A Practical Guide to Radial Basis Functions

R. Schaback

April 16, 2007

Contents

Preface

This is "my" part of a future book "Scientific Computing with Radial Basis Functions" I am currently writig with my colleagues C.S. Chen and Y.C. Hon. I took a preliminary version out of the current workbench for the book and made only very few changes. Readers should be aware that this text just sets the stage for certain kinds of "meshless methods" for solving partial differential equations. Thus there are quite a few "hanging" references to future chapters for which I apologize.

R. Schaback Spring 2007

1 Radial Basis Functions

Scientific Computing with Radial Basis Functions focuses on the reconstruction 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 can only be done if the space furnishing the "trial" functions is not fixed in advance, but is data–dependent [99]. **Finite elements** (see *e.g.:* [18, 19]) 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 [16]. 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** [66]

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

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

$$\Phi(\mathbf{x}) = \phi(\|\mathbf{x}\|_2) = \phi(r),\ \mathbf{x} \in I\!R^d \tag{1.1}$$

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

Reconstruction of functions is then made by **trial functions** u which are linear combinations

$$u(\mathbf{x}) := \sum_{k=1}^{n} \alpha_k \phi(\|\mathbf{x} - \mathbf{y}_k\|_2) \tag{1.2}$$

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 $\mathbb{R}^d$, sometimes called **centers**, 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 (1.2), 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 [24, 135] 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 [124]. The rest of this chapter gives an overview over the applications we cover in this book.

1.1 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 $\mathbb{R}^d$ forming a set $X := \{\mathbf{x}_1, \ldots, \mathbf{x}_m\}$ are known. In contrast to the n **trial points** $\mathbf{y}_1, \ldots, \mathbf{y}_n$ of (1.2), 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 (1.2) which reproduces the data $u^*(\mathbf{x}_1), \ldots, u^*(\mathbf{x}_m)$ well, we have to solve the $m \times n$ linear system

$$\sum_{k=1}^{n} \alpha_k \phi(\|\mathbf{x}_i - \mathbf{y}_k\|_2) \approx u^*(\mathbf{x}_i),\ 1 \le i \le m \tag{1.3}$$

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 $m \geq n$ holds by picking at least as many test points as trial points, but the easiest case will occur when the centers $\mathbf{y}_k$ of trial functions (1.2) are chosen to be identical to the data locations $\mathbf{x}_j$ for $1 \leq j \leq m = n$. If there is no noise in the data, it then makes sense to reconstruct u^* by a function u of the form (1.2) by enforcing the exact interpolation conditions

$$u^*(\mathbf{x}_j) = \sum_{k=1}^{n} \alpha_j \phi(\|\mathbf{x}_j - \mathbf{x}_k\|_2),\ 1 \leq j \leq m = n. \tag{1.4}$$

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

$$\mathbf{A}_X := (\phi(\|\mathbf{x}_j - \mathbf{x}_k\|_2))_{1 \leq j,k \leq m} \tag{1.5}$$

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 1.6 *A radial basis function ϕ on $[0, \infty)$ is* **positive definite** *on $\mathbb{R}^d$, if for all choices of sets $X := \{\mathbf{x}_1, \ldots, \mathbf{x}_m\}$ of finitely many points $\mathbf{x}_1, \ldots, \mathbf{x}_m \in \mathbb{R}^d$ and arbitrary m the symmetric $m \times m$ symmetric matrices $\mathbf{A}_X$ of (1.5) are positive definite.*

Consequently, solvability of the system (1.4) is guaranteed, if ϕ satisfies the above definition. This holds for several standard radial basis function provided in Table 1, but users must be aware to run into problems when using other scalar functions such as $\exp(-r)$. A more complete list of radial basis functions will follow later on page 16.

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 (1.2). 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 then q comes out to be $q = \binom{Q-1+d}{d}$, and the trial functions of (1.2) are augmented to

$$u(\mathbf{x}) := \sum_{k=1}^{n} \alpha_k \phi(\|\mathbf{x} - \mathbf{y}_k\|_2) + \sum_{\ell=1}^{q} \beta_\ell p_\ell(\mathbf{x}). \tag{1.7}$$

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: Positive definite radial basis functions

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

$$\sum_{k=1}^{n} \alpha_k p_\ell(\mathbf{x}_k) = 0,\ 1 \le \ell \le q \tag{1.8}$$

restricting the coefficients $\alpha_1, \ldots, \alpha_n$ in (1.7). Unique solvability of the extended system

$$\begin{aligned} \sum_{k=1}^{n} \alpha_k \phi(\|\mathbf{x}_j - \mathbf{y}_k\|_2) + \sum_{\ell=1}^{q} \beta_\ell p_\ell(\mathbf{x}_j) &= u(\mathbf{x}_j), \quad 1 \le j \le n \\ \sum_{k=1}^{n} \alpha_k p_\ell(\mathbf{x}_k) &= 0, \qquad 1 \le \ell \le q \end{aligned} \tag{1.9}$$

is assured if

$$p(\mathbf{x}_k) = 0 \text{ for all } 1 \le k \le n \text{ and } p \in P^d_{Q-1} \text{ implies } p = 0. \tag{1.10}$$

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, (1.8) is obsolete, and (1.7) reduces to (1.2). 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 2 provides a selection of the most useful conditionally positive definite functions, and again we refer to page 16 for other radial basis functions.

1.2 Stability and Scaling

The system (1.4) is easy to program, and it is always solvable 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 nonvanishing radial basis functions. To handle bad condition of moderately large

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

Table 2: Conditionally positive definite radial basis functions

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 (1.4) or (1.9) 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 (1.2) or (1.7). 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 condition. 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 (1.2) 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 trick 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 [142, 132, 23]. The most useful example is Wendland's function

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

which is positive definite in $\mathbb{R}^d$ for $d \leq 3$ and twice differentiable in $\mathbf{x}$ when $r = \|\mathbf{x}\|_2$ (see Table 1 and other cases in 3 on page 16). Other localization techniques use fast **multipole methods** [11, 12] or a **partition of unity** [134]. This technique originated from **finite elements** [102, 6], 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** [88, 92, 133]. 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.

1.3 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 $\Omega \subset \mathbb{R}^3$ 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

$$\begin{aligned} -\Delta u &= f^\Omega \quad \text{in } \Omega \\ u &= f^\Gamma \quad \text{in } \Gamma \end{aligned} \tag{1.11}$$

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 $\mathbf{x} = (x_1, x_2, x_3)^T \in \mathbb{R}^3$. This way the problem is completely posed in terms of evaluations of functions and derivatives, without any integrations. However, it requires to take 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 exclusively deal 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.

1.4 Comparison of Strong and Weak Problems

Concerning the range of partial differential equation techniques we handle here 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.

On the analytical side, we shall only consider 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 adapt 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 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 [74] for a class of symmetric meshless methods. The meshless local Petrov–Galerkin (**MLPG**) method [3, 4, 5] 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, it only recently was put on a solid theoretical foundation [121]

Finally, the papers [74, 121] also indicate that **mixed** weak and strong problems are possible, confining the weak approach to areas where problems occur or data are distributional. Together with adaptivity, this technique will surely prove useful in the future.

