

The case of data functions which do not allow point evaluations (i.e. $f^\Omega \in L_2(\Omega)$ or even distributional data for the Poisson problem) and still require integration can be ruled out too, because on the one hand we do not know a single case from applications, and on the other hand we would like to know how to handle this case with a standard finite element code, which usually integrates by applying integration formulae. The latter can never work for L_2 functions.

Things are fundamentally different when applications in science or engineering insist on **distributional data**. Then weak forms are unavoidable, and we address this situation now.

Many of the techniques here can be transferred to weak forms, if absolutely necessary. This is explained to some extent in [HS05] for a class of symmetric meshless methods. The meshless local Petrov–Galerkin (**MLPG**) method [AZ98a, AZ98b, AZ00] of S.N. Atluri and collaborators is a good working example of a weak meshless technique with plenty of successful applications in engineering, Because it is both weak and unsymmetric, its mathematical analysis is hard, and thus it only recently was put on a solid theoretical foundation [Sch06c].

Finally, **mixed** weak and strong problems are possible [HS05, Sch06c], confining the weak approach to areas where severe regularity problems occur or data are distributional. Together with adaptivity, mixed methods will surely prove useful in the future.

1.6 Collocation Techniques

This approach applies to problems in strong form and does not require numerical integration. Consequently, it avoids all kinds of meshes. In order to cope with scattered multivariate data, it uses methods based on radial basis function approximation, generalizing the interpolation problem described in Section 1.2. Numerical computations indicate that these meshless methods are ideal for solving complex physical problems in strong form on irregular domains. Section 3 will select some typical examples out of a rich literature, but here we want to sketch the basic principles.

Consider the following linear **Dirichlet boundary value problem**:

$$\begin{array}{rcll} Lu & = & f^\Omega & \text{in} \quad \Omega \subset I\!R^d \\ u & = & f^\Gamma & \text{on} \quad \Gamma := \partial\Omega \end{array} \tag{1.6.1}$$

where L is a linear differential or integral operator. **Collocation** is a technique that interprets the above equations in a strong pointwise sense and discretizes them by imposing finitely many conditions

$$\begin{array}{rcll} Lu(x_j^\Omega) & = & f^\Omega(x_j^\Omega), & x_j^\Omega \in \Omega, \quad 1 \le j \le m_\Omega \\ u(x_j^\Gamma) & = & f^\Gamma(x_j^\Gamma), & x_j^\Gamma \in \Gamma \quad\; 1 \le j \le m_\Gamma \end{array} \tag{1.6.2}$$

on $m := m_\Omega + m_\Gamma$ **test points** in Ω and Γ. Note that this is a generalization of a standard multivariate interpolation problem as sketched in Section 1.2 and to be described in full generality in the following chapter. The exact solution u^* of the Dirichlet problem (1.6.1) will satisfy (1.6.2), but there are plenty of other functions u which will also satisfy these equations. Thus one has to fix a finite-dimensional space U of **trial functions** to pick solutions u of (1.6.2) from, and it is reasonable to let U be at least m-dimensional. But then the fundamental problem of all collocation methods is to guarantee solvability of the linear system (1.6.2) when restricted to trial functions from U. This problem is hard to solve, and therefore collocation methods have not attracted much attention so far from the mathematical community.

However, as we know from Chapter 1, kernel-based trial spaces allow nonsingular matrices for multivariate interpolation problems, and so there is some hope that kernel-based trial spaces also serve well for collocation. Unfortunately, things are not as easy as for interpolation, but they proved to work well in plenty of applications.

The first attempt to use radial basis functions to solve partial differential equations is due to Ed Kansa [Kan86]. The idea is to take trial functions of the form (1.1.1) or (1.2.5), depending on the order of the positive definiteness of the radial basis function used. For positive q one also has to postulate (1.2.6), and thus one should take $n := m + q$ to arrive at a problem with the correct degrees of freedom. The collocation equations come out in general as

$$
\begin{array}{lclcll}
\sum_{k=1}^{n} \alpha_k \Delta\phi(\|\mathbf{x}_j^\Omega - \mathbf{y}_k\|_2) & + & \sum_{\ell=1}^{q} \beta_\ell \Delta p_\ell(\mathbf{x}_j^\Omega) & = & f^\Omega(\mathbf{x}_j^\Omega), & 1 \le j \le m_\Omega \\
\sum_{k=1}^{n} \alpha_k \phi(\|\mathbf{x}_j^\Gamma - \mathbf{y}_k\|_2) & + & \sum_{\ell=1}^{q} \beta_\ell p_\ell(\mathbf{x}_j^\Gamma) & = & f^\Gamma(\mathbf{x}_j^\Gamma), & 1 \le j \le m_\Gamma \\
\sum_{k=1}^{n} \alpha_k p_\ell(\mathbf{y}_k) & + & 0 & = & 0, & 1 \le \ell \le q,
\end{array}
\tag{1.6.3}
$$

forming a linear unsymmetric $n \times n = (m_\Omega + m_\Gamma + q) \times (m_\Omega + m_\Gamma + q)$ system of equations. In all known applications, the system is nonsingular, but there are specially constructed cases [HS01] where the problem is singular.

A variety of experimental studies, e.g. by Kansa [Kan90a, Kan90b], Golberg and Chen [GCK96], demonstrated this technique to be very useful for solving partial differential and integral equations in strong form. Hon et. al. further extended the applications to the numerical solutions of various ordinary and partial differential equations including general initial value problems [HM97], the nonlinear **Burgers equation** with a **shock wave** [HM98], the **shallow water equation** for tide and current simulation in domains with irregular boundaries [Hon93], and **free boundary problems** like the American **option pricing** [HM99, Hon02]. These cases will be reported in Chapter 3. Due to the unsymmetry, the theoretical possibility of degeneration, and the lack of a seminorm-minimization in the analytic background, a theoretical justification is difficult but was provided recently [Sch07b] for certain variations of the basic approach.

The lack of symmetry may be viewed as a bug, but it also can be seen as a feature. In particular, the method does not assume ellipticity or self-adjointness of differential operators. Thus it applies to a very general class of problems, as many applications show.

On the other hand, symmetry can be brought back again by a suitable change of the trial space. In the original method, there is no connection between the test points $\mathbf{x}_j^\Omega$, $\mathbf{x}_j^\Gamma$ and the trial centers $\mathbf{y}_k$. If the trial points are dropped completely, one can recycle the test points to define new trial functions by

$$u(\mathbf{x}) := \sum_{i=1}^{m_\Omega} \alpha_i^\Omega \Delta\phi(\|\mathbf{x}-\mathbf{x}_i^\Omega\|_2)+\sum_{j=1}^{m_\Gamma} \alpha_j^\Gamma \phi(\|\mathbf{x}-\mathbf{x}_j^\Gamma\|_2)+\sum_{\ell=1}^{q} \beta_\ell p_\ell(\mathbf{x}) \quad (1.6.4)$$

providing the correct number $n := m_\Omega + m_\Gamma + q$ of degrees of freedom. Note how the test points $\mathbf{x}_i^\Omega$ and $\mathbf{x}_j^\Gamma$ lead to different kinds of trial functions, since they apply "their" differential or boundary operator to one of the arguments of the radial basis function.

The collocation equations now come out as a symmetric square linear system with block structure. If we define vectors

$$\begin{array}{lcl}
\mathbf{f}^\Omega & := & (f^\Omega(\mathbf{x}_1^\Omega), \ldots, f^\Omega(\mathbf{x}_{m_\Omega}^\Omega))^T \in I\!R^{m_\Omega} \\
\mathbf{f}^\Gamma & := & (f^\Gamma(\mathbf{x}_1^\Gamma), \ldots, f^\Gamma(\mathbf{x}_{m_\Gamma}^\Gamma))^T \in I\!R^{m_\Gamma} \\
\mathbf{0}_q & := & (0, \ldots, 0)^T \in I\!R^q \\
\mathbf{a}^\Omega & := & (\alpha_1^\Omega, \ldots, \alpha_{m_\Omega}^\Omega)^T \in I\!R^{m_\Omega} \\
\mathbf{a}^\Gamma & := & (\alpha_1^\Gamma, \ldots, \alpha_{m_\Gamma}^\Gamma)^T \in I\!R^{m_\Gamma} \\
\mathbf{b}_q & := & (\beta_1, \ldots, \beta_q)^T \in I\!R^q,
\end{array}$$

we can write the system with a slight abuse of notation as

$$\begin{pmatrix} \Delta^2\phi(\|\mathbf{x}_r^\Omega - \mathbf{x}_i^\Omega\|_2) & \Delta\phi(\|\mathbf{x}_r^\Omega - \mathbf{x}_j^\Gamma\|_2) & \Delta p_\ell(\mathbf{x}_r^\Omega) \\ \Delta\phi(\|\mathbf{x}_s^\Gamma - \mathbf{x}_i^\Omega\|_2) & \phi(\|\mathbf{x}_s^\Gamma - \mathbf{x}_j^\Gamma\|_2) & p_\ell(\mathbf{x}_s^\Gamma) \\ \Delta p_t(\mathbf{x}_i^\Omega) & p_t(\mathbf{x}_j^\Gamma) & 0 \end{pmatrix} \begin{pmatrix} \mathbf{a}^\Omega \\ \mathbf{a}^\Gamma \\ \mathbf{b}_q \end{pmatrix} = \begin{pmatrix} \mathbf{f}^\Omega \\ \mathbf{f}^\Gamma \\ \mathbf{0}_q \end{pmatrix}$$

where indices in the submatrices run over

$$\begin{array}{ccccc}
1 & \le & i, r & \le & m_\Omega \\
1 & \le & j, s & \le & m_\Gamma \\
1 & \le & \ell, t & \le & q.
\end{array}$$

The first set of equations arises when applying Δ to (1.6.4) on the domain test points $\mathbf{x}_r^\Omega$. The second is the evaluation of (1.6.4) on the boundary test points $\mathbf{x}_s^\Gamma$. The third is a natural generalization of (1.2.6) to the current trial space. Note that the system has the general symmetric form

$$\begin{pmatrix} \mathbf{A}^{\Omega,\Omega} & \mathbf{A}^{\Omega,\Gamma} & \mathbf{P}^\Omega \\ \mathbf{A}^{\Omega,\Gamma^T} & \mathbf{A}^{\Gamma,\Gamma} & \mathbf{P}^\Gamma \\ \mathbf{P}^{\Omega^T} & \mathbf{P}^{\Gamma^T} & \mathbf{0}_{q\times q} \end{pmatrix} \begin{pmatrix} \mathbf{a}^\Omega \\ \mathbf{a}^\Gamma \\ \mathbf{b}_q \end{pmatrix} = \begin{pmatrix} \mathbf{f}^\Omega \\ \mathbf{f}^\Gamma \\ \mathbf{0}_q \end{pmatrix} \quad (1.6.5)$$

with evident notation when compared to the previous display.

Under weak assumptions, such matrices are nonsingular [Wu92, Isk95] because they arise as Hermite interpolation systems generalizing (1.2.2). The approach is called **symmetric collocation** and has a solid mathematical foundation [FS98b, FS98a] making use of the symmetry of the discretized problem. We provide specific applications in Chapter 3 and some underlying theory in Section 2.2.

1.7 Method of Fundamental Solutions

This method is a highly effective technique for solving **homogeneous** differential equations, e.g. the potential problem (1.4.1) with $f^{\Omega} = 0$. The basic idea is to use trial functions that satisfy the differential equation and to superimpose the trial functions in such a way that the additional boundary conditions are satisfied with sufficient accuracy. It reduces a homogeneous partial differential equation problem to an approximation or interpolation problem on the boundary by fitting the data on the boundary. Since fundamental solutions are special homogeneous solutions which are well-known and easy to implement for many practically important differential operators, the method of fundamental solutions is a relatively easy way to find the desired solution of a given homogeneous differential equation with the correct boundary values.

For example, the function $u_{\mathbf{y}}(\mathbf{x}) := \|\mathbf{x} - \mathbf{y}\|_2^{-1}$ satisfies $(\Delta u_{\mathbf{y}})(\mathbf{x}) = 0$ everywhere in $I\!R^3$ except for $\mathbf{x} = \mathbf{y}$, where it is singular. But if points $\mathbf{y}_1, \ldots, \mathbf{y}_n$ are placed outside the domain Ω, any linear combination u of the $u_{\mathbf{y}_1}, \ldots, u_{\mathbf{y}_n}$ will satisfy $\Delta u = 0$ on all of Ω. Now the freedom in the coefficients can be used to make u a good approximation to f^{Γ} on the boundary. For this, several methods are possible, but we do not want to provide details here. It suffices to see that we have got rid of the differential equation, arriving at a plain approximation problem on the boundary of Ω.

The method of fundamental solutions was first proposed by Kupradze and Aleksidze [KA64b] in 1964. During the past decades, the method has re-emerged as a popular boundary-type meshless method and has been applied to solve various science and engineering problems. One of the reasons for the renewed interest for this method is that it has been successfully extended to solve inhomogeneous and time–dependent problems. As a result, the method now is applicable to a larger class of partial differential equations. Furthermore, it does not require numerical integration and is "truly meshless" in the sense that no tedious domain or boundary mesh is necessary. Hence the method is extremely simple to implement, making it especially attractive to scientists and engineers working in applications.

In many cases, e.g. for the potential equation, the underlying mathematical analysis has a **maximum principle** [PW67, Jos02] for homogeneous solutions, and then the total error is bounded by the error on the boundary, which can be evaluated easily. Furthermore, adaptive versions are possible, introducing more trial functions to handle places where the boundary error is not tolerable. In very restricted cases, convergence of these methods can be proven to be spectral (i.e. faster than any fixed order), and for "smooth" application problems this technique shows an extremely good convergence behavior in practice.

This book is the first to give a comprehensive treatment of the method of fundamental solutions (**MFS**). The connection to radial basis function techniques is that fundamental solutions of radially invariant differential operators like the Laplace or the Helmholtz operator have radial form around a singularity, as in the above case. For example, one of the most widely used radial basis functions, the **thin–plate spline** $\phi(r) := r^2 \log r$ is the fundamental solution at the origin to the thin–plate equation $\Delta^2 u = 0$ in $I\!R^2$.

Methods which solve homogeneous equations by superposition of general solutions and an approximation on the boundary have quite some history, dating back to Trefftz [Tre26]. In particular, the work of L. Collatz [MNRW91] contains plenty of examples done in the 1960's. Later, this subject was taken up again and called **Boundary Knot Method** [CT00, Che02, CH03, HC03], but we stick to the Method of Fundamental Solutions here. A recent technique [Sch07c] for solving homogeneous problems is based on singularity–free kernels. It has a full mathematical background theory, but is still rather special and under investigation.

1.8 Method of Particular Solutions

Inhomogeneous differential equations with linear differential operators L can be reduced to homogeneous cases, if trial functions u_j are used for which $Lu_j = f_j$ is known. If $Lu = f^\Omega$ is to be solved, a good approximation f to f^Ω by a linear combination of the f_j will have the form $f = Lu$ with u being a linear combination of the u_j, using the same coefficients. This is the **Method of Particular Solutions** (**MPS**). It reduces the solution of an inhomogeneous differential equation to an approximation problem for the inhomogeneity.

After this first stage, $Lu = f$ is close to f^Ω, and the original problem $Lu = f^\Omega$ can be replaced by a homogeneous problem due to

$$L(u^* - u) \approx f^\Omega - f \approx 0,$$

and then the **Method of Fundamental Solutions** (**MFS**) can be applied. The approximation of f^Ω by f can be done by interpolation or approximation techniques of the previous sections, provided that the f_j are translates of radial basis functions.

$$\text{Inhom. PDE} \overset{MPS}{\Rightarrow} \begin{cases} \text{App. in interior} \\ \text{Homog. PDE} \qquad \overset{MFS}{\Rightarrow} \text{ App. on boundary} \end{cases}$$

This is how the major techniques of this book are related. For the most important differential operators and radial basis functions, we provide useful (u_j, f_j) pairs with $Lu_j = f_j$ and show their applications.

1.9 Time–dependent Problems

In the final chapter, we extend the method of fundamental solutions and the method of particular solutions to solving time–dependent problems. A common feature of the methods in this chapter is that a given time–dependent problem is reduced to an inhomogeneous modified Helmholtz equation through the use of two basic techniques:

- **Laplace transforms** and
- **time–stepping algorithms**.

Using the Laplace transform, the given time–dependent problem can be solved in one step in Laplace space and then converted back to the original time space using the inverse Laplace transform. By time-stepping, the given time–dependent problem is transformed into a sequence of modified Helmholtz equations which in turn can be solved by the numerical procedures described in the previous chapters. In the parabolic case, we consider both linear and nonlinear heat equations. In the hyperbolic case, we only consider the wave equation using the time-stepping algorithm. Readers are encouraged to apply this approach to solve more challenging time–dependent problems.

1.10 Lists of Radial Basis Functions

Table 1.3 shows a selection of the most popular radial basis functions $\phi(r)$ with non–compact support. We provide the minimal order Q of conditional positive definiteness and indicate the range of additional parameters.

Classes of **compactly supported** radial basis functions were provided by Wu [Wu95], Wendland [Wen95], and Buhmann [Buh98]. We list

Name	$\phi(r)$	Q	condition
Gaussian	$exp(-r^2)$	0	
Matern	$r^\nu K_\nu(r)$	0	$\nu > 0$
inverse multiquadric	$(1+r^2)^{\beta/2}$	0	$\beta < 0$
multiquadric	$(-1)^{\lceil\beta/2\rceil}(1+r^2)^{\beta/2}$	$\lceil\beta/2\rceil$	$\beta > 0,\ \beta \notin 2I\!N$
polyharmonic	$(-1)^{\lceil\beta/2\rceil}r^\beta$	$\lceil\beta/2\rceil$	$\beta > 0,\ \beta \notin 2I\!N$
polyharmonic	$(-1)^{1+\beta/2}r^\beta \log r$	$1+\beta/2$	$\beta > 0,\ \beta \in 2I\!N$

Table 1.3. *Global RBFs*

a selection of Wendland's functions in Table 1.4. These are always positive definite up to a maximal space dimension d_{max}, and have smoothness C^k as indicated in the table. Their polynomial degree is minimal for given smoothness, and they have a close connection to certain Sobolev spaces.

$\phi(r)$	k	d_{max}
$(1-r)_+^2$	0	3
$(1-r)_+^4(4r+1)$	2	3
$(1-r)_+^6(35r^2+18r+3)$	4	3
$(1-r)_+^8(32r^3+25r^2+8r+1)$	6	3
$(1-r)_+^3$	0	5
$(1-r)_+^5(5r+1)$	2	5
$(1-r)_+^7(16r^2+7r+1)$	4	5

Table 1.4. *Selection of Wendland's compactly supported radial basis functions*

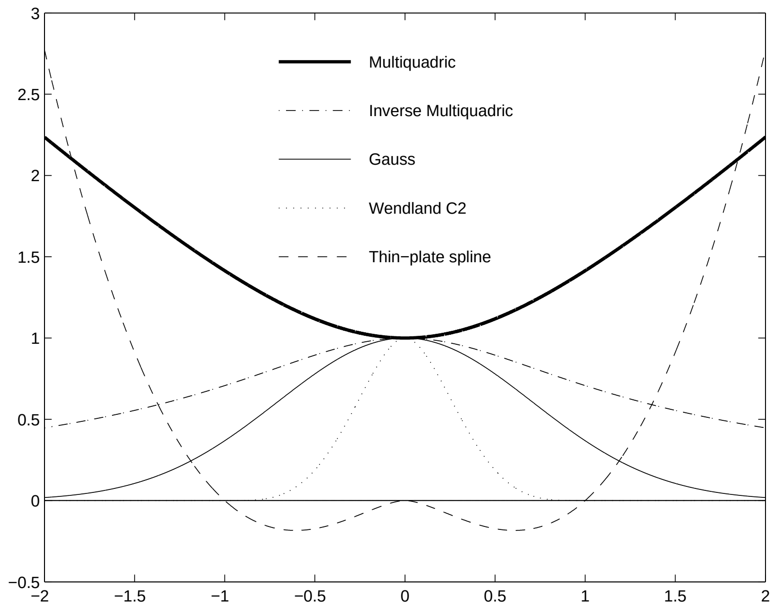

Fig. 1.1. Some radial basis functions

2
Basic Techniques for Function Recovery

This chapter treats a basic problem of Scientific Computing: the **recovery** of multivariate functions from discrete data. We shall use **radial basis functions** for this purpose, and we shall confine ourselves to reconstruction from **strong data** consisting of evaluations of the function itself or its derivatives at discrete points. Recovery of functions from **weak data**, i.e. from data given as integrals against test functions, is a challenging research problem [Sch06b, Sch06c], but it has to be ignored here. Note that weak data require integration, and we want to avoid unnecessary background meshes used for this purpose.

2.1 Interpolation of Lagrange Data

Going back to Section 1.2, we assume data values $y_1, \ldots, y_m \in I\!R$ to be given, which are supposed to be values $y_k = u^*(\mathbf{x}_k)$ of some unknown function u^* at scattered points $\mathbf{x}_1, \ldots, \mathbf{x}_m$ in some domain Ω in $I\!R^d$. We then pick a positive definite radial basis function ϕ and set up the linear system (1.2.2) of m equations for the m coefficients $\alpha_1, \ldots, \alpha_m$ of the representation (1.1.1) where $n = m$ and $\mathbf{y}_k = \mathbf{x}_k$ for all k. In case of conditionally positive radial basis functions, we have to use (1.2.5) and add the conditions (1.2.6).

In Figure 2.1 we have 150 scattered data points in $[-3, 3]^2$ in which we interpolate the MATLAB `peaks` function (top right). The next row shows the interpolant using Gaussians and the absolute error. The lower row shows MATLAB's standard technique for interpolation of scattered data using the `griddata` command. The results are typical for such problems: radial basis function interpolants recover smooth functions very well from a sample of scattered values, provided that the values are noiseless and the underlying function is smooth.

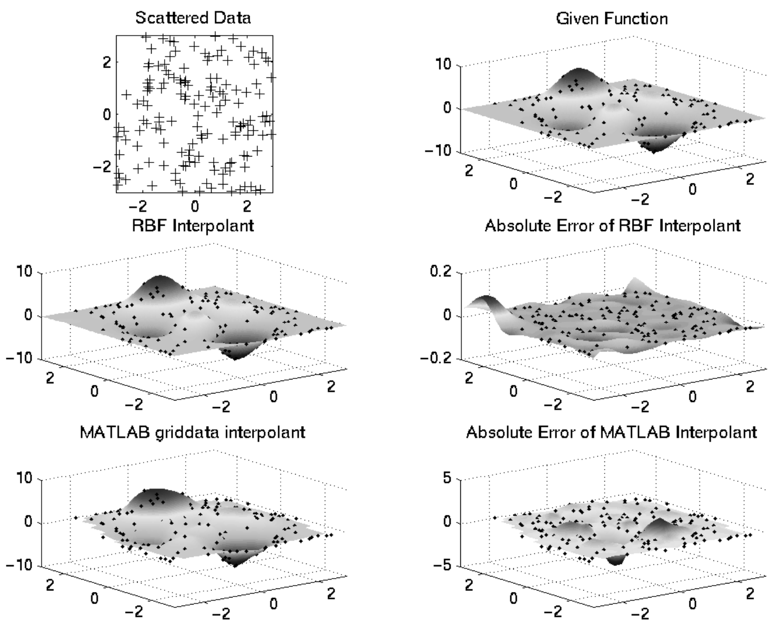

Fig. 2.1. Interpolation by radial basis functions

The ability of radial basis functions to deal with arbitrary point locations in arbitrary dimensions is very useful when geometrical objects have to be constructed, parametrized, or warped, see e.g. [ADR94, CFB97, NFN00, CBC$^+$01, OBS03, RTSD03, WK05, BK05]. In particular, one can use such transformations to couple incompatible finite element codes [ABW06].

Furthermore, interpolation of functions has quite some impact on methods solving partial differential equations. In Chapter 5 we shall solve inhomogeneous partial differential equations by interpolating the right-hand sides by radial basis functions which are related to particular solutions of the partial differential equation in question.

Another important issue is the possibility parametrizing spaces of translates of kernels not via coefficients, but via function values at the translation centers. This simplifies **meshless methods** "*constructing the approximation entirely in terms of nodes*" [BKO$^+$96]. Since kernel interpolants approximate higher derivatives well, local function values can be used to provide good estimates for derivative data [WHW05]. This has connections to **pseudospectral methods** [Fas04].

2.2 Interpolation of Mixed Data

It is quite easy to allow much more general data for interpolation by radial basis functions. For example, consider recovery of a multivariate function f from data including the values $\frac{\partial f}{\partial x_2}(\mathbf{z})$, $\int_\Omega f(\mathbf{t})d\mathbf{t}$. The basic trick, due to Z.M. Wu [Wu92], is to use special trial functions

$$\begin{array}{ccc} \dfrac{\partial\phi(\|\mathbf{x}-\mathbf{z}\|_2)}{\partial x_2} & \text{for} & \dfrac{\partial f}{\partial x_2}(\mathbf{z}) \\ \displaystyle\int_\Omega \phi(\|\mathbf{x}-\mathbf{t}\|_2)d\mathbf{t} & \text{for} & \displaystyle\int_\Omega f(\mathbf{t})d\mathbf{t} \end{array}$$

to cope with these requirements. In general, if a linear functional λ defines a data value $\lambda(f)$ for a function f as in the above cases with $\lambda_1(f) = \frac{\partial f}{\partial x_2}(\mathbf{z})$, $\lambda_2(f) = \int_\Omega f(\mathbf{t})d\mathbf{t}$, the special trial function $u_\lambda(\mathbf{x})$ to be added is

$$u_\lambda(\mathbf{x}) := \lambda^{\mathbf{t}}\phi(\|\mathbf{x}-\mathbf{t}\|_2) \text{ for } \lambda^{\mathbf{t}}(f(\mathbf{t}))$$

where the upper index denotes the variable the functional acts on. If $m = n$ functionals $\lambda_1, \ldots, \lambda_m$ are given, the span (1.1.1) of trial functions is to be replaced by

$$u(\mathbf{x}) = \sum_{k=1}^{n} \alpha_k \lambda_k^{\mathbf{t}}\phi(\|\mathbf{x}-\mathbf{t}\|_2).$$

The interpolation system (1.2.2) turns into

$$\lambda_j u = \sum_{k=1}^{n} \alpha_k \lambda_k^{\mathbf{t}}\lambda_j^{\mathbf{x}}\phi(\|\mathbf{x}-\mathbf{t}\|_2),\ 1 \le j \le n \qquad (2.2.1)$$

with a symmetric matrix composed of $\lambda_k^{\mathbf{t}}\lambda_j^{\mathbf{x}}\phi(\|\mathbf{x}-\mathbf{t}\|_2)$, $1 \le j,k \le n$ which is positive definite if the functionals are linearly independent and ϕ is positive definite. Thus a fully general **Hermite–Birkhoff interpolation** is possible as long as the entries of the matrix in (2.2.1) are meaningful.

To give an example with general functionals, Figure 2.2 shows an interpolation to Neumann data +1 and -1 on each half of the unit circle, respectively, in a total of 64 points by linear combinations of properly scaled Gaussians.

In case of conditionally positive definite radial basis functions, the span of (1.2.5) turns into

$$u(\mathbf{x}) := \sum_{k=1}^{n} \alpha_k \lambda_k^{\mathbf{t}}\phi(\|\mathbf{x}-\mathbf{t}\|_2) + \sum_{\ell=1}^{q} \beta_\ell p_\ell(\mathbf{x})$$

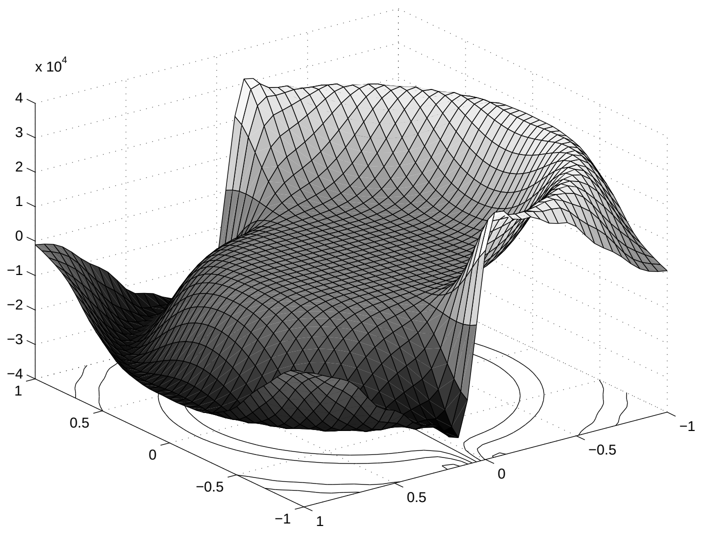

Fig. 2.2. Generalized interpolant to Neumann data

while the additional condition (1.2.6) is replaced by

$$\sum_{k=1}^{n} \alpha_k \lambda_k^{\mathbf{t}} p_\ell(\mathbf{t}) = 0,\ 1 \le \ell \le q,$$

and the interpolation problem is solvable, if the standard **polynomial unisolvency constraint**

$$\lambda_k^{\mathbf{t}} p(\mathbf{t}) = 0 \text{ for all } 1 \le k \le n \text{ and } p \in P_{Q-1}^d \text{ implies } p = 0$$

is imposed, replacing (1.2.8).

Another example of recovery from non–Lagrange data is the construction of **Lyapounov basins** from data consisting of **orbital derivatives** [GW06a, GW06b].

The flexibility to cope with general data is the key to various applications of radial basis functions within methods solving partial differential equations. Collocation techniques, as sketched in Section 1.6 and treated in Chapter 3 in full detail, solve partial differential equations numerically by interpolation of values of differential operators and boundary conditions.

Another important aspect is the possibility of implementing **addi-**

tional linear conditions or **constraints** like

$$\lambda(u) := \int_\Omega u(\mathbf{x})\mathrm{d}\mathbf{x} = 1$$

on a trial function. For instance, this allows us to handle **conservation laws** and is inevitable for Finite Volume Methods. A constraint like the one above, when used as additional data, adds another degree of freedom to the trial space by addition of the basis function $u_\lambda(\mathbf{x}) := \lambda^{\mathbf{t}}\phi(\|\mathbf{x}-\mathbf{t}\|_2)$, and at the same time it uses this additional degree of freedom to satisfy the constraint. This technique deserves much more attention in applications.

2.3 Error Behavior

If exact data come from smooth functions f, and if smooth radial basis functions ϕ are used for interpolation, users can expect very small interpolation errors. In particular, the error goes to zero when the data samples are getting dense. The actual error behavior is limited by the smoothness of both f and ϕ. Quantitative error bounds can be obtained from the standard literature [Buh03, Wen05] and recent papers [NWW06]. They are completely *local*, and they are in terms of the **fill distance**

$$h := h(X,\Omega) := \sup_{\mathbf{y}\in\Omega}\min_{\mathbf{x}\in X}\|\mathbf{x}-\mathbf{y}\|_2 \tag{2.3.1}$$

of the discrete set $X = \{\mathbf{x}_1,\dots,\mathbf{x}_n\}$ of centers with respect to the domain Ω where the error is measured. The fill distance is the radius of the largest data–less ball around points of the domain, i.e. it measures the largest gap in the data.

The interpolation error then converges to zero for $h \to 0$ at a rate dictated by the minimum smoothness of f and ϕ. For infinitely smooth radial basis functions like the Gaussian or multiquadrics, convergence is even exponential [MN92, Yoo01, RZ06]. Derivatives are also convergent as far as the smoothness of f and ϕ allows, but at a smaller rate, of course. This is particularly important when applications require good reproductions of derivatives, e.g. velocity fields or stress tensors.

For interpolation of the smooth `peaks` function provided by MATLAB and used already in Figure 2.1, the error behavior on $[-3,3]^2$ as a function of fill distance h is given by Figure 2.3. It can be clearly seen that smooth ϕ yield smaller errors with higher convergence rates. In contrast

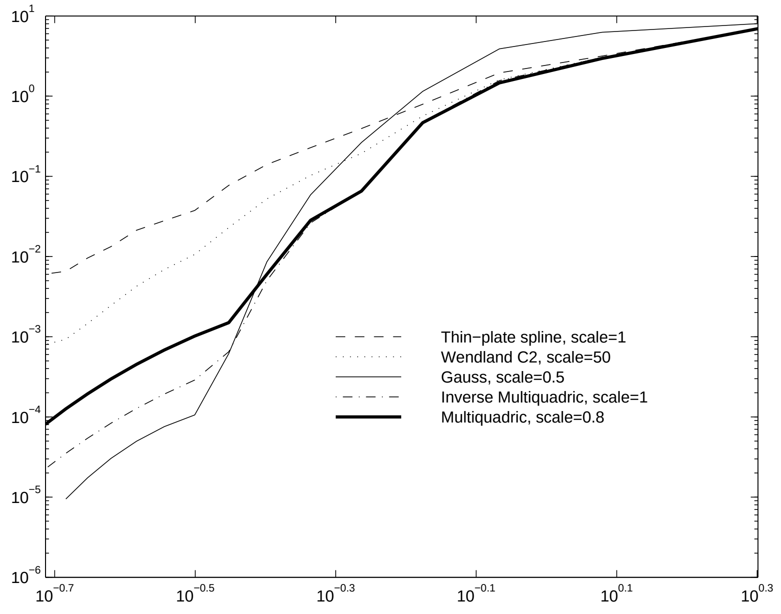

Fig. 2.3. Nonstationary interpolation to a smooth function as a function of fill distance

to this, Figure 2.4 shows interpolation to the nonsmooth function

$$f(x,y) = 0.03 * \max(0, 6 - x^2 - y^2)^2, \tag{2.3.2}$$

on $[-3,3]^2$, where now the convergence rate is dictated by the smoothness of f instead of ϕ and is thus more or less fixed. Excessive smoothness of ϕ never spoils the error behavior but induces excessive instability, as we shall see later.

2.4 Stability

But there is a serious drawback when using radial basis functions on dense data sets, i.e. with small fill distance. The condition of the matrices used in (1.2.2) and (2.2.1) will get extremely large if the **separation distance**

$$S(X) := \frac{1}{2} \min_{1 \le i < j \le n} \|\mathbf{x}_i - \mathbf{x}_j\|_2$$

of points of $X = \{\mathbf{x}_1, \dots, \mathbf{x}_n\}$ gets small. Figure 2.5 shows this effect in the situation of Figure 2.3.

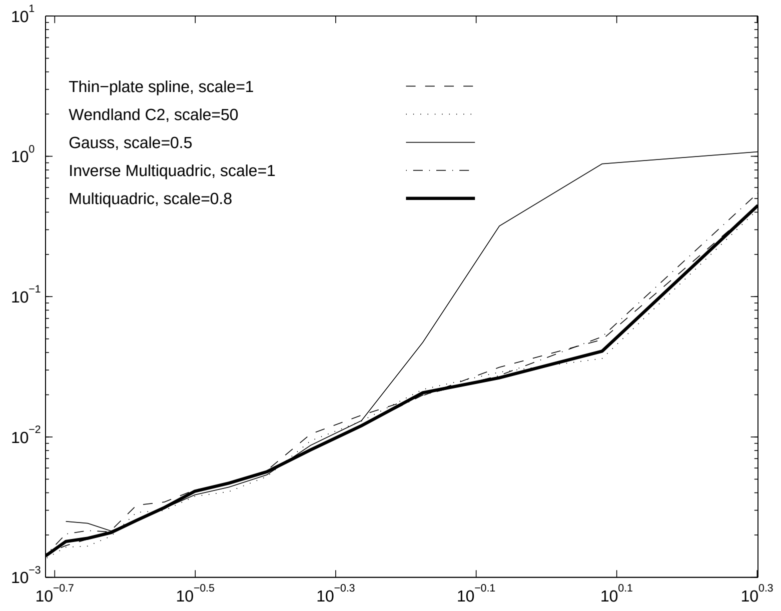

Fig. 2.4. Nonstationary interpolation to a nonsmooth function as a function of fill distance

If points are distributed well, the separation distance $S(X)$ will be proportional to the fill distance $h(X, \Omega)$ of (2.3.1). In fact, since the fill distance is the radius of the largest ball with arbitrary center in the underlying domain Ω without any data point in its interior, the separation distance $S(X)$ is the radius of the smallest ball anywhere without any data point in its interior, but with at least two points of X on the boundary. Thus for convex domains one always has $S(X) \leq h(X, \Omega)$. But since separation distance depends only on the closest pair of points and ignores the rest, it is reasonable to avoid unusually close points leading to some $S(X)$ which is considerably smaller than $h(X, \Omega)$. Consequently, a distribution of data locations in X is called **quasi–uniform** if there is a positive **uniformity constant** $\gamma \leq 1$ such that

$$\gamma\, h(X, \Omega) \leq S(X) \leq h(X, \Omega). \tag{2.4.1}$$

To maintain quasi-uniformity, it suffices in most cases to delete "duplicates". Furthermore, there are sophisticated **thinning algorithms** [FI98, DDFI05, WR05] to keep fill and separation distance proportional,

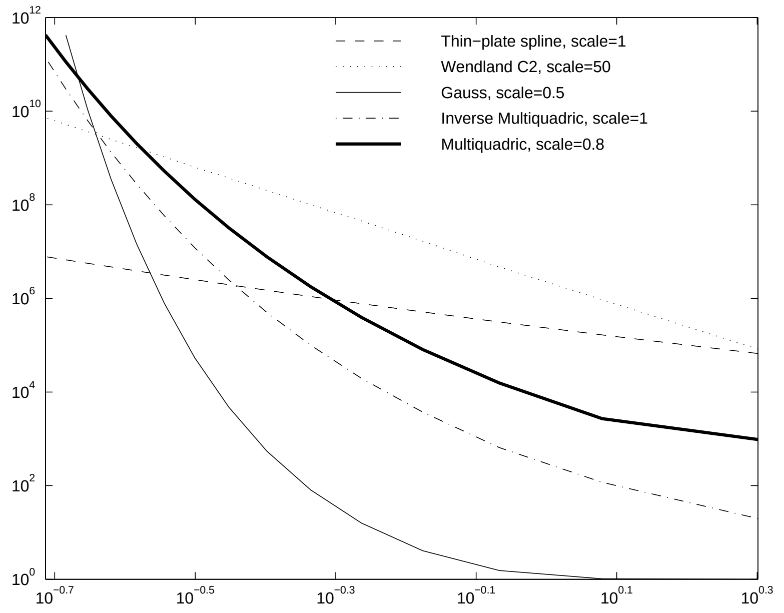

Fig. 2.5. Condition as function of separation distance

i.e. to assure quasi-uniformity at multiple scaling levels. We shall come back to this in Section 5.8. Finally, we point out that adding a properly scaled positive regularization parameter into the diagonal of the kernel matrix allows to get rid of the negative influence of small separation distance [WR05].

Unless radial basis functions are rescaled in a data-dependent way, it can be proven [Sch95] that there is a close link between error and stability, even if fill and separation distance are proportional. In fact, both are tied to the smoothness of ϕ, letting stability become worse and errors become smaller when taking smoother radial basis functions. This is kind of an **Uncertainty Principle**:

> It is impossible to construct radial basis functions which guarantee good stability and small errors at the same time.

We illustrate this by an example. Since [Sch95] proves that the square of the L_∞ error roughly behaves like the smallest eigenvalue of the interpolation matrix, Figure 2.6 plots the product of the MATLAB condition estimate `condest` with the square of the L_∞ error for the nonstationary interpolation of the MATLAB `peaks` function, used already for Figures

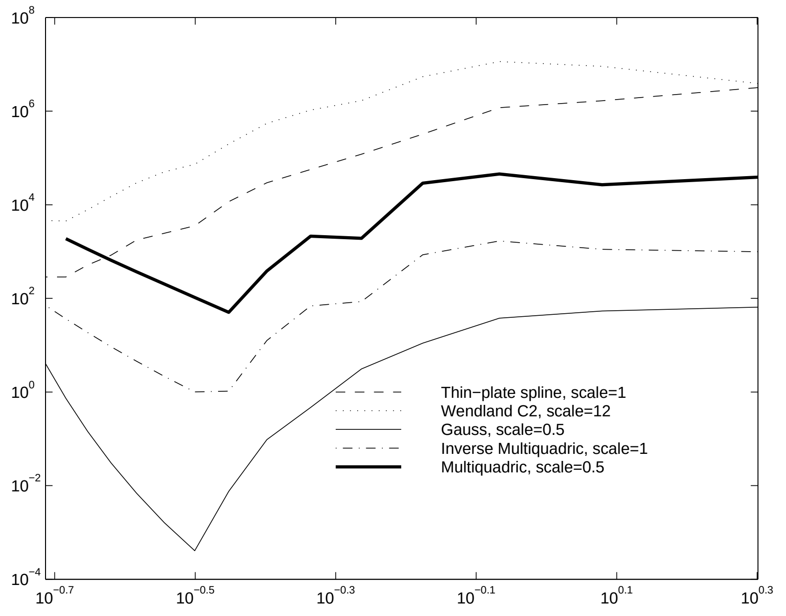

Fig. 2.6. Squared L_∞ error times condition as a function of fill distance

2.3, 2.5, and 2.7 to show the error and condition behavior there. Note that the curves do not vary much if compared to Figure 2.5. Example 4.5.1 for the Method of Fundamental Solutions shows a similarly close link between error and condition. But the inherent instability of interpolation matrices for small separation distances is dependent on having chosen the standard basis consisting of translates of the given radial basis function. More sophisticated bases will show a better behaviour, in particular those which parametrize trial functions in terms of values at nodes [BKO$^+$96].

Thus **smoothness** of radial basis functions must be chosen with some care and selected dependent on the smoothness of the function to be approximated. From the point of view of reproduction quality, smooth radial basis functions can well recover nonsmooth functions, as proven by papers concerning error bounds [NWW06]. On the other hand, non–smooth radial basis functions will not achieve high convergence rates when approximating smooth functions [SW02]. This means that using too much smoothness in the chosen radial basis function is not critical for the error, but rather for the stability. But in many practical cases, the choice of smoothness is not as sensible as the choice of scale, as will be discussed in Section 2.6.

2.5 Regularization

The linear systems arising in radial basis function methods have a special form of degeneration: the large eigenvalues of kernel matrices usually are moderate, but there are very small ones leading to bad condition. This is a paradoxical consequence of the good error behavior we demonstrated in Section 2.3. In fact, since trial spaces spanned by translates of radial basis functions have very good approximation properties, the linear systems arising in all sorts of recovery problems throughout this book will have good approximate solutions reproducing the right-hand sides well, no matter what the condition number of the system is. And the condition will increase if trial centers are getting close, because then certain rows and columns of the matrices $\mathbf{A}_X$ of (1.2.3) are approximately the same.

Therefore it makes sense to go for *approximate* solutions of the linear systems, for instance by projecting the right-hand sides to spaces spanned by eigenvectors corresponding to large eigenvalues. One way to achieve this is to calculate a **singular value decomposition** first and then use only the subsystem corresponding to large singular values. This works well beyond the standard condition limits, as we shall demonstrate now. This analysis will apply without changes to all linear systems appearing in this book.

Let $\mathbf{G}$ be an $m \times n$ matrix and consider the linear system

$$\mathbf{G}\mathbf{x} = \mathbf{b} \in I\!R^m \tag{2.5.1}$$

which is to be solved for a vector $\mathbf{x} \in I\!R^n$. The system may arise from any method using radial basis functions, including (1.2.1), (1.6.3), (1.6.5), (2.2.1) and those of subsequent chapters, e.g. (4.2.7), and (5.4.4). In case of collocation (Chapter 3) or the Method of Fundamental Solutions (Chapter 4), or already for the simple recovery problem (1.2.1) there may be more test or collocation points than trial centers or source points. Then the system will have $m \geq n$ and it usually is overdetermined.

But if the user has chosen enough well-placed trial centers and a suitable radial basis function for constructing trial functions, the previous section told us that chances are good that the true solution can be well approximated by functions from the trial space. Then there is an approximate solution $\hat{\mathbf{x}}$ which at least yields $\|\mathbf{G}\hat{\mathbf{x}} - \mathbf{b}\|_2 \leq \eta$ with a small tolerance η, and which has a coefficient vector $\hat{\mathbf{x}}$ representable on a standard computer. Note that η may also contain noise of a certain unknown level. The central problem is that there are many vectors $\hat{\mathbf{x}}$ leading to small values of $\|\mathbf{G}\hat{\mathbf{x}} - \mathbf{b}\|_2$, and the selection of just one of them is an unstable process. But the reproduction quality is much more important than the actual accuracy of the solution vector $\hat{\mathbf{x}}$, and thus matrix condition alone is not the right aspect here.

Clearly, any reasonably well-programmed least-squares solver [GvL96] should do the job, i.e. produce a numerical solution $\tilde{\mathbf{x}}$ which solves

$$\min_{\mathbf{x} \in \mathbb{R}^n} \|\mathbf{G}\mathbf{x} - \mathbf{b}\|_2 \tag{2.5.2}$$

or at least guarantees $\|\mathbf{G}\tilde{\mathbf{x}} - \mathbf{b}\|_2 \leq \eta$. It should at least be able not to overlook or discard $\hat{\mathbf{x}}$. This **regularization** by **optimization** works in many practical cases, but we shall take a closer look at the joint error and stability analysis, because even an optimizing algorithm will recognize that it has problems in determining $\hat{\mathbf{x}}$ reliably if columns of the matrix $\mathbf{G}$ are close to being linearly dependent.

By **singular value decomposition** (**SVD**) [GvL96], the matrix $\mathbf{G}$ can be decomposed into

$$\mathbf{G} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T \tag{2.5.3}$$

where $\mathbf{U}$ is an $m \times m$ orthogonal matrix, $\mathbf{\Sigma}$ is an $m \times n$ matrix with zeros except for **singular values** $\sigma_1, \ldots, \sigma_n$ on the diagonal, and where $\mathbf{V}^T$ is an $n \times n$ orthogonal matrix. Due to some sophisticated numerical tricks, this decomposition can under normal circumstances and within standard accuracy limits be done with $\mathcal{O}(mn^2 + nm^2)$ complexity, though it needs an eigenvalue calculation. One can assume

$$\sigma_1^2 \geq \sigma_2^2 \geq \ldots \geq \sigma_n^2 \geq 0,$$

and the σ_j^2 are the nonnegative eigenvalues of the positive semidefinite $n \times n$ matrix $\mathbf{G}^T\mathbf{G}$.

The condition number of the non–square matrix $\mathbf{G}$ is then usually defined to be σ_1/σ_n. This is in line with the usual **spectral condition number** $\|\mathbf{G}\|_2\|\mathbf{G}^{-1}\|_2$ for the symmetric case $m = n$. The numerical computation of $\mathbf{U}$ and $\mathbf{V}$ usually is rather stable, even if the total condition is extremely large, but the calculation of small singular values is hazardous. Thus the following arguments can rely on $\mathbf{U}$ and $\mathbf{V}$ but not on small singular values.

Using (2.5.3), the solution of either the minimization problem (2.5.2) or, in the case $m = n$, the solution of (2.5.1) can be obtained and analyzed as follows. We first introduce new vectors

$$\mathbf{c} := \mathbf{U}^T\mathbf{b} \in I\!R^m \text{ and } \mathbf{y} := \mathbf{V}^T\mathbf{x} \in I\!R^n$$

by transforming the data and the unknowns orthogonally. Since orthogonal matrices preserve Euclidean lengths, we rewrite the squared norm as

$$\begin{aligned} \|\mathbf{Gx} - \mathbf{b}\|_2^2 &= \|\mathbf{U\Sigma V}^T\mathbf{x} - \mathbf{b}\|_2^2 \\ &= \|\mathbf{\Sigma V}^T\mathbf{x} - \mathbf{U}^T\mathbf{b}\|_2^2 \\ &= \|\mathbf{\Sigma y} - \mathbf{c}\|_2^2 \\ &= \sum_{j=1}^{n}(\sigma_j y_j - c_j)^2 + \sum_{j=n+1}^{m} c_j^2 \end{aligned}$$

where now $y_1, \ldots, y_n$ are variables. Clearly, the minimum exists and is given by the equations

$$\sigma_j y_j = c_j,\ 1 \leq j \leq n,$$

but the numerical calculation runs into problems when the σ_j are small and imprecise in absolute value, because then the resulting y_j will be large and imprecise. The final transition to the solution $\mathbf{x} = \mathbf{Vy}$ by an orthogonal transformation does not improve the situation.

If we assume existence of a good solution candidate $\hat{\mathbf{x}} = \mathbf{V}\hat{\mathbf{y}}$ with $\|\mathbf{G}\hat{\mathbf{x}} - \mathbf{b}\|_2 \leq \eta$, we have

$$\sum_{j=1}^{n}(\sigma_j \hat{y}_j - c_j)^2 + \sum_{j=n+1}^{m} c_j^2 \leq \eta^2. \tag{2.5.4}$$

A standard **regularization** strategy to construct a reasonably stable approximation $\mathbf{y}$ is to choose a positive tolerance ϵ and to define

$$y_j^\epsilon := \begin{cases} \frac{c_j}{\sigma_j} & |\sigma_j| \geq \epsilon \\ 0 & |\sigma_j| < \epsilon \end{cases}$$

i.e. to ignore small singular values, because they are usually polluted by roundoff and hardly discernible from zero. This is called the **truncated singular value decomposition** (**TSVD**). Fortunately, one often has small c_j^2 whenever σ_j^2 is small, and then chances are good that

$$\|\mathbf{Gx}^\epsilon - \mathbf{b}\|_2^2 = \sum_{\substack{1 \leq j \leq n \\ |\sigma_j| \geq \epsilon}} c_j^2 + \sum_{j=n+1}^{m} c_j^2 \leq \eta^2$$

holds for $\mathbf{x}^\epsilon = \mathbf{V}\mathbf{y}^\epsilon$.

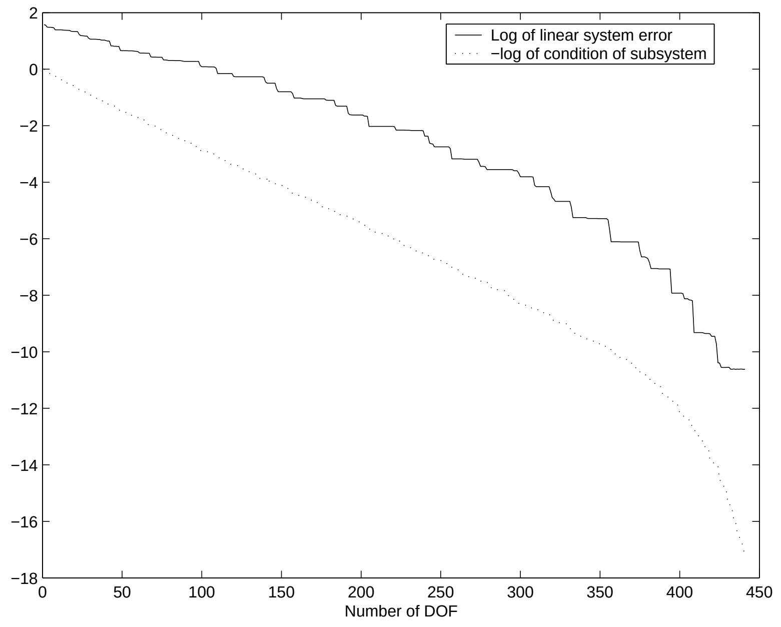

Fig. 2.7. Error and condition of linear subsytems via SVD

Figure 2.7 is an example interpolating the MATLAB `peaks` function in $m = n = 441$ regular points on $[-3, 3]^2$ by Gaussians with scale 1, using the standard system (1.2.2). Following a fixed 441×441 singular value decomposition, we truncated after the k largest singular values, thus using only k degrees of freedom. The results for $1 \leq k \leq 441$ show that there are low-rank subsystems which already provide good approximate solutions. A similar case for the Method of Fundamental Solutions will be provided by Example 4.5.1 in Chapter 4.

But now we proceed with our analysis. In case of large c_j for small σ_j, truncation is insufficient, in particular if the dependence on the unknown noise level η comes into focus. At least, the numerical solution should not spoil the reproduction quality guaranteed by (2.5.4), which is much more important than an exact calculation of the solution coefficients. Thus one can minimize $\|\mathbf{y}\|_2^2$ subject to the essential constraint

$$\sum_{j=1}^{n}(\sigma_j y_j - c_j)^2 + \sum_{j=n+1}^{m} c_j^2 \leq \eta^2, \tag{2.5.5}$$

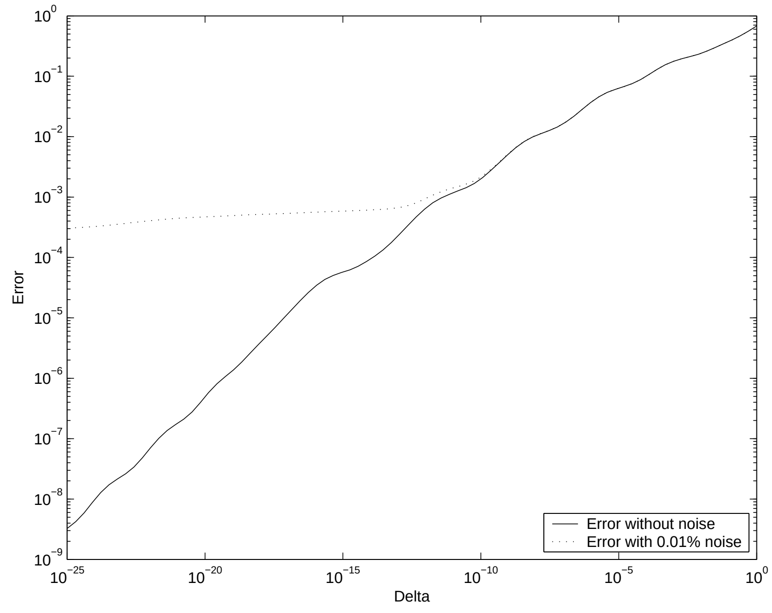

Fig. 2.8. Error as function of regularization parameter δ^2

but we suppress details of the analysis of this optimization problem. Another, more popular possibility is to minimize the objective function

$$\sum_{j=1}^{n}(\sigma_j y_j - c_j)^2 + \delta^2 \sum_{j=1}^{n} y_j^2$$

where the positive weight δ allows placing more emphasis on small coefficients if δ is increased. This is called **Tikhonov regularization**.

The solutions of both settings coincide and take the form

$$y_j^\delta := \frac{c_j \sigma_j}{\sigma_j^2 + \delta^2},\ 1 \le j \le n,$$

depending on the positive parameter δ of the Tikhonov form, and for $\mathbf{x}^\delta := \mathbf{V}\mathbf{y}^\delta$ we get

$$\|\mathbf{G}\mathbf{x}^\delta - \mathbf{b}\|_2^2 = \sum_{j=1}^{n} c_j^2 \left(\frac{\delta^2}{\delta^2 + \sigma_j^2}\right)^2 + \sum_{j=n+1}^{m} c_j^2,$$

which can be made smaller than η^2 for sufficiently small δ. The optimal value δ^* of δ for a known noise level η in the sense of (2.5.5) would be

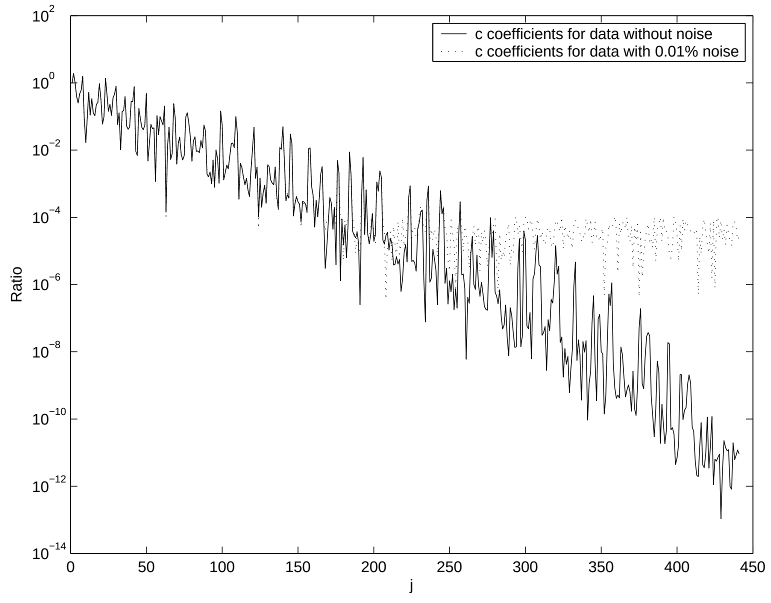

Fig. 2.9. Coefficients $|c_j|$ as function of j

defined by the equation $\|\mathbf{G}\mathbf{x}^{\delta^*} - \mathbf{b}\|_2^2 = \eta^2$, but since the noise level is only rarely known, users will be satisfied to achieve a tradeoff between reproduction quality and stability of the solution by inspecting the error $\|\mathbf{G}\mathbf{x}^{\delta} - \mathbf{b}\|_2^2$ for varying δ experimentally.

We now repeat the example leading to Figure 2.7, replacing the truncation strategy by the above regularization. Figure 2.8 shows how the error $\|\mathbf{G}\mathbf{x}^{\delta} - \mathbf{b}\|_{\infty,X}$ depends on the regularization parameter δ. In case of noise, users can experimentally determine a good value for δ even for an unknown noise level. The condition of the full matrix was calculated by MATLAB as $1.46 \cdot 10^{19}$, but it may actually be higher. Figure 2.9 shows that the coefficients $|c_j|$ are indeed rather small for large j, and thus regularization by truncated SVD will work as well in this case.

From Figures 2.9 and 2.8 one can see that the error $\|\mathbf{G}\mathbf{x}^{\delta} - \mathbf{b}\|$ takes a sharp turn at the noise level. This has led to the L–**curve method** for determining the optimal value of δ, but the L-curve is defined differently as the curve

$$\delta \mapsto (\log \|\mathbf{y}^{\delta}\|_2^2, \log \|\mathbf{G}\mathbf{x}^{\delta} - \mathbf{b}\|_2^2).$$

The optimal choice of δ is made where the curve takes its turn, if it does

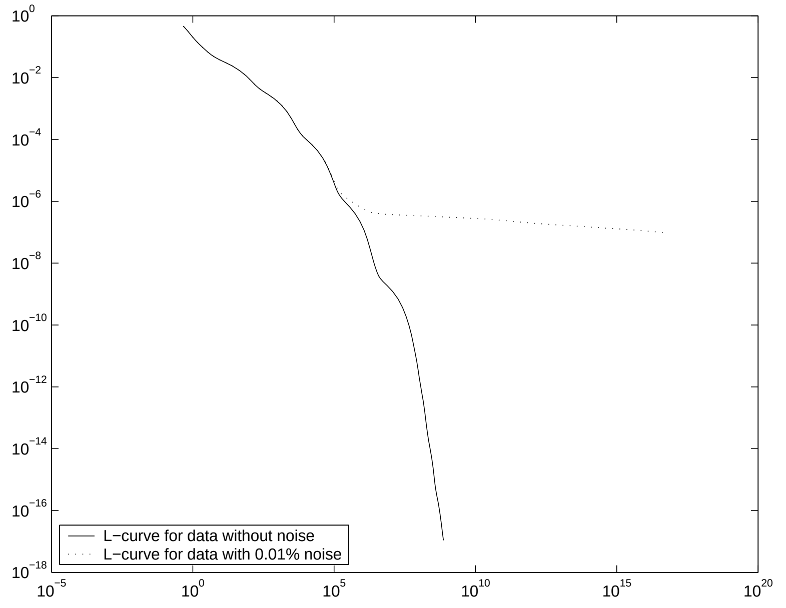

Fig. 2.10. The L-curve for the same problem

so, and there are various ways to estimate the optimal δ (see [Han92, Han94, Han00]) including a MATLAB software package.

Figure 2.10 shows the typical L-shape of the L-curve in case of noise, while in the case of exact data there is no visible sharp turn within the plot range. The background problem is the same as for the previous figures. A specific example within the Method of Fundamental Solutions will be presented in Section 4.9 on inverse problems.

Consequently, users of radial basis function techniques are strongly advised to take some care when choosing a linear system solver. The solution routine should incorporate a good regularization strategy or at least automatically project to stable subspaces and not give up quickly due to bad condition. Further examples for this will follow in later chapters of the book.

But for large systems the above regularization strategies are debatable. A singular-value decomposition of a large system is computationally expensive, and the solution vector will usually not be sparse, i.e. the evaluation of the final solution at many points is costly. In Section 2.9 we shall demonstrate that linear systems arising from radial basis functions often have good approximate solutions with only few nonzero coefficients. The corresponding numerical techniques are other, and possibly preferable regularizations which still are under investigation. A simple but efficient recent strategy [Sch07a] is to project the right–hand side of the system (2.5.1) to the span of adaptively selected columns of $\mathbf{G}$.

2.6 Scaling

If radial basis functions are used directly, without any additional tricks and treats, users will quickly realize that **scaling** is a crucial issue. The literature has two equivalent ways of scaling a given radial basis function ϕ, namely replacing it by either $\phi(\|\mathbf{x}-\mathbf{y}\|_2/c)$ or by $\phi(\epsilon\|\mathbf{x}-\mathbf{y}\|_2)$ with c and ϵ being positive constants. Of course, these scalings are equivalent, and the case $\epsilon \to 0,\ c \to \infty$ is called the **flat limit** [DF02]. In numerical methods for solving differential equations, the **scale factor** c is preferred, and it is called **shape factor** there. Readers should not be irritated by slightly different ways of scaling, e.g.

$$\phi_c(\|\mathbf{x}\|_2) := \sqrt{c^2 + \|\mathbf{x}\|_2^2} = c \cdot \sqrt{1 + \frac{\|\mathbf{x}\|_2^2}{c^2}} = c \cdot \phi_1\left(\frac{\|\mathbf{x}\|_2}{c}\right) \qquad (2.6.1)$$

for multiquadrics, because the outer factor c is irrelevant when forming trial spaces from functions (1.1.1). Furthermore, it should be kept in mind that only the **polyharmonic spline** and its special case, the **thin–plate spline**, generate trial spaces which are scale-invariant.

Like the tradeoff between error and stability when choosing smoothness (see the preceding section), there often is a similar tradeoff induced by scaling: a “wider” scale improves the error behavior but induces instability. Clearly, radial basis functions in the form of sharp spikes will lead to nearly diagonal and thus well-conditioned systems (1.2.2), but the error behavior is disastrous, because there is no reproduction quality between the spikes. The opposite case of extremely “flat” and locally close to constant radial basis functions leads to nearly constant and thus badly conditioned matrices, while many experiments show that the reproduction quality is even improving when scales are made wider, as far as the systems stay solvable.

For **analytic** radial basis functions (i.e. in C^∞ with an expansion into a power series), this behavior has an explanation: the interpolants often converge towards polynomials in spite of the degeneration of the linear systems [DF02, Sch05, LF05, LYY05, Sch06a]. This has implications for many examples in this book which approximate analytic solutions of partial differential equations by analytic radial basis functions like

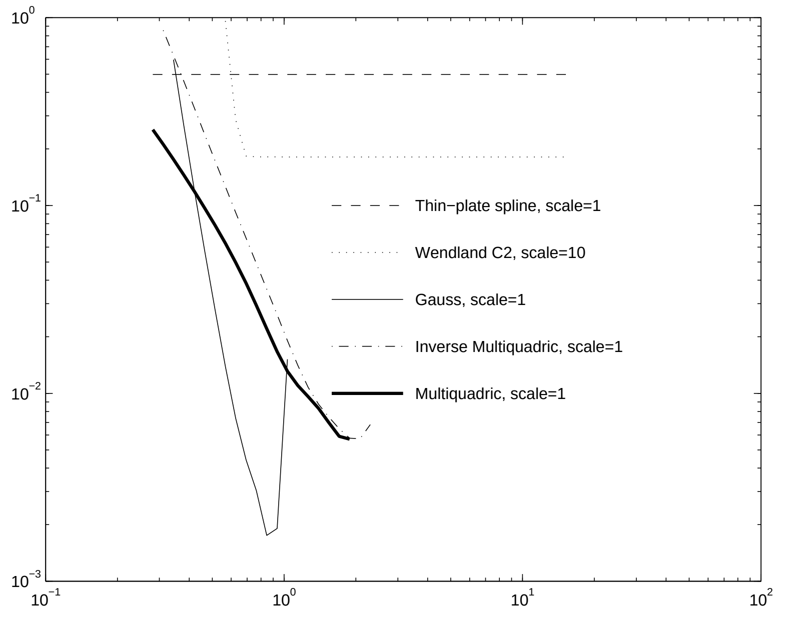

Fig. 2.11. Error as function of relative scale, smooth case

Gaussians or multiquadrics: whatever is calculated is close to a good polynomial approximation to the solution. Users might try using polynomials right away in such circumstances, but the problem is to pick a good polynomial basis. For multivariate problems, choosing a good polynomial basis must be data-dependent, and it is by no means clear how to do that. It is one of the intriguing properties of analytic radial basis functions that they automatically choose good data-dependent polynomial bases when driven to their "flat limit". There are new techniques [LF03, FW04] which circumvent the instability at large scales, but these are still under investigation.

Figure 2.11 shows the error for interpolation of the smooth MATLAB `peaks` function on a fixed data set, when interpolating radial basis functions ϕ are used with varying scale relative to a ϕ-specific starting scale given in the legend. Only those cases are plotted which have both an error smaller than 1 and a condition not exceeding 10^{12}. Since the data come from a function which has a good approximation by polynomials, the analytic radial basis functions work best at their condition limit. But since the `peaks` function is a superposition of Gaussians of different

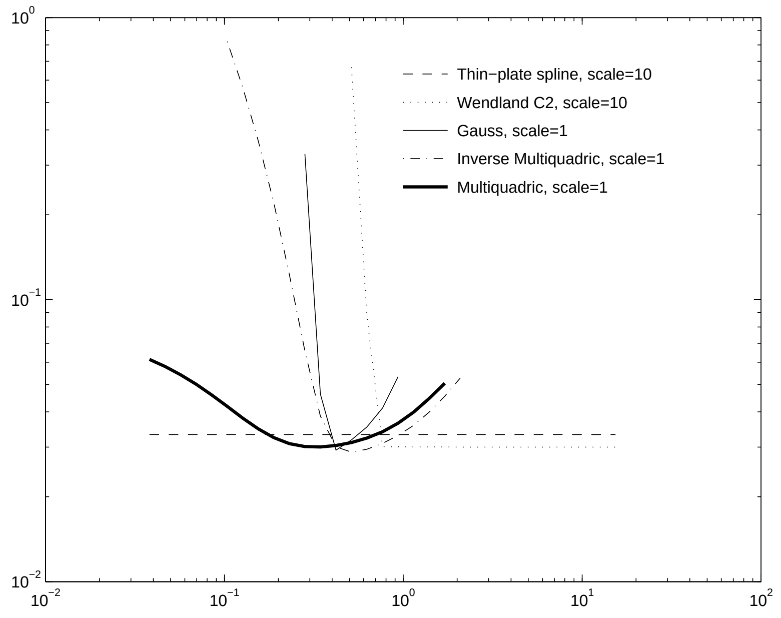

Fig. 2.12. Error as function of relative scale, nonsmooth case

scales, the Gaussian radial basis function still shows some variation in the error as a function of scale.

Interpolating the nonsmooth function (2.3.2) shows a different behavior (see Figure 2.12), because now the analytic radial basis functions have no advantage for large scales. In both cases one can see that the analytic radial basis functions work well only in a rather small scale range, but there they beat the other radial basis functions. Thus it often pays off to select a good scale or to circumvent the disadvantages of large scales [LF03, FW04].

As in finite element methods, users might want to scale the basis functions in a data-dependent way, making the scale c in the sense of using $\phi(\|\mathbf{x} - \mathbf{y}\|_2/c)$ proportional to the fill distance h as in (2.3.1). This is often called a **stationary** setting, e.g. in the context of wavelets and quasi-interpolation. If the scale is fixed, the setting is called **nonstationary**, and this is what we have been considering up to this point. Users must be aware that the error and stability analysis, as described in the previous sections, apply to the nonstationary case, while the stationary case will not converge for $h \to 0$ in case of unconditionally

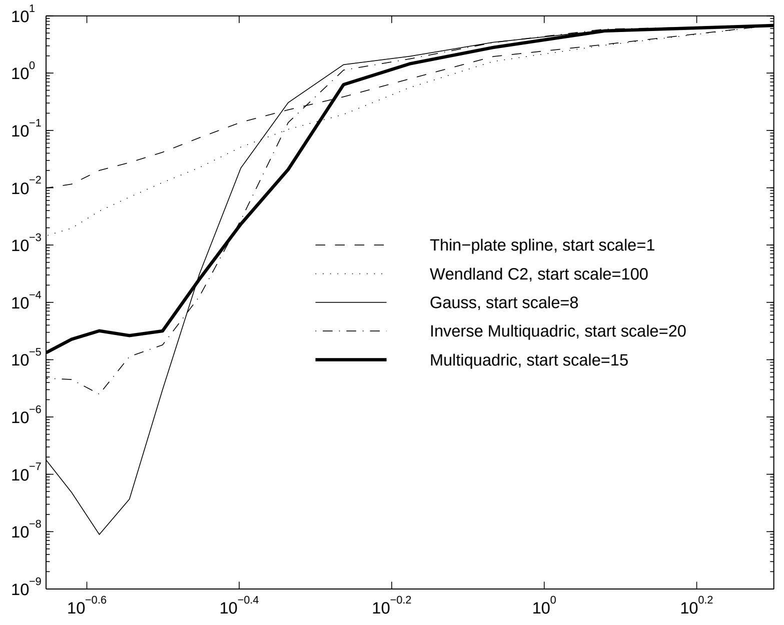

Fig. 2.13. Stationary interpolation to a smooth function at small starting scales

positive definite radial basis functions [Buh88, Buh90]. But there is a way out: users can influence the "relative" scale of c with respect to h in order to achieve a good compromise between error and stability. The positive effect of this can easily be observed [Sch97], and for special situations there is a sound theoretical analysis called **approximate approximation** [MS96]. Figure 2.13 shows the stationary error behavior for interpolation of the smooth MATLAB `peaks` function when using different radial basis functions ϕ at different starting scales. It can be clearly seen how the error goes down to a certain small level depending on the smoothness of ϕ and then stays roughly constant. Using larger starting radii decreases these saturation levels, as Figure 2.14 shows.

Due to the importance of *relative* scaling, users are strongly advised to always run their programs with an *adjustable* scale of the underlying radial basis functions. Experimenting with small systems at different scales give a feeling of what happens, and users can fix the relative scale of c versus h rather cheaply. Final runs on large data can then use this relative scaling. In many cases, given problems show a certain "intrinsic" preference for a certain scale, as shown in Figure 2.12, but this

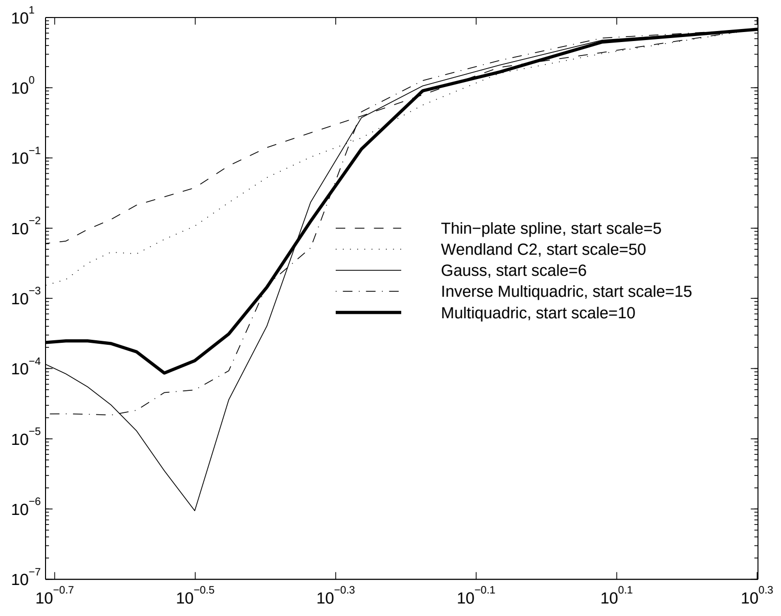

Fig. 2.14. Stationary interpolation to a smooth function at wider starting scales

is an experimental observation which still is without proper theoretical explanation.

2.7 Practical Rules

If users adjust the smoothness and the scaling of the underlying radial basis function along the lines of the previous sections, chances are good to get away with relatively small and sufficiently stable systems. The rest of the book contains plenty of examples supporting this observation.

For completeness, we add a few rules for Scientific Computing with radial basis functions, in particular concerning good choices of scale and smoothness. Note that these apply also to methods for solving partial differential equations in later chapters.

- Always allow a scale adjustment.
- If possible, allow different RBFs to choose from.
- Perform some experiments with scaling and choice of RBF before you turn to tough systems for final results.

- If you do not apply iterative solvers, do not worry about large condition numbers, but use a stabilized solver, e.g. based on Singular Value Decomposition (SVD). Remember that unless you apply certain tricks, getting a good reproduction quality will always require bad condition. If you need k decimal digits of final accuracy for an application, do not bother about condition up to 10^{12-k}.
- If you use compactly supported radial basis functions, do not expect them to work well when each support contains less than about 50 neighbors. This means that the bandwidth of large sparse systems should not be below 50. Increasing bandwidth will usually improve the quality of the results at the expense of computational complexity.
- When using either compactly supported or quickly decaying radial basis functions of high smoothness, the theoretical support and the practical support do not coincide. In such cases one should enforce sparsity by chopping the radial basis functions, in spite of losing positive definiteness properties. But this should be done with care, and obeying the "50 neighbors" rule above.
- If systems get large and ill–conditioned, and if change of scale and RBF do not improve the situation, try methods described in the following section.
- Use blockwise iteration ("domain decomposition") first, because it is simple and often rather efficient.
- Blockwise iteration can be speeded up by precalculation of LR decompositions of blocks.
- If all of this does not work, try partitions of unity, multilevel methods, or special preconditioning techniques. You are now at current research level, and you should look into the next section.

2.8 Large Systems: Computational Complexity

Handling unavoidably large problems raises questions of computational complexity which deserve a closer look. First, there is the difference between the complexities of *solving* and *evaluation*. The latter addresses the evaluation of trial functions like (1.2.5) for large n at many evaluation points $x \in \mathbb{R}^d$, while the former concerns the calculation of the coefficients.

Evaluation complexity can be kept at bay by **localization** techniques needing only a few "local" coefficients to evaluate the trial function. There are several possibilities for localization:

- Using **compactly supported radial basis functions** [Wu95, Wen95, Buh98] leads to sparse systems and localized evaluation. In particular, Wendland's functions have been applied successfully in plenty of applications, e.g. [FI96, CBP99, SW99, CBP99, CMC99, WHL$^+$99, CYT00a, CYT00b, GCG00, MYP$^+$01, CGGC02, WHG02a, WHG02b, KSBH02, OBS03, Fas03, RTSD03, WK05, ABW06] and many others. Since the correspondent radial basis functions have limited smoothness (and thus low convergence rates, following section 2.3), the error will be larger than when using analytic radial basis functions, but the stability is much better. However, they again need careful scaling, which now influences the evaluation complexity and the sparsity. "Flat" scaling improves the error behavior at the price of increasing instability and complexity. Together with **thinning algorithms** providing data at different resolution levels, compactly supported radial basis functions also allow efficient **multiscale techniques** [FI96, Fas98, Fas99, NSW99, GFJ00, CGGC02, Isk04].
- **partition of unity** methods [MB96, BM97, GS00, GS02a, GS02b, Wen02, Sch03, OBS03, TRS04] are a flexible and general tool for localizing **any** trial space. They have the advantage of not spoiling the local error behavior of the original trial space while localizing both the evaluation and the solving. The basic idea is to start with a selection of smooth "weight" functions $\varphi_i : I\!R^d \to I\!R$ with overlapping compact supports Ω_i and summing up to 1 globally. If a trial space U_i is given on each of the subdomains Ω_i, a global trial function u can be superimposed from local trial functions $u_i \in U_i$ by the localized summation

  $$u(\mathbf{x}) := \sum_i u_i(\mathbf{x})\varphi_i(\mathbf{x}) = \sum_{i\,:\,\mathbf{x}\in\Omega_i} u_i(\mathbf{x})\varphi_i(\mathbf{x}).$$

  Depending on the problem to be solved, one can plug the above representation into the full problem or use local methods to generate the local trial functions u_i more or less independently, thus localizing also the solution stage. This class of techniques deserves much more attention from scientists and engineers working in applications.
- **multipole expansions** work best for radial basis functions with series expansions around infinity. They aggregate "far" points into "panels" and use expansions to simplify evaluation. This technique is very successful in certain applications [BN92, BG97, BL97, BC00, BCR00, CBN02] though it is not easy to code.

- Fast evaluation using **transforms** is another choice [PS03, FS04, RB05], but it has not yet found its way into applications.

The dominant methods for reducing the complexity of *solving* large systems like (1.2.2) are **domain decomposition** and **preconditioning**. In classical analysis, domain decomposition means the splitting of a boundary value problem into smaller boundary value problems, using interface conditions for coupling the local problems. This was also done for problems solved via radial basis functions (e.g. [ZHL03a, LH04, ICT04]), but the majority of authors working with radial basis functions use the term in a different way. We explain it below, together with its close connection to preconditioning.

Of course, one can solve a huge problem (1.2.2) or (1.2.7) by a block–wise Gauss-Seidel or Jacobi iteration, where each "block" is defined by taking a small set of points in a small subdomain. Each block defines a local linear system where the unused data are shifted to the right-hand side [WHL$^+$99]. These local systems are solved independently and in turn. It does not matter whether the domains overlap or not. In most cases the numerical results for suitably chosen subdomains usually are much better than for direct iterative methods. In particular, the LU decompositions of the small local systems can be stored and re-used all over again in order to save computation time. Furthermore, there are different strategies for choosing "good" blocks.

This basic technique comes in various forms. It can be reformulated as a **block–wise preconditioner** or as a **Krylov subspace iteration** [FP00, FGP05] employing local **cardinal functions** in case of plain interpolation [BCM99, BLB00, KH00, Mou01, LK04, LK05, BLKL05]. It can also be seen as an additive [HW00] or as a multiplicative **Schwarz decomposition** [BLB00] depending whether Jacobi or Gauss-Seidel is used as the inner iteration. For regular data these preconditioners can achieve a fixed accuracy by a fixed number of iterations of the conjugate gradient method which is not dependent on the number of equations [Bax02].

Altogether, there are many successful techniques for handling large and ill–conditioned systems (see e.g. an overview in [KH00]), and there are a few promising theoretical investigations [BCM99, FP00, Bax02, BLKL05, FGP05, Sch05], but a general and complete mathematical foundation for handling large RBF–based systems arising in Partial Differential Equations still is missing.

2.9 Sensitivity to Noise

So far the discussion has focused on noiseless data, with the exception of Figure 2.8. If users expect **noise** in the data, an interpolatory recovery along the lines of Section 2.1 is not appropriate, because it treats noise as data. In most of the later sections of this book, data are right-hand sides or boundary values for partial differential equations, and they usually are given as noiseless functions which can be evaluated anywhere. Thus the rest of the book does not treat noisy inputs in detail. But at this point some remarks are appropriate.

In all noisy situations, interpolation should be replaced by approximation. This can be done in various ways leading to **stabilization**.

A primitive, but often quite sufficient technique is to run a smoothing process on the raw data and to recover the unknown function from the smoothed data instead of the raw data.

Another standard trick is to solve (1.2.2) in the L_2 sense with oversampling, if only $n << m$ trial centers $\mathbf{x}_j$ are used for m data points $\mathbf{y}_k$. The trial centers can then be placed rather freely with a large separation distance, while a small separation distance of data points will not have a dramatic effect on stability any more. However, there is not very much theoretical and practical work done on unsymmetric recovery techniques [Sch07b, Sch06c, Sch06a].

A third possibility is the old Levenberg–Marquardt trick of adding a positive value λ into the diagonal of the **kernel matrix** of (1.2.2) with entries $\phi(\|\mathbf{x}_j - \mathbf{x}_k\|_2)$. As is well-known from literature on spline smoothing, this leads to an approximant achieving a tradeoff between smoothness and reproduction quality which can be controlled by λ. If a stochastic background is available, there are methods to estimate λ properly, e.g. by **cross–validation**. However, in most cases users adjust λ experimentally. This technique also helps to fight instability when working on irregularly distributed data [WR05], because it is able to shift the stability from dependence on the separation distance to dependence on the fill distance (see Section 2.4).

A fourth possibilty is **regularization**, for example, using a singular-value decomposition as described in Section 2.5.