1.5 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 1.1. Numerical computations indicate that these meshless methods are ideal for solving complex physical problems in strong form on irregular domains. Section **??** 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**:

$$\begin{array}{rcll} Lu & = & f^\Omega & \text{in} \quad \Omega \subset I\!R^d \\ u & = & f^\Gamma & \text{on} \quad \Gamma := \partial\Omega \end{array} \tag{1.12}$$

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

$$\begin{array}{rcll} Lu(x_j^\Omega) & = & f^\Omega(x_j^\Omega), & x_j^\Omega \in \Omega, \quad 1 \le j \le m_\Omega \\ u(x_j^\Gamma) & = & f^\Gamma(x_j^\Gamma), & x_j^\Gamma \in \Gamma \quad 1 \le j \le m_\Gamma \end{array} \tag{1.13}$$

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 1.1 and to be described in full generality in the following chapter. The exact solution u^* of the Dirichlet problem (1.12) will satisfy (1.13), 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 (1.13) 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 (1.13) when restricted to trial functions from U. This problem is hard to solve, and therefore collocation methods did not attract much attention so far from the mathematical community.

However, as we know from Chapter **??**, 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 [82]. The idea is to take trial functions of the

form (1.2) or (1.7), depending on the order of the positive definiteness of the radial basis function used. For positive q one also has to postulate (1.8), 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

$$\begin{array}{lcll} \sum_{k=1}^{n} \alpha_k \Delta\phi(\|\mathbf{x}_j^\Omega - \mathbf{y}_k\|_2) & + & \sum_{\ell=1}^{q} \beta_\ell \Delta p_\ell(\mathbf{x}_j^\Omega) & = f^\Omega(\mathbf{x}_j^\Omega), \quad 1 \le j \le m_\Omega \\ \sum_{k=1}^{n} \alpha_k \phi(\|\mathbf{x}_j^\Gamma - \mathbf{y}_k\|_2) & + & \sum_{\ell=1}^{q} \beta_\ell p_\ell(\mathbf{x}_j^\Gamma) & = f^\Gamma(\mathbf{x}_j^\Gamma), \quad 1 \le j \le m_\Gamma \\ \sum_{k=1}^{n} \alpha_k p_\ell(\mathbf{y}_k) & + & 0 & = 0, \qquad 1 \le \ell \le q, \end{array} \tag{1.14}$$

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 [73] where the problem is singular.

A variety of experimental studies, *e.g.:* by Kansa [83, 84], Golberg and Chen [57], 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 [70], the nonlinear **Burgers equation** with a shock wave [71], the shallow water equation for tide and current simulation in domains with irregular boundaries [67], and free boundary problems like the American option pricing [72, 68]. These cases will be reported in Chapter **??**. 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 [118] 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 points $\mathbf{y}_k$. If the trial points are dropped completely, one can recycle the test points to define new trial

functions by

$$u(\mathbf{x}) := \sum_{i=1}^{m_\Omega} \alpha_i^\Omega \Delta\phi(\|\mathbf{x} - \mathbf{x}_i^\Omega\|_2) + \sum_{j=1}^{m_\Gamma} \alpha_j^\Gamma \phi(\|\mathbf{x} - \mathbf{x}_j^\Gamma\|_2) + \sum_{\ell=1}^{q} \beta_\ell p_\ell(\mathbf{x}) \quad (1.15)$$

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

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

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

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

The first set of equations arises when applying Δ to (1.15) on the domain test points $\mathbf{x}_r^\Omega$. The second is the evaluation of (1.15) on the boundary test points $\mathbf{x}_s^\Gamma$. The third is a natural generalization of (1.8) to the current trial space. Note that the system has the general symmetric form

$$\begin{pmatrix} \mathbf{A}^{\Omega,\Omega} & \mathbf{A}^{\Omega,\Gamma} & \mathbf{P}^\Omega \\ \mathbf{A}^{\Omega,\Gamma T} & \mathbf{A}^{\Gamma,\Gamma} & \mathbf{P}^\Gamma \\ \mathbf{P}^{\Omega T} & \mathbf{P}^{\Gamma T} & \mathbf{0}_{q\times q} \end{pmatrix} \begin{pmatrix} \mathbf{a}^\Omega \\ \mathbf{a}^\Gamma \\ \mathbf{b}_q \end{pmatrix} = \begin{pmatrix} \mathbf{f}^\Omega \\ \mathbf{f}^\Gamma \\ \mathbf{0}_q \end{pmatrix} \quad (1.16)$$

with evident notation when compared to the previous display.

Under weak assumptions, such matrices are nonsingular [141, 79] because they arise as Hermite interpolation systems generalizing (1.4). The approach is called **symmetric collocation** and has a solid mathematical foundation [52, 51] making use of the symmetry of the discretized problem. We provide specific applications in Chapter **??** and some underlying theory in section 2.2.

1.6 Method of Fundamental Solutions

This method is a highly effective technique for solving **homogeneous** differential equations, *e.g.:* the potential problem (1.11) 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 $\mathbb{R}^3$ 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 [87] in 1964. During the past decade, 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, which is 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** [111] 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, like 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 $\mathbb{R}^2$.

Methods which solve homogeneous equations by superposition of general solutions and an approximation on the boundary have quite some history, dating back to Trefftz [130]. In particular, the work of L. Collatz [101] contains plenty of examples done in the 1960's. Recently, this subject was taken up again and called **boundary knot method** [32, 30, 31, 69], but we stick to the Method of Fundamental Solutions here.

1.7 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.

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

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.

1.8 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.

1.9 Lists of Radial Basis Functions

Table 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 [142], Wendland [132], and Buhmann [23]. We list a selection of

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 3: Global RBFs

Wendland's functions in Table 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 4: Selection of Wendland's compactly supported 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

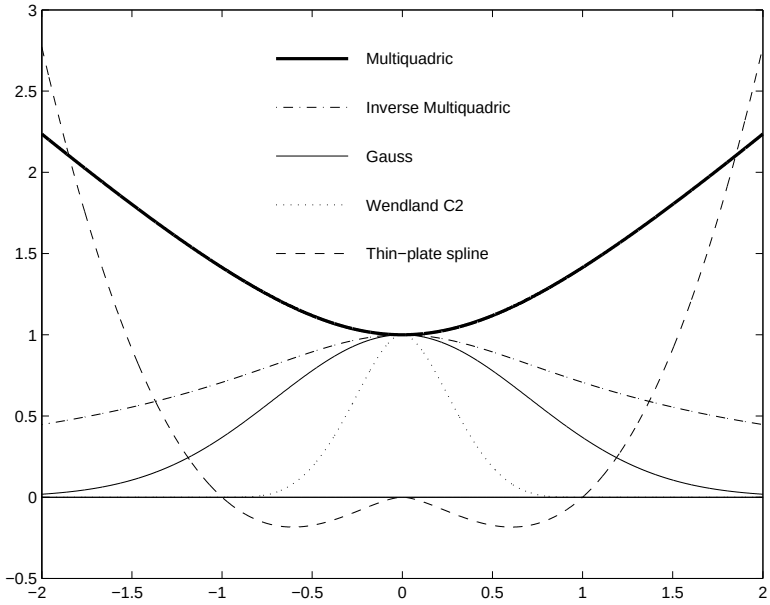

Figure 1: Some radial basis functions

[120, 121], 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 1.1, we assume data values $y_1, \ldots, y_m \in \mathbb{R}$ 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 $\mathbb{R}^d$. We then pick a positive definite radial basis function ϕ and set up the linear system (1.4) of m equations for the m coefficients $\alpha_1, \ldots, \alpha_m$ of the representation (1.2) 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 (1.7) and add the conditions (1.8).

In Figure 2 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.

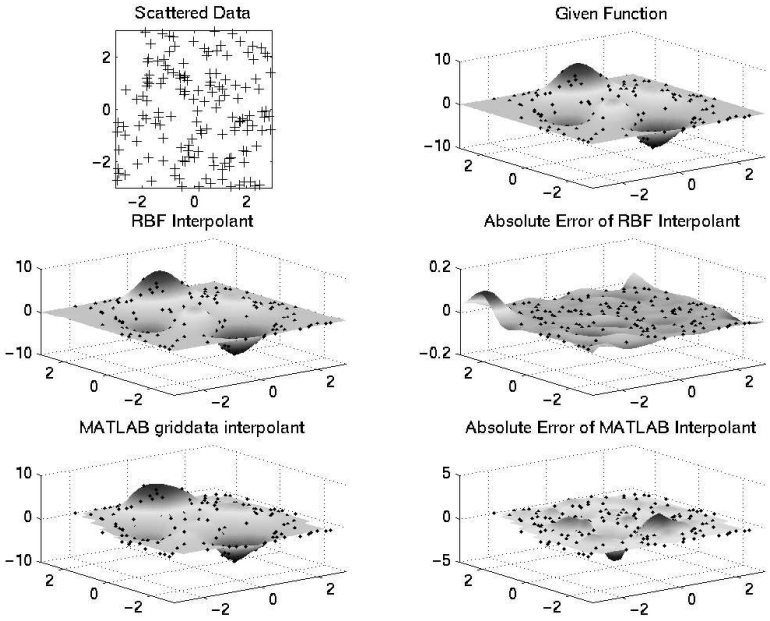

Figure 2: Interpolation by radial basis functions

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.:* [2, 26, 108, 25, 109, 112, 140, 17]. In particular, one can use such transformations to couple incompatible finite element codes [1].

Furthermore, interpolation of functions has quite some impact on methods solving partial differential equations. In Chapter **??** 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 to parametrize 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*" [16]. Since kernel interpolants approximate higher derivatives well, local function values can be used to provide good estimates for derivative data [131]. This has connections to **p**seudospectral methods

[44].

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 [141], is to use special trial functions

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

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 (1.2) 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 (1.4) turns into

$$\lambda_j u = \sum_{k=1}^{n} \alpha_k \lambda_k^{\mathbf{t}} \lambda_j^{\mathbf{x}} \phi(\|\mathbf{x}-\mathbf{t}\|_2), \ 1 \le j \le n \tag{2.1}$$

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.

To give an example with general functionals, Figure 3 shows an interpolation to Neumann data +1 and -1 on each half of the unit circle, respectively, in altogether 64 points by linear combinations of properly scaled Gaussians.

In case of conditionally positive definite radial basis functions, the span of (1.7) turns into

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

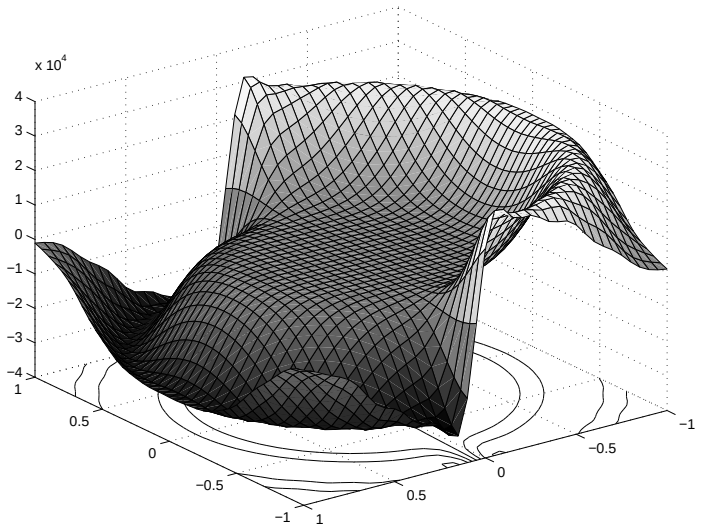

Figure 3: Generalized interpolant to Neumann data

while the additional condition (1.8) 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 additional condition

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

is imposed, replacing (1.10).

Another example of recovery from non-Lagrange data is the construction of **Lyapounov basins** from data consisting of **orbital derivatives** [54, 55].

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 1.5 and treated in Chapter **??** in full detail, solve partial differential equations numerically by interpolation of values of differential operators and boundary conditions.

Another important aspect is the possibility to implement additional 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 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 [24, 135] and recent papers [106]. They are completely *local*, and they are in terms of the **fill distance**

$$h := h(X,\Omega) := \sup_{\mathbf{y}\in\Omega}\min_{\mathbf{x}\in X}\|\mathbf{x}-\mathbf{y}\|_2 \tag{2.2}$$

of the discrete set $X = \{\mathbf{x}_1,\ldots,\mathbf{x}_n\}$ of centers with respect to the domain Ω where the error is measured. The interpolation error 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 even is exponential [98, 143] like $\exp(-c/h)$. Derivatives are also convergent as far as the smoothness of f and ϕ allows, but at a smaller rate, of course.

For interpolation of the smooth `peaks` function provided by MATLAB and used already in Figure 2, the error behavior on $[-3,3]^2$ as a function of fill distance h is given by Figure 4. It can be clearly seen that smooth ϕ yield smaller errors with higher convergence rates. In contrast to this, Figure 5 shows interpolation to the nonsmooth function

$$f(x,y) = 0.03 * \max(0, 6 - x^2 - y^2)^2, \tag{2.3}$$

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.

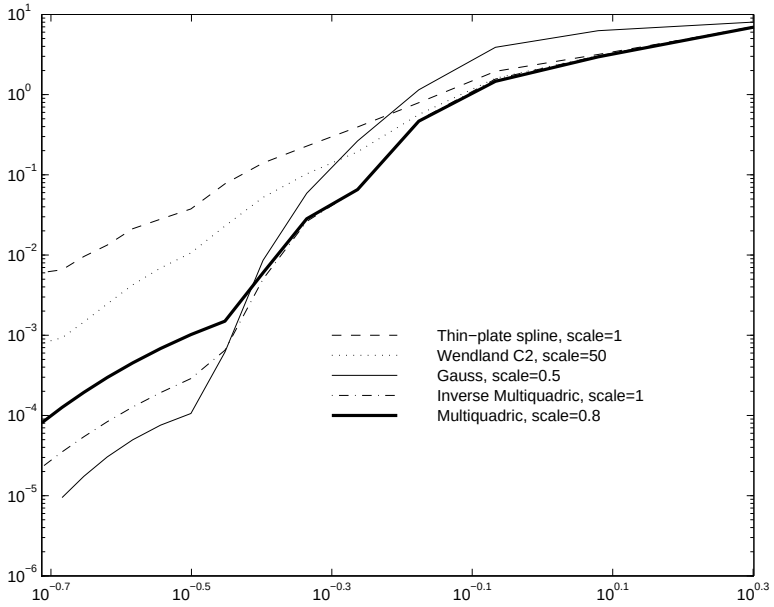

Figure 4: Nonstationary interpolation to a smooth function as a function of fill distance

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 (1.4) and (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 $X = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$ gets small. Figure 6 shows this effect in the situation of Figure 4.