In general, one can replace the system (1.2.2) by an **optimization method** which penalizes the reproduction error on the one hand and either a complexity or smoothness criterion on the other, allowing a fair amount of control over the tradeoff between error and stability. Penalties for the discrete reproduction error can be made in various discrete norms, the ℓ_1 and ℓ_∞ case having the advantage of leading to linear optimization restrictions, while the discrete ℓ_2 norm leads to quadratic ones. For radial basis functions of the form (1.1.1) or (1.2.5), the quadratic form

$$\|u\|_\phi^2 := \sum_{j,k=1}^{n} \alpha_j \alpha_k \phi(\|\mathbf{x}_j - \mathbf{x}_k\|_2) \tag{2.9.1}$$

is a natural candidate for penalizing high derivatives without evaluating any. This is due to the standard fact that the above expression is a squared norm in a **native Hilbert space** of functions with about half the smoothness of ϕ, penalizing all available derivatives there. For details, we have to refer to basic literature [Buh03, Wen05] on the theory of radial basis functions. But even though we skip over native spaces here, all users should be aware that they always lurk in the theoretical background, and that all methods based on radial basis functions implicitly minimize the above quadratic form under all functions in the native space having the same data. This has a strong **regularization** effect which is the background reason why radial basis functions or more general **kernel methods** work well for many **ill-posed** and **inverse problems** [HW03, Li04, TWN04, CC05b, CC05a, HW05, JZ05, Li05, Sai05, Nas06]. The above strategy of minimizing the quadratic form (2.9.1) also is central for modern methods of **machine learning**, but we cannot pursue this subject in detail here [CST00, SS02, STC04].

Let us use minimization of the quadratic form (2.9.1) to provide an example of the tradeoff between error and complexity. Again, the basic situation is interpolation to the MATLAB `peaks` function, this time in 14×14=196 regularly distributed points in $[-3,3]^2$ by Gaussians of scale 1. The global $L_\infty[-3,3]^2$ error of the exact interpolation on these data is 0.024, evaluated on a fine grid with 121×121=14641 points. But now we minimize the quadratic form (2.9.1) under the constraints

$$-\epsilon \leq \sum_{j=1}^{n} \alpha_j \phi(\|\mathbf{x}_j - \mathbf{x}_k\|_2) - f(\mathbf{x}_k) \leq \epsilon,\ 1 \leq k \leq n \tag{2.9.2}$$

for positive ϵ. The case of $\epsilon = 0$ is exact interpolation using all 196 data points and trial functions. For positive ϵ, the usual Karush-Kuhn-Tucker conditions imply that only those points $\mathbf{x}_k$ are actually used where one of the bounds in (2.9.2) is attained with equality. The number $n(\epsilon)$ of required points increases to the maximally possible $n(0) = 196$ when ϵ decreases. Figure 2.15 shows this for the case of exact and noisy data.

But even more interesting is the behavior of the global $L_\infty[-3,3]^2$

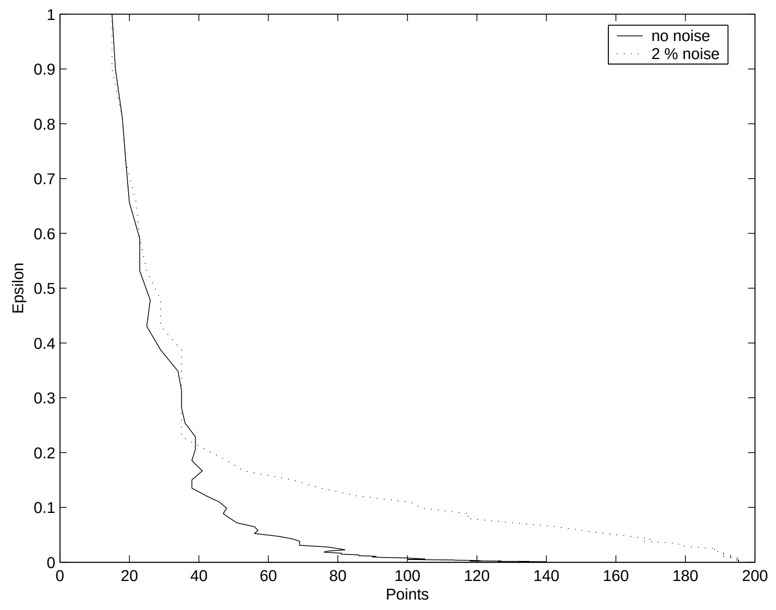

Fig. 2.15. Connection between ϵ and the number $n(\epsilon)$ of necessary points

error $E(\epsilon)$ as a function of ϵ. Figure 2.16 shows that $E(\epsilon)$ roughly follows the behavior of ϵ when plotted as a function of the required points $n(\epsilon)$. Both curves are experimentally available, and one can read off that the optimal choice of ϵ in the noisy case is at the point where the curve takes its L-turn, i.e. at the point of largest curvature around $n = 40$. This can be viewed as an experimental method to determine the noise level. Note that it does not pay off to use more points, and note the similarity to the L-curve technique [HO93].

But these curves are also useful for exact data,. Since the maximum value of the `peaks` function is about 8.17, one can get a relative global accuracy of 1% using roughly 60 points for exact data. It makes no sense to use the full 196 points, even for exact data, if exact results are not required. Of course, larger noise levels lead to smaller numbers of required points, but a thorough investigation of these tradeoff effects between error and complexity is still a challenging research topic.

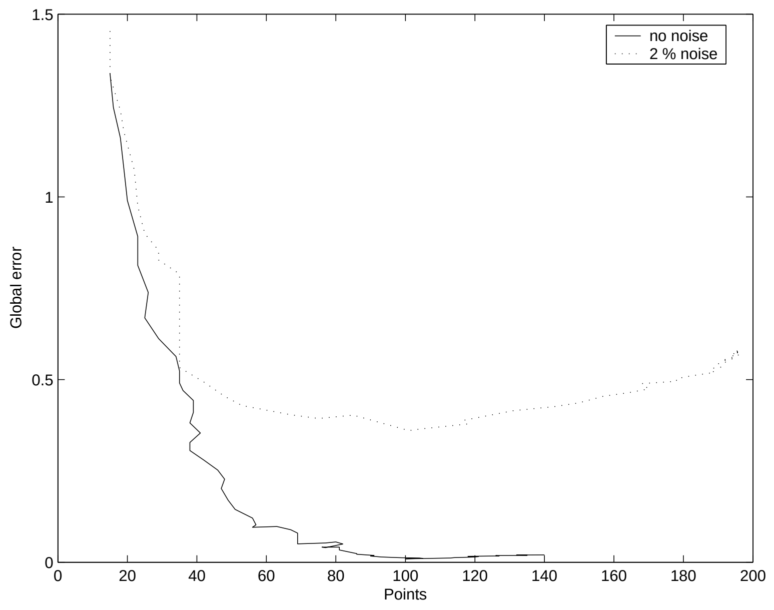

Fig. 2.16. Error $E(\epsilon)$ as a function of the number $n(\epsilon)$ of necessary points

2.10 Time–dependent Functions

Interpolation and approximation of time–dependent functions $u(\mathbf{x}, t)$ can easily be achieved by choosing a fixed spatial discretization via points $\mathbf{y}_1, \dots, \mathbf{y}_n$ and letting the coefficients in the representations (1.1.1) and (1.2.5) be time–dependent. If β_{ij} are the coefficients of the fixed inverse of the matrix in (1.2.2) (the case of (1.2.7) can be treated similarly), then the approximation to $u(\mathbf{x}, t)$ is of the simple form

$$\begin{aligned}\tilde{u}(\mathbf{x}, t) &= \sum_{k=1}^{n} \underbrace{\sum_{j=1}^{n} \beta_{kj} u(\mathbf{x}_j, t)}_{\alpha_k(t)} \phi(\|\mathbf{x} - \mathbf{x}_k\|_2) \\ &= \sum_{k=1}^{n} \alpha_k(t) \phi(\|\mathbf{x} - \mathbf{x}_k\|_2)\end{aligned} \qquad (2.10.1)$$

which can be plugged into other parts of the underlying problem. For instance, it allows an easy spatial resampling for fixed t, and it provides arbitrary approximate derivatives such as

$$\frac{\partial \tilde{u}}{\partial \mathbf{x}_j} = \sum_{k=1}^{n} \alpha_k(t) \frac{\partial \phi(\|\mathbf{x}-\mathbf{x}_k\|_2)}{\partial \mathbf{x}_j} = \sum_{k=1}^{n}\sum_{j=1}^{n} \beta_{kj} u(\mathbf{x}_j, t) \frac{\partial \phi(\|\mathbf{x}-\mathbf{x}_k\|_2)}{\partial \mathbf{x}_j}$$

in terms of time–dependent values of either u or the coefficients α_k. Formulas like this are useful when considering time-dependent meshless spatial discretizations, because they make resampling easy, avoiding remeshing.

But the above technique can also serve for **line methods** solving partial differential equations like

$$\frac{\partial u}{\partial t}(\mathbf{x}, t) = F(L(u(\mathbf{x}, t)), \mathbf{x}, t) \tag{2.10.2}$$

with a linear spatial differential operator L because of

$$\begin{aligned} \alpha_k'(t) &= \sum_{j=1}^{n} \beta_{kj} \frac{\partial \tilde{u}}{\partial t}(\mathbf{x}_j, t),\ 1 \le k \le n \\ &= \sum_{j=1}^{n} \beta_{kj} F\left(\sum_{k=1}^{n} \alpha_k(t) L(\phi(\|\mathbf{x}-\mathbf{x}_k\|_2))_{|\mathbf{x}=\mathbf{x}_j}, \mathbf{x}_j, t\right). \end{aligned}$$

leading to a system of ordinary differential equations. Along these lines. radial basis function methods will be used in later parts of the book, in particular in chapters 3 and 6.

3
Collocation Techniques

The history of Scientific Computing shows that important numerical techniques like

- finite difference methods,
- finite element methods,
- finite volume methods,
- boundary element methods,
- particle methods, and
- multipole methods

were introduced and improved by engineers or physicists working in applications. Similarly, the use of radial basis functions for scattered data interpolation, as described in Chapter 2, was first introduced in 1951 by the South African geophysicist D.G. Krige [Kri51] within a statistical background, and called **Kriging** afterwards. It was further developed by G. Matheron [Mat62] and used without any statistical background by the geophysicist R. Hardy [Har71] who introduced **multiquadrics**.

Also, the use of radial basis functions for solving partial differential equations was started not by a mathematician, but rather by the physicist E. Kansa [Kan86] who proposed the meshless collocation method for solving partial differential equations using multiquadrics, as sketched in Section 1.6. Since the resulting system (1.6.3) of linear equations is asymmetric, the method is now known as the **asymmetric collocation** method.

Subsequently, **meshless methods** based on radial basis functions have reached more and more application areas, because they provide efficient and easy-to-use numerical techniques for solving various kinds of partial differential equations. In the following sections, the advantages of the asymmetric meshless collocation method are demonstrated by solving some typical partial differential equation problems. Since it is impossible to give a comprehensive survey over all relevant applications, we confine ourselves to a few special cases. These include time-dependent problems where we use asymmetric collocation in the spatial variables only, letting the coefficients of the spatial representation (1.1.1) be time-dependent as in (2.10.1) of Section 2.10. Readers will notice that the general rules governing error, stability, and scaling will be the same as in the previous chapter.

For other time-dependent problems, the combination of the asymmetric meshless collocation method with **Laplace transform** techniques was successfully demonstrated in [ME94] to greatly reduce the spatial and temporal truncation errors, and we shall deal with this special approach in Chapter 6 because it is closely related to the Helmholtz equation arising in Chapters 4 and 5.

3.1 High Dimensional Problems

Due to the restrictive requirements of grid generation when using finite difference methods or mesh generation in finite elements, the extendability of these mesh-dependent numerical methods to solve high-dimensional problems or problems with complicated boundaries has been prohibitive. To illustrate the adaptability of the asymmetric meshless collocation method described in Section 1.6 for problems in higher dimensions, we consider the following

Example 3.1.1

$$\begin{array}{rcll} \Delta u & = & f^{\Omega} & \text{in } \Omega := [0,1]^d \\ u & = & f^{\Gamma} & \text{in } \Gamma := \partial\Omega, \end{array} \tag{3.1.2}$$

where d is the dimension of the space. This is of the form (1.6.1) with the **Laplace operator** $L := \Delta$.

The exact solution of (3.1.2) is given by

$$u(\mathbf{x}) = \prod_{i=1}^{d} \sin(x_i),\ \mathbf{x} = (x_1, x_2, \ldots, x_d)^T$$

from which the values f^{Ω} and f^{Γ} can then be determined.

For numerical comparison we use both the asymmetric and symmetric collocation methods described in Section 1.6 to solve the boundary value problem (3.1.2), respectively. In the computations, the radial basis function $\phi(r)$ is chosen to be r^5 to avoid scaling problems. To keep the maximum total number of points used in the computations less than 750, we chose the trial centers and the test points to be the same $\mathbf{x}_i, i = 1 \dots m := m_\Omega + m_\Gamma$ which are uniformly distributed in the domain $\Omega \cup \Gamma$ so that $m = s^d$ where $s = 5, \dots, 21$ for $d = 2$, $s = 5, \dots, 9$ when $d = 3$, $s = 4$ and 5 when $d = 4$. The maximal degree of the additional polynomials is 2.

By assuming that the solution u of (3.1.2) can be approximated by

$$u(\mathbf{x}) := \sum_{i=1}^{m} \alpha_i \phi(\|\mathbf{x} - \mathbf{x}_i\|_2) + \sum_{\ell=1}^{q} \beta_\ell p_\ell(\mathbf{x}),$$

where $\phi(r) := r^5$, it follows from Section 1.6 that the system of linear equations for asymmetric collocation is a special case of (1.6.3) in the form

$$\begin{array}{lclcll}
\sum_{k=1}^{m} \alpha_k \Delta\phi(\|\mathbf{x}_j^\Omega - \mathbf{x}_k\|_2) & + & \sum_{\ell=1}^{q} \beta_\ell \Delta p_\ell(\mathbf{x}_j^\Omega) & = & f^\Omega(\mathbf{x}_j^\Omega), & j \le m_\Omega \\
\sum_{k=1}^{m} \alpha_k \phi(\|\mathbf{x}_j^\Gamma - \mathbf{x}_k\|_2) & + & \sum_{\ell=1}^{q} \beta_\ell p_\ell(\mathbf{x}_j^\Gamma) & = & f^\Gamma(\mathbf{x}_j^\Gamma), & j \le m_\Gamma \\
\sum_{k=1}^{m} \alpha_k p_\ell(\mathbf{x}_k) & + & 0 & = & 0, & \ell \le q.
\end{array}$$

For symmetric collocation, the system of linear equations is similarly obtained from Section 1.6 as (1.6.5). Both resultant systems of equations for the undetermined coefficients are solved by Gaussian elimination. The results of computations for dimensions 2, 3, and 4 are listed in Tables 3.1 and 3.2. □

Two features can be read from these tables:

(i) For each fixed space dimension, the accuracy of the numerical solution improves with the total number of trial centers.

(ii) If rows with $m = s^d$ points are compared for fixed s, leading to a dimension-independent fill distance of $h = 1/2s$ in the L_∞ norm, the convergence rates of both asymmetric and symmetric collocation increase with the space dimension.

The first feature must be expected from results on interpolation and approximation, as summarized in Section 2.3, but the second observation is surprising. It is well known from certain stationary interpolation processes on regular data [Buh03], but its mathematical foundation must

Table 3.1. *Asymmetric collocation with $\phi(r) = r^5$*

Dimension	Points	Max Error	Average Error	Condition Number
2	5^2	2.605E-3	1.391E-3	2.673E+5
2	10^2	2.179E-4	1.345E-4	1.580E+8
2	15^2	5.774E-5	3.292E-5	3.926E+9
2	20^2	2.036E-5	1.200E-5	3.445E+10
3	5^3	2.521E-3	1.255E-3	9.420E+6
3	7^3	5.973E-4	3.756E-4	2.083E+8
3	9^3	2.126E-4	1.440E-4	1.964E+9
4	4^4	7.475E-3	2.788E-3	4.555E+7
4	5^4	1.041E-3	8.518E-4	6.313E+8
2	5^2	2.605E-3	1.391E-3	2.673E+5
3	5^3	2.521E-3	1.255E-3	9.420E+6
4	5^4	1.041E-3	8.518E-4	6.313E+8

Table 3.2. *Symmetric collocation with $\phi(r) = r^5$*

Dimension	Points	Max Error	Average Error	Condition Number
2	5^2	4.429E-3	3.138E-3	2.897E+6
2	10^2	4.667E-4	3.307E-4	1.924E+9
2	15^2	1.246E-4	8.440E-5	5.190E+10
2	20^2	5.011E-5	3.196E-5	4.888E+11
3	5^3	4.749E-3	2.054E-3	1.188E+8
3	7^3	2.811E-3	1.073E-3	5.054E+9
3	9^3	1.448E-3	5.895E-4	6.120E+10
4	4^4	6.489E-3	2.807E-3	1.370E+8
4	5^4	3.343E-3	1.668E-3	4.673E+9
2	5^2	4.429E-3	3.138E-3	2.897E+6
3	5^3	4.749E-3	2.054E-3	1.188E+8
4	5^4	3.343E-3	1.668E-3	4.673E+9

be omitted here. For solving partial differential equations, the observed increase of the convergence rate with the space dimension has no theoretical basis yet.

3.2 Irregular Boundary Shape Problems

Other than the capability of solving higher dimensional problems as shown in the previous section, RBF meshless method (Kansa's method) is also very attractive in solving problems with irregular domain. To demonstrate the simplicity of Kansa's method, we consider the following convection equation.

$$\begin{aligned}
\Delta u - x^2 y \frac{\partial u}{\partial x} &= e^x + e^y - x^2 y e^x, \quad &(x,y) \in \Omega, \\
u &= e^x + e^y, \quad &(x,y) \in \partial\Omega^D, \\
\frac{\partial u}{\partial n} &= \left(\nabla \left(e^x + e^y\right)\right) \cdot \mathbf{n}, \quad &(x,y) \in \partial\Omega^N,
\end{aligned} \tag{3.2.1}$$

where $\bar{\Omega} = \Omega \cup \partial\Omega$ is an amoeba-like domain (see Figure 3.1) and its boundary is defined by the following parametric equation:

$$\partial\Omega = \left\{ (r\cos\theta, r\sin\theta) : r = e^{\sin\theta}\left(\sin^2 2\theta\right) + e^{\cos\theta}\left(\cos^2 2\theta\right) \right\}.$$

Note that $\partial\Omega = \partial\Omega^D \cup \partial\Omega^N$ and $\partial\Omega^D \cap \partial\Omega^N = \emptyset$. Let $\partial\Omega^D$ be

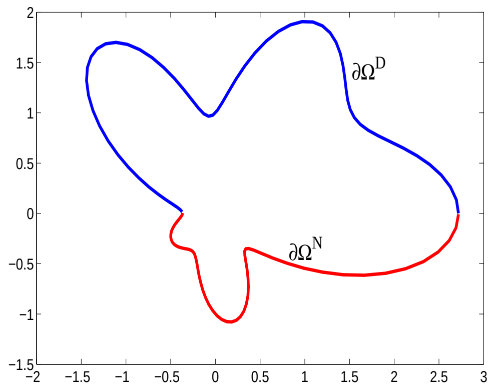

Fig. 3.1. Computational domain: Amoeba-like shape.

the boundary above the x-axis and $\partial\Omega^N$ be the boundary below the x-axis. The exact solution of (3.2.1) is given by $u(x,y) = e^x + e^y$ for all $(x,y) \in \bar{\Omega}$.

For numerical computation, we choose Inverse MQ ($\phi = 1/\sqrt{r^2+c^2}$) as the basis function. Furthermore, we select 300 uniformly distributed interior points, 100 uniformly distributed boundary points, and 120 randomly distributed test points inside the domain. The distribution of boundary and interior points are shown in Figure 3.2.

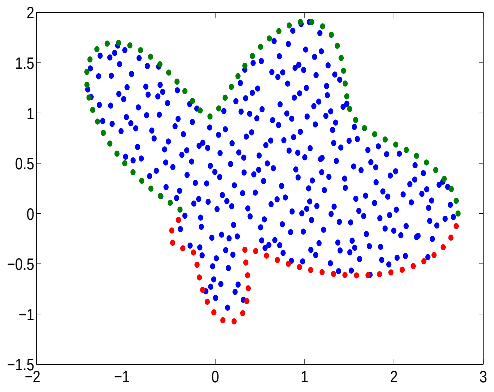

Fig. 3.2. 300 interior points and 100 boundary points.

We evaluate the Root Mean Square Error (RMSE) on the 120 test points to measure the numerical results. RMSE is defined as follows:

$$\text{RMSE} = \sqrt{\frac{1}{n_t}\sum_{i=1}^{n_t}\left(u(x_i, y_i) - \hat{u}(x_i, y_i)\right)^2},$$

where $\hat{u}$ is the approximate solution. The profile of RMSE with respect to various of shape parameter is shown in Figure 3.3. We observe that the accuracy is improving along with the increasing shape parameter until $c = 1.5$. After that, the solution becomes unstable and the best result is near $c = 2$. One has to be careful to balance between accuracy and stability.

The result in Figure 3.3 is obtained by the following MATLAB code. Before execute the program, we need to prepare four data sets: interior, boundary, test points, and unit normal vector at the boundary points with Neumann boundary condition. bdpt.txt contains all the points on the entire boundary and the same for all the unit normal vector in the file normal.txt. Let n_i be the number of interior points, m be the number of boundary points on the Neumann boundary, nb be the number of all the boundary points, and $n = n_i + nb$. Let $\{(x_j, y_j)\}_{j=1}^{m_1}$ and $\{(s_j, t_j)\}_{j=1}^{m_2}$ be two sets of points. We define the distance matrix DM of these two

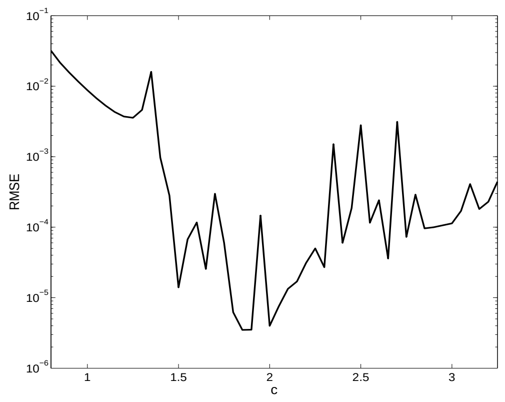

Fig. 3.3. RMSE verses shape parameter c.

set of points as follows:

$$DM = \begin{bmatrix} r_{11} & r_{12} & \cdots & r_{1m_2} \\ r_{21} & r_{22} & \cdots & r_{2m_2} \\ \vdots & \vdots & \ddots & \vdots \\ r_{m_11} & r_{m_12} & \cdots & r_{m_1m_2} \end{bmatrix}, \tag{3.2.2}$$

where $r_{ij} = \sqrt{(x_j - s_i)^2 + (y_j - t_i)^2}$.

Lines 16–21 establish the following block matrix $A1$ for governor equation using the interior points:

$$\begin{aligned} A1 &= \left[\Delta\phi_{ij} - x_j^2 y_j \frac{\partial \phi_{ij}}{\partial x}\right]_{1\le j\le n_i, 1\le i\le n}, \\ &= \left[r_{ij} - \frac{2c^2}{(r_{ij}+c^2)^{5/2}} - x_j^2 y_j \left(\frac{-(x_j - x_i)}{(r_{ij}+c^2)^{3/2}}\right)\right]_{1\le j\le n_i, 1\le i\le n.} \end{aligned} \tag{3.2.3}$$

Lines 22–23 establish the block matrix $A2$ for the Dirichlet boundary condition:

$$A2 = [\phi_{ij}] = \left[\left(r_{ij}^2 + c^2\right)^{-1/2}\right]_{1\le j\le n_b^D, 1\le i\le n.} \tag{3.2.4}$$

Note that in line 22, pdist2 is a MATLAB command which constructs a distance matrix containing pairwise distance between two sets. pdist2 is used to be available only in statistics toolbox. It is now available in the version R2010a. The readers who do not have updated version of

MATLAB can download the code from internet. DM in line 22 represents the distance matrix defined in (3.2.2) where the first set of data is the boundary points on Dirichlet boundary (BD) and the second set of points are all the points including boundary and interior (center(:,:)). Using the distance matrix DM, A2 in (3.2.4) can be established efficiently by line 23. Lines 24–29 formulate the block matrix $A3$ for the Neumann boundary condition using the boundary points below the x-axis.

$$A3 = \left[\frac{-(x_j - x_i)}{(r_{ij} + c^2)^{3/2}} \frac{\partial x_j}{\partial n} + \frac{-(y_j - y_i)}{(r_{ij} + c^2)^{3/2}} \frac{\partial y_j}{\partial n} \right]_{1 \le j \le n_b^N, 1 \le i \le n.}$$

In line 30, three block matrices are merged to form the resultant matrix of Kansa's method as follows:

$$A = \begin{bmatrix} A1 \\ A2 \\ A3 \end{bmatrix}.$$

Lines 36–38 perform the evaluation of approximate solutions at all the test points. The readers should be able to follow the rest of the code.

Program 3.2

```
 1   clear all;
 2   load -ascii testpoints.txt; % Evaluation points
 3   load -ascii intnode300.txt; % Interior points
 4   load -ascii DB.txt; % Dirichlet boundary points
 5   load -ascii NB.txt; % Neumann boundary points
 6   load -ascii NV.txt; % Normal vectors at NB.txt
 7   Xin = intnode300(:,1); Yin = intnode300(:,2);
 8   bdpt = [DB; NB];
 9   center = [intnode300;bdpt]; %interior+boundary points
10   ni = length(Xin); % # of Interior points
11   nb = length(bdpt); % # of Boundary points
12   m = length(NB); n = ni + nb;
13   c = 1.53; % shape parameter
     % Reset the used matrices for [A][Ci]=[F]
14   A1 = zeros(ni,n); A3 = zeros(m,n);
15   c2 = c * c;
     % [A1](ni x n): obtained by interior points
16   for i = 1:ni
17       r2 = (Xin(i) - center(:,1)).^2 + (Yin(i) - center(:,2)).^2;
```

```
% Laplace IMQ
    Lphir = (r2-2*c2)./((r2 + c2).∧(5/2));
% d/dx IMQ
    dxphir = -(Xin(i) - center(:,1))./((r2 + c2).∧(3/2));
    A1(i,:) = Lphir - (Xin(i)∧ 2) * Yin(i) * dxphir;
end
% [A2] (Dirichlet boundary condition)
DM = pdist2(DB,center);
A2 = 1 ./sqrt(DM.∧2 + c2);
% [A3] (Neumann boundary condition)
for i = 1:m
    r2 = (NB(i,1)-center(:,1)).∧2+(NB(i,2) - center(:,2)).∧2;
% d/dx IMQ
    dxphir = -(NB(i,1) - center(:,1))./((r2 + c2).∧(3/2));
% d/dy IMQ
    dyphir = -(NB(i,2) - center(:,2))./((r2 + c2).∧(3/2));
    A3(i,:) = dxphir * NV(i,1) + dyphir * NV(i,2);
end
% Resultant matrix of Kansa's method
A = [A1; A2; A3];
% forcing term
f1 = exp(Xin) + exp(Yin) - Xin.∧2.*Yin.*exp(Xin);
% Dirichlet boundary condition
g1 = exp(DB(:,1)) + exp(DB(:,2));
% Neumann boundary condition
g2 = exp(NB(:,1)).*NV(:,1) + exp(NB(:,2)).*NV(:,2);
% [F] (n x 1):
F = [f1; g1; g2];
% solve the linear system of equations
Ci = A \ F;
% Calculate the approximate solutions at all the test points
DM=pdist2(testpoints, center);
B = 1./sqrt(DM.∧2 + c2);
appro = B * Ci;
% exact solutions at the test points
exact = exp(testpoints(:,1)) + exp(testpoints(:,2));
% Calculate RMSE
RMSE = norm(exact-appro)/sqrt(length(testpoints));
fprintf('RMSE: %e\ n', RMSE)
```

3.3 Transport Problems

The advantage of using radial basis functions within collocation methods will further be illustrated by solving the nonlinear **Burgers equation**

$$u_t + uu_x = \nu u_{xx}, \qquad \nu > 0. \tag{3.3.1}$$

It is an extremely simplified limit form of the **Navier-Stokes equation**, and it models flows of viscous liquids as well as of cars in traffic. This equation has been studied by many researchers for the following reasons:

(i) It contains the simplest form of a nonlinear advection term uu_x and a dissipation term νu_{xx} for simulating the physical phenomena of wave motion,

(ii) its analytical solution was obtained by Cole [Col51], so that errors of numerical computations can be evaluated, and

(iii) its solutions show a shock wave behavior [Bur40] when the Reynolds number $1/\nu$ is large, i.e. when ν is small.

Various techniques have been applied to solve equation (3.3.1) numerically under the boundary conditions

$$u(0,t) = 0 = u(1,t),\ t > 0, \tag{3.3.2}$$

and the initial condition

$$u(x,0) = f(x),\ x \in \Omega = (0,1). \tag{3.3.3}$$

We first discretize the equation (3.3.1) by using a forward difference approximation for the time derivative to obtain

$$u^k + \delta_t(u^{k-1}u_x^k - \nu u_{xx}^k) = u^{k-1},\ k \geq 1, \tag{3.3.4}$$

where δ_t is the length of the time step and u^k denotes the k^{th} iterate of the solution. We approximate u^k at each iteration k by multiquadrics

$$u^k(x) \simeq \sum_{j=0}^{m} \alpha_j^k \sqrt{(x-x_j)^2 + c_j^2} + \alpha_{m+1}^k x + \alpha_{m+2}^k, \tag{3.3.5}$$

where x_j are $(m+1)$ distinct uniformly distributed centers in $[0,1]$. The scale factor c_j is one of the key factors for obtaining an accurate solution, but we do not elaborate on its choice here. We just note that an exponential variation in the scaling factors was proposed in [Kan90b], and the ill–conditioning problem was further studied experimentally in [KH00].

For each iteration k we have to determine the $m+3$ coefficients α_j^k for $0 \le j \le m+2$. The boundary conditions (3.3.2) already give the two equations

$$u^k(x_0) = 0 = u^k(x_m). \tag{3.3.6}$$

We then collocate u^k at $m+1$ distinct uniformly distributed collocation points $\hat{x}_j$ in $(0,1)$ using equation (3.3.4) to obtain

$$u^k(\hat{x}_j) + \delta_t \left(u^{k-1}(\hat{x}_j) \frac{d}{dx} u^k(\hat{x}_j) - \nu \frac{d^2}{dx^2} u^k(\hat{x}_j) \right) = u^{k-1}(\hat{x}_j), \quad k \ge 1, \tag{3.3.7}$$

for $j = 1, 2, \ldots, m+1$, where $u^0(\hat{x}_j)$ is taken to be $f(\hat{x}_j)$ from the initial condition (3.3.3). The system of equations (3.3.6) and (3.3.7) can then be solved by using Gaussian elimination with partial pivoting to obtain the coefficients α_j^k. Note that the test points x_j in (3.3.5) are different from the trial centers $\hat{x}_j$ in (3.3.7).

The analytical solution given in [Col51] for the Burgers equation (3.3.1) subject to the boundary conditions (3.3.2) and the initial condition (3.3.3) is

$$u(x,t) = \frac{2\pi\nu \displaystyle\sum_{k=1}^{\infty} k A_k \sin(k\pi x) \exp(-k^2 \nu \pi^2 t)}{A_0 + \displaystyle\sum_{k=1}^{\infty} A_k \cos(k\pi x) \exp(-k^2 \nu \pi^2 t)},$$

where

$$A_k = 2 \int_0^1 \cos(k\pi x) \exp\left(-\frac{1}{2\nu} \int_0^x f(y)dy \right) dx, \qquad k \ge 1,$$

and

$$A_0 = \int_0^1 \exp\left(-\frac{1}{2\nu} \int_0^x f(y)dy \right) dx.$$

In the case when $f(x) = \sin(\pi x)$, the analytical solution derived in [CS82] is

$$u(x,t) = \frac{4\pi\nu \displaystyle\sum_{k=1}^{\infty} k I_k(1/2\pi\nu) \sin(k\pi x) \exp(-k^2 \nu \pi^2 t)}{I_0(1/2\pi\nu) + 2\displaystyle\sum_{k=1}^{\infty} I_k(1/2\pi\nu) \cos(k\pi x) \exp(-k^2 \nu \pi^2 t)}, \tag{3.3.8}$$

where I_k denotes the modified Bessel function of first kind and order k. The difficulty in evaluating an accurate solution $u(x,t)$ based on the analytical formula (3.3.8) is due to the exponentially increasing term $I_k(1/2\pi\nu)$ in the series when ν is small. Using double precision floating-point numbers with a total word length of 64 bits, we can compute $u(x,t)$ to four decimal digits by evaluating the two series in formula (3.3.8) for ν down to 0.01 approximately. Unless an asymptotic formula is used, it is not possible to compute $u(x,t)$ using formula (3.3.8) for smaller ν. In fact, the magnitude of $I_k(1/2\pi\nu)$ already exceeds the limits of a 64 bit computation when $\nu \leq 1/4500$. For comparison purposes, in the case when $\nu = 0.0001$, we adopt the accurate solution computed in [CM78] using the Galerkin method with fully upwinded cubic functions and a particularly small value of the spatial fill distance h.

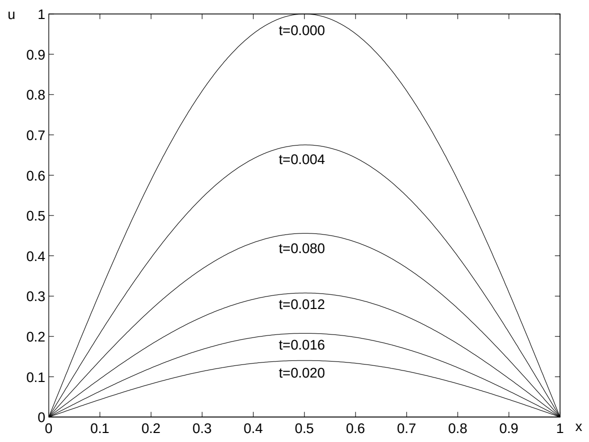

Fig. 3.4. Solution of the Burgers equation for $\nu = 10$

We solve the Burgers equation (3.3.1) subject to the boundary conditions (3.3.2) and the initial condition $f(x) = \sin(\pi x)$ using $m = 10$ for different values of ν:

Case $\nu = 10$ Here, the dissipation term dominates the advection term. In fact, the curve of the solution u drops dramatically from the initial data to zero within the first 0.05 seconds.

Case $\nu = 0.1$ Now both the dissipation term and the advection term have a balanced influence on the solution.

Case $\nu = 0.01$ This is a first case when the advection term dominates the dissipation term. The behavior of the solution u is very different from the previous case as soon as advection dominates. The solution u is a shock wave moving to the right with speed

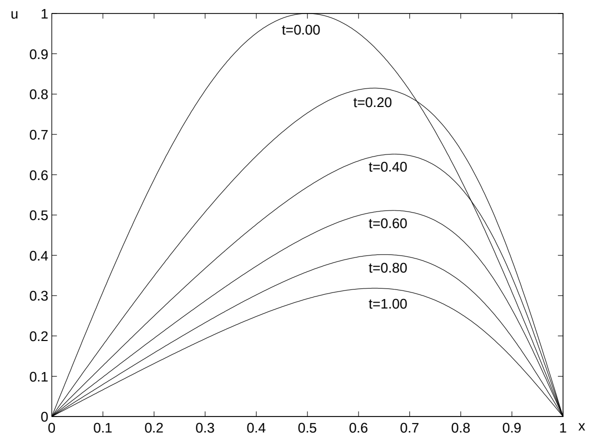

Fig. 3.5. Solution of the Burgers equation for $\nu = 0.1$

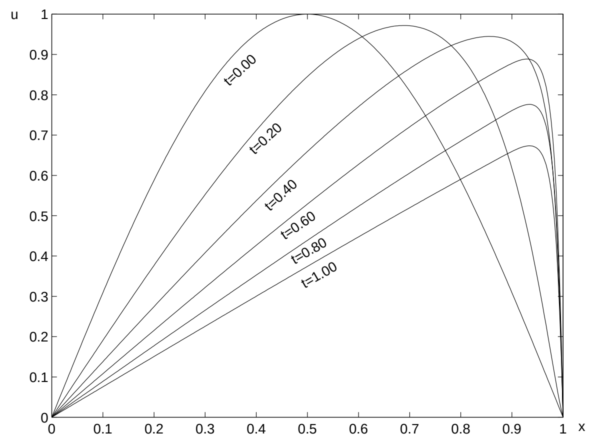

Fig. 3.6. Solution of the Burgers equation for $\nu = 0.01$

proportional to the magnitude of u. The numerical value of the analytical solution is still available for comparison.

Case $\nu = 0.0001$ Here, the advection term strongly dominates the dissipation term. The peak of the shock wave remains high and moves to the right during the first 0.5 seconds. The numerical value of the analytical solution cannot be computed accurately by standard methods.

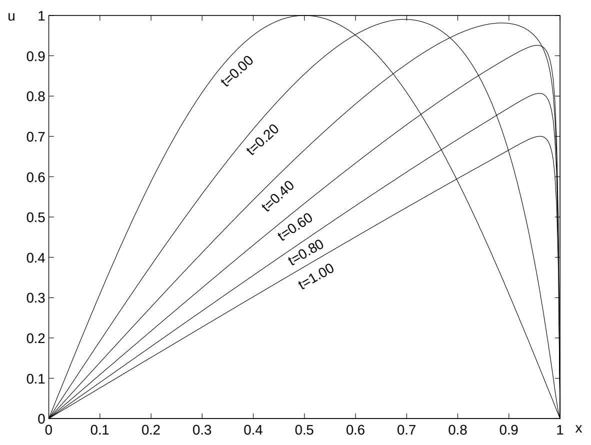

Fig. 3.7. Solution of the Burgers equation for $\nu = 0.0001$

In Figure 3.4 to Figure 3.7, we display the profiles of the numerical solutions u^k using the global formula (3.3.5) for all cases. Similar results can also be found in [OP86] for comparison. For $\nu = 0.0001$, we compare the numerical results with the values obtained using the **Compact Difference Technique** [MG80], the **Finite Element Method** with a special splitting technique [IM92], and with moving nodes [CS82]. The meshless spatial collocation method offers better results than the finite element method with moving nodes and much better results than both the compact difference technique and finite elements with splitting. Results are shown in Table 3.4. With a fixed number of spatial points, the ℓ_∞-norm errors of the numerical scheme are approximately 10^{-2}, 10^{-3}, and 10^{-4} for time steps $\delta_t = 0.001$, 0.0001, and 0.00001, respectively. Altogether, this truly meshless method not only is easy to implement, but also offers an excellent approximation for large values of the Reynolds number. Finally, we refer the reader to [IK04] for a rather sophisticated adaptive meshless treatment of 2D advection using radial basis functions.

3.4 Free Boundary Value Problems

As another illustration of the advantages of meshless collocation, we now treat the **Black–Scholes equation** for both European and American option valuation. In particular, the American option valuation can be treated as a free boundary diffusion problem in which no analytic

		Compact	FEM	FEM	Meshless	
	accurate	difference	splitting	moving	(m = 10 points)	
x	solution	δ_t=0.001	δ_t=0.1*	δ_t=0.001*	δ_t=0.1	δ_t=0.001
0.00	0.000	0.000	0.000	0.000	0.000	0.000
0.05	0.042	0.050	0.041	0.042	0.042	0.042
0.11	0.084	0.075	0.083	0.084	0.084	0.084
0.16	0.126	0.147	0.125	0.126	0.126	0.126
0.22	0.168	0.135	0.169	0.168	0.168	0.168
0.27	0.210	0.261	0.203	0.210	0.210	0.210
0.33	0.252	0.209	0.266	0.252	0.251	0.252
0.38	0.293	0.334	0.252	0.294	0.293	0.293
0.44	0.335	0.304	0.396	0.336	0.334	0.335
0.50	0.376	0.417	0.235	0.377	0.375	0.376
0.55	0.418	0.374	0.548	0.419	0.415	0.418
0.61	0.459	0.505	0.257	0.460	0.455	0.459
0.66	0.500	0.463	0.604	0.500	0.495	0.499
0.72	0.540	0.580	0.601	0.541	0.534	0.540
0.77	0.580	0.536	0.463	0.581	0.572	0.580
0.83	0.620	0.667	0.701	0.621	0.609	0.620
0.88	0.659	0.620	0.671	0.660	0.645	0.660
0.94	0.698	0.741	0.726	0.699	0.678	0.695
1.00	0.000	0.000	0.000	0.000	0.000	0.000

* with 16 intervals

Table 3.4. *Comparison of results for* $\nu = 0.0001$.

solution is available. The difference between the two only lies in the boundary conditions, and this is why we start with the Black–Scholes equation and postpone the boundary conditions.

As outlined in Section 2.10 concerning line methods for time-dependent problems, we discretize the spatial part of the computational domain by superpositions (1.1.1) of translates of a positive definite radial basis function ϕ and then let the coefficients be functions of time as in (2.10.1). Since the radial basis function can be chosen to be arbitrarily continuously differentiable, higher order partial derivatives of the trial functions can directly be computed via derivatives of the radial basis function. This is particularly useful in computing special financial derivative terms like Delta values. To handle the time derivative variable, the Black–Scholes equation is transformed as in (2.10.2) to a system of first order differential equations in time whose solution can be obtained by using an ad-hoc time integration scheme.

To be more explicit, consider the Black–Scholes equation

$$\frac{\partial V}{\partial t} + \frac{1}{2}\sigma^2 S^2 \frac{\partial^2 V}{\partial S^2} + rS\frac{\partial V}{\partial S} - rV = 0,\ 0 < t < T,\ S > 0, \qquad (3.4.1)$$

with the terminal condition

$$V(S,T) = \max(E - S, 0) \qquad (3.4.2)$$

for European put options, where r is the risk-free interest rate, σ is the volatility of the stock price S, and $V(S,t)$ is the option value at time t and stock price S. The transformations

$$S = Ee^x, \quad t = T - \frac{\tau}{\sigma^2/2}, \quad V = Ee^{\alpha x + \beta\tau}u(x,\tau)$$

given in [WDH95] change equations (3.4.1) and (3.4.2) to the simple diffusion equation

$$\frac{\partial u}{\partial \tau} = \frac{\partial^2 u}{\partial x^2},\ -\infty < x < \infty,\ 0 < \tau < \frac{1}{2}\sigma^2 T, \qquad (3.4.3)$$

with initial condition

$$u(x,0) = \max(e^{\frac{1}{2}(\gamma-1)x} - e^{\frac{1}{2}(\gamma+1)x}, 0), \qquad (3.4.4)$$

where $\gamma = r\sigma^2/2,\ \alpha = -(\gamma-1)/2$, and $\beta = \alpha^2 + (\gamma-1)\alpha - \gamma$.

Using meshless collocation in the spatial variable, an approximation of the solution of (3.4.3) subject to the initial condition (3.4.4) is sought as in (2.10.1) by

$$\tilde{u}(x,\tau) = \sum_{k=1}^{m} \alpha_k(\tau)\phi(x - x_k), \qquad (3.4.5)$$

where $(x,\tau) \in [a,b] \times [0, \sigma^2 T/2]$, and $X := \{x_1 \ldots, x_m\}$ is a set of m distinct points in the interval $[a,b]$. Define

$$\begin{array}{rcl} \mathbf{\Phi}_X^T(x) & := & (\phi(x-x_1), \ldots, \phi(x-x_m)) \in I\!R^m \\ \mathbf{a}^T(t) & := & (\alpha_1(t), \ldots, \alpha_m(t)) \in I\!R^m. \end{array}$$

The trial space is then spanned by all functions of the form $\mathbf{\Phi}_X^T(x)\mathbf{a}(\tau)$.

It is well known from the theory of radial basis function approximations that smooth functions on $\Omega \times [0, \sigma^2 T/2]$ can be well approximated by functions $\mathbf{\Phi}^T(x)\mathbf{a}(\tau)$ if the scattered points from $X := \{x_1, \ldots, x_m\}$ have a sufficiently small fill distance (2.3.1) in the spatial domain $\Omega \subset I\!R^d$. This was outlined in Section 2.3 and has a long history dating back to e.g. [MN88, WS93] and summarized in [Buh03, Wen05]. To handle time derivatives, an approximation of the solution in the trial space is constructed which satisfies (3.4.3) on the lines $x = x_j$.

Assume that a function $\tilde{u}$ of the form (3.4.5) satisfies equation (3.4.3) on the lines $x = x_j$. Then the vector function

$$\tilde{\mathbf{u}}(\tau)^T := (\tilde{u}(x_1, \tau), \ldots, \tilde{u}(x_m, \tau)) \in I\!R^m$$

has the form $\tilde{\mathbf{u}}(\tau) = \mathbf{A}_X \mathbf{a}(\tau)$ with the time-independent positive definite kernel matrix

$$\mathbf{A}_X := (\phi(\|x_j - x_k\|_2))_{1 \le j,k \le m}$$

arising already in (1.2.2).

Substituting the representation $\tilde{\mathbf{u}}(\tau) = \mathbf{A}_X \mathbf{a}(\tau)$ into equation (3.4.3), we obtain the following system

$$\mathbf{A}_X \mathbf{a}'(\tau) = -\mathbf{A}_{X,2} \mathbf{a}(\tau)$$

of ordinary differential equations (ODEs) for the unknown vector function $\mathbf{a}(\tau)$, where $\mathbf{A}_{X,2}$ is defined to be the matrix with entries consisting of spatial derivatives $-\phi''(x_j - x_k)$ for $1 \le j, k \le m$.

Since $\mathbf{A}_X$ is positive definite, its inverse $\mathbf{A}_X^{-1}$ exists and we have

$$\mathbf{a}'(\tau) = -\mathbf{A}_X^{-1} \mathbf{A}_{X,2} \mathbf{a}(\tau) \tag{3.4.6}$$

or equivalently

$$\tilde{\mathbf{u}}'(\tau) = -\mathbf{A}_{X,2} \mathbf{A}_X^{-1} \tilde{\mathbf{u}}(\tau). \tag{3.4.7}$$

Since $\phi(x)$ is a positive definite function, the function $-\phi''(x)$ is a positive definite function (this follows from univariate Fourier transform arguments we omit here), and hence $\mathbf{A}_{X,2}$ is also a positive definite matrix.

These linear ODE systems (3.4.6) or (3.4.7) can then be solved by using various methods. With the initial condition $\mathbf{u}(0)$ obtained from the initial condition (3.4.4), the systems (3.4.6) or (3.4.7) are standard linear initial value problems from any textbook in ordinary differential equations, and they have explicitly available exponentially decaying solutions due to the negative definiteness of the system matrix $-\mathbf{A}_{X,2} \mathbf{A}_X^{-1}$. Using the **matrix exponential**

$$e^{t\mathbf{B}} \mathbf{x} := \sum_{n=0}^{\infty} \frac{t^n}{n!} \mathbf{B}^n \mathbf{x}$$

for square $m \times m$ matrices $\mathbf{B}$, scalars t and vectors $\mathbf{x} \in I\!\!R^m$, our approximate solution is

$$\tilde{\mathbf{u}}(\tau) = e^{-\tau \mathbf{A}_{X,2} \mathbf{A}_X^{-1}} \mathbf{u}(0). \tag{3.4.8}$$

So far, the above results hold for positive definite radial basis functions, but they can be extended to conditionally positive definite functions. For instance, if we use scaled multiquadrics $\phi(x) = \sqrt{c^2 + x^2}$, an approximation order $\mathcal{O}(h^\rho)$ for any $\rho > 0$ can be achieved if the time steps are small enough. Note that the time stepping cannot be avoided since the numerical computation of the eigenvalues and eigenvectors from the explicit formula can be difficult.

We can treat the case of American options similarly. Their pricing can be treated as a free boundary value problem, but until recently no analytical formula was available. American options allow early exercise at any time $t \in [0, T]$ with an optimal exercise stock value of $S = B(t)$. The difficulty for most numerical methods to compute an accurate solution for the American options is due to the unknown free boundary $B(t)$. To model an optimal early exercise, the Black–Scholes equation for the American put options valuation now is

$$\begin{array}{rcll} \dfrac{\partial V}{\partial t} + \dfrac{1}{2}\sigma^2 S^2 \dfrac{\partial^2 V}{\partial S^2} + rS\dfrac{\partial V}{\partial S} - rV &=& 0, & S > B(t) \\ V(S,t) &=& V(S,T), & S \leq B(t). \end{array} \tag{3.4.9}$$

The same transformations as given before change equations (3.4.9) and (3.4.2) to

$$\begin{array}{rcll} \dfrac{\partial u}{\partial \tau} &=& \dfrac{\partial^2 u}{\partial x^2}, & x > x_f(\tau), \\ u(x,\tau) &=& u(x,0), & x \leq x_f(\tau), \end{array} \tag{3.4.10}$$

with the initial condition

$$u(x,0) = \max(e^{\frac{1}{2}(\gamma-1)x} - e^{\frac{1}{2}(\gamma+1)x}, 0) \tag{3.4.11}$$

where $x_f(\tau) = log(B(t)/E)$ is the corresponding optimal exercise point. The region $x \leq x_f(\tau)$ corresponds to where the American options should be early exercised to attain the optimal value $u(x,\tau) = u(x,0)$ (i.e. $\partial u/\partial \tau = 0$).

The difficulty in solving equation (3.4.10) under (3.4.11) is due to the unknown optimal exercise point $x_f(\tau)$. It is possible to find a suitable transformation which can translate the unknown $x_f(\tau)$ into the equation [WK97]. However, the difficulty still persists due to the nonlinearity added to the equation.

From a physical point of view, the only difference between the American options and the European options problem is the propagation process induced by the moving unknown boundary value $x_f(\tau)$. This places an additional restriction at any time τ on the solution to let its value be at least $u(x, 0)$.

To solve this free boundary value problem, we apply the meshless collocation method to solve equations (3.4.10) under the initial condition (3.4.11), additionally guaranteeing at each time τ that $u(x, \tau)$ is at least $u(x, 0)$. Then there will be a similar estimate as in the European option case for the error between the exact solution and the approximate solution.

Since the only difference in the valuation of the European and the American options is reflected by the updating procedure at the free boundary, the American options case still is relatively simple. For illustration, we show how to use the first order explicit forward difference scheme with the necessary modification at the free boundary. Handling higher order forward difference or implicit difference schemes is similar. We employ the following algorithm:

- Time discretization: With chosen discrete time values $\{\tau_k\}$ we can discretize equation (3.4.7) by using a first order forward difference scheme to get an intermediate value

  $$\mathbf{u}^\dagger(\tau_{k+1}) := \tilde{\mathbf{u}}(\tau_k) - (\tau_{k+1} - \tau_k)\mathbf{A}_{X,2}\mathbf{A}_X^{-1}\tilde{\mathbf{u}}(\tau_k).$$

- Updating at the free boundary: To satisfy the optimal exercise condition for the American options valuation, we update the approximate solution $\tilde{\mathbf{u}}(\tau_{k+1})$ before the next time step to satisfy the free boundary condition via

  $$\tilde{\mathbf{u}}(\tau_{k+1}) := \max(\mathbf{u}^\dagger(\tau_{k+1}), \mathbf{u}(0)).$$

- The numerical approximation $\tilde{u}(x, \tau)$ to the exact solution $u^*(x, \tau)$ is then given by

  $$\tilde{u}(x, \tau) := \mathbf{\Phi}_X^T(x)\mathbf{A}_X^{-1}\tilde{\mathbf{u}}(\tau),$$

  where we can use values $\tilde{\mathbf{u}}(\tau_k)$, $\tilde{\mathbf{u}}(\tau_{k+1})$ to approximate $\tilde{\mathbf{u}}(\tau)$ for $\tau \in (\tau_k, \tau_{k+1})$.

As the Black–Scholes equation for the American put options is the limiting case of the above algorithm when $\delta_\tau \geq \tau_{k+1} - \tau_k$ tends to zero (see [WDH95] page 112), the proposed algorithm is a reasonable discretization of the Black–Scholes equation for the American put options represented by equations (3.4.10) and (3.4.11).

The spatial error estimation for the radial basis function approximation for the American options problem is the same as for the European options. The remaining question is then the time discretization error of the above algorithm when handling the free boundary condition for the American options. From [WH03] we have the error behavior

$$\mathcal{O}(\delta_\tau \|\mathbf{A}_{X,2}\mathbf{A}_X^{-1}\|_2^2) + \mathcal{O}(h^{\frac{\varrho}{2}})$$

provided that $\delta_\tau \|\mathbf{A}_{X,2}\mathbf{A}_X^{-1}\|_2^2$ and h tend to zero.

Remark: The spectral radius of the matrix $\mathbf{A}_{X,2}\mathbf{A}_X^{-1}$ is less than $\mathcal{O}(h^{-\rho-1})$. Hence, $\delta_\tau \leq o(h^{3\rho/2+1})$ is a reasonable choice for practical computation.

For illustration, we consider a European put option with

$$E = 10,\ r = 0.05,\ \sigma = 0.20 \text{ and } T = 0.5 \text{ (year)}$$

as given in [WDH95]. Assume $S \in [S_{min}, S_{max}]$ and hence

$$x \in [\log(S_{min}/E), \log(S_{max}/E)].$$

In our computations, we chose $S_{min} = 1$ and $S_{max} = 30$ sufficiently large so that $V(S_{max}, \tau)$ is very close to zero. Let

$$\delta_x = \log(S_{max}/E)/(m-1)$$

and

$$x_j = (j-1)\delta_x \text{ for } j = 1, 2, \ldots, m.$$

With these fixed values of x_j, the constant matrices $\mathbf{A}_X$ and $\mathbf{A}_{X,2}$ can be determined, using the conditionally positive definite polyharmonic spline r^3 plus a linear polynomial in x. By Gaussian elimination with partial pivoting, the inverse matrix $\mathbf{A}_X^{-1}$ is obtained. Note that our theoretical description above ignores the additional linear polynomials for simplicity of presentation.

From the initial condition (3.4.4), the initial vector $\mathbf{u}(0)$ of values $u(x_j, 0)$ is given by $u(x_j, 0) = \max(e^{\frac{1}{2}(\gamma-1)x_j} - e^{\frac{1}{2}(\gamma+1)x_j}, 0)$. The numerical approximation $\tilde{u}(x, \tau)$ can then be computed directly by using the analytical formula (3.4.8). If all the eigenvectors of the matrix $-\mathbf{A}_{X,2}\mathbf{A}_X^{-1}$ are linearly independent (this still needs to be proven), the analytical formula (3.4.8) can be computed by

$$\tilde{\mathbf{u}}(\tau) = \sum_{i=1}^{m} c_i e^{\lambda_i \tau} \mathbf{w}_i, \tag{3.4.12}$$

where λ_i and $\mathbf{w}_i$ are the m eigenvalues and linearly independent eigenvectors of the matrix $-\mathbf{A}_{X,2}\mathbf{A}_X^{-1}$. Denote $\mathbf{W}$ to be the $m \times m$ matrix of $(\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_m)$. Since the eigenvectors are assumed to be linearly independent, the matrix $\mathbf{W}$ is invertible. The constant vector $\mathbf{c}$ of c_i can then be determined from $\mathbf{u}(0)$ as

$$\mathbf{c} = \mathbf{W}^{-1}\mathbf{u}(0).$$

In the numerical computation, the eigenvalues λ_i and eigenvectors $\mathbf{w}_i$ in equation (3.4.12) are computed by using the standard matrix computation software MATLAB. For a real and physical solution, we take the real part of the right-hand side of equation (3.4.12) to obtain the analytical approximated solution $\mathbf{u}$. The European put option values

$$V(S,0) \cdot E \cdot \exp\left(\alpha \log(S/E) + \beta\sigma^2 T/2\right) \cdot \tilde{u}(\log(S/E), \sigma^2 T/2)$$

can then be determined. Unlike finite difference methods, which approximate the solution at grid points only, this meshless method gives a global approximation formula. With the obtained values of the coefficients $\mathbf{a}(\tau)$, the option values and all of their derivatives can be obtained directly by using the derivatives of the radial basis function used.

To indicate the numerical accuracy of the meshless spatial collocation method, we also used the analytical time integration scheme to compute the European put option values with m equals 41, 81, and 121, respectively. The comparison with the exact solution is given in Table 3.5. These numerical results indicate that meshless spatial collocation provides a highly accurate approximation to the solution of the European option pricing problem. With $\delta_x = 0.0283$ when $m = 121$, the root-mean-square-error (RMSE) has already been reduced to 0.0002.

Furthermore, the benefit of this truly meshless computational method can be seen from the following example calculating the value of an American put option with

$$E = 100,\ r = 0.1,\ \sigma = 0.30 \text{ and } T = 1 \text{ (year)}.$$

Let $S \in [1, 450]$ so that $x \in [\log(0.01), \log(4.5)]$. Assume

$$\delta_x = [(\log(4.5) - \log(0.01))]/(m-1) \text{ and } \delta_\tau = \sigma^2 T/2n$$

		Analytical Solution		
Stock S	Exact	m=41	m=81	m=121
2.00	7.7531	7.7531	7.7531	7.7531
4.00	5.7531	5.7531	5.7531	5.7531
6.00	3.7532	3.7531	3.7531	3.7532
7.00	2.7568	2.7537	2.7562	2.7566
8.00	1.7987	1.7912	1.7974	1.7985
9.00	0.9880	0.9833	0.9881	0.9879
10.00	0.4420	0.4407	0.4432	0.4417
11.00	0.1606	0.1575	0.1609	0.1603
12.00	0.0483	0.0433	0.0476	0.0481
13.00	0.0124	0.0082	0.0115	0.0123
14.00	0.0028	0.0006	0.0023	0.0027
15.00	0.0006	0.0003	0.0004	0.0005
16.00	0.0001	0.0001	0.0001	0.0001
RMSE		0.0034	0.0006	0.0002

Table 3.5. *Comparison of accuracy for the European put option*

so that

$$\begin{aligned} x_j &= (j-1)\delta_x \text{ for } j = 1, 2, \ldots, m \\ \tau_k &= (k-1)\delta_\tau \text{ for } k = 1, 2, \ldots, n. \end{aligned}$$

In our computation, we take $m = 101$, $n = 100$ and apply our algorithm with the first order explicit time discretization scheme. Table 3.6 gives the result of comparisons of meshless spatial collocation (MSC) with the **Binomial method** (Bin) and the **Front Finite Difference method** (F-F-F) [WK97] for the American put option values V and their Delta values $\partial V(S,0)/\partial S$. In the numerical comparison, the parameter $h = 0.045$ used in the F-F-F method generated a total of 134 grid points whereas the MSC used only $m = 101$ to calculate the values given in the table.

Since there is still no exact solution for the American options valuation, the RMSE values are not given in Table 3.6. However, it can be observed that meshless spatial collocation provides a reasonable approximation to the American put option. A better approximation can be expected by using higher order or implicit time discretization schemes. More computational examples for meshless spatial collocation solving the option pricing problem can be found in [HM99].

Unlike the finite element method, which interpolates the solution by low-order piecewise continuous polynomials, or the finite difference

S	Option values Bin	F-F-F	MSC	Delta values Bin	F-F-F	MSC
80.0	20.2689	20.2662	20.3041	-0.8631	-0.8661	-0.8666
85.0	16.3467	16.3396	16.3646	-0.7109	-0.7133	-0.7137
90.0	13.1228	13.1124	13.1269	-0.5829	-0.5848	-0.5851
95.0	10.4847	10.4733	10.4797	-0.4755	-0.4769	-0.4770
100.0	8.3348	8.3277	8.3278	-0.3856	-0.3866	-0.3865
105.0	6.6071	6.5936	6.5893	-0.3108	-0.3116	-0.3113
110.0	5.2091	5.2004	5.1932	-0.2491	-0.2497	-0.2492
115.0	4.0976	4.0872	4.0784	-0.1986	-0.1990	-0.1984
120.0	3.2059	3.2023	3.1928	-0.1575	-0.1578	-0.1572

Table 3.6. *Comparison of accuracy for the American put option*

method where the derivatives of the solution are approximated by divided differences, the proposed meshless spatial collocation method provides a global interpolation, not only for the solution but also for the derivatives of the solution. This provides a cheap and stable computation of important indicators like Delta values as a bonus without the need to use extra interpolation techniques. The free boundary condition in the valuation of the American options usually leads to difficulties for most existing numerical methods. This, however, does not apply to the meshless spatial collocation method. The valuation of American options is roughly the same as the valuation of European options except for a special updating formula on the free boundary.

But a disadvantage of the method is the full matrix $\mathbf{A}_{X,2}\mathbf{A}_X^{-1}$ which normally hinders its application to large scale problems. Fortunately, recent developments (see Section 2.8) including preconditioning, domain decomposition [WHL$^+$98], and compactly supported radial basis functions [WHG02b] have shown to successfully improve the computational efficiency.

3.5 Moving Boundary Value Problems

Finally, the advantages of meshless spatial collocation will be illustrated by solving a **wave–runup** and a **dam–breaking problem**, which are typical **moving boundary value problems**.

We first consider the following nonlinear, nondispersive **shallow water model** based on hydrostatic pressure and uniform velocity distributions in the depth direction. Let $h(\zeta)$ be the water depth, g the gravitational acceleration, and t and ζ denote the time and spatial depth variable, respectively. Furthermore, let $y(\zeta, t)$ denote the wave amplitude and $u(\zeta, t)$ the depth-average velocity. Then the one-dimensional governing equations in Eulerian form are given by

$$\frac{\partial y}{\partial t} + \frac{\partial (h+y)u}{\partial \zeta} = 0, \ t > 0, \tag{3.5.1}$$

$$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial \zeta} + g\frac{\partial y}{\partial \zeta} = 0. \tag{3.5.2}$$

The Lagrangian–Eulerian hybrid method [BS87] captures the moving waterline by combining the Lagrangian description for the moving waterline with the Eulerian description for the interior flow field. If $W(t)$ denotes the position of the waterline, then

$$\frac{dW}{dt} = u(W(t), t) =: U_0(t).$$

Transformation of the equation (3.5.2) to the Lagrangian form gives

$$\frac{dU_0(t)}{dt} = -g\frac{\partial y}{\partial \zeta}(W(t), t), \tag{3.5.3}$$

where $U_0(t)$ represents the Lagrangian velocity of the waterline. Introducing a geometric transformation

$$\zeta = (1 + W(t)/L)x + W(t)$$

replacing the spatial variable ζ by x, the variable domain is converted to $[-L, 0]$, with $-L$ denoting the fixed end of the spatial computational region, while the time-varying region $-L \leq \zeta \leq W(t)$ is transformed into the fixed spatial region $-L \leq x \leq 0$. If we rewrite the unknown functions U and y as functions of x and t, the equations (3.5.1), (3.5.2) and (3.5.3), respectively, become

$$\frac{\partial y}{\partial t} - c_1(x, W(t))U_0(t)\frac{\partial y}{\partial x} + c_2(W(t))\frac{\partial (h+y)u}{\partial x} = 0, \tag{3.5.4}$$

$$\frac{\partial u}{\partial t} - c_1(x, W(t))U_0(t)\frac{\partial u}{\partial x} + c_2(W(t))\left(u\frac{\partial u}{\partial x} + g\frac{\partial y}{\partial x}\right) = 0, \tag{3.5.5}$$

$$\frac{dU_0(t)}{dt} = -c_2(W(t))g\frac{\partial y}{\partial x}(0, t), \tag{3.5.6}$$

in which

$$c_1(x, W(t)) := \frac{1 + x/L}{1 + W(t)/L},$$

$$c_2(W(t)) := \frac{1}{1 + W(t)/L}.$$

The Lagrangian–Eulerian transformation introduces additional nonlinear terms into the shallow water equations (3.5.4) and (3.5.5) in exchange for a time-independent domain. These nonlinear terms need to be accurately evaluated to provide a good numerical solution.

The surface elevation y and the depth-average velocity u in the shallow-water equations (3.5.4) and (3.5.5) are approximated by using the meshless spatial collocation method like in (2.10.1) as

$$y(x,t) = \sum_{j=1}^{m} \alpha_j(t)\phi_j(x),$$

$$u(x,t) = \sum_{j=1}^{m} \beta_j(t)\phi_j(x),$$

where $\phi_j(x) = \phi(\|x - x_j\|_2)$ are translates of a single radial basis function.

Collocation of the governing equations at the same m testing points of $X := \{x_1, \ldots, x_m\}$ transforms equations (3.5.4) and (3.5.5) into a nonlinear first-order system of ordinary differential equations

$$\mathbf{P} \cdot \dot{\mathbf{B}}(t) + \mathbf{Q}(t, U_0(t), W(t), \mathbf{B}(t)) \cdot \mathbf{B}(t) = \mathbf{F} \qquad (3.5.7)$$

where $\mathbf{F}$ is the zero vector in $I\!R^m$ and $\mathbf{Q}(t, U_0(t), W(t), \mathbf{B}(t))$ is a rather complicated $m \times m$ matrix whose detailed form we suppress here. The system uses standard notation

$$\mathbf{A}_X := (\phi(\|\mathbf{x}_j - \mathbf{x}_k\|_2))_{1 \le j,k \le m}$$

$$\mathbf{a}(t)^T := (\alpha_1(t), \ldots, \alpha_m(t)),\ \dot{\mathbf{a}}(t)^T := \left(\frac{\partial \alpha_1}{\partial t}, \ldots, \frac{\partial \alpha_m}{\partial t}\right)$$

and

$$\mathbf{b}(t)^T := (\beta_1(t), \ldots, \beta_m(t)),\ \dot{\mathbf{b}}(t)^T := \left(\frac{\partial \beta_1}{\partial t}, \ldots, \frac{\partial \beta_m}{\partial t}\right),$$

$$\mathbf{B}(t) \quad = \quad \begin{pmatrix} \mathbf{a}(t) \\ \mathbf{b}(t) \end{pmatrix},\ \dot{\mathbf{B}}(t) = \begin{pmatrix} \dot{\mathbf{a}}(t) \\ \dot{\mathbf{b}}(t) \end{pmatrix},\ \mathbf{P} = \begin{pmatrix} \mathbf{A}_X & 0 \\ 0 & \mathbf{A}_X \end{pmatrix}.$$

The coefficient matrix $\mathbf{A}_X$ that we know well from (1.2.2) is usually full and symmetric, and may become ill-conditioned when a large system of equations is involved. The recently developed domain decomposition method in [ZHL03b] and other techniques summarized in Section 2.8 will enable to solve large-scale problems.

Analogously, the differential equation (3.5.6) turns after spatial discretization into

$$\dot{U}_0(t) = R(W(t), \mathbf{B}(t))$$

after introduction of a suitable function R.

Altogether, the full system of $2m+2$ differential equations consists of

$$\begin{array}{rcl} \dot{\mathbf{B}}(t) & = & \mathbf{P}^{-1}\left(\mathbf{F} - \mathbf{Q}(t, U_0(t), W(t), \mathbf{B}(t)) \cdot \mathbf{B}(t)\right) \\ \dot{U}_0(t) & = & R(W(t), \mathbf{B}(t)) \\ \dot{W}(t) & = & U_0(t). \end{array} \tag{3.5.8}$$

The numerical solution is obtained by a time integration based on given initial conditions. The vector $\mathbf{F}$ becomes non–zero after implementing the boundary conditions at $x = -L$ and $x = 0$. At each time step t_k, the governing equations (3.5.8) are written respectively in the form

$$\begin{array}{rcl} \dot{\mathbf{B}}^k & = & \mathbf{P}^{-1}\left(\mathbf{F}^k - \mathbf{Q}^k \cdot \mathbf{B}^k\right) \\ \dot{U}_0^k & = & R(W^k, \mathbf{B}^k) \\ \dot{W}^k & = & U_0^k \end{array}$$

with $\mathbf{Q}^k = \mathbf{Q}(t_k, U_0^k, W^k, \mathbf{B}^k)$. By employing the Wilson-θ method to $\mathbf{B}^k$ we get the system

$$\mathbf{B}^k = \mathbf{B}^{k-1} + \delta_t \left((1-\theta)\dot{\mathbf{B}}^{k-1} + \theta\dot{\mathbf{B}}^k\right),$$

and similarly for U_0^k and W^k we obtain

$$\left(\mathbf{P}^k + \theta\delta_t\mathbf{Q}^k\right)\dot{\mathbf{B}}^k = \mathbf{F}^k - \mathbf{Q}^k\left(\mathbf{B}^{k-1} + (1-\theta)\delta_t\dot{\mathbf{B}}^{k-1}\right), \tag{3.5.9}$$

$$U_0^k = U_0^{k-1} + \delta_t\left((1-\theta)\dot{U}_0^{k-1} - \theta R(W^k, \mathbf{B}^k)\right),$$

$$W^k = W^{k-1} + \delta_t\left((1-\theta)\dot{W}^{k-1} + \theta U_0^k\right),$$

where δ_t is the time-step and $0 \leq \theta \leq 1$. When θ is equal to 0 or 1, the integration scheme is explicit and may need an extra stability constraint for convergence. In the following computations, the choice of $\theta = 0.5$ makes the time integration scheme implicit and unconditionally stable.

Newton's iterative method provides a fast convergent solution to the nonlinear matrix equation (3.5.9).

To verify the efficiency and accuracy of the meshless spatial collocation method, we implement the formulation derived in the last section to calculate the flow of floodwater resulting from the collapse of a reservoir dam. The analytical solution is known for a semi-infinite reservoir with a constant initial depth h. The bottom of the semi-infinite channel in front of the reservoir is flat and the bottom frictionless. The solution provided in [Sto92] is given as:

$$u(\zeta,t)=\frac{2}{3}\left(\frac{\zeta}{t}+c_0\right),\ -c_0t\le\zeta\le 2c_0t,\ t>0, \tag{3.5.10}$$

$$y(\zeta,t)=\frac{1}{9g}\left(-\frac{\zeta}{t}+2c_0\right)^2-h,\ -c_0t\le\zeta\le 2c_0t,\ t>0, \tag{3.5.11}$$

where u denotes the flow velocity, y is the flood wave amplitude, and $c_0=\sqrt{gh}$. The dam is located at $\zeta=0$ and the solution describes the propagation of the flood wave in both directions following the collapse of the dam at time $t=0$. Equations (3.5.10) and (3.5.11) reveal that the waterline has a zero surface slope and moves at a constant speed of $2c_0$, while the water depth and flow velocity at $\zeta=0$ are always equal to $\frac{4}{9}h$, and $\frac{2}{3}c_0$ respectively.

For comparison purposes, the finite difference method (FDM) is also applied to obtain numerical approximations to the solutions y and u of equations (3.5.4) and (3.5.5) by using the Richtmyer two-step Lax-Wendroff scheme with the following parameters:

- Spatial domain for ζ : [-10m, 0]
- Time domain: [0, 12s]
- Initial water depth behind the dam: h=1m
- Total number of time steps: 600 (in the meshless method) and 1200 (in FDM)
- Total number of spatial discretization points: 101 (in the meshless method) and 201 (in FDM)
- The radial basis function used: r^3.

The boundary conditions $y(0,t)=-h$ and $u(-10,t)=0$ of the transformed domain can be implemented directly in the finite difference scheme. In the meshless collocation scheme, the boundary conditions gives rise to the following numerical approximations:

$$\sum_{j=1}^{m} \alpha_j(t)\phi_j(x_m) = -h,$$

$$\sum_{j=1}^{m} \beta_j(t)\phi_j(x_1) = 0,$$

which are incorporated into equation (3.5.7) to obtain the non–zero vector **F** as a right-hand side of the system.

Figures 3.8 and 3.9 respectively give the time histories of the water surface and flow velocity. The meshless spatial collocation method accurately describes the zero surface slope and the finite propagation velocity at the waterline. These features at the waterline have presented a challenge to most traditional numerical methods. The comparison with the analytical solution and the finite difference approximation is made at time $t = 1.7s$ and indicates excellent agreement. Figure 3.10 shows the propagation of the waterline along the channel in front of the dam. The asterisks indicate the analytical solution before the back-propagating flood wave reaches the back of the reservoir at $x = -10m$, when the numerical solution is influenced by the boundary condition. The computed propagation velocity of the waterline is always equal to the constant $2c_0$, which matches the analytical solution. The computed water depth at the location of the dam is compared with the analytical solution in Figure 3.11. The computed water depth remains constant until the reflection of the back-propagating flood wave from the back of the reservoir arrives. In the numerical comparisons, the maximum relative error is less than 1.5×10^{-3}.

Although the numerical results obtained by the FDM as illustrated in Figure 3.8 to Figure 3.10 are also in good agreement with the analytical solutions, they require more time steps and a higher grid density than the proposed meshless method for the same accuracy. The extension of the FDM to solve similar problems on irregular domains, of course, will be much more difficult than using the meshless method.

The following example further illustrates the application of the meshless collocation method to simulate wave run-up on a plane beach. The

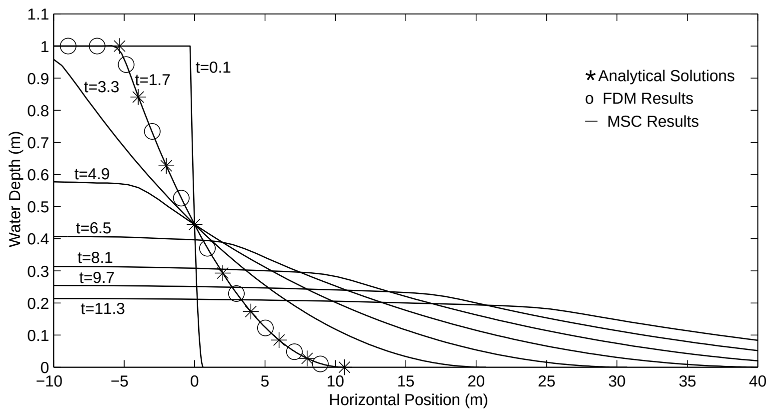

Fig. 3.8. Water-surface profiles in a dam–breaking problem

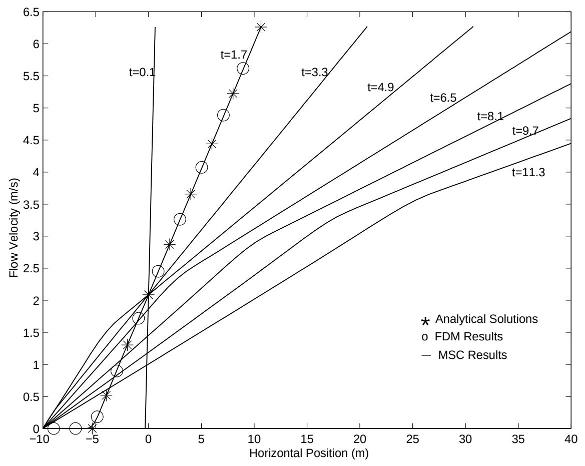

Fig. 3.9. Flow-velocity profiles in a dam–breaking problem

beach profile is defined by

$$h = \begin{cases} \alpha\zeta & : \quad -\frac{h_0}{\alpha} \le \zeta \le 0, \\ h_0 & : \quad \zeta < -\frac{h_0}{\alpha}, \end{cases}$$

where h_0 is the constant water depth in front of the beach, α denotes the slope of the plane beach, and the still waterline is located at $\zeta = 0$. With the still-water initial conditions, the incident waves at the offshore

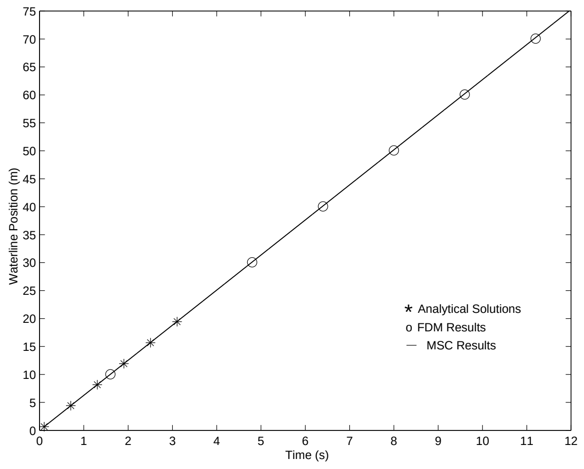

Fig. 3.10. Waterline position in a dam–breaking problem

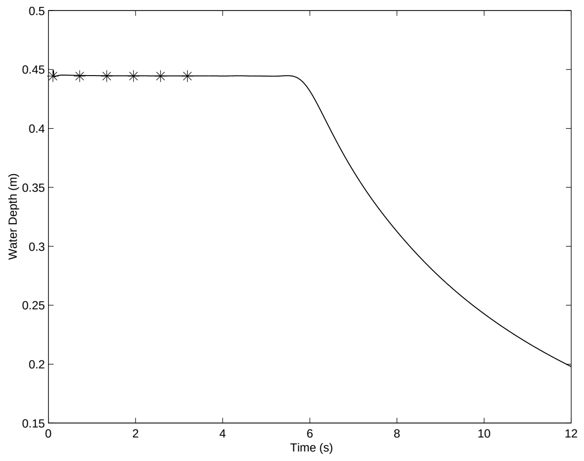

Fig. 3.11. Water depth at dam site, * : analytical solution

end of the profile are generated by

$$\tilde{u}(t) = \gamma \sin(\omega t),$$

where ω denotes the **wave frequency** and γ is the amplitude of the velocity. The boundary conditions of the transformed domain $y(0,t) = \alpha W(t)$ and $u(-L,t) = \tilde{u}(t)$ can be directly incorporated into equation (3.5.7). The parameters used in the computation are:

- $h_0 = 1m,\ \alpha = 0.5,\ \gamma = 0.06,\ \omega = \sqrt{g},\ L = 12m$
- Spatial domain for ζ: [-12m, 0]
- Time domain: [0, 10s]
- Total number of time steps: 500
- Total number of the points used: 101
- Radial basis function used: $(r^2 + 16)^{\frac{1}{2}}$.

The multiquadric radial basis function is used with a scale factor of $c = 4$, which was found to give an accurate solution for this problem.

Figure 3.12 shows the time history of the vertical position of the waterline at the beach. The results show that the initial wave arrives at the beach at $t = 3.8s$ and a steady state solution develops within one wave cycle. The unsteadiness associated with the transition from the still water initial condition decays rapidly with time. Figure 3.13 shows the surface profiles over a wave cycle after the initial wave reaches the beach at $t = 3.8s$. It can be observed from the figures that the meshless collocation method has reasonably simulated the complicated wave motion on the beach. Unlike traditional numerical methods, the meshless collocation method produces a continuous solution over the domain and thus is a good candidate for solving physical problems with high order derivatives.

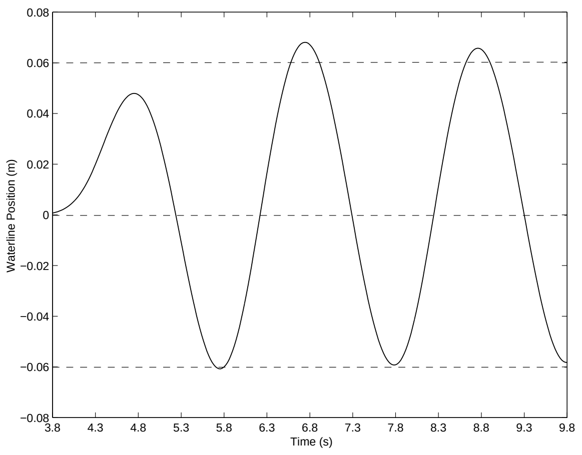

Fig. 3.12. Waterline in a wave run-up problem

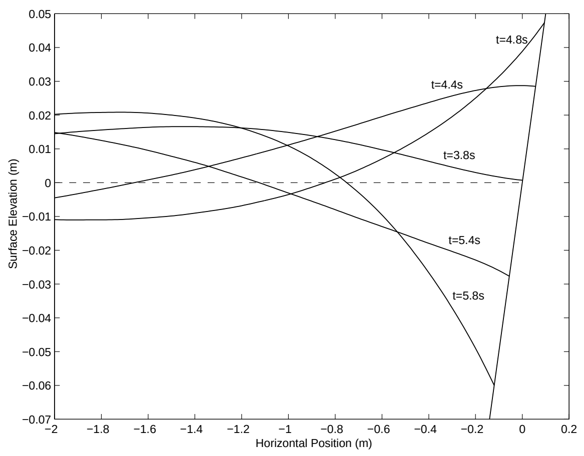

Fig. 3.13. Wave profiles at plane beach

4
The Method of Fundamental Solutions

4.1 Introduction

During the past three decades, the **Finite Difference Method** (FDM) and the **Finite Element Method** (FEM) have become the dominating numerical methods for solving mathematical problems in science and engineering. These methods are **domain–type methods** in contrast to **boundary–type methods** which transfer the problem to the boundary.

Each has its advantages and disadvantages. In general, the FEM and the FDM require tedious domain meshing and are relatively expensive to set up due to the complex bookkeeping process needed for tracking the connectivity of the mesh in the solution domain. In particular, the difficulty of meshing becomes a major hurdle for solving problems in 3D, on irregular domains, and for moving boundaries and singularities.

From a numerical point of view, it is highly desirable to solve a boundary value problem by only discretizing the boundary or by distribution of unstructured discretization points on boundary-like, lower-dimensional geometries. As a result, **boundary element methods** (BEM) have gradually developed as alternative techniques for solving partial differential equations (PDEs) due to their computational efficiency. During the past two decades, they have rapidly improved, and are nowadays considered as mainstream. Despite the fact that the BEM requires only meshing on the boundary, it involves quite sophisticated mathematics and some difficult numerical integrations of singular functions. Moreover, surface meshing in 3D is still a nontrivial task.