If points are distributed well, the separation distance $S(X)$ will be proportional to the fill distance $h(X, \Omega)$ of (2.2). 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 $S(X) \le h(X, \Omega)$. But since separation distance only depends on the closest pair of points and ignores the rest, it is reasonable to avoid

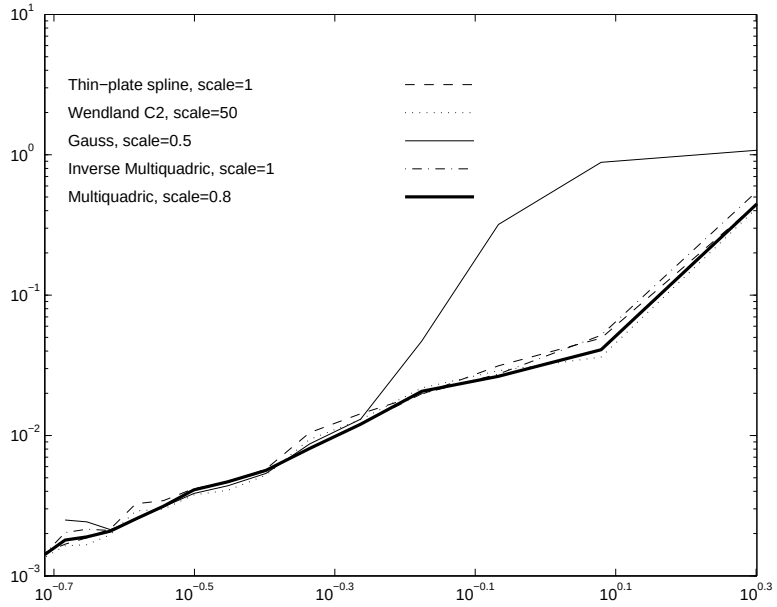

Figure 5: Nonstationary interpolation to a nonsmooth function as a function of fill distance

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

$$\gamma \, h(X, \Omega) \leq S(X) \leq h(X, \Omega). \tag{2.4}$$

To maintain quasi-uniformity, it suffices in most cases to delete "duplicates". Furthermore, there are sophisticated "thinning" techniques [49, 39, 136] to keep fill and separation distance proportional, *i.e.:* to assure quasi-uniformity at multiple scaling levels. We shall come back to this in section **??**.

Unless radial basis functions are rescaled in a data-dependent way, it can be proven [115] 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.

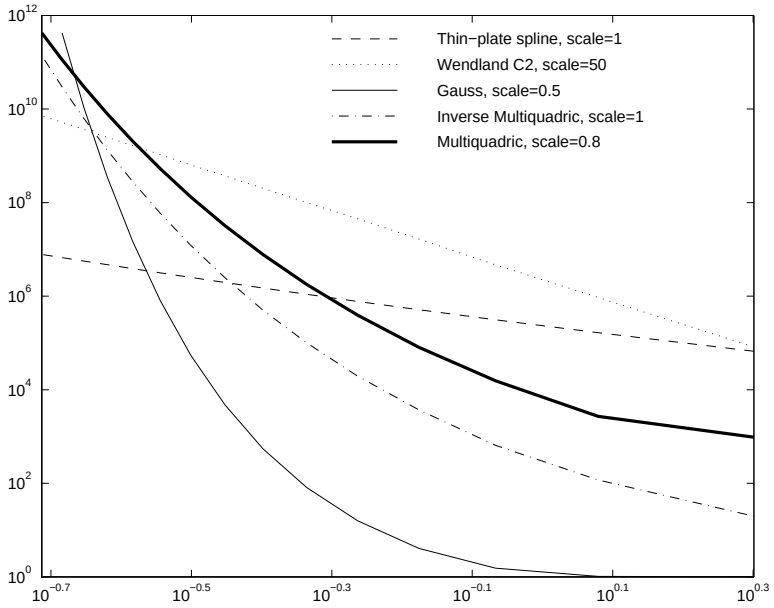

Figure 6: Condition as function of separation distance

We illustrate this by an example. Since [115] proves that the square of the L_∞ error roughly behaves like the smallest eigenvalue of the interpolation matrix, Figure 7 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 4, 8, and 6 to show the error and condition behavior there. Note that the curves do not vary much if compared to Figure 6. Example **??** for the Method of Fundamental Solutions shows a similarly close link between error and condition.

Thus **smoothness** of radial basis functions must be chosen with some care, and chosen 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 [106]. On the other hand, non-smooth radial basis functions will not achieve high convergence rates when approximating smooth functions [123]. 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 discussed in section 2.6.

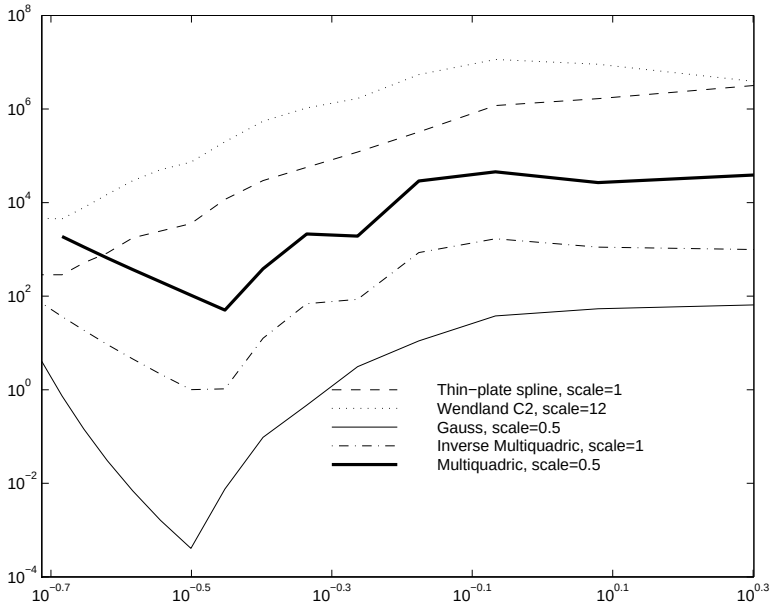

Figure 7: Squared L_∞ error times condition as a function of fill distance

2.5 Regularization

The linear systems arising in radial basis function methods have a special form of degeneration: the large eigenvalues 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 (1.5) 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

$$\mathbf{G}\mathbf{x} = \mathbf{b} \in I\!R^m \tag{2.5}$$

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 (1.3), (1.14), (1.16), (2.1) and those of subsequent chapters, *e.g.:* (**??**), and (**??**). In case of collocation (Chapter **??** or the Method of Fundamental Solutions (Chapter **??**), or already for the simple recovery problem (1.3) 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 [58] should do the job, *i.e.:* produce a numerical solution $\tilde{\mathbf{x}}$ which solves

$$\min_{\mathbf{x} \in I\!R^n} \|\mathbf{G}\mathbf{x} - \mathbf{b}\|_2 \tag{2.6}$$

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 to determine $\hat{\mathbf{x}}$ reliably if columns of the matrix $\mathbf{G}$ are close to being linearly dependent.

By **singular-value decomposition** [58], the matrix $\mathbf{G}$ can be decomposed into

$$\mathbf{G} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T \tag{2.7}$$

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 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 (2.7), the solution of either the minimization problem (2.6) or, in the case $m = n$, the solution of (2.5) 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

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

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 $\mathbf{x} = \mathbf{V}\mathbf{y}$ 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

$$\sum_{j=1}^{n}(\sigma_j \hat{y}_j - c_j)^2 + \sum_{j=n+1}^{m} c_j^2 \leq \eta^2. \tag{2.8}$$

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

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

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

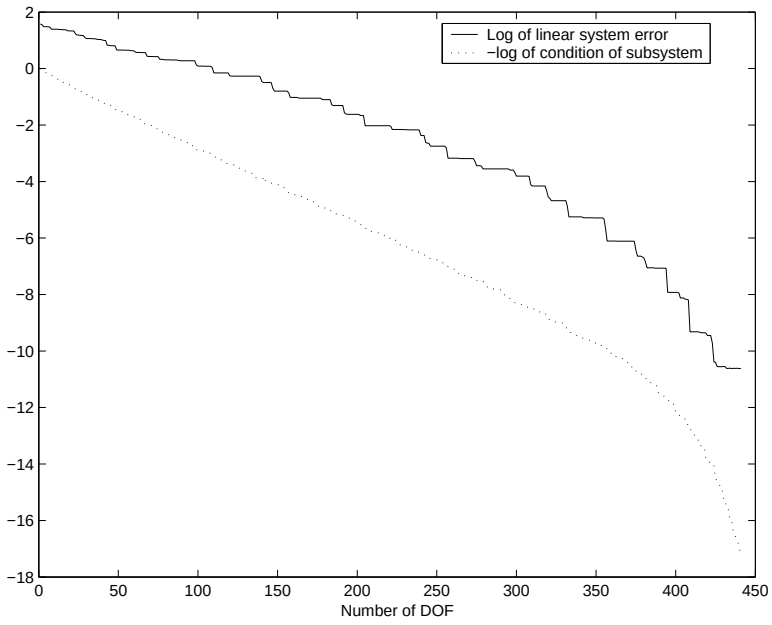

Figure 8: Error and condition of linear subsytems via SVD

Figure 8 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 (1.4). Following a fixed 441×441 singular value decomposition, we truncated after the k largest singular values, thus using only k degrees of freedom (DOF). 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 **??** in Chapter **??**.

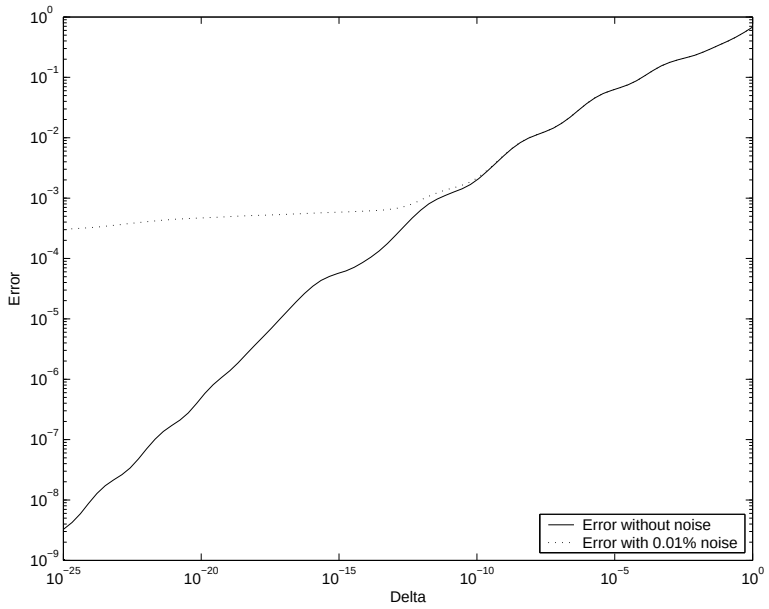

Figure 9: Error as function of regularization parameter δ^2

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 (2.8), 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

$$\sum_{j=1}^{n}(\sigma_j y_j - c_j)^2 + \sum_{j=n+1}^{m} c_j^2 \leq \eta^2, \tag{2.9}$$

but we suppress details of the analysis of this optimization problem. Another,

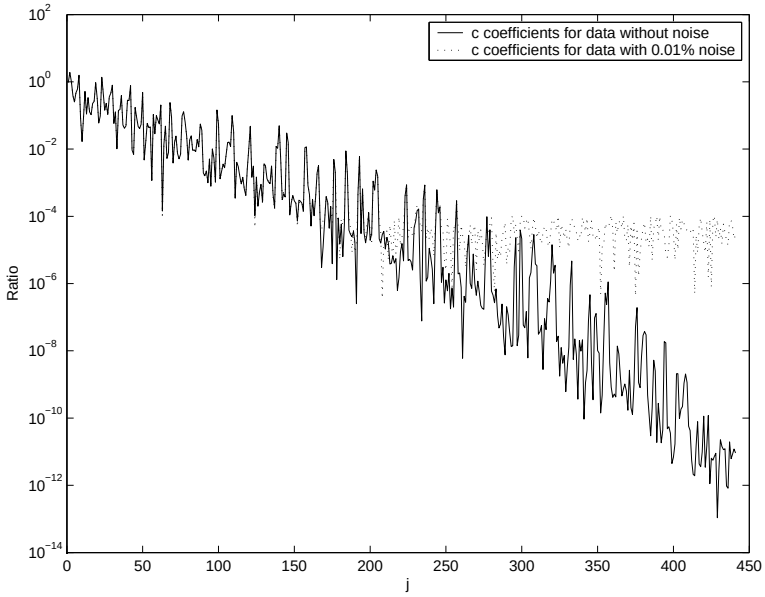

Figure 10: Coefficients $|c_j|$ as function of j

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 to put 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 me 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 (2.9) would be 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 $\|\mathbf{G}\mathbf{x}^\delta - \mathbf{b}\|_2^2$ for varying δ experimentally.

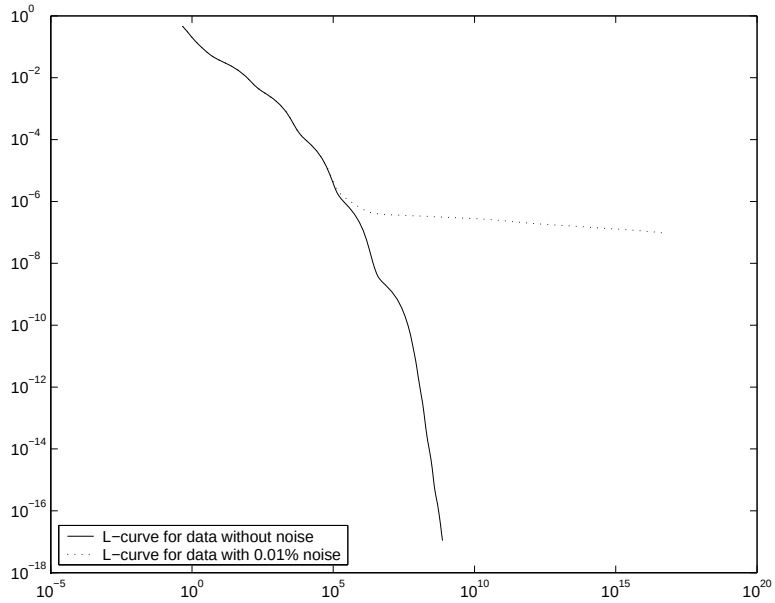

Figure 11: The L-curve for the same problem

We now repeat the example leading to Figure 8, replacing the truncation strategy by the above regularization. Figure 9 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 10 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 10 and 9 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 so, and there are various way to estimate the optimal δ, see [62, 63, 64] including a MATLAB software package.

Figure 11 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 **??** 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, and the corresponding numerical techniques are other, and possibly preferable regularizations which still are under investigation.

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** [40]. In numerical methods for solving differential equations, the **scale parameter** c is preferred, and it is called **shape factor** there. Readers should not be irritated by slighly other ways of scaling, *e.g.:*

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

for multiquadrics, because the outer factor c is irrelevant when forming trial spaces from functions (1.2). 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 (1.4), 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.