Thus, over the past decade, some considerable effort was expanded on eliminating the need for meshing. This led to the development of **meshless methods** which require neither domain nor boundary meshing. They still require discretizations via sets of "nodes", but these nodes need not have any connectivity, and the trial functions are built “entirely in terms of nodes” [BKO$^+$96]. Among these methods, the **Method of Fundamental Solutions** (MFS) has emerged as an effective boundary-only meshless method for solving homogeneous linear partial differential equations. It can be viewed as an indirect BEM and works exclusively with homogeneous solutions as trial functions. Thus only the satisfaction of the boundary conditions is required. Like other boundary methods, the MFS is applicable to any homogeneous elliptic boundary value problem, provided the fundamental solution of the governing equation is known. In fact, the MFS is not new at all. It was originally proposed by Kupradze and Aleksidze [Kup67, KA64a, KA64b, KA65] and has been successfully applied to many problems in science and engineering. The method is also known under different titles such as

- **General Integral Method**,
- **Desingularized Method**,
- **Superposition Method**,
- **Charge Simulation Method**, and
- **Indirect Boundary Element Method**,

and it is a special case of Trefftz’s [Tre26] idea to use superpositions of special solutions of a differential equation in order to satisfy boundary conditions. Detailed reviews of the MFS have been given independently by Fairweather and Karageorghis [FK98], Golberg and Chen [GC98], Fairweather et. al. [FKM03], and Cho et. al. [CGML04].

The MFS has the following key features:

(i) It requires neither domain nor surface discretization.
(ii) It requires no numerical integration, and in particular the difficulties of singular integrands can be avoided.
(iii) For smooth data and domains it was found to be spectrally convergent.
(iv) It is insensitive to the dimensionality of the problem and thus is very attractive for high dimensional problems.
(v) The computational cost is relatively low.
(vi) It can adapt to sharp changes in the geometry of the domain.
(vii) It is easy to implement.
(viii) If a maximum principle holds, its global error is controlled by the error committed on the boundary, and the latter can be easily evaluated. Thus it is safe to apply in practice.

4.2 Theoretical Background

The basic idea of the method of fundamental solutions is to approximate the solution of a homogeneous linear boundary value problem in terms of a superposition of fundamental solutions of the governing differential operator. To be more specific, a **fundamental solution** G for a linear differential operator L is a formulation in the context of distribution theory. In terms of the Dirac delta functional δ, a fundamental solution G is the Green's function for the unbounded space and solves the inhomogeneous equation

$$L\,G(\cdot,\mathbf{y}) = -\delta(\cdot,\mathbf{y}) \text{ for all } \mathbf{y} \in I\!R^d.$$

It is defined on the whole of $I\!R^d \times I\!R^d$ except the **diagonal** $\{(\mathbf{y},\mathbf{y}) : \mathbf{y} \in I\!R^d\}$. The minus sign of the Dirac distribution is introduced for convenience so that the obtained function of two variables is a positive definite kernel if the differential operator L is elliptic. In other words, the function $G(\cdot,\mathbf{y})$ is a solution of the homogeneous partial differential equation

$$Lu = 0 \text{ in } \Omega \tag{4.2.1}$$

for all domains Ω which do not contain the **source point y** of $G(\cdot,\mathbf{y})$.

If $\Omega \subset I\!R^d$, $d = 2,3$, is a bounded open nonempty connected domain with sufficiently regular boundary $\partial\Omega =: \Gamma$ and f^Γ is a known function, then the additional **boundary condition**

$$u = f^\Gamma \text{ on } \Gamma \tag{4.2.2}$$

defines together with (4.2.1) a **homogeneous Dirichlet boundary value problem**. To solve such a problem, the traditional **Boundary Integral Method** is to represent u in terms of the **double layer potential**

$$u(\mathbf{x}) = \int_\Gamma \frac{\partial G(\mathbf{x},\cdot)}{\partial n}(\mathbf{y})\sigma(\mathbf{y})d\gamma(\mathbf{y}), \quad \mathbf{x} \in \Gamma, \tag{4.2.3}$$

where n is the outward pointing normal at $\mathbf{y} \in \Gamma$, σ is the unknown density and $G(\mathbf{x},\cdot)$ is the fundamental solution of L with a source point $\mathbf{x}$ placed on the boundary. This reduces the given homogeneous boundary value problem to a singular **boundary integral equation**

$$f^\Gamma(\mathbf{x}) = \int_\Gamma \frac{\partial G(\mathbf{x},\cdot)}{\partial n}(\mathbf{y})\sigma(\mathbf{y})d\gamma(\mathbf{y}), \quad \mathbf{x} \in \Gamma$$

to be solved for σ on the boundary.

Although the double layer approach to solve the Dirichlet problem has been the most common boundary integral equation method, there has been a substantial amount of work in recent years using the single layer potential representation of u for solving (4.2.1) and (4.2.2) via

$$u(\mathbf{x}) = \int_{\Gamma} G(\mathbf{x}, \mathbf{y})\sigma(\mathbf{y})d\gamma(\mathbf{y}), \quad \mathbf{x} \in \Gamma,$$

for the determination of the source density distribution $\sigma(\mathbf{y})$ on the boundary. This yields an integral equation

$$\int_{\Gamma} G(\mathbf{x}, \mathbf{y})\sigma(\mathbf{y})d\gamma(\mathbf{y}) = f^{\Gamma}(\mathbf{x}) \tag{4.2.4}$$

of first kind on the boundary. One of the advantages of using (4.2.4) over (4.2.3) is that there are fewer integration difficulties since the singularity of the single layer potential is weaker than that of the double layer potential. The disadvantage is that the linear systems arising from discretizations of (4.2.4) tend to be more ill–conditioned than those from (4.2.3).

To alleviate the difficulties of singular integrals, the integration domain can be moved outside Ω to avoid singularities, while still constructing a homogeneous solution. This generates trial functions

$$u(\mathbf{x}) = \int_{\hat{\Gamma}} G(\mathbf{x}, \mathbf{y})\sigma(\mathbf{y})d\gamma(\mathbf{y}), \quad \mathbf{x} \in \Gamma, \tag{4.2.5}$$

where the **fictitious boundary** $\hat{\Gamma}$ is the boundary of a domain $\hat{\Omega}$ containing Ω. Since u in (4.2.5) satisfies (4.2.1) automatically, we need only to satisfy the boundary condition (4.2.2) by a trial function from (4.2.5). This leads to the equation

$$\int_{\hat{\Gamma}} G(\mathbf{x}, \mathbf{y})\sigma(\mathbf{y})d\gamma(\mathbf{y}) = f^{\Gamma}(\mathbf{x}), \quad \mathbf{x} \in \Gamma, \tag{4.2.6}$$

where the source density distribution σ is now to be determined on $\hat{\Gamma}$ instead of Γ. Once σ is determined, the solution u of (4.2.1) and (4.2.2) is constructed.

The integral representation (4.2.6) means that there are infinitely many source points on $\hat{\Gamma}$. In order to apply numerical solution techniques, it is necessary to discretize $\sigma(\mathbf{y})$ and the integral in (4.2.6). First, we reduce the infinite number of points $\mathbf{x} \in \Gamma$ to a finite set of m **test points** or **collocation points** $\{\mathbf{x}_j\}_{j=1}^{m} \in \Gamma$. Second, we combine integration weights and values of σ attained in a set $\{\mathbf{y}_k\}_{k=1}^{n}$ of **source points** or **trial centers** into a set $\{\alpha_k\}_{k=1}^{n}$ of real numbers to get the discretization

$$\sum_{k=1}^{n} \alpha_k G(\mathbf{x}_j, \mathbf{y}_k) = f^\Gamma(\mathbf{x}_j), \quad 1 \le j \le m \tag{4.2.7}$$

of (4.2.6). Even if this linear system is solved inexactly, the trial function

$$u_n(\mathbf{x}) := \sum_{k=1}^{n} \alpha_k G(\mathbf{x}, \mathbf{y}_k) \tag{4.2.8}$$

will exactly satisfy the homogeneous differential equation (4.2.1) and approximately satisfy the boundary condition (4.2.2). If a maximum principle holds, the global error of the trial function u_n with respect to the exact solution u^* will have the explicitly accessible bound

$$\|u - u_n\|_{\infty,\Omega} \le \|f^\Gamma - u_n\|_{\infty,\Gamma}, \tag{4.2.9}$$

thus reducing the solution of a homogeneous boundary value problem to an L_∞ approximation problem on the boundary. The approximate solution of (4.2.7) by trial functions of the form (4.2.8) is called the **Method of Fundamental Solutions**.

Although the above introduction of the MFS proceeded via the boundary integral equation (4.2.6), there is no numerical integration needed at all. It suffices to superimpose a number of fundamental solutions to approximate the boundary values well, but the system (4.2.7) need not be interpreted as a numerical integration formula for a boundary integral. In particular, the MFS usually needs much less source or collocation points than is necessary for any numerical integration of reasonable accuracy.

In order to evaluate numerically the maximum absolute error $\|f^\Gamma - u_n\|_{\infty,\Gamma}$ on the boundary, we often need additional **evaluation points** on the boundary which did not enter in the actual calculation of the approximate solution.

It is important to note that the MFS can also handle other types of boundary conditions, including nonlinear ones. For example, Neumann boundary conditions on the boundary Γ of the solution domain Ω can be matched by setting

$$\frac{\partial u_n(\mathbf{x})}{\partial n} = \sum_{k=1}^{n} \alpha_k \frac{\partial G(\mathbf{x}, \mathbf{y}_k)}{\partial n}$$

equal to certain prescribed values.

We illustrate this in two dimensions using the fundamental solution

$G(\mathbf{x}, \mathbf{y}) = \ln(\|\mathbf{x} - \mathbf{y}\|_2)$ of the Laplace operator $L = \Delta$ which, remarkably, is another radial basis function. Let

$$\mathbf{p} = (p_1, p_2), \mathbf{q} = (q_1, q_2), r = \|\mathbf{p} - \mathbf{q}\|_2 = \sqrt{(p_1 - q_1)^2 + (p_2 - q_2)^2}$$

and let $\mathbf{p}$ be on the boundary Γ with a unit normal $n = (n_1, n_2)$ at $\mathbf{p}$. Then

$$\begin{aligned} \frac{\partial G(\mathbf{p}, \cdot)}{\partial n}(\mathbf{q}) &= \frac{\partial}{\partial n} \ln r = \frac{1}{r} \left(\frac{\partial r}{\partial q_1} n_1 + \frac{\partial r}{\partial q_2} n_2 \right) \\ &= \frac{1}{r^2} \left((p_1 - q_1) n_1 + (p_2 - q_2) n_2 \right) \end{aligned} \tag{4.2.10}$$

See Figure 4.1 for illustration.

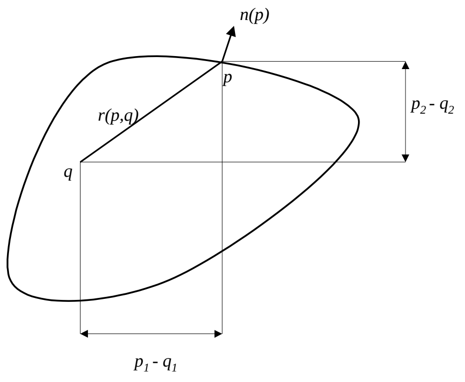

Fig. 4.1. Source point and its boundary nodal point.

In the following, we shall give a more detailed mathematical analysis and a density-based convergence argument. Application-oriented readers may skip to Section 4.4 from here.

Assume that $\{\varphi_\ell(\mathbf{y})\}_{\ell=1}^{\infty}$ is a complete set of functions on $\hat{\Gamma}$. Then the source distribution σ of (4.2.3) can be well approximated by *trial* functions

$$\sigma_L(\mathbf{y}) = \sum_{\ell=1}^{L} c_\ell \varphi_\ell(\mathbf{y}), \quad \mathbf{y} \in \hat{\Gamma}. \tag{4.2.11}$$

This means that we can find a sufficiently large L and coefficients $\hat{c}_\ell$ such that

$$\|\sigma - \sigma_L\|_{\infty, \hat{\Gamma}} = \left\| \sigma(\mathbf{y}) - \sum_{\ell=1}^{L} \hat{c}_\ell \varphi_\ell(\mathbf{y}) \right\|_{\infty, \hat{\Gamma}} \leq \epsilon \tag{4.2.12}$$

holds for an arbitrarily small ϵ. Substituting (4.2.11) into (4.2.6) and collocating at m *test* points $\{\mathbf{x}_j\}_{j=1}^m \in \Gamma$, we should pose a linear equation system

$$\sum_{\ell=1}^{L} c_\ell \int_{\hat{\Gamma}} G(\mathbf{x}_j,\mathbf{y})\varphi_\ell(\mathbf{y})d\gamma(\mathbf{y}) = f^\Gamma(\mathbf{x}_j), \quad 1 \le j \le m. \tag{4.2.13}$$

We further assume that standard quadrature rules can be used which guarantee a bound

$$\left| \int_{\hat{\Gamma}} G(\mathbf{x}_j,\mathbf{y})\varphi_\ell(\mathbf{y})d\mathbf{y} - \sum_{k=1}^{n} w_k G(\mathbf{x}_j,\mathbf{y}_k)\varphi_\ell(\mathbf{y}_k) \right| \le \delta \tag{4.2.14}$$

with suitable weights w_k and integration nodes $\mathbf{y}_k$, $1 \le k \le n$ for arbitrarily small δ, all collocation points $\mathbf{x}_j$ for $1 \le j \le m$, and all functions φ_ℓ, $1 \le \ell \le L$. Then from (4.2.13) and (4.2.14), we derive the linear system

$$\begin{aligned} & \sum_{\ell=1}^{L} c_\ell \left(\sum_{k=1}^{n} w_k G(\mathbf{x}_j,\mathbf{y}_k)\varphi_\ell(\mathbf{y}_k) \right) \\ = \; & \sum_{k=1}^{n} \underbrace{w_k \left(\sum_{\ell=1}^{L} c_\ell \varphi_\ell(\mathbf{y}_k) \right)}_{=:\alpha_k} G(\mathbf{x}_j,\mathbf{y}_k) \\ = \; & f^\Gamma(\mathbf{x}_j), \quad 1 \le j \le m. \end{aligned}$$

This is exactly the standard MFS system (4.2.7) where now the coefficients have the interpretation

$$\alpha_k = w_k \sum_{\ell=1}^{L} c_\ell \varphi_\ell(\mathbf{y}_k), \; 1 \le k \le n.$$

If we define

$$\hat{\alpha}_k := w_k \sum_{\ell=1}^{L} \hat{c}_\ell \varphi_\ell(\mathbf{y}_k), \; 1 \le k \le n$$

with the coefficients of (4.2.12), we find

$$\begin{aligned}
&\left|\sum_{k=1}^{n} \hat{\alpha}_k G(\mathbf{x}_j, \mathbf{y}_k) - f^{\Gamma}(\mathbf{x}_j)\right| \\
= &\left|\sum_{\ell=1}^{L} \hat{c}_\ell \sum_{k=1}^{n} w_k G(\mathbf{x}_j, \mathbf{y}_k)\varphi_\ell(\mathbf{y}_k) - f^{\Gamma}(\mathbf{x}_j)\right| \\
\leq &\left|\sum_{\ell=1}^{L} \hat{c}_\ell \int_{\hat{\Gamma}} G(\mathbf{x}_j, \mathbf{y})\varphi_\ell(\mathbf{y})dy - f^{\Gamma}(\mathbf{x}_j)\right| + \delta\|\hat{c}\|_\infty \\
\leq &\left|\int_{\hat{\Gamma}} G(\mathbf{x}_j, \mathbf{y})\sigma_L(\mathbf{y})dy - \int_{\hat{\Gamma}} G(\mathbf{x}_j, \mathbf{y})\sigma(\mathbf{y})dy\right| + \delta\|\hat{c}\|_\infty \\
\leq &\ \epsilon \int_{\hat{\Gamma}} |G(\mathbf{x}_j, \mathbf{y})|dy + \delta\|\hat{c}\|_\infty
\end{aligned}$$

and see that the MFS system (4.2.7) has a good approximate solution under the above hypotheses. This means that for sufficiently dense test points and source points, and for sufficiently smooth boundary functions, the MFS system can be expected to be solvable by a regularized least-squares solver, and the error in the system can be made arbitrarily small. If, finally, the actual error

$$\left|\sum_{k=1}^{n} \alpha_k G(\mathbf{x}, \mathbf{y}_k) - f^{\Gamma}(\mathbf{x})\right|$$

for the calculated coefficients α_k is evaluated on the boundary, and if a maximum principle holds, users have sharp and arbitrarily small bounds on the total error. This argument applies to most of the examples in this chapter.

But users should note that the above analysis uses an unrealistically large number of source points $\mathbf{y}_k$, because they are chosen to supply a good integration formula on the fictitious boundary. A much smaller number may be sufficient to achieve a small error in the approximation on the boundary, as described by (4.2.9). This is supported by the many examples of this chapter. Unfortunately, the approximation power of traces of fundamental solutions on boundary curves or surfaces is a difficult theoretical problem still under investigation.

4.3 Fundamental Solutions

Table 4.1 lists fundamental solutions of some commonly used linear differential operators which will frequently occur in later sections or chapters. Since all of these operators are radially invariant, their fundamental solutions are radial kernels $G(\mathbf{x}, \mathbf{y}) = \gamma(\|\mathbf{x} - \mathbf{y}\|_2)$, and we only have to list scalar radial basis functions γ of the argument $r := \|\mathbf{x} - \mathbf{y}\|_2$.

L	$\gamma(r)$ in $I\!R^2$	$\gamma(r)$ in $I\!R^3$
Δ	$\frac{-1}{2\pi} \ln r$	$\frac{1}{4\pi r}$
Δ^2	$\frac{-1}{8\pi} r^2 \ln r$	$\frac{r}{8\pi}$
$\Delta - \lambda^2$	$\frac{1}{2\pi} K_0(\lambda r)$	$\frac{e^{-\lambda r}}{4\pi r}$
$\Delta + \lambda^2$	$\frac{i}{4} H_0^{(2)}(\lambda r)$	$\frac{e^{-i\lambda r}}{4\pi r}$
$\left(\Delta - \lambda^2\right)^2$	$\frac{-r}{4\pi\lambda} K_1(\lambda r)$	$-\frac{e^{-\lambda r}}{8\pi\lambda}$
$\Delta\left(\Delta - \lambda^2\right)$	$\frac{-1}{2\pi\lambda^2}\left(K_0(\lambda r) + \ln r\right)$	$-\frac{e^{-\lambda r} - 1}{4\pi\lambda^2 r}$
$\Delta^2 - \lambda^4$	$\frac{1}{8\lambda^2}\left(-iH_0^{(1)}(\lambda r) + \frac{2}{\pi} K_0(\lambda r)\right)$	$\frac{e^{-\lambda r} + e^{-i\lambda r}}{4\pi\lambda^2}$

Table 4.1. *Fundamental solutions for various differential operators.*

In Table 4.1, K_0 and K_1 denote modified **Bessel functions** of second kind, while $H_0^{(2)}$ is the Hankel function of order zero, and $i = \sqrt{-1}$ as usual.

For the fundamental solutions of the product of Helmholtz operators, we refer readers to the Reference [CAO94].

4.4 Static Implementation

It is an important issue to determine the optimal location of the fictitious boundary. In general, there are two different approaches in the literature: **static** and **dynamic**. A third technique is also proposed in this book, combining these two approaches. We shall treat these three approaches in separate sections, starting with the **static approach**.

In the static approach, the source locations $\{\mathbf{y}_k\}_{k=1}^n$ are chosen *a-priori* in some fashion. Bogomolny [Bog85] and Cheng [Che87] have laid the theoretical foundations in this direction. In both papers, the fictitious boundary was chosen to be a circle in 2D and a sphere in 3D with center located at the centroid of the solution domain. In fact, the numerical results for problems with smooth domains and smooth solutions reveal that the shape of the fictitious boundary has only a small effect on the results, as long as it stays well away from the boundary of the solution domain. As a result, it is convenient to generate the source points uniformly distributed on a circle or sphere. Thus, in our two-dimensional numerical examples that we show later, we will mostly choose circles as fictitious boundaries. Theoretically, the optimal choice of the fictitious boundary is a circle or sphere with infinite radius under certain assumptions eliminating cases with singularities. But since the resulting problems become more and more ill–conditioned when the fictitious boundary moves out to infinity, the radius should be chosen as large as to make the systems still solvable on a given computer precision, as suggested by Bogomolny [Bog85]. However, users must be aware that the numerical solution will always extend out to the fictitious boundary, and this rules out problems where singularities of the solution on the boundary can be expected, e.g. for elliptic problems on domains with incoming corners and general boundary data. See Section 4.8.2 for an example.

Cheng's result [Che87] was generalized by Katsurada and Okamoto [Kat90, Kat94, KO88, KO96], who showed that if Γ is a closed Jordan curve in the plane and the data are analytic, then

$$\|u - u_n\|_{\infty,\Gamma} \le c\left(\frac{r}{R}\right)^n$$

where c is a constant, r and R are the diameters of Γ and $\hat{\Gamma}$, respectively.

Once the source points are chosen on $\hat{\Gamma}$, the coefficients $\{\alpha_k\}_{k=1}^n$ in (4.2.8) can be obtained by either least-squares or collocation methods. For least-squares, $m \ge n$ **test** or **collocation points** $\{\mathbf{x}_j\}_{j=1}^m$ are chosen on Γ and then $\{\alpha_k\}_{k=1}^n$ are chosen to minimize [Bog85]

$$\sum_{j=1}^{m}\left(\sum_{k=1}^{n}\alpha_k G(\mathbf{x}_j, \mathbf{y}_k) - f^{\Gamma}(\mathbf{x}_j)\right)^2$$

in a least squares sense. Note that this was the suggestion for solving the non-quadratic asymmetric system (4.2.7) of Section 4.2 approximately.

For collocation, the number n of source points on the fictitious boundary and the number m of test points on the physical boundary are chosen to be equal, so that (4.2.7) becomes a quadratic symmetric linear system of equations

$$\sum_{k=1}^{n} \alpha_k G(\mathbf{x}_j, \mathbf{y}_k) = f^{\Gamma}(\mathbf{x}_j),\ 1 \le j \le n. \tag{4.4.1}$$

In view of the maximum principle and for getting an optimal error bound in the sense of (4.2.9), the optimal way to handle (4.2.7) is to take m very large and to calculate the coefficients that solve this overdetermined system by minimizing not the sum of squares, but the maximum of the absolute values of the residuals. This can be rewritten as a linear optimization algorithm and solved efficiently by the dual revised simplex method.

Due to the density results shown in [Bog85], we generally need to add a constant to (4.4.1) for the Laplace equation. This means that

$$u_n(\mathbf{x}) = \sum_{k=1}^{n} \alpha_k G(\mathbf{x}, \mathbf{y}_k) + \alpha_0, \quad \mathbf{x} \in \Omega.$$

In this case, we have to collocate $n+1$ points $\{\mathbf{x}_j\}_{j=0}^{n}$ on the physical boundary and n source points $\{\mathbf{y}_k\}_{k=1}^{n}$ on $\hat{\Gamma}$. This gives the system

$$\sum_{k=1}^{n} \alpha_k G(\mathbf{x}_j, \mathbf{y}_k) + \alpha_0 = g(\mathbf{x}_j), \quad 0 \le j \le n \tag{4.4.2}$$

along with the constraint

$$\sum_{k=1}^{n} \alpha_k = 0. \tag{4.4.3}$$

Of course, a similar addition can be made when solving (4.2.7) approximately by minimizing minimizing the sum of squares or the maximum of the residuals.

In [Bog85], the author also investigated the influence of the inclusion of a constant function into the set of trial functions. It was found that the difference between (4.4.1) and (4.4.2) is perceptible only when the fictitious boundary is close to the physical boundary. Hence, for convenience, we use (4.4.1) throughout in the next few sections for solving the Laplace equation. For other differential equations, α_0 in (4.4.2) is not required.

For domains with interior holes, other than the source points on the exterior fictitious boundary, additional source points should be placed inside the holes as shown in Figure 4.2. When the interior holes are tiny, it is a challenging how to place the source points inside the interior holes.

This is not a desirable situation and we are expecting the condition number of the resultant MFS matrix gets worse. We will address this issue in one of the following examples.

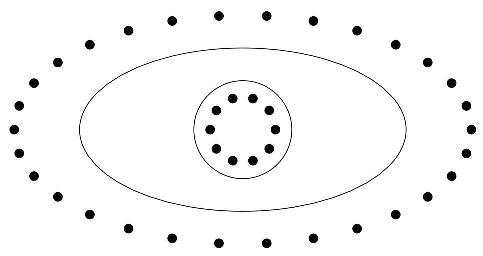

Fig. 4.2. The location of sources points.

The following examples are given to illustrate the effectiveness of the method of fundamental solutions described above. We proceed from rather simple examples to more complicated application-oriented ones.

Example 4.4.4 Here, we present a benchmark case considered by Fen-

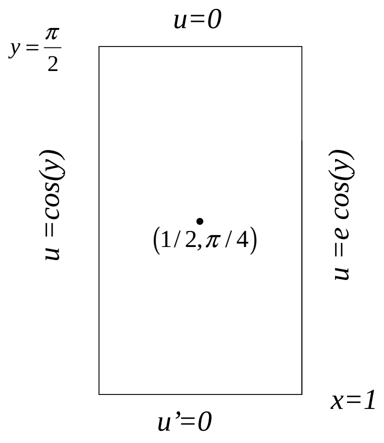

Fig. 4.3. The rectangular domain and boundary conditions.

ner [Fen91]. The governing equation is the homogeneous Laplace operator, and the appropriate fundamental solution is the singular radial basis function $\phi(r) = \ln r$. As shown in Figure 4.3, the boundary conditions prescribe a potential on three sides of the domain, and a potential gradient on the fourth. The exact solution is $u^*(x,y) = e^x \cos y$, causing no problems when source points are far away from the domain. As indicated in [Fen91], the problem is considered as one of moderate complexity and has been used before as an example for testing an Hermitian cubic boundary element formulation.

Let $n = 20$ be the number of collocation points which are uniformly distributed on the physical boundary and let the same number of sources be uniformly distributed on a circle with center at $(0.5, \pi/4)$ and radius R.

R	$\|u - u_n\|_\infty$	$\|\partial u/\partial x - \partial u_n/\partial x\|_\infty$	$\|\partial u/\partial y - \partial u_n/\partial y\|_\infty$
2.5	1.18E−7	1.21E−6	7.40E−6
4.0	3.03E−7	2.46E−6	1.70E−6
8.0	2.04E−7	2.13E−6	1.48E−6
12	2.41E−7	1.93E−6	1.34E−6
15	2.34E−7	1.88E−6	1.31E−6
18	2.28E−7	1.84E−6	1.28E−6

Table 4.2. *Maximum absolute errors of* u, $\frac{\partial u}{\partial x}$, *and* $\frac{\partial u}{\partial y}$.

The absolute maximum errors of u, $\partial u/\partial x$, and $\partial u/\partial y$ evaluated at 400 boundary points are shown in Table 4.2 as functions of the radius R of the fictitious boundary circle. The accuracy is relatively independent of R in this example. This is due to the reason that n is small and the resultant MFS matrix is not very ill-conditioned. Note that we only use 20 collocation points on the boundary to achieve the accuracy of $E-07$ which is remarkable. We can further improve the accuracy to higher level, if necessary, by increasing n. □

Example 4.4.5 We consider a bounded domain Ω with an irregular boundary $\partial\Omega = \Gamma^D \cup \Gamma^N$ and the mixed boundary conditions

$$\begin{aligned} \Delta u &= 0, && (x_1,x_2) \in \Omega, \\ u &= e^{x_1}\cos x_2, && (x_1,x_2) \in \Gamma^D, \\ \frac{\partial u}{\partial n} &= e^{x_1}(\cos x_2 n_{x_1} - \sin x_2 n_{x_2}), && (x_1,x_2) \in \Gamma^N, \end{aligned}$$

where (n_{x_1}, n_{x_2}) is a unit normal vector and

$$\partial\Omega = \left\{(r\cos\theta, r\sin\theta) : r = e^{\sin\theta}\sin^2 2\theta + e^{\cos\theta}\cos^2 2\theta, 0 \le \theta \le 2\pi\right\}.$$

Then Γ^D is the upper half curve of $\partial\Omega$ with $0 \leq \theta < \pi$ and Γ^N is the

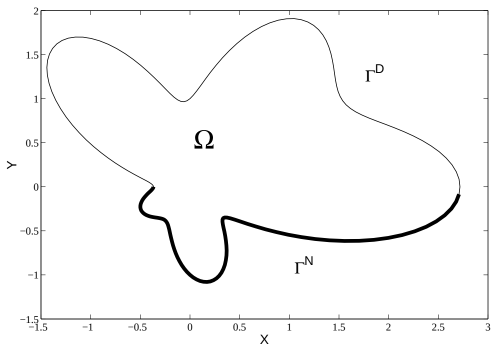

Fig. 4.4. The profile of an amoeba-like boundary.

lower half curve with $\pi \leq \theta < 2\pi$. Figure 4.4 shows the profile of the amoeba-like boundary $\Gamma^N \cup \Gamma^D$.The exact solution is given by the global function $u^*(x_1, x_2) = e^{x_1} \cos x_2$ such that the incoming corners and the change of boundary conditions induce no boundary singularities.

For the numerical implementation, n evenly distributed collocation points were chosen on the boundary $\partial\Omega$. The fictitious boundary is chosen to be a circle with radius R and center at the origin. The profile of source and collocation points are shown in Figure 4.5. To show

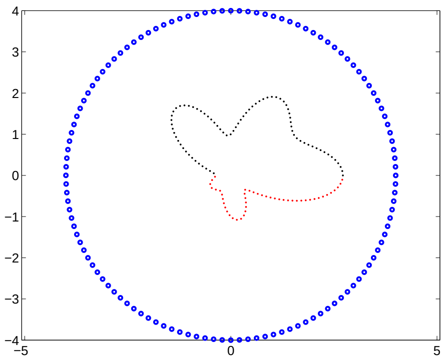

Fig. 4.5. The profile of source and collocation points.

the numerical results, we choose 200 uniformly distributed points inside the domain for testing. The approximate solution and derivatives are denoted by $\hat{u}$, $\hat{u}_{x_1}$, and $\hat{u}_{x_2}$ respectively. Table 4.3 shows the absolute maximum errors of u, u_{x_1}, and u_{x_2} for various n and R. When n is larger, R tends to be smaller to maintain the same level of accuracy.

n	R	$\|u-\hat{u}\|_\infty$	$\|u_{x_1}-\hat{u_{x_1}}\|_\infty$	$\|u_{x_2}-\hat{u}_{x_2}\|_\infty$
120	5.0	1.25$E-$13	5.75$E-$13	4.98$E-$13
140	4.0	4.44$E-$14	9.83$E-$13	7.59$E-$13
160	4.5	9.79$E-$14	6.65$E-$13	5.58$E-$13
180	4.0	9.77$E-$14	6.66$E-$13	5.00$E-$13
200	3.8	3.31$E-$14	4.94$E-$13	3.97$E-$13

Table 4.3. *Absolute maximum errors for various n and R.*

Table 4.3 was obtained by the following MATLAB program. Before executing this program, the users need to prepare four data sets as shown in lines 2–5. In lines 16 and 26, pdist2 is a MATLAB keyword which is used to produce the distance matrix. More details have been given in Section 3.2. The code is quite easy to follow. We will not give further detailed explanation for the rest of the code.

Program 4.4

```
    %The MFS for solving Laplace equation
    %with Dirichlet and Neumann boundary conditions
 1  clear all;
 2  load intnode.txt; % evaluation points
 3  load DB.txt; % Boundary points with Dirichlet condition
 4  load NB.txt; % Boundary points with Neumann condition
 5  load NV.txt % Normal vector (nx,ny) at each point in NB.txt
 6  Dirichlet = @ (x,y) exp(x).*cos(y);
 7  Neumann = @ (x,y,n1,n2) exp(x).*cos(y).*n1 - exp(x).*sin(y).*n2;
 8  bdpt = [DB;NB];
 9  n = length(bdpt); m=length(NB);
10  r = 5; theta = (1:n)/n*2*pi;
11  [s,t] = pol2cart(theta,r); % Sources on circle with radius r
12  source = [s',t'];
13  DM = pdist2(DB, source); % Distance matrix
```

```
A1 = log(DM); % MFS matrix for Dirichlet boundary condition
A2 = zeros(m,n); ux = zeros(1,length(intnode)); uy = ux;
DM1 = pdist2(NB,source);
for i = 1:m %The MFS matrix Neumann boundary condition
    A2(i,:) = (NB(i,1)-s')*NV(i,1) + (NB(i,2)-t')*NV(i,2);
end
A2 = A2./(DM1.∧ 2);
A = [A1;A2]; %The overall MFS matrix
g1 = Dirichlet(DB(:,1),DB(:,2));
g2 = Neumann(NB(:,1),NB(:,2),NV(:,1),NV(:,2));
g = [g1; g2]; % RHS boundary conditions
coef = A\ g; % Solve Ax = b
TM = pdist2(intnode,source);
u = log(TM)*coef; % Approximate solution on the evaluation points
%Approximate solution of du/dx at the evaluation points
for i = 1:length(intnode);
    r2 = (intnode(i,1)-s').∧ 2 + (intnode(i,2)-t').∧ 2;
    ux(i) = ((intnode(i,1)-s')./r2)'*coef;
    uy(i) = ((intnode(i,2)-t')./r2)'*coef;
end
exact = Dirichlet(intnode(:,1), intnode(:,2));
exactux = exp(intnode(:,1)).*cos(intnode(:,2));
exactuy = -exp(intnode(:,1)).*sin(intnode(:,2));
error = max(abs(u-exact)); %absolute maximum error
errorx = max(abs(ux'-exactux)); % absolute maximum d/dx error
errory = max(abs(uy'-exactuy)); % absolute maximum d/dx error
fprintf('Absolute maximum error: %e\n', error)
fprintf('Absolute maximum d/dx error: %e\n', errorx)
fprintf('Absolute maximum d/dy error: %e\n', errory)
```

□

Example 4.4.6 We consider

$$\begin{aligned} \Delta u &= 0, & (x_1, x_2) &\in \Omega, \\ u &= \sin x_1 \cosh x_2, & (x_1, x_2) &\in \partial\Omega, \end{aligned} \qquad (4.4.7)$$

on a multi-connected domain which consists of a unit circle and five embedded circles with radius 0.2 as shown in the shaded area of Figure

4.6. The analytic solution is

$$u^* = \sin x_1 \cosh x_2.$$

Let m denotes the number of source and collocation points on each

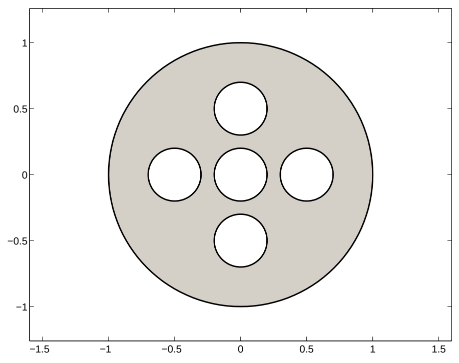

Fig. 4.6. Profile of a circle embedded with five circular holes.

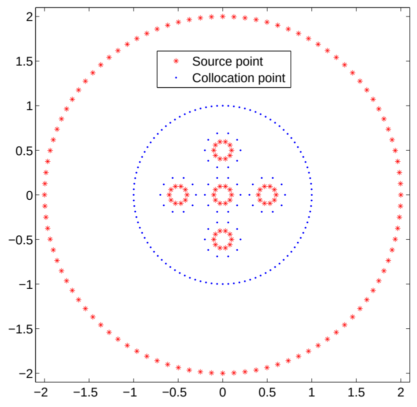

Fig. 4.7. Distribution of collocation and source points.

of the five inner holes, n the number of source and collocation points on exterior fictitious and physical boundary, R the radius of exterior source circle, r the radius of the interior source circle. Note that $R > 1$

and $r < 0.2$. The interior source points need to place inside each hole. We choose the same number of source and collocation points on each corresponding circle. The distribution of source and collocation points is shown in Figure 4.7. We perform the numerical test using various m, n, R, r. The approximate solution is denoted by $\hat{u}$. The results can be seen in Table 4.5. We observe that we can achieve high accuracy with rather small number of m and n. To show the effect of the size of

n	m	R	r	$\parallel u - \hat{u} \parallel_\infty$
40	10	2	0.10	$4.90E{-}12$
40	10	3	0.10	$1.88E{-}15$
60	10	2	0.10	$2.00E{-}15$
80	10	2	0.10	$1.83E{-}15$
100	10	2	0.10	$1.88E{-}15$
100	10	3	0.10	$9.24E{-}15$
100	10	2	0.15	$2.55E{-}15$
100	20	2	0.10	$2.11E{-}15$

Table 4.5. *Absolute maximum error on a multi-connected domain.*

the inner holes, we shrink the radius of each hole to 0.0125 at the same position as above. We choose the radius of each inner source circle to be 0.00625. For $m = 10, n = 60, R = 2$, we can still achieve maximum absolute error $2.44E{-}15$. We continue our test to increase the number of n and m and we still obtain the same level of accuracy. The results seems too good to be true!

Table 4.5 is obtained by the following MATLAB program. We need to provide a subprogram called 'fiveholes' to produce collocation, source, and evaluation points as input data which is shown in line 2.

Program 4.6

```
1  clear all;
   %obtain collocation, source, and evaluation points
2  [coll, source, testpt] = fiveholes;
3  g = sin(coll(:,1)).*cosh(coll(:,2)); % Boundary condition
4  DM = pdist2(coll, source); %Distance matrix
5  coef = log(DM) \ g; % Solve Ax=b
6  TM = pdist2(testpt, source);
7  u = log(TM)*coef; % Approximate solution on the evaluation points
8  exact = sin(testpt(:,1)).*cosh(testpt(:,2)); % Exact solution
```

```
error = max(abs(u-exact)); % Absolute maximum error
fprintf('Absolute maximum error: %e\ n', error)
```

□

In this and several following examples, we consider the **Helmholtz equation**

$$\left(\Delta + k^2\right) u(\mathbf{x}) \quad = \quad 0, \quad \mathbf{x} \in \Omega,$$

which often arises in the study of physical problems involving partial differential equations. This equation generalizes the standard ordinary differential equation

$$u''(x) + k^2 u(x) = 0,\ x \in I\!R$$

for stationary waves with solutions $\cos kx,\ \sin kx$, where k is called the **wave number** related to the **frequency** λ by $k = 2\pi/\lambda$. It arises from the standard technique of separation of variables applied to the **wave equation**

$$\Delta u(\mathbf{x}, t) - \frac{1}{c^2}\frac{\partial^2 u}{\partial t^2}(\mathbf{x}, t) = 0,$$

because general waves can usually be superimposed from stationary waves. This reduces solving the time-dependent wave equation to solving a sequence of spatial Helmholtz equations. Similarly, we shall apply various techniques to reduce time–dependent problems to a series of Helmholtz equations in Chapter 6. Hence, it is important to show the effectiveness of the method of fundamental solutions for solving this type of equation. Note that the equation is not elliptic unless it is rewritten for purely imaginary k as

$$-\Delta u(\mathbf{x}) + \gamma u(\mathbf{x}) = 0,\ \mathbf{x} \in \Omega$$

for nonnegative $\gamma \in I\!R$.

Example 4.4.8 Consider the special Helmholtz equation

$$\begin{aligned} \left(-\Delta + \frac{\pi^2}{2}\right) u &= 0, \quad (x_1, x_2) \in \Omega, \\ u &= \sinh\left(\frac{\pi x_1}{2}\right) \cosh\left(\frac{\pi x_2}{2}\right), \quad (x_1, x_2) \in \Gamma, \end{aligned}$$

where $\Omega \cup \Gamma = \left\{(x_1, x_2) : x_1^2 + x_2^2 \leq 4\right\}$. The exact solution is given by

$$u^* = \sinh\left(\frac{\pi x_1}{2}\right) \cosh\left(\frac{\pi x_2}{2}\right),$$

and the fundamental solution is $\phi(r) = K_0(\pi r/\sqrt{2})$ (see Table 4.1).

The circle around $(0, 0)$ with radius 5 is the fictitious boundary $\hat{\Gamma}$ on which various numbers of uniformly distributed source points were chosen. The same number of collocation points was chosen on the boundary. The numerical error was calculated at 400 uniformly distributed evaluation points on the domain boundary. Absolute maximum errors for various numbers of source points are shown in Table 4.7. Note that the errors decrease rapidly when n becomes larger. When n is sufficiently large, the error started to decrease due to ill–conditioning. □

n	$\|u^* - u_n\|_\infty$
20	$3.3E{-3}$
40	$3.6E{-11}$
60	$1.4E{-14}$
80	$2.1E{-14}$
100	$2.5E{-13}$

Table 4.7. *Errors for n source points.*

Many traditional numerical methods such as finite element, finite difference, and boundary element methods are tedious for irregular three-dimensional domains due to the difficulties of mesh generation on the boundary or inside the domain. The method of fundamental solutions does not have such difficulties and can be easily applied for solving three-dimensional problems on nontrivial domains, as the following case shows.

Example 4.4.9 Consider the 3D Laplace equation with Dirichlet boundary condition

$$\begin{aligned} \Delta u &= 0, && (x_1, x_2, x_3) \in \Omega, \\ u &= x_1^2 + x_2^2 - 2x_3^2, && (x_1, x_2, x_3) \in \Gamma \end{aligned} \tag{4.4.10}$$

on a three–dimensional peanut-shaped domain represented by the parametric surface

$$\mathbf{r}(\phi, \theta) = (f(\theta)\cos\theta, f(\theta)\sin\theta\cos\phi, f(\theta)\sin\theta\sin\phi) \tag{4.4.11}$$

where $\theta \in [0, \pi), \phi \in [0, 2\pi)$ and $f(\theta) := \sqrt{\cos 2\theta + \sqrt{1.1 - \sin^2 2\theta}}$. Figure 4.8 shows the 3D graph of the parametric surface (4.4.11). The analytical solution of (4.4.10) is given by

$$u^* = x_1^2 + x_2^2 - 2x_3^2, \quad (x_1, x_2, x_3) \in \Omega \cup \Gamma,$$

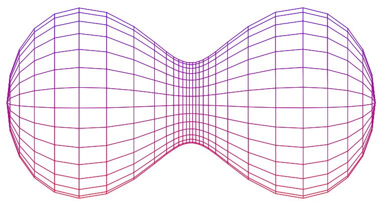

Fig. 4.8. Peanut-shaped domain.

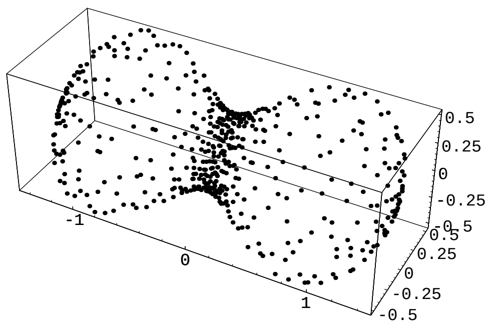

Fig. 4.9. Collocation points on the boundary.

and the fundamental solution of the Laplace operator in 3D is $\phi(r) = 1/r$ from Table 4.1.

A set of n quasi-random collocation points were chosen on the surface of Figure 4.9, and the same number of quasi-random source points were chosen on the surface of a sphere with center at $(0, 0, 0)$ and radius R. We used a quasi-random number generator to ensure quasi-uniform distributions (in terms of the angle parameters of (4.4.11) and the outer sphere). Since no connectivity is required, the generation of these points is straightforward. The distribution of these two sets of points is shown in Figure 4.10. The absolute maximum errors for various n and R are shown in Table 4.8. The accuracy is rapidly improving with increasing n and R.

	$n = 50$	$n = 100$	$n = 150$	$n = 200$
$R = 4$	$1.25E{-3}$	$2.79E{-9}$	$4.07E{-12}$	$1.64E{-13}$
$R = 6$	$5.74E{-4}$	$1.28E{-11}$	$5.83E{-13}$	$3.66E{-13}$
$R = 8$	$7.87E{-6}$	$1.48E{-12}$	$8.91E{-13}$	$3.34E{-13}$
$R = 10$	$1.76E{-6}$	$1.64E{-12}$	$2.74E{-13}$	$6.66E{-13}$

Table 4.8. *Absolute maximum errors for various of n and R*

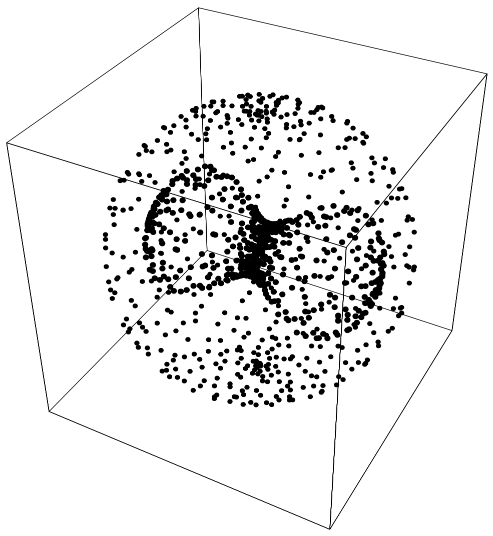

Fig. 4.10. The distribution of collocation points and source points.

$\square$

This example shows that flat analytic radial basis functions, if superimposed properly, and even if they are singular outside the domain, can recover multivariate polynomials well. This is another example of the fact mentioned in Section 2.6, namely that analytic radial basis function interpolants often converge towards polynomials in their “flat limit”.

4.5 Interlude: Stability Issues

It must be expected that the condition numbers of (4.4.1) and (4.4.2) rapidly deteriorates as the radius R of the fictitious boundary circle or sphere increases. Despite this ill–conditioning, a greater distance of the source points $\{y_j\}_{j=1}^{M}$ from the boundary Γ will improve the quality of the approximation of the solution [Bog85] if the boundary is sufficiently smooth and the given boundary data are exact. This is a curious feature of the method which has puzzled many researchers, but it is in line with the observations described in Section 2.4, because the usual linear systems arising in RBF-based methods have the property that the right-hand sides usually are close to the span of the columns of the matrix. Consequently, optimization methods or singular value decompositions like those described in Section 2.5 and 2.9 will help to overcome stability problems, independent of the actual value of the condition number. The latter measures the blow-up of relative errors in the coefficients, but the coefficients are much less relevant than the error committed in reproducing the right-hand side of the system by the constructed approximate solution.

Also in case of the method of fundamental solutions, the coefficients $\{\alpha_j\}_{j=1}^{n}$ in (4.2.8) are usually not of independent interest, but rather the approximation quality of the numerical solution

$$u_n(\mathbf{x}) = \sum_{j=1}^{n} \alpha_j G(\mathbf{x}, \mathbf{y}_j).$$

The crucial point is how data errors are amplified in $u_n(\mathbf{x})$, not necessarily in $\{\alpha_j\}_{j=1}^{n}$.

Despite the notorious ill–conditioning problem, the method of fundamental solutions can still produce extremely accurate results, because the **Uncertainty Principle** for radial basis function approximations described in Section 2.4 can also be observed in the method of fundamental solutions. Subject to machine precision, the worse the condition numbers, the better the approximate solution. As we shall see in the following example, the Method of Fundamental Solutions can cope with large condition numbers, if suitable projections or approximations are used.

Example 4.5.1 To show how the number of collocation points and the location of source points on the fictitious boundary affect the solution of the MFS, we consider the Laplace equation with Dirichlet boundary conditions as in (4.4.7) and boundary conditions

$$u = \sin x_1 \cosh x_2, \quad (x_1, x_2) \in \Gamma,$$

where the domain is

$$\Omega \cup \Gamma = \{(x_1, x_2) : \frac{x_1^2}{4} + x_2^2 \leq 1\}.$$

The numerical evaluation of the error was performed using 276 uniformly distributed evaluation points on the boundary. For various numbers n of source and collocation points, Figure 4.11 shows the absolute maximum errors as functions of the radii of the fictitious boundary circles, while the condition numbers for various radii and source points are shown in Figure 4.12. With an increasing number of sources, the accuracy improves at the expense of the condition number. For $n = 20$, the accuracy seems independent of R, the radius of the fictitious circle. For $n = 40$, the accuracy improves sharply with the increase of R, but deteriorates after $R > 5$ due to the extreme ill–conditioning.

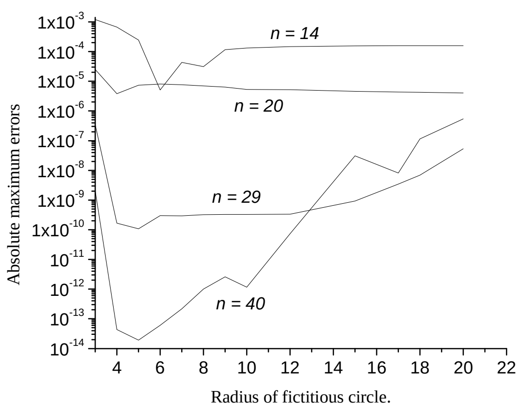

Fig. 4.11. The maximum absolute errors versus radius of fictitious circle.

$\square$

As will be shown in examples below, despite the ill–conditioning, many problems can be solved directly using Gaussian elimination with partial pivoting without problems. However, this requires care since the ill–conditioning may still cause stability problems for some cases [CGML04, Kit88, Kit91, Ram02].

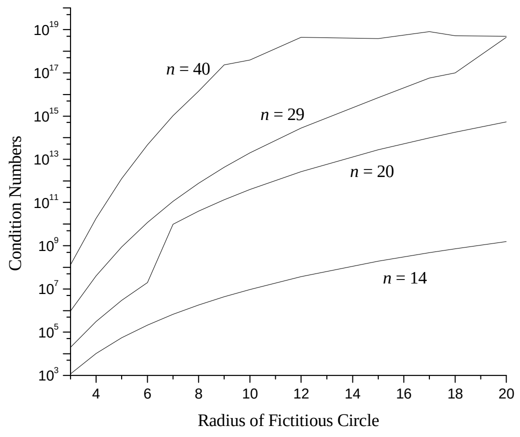

Fig. 4.12. Condition numbers versus the radius of the fictitious boundary circle.

In many practical situations, the ill–conditioning is inherent in the problem and cannot be avoided in scientific computing. Ill–conditioning is one of the problems that need to be further investigated, not only in the solution of the MFS, but for RBF-based methods in general. As we have already observed before, the MFS has the surprising nature that the ill–conditioning has little effect, to a certain extent, on the solution of the MFS. In fact, the so-called effective condition number is a better way to measure the condition number of a matrix. We will briefly introduce the concept of effective condition number and its applicability in the MFS [DL09].

Consider for solving the following system of equations

$$A\mathbf{x} = \mathbf{b} \tag{4.5.2}$$

where A is a $n \times n$ real matrix. The conditioning number of a nonsingular matrix A is defined by $\kappa = \kappa(A) = \|A\| \, \|A\|^{-1}$. If $\|\cdot\|$ is a norm (usually denoted as $\|\cdot\|_2$), then the condition number can be expressed as $\kappa(A) = \sigma_{\min}/\sigma_{\max}$ where $\sigma_{\min}$ and $\sigma_{\max}$ are the smallest and largest singular values of A respectively. When $\kappa(A)$ is small, A is said to be well-conditioned, while $\kappa(A)$ is large A is said to be ill-conditioned. As a general rule of thumb, if the condition number $\kappa(A) = 10^k$, then we may lost up to k digits of accuracy on top of what would be lost to the numerical method due to loss of precision from arithmetic methods.

However, the condition number does not give the exact value of the maximum inaccuracy that may occur in the algorithm. It generally just bounds it with an estimate. The condition number is normally an upper bound and sometimes overestimate the error.

Consider a small perturbation $\Delta\mathbf{b}$ on the right-hand side vector $\mathbf{b}$ in (4.5.2). The condition number will give an estimation of the stability of solution in (4.5.2). As a result, we have $A(\mathbf{x} + \mathbf{\Delta x}) = \mathbf{b} + \mathbf{\Delta b}$. Then,

$$\frac{\|\mathbf{\Delta x}\|}{\|\mathbf{x}\|} \leq \kappa \frac{\|\mathbf{\Delta b}\|}{\|\mathbf{b}\|}. \tag{4.5.3}$$

In the numerical solution using the MFS, we are constantly dealing with extremely high condition number, often $\kappa > 10^{20}$. We have noticed that κ is only depend on the matrix A and does not involve the righ-hand side $\mathbf{b}$. The accuracy of the MFS has an obvious dependence on $\mathbf{b}$. To better estimate the accuracy and stability of the MFS, we introduce the concept of effective condition number $\kappa_{eff} = \kappa_{eff}(A, \mathbf{b})$ MFS [DL09].

By singular value decomposition (SVD), A in (4.5.2) can be decomposed as

$$A = \mathbf{UDV}^T \tag{4.5.4}$$

where $\mathbf{U}$ and $\mathbf{V}$ are $n \times n$ orthogonal matrices and $\mathbf{D}$ is a diagonal matrix with diagonal elements

$$\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_n > 0. \tag{4.5.5}$$

$\sigma_i, 1 \leq i \leq n$, are called the singular values of A.Note that

$$\mathbf{U}^T\mathbf{U} = \mathbf{I}, \quad \mathbf{V}^T\mathbf{V} = \mathbf{I}. \tag{4.5.6}$$

Using SVD, (4.5.2) becomes

$$\mathbf{UDV}^T\mathbf{x} = \mathbf{b}$$

or

$$\mathbf{V}^T\mathbf{x} = \mathbf{D}^{-1}\left(\mathbf{U}^T\mathbf{b}\right). \tag{4.5.7}$$

Let

$$\beta = [\beta_1, \beta_2, \cdots, \beta_n]^T = \mathbf{U}^T\mathbf{b} \tag{4.5.8}$$

and

$$\Delta\beta = [\Delta\beta_1, \Delta\beta_2, \cdots, \Delta\beta_n]^T = \mathbf{U}^T\Delta\mathbf{b}. \tag{4.5.9}$$

Since $\mathbf{D}$ is a diagonal matrix, $\mathbf{D}^{-1}$ is also a diagonal matrix with diagonal elements

$$\sigma_1^{-1} \leq \sigma_2^{-1} \leq \cdots \leq \sigma_n^{-1}.$$

Hence, we have

$$\mathbf{D}^{-1}\left(\mathbf{U}^T\mathbf{b}\right) = \left[\frac{\beta_1}{\sigma_1}, \frac{\beta_2}{\sigma_2}, \cdots, \frac{\beta_n}{\sigma_n}\right]. \tag{4.5.10}$$

Since $\mathbf{V}$ is orthogonal matrix, it preserves the inner product of two real vectors; i.e.,

$$\left\|\mathbf{V}^T\mathbf{x}\right\|^2 = \mathbf{V}^T\mathbf{x}\cdot\mathbf{V}^T\mathbf{x} = \mathbf{x}\cdot\mathbf{x} = \|\mathbf{x}\|^2, \tag{4.5.11}$$

which means $\left\|\mathbf{V}^T\mathbf{x}\right\| = \|\mathbf{x}\|$. Similarly, from (4.5.9), we have

$$\|\Delta\beta\| = \left\|\mathbf{U}^T\Delta\mathbf{b}\right\| = \|\Delta\mathbf{b}\|. \tag{4.5.12}$$

Taking the norm on the both sides of (4.5.7), we have

$$\|\mathbf{x}\| = \sqrt{\sum_{i=1}^{n}\left(\frac{\beta_i}{\sigma_i}\right)^2}. \tag{4.5.13}$$

From (4.5.12) and (4.5.13),

$$\|\Delta\mathbf{x}\| = \sqrt{\sum_{i=1}^{n}\left(\frac{\Delta\beta_i}{\sigma_i}\right)^2} \leq \sqrt{\frac{1}{\sigma_n^2}\sum_{i=1}^{n}(\Delta\beta_i)^2} = \frac{\|\Delta\beta\|}{\sigma_n} = \frac{\|\Delta\mathbf{b}\|}{\sigma_n}. \tag{4.5.14}$$

Form (4.5.3), (4.5.13), and (4.5.14), we obtain

$$\kappa_{eff}(A,\mathbf{b}) = \frac{\|\mathbf{b}\|}{\sigma_n\sqrt{\sum_{i=1}^{n}\left(\frac{\beta_i}{\sigma_i}\right)^2}}. \tag{4.5.15}$$

In [DL09], the discussion of the relation between effective condition number and the noisy boundary data was given. We tried to reproduce a similar example shown in [DL09].

Example 4.5.16 Consider the Laplace equation in the unit disk subject to the Dirichlet boundary condition prescribed by the following harmonic function $g(x,y) = x^2 - y^2$. We choose 80 collocation points on the unit circle and the same number of source points on a fictitious circle of radius 2 with center at $(0,0)$. Furthermore, 161 evaluation points are chosen on the circular boundary of the unit circle.

The MFS is highly accurate numerical method but very sensitive to the noisy boundary data. In Table 4.9, we show that the MFS produces very accurate result when there is a noise-free boundary condition. As the noise level on the boundary condition increase, the effective condition number along with the accuracy decrease. Note that the condition number of the system remain constant $2.32E-10$ through the added noise while the effective condition number and the absolute maximum error (L_∞) are correlated. The resulting effective condition number and L_∞ are quite similar for test cases with 1% and 5% noise. This suggests that the systems are not sensitive to the added noise. In Figure 4.13, we show the plot of L_∞ and $1/k_{eff}$ for the data in Table 4.9. There is a strong correlation between them. As we can see in this case, the condition number is completely useless in predicting the error when the noise data presented.

Noise (%)	k_{eff}	L_∞
0.000	$2.01E9$	$1.67E-14$
0.001	$4.61E4$	$1.41E-4$
0.010	$6.96E3$	$1.28E-3$
0.100	$6.52E3$	$1.22E-3$
1.000	$4.50E2$	$1.13E-2$
5.000	$2.87E2$	$5.87E-2$

Table 4.9. *Effective condition number and the absolute maximum error for various noise levels.*

$\square$

In the following, we present examples showing how to counteract ill–conditioning using the singular value decomposition (SVD) and its regularization by truncation [CCG06]. Readers are referred to Section 2.5 for the background theory.

Due to the maximum principle [PW67], the absolute maximum error for solutions of the Dirichlet problem for the Laplace equation occurs on the boundary. Hence, through the next three examples, we calculated the boundary error using 121 evaluation points that were uniformly distributed on Γ. Then we chose $n < 121$ collocation points and source

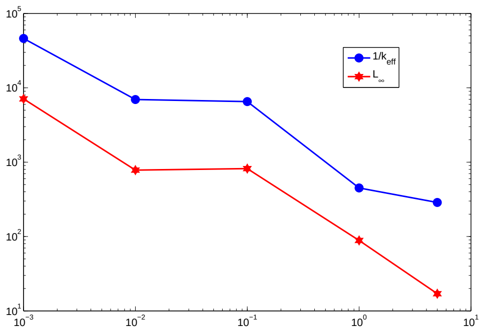

Fig. 4.13. The absolute maximum error and reciprocal of effective condition number for various noise levels.

points which were uniformly distributed on the physical boundary and the fictitious boundary, respectively. Furthermore, we chose a circle with center at $(0, 0)$ and radius R as the fictitious boundary. We denote by e, e_{x_1}, e_{x_2} the absolute maximum errors of u and $\partial u/\partial x_1$ and $\partial u/\partial x_2$, respectively; i.e. ,

$$e = \|u - u_n\|_\infty, \quad e_{x_1} = \left\| \frac{\partial u}{\partial x_1} - \frac{\partial u_n}{\partial x_1} \right\|_\infty, \quad e_{x_2} = \left\| \frac{\partial u}{\partial x_2} - \frac{\partial u_n}{\partial x_2} \right\|_\infty.$$

Example 4.5.17 Consider the Laplace equation with the following Dirichlet boundary conditions

$$\begin{array}{rcll} \Delta u & = & 0, & (x_1, x_2) \in \Omega \\ u & = & \sin x_1 \cosh x_2, & (x_1, x_2) \in \Gamma, \end{array}$$

where

$$\Gamma = \left\{ (r\cos\theta, r\sin\theta) : r = \sqrt{\cos 2\theta + \sqrt{1.1 - \sin^2 2\theta}}, 0 \le \theta \le 2\pi \right\}.$$

Figure 4.14 shows the profile of Γ. The exact solution is given by $u^* = \sin x_1 \cosh x_2$.

First, we chose 50 uniformly distributed collocation and source points on Γ (see Figure 4.14) and the fictitious boundary, respectively. Let R denote the radius of the fictitious circle. In Table 4.10, we show the absolute maximum errors using SVD and Gaussian elimination. In the SVD, we used all 50 singular values. We found that the numerical results using these two approaches showed practically no difference. There is

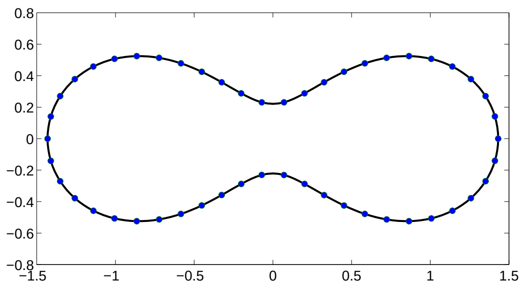

Fig. 4.14. Profile of the Oval of Cassini.

no evidence that the SVD is superior to Gaussian elimination here, as can be expected in all numerically stable cases. □

	SVD		Gaussian	
R	e	e_{x_1}	e	e_{x_1}
4	$9.00E-9$	$1.35E-7$	$2.78E-8$	$4.84E-7$
6	$7.83E-10$	$1.13E-8$	$4.33E-10$	$6.44E-9$
8	$1.70E-10$	$2.88E-9$	$24.25E-10$	$7.18E-9$

Table 4.10. *Comparison of Gaussian elimination and SVD using 50 collocation and source points.*

Example 4.5.18 We consider Laplace equation with mixed boundary conditions. Let $\Gamma = \Gamma^D \cup \Gamma^N$. As shown in Figure 4.15, we imposed a Dirichlet condition on Γ^D, which contains the upper half of Γ, and a Neumann condition on Γ^N, which contains the lower half of Γ. The boundary conditions are imposed in such a way that the exact solution still is $u^* = \sin x_1 \cosh x_2$.

We chose 200 collocation and source points and performed the same tests as shown in Example 4.5.17. Again, as displayed in Table 4.11, the results do not indicate that the SVD is superior to Gaussian elimination.

For testing the **Truncated Singular Value Decomposition** (TSVD) described in Section 2.5, we chose the first 50 singular values and conducted the same numerical tests as above. The numerical results in

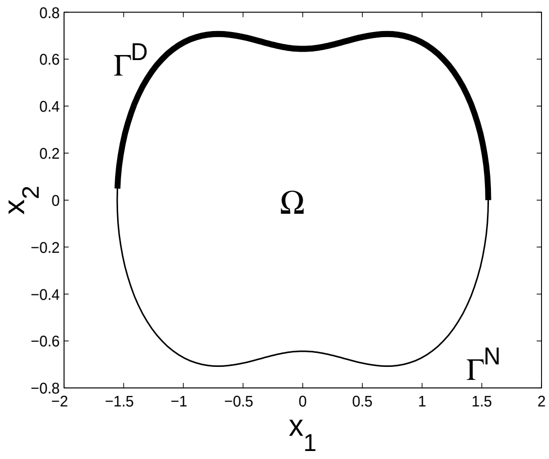

Fig. 4.15. Profile of the Neumann and Dirichlet boundary conditions.

	SVD		Gaussian	
R	e	e_{x_1}	e	e_{x_1}
4	$6.47E-14$	$2.07E-13$	$5.05E-14$	$3.91E-13$
6	$1.34E-12$	$1.80E-12$	$5.00E-13$	$1.95E-12$
8	$5.42E-10$	$4.29E-9$	$5.42E-10$	$4.29E-9$

Table 4.11. *Absolute maximum errors e and e_{x_1} for mixed boundary conditions.*

Table 4.12 show that there are no differences for $R = 4$. However, the numerical results are more stable when R increases. This indicates that the TSVD is superior to the SVD in terms of accuracy and efficiency for using large numbers of collocation and source points. We will further elaborate on this observation in the next two examples.

□

Example 4.5.19 To study the effect of the TSVD on the accuracy, we consider again the Laplace equation with mixed boundary conditions.

TSVD		
R	e	e_{x_1}
4	$3.96E-14$	$1.72E-13$
6	$8.45E-14$	$1.67E-13$
8	$1.35E-12$	$1.67E-12$

Table 4.12. *Absolute maximum errors e and e_{x_1} for mixed boundary conditions.*

Let

$$\Gamma = \{(r\cos\theta, r\sin\theta) : r = e^{\sin\theta}(\sin^2(2\theta)) + e^{\cos\theta}(\cos^2(2\theta))\}.$$

The boundary Γ is an amoeba-like irregular shape, see Figure 4.16. The

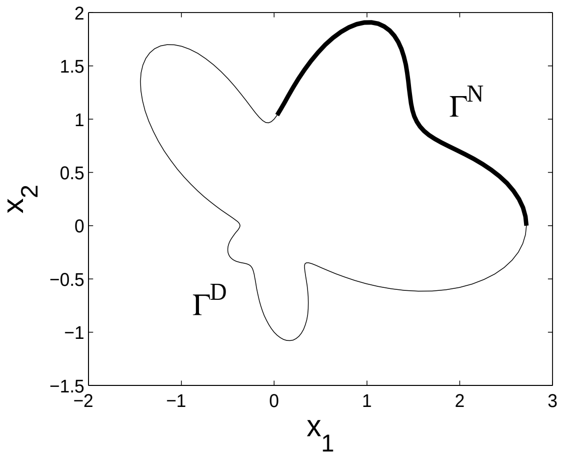

Fig. 4.16. Neumann and Dirichlet boundary conditions.

boundary conditions are given by

$$\begin{aligned}\frac{\partial u}{\partial n} &= (\cos x_1 \cosh x_2)\, n_{x_1} + (\sin x_1 \sinh x_2)\, n_{x_2}, \quad (x_1, x_2) \in \Gamma^N,\\ u &= \sin x_1 \cosh x_2, \quad (x_1, x_2) \in \Gamma^D,\end{aligned}$$

where (n_{x_1}, n_{x_2}) is a unit normal vector, $\Gamma^N = \Gamma$ with $0 \le \theta < \pi/2$, and $\Gamma^D = \Gamma$ with $\pi/2 \leq \theta < 2\pi$. The exact solution is given by $u^* = \sin x_1 \cosh x_2$. As shown in Figure 4.16, the Neumann condition is imposed on the first segment of the amoeba-like curve, and the Dirichlet boundary condition on the rest of Γ.

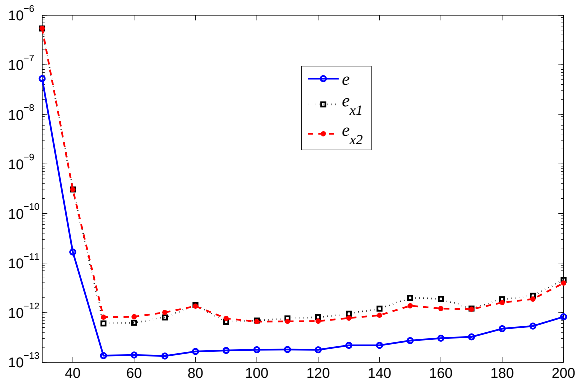

Fig. 4.17. e, e_{x_1}, e_{x_2} using various number of singular values.

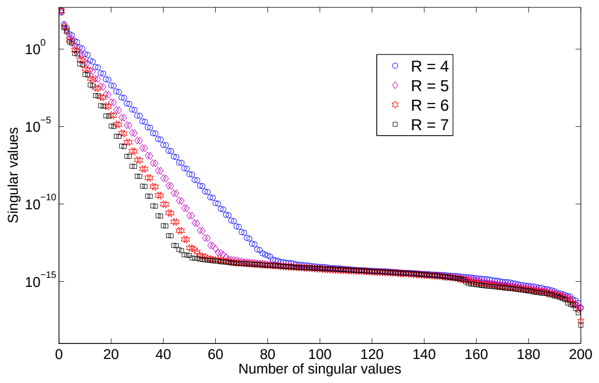

Fig. 4.18. Distribution of singular values.

We first chose 200 collocation on Γ and the same number of source points on the fictitious circle with radius $R = 4$ and center $(0, 0)$. In Figure 4.17, we show the absolute maximum errors of e, e_{x_1}, and e_{x_2} versus the number of singular values being used. In Figure 4.18, we also present the distribution of the singular values for various R. For larger R, the system is more ill-conditioned and more smaller singular values have been observed. We observe that there is no significant difference in accuracy when using more than 50 singular values. This indicates that the truncation error and round off error have little effect on the accuracy, but users are advised not to use more singular values than necessary. Table 4.13 shows again that there is no significant difference between

Source radius R	e (Gaussian)	e (SVD)	e (TSVD)
4	$5.86E-14$	$8.22E-13$	$1.64E-13$
5	$1.03E-12$	$7.04E-13$	$2.68E-13$
6	$1.49E-12$	$7.13E-12$	$7.82E-13$
7	$1.63E-12$	$1.29E-12$	$1/05E-12$

Table 4.13. *Comparison of Gaussian elimination, SVD, and TSVD.*

Gaussian elimination and SVD using 200 collocation and source points using $R = 4$. We have further conducted several tests using various boundary shapes and conditions, and found no significant difference, even for mixed boundary conditions, when using a circle as the fictitious boundary.

The results in Table 4.13 is produced by the following MATLAB program. We need to prepare the input data in lines 2–5 for boundary points and normal vectors at Neumann boundary. Lines 6–7 are the boundary conditions. Line 11 is the MATLAB command for producing the source points on the fictitious circle. Lines 27–34 compute the approximate solutions of $u, \partial u/\partial x, \partial u/\partial y$ at all the evaluation points using Gaussian elimination. Lines 41–54 perform the same task using SVD and TSVD. Note that in line 44, the order of operation of inverse SVD is important. U'∗g has to be computed first instead of using $(V * D^{-1} * U')$∗g.

Program 4.5.19

```
   %The MFS for solving Laplace equation
   %with Dirichlet and Neumann boundary conditions
1  clear all;
2  load -ascii eval.txt; %Evaluation points
3  load -ascii DB.txt; % Boundary points on Dirichlet boundary
```

```
load -ascii NB.txt; %Boundary points on Neumann boundary
load -ascii NV.txt %Normal vector (nx,ny) at each point in NB.txt
Dirichlet = @(x,y) exp(x).*cos(y);
Neumann = @(x,y,nx,ny) exp(x).*cos(y).*nx-exp(x).*sin(y).*ny;
bdpt = [DB;NB];
n = length(bdpt); m = length(NB);
r = 4; theta = (1:n)\n*2*pi;
[s,t] = pol2cart(theta,r); %Sources on circle with radius r
source = [s',t'];
DM = pdist2(DB, source); %Distance matrix
A1 = log(DM); % MFS matrix for Dirichlet boundary condition
A2 = zeros(m,n); ux = zeros(1,length(eval)); uy = ux;
uxsvd = ux; uysvd = ux; uxtsvd = ux; uytsvd = ux;
DM1 = pdist2(NB,source);
for i = 1:m %The MFS matrix Neumann boundary condition
    A2(i,:) = (NB(i,1)-s')*NV(i,1)+(NB(i,2)-t')*NV(i,2);
end
A2 = A2.\(DM1.∧2);
A = [A1;A2]; %The overall MFS matrix
g1 = Dirichlet(DB(:,1),DB(:,2));
g2 = Neumann(NB(:,1),NB(:,2),NV(:,1),NV(:,2));
g = [g1;g2]; % RHS boundary conditions
coef = A\g; %solve Ax=b
TM = pdist2(eval,source); %Distance matrix for evaluation points
u = log(TM)*coef; %Approximate solution on the evaluation points
%Approximate solution of du/dx, d/dy at the evaluation points
for i=1:length(eval);
    r2 = (eval(i,1)-s').∧2 + (eval(i,2)-t').∧2;
    ux(i) = ((eval(i,1)-s')./r2)'*coef;
    uy(i) = ((eval(i,2)-t')./r2)'*coef;
end
exact = Dirichlet(eval(:,1), eval(:,2));
exactux = exp(eval(:,1)).*cos(eval(:,2));
exactuy = -exp(eval(:,1)).*sin(eval(:,2));
err = max(abs(u-exact)); %Absolute maximum error u
errx = max(abs(ux'-exactux));% Absolute maximum du/dx error
erry = max(abs(uy'-exactuy));% Absolute maximum du/dy error
[U,D,V]= svd(A); %Singular value decomposition
dd = 1.\diag(D); % inverse of D
td = dd; td(40:200)=0; %Truncate singular values
```

```
c1 = V*(dd.*(U'*g)); %SVD
c2 = V*(td.*(U'*g)); %TSVD
usvd = log(TM)*c1; %Approximate solution using SVD
utsvd = log(TM)*c2;%Approximate solution using TSVD
%Approximate solution of du/dx, du/dy using SVD & TSVD
for i = 1:length(eval);
    r2 = (eval(i,1)-s').∧2+(eval(i,2)-t').∧2;
    uxsvd(i) = ((eval(i,1)-s')./r2)'*c1; %SVD du/dx
    uysvd(i) = ((eval(i,2)-t')./r2)'*c1; %SVD du/dy
    uxtsvd(i) = ((eval(i,1)-s')./r2)'*c2; %TSVD du/dx
    uytsvd(i) = ((eval(i,2)-t')./r2)'*c2; %TSVD du/dy
end
esvd = max(abs(usvd-exact));%Absolute maximum error u (SVD)
esvdx = max(abs(uxsvd'-exactux));% Absolute maximum du/dx error
esvdy = max(abs(uysvd'-exactuy));% Absolute maximum du/dy error
tsvd = max(abs(utsvd-exact));
tsvdx = max(abs(uxtsvd'-exactux));
tsvdy = max(abs(uytsvd'-exactuy));
%Output the absolute maximum errors
fprintf('Gauss; u= %e,du/dx= %e,du/dy= %e\',err,errx,erry)
fprintf('SVD; u= %e,du/dx= %e,du/dy=%e\n',esvd,esvdx,esvdy)
fprintf('TSVD; u= %e,du/dx= %e,du/dy= %e\n',tsvd,tsvdx,tsvdy)
```

□

Example 4.5.20 We consider Example 4.5.19 adding random noise to the boundary conditions as

$$\begin{aligned}\frac{\partial u}{\partial n} &= (\sin x_1 \cosh x_2)\, n_{x_1} + (\sin x_1 \cosh x_2)\, n_{x_2} + \delta, \quad (x_1, x_2) \in \Gamma^N, \\ u &= \sin x_1 \cosh x_2 + \delta, \quad (x_1, x_2) \in \Gamma^D,\end{aligned}$$

where δ stands for uniformly distributed random numbers drawn from $[-\varepsilon, \varepsilon]$, such that ε denotes the noise level. As in the last example, we chose 200 collocation and source points. We chose 60 singular values for the truncated singular-value decomposition (TSVD). The noise level ε was set to be at 1%. The absolute maximum errors for e, e_{x_1} and e_{x_2} versus the source radius are shown in Figures 4.19 and 4.20. It is clear that the TSVD is superior to the SVD and Gaussian elimination. Again, there is little difference between Gaussian elimination and the SVD. Note how a careful analysis of the singular values can help to determine the noise level and the correct number of signgular values to use. This is in line with the results of Section 2.5 on regularization.

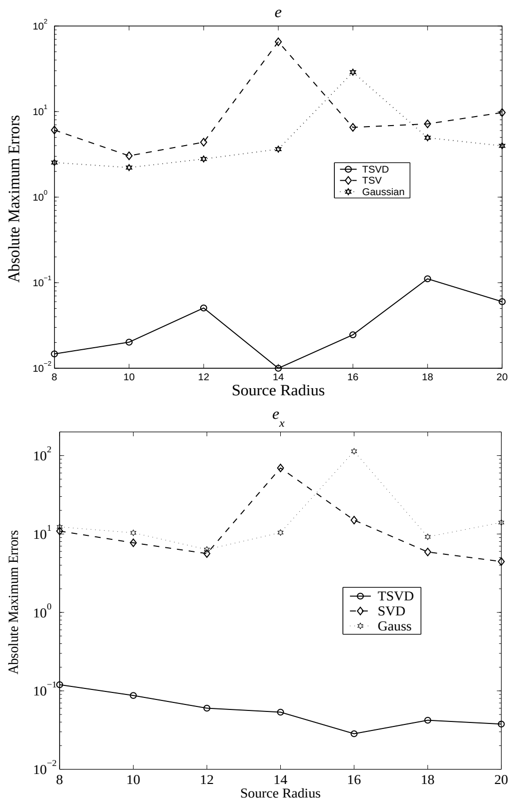

Fig. 4.19. Absolute maximum errors of e (above) and e_{x_1} (below).

□

In [HW05, Jin04] the authors discussed in detail how to choose certain regularization parameters using various techniques. However, we found that the simple truncated singular-value decomposition (TSVD) is sufficient to produce satisfactory results in our cases, in particular if condition numbers are extremely large and if there is noise. We suggest choosing a sufficiently large number of collocation and source points in the formulation and then to comfortably cut-off at least half of the singular values without loss of accuracy. Using SVD, we know how to eliminate the smaller singular values, and using TSVD, the radius of the

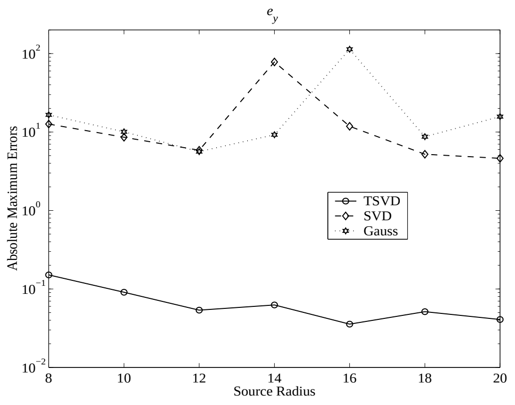

Fig. 4.20. Absolute maximum errors of e_{x_2}.

fictitious circus is no longer a critical issue. This follows since the larger the radius of the source circle, the worse the ill–conditioning becomes. However, since the ill–conditioning is caused by the small singular values, it is rather easy to remove them in order to have a stable computation. The price we have to pay for a simple and maybe too large choice of the source radius is to choose many more source points than needed. In the last and current example, we found that only 60 out of 200 singular values are enough to produce accurate results. This is in perfect agreement with the results of Section 2.5 and the simple example yielding Figure 2.7.

Examples 4.5.17 – 4.5.20 show that the SVD without truncation or regularization does not appear to be more reliable than Gaussian elimination for solving these equations for non–noisy boundary conditions. Since Gaussian elimination is cheaper than SVD, it appears that it can be used to efficiently and accurately implement the MFS. But this requires that the numerical routine used for Gaussian elimination be well-programmed and its pivoting process be cut off properly. Users are discouraged from using self-made solvers. For noisy data and highly ill–conditioned situations, the truncated singular-value decomposition (TSVD) is clearly superior to Gaussian elimination. However, we shall see in Section 4.9 that inverse problems call for even more careful regularization, and then we shall use the Tikhonov strategy, as described in Section 2.5. Note that TSVD is considered as a regularization method, not as a plain solution technique.

4.6 Dynamic Implementation

This approach was first proposed by Mathon and Johnson [MJ77] and further developed by Fairweather and Karageorghis [FK98] and extended to a wide variety of problems, see [Kar92, KF87, KF89a, KO88, PKG98], and references cited therein. In this approach, the locations of n source points $\{\mathbf{y}_k\}_{k=1}^n$ and the corresponding source strength factors $\{\alpha_k\}_{k=1}^n$ in (4.2.8) are both treated as unknown and are to be found as part of the solution. A set of m collocation points $\{\mathbf{x}_j\}_{j=1}^m$ is selected on the boundary where e.g. a Dirichlet boundary condition is provided by a known function g. Then the objective function

$$\varepsilon_m(\mathbf{a}, \mathbf{Y}) = \sum_{j=1}^{m} \left(u_n(\mathbf{x}_j; \mathbf{a}, \mathbf{Y}) - g(\mathbf{x}_j)\right)^2 \tag{4.6.1}$$

is minimized in a non–linear least-squares sense with respect to $\mathbf{a} := \{\alpha_j\}_{j=1}^n$ and $\mathbf{Y} := \{\mathbf{y}_j\}_{j=1}^n$. This is a nonlinear optimization problem which has high computational cost. There are $3n$ unknown to be determined in 2D problems and $4n$ for 3D problems. Since significantly more source points are required for the 3D problems, it becomes inefficient in implementation unless one can get away with very small n, i.e. with a very well-chosen trial space.