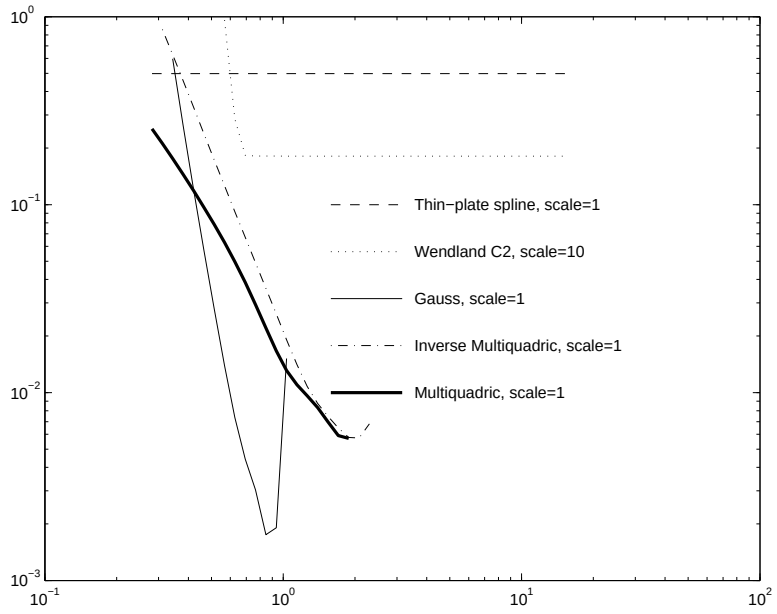

Figure 12: Error as function of relative scale, smooth case

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 [40, 117, 90, 91, 119]. This has implications for many examples in this book which approximate analytic solutions of partial differential equations by analytic radial basis functions like Gaussians or multiquadrics: whatever is calculated is close to a good polynomial approximation to the solution. Users might suggest to use 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 [89, 50] which circumvent the instability at large scales, but these are still under investigation.

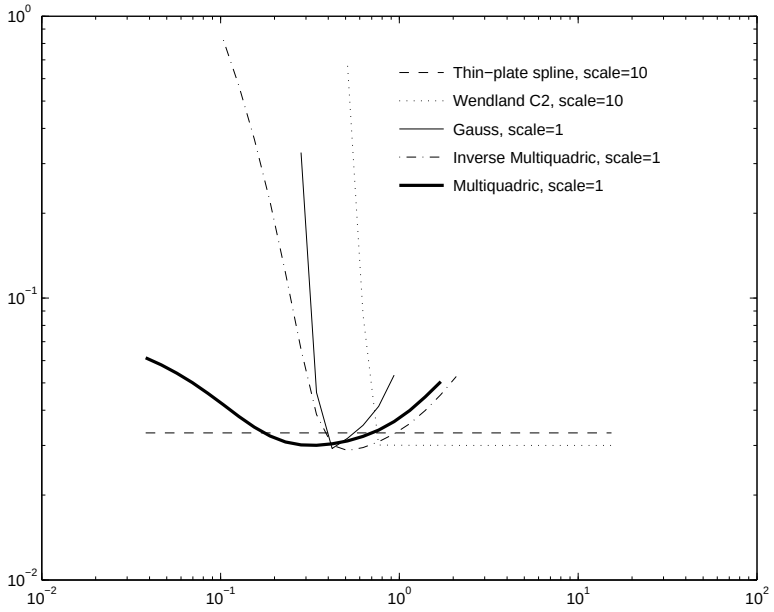

Figure 13: Error as function of relative scale, nonsmooth case

Figure 12 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 scales, the Gaussian radial basis function still shows some variation in the error as a function of scale.

Interpolating the nonsmooth function (2.3) shows a different behavior (see Figure 13), 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 [89, 50].

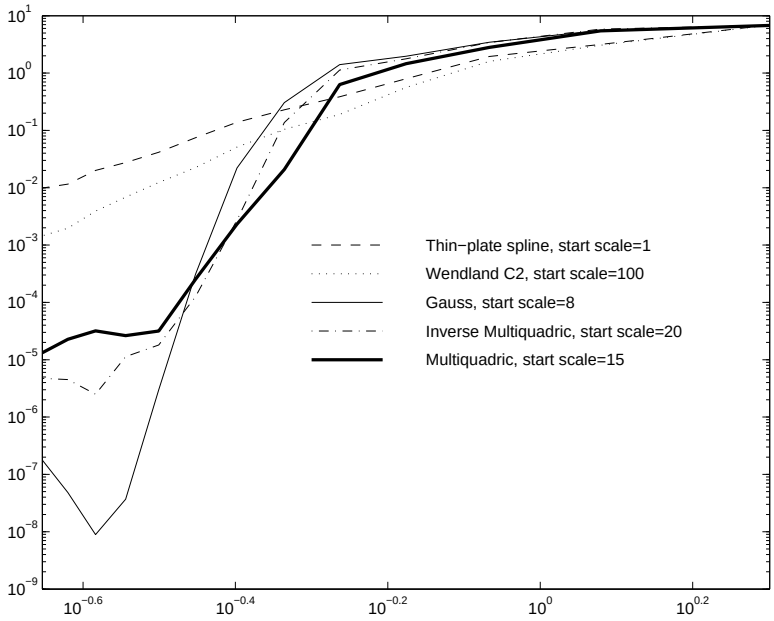

Figure 14: Stationary interpolation to a smooth function at small starting scales

Like 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 (2.2). 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 were 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 positive definite radial basis functions [21, 22]. 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 [116], and for special situations there is a sound theoretical analysis called **approximate**

approximation [100]. Figure 14 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 15 shows.

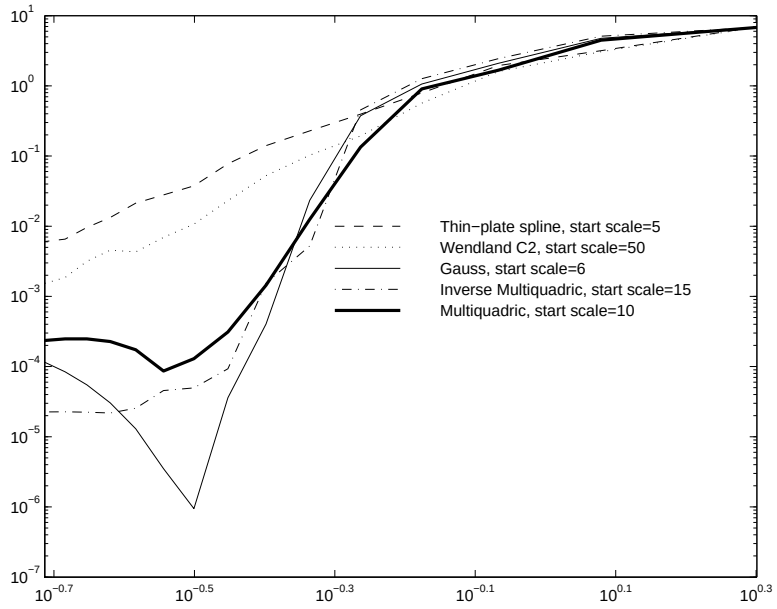

Figure 15: Stationary interpolation to a smooth function at wider starting scales

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 13, but this 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 for 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 (1.7) 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** [142, 132, 23] leads to sparse systems and localized evaluation. In particular, Wendland's functions have been applied successfully in plenty of applications, *e.g.:* [48, 27, 122, 27, 29, 139, 35, 36, 56, 103, 28, 137, 138, 86, 109, 43, 112, 140, 1] 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** [48, 41, 42, 105, 53, 28, 80].
- **partition of unity** methods [102, 6, 59, 60, 61, 134, 126, 109, 129] are a flexible and general tool for localizing **any** trial space. They have the advantage not to spoil 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 $\varphi_i : \mathbb{R}^d \to \mathbb{R}$ 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 [15, 12, 13, 8, 10, 37] though it is not easy to code.

- Fast evaluation using **transforms** is another choice [110, 47, 113], but it has not yet found its way into applications.

The dominant methods for reducing the complexity of *solving* large systems like (1.4) 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.:* [144, 95, 78]), but the majority of authors working with radial basis functions uses 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 (1.4) or (1.9) 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 [139]. 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** [46, 45] employing local **cardinal functions** in case of plain interpolation [9, 14, 85, 104, 96, 97, 20]. It can also be seen as an additive [77] or as a multiplicative **Schwarz decomposition** [14] 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 [7].

Altogether, there are many successful techniques now for handling large and ill-conditioned systems (see *e.g.:* an overview in [85]), and there are a few promising theoretical investigations [9, 46, 7, 20, 45, 117], but a general and complete mathematical foundation for handling large systems arising in Partial Differential Equations still is missing.

2.9 Sensitivity to Noise

So far, the discussion focused on noiseless data, with the exception of Figure 9. 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 (1.4) in the L_2 sense with oversampling, if only $n << m$ trial points $\mathbf{x}_j$ are used for m data points $\mathbf{y}_k$. The trial points 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 [118, 121, 119].

A third possibility is the old Levenberg-Marquardt trick of adding a positive value λ into the diagonal of the kernel matrix of (1.4) 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 [136], 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 (1.4) by an **optimization method** which penalizes the reproduction error on 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 to lead to linear optimization restrictions, while the discrete ℓ_2 norm leads to quadratic ones. For radial basis functions of the form (1.2) or (1.7), the quadratic form

$$\|u\|_\phi^2 := \sum_{j,k=1}^{n} \alpha_j \alpha_k \phi(\|\mathbf{x}_j - \mathbf{x}_k\|_2) \tag{2.10}$$

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 space** of functions with about half the smoothness of ϕ, penalizing all available derivatives there. For details, we have to refer to basic literature [24, 135] on the theory of radial basis functions. But even though we skip over native spaces here, all users should be aware that they always lure 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 function or more general **kernel methods** work well for many **ill-posed** and **inverse problems** [75, 93, 128, 34, 33, 76, 81, 94, 114, 107]. The above strategy of minimizing the quadratic form (2.10) also is central for modern methods of **machine learning**, but we cannot pursue this subject in detail here [38, 125, 127].

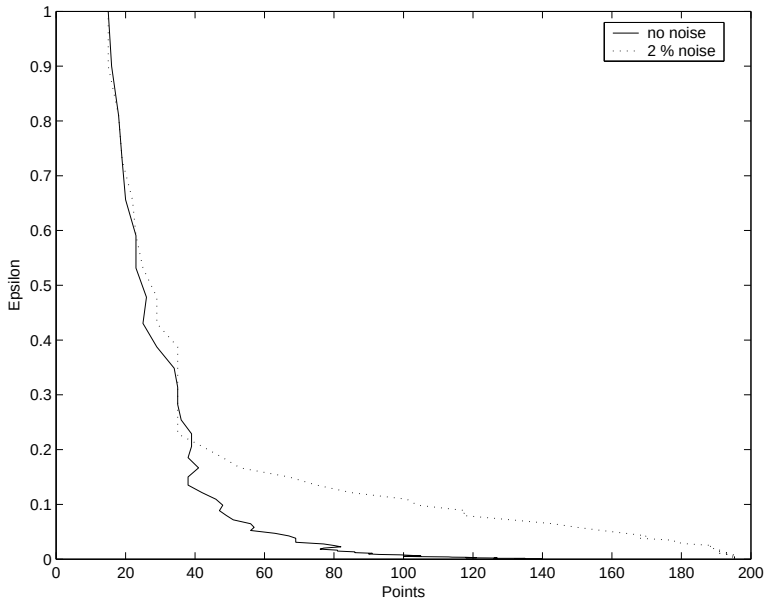

Figure 16: Connection between ϵ and the number $n(\epsilon)$ of necessary points

Let us use minimization of the quadratic form (2.10) to provide an example for 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 (2.10) under the constraints

$$-\epsilon \le \sum_{j=1}^{n} \alpha_j \phi(\|\mathbf{x}_j - \mathbf{x}_k\|_2) - f(\mathbf{x}_k) \le \epsilon,\ 1 \le k \le n \tag{2.11}$$

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 (2.11) is attained with equality. The number $n(\epsilon)$ of required points grows up to the maximally possible $n(0) = 196$ when ϵ decreases. Figure 16 shows this for the case of exact and noisy data.

But even more interesting is the behavior of the global $L_\infty[-3,3]^2$ error $E(\epsilon)$ as a function of ϵ. Figure 17 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 [65].

But also for exact data, these curves are useful. 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.

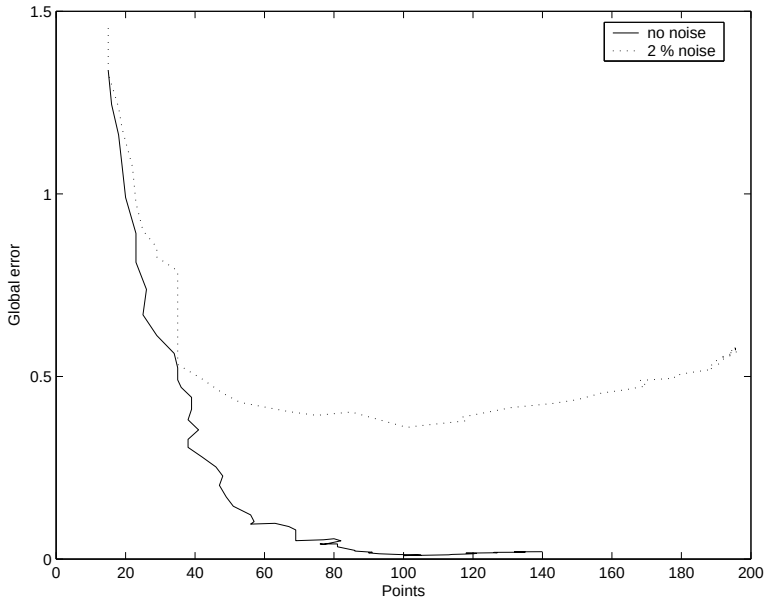

Figure 17: Error $E(\epsilon)$ as a function of the number $n(\epsilon)$ of necessary points

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 $\mathbf{y}_1, \ldots, \mathbf{y}_n$

and letting the coefficients in the representations (1.2) and (1.7) be time-dependent. If β_{ij} are the coefficients of the fixed inverse of the matrix in (1.4) (the case of (1.9) can be treated similarly), then the approximation to $u(\mathbf{x}, t)$ is of the simple form

$$\begin{aligned} \tilde{u}(\mathbf{x}, t) &= \sum_{k=1}^{n} \underbrace{\sum_{j=1}^{n} \beta_{kj} u(\mathbf{x}_j, t)}_{\alpha_k(t)} \phi(\|\mathbf{x} - \mathbf{x}_k\|_2) \\ &= \sum_{k=1}^{n} \alpha_k(t) \phi(\|\mathbf{x} - \mathbf{x}_k\|_2) \end{aligned} \tag{2.12}$$

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 like *e.g.:*

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

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 re-meshing.