Consider the following **biharmonic problem**

$$\Delta^2 u = 0, \quad \mathbf{x} \in \Omega, \tag{4.6.2}$$

subject to either

$$u = g_1(\mathbf{x}), \quad \frac{\partial u}{\partial n} = h_1(\mathbf{x}), \quad \mathbf{x} \in \Gamma,$$

or

$$u = g_2(\mathbf{x}), \quad \Delta u = h_2(\mathbf{x}), \quad \mathbf{x} \in \Gamma. \tag{4.6.3}$$

Karageorghis and Fairweather [KF87] proposed the solution to be approximated by a linear combination of fundamental solutions of both the Laplace and biharmonic equations; i.e. ,

$$u_n(\mathbf{x}) = \sum_{k=1}^{n} \alpha_k G_1(\mathbf{x}, \mathbf{y}_k) + \sum_{k=1}^{n} \beta_k G_2(\mathbf{x}, \mathbf{y}_k)$$

where G_1 and G_2 are the fundamental solutions of Δ and Δ^2, respectively, and where we dropped the dependence of u_n on the $\mathbf{y}_k$ and the coefficients α_k, β_k. Similar to the formulation of (4.6.1), the approximate solution u_n must satisfy the biharmonic equation (4.6.2), and

$\{\alpha_j\}_{j=1}^n$,$\{\beta_j\}_{j=1}^n$ and the positions of the source points $\{\mathbf{y}_j\}_{j=1}^n$ must be chosen in such a way that the boundary conditions (4.6.3) are satisfied. To achieve this goal, the following nonlinear functions are minimized

$$\varepsilon_m = \sum_{j=1}^{m} \left((u_n - g_1(\mathbf{x}_j))^2 + \left(\frac{\partial u_n}{\partial n} - h_1(\mathbf{x}_j) \right)^2 \right) \tag{4.6.4}$$

or

$$\varepsilon_m = \sum_{j=1}^{m} \left((u_n - g_2(\mathbf{x}_j))^2 + (\Delta u_n - h_2(\mathbf{x}_j))^2 \right), \tag{4.6.5}$$

depending on the type of the boundary conditions, and where the source points and coefficients hidden in u_n are varying.

The minimization of the functionals in (4.6.1), (4.6.4) or (4.6.5) can be performed by using various least squares minimization algorithms. Among them, MINPACK, IMSL [IMS01] and NAG [NAG01] are commonly used. The routines LMDIF and LMDER from MINPACK minimize the sum of squares of n nonlinear functionals in m variables using a modified version of the **Levenberg–Marquardt algorithm**. Users should be aware that it implicitly uses a regularization strategy which is similar to Tikhonov regularization. LMDIF evaluates the Jacobian internally by a forward-difference approximation, whereas LMDER requires the user provide a subroutine that evaluates the Jacobian analytically. It terminates when either a user-specified tolerance is achieved or a user-specified maximum number of function evaluations is reached. The subroutine E04UPF from NAG can also be used. This subroutine employs a sequential quadratic programming algorithm and minimizes a functional consisting of a sum of squares. This subroutine can be used either for unconstrained or constrained optimization. The subroutine E04UPF terminates when a user-specified tolerance is found. The tolerance can be supplied through an optional input parameter which specifies the accuracy to which the user wishes the final iterate to approximate the solution of the problem. The subroutines UNLSF and UNSLSJ from IMSL can also be used. UNLSF is based on a modified Levenberg–Marquardt and a finite-difference Jacobian while UNSLSJ uses the same algorithm with a user-supplied Jacobian. Note that if the exact Jacobian is supplied in above mentioned subroutines, the efficiency, as we shall see in the numerical implementation later, will be significantly improved. However, the complexity of the code will increase, too.

The initial placement of the moving sources and the positioning of the fixed boundary points are extremely important as they greatly affect the convergence of the least squares procedure. In general, the sources are initially distributed uniformly at a fixed distance from the boundary [KF87], and the boundary points are selected uniformly on the boundary. Following the recommendation of Oliveira [Oli68], the number m of test or collocation points on the boundary is chosen to be approximately three times the number of unknowns determined by the number n of source points and the space dimension. The tendency of the sources to move to the interior of the solution domain is overcome by an internal check of the position of singularities during the iterative process. If a source is moved inside Ω, it is repositioned at the exterior of the domain [KF87].

Since the minimization problem of (4.6.4) or (4.6.5) is non–linear, there is no guarantee that it has a unique global solution, i.e. the solution obtained may be just an approximation of a local minimum instead of the global minimum. Hence, one has to observe convergence using various parameters and use the algorithms with care.

In the implementation of the MFS, it is often difficult to decide a priori how many sources and boundary collocation points should be used in order to fit the boundary condition satisfactorily. MacDonell [Mac85] introduced the idea of starting with a certain number of sources, say n_1, and a corresponding number of boundary points m_1, and after a certain number of function evaluations, adding sources so that the total number becomes n_2 and the corresponding number of boundary collocation points is increased to m_2, a process which can be repeated until the desired accuracy is reached. The additional sources are distributed uniformly around the region at a user-specified distance along the normal of the surface. This improved version of the MFS was used in the solution of biharmonic problems [KF87] and produced faster convergence of the method. However, it reports no improvement for some nonlinear boundary problems [KF89a].

Example 4.6.6 Consider the harmonic test problem [PKG98]

$$\begin{aligned} \Delta u &= 0, \quad (x_1, x_2) \in \Omega, \\ u &= x_1, \quad (x_1, x_2) \in \Gamma, \end{aligned}$$

where $\Omega \cup \Gamma = \{(x_1, x_2) : -1 \leq x_1, x_2 \leq 1\}$.

The sources were initially positioned at uniformly distributed points, at a fixed distance $d = 0.1$ from the boundary. In this example, the CPU time used was measured as a function of the user-specified tolerance ϵ for $\|u - u_m\| \leq \epsilon$ on the boundary. Let n denote the number of source points and m the number of collocation points on Γ. All computations were performed in double precision on an IBM RISC 6000 computer. The CPU times in seconds for various ϵ, m and n, and for various least squares software packages are shown in Table 4.15. There, E04UPF* denotes that the Jacobian was evaluated internally, while E04UPE** denotes that the Jacobian was provided by the user. From the table we notice that the CPU times required by using LMDER and E04UPF** are much less than those of LMDIF and E04UPF**.

ϵ	m	n	LMDIF	LMDER	E04UPF*	E04UPF**
10^{-2}	6	56	1.1	0.6	1.4	0.2
	7	64	2.3	1.2	1.3	0.2
	8	72	3.7	2.0	93.1	0.4
	9	84	3.4	1.9	141.2	1.3
	10	92	3.5	3.3	151.8	1.1
10^{-3}	6	56	1.1	0.7	2.6	1.6
	7	64	20.1	1.8	4.3	0.7
	8	72	18.6	10.2	129.6	3.9
	9	84	22.4	12.6	243.4	6.1
	10	92	22.3	15.6	260.0	8.3
10^{-4}	6	56	3.3	0.8	3.7	1.9
	7	64	56.8	56.0	60.3	10.5
	8	72	159.7	93.4	339.9	46.7
	9	84	398.6	146.3	506.4	95.4
	10	92	463.3	178.3	617.7	110.4

Table 4.15. *CPU times in seconds required by the least squares minimization routines.*

□

Example 4.6.7 Consider the **biharmonic equation** (4.6.2) subject to the boundary conditions $u(x_1, x_2) = x_1^2$ and $\Delta u(x_1, x_2) = 2$ where $\Omega \cup \Gamma$ is the same as in the previous example. This problem is considered by [PKG98]. The exact solution is $u^*(x_1, x_2) = x_1^2$.

The performance of the various routines shown in Table 4.16 is similar to those for the harmonic problem in the previous example, and the notations are the same. The relative efficiency of the routines is the same as above, but the CPU times required in this example are much higher than in the previous example.

ϵ	m	n	LMDIF	LMDER	E04UPF*	E04UPF**
10^{-2}	6	72	3.2	1.7	51.4	1.5
	7	84	7.5	4.0	69.2	1.5
	8	96	8.2	4.1	146.7	3.3
	9	108	9.8	5.1	343.2	4.6
	10	120	12.1	6.2	467.9	4.5
10^{-3}	6	72	27.9	15.4	89.4	9.6
	7	84	53.3	30.2	159.7	21.7
	8	96	78.7	39.2	462.7	26.4
	9	108	125.7	64.4	767.3	53.9
	10	120	120.9	60.0	957.4	62.7
10^{-4}	6	72	411.8	352.8	461.4	255.4
	7	84	860.4	274.4	735.6	310.8
	8	96	829.2	427.2	920.7	422.3
	9	108	1153.1	637.2	1541.2	546.4
	10	120	2235.3	544.5	2462.4	601.9

Table 4.16. *CPU times in seconds required by the least squares minimization routines.*

□

Example 4.6.8 In this example we consider

$$\begin{aligned} \Delta u &= 0, & (x_1, x_2) \in \Omega, \\ u &= x_1^2 - x_2^2, & (x_1, x_2) \in \Gamma, \end{aligned}$$

where Ω is an ellipse as shown in Example 4.5.1. We used the subroutine DUNLSF from the IMSL library. A total of 72 collocation points were uniformly distributed on the boundary and 8 source points were initially placed on a uniformly distributed fictitious boundary ellipse $\{(3\cos\theta, 2\sin\theta) : 0 \leq \theta < 2\pi\}$. The initial values of $\{\alpha_j\}_{j=1}^n$ were all set equal to 1. Both the absolute and relative function tolerance are set equal to 10^{-5}. The numerical computation was performed using 276 evenly distributed points on the boundary. We obtained an absolute maximum error of $\|u - u_m\|_\infty = 2.75E{-}4$. The final locations of the source points are shown in Figure 4.21. We have tested different initial values, and found the final locations of the sources to vary greatly. This is because there are many ways to approximate a quadratic function well by a few distant fundamental solutions. □

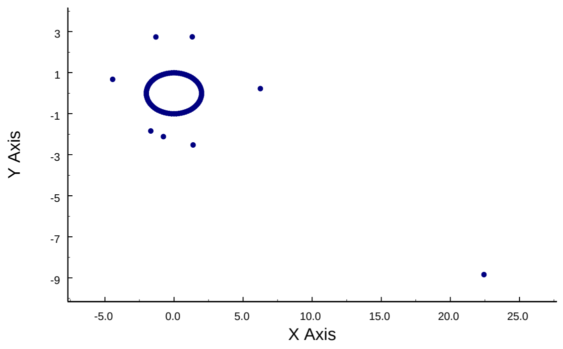

Fig. 4.21. Final locations of source points.

4.7 Modified Implementation

Each method of the previous two approaches has their advantages and disadvantages. A **modified MFS** combining both approaches is presented. In the dynamic approach, each source contains two unknowns in 2D (three in 3D) to be determined. As a result, using the nonlinear routines becomes costly. To reduce the number of unknowns, it is suggested now that the sources are initially fixed on a fictitious circle in 2D (sphere in 3D) with radius R. Then the dynamic approach is applied to determine the radius of the circle. In this way, there are only $m+1$ unknowns for either the 2D or 3D cases. This modified approach is particularly attractive for the 3D case.

Example 4.7.1 We use the modified approach to compare with the static approach in Example 4.5.1. The sources initially are located on a circle of radius 10, and the initial values of $\{\alpha_j\}_{j=1}^{n}$ are all set equal to 1. We used the IMSL routine DUNLSF (double precision) for the nonlinear least squares computation. In the implementation, we supplied the absolute function tolerance and the relative function tolerance as 10^{-8}. Let R denote the computed radius for the fictitious circle. Table 4.17 shows the absolute maximum errors using various sources on the fictitious boundary and collocation points on the physical boundary. It is obvious that the accuracy of the modified approach is not as high as the static approach, but it is better than dynamic approach.

m	n	R	$\|u - u_m\|_\infty$
10	22	4.378	$3.258E{-4}$
15	32	6.672	$3.172E{-5}$
20	21	4.407	$3.702E{-5}$
	40	4.412	$2.651E{-5}$
	60	4.512	$2.097E{-5}$
30	31	4.699	$4.699E{-7}$
	40	4.562	$3.146E{-6}$
	62	4.631	$1.049E{-5}$

Table 4.17. *Absolute maximum errors using the modified approach for Example 4.5.1*

Next we compare the modified approach with the dynamic approach in Example 4.6.8, where the boundary condition is $u = x_1^2 - x_2^2$. We used the same parameters as mentioned above and obtained the numerical results as shown in Table 4.18. We observe that the modified approach is more accurate than the dynamic approach. Since fewer parameters are involved in the modified approach, the method is definitely more efficient. Overall, the modified approach appears to be good compromise between the static and dynamic approaches. □

m	n	R	$\|u - u_m\|_\infty$
10	22	16.58	$3.841E{-6}$
15	32	5.378	$7.671E{-6}$
20	21	5.289	$7.301E{-7}$
	40	5.536	$7.603E{-8}$
	60	5.549	$4.405E{-6}$
30	31	6.007	$1.672E{-5}$
	40	5.934	$1.023E{-5}$
	62	5.960	$9.371E{-6}$

Table 4.18. *Absolute maximum errors using the modified approach for Example 4.6.8*

.

4.8 Special Problems

4.8.1 Calculation of Eigenvalues of the Helmholtz Equation

In this section, we apply the static approach of the Method of Fundamental Solutions for the solution of a Helmholtz eigenvalue problem . This is of central importance in areas like acoustic and electromagnetic waves, e.g. in order to study the forced motion of a membrane. In this section, we only discuss the direct approach using the method of fundamental solutions for simple connected domain as shown in [DM76, Kar01]. We refer readers to the more sophisticated approaches using the method of fundamental solutions for simple and multi-connected domains in References [AA05, Reu05, Reu06b, Reu06a].

The **Helmholtz eigenvalue problem** requires the determination of a scalar λ called the **eigenvalue** and a nonzero function u called the **eigenfunction** solving

$$\begin{aligned} \left(\Delta u + \lambda^2\right) u &= 0 \quad \text{in } \Omega, \\ u &= 0 \quad \text{in } \Gamma, \end{aligned} \tag{4.8.1}$$

where Ω is a bounded domain in $I\!R^2$ with boundary Γ.

The fundamental solution of the Helmholtz equation for fixed λ is given by

$$G_\lambda(\mathbf{x}, \mathbf{y}) = -\frac{i}{4} H_0^{(2)}(\lambda r)$$

where $i = \sqrt{-1}$ and $H_0^{(2)}$ is the Hankel function of the second kind of order zero (see Table 4.1). Thus we can use the MFS to superimpose translates of the fundamental solution to make the superimposed boundary values as small as possible. For actual eigenvalues λ this will work better than for other values, and this describes our basic approach to the eigenvalue problem.

The fictitious boundary $\hat{\Gamma}$ where the n source points $\mathbf{y}_k$ will be located is taken to be of a similar shape as Γ at a fixed distance R from it. The same number n of collocation and source points are chosen in the implementation. The MFS for (4.8.1) yields a homogeneous linear system of the form

$$\sum_{k=1}^{n} \alpha_k G_\lambda(\mathbf{x}_j, \mathbf{y}_k) = 0, \quad j = 1, ..., n. \tag{4.8.2}$$

The above equation can be written in matrix form

$$\mathbf{G}_\lambda \mathbf{a} = \mathbf{0}, \tag{4.8.3}$$

where the entries of $\mathbf{G}_\lambda \in C^{n\times n}$ are given by

$$\mathbf{G}_\lambda(\mathbf{x}_k, \mathbf{y}_j) = -\frac{i}{4} H_0^{(2)}(\lambda r(\mathbf{x}_k, \mathbf{y}_j)), \quad j, k = 1, 2, ..., n,$$

and $\mathbf{a} = \{\alpha_1, \alpha_2, \cdots, \alpha_n\}$. Note that (4.8.3) is a set of complex algebraic equations for the unknowns α_j. In order to obtain a nontrivial solution of the system (4.8.3), the determinant det $(\mathbf{G}_\lambda)$ of the coefficient matrix $\mathbf{G}_\lambda$ must be zero. Thus, the eigenvalues of (4.8.1) can hopefully be approximately obtained if we can find λ such that

$$\det(\mathbf{G}_\lambda) = 0.$$

Taking the real and imaginary parts of det $(\mathbf{G}(\lambda))$, we have

$$\mathcal{R}\left[\det(\mathbf{G}(\lambda))\right] = 0 \tag{4.8.4}$$

and

$$\mathcal{I}\left[\det(\mathbf{G}(\lambda))\right] = 0 \tag{4.8.5}$$

where $\mathcal{R}$ denotes the real part and $\mathcal{I}$ the imaginary part. The two nonlinear equations (4.8.4) and (4.8.5) can be solved independently using standard numerical library software such as IMSL, NAG, or Netlib.

In [Kar01], the MFS was used to calculate the smallest eigenvalue λ_0 of (4.8.1) in the case when Ω is the unit disk. The exact value of λ_0 is the smallest positive zero $\lambda_0 \simeq 2.4048255577$ of the Bessel function J_0 of first kind [AS65]. Using the MFS, the numerical results for λ_0 and the corresponding absolute errors for various numbers of degrees of freedom for (4.8.2) are presented in Table 4.19 [Kar01], where R was taken to be 0.6. The results were not particularly sensitive to the value of R, provided it was not too small or too large. In this case, the MFS approach is much superior to the boundary integral method [DM76] in terms of accuracy and efficiency.

4.8.2 Nonsmooth Boundary Conditions

The treatment of boundary singularities by various techniques using boundary element methods has been the subject of several studies. Karageorghis and his co-workers [Kar92, KF87, KF88, KF89b, PKG98] modified and extended these techniques in conjunction with the MFS. There are two main reasons for using the MFS. First, due to its adaptivity, the MFS to some extent absorbs the effects caused by the presence of boundary singularities [JFK87, KF87]. Second, since in the MFS the solution is expressed in terms of a linear combination of fundamental

	$\mathcal{R}\,[\det\,(\mathbf{G})]$		$\mathcal{I}\,[\det\,(\mathbf{G})]$	
n	λ_0	Error	λ_0	Error
12	2.409537	$0.471E-2$	2.400311	$0.451E-2$
16	2.404908	$0.821E-4$	2.404662	$0.163E-3$
20	2.404835	$0.943E-5$	2.404806	$0.197E-4$
24	2.404827	$0.117E-5$	2.404823	$0.245E-5$
28	2.404826	$0.150E-6$	2.404825	$0.316E-6$
32	2.404826	$0.198E-7$	2.404806	$0.417E-7$
50 [DM76]	2.4121		2.4119	

Table 4.19. *The smallest eigenvalue for various n.*

solutions, it is only natural to include terms prescribing the analytical behavior of the boundary singularity in this expansion.

Let us consider the problem (4.2.1)–(4.2.2) with $L = \Delta$. Assume that the problem is posed in such a way (e.g. by incoming domain corners) that the solution has a singularity at some boundary point O on Γ. A special singular solution u of the problem can be expressed in a neighborhood of O by

$$u(\mathbf{x}) = \sum_{k=1}^{\infty} \alpha_k r^{\lambda_k} f_k(\theta),$$

where r and θ are local polar coordinates for $\mathbf{x} = (r\cos\theta, r\sin\theta)$ centered at O, and where r^{λ_k} and $f_k(\theta)$ are known. Based on this observation, Karageorghis [Kar92] proposed to assemble the approximate solution as the sum of a singular solution u_n^s and a regular solution u_n^r as

$$\begin{aligned} u_n(\mathbf{x}) &= u_n^s(\mathbf{x}) + u_n^r(\mathbf{x}) \\ &= \sum_{i=1}^{k} \alpha_i r^{\lambda_i} f_i(\theta) + \sum_{k=1}^{n} a_k G(\mathbf{x}, \mathbf{y}_k), \quad \mathbf{y}_k \in \hat{\Gamma}, \end{aligned} \tag{4.8.6}$$

where $\hat{\Gamma}$ denotes the fictitious boundary. Note that u_n satisfies (4.2.1) since both the fundamental solution $G(\mathbf{x}, \mathbf{y})$ and singular solutions $r^{\lambda_i} f(\theta)$ are harmonic, provided that the λ_i are chosen appropriately, e.g. dependent on the angle of incoming domain corners.

A similar approach can be applied to the biharmonic problem. Then the solution of the boundary value problem (4.6.2)–(4.6.3) can be approximated by (see [Kar92])

$$u_n(\mathbf{x}) = \sum_{i=1}^{k} \alpha_i r^{\lambda_i} f_i(\theta) + \sum_{j=1}^{n} a_j G_1(\mathbf{x}, \mathbf{y}_j) + \sum_{j=1}^{n} b_j G_2(\mathbf{x}, \mathbf{y}_j), \quad y_j \in \hat{\Gamma},$$

where G_1 and G_2 are the fundamental solutions of Δ and Δ^2.

In contrast to the modified MFS described above, a new version of the MFS was proposed by Poullikkas et. al. [PKG98] in which the singular term u_n^s in (4.8.6) only contains one term

$$u_n^s = \alpha r^\beta \cos(\beta\theta), \tag{4.8.7}$$

where α is the unknown coefficient of the special singular solution.

To show the effectiveness of the above modified versions of the MFS, the following problem, known as the **Motz problem**, was considered by Karageorghis [Kar92].

Example 4.8.8 Consider

$$\begin{aligned}
\Delta u &= 0, && \mathbf{x} \in \Omega, \\
\frac{\partial u}{\partial n} &= 0, && \mathbf{x} \in OA, BC, CD \\
u &= 1000, && \mathbf{x} \in AB \\
u &= 500, && \mathbf{x} \in DO,
\end{aligned}$$

where Ω is shown in Figure 4.22. The Motz problem is considered as a benchmark problem for testing various numerical methods for problems with boundary singularities. The singularity occurs at the point O where the boundary condition suddenly changes from $u = 500$ to $\partial u/\partial x_2 = 0$.

The point O represents the tip of the slit, and the problem possesses a singular solution of the form

$$u(\mathbf{x}) = 500 + \alpha_1 r^{1/2} \cos\frac{\theta}{2} + \alpha_2 r^{3/2} \cos\frac{3\theta}{2} + \cdots$$

in the neighborhood of O [Sym73]. From (4.8.6), two terms of u_n^s were chosen. The solution is approximated by

$$u_n(\mathbf{x}) = \alpha_1 r^{1/2} \cos\frac{\theta}{2} + \alpha_2 r^{3/2} \cos\frac{3\theta}{2} + \sum_{j=1}^{n} a_j G(\mathbf{x}, \mathbf{y}_j), \quad \mathbf{y}_j \in \hat{\Gamma}.$$

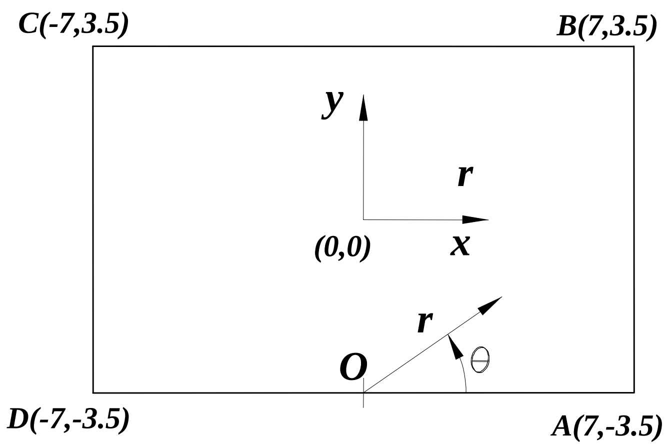

Fig. 4.22. Motz Problem.

By taking derivatives with respect to x and y, we obtain

$$\frac{\partial}{\partial x} r^{\frac{1}{2}} \cos\frac{\theta}{2} = \frac{1}{2} r^{\frac{-1}{2}} \cos\frac{\theta}{2}, \quad \frac{\partial}{\partial y} r^{\frac{1}{2}} \cos\frac{\theta}{2} = \frac{1}{2} r^{\frac{-1}{2}} \sin\frac{\theta}{2}$$

$$\frac{\partial}{\partial x} r^{\frac{3}{2}} \cos\frac{\theta}{2} = \frac{3}{2} r^{\frac{1}{2}} \cos\frac{\theta}{2}, \quad \frac{\partial}{\partial y} r^{\frac{3}{2}} \cos\frac{\theta}{2} = -\frac{3}{2} r^{\frac{1}{2}} \sin\frac{\theta}{2}.$$

The boundary conditions become

$$\begin{cases} \dfrac{\partial u_n}{\partial n} = 0, & \mathbf{x} \in OA, \\ u_n + \alpha_1 r^{\frac{1}{2}} \cos\dfrac{\theta}{2} + \alpha_2 r^{\frac{3}{2}} \cos\dfrac{3\theta}{2} - 1000 = 0, & \mathbf{x} \in AB, \\ \dfrac{\partial u_n}{\partial n} + \dfrac{\alpha_1}{2} r^{\frac{-1}{2}} \sin\dfrac{\theta}{2} - \dfrac{3\alpha_2}{2} r^{\frac{1}{2}} \sin\dfrac{\theta}{2} = 0, & \mathbf{x} \in BC, \\ \dfrac{\partial u_n}{\partial n} - \dfrac{\alpha_1}{2} r^{\frac{-1}{2}} \cos\dfrac{\theta}{2} - \dfrac{3\alpha_2}{2} r^{\frac{1}{2}} \cos\dfrac{\theta}{2} = 0, & \mathbf{x} \in CD, \\ u_n - 500 = 0, & \mathbf{x} \in DO. \end{cases}$$

The problem was tackled by the following four different methods [PKG98]:

MFS1 Naive MFS.
MFS2 Modified MFS with α_1 as only additional unknown.
MFS3 Modified MFS with α and β in (4.8.7) as additional unknowns.
MFS4 Modified MFS with α_1 and α_2 as unknowns [Kar92].

The sources were initially placed outside the domain at different distances d from the boundary, as before. Due to the strong nonlinearity of the resulting problem in the new modified method, convergence was achieved only for a restricted range of d less than 0.4. The exact solution [RP75] and approximate solutions using the MFS1 − MFS4 with $n = 66, m = 7$, and a maximum of 4000 function evaluations are shown in Table 4.20. The numerical results reveal that MFS4 is the most accurate.

(x, y)	MFS1	MFS2	MFS3	MFS4	Exact
$(-0.5, -3.0)$	578.95	547.59	546.04	546.31	546.24
$(-0.25, -3.0)$	605.05	559.30	557.32	557.72	557.64
$(0.0, -3.0)$	631.08	576.48	574.20	574.70	574.61
$(0.25, -3.0)$	655.27	598.17	595.82	596.31	596.23
$(0.5, -3.0)$	677.50	620.75	618.49	618.92	618.85
$(-0.5, -3.25)$	553.03	526.03	524.63	524.89	524.81
$(-0.25, -3.25)$	590.52	535.61	533.17	533.71	533.59
$(0.0, -3.25)$	622.50	555.54	552.54	553.30	553.19
$(0.25, -3.25)$	649.86	586.07	583.12	583.76	583.67
$(0.5, -3.5)$	673.90	614.14	611.44	611.93	611.86
$(-0.5, -3.5)$	530.90	501.69	499.97	500.11	500.00
$(-0.25, -3.5)$	582.26	502.50	499.63	500.18	500.00
$(0.0, -3.5)$	618.48	503.46	496.80	500.26	500.00
$(0.25, -3.5)$	647.70	579.39	575.80	576.48	576.41
$(0.5, -3.5)$	672.73	611.66	608.48	608.96	608.91

Table 4.20. *Approximate solution near the singularity O.*

The calculated values of α_1 and β_1 are given in Table 4.21 for various values of the initial distance of the moving sources from the boundary. These are in excellent agreement with the exact values $\alpha_1 = 151.63, \beta_1 = 0.5$.

□

m	n	d	$NFEV$	α_1	β_1
7	66	0.4	2500	149.72	0.45
			3000	150.85	0.44
			3500	150.88	0.44
			4000	150.88	0.44
7	66	0.6	2500	150.14	0.47
			3000	150.14	0.47
			3500	150.14	0.47
			4000	150.14	0.47
10	96	0.5	2500	150.92	0.47
			3000	150.51	0.47
			3500	150.43	0.47
			4000	150.39	0.47
10	96	0.6	2500	149.77	0.48
			3000	149.17	0.48
			3500	149.13	0.48
			4000	149.13	0.48

Table 4.21. *Approximate value of* α_1 *and* β_1.

4.8.3 Axisymmetric Potential Problems

When working on an axisymmetric three-dimensional domain Ω, the **potential equation** using the **Laplace operator** can be rewritten as

$$\Delta u = \frac{\partial^2 u}{\partial r^2} + \frac{1}{r}\frac{\partial u}{\partial r} + \frac{\partial^2 u}{\partial z^2} = 0, \quad \mathbf{x} \in \Omega, \tag{4.8.9}$$

where z is the coordinate along the axis and r is the distance to the axis. If the boundary conditions are also axisymmetric, a 3D problem can be reduced to a 2D problem in the variables r and z.

Karageorghis and Fairweather [KF99] considered the following axisymmetric mixed potential problem

$$\begin{aligned} \Delta u &= 0, \quad \mathbf{x} \in \Omega, \\ Bu &= 0, \quad \mathbf{x} \in \Gamma, \end{aligned} \tag{4.8.10}$$

on an axisymmetric domain $\Omega \subset I\!R^3$ with boundary Γ. The boundary operator B has the form

$$Bu = \begin{cases} \alpha(\mathbf{x}) + u(\mathbf{x}), & \mathbf{x} \in \Gamma_1^D, \quad \text{(Dirichlet part)} \\ \alpha(\mathbf{x}) + \dfrac{\partial u}{\partial n}(\mathbf{x}), & \mathbf{x} \in \Gamma_2^N, \quad \text{(Neumann part)} \\ \alpha(\mathbf{x}) + \beta(\mathbf{x})u(\mathbf{x}) + \gamma(\mathbf{x})\dfrac{\partial u}{\partial n}(\mathbf{x}), & \mathbf{x} \in \Gamma_3^R, \quad \text{(Robin part)} \end{cases}$$

where α, β, and γ are prescribed axisymmetric functions.

4.8.4 Axisymmetric boundary conditions

Since the boundary conditions are assumed to be axisymmetric, the 3D problem in (4.8.10) reduces to a 2D problem via (4.8.9). Let $\mathbf{p} = (r_\mathbf{p}, z_\mathbf{p})$ and $\mathbf{q} = (r_\mathbf{q}, z_\mathbf{q})$ be two points in Ω in axisymmetric coordinates (r, z) and define

$$\rho(\mathbf{p}, \mathbf{q}) = \sqrt{(r_\mathbf{p} - r_\mathbf{q})^2 + (z_\mathbf{p} - z_\mathbf{q})^2} \qquad (4.8.11)$$

to be their standard Euclidean distance in these coordinates. Then the fundamental solution of equation (4.8.9) is given by

$$G(\mathbf{p}, \mathbf{q}) = \frac{4K(\kappa(\mathbf{p}, \mathbf{q}))}{\rho(\mathbf{p}, \mathbf{q})} \qquad (4.8.12)$$

where

$$K(\kappa) = \int_0^{2\pi} \frac{1}{\sqrt{1 - \kappa^2 \sin^2(\theta)}}\, d\theta \quad \text{and} \quad \kappa^2(\mathbf{p}, \mathbf{q}) = \frac{4 r_\mathbf{p} r_\mathbf{q}}{\rho^2(\mathbf{p}, \mathbf{q})}. \qquad (4.8.13)$$

is the complete elliptic integral of the first kind. Furthermore, since

$$\frac{\partial G(\mathbf{p}, \mathbf{q})}{\partial n} = \frac{\partial G(\mathbf{p}, \mathbf{q})}{\partial r} n_r + \frac{\partial G(\mathbf{p}, \mathbf{q})}{\partial z} n_z$$

where $\partial/\partial n$ denotes the outward normal derivative at the boundary point $\mathbf{q}$, and n_r and n_z are the components of the outward unit vector to Γ in the r and z direction, respectively, one obtains

$$\begin{aligned} &\frac{\partial G(\mathbf{p}, \mathbf{q})}{\partial n} \\ = \; &\frac{2\left\{\rho^{3/2}\left[E(\kappa) - K(\kappa)(1-\kappa^2)\right] - 2r_\mathbf{q}(r_\mathbf{q} + r_\mathbf{p})E(\kappa)\right]\}}{r_\mathbf{q}\rho^{3/2}(1-\kappa)^2} n_r \\ &- \frac{4(z_\mathbf{q} - z_\mathbf{p})E(\kappa)}{\rho^{3/2}(1-\kappa^2)} n_z \end{aligned}$$

where $E(\kappa)$ is the complete elliptic integral of the second kind defined by

$$E(\kappa) = \int_0^{2\pi} \sqrt{1 - \kappa^2 \sin^2(\theta)} d\theta,$$

and where the dependence of ρ and κ on $\mathbf{p}$ and $\mathbf{q}$ is dropped to simplify the notation. Once $G(\mathbf{p}, \mathbf{q})$ and $\partial G(\mathbf{p}, \mathbf{q})/\partial n$ are available, the numerical approximation u_n of the true solution can be represented by (4.2.8). Note that in the case of simple connected domains, no boundary points are placed on the axis of rotation.

The complete elliptic integrals $K(\kappa)$ and $E(\kappa)$ are evaluated using the NAG routines S21BBF and S21BCF, or the IMSL routines ELK and ELE, respectively.

Example 4.8.14 The problem of steady-state heat conduction through a hollow cylinder with insulated ends is investigated [KF99] using the MFS. The inner and outer radii of the cylinder are denoted by r_i and r_o, respectively. The boundary conditions are prescribed as

$$\begin{aligned} u &= f, \quad \text{for} \quad r = r_i, \\ u &= g, \quad \text{for} \quad r = r_o, \\ \frac{\partial u}{\partial z} &= 0, \quad \text{for} \quad z = \pm z_0. \end{aligned}$$

The analytical solution [Cha67] is

$$u = f + (g - f)\frac{\ln(r/r_i)}{\ln(r_o/r_i)}.$$

The following parameter values were chosen:

$$r_i = 1, \ r_o = 2, \ z_0 = 0.5, \ f = 1, \ g = 2.$$

The numerical computations were performed on a 0.1×0.1 grid for various values of

NFEV	=	number of function evaluations
m	=	number of collocation points
n	=	number of source points

The errors $\|u - u_n\|_\infty$ and $\|\partial u/\partial r - \partial u_n/\partial r\|_\infty$ are shown in Table 4.22 and 4.23.

□

NFEV	$\|u-u_n\|_\infty$		
	$m=32, n=4$	$m=64, n=8$	$m=128, n=16$
500	$7.0E{-}3$	$1.5E{-}3$	$3.8E{-}4$
1000	$6.9E{-}3$	$5.9E{-}4$	$1.9E{-}5$
2000	$4.2E{-}3$	$3.6E{-}4$	$9.9E{-}6$
3000	$3.1E{-}3$	$2.3E{-}5$	$8.1E{-}6$
4000	$3.0E{-}3$	$2.0E{-}5$	$5.6E{-}6$
5000	$3.0E{-}3$	$1.7E{-}5$	$3.7E{-}6$

Table 4.22. *Results for the temperature distribution.*

NFEV	$\|\partial u/\partial r-\partial u_n/\partial r\|_\infty$		
	$m=32, n=4$	$m=64, n=8$	$m=128, n=16$
500	$4.0E{-}2$	$1.5E{-}2$	$3.8E{-}3$
1000	$6.9E{-}2$	$5.9E{-}3$	$1.9E{-}4$
2000	$4.2E{-}2$	$3.6E{-}3$	$9.9E{-}4$
3000	$3.1E{-}2$	$2.3E{-}4$	$8.1E{-}5$
4000	$3.0E{-}2$	$2.0E{-}4$	$5.6E{-}5$
5000	$3.0E{-}2$	$1.7E{-}4$	$3.7E{-}5$

Table 4.23. *Results for the flux.*

4.8.5 Non–axisymmetric boundary conditions

For non–axisymmetric boundary conditions, the solution u is expanded in Fourier series as

$$u(\mathbf{p}) = \frac{u_0(r_\mathbf{p}, z_\mathbf{p})}{2} + \sum_{i=1}^{\infty} \left[u_i^c(r_\mathbf{p}, z_\mathbf{p})\cos(i\theta_p) + u_i^s(r_\mathbf{p}, z_\mathbf{p})\sin(i\theta_p)\right] \tag{4.8.15}$$

where $\mathbf{p} = (r_\mathbf{p}, z_\mathbf{p})$, and where the functions u_i^c and u_i^s satisfy a differential equation of the form

$$\frac{\partial^2 u(\mathbf{p})}{\partial r^2} + \frac{1}{r}\frac{\partial u(\mathbf{p})}{\partial r} + \frac{\partial^2 u(\mathbf{p})}{\partial z^2} - \left(\frac{i}{r}\right)^2 u = 0,\ \mathbf{p} \in \Omega, \text{for}\, i = 0, 1, 2, \cdots. \tag{4.8.16}$$

The fundamental solutions of (4.8.16) can be written in terms of complete elliptic integrals [Gup79] as

$$G_i = \frac{4K_i(\kappa)}{\rho},\ i \geq 0$$

as in (4.8.12) and (4.8.13), with

$$\begin{aligned} K_0(\kappa) &= K(\kappa), \\ K_i(\kappa) &= (-1)^i i \sum_{j=0}^{m} L_j^i 2^{2(i-j)} C_{i-j}, \quad i \geq 1, \\ L_j^i &= (-1)^j \frac{(2i-j-1)!}{j!(2i-2j)!}, \quad j = 0, 1, \cdots, i, \end{aligned}$$

$$\begin{aligned} C_0 &:= K(\kappa), \\ C_1 &:= \frac{1}{\kappa^2}\left(E(\kappa) - (1-\kappa^2)K(\kappa)\right) \\ C_n &:= \frac{(2n-2)\left(2\kappa^2-1\right)C_{n-1} + (2n-3)\left(1-\kappa^2\right)C_{n-2}}{(2n-1)\kappa^2}, \ n \geq 2. \end{aligned}$$

The normal derivatives of the fundamental solutions are [Gup79]

$$\begin{aligned} &\frac{\partial G_i(\mathbf{p},\mathbf{q})}{\partial n} \\ = \ &\frac{4}{\rho^2}\left[\left(\rho \frac{\partial K_i}{\partial \kappa}\frac{\partial \kappa}{\partial r_{\mathbf{q}}} - K_i \frac{\partial \rho}{\partial r_{\prime}}\right) n_r + \left(\rho \frac{\partial K_m}{\partial \kappa}\frac{\partial \kappa}{\partial z_{\mathbf{q}}} - K_i \frac{\partial \rho}{\partial z_{\mathbf{q}}}\right) n_z\right], \end{aligned}$$

where

$$\begin{aligned} \frac{\partial \rho}{\partial r_{\mathbf{q}}} &= \frac{r_{\mathbf{q}} + r_{\mathbf{p}}}{\rho}, \\ \frac{\partial \rho}{\partial z_{\mathbf{q}}} &= \frac{z_{\mathbf{q}} - z_{\mathbf{p}}}{\rho}, \\ \frac{\partial \kappa}{\partial r_{\mathbf{q}}} &= \frac{2r_{\mathbf{p}} - \kappa^2(r_{\mathbf{q}} + r_{\mathbf{p}})}{\kappa \rho^2}, \\ \frac{\partial \kappa}{\partial z_{\mathbf{q}}} &= \frac{\kappa\left(z_{\mathbf{q}} - z_{\mathbf{p}}\right)}{\rho^2}. \end{aligned}$$

Furthermore,

$$\begin{aligned} \frac{\partial K_0(k)}{\partial k} &= \frac{\partial K(k)}{\partial k} \\ \frac{\partial K_i(k)}{\partial k} &= (-1)^i i \sum_{j=0}^{i} L_j^m 2^{2(m-j)} \frac{\partial C_{i-j}}{\partial k}, \quad i \geq 1, \end{aligned}$$

$$\frac{\partial C_\ell}{\partial \kappa} = \begin{cases} \dfrac{E(\kappa) - (1-\kappa^2)K(\kappa)}{\kappa(1-\kappa^2)}, & \ell = 0, \\ \dfrac{(2-\kappa^2)K(\kappa) - 2E(\kappa)}{\kappa^3}, & \ell = 1, \\ \dfrac{2\ell-2}{(2\ell-1)\kappa^2}\left[\left(2\kappa^2-1\right)\dfrac{\partial C_{\ell-1}}{\partial \kappa} + \dfrac{2}{\kappa}C_{\ell-1}\right] & \\ \quad + \dfrac{2\ell-3}{(2\ell-1)\kappa^2}\left[\left(1-\kappa^2\right)\dfrac{\partial C_{\ell-2}}{\partial \kappa} - \dfrac{2}{\kappa}C_{\ell-2}\right], & \ell \geq 2. \end{cases}$$

Example 4.8.17 Here, the problem of uniaxial flow in a finite cylindrical region of an infinite medium is considered. The direction of heat flow in this case is not along the axis of the cylindrical region, thus resulting in non–axisymmetric boundary conditions. Let the boundary conditions be as follows [Gup79]:

$$\begin{aligned} u &= 2r\cos\theta \quad \text{on } z = 0, \\ u &= 2r\cos\theta \quad \text{on } z = 2, \\ u &= 2\cos\theta \quad \text{on } r = 2. \end{aligned}$$

The exact solution of the problem is given by $u = 2r\cos\theta$. Note that the only non–zero Fourier coefficient is u_1^c in (4.8.15); i.e. $i = 1$. The errors $e = \|u - u_m\|_\infty$ and $e_r = \|\partial u/\partial r - \partial u_m/\partial r\|_\infty$ are computed on a grid of cell width 0.2 and the results [KF99] are shown in Table 4.24.
□

	$n=3, m=24$		$n=6, m=48$		$n=12, m=96$	
$NFEV$	e	e_r	e	e_r	e	e_r
500	$2.1E{-5}$	$5.1E{-5}$	$9.6E{-5}$	$5.4E{-4}$	$2.9E{-3}$	$3.9E{-3}$
1000	$1.5E{-5}$	$3.8E{-5}$	$2.2E{-5}$	$1.9E{-4}$	$2.2E{-4}$	$7.4E{-4}$
2000	$1.0E{-5}$	$2.5E{-5}$	$8.7E{-6}$	$4.7E{-5}$	$1.3E{-5}$	$3.8E{-5}$
3000	$7.6E{-6}$	$1.8E{-5}$	$4.2E{-6}$	$2.2E{-5}$	$4.2E{-6}$	$2.9E{-5}$
4000	$5.6E{-6}$	$1.3E{-5}$	$2.1E{-6}$	$1.4E{-5}$	$3.2E{-6}$	$1.9E{-5}$
5000	$4.3E{-4}$	$1.0E{-5}$	$1.5E{-6}$	$9.4E{-6}$	$1.5E{-6}$	$8.4E{-6}$

Table 4.24. *Errors of* $e = \|u - u_m\|_\infty$ *and* $e_r = \|\partial u/\partial r - \partial u_m/\partial r\|_\infty$

4.8.6 Acoustic Scattering and Radiation Problems

Such problems are governed by the Helmholtz equation

$$\begin{aligned}(\Delta + \lambda^2)u(\mathbf{p}) &= 0, \qquad \mathbf{p} \in \Omega,\\ Bu(\mathbf{p}) &= f(\mathbf{p}), \quad \mathbf{p} \in \Gamma,\end{aligned}$$

where B is a boundary operator and λ the wave number. Let $\mathbf{p} = (r_{\mathbf{p}}, z_{\mathbf{p}}, \theta_{\mathbf{p}}), \mathbf{q} = (r_{\mathbf{q}}, z_{\mathbf{q}}, \theta_{\mathbf{q}})$. Define

$$\rho\left(\mathbf{p}, \mathbf{q}\right) = \sqrt{r_{\mathbf{p}}^2 + r_{\mathbf{q}}^2 - 2r_{\mathbf{p}}r_{\mathbf{q}}\cos(\theta_{\mathbf{q}} - \theta_{\mathbf{p}}) + (z_{\mathbf{q}} - z_{\mathbf{p}})^2}.$$

The fundamental solution of the axisymmetric version of the Helmholtz equation is given by

$$G(\mathbf{p}, \mathbf{q}) = \int_0^{2\pi} \frac{e^{-i\lambda\rho(\mathbf{p},\mathbf{q})}}{\rho\left(\mathbf{p}, \mathbf{q}\right)} d\theta(\mathbf{q}).$$

$G(\mathbf{p}, \mathbf{q})$ can be written into two parts, consisting of a nonsingular and a singular integral

$$\begin{aligned}G(\mathbf{p}, \mathbf{q}) &= \int_0^{2\pi} \frac{e^{-i\lambda\rho(\mathbf{p},\mathbf{q})} - 1}{\tilde{\rho}\left(\mathbf{p}, \mathbf{q}\right)} d\theta(\mathbf{q}) + \int_0^{2\pi} \frac{1}{\rho\left(\mathbf{p}, \mathbf{q}\right)} d\theta(\mathbf{q})\\ &= G_1(\mathbf{p}, \mathbf{q}) + G_2(\mathbf{p}, \mathbf{q}).\end{aligned}$$

Notice that G_1 can be evaluated using a standard quadrature rule. As shown in the previous example, we have

$$G_2(\mathbf{p}, \mathbf{q}) = \frac{4K(\kappa)}{\rho}$$

where $K(\kappa)$ and ρ are given in (4.8.11) and (4.8.13), and the normal derivative of $G(\mathbf{p}, \mathbf{q})$ can be written in terms of $K(\kappa)$ and $E(\kappa)$. The formulation of the MFS is similar to the bi-harmonic equation described in Section 4.6. As a result, there are $4M$ unknowns to be determined using dynamic approach. For the acoustic radiation and scattering problems for axisymmetric bodies in the half space $\{z > 0\}$, the problem of discretizing the plane $z = 0$ is avoided by using an appropriate fundamental solution. When the plane is rigid, the following fundamental solution is available:

$$G^H(\mathbf{p}, \mathbf{q}) = \int_0^{2\pi} \frac{e^{-i\lambda\rho(\mathbf{p},\mathbf{q})} - 1}{\tilde{\rho}\left(\mathbf{p}, \mathbf{q}\right)} d\theta(\mathbf{q}) + \int_0^{2\pi} \frac{e^{-i\lambda\rho(\mathbf{p}',\mathbf{q})} - 1}{\rho\left(\mathbf{p}', \mathbf{q}\right)} d\theta(\mathbf{q})$$

where the point $\mathbf{p}'$ is the image point of $\mathbf{p}$ in the plane. This fundamental solution satisfies the axisymmetric version of the Helmholtz equation and the boundary condition for a rigid plane, $\partial u/\partial z = 0$ on $z = 0$.

For numerical examples, we refer the reader to [KF98].

4.8.7 Nonlinear Boundary Conditions

One of the criticisms of the MFS, when the dynamic approach is used, is that linear problems are solved by means of a nonlinear technique. In this subsection, we present problems which are governed by Laplace's equation and are subject to **nonlinear** boundary conditions [KF89a]. The mathematical formulation of the MFS does not depend on the linearity of the boundary conditions, as we shall see, and it can be easily adapted to such problems.

Consider the following Laplace equation subject to the nonlinear boundary condition

$$\begin{aligned} \Delta u &= 0, \quad \mathbf{x} \in \Omega, \\ Bu &= 0, \quad \mathbf{x} \in \Gamma, \end{aligned}$$

where B is a nonlinear operator. Similar to the problems with linear boundary conditions, the approximate solution u_n can be represented by a linear combination of fundamental solutions as shown in (4.2.8). In particular, Karageorghis and Fairweather [KF89a] considered operators of the form

$$Bu(\mathbf{x}) = \alpha(\mathbf{x})u(\mathbf{x}) + \beta(\mathbf{x})\frac{\partial u}{\partial n}(\mathbf{x}) + \gamma(\mathbf{x})u(\mathbf{x})^4 + \delta(\mathbf{x}), \quad \mathbf{x} \in \Gamma.$$

The least squares subroutines outlined in Section 4.6 can be used to minimize the sum of squares

$$\varepsilon_m = \sum_{i=1}^{m} |f_i|^2$$

where

$$\begin{aligned} f_i \;=\; & \alpha(\mathbf{x}_i)\sum_{k=1}^{n} a_k G(\mathbf{x}_i, \mathbf{y}_k) + \beta(\mathbf{x}_i)\sum_{k=1}^{n} a_k \frac{\partial}{\partial n} G(\mathbf{x}_i, \mathbf{y}_k) \\ & + \gamma(\mathbf{x}_i)\left(\sum_{k=1}^{n} a_k G(\mathbf{x}_i, \mathbf{y}_k)\right)^4 + \delta(\mathbf{x}_i), \; 1 \le i \le m. \end{aligned}$$

The initial location of source points and the initial values of a_k are taken to be the same as in the linear cases mentioned in the previous sections.

Example 4.8.18 Consider the following nonlinear boundary problem

$$
\begin{aligned}
\Delta u &= 0, && \mathbf{x} \in \Omega, \\
\frac{\partial u}{\partial n} &= u(\mathbf{x}) + u^4(\mathbf{x}), && \mathbf{x} \in AB, \\
\frac{\partial u}{\partial n} &= -u(\mathbf{x}) - u^4(\mathbf{x}), && \mathbf{x} \in BC, \\
\frac{\partial u}{\partial n} &= 0, && \mathbf{x} \in CD, \\
u &= 1, && \mathbf{x} \in DA,
\end{aligned}
$$

where $\Omega = \{\mathbf{x} = (x_1, x_2) \, : \, 0 \leq x_1 \leq 1, \ 0 \leq x_2 \leq 0.1\}$. The problem domain Ω is shown in Figure 4.23. This problem arises in the study of heat transfer from finned surfaces (Kern and Kraus [KK72]), which was solved using a boundary element approach by Ingham et. al. [IHM81]. The problem was also solved by Karageorghis and Fairweather [KF89a] using the MFS, and their results were in good agreement with those obtained by Ingham et. al. [IHM81]. In addition to the difficulties of nonlinear boundary conditions, the large aspect ratio of the region $(AB = 1, BC = 0.1)$ causes further complications.

The numerical results at some selected points obtained by Karageorghis and Fairweather [KF89a] using the dynamic approach are shown in Table 4.25. Various m, n and maximal numbers $NFEV$ of function evaluations were used in their tests. Table 4.25. uses the following notations:

MFS1: $m = 13, n = 120, NFEV = 1500$
MFS2: $m = 13, n = 120, NFEV = 2000$
BEM: Ingham et. al. [IHM81]

□

4.9 Inverse Problems

Let Ω be a simply connected domain in $I\!R^d$, $d = 2, 3$, and Γ_1, Γ_2, Γ_3 be three parts of the boundary Γ. Suppose that $\Gamma_1 \cup \Gamma_2 \cup \Gamma_3 = \Gamma$, $\Gamma_i \cap \Gamma_j = \phi$ for $i \neq j$, and either Γ_1 or Γ_2 can be empty set. We want to predict the temperature and heat flux on the boundary Γ_3 from given Dirichlet data on Γ_1, Neumann data on Γ_2, and scattered measurement data at some interior points. This is a typical **inverse problem**.

Consider the standard heat equation:

$$
\frac{\partial u}{\partial t}(\mathbf{x}, t) = a^2 \Delta u(\mathbf{x}, t), \qquad \mathbf{x} \in \Omega \subset I\!R^d, \quad t \in (0, t_{max}), \tag{4.9.1}
$$

(x_1, x_2)	MFS1	MFS2	BEM
$(0.2, 0.02)$	0.5091	0.5110	0.5073
$(0.4, 0.02)$	0.2714	0.2745	0.2734
$(0.6, 0.02)$	0.1494	0.1510	0.1510
$(0.2, 0.04)$	0.5171	0.5192	0.5152
$(0.4, 0.04)$	0.2751	0.2783	0.2772
$(0.6, 0.04)$	0.1515	0.1531	0.1531
$(0.2, 0.06)$	0.5229	0.5250	0.5210
$(0.4, 0.06)$	0.2778	0.2810	0.2800
$(0.6, 0.06)$	0.1530	0.1546	0.1546

Table 4.25. *Approximate solution of u at 9 selected points.*

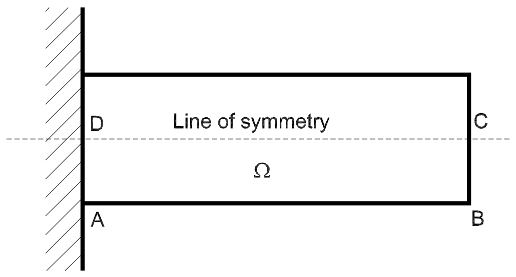

Fig. 4.23. The problem domain Ω.

with the initial condition

$$u(\mathbf{x}, 0) = \varphi(\mathbf{x}), \qquad \mathbf{x} \in \overline{\Omega}, \tag{4.9.2}$$

the Dirichlet boundary condition

$$u(\mathbf{x}, t) = f(\mathbf{x}, t), \qquad \mathbf{x} \in \Gamma_1, \quad t \in (0, t_{max}], \tag{4.9.3}$$

and the Neumann boundary condition

$$\frac{\partial u}{\partial n}(\mathbf{x}, t) = g(\mathbf{x}, t), \qquad \mathbf{x} \in \Gamma_2, \quad t \in (0, t_{max}], \tag{4.9.4}$$

where n is the outer unit normal with respect to Γ_2.

Let $\{\mathbf{x}_i\}_{i=1}^{\ell}$ be a set of points contained in $\overline{\Omega}$. At each point $\mathbf{x}_i$, there are I_i noisy measurements $\hat{h}_i^{(k)},\ k = 1, 2, \cdots, I_i$ of temperature at times $t_i^{(k)}$ each, such that we have the additional conditions

$$u(\mathbf{x}_i, t_i^{(k)}) = \hat{h}_i^{(k)}, \quad i = 1, 2, \cdots, \ell, \quad k = 1, 2, \cdots, I_i. \tag{4.9.5}$$

The absolute error between the noisy measurements and exact data is assumed to be bounded as $|\hat{h}_i^{(k)} - h_i^{(k)}| \leq \eta$ for all measurement points at all measured times. Here, the constant η is called the noise level of the input data.

Our goal is to reconstruct the temperature u and the heat flux $\partial u/\partial n$ on the boundary Γ_3 from the given conditions in (4.9.1)–(4.9.4) and the scattered noisy measurements (4.9.5).

It is well known that the fundamental solution of the heat equation (4.9.1) is given by

$$G(\mathbf{x}, \mathbf{y}, t) = \frac{1}{(4\pi a^2 t)^{d/2}} \exp\left(-\frac{r^2}{4a^2 t}\right) H(t),$$

where $r = \|\mathbf{x} - \mathbf{y}\|$ and $H(t)$ is the Heaviside function. Assuming that $T > t_{max}$ is a constant, the function

$$\phi(\mathbf{x}, t) := G(\mathbf{x}, t + T) \tag{4.9.6}$$

is a solution of (4.9.1) in the solution domain $\Omega \times [0, t_{max}]$.

We rearrange the measurement points to be $\{(\mathbf{x}_j, t_j)\}, j = 1, \cdots, m = \sum_{i=1}^{\ell} I_i$. Note that two measurements taken at the same location $\mathbf{x}_i$ but at different times are now treated as two distinct collocation points. The collocation points are renumbered as

$$\{(\mathbf{x}_j, t_j)\}, \quad j = \quad m + 1, \cdots, m + n \quad \text{on } \overline{\Omega} \times \{0\},$$

$$\{(\mathbf{x}_j, t_j)\}, \quad j = \quad m + n + 1, \cdots, m + n + p \quad \text{on } \Gamma_1 \times (0, t_{max}],$$

$$\{(\mathbf{x}_j, t_j)\}, \quad j = \quad m + n + p + 1, \cdots, m + n + p + q \quad \text{on } \Gamma_2 \times (0, t_{max}].$$

Here, n, p, q denote the total number of collocation points for the initial condition (4.9.2), Dirichlet boundary condition (4.9.3), and Neumann boundary condition (4.9.4), respectively. In this way, we treat time as an additional spatial dimension. As a result, the collocation points are now distributed in a $(d + 1)$-dimensional space-time.

Using the MFS, an approximation $\tilde{u}$ to the solution of (4.9.1) under the conditions (4.9.2)-(4.9.4) with the noisy measurements $\hat{h}_j$ can be expressed as follows:

$$\tilde{u}(\mathbf{x},t) = \sum_{j=1}^{n+m+p+q} \hat{\alpha}_j \phi(\mathbf{x}-\mathbf{x}_j, t-t_j),$$

where $\phi(\mathbf{x},t)$ is given by (4.9.6) and $\hat{\alpha}_j$ are unknown coefficients to be determined.

In this way, the procedure for solving homogeneous time–dependent problems is the same as for solving elliptic problems. Since ϕ is the fundamental solution of the heat equation, the approximate solution $\tilde{u}$ automatically satisfies the original heat equation (4.9.1) inside the domain. We need only to fit the boundary conditions. Using the initial condition (4.9.2) and collocating the boundary conditions (4.9.3) and (4.9.4), we then obtain the following system of linear equations:

$$\mathbf{G}\hat{\alpha} = \hat{\mathbf{b}}, \tag{4.9.7}$$

where

$$\mathbf{G} = \begin{pmatrix} \phi(\mathbf{x}_i - \mathbf{x}_j, t_i - t_j) & 1 \le i \le m \\ \phi(\mathbf{x}_i - \mathbf{x}_j, t_i - t_j) & m+1 \le i \le m+n \\ \phi(\mathbf{x}_i - \mathbf{x}_j, t_i - t_j) & m+n+1 \le i \le m+n+p \\ \frac{\partial u}{\partial n}(\mathbf{x}_k - \mathbf{x}_j, t_k - t_j) & m+n+p+1 \le i \le m+n+p+q \end{pmatrix}$$

where the column index j always runs from 1 to $m+n+p+q$ to match the vector

$$\hat{\alpha} = (\hat{\alpha}_1, \hat{\alpha}_2, \ldots, \hat{\alpha}_{n+m+p+q})^T$$

and where

$$\hat{\mathbf{b}} = \begin{pmatrix} \hat{h}_i & 1 \le i \le m \\ \varphi(\mathbf{x}_i, t_i) & m+1 \le i \le m+n \\ f(\mathbf{x}_i, t_i) & m+n+1 \le i \le m+n+p \\ g(\mathbf{x}_i, t_i) & m+n+p+1 \le i \le m+n+p+q \end{pmatrix}.$$

Note that this method, in contrast to many standard methods for inverse problems, does not need a separate solver for the direct problem. It solves both the direct and the inverse problem simultaneously, but it is not recommended for solving the direct problem alone.

The solvability of the system (4.9.7) depends on the non–singularity of the matrix $\mathbf{G}$, which is still an open research problem. Due to the large number of collocation points, the resultant matrix $\mathbf{G}$ is very large, and in addition it is extremely ill–conditioned and thus ill-posed, if the trial centers get dense. To alleviate the extreme ill–conditioning for the inverse problem indicated above, a regularization technique is needed. As we note in Section 2.5, regularization can produce a stable and accurate solution for the unknown parameters $\hat{\alpha}$ of the matrix equation (4.9.7). The singular value decomposition (SVD) usually works well for the direct problem [Ram02] but fails to provide a stable and accurate solution to the system (4.9.7) for the inverse problem. A number of regularization methods have been applied to solving this kind of ill–conditioning problem, and in Section 2.5 we described **Tikhonov regularization**, the **truncated singular value decomposition**, and the **L–curve method**.

After solving (4.9.7) by regularization with a parameter δ along the lines mentioned in Section 2.5, the corresponding approximate solution for the problem (4.9.1)–(4.9.4) with noisy measurement data is then given by

$$\hat{u}^{\delta}(\mathbf{x},t) := \sum_{j=1}^{N} \hat{\alpha}_j^{\delta} \phi(\mathbf{x}-\mathbf{x}_j, t-t_j)$$

for $N = m+n+p+q$. The temperature and heat flux at the surface Γ_3 can also be calculated accordingly.

Example 4.9.8 Let the heat conduction coefficient $a = 1$ and $t_{max} = 1$ in (4.9.1). The noisy data are generated by $\widetilde{h_i} = h_i + \eta\ r(i)$, where h_i is the exact data and $r(i)$ is a random number in $[-1, 1]$. The scalar η indicates the noise level of the measurement data.

To validate the accuracy of the approximate solution, we compute the Root Mean Square Error (RMSE) which is defined by

$$E(u) = \sqrt{\frac{1}{N_t}\sum_{i=1}^{N_t} \left(\hat{u}^{\delta}(\mathbf{z}_i,\tau_i) - u(\mathbf{z}_i,\tau_i)\right)^2},$$

with a total of N_t evaluation points $(\mathbf{z}_i, \tau_i)$ in the domain $\Gamma_3 \times [0,1]$ where $\hat{u}^{\delta}(\mathbf{z}_i,\tau_i)$ and $u(\mathbf{z}_i,\tau_i)$ are the approximate and exact temperature respectively at a evaluation point. The error $E(\partial u/\partial n)$ for the heat flux is defined in a similar way.

The boundary conditions $f(\mathbf{x},t)$, $g(\mathbf{x},t)$ and the initial temperature

$\varphi(\mathbf{x}, t)$ are chosen in such a way that they satisfy the exact solution

$$u^*(x_1, x_2, t) = 2t + \frac{1}{2}(x_1^2 + x_2^2) + e^{-t}(\cos(x_1) + \cos(x_2))$$

of (4.9.1).

Three different domain and boundary configurations are considered:

Case 1:
$\Omega = \{\ (x_1, x_2) \mid 0 < x_1 < 1,\ 0 < x_2 < 1\}$,
$\Gamma_1 = \{\ (x_1, x_2) \mid x_1 = 1,\ 0 < x_2 < 1\}$,
$\Gamma_2 = \{\ (x_1, x_2) \mid 0 < x_1 < 1,\ x_2 = 1\}$,
$\Gamma_3 = \partial\Omega \setminus \{\Gamma_1 \cup \Gamma_2\}$.

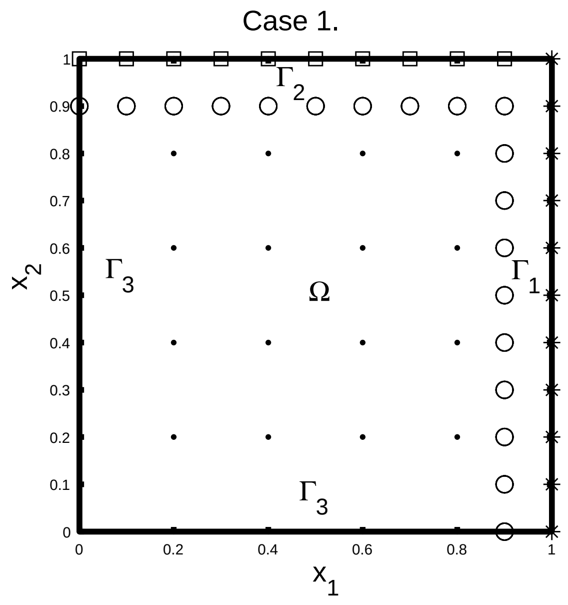

Fig. 4.24. The distribution of measurement and collocation points for the Case 1.

Case 2:
$\Omega = \{\ (x_1, x_2) \mid x_1^2 + x_2^2 < 1\}$,
$\Gamma_1 = \emptyset$,
$\Gamma_2 = \{\ (x_1, x_2) \mid x_1^2 + x_2^2 = 1,\ x_1 > 0\}$,
$\Gamma_3 = \partial\Omega \setminus \{\Gamma_1 \cup \Gamma_2\}$.

Case 3: Ω as in Case 1, $\Gamma_1 = \Gamma_2 = \emptyset$, $\Gamma_3 = \partial\Omega \setminus \{\Gamma_1 \cup \Gamma_2\}$.

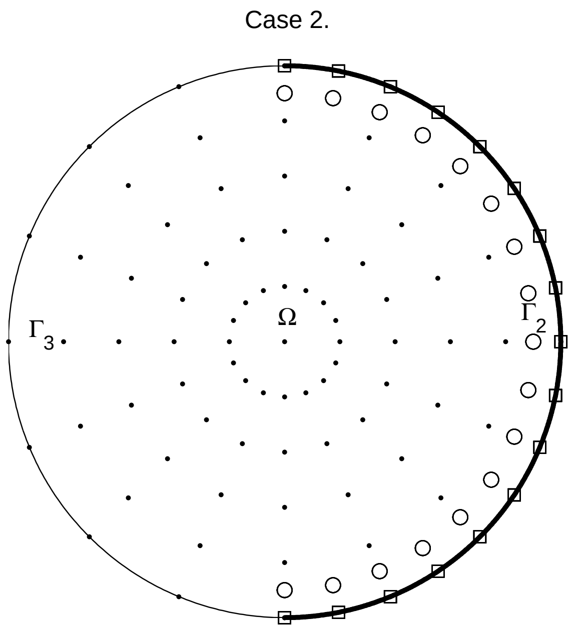

Fig. 4.25. The distribution of measurement and collocation points for the Case 2.

The distributions of the internal measurements and collocation points over Ω for the three cases are shown in Figures 4.24-4.26. In these figures,

$*$ represents a collocation point for Dirichlet data,
$\square$ represents a collocation point for Neumann data,
$\cdot$ represents a collocation point for initial data, and
$\circ$ represents points with a sensor for internal measurement.

Numerical results are obtained by taking the constant $T = 1.8$ in all three cases. In Case 1,

$$n = 36,\ m = 100,\ p = 55,\ q = 50.$$

Hence, the total number of collocation points and testing points are $N = 241$ and $N_t = 882$, respectively. In Case 2,

$$n = 96,\ m = 85,\ p = 0,\ q = 85.$$

which implies $N = 266, N_t = 693$. In Case 3,

$$n = 36,\ m = 30,\ p = 0,\ q = 0,$$

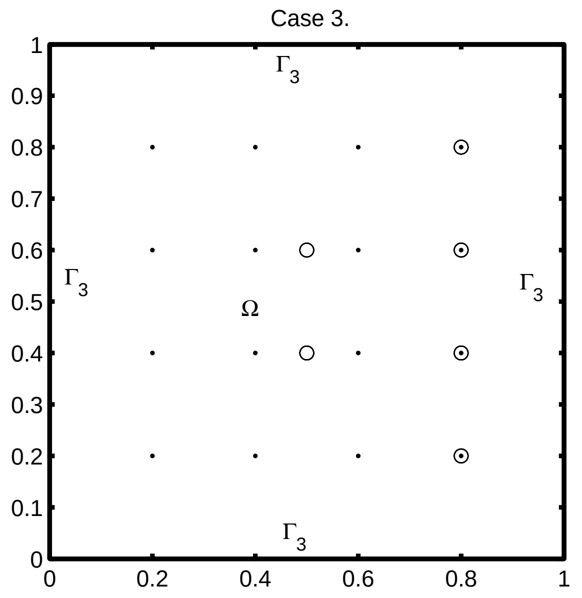

Fig. 4.26. The distribution of measurement and collocation points for the Case 3.

which implies $N = 66$, $N_t = 1764$.

Numerical results by using only the SVD are presented in Table 4.26. The computed root mean-square errors show that even for exact input data the direct method cannot produce an acceptable solution to such extremely ill–conditioned linear systems. In fact, the condition numbers in Case 1 and Case 2 are 8.355×10^{33} and 4.878×10^{34} which are too large to obtain an accurate solution without the use of a regularization technique. This is because inverse problems like this will often be ill-posed even for exact data. For Case 3, the condition number is 4.5×10^{19} which is much smaller than the other two cases and hence the results in Case 3 are better.

The use of regularization techniques, as we shall see, gives a stable and much more accurate solution. Table 4.27 shows the root mean-square errors of the temperature and heat flux in domain $\Gamma_3 \times [0, 1]$ with exact data, using Tikhonov regularization with an L-curve-based choice of the regularization parameter δ. In contrast to Table 4.26, the RMSE errors in Table 4.27 are greatly reduced. We remark that the L-curve method works well for determining the crucial regularization parameter, but it can be further improved if a scaling factor is used.

Numerical results for the example with noisy data (noise level $\eta = 0.01$ in all cases) are shown in Table 4.28. The regularization method with the L-curve technique provides an acceptable approximation to the solution whilst the direct method fails completely. The problem of how to choose an optimal regularization parameter is still an open question.

Note that the setting for Case 3 comes from a real-life problem in a steel company. In fact, the solution for this case may not be unique, but our computational results demonstrate that the proposed method is flexible enough to give a reasonable approximation to the solution under insufficient information.

Case 1	E(u)= 11.2115 E($\partial u/\partial \mathbf{n}$) = 4.6189
Case 2	E(u) = 12.2759 E($\partial u/\partial \mathbf{n}$) = 2.1023
Case 3	E(u) = 0.0104 E($\partial u/\partial \mathbf{n}$) = 0.0556

Table 4.26. *RMSE in domain* $\Gamma_3 \times [0,1]$ *with exact data and no regularization technique.*

Case 1	$\delta = 2.7404E-12$	E(u)= $2.7289E-04$ E($\partial u/\partial \mathbf{n}$) = $6.7662E-04$
Case 2	$\delta = 4.1551E-13$	E(u) = 0.0011 E($\partial u/\partial \mathbf{n}$)= 0.0029
Case 3	$\delta = 1.0370E-8$	E(u) = 0.0013 E($\partial u/\partial \mathbf{n}$) = 0.0062

Table 4.27. *RMSE in domain* $\Gamma_3 \times [0,1]$ *with exact data by using Tikhonov regularization with L-curve parameter* δ*.*

The relationship between the RMSE and the value of the constant T in Case 1 with noisy data ($\eta = 0.01$) is displayed in Figure 4.27. The regularization parameter δ is obtained again by using the L-curve

Case 1	$\delta = 6.9833E-5$	E(u)= 0.0266
		E($\partial u/\partial \mathbf{n}$) = 0.0538
Case 2	$\delta = 0.003$	E(u)= 0.2456
		E($\partial u/\partial \mathbf{n}$) = 0.4211
Case 3	$\delta = 6.7772E-7$	E(u)= 0.0290
		E($\partial u/\partial \mathbf{n}$) = 0.0608

Table 4.28. *RMSE in domain* $\Gamma_3 \times [0,1]$ *with noisy data (*$\eta = 0.01$*) by Tikhonov regularization with L-curve parameter* δ

method. The numerical results indicate that the choice of the parameter T plays an important role for obtaining an accurate approximation.

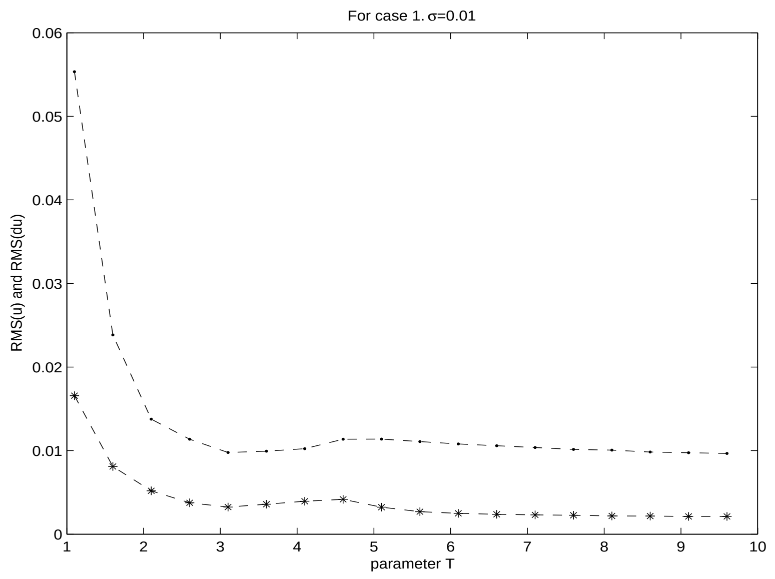

Fig. 4.27. RMSE of temperature and heat flux on $\Gamma_3 \times [0,1]$ with respect to the parameter T.

□

Example 4.9.9 Let

$$\begin{aligned}
\Omega &= \{(x_1,x_2,x_3) \,:\, 0<x_i<1,\ i=1,2,3\}\\
\Gamma_1 &= \{(x_1,x_2,x_3) \,:\, 0<x_1<1,\ 0<x_2<1,\ x_3=1\}\\
\Gamma_2 &= \{(x_1,x_2,x_3) \,:\, 0<x_1<1,\ 0<x_2<1,\ x_3=0\}\\
\Gamma_3 &= \partial\Omega \setminus (\Gamma_1 \cup \Gamma_2).
\end{aligned}$$

The locations of the measurement points and the collocation points in the domain Ω are shown in Figure 4.28 where

$*$ represents a measurement point with Dirichlet data,
$\square$ represents a measurement point with Neumann data, and
$\circ$ represents a point with sensor data.

In this computation, we set

$$n=245,\ m=250,\ p=180,\ q=180,\ N=855,\ N_t=5324.$$

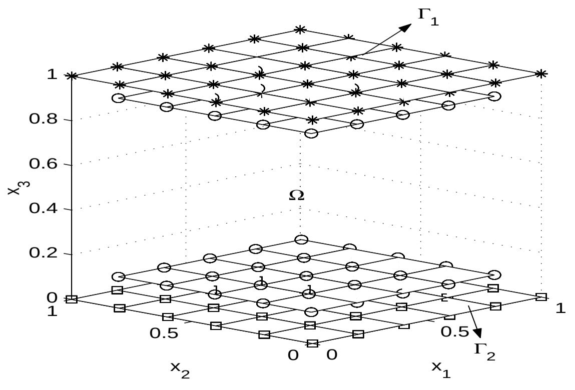

Fig. 4.28. Locations of measurement and collocation points.

The exact solution is given by

$$u^*(x_1,x_2,x_3,t) = e^{(-4t)}(\cos(2x_1)+\cos(2x_2)+\cos(2x_3)).$$

In the computation, the value of the parameter T is 2.3 and the noise level is set to be $\eta = 0.01$. The errors between the exact solution and the approximate solution for the temperature and heat flux on boundary Γ_3 at time $t=1$ are shown in Figure 4.29 and Figure 4.30, respectively. Note that the L^2 norm of u and $\partial u/\partial n$ over $\Gamma_3 \times [0,1]$ are about 0.62 and 0.52 respectively. Their relative errors are approximately twice the

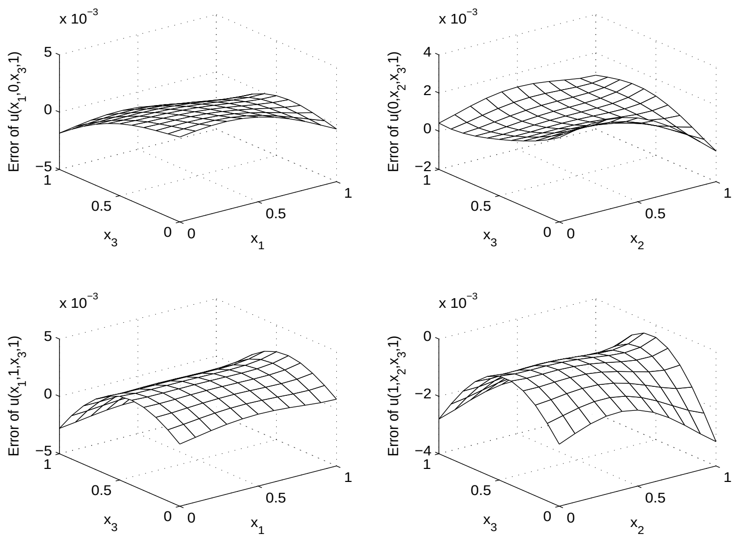

Fig. 4.29. Surface plots of temperature errors on boundary Γ_3

values given in Figures 4.29 and 4.30. These small relative errors show that the proposed scheme is effective for solving the three-dimensional inverse heat conduction problem. □

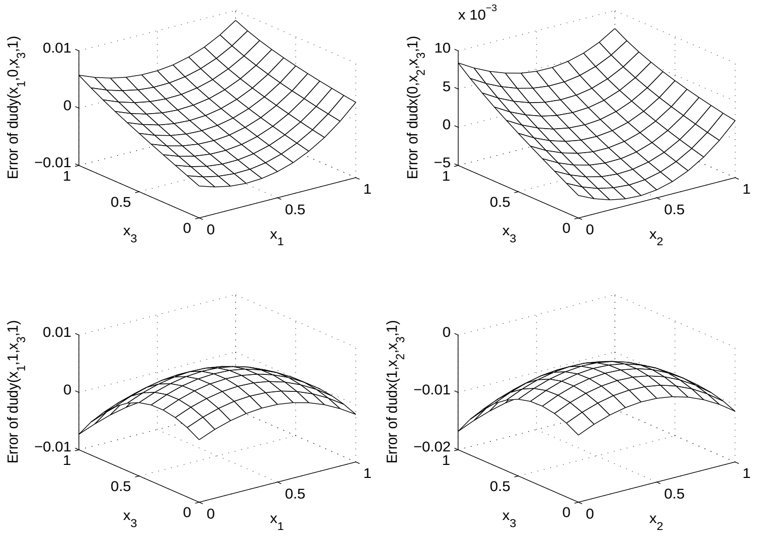

Fig. 4.30. Surface plots of heat flux errors on boundary Γ_3

5
The Method of Particular Solutions

5.1 Introduction

In the last chapter, we considered only linear homogeneous equations having fundamental solutions. In this chapter we extend our discussion to linear inhomogeneous problems

$$\begin{array}{rcll} Lu &=& f^{\Omega} & \text{in } \Omega \subset I\!R^d \\ u &=& f^{D} & \text{in } \Gamma^D \subseteq \Gamma := \partial\Omega \\ \dfrac{\partial u}{\partial n} &=& f^{N} & \text{in } \Gamma^N \subset \Gamma = \Gamma^D \cup \Gamma^N,\ \Gamma^D \cap \Gamma^N = \emptyset \end{array} \tag{5.1.1}$$

with Dirichlet boundary conditions on Γ^D and Neumann boundary conditions on Γ^N. Here, $f^{\Omega},\ f^D, f^N$ are given functions on $\Omega,\ \Gamma^D,\ \Gamma^N$, respectively. We shall later replace the right-hand side $f^{\Omega}\ :\ \Omega \to I\!R$ by $f^{\Omega}(\mathbf{x}, u, \nabla u, ...)$ to treat more general quasi-linear or nonlinear problems.

In the **Boundary Element Method**, it is well-known that (5.1.1) can be reformulated as an integral equation [PBW92]

$$\begin{array}{rcl} c(\mathbf{x})u(\mathbf{x}) &=& \displaystyle\int_{\Gamma} \left(u(\mathbf{y}) \frac{\partial G(\mathbf{x}, \cdot)}{\partial n}(\mathbf{y}) - \frac{\partial u}{\partial n}(\mathbf{y}) G(\mathbf{x}, \mathbf{y}) \right) \mathrm{d}\mathbf{y} \\ && + \displaystyle\int_{\Omega} G(\mathbf{x}, \mathbf{y}) f^{\Omega}(\mathbf{y}) \mathrm{d}\mathbf{y} \end{array} \tag{5.1.2}$$

where $\mathbf{x} \in \Omega \cup \Gamma$ and $c(\mathbf{x})$ is a geometric factor depending on the location of $\mathbf{x}$, i.e.

$$c(\mathbf{x}) = \begin{cases} 1, & \mathbf{x} \text{ in the interior,} \\ \dfrac{1}{2}, & \mathbf{x} \text{ on a smooth boundary,} \\ \dfrac{\theta}{2\pi}, & \mathbf{x} \text{ at a corner with angle } \theta, \end{cases}$$

and $G(\mathbf{x}, \mathbf{y})$ is the fundamental solution of the differential operator L.

In (5.1.2), a domain integration is required. Due to this, the **Boundary Element Method** (BEM) loses its attractiveness of having a boundary integration only. During the past two decades, considerable effort has been devoted to the problem of either efficiently computing the domain integrals or transferring them to the boundary. Even though the domain integrals usually do not introduce any new unknowns in the numerical implementation, the singularity in the integrand and the possibly irregular shape of the domain make it difficult to evaluate the integral with efficient numerical schemes, especially in the 3D case. In fact, the accurate evaluation of these integrals becomes the dominating computational task in the whole numerical process for solving partial differential equations.

In this chapter we will briefly introduce a number of approaches to alleviate the difficulties of domain integration in irregular domains. Later, we shall focus on how to eliminate these domain integrations by using radial basis functions.

5.2 The Dual Reciprocity Method (DRM)

A **particular solution** of a boundary-value problem (5.1.1) is defined as a function u_p on $\Omega \cup \Gamma$ which satisfies the inhomogeneous equation

$$Lu_p = f^{\Omega} \text{ in } \Omega \tag{5.2.1}$$

but does not necessarily satisfy the boundary conditions of (5.1.1). Notice that the particular solution is not unique.

Suppose a particular solution u_p of (5.2.1) is known. Then the integral equation (5.1.2) can be rewritten without any domain integral. To see this, let $I(\mathbf{x})$ be the domain integral term in (5.1.2). Then

$$I(\mathbf{x}) = \int_{\Omega} G(\mathbf{x}, \mathbf{y}) f^{\Omega}(\mathbf{y}) \mathrm{d}\mathbf{y} = \int_{\Omega} G(\mathbf{x}, \mathbf{y}) Lu_p(\mathbf{y}) \mathrm{d}\mathbf{y}. \tag{5.2.2}$$

Using Green's Theorem, we have

$$\begin{aligned} & \int_{\Omega} \left(G(\mathbf{x}, \mathbf{y}) Lu_p(\mathbf{y}) - u_p(\mathbf{y}) LG(\mathbf{x}, \mathbf{y}) \right) \mathrm{d}\mathbf{y} \\ = & \int_{\Gamma} \left(G(\mathbf{x}, \mathbf{y}) \frac{\partial u_p}{\partial n}(\mathbf{y}) - u_p(\mathbf{y}) \frac{\partial G(\mathbf{x}, \cdot)}{\partial n}(\mathbf{y}) \right) \mathrm{d}\mathbf{y}, \end{aligned}$$

and thus get

$$\begin{aligned}
I(\mathbf{x}) &= \int_\Omega G(\mathbf{x},\mathbf{y})Lu_p(\mathbf{y})\mathrm{d}\mathbf{y} \\
&= \int_\Omega u_p(\mathbf{y})(LG(\mathbf{x},\cdot))(\mathbf{y})\mathrm{d}\mathbf{y} \\
&\quad + \int_\Gamma \left(G(\mathbf{x},\mathbf{y})\frac{\partial u_p}{\partial n}(\mathbf{y}) - u_p(\mathbf{y})\frac{\partial G(\mathbf{x},\cdot)}{\partial n}(\mathbf{y}) \right) \mathrm{d}\mathbf{y}.
\end{aligned} \tag{5.2.3}$$

We note that fundamental solutions satisfy $LG(\mathbf{x},\cdot) = \delta_\mathbf{x}$. Furthermore, the domain integral term in the right hand side of (5.2.3) is equal to $c(\mathbf{x})u_p(\mathbf{x})$. From (5.2.2) and (5.2.3), we finally get

$$I(\mathbf{x}) = c(\mathbf{x})u_p(\mathbf{x}) + \int_\Gamma \left(G(\mathbf{x},\mathbf{y})\frac{\partial u_p}{\partial n}(\mathbf{y}) - u_p(\mathbf{y})\frac{\partial G(\mathbf{x},\cdot)}{\partial n}(\mathbf{y}) \right) \mathrm{d}\mathbf{y},$$

which contains no domain integral any more. In this way the boundary element method can keep its advantage of having boundary integrations only.

The above procedure to transfer the domain integral to the boundary can be successful only if an explicit particular solution u_p is available. However, the task of deriving an explicit particular solution is nontrivial and often impossible when f^Ω becomes complicated. In Section 5.5, we will introduce a numerical scheme to approximate the particular solution. Coupling Boundary Element Methods with such a scheme is known as the **Dual Reciprocity Method** (DRM). We refer the readers to the book [PBW92] for further details on the DRM.

5.3 The Method of Particular Solutions (MPS)

Another approach for solving the inhomogeneous problem (5.1.1) is to split the solution u into a particular solution u_p and its associated homogeneous solution u_h as

$$u = u_h + u_p. \tag{5.3.1}$$

From (5.1.1) and (5.2.1) we get

$$Lu_h = Lu - Lu_p = f^\Omega - f^\Omega = 0.$$

Thus we have the homogeneous problem

$$\begin{aligned}
Lu_h &= 0 && \text{in } \Omega \\
u_h &= f^D - u_p && \text{in } \Gamma^D \\
\frac{\partial u_h}{\partial n} &= f^N - \frac{\partial u_p}{\partial n} && \text{in } \Gamma^N.
\end{aligned} \tag{5.3.2}$$

Once u_p is known, the influence of the solution by the inhomogeneous term has in fact been transferred to the boundary and the homogeneous solution u_h in (5.3.2) can be calculated by standard boundary methods such as the **Boundary Element Method**, the **Method of Fundamental Solutions**, the **Boundary Integral Method**, etc. For simplicity and the advantages we have discussed in the previous chapter, the **Method of Fundamental Solutions** will be employed throughout this chapter to find the homogeneous solution u_h. The final solution u can be recovered by adding u_h and u_p as in (5.3.1). The key step of the **Method of Particular Solutions** is to obtain an explicit particular solution or, at least, a good explicit approximation to it. In the next section we will focus on the general idea of the approximation of particular solutions via radial basis functions.

5.4 Approximate Particular Solutions

To construct a particular solution for the boundary-value problem (5.1.1) approximately, one can take particular solutions of other boundary-value problems

$$Lu_k = f_k,\ 1 \leq k \leq n \tag{5.4.1}$$

with the same linear differential operator, and approximate f^Ω by a superposition

$$f^\Omega \simeq \sum_{k=1}^{n} \alpha_k f_k \tag{5.4.2}$$

of the functions f_k to some accuracy. Then the function

$$\tilde{u}_p := \sum_{k=1}^{n} \alpha_k u_k$$

satisfies

$$L\tilde{u}_p = \sum_{k=1}^{n} \alpha_k L u_k = \sum_{k=1}^{n} \alpha_k f_k \simeq f^\Omega$$

and thus is an **approximate particular solution** for the boundary-value problem (5.1.1). This reduces the construction of approximate particular solutions of boundary-value problems to a standard approximation problem like those considered in Chapter 2.

Furthermore, the Method of Particular Solutions combined with the Method of Fundamental Solutions leads to reliable results, because there will be explicitly available functions $\tilde{u}_p$ and $\tilde{u}_h$ such that in the situation of (5.2.1) we have small residuals

$$\begin{array}{rcllll} L\tilde{u}_p & - & f^\Omega & =: & r^\Omega & \text{small in } \Omega \\ \tilde{u}_h & - & (f^D - \tilde{u}_p) & =: & r^D & \text{small in } \Gamma^D \\ \dfrac{\partial \tilde{u}_h}{\partial n} & - & (f^N - \dfrac{\partial \tilde{u}_p}{\partial n}) & =: & r^N & \text{small in } \Gamma^N. \end{array}$$

If there is continuous dependence of the exact solution u on the data in the sense that for each other approximate solution $\tilde{u}$ the inequality

$$\|u - \tilde{u}\| \le c_\Omega \|Lu - L\tilde{u}\|_\Omega + c_{\Gamma^D} \|u - \tilde{u}\|_{\Gamma^D} + c_{\Gamma^N} \|\frac{\partial u}{\partial n} - \frac{\partial \tilde{u}}{\partial n}\|_{\Gamma^N}$$

holds in suitable norms, we have a final small error

$$\begin{array}{rcl} \|u - \tilde{u}_p - \tilde{u}_h\| & \le & c_\Omega \|Lu - L\tilde{u}_p - L\tilde{u}_h\|_\Omega \\ & & +c_{\Gamma^D} \|u - \tilde{u}_p - \tilde{u}_h\|_{\Gamma^D} \\ & & +c_{\Gamma^N} \|\dfrac{\partial u}{\partial n} - \dfrac{\partial \tilde{u}_p}{\partial n} - \dfrac{\partial \tilde{u}_h}{\partial n}\|_{\Gamma^N} \\ & = & c_\Omega \|f^\Omega - L\tilde{u}_p\|_\Omega \\ & & +c_{\Gamma^D} \|f^D - \tilde{u}_p - \tilde{u}_h\|_{\Gamma^D} \\ & & +c_{\Gamma^N} \|f^N - \dfrac{\partial \tilde{u}_p}{\partial n} - \dfrac{\partial \tilde{u}_h}{\partial n}\|_{\Gamma^N} \\ & = & c_\Omega \|r^\Omega\|_\Omega + c_{\Gamma^D} \|r^D\|_{\Gamma^D} + c_{\Gamma^N} \|r^N\|_{\Gamma^N}. \end{array}$$

This is a good theoretical *a-posteriori* justification for the method, but in practice the constants c_Ω, c_{Γ^D}, c_{Γ^N} are only rarely known explicitly.

For the particular solutions arising in (5.4.1) one can use any pair of functions u, f with $Lu = f$, and each such pair can be called a **particular solution** for the differential operator L. A simple and easy technique to get approximate particular solutions is to take some arbitrary smooth and explicit u and to evaluate $f := Lu$ explicitly in a straightforward way. We shall use this **direct** technique below in some cases, but it has the disadvantage that it is not always guaranteed that the functions f_k obtained this way are able to recover f^Ω up to high accuracy within a well-developed theory of multivariate approximation. This limits the applicability of the direct technique somewhat, since we require (5.4.2) to work properly. But in many cases one can start with functions f_k which provide useful approximations, and then one has to solve for u_k with $Lu_k = f_k$. We call this the **indirect** technique.

Since Chapter 2 provides a well-established theory for approximation by radial basis trial functions, this chapter focuses on the indirect technique to derive particular solutions u such that $f = Lu$ is a translate $\phi(\|\cdot - \mathbf{y}\|_2)$ of a radial basis function ϕ with respect to a trial point $\mathbf{y} \in I\!R^d$. If n such trial centers $\{\mathbf{y}_k\}_{k=1}^n$ are chosen, the inhomogeneous term f^Ω is approximated as in (5.4.2) via a finite combination of translates of radial basis functions as

$$f^\Omega(\mathbf{x}) \simeq \tilde{f}(\mathbf{x}) := \sum_{k=1}^{n} \alpha_k \phi(\|\mathbf{x} - \mathbf{y}_k\|_2) \tag{5.4.3}$$

where $\{\alpha_k\}_{k=1}^n$ is a vector of coefficients to be determined. If we do the approximation in (5.4.3) by interpolation, and take the trial centers $\{\mathbf{y}_k\}_{k=1}^n$ also as test points, we have to satisfy the conditions

$$f^\Omega(\mathbf{y}_j) = \tilde{f}(\mathbf{y}_j), \quad 1 \le j \le n,$$

which lead to the linear system

$$f^\Omega(\mathbf{y}_j) = \sum_{k=1}^{n} \alpha_k \phi(\|\mathbf{y}_j - \mathbf{y}_k\|_2), \quad 1 \le j \le n, \tag{5.4.4}$$

which is uniquely solvable if the symmetric $n \times n$ matrix

$$\mathbf{A} = \begin{pmatrix} \phi(\|\mathbf{y}_1 - \mathbf{y}_1\|_2) & \cdots & \phi(\|\mathbf{y}_1 - \mathbf{y}_n\|_2) \\ \vdots & \ddots & \vdots \\ \phi(\|\mathbf{y}_n - \mathbf{y}_1\|_2) & \cdots & \phi(\|\mathbf{y}_n - \mathbf{y}_n\|_2) \end{pmatrix}$$

is nonsingular. If ϕ is a positive definite radial basis function, Definition 1.2.4 in Chapter 1 even guarantees that the matrix is positive definite which implies that the matrix is nonsingular.

Therefore we need to find particular solutions u_k such that

$$Lu_k(\mathbf{x}) = \phi(\|\mathbf{x} - \mathbf{y}_k\|_2).$$

If L is a radially invariant operator like the Laplace or Helmholtz operator, we get particular solutions that itself are radial basis functions

$$u_k(\mathbf{x}) = \psi(\|\mathbf{x} - \mathbf{y}_k\|_2).$$

In this case, the indirect method also works, provided that the differential operator is elliptic, e.g. for the negative Laplace operator or the modified Helmholtz equation. Radiality will not be conserved for general operators, but the overall logic of the Dual Reciprocity Method stays the same. Since this book focuses on radial basis functions, we shall also focus on radially invariant linear differential operators and radial particular solutions.

Once $\tilde{f}$ in (5.4.3) is established, an approximate particular solution $\tilde{u}_p$ to a particular solution u_p of (5.2.1) can be written as

$$\tilde{u}_p(\mathbf{x}) = \sum_{k=1}^{n} \alpha_k \psi\left(\|\mathbf{x} - \mathbf{y}_k\|_2\right), \quad \mathbf{x} \in \Omega, \tag{5.4.5}$$

where ψ satisfies

$$L\psi(\|\mathbf{x} - \mathbf{y}_k\|_2) = \phi(\|\mathbf{x} - \mathbf{y}_k\|_2), \quad \mathbf{x} \in \Omega, \quad 1 \le k \le n. \tag{5.4.6}$$

The normal derivative of a particular solution (5.4.5) is

$$\frac{\partial \tilde{u}_p}{\partial n}(\mathbf{x}) = \sum_{k=1}^{n} \alpha_k \frac{\partial}{\partial n} \psi(\|\mathbf{x} - \mathbf{y}_k\|_2), \quad \mathbf{x} \in \Omega. \tag{5.4.7}$$

The above approximation of the particular solution requires knowing ψ in (5.4.6) in closed form. Since ϕ is radially symmetric, the analytical solvability of (5.4.6) for differential operators L that are radially and translationally invariant can be expected. For these operators such as the Laplace operator Δ and the bi-harmonic operator Δ^2, a variety of closed-form radial particular solutions ψ are available.

Consequently, an appropriate choice of a radial basis function is of considerable interest. In the past, the function $\phi(r) = 1+r$ was used almost exclusively in the literature on Boundary Element Methods [PBW92]. Unfortunately, there is no theoretical basis to justify the ad-hoc choice of $1 + r$. However the introduction of a positive definite radial basis functions gives a firm theoretical justification of the Dual Reciprocity Method.

Moreover, the traditional approach of the Dual Reciprocity Method coupled the boundary values and the domain term [PBW92] and thus placed a substantial restriction on the choice of the interpolation points and basis functions in (5.4.3). One of the advantages of using the Method of Particular Solutions is that it decouples the original differential equation into two parts as shown in the previous section. Since the particular solution is independent of the geometric shape of the solution domain, this allows us to use a domain-embedding approach to calculate an approximate particular solution. As shown in Figure 5.1, the interpolation points can reside inside and outside the solution domain and can be scattered points or regular grid points in a box containing any irregular domain Ω. The simplicity and flexibility of choosing interpolation points in such a way are desirable when preparing the data input.

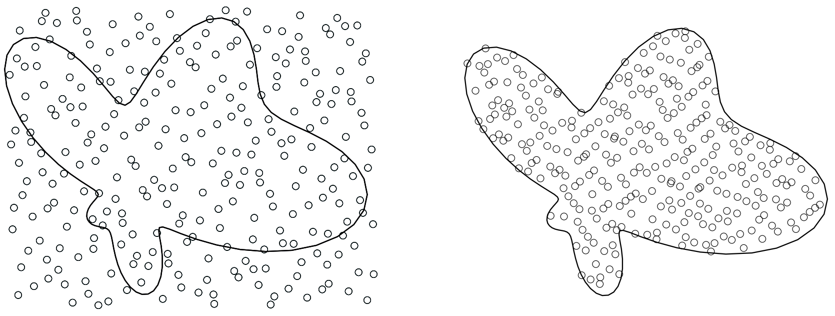

Fig. 5.1. Domain embedding method using scattering points (left) or irregular grid points inside the domain (right).

In the past, due to the difficulty of obtaining closed-form particular solutions, the Dual Reciprocity Method was restricted mostly to Poisson-type equations [PBW92]. Other types of differential equations were treated as a generalized Poisson equation by keeping the Laplace operator operator on the left-hand-side and moving the rest to the right hand side, i.e.

$$\Delta u = f(\mathbf{x}, u, \nabla u, \frac{\partial u}{\partial t}, ...).$$

The right-hand-side of the equation is treated as a known quantity by means of iterative updating in nonlinear problems, or by time-stepping in time–dependent problems. This is possible because RBF methods provide smooth approximate solutions which can be evaluated or differentiated everywhere.

In recent years, significant progress in deriving closed–form particular solutions for other differential operators has been reported [CR98, MGC99]. As we shall see in the later sections, significant improvement in efficiency and accuracy of numerical computations has been achieved this way.

5.5 Particular Solutions for the Laplace operator

To obtain ψ in (5.4.6) for $L = \Delta$, we notice that Δ is both translationally and rotationally invariant. In polar coordinates, we have

$$\Delta\psi = \begin{cases} \dfrac{1}{r}\dfrac{d}{dr}\left(r\dfrac{d\psi}{dr}\right), & \text{in } 2D, \\ \dfrac{1}{r^2}\dfrac{d}{dr}\left(r^2\dfrac{d\psi}{dr}\right), & \text{in } 3D. \end{cases}$$

Thus one can easily determine particular solutions by the direct method, while the indirect method requires solving a second-order ordinary differential equation.

All cases allow a simple scaling technique. If we scale the radial basis function ϕ by a positive scale factor c as $\phi_c(r) := \phi(r/c)$ as we did in Section 2.6 for various numerical reasons, the function

$$\Delta(c^2\psi(r/c)) = \phi(r/c) \quad \text{or} \quad \Delta(c^2\psi_c(r)) = \phi_c(r) \qquad (5.5.1)$$

will solve the scaled problem. Thus we only need to focus on unscaled particular solutions here, but note that this argument works only for homogeneous differential operators in this straightforward way.

We first treat the direct case, which goes back to Schclar [Sch94]. It can be proven by Fourier transform arguments that any linear elliptic and radially symmetric operator transforms smooth positive definite radial basis functions into positive definite radial basis functions. Thus if we start with a smooth positive definite radial basis function ψ and take $-\phi := -\Delta\psi$, the function $-\phi$ will be positive definite and radial again, because $-\Delta$ is elliptic and radially symmetric. A more careful analysis reveals that if ψ is a conditionally positive definite radial basis function of positive order Q from Table 1.3, then $-\Delta\psi$ will be conditionally positive definite of order $Q-1$. This technique generates plenty of new radial basis functions, if $(-\Delta)^m$ is applied to the ones we gave in Table 1.3. Note, however, that polyharmonic splines will go into polyharmonic splines again. Furthermore, the procedure comes to an end when generating singular radial basis functions after application of $-\Delta$.

In this way, it is mathematically justified to choose ψ as the particular solution corresponding to $\phi := \Delta\psi$ which is used to interpolate the right–hand side of the given differential equation. In this way, not only the invertibility of the interpolation matrix is guaranteed, but also the particular solution can be obtained cheaply.

For the inverse multiquadric

$$\psi = \frac{1}{\sqrt{r^2 + c^2}}$$

the direct technique yields

$$\phi = \Delta\psi = \begin{cases} \dfrac{r^2 - 2c^2}{(r^2 + c^2)^{5/2}}, & \text{in 2D,} \\ \dfrac{-3c^2}{(r^2 + c^2)^{5/2}}, & \text{in 3D.} \end{cases}$$

For the unscaled compactly supported Wendland–type RBF

$$\psi = (1 - r)_+^6 \left(35r^2 + 18r + 3\right), \tag{5.5.2}$$

the new interpolation function is given by

$$\phi = \Delta\psi = \begin{cases} 112\,(1 - r)_+^4 \left(20r^2 - 4r - 1\right), & \text{in 2D,} \\ 168\,(1 - r)_+^4 \left(15r^2 - 4r - 1\right), & \text{in 3D.} \end{cases} \tag{5.5.3}$$

In this approach, the smoothness of the new radial basis function ϕ is two orders lower than ψ. Hence we have to choose sufficiently smooth functions for ψ if we want to end up with a certain smoothness for ϕ. Note that since the inverse multiquadric is C^∞, the smoothness of ϕ will not be affected. In this way, a wide class of new radial basis functions can be generated.

Example 5.5.4 As an application of the direct method, we consider the Poisson problem

$$\begin{aligned} \Delta u &= 2e^{x-y}, & (x, y) \in \Omega, \\ u &= e^{x-y} + e^x \cos y, & (x, y) \in \Gamma, \end{aligned}$$

where $\Omega \cup \Gamma$ is the unit square. The exact solution is given by $u^* = e^{x-y} + e^x \cos y$.

In this example, we chose ψ in (5.5.2) as the basis function for the particular solution and ϕ in (5.5.3) as the basis for the interpolation function. A special feature of this approach is that the particular solution is compactly supported, unlike the results of the indirect technique to be described below. The L_∞ norms for various scaling factors can be found in Table 5.1 which shows little difference in L_∞ when c becomes large. To perform the numerical computations, we used 81 evenly distributed points on a regular grid as the interpolation points in $[0, 1]^2$. The L_∞ errors of u were computed on a 20×20 uniform grid in the domain. The numerical tests were performed using MATLAB which has a built–in sparse matrix solver. The method of fundamental solutions was employed to find the homogeneous solution with 20 uniformly distributed collocation points on the boundary Γ and 20 uniformly distributed source points on a circle with radius 10 and center $(0, 0)$. Figure 5.2 shows the absolute error using $c = 0.7$.

c	L_∞	c	L_∞
0.3	5.77E-2	0.9	2.47E-4
0.4	2.68E-2	1.0	1.75E-4
0.5	9.10E-3	1.1	2.32E-4
0.6	3.52E-3	1.2	1.28E-4
0.7	1.51E-3	1.3	1.40E-4
0.8	5.68E-4	1.4	1.57E-4

Table 5.1. L_∞ *norm error using the direct approach*

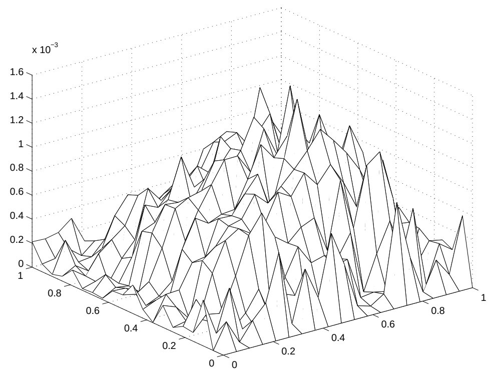

Fig. 5.2. The profile of absolute error using $c = 0.7$.

$\square$

We now turn to the **indirect** technique. For a given radial basis function ϕ, a radial basis function ψ with $\Delta\psi = \phi$ can be obtained by repeated integration. With modern symbolic software such as Maple© or Mathematica©, it is convenient to obtain ψ explicitly. However, since the integration constants are normally omitted by the symbolic software, one needs to treat them with care.

It should be noted that $-\psi$ is a positive definite radial basis function whenever ϕ is positive definite. Thus the methods of this section generate new classes of useful radial basis functions.

For simple f^{Ω} such as polynomials or finite trigonometric sums, particular solutions may be obtained explicitly by the method of undetermined coefficients.

Example 5.5.5 Assume that $f_n(x_1, x_2)$ is a homogeneous polynomial of degree n, i.e.

$$f_n = \sum_{k=0}^{n} \alpha_k x_1^{n-k} x_2^k.$$

Then a particular solution of

$$\Delta u = f_n$$

may be of the form

$$u_p = \sum_{k=0}^{n+2} \beta_k x_1^{n-k+2} x_2^k$$

that is a polynomial of degree $n+2$. With given $\{\alpha_k\}_{k=0}^{n}$, the unknowns $\{\beta_k\}_{k=0}^{n+2}$ can be determined by direct substitution. A detailed derivation can be found in [CMG99]. A particular choice of $\{\beta_k\}_{k=0}^{n+2}$ is given by $\beta_{n+1} = 0, \beta_{n+2} = 0$ and

$$\beta_k = \sum_{m=0}^{\lfloor \frac{n-k}{2} \rfloor} \frac{(-1)^m (k+2m)!(n-k-2m)!}{k!(n-k+2)!} \alpha_{k+2m}, \quad 1 \le k \le n$$

where $\lfloor (n-k)/2 \rfloor$ denotes the integer M satisfying $M \le (n-k)/2 < M+1$. For instance, for

$$f_n = \frac{1}{12} \left(x_1^4 - 4x_1^3 x_2 + 6x_1^2 x_2^2 - 4x_1 x_2^3 + x_2^4 \right),$$

a particular solution is given by

$$u_p = \frac{x_1^6}{360} - \frac{x_1^3 x_2^3}{18} + \frac{x_1^2 x_2^4}{24}.$$

□

For general $f(\mathbf{x})$, the derivation of closed–form particular solutions is not trivial. The classical method for obtaining a particular solution of

(5.2.1) is to construct the associated Newton potential which is given by

$$u_p(\mathbf{x}) = \frac{1}{2\pi} \int_{\Omega} G(\mathbf{x}, \mathbf{y}) f(\mathbf{y}) d\Omega, \quad \mathbf{x} \in \Omega \tag{5.5.6}$$

where G is the fundamental solution as used in Chapter 4. In fact, u_p in (5.5.6) is equivalent to the domain integral term in (5.1.2). As observed by Atkinson [Atk85], if f can be extended smoothly to $\tilde{\Omega} \supset \Omega$ with the same order of differentiability as it had on Ω, then

$$u_p(\mathbf{x}) = \frac{1}{2\pi} \int_{\tilde{\Omega}} G(\mathbf{x}, \mathbf{y}) f(\mathbf{y}) d\Omega, \quad \mathbf{x} \in \Omega, \tag{5.5.7}$$

is also a particular solution of (5.2.1). As a result, u_p in (5.5.7) can be numerically approximated on a larger regular domain $\tilde{\Omega}$ instead of an irregular domain Ω. In general, $\tilde{\Omega}$ can be chosen as an ellipse in $\mathbb{R}^2$ and an ellipsoid in $\mathbb{R}^3$. This offers a general approach as long as the fundamental solution G of L is available. In his paper, Atkinson [Atk85] provided three different numerical methods to approximate particular solutions.

5.5.1 Globally Supported Radial Basis Functions

For thin plate splines $\phi(r) = r^2 \ln r$, by direct integration of

$$\frac{1}{r}\frac{d}{dr}\left(r\frac{d\psi}{dr}\right) = r^2 \ln r$$

we can easily obtain

$$\psi(r) = \frac{r^4 \ln r}{16} - \frac{r^4}{32} + C_1 \ln r + C_2. \tag{5.5.8}$$

Through this subsection, C_1 and C_2 denote the integration constants. Since the first two terms of ψ in (5.5.8) are non–singular at $r = 0$, we can choose $C_1 = C_2 = 0$. Consequently, we have

$$\psi(r) = \frac{r^4 \ln r}{16} - \frac{r^4}{32}. \tag{5.5.9}$$

Beware that if we use symbolic software to do the integration, the term $C_1 \ln r + C_2$ in (5.5.8) is not displayed. As we shall see, these terms play an important role canceling the singularity of $\psi(r)$ if there is any.

For polyharmonic splines $\phi(r) = r^{2n} \ln r$, an analytic expression of $\psi(r)$ is given by

$$\psi(r) = \frac{r^{2n+2}}{4(n+1)^2} \ln r - \frac{r^{2n+2}}{4(n+1)^3} \tag{5.5.10}$$

For the implementation of Neumann boundary condition, we need to evaluate the following in $\mathbf{R}^2$:

$$\frac{\partial\psi(r)}{\partial\mathbf{n}} = \frac{\partial\psi(r)}{\partial x}n_x + \frac{\partial\psi(r)}{\partial y}n_y = \frac{1}{r}\frac{\partial\psi(r)}{\partial r}\left(x\, n_x + y\, n_y\right), \tag{5.5.11}$$

where $\mathbf{n} = (n_x, n_y)$ is an outward normal unit vector. As a result, we also need to derive $1/r(\partial\psi(r)/\partial r)$ after we obtain $\psi(r)$.

From (5.5.10), we have the following

$$\frac{1}{r}\frac{\partial\psi(r)}{\partial r} = \frac{r^{2n}\ln r}{2(n+1)} - \frac{r^{2n}}{4(n+1)^2}. \tag{5.5.12}$$

For multiquadrics $\phi(r) = \sqrt{r^2+c^2}$ in 2D, by direct integration using Mathematica© we obtain

$$\psi(r) = \frac{\left(4c^2+r^2\right)}{9}\sqrt{r^2+c^2} - \frac{c^3}{3}\ln\left(\frac{2}{c^3 r} + \frac{2\sqrt{r^2+c^2}}{c^4 r}\right) + C_1\ln r + C_2. \tag{5.5.13}$$

We observe that $\psi(r)$ is singular at $r=0$. In this case, the integration constants in $C_1\ln r + C_2$ are crucial. One can choose C_1 and C_2 so that the singularity at $r=0$ is canceled. This can be achieved by rewriting the second term of ψ in (5.5.13) as

$$-\frac{c^3}{3}\ln 2 - \frac{c^3}{3}\ln(c+\sqrt{r^2+c^2}) + \frac{c^3}{3}\ln c^4 + \frac{c^3}{3}\ln r.$$

To cancel the singularity at $r=0$, it is clear that we can choose

$$C_1 = -\frac{c^3}{3}, \quad C_2 = \frac{c^3}{3}\ln 2 - \frac{c^3}{3}\ln c^4.$$

Consequently, in the 2D case, we have

$$\psi(r) = \frac{1}{9}\left(4c^2+r^2\right)\sqrt{r^2+c^2} - \frac{c^3}{3}\ln\left(c+\sqrt{r^2+c^2}\right). \tag{5.5.14}$$

Furthermore, we have

$$\frac{1}{r}\frac{\partial\psi(r)}{\partial r} = \frac{1}{3}\frac{c\sqrt{r^2+c^2}+r^2+2c^2}{c+\sqrt{r^2+c^2}}. \tag{5.5.15}$$

The analytic form of $\psi(r)$ of the inverse multiquadric $\phi(r) = 1/\sqrt{r^2+c^2}$ in 2D should also be treated with care to avoid the singularity at $r=0$. It can be obtained in a similar fashion as

$$\psi(r) = \sqrt{r^2+c^2} - c\ln\left(c+\sqrt{r^2+c^2}\right) \tag{5.5.16}$$

and

$$\frac{1}{r}\frac{\partial\psi(r)}{\partial r} = \frac{1}{\sqrt{r^{2}+c^{2}}} - \frac{c}{\sqrt{r^{2}+c^{2}}\left(c+\sqrt{r^{2}+c^{2}}\right)}. \tag{5.5.17}$$

Even for a simple differential operator such as the Laplace operator, a closed form of ψ with $\Delta\psi = \phi$ for a given ϕ can be difficult to obtain and evaluate. For instance, for the Gaussian $\phi(r) = e^{-r^2}$ one has

$$\psi(r) = \frac{1}{4}\left(\ln r^2 + \int_{r^2}^{\infty}\frac{e^{-t}}{t}dt\right).$$

For the 3D case, repeated integration can also be employed as in the 2D case. But an alternative approach is also possible. We observe that by defining

$$\psi(r) = \frac{z(r)}{r} \tag{5.5.18}$$

the problem $\Delta\psi = \phi$ can be reformulated as

$$\frac{1}{r^2}\frac{d}{dr}\left(r^2\frac{d\psi}{dr}\right) = \frac{1}{r}\frac{d^2z}{dr^2} = \phi.$$

This implies that we only have to solve

$$\frac{d^2z}{dr^2} = r\phi.$$

An ordinary differential equation solver or direct integration can be applied to obtain z and hence ψ.

For multiquadrics $\phi = \sqrt{r^2+c^2}$ we have

$$z\left(r\right) = \frac{r\left(2r^2+5c^2\right)}{24}\sqrt{r^2+c^2} + \frac{c^4}{8}\ln\left(r+\sqrt{r^2+c^2}\right) + C_1r + C_2$$

Using (5.5.18), we get

$$\psi\left(r\right) = \frac{\left(2r^2+5c^2\right)}{24}\sqrt{r^2+c^2} + \frac{c^4}{8r}\ln\left(r+\sqrt{r^2+c^2}\right) + C_1 + \frac{C_2}{r}.$$

We observe the second term $(c^4/8r)\ \ln(r+\sqrt{r^2+c^2})$ has a singularity at $r = 0$. To cancel the singularity, we choose

$$C_1 = 0, \quad C_2 = -\frac{c^4\ln c}{8}.$$

This implies

$$\begin{aligned} & \lim_{r\to 0}\left(\frac{c^4}{8r}\ln\left(r+\sqrt{r^2+c^2}\right)+\frac{C_2}{r}\right) \\ = \quad & \lim_{r\to 0}\frac{c^4}{8r}\left(\ln\left(r+\sqrt{r^2+c^2}\right)-\ln c\right) \\ = \quad & \frac{c^3}{8}. \end{aligned}$$

Thus,

$$\psi\left(r\right)=\begin{cases} \dfrac{\left(2r^2+5c^2\right)}{24}\sqrt{r^2+c^2}+\dfrac{c^4}{8r}\ln\left(\dfrac{r+\sqrt{r^2+c^2}}{c}\right), & r>0, \\ \dfrac{c^3}{3}, & r=0. \end{cases} \tag{5.5.19}$$

By taking the derivative of above expression with respect to r, we have

$$\begin{aligned} \frac{1}{r}\frac{\partial\psi(r)}{\partial r} = \quad & \frac{\sqrt{r^{\,2}+c^{\,2}}}{6}+\frac{2r^{\,2}+5c^{\,2}}{24\sqrt{r^{\,2}+c^{\,2}}} \\ & -\frac{c^4}{8r^{\,3}}\ln\left(\frac{r+\sqrt{r^{\,2}+c^{\,2}}}{c}\right)+\frac{c^4}{8r^{\,2}\sqrt{r^{\,2}+c^{\,2}}}. \end{aligned} \tag{5.5.20}$$

There are singularities in last two terms at $r=0$ in (5.5.20). By l'Hôpital's Rule, we have

$$\begin{aligned} & \lim_{r\to 0}\left(-\frac{c^4}{8r^{\,3}}\ln\left(\frac{r+\sqrt{r^{\,2}+c^{\,2}}}{c}\right)+\frac{c^4}{8r^{\,2}\sqrt{r^{\,2}+c^{\,2}}}\right) \\ = \quad & \frac{c^4}{8}\lim_{r\to 0}\frac{r+\ln c\sqrt{r^{\,2}+c^{\,2}}-\sqrt{r^{\,2}+c^{\,2}}\ln\left(r+\sqrt{r^{\,2}+c^{\,2}}\right)}{r^{\,3}\sqrt{r^{\,2}+c^{\,2}}} \\ = \quad & \frac{c^4}{8}\lim_{r\to 0}\frac{\ln c-\ln\left(r+\sqrt{r^{\,2}+c^{\,2}}\right)}{4r^{\,3}+3c^{\,2}r} \\ = \quad & -\frac{c^4}{8}\lim_{r\to 0}\frac{1}{12r^{\,2}+3c^{\,2}} \\ = \quad & -\frac{c^{\,2}}{24}. \end{aligned}$$

Then (5.5.20) can be simplified as

$$\frac{1}{r}\frac{\partial\psi(r)}{\partial r}=\frac{\sqrt{r^{\,2}+c^{\,2}}}{6}+\frac{2r^{\,4}+5c^{\,2}r^{\,2}+3c^4}{24r^{\,2}\sqrt{r^{\,2}+c^{\,2}}}-\frac{c^4}{8r^{\,3}}\ln\left(\frac{r+\sqrt{r^{\,2}+c^{\,2}}}{c}\right),$$

for $r \neq 0$, and

$$\lim_{r\to 0} \frac{1}{r}\frac{\partial \psi(r)}{\partial r} = \frac{c}{6} + \frac{5c^2}{24c} - \frac{c^2}{24} = \frac{c(9-c)}{24}.$$

Thus,

$$\frac{1}{r}\frac{\partial \psi(r)}{\partial r} = \begin{cases} \dfrac{\sqrt{r^2+c^2}}{6} - \dfrac{c^4}{8r^3}\ln\left(\dfrac{r+\sqrt{r^2+c^2}}{c}\right) & r \neq 0, \\ +\dfrac{2r^4+5c^2r^2+3c^4}{24r^2\sqrt{r^2+c^2}}, & \\ \dfrac{c(9-c)}{24}, & r = 0. \end{cases}$$

Similarly, for inverse multiquadrics we obtain

$$\psi(r) = \begin{cases} \dfrac{\sqrt{r^2+c^2}}{2} + \dfrac{c^2}{2r}\ln\left(\dfrac{r+\sqrt{r^2+c^2}}{c}\right), & r > 0, \\ \dfrac{c}{2}, & r = 0, \end{cases}$$

and

$$\frac{1}{r}\frac{\partial \psi(r)}{\partial r} = \begin{cases} \dfrac{1}{2\sqrt{r^2+c^2}} - \dfrac{c^2}{2r}\ln\left(\dfrac{r+\sqrt{r^2+c^2}}{c}\right) & r \neq 0, \\ -\dfrac{c^2}{3r^2\sqrt{r^2+c^2}}, & \\ -\dfrac{1}{6c}, & r = 0. \end{cases}$$

Example 5.5.21 Consider the Poisson problem

$$\begin{aligned} \Delta u &= 8\exp(-2x+2y), && (x,y) \in \Omega, \\ u &= \exp(-2x+2y), && (x,y) \in \Gamma^D, \\ \frac{\partial u}{\partial n} &= \exp(-2x+2y)(-2n_x+2n_y), && (x,y) \in \Gamma^N, \end{aligned}$$

where (n_x, n_y) is a unit normal vector. The computational domain is bounded by the following curve:

$$\Gamma^D \cup \Gamma^N = \left\{ (r\cos\theta, r\sin\theta) : r = \left(\cos(3\theta) + \sqrt{2 - \sin^2(3\theta)}\right)^{1/3} \right\}.$$

Let Γ^N be the boundary above the x-axis and Γ^D be the boundary

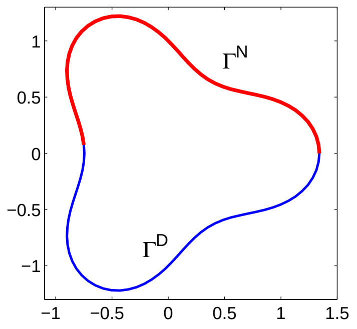

Fig. 5.3. Computational domain: Cassini

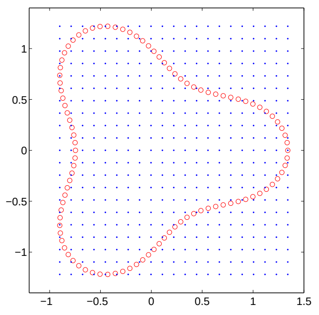

Fig. 5.4. Domain embedding.

below the x-axis (see Figure 5.3. The exact solution is given by $u(x, y) = \exp(-2x + 2y)$.

We choose 50 points on each of the boundary Γ^D and Γ^N respectively. For the evaluation of the particular solutions, 441 evenly distributed interpolation points were generated to cover the computation domain (see Figure 5.4). The inverse MQ is selected as the basis function. To obtain the homogeneous solution, we chose 50 points on Γ^N, 50 points on Γ^D, and 100 points on the fictitious boundary which consists of a circle with center at (0,0) and radius 3. The RMSE for interpolating $f(x, y)$ and $u(x, y)$ with various of shape parameters of inverse MQ are shown in Figure 5.5. We notice that the best error occurs at $c = 1.3$ for $f(x, y)$ while the best error for $u(x, y)$ occurs at $c = 1.1$. Since $f(x, y)$ is a known function, we can use it as a guidance to find the suitable shape parameter c.

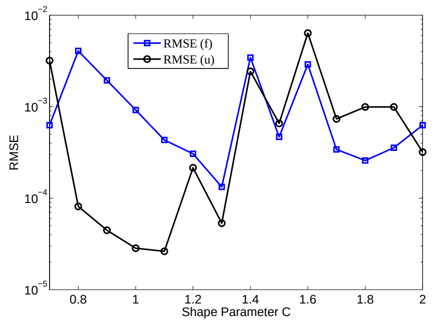

Fig. 5.5. RMSE for f and u with respect to various shape parameters.

We provide the MATLAB code of this example. In lines 15–20, we generate the evenly distributed interpolation points to cover the computational domain. In lines 24–27, we perform the following interpolation to obtain $\{\alpha_k\}$:

$$f(x_i, y_i) = \sum_{k=1}^{n} \alpha_k \frac{1}{\sqrt{r^2 + c^2}}, \quad 1 \leq i \leq n.$$

We then use $\{\alpha_k\}$ to evaluate the particular solution for the rest of the program. In line 30, we update the boundary condition using newly obtained particular solutions at each point on Dirichlet boundary. In lines 31–36, we perform the same task to update the Neumann boundary condition. In lines 40–48, we establish the MFS. In lines 50 and 52, we obtain the particular solutions and homogeneous solutions, respectively, at the evaluation points. In line 55, we calculate the RMSE errors at all the evaluation points.

Program 5.5.21

```
1  clear all;
2  load -ascii test.txt; %Evaluation points
3  load -ascii DB.txt; %Boundary points on Dirichlet boundary
4  load -ascii NB.txt; %Boundary points on Neumann boundary
5  load -ascii NV.txt %Normal vector (nx,ny) for NB.txt
6  Dirichlet = @ (x,y) exp(-2*x+2*y);
```

```
Neumann = @ (x,y,nx,ny) exp(-2*x+2*y).*(-2*nx+2*ny);
exact = @ (x,y) exp(-2*x+2*y); % Exact solution
f = @ (x,y) 8*exp(-2*x+2*y); % Inhomogeneous term
mq = @ (r,c) sqrt(r.∧2+c*c);
mps = @ (r,c) (-c*log(mq(r,c)+c)+mq(r,c));%particular solution
rad = 3; %Radius of source points
c = 1.0; %Shape parameter of Inverse MQ
bdpt = [DB;NB]; nx = NV(:,1); ny = NV(:,2);
%Generate mesh points covering the domain
x1 = min(bdpt(:,1)); x2 = max(bdpt(:,1));
y1 = min(bdpt(:,2)); y2 = max(bdpt(:,2));
[X,Y] = meshgrid((x2-x1)/20*(0:20),(y2-y1)/20*(0:20));
intx = reshape(X,[ ],1)-abs(x1);
inty = reshape(Y,[ ],1)-abs(y1);
intp = [intx,inty]; %Uniform mesh points
m = length(bdpt); % # of boundary point
m1 = length(NB); % # of Neumann boundary point
upn = zeros(m1,length(intp)); B2 = zeros(m1,m);
RHS = f(intp(:,1),intp(:,2)); %RHS of governor equation
DM1 = pdist2(intp,intp); %Distance matrix
A = 1./mq(DM1,c); %RBFs matrix
a = A\RHS; % Solve RBF matrix
DM1A = pdist2(DB,intp);
%Particular solutions at the boundary collocation points
ps = mps(DM1A,c)*a;
%Update Dirichlet boundary condition
g1 = Dirichlet(DB(:,1),DB(:,2))-ps;
DM1B = pdist2(NB,intp);
tmp = (1-c./(c+mq(DM1B,c)))./mq(DM1B,c);
for i = 1:m1 %The RBF matrix Neumann boundary condition
    upn(i,:)=(NB(i,1)-intx)*nx(i)+(NB(i,2)-inty)*ny(i);
end
%Updated Neumann boundary condition
g2 = Neumann(NB(:,1),NB(:,2),NV(:,1),NV(:,2))-(upn.*tmp)*a;
g = [g1;g2]; %Updated boundary conditions
[sx,sy] = pol2cart(2*pi*(1:m)/m,rad); %Source points
source = [sx’,sy’];
%Find coefficients for homogeneous solution
DM2 = pdist2(DB,source);
B1 = log(DM2); %The MFS matrix
```

```
DM3 = pdist2(NB,source);
for i = 1:m1 %The MFS matrix Neumann boundary condition
    B2(i,:) = (NB(i,1)-sx')*NV(i,1)+(NB(i,2)-sy')*NV(i,2);
end
B2 = B2.\(DM3.∧2);
B = [B1;B2]; %The overall MFS matrix
b = B\ g; %Solve Ax=b
DM3 = pdist2(test,intp);
%Particular solutions at the evaluation points
up = mps(DM3,c)*a;
DM4 = pdist2(test,source);
%Homogeneous solution at the evaluation points
uh = log(DM4)*b;
u = uh + up; %Homogeneous solutions+particular solutions
sol = exact(test(:,1),test(:,2));
rmse = norm(u-sol)/sqrt(length(test)); %RMSE errors of u
fprintf('RMSE: %e\n', rmse)
```

5.5.2 Monomial Basis Functions

For conditionally positive definite radial basis functions, the additional polynomial terms are required as indicated in Section 1.2 and equation (1.2.5). Hence the corresponding particular solutions are also necessary. We note that the particular solutions for monomial right-hand sides for the 2D and 3D cases are available in the literature [CLG94, GMCC03]. The results are as follows.

A particular solution of

$$\Delta\psi = x^n y^m, \quad m \geq 0, n \geq 0,$$

is given by [CLG94]

$$\psi(x,y) = \begin{cases} \displaystyle\sum_{k=1}^{\lfloor \frac{n+2}{2} \rfloor} (-1)^{k+1} \frac{m!n!x^{m+2k}y^{n-2k+2}}{(m+2k)!(n-2k+2)!}, & m \geq n, \\ \displaystyle\sum_{k=1}^{\lfloor \frac{m+2}{2} \rfloor} (-1)^{k+1} \frac{m!n!x^{m-2k+2}y^{n+2k}}{(m-2k+2)!(n+2k)!} & m < n. \end{cases}$$

A particular solution of

$$\Delta\psi = x^\ell y^m z^n$$

in 3D can be obtained similar to the 2D case [GMCC03]:

$$\psi = \sum_{i=1}^{\lfloor \frac{m}{2} \rfloor + \lfloor \frac{n}{2} \rfloor + 1} x^{\ell+2i} \sum_{j=\max\left\{1, i-\lfloor \frac{n}{2} \rfloor\right\}}^{\min\left\{i, \lfloor \frac{m}{2} \rfloor + 1\right\}} a_{ij} y^{m-2j+2} z^{n-2i+2j}$$

where

$$\begin{aligned} a_{ij} &= \frac{-1}{(\ell+2i)(\ell+2i-1)} \left[(m-2j+4)(m-2j+3)a_{i-1,j-1} \right. \\ &\quad \left. + (n-2i+2j+2)(n-2i+2j+1)a_{i-1,j}\right], \\ a_{11} &= a_{i0} = a_{0j} = 0. \end{aligned}$$

5.5.3 Compactly Supported Radial Basis Functions

As was already pointed out in Section 2.8, there are good reasons to use compactly supported radial basis functions (CS-RBFs) for dealing with large problems involving data with limited smoothness. Therefore we now focus on the derivation of particular solutions based on compactly supported radial basis functions [CGS03]. Note that we already have some direct cases in (5.5.2) and (5.5.3). For practical implementation, all radial basis functions in Table 1.4 should be scaled, but due to (5.5.1) the general scaling can be deduced from the unscaled case easily.

For the 2D case, and for the original radial basis function ϕ supported in $[0,1]$, an explicit representation of $\psi(r)$ can be derived by straightforward integration as follows [CBP99]:

$$\psi(r) = \begin{cases} \displaystyle\int_0^r \frac{1}{s}\left[\int_0^s t\phi(t)\,dt\right] ds, & r \le 1, \\ \displaystyle\int_0^1 \frac{1}{s}\left[\int_0^s t\phi(t)\,dt\right] ds + \int_1^r \frac{1}{s}\left[\int_0^1 t\phi(t)\,dt\right] ds, & r > 1. \end{cases}$$

Note that the part for $r > 1$ can be rewritten as

$$\psi(r) = \psi(1) + \psi'(1)\ln r, \ r > 1, \tag{5.5.22}$$

and this also explains why the resulting function ψ is not compactly supported. At $r = 1$ there must be a smooth extension with a harmonic function of r, which is exactly what the above formula says. The double integration shown above can be performed easily by using symbolic software. A list of particular solutions $\psi(r)$ corresponding to various compactly supported functions of Wendland type is given in Table 5.3 [CBP99]. Note that α in Table 5.3 is the scaling factor of the CS-RBFs.

ϕ	ψ
$\left(1-\frac{r}{\alpha}\right)_+^2$	$\begin{cases} \frac{r^4}{16\alpha^2}-\frac{2r^3}{9\alpha}+\frac{r^2}{4}, & r\le\alpha, \\ \frac{13\alpha^2}{144}+\frac{\alpha^2}{12}\ln\left(\frac{r}{\alpha}\right), & r>\alpha. \end{cases}$
$\left(1-\frac{r}{\alpha}\right)_+^4\left(4\frac{r}{\alpha}+1\right)$	$\begin{cases} \frac{4r^7}{49\alpha^5}-\frac{5r^6}{12\alpha^4}+\frac{4r^5}{5\alpha^3}-\frac{5r^4}{8\alpha^2}+\frac{r^2}{4}, & r\le\alpha, \\ \frac{529\alpha^2}{5880}+\frac{\alpha^2}{14}\ln\left(\frac{r}{\alpha}\right), & r>\alpha. \end{cases}$
$\left(1-\frac{r}{\alpha}\right)_+^6 * \left(35\frac{r^2}{\alpha^2}+18\frac{r}{\alpha}+3\right)$	$\begin{cases} \frac{7r^{10}}{20\alpha^8}-\frac{64r^9}{27\alpha^7}+\frac{105r^8}{16\alpha^6}-\frac{64r^7}{7\alpha^5} +\frac{35r^6}{6\alpha^4}-\frac{7r^4}{4\alpha^2}+\frac{3r^2}{4}, & r\le\alpha \\ \frac{3517\alpha^2}{15120}+\frac{\alpha^2}{6}\ln\left(\frac{r}{\alpha}\right), & r>\alpha. \end{cases}$
$\left(1-\frac{r}{\alpha}\right)_+^8 * \left(32\frac{r^3}{\alpha^3}+25\frac{r^2}{\alpha^2}+8\frac{r}{\alpha}+1\right)$	$\begin{cases} \frac{32r^{13}}{169\alpha^{11}}-\frac{77r^{12}}{48\alpha^{10}}+\frac{64r^{11}}{11\alpha^9}-\frac{231r^{10}}{20\alpha^8} +\frac{352r^9}{27\alpha^7}-\frac{231r^8}{32\alpha^6}+\frac{11r^6}{6\alpha^4}-\frac{11r^4}{16\alpha^2}+\frac{r^2}{4}, & r\le\alpha, \\ \frac{541961\alpha^2}{8030880}+\frac{7\alpha^2}{156}\ln\left(\frac{r}{\alpha}\right), & r>\alpha. \end{cases}$

Table 5.3. *A list of ψ corresponding to various CS-RBFs ϕ in 2D.*

The explicit representation of $\psi(r)$ for the 3D case is similar to the 2D case. In particular,

$$\psi(r)=\begin{cases} \int_0^r \frac{1}{s^2}\left[\int_0^s t^2\phi(t)\,dt\right]ds, & r\le 1, \\ \int_0^1 \frac{1}{s^2}\left[\int_0^s t^2\phi(t)\,dt\right]ds+\int_1^r \frac{1}{s^2}\left[\int_0^1 t^2\phi(t)\,dt\right]ds, & r>1, \end{cases}$$

where now instead of (5.5.22) we have

$$\psi(r)=\psi(1)+\psi'(1)(1-1/r),\ r>1$$

as a harmonic extension. Explicit $\psi(r)$ corresponding to various Wendland CS-RBFs $\phi(r)$ are given in Table 5.4. Note that $\{\phi_i\}_{i=1}^4$ in this table is the same as those in Table 5.3.

ϕ	ψ
$\phi_1(r)$	$\begin{cases} \dfrac{r^4}{20\alpha^2}-\dfrac{r^3}{6\alpha}+\dfrac{r^2}{6}, & r\le\alpha, \\ \dfrac{\alpha^2}{12}-\dfrac{\alpha^3}{30r}, & r>\alpha. \end{cases}$
$\phi_2(r)$	$\begin{cases} \dfrac{r^7}{14\alpha^5}-\dfrac{5r^6}{14\alpha^4}+\dfrac{2r^5}{3\alpha^3}-\dfrac{r^4}{2\alpha^2}+\dfrac{r^2}{6}, & r\le\alpha, \\ \dfrac{\alpha^2}{14}-\dfrac{\alpha^3}{42r}, & r>\alpha. \end{cases}$
$\phi_3(r)$	$\begin{cases} \dfrac{7r^{10}}{22\alpha^8}-\dfrac{32r^9}{15\alpha^7}+\dfrac{35r^8}{6\alpha^6}-\dfrac{8r^7}{\alpha^5}+\dfrac{5r^6}{\alpha^4}-\dfrac{7r^4}{5\alpha^2}+\dfrac{r^2}{2}, & r\le 1, \\ \dfrac{\alpha^2}{6}-\dfrac{8\alpha^3}{165r}, & r>1. \end{cases}$
$\phi_4(r)$	$\begin{cases} \dfrac{16r^{13}}{91\alpha^{11}}-\dfrac{77r^{12}}{52\alpha^{10}}+\dfrac{16r^{11}}{3\alpha^9}-\dfrac{21r^{10}}{2\alpha^8}+\dfrac{176r^9}{15\alpha^7}-\dfrac{77r^8}{12\alpha^6} \\ +\dfrac{11r^6}{7\alpha^4}-\dfrac{11r^4}{20\alpha^2}+\dfrac{r^2}{6}, & r\le 1, \\ \dfrac{4903\alpha^2}{60060}-\dfrac{8\alpha^3}{165r}, & r>1. \end{cases}$

Table 5.4. *A list of ψ corresponding to various CS-RBFs ϕ in 3D.*

We now implement the particular solutions we derived above to solve the following Poisson problems in 2D and 3D.

Example 5.5.23 Consider the Poisson problem

$$\begin{aligned} \Delta u(x,y) &= -\frac{5\pi^2}{4}\sin\pi x\cos\frac{\pi y}{2}, && (x,y)\in\Omega, \\ u(x,y) &= \sin\pi x\cos\frac{\pi y}{2}, && (x,y)\in\Gamma. \end{aligned}$$

where the computational domain $\Omega\cup\Gamma$ is the same as Example 5.5.21. The exact solution is given by $u^*=\sin\pi x\cos\pi y/2$.

In this example, we select the CS-RBF $\phi(r)=(1-r/\alpha)_+^4(4r/\alpha+1)$ as the basis function. For evaluating the particular solution, we chose 441 uniformly and irregularly distributed interpolation points to cover the whole domain (see Figures 5.6 and 5.7). The profiles of the,interpolation matrix are shown on the right side of above two figures. For the case of uniformly distributed interpolation points, we set $\alpha = 0.26$, and the interpolation matrix is sparse and banded with 6837 nonzero entries as shown in the right hand side of Figure 5.6. For the case of irregularly distributed points, we set $\alpha = 0.3$, and we found 7427 nonzero entries. For the implementation of the MFS, we chose 300 evenly distributed collocation and source points on the boundary and fictitious boundary. The fictitious is a circle with radius 4 and center (0,0). For the uniform case, we obtain the accuracy of $1.09E-2$ and the matrix is 3.5% full. For the irregular case, the accuracy is $2.69E-2$ and the matrix is 3.8% full. To improve the accuracy, we need to increase the number of interpolation points.

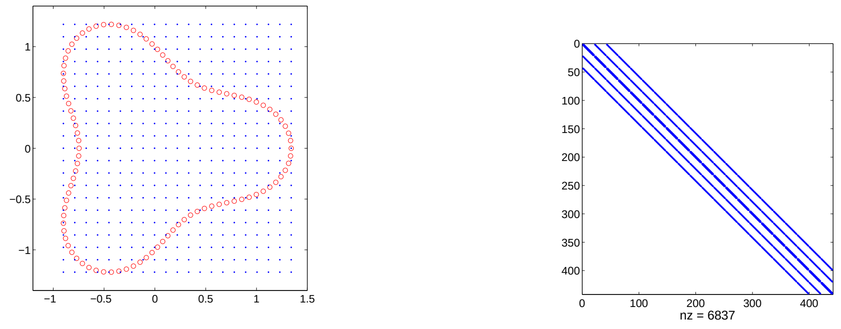

Fig. 5.6. Profile of uniformly distributed interpolation points (left) and the resulting sparse matrix using $\alpha = 0.26$.

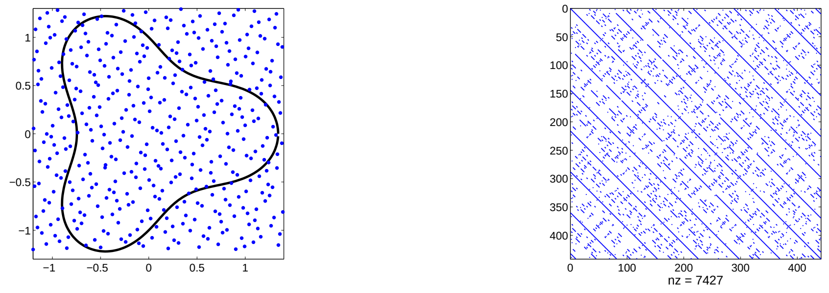

Fig. 5.7. Profile of irregularly distributed interpolation points (left) and the resulting sparse matrix using $\alpha = 0.3$.

To compare the accuracy using uniform and irregular points, we used

2916 points for testing. In Table 5.5, NZ denotes the number of nonzero entry in the interpolation matrix. With the same scaling factor α, the case of uniformly distributed points contains slightly more NZ than the irregular case. The performance of the uniformly distributed points is much superior than the irregular case.

	Uniform		Irregular	
α	RMSE	NZ	RMSE	NZ
0.1	$9.89E-3$	47856	$5.26E-2$	37508
0.2	$3.98E-4$	176356	$2.63E-3$	147368
0.3	$1.27E-4$	384012	$1.01E-3$	322068
0.4	$6.19E-5$	641936	$4.52E-4$	552098

Table 5.5. *The RMSE errors using 2916 uniformly and irregularly distributed interpolation points.*

□

Example 5.5.24 Let us consider the following 3D problem using compactly supported radial basis functions:

$$\begin{aligned} \Delta u &= -3\cos x\cos y\cos z, && \text{in } \Omega \\ u &= \cos x\cos y\cos z, && \text{on } \Gamma. \end{aligned}$$

Define $R(\theta) = \sqrt{\cos(2\theta) + \sqrt{1.1 - \sin^2(2\theta)}}$. Then the surface of the domain Ω is represented by the parametric surface

$$\mathbf{x}(\theta,\phi) = \begin{pmatrix} R(\theta)\cos(\theta) \\ R(\theta)\sin(\theta)\cos(\phi) \\ R(\theta)\sin(\theta)\sin(\phi) \end{pmatrix} \qquad (5.5.25)$$

where $\theta \in [0,\pi), \phi \in [0,2\pi)$. The 3D graph of the parametric surface in (5.5.25) can be found in Example 4.4.11. The exact solution is given by

$$u^* = \cos x\cos y\cos z, \qquad (x,y,z) \in \Omega \cup \Gamma.$$

In this example, we chose Wendland's basis function $\varphi = (1-r)_+^4(4r+1)$ to approximate the forcing term. We chose 300 quasi-random points in a box $[-1.5, 1, 5] \times [-0.5, 0.5] \times [0.5, 0.5]$. In the method of fundamental solutions, we chose 100 quasi-random test points on the peanut-shape surface and 100 quasi-random source points on a sphere with center at the origin with radius 9. The numerical results were computed along the x-axis with $y = z = 0$. The results in terms of relative errors in percent with three different scaling factors are shown in Figure 5.8. We used scaling factors $\alpha = 0.7,\ 1.0$, and 1.4 in the sense of the parameter c in (5.5.1). Notice that these results are consistent with our intuition. With larger support, more interpolation points are included in the approximation. Therefore as more information is provided, more accurate solutions are expected.

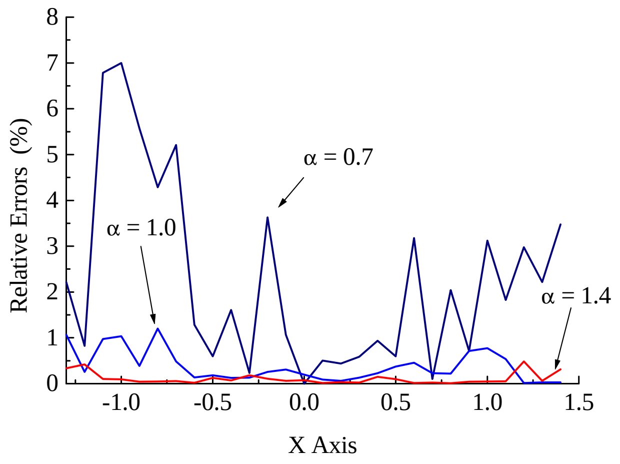

Fig. 5.8. The effect of various scaling factors.

□

5.6 Particular Solutions for Biharmonic Equations

The derivation of particular solutions for the biharmonic operator is similar to the one for the Laplace operator which was described in the previous sections. However, the algebraic manipulations are getting more complicated, and the correct scaling now is

$$\Delta^2(c^4\psi(r/c)) = \phi(r/c).$$

In some cases such as multiquadrics, the continuing integration on (5.5.13) two more times can become very difficult.

But in case of factorizable differential operators, the Method of Particular Solutions can always be iterated. If $L = L_1L_2$ is a factorizable differential operator, the problem to find a particular solution pair (u, f) with $Lu = f$ can be split into finding particular solution pairs (v, f) with $L_1v = f$ and (u, v) with $L_2u = v$ due to

$$Lu = L_1L_2u = L_1v = f.$$

For approximate particular solutions using radial basis functions, users can first find an approximate particular solution v for $L_1v = f$ using pairs (ψ_1, ϕ_1) of particular RBF solutions for the operator L_1. This requires an approximation of f in terms of translates of ϕ_1 and yields an approximation of v in terms of translates of ψ_1. Then fundamental RBF solutions (ψ_2, ϕ_2) for L_2 are used, and v must be re-represented in terms of translates of ϕ_2 to get an approximate particular solution u to $Lu = f$ or $L_2u = v$ in terms of translates of ψ_2.

5.7 Particular Solutions for Helmholtz Operators

Before we go into details, let us settle the scaling problem for particular solutions of Helmholtz-type operators $\Delta \pm \lambda^2$.

Assume that we have a pair $(\psi_{\pm\lambda}, \phi_{\pm\lambda})$ satisfying

$$(\Delta \pm \lambda^2)\psi_{\pm\lambda} = \phi_{\pm\lambda}.$$

Then, by some easy calculation, we get pairs $(c^2\psi_{\pm c\lambda}(r/c), \phi_{\pm c\lambda}(r/c))$ of scaled particular solutions for $\Delta \pm \lambda^2$ via

$$(\Delta \pm \lambda^2)(c^2\psi_{\pm c\lambda}(r/c)) = \phi_{\pm c\lambda}(r/c) \tag{5.7.1}$$

for all positive scaling factors c. Note how the scaling factor c interacts with λ.

Furthermore, if the following discussion allows complex λ, we can fix the sign in the differential equation *ad libitum.* But note that only the case $-\Delta - \lambda^2$ for nonnegative λ will be elliptic in general.

5.7.1 Historical Remarks

In most applications so far, the Dual Reciprocity Method was applied in the case of the Laplace or biharmonic operator on the left hand side, while the remaining terms in the original differential equation were moved to the right hand side and treated as source terms. This was primarily due to the difficulty in obtaining particular solutions in closed form as we have indicated in the earlier sections. As a result, the Dual Reciprocity Method gets additional source terms and needs an additional iteration. For instance, to find an approximate particular solution to the Helmholtz-type equation

$$\Delta u \pm \lambda^2 u = f(x,y), \tag{5.7.2}$$

one has to rearrange (5.7.2) to the following form

$$\Delta u = f(x,y) \mp \lambda^2 u. \tag{5.7.3}$$

Then one has to iterate as follows:

- Given an approximation $\tilde{u}$ of u in (5.7.2), use it in the right–hand side of (5.7.3).
- Then apply the Method of Particular Solutions for Δ in (5.7.3) to get a new approximate solution $\hat{u}$ for the Helmholtz–type differential equation (5.7.2).
- Repeat this for $\tilde{u} := \hat{u}$ until changes are small.

The disadvantages of (5.7.3) instead of (5.7.2) are obvious. Not only is the source term in (5.7.3) more difficult to approximate, but also the above iteration scheme is required and the convergence of the iteration is uncertain.

Zhu [Zhu93] worked along these lines using Helmholtz-type operators as the main differential operators in the Dual Reciprocity Method. However, no closed form particular solutions are possible by using polynomial radial basis functions, as is shown in his paper. Instead, a non–trivial recursive scheme was developed to evaluate the particular solutions.

Example 5.7.4 We give an example to demonstrate the problems incurred by iterations when shifting terms to the right–hand side. For simplicity, we only use the Thin Plate Splines as basis functions in this example. Let us consider the following modified Helmholtz equation

$$\begin{aligned} \Delta u - \lambda^2 u &= (e^x + e^y)(1-\lambda^2), \quad (x,y) \in \Omega, \\ u &= e^x + e^y, \qquad\qquad\quad (x,y) \in \Gamma, \end{aligned} \tag{5.7.5}$$

where $\Omega \cup \Gamma = \{(x,y) : x^2 + y^2 \leq 1\}$. The exact solution is given by $u^* = e^x + e^y$. In the past, due to the lack of closed form particular solutions for the Helmholtz equation, the governing equation of (5.7.5) was often reformulated as

$$\Delta u_{n+1} = (e^x + e^y)(1 - \lambda^2) + \lambda^2 u_n, \qquad (x, y) \in \Omega. \qquad (5.7.6)$$

in which numerical iterations are required.

To obtain an approximate particular solution by this technique, we chose 60 quasi-random points in the interior to interpolate the right hand side. To approximate the homogeneous solution, the method of fundamental solutions was used and 40 evenly distributed points on the boundary were chosen. The source points of the method of fundamental solutions were chosen on a fictitious boundary of a circle with radius 8 and center at the origin. In the iteration of (5.7.6), we set the tolerance to $\varepsilon = 10^{-4}$, i.e. when $\max_{(x,y)\in\Omega} |u_{n+1}(x, y) - u_n(x, y)| < \varepsilon$, the iteration will be terminated. In Table 5.7.6, it turns out that the iteration diverges when $\lambda^2 > 6$.

λ^2	# of Iteration	CPU
1	8	3.30
2	12	4.56
3	18	5.38
4	31	7.14
5	78	17.41
6	Divergence	

Table 5.6. *Iterations and relative CPU times for various* λ^2.

As an alternative approach, we avoided the iteration by using our special particular solution (5.7.15) for the modified Helmholtz equation to that is derived below. Then the absolute error of the solution is shown in Figure 5.9. It took only 0.22 relative CPU time. Here we chose $\lambda^2 = 1000$. Solution with higher wave numbers can also be achieved easily.

$\square$

Ingber et. al. [IPT92] also developed a numerical scheme to find an approximate particular solution for (5.7.2) using radial basis functions. We briefly state their approach because it has connections to our construction below. To find an approximate particular solution for (5.7.2),

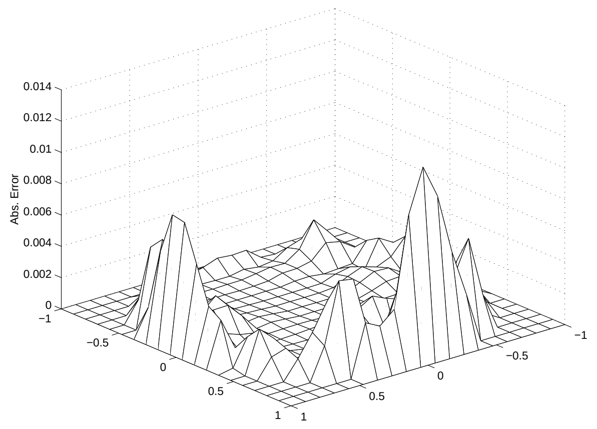

Fig. 5.9. The profile of the absolute error of solution of (5.7.5)for $\lambda^2 = 1000$.

it suffices to determine a particular solution $\psi(r)$ for the equation

$$\frac{1}{r}\frac{d}{dr}\left(r\frac{d\psi}{dr}\right) - \lambda^2\psi = \phi(r) \tag{5.7.7}$$

where $\phi(r)$ is a radial basis function and λ is a complex number. By letting $z = \lambda r$ and rewriting both functions in terms of z, we get

$$z^2\frac{d^2\psi}{dz^2} + z\frac{d\psi}{dz} - z^2\psi = \frac{z^2\phi}{\lambda^2}.$$

This equation is an inhomogeneous modified Bessel equation of order zero. The solutions of the homogeneous problem are linear combinations of the modified Bessel functions I_0 and K_0 of first and second kind. One can get a particular solution by using the classical variation of parameters method, i.e.

$$\psi(z) = A(z)I_0(z) + B(z)K_0(z). \tag{5.7.8}$$

The coefficient functions $A(z)$ and $B(z)$ are determined by

$$\begin{array}{rcrcl} A'I_0 & + & B'K_0 & = & 0, \\ A'I_0' & + & B'K_0' & = & \dfrac{\phi}{\lambda^2}. \end{array}$$

Solving for A and B yields

$$\begin{aligned} A(z) = & \frac{1}{\lambda^2}\int \frac{K_0\phi}{I_0'K_0 - K_0'I_0}, \\ B(z) = & \frac{1}{\lambda^2}\int \frac{I_0\phi}{I_0'K_0 - K_0'I_0}. \end{aligned}$$

Thus the determination of particular solutions of Helmholtz-type equations is much more sophisticated than for the Laplace operator case.

The particular solution of (5.7.8) involves integration of Bessel functions. But for efficiency in applications, it is important to be able to solve (5.7.7) explicitly. Consequently, this motivates further search for closed forms of particular solutions using various differential operators. Along this line, Chen and Rashed [CR98] discovered that a closed form particular solution can be obtained by choosing Helmholtz-type differential operators as main differential operators and thin plate splines (TPS) as the basis functions in the context of the Dual Reciprocity Method. The discovery of these particular solutions was purely by observation and no systematic derivation was given.

Golberg et. al. [GCR99] re-derived the results of [CR98] by an approach which generalizes the **annihilator method** used to obtain particular solutions for ordinary differential equations. To be more specific, suppose there is a differential operator M such that $M\phi = 0$. Then operating on $L\psi = \phi$ gives

$$ML\psi = M\phi = 0$$

so ψ now satisfies a homogeneous equation. In addition, if we can choose M to commute with L, then one can write ψ as

$$\psi = \Theta + \Psi$$

where $M\Theta = 0$ and $L\Psi = 0$ so that ψ can be found by solving a pair of homogeneous equations, i.e. when $ML = LM$ then

$$ML\psi = ML(\Theta + \Psi) = M(L\Theta) + M(L\Psi) = L(M\Theta) + M(L\Psi) = 0.$$

We shall implement this concept in the next section.

5.7.2 Thin–Plate Splines in 2D

To obtain a specific closed form particular solution ψ for $L = \Delta - \lambda^2$ in 2D, we choose thin–plate splines $\phi = r^2 \ln r$ and have to solve

$$\left(\Delta - \lambda^2\right) \psi = r^2 \ln r. \tag{5.7.9}$$

If we choose $M = \Delta^2$ for the annihilator method, we can use $\Delta^2 r^2 \ln r = 0$ outside the origin. Operating on (5.7.9) by $M := \Delta^2$ then gives

$$\Delta^2 \left(\Delta - \lambda^2\right) \psi = 0.$$

Since Δ^2 and $\Delta - \lambda^2$ are radially and translationally invariant, we can focus on

$$\Delta_r^2 \left(\Delta_r - \lambda^2\right) \psi = 0 \tag{5.7.10}$$

where

$$\Delta_r = \frac{1}{r}\frac{d}{dr}(r\frac{d}{dr}) = \frac{d^2}{dr^2} + \frac{1}{r}\frac{d}{dr}.$$

Thus (5.7.10) is a sixth order ODE which has a six dimensional solution space [DG76]. Since Δ_r^2 and $\Delta_r - \lambda^2$ commute and their solution spaces have empty intersection (as we shall see), then we have

$$\psi = \Theta + \Psi$$

where

$$\left(\Delta_r - \lambda^2\right) \Psi = 0 \tag{5.7.11}$$

and

$$\Delta_r^2 \Theta = 0. \tag{5.7.12}$$

As is well known, (5.7.11) is a Bessel equation, and we get

$$\Psi = AI_0(\lambda r) + BK_0(\lambda r)$$

where I_0 and K_0 are modified Bessel functions of first and second kind with order 0, respectively.

By straightforward algebra, we find that (5.7.12) is equivalent to the fourth order Euler equation

$$r^4\frac{d^4\Theta}{dr^4} + 2r^3\frac{d^3\Theta}{dr^3} - r^2\frac{d^2\Theta}{dr^2} + r\frac{d\Theta}{dr} = 0$$

whose characteristic polynomial is $P(r) = r^2(r-2)^2$. Thus it follows from standard theory of ordinary differential equations [DG76] that

$$v = a + b\ln r + cr^2 + dr^2 \ln r,$$

where a, b, c, and d are constants to be determined. Hence,

$$\psi = AI_0(\lambda r) + BK_0(\lambda r) + a + b\ln r + cr^2 + dr^2 \ln r. \tag{5.7.13}$$

To determine the constants in (5.7.13) we must use the additional condition that $\left(\Delta_r - \lambda^2\right) \psi = r^2 \ln r$ to guarantee that ψ is a particular solution of $\Delta_r - \lambda^2$. Since $\left(\Delta_r - \lambda^2\right) I_0 = \left(\Delta_r - \lambda^2\right) K_0 = 0$, we have to solve

$$\left(\Delta_r - \lambda^2\right) a + \left(\Delta_r - \lambda^2\right) b\ln r + \left(\Delta_r - \lambda^2\right) cr^2 + \left(\Delta_r - \lambda^2\right) dr^2 \ln r$$
$$= r^2 \ln r.$$

By the method of undeterminate coefficients and some tedious algebra, we obtain

$$a = \frac{-4}{\lambda^4}, \quad b = \frac{-4}{\lambda^4}, \quad c = 0, \quad d = \frac{-1}{\lambda^2}.$$

Thus we end up with

$$\psi = AI_0(\lambda r) + BK_0(\lambda r) - \frac{4}{\lambda^4} - \frac{4\ln r}{\lambda^4} - \frac{r^2 \ln r}{\lambda^2}. \qquad (5.7.14)$$

To get a particular solution which is continuous at $r = 0$, we need to choose B to cancel the $\ln r$ term at $r = 0$. As in [CR98], this can be done by using

$$K_0(\lambda r) \to -\gamma - \ln\left(\frac{\lambda r}{2}\right), \quad r \to 0,$$

where $\gamma \simeq 0.5772156649015328$ is Euler's constant [AS65]. Choosing $B = -4/\lambda^4$ gives the required result. Furthermore, since $\left(\Delta_r - \lambda^2\right) I_0 = 0$ and particular solutions are not unique, we can take $A = 0$. Hence our final result is

$$\psi = \begin{cases} -\dfrac{4}{\lambda^4} - \dfrac{4\ln r}{\lambda^4} - \dfrac{r^2 \ln r}{\lambda^2} - \dfrac{4K_0(\lambda r)}{\lambda^4}, & r > 0, \\[2ex] -\dfrac{4}{\lambda^4} + \dfrac{4\gamma}{\lambda^4} + \dfrac{4}{\lambda^4}\ln\left(\dfrac{\lambda}{2}\right), & r = 0. \end{cases} \qquad (5.7.15)$$

The derivation of the derivative of ψ in (5.7.15) can proceed as follows. The singularities have to be cancelled in a similar way shown above.

$$\begin{aligned} \frac{\partial \psi}{\partial n} &= \frac{\partial \psi}{\partial r}\left(\frac{\partial r}{\partial x} n_x + \frac{\partial r}{\partial y} n_y\right) \\ &= \frac{\partial \psi}{\partial r}\left(\frac{x-\xi}{r} n_x + \frac{y-\eta}{r} n_y\right) \\ &= \frac{1}{r}\frac{\partial \psi}{\partial r}\left((x-\xi)\, n_x + (y-\eta)\, n_y\right) \end{aligned}$$

where $r^2 = (x-\xi)^2 + (y-\eta)^2$, and (n_x, n_y) is a unit normal vector. Hence it is sufficient to find $(1/r)\partial\psi/\partial r$. Note that

$$\frac{d}{dr} K_0(\lambda r) = -\lambda K_1(\lambda r).$$

Form (5.7.15), we have

$$\frac{1}{r}\frac{\partial \psi}{\partial r} = \frac{-4}{r^2 \lambda^4} - \frac{2\ln r}{\lambda^2} - \frac{1}{\lambda^2} + \frac{4K_1(\lambda r)}{\lambda^3}, \quad r \neq 0. \qquad (5.7.16)$$

We observe that

$$K_1(\lambda r) = \left(\ln\left(\frac{\lambda r}{2}\right) + \gamma\right) I_1(\lambda r) + \frac{1}{\lambda r}$$

where

$$I_1(\lambda r) = \frac{\lambda r}{2} + \frac{(\lambda r)^3}{2^2 \cdot 4} + \cdots$$

Considering the expansion of the first term of $I_1(\lambda r)$, we obtain

$$K_1(\lambda r) = \left(\ln\left(\frac{\lambda r}{2}\right) + \gamma\right) \frac{\lambda r}{2} + \frac{1}{\lambda r} + O(r^3).$$

The last term of (5.7.16) can be expanded as follows:

$$\frac{4K_1(\lambda r)}{\lambda^3 r} = \frac{2}{\lambda^2} \ln\left(\frac{\lambda}{2}\right) + \frac{2}{\lambda^2} \ln r + \frac{2\gamma}{\lambda^2} + \frac{4}{r^2\lambda^4} + O(r^2).$$

As $r \to 0$, (5.7.16) can be further simplified as follows:

$$\frac{1}{r}\frac{\partial \psi}{\partial r} = \frac{2}{\lambda^2} \ln\left(\frac{\lambda}{2}\right) + \frac{2\gamma}{\lambda^2} - \frac{1}{\lambda^2}.$$

Thus, we obtain

$$\frac{1}{r}\frac{\partial \psi}{\partial r} = \begin{cases} \dfrac{-4}{r^2\lambda^4} - \dfrac{2\ln r}{\lambda^2} - \dfrac{1}{\lambda^2} + \dfrac{4K_1(\lambda r)}{\lambda^3}, & r \neq 0, \\ \dfrac{2}{\lambda^2} \ln\left(\dfrac{\lambda}{2}\right) + \dfrac{2\gamma}{\lambda^2} - \dfrac{1}{\lambda^2}, & r = 0. \end{cases} \tag{5.7.17}$$

As we know in the implementation of TPS, we need to add the polynomial term as shown in (1.2.5); i.e.,

$$f(x, y) = \sum_{i=1}^{n} \alpha_i r^2 \ln(r) + \beta_1 + \beta_2 x + \beta_3 y. \tag{5.7.18}$$

The particular solutions of the corresponding linear polynomial terms in (5.7.18) for the modified Helmholtz equation is trivial and can be obtained by inspection; i.e.,

$$u_p(x, y) \simeq \sum_{i=1}^{n} \alpha_i \psi - \frac{1}{\lambda^2}(\beta_1 + \beta_2 x + \beta_3 y).$$

Example 5.7.19 Consider the following modified Helmholtz equation

$$\begin{aligned}
(\Delta - 25)u &= -23\exp(x+y), && (x,y) \in \Omega, \\
u &= \exp(x+y), && (x,y) \in \Gamma^D, \\
\frac{\partial u}{\partial n} &= \exp(x+y)(n_x+n_y), && (x,y) \in \Gamma^N,
\end{aligned} \tag{5.7.20}$$

where (n_x, n_y) is a unit normal vector. The computational domain is the same as Figure 5.3 in Example 5.5.21. Let Γ^N be the boundary above the x-axis and Γ^D be the boundary below the x-axis. The exact solution is given by $u(x,y) = \exp(x+y)$.

For the evaluation of the particular solution, we choose the same 441 uniformly distributed interpolation points as shown in Figure 5.4. The Thin Plate Splines (TPS) is selected as the basis function. To obtain the homogeneous solution, we choose 150 collocation points on Γ^N and Γ^D respectively. For the fictitious boundary, we choose 300 source points on a circle with radius of 5 and center (0,0). The RMSE error on the 120 interior evaluation points is $4.94E-5$.

The MATLAB code of the above example is shown as follows. Note that the particular solution in (5.7.15) is shown in lines 10 and 11. The derivative of particular solution in (5.7.17) is shown in lines 12 and 13. For convenience, we do not include the case $r = 0$ in our code. In practice, it is unlikely the interpolation points will collide with the boundary collocation points or evaluation points. In line 31, we add 1 to the diagonal of the distance matrix of interpolation. In this way, we can avoid evaluating $\log 0$ in line 32 and not affect the evaluation of $r^2 \ln r$ for $r = 0$. In lines 29–35, we perform the RBF interpolation using TPS. alpha and beta in lines 36 and 37 are used for the evaluation of approximate particular solution and its derivative in the rest of the code. In lines 40–48, we perform the update of the Neumann boundary condition as follows:

$$\frac{\partial u_h}{\partial n}(x_j, y_j) \quad = \quad e^{(x_j+y_j)}(n_x+n_y) - \left(\sum_{i=1}^{N} \alpha_i \frac{\partial \psi}{\partial n} - \frac{1}{\lambda^2}(\beta_2 n_x + \beta_3 n_y) \right),$$

for $1 \le j \le Neum$. $\{(x_j, y_j)\}_{j=1}^{Neum}$ is the collocation points on the Neumann boundary, and $\{(v_i, w_i)\}_{i=1}^{N}$ is the uniform interpolation points. In line 49–51, we update the Dirichlet boundary condition. In lines 54–62, we formulate the matrix by the MFS.

$$\begin{bmatrix} \frac{-\lambda}{r}K_1(\lambda r_{ij})((x_i-s_j)n_x+(y_i-t_j)n_y) \\ K_0(\lambda r_{kj}) \end{bmatrix}_{1\leq j\leq M, 1\leq i\leq Neum, 1\leq k\leq Deum}$$

where $\{(x_i,y_i)\}_{i=1}^{Neum}$ is the collocation points on the Neumann boundary, $\{(s_j,t_j)\}_{j=1}^{M}$ is source points, and $r_{ij}=\sqrt{(x_i-s_j)^2+(y_i-t_j)^2}$. $Deum$ is the number of collocation on the Dirichlet boundary, and M is the number of source points. r_{kj} defines in a similar way as r_{ij}.

Program 5.7.19

```
clear all;
load -ascii test.txt; %Evaluation points
load -ascii DB.txt; %Boundary points on Dirichlet boundary
load -ascii NB.txt; %Boundary points on Neumann boundary
load -ascii NV.txt %Unit normal vector (nx,ny) for NB.txt
f = @ (x, y) –23*exp(x+y); %Inhomogeneous term
g1 = @ (x, y, nx, ny) exp(x+y).*(nx+ny); %Neumann condition
g2 = @ (x, y) exp(x+y); % Dirichlet boundary condition
exact = @ (x, y) exp(x+y); %Exact solution
parnol = @ (r, k) –(2*log(r)+1)/(k*k)–4./((k∧4)*r.∧2)...
                   +(4*besselk(1,k.*r))./((k∧3)*r);
parsol = @ (r, k) –(r.∧2.*log(r)/k∧2)–(4*(log(r)+1)/k∧4)...
                   –(4*besselk(0,k*r)/k∧4);
bdpt = [DB; NB]; nx = NV(:,1); ny = NV(:,2);
lamda = 5; %wave number
rad = 3; %radius of souce circle
N = length(bdpt); %# of boundary points
Neum = length(NB); %# of Neumann boundary points
%————MPS——————
%Generate 21x21 mesh points covering the domain
x1 = min(bdpt(:,1)); x2 = max(bdpt(:,1));
y1 = min(bdpt(:,2)); y2 = max(bdpt(:,2));
[X,Y] = meshgrid((x2–x1)/20*(0:20),(y2–y1)/20*(0:20));
intx = reshape(X,[ ],1)–abs(x1);
inty = reshape(Y,[ ],1)–abs(y1);
unif = [intx, inty]; %Uniform mesh points
M=length(unif);
p = ones(M,3);
dnx = zeros(Neum, N); dny = dnx;
```

```
dnx1 = zeros(Neum, M); dny1 = dnx1;
r = pdist2(unif, unif); %Distance matrix
p(:, 2:3) = unif(:, 1:2);
r = r+diag(diag(r)+1);%set diagonal of r to 1
rbf = r.∧2.*log(r); %TPS
MPS = [rbf, p; p', zeros(3,3)]; %Interpolation matrix
RHS = [f(unif(:, 1), unif(:, 2)); zeros(3,1)]; %forcing term
cof1 = MPS\RHS;
alpha = cof1(1:M);
beta = cof1(M+1:M+3);
%————MFS——————
[sx,sy] = pol2cart(2*pi*(1:N)/N,rad); %Source points
source = [sx', sy'];
NBC = g1(NB(:,1),NB(:,2),NV(:,1),NV(:,2));
for i = 1:Neum
    dnx1(i,:) = nx(i).*(NB(i,1)–unif(:,1));
    dny1(i,:) = ny(i).*(NB(i,2)–unif(:,2));
end
r1 = pdist2(NB(:,1:2),unif);
F1 = parnol(r1,lamda).*(dnx1+dny1)*alpha;
F1 = F1–1/lamda∧2*(beta(2)*nx+beta(3)*ny);
NBC = NBC–F1; % Update Neumann boundary condition
r1 = pdist2(DB(:,1:2),unif);
%particular solutions at the Dirichlet boundary points
F2 = parsol(r1,lamda)*alpha;
F2 = F2–(beta(1)+DB(:,1)*beta(2)+DB(:,2)*beta(3))/lamda∧2;
%Update Dirichlet boundary condition
DBC = g2(DB(:,1), DB(:,2))–F2;
BC = [NBC;DBC]; %Updated boundary conditions
rN = pdist2(NB(:,1:2), source);
rD = pdist2(DB(:,1:2),source);
for i = 1:Neum
    dnx(i,:) = nx(i).*(NB(i,1)–source(:,1));
    dny(i,:) = ny(i).*(NB(i,2)–source(:,2));
end
Neum = (–lamda./rN).*besselk(1,lamda.*rN).*(dnx+dny);
Dirch = besselk(0,lamda.*rD);
MFS = [Neum; Dirch]; %MFS matrix
cof2 = MFS\BC;
%———Test accuracy at the test points————-
```

```
rh = pdist2(test,source);
testuh = besselk(0,lamda*rh)*cof2; %homogeneous solution
rp = pdist2(test,unif);
testup = parsol(rp,lamda)*alpha;
pol=-(beta(1)+test(:,1)*beta(2)+test(:,2)*beta(3))/lamda^2;
testup = testup+pol; %particular solution
uN = testuh+testup; %approximate solution
uA = exact(test(:,1),test(:,2)); %exact solution
RMSE = sqrt(mse(uN-uA)); %RMSE error
fprintf('The Root Mean Square Error is: %e\n',RMSE)
```

Alternative approach:

Instead of using particular solution in (5.7.15) using TPS as the basis function, we can using the inverse MQ as the basis function and Laplacian as the differential operator. Let

$$u_p \simeq \sum_{i=1}^{N} \alpha_i \psi$$

where $\Delta\psi = \phi$. As a result, (5.7.19) can be written as

$$\sum_{i=1}^{N} \alpha_i \left(\phi - 25\psi\right) = -23\exp(x+y).$$

If we choose $\phi = 1/\sqrt{r^2+c^2}$, then $\psi = \sqrt{r^2+c^2} - c\ln(c+\sqrt{r^2+c^2})$ as shown in Section 5.5.1. The above problem can be solved by slightly modify Program 5.5.21. Based on the forcing term and boundary conditions, lines 6–9 have to be changed accordingly. We add a line after line 13 with "lamda = 5". For the rest of the code, we only need to make the following changes:

Line 26: A = 1./mq(DM1,c)–lamda^2*mps(mq,c);
Line 41: B1 = besselk(0,lamda*DM2);
Line 46: B2 = –B2.*besselk(1,lamda*DM3)./DM3*lamda;
Line 52: uh = besselk(0,lamda*DM4)*b;

Using the modified program and the same data in Program 5.7.19 and the shape parameter of inverse MQ $c = 1.1$, we can achieve the RMSE error $7.37E-7$ which is two order of magnitude more accurate than the the result using TPS. Note that if we apply the particular solution derived in the next section using polyharmonic splines, we will expect to obtain much better accuracy than using TPS.

5.7.3 Polyharmonic Splines in 2D

This argument can be extended to find particular solutions for $\Delta - \lambda^2$ [MGC99] when $\phi^{(n)}$ is a polyharmonic spline

$$\phi^{(n)}(r) = \begin{cases} r^{2n} \ln r, & \text{in 2D}, \\ r^{2n-1}, & \text{in 3D}. \end{cases}$$

Since $\Delta_r^{n+1} \phi^{(n)} = 0, r \neq 0$, we find that particular solutions can be found by solving

$$\Delta_r^{n+1} \left(\Delta_r - \lambda^2 \right) \psi = 0.$$

As before, we have

$$\psi = \Theta + \Psi$$

where

$$\left(\Delta_r - \lambda^2 \right) \Theta = 0, \text{ and } \Delta_r^{n+1} \Psi = 0.$$

Since Δ_r^{n+1} is a multiple of an Euler operator, we look for solutions of $\Delta_r^{n+1}\Psi$ in the form $\Psi = r^q$ where q is the characteristic exponent. Using $\Delta_r^{n+1} r^q = q^2 r^{q-2}$ repeatedly, we get

$$\Delta_r^{n+1} r^q = q^2 (q-2)^2 \cdots (q-2n)^2 r^{q-2n}.$$

Hence the characteristic polynomial of the differential equation is

$$P(r) = r^2 (r-2)^2 \cdots (r-2n)^2 = 0,$$

and the characteristic exponents are $q = 0, 2, 4, \ldots, 2n$. Since the roots are double, the theory of the Euler equation shows that the solution space W is spanned by $\left\{ r^{2k}, r^{2k} \ln r \right\}, 0 \leq k \leq n$ $(\dim W = 2n + 2)$ [MGC99] so that

$$\psi(r) = AI_0(\lambda r) + BK_0(\lambda r) + \sum_{k=1}^{n+1} c_k r^{2k-2} \ln r + \sum_{k=1}^{n+1} d_k r^{2k-2}. \quad (5.7.21)$$

We now need to choose $A, B, \{c_k\}, \{d_k\}$ so that $\left(\Delta - \lambda^2 \right) \psi = r^{2n} \ln r$ with ψ having the maximum differentiability. Here we can choose $A = 0$.

By some tedious algebraic derivations, one obtains

$$\begin{aligned}
B &= -\frac{(2n)!!^2}{\lambda^{2n+2}} \\
c_k &= -\frac{(2n)!!^2}{(2k-2)!!^2}\lambda^{2k-2n-4}, \quad 1 \le k \le n+1, \\
d_k &= c_k \sum_{j=k}^{n} \frac{1}{j}, \qquad\qquad\qquad 1 \le k \le n.
\end{aligned}$$

Here $m!!$ denotes the product of all positive integers that are less than or equal to m and have the same remainder as m when divided by 2, e.g. $5!! = 1 \times 3 \times 5, 8!! = 2 \times 4 \times 6 \times 8$. Furthermore, by conventional notation, $2!! = 2, 0!! = 1, 1!! = 1$, and $(-1)!! = 1$. Since A is arbitrary, it can be chosen as zero. We refer readers to [MGC99] for detailed derivation.

From a theoretical point of view, we should prefer to use splines of highest possible order due to their higher convergence rate. However, from a computational point of view, ill–conditioning will pose limitations for using high–order splines. As a result, within machine precision, we try to push the order of the splines as high as possible within reasonable stability limits.

By our construction, the singularities in equations (5.7.21) have been nicely canceled out. To be more specific, (5.7.21) can be expanded and rewritten as

$$\psi(r) = B\sum_{k=0}^{\infty} a_k r^{2k} - B\sum_{k=n+1}^{\infty} \frac{\lambda^{2k}}{(2k)!!^2} r^{2k} \ln r + \sum_{k=1}^{n} d_k r^{2k-2}. \quad (5.7.22)$$

From (5.7.22) we notice that $\psi \in C^{2n+1}$. Furthermore, for computational purposes, we prefer to use (5.7.22) for small r, since only a few terms in the summation are required. For larger r, we switch to equation (5.7.21) by using a library subroutine to compute K_0.

5.7.4 Thin–Plate Splines in 3D

The derivation of a particular solution for $\Delta - \lambda^2$ in the 3D case is similar to the 2D case [MGC99]. We start with the thin–plate spline $\phi = r$ as the basis function, which is the fundamental solution of the biharmonic equation in $I\!R^3$. We need to solve

$$\left(\Delta - \lambda^2\right)\psi = r.$$

As in the 2D case, we have $\Delta^2 r = 0$ so that

$$\Delta^2\left(\Delta - \lambda^2\right)\psi = 0.$$

As before, radially symmetric solutions ψ can be obtained by solving the following ordinary differential equation

$$\Delta_r^2\left(\Delta_r - \lambda^2\right)\psi = 0$$

where

$$\Delta_r = \frac{1}{r^2}\frac{d}{dr}\left(r^2\frac{d}{dr}\right) = \left(\frac{d^2}{dr^2} + \frac{2}{r}\frac{d}{dr}\right).$$

Once again, we split ψ into the solutions of two homogeneous equations

$$\psi = \Theta + \Psi$$

where

$$\left(\Delta_r - \lambda^2\right)\Theta = 0 \quad \text{and} \quad \Delta_r^2\Psi = 0. \tag{5.7.23}$$

To solve the first part of (5.7.23), we make the change of variable

$$\Psi = \frac{z}{r}.$$

Then z satisfies

$$\frac{d^2 z}{dr^2} - \lambda^2 z = 0,$$

whose general solution is

$$z = Ae^{-\lambda r} + Be^{\lambda r}$$

so that

$$\Psi = \frac{Ae^{-\lambda r}}{r} + \frac{Be^{\lambda r}}{r}.$$

To solve the second part of (5.7.23), we note that it is equivalent to the fourth order Euler equation

$$r^4\frac{d^4\Theta}{dr^4} + 4r^3\frac{d^3\Theta}{dr^3} = 0 \tag{5.7.24}$$

whose characteristic polynomial is $P(r) = r(r-1)(r+1)(r-2)$. Hence, the general solution of (5.7.24) is

$$\Theta = a + br + \frac{c}{r} + dr^2.$$

Thus we superimpose everything into

$$\psi = \frac{Ae^{-\lambda r}}{r} + \frac{Be^{\lambda r}}{r} + a + br + \frac{c}{r} + dr^2$$

where the coefficients are determined by the condition $\left(\Delta_r - \lambda^2\right)\psi = r$. Doing this gives

$$a = d = 0, \quad b = \frac{-1}{\lambda^2}, \quad c = \frac{-2}{\lambda^4}$$

so that

$$\psi = \frac{Ae^{-\lambda r}}{r} + \frac{Be^{\lambda r}}{r} - \frac{r}{\lambda^2} - \frac{2}{\lambda^4 r}.$$

Since A and B are arbitrary, we choose $A = 2/\lambda^4, B = 0$ to get a continuous particular solution at $r = 0$. Doing this yields

$$\psi = \begin{cases} -\dfrac{r}{\lambda^2} - \dfrac{2}{\lambda^4 r} + \dfrac{2e^{-\lambda r}}{\lambda^4 r}, & r \neq 0, \\ \dfrac{-2}{\lambda^3}, & r = 0. \end{cases}$$

5.7.5 Polyharmonic Splines in 3D

Similar to the 2D case, if we consider polyharmonic splines $\phi = r^{2n-1}$, we have to solve $\left(\Delta_r - \lambda^2\right)\Theta = 0$ and $\Delta_r^{n+1}\Theta = 0$, since $\Delta_r^{n+1} r^{2n-1} = 0$. It is easily shown that the solution to $\left(\Delta_r - \lambda^2\right)\Theta = 0$ is

$$\Theta(r) = A\frac{\cosh(\lambda r)}{r} + B\frac{\sinh(\lambda r)}{r}.$$

As in $I\!R^2$, the remaining problem is to solve $\Delta_r^{n+1}\Theta = 0$. Again one can show that this is a multiple of an Euler equation. Thus we look for solutions of the form $\Theta = r^p$. Since $\Delta_r r^p = (p+1)pr^{p-2}$, we find by repeated differentiation that the characteristic polynomial is

$$P(r) = (r+1)r(r-1)\cdots(r-2n) = 0.$$

Thus the characteristic exponents are $p = -1, 1, 2, ..., 2n$ so the solution is spanned by $\left\{r^k\right\}, -1 \leq k \leq 2n$. Hence the particular solution ψ of $(\Delta_r - \lambda^2)\psi = r^{2n-1}$ is of the form

$$\psi(r) = A\frac{\cosh(\lambda r)}{r} + B\frac{\sinh(\lambda r)}{r} + \sum_{k=-1}^{2n} a_k r^k.$$

To obtain solutions which are regular at $r = 0$ we use the Taylor series expansions of $\cosh(\lambda r)$ and $\sinh(\lambda r)$ at $r = 0$. Comparing coefficients gives

$$\begin{array}{rclrcl} B & = & 0, & A & = & \dfrac{(2n)!}{\lambda^{2n+2}}, \\ a_{2k} & = & 0, & a_{2k-1} & = & \dfrac{-(2n)!}{(2k)!\lambda^{2n+2k+2}}, \quad 0 \le k \le n. \end{array}$$

Thus

$$\psi(r) = \frac{(2n)!\cosh(\lambda r)}{r\lambda^{2n+3}} - \sum_{k=0}^{n} \frac{(2n)!}{(2k)!} \frac{r^{2k-1}}{\lambda^{2n-2k+2}} \tag{5.7.25}$$

is the particular solution of $\Delta - \lambda^2$ corresponding to $\phi = r^{2n-1}$.

Similar to (5.7.22) for small r, (5.7.25) can be expanded and rewritten as

$$\psi(r) = \sum_{k=n+1}^{\infty} \frac{(2n+1)!}{(2k+1)!} \frac{r^{2k}}{\lambda^{2n-2k+2}}$$

showing the smoothness properties.

5.7.6 Particular Solutions for $\Delta + \lambda^2$ in 2D and 3D

The remaining cases for Helmholtz–type operators $\Delta + \lambda^2$ in $I\!R^2$ and $I\!R^3$ can be obtained using an analysis similar to that above. The detailed derivation can be found in [MGC99]. We quote the results which the reader can verify by differentiation.

For $\Delta + \lambda^2$ in $I\!R^2$ and $\phi = r^2 \ln r$, we find

$$\psi(r) = \begin{cases} \dfrac{2}{\lambda^4}\left(\pi Y_0(\lambda r) - 2\ln r\right) + \dfrac{r^2 \ln r}{\lambda^2} - \dfrac{4}{\lambda^4}, & r > 0, \\ \dfrac{4}{\lambda^4}\left(\gamma + \ln\left(\dfrac{\lambda}{2}\right)\right) - \dfrac{4}{\lambda^4}, & r = 0. \end{cases}$$

If we choose polyharmonic splines $\phi = r^{2n} \ln r$, we have

$$\psi(r) = BY_0(\lambda r) + \sum_{k=1}^{n+1} c_k r^{2k-2} \ln r + \sum_{k=1}^{n} d_k r^{2k-2} \tag{5.7.26}$$

where

$$\begin{aligned}
B &= \frac{\pi}{2}\frac{(-1)^{n+1}(2n)!!^2}{\lambda^{2n+2}}, \\
c_k &= \frac{(-1)^{n+k+1}(2n)!!^2\lambda^{2k-2n-4}}{(2k-2)!!^2}, \quad 1 \le k \le n+1, \\
d_k &= c_k\sum_{j=k}^{n}\frac{1}{j}, \qquad\qquad 1 \le k \le n,
\end{aligned}$$

where Y_0 is the Bessel function of second kind of order 0. For small r, (5.7.26) can be expanded and rewritten as

$$\psi(r) = B\sum_{k=0}^{\infty} a_k r^{2k} + \frac{2B}{\pi}\sum_{k=n+1}^{\infty}\frac{(-1)^k\lambda^{2k}}{(2k)!!^2}r^{2k}\ln r + \sum_{k=1}^{n} d_k r^{2k-2}.$$

For $\Delta + \lambda^2$ in $I\!R^3$ and $\phi = r$, we have

$$\psi = \begin{cases} \dfrac{r}{\lambda^2} + \dfrac{2(\cos(\lambda r) - 1)}{\lambda^4 r}, & r \neq 0, \\ \dfrac{r}{\lambda^2}, & r = 0. \end{cases}$$

Similarly, for polyharmonic splines $\phi = r^{2n-1}$ we have

$$\psi(r) = \frac{(-1)^{n+1}(2n)!}{r\lambda^{2n+2}}\cos(\lambda r) + \sum_{k=0}^{n}\frac{(2n)!}{(2k)!}\frac{(-1)^{n+k}r^{2k-1}}{\lambda^{2n-2k+2}}.$$

For small r, $\psi(r)$ can be expanded and rewritten as

$$\psi(r) = \sum_{k=n+1}^{\infty}\frac{(2n+1)!}{(2k+1)!}\frac{(-1)^{n+k+1}r^{2k}}{\lambda^{2n-2k+2}}.$$

5.7.7 Polynomial Basis Functions

Since polyharmonic splines are conditionally positive definite as explained in Section 1.2, additional polynomial terms of the form (1.2.5) are required to assure the solvability of the resulting matrix system.

Theorem 5.7.27 *Assume $\varepsilon = \pm 1$ and consider the Helmholtz operator $\Delta + \varepsilon\lambda^2$ in $I\!R^2$. A particular solution for*

$$\Delta\psi + \varepsilon\lambda^2\psi = x^n y^m \tag{5.7.28}$$

for nonnegative integers m, n is given by

$$\psi(x,y) = \sum_{k=0}^{\lfloor \frac{m}{2} \rfloor} \sum_{\ell=0}^{\lfloor \frac{n}{2} \rfloor} \frac{\varepsilon\,(-\varepsilon)^{k+\ell}\,(k+\ell)!m!n!x^{m-2k}y^{n-2\ell}}{\lambda^{2k+2\ell+2}k!\ell!(m-2k)!(n-2\ell)!}.$$

where $\lfloor t \rfloor$ is the largest integer that is less than or equal to t.

Proof See [MCGC00]. □

Example 5.7.29 Consider the following 2D modified Helmholtz differential equation

$$\Delta\psi - 16\psi = x^2y^5. \tag{5.7.30}$$

In (5.7.28), $\epsilon = -1, \lambda = 4, m = 2, n = 5, \lfloor \frac{m}{2} \rfloor = 1$ and $\lfloor \frac{n}{2} \rfloor = 2$. Thus a particular solution u is given by

$$\begin{aligned}\psi &= -\sum_{k=0}^{1}\sum_{\ell=0}^{2} \frac{2!5!(k+\ell)!x^{2-2k}y^{5-2\ell}}{4^{2k+2\ell+2}k!\ell!(2-2k)!(5-2\ell)!} \\ &= -\frac{1}{16}x^2y^5 - \frac{5}{64}x^2y^3 - \frac{15}{512}x^2y - \frac{1}{128}y^5 - \frac{5}{256}y^3 - \frac{45}{4096}y.\end{aligned}$$

It can be easily checked that ψ obtained above satisfies the modified Helmholtz differential equation in (5.7.30). Note that the above calculation can be obtained effortlessly by symbolic computation. □

We now provide a somewhat more general and direct approach to obtain particular solutions of Helmholtz equations for polynomial forcing terms [CLH07]. Suppose that f is a polynomial of degree m and we consider solving the following modified Helmholtz equation

$$\Delta u_p - \lambda^2 u_p = f(x,y) \tag{5.7.31}$$

for a particular solution u_p and $\lambda \in \mathbb{C} \setminus \{0\}$. The above expression can be written in the following form

$$\left(\Delta - \lambda^2 I\right) u_p = f(x,y), \tag{5.7.32}$$

where I is the identity operator. We observe that $\Delta^{L+1}f(x,y) = 0$ for some positive integer L, and then

$$\left(I - \left(\frac{\Delta}{\lambda^2}\right)^{L+1}\right) f(x,y) = f(x,y). \tag{5.7.33}$$

By geometric series expansion, we have

$$I - \left(\frac{\Delta}{\lambda^2}\right)^{L+1} = \left(I - \frac{\Delta}{\lambda^2}\right)\sum_{k=0}^{L}\left(\frac{\Delta}{\lambda^2}\right)^k. \qquad (5.7.34)$$

From (5.7.33) and (5.7.34), we obtain

$$\left(I - \frac{\Delta}{\lambda^2}\right)\sum_{k=0}^{L}\left(\frac{\Delta}{\lambda^2}\right)^k f(x,y) = f(x,y). \qquad (5.7.35)$$

On the other hand, (5.7.32) can be re-written as

$$-\lambda^2\left(I - \frac{\Delta}{\lambda^2}\right)u_p(x,y) = f(x,y). \qquad (5.7.36)$$

Comparing (5.7.35) and (5.7.36), it follows that

$$-\lambda^2 u_p(x,y) = \sum_{k=0}^{L}\left(\frac{\Delta}{\lambda^2}\right)^k f(x,y).$$

Consequently, we have

$$u_p(x,y) = \frac{-1}{\lambda^2}\sum_{k=0}^{L}\left(\frac{\Delta}{\lambda^2}\right)^k f(x,y). \qquad (5.7.37)$$

Similarly, the particular solution for the Helmholtz equation

$$\Delta u_p + \lambda^2 u_p = f(x,y)$$

can be obtained as

$$u_p = \sum_{k=0}^{L}(-1)^k \frac{\Delta^k}{\lambda^{2k+2}} f(x,y).$$

Example 5.7.38 Consider the following modified Helmholtz equation

$$\Delta u_p - 9u_p = x^2y^3 + x^2y^5 + 3xy.$$

Notice that

$$\begin{aligned}
\Delta\left(x^2y^3 + x^2y^5 + 3xy\right) &= 2y^3 + 2y^5 + 6x^2y + 20x^2y^3,\\
\Delta\left(2y^3 + 2y^5 + 6x^2y + 20x^2y^3\right) &= 24y + 80y^3 + 120x^2y,\\
\Delta\left(24y + 80y^3 + 120x^2y\right) &= 720y,
\end{aligned}$$

From (5.7.37), we obtain the particular solution

$$\begin{aligned} u_p &= -\frac{1}{3^2}\left(x^2y^3+x^2y^5+3xy\right)-\frac{1}{3^4}\left(2y^3+2y^5+6x^2y+20x^2y^3\right) \\ &\quad -\frac{1}{3^6}\left(24y+80y^3+120x^2y\right)-\frac{1}{3^8}\left(720y\right) \\ &= -\frac{29}{81}x^2y^3-\frac{1}{9}x^2y^5-\frac{1}{3}xy-\frac{98}{729}y^3-\frac{2}{81}y^5-\frac{58}{243}x^2y-\frac{104}{729}y. \end{aligned}$$

□

Theorem 5.7.39 *Let $\varepsilon = \pm 1$ and assume that p, q and r are positive integers. A particular solution for the Helmholtz problem*

$$\Delta\psi + \varepsilon\lambda^2\psi = x^p y^q z^r,$$

in $I\!R^3$ is given by

$$\psi(x,y,z)=\sum_{j=0}^{\lfloor \frac{p}{2}\rfloor}\sum_{k=0}^{\lfloor \frac{q}{2}\rfloor}\sum_{\ell=0}^{\lfloor \frac{r}{2}\rfloor}\frac{\varepsilon\left(-\varepsilon\right)^{k+\ell}(j+k+\ell)!p!q!r!x^{p-2j}y^{q-2k}z^{r-2\ell}}{\lambda^{2j+2k+2\ell+2}j!k!\ell!(p-2j)!(q-2k)!(r-2\ell)!}$$

where $\lfloor t\rfloor$ is the largest integer that is less than or equal to t.

Proof See [GMCC03]. □

Example 5.7.40 Consider the following interior Dirichlet problem for the modified Helmholtz equation [MGC99]

$$\begin{aligned} \left(\Delta-\lambda^2\right)u &= (e^x+e^y)(1-\lambda^2), && (x,y)\in\Omega, \\ u &= e^x+e^y, && (x,y)\in\Gamma, \end{aligned} \qquad (5.7.41)$$

where $\Omega\cup\Gamma=[0,1]^2$.

The exact solution is given by

$$u^* = e^x+e^y, \quad (x,y)\in\Omega\cup\Gamma.$$

To calculate an approximate particular solution $\tilde{u}_p$, we chose polyharmonic splines as the basis function to interpolate the forcing term in (5.7.41). In order to do so, we took 36 uniform grid points on the unit square. Furthermore, we used $\lambda^2 = 25$.

To approximate the homogeneous solution, we used the method of fundamental solutions. We chose 16 uniformly distributed collocation points on Γ and the same number of source points on the fictitious boundary which is a circle with center at $(0,0)$ and radius 8.

Then we observed the absolute errors of approximate solutions $\tilde{u}$ along the line $\{(x, 0.4) : 0 \leq x \leq 1\}$. In Figure 5.10, we show the absolute error of $\tilde{u}$ in a logarithmic scale using polyharmonic splines of order 1 through 5. In the figure the solution with polyharmonic splines of order * is denoted by $S*$.

The numerical results in Figure 5.10 show the accuracy of the higher order splines which improves from thin plate splines (TPS) up to three orders of magnitude. In Figure 5.11, we show the profile of the overall relative errors using polyharmonic splines of order 4. We observe the high numerical accuracy with a maximum relative error within 7×10^{-7}. Comparing with thin–plate splines, the results are remarkable with very little additional computational cost. $\Box$

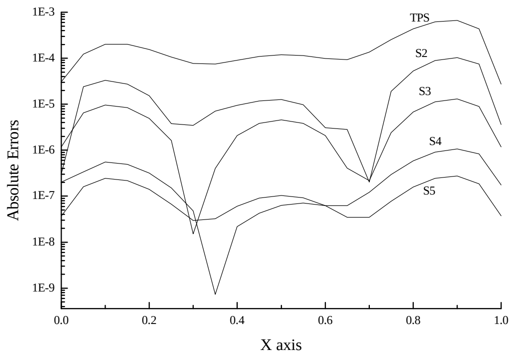

Fig. 5.10. Absolute errors of u along the line $y = 0.4$ using polyharmonic splines of various orders.

5.7.8 Compactly Supported RBFs in 3D

In this section, we focus on the indirect method for derivation of particular solutions ψ to compactly supported radial basis functions ϕ in 3D [GCG00]. Unfortunately, no explicit results are known for the 2D case, but fortunately there still is the direct method. We shall give an example at the end of this section. Furthermore, we recall that we can handle scaling easily via (5.7.1), so that we can ignore scaling from now on.

Consider the following equation

$$\left(\Delta - \lambda^2\right) \psi(r) = \phi(r)$$

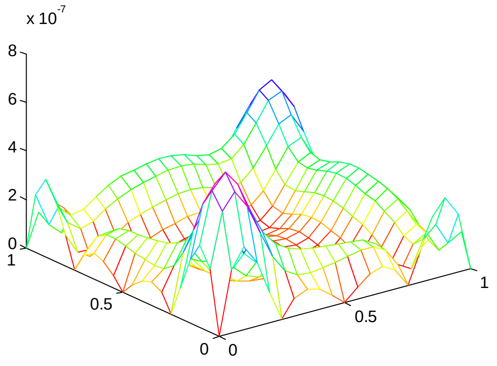

Fig. 5.11. Profile of relative errors for u using splines of order 4.

where ϕ is a Wendland–type compactly supported function with support in $[0,1]$. This is equivalent to solving

$$\frac{1}{r^2}\frac{d}{dr}(r^2\frac{d\psi}{dr}) - \lambda^2\psi = \begin{cases} (1-r)^n\, p\,(r), & 0 \le r \le 1, \\ 0, & r > 1, \end{cases} \tag{5.7.42}$$

where p is an appropriately chosen polynomial of fixed degree $k \ge 0$ so that ϕ is a C^{2k} compactly supported radial basis function [GCG00]. For $r = 0$, (5.7.42) is to be considered in the limiting case as $r \to 0^+$. If we let

$$\psi(r) = \frac{z(r)}{r}, \quad r > 0, \quad \psi^j(0) = \lim_{r\to 0^+}\frac{d^j}{dr^j}\left(\frac{z}{r}\right), \quad j = 0,1,2, \tag{5.7.43}$$

then

$$\frac{1}{r^2}\left(\frac{d}{dr}r^2\frac{d\psi}{dr}\right) = \frac{1}{r}\frac{d^2z}{dr^2},$$

and hence (5.7.42) is equivalent to

$$\frac{d^2z}{dr^2} - \lambda^2 z = \begin{cases} r\,(1-r)^n\, p\,(r), & 0 \le r \le 1, \\ 0, & r > 1. \end{cases} \tag{5.7.44}$$

By elementary arguments from ordinary differential equations, the general solution of the above equation is of the form

$$z(r) = \begin{cases} Ae^{-\lambda r} + Be^{\lambda r} + q(r), & 0 \le r \le 1, \\ Ce^{-\lambda r} + De^{\lambda r}, & r > 1, \end{cases} \tag{5.7.45}$$

where $q(r)$ is a particular smooth solution of the first equation in (5.7.44). The task of determining $q(r)$ by the method of undetermined coefficients could be tedious. But using modern computer symbolic ordinary differential equation solvers such as Maple© or Mathematica©, $q(r)$ can be obtained easily.

The four constants A, B, C and D in (5.7.45) are to be determined so that ψ given by (5.7.43) is at least twice continuously differentiable at $r = 0$ and $r = 1$, and hence on $[0, \infty)$. Since z is a solution of an inhomogeneous linear second–order equation with a piecewise smooth right-hand side, it is piecewise smooth everywhere. At $r = 0$ we only have to make sure that $z(0) = 0$ holds, because we want a nonsingular ψ. The additional degrees of freedom can be used to ensure smoothness at $r = 1$.

Clearly, the condition $z(0) = 0$ is

$$A + B + q(0) = 0. \tag{5.7.46}$$

Let us check the derivatives at $r = 1$ and write down the equations for smoothness. The m^{th} derivative is continuous at $r = 1$ if

$$(-1)^m \lambda^m A e^{-\lambda} + \lambda^m B e^{\lambda} + q^{(m)}(1) = (-1)^m \lambda^m C e^{-\lambda} + \lambda^m D e^{\lambda}.$$

This means

$$(-1)^m (C - A)e^{-\lambda} + (D - B)e^{\lambda} = \frac{q^{(m)}(1)}{\lambda^m}$$

for $\lambda \neq 0$. There are essentially only two different equations for odd and even m, but with infinitely many right-hand sides. There is no unique solution, and it is easy to come up with a solution for the first two equations for $m = 0$ and $m = 1$. Since

$$\begin{aligned} (D - B)e^{\lambda} &= \frac{1}{2}\left(q(1) + \frac{q'(1)}{\lambda}\right) \\ (C - A)e^{-\lambda} &= \frac{1}{2}\left(q(1) - \frac{q'(1)}{\lambda}\right), \end{aligned}$$

only $C - A$ and $D - B$ are uniquely defined, but we have the additional condition (5.7.46). Hence one of the variables can be chosen arbitrarily.

For computational efficiency, we set $D := 0$. Consequently, we get

$$\begin{cases} A = -[B + q(0)], \\ B = -\dfrac{e^{-\lambda}\left[q'(1) + \lambda q(1)\right]}{2\lambda}, \\ C = B\left(e^{2\lambda} - 1\right) + q(1)e^{\lambda} - q(0). \end{cases} \tag{5.7.47}$$

As we now know $z(r)$ explicitly, we have a particular solution of (5.7.42) given by

$$\psi(r) = \begin{cases} \lambda(2B + q(0)) + q'(0), & r = 0 \\ \dfrac{Ae^{-\lambda r} + Be^{\lambda r} + q(r)}{r}, & 0 < r \le 1, \\ \dfrac{Ce^{-\lambda r}}{r}, & r > 1, \end{cases} \tag{5.7.48}$$

where A, B and C are as in (5.7.47). Notice that for $r > 1$ and for $\lambda > 0$ and large, $\psi(r)$ in (5.7.48) decays exponentially. Higher orders of smoothness can only occur if q has additional properties. For instance, second-order smoothness needs

$$\frac{q^{(2)}(1)}{\lambda^2} = \frac{q(0)}{\lambda^0} = q(0).$$

Example 5.7.49 Let $\phi(r) = (1 - r)_+^2$. The general solution of

$$\frac{d^2 z}{dr^2} - \lambda^2 z = r\left(1 - r\right)^2, \; r \in I\!R$$

is given by

$$z\left(r\right) = -\frac{-4 + r\lambda^2 + 6r - 2r^2\lambda^2 + r^3\lambda^2}{\lambda^4} + Ae^{-\lambda r} + Be^{\lambda r}.$$

Hence $q(r),\ 0 \le r \le 1$, in (5.7.45) is given by

$$q(r) = \frac{4}{\lambda^4} - \frac{1}{\lambda^2}r - \frac{6}{\lambda^4}r + \frac{2}{\lambda^2}r^2 - \frac{1}{\lambda^2}r^3$$

and

$$\frac{d}{dr}q(r) = -\frac{\lambda^2 + 6 - 4r\lambda^2 + 3r^2\lambda^2}{\lambda^4}.$$

Since

$$q(0) = \frac{4}{\lambda^4}, \qquad q(1) = -\frac{2}{\lambda^4},$$

$$\frac{dq}{dr}(0) = -\frac{\lambda^4+6}{\lambda^4}, \qquad \frac{dq}{dr}(1) = -\frac{6}{\lambda^4},$$

using (5.7.47) we get

$$\begin{aligned} A &= -\frac{e^{-\lambda}(3+\lambda)+4\lambda}{\lambda^5}, \\ B &= \frac{e^{-\lambda}(3+\lambda)}{\lambda^5} \\ C &= \frac{3e^{\lambda}-e^{-\lambda}(3+\lambda)-\lambda(4+e^{\lambda})}{\lambda^5}. \end{aligned}$$

Using the above values in (5.7.48), we get ψ. □

Example 5.7.50 Let $\phi(r) = (1-r)_+^4\,(4r+1)$. In this case, as in the previous example we can show that

$$\begin{aligned} q(r) &= -\frac{480}{\lambda^6} - \frac{2880}{\lambda^8} + r\left(\frac{60}{\lambda^4}+\frac{1800}{\lambda^6}-\frac{1}{\lambda^2}\right) + r^2\left(-\frac{240}{\lambda^4}-\frac{1440}{\lambda^6}\right) \\ &+ r^3\left(\frac{10}{\lambda^2}+\frac{300}{\lambda^4}\right) - r^4\left(\frac{20}{\lambda^2}+\frac{120}{\lambda^4}\right) + \frac{15}{\lambda^2}r^5 - \frac{4}{\lambda^2}r^6. \end{aligned}$$

Hence A, B, C and the solution ψ can now be easily computed using (5.7.47) and (5.7.48). □

Example 5.7.51 Consider the following Helmholtz problem in three dimensions [GCG00]

$$\begin{aligned} (\Delta - 400)u &= -\frac{397}{400}e^{x+y+z}, \quad (x,y,z) \in \Omega, \\ u &= \frac{e^{x+y+z}}{400}, \qquad\quad (x,y,z) \in \Gamma, \end{aligned} \tag{5.7.52}$$

where the physical domain Ω is two connected spheres in $I\!R^3$ (see Figure 5.12); i.e. ,

$$\Omega = \left\{(x,y,z) \in I\!R^3 : H(x,y,z) < 1\right\}$$

with

$$H(x,y,z) = \min\left\{\left(x-\frac{3}{4}\right)^2, \left(x+\frac{3}{4}\right)^2\right\} + y^2 + z^2.$$

To construct an approximate particular solution, 400 quasi-random points were generated to serve as the interpolation points in $\Omega \cup \Gamma$. The

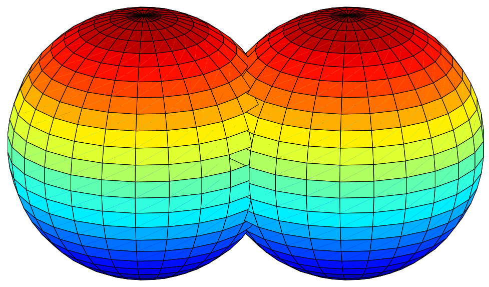

Fig. 5.12. The solution domain Ω.

CS-RBF $\phi_c = (1 - r/c)_+^2$ was chosen as the basis function to interpolate the inhomogeneous term in (5.7.52). The sparseness of the interpolation matrix depends on the scaling factor c. To approximate the homogeneous solution, the method of fundamental solutions with 100 uniformly distributed collocation points on the surface of the physical domain was employed. The same number of source points on the fictitious surface which is a sphere with radius 10 and center $(0, 0)$ was chosen.

The L_∞ norm error was computed at 500 random points in Ω for various choices of the scaling factor c. The number nz of nonzero elements and the fill–in (i.e. percentage of nonzero matrix elements) of the interpolation matrix, L_∞ norm, and the corresponding computing time are shown in Table 5.8. The numerical results shown here are especially encouraging for future work in solving a large class of time–dependent problems. □

Example 5.7.53 In this example, we consider the case where the closed form particular solution is not available for the using compactly supported radial basis functions. Let us choose the same particular solution ψ as in Example 5.5.4. Then

$$\begin{aligned}
\phi(r) &= \left(\Delta - \lambda^2\right)\psi(r) \\
&= \left(1 - \frac{r}{c}\right)_+^4 \left(\frac{112}{c^4}\left(20r^2 - 4rc - c^2\right) - \right. \\
&\qquad \left. \lambda^2 \left(1 - \frac{r}{c}\right)^6 \left(\frac{35r^2}{c^2} + \frac{18r}{c} + 3\right)\right).
\end{aligned}$$

c	nz	fill–in (%)	L_∞	CPU (sec.)
0.2	756	0.40	2.28E-2	01.40
0.4	3392	2.10	1.61E-2	04.43
0.6	10562	6.60	9.77E-3	07.48
0.8	21526	13.5	6.45E-3	10.63
1.0	36476	22.8	4.50E-3	13.71
1.2	54116	22.8	3.82E-3	16.98
1.4	73722	46.0	3.30E-3	20.53
1.6	92722	57.9	2.90E-3	24.08
1.8	110498	69.0	2.63E-3	27.60

Table 5.8. *Sparseness, error estimates and CPU time for various scaling factor c.*

We consider the following modified Helmholtz equation: :

$$\begin{aligned} \left(\Delta - \lambda^2\right) u &= \left(1 - \lambda^2\right)\left(e^x + e^y\right), \quad && (x,y) \in \Omega, \\ u &= e^x + e^y, && (x,y) \in \Gamma, \end{aligned}$$

where Ω is a unit circle. In the implementation we choose $\lambda^2 = 100$, and 150 quasi-random interpolation points were selected in the domain. To find the homogeneous solution using the method of fundamental solutions, 35 evenly distributed collocation points and the same number of source points were chosen. The fictitious boundary is a circle with radius 10 and center at the origin. The L_∞ norm errors are shown in Table 5.9.

□

5.8 Multiscale Techniques

When compactly supported radial basis functions were developed in mid-1990 [Wu95, Wen95, Buh98], they were regarded as a cure to the problems of the dense and ill–conditioned matrices arising when using traditional globally supported radial basis functions. Since then, they have been widely applied for solving various kinds of problems in science and engineering [FI96, CBP99, CMC99, SW99, CYT00a, CYT00b,

c	L_∞	c	L_∞
0.3	2.618	1.1	2.982E-2
0.4	1.711	1.2	2.415E-2
0.5	0.821	1.3	1.978E-2
0.6	0.289	1.4	1.634E-2
0.7	0.098	2.0	6.425E-3
0.8	0.051	4.0	1.228E-3
0.9	4.469E-2	8.0	3.828E-4
1.0	3.658E-2	10	2.965E-4

Table 5.9. *Absolute maximum errors for the modified Helmholtz equation*

GCG00, MYP$^+$01, WHG02b, CGGC02, WHG02a, OBS03] and numerous others to follow.

However, several difficulties of compactly supported radial basis functions have been observed:

(i) the accuracy and efficiency depends on the scale of the support,
(ii) there is no theoretical background for determining the optimal scale of the support,
(iii) the convergence rate of compactly supported radial basis functions is low due to their limited smoothness.

These effects have explanations along the lines of earlier sections on scaling and error estimates (see sections 2.6 and 2.3). In order to obtain a very sparse matrix system, the support needs to be very small, but then the interpolation error becomes unacceptable. When the support is large enough to make the error acceptable, the matrix sparsity may be too low to be acceptable to the user. As a result, the application of compactly supported radial basis functions with a fixed support requires some experience and some additional techniques. Section 2.7 presents practical rules-of-thumb for this. But one can also use compactly supported radial basis functions with a **variable support**, and this is the topic of this section.

As described in Section 2.3, the accuracy of any RBF approximation based on a set X of data locations within a domain Ω is controlled by the **fill distance** $h = h(X, \Omega)$ of (2.3.1), i.e.

$$h = h(X, \Omega) := \sup_{\mathbf{x}\in\Omega} \min_{\mathbf{y}\in X} \|\mathbf{x} - \mathbf{y}\| .$$

On the other hand, stability is dominated by the **separation distance**

$$S(X) := \frac{1}{2} \min_{\mathbf{y}\neq\mathbf{z}\in X} \|\mathbf{y} - \mathbf{z}\|$$

following Section 2.4. Ideally, the distribution of data locations in X should be **quasi–uniform** as defined in (2.4.1) in order not to have spurious near-duplicate points that spoil stability.

But both the fill and the separation distance should be closely tied to the **scale factor** c of the underlying radial basis function, in particular in case of compact support when c is the support radius. As in Section 2.6, we use c here in the sense

$$\phi_c(\|\mathbf{x} - \mathbf{y}\|_2) := \phi_c(\|\mathbf{x} - \mathbf{y}\|_2/c).$$

Ideally, one could link c closely to h and S by making it proportional to these quantities in a quasi-uniform setting. In the case of compactly supported radial basis functions, the ratio

$$B := \frac{c}{h}$$

controls the number of data locations in each support, and it is closely connected to the **bandwidth** of the system matrix. Increasing B will increase accuracy and computational complexity, while stability decreases.

The basic idea of **multiscale techniques** is to first recover the main features of the unknown solution on a coarse discretization level using only a few data points and applying a radial basis function with a large support. This means using large $S \approx c \approx h$ first. Then errors are evaluated, and in subsequent levels smaller details are resolved using smaller values of $S \approx c \approx h$. Thus multiscale techniques are often called **multilevel methods** or **multiresolution methods**, because they work at different levels of $S \approx c \approx h$ but with roughly fixed bandwidth B in order to improve resolution.

Multiscale methods came up in 1996 [FI96, Sch97] in the context of multivariate interpolation. They were further investigated by various authors [Fas99, NSW99, GFJ00, OBS03], and readers are referred to the book [Isk04] of Iske for details and applications. Chen et. al. [CGGC02] applied the concept of multiscale schemes to solving partial differential equations via particular solutions. Here we will briefly introduce two multiscale schemes using compactly supported radial basis functions following [CGGC02].

5.8.1 Notation

Let $X = \{\mathbf{x}_i\}_{i=1}^N$ be a large set of interpolation points. We assume that X is split into a chain of quasi–uniformly distributed point sets

$$X^1 \subset X^2 \subset \ldots \subset X^k \subset \ldots \subset X^K = X \tag{5.8.1}$$

where X^k consists of N_k points with $N_K = N$. To simplify notation, we introduce index sets $I_k \subseteq \{1, \ldots, N\}$ with $|I_k| = N_k$ and write $X^k = \{\mathbf{x}_i \ : \ i \in I_k\}$.

At this point, it is open whether the user works "bottom–up" from coarse sets $X^1 \subset X^2 \subset \ldots$ towards finer and finer sets, or if a set X large enough for final accuracy is hierarchically decomposed "top–down" into smaller and coarser sets. Both approaches arise in specific circumstances, depending e.g. whether the fine data are at the user's disposal or not.

For each level index $k = 1, \ldots, K$ there is a different scaling factor c_k, a different separation distance S_k, and a fill distance h_k which all decrease with increasing k. Ideally, the ratios of these parameters should be independent of k to ensure a fixed generalized bandwidth B for all levels, but in general the choice of the parameters depends on the required minimal accuracy of the approximation and the size and sparsity constraints of the interpolation or collocation matrix A_k at level k.

Note that special **thinning algorithms** are available [FI98, DDFI05] for maintaining quasi–uniformity efficiently when extracting coarser subsets X^{k-1} from finer sets X^k in the top–down scenario. In the bottom–up situation, new data locations are usually placed adaptively where residuals are large, and this automatically serves to avoid near–duplicate data locations, though it does not necessarily guarantee quasi–uniformity.

Figures 5.13 and 5.14 show three sets of interpolation points and their corresponding scaling factors. In these examples, we used a **quasi–Monte Carlo method** [PTVF96] to generate a sequence of quasi-random points to ensure quasi–uniformity of interpolation points at each level. In [CGGC02], two multiscale schemes were employed for solving partial differential equations. Next, we briefly summarize these approaches. We do this in the context of the Method of Particular Solutions as in the previous sections.

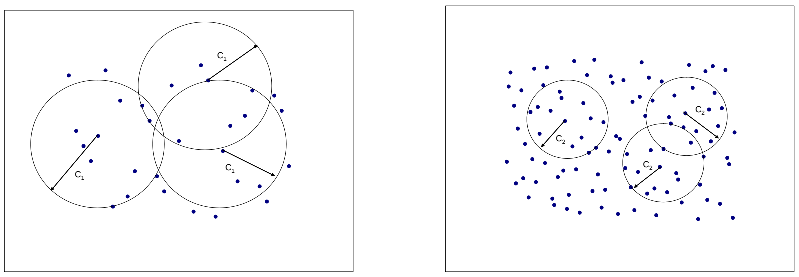

Fig. 5.13. $N_1 = 30$ with support radius c_1, $N_2 = 100$ with scale c_2, and $N_3 = 303$ with scale c_3.

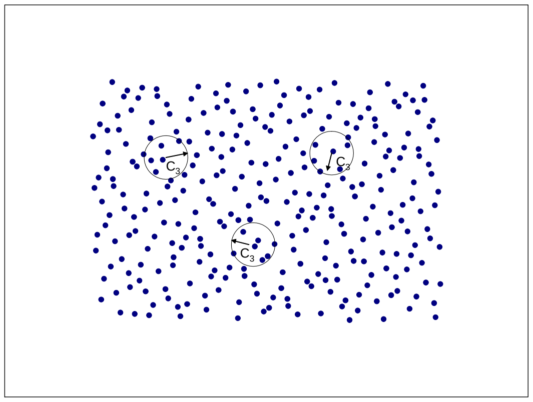

Fig. 5.14. $N_3 = 303$ with scale c_3.

5.8.2 Multiscale MPS Algorithm

For $k = 1, \ldots, K$ with c_k being the scaling factor for X^k, we let

$$\tilde{f}^k(\mathbf{x}) = \sum_{j \in I_k} \alpha_j^{(k)} \phi_{c_k} \left(\|\mathbf{x} - \mathbf{x}_j\|_2 \right) \tag{5.8.2}$$

and at level k we construct an approximate particular solution of (5.4.5) as

$$\tilde{u}_p^k(\mathbf{x}) = \sum_{j \in I_k} \alpha_j^{(k)} \psi_{c_k}(\|\mathbf{x} - \mathbf{x}_j\|_2), \qquad \mathbf{x} \in \Omega, \tag{5.8.3}$$

where $\psi_{c_k}(\|\mathbf{x} - \mathbf{x}_j\|_2)$ is a solution of

$$L\psi_{c_k}(\|\mathbf{x} - \mathbf{x}_j\|_2) = \phi_{c_k} \left(\|\mathbf{x} - \mathbf{x}_j\|_2 \right), \qquad \mathbf{x} \in \Omega, \qquad j \in I_k, \; \mathbf{x}_j \in X^k.$$

The solutions $\psi_{c_k}(\|\mathbf{x}-\mathbf{x}_j\|_2),\ j \in I_k,\ k=1,\ldots,K$, can be computed using explicit formulas such as those described in the previous sections. For $k=1$, the coefficients $\alpha_j^{(1)},\ j \in I_1$, in (5.8.2) and (5.8.3) are determined by

$$\tilde{f}^1(\mathbf{x}_i) = f(\mathbf{x}_i), \quad i \in I_1, \tag{5.8.4}$$

while for $k=2,\ldots,K$ the coefficients $\alpha_j^{(k)},\ j \in I_k$ in (5.8.2) and (5.8.3) are computed using the interpolatory constraints

$$\tilde{f}^k(\mathbf{x}_i) = f(\mathbf{x}_i) - \sum_{j=1}^{k-1} \tilde{f}^j(\mathbf{x}_i), \quad i \in I_k. \tag{5.8.5}$$

Consequently, at each level $k=1,\ldots,K$, the inhomogeneous function f is approximated by $\sum_{i=1}^{k} \tilde{f}^i$.

At the first level one chooses the support radius c_1 large and the number of points N_1 in X^1 small and obtains the unknown coefficient vector $\mathbf{a}^{(1)}$ with components $\alpha_j^{(1)},\ j \in I_1$ by solving the $N_1 \times N_1$ system

$$\mathbf{A}_{\phi_{c_1}} \mathbf{a}^{(1)} = \mathbf{f}^{(1)}$$

where $\mathbf{f}^{(1)}$ contains the values $f(\mathbf{x}_j),\ j \in I_1$.

For subsequent levels $k=2,\ldots,K$ one interpolates the residual of the previous levels. To compute the vector $\mathbf{a}^{(k)}$ with components $\alpha_j^{(k)},\ j \in I_k$, one solves the $N_k \times N_k$ system of the form

$$\mathbf{A}_{\phi_{c_k}} \mathbf{a}^{(k)} = \mathbf{f}^{(k)}$$

with the vector $\mathbf{f}^{(k)}$ consisting of the values $\tilde{f}^k(\mathbf{x}_j),\ j \in I_k$. Due to (5.8.5) and the inclusion (5.8.1), there are N_{k-1} zero entries of $\mathbf{f}^{(k)}$ corresponding to points $\mathbf{x}_i,\ i \in I_{k-1}$, and the remaining $N_k - N_{k-1}$ entries are given by (5.8.5) for points $\mathbf{x}_i$ with $i \in I_k \setminus I_{k-1}$.

As the level increases, one decreases the scaling factor c_k and increases the number of interpolation points.

The approximate particular solution $\tilde{u}_p$ of (5.4.5) can be written as

$$\tilde{u}_p = \sum_{k=1}^{K} \tilde{u}_p^k \tag{5.8.6}$$

with $\tilde{u}_p^k,\ \ k=1,\ldots,K$, given by (5.8.3).

Using the approximate particular solution (5.8.6), the final step in our algorithm is to compute an approximate solution $\tilde{u}_h$ of the associated homogeneous problem. One may apply boundary integral methods or the method of fundamental solutions to find the homogeneous solution $\tilde{u}_h$.

Finally, as in the DRM, one takes $\tilde{u} = \tilde{u}_h + \tilde{u}_p$ as an approximation to the unique solution u of (5.1.1).

If the algorithm is run bottom–up, users can monitor the full residuals

$$r_K := f - L\left(\sum_{k=1}^{K} \tilde{u}_p^k\right) = f - \sum_{k=1}^{K} \tilde{f}^k$$

and stop if a given tolerance on $\|r_K\|_\infty$ is reached. If new points for $X^{k+1} \supset X^k$ are needed, one can add points $\mathbf{y}$ where $|r_K(\mathbf{y})|$ is maximal and thus well above $\|r_K\|_\infty$ and the final tolerance. If the scaling is not changed, this will decrease the residuals under weak assumptions, as is proven in [SW00, HSZ03] on greedy algorithms. Users should select a smaller scale for level $k+1$ only if a reasonable decrease of the residuals results. If not, the point density at the old level was not large enough, and the scale should be kept.

With these precautions, the above multiscale algorithm can be run without difficulty. It ignores the additional homogeneous solution until it has approximated the particular solution up to some accuracy. On the downside, it controls only the MPS residual, not the full residual of the PDE problem. Users will have to wait through all steps of the algorithm until finally a full approximate solution to the PDE problem can be displayed.

Thus it is desirable to have a technique which allows us to display an approximate solution of the problem at each resolution level. This will be done by the method we describe in the next section.

5.8.3 Multiscale MPS–MFS Algorithm

We now reformulate the previous method in such a way that it calculates a full approximate solution to the PDE at each scale. The trade–off is some additional increase in computation costs, because the algorithm must include the homogeneous solution in each multiscale calculation.

For the starting level $k = 1$, we first calculate a particular solution $\tilde{u}_p^1$ of

$$Lu_p^1 \quad = \quad \tilde{f}^1 \quad \text{in } \Omega,$$

using the representations (5.8.3) and (5.8.4). Then $\tilde{u}_h^1$ is calculated as the approximate solution of the homogeneous problem

$$\begin{aligned} Lu_h^1 &= 0 && \text{in } \Omega, \\ u_h^1 &= g - \tilde{u}_p^1 && \text{in } \Gamma. \end{aligned} \tag{5.8.7}$$

The above homogeneous boundary value problem can be solved by using the Method of Fundamental Solutions or other boundary methods. Then we write $\tilde{v}^1 := \tilde{u}_p^1 + \tilde{u}_h^1$ and see that it approximately solves

$$\begin{aligned} Lv^1 &= \tilde{f}^1 && \text{in } \Omega, \\ v^1 &= g && \text{in } \Gamma. \end{aligned}$$

For $k = 2, \dots, K$, we write $v^k := \tilde{u}_p^k + \tilde{u}_h^k$, where $\tilde{u}_p^k$ is a particular solution of

$$Lu_p^k = \tilde{f}^k \quad \text{in } \Omega$$

computed using (5.8.3) and (5.8.5), while $\tilde{u}_h^k$ is the approximate solution of the homogeneous problem

$$\begin{aligned} Lu_h^k &= 0, && \text{in } \Omega, \\ u_h^k &= -\tilde{u}_p^k && \text{in } \Gamma. \end{aligned} \tag{5.8.8}$$

Thus the algorithm solves the problem

$$\begin{aligned} Lv^k &= \tilde{f}^k && \text{in } \Omega, \\ v^k &= 0 && \text{in } \Gamma \end{aligned}$$

approximately at each level $k > 1$.

If the steps are superimposed, we have approximate solutions to the problem

$$\begin{aligned} L\left(\sum_{k=1}^{K} v^k\right) &= \sum_{k=1}^{K} \tilde{f}^k && \text{in } \Omega, \\ \sum_{k=1}^{K} v^k &= g && \text{in } \Gamma. \end{aligned}$$

In contrast to the previous purely MPS–based multiscale method, the multiscale MPS–MFS algorithm additionally requires solving the homogeneous problems (5.8.7) and (5.8.8) at levels $k = 1, \dots, K$. Since in practice the maximum number of levels K is small, the additional computational cost involved in the multiscale MPS–MFS algorithm is justified, because the algorithm allows us to monitor a complete solution at each step. For simple test problems, the multiscale MPS algorithm should be sufficient.

Example 5.8.9 Consider the following Poisson problem [CGGC02]:

$$\begin{aligned} \Delta u &= f(x,y), \quad (x,y) \in \Omega, \\ u &= g(x,y), \quad (x,y) \in \Gamma \end{aligned} \qquad (5.8.10)$$

where $\Omega \cup \Gamma = [1,2]^2$. For testing purposes we chose f and g in such a way that the exact solution of (5.8.10) is

$$u^*(x,y) = \sin\frac{\pi x}{6}\sin\frac{7\pi x}{4}\sin\frac{3\pi y}{4}\sin\frac{5\pi y}{4}, \qquad (x,y) \in \Omega. \qquad (5.8.11)$$

The choice of u^* in (5.8.11) is possible if the boundary data $g(x,y)$ in (5.8.10) is the same as in (5.8.11) and if the inhomogeneous term $f(x,y)$ is given by

$$\begin{aligned} & f(x,y) \\ = \; & \frac{751\pi^2}{144}\sin\frac{\pi x}{6}\sin\frac{7\pi x}{4}\sin\frac{3\pi y}{4}\sin\frac{5\pi y}{4} + \frac{7\pi^2}{12}\cos\frac{\pi x}{6}\cos\frac{7\pi x}{4} \\ & \times\sin\frac{3\pi y}{4}\sin\frac{5\pi y}{4} + \frac{15\pi^2}{8}\sin\frac{\pi x}{6}\sin\frac{7\pi x}{4}\cos\frac{3\pi y}{4}\cos\frac{5\pi y}{4}. \end{aligned}$$

The profiles of the exact solution (left) and the forcing term $f(x,y)$ (right) are shown in Figure 5.15. In this example, we employed the multiscale MPS–MFS algorithm described above and chose $\varepsilon = 10^{-3}$.

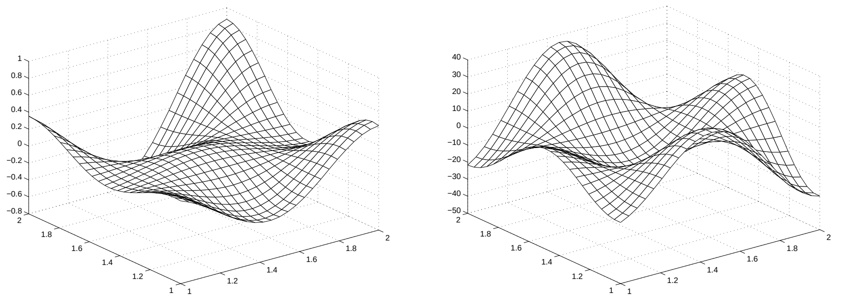

Fig. 5.15. The profiles of exact solution (left) and forcing term $f(x,y)$ (right).

To interpolate $f(x,y)$, we chose the Wendland–type compactly supported radial basis function $\varphi(r) = (1-r)_+^4\,(4r+1)$. Using the quasi-Monte Carlo based subroutine SOBSEQ [PTVF96], we generated $N = 500$ quasi-random points in the domain. Following our earlier notation, we chose four levels with scales

$$c_1 = 1.0,\; c_2 = 0.8,\; c_3 = 0.5,\; c_4 = 0.2$$

and

$$N_1 = 30,\ N_2 = 150,\ N_3 = 300,\ N_4 = 500$$

and for $k = 1, 2, 3, 4$ the sets X^k consisted of the first N_k points from the generated quasi-random points in $[1, 2]^2$.

For $k = 1, 2, 3, 4$ the sparsity structure (with non–zero entries nz_k and fill–in in %) of the resulting $N_k \times N_k$ matrix $A_{\phi_{c_k}}$ and the absolute maximum error are given in Table 5.10.

N_k	nz_k	fill–in (%)	L_∞ error
30	459	51	0.1984
150	9729	43.24	0.59E-3
300	22057	24.50	8.86E-3
500	13299	5.31	8.758E-3

Table 5.10. *Sparsity pattern of the interpolation matrix and L_∞ error*

The profiles of the interpolation error $e_k = f - \sum_{i=1}^{k} \tilde{f}^k$ of the forcing term at each level $k = 1, 2, 3, 4$ are given in Figure 5.16. The main interest is in the particular solutions obtained at each level. Their profiles are given in Figure 5.17. As expected, from Figures 5.16 and 5.17, one observes that $\|e_k\|_\infty$ and $||\tilde{u}_p^k||_\infty$ get smaller as the level increases; i.e. the scaling factor c shrinks and the number of interpolation points increases. The profiles of the solution produced at each level, $\tilde{u}_p^k + u_h^k, k = 1, 2, 3, 4,$ are shown in Figure 5.18. One notes that the contribution of the solution at levels 3 and 4 is almost insignificant. The overall errors on each level are shown in Figure 5.19 which are consistent with the solution profile in Figure 5.18. $\square$

5.8.4 Series Expansions for Differential Operators

The derivation of particular solutions for Helmholtz-type equations by solving linear ordinary differential equations as shown in Section 5.7 is somewhat tedious. In this section, we come back to the technique we used in Section 5.7.7 to derive particular solutions for polynomial right–hand sides. Coupled with symbolic software such as MATHEMATICA© or MAPLE©, this approach can be applied to more complicated operators, but it is restricted to the case that ϕ is a polyharmonic function.

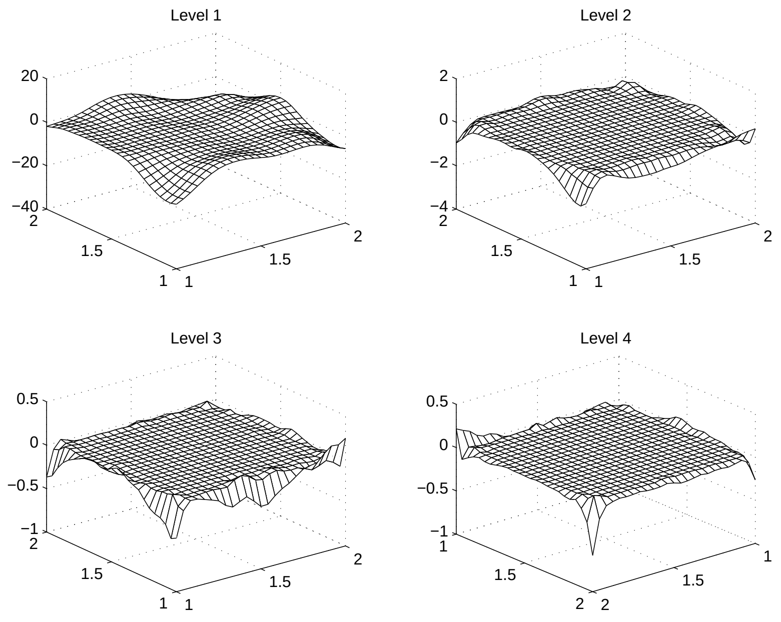

Fig. 5.16. Interpolation errors after each level.

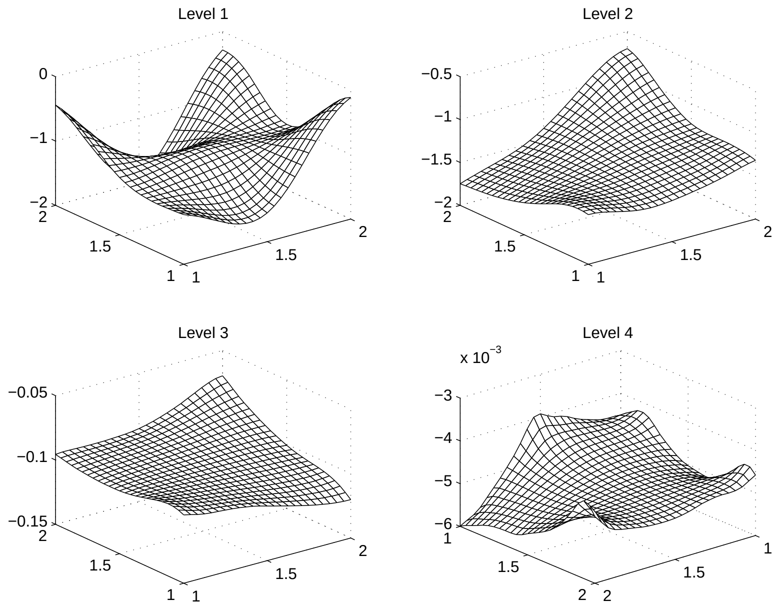

Fig. 5.17. Particular solutions produced at each level.

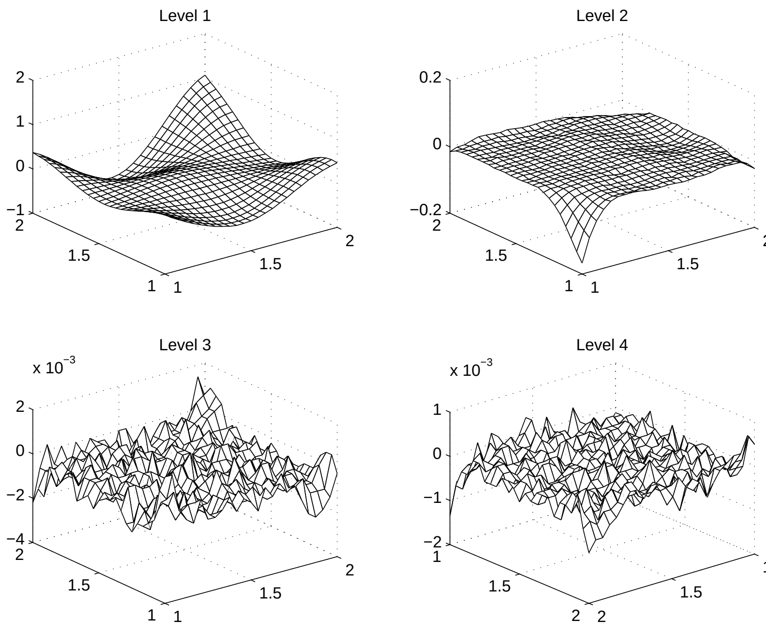

Fig. 5.18. Solution contributed at each level.

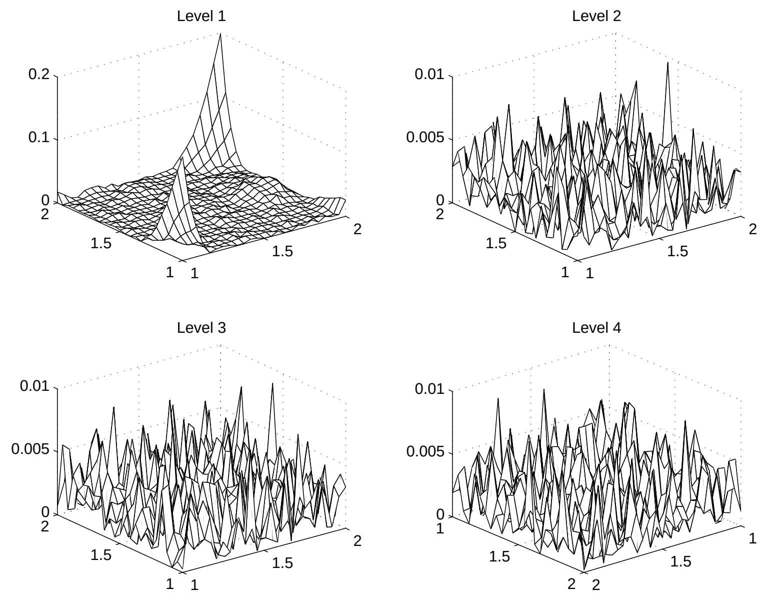

Fig. 5.19. Overall errors after each level.

From (5.7.31) and (5.7.37), a particular solution ψ of the Helmholtz equation

$$\Delta\psi - \lambda^2\psi = \phi$$

is given by

$$\psi = \frac{-1}{\lambda^2}\sum_{k=0}^{L}\left(\frac{\Delta}{\lambda^2}\right)^k \phi \tag{5.8.12}$$

where L is a sufficiently large positive integer. Notice that (5.8.12) is valid only if ϕ satisfies $\Delta^{L+1}\phi = 0$ for some positive integer L. Following and generalizing Section 5.7.2, we take fundamental solutions of the iterated Laplace operator, which are called **polyharmonic splines** (see Table 1.3). Like fundamental solutions of the Laplace operator itself, polyharmonic splines must depend on the space dimension d. In even space dimensions, they are of the form $\phi(r) = r^{2n}\ln r$ with $L+1 = (2n+d)/2$, while they are $\phi(r) = r^{2n-1}$ with $L+1 = (2n-1+d)/2$ in odd space dimensions. For illustration, let us choose $\phi = r^2\ln r$ in (5.8.12) for two dimensions. It is called the **thin–plate spline** because it is the fundamental solution of the thin–plate or biharmonic operator Δ^2 in two dimensions, leading to $L = 1$ above. The following Mathematica code can be used to produce ψ in (5.8.12) symbolically:

```
ψ[λ_]  :=  Module[ {ϕ}, ϕ = r^2 * Log[r]; g = 0; L = 20;
           For [ i = 0, i ≤ L, i++, g = ϕ + g;
           ϕ = Simplify [((1/r) * ∂_r * (r * ∂_r (ϕ))) /λ^2] ;
           If [ϕ == 0, Break[ ]] ] ;
           g = Simplify [−g/λ^2]] ;
```

In the above code, we choose $L = 20$ which is considered to be sufficiently large to handle most of the cases. Using the above Mathematica code, we obtain

$$\psi[\lambda] = -\frac{4 + \left(4 + r^2\lambda^2\right)\mathrm{Log}[r]}{\lambda^4}. \tag{5.8.13}$$

The particular solution ψ in (5.8.13) is equivalent to the last three terms in (5.7.14) which we have derived in Section 5.7.2. In (5.8.13), there is a singular term $\ln r$ that needs to be de-singularized. Hence, we still need to follow the procedure as shown in Section 5.7.2 to cancel the singularity in (5.8.13).

A similar technique for series expansions of differential operators can be applied to other types of differential operators such as $\left(\Delta - \lambda^2\right)^2$, $\Delta^2 - \lambda^4$, product of Helmholtz operators, etc. It is almost impossible to derive the particular solutions for these types of differential operators using the technique of Section 5.7.2. More specifically, let us consider the particular solution of the following equation

$$\left(\Delta^2 - \lambda^4\right)\psi = \varphi$$

which arises in solving the **Berger equation**. For $\varphi = r^2 \ln r$, it is trivial that

$$\psi = \frac{-r^2 \ln r}{\lambda^4}.$$

For $\varphi = r^4 \ln r$, we have

$$\psi = \begin{cases} \dfrac{r^4 \ln r}{\lambda^4} - \dfrac{4 \ln r + 96}{\lambda^8} - \dfrac{64 K_0(\lambda r)}{\lambda^8}, & r > 0, \\ \dfrac{1}{\lambda^8}\left(64\gamma + \ln\left(\dfrac{\lambda}{2}\right) - 96\right), & r = 0. \end{cases}$$

For $\varphi = r^6 \ln r$, we have

$$\psi = -\frac{1}{\lambda^4}\left(r^6 \ln r + \frac{1}{\lambda^4}\left(576 r^2 \ln r + 480\right)\right).$$

The fundamental solution for the operator $\Delta^2 - \lambda^4$ in 2D can be found in Table 4.1. With particular solution and fundamental solution both available, the Berger equation can be solved without much effort.

5.8.5 Particular Solutions from Fundamental Solutions

We now provide a technique for constructing particular solutions to Laplace or Helmholtz equations starting from fundamental solutions of the Helmholtz equation [AC05].

We take a fundamental solution ψ_λ of the modified Helmholtz equation, i.e.

$$\left(\Delta - \lambda\right)\psi_\lambda = -\delta \tag{5.8.14}$$

where $\lambda \in I\!R$ and

$$\psi_\lambda(r) = \begin{cases} \dfrac{1}{2\pi} K_0(\sqrt{\lambda} r), & \text{if } \lambda > 0 \\ \dfrac{-1}{2\pi} \ln(r), & \text{if } \lambda = 0 \\ \dfrac{i}{4} H_0^{(1)}(\sqrt{-\lambda} r), & \text{if } \lambda < 0 \end{cases}$$

in 2D and

$$\psi_\lambda(x) = \frac{e^{-\sqrt{\lambda}r}}{4\pi r}$$

in 3D. Here K_0 is the modified Bessel function of the second kind of order zero, and $H_0^{(1)}$ is the Hankel function of order zero. Recall that $H_0^{(1)} = J_0 + iY_0$, where J_0, Y_0 are Bessel functions of the first and second kind, respectively. The Bessel function J_0 is analytic everywhere, while K_0 and Y_0 exhibit logarithmic singular behavior at 0.

From (5.8.14) we get a particular solution to the Laplace operator via

$$\Delta\psi_\lambda = \lambda\psi_\lambda$$

and to the Helmholtz operator $\Delta - \mu$ as

$$(\Delta - \mu)\psi_\lambda = (\lambda - \mu)\psi_\lambda \tag{5.8.15}$$

except for the singularity at zero, which we have to cancel or shift away by methods of the previous sections.

Suppose we can approximate a function f using the above fundamental solutions of Helmholtz equations as basis functions as a superposition

$$f(\mathbf{x}) \simeq \tilde{f}(\mathbf{x}) = \sum_{i=1}^{p}\sum_{j=1}^{n} a_{ij}\psi_{\lambda_i}(\|\mathbf{x} - \mathbf{y}_j\|), \tag{5.8.16}$$

where $\{\lambda_1, \ldots, \lambda_p\}$ are p different nonzero real numbers and $\{\mathbf{y}_1, \ldots, \mathbf{y}_n\}$ are n different source points located on a fictitious boundary $\hat{\Gamma}$ outside the solution domain Ω.

For all $\mathbf{x} \in I\!R^d \backslash \{0\}$ we have $\Delta\psi_\lambda(\mathbf{x}) = \lambda\psi_\lambda(\mathbf{x})$ and if

$$\tilde{u}_P(\mathbf{x}) := \sum_{i=1}^{p}\sum_{j=1}^{n} \frac{a_{ij}}{\lambda_i}\psi_{\lambda_i}(\|\mathbf{x} - \mathbf{y}_j\|) \tag{5.8.17}$$

then

$$\Delta\tilde{u}_P = \tilde{f}. \tag{5.8.18}$$

By simply dividing a_{ij} by λ_i, a particular solution (5.8.18) can be obtained from (5.8.16).

To justify the approximation of any function $f \in L^2(\Omega)$ using the basis function ψ_{λ_i} in (5.8.16), we refer to the theoretical background provided by [AC01].

The following MFS→MPS algorithm implements the above technique. It works with fundamental solutions throughout and uses them to calculate particular solutions.

MFS→MPS Algorithm

(i) Choose m collocation points $\mathbf{x}_1, ..., \mathbf{x}_m$ in the domain Ω,

(ii) choose n source points $\mathbf{y}_1, ..., \mathbf{y}_n$ on the fictitious boundary $\hat{\Gamma}$,

(iii) choose p nonzero frequencies $\lambda_1, ..., \lambda_p \in I\!R$,

(iv) define the $m \times (np)$ matrix M

$$\mathbf{M} \quad := \quad (\mathbf{M_1}, \dots, \mathbf{M_p})$$

consisting of p blocks

$$\mathbf{M}_i \quad := \quad (\psi_{\lambda_i}(\|\mathbf{x}_j - \mathbf{y}_k\|))_{1 \le j \le m,\ 1 \le k \le n},\ 1 \le i \le p$$

of size $m \times n$ each,

(v) solve the $(np) \times (np)$ least-squares system

$$\mathbf{M}^T \mathbf{M} \mathbf{a} = \mathbf{M}^T \mathbf{f}\ ,$$

where $\mathbf{f} = (f(\mathbf{x}_1), \dots, f(\mathbf{x}_m))^T$.

The vector solution $\mathbf{a} = (a_{11}, \dots, a_{n1}, \dots, a_{1p}, \dots, a_{np}) \in I\!R^{np}$ will provide an approximation of f in the form

$$\tilde{f}(\mathbf{x}) = \sum_{k=1}^{p} \sum_{j=1}^{n} a_{jk} \psi_{\lambda_k}(\|\mathbf{x} - \mathbf{y}_j\|). \tag{5.8.19}$$

Since we just want to approximate a real–valued function, there is no need to consider complex fundamental solutions. We can simply use the real part of the complex fundamental solution as our basis function. For instance, in the 2D case, it will be sufficient to use J_0, which is a regular function, as the basis. As a result, this allows us to admit points $\mathbf{y}_j$ inside Ω. This means that we can take our approximations from the space

$$\text{span}\{J_0(\sqrt{-\lambda}\|\mathbf{x} - \mathbf{y}\|)|_\Omega : \mathbf{y} \in I\!R^2,\ \lambda \in (-\infty, 0]\}.$$

Similarly, in the 3D case, it suffices to consider

$$\text{span}\left\{\frac{\sin(\sqrt{-\lambda}\|\mathbf{x} - \mathbf{y}\|)}{\|\mathbf{x} - \mathbf{y}\|}|_\Omega : \mathbf{y} \in I\!R^3,\ \lambda \in (-\infty, 0]\right\}.$$

5.8.6 Inhomogeneous Helmholtz Equation

A similar technique can be applied to find a particular solution of the inhomogeneous Helmholtz-type equation

$$\Delta u - \mu u = f$$

for any $\mu \in I\!R$. We already pointed this out in (5.8.15). In fact, assuming that we have an approximation of f given by $\tilde{f}$ in formula (5.8.19), then from

$$\tilde{u}_P(\mathbf{x}) = \sum_{k=1}^{p}\sum_{j=1}^{n} \frac{a_{jk}}{\lambda_k - \mu}\psi_{\lambda_k}(\|\mathbf{x}-\mathbf{y}_j\|) \tag{5.8.20}$$

with $\lambda_1, ..., \lambda_p \neq \mu$, we get

$$\begin{aligned}(\Delta - \mu)\tilde{u}_P(\mathbf{x}) &= \sum_{k=1}^{p}\sum_{j=1}^{n} \frac{a_{jk}}{\lambda_k - \mu}(\Delta - \mu)\psi_{\lambda_k}(\|\mathbf{x}-\mathbf{y}_j\|)\\ &= \sum_{k=1}^{p}\sum_{j=1}^{n} \frac{a_{jk}}{\lambda_k - \mu}(\lambda_k - \mu)\psi_{\lambda_k}(\|\mathbf{x}-\mathbf{y}_j\|)\\ &= \tilde{f}(\mathbf{x}).\end{aligned}$$

Once we have obtained the approximation of the particular solution $\tilde{u}_P$ in (5.8.20), we must then solve the problem

$$\begin{cases} \Delta u_H - \mu u_H = 0, & \text{in } \Omega,\\ \mathcal{B}u_H = g - \mathcal{B}\tilde{u}_P, & \text{on } \Gamma,\end{cases}$$

and this can be done using the classical method of fundamental solutions. We must keep in mind that the condition on the boundary is now given with an approximated $\tilde{u}_P$ and a poor approximation of u_P might carry more significant errors to $\tilde{u}_H$.

We recall that the solution obtained by the method of fundamental solutions will now be written in the form

$$\tilde{u}_H(x) = \sum_{j=1}^{m} a_j\psi_\mu(\|\mathbf{x}-\mathbf{y}_j\|).$$

Example 5.8.21 We consider the Poisson problem given in Example 5.8.9. To approximate the forcing term $f(x, y)$ and thus the particular solution, we used the Bessel function J_0 as the basis function. We chose 400 quasi-random interior points in an extended domain $[0.85, 2.15] \times [0.85, 2.15]$ and 40 evenly distributed source points on the fictitious boundary which is a circle with center at $(1.5, 1.5)$ and radius 5. The wave numbers in (5.8.16) were chosen to be $\{1, 10, 20, \ldots, 80\}$.

To evaluate the approximate homogeneous solution $\tilde{u}_H$, we chose 40 uniformly distributed points on the physical boundary and the same number of points on the fictitious boundary. We performed our numerical error evaluation on 400 uniform grid points in the domain. In

# of wave lengths	$\lVert u - \tilde{u} \rVert_\infty$	$\lVert f - \tilde{f} \rVert_\infty$
5	$1.30E-1$	$6.98E-2$
6	$8.72E-3$	$2.91E-3$
7	$2.38E-4$	$1.16E-4$
8	$2.21E-6$	$5.75E-7$
9	$1.46E-8$	$3.02E-9$

Table 5.11. *The errors of u and f for various numbers of wave lengths.*

Table 5.11, we show the maximum absolute errors of u and f for various numbers of wave lengths with 40 fixed source points on the fictitious boundary. This reveals that the number of wave lengths has great impact on the numerical accuracy of the approximate solution $\tilde{u}$. This is because the number of the basis function in (5.8.16) and (5.8.17) is increased. □

Example 5.8.22 We consider the following Helmholtz problem [AC05]

$$\begin{aligned} (\Delta - \mu)u &= f(x,y), \quad (x,y) \in \Omega, \\ u &= g(x,y), \quad (x,y) \in \Gamma, \end{aligned}$$

where $\Omega = [-1,1]^2$, and

$$\begin{aligned} f(x,y) &= \frac{\mu}{1+x^4+y^2} - \frac{2+2x^4-20x^6-6y^2+12x^2(1+y^2)}{(1+x^4+y^2)^3} \\ g(x,y) &= \frac{1}{1+x^4+y^2}. \end{aligned}$$

The exact solution is given by

$$u^*(x,y) = \frac{1}{1+x^4+y^2}.$$

The solution parameters were chosen as follows:

- 400 evenly distributed collocation points in an extended domain $W \supset \Omega$ where $W = [-1.2, 1.2]^2$.
- 16 source points evenly distributed in the extended domain $[-1.1, 1.1]^2$.
- 8 frequencies: $\{\lambda_k\}_{k=1}^{8} = \{-1, -4, -9, -16, -25, -36, -49, -64\}$. Note that we shall use values of μ different from these frequencies.

This information is required to approximate f and u_P. Note that we considered an extended domain W to produce a better approximation of f in Ω.

To approximate the homogeneous solution u_H, we chose the following parameters:

- 40 collocation points on the physical boundary Γ.
- 40 source points on the fictitious boundary $\hat{\Gamma}$ which is a circle with center at $(0, 0)$ and radius 5.

In this example, we placed the source points inside the domain for the approximation of particular solutions. This distribution of the collocation and source points for the MFS→MPS algorithm on page 219 and the traditional method of fundamental solutions are shown in Figure 5.20. This does not present a singularity problem because we have used the nonsingular part of the Bessel function J_0 as the basis. In fact, we could have also placed the 16 source points on a circle outside the domain. Now consider the Helmholtz equation with $\mu = -18$. Using

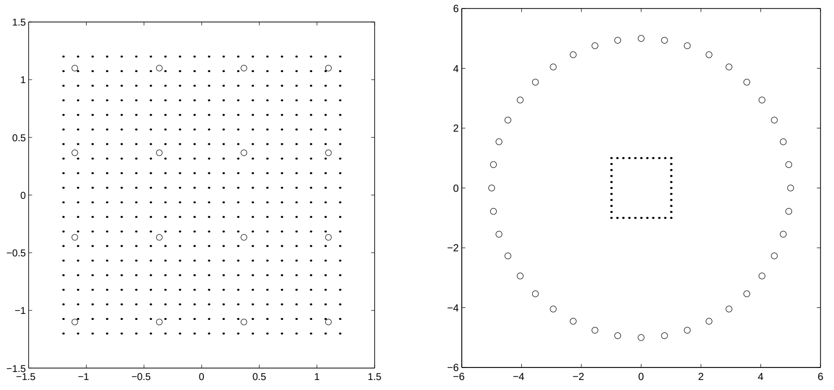

Fig. 5.20. Source points (circle) and collocation points (dot) using MFS→MPS for evaluating $\tilde{u}_P$(left) and the traditional method of fundamental solutions for evaluating $\tilde{u}_H$ (right).

the previous data with interior point sources, we obtained about 0.5% maximum relative errors in the function approximation and about 0.1% maximum relative errors in the approximation of the solution of the Helmholtz equation. The profiles of the relative errors of f and u are shown in Figure 5.21.

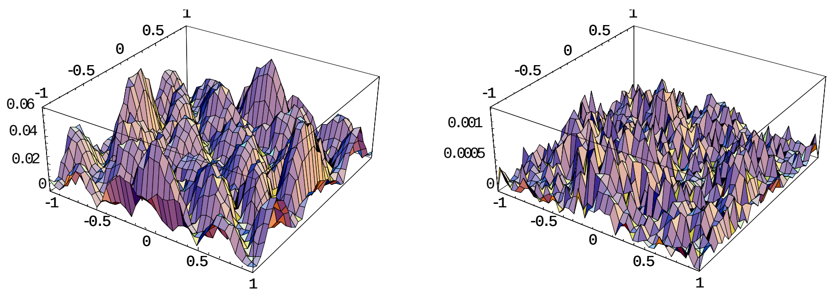

Fig. 5.21. The profiles of relative errors of f and u.

□

5.8.7 Fast Solution of the MFS for Modified Helmholtz Equations

In the formulation of the Method of Fundamental Solutions, we usually obtain a dense matrix which is expensive to handle, especially for the 3D case. But for the modified Helmholtz equation, we have noticed that its fundamental solutions $K_0(\lambda r)$ in 2D and $\exp(-\lambda r)/r$ in 3D have exponential decay. As a result, when λr is sufficiently large, most of the entries of the formulated matrix become so small that they can be considered as negligible and thus can be truncated without loss of numerical accuracy. After the truncation, the new resulting matrix becomes sparse and can be treated efficiently using some sparse solver.

The wave number λ in $K_0(\lambda r)$ or $\exp(-\lambda r)/r$ depends on the governed equation, and r depends on the size and geometric shape of the domain. Thus the efficiency of the algorithm depends on the combination of these two factors.

Example 5.8.23 To be more specific, we consider the following modified Helmholtz equation with Dirichlet boundary condition

$$\begin{array}{rcl} \Delta u - \lambda^2 u &=& (1-\lambda^2)(e^x + e^y), \qquad (x,y) \in \Omega, \\ u &=& e^x + e^y, \quad (x,y) \in \Gamma, \end{array} \tag{5.8.24}$$

where Ω is a bounded domain and Γ satisfies the following parametric equation:

$$\begin{aligned} x &= r(\theta)\cos\theta, \quad y = r(\theta)\sin\theta, \\ r(\theta) &= \sqrt{\cos^2 2\theta + \sqrt{1.1 - \sin^2 2\theta}}, \quad 0 \le \theta \le 2\pi. \end{aligned}$$

By the method of particular solutions, we reduce (5.8.24) to the following homogeneous equation:

$$\begin{aligned} \Delta v - \lambda^2 v &= 0, & (x,y) \in \Omega, \\ v &= e^x + e^y - \tilde{u}_p, & (x,y) \in \Gamma. \end{aligned}$$

Here we choose $\lambda^2 = 1000$. To evaluate the approximate particular solution, 100 evenly distributed grid points were chosen in the extended domain $[-1.5, 1.5] \times [-1, 1]$ plus 100 boundary points which were also used as the collocation points for the method of fundamental solutions. The distribution of these points is shown in Figure 5.22. Furthermore, third order polyharmonic splines were chosen as the basis function to interpolate the forcing term of (5.8.24). Let A be the coefficient ma-

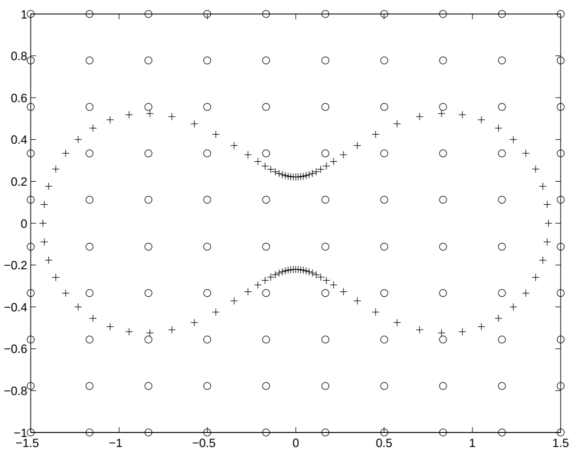

Fig. 5.22. The distribution of interpolation points and collocation points.

trix in the formulation of the method of fundamental solutions. The maximum and minimum coefficients of A are $\varepsilon_{\max} = 5.346 \times 10^{-92}$ and $\varepsilon_{\min} = 2.157 \times 10^{-131}$ respectively. We notice that there is a wide margin between $\varepsilon_{\max}$ and $\varepsilon_{\min}$. Let the truncation value be ε. This means

$$a_{ij} = \begin{cases} 0, & a_{ij} < \varepsilon, \\ a_{ij}, & i = j, \\ a_{ij}, & a_{ij} \geq \varepsilon, \end{cases}$$

where a_{ij} is the entry in A. To ensure the matrix is nonsingular after the truncation, we should always keep the diagonal entries of A. For $\varepsilon = 10^{-95}$, there are only 412 nonzero entries in A. This means 96% of the entries of A were truncated. The profile of nonzero entries of A is shown in Figure 5.23. Surprisingly, the maximum absolute errors of the solution (3.35×10^{-5}) before and after the truncation are identical. The absolute maximum error were computed on the 60 test points located on the boundary. We also observe that the condition numbers of A before and after the truncation are 10^{35} and 10^{18} respectively. The improvement of the condition number does not affect the final result.

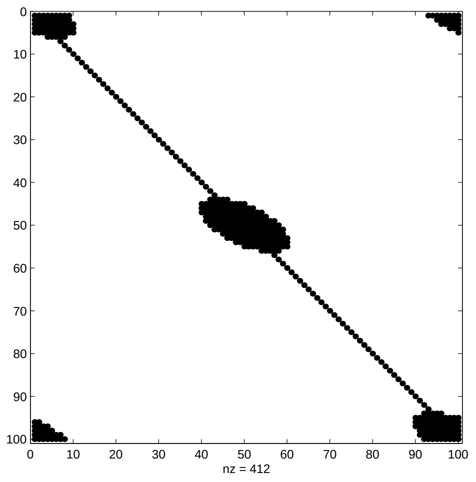

Fig. 5.23. The profile of nonzero entries of A.

Using the same setting as just described, some further numerical results using various wave number λ^2 are given in Table 5.12 where L_∞ denotes the absolute maximum error. In all cases, the L_∞ are exactly the same before and after the truncation. The last column of Table 5.12 shows the condition numbers before and after the truncation.

□

One of the important applications of using modified Helmholtz equa-

λ^2	$\varepsilon_{\min}$	$\varepsilon_{\max}$	ε	L_∞	$\#(\neq 0)$	Cond. #
10	2.60E-10	2.54E-14	2.60E-11	5.49E-2	12.32%	$10^{20} : 10^{9}$
100	4.58E-30	1.41E-42	4.58E-32	1.023E-3	7.72%	$10^{23} : 10^{8}$
500	1.68E-65	2.22E-93	1.68E-67	8.55E-5	3.76%	$10^{29} : 10^{13}$

Table 5.12. *Numerical results for various values of* λ^2*.*

tions is in the area of time–dependent problems which will be described in the next chapter. The above described fast solution algorithm can be applied to the time–dependent problems in case of large wave numbers.

5.9 Solving Large Problems on Circular Points

In case of specially distributed interpolation or collocation points, a surprisingly effective numerical technique for reconstruction of functions from data can be applied [KCS07]. Though it could also be presented at the end of Chapter 2, we present it here due to its applicability within the Method of Particular Solutions.

Assume that the forcing term of a two–dimensional PDE is given as a global formula, and assume that we want to approximate it by a linear combination of particular solutions. Then we are free to choose the data points for reconstruction, and we can make use of this freedom.

We choose a large circle around the domain in question, and choose a number m of concentric circles within this circle. On each circle, we choose the same number n of equidistant points. In this way, the resultant interpolation matrix has the special feature of blocks of circulant matrices due to symmetry. The circulant matrix has many attractive properties which allow us to solve linear equations that contains them efficiently through the use of the fast Fourier transform.

For simplicity, assume that $12 = 3 \cdot 4$ interpolation points are distributed in the circular form with $n = 4$ equidistant points on $m = 3$ circles as shown in Figure 5.24.

We focus on reconstruction by interpolation, and we let $f(\mathbf{x})$ be the function to be interpolated. By the usual radial basis function interpo-

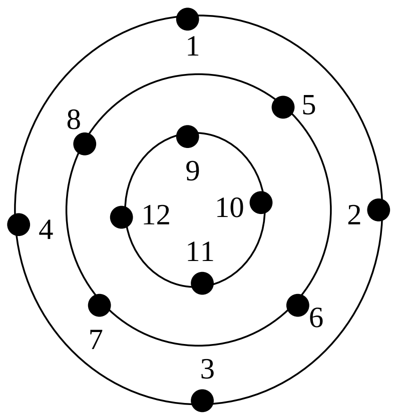

Fig. 5.24. Distribution of 12 circulant points.

lation scheme as shown in Chapter 1.1, we have

$$f(\mathbf{x}) \simeq \widetilde{f}(\mathbf{x}) = \sum_{j=1}^{12} a_j \varphi\left(\|\mathbf{x} - \mathbf{x}_j\|\right).$$

As the standard linear system for interpolation, we have

$$f(\mathbf{x}_i) = \widetilde{f}(\mathbf{x}_i) = \sum_{j=1}^{12} a_j \varphi\left(\|\mathbf{x}_i - \mathbf{x}_j\|\right),\ 1 \le i \le 12, \tag{5.9.1}$$

which can be formulated in the matrix form $\mathbf{A}_\varphi \mathbf{a} = \mathbf{f}$ where

$$\begin{array}{rcl} \mathbf{a} & = & (a_1, a_2, \ldots, a_{12})^T, \\ \mathbf{f} & = & (f(\mathbf{x}_1), f(\mathbf{x}_2), \ldots, f(\mathbf{x}_{12}))^T, \end{array}$$

and

$$\mathbf{A}_\varphi = \begin{pmatrix} \varphi_{1,1} & \varphi_{1,2} & \cdots & \varphi_{1,12} \\ \varphi_{2,1} & \varphi_{2,2} & \cdots & \varphi_{2,12} \\ \vdots & \vdots & \ddots & \vdots \\ \varphi_{1,12} & \varphi_{12,12} & \cdots & \varphi_{12,12} \end{pmatrix},$$

with $\varphi_{ij} := \varphi\left(\|\mathbf{x}_i - \mathbf{x}_j\|\right)$. Let

$$\begin{aligned}
\mathbf{A}_{11} &= \begin{pmatrix} \varphi_{1,1} & \varphi_{1,2} & \varphi_{1,3} & \varphi_{1,4} \\ \varphi_{2,1} & \varphi_{2,2} & \varphi_{2,3} & \varphi_{2,4} \\ \varphi_{3,1} & \varphi_{3,2} & \varphi_{3,3} & \varphi_{3,4} \\ \varphi_{4,1} & \varphi_{4,2} & \varphi_{4,3} & \varphi_{4,4} \end{pmatrix}, \\
\mathbf{A}_{12} &= \begin{pmatrix} \varphi_{1,5} & \varphi_{1,6} & \varphi_{1,7} & \varphi_{1,8} \\ \varphi_{2,5} & \varphi_{2,6} & \varphi_{2,7} & \varphi_{2,8} \\ \varphi_{3,5} & \varphi_{3,6} & \varphi_{3,7} & \varphi_{3,8} \\ \varphi_{4,5} & \varphi_{4,6} & \varphi_{4,7} & \varphi_{4,8} \end{pmatrix}, \\
\mathbf{A}_{13} &= \begin{pmatrix} \varphi_{1,9} & \varphi_{1,10} & \varphi_{1,11} & \varphi_{1,12} \\ \varphi_{2,9} & \varphi_{2,10} & \varphi_{2,11} & \varphi_{2,12} \\ \varphi_{3,9} & \varphi_{3,10} & \varphi_{3,11} & \varphi_{3,12} \\ \varphi_{4,9} & \varphi_{4,10} & \varphi_{4,11} & \varphi_{4,12} \end{pmatrix}.
\end{aligned}$$

Similarly, $\mathbf{A}_{21}, \mathbf{A}_{22}, \mathbf{A}_{23}, \mathbf{A}_{31}, \mathbf{A}_{32}$, and $\mathbf{A}_{33}$ can be defined in a similar way as above, blocking each set of 4 points on each circle. Then $\mathbf{A}_\varphi$ can be written as a block matrix as follows

$$\mathbf{A}_\varphi = \begin{pmatrix} \mathbf{A}_{11} & \mathbf{A}_{12} & \mathbf{A}_{13} \\ \mathbf{A}_{21} & \mathbf{A}_{22} & \mathbf{A}_{23} \\ \mathbf{A}_{31} & \mathbf{A}_{32} & \mathbf{A}_{33} \end{pmatrix}. \tag{5.9.2}$$

Due to the special geometric placement of points, each block matrix $\mathbf{A}_{ij}$ has the structure of a circulant matrix which is a special kind of Toeplitz matrix where each row vector is rotated one element to the right relative to the preceding row vector. The circulant matrix $\mathrm{circ}(a_0, a_1, \cdots, a_n)$ is defined as follows:

$$\mathrm{circ}(a_0, a_1, \cdots, a_n) =: \begin{pmatrix} a_0 & a_1 & \cdots & a_{n-1} & a_n \\ a_n & a_0 & \cdots & a_{n-2} & a_{n-1} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ a_2 & a_3 & \cdots & a_0 & a_1 \\ a_1 & a_2 & \cdots & a_n & a_0 \end{pmatrix}.$$

For example, in $\mathbf{A}_{11}$, due to the symmetry, we have

$$\begin{aligned}
\varphi_{1,1} &= \varphi_{2,2} = \varphi_{3,3} = \varphi_{4,4} \\
\varphi_{1,2} &= \varphi_{2,1} = \varphi_{2,3} = \varphi_{3,2} = \varphi_{4,1} = \varphi_{1,4} = \varphi_{3,4} = \varphi_{4,3} \\
\varphi_{1,3} &= \varphi_{3,1} = \varphi_{2,4} = \varphi_{4,2}.
\end{aligned}$$

Hence,

$$\mathbf{A}_{11} = \begin{pmatrix} \varphi_{1,1} & \varphi_{1,2} & \varphi_{1,3} & \varphi_{1,2} \\ \varphi_{1,2} & \varphi_{1,1} & \varphi_{1,2} & \varphi_{1,3} \\ \varphi_{1,3} & \varphi_{1,2} & \varphi_{1,1} & \varphi_{1,2} \\ \varphi_{12} & \varphi_{1,3} & \varphi_{1,2} & \varphi_{1,1} \end{pmatrix} =: \operatorname{circ}(\varphi_{1,1}, \varphi_{1,2}, \varphi_{1,3}, \varphi_{1,2}).$$

For $\mathbf{A}_{12}$, we obtain

$$\mathbf{A}_{12} = \operatorname{circ}(\varphi_{1,5}, \varphi_{1,6}, \varphi_{1,6}, \varphi_{1,5}).$$

Similarly, all the other block matrices $\mathbf{A}_{13}, \mathbf{A}_{21}, \mathbf{A}_{22}, \mathbf{A}_{23}, \mathbf{A}_{31}, \mathbf{A}_{32}$, and $\mathbf{A}_{33}$ in (5.9.2) have the similar circulant structure. Due to this particular form, (5.9.1) can be efficiently solved using special properties of circulant matrices [Dav79]. The main idea is to break the full system of equations into smaller block systems and solve them individually and efficiently. We will now give a general procedure for this.

In this approach, we assume that the given function to be approximated can be extended to a larger circle containing the Ω on which we actually need the reconstruction. We can assume that the circle is centered at the origin. To generalize the above particular case, we choose m concentric circles with n equidistant interpolation points on each circle. This can be achieved in the following way:

$$\Omega_{R_i} = \{\mathbf{x} \in I\!R^2 \, : \, |\mathbf{x}| < R_i\}, \ \ 1 \le i \le m,$$

where $R_1 < R_2 < \ldots < R_m$ and $\Omega \subseteq \Omega_{R_m}$. A special case is shown in Figure 5.25.

On the circles $\partial\Omega_{R_i}, \ 1 \le i \le m$, we define the mn collocation points $\{\mathbf{x}_{i,j}\}_{i=1,j=1}^{m,n} = \{(x_{i,j}, y_{i,j})\}_{i=1,j=1}^{m,n}$, by

$$\begin{aligned} x_{i,j} &= R_i \cos\left(\frac{2(j-1)\pi}{n} + \frac{2\alpha_i\pi}{n}\right) \\ y_{i,j} &= R_i \sin\left(\frac{2(j-1)\pi}{n} + \frac{2\alpha_i\pi}{n}\right), \end{aligned}$$

where $0 \le \alpha_i \le 1, 1 \le j \le n$. In this way, the starting position of the points on each circle of radius R_i is rotated by an angle $2\alpha_i\pi/n$. Such a rotation will ensure that the interpolation points are more uniformly distributed. The radii of the concentric circles can be chosen arbitrarily, but may be evenly divided as follows:

$$R_i = \frac{i}{m} r_m, \ \ 1 \le i \le m,$$

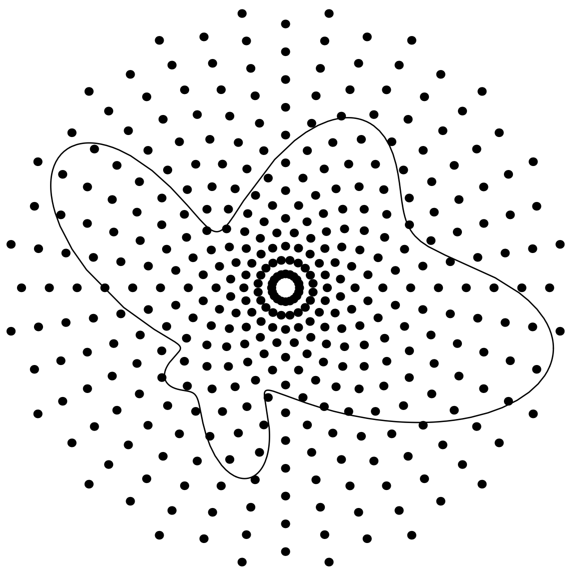

Fig. 5.25. The domain of the function and the circular interpolation points.

where r_m is the radius of the largest circle. In Figure 5.25, we present a typical distribution of collocation points with $n = 20, m = 20, r_{\max} = 1$, and $\alpha_i = 0$ for $i = 1, 3, \cdots, 19$, $\alpha_i = 0.5$, for $i = 2, 4, \cdots, 20$.

For the mn circular points given above, the collocation equations (5.9.1) can be generalized to the following system

$$f(\mathbf{x}_{i,k}) = \sum_{\ell=1}^{m} \sum_{j=1}^{n} a_{\ell,j} \varphi(\|\mathbf{x}_{\ell,j} - \mathbf{x}_{i,k}\|_2), \quad 1 \le i \le m, \ 1 \le k \le n. \tag{5.9.3}$$

In matrix form, we have

$$\mathbf{A}_\varphi \mathbf{a} \ = \ \mathbf{f} \tag{5.9.4}$$

where the $mn \times mn$ matrix $\mathbf{A}_\varphi$ has the structure

$$\mathbf{A}_\varphi = \begin{pmatrix} \mathbf{A}_{11} & \mathbf{A}_{12} & \dots & \mathbf{A}_{1,m} \\ \mathbf{A}_{21} & \mathbf{A}_{22} & \dots & \mathbf{A}_{2,m} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{A}_{m1} & \mathbf{A}_{m2} & \dots & \mathbf{A}_{m,m} \end{pmatrix} \tag{5.9.5}$$

where each of the $n \times n$ submatrices $\mathbf{A}_{i,\ell}$ is circulant due to

$$\begin{aligned}
\mathbf{A}_{i,\ell} &= (\varphi(\|\mathbf{x}_{i,k} - \mathbf{x}_{\ell,j}\|_2))_{1\le j,k\le n} \\
&= (\varphi(\|R^{k-1}\mathbf{x}_{i,1} - R^{j-1}\mathbf{x}_{\ell,1}\|_2))_{1\le j,k\le n} \\
&= (\varphi(\|R^{k-1}(\mathbf{x}_{i,1} - R^{j+n-k}\mathbf{x}_{\ell,1})\|_2))_{1\le j,k\le n} \\
&= (\varphi(\|\mathbf{x}_{i,1} - R^{j+n-k}\mathbf{x}_{\ell,1}\|_2))_{1\le j,k\le n} \\
&= \operatorname{circ}(\varphi\left\|\mathbf{x}_{i,1} - \mathbf{x}_{\ell,1}\right\|_2, \ldots, \varphi\left\|\mathbf{x}_{i,1} - \mathbf{x}_{\ell,n}\right\|_2),\ 1 \le i,\ell \le m,
\end{aligned}$$

if we denote rotation by an angle of $2\pi/n$ by R. Also, we have

$$\mathbf{f}_{(i-1)n+j} = f(\mathbf{x}_{i,j}), \quad 1 \le i \le m, \quad 1 \le j \le n.$$

Altogether, our system is **block–circulant**, and such systems have an efficient solution procedure which we outline here.

It is known that any circulant matrix can be diagonalized in the following way. If $\mathbf{C} = \operatorname{circ}(c_1, \ldots, c_n)$, then

$$\mathbf{C} = \mathbf{U}^*\mathbf{D}\mathbf{U}$$

where $\mathbf{D} = \operatorname{diag}(d_1, \ldots, d_n), d_j = \sum_{k=1}^n c_k \omega^{(k-1)(j-1)}$, and the $n \times n$ matrix $\mathbf{U}$ is the matrix arising in the **Discrete Fourier Transform**, and is the conjugate of the matrix (see [Dav79])

$$\mathbf{U}^* = \frac{1}{n^{1/2}} \begin{pmatrix} 1 & 1 & 1 & \ldots & 1 \\ 1 & \omega & \omega^2 & \ldots & \omega^{n-1} \\ 1 & \omega^2 & \omega^4 & \ldots & \omega^{2(n-1)} \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & \omega^{n-1} & \omega^{2(n-1)} & \ldots & \omega^{(n-1)(n-1)} \end{pmatrix},$$

with $\omega = e^{2\pi\iota/n}$.

Next, we denote the matrix tensor product by $\otimes$. The tensor product of the $m \times n$ matrix $\mathbf{V}$ and the $\ell \times k$ matrix $\mathbf{W}$ is the $m\ell \times nk$ matrix [Mey00]

$$\mathbf{V} \otimes \mathbf{W} = \begin{pmatrix} v_{11}\mathbf{W} & v_{12}\mathbf{W} & \cdots & v_{1n}\mathbf{W} \\ v_{21}\mathbf{W} & v_{22}\mathbf{W} & \cdots & v_{2n}\mathbf{W} \\ \vdots & \vdots & \ddots & \vdots \\ v_{m1}\mathbf{W} & v_{m2}\mathbf{W} & \cdots & v_{mn}\mathbf{W} \end{pmatrix}.$$

We therefore premultiply the system (5.9.4) by the block diagonal $mn \times mn$ matrix $\mathbf{I}_m \otimes \mathbf{U}$ and, using the fact that $\mathbf{U}$ is unitary, we have

$$(\mathbf{I}_m \otimes \mathbf{U})\,\mathbf{A}_\varphi\,(\mathbf{I}_m \otimes \mathbf{U}^*)\,(\mathbf{I}_m \otimes \mathbf{U})\,\mathbf{a} = (\mathbf{I}_m \otimes \mathbf{U})\,\mathbf{f},$$

since $(\mathbf{I}_m \otimes \mathbf{U}^*)\,(\mathbf{I}_m \otimes \mathbf{U}) = \mathbf{I}_{mn}$. This yields

$$\tilde{\mathbf{A}}_\varphi \tilde{\mathbf{a}} \;=\; \tilde{\mathbf{f}}\,, \tag{5.9.6}$$

where $\tilde{\mathbf{a}} = (\mathbf{I}_m \otimes \mathbf{U})\,\mathbf{a}$, $\tilde{\mathbf{f}} = (\mathbf{I}_m \otimes \mathbf{U})\,\mathbf{f}$,

$$\tilde{\mathbf{A}}_\varphi = \begin{pmatrix} \mathbf{D}_{11} & \mathbf{D}_{12} & \dots & \mathbf{D}_{1,m} \\ \mathbf{D}_{21} & \mathbf{D}_{22} & \dots & \mathbf{D}_{2,m} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{D}_{m1} & \mathbf{D}_{m2} & \dots & \mathbf{D}_{m,m} \end{pmatrix} \tag{5.9.7}$$

and each $n \times n$ submatrix $\mathbf{D}_{k,\ell}$ is diagonal and symmetric. If the submatrix $\mathbf{A}_{i,\ell}$ of $\mathbf{A}_\varphi$ in (5.9.5) is circulant, i.e.

$$\begin{aligned} \mathbf{A}_{i,\ell} \;&=\; \operatorname{circ}(c_1^{i,\ell}, \dots, c_n^{i,\ell}) \\ &=\; \begin{pmatrix} c_1^{i,\ell} & c_2^{i,\ell} & c_3^{i,\ell} & \dots & c_n^{i,\ell} \\ c_n^{i,\ell} & c_1^{i,\ell} & c_2^{i,\ell} & \dots & c_{n-1}^{i,\ell} \\ c_{n-1}^{i,\ell} & c_n^{i,\ell} & c_1^{i,\ell} & \dots & c_{n-2}^{i,\ell} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ c_2^{i,\ell} & c_3^{i,\ell} & c_4^{i,\ell} & \dots & c_1^{i,\ell} \end{pmatrix}, \end{aligned}$$

then the corresponding submatrix is $\mathbf{D}_{i,\ell} = \operatorname{diag}(d_1^{i,\ell}, \dots, d_n^{i,\ell})$, where

$$d_j^{i,\ell} \;=\; \sum_{k=1}^{n} c_k^{i,\ell} \omega^{(k-1)(j-1)},\ 1 \le j \le n.$$

Equipped with this important property, the system (5.9.6) can be decomposed into n complex systems of equations, each of size $m \times m$, i.e.

$$\mathbf{E}_j\, \tilde{\mathbf{a}}_j \;=\; \tilde{\mathbf{f}}_j\,,\ \ 1 \le j \le n, \tag{5.9.8}$$

where

$$(\mathbf{E}_j)_{i,\ell} = d_j^{i,\ell},\ 1 \le i, \ell \le m,\ 1 \le j \le n,$$

$$(\tilde{\mathbf{f}}_j)_i = \mathbf{f}_{(i-1)n+j} = f(\mathbf{x}_{i,j}),\ 1 \le j \le n, \quad 1 \le i \le m,$$

and

$$(\tilde{\mathbf{a}}_j)_i = \mathbf{a}_{(i-1)n+j} = a_{i,j},\ 1 \le j \le n, \quad 1 \le i \le m.$$

In other words, the entries (i, ℓ) of the matrix $\mathbf{E}_j$ are formed by extracting the j^{th} diagonal entry of each $\mathbf{D}_{i,\ell}$.

We summarize the above procedure in the following matrix decomposition algorithm for the evaluation of coefficients of $\mathbf{a} = \{a_j\}_{j=1}^{mn}$ in (5.9.3):

Algorithm of Matrix Decomposition

Step 1 Compute $\tilde{\mathbf{f}} = (\mathbf{I}_m \otimes \mathbf{U})\,\mathbf{f}$.

Step 2 Evaluate the $m^2 n$ elements $d_j^{i,\ell}$, $1 \le i, \ell \le m,\ 1 \le j \le n$.

Step 3 Solve the $m \times m$ systems $\mathbf{E}_j\,\tilde{\mathbf{a}}_j = \tilde{\mathbf{f}}_j,\ 1 \le j \le n$.

Step 4 Compute $\mathbf{a} = (\mathbf{I}_m \otimes \mathbf{U}^*)\ \tilde{\mathbf{a}}$.

Due to the form of the unitary matrix $\mathbf{U}$, the operation in **Step 1** can be efficiently performed using the **Fast Fourier Transform** (FFT). The operation count then is $O(mn \ln n)$. Similarly, because of the form of the matrix $\mathbf{U}^*$, the operation in **Step 4** can be carried out via inverse FFTs at a cost of order $O(mn \ln n)$.

In **Step 3**, we need to solve n complex linear $m \times m$ systems. Gauss elimination with partial pivoting can be employed at a cost of $O(nm^3)$ operations. In **Step 2**, we apply FFTs at a cost of $O(m^2 n \ln n)$ operations.

It is noteworthy that in this algorithm we do not have to physically store the interpolation points. All we need is to store the diagonal elements of $\mathbf{D}_{ij}$ as shown in (5.9.7). When m and n are large, this results in substantial saving in storage. Furthermore, the diagonal entries of $\mathbf{D}_{ij}$ are conjugate complex numbers. As a result, we need to store only half of them in the computer memory. The other half of the conjugate numbers can be generated when needed. This also presents a great saving in memory space on a computer. This allows us to solve large-scale problems of this specific form without using the domain decomposition method or any other of the techniques described in Section 2.8.

Some limitations of the above algorithm include the fact that it is not suitable for the interpolation of data at fixed scattered locations. If working on a non–circular domain, we also need to be able to smoothly extend the function to be reconstructed to the exterior of the domain under consideration. Furthermore, radial basis functions with polynomial augmentation cannot be used.

Example 5.9.9 Here we perform some numerical tests [KCS07] by using two of the benchmark functions of R. Franke [Fra82] on the unit square.

Two of these test functions are given as follows:

$$F_1(x,y) = \frac{3e^{h_1(x,y)}}{4} + \frac{3e^{h_2(x,y)}}{4} + \frac{e^{h_3(x,y)}}{2} - \frac{e^{h_4(x,y)}}{5}$$

where

$$\begin{aligned}
h_1(x,y) &= \frac{-1}{4}\left((9x-2)^2 + (9y-2)^2\right), \\
h_2(x,y) &= \frac{-1}{49}(9x+1)^2 - \frac{1}{10}(9y+1)^2, \\
h_3(x,y) &= \frac{-1}{4}(9x-7)^2 - \frac{1}{4}(9y-3)^2, \\
h_4(x,y) &= -(9x-4)^2 - (9y-7)^2, \\
F_2(x,y) &= \tfrac{1}{9}\left[\tanh(9y-9x)+1\right].
\end{aligned}$$

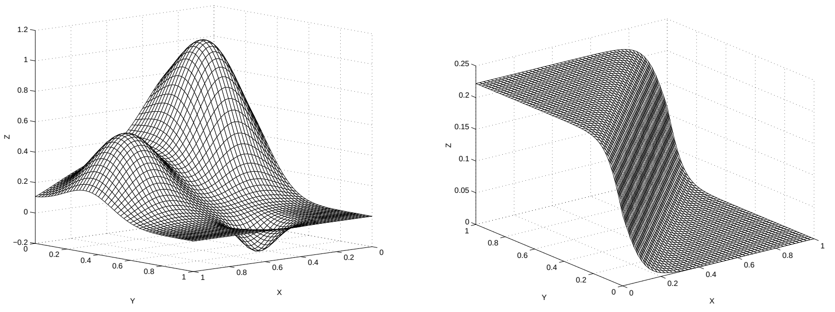

Fig. 5.26. Profiles of test functions F1 and F2.

We denote by m the number of concentric circles, by n the number of interpolation points that are evenly distributed on each circle, and by r_m the radius of the largest circle centered at $(0.5, 0.5)$ which covers the unit square $[0,1]\times[0,1]$. We used multiquadrics with a scaling constant c as in (2.6.1).

We chose 400 evenly distributed test points in the unit square for the evaluation of the absolute maximum error. The numerical results in Table 5.13 were obtained by choosing various parameters c, r_m, m, and n. Consequently, the total number of interpolation points used for $m = n = 90, 110,\ 130, 150$ is $8100, 12100,\ 16900$, and 22500, respectively. The accuracy of the results presented in Table 5.13 is excellent. Furthermore, the matrix decomposition algorithm is also highly efficient. Using a Pentium 4 (3.19GHZ) PC, the CPU time for various m and n is also listed in Table 5.13.

m	n	c	$F_1\left(\left\lVert f-\widetilde{f}\right\rVert_\infty\right)$	$F_2\left(\left\lVert f-\widetilde{f}\right\rVert_\infty\right)$	CPU (seconds)
90	90	0.1	$1.17E-09$	$2.18E-09$	1.57
110	110	0.1	$2.10E-10$	$2.81E-10$	2.56
130	130	0.09	$5.43E-11$	$5.41E-11$	3.92
150	150	0.07	$2.92E-11$	$4.08E-11$	5.67

Table 5.13. *Absolute maximum errors for the two test functions with* $r_m = 1.0$.

As is well known, a challenge when using radial basis functions is the limitation in the number of interpolation points we can use. Here we wish to make m and n as large as possible. To break the barrier of memory limitation, further consideration of memory management is required. Instead of storing all the $\mathbf{E}_j$, $j = 1, \ldots, n$ in (5.9.8), we simply subdivide n into k equal groups, depending on the capacity of the computer, and store $\mathbf{E}_j$ accordingly. We then solve these group by group. The price one pays for this is the additional evaluation of $\mathbf{D}_{ij}$ in (5.9.7) k times. In this way, we use computing time in exchange for memory space. The number k can be adjusted depending on the computer hardware system used. As shown in Table 5.14, we managed to reach $m = 250, n = 400$ with $k = 2$ (i.e. $100,000$ interpolation points) without encountering problems. Without memory management ($k = 1$), the CPU times in Table 5.14 are 26.92, 29.75, 44.56 respectively. Thus, this method enables us to solve large–scale problems on a standard PC. Here we used multiquadrics with a scale factor $c = 0.04$. The better accuracy with larger r_m is consistent with the results in Table 5.13.

Since we solve a much smaller $m \times m$ system n times instead of solving a large $mn \times mn$ system, the algorithm is suitable for parallel implementation. The computer running time can thus be significantly reduced.

□

For engineering applications, it is important to have good approximations of the gradient and second derivatives of a given function. But it is known that numerical differentiation is inherently ill-posed. A small change of the function may cause significant change in its derivatives. The approximation of the first derivative of a given function is unstable by any numerical algorithm, although the instability is often manage-

m	n	$\left\| f - \widetilde{f} \right\|_\infty$	CPU time (seconds)
250	250	2.68E-11	43.53
250	300	4.07E-11	47.15
250	400	7.29E-11	54.23

Table 5.14. *Absolute maximum error and CPU time for F_1 with $c = 0.04$ and $r_m = 1.0$.*

able. The task of approximating higher derivatives is even more challenging. Thus we also treat derivative evaluation here, but we remind the reader that interpolation of smooth functions by smooth radial basis functions will always yield good approximations also to the derivatives of the given function (see Section 2.3).

Example 5.9.10 In this example, we further consider the approximation of the first and second partial derivatives of the following function [KCS07]:

$$f(x,y) := F_3(x,y) := x\exp(-x^2 - y^2), \qquad (x,y) \in [0,1] \times [0,1]\,.$$

Then,

$$\begin{aligned}
\nabla f(x,y) &= \Big((1-2x^2)\exp(-x^2-y^2), 2xy\exp(-x^2-y^2)\Big), \\
\frac{\partial^2 f}{\partial x^2}(x,y) &= (4x^3 - 6x)\exp(-x^2-y^2), \\
\frac{\partial^2 f}{\partial y^2}(x,y) &= -(2x - 4xy^2)\exp(-x^2-y^2).
\end{aligned}$$

To approximate the partial derivatives of a given function, we take the partial derivatives of the basis function in (5.9.3); i.e. ,

$$\begin{aligned}
\frac{\partial f}{\partial x}(x,y) &\simeq \frac{\partial \widetilde{f}}{\partial x}(x,y)) = \sum_{j=1}^{mn} a_j \frac{\partial \varphi_j}{\partial x}(x,y), \\
\frac{\partial^2 f}{\partial x^2}(x,y) &\simeq \frac{\partial^2 \widetilde{f}}{\partial x^2}(x,y) = \sum_{j=1}^{mn} a_j \frac{\partial^2 \varphi_j}{\partial x^2}(x,y),
\end{aligned}$$

$$\frac{\partial f}{\partial y}(x,y) \simeq \frac{\partial \widetilde{f}}{\partial y}(x,y) = \sum_{j=1}^{mn} a_j \frac{\partial \varphi_j}{\partial y}(x,y),$$

$$\frac{\partial^2 f}{\partial y^2}(x,y) \simeq \frac{\partial^2 \widetilde{f}}{\partial y^2}(x,y) = \sum_{j=1}^{mn} a_j \frac{\partial^2 \varphi_j}{\partial y^2}(x,y),$$

with

$$\varphi_j(x,y) = \sqrt{\rho_j^2(x,y) + c^2},\ \rho_j^2(x,y) := (x-x_j)^2 + (y-y_j)^2$$

and

$$\frac{\partial \varphi_j}{\partial x} = \frac{x - x_j}{\sqrt{\rho_j^2 + c^2}}, \qquad \frac{\partial \varphi_j}{\partial y} = \frac{y - y_j}{\sqrt{\rho_j^2 + c^2}},$$

$$\frac{\partial^2 \varphi_j}{\partial x^2} = \frac{(y-y_j)^2 + c^2}{\left(\rho_j^2 + c^2\right)^{3/2}}, \qquad \frac{\partial^2 \varphi_j}{\partial y^2} = \frac{(x-x_j)^2 + c^2}{\left(\rho_j^2 + c^2\right)^{3/2}}.$$

We chose multiquadrics as the basis functions, and $r_m = 1.0$. Similar to the last example, we chose 400 evenly distributed points in the given domain for the evaluation of the absolute maximum error. From Table 5.15, we observe that the approximations of the first partial derivatives are less accurate than the function approximation by two orders of magnitude. The second derivatives are two to three orders of magnitude less accurate than the approximations of the first derivatives. These results are consistent with Section 2.3 and much better than any other traditional numerical method using divided differences. In order to achieve higher accuracy in approximating the second derivatives, we can in this case extend the region covered by the interpolation circles. For instance, if we let $m = n = 140$, $c = 0.05, r_m = 1.2$, we obtain the following results:

$$\begin{aligned}
\epsilon &:= \left\| f - \widetilde{f} \right\|_\infty &&= 2.58\mathtt{E}-13\,,\\
\epsilon_x &:= \left\| \tfrac{\partial f}{\partial x} - \tfrac{\partial \widetilde{f}}{\partial x} \right\|_\infty &&= 3.63\mathtt{E}-11\,,\\
\epsilon_y &:= \left\| \tfrac{\partial f}{\partial y} - \tfrac{\partial \widetilde{f}}{\partial y} \right\|_\infty &&= 3.69\mathtt{E}-11\,,\\
\epsilon_{xx} &:= \left\| \tfrac{\partial^2 f}{\partial x^2} - \tfrac{\partial^2 \widetilde{f}}{\partial x^2} \right\|_\infty &&= 6.16\mathtt{E}-9\,,\\
\epsilon_{yy} &:= \left\| \tfrac{\partial^2 f}{\partial y^2} - \tfrac{\partial^2 \widetilde{f}}{\partial y^2} \right\|_\infty &&= 4.84\mathtt{E}-9\,.
\end{aligned}$$

The above notation is also used in Table 5.15. □

$m=n$	c	ϵ	ϵ_x	ϵ_y	ϵ_{xx}	ϵ_{yy}
60	0.2	1.17E-09	7.07E-08	9.79E-07	9.01E-06	8.48E-06
80	0.2	5.69E-10	1.06E-08	1.16E-08	1.30E-06	1.32E-06
100	0.1	5.73E-10	3.45E-08	3.98E-08	1.01E-05	8.54E-06
120	0.08	4.75E-11	2.43E-08	1.98E-08	6.17E-07	7.98E-07
140	0.07	2.26E-11	3.69E-09	4.13E-09	1.03E-06	7.17E-07
160	0.05	4.22E-12	3.61E-09	4.10E-09	4.23E-07	3.63E-07

Table 5.15. *Absolute maximum error for function F_3 and its partial derivatives.*

Example 5.9.11 In this example we applied the above matrix decomposition algorithm and the Method of Particular Solutions to solve the following Poisson problem:

$$\begin{aligned} \Delta u &= 2e^{x-y}, \quad (x,y) \in \Omega, \\ u &= e^{x-y} + e^x \cos y, \quad (x,y) \in \Gamma, \end{aligned}$$

where the parametric representation of the boundary curve Γ is given by

$$\begin{cases} x = (1+\cos^2 4t)\cos t, \\ y = (1+\cos^2 4t)\sin t, \end{cases} \qquad t \in [0, 2\pi],$$

which is illustrated in Figure 5.27.

To get an approximate particular solution, we chose multiquadrics as the basis functions. The close form approximate particular solution can be found in (5.5.14) and (5.5.19). The collocation points were chosen in the form of concentrated circles as described in this section. We chose $m = n = 100, c = 0.1$, and $r_m = 2.8$. To approximate the homogeneous solution, we applied the Method of Fundamental Solutions with 80 collocation points evenly distributed (in terms of angle) on the boundary and the same number of source points on the fictitious circle with center at $(0,0)$ and radius 8. Using the Maximum Principle [PW67], it can be shown that the maximum error of the Method of Fundamental Solutions occurs on the boundary. Hence, we chose 200 testing points on the boundary and computed the absolute maximum error. The result of

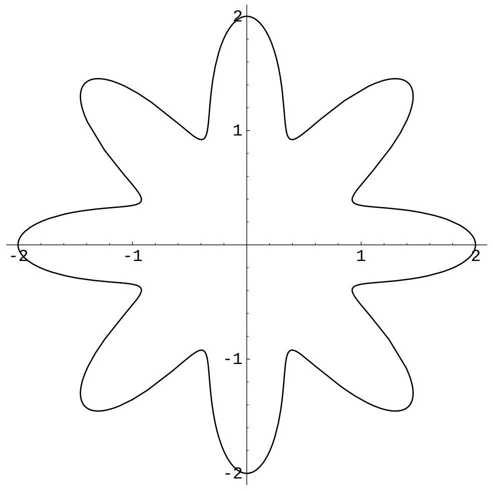

Fig. 5.27. The profile of of the star–shaped boundary.

combining these two approaches is amazing. We obtained an absolute maximum error $1.231E-10$.

As we know, the method of fundamental solutions can be almost as accurate as machine accuracy for the homogeneous equations if the exact boundary conditions are given. One of the reasons preventing us from obtaining such accuracy for inhomogeneous equations is the accuracy of evaluating particular solutions. This example shows that if we can increase the accuracy of approximating the inhomogeneous term, we can obtain extremely high accuracy of solving inhomogeneous problems. $\square$

6
Time–Dependent Problems

Simple time–dependent problems can be tackled by the **Method of Lines** we sketched in Section 2.10. It requires a spatial discretization and solves a system of ordinary differential equations in time for the coefficients of the spatial representation of the solution. But in this chapter we will further extend the methodology we introduced for solving elliptic problems in the last two chapters to various types of time–dependent problems which require neither domain nor boundary discretizations. We shall focus on algorithms which are primarily based on the reduction of boundary value problems for various time–dependent partial differential equations to time-independent inhomogeneous Helmholtz-type equations. Using the particular solutions derived in Chapter 5, the inhomogeneities can be further eliminated. Consequently, the resulting homogeneous equations can be solved using a variety of standard boundary methods. As we have shown in Chapter 4, the Method of Fundamental Solutions is a highly efficient boundary meshless method. In this chapter, the MFS will be our main tool for solving homogeneous problems. For reducing time–dependent problems to time-independent ones, we shall employ two techniques: the Laplace transform in Section 6.1 and a specific time-stepping method in Section 6.2.

6.1 Method of Laplace Transforms

In this section several important numerical methods are combined to solve time–dependent problems. The overall solution procedure can be summarized as follows:

(i) The Laplace transform temporarily removes the time variable.
(ii) The Method of Particular Solutions (MPS) from Chapter 5 splits

the resulting elliptic Helmholtz-type boundary value problem into two subproblems: finding a particular solution and a homogeneous solution.

(iii) Construct an approximate particular solution using radial basis functions, ignoring boundary conditions.

(iv) The Method of Fundamental Solutions (MFS) from Chapter 4 solves the homogeneous boundary value problem without any domain or boundary discretization.

(v) The well-known **Stehfest algorithm** [Ste70] inverts the Laplace transform and finally provides a solution of the time–dependent problem we started with.

6.1.1 Applying the Laplace Transform

We shall explain this by boundary value problems for the diffusion equation. Let Ω be a bounded domain in $I\!R^d$ with Dirichlet and Neumann boundaries $\Gamma = \Gamma^D \cup \Gamma^N, \Gamma^D \cap \Gamma^N = \emptyset$. We consider the following inhomogeneous diffusion equation

$$\frac{1}{k}\frac{\partial u(\mathbf{x},t)}{\partial t} + f(\mathbf{x},t) = \Delta u(\mathbf{x},t), \quad \mathbf{x} \in \Omega, \quad t > 0, \tag{6.1.1}$$

with boundary conditions

$$\begin{aligned} u(\mathbf{x},t) &= g_1(\mathbf{x},t), \quad \mathbf{x} \in \Gamma^D, \quad t > 0, \\ \frac{\partial u}{\partial n}(\mathbf{x},t) &= g_2(\mathbf{x},t), \quad \mathbf{x} \in \Gamma^N, \quad t > 0, \end{aligned}$$

and initial condition

$$u(\mathbf{x},0) = u_0(\mathbf{x}), \quad \mathbf{x} \in \Omega,$$

where n is the outward unit vector normal to Γ. The diffusion coefficient k is assumed to be constant with respect to space and time. The functions $u_0(\mathbf{x}), f(\mathbf{x},t), g_1(\mathbf{x},t)$ and $g_2(\mathbf{x},t)$ are given.

To temporarily remove the time variable, we first define the **Laplace transform** of a given real-valued function g on $[0,\infty)$, when it exists, by

$$\mathcal{L}\left[g(\cdot)\right](s) := G(s) := \int_0^\infty g(t)e^{-st}\mathrm{dt} \tag{6.1.2}$$

where the transform parameter s is real and positive. By integration by parts, we have

$$\mathcal{L}\left[\frac{\partial u}{\partial t}(\mathbf{x},t)\right](s) = \int_0^\infty \frac{\partial u}{\partial t}(\mathbf{x},t)e^{-st}\mathrm{dt} = sU(\mathbf{x},s) - u_0(\mathbf{x}). \qquad (6.1.3)$$

By direct substitution of (6.1.2) and (6.1.3) into (6.1.1), we obtain

$$\begin{aligned} \left(\Delta - \frac{s}{k}\right)U(\mathbf{x},s) &= F(\mathbf{x},s) - \frac{u_0(\mathbf{x})}{k}, && \mathbf{x}\in\Omega, \\ U(\mathbf{x},s) &= G_1(\mathbf{x},s), && \mathbf{x}\in\Gamma^D, \\ \frac{\partial U}{\partial n}(\mathbf{x},s) &= G_2(\mathbf{x},s), && \mathbf{x}\in\Gamma^N, \end{aligned} \qquad (6.1.4)$$

where $F(\mathbf{x},s) = \mathcal{L}\left[f(\mathbf{x},\cdot)\right](s), G_i(\mathbf{x},s) = \mathcal{L}\left[g_i(\mathbf{x},\cdot)\right](s), i = 1,2$. The modified Helmholtz equation in (6.1.4) is inhomogeneous. The Method of Particular Solutions (MPS) and the Method of Fundamental Solutions (MFS) introduced in the last two chapters can be applied to solve (6.1.4).

A similar procedure can be applied to the following wave equation

$$\begin{aligned} \Delta u(\mathbf{x},t) &= \frac{\partial^2 u}{\partial t^2}(\mathbf{x},t), && \mathbf{x}\in\Omega, \quad t>0, \\ u(\mathbf{x},0) &= u_0(\mathbf{x}), && \mathbf{x}\in\Gamma^D \\ \frac{\partial u}{\partial t}(\mathbf{x},0) &= v_0(\mathbf{x}), && \mathbf{x}\in\Gamma^N \end{aligned} \qquad (6.1.5)$$

with Dirichlet boundary conditions

$$u(\mathbf{x},t) = g(\mathbf{x},t), \quad \mathbf{x}\in\Gamma^D, \quad t>0. \qquad (6.1.6)$$

By Laplace transform, we have

$$\begin{aligned} \mathcal{L}\left[\frac{\partial^2 u}{\partial t^2}(\mathbf{x},t)\right](s) &= \int_0^\infty \frac{\partial^2 u}{\partial t^2}e^{-st}\mathrm{dt} \\ &= -\frac{\partial u}{\partial t}(\mathbf{x},0) + s\int_0^\infty \frac{\partial u}{\partial t}(\mathbf{x},t)e^{-st}\mathrm{dt} \\ &= s^2U(\mathbf{x},s) - su(\mathbf{x},0) - \frac{\partial u}{\partial t}(\mathbf{x},0) \\ &= s^2U(\mathbf{x},s) - su_0(\mathbf{x}) - v_0(\mathbf{x}). \end{aligned}$$

Once again, (6.1.5)-(6.1.6) can be reduced to the following inhomogeneous modified Helmholtz problem

$$\begin{aligned} \Delta U(\mathbf{x},s) - s^2U(\mathbf{x},s) &= -su_0 - v_0, && \mathbf{x}\in\Omega, \\ U(\mathbf{x},s) &= G(\mathbf{x},s), && \mathbf{x}\in\Gamma^D, \end{aligned}$$

where $U(\mathbf{x},s) = \mathcal{L}\left[u(\mathbf{x},\cdot)\right](s)$ and $G(\mathbf{x},s) = \mathcal{L}\left[g(\mathbf{x},\cdot)\right](s)$.

Notice that the solution processes for solving the diffusion equations and wave equations are practically the same. If the parameter k behaves nicely, the same approach can also be applied to reaction-diffusion equations of the type

$$\frac{1}{k}\frac{\partial u(\mathbf{x},t)}{\partial t} = \Delta u(\mathbf{x},t) - \lambda^2 u$$

without additional difficulty. After suitable modifications the procedure can be implemented similarly for convection-diffusion and nonlinear equations.

6.1.2 Inverting the Laplace Transform

Once the approximate solution of $U(\mathbf{x},s)$ is found in the Laplace space, we need to invert it back to the original time space which can be achieved by applying numerical inverse Laplace transform schemes. There are many such schemes in the literature. Among them, the Stehfest algorithm [Ste70] has been successfully implemented in the context of boundary element methods [CAB92, ZSL94]. g There are two major steps in the Stehfest algorithm. First, one needs to use n_s distinct parameters s for a given observation time t, e.g.

$$s_\nu = \frac{\ln 2}{t}\nu, \quad \nu = 1, 2, \cdots, n_s,$$

where n_s is the number of terms in the Stehfest algorithm and must be an even number. For each s_ν, we need to obtain a solution $U(\mathbf{x}, s_\nu)$ from (6.1.4) at any given point $\mathbf{x} \in \Omega \cup \Gamma^D$.

Second, one needs to invert the solution from the Laplace transform space to the time domain. The inversion procedure is

$$u(\mathbf{x},t) = \frac{\ln 2}{t}\sum_{\nu=1}^{n_s} W_\nu \cdot U(\mathbf{x}, s_\nu) \tag{6.1.7}$$

where

$$W_\nu = (-1)^{\frac{n_s}{2}+\nu} \sum_{k=\lfloor\frac{1}{2}(\nu+1)\rfloor}^{\min\{\nu,\frac{n_s}{2}\}} \frac{k^{\frac{n_s}{2}}(2k)!}{\left(\frac{n_s}{2}-k\right)!k!(k-1)!(\nu-k)!(2k-\nu)!}.$$

The accuracy of the Stehfest algorithm depends on the correct choice of the number n_s of terms in (6.1.7). As n_s increases, the accuracy improves first, but then round-off errors become dominant and finally the accuracy declines. This is a rather common phenomenon in practical numerical computation. The optimal n_s has a significant impact on the quality of the final solution of our proposed method. According to Stehfest's test of his algorithm on 50 test functions with known inverse Laplace transforms, he concluded that the optimal value of n_s is 10 for single precision variables and 18 for double precision variables. The accuracy of the results provided by Moridis and Reddell [MR91] and Zhu et. al. [ZSL94] differed little for values of n_s between 6 and 16.

Example 6.1.8 As a simple model problem [CGR98], we consider the flow of heat in Eq. (6.1.1) with $f(x, y, t) = 0$ in the finite rectangle $\Omega = [a, b]^2$ where $a = b = 0.2$ meters, and with a unit initial temperature and boundaries kept at zero temperature. The exact solution for the temperature distribution was given by Carslaw and Jaeger [CJ59] as

$$u^*(x, y, t) = \frac{16}{\pi^2} \sum_{n=0}^{\infty} \sum_{m=0}^{\infty} L_{n,m} \cos\frac{(2n+1)\pi x}{2a} \cos\frac{(2m+1)\pi y}{2b} e^{-D_{n,m}t}$$

where

$$\begin{aligned} L_{n,m} &= \frac{(-1)^{n+m}}{(2n+1)(2m+1)}, \\ D_{n,m} &= \frac{k\pi^2}{4}\left(\frac{(2n+1)^2}{a^2} + \frac{(2m+1)^2}{b^2}\right). \end{aligned}$$

Note that k is the thermal diffusivity of the substance with units of $[L^2][T^{-1}]$. In this example, we chose a thermal diffusivity of $k = 5.8 \times 10^{-7}\ m^2/\sec$ and an observation time t_{obs} of 9000 seconds. The solution profile is shown in Figure 6.1.

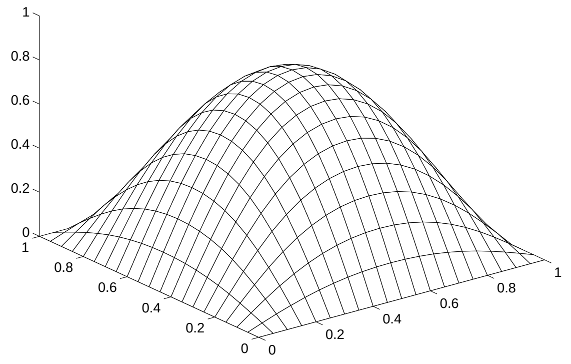

Fig. 6.1. The profile of the temperature distribution on the whole domain

To approximate the forcing term by thin–plate splines, we chose 20 evenly distributed collocation points in the domain. To obtain an approximate homogeneous solution by the Method of Fundamental Solutions, we chose 16 equally distributed collocation points on the boundary and the same number of source points evenly distributed on a circle with center at $(0, 0)$ and radius $r = 3.0$ meters.

The relative errors for various n_s on $(x, 0.025)$, $0 \le x \le 0.2$, are shown in Figure 6.2. We observed that the relative errors were bounded within 0.16% for $n_s \ge 10$. With increasing n_s the relative error becomes stable. The overall profile of the absolute error for $n_s = 12$ is shown in Figure 6.3. In the Laplace transform space, the forcing term of the modified Helmholtz equation is the constant $1/k$, and the thin–plate splines with the augmented linear term approximates $1/k$ exactly. Hence, the dominant error comes from the numerical inversion of the Laplace transform. Similar results can be expected for problems on general domains and with unknown exact solutions.

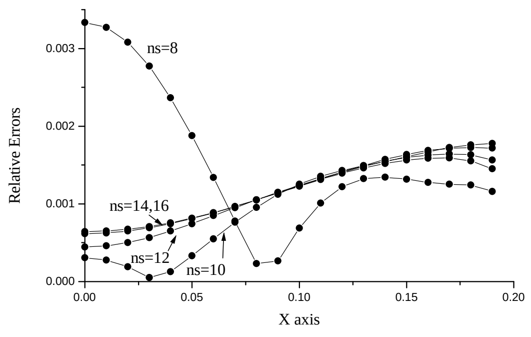

Fig. 6.2. Relative errors for various n_s

$\square$

6.2 Time–stepping Methods

Since numerical inversion of the Laplace transform is an ill-posed problem, small truncation errors are magnified in the numerical inversion process. As a result, the accuracy of the final result is greatly affected.

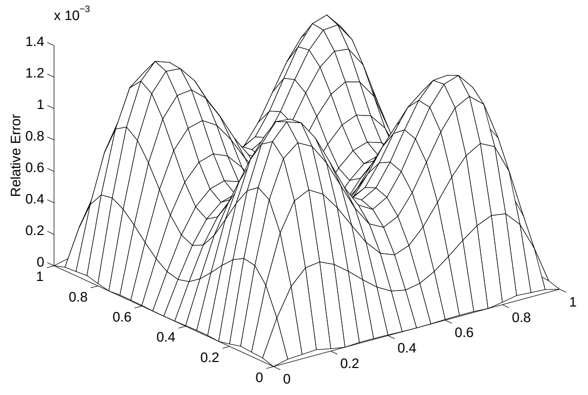

Fig. 6.3. Profile of absolute error using $n_s = 12$

In this section, we introduce another popular method for solving time–dependent problems using a time difference scheme.

The standard **Method of Lines** takes a spatial discretization of the domain Ω by points $x_j \in \Omega$, defines univariate functions $u_j(t) \approx u(x_j, t)$ and transforms the partial differential equation into a system of spatially coupled **ordinary differential equations** (ODEs) for these functions. Then one can use any package for solving large systems of ODEs for the numerical determination of the $u_j(t)$. These methods construct spatial discretizations at various time levels t_k determined by the ODE solver. Radial basis functions can then serve to provide the full spatial approximations $\tilde{u}(\mathbf{x}, t_k)$ as interpolants or approximants based on the data provided by the values $u_j(t_k) \approx u(\mathbf{x}_j, t_k)$. Therefore classical line methods based on radial basis functions reduce to spatial interpolation or approximation, at each time level, as described in the introductory sections. A somewhat more sophisticated recent variation expressing spatial derivatives of radial basis function approximations directly in terms of the spatial data is due to Fasshauer [Fas04, FF07] and closely related to **pseudospectral methods**.

But these methods do not take advantage of fundamental or particular solutions, as provided by the previous sections. Thus we now describe a technique which use these to construct approximations at various time levels, connecting them in time by certain **time–stepping methods**.

6.2.1 Diffusion Equation

A simple thermal diffusion process can be modeled by

$$\frac{1}{k}\frac{\partial u}{\partial t}(\mathbf{x},t) + f(\mathbf{x},t) = \Delta u(\mathbf{x},t) \tag{6.2.1}$$

for all $\mathbf{x}$ in a closed spatial domain $\Omega \subset I\!R^d, d = 2, 3$ with boundary Γ and for time $t \geq 0$. Here u is the temperature, f is the internal heat generator, and k is the thermal conductivity. If n is the outward unit normal vector and $q = \partial u/\partial n$ is the heat flux, one can prescribe three different types of boundary conditions

$$\begin{array}{rcrcll}
\text{temperature} & : & u(\mathbf{x},t) & = & \bar{u}(\mathbf{x}), & \mathbf{x} \in \Gamma^D, \\
\text{flux} & : & \frac{\partial u}{\partial n}(\mathbf{x},t) & = & -\bar{q}(\mathbf{x}), & \mathbf{x} \in \Gamma^N, \\
\text{convection} & : & \frac{\partial u}{\partial n}(\mathbf{x},t) & = & -h(u(\mathbf{x},t) - u_\infty), & \mathbf{x} \in \Gamma^C,
\end{array}$$

for all $t \geq 0$, where h is the heat transfer coefficient and u_∞ is the ambient temperature. The initial condition is given by

$$u(\mathbf{x}, 0) = g(\mathbf{x}),\ \mathbf{x} \in \Omega.$$

Now the spatial variables are discretized using a combination of the Method of Fundamental Solutions of Chapter 4 and the Method of Particular Solutions of Chapter 5, while time is discretized using the finite difference method.

Generalized trapezoidal methods (θ-methods) are a standard way of doing this. They are defined as follows [PBW92]. Let the time step be $\delta t > 0$ and define the time mesh $t_n = n\delta t, n \geq 0$. For $t_{n-1} \leq t \leq t_n$, approximate $u(\mathbf{x}, t)$ by

$$\begin{aligned}
u(\mathbf{x},t) &\simeq \theta u(\mathbf{x},t_n) + (1-\theta)u(\mathbf{x},t_{n-1}) \qquad (6.2.5)\\
f(\mathbf{x},t) &\simeq \theta f(\mathbf{x},t_n) + (1-\theta)f(\mathbf{x},t_{n-1}) \qquad (6.2.6)
\end{aligned}$$

so that

$$\Delta u(\mathbf{x},t) \simeq \theta\Delta u(\mathbf{x},t_n) + (1-\theta)\Delta u(\mathbf{x},t_{n-1}) \tag{6.2.7}$$

and

$$\frac{\partial u}{\partial t}(\mathbf{x},t) \simeq \frac{u(\mathbf{x},t_n) - u(\mathbf{x},t_{n-1})}{\delta t}.$$

Note that the implicit **Euler method** arises for $\theta = 1$ while the **Crank–Nicholson method** has $\theta = 1/2$. Let $\Gamma = \Gamma^D \cup \Gamma^N \cup \Gamma^C$ be a disjoint splitting of the spatial boundary. Then (6.2.1) can be discretized as

$$\frac{1}{k}\frac{u^n - u^{n-1}}{\delta t} = \theta(\Delta u^n - f^n) + (1-\theta)(\Delta u^{n-1} - f^{n-1}) \qquad (6.2.8)$$

where δt is the time step, $u^n = u(\mathbf{x}, t_n)$, $f^n = f(\mathbf{x}, t_n)$, and $0 < \theta \leq 1$. In particular, forward differences ($\theta = 0$) cannot be used in the present formulation. The transformed temperature is defined by

$$v^n = u^n + \frac{(1-\theta)}{\theta}u^{n-1}. \qquad (6.2.9)$$

Rearranging (6.2.8) gives

$$\Delta v^n - \lambda^2 v^n = -\frac{1}{\theta^2 k\delta t}u^{n-1} + \frac{(1-\theta)}{\theta}f^{n-1} + f^n \qquad (6.2.10)$$

where $\lambda^2 = 1/\theta k\delta t$ and $u^0(\mathbf{x}) = g(\mathbf{x})$. We note that after v^n is evaluated, u^n can be retrieved from (6.2.9).

Now the diffusion equation (6.2.1) has been transformed into a series of inhomogeneous modified Helmholtz equations (6.2.10) which can be solved by a boundary type approach. The most challenging part of the solution procedure is to effectively obtain an approximate particular solution.

To solve (6.2.1) at each time step, our standard procedure of splitting the given problem into a homogeneous solution v_h^n, and a particular solution v_p^n is applied, i.e. $v^n = v_h^n + v_p^n$. The governing equation for the particular solution is given by

$$\Delta v_p^n - \lambda^2 v_p^n = -\frac{1}{\theta^2 k\delta t}u^{n-1} + \frac{(1-\theta)}{\theta}f^{n-1} + f^n. \qquad (6.2.11)$$

As we have indicated in the previous chapters, the particular solution v_p^n is not required to satisfy any boundary condition. The governing system for the homogeneous solution v_h^n is then given by

$$\begin{aligned}
\Delta v_h^n(\mathbf{x}) - \lambda^2 v_h^n(\mathbf{x}) &= 0, && \mathbf{x} \in \Omega,\\
v_h^n(\mathbf{x}) &= \bar{u}(\mathbf{x}) - v_p^n(\mathbf{x}), && \mathbf{x} \in \Gamma^D,\\
q_h^n(\mathbf{x}) &= \bar{q}(\mathbf{x}) - q_p^n(\mathbf{x}), && \mathbf{x} \in \Gamma^N,\\
(kq_h^n + hv_h^n)(\mathbf{x}) &= -kq_p^n(\mathbf{x}) - hv_p^n(\mathbf{x}) + hu_\infty(\mathbf{x}), && \mathbf{x} \in \Gamma^C.
\end{aligned}$$

The right-hand side of (6.2.11) is completely known at each time step and can be approximated by polyharmonic splines as shown in Chapter 2.

Example 6.2.12 Consider the parabolic equation

$$\frac{\partial u}{\partial t}(x_1, x_2, t) = \Delta u(x_1, x_2, t) + f(x_1, x_2, t)$$

in the 2D domain $[-2, 2] \times [-2, 2]$. The forcing term is given by

$$f(x_1, x_2, t) = \sin x_1 \sin x_2 (2 \sin t + \cos t).$$

The initial and boundary conditions lead to the solution

$$u^*(x_1, x_2, t) = \sin x_1 \sin x_2 \sin t.$$

To approximate the forcing term f, we chose 132 interpolation points in which 32 boundary points were used as the collocation points for the Method of Fundamental Solutions. We chose the polyharmonic spline ($r^4 \ln r$) of order two from Table 1.3 as the radial basis function. To evaluate the homogeneous solution using the Method of Fundamental Solutions, we chose 32 source points on a circle with center at $(0, 0)$ and radius 8.

To verify the effectiveness of the method, we computed the errors on a point randomly chosen at $(0.889, 0.889)$. The errors for $\delta t = 0.05$ and $\delta t = 0.025$ are shown in Figure 6.4. Similar results were also obtained at other points. The results are highly accurate for the smaller time step δt. Increasing the number of interpolation points has only a small influence on the numerical accuracy.

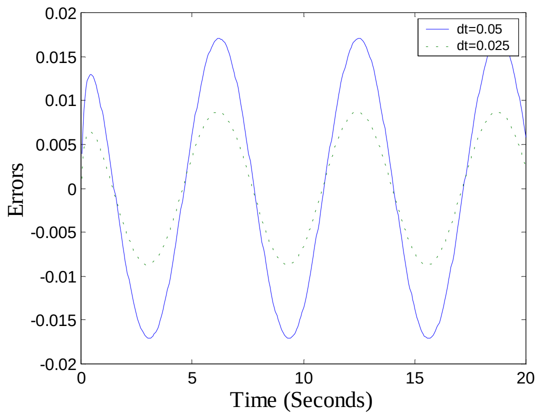

Fig. 6.4. Example 6.2.12. Errors for $\delta t = 0.05$ and $\delta t = 0.025$ at $(0.889, 0.889)$.

To evaluate the global error, we chose 400 evenly distributed points on the domain and computed the errors using $\delta t = 0.0125$ at $t = 10$. The profile of the error over the domain is shown in Figure 6.5, while the exact solution at $t = 10$ is displayed in Figure 6.6. The error decreases linearly with the time step chosen, as shown in Figure 6.7.

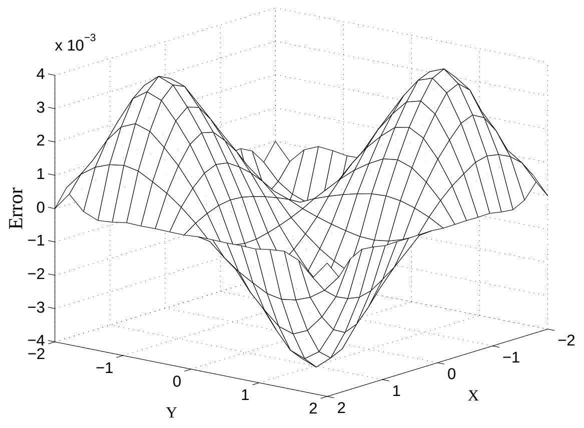

Fig. 6.5. Example 6.2.12. Errors at $t = 10$ using $\delta t = 0.0125$.

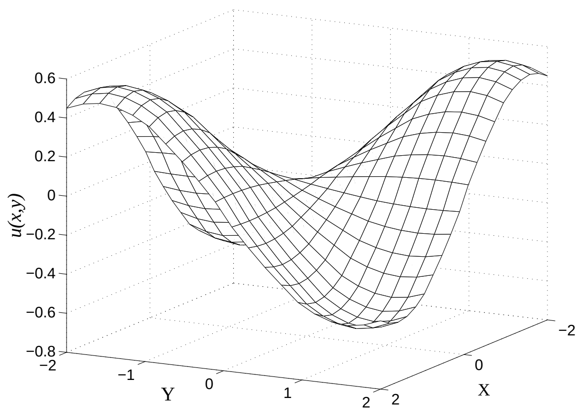

Fig. 6.6. Example 6.2.12. Exact solution at $t = 10$.

□

Example 6.2.13 Consider the diffusion equation (6.2.1) in a 3D cube $0 \leq x_1, x_2, x_3 \leq 1$ with $k = 1$ [ICT04]. The boundary conditions are

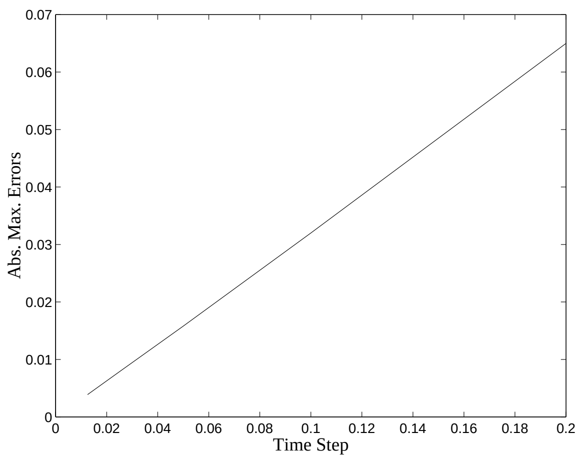

Fig. 6.7. Example 6.2.12. Error in terms of the time discretization.

given by

$$\begin{aligned} u(0,x_2,x_3,t) &= 0, u(1,x_2,x_3,t) = 1, && (6.2.14) \\ \frac{\partial u}{\partial n}(x_1,x_2,0,t) &= \frac{\partial u}{\partial n}(x_1,x_2,1,t) = 0, && (6.2.15) \\ \frac{\partial u}{\partial n}(x_1,0,x_3,t) &= \frac{\partial u}{\partial n}(x_1,1,x_3,t) = 0 && (6.2.16) \end{aligned}$$

on the sides of the cube and for positive t. The initial condition is given by $u(x_1,x_2,x_3,0) = 0$ for $0 < x_1,x_2,x_3 < 1$. This problem represents the problem of an insulated unit bar with initial temperature 0 whose left-hand boundary is isothermal at 0 and whose right hand boundary is impulsively raised to 1 at time $t = 0$. The exact solution can be determined using a separation of variables and is given in [O'N99].

The domain is discretized with 218 source points outside the domain and 343 collocation points, of which 218 are located on the surface of the domain while the remaining 125 collocation points are located in the interior of the domain. The radius of the fictitious boundary is 8. In Table 6.1, PS1, PS2 and PS3 denote errors obtained via polyharmonic splines $r^{2k-1}, k = 1,2,3$, respectively. In general, the results get better with higher-order polyharmonic splines and reduced time steps, but this is not always the case. Care must be taken when either reducing the time step or going to higher-order polyharmonic splines since the higher-order polyharmonic splines result in worse conditioning of the linear system associated with the particular solution, while smaller time steps result in worse condition of the linear system associated with the homogeneous solution. In fact, for small time steps, the Euler implicit method ($\theta = 1$) is more accurate than the Crank–Nicholson method ($\theta = 1/2$) despite the difference in the local truncation error for the two methods. The current results are consistent with findings of Muleshkov et. al. [MGC99] who showed that significant improvement in accuracy can be obtained using higher-order polyharmonic splines for elliptic boundary value problems. Further, they observed limited improvement for time–dependent problems, presumably because the dominating error was caused by the time-stepping scheme. □

θ	Δt	PS1	PS2	PS3
0.5	0.01	$1.00E-2$	$2.83E-3$	$1.38E-2$
	0.005	$1.57E-2$	$2.77E-3$	$2.40E-3$
	0.002	$2.92E-2$	$2.47E-3$	$1.96E-3$
	0.001	$4.56E-2$	$2.50E-3$	overflow
1	0.01	$1.73E-2$	$1.39E-2$	$1.38E-2$
	0.005	$1.35E-2$	$7.20E-3$	$7.07E-3$
	0.002	$1.84E-2$	$3.31E-3$	$3.02E-3$
	0.001	$2.86E-2$	$2.26E-3$	$1.76E-3$

Table 6.1. *Example 6.2.13. Absolute maximum errors at $t = 1$ using polyharmonic splines.*

6.2.2 Convection-Diffusion Problems

The standard two-dimensional **convection-diffusion equation** is of the form

$$\frac{1}{k}\frac{\partial u}{\partial t} + w_x \frac{\partial u}{\partial x} + w_y \frac{\partial u}{\partial y} + f = \Delta u, \tag{6.2.17}$$

where w_x and w_y are the components of a variable velocity vector field and f is the source term. The solution to this problem is formulated in the same manner as before by explicitly combining the convection and source terms, i.e.

$$F = w_x \frac{\partial u}{\partial \mathbf{x}} + w_y \frac{\partial u}{\partial y} + f.$$

In this way, (6.2.17) can be solved like (6.2.1) in the diffusion case. Consequently,

$$\Delta v_p^n - \lambda^2 v_p^n = -\frac{1}{\theta^2 k \delta t} u^{n-1} + \frac{(1-\theta)}{\theta} F^{n-1} + F^n$$

is another inhomogeneous modified Helmholtz equation which can be solved similar to the case of diffusion equations as before.

6.2.3 Semi-nonlinear Diffusion Equation

If the convection term in (6.2.17) is replaced by a general nonlinear term of the form $f(\mathbf{x}, t, u, \nabla u)$, the method given above can be modified to solve the **nonlinear diffusion equation**

$$\frac{1}{k}\frac{\partial u}{\partial t}(\mathbf{x}, t) + f(\mathbf{x}, t, \nabla u, u) = \Delta u(\mathbf{x}, t). \tag{6.2.18}$$

Again, $u(\mathbf{x}, t)$ can be approximated by (6.2.5) using the θ–method. In the next example, we choose $\theta = 1$ Similar to the derivation of (6.2.10), we can reformulate (6.2.18) as

$$\Delta v^n - \lambda^2 v^n = -\frac{1}{\theta^2 k \delta t} u^{n-1} + f^{n-1},$$

where v^n is defined in (6.2.9). Interestingly, this scheme requires no spatial iteration. This is in sharp contrast to the steady state case $\partial u/\partial t = 0$ [Che95].

Example 6.2.19 We first consider the two-dimensional **Fisher equation** [Bri86, Lan99]

$$\frac{\partial u}{\partial t} = \Delta u + \kappa u(1-u) \quad \text{in} \quad \Omega \times (0, T],\ \kappa \geq 0, \tag{6.2.20}$$

with initial and boundary conditions

$$\begin{aligned} u(\mathbf{x}, 0) &= J_0\left(c\sqrt{\frac{x_1^2}{9} + x_2^2}\right) \quad \text{in} \quad \Omega, && (6.2.21) \\ u(\mathbf{x}, t) &= 0 \quad \text{on} \quad \Gamma \times (0, T], && (6.2.22) \end{aligned}$$

where J_0 is the first kind Bessel function of order zero, $c \simeq 2.4048$ is the first zero of J_0 and $\mathbf{x} = (x_1, x_2)$. The physical domain is an ellipse $\Omega \cup \Gamma = \{(x_1, x_2) : x_1^2/9 + x_2^2 \leq 1\}$.

The existence and uniqueness of a solution to (6.2.20)-(6.2.22) was proven by Britton [Bri86]. Furthermore, there exists a critical value $\delta^* > 0$ such that the zero solution is a stable steady state for $\kappa < \delta^*$, while it is unstable for $\kappa > \delta^*$. Finally, for $\kappa > \delta^*$ there exists at least one non–trivial non–negative solution of (6.2.20)-(6.2.22). Langdon [Lan99] verified these results numerically by using a domain embedding boundary integral method. Both approaches were carried out in a natural way and do not require spatial iteration; i.e. a nonlinear problem can be solved the same way as a linear one.

To handle this problem, we used thin–plate splines as basis functions [GC01]. A total of 110 quasi-random points, including 32 uniformly distributed points on the boundary, were chosen in $\Omega \cup \Gamma$ for interpolating the forcing term. The Method of Fundamental Solutions was used to calculate homogeneous solutions, based on 32 evenly spaced points chosen on a fictitious boundary which is a circle with center at $(0, 0)$ and radius 15.

Figure 6.8 shows the convergence to a trivial steady state for $\kappa = 1, 3$ and to non–trivial steady states for $\kappa = 5$ and 10 at the observation point $(0, 0)$. Here we used the backward time difference scheme ($\theta = 1$) with $\delta t = 0.02$.

From Figure 6.8, we can deduce that the critical value δ^* satisfies $3 < \delta^* < 5$. Refined bounds on δ^* can be obtained by the bisection method [Che95]. In Table 6.2, the results of our technique are in excellent agreement with [Lan99]. To evaluate the non–trivial steady state solution of (6.2.20)-(6.2.22), we take t sufficiently large such that $\|\hat{v}_{n+1} - \hat{v}_n\|_\infty < 10^{-6}$. The profiles of non–trivial steady state solutions for $\kappa = 5$ are given in Figure 6.9. Based on our numerical results on the 110 quasi-random points, we used a standard thin–plate spline surface reconstruction method to produce the graph in Figure 6.9.

□

Example 6.2.23 We further consider the **Burgers equation**

$$\begin{aligned} \frac{\partial u}{\partial t} - \alpha \Delta u &= -u\left(\frac{\partial u}{\partial x_1} + \frac{\partial u}{\partial x_2}\right), && \text{in } \Omega \times (0, T], \\ u(\mathbf{x}, t) &= g_1(\mathbf{x}, t), && \text{on } \Gamma \times (0, T], \\ u(\mathbf{x}, 0) &= g_2(\mathbf{x}), && \text{in } \Omega, \end{aligned}$$

in two dimensions on $\Omega = [0, 1] \times [0, 1]$ [LHC02]. The boundary condition $g_1(\mathbf{x}, t)$ and initial condition $g_2(\mathbf{x})$ are chosen to satisfy the exact solution

$$u^*(x_1, x_2, t) = \frac{1}{1 + \exp\left(\dfrac{x_1 + x_2 - t}{2\alpha}\right)}. \tag{6.2.24}$$

	$u(0,0)$ (Langdon)			$u(0,0)$ (MFS-MPS)		
Time	$k=1$	$k=5$	$k=10$	$k=1$	$k=5$	$k=10$
0.5	3.46E-1	6.14E-1	9.04E-1	3.45E-1	6.15E-1	9.03E-1
1.0	1.41E-1	5.48E-1	8.67E-1	1.40E-1	5.49E-1	8.69E-1
1.5	5.96E-2	5.22E-1	8.50E-1	5.89E-2	5.23E-1	8.54E-1
2.0	2.54E-2	5.10E-1	8.42E-1	2.50E-2	5.10E-1	8.47E-1
2.5	1.08E-2	5.04E-1	8.37E-1	1.06E-2	5.04E-1	8.43E-1
3.0	4.66E-3	5.01E-1	8.34E-1	4.55E-3	5.01E-1	8.40E-1
3.5	1.99E-3	5.00E-1	8.32E-1	1.94E-3	5.00E-1	8.39E-1
4.0	8.57E-4	4.99E-1	8.31E-1	8.31E-4	4.99E-1	8.38E-1
4.5	3.67E-4	4.99E-1	8.30E-1	3.55E-4	4.99E-1	8.38E-1
5.0	1.57E-4	4.99E-1	8.29E-1	1.51E-4	4.99E-1	8.37E-1

Table 6.2. *Example 6.2.19. Comparison of* $u(0,0)$ *using different methods.*

The solution in (6.2.24) is a wave front moving along the line x_1+x_2-t. The profile for $\alpha = 0.05$ and $t = 1.0$ is shown in Figure 6.10. The numerical solutions were obtained by choosing the fully implicit time stepping scheme $\theta = 1$ and the time step $\delta t = 0.01$. From (6.2.9)-(6.2.10), we have

$$\Delta u^n - \frac{1}{\alpha\delta t}u^n = -\frac{1}{\alpha\delta t}u^{n-1} + \frac{1}{\alpha}u^{n-1}\left(\frac{\partial u^{n-1}}{\partial x_1} + \frac{\partial u^{n-1}}{\partial x_2}\right).$$

The spatial domain Ω is discretized by 19×19 uniform grid points inside Ω. Third order polyharmonic splines ($r^6 \ln r$) are chosen to approximate the forcing term. For the implementation of the Method of Fundamental Solutions, 32 points were evenly distributed on the boundary Γ. For the initial step, $u^0 = g_2(\mathbf{x})$ and

$$\frac{\partial u^0}{\partial x_i} = \frac{\partial g_2}{\partial x_i},\ i = 1, 2,$$

are used. For the time step $n \geq 1$, we have

$$\frac{\partial u^n}{\partial x_i} = \frac{\partial u_h^n}{\partial x_i} + \frac{\partial u_p^n}{\partial x_i},\ i = 1, 2,$$

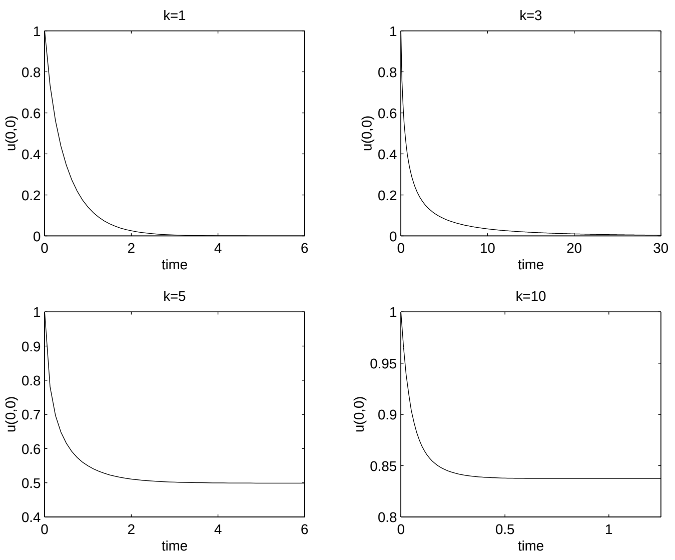

Fig. 6.8. Example 6.2.19. Approximate value of $u(0,0)$ against time for $\kappa = 1, 3, 5$, and 10.

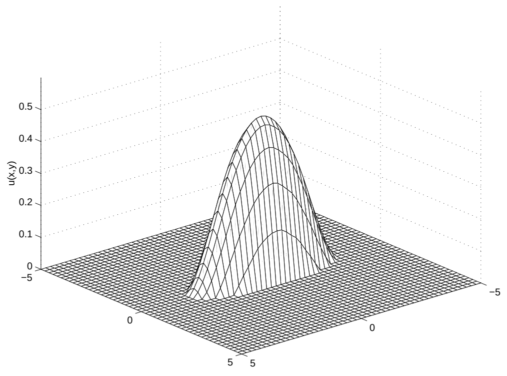

Fig. 6.9. Example 6.2.19. Profile of steady state solution for $\kappa = 5$.

where $\partial u_h^n / \partial x_i$ and $\partial u_p^n / \partial x_i$ can be obtained using (4.2.10) and (5.4.7), respectively. The profiles of the absolute error (left) and exact solution (right) at time $t = 1$ are shown in Figure 6.10. $\square$

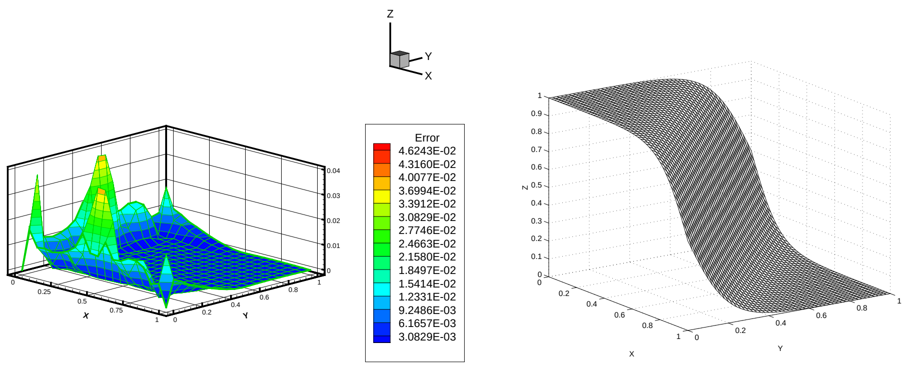

Fig. 6.10. Error and solution profiles at $t = 1$.

6.3 Operator-Splitting Methods

Consider the **nonlinear diffusion–reaction equation**

$$\frac{\partial u}{\partial t} = \Delta u + f(u) \quad \text{on} \quad \Omega \tag{6.3.1}$$

with boundary conditions

$$\begin{aligned} u(\mathbf{x}, t) &= u_0(\mathbf{x}, t), && \text{on } \Gamma^D, \\ \frac{\partial u}{\partial n}(\mathbf{x}, t) &= 0, && \text{on } \Gamma^N, \end{aligned}$$

where $\Gamma = \Gamma^D \cup \Gamma^N$ denotes the boundary of Ω, and f is a nonlinear forcing function. The **operator splitting method** is a procedure to separate the elliptic operator from the nonlinear forcing term, followed by an approach similar to the alternating direction implicit (**ADI**) procedure proposed by Peaceman and Rachford [PR55]. Balakrishnan et. al. [BSR02] generalized the procedure in the context of the Method of Fundamental Solutions and Radial Basis Functions.

In general, a time–dependent problem can be written as

$$\frac{\partial u}{\partial t} = L_1 + L_2 + \cdots L_n$$

via a sum of n operators $L_1, L_2, \cdots, L_n$. In (6.3.1), we have the following expressions

$$\begin{aligned} L_1 &= f(u) \qquad (6.3.2) \\ L_2 &= \Delta u. \qquad (6.3.3) \end{aligned}$$

To discretize $\partial u / \partial t$, a two-time-step finite difference scheme can be used.

More specifically, the non–linear forcing term in (6.3.2) is evaluated explicitly at each half time step and the resulting solution is then used for an implicit solution of the Laplacian term in (6.3.3). For the first half step, we can use the second order explicit **Adams-Bashforth scheme** and obtain

$$\frac{u^{n+1/2} - u^n}{\delta t} = \frac{3}{2} f(u^n) - \frac{1}{2} f(u^{n-1}). \tag{6.3.4}$$

The solution $u^{n+1/2}$ from (6.3.4) is then used in the next half step, which can be discretized implicitly using the second order **Adams-Moulton scheme** as

$$\frac{u^{n+1} - u^{n+1/2}}{\delta t} = \frac{1}{2}(\Delta u^{n+1} + \Delta u^n). \tag{6.3.5}$$

We observe that the non–linear term in (6.3.4) is evaluated explicitly without any spatial iteration. We also note that the evaluation of the Laplace operator for the previous step in (6.3.5) will add some error to the one induced by the time difference scheme. To circumvent this difficulty, the dependent variable in (6.3.5) is transformed to u^* via

$$u^* \equiv \frac{u^n + u^{n+1}}{2}. \tag{6.3.6}$$

Then

$$\Delta u^* - \frac{2u^*}{\delta t} = -(\frac{3}{2} f^n - \frac{1}{2} f^{n-1}) - \frac{2u^n}{\delta t} \tag{6.3.7}$$

can be achieved by using (6.3.4)-(6.3.6) and some algebraic manipulations. Again, we observe that (6.3.7) is a modified Helmholtz equation, solvable by the procedures of the previous chapters. Fortunately, at each time step the non–homogeneous term is explicitly known. Furthermore, u^* can be computed at each time step by the solution of (6.3.7). The values of u^{n+1} are then extracted from the computed values of u^* through (6.3.6). We remark that the whole solution process needs the function values at the current and previous time step. Hence, the method itself is not self-starting, and an extrapolated explicit forward Euler method is used for simplicity for the first time step to initiate the time stepping.

For numerical examples and further details, we refer the reader to the work of Balakrishnan et. al. [BSR02].

6.3.1 Stokes Flow

The **Stokes flow problem** is a simplification of the **Navier-Stokes problem**, in which the nonlinear convective terms are very small and can be neglected. For the solution of Stokes flows using the **velocity–vorticity formulation**, the governing equations can be written as a system of diffusion-type and Poisson–type equations for the components of the vorticity and velocity field, respectively. This means

$$\begin{cases} \dfrac{\partial \omega}{\partial t} = \Delta \omega \\ \Delta \mathbf{u} = -\nabla \times \omega \end{cases} \quad \text{in } \Omega \tag{6.3.8}$$

$$\mathbf{u} = \mathbf{U} \quad \text{on } \Gamma \tag{6.3.9}$$

where $\mathbf{u}$ is the velocity vector, ω the vorticity vector, $\mathbf{U}$ the given boundary velocity and $\Omega,\ \Gamma$ denote the domain and its boundary, respectively. Furthermore, the vorticity vector ω can be expressed as

$$\omega = \nabla \times \mathbf{u}.$$

One of the advantages of this formulation is the separation of the kinematic and kinetic aspects of the fluid flow.

The general numerical procedure for solving (6.3.8) and (6.3.9) can be described as follows:

(i) Set the initial value to be zero:

$$\begin{aligned} \mathbf{u}^0(\mathbf{x}) &= (u_1^0(\mathbf{x}), u_2^0(\mathbf{x}), u_3^0(\mathbf{x})) &&= (0,0,0) \\ \omega^0(\mathbf{x}) &= (\omega_1^0(\mathbf{x}), \omega_2^0(\mathbf{x}), \omega_3^0(\mathbf{x})) &&= (0,0,0) \end{aligned}$$

and $\mathbf{u}^n = \mathbf{u}(\mathbf{x}, n\delta t), \omega^n = \omega(\mathbf{x}, n\delta t)$.

(ii) Solve $\Delta \mathbf{u}^{n+1} = 0$ with the given boundary velocity condition $\mathbf{u} = \mathbf{U}$ (Laplace equation).

(iii) Obtain the vorticity on the boundary using $\omega^{n+1} = \nabla \times \mathbf{u}^{n+1}$.

(iv) Solve the diffusion equation in (6.3.8) for ω^{n+1} with the boundary condition from step (iii).

(v) Solve the velocity equation in (6.3.9) with the given boundary condition (Poisson equation) for $\mathbf{u}^{n+1}$. The forcing term is the derivative of the vorticity components obtained from step (iv).

(vi) Repeat step (iii)-(v) until the solutions are convergent.

6.4 Wave Equation

The initial-boundary value problem for the **wave equation** is

$$\begin{aligned}
\frac{\partial^2 u}{\partial t^2}(\mathbf{x},t) &= \Delta u(\mathbf{x},t) - cu(\mathbf{x},t) - f(\mathbf{x},t), & \mathbf{x}\in\Omega,\ t>0,\\
u(\mathbf{x},0) &= g_1(\mathbf{x}), & \mathbf{x}\in\Omega,\\
\frac{\partial u}{\partial t}(\mathbf{x},0) &= g_2(\mathbf{x}), & \mathbf{x}\in\Omega,\\
u(\mathbf{x},t) &= h(\mathbf{x},t), & \mathbf{x}\in\Gamma,\ t\geq 0.
\end{aligned} \tag{6.4.1}$$

Again we consider only the class of generalized trapezoidal methods as described in Section 6.2.1. Defining $u^n \equiv u(\mathbf{x}, t_n)$ again, we approximate

$$\frac{\partial^2 u}{\partial t^2} \simeq \frac{u^{n+1} - 2u^n + u^{n-1}}{\delta t^2} \tag{6.4.2}$$

and use the formulas of the $\theta-$method as in (6.2.5)-(6.2.7) to get

$$\begin{aligned}
\Delta u - cu - f \quad \simeq \quad & \theta\left(\Delta u^{n+1} - cu^{n+1} - f^{n+1}\right)\\
& + (1-\theta)\left(\Delta u^n - cu^n - f^n\right).
\end{aligned} \tag{6.4.3}$$

When using (6.4.2)-(6.4.3) for (6.4.1), u_n satisfies

$$\begin{aligned}
\frac{u^{n+1} - 2u^n + u^{n-1}}{\delta t^2} = \quad & \theta\Delta u^{n+1} + (1-\theta)\Delta u^n\\
& -c\left(\theta u^{n+1} + (1-\theta)u^n\right)\\
& -\left(\theta f^{n+1} + (1-\theta)f^n\right).
\end{aligned} \tag{6.4.4}$$

To avoid evaluating Δu^n at each step, we use the same transformation as in (6.2.9). Rearranging (6.4.4) gives

$$\Delta v^{n+1} - \lambda^2 v^{n+1} = -\frac{1+\theta}{\theta^2 \delta t^2}u^n + \frac{1}{\theta\delta t^2}u^{n-1} + f^{n+1} + \frac{1-\theta}{\theta}f^n \tag{6.4.5}$$

where

$$\lambda^2 = c + \frac{1}{\theta\delta t^2},$$

and v is defined in (6.2.9). The right hand side of (6.4.5) contains only the known values of the previous step. When $\theta = 1$, we obtain the following implicit formulation

$$\Delta u^{n+1} - \lambda^2 u^{n+1} = -\left(\frac{2u^n - u^{n-1}}{\delta t^2}\right) + f^n.$$

With the approximation

$$\frac{\partial u}{\partial t} \simeq \frac{u^{n+1} - u^n}{\delta t},$$

we get the initial conditions

$$u^0(\mathbf{x}) = g_1(\mathbf{x}), \ \mathbf{x} \in \Omega \cup \Gamma, \tag{6.4.6}$$

$$u^1(\mathbf{x}) = g_1(\mathbf{x}) + \delta t g_2(\mathbf{x}), \ \mathbf{x} \in \Omega \cup \Gamma, \tag{6.4.7}$$

and boundary conditions

$$u^{n+1}(\mathbf{x}) = h^{n+1}(\mathbf{x}), \ \mathbf{x} \in \Gamma.$$

Note that (6.4.5) is again a modified Helmholtz equation. Hence at each time step u^n satisfies a boundary-value problem for a modified Helmholtz equation. The solution procedure is quite similar to the diffusion equation using the approach described in Section 6.2.1.

Example 6.4.8 Consider the wave equation

$$\begin{aligned}
\frac{\partial^2 u}{\partial t^2}(\mathbf{x},t) &= \Delta u(\mathbf{x},t), && \mathbf{x} \in \Omega, \ t > 0\\
u(\mathbf{x},0) &= \sin^2(\pi x_1)\sin(\pi x_2), && \mathbf{x} \in \Omega,\\
\frac{\partial u}{\partial t}(\mathbf{x},0) &= 0, && \mathbf{x} \in \Omega,\\
u(\mathbf{x},t) &= 0, && \mathbf{x} \in \Gamma,
\end{aligned}$$

where $\mathbf{x} = (x_1, x_2)$, $\Omega \cup \Gamma = [0,1] \times [0,1]$. The exact solution is given by

$$u(x_1,x_2,t) = \frac{-8}{\pi} \sum_{n=1,3,5,\cdots}^{\infty} \frac{\cos\left(\pi t\sqrt{n^2+1}\right)}{n(n^2-4)} \sin n\pi x_1 \sin \pi x_2.$$

To evaluate the particular solution at each time step, we chose 121 evenly distributed grid points on $\Omega \cup \Gamma$. As our radial basis function we chose polyharmonic splines (1.3) of order two, i.e. $\phi(r) = r^4 \ln r$. To solve the homogeneous equation, we chose 32 collocation points on the boundary and 32 source points on the fictitious circle with center at $(0.5, 0.5)$ and radius $r = 2$. For the time-stepping scheme, we chose the fully implicit method with $\theta = 1$. Figure 6.11 shows both the profiles of the exact and the approximate solution at $(0.5, 0.5)$ over the time interval $0 \leq t \leq 6$. We have noticed that the larger errors occur near turning points of each cycle. The accuracy when using the smaller time step (dt $= 0.001$) is far better than the larger time step (dt $= 0.01$). The profiles of the exact solution and the error $u - u^n$ on the whole domain at $t = 2.8, 6$ using $\delta t = 0.001$ are shown in Figures 6.12-6.13. □

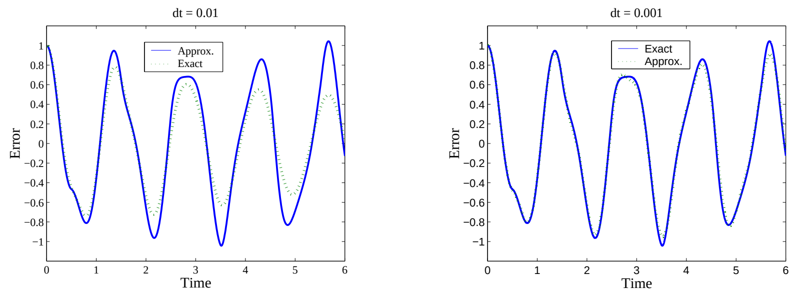

Fig. 6.11. Example 6.4.8: $u^n(0.5, 0.5)$ for $\delta t = 0.01$ and 0.001

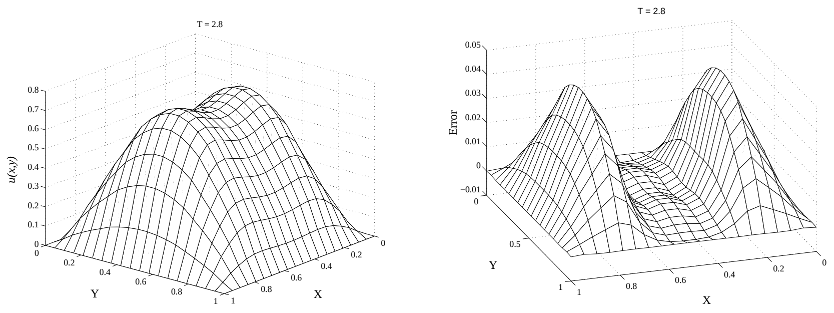

Fig. 6.12. Example 6.4.8: The profiles of u^n (left) and errors (right) at $t = 2.8$ using $\delta t = 0.001$

Example 6.4.9 Consider the wave equation

$$\begin{aligned}
\frac{\partial^2 u}{\partial t^2}(\mathbf{x}, t) &= \Delta u(\mathbf{x}, t) + f(\mathbf{x}, t), && \mathbf{x} \in \Omega,\ t > 0, \\
u(\mathbf{x}, 0) &= x_1(1 - x_1)x_2(1 - x_2), && \mathbf{x} \in \Omega, \\
\frac{\partial u}{\partial t}(\mathbf{x}, 0) &= 0, && \mathbf{x} \in \Omega, \\
u(\mathbf{x}, t) &= x_1(1 - x_1)x_2(1 - x_2)\cos(t), && \mathbf{x} \in \Gamma,
\end{aligned}$$

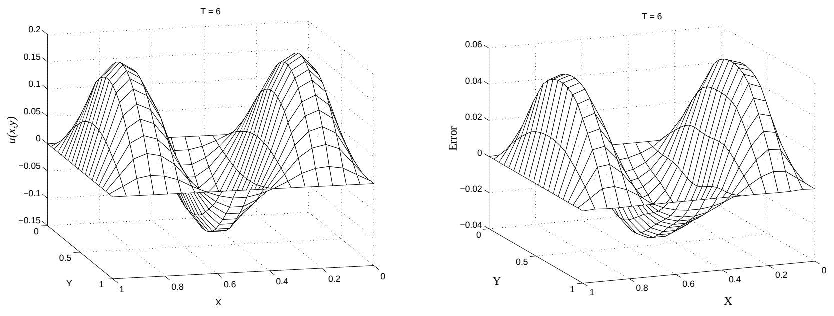

Fig. 6.13. The profiles of u^n (left) and errors (right) at $t = 6$ using $\delta t = 0.001$.

where

$$f(\mathbf{x},t) = (2x_1(1-x_1) + 2x_2(1-x_2) - x_1(1-x_1)x_2(1-x_2))\cos(t)$$

and $\mathbf{x} = (x_1, x_2)$, $\Omega \cup \Gamma = [0,1] \times [0,1]$. The exact solution is

$$u^*(x_1, x_2, t) = x_1(1-x_1)x_2(1-x_2)\cos(t).$$

The number of collocation points, the radial basis function, and the time-stepping scheme are like in the previous example. For the time step we chose $\delta t = 0.05$. From (6.4.7), the initial condition $\frac{\partial u}{\partial t} = 0$ implies $u^0 = u^1$. The profiles of $u(x_1, x_2)$ and the errors at $t = 1, 3, 5$ are shown in Figures 6.14 to 6.16. The solution u is vibrating up and down. In Figure 6.17, we show the errors of $u(0.5, 0.5, t)$ for $0 \le t \le 20$ using time steps $\delta t = 0.01$, and 0.025. This shows that the error is oscillating, but stable. However, the smaller time step did not improve the error as much as in the last example and for the diffusion equation in Example 6.2.12. We offer no further explanation here as no error analysis is available. It is possible that the source term $f(\mathbf{x}, t)$ cancels the damping effect as shown in the previous example.

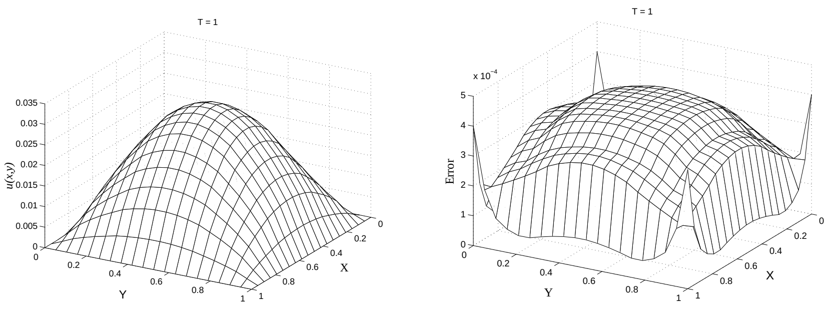

Fig. 6.14. Example 6.4.9: The profiles of u^n (left) and errors (right) at $t = 1$.

□

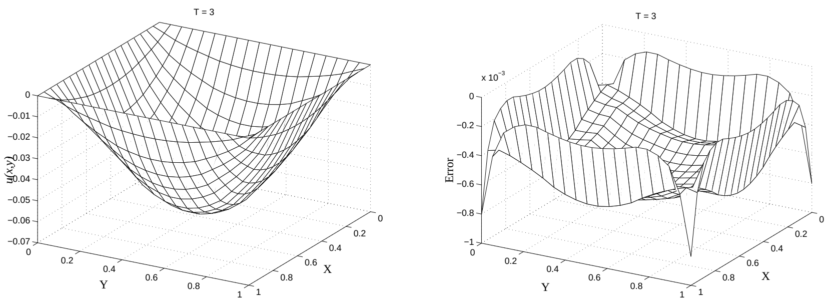

Fig. 6.15. Example 6.4.9: The profiles of u^n (left) and errors (right) at $t = 3$.

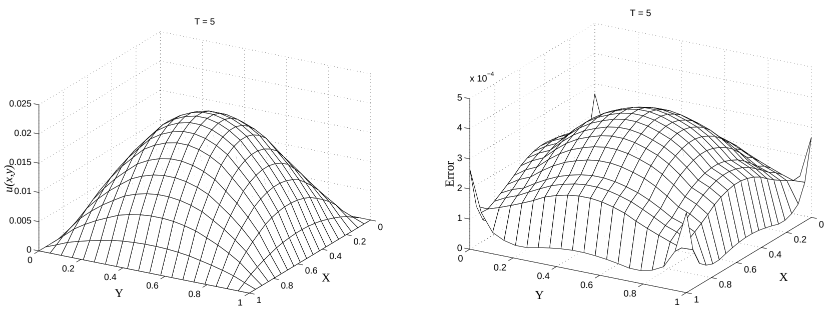

Fig. 6.16. Example 6.4.9: The profiles of u^n (left) and errors (right) at $t = 5$.

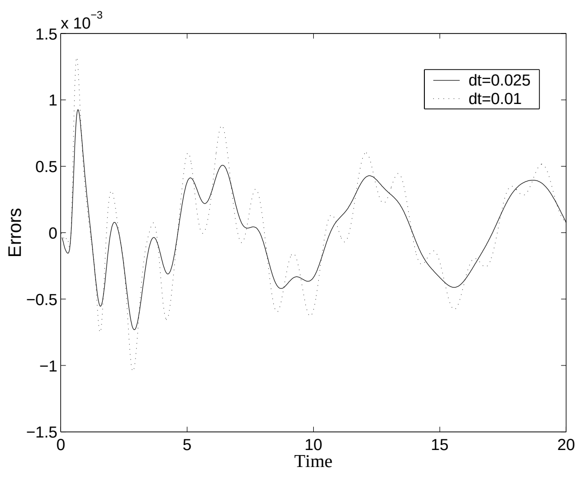

Fig. 6.17. Example 6.4.9: $u(0.5, 0.5, t)$ for $\delta t = 0.01$ and 0.025.

References

Bibliography

[AA05] C.J.S. Alves and P.R.S. Antunes, *The method of fundamental solutions applied to the calculation of eigenfrequencies and eigenmodes of 2d simply connected shapes*, Compters, Materials, and Continua (2005), 251–266.

[ABW06] R. Ahrem, A. Beckert, and H. Wendland, *A meshless spatial coupling scheme for large-scale fluid-structure-interaction problems*, Computer Modeling in Engineering and Sciences **12** (2006), 121–136.

[AC01] C.J.S. Alves and C.S. Chen, *The MFS method adapted for a nonhomogeneous equation*, Advances in Computational Engineering & Sciences (S.N. Atluri, T. Nishioka, and M. Kikuchi, eds.), Tech Science Press, Los Angeles, CD Rom, # 135, 2001.

[AC05] ———, *Approximating functions and solutions of non homogeneous partial differential equations using the method of fundamental solutions*, Advances in Computational Mathematics (2005), 125–142.

[ADR94] N. Arad, N. Dyn, and D. Reisfeld, *Image warping by radial basis functions: applications to facial expressions*, Graphical Models and Image Processing **56** (1994), 161–172.

[AS65] M. Abramowitz and I.A. Stegun, *Handbook of Mathematical Functions*, Dover, New York, 1965.

[Atk85] K.E. Atkinson, *The numerical evaluation of particular solutions for Poisson's equation*, IMA Journal of Numerical Analysis **5** (1985), 319–338.

[AZ98a] S.N. Atluri and T.-L. Zhu, *A new meshless local Petrov-Galerkin (MLPG) approach in Computational mechanics*, Computational Mechanics **22** (1998), 117–127.

[AZ98b] ———, *A new meshless local Petrov-Galerkin (MLPG) approach to nonlinear problems in Computer modeling and simulation*, Computer Modeling and Simulation in Engineering **3** (1998), 187–196.

[AZ00] ———, *The meshless local Petrov-Galerkin (MLPG) approach for solving problems in elasto-statics*, Computational Mechanics **25** (2000), 169–179.

[Bax02] B.J.C. Baxter, *Preconditioned conjugate gradients, radial basis functions, and Toeplitz matrices*, Comput. Math. Appl. **43** (2002), 305–318.

[BC00] R.K. Beatson and E. Chacko, *Fast evaluation of radial basis functions: A multivariate momentary evaluation scheme*, Curve and Surface Fitting: Saint-Malo 1999 (A. Cohen, C. Rabut, and L.L. Schumaker, eds.), Vanderbilt University Press, 2000, pp. 37–46.

[BCM99] 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** (1999), 253–270.

[BCR00] R.K. Beatson, J.B. Cherrie, and D.L. Ragozin, *Polyharmonic splines in $I\!R^d$: Tools for fast evaluation*, Curve and Surface Fitting: Saint-Malo 1999 (A. Cohen, C. Rabut, and L.L. Schumaker, eds.), Vanderbilt University Press, 2000, pp. 47–56.

[BG97] R.K. Beatson and L. Greengard, *A short course on fast multipole methods*, Wavelets, Multilevel Methods and Elliptic PDEs (M. Ainsworth, J. Levesley, W. Light, and M. Marletta, eds.), Oxford University Press, 1997, pp. 1–37.

[BGP96] R.K. Beatson, G. Goodsell, and M.J.D. Powell, *On multigrid techniques for thin plate spline interpolation in two dimensions*, The mathematics of numerical analysis, Lectures in Appl. Math., vol. 32, Amer. Math. Soc., Providence, RI, 1996, pp. 77–97.

[BK05] M. Botsch and L. Kobbelt, *Real-time shape editing using radial basis functions*, Computer Graphics Forum, vol. 24, 2005, pp. 611–621.

[BKO$^+$96] T. Belytschko, Y. Krongauz, D.J. Organ, M. Fleming, and P. Krysl, *Meshless methods: an overview and recent developments*, Computer Methods in Applied Mechanics and Engineering, special issue **139** (1996), 3–47.

[BL97] R.K. Beatson and W.A. Light, *Fast evaluation of radial basis functions: Methods for two–dimensional polyharmonic splines*, IMA Journal of Numerical Analysis **17** (1997), 343–372.

[BLB00] 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** (2000), no. 5, 1717–1740.

[BLKL05] 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 **19** (2005), 343–353.

[BM97] I. Babuska and J.M. Melenk, *The Partition of Unity Method*, Int. J. Numer. Meths. Eng. **40** (1997), 727–758.

[BN92] R.K. Beatson and G.N. Newsam, *Fast evaluation of radial basis functions. I*, Comput. Math. Appl. **24** (1992), no. 12, 7–19, Advances in the theory and applications of radial basis functions.

[Bog85] A. Bogomolny, *Fundamental solutions method for elliptic boundary value problems*, SIAM J. Numer. Anal. **22** (1985), 644–669.

[Bra01] D. Braess, *Finite elements. Theory, fast solvers and applications in solid mechanics*, Cambridge University Press, 2001.

[Bri86] N.F. Britton, *Reaction-diffusion equations and their applications to Biology*, Academic Press, 1986.

[BS87] C.V. Bellos and J.G. Sakkas, *1-D dam-break flood-wave propagation on dry bed*, Journal of Hydraulic Engineering **113** (1987), 1510–1524.

[BS02] S.C. Brenner and L.R. Scott, *The mathematical theory of finite element methods, second edition*, Springer, 2002.

[BSR02] K. Balakrishnan, R. Sureshkumar, and P.A. Ramachandran, *An operator splitting-RBF method for the solution of transient nonlinear Poisson problems*, Computers and Mathematics with Applications **43** (2002), 289–304.

[Buh88] M.D. Buhmann, *Convergence of univariate quasi-interpolation using multiquadrics*, IMA J. Numer. Anal. **8** (1988), 365–383.

[Buh90] ———, *Multivariate cardinal interpolation with radial–basis functions*, Constr. Approx. **6** (1990), 225–255.

[Buh98] ———, *Radial functions on compact support*, Proceedings of the Edinburgh Mathematical Society **41** (1998), 33–46.

[Buh03] ———, *Radial basis functions, theory and implementations*, Cambridge University Press, 2003.

[Bur40] J.M. Burgers, *Application of a model system to illustrate some points of the statistical theory of free turbulence*, Proc. Acad. Sci. Amsterdam **43** (1940), 2–12.

[CAB92] A.H.-D. Cheng, Y. Abousleiman, and T. Badmus, *A Laplace transform BEM for axisymmetric diffusion utilizing pre-tabulated Greeen's function*, Eng. Anal. Boundary Elements **9** (1992), 39–46.

[CAO94] A.H.-D. Cheng, H. Ahtes, and N. Ortner, *Fundamental solutions of product of Helmholtz and polyharmonic operators*, Eng. Anal. Boundary Elements **14** (1994), 187–191.

[CBC$^+$01] 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*, Computer Graphics Proceedings, Addison Wesley, 2001, pp. 67–76.

[CBN02] J.B. Cherrie, R.K. Beatson, and G.N. Newsam, *Fast evaluation of radial basis functions: Methods for generalised multiquadrics in $I\!R^n$*, SIAM J. Sci. Comput. **23** (2002), 1272–1310.

[CBP99] C.S. Chen, C.A. Brebbia, and H. Power, *Dual reciprocity method using compactly supported radial basis functions*, Comm. Num. Meth. Eng. **15** (1999), 137–150.

[CC05a] A.H.-D. Cheng and J.J.S.P. Cabral, *Direct solution of certain ill-posed boundary value problems by collocation method*, Boundary Elements XXVII (A. Kassab, C.A. Brebbia, E. Divo, and D. Poljak, eds.), 2005, pp. 35–44.

[CC05b] ———, *Direct solution of ill-posed boundary value problems by radial basis function collocation method*, Internat. J. Numer. Methods Engrg. **64** (2005), no. 1, 45–64.

[CCG06] H.A. Cho, C.S. Chen, and M.A. Golberg, *Some comments on mitigating the ill-conditioning of the method of fundamental solutions*, Engineering Analysis with Boundary Elements **30** (2006), 405–410.

[CFB97] J.C. Carr, W.R. Fright, and R.K. Beatson, *Surface interpolation with radial basis functions for medical imaging*, IEEE Transactions on Medical Imaging **16** (1997), 96–107.

[CGGC02] C.S. Chen, M.A. Golberg, M. Ganesh, and A.H.-D. Cheng, *Multilevel compact radial functions based computational schemes for some elliptic problems*, Computers and Mathematics with Application **43** (2002), 359–378.

[CGML04] H.A. Cho, M.A. Golberg, A.S. Muleshkov, and X. Li, *Trefftz methods for time dependent partial differential equations*, Computers, Materials, and Continua **1** (2004), 1–38.

[CGR98] C.S. Chen, M.A. Golberg, and Y.F. Rashed, *A mesh free method for linear diffusion equations*, Numerical Heat Transfer, Part B **33** (1998), 469–486.

[CGS03] C.S. Chen, M.A. Golberg, and R. Schaback, *Recent developments of* *the dual reciprocity method using compactly supported radial basis functions*, Transformation of Domain Effects to the Boundary (Y.F. Rashed, ed.), WIT Press, 2003, pp. 183–225.

[CH03] W. Chen and Y.C. Hon, *Numerical convergence of boundary knot method in the analysis of Helmholtz, modified Helmholtz, and convection-diffusion problems*, Computer Methods Appl. Mech. Engng. **192** (2003), 1859–1875.

[Cha67] A.J. Chapman, *Heat transfer*, Macmillan, New York, 1967.

[Che87] R.S.C. Cheng, *Delta-trigonometric and spline methods using the single-layer potential representation*, Ph.D. thesis, University of Maryland, 1987.

[Che95] C.S. Chen, *The method of fundamental solutions for non-linear thermal explosions*, Comm. Numer. Methods Engrg. **11** (1995), no. 8, 675–681.

[Che02] W. Chen, *Symmetric boundary knot method*, Engineering Analysis with Boundary Elements **26** (2002), 489–494.

[CJ59] H.S. Carslaw and J.C. Jaeger, *Conduction of heat in solids*, Oxford University Press, London, 1959.

[CLG94] A.H.-D. Cheng, O. Lafe, and S. Grilli, *Dual reciprocity BEM based on global interpolation functions*, Eng. Anal. Boundary Elements **13** (1994), 303–311.

[CLH07] C.S. Chen, Sungwook Lee, and C.-S. Huang, *The method of particular solutions using chebyshev polynomial based functions*, International Journal of Computational Methods **4(1)** (2007), 15–32.

[CM78] I. Christie and A.R. Mitchell, *Upwinding of high order Galerkin methods in conduction-convection problems*, Int. J. Num. Meth. Eng. **12** (1978), 1764–1771.

[CMC99] C.S. Chen, M.D. Marcozzi, and S. Choi, *The method of fundamental solutions and compactly supported radial basis functions: a meshless approach to 3D problems*, Boundary elements, XXI (Oxford, 1999), Int. Ser. Adv. Bound. Elem., vol. 6, WIT Press, Southampton, 1999, pp. 561–570.

[CMG99] C.S. Chen, A.S. Muleshkov, and M.A. Golberg, *The numerical evaluation of particular solution for Poisson's equation - a revisit*, Boundary Elements XXI (C.A. Brebbia and H. Power, eds.), WIT Press, 1999, pp. 313–322.

[Col51] J.D. Cole, *On a quasi-linear parabolic equation occuring in aerodynamics*, Q. Appl. Math **9** (1951), 225–236.

[CR98] C.S. Chen and Y.F. Rashed, *Evaluation of thin plate spline based particular solutions for Helmholtz-type operators for the DRM*, Mech. Res. Comm. **25** (1998), 195–201.

[CS82] J Caldwell and P. Smith, *Solution of Burgers' equation with a large Reynolds number*, Appl. Math. Modelling **6** (1982), 381–385.

[CST00] N. Cristianini and J. Shawe-Taylor, *An introduction to support vector machines and other kernel-based learning methods*, Cambridge University Press, Cambridge, 2000.

[CT00] W. Chen and M. Tanaka, *New insights into boundary-only and domain-type RBF methods*, Int. J. Nonlinear Sci. & Numer. Simulation **1** (2000), 145–151.

[CYT00a] A.H.-D. Cheng, D.-L. Young, and J.-J. Tsai, *Solution of Poisson's equation by iterative DRBEM using compactly supported, positive definite radial basis function*, Eng. Analy. Boundary Elements **24** (2000), 549–557.

[CYT00b] ———, *Solution of Poisson's equation by iterative DRBEM using compactly supported, positive definite radial basis function*, Eng. Analy.

Boundary Elements **24** (2000), 549–557.

[Dav79] P.J. Davis, *Circulant matrices*, John Wiley & Sons, New York, 1979.

[DDFI05] L. Demaret, N. Dyn, M.S. Floater, and A. Iske, *Adaptive thinning for terrain modelling and image compression*, Advances in multiresolution for geometric modelling, Math. Vis., Springer, Berlin, 2005, pp. 319–338.

[DF02] T.A. Driscoll and B. Fornberg, *Interpolation in the limit of increasingly flat radial basis functions*, Comput. Math. Appl. **43** (2002), 413–422.

[DG76] W.R. Derrick and S.I. Grossman, *Elementary differential equations with applications*, Addison-Wessley Publishing Co., 1976.

[DL09] A.L. Drombosky, T.W. amd Meyer and L. Ling, *Applicability of the method of fundamental solutions*, Engineering Analysis with Boundary Elements **33** (2009), 637–643.

[DM76] G. De Mey, *Calculation of eigenvalues of the Helmholtz equation by an integral equation*, International Journal for Numerical Methods in Engineering **10** (1976), 59–66.

[Fas98] G.E. Fasshauer, *On smoothing for multilevel approximation with radial basis functions*, Approximation Theory IX, Vol. 2: Computational Aspects (Nashville) (C. K. Chui and L.L. Schumaker, eds.), Vanderbilt University Press, 1998, pp. 55–62.

[Fas99] ———, *Solving differential equations with radial basis functions: Multilevel methods and smoothing*, Adv. Comput. Math. **11** (1999), 139–159.

[Fas03] ———, *Approximate moving least-squares approximation with compactly supported radial weights*, Meshfree methods for partial differential equations (Bonn, 2001), Lect. Notes Comput. Sci. Eng., vol. 26, Springer, Berlin, 2003, pp. 105–116.

[Fas04] ———, *RBF collocation methods and pseudospectral methods*, Preprint, Illinois Institute of Technology, Chicago, 2004.

[Fas07] ———, *Meshfree approximation methods with MATLAB*, Interdisciplinary Mathematical Sciences, vol. 6, World Scientific, 2007.

[Fen91] R.T. Fenner, *Source field superposition analysis of two-dimensional potential problems*, International Journal for Numerical Methods in Engineering **32** (1991), 1079–1091.

[FF07] G.E. Fasshauer and A.J.M. Ferreira, *Analysis of natural frequencies of composite plates by an RBF–pseudospectral method*, Composite Structures **79** (2007), 202–210.

[FGP05] 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** (2005), no. 1, 1–24.

[FI96] M.S. Floater and A. Iske, *Multistep scattered data interpolation using compactly supported radial basis functions*, J. Comput. Applied Math. **73** (1996), 65–78.

[FI98] ———, *Thinning algorithms for scattered data interpolation*, BIT **38:4** (1998), 705–720.

[FK98] G Fairweather and A. Karageorghis, *The method of fundamental solution for elliptic boundary value problems*, Advances in Computatonal Mathematics **9** (1998), 69–95.

[FKM03] G Fairweather, A. Karageorghis, and P.A. Martin, *The method of fundamental solution for scattering and radiation problems*, Engineering Analysis with Boundary Elements **27** (2003), 759–769.

[FP00] A.C. Faul and M.J.D. Powell, *Krylov subspace methods for radial basis function interpolation*, Numerical analysis 1999 (Dundee), Chapman

& Hall/CRC Res. Notes Math., vol. 420, Chapman & Hall/CRC, Boca Raton, FL, 2000, pp. 115–141.

[Fra82] R. Franke, *Scattered data interpolation: tests of some methods*, Mathematics of Computation **48** (1982), 181–200.

[FS98a] C. Franke and R. Schaback, *Convergence order estimates of meshless collocation methods using radial basis functions*, Adv. Comput. Math. **8** (1998), 381–399.

[FS98b] ______, *Solving partial differential equations by collocation using radial basis functions*, Appl. Math. Comput. **93** (1998), 73–82.

[FS04] M. Fenn and G. Steidl, *Fast NFFT based summation of radial functions*, Sampl. Theory Signal Image Process. **3** (2004), no. 1, 1–28.

[FW04] B. Fornberg and G. Wright, *Stable computation of multiquadric interpolants for all values of the shape parameter*, Comput. Math. Appl. **48** (2004), no. 5-6, 853–867.

[GC98] M.A. Golberg and C.S. Chen, *The method of fundamental solutions for potential, Helmholtz and diffusion problems*, Boundary Integral Methods: Numerical and Mathematical Aspects (M.A. Golberg, ed.), WIT Press, 1998, pp. 103–176.

[GC01] ______, *A mesh free method for solving nonlinear reaction-diffusion equations*, Journal of Computer Modeling in Engineering & Science **2** (2001), 87–95.

[GCG00] M.A. Golberg, C.S. Chen, and M. Ganesh, *Particular solutions of 3D Helmholtz-type equations using compactly supported radial basis functions*, Eng. Anal. with Boundary Elements **24** (2000), 539–547.

[GCK96] M.A. Golberg, C.S. Chen, and S. Karur, *Improved multiquadric approximation for partial differential equations*, Engineering Analysis with Boundary Elements. **18** (1996), no. 1, 9–17.

[GCR99] M.A. Golberg, C.S. Chen, and Y.F. Rashed, *The annihilator method for computing particular solutions to partial differential equations*, Eng. Analy. Boundary Elements **23** (1999), 275–279.

[GFJ00] E.C. Gartland G.E. Fasshauer and J.W. Jerome, *Algorithms defined by Nash iteration: some implementations via multilevel collocation and smoothing*, J. Comp. Appl. Math. **119** (2000), 161–183.

[GMCC03] M.A Golberg, A.S. Muleshkov, C.S. Chen, and A.H.-D. Cheng, *Polynomial particular solutions for certain kind of partial differential operators*, Numerical Methods for Partial Differential Equations **19** (2003), 112–133.

[GS00] M. Griebel and M.A. Schweitzer, *A Particle-Partition of Unity Method for the solution of Elliptic, Parabolic and Hyperbolic PDE*, SIAM J. Sci. Comp. **22** (2000), no. 3, 853–890.

[GS02a] ______, *A particle-partition of unity method - part II: Efficient cover construction and reliable integration*, SIAM J. Sci. Comput. **23** (2002), 1655–1682.

[GS02b] ______, *A particle-partition of unity method - part III: A multilevel solver*, SIAM J. Sci. Comput. **24** (2002), 377–409.

[Gup79] A.K. Gupta, *The boundary integral method for potential problems involving axisymmetric geometry and arbitrary boundary conditions*, Ph.D. thesis, M.S. thesis, Department of Engineering Mechanics, University of Kentucky, 1979.

[GvL96] G. Golub and C. van Loan, *Matrix computations*, The Johns Hopkins University Press, 1996, Third edition.

[GW06a] P. Giesl and H. Wendland, *Approximating the basin of attraction of time-periodic ODEs by meshless collocation*, Preprint Göttingen/München, 2006.

[GW06b] ———, *Meshless collocation: Error estimates with application to dynamical systems*, Preprint Göttingen/München, 2006.

[Han92] P.C. Hansen, *Analysis of discrete ill-posed problems by means of the l-curve*, SIAM Review **34** (1992), 561–580.

[Han94] ———, *Regularization tools: a MATLAB package for analysis and solution of discrete ill-posed problems*, Numerical Algorithms **6** (1994), 1–35.

[Han00] ———, *The L-curve and its use in the numerical treatment of inverse problems*, Computational Inverse Problems in Electrocardiology, Advances in Computational Bioengineering Series (P. Johnston, ed.), WIT Press, Southampton, 2000.

[Har71] R.L. Hardy, *Multiquadric equations of Topography and other irregular surfaces*, Journal of Geophysical Research **176** (1971), 1905–1915.

[HC03] Y.C. Hon and W. Chen, *Boundary knot method for 2D and 3D Helmholtz and convection-diffusion problems with complicated geometry*, Int. J. Numer. Methd. Engng **56** (2003), 1931–1948.

[HM97] Y.C. Hon and X.Z. Mao, *A multiquadric interpolation method for solving initial value problems*, Sci. Comput. **12** (1997), no. 1, 51–55.

[HM98] ———, *An efficient numerical scheme for Burgers' equation*, Appl. Math. Comput. **95** (1998), no. 1, 37–50.

[HM99] ———, *A radial basis function method for solving option pricing model*, Int. J. Financial Engineering **8** (1999), no. 1, 31–49.

[HO93] P.C. Hansen and D.P. O'Leary, *The use of the l-curve in the regularization of discrete ill-posed problems*, SIAM Journal of Scientific Computing **14** (1993), 1487–1503.

[Hon93] Y.C. Hon, *Typhoon surge in Tolo Harbour of Hong Kong - an approach using finite element method with quadrilateral elements and parallel processing technique*, Chinese J. Num. Math. Appl. **15** (1993), no. 4, 21–33.

[Hon02] ———, *A quasi-radial basis function method for American options pricing*, Comput. Math. Applic. **43** (2002), no. 3-5, 513–524.

[HS01] Y.C. Hon and R. Schaback, *On unsymmetric collocation by radial basis functions*, J. Appl. Math. Comp. **119** (2001), 177–186.

[HS05] ———, *Solvability of partial differential equations by meshless kernel methods*, to appear in Adv. in Comp. Math., 2005.

[HSZ03] Y. C. Hon, R. Schaback, and X. Zhou, *An adaptive greedy algorithm for solving large RBF collocation problems*, Numer. Algorithms **32** (2003), no. 1, 13–25. MR 2003k:65157

[HW00] Y.C. Hon and Z.M. Wu, *Additive Schwarz domain decomposition with a radial basis approximation*, Int. J. Appl. Math. **4** (2000), 81–98.

[HW03] Y.C. Hon and T. Wei, *A meshless scheme for solving inverse problems of Laplace equation*, Recent development in theories & numerics, World Sci. Publishing, River Edge, NJ, 2003, pp. 291–300.

[HW05] ———, *The method of fundamental solution for solving multidimensional inverse heat conduction problems*, Comput. Model. Eng. Sci. **7** (2005), no. 2, 119–132.

[ICT04] M.S. Ingber, C.S. Chen, and J.A. Tanski, *A mesh free approach using radial basis functions and parallel domain decomposition for solving three-dimensional diffusion equations*, Internat. J. Numer. Methods Engrg. **60**

(2004), 2183–2201.

[IHM81] D.B. Ingham, P.J. Heggs, and M Manzoor, *Boundary integral equation solution of non-linear plane potential problems*, IMA J. Numer. Anal. **1** (1981), 415–426.

[IK04] A. Iske and M. Käser, *Conservative semi-Lagrangian advection on adaptive unstructured meshes*, Numerical Methods for Partial Differential Equations **20** (2004), 388–411.

[IM92] L. Iskandar and A. Mohsen, *Some numerical experiments on the splitting of Burgers' equation*, Numer. Methods for Partial Diff. Equations **8** (1992), 267–276.

[IMS01] IMSL, *IMSL Software*, Visual Numerics Inc. Houston, Texas, USA, 2001.

[IPT92] M.S. Ingber and N. Phan-Thien, *A boundary element approach for parabolic differential equations using a class of particular solutions*, Appl. Math. Modelling **16** (1992), 124–132.

[Isk95] A. Iske, *Reconstruction of functions from generalized Hermite-Birkhoff data*, Approximation Theory VIII, Vol. 1 (C.K. Chui and L.L. Schumaker, eds.), World Scientific, Singapore, 1995, pp. 257–264.

[Isk04] ———, *Multiresolution methods in scattered data modelling*, Lecture Notes in Computational Science and Engineering, vol. 37, Springer-Verlag, Berlin, 2004.

[JFK87] R.L. Johnston, G. Fairweather, and A. Karageorghis, *An adaptive indirect boundary element method with applications*, Boundary Elements VIII (M. Tanaka and C.A. Brebbia, eds.), Springer-Verlag, Tokyo, 1987, pp. 587–597.

[Jin04] B. Jin, *A meshless method for the Laplace and biharmonic equations subjected to noisy boundary data*, Computer Modeling in Engineering & Sciences **6(3)** (2004), 253–261.

[Jos02] J. Jost, *Partial differential equations*, Springer, 2002.

[JZ05] B. Jin and Y. Zheng, *Boundary knot method for some inverse problems associated with the Helmholtz equation*, Internat. J. Numer. Methods Engrg. **62** (2005), no. 12, 1636–1651.

[KA64a] V.D. Kupradze and M.A. Aleksidze, *A method for the approximate solution of limiting problems in mathematical physics*, U.S.S.R. Computational Mathematics and Mathematical Physics **4** (1964), 199–205.

[KA64b] ———, *The method of functional equations for the approximate solution of certain boundary value problems*, U.S.S.R. Computational Mathematics and Mathematical Physics **4** (1964), 82–126.

[KA65] ———, *Potential methods in the theory of elasticity*, Israel Program for Scientific Translations, Jerusalem, 1965.

[Kan86] E.J. Kansa, *Application of Hardy's multiquadric interpolation to hydrodynamics*, Proc. 1986 Simul. Conf., Vol. 4, 1986, pp. 111–117.

[Kan90a] ———, *Multiquadrics - a scattered data approximation scheme with applications to computational fluid dynamics - I*, Comput. Math. Applic. **19** (1990), no. 8/9, 127–145.

[Kan90b] ———, *Multiquadrics - a scattered data approximation scheme with applications to computational fluid dynamics - II*, Comput. Math. Applic. **19** (1990), no. 8/9, 147–161.

[Kar92] A. Karageorghis, *Modified methods of fundamental solutions for harmonic and biharmonic problems with boundary singularities*, Numerical Methods for Partial Differential Equations **8** (1992), 1–19.

[Kar01] ———, *The method of fundamental solutions for calculation of the eigenvalues of the Helmholtz equation*, Applied Mathematics Letters **14** (2001), 837–842.

[Kat90] M. Katsurada, *Asymptotic error analysis of the charge simulation method in a jordan region with an analytic boundary*, Journal of the Faculty of Science of Tokyo University, Section 1A **37** (1990), 635–657.

[Kat94] ———, *Charge simulation method using exterior mapping functions*, Japan Journal of Industrial and Applied Mathematics **11** (1994), 47–61.

[KCS07] A. Karageorghis, C.S. Chen, and Y-S Smyrlis, *A matrix decomposition method for solving large scale problems using radial basis functions - approximation of functions and their derivatives*, Applied Numerical Mathematics **57** (2007), 304–319.

[KF87] A. Karageorghis and G. Fairweather, *The method of fundamental solutions for the numerical solution of the biharmonic equation*, Journal of Computational Physics **69** (1987), 435–459.

[KF88] ———, *The Almansi method of fundamental solutions for solving biharmonic problems*, International Journal for Numerical Methods in Engineering **26** (1988), 1668–1682.

[KF89a] ———, *The method of fundamental solutions for the solution of nonlinear plane potential problems*, IMA Journal of Numerical Analysis **9** (1989), 231–242.

[KF89b] ———, *The simple layer potential method of fundamental solutions for certain biharmonic problems*, International Journal for Numerical Methods in Fluids **9** (1989), 1221–1234.

[KF98] ———, *The method of fundamental solutions for axisymmetric accoustic scattering and radiation problems*, The Journal of the Acoustical Society of America **104** (1998), 3212–3218.

[KF99] ———, *The method of fundamental solutions for axisymmetric potential problems*, International Journal for Numerical Methods in Engineering **44** (1999), 1653–1669.

[KH00] 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** (2000), no. 7-8, 123–137.

[Kit88] T. Kitagawa, *On the numerical stability of the method of fundamental solutions applied to the Dirichlet problem*, Japan Journal of Applied Mathematics **35** (1988), 507–518.

[Kit91] ———, *Asymptotic stability of the fundamental solution method*, Journal of Computational and Applied Mathematics **38** (1991), 263–269.

[KK72] D.Q. Kern and A.D. Kraus, *Extended surface heat transfer*, McGraw-Hill, New York, 1972.

[KO88] M. Katsurada and H. Okamoto, *A mathematical study of the charge simulation method*, Journal of the Faculty of Science, University of Tokyo, Section 1A **35** (1988), 507–518.

[KO96] ———, *The collocation points of the fundamental solution method for the potential problem*, Computers and Mathematics with Applications **31** (1996), 123–137.

[Kri51] D.G. Krige, *A statistical approach to some basic mine valuation problems on the witwatersrand*, J. of the Chem., Metal. and Mining Soc. of South Africa **52** (1951), 119–139.

[KSBH02] N. Kojekine, V. Savchenko, D. Berzin, and I. Hagiwara, *Computer graphics applications based on compactly supported radial basis functions* *toolkit*, Sūrikaisekikenkyūsho Kōkyūroku **1288** (2002), 64–72, Nonlinear problems in industry, domain decomposition methods and numerical simulations (Japanese) (Kyoto, 2002).

[Kup67] V.D. Kupradze, *On the approximate solution of problems in mathematical physics*, Russian Math. Surveys **22** (1967), 58–108.

[Lan99] S. Langdon, *Domain embedding boundary integral equation methods and parabolic pdes*, Ph.D. thesis, University of Bath, 1999.

[Lev98] D. Levin, *The approximation power of moving least-squares*, Math. Comput. **67** (1998), 1517–1531.

[LF03] E. Larsson and B. Fornberg, *A numerical study of some radial basis function based solution methods for elliptic PDEs*, Comput. Math. Appl. **46** (2003), 891–902.

[LF05] ______, *Theoretical and computational aspects of multivariate interpolation with increasingly flat radial basis functions*, Comput. Math. Appl. **49** (2005), no. 1, 103–130.

[LH04] J. Li and Y.C. Hon, *Domain decomposition for radial basis meshless methods*, Numer. Methods for PDEs **20** (2004), 450–462.

[LHC02] J. Li, Y.C. Hon, and C.S. Chen, *Numerical comparisons of two meshless methods using radial basis functions*, Eng. Anal. Boundary Elements **26** (2002), 205–225.

[Li04] J. Li, *A radial basis meshless method for solving inverse boundary value problems*, Comm. Numer. Methods Engrg. **20** (2004), no. 1, 51–61.

[Li05] ______, *Application of radial basis meshless methods to direct and inverse biharmonic boundary value problems*, Comm. Numer. Methods Engrg. **21** (2005), no. 4, 169–182.

[LK04] L. Ling and E.J. Kansa, *Preconditioning for radial basis functions with domain decomposition methods*, Math. Comput. Modelling **40** (2004), 1413–1427.

[LK05] ______, *A least-squares preconditioner for radial basis functions collocation methods*, Adv. Comput. Math. **23** (2005), no. 1-2, 31–54.

[LS81] P. Lancaster and K. Salkauskas, *Surfaces generated by moving least squares methods*, Math. Comput. **37** (1981), 141–158.

[LYY05] Y.J. Lee, G.J. Yoon, and J. Yoon, *Convergence property of increasingly flat radial basis function interpolation to polynomial interpolation*, preprint, 2005.

[Mac85] M. MacDonell, *A boundary method applied to the modified Helmholtz equation in three dimensions and its application to a waste disposal problem in the deep ocean*, Ph.D. thesis, M.S. Thesis, Department of Computer Science, University of Toronto, 1985.

[Mai56] J.C. Mairhuber, *On Haar's theorem concerning Chebychev approximation problems having unique solutions*, Proc. Am. Math. Soc. **7** (1956), 609–615.

[Mat62] G. Matheron, *Traité de géostatistique appliquée*, Editions Technip, 1962.

[MB96] J.M. Melenk and I. Babuska, *The partition of unity finite element method: Basic theory and applications*, Comput. Meths. Appl. Mech. Engrg. **139** (1996), 289–314.

[MCGC00] A.S. Muleshkov, C.S. Chen, M.A. Golberg, and A.H-D. Cheng, *Analytic particular solutions for inhomogeneous Helmholtz-type equations*, Advances in Computational Engineering & Sciences (S.N. Atluri and F.W. Brust, eds.), Tech Science Press, 2000, pp. 27–32.

[ME94] G.J. Moridis and Kansa E.J., *The Laplace transform multiquadric method: A highly accurate scheme for the numerical solution of linear partial differential equations*, J. Appl. Sci. Comp. **1** (1994), 375–407.

[Mey00] C.D. Meyer, *Matrix analysis and applied linear algebra*, SIAM, 2000.

[MG80] A.R. Mitchell and D.F. Griffiths, *The finite difference method in partial differential equations*, John Wiley & Sons Ltd, 1980, pp. 233.

[MGC99] A.S. Muleshkov, M.A. Golberg, and C.S. Chen, *Particular solutions of Helmholtz-type operators using higher order polyharmonic splines*, Comp. Mech. **23** (1999), 411–419.

[MJ77] R. Mathon and R.L. Johnston, *The approximate solution of elliptic boundary-value problems by fundamental solutions*, SIAM J. Numer. Anal. **14** (1977), 638–650.

[MN88] W.R. Madych and S.A. Nelson, *Multivariate interpolation and conditionally positive definite functions*, Approximation Theory Appl. **4** (1988), 77–89.

[MN92] ______, *Bounds on multivariate polynomials and exponential error estimates for multiquadric interpolation*, J. Approx. Theory **70** (1992), 94–114.

[MNRW91] G. Meinardus, G. Nürnberger, Th. Riessinger, and G. Walz, *In memoriam: the work of Lothar Collatz in approximation theory*, J. Approx. Theory **67** (1991), no. 2, 119–128.

[Mou01] C.T. Mouat, *Fast algorithms and preconditioning techniques for fitting radial basis functions*, Ph.D. thesis, Mathematics Department, University of Canterbury, Christchurch, New Zealand, 2001.

[MR91] G.L. Moridis and D.L. Reddell, *The Laplace transform boundary element (LTBE) method for the solution of diffusion-type equations*, Boundary Elements XIII (C.A. Brebbia, ed.), Springer-Verlag, Berlin, 1991, pp. 83–97.

[MS96] V. Maz'ya and G. Schmidt, *On approximate approximations using Gaussian kernels*, IMA Journal of Numerical Analysis **16** (1996), 13–29.

[MYP$^+$01] B. Morse, T.S. Yoo, P.Rheingans, D.T. Chen, and K.R. Subramanian, *Interpolating implicit surfaces from scattered surface data using compactly supported radial basis functions*, International Conference on Shape Modeling and Applications (SMI 2001), Computer Society Press, 2001, pp. 89–98.

[NAG01] NAG, *Numerical algorithms group library mark 20*, NAG Ltd, Wilkinson House, Jordan Hill Road, Oxford, UK, 2001.

[Nas06] Z. Nashed, *Applications of wavelets and kernel methods in inverse problems*, Integral methods in science and engineering, Birkhäuser Boston, Boston, MA, 2006, pp. 189–197.

[NFN00] J.Y. Noh, D. Fidaleo, and U. Neumann, *Animated deformations with radial basis functions*, ACM Virtual Reality and Software Technology (VRST), 2000, pp. 166–174.

[NSW99] F.J. Narcowich, R. Schaback, and J.D. Ward, *Multilevel interpolation and approximation*, Appl. Comput. Harmon. Anal. **7** (1999), 243–261.

[NWW06] F.J. Narcowich, J.D. Ward, and H. Wendland, *Sobolev error estimates and a Bernstein inequality for scattered data interpolation via radial basis functions*, Constr. Approx. **24** (2006), 175–186.

[OBS03] Y. Ohtake, A. Belyaev, and H.-P. Seidel, *A multi-scale approach to 3D scattered data interpolation with compactly supported basis functions*, SMI '03: Proceedings of the Shape Modeling International 2003 (Washington, DC, USA), IEEE Computer Society, 2003, p. 292.

[Oli68] E.R. Oliveira, *Plane stress analysis by a general integral method*, J. Eng. Mech. Div., ASCE **94** (1968), 79–101.

[O'N99] P.V. O'Neil, *Beginning partial differential equations*, John Wiley & Son, New York, 1999.

[OP86] E.L. Ortiz and K.S. Pun, *A bidimensional tau-elements method for the numerical solution of nonlinear partial differential equations with an application to Burgers' equation*, Comput. Math. Appl. **12B** (1986), 1225–1240.

[PBW92] P.W. Partridge, C.A. Brebbia, and L.C. Wrobel, *The dual reciprocity boundary element method*, CMP/Elsevier, 1992.

[PKG98] A. Poullikkas, A. Karageorghis, and G. Georgiou, *The method of fundamental solutions for inhomogeneous elliptic problems*, Computational Mechanics **22** (1998), 100–107.

[PR55] D. Peaceman and H. Rachford, *The numerical solution of parabolic and elliptic differential equations*, J. SIAM **3** (1955), 28–41.

[PS03] D. Potts and G. Steidl, *Fast summation at nonequispaced knots by NFFTs*, SIAM J. Sci. Comput. **24** (2003), no. 6, 2013–2037 (electronic).

[PTVF96] W.H. Press, S.A. Teukolsky, W.T. Vetterling, and B.P. Flannery, *Numerical Recipes in Fortran: the Art of Scientifc Computing*, Cambridge University Press, 1996.

[PW67] M.H. Protter and H.F. Weinberger, *Maximum principles in differential equations*, Englewood Cliffs, 1967.

[Ram02] P.A. Ramachandran, *Method of fundamental solutions: singular value of decomposition analysis*, Commum. Numer. Meth. Engng **18** (2002), 789–801.

[RB05] G. Roussos and Brad J.C. Baxter, *Rapid evaluation of radial basis functions*, J. Comput. Appl. Math. **180** (2005), no. 1, 51–70.

[Reu05] S.Y. Reutskiy, *The method of fundamental solutions for eigenproblems with Laplace and biharmonic operators*, Computers, Materials, and Continua **2** (2005), 177–188.

[Reu06a] ———, *The method of external sources (MES) for eigenvalue problems with Helmholtz equation*, CMES **12** (2006), 27–40.

[Reu06b] ———, *The method of fundamental solutions for Helmholtz eigenvalue problems in simply and multiply connected domains*, Engineering Analysis with Boundary Elements **30** (2006), 150–159.

[RP75] J.B. Rosser and N. Papamichael, *A power series solution of a harmonic mixed boundary value problem*, MRC Technical Summary, Rept. 1405, University of Wisconsin, 1975.

[RTSD03] P. Reuter, I. Tobor, C. Schlick, and S. Dedieu, *Point-based modelling and rendering using radial basis functions*, Proceedings of ACM Graphite 2003, 2003, pp. 111–118.

[RZ06] C. Rieger and B. Zwicknagl, *Sampling inequalities for infinitely smooth functions, with applications to interpolation and machine learning*, Preprint Göttingen/Leipzig, 2006.

[Sai05] S. Saitoh, *Applications of reproducing kernels to best approximations, Tikhonov regularizations and inverse problems*, Advances in analysis, World Sci. Publ., Hackensack, NJ, 2005, pp. 439–445.

[Sch94] N.A. Schclar, *Anisotropic analysis using boundary elements*, Computational Mechanics Publication, 1994.

[Sch95] R. Schaback, *Error estimates and condition numbers for radial basis* *function interpolation*, Adv. Comput. Math. **3** (1995), 251–264.

[Sch97] ———, *On the efficiency of interpolation by radial basis functions*, Surface Fitting and Multiresolution Methods (A. LeMéhauté, C. Rabut, and L.L. Schumaker, eds.), Vanderbilt University Press, Nashville, TN, 1997, pp. 309–318.

[Sch03] M.A. Schweitzer, *A Parallel Multilevel Partition of Unity Method for Elliptic Partial Differential Equations*, Lecture Notes in Computational Science and Engineering, vol. 29, Springer, 2003.

[Sch05] R. Schaback, *Convergence analysis of methods for solving general equations*, Boundary Elements XXVII (A. Kassab, C.A. Brebbia, E. Divo, and D. Poljak, eds.), WITPress, Southampton, 2005, pp. 17–24.

[Sch06a] ———, *Limit problems for interpolation by analytic radial basis functions*, Preprint, to appear in J. of Computational and Applied Mathematics, 2006.

[Sch06b] ———, *Recovery of functions from weak data using unsymmetric meshless kernel-based methods*, to appear in APNUM, 2006.

[Sch06c] ———, *Unsymmetric meshless methods for operator equations*, Preprint, 2006.

[Sch07a] ———, *Adaptive numerical solution of MFS systems*, Preprint Göttingen, 2007.

[Sch07b] ———, *Convergence of unsymmetric kernel-based meshless collocation methods*, SIAM J. Numer. Anal. **45** (2007), 333–351.

[Sch07c] ———, *Solving the Laplace equation by meshless collocation using harmonic kernels*, Preprint Göttingen, 2007.

[SS02] B. Schölkopf and A.J. Smola, *Learning with Kernels*, MIT Press, Cambridge, 2002.

[STC04] J. Shawe-Taylor and N. Cristianini, *Kernel methods for pattern analysis*, Cambridge University Press, 2004.

[Ste70] H. Stehfest, *Algorithm 368: numerical inversion of Laplace transform*, Commun. ACM **13** (1970), 47–49.

[Sto92] J.J. Stoker, *Water waves*, John Wiley & Sons, Inc., 1992.

[SW99] R. Schaback and H. Wendland, *Using compactly supported radial basis functions to solve partial differential equations*, Boundary Element Technology XIII (C.S. Chen, C.A. Brebbia, and D.W. Pepper, eds.), WitPress, Southampton, Boston, 1999, pp. 311–324.

[SW00] ———, *Adaptive greedy techniques for approximate solution of large RBF systems*, Numerical Algorithms **24** (2000), 239–254.

[SW02] ———, *Approximation by positive definite kernels*, for the IDO-Mat2001 proceedings (invited lecture), 2002.

[SW06] ———, *Kernel techniques: From machine learning to meshless methods*, Acta Numerica **15** (2006), 543–639.

[Sym73] G.T. Symm, *Treatment of singularities in the solution of Laplace's equation by an integral equation method*, National Physical Laboratory Report NAC 31, 1973.

[Tre26] E. Trefftz, *Ein gegenstück zum ritzschen verfahren*, 2. Int. Kongr. f. Techn. Mechanik, Zürich, 1926, pp. 131–137.

[TRS04] I. Tobor, P. Reuter, and C. Schlick, *Efficient reconstruction of large scattered geometric datasets using the partition of unity and radial basis functions*, Journal of WSCG 2004 **12** (2004), no. 3, 467–474.

[TWN04] Y.J. Tan, X.H. Wang, and B.G. Niu, *Solving the inverse problem of determining the boundary of a parabolic equation by radial basis method*,

J. Fudan Univ. Nat. Sci. **43** (2004), no. 3, 285–291.

[WDH95] P. Wilmott, J. Dewynne, and S. Howison, *The mathematics of financial derivatives*, Cambridge University Press, 1995.

[Wen95] H. Wendland, *Piecewise polynomial, positive definite and compactly supported radial functions of minimal degree*, Adv. Comput. Math. **4** (1995), 389–396.

[Wen01] ______, *Local polynomial reproduction and moving least squares approximation*, IMA Journal of Numerical Analysis **21** (2001), 285–300.

[Wen02] ______, *Fast evaluation of radial basis functions: Methods based on partition of unity*, Approximation Theory X: Wavelets, Splines, and Applications (C.K. Chui, L.L. Schumaker, and J. Stöckler, eds.), Vanderbilt University Press, 2002, pp. 473–483.

[Wen05] ______, *Scattered data approximation*, Cambridge University Press, 2005.

[WH03] Z.M. Wu and Y.C. Hon, *Convergence error estimate in solving free boundary diffusion problem by radial basis functions method*, Engng. Anal. Bound. Elem. **27** (2003), no. 1, 73–79.

[WHG02a] S.M. Wong, Y.C. Hon, and M.A. Golberg, *Compactly supported radial basis functions for shallow water equations*, Appl. Math. Comput. **127** (2002), no. 1, 79–101.

[WHG02b] ______, *Compactly supported radial basis functions for the shallow water equations*, Appl. Math. Comput. **127** (2002), 79–101.

[WHL$^+$98] S.M. Wong, Y.C. Hon, T.S. Li, S.L. Chung, and E.J. Kansa, *Multizone decomposition for simulation of time-dependent problems using the multiquadric scheme*, to appear, 1998.

[WHL$^+$99] ______, *Multizone decomposition for simulation of time-dependent problems using the multiquadric scheme*, Comput. Math. Appl. **37** (1999), no. 8, 23–43.

[WHW05] T. Wei, Y.C. Hon, and Y.B. Wang, *Reconstruction of numerical derivatives from scattered noisy data*, Inverse Problems **21** (2005), 657–672.

[WK97] L. Wu and Y.K. Kwok, *A front-fixing finite difference method for the valuation of american options*, Comput. Math. Applic. **6** (1997), 83.

[WK05] J.H. Wu and L. Kobbelt, *Efficient spectral watermarking of large meshes with orthogonal basis functions*, The Visual Computers (Pacific Graphics 2005 Proceedings), vol. 21 (8-10), 2005, pp. 848–857.

[WR05] H. Wendland and C. Rieger, *Approximate interpolation with applications to selecting smoothing parameters*, Numer. Math. **101** (2005), 643–662.

[WS93] Z.M. Wu and R. Schaback, *Local error estimates for radial basis function interpolation of scattered data*, IMA Journal of Numerical Analysis **13** (1993), 13–27.

[Wu92] Z.M. Wu, *Hermite–Birkhoff interpolation of scattered data by radial basis functions*, Approximation Theory Appl. **8/2** (1992), 1–10.

[Wu95] ______, *Multivariate compactly supported positive definite radial functions*, Adv. Comput. Math. **4** (1995), 283–292.

[Yoo01] J. Yoon, *Spectral approximation orders of radial basis function interpolation on the Sobolev space*, SIAM J. Math. Anal. **33** (2001), no. 4, 946–958 (electronic).

[ZHL03a] X. Zhou, Y.C. Hon, and J. Li, *Overlapping domain decomposition method by radial basis functions*, Appl. Numer. Math. **44** (2003), 241–

255.

[ZHL03b] ______, *Overlapping domain decomposition method by radial basis functions*, Ann. Numer. Math. **44** (2003), no. 1-2, 243–257.

[Zhu93] S. Zhu, *Particular solutions associated with the Helmholtz operator used in DRBEM*, Boundary Element Abstracts **4** (1993), 231–233.

[ZSL94] S. Zhu, P. Satravaha, and X. Lu, *Solving linear diffusion equations with the Dual Reciprocity Method in Laplace space*, Eng. Anal. with Boundary Elements **13** (1994), 1–10.

Index

acoustic scattering problem, 128
Adams-Bashforth scheme, 240
Adams-Moulton scheme, 240
ADI, 239
American options, 61
analytic RBFs, 33, 98
annihilator method, 167
approximate approximation, 36
approximate particular solution, 146
approximate solution, 5
asymmetric collocation, 46, 47
axisymmetric potential problems, 122

bandwidth, 38, 188
Berger equation, 199
Bessel functions, 57, 86, 118, 167, 169, 176, 200, 202, 204, 205, 235
biharmonic equation, 110, 112, 163
biharmonic problem, 110, 119
Binomial method, 68
Black–Scholes equation, 61, 63
block–circulant, 213
block–wise preconditioner, 40
Boundary Element Method, 46, 78, 143–146
boundary integral equation, 80
Boundary Integral Method, 80, 146
Boundary Knot Method, 13
boundary–type methods, 78
Burgers equation, 10, 55, 236

cardinal functions, 40
Cartesian coordinates, 6
centers, 2
Charge Simulation Method, 79
circulant matrix, 208
Circular Points Method, 208
collocation, 9, 20, 46, 87
collocation points, 2, 26, 81, 87
Compact Difference Technique, 60
compactly supported RBF, 14, 38, 39, 69, 152, 158, 159, 161, 180, 186, 187, 194
Computational Mechanics, vi
Computer Graphics, vi
conditionally positive definite, 4, 17, 19, 36, 151
conservation laws, 21
convection problem, vi
convection-advection problem, vi
convection-diffusion equation, 234
Crank–Nicholson method, 229, 234
cross–validation, 41

dam–breaking problem, 69
derivative approximation, 18
Desingularized Method, 79
diagonal, 80
diffusion equation, 61, 223, 229, 232, 241
diffusion problem, vi, 61, 229
diffusion–reaction equation, 239
direct MPS, 147, 151, 152
Dirichlet boundary condition, 97, 100, 102
Dirichlet problem, 9, 80, 102, 179
Discrete Fourier Transform, 213
distributional data, 8
domain decomposition, 38, 40, 69
domain decomposition method, 71, 215
domain–type methods, 78
double layer potential, 80
DRM, 145
Dual Reciprocity Method, 145, 164
dynamic MFS, 109

Economics, vi
eigenfunction, 116
eigenvalue problem, 116
Elasticity, vi
error analysis, 21

Euclidean norm, 1
Euler method, 229, 234
European options, 61
evaluation points, 38, 82

Fast Fourier Transform, 215
fictitious boundary, 81, 86
fill distance, 21, 188
Finite Difference Method, 46, 78
Finite Element Method, 1, 6, 46, 60, 78
Finite Volume Method, 21, 46
Fisher equation, 235
flat limit, 33, 98
flow problem, 55, 69, 72, 128, 226, 240
Fluid Dynamics, vi
Fourier transform, 63, 151
free boundary value problem, 10, 61, 69
frequency, 75, 95
Front Finite Difference method, 68
fundamental solution, 80, 85, 145

Gaussian, 1, 3, 15, 21, 34
General Integral Method, 79
generalized interpolation, 19
generalized trapezoidal methods, 229

Hankel function, 86, 117, 200
heat conduction problem, 124, 131, 226
heat equation, 14, 132–134
Helmholtz eigenvalue problem, 116
Helmholtz equation, 95, 128, 164, 178, 195, 202, 204
Helmholtz equation, modified, 186, 205, 224, 235
Helmholtz operator, 164, 175, 176, 200
Helmholtz problem, 179, 184, 203
Hermite–Birkhoff interpolation, 19

ill-posed problem, vi, 42
Image Analysis, vi
incoming corners, 8, 87, 119, 160
Indirect Boundary Element Method, 79
indirect MPS, 147, 153, 180
interpolation, 2, 12, 17
inverse multiquadric, 3, 15, 152, 156, 157
inverse problem, vi, 42, 132

kernel, 2
kernel matrix, 2, 41
kernel methods, 42
Kriging, 46
Krylov subspace iteration, 40

L–curve method, 31, 135
Lagrange interpolation, 17
Laplace equation, 88, 97, 100, 102, 104, 130, 241
Laplace operator, 6, 47, 122, 151, 200
Laplace transform, 14, 47, 223
Laplace transform, inversion, 225
Laplace transforms, method of, 222
learning algorithms, vi
Levenberg–Marquardt algorithm, 41, 110, 111
line method, 45, 61, 228
localization, 5, 6, 38
Lyapounov basin, 20

machine learning, 2, 42
Matern RBF, 3, 15
matrix exponential, 63
maximum principle, 12, 79, 82, 85, 88
meshless methods, vi, 1, 2, 8, 18, 46, 78
Method of Fundamental Solutions, 12, 13, 79, 82, 146
Method of Laplace transforms, 222
Method of Lines, 222, 228
Method of Particular Solutions, 13, 146
MFS, 13
MFS$\rightarrow$MPS algorithm, 201
mixed boundary conditions, 91, 103, 104, 106
mixed interpolation, 19
mixed problem, 8
MLPG, 8
modified Helmholtz equation, 165, 177–179, 186, 199, 205, 230, 243
modified MFS, 115
Motz problem, 119
moving boundary value problem, 69
moving least squares, 6
MPS, 13
multilevel methods, 38, 188
multipole expansions, 39
multipole method, 6, 46
multiquadric, 1, 4, 15, 21, 34, 46
multiresolution methods, 188
multiscale MPS–MFS algorithm, 192
multiscale techniques, 39, 186, 188, 192
multivariate interpolation, 2

native Hilbert space, 42
Navier-Stokes equation, 55
Navier-Stokes problem, 240
neural networks, vi
non–smooth data, 8
nonlinear boundary conditions, 82, 129
nonlinear Burgers equation, 55
nonlinear diffusion equation, 235
nonlinear diffusion–reaction equation, 239
nonstationary, 35

operator splitting method, 239
optimization, 5, 27, 41
option pricing, vi, 10, 61
orbital derivative, 20
order of conditional positive
  definiteness, 4
ordinary differential equations, 228

partial differential equations, vi, 6
Particle Method, 46
particular solution, 144, 147
particular solution, approximate, 146
partition of unity, 6, 38, 39
PDE, vi, 2
Poisson equation, 241
Poisson problem, 6, 152, 159, 194, 202,
  220
polyharmonic spline, 4, 15, 33, 66, 151,
  171, 174–176, 198, 206, 230, 233,
  237, 243
polynomial unisolvency constraint, 20
positive definite, 3, 19, 151
potential equation, 122
potential problem, 12, 122
preconditioning, 38, 40, 69
pseudospectral methods, 18, 228

quasi–Monte Carlo method, 189
quasi–uniform, 23, 188

radial, 1
radial basis functions, vi, 1, 15
radiation problem, 128
RBF, 1
reaction-diffusion problem, vi
recovery, vi, 17
regularity, 7
regularization, 26–28, 41, 42, 102, 107,
  108, 134

scale factor, 33, 151, 188
scaling, 5, 33, 151, 188
Schwarz decomposition, 40
sensitivity to noise, 41
separation distance, 22, 41, 188, 189
series expansions for differential
  operators, 195
shallow water equation, vi, 10
shallow water model, 69
shape factor, 33
shape modeling, vi
shock wave, 10
single layer potential, 81
singular problem, 119
singular value decomposition, 5, 26, 27
singular values, 27
smoothness, 25
Sobolev RBF, 3
source points, 26, 80, 81
spectral condition number, 27
spectral convergence, 13
spectral methods, 7
stability, 5, 22, 99, 188
stabilization, 41
static approach, 86
static MFS, 86
stationary, 35
Stehfest algorithm, 223
Stokes flow problem, 240
Stress Analysis, vi
strong data, 17
strong form, 7
Superposition Method, 79
SVD, 27
symmetric collocation, 11, 48

test function, 7
test points, 2, 9, 26, 81, 87
theta method, 229
thin–plate spline, 4, 13, 33, 168, 173,
  198, 227
thinning algorithms, 23, 39, 189
three-dimensional problem, 96
Tikhonov regularization, 30, 111, 135,
  138
time–dependent problems, 14, 47, 133,
  150, 208, 222, 239
time–dependent RBF approximations,
  44
time–stepping method, 14, 228
Toeplitz matrix, 210
traffic flow, 55
translates, 2
transport problems, 55
Trefftz method, 13, 79
trial centers, 2, 26, 81
trial functions, 2, 9
trial points, 2
truly meshless, 7, 12
truncated singular value decomposition,
  28, 103, 135
TSVD, 28

Uncertainty Principle, 24, 100
uniformity constant, 23
unsymmetric collocation, 9

variable support, 187
velocity–vorticity formulation, 241
viscuous flow, 55

warping, 18
wave equation, 95, 224, 242–244
wave frequency, 75

wave number, 95
wave–runup problem, 69
weak data, 17
weak form, 7
Wendland RBF, 5, 15, 39, 152, 158,
159, 162, 181, 185, 186, 194