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

$$\frac{\partial u}{\partial t}(\mathbf{x}, t) = F(L(u(\mathbf{x}, t)), \mathbf{x}, t) \tag{2.13}$$

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 **??** and **??**.

References

References

[1] 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*, 12:121–136, 2006.

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

[3] S.N. Atluri and T.-L. Zhu. A new meshless local Petrov-Galerkin (MLPG) approach in Computational mechanics. *Computational Mechanics*, 22:117–127, 1998.

[4] S.N. Atluri and T.-L. Zhu. A new meshless local Petrov-Galerkin (MLPG) approach to nonlinear problems in Computer modeling and simulation. *Computer Modeling and Simulation in Engineering*, 3:187–196, 1998.

[5] S.N. Atluri and T.-L. Zhu. The meshless local Petrov-Galerkin (MLPG) approach for solving problems in elasto-statics. *Computational Mechanics*, 25:169–179, 2000.

[6] I. Babuska and J.M. Melenk. The Partition of Unity Method. *Int. J. Numer. Meths. Eng.*, 40:727–758, 1997.

[7] B.J.C. Baxter. Preconditioned conjugate gradients, radial basis functions, and Toeplitz matrices. *Comput. Math. Appl.*, 43:305–318, 2002.

[8] R.K. Beatson and E. Chacko. Fast evaluation of radial basis functions: A multivariate momentary evaluation scheme. In A. Cohen, C. Rabut, and L.L. Schumaker, editors, *Curve and Surface Fitting: Saint-Malo 1999*, pages 37–46. Vanderbilt University Press, 2000.

[9] 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.*, 11:253–270, 1999.

[10] R.K. Beatson, J.B. Cherrie, and D.L. Ragozin. Polyharmonic splines in $\mathbb{R}^d$: Tools for fast evaluation. In A. Cohen, C. Rabut, and L.L. Schumaker, editors, *Curve and Surface Fitting: Saint-Malo 1999*, pages 47–56. Vanderbilt University Press, 2000.

[11] R.K. Beatson, G. Goodsell, and M.J.D. Powell. On multigrid techniques for thin plate spline interpolation in two dimensions. In *The mathematics of numerical analysis*, volume 32 of *Lectures in Appl. Math.*, pages 77–97. Amer. Math. Soc., Providence, RI, 1996.

[12] R.K. Beatson and L. Greengard. A short course on fast multipole methods. In M. Ainsworth, J. Levesley, W. Light, and M. Marletta, editors, *Wavelets, Multilevel Methods and Elliptic PDEs*, pages 1–37. Oxford University Press, 1997.

[13] R.K. Beatson and W.A. Light. Fast evaluation of radial basis functions: Methods for two–dimensional polyharmonic splines. *IMA Journal of Numerical Analysis*, 17:343–372, 1997.

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

[15] R.K. Beatson and G.N. Newsam. Fast evaluation of radial basis functions. I. *Comput. Math. Appl.*, 24(12):7–19, 1992. Advances in the theory and applications of radial basis functions.

[16] 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*, 139:3–47, 1996.

[17] M. Botsch and L. Kobbelt. Real-time shape editing using radial basis functions. In *Computer Graphics Forum*, volume 24, pages 611 – 621, 2005.

[18] D. Braess. *Finite Elements. Theory, Fast Solvers and Applications in Solid Mechanics*. Cambridge University Press, 2001.

[19] S.C. Brenner and L.R. Scott. *The Mathematical Theory of Finite Element Methods, Second edition*. Springer, 2002.

[20] 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*, 19:343–353, 2005.

[21] M.D. Buhmann. Convergence of univariate quasi-interpolation using multiquadrics. *IMA J. Numer. Anal.*, 8:365–383, 1988.

[22] M.D. Buhmann. Multivariate cardinal interpolation with radial–basis functions. *Constr. Approx.*, 6:225–255, 1990.

[23] M.D. Buhmann. Radial functions on compact support. *Proceedings of the Edinburgh Mathematical Society*, 41:33–46, 1998.

[24] M.D. Buhmann. *Radial Basis Functions, Theory and Implementations.* Cambridge University Press, 2003.

[25] J.C. Carr, R.K. Beatson, J.B. Cherrie, T.J. Mitchell, W.R. Fright, B.C. McCallum, and T.R. Evans. Reconstruction and representation of $3D$ objects with radial basis functions. In *Computer Graphics Proceedings*, pages 67–76. Addison Wesley, 2001.

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

[27] C.S. Chen, C.A. Brebbia, and H. Power. Dual reciprocity method using compactly supported radial basis functions. *Comm. Num. Meth. Eng.*, 15:137–150, 1999.

[28] 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*, 43:359–378, 2002.

[29] 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. In *Boundary elements, XXI (Oxford, 1999)*, volume 6 of *Int. Ser. Adv. Bound. Elem.*, pages 561–570. WIT Press, Southampton, 1999.

[30] W. Chen. Symmetric boundary knot method. *Engineering Analysis with Boundary Elements*, 26:489–494, 2002.

[31] 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.*, 192:1859–1875, 2003.

[32] W. Chen and M. Tanaka. New insights into boundary-only and domain-type RBF methods. *Int. J. Nonlinear Sci. & Numer. Simulation*, 1:145–151, 2000.

[33] A.H.-D. Cheng and J.J.S.P. Cabral. Direct solution of certain ill-posed boundary value problems by collocation method. In A. Kassab, C.A. Brebbia, E. Divo, and D. Poljak, editors, *Boundary Elements XXVII*, pages 35–44, 2005.

[34] A.H.-D. Cheng and J.J.S.P. Cabral. Direct solution of ill-posed boundary value problems by radial basis function collocation method. *Internat. J. Numer. Methods Engrg.*, 64(1):45–64, 2005.

[35] 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*, 24:549–557, 2000.

[36] 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*, 24:549–557, 2000.

[37] 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.*, 23:1272–1310, 2002.

[38] N. Cristianini and J. Shawe-Taylor. *An introduction to support vector machines and other kernel-based learning methods*. Cambridge University Press, Cambridge, 2000.

[39] L. Demaret, N. Dyn, M.S. Floater, and A. Iske. Adaptive thinning for terrain modelling and image compression. In *Advances in multiresolution for geometric modelling*, Math. Vis., pages 319–338. Springer, Berlin, 2005.

[40] T.A. Driscoll and B. Fornberg. Interpolation in the limit of increasingly flat radial basis functions. *Comput. Math. Appl.*, 43:413–422, 2002.

[41] G.E. Fasshauer. On smoothing for multilevel approximation with radial basis functions. In C. K. Chui and L.L. Schumaker, editors, *Approximation Theory IX, Vol. 2: Computational Aspects*, pages 55–62, Nashville, 1998. Vanderbilt University Press.

[42] G.E. Fasshauer. Solving differential equations with radial basis functions: Multilevel methods and smoothing. *Adv. Comput. Math.*, 11:139–159, 1999.

[43] G.E. Fasshauer. Approximate moving least-squares approximation with compactly supported radial weights. In *Meshfree methods for partial differential equations (Bonn, 2001)*, volume 26 of *Lect. Notes Comput. Sci. Eng.*, pages 105–116. Springer, Berlin, 2003.

[44] G.E. Fasshauer. RBF collocation methods and pseudospectral methods. Preprint, 2006.

[45] A.C. Faul, G. Goodsell, and M.J.D. Powell. A Krylov subspace algorithm for multiquadric interpolation in many dimensions. *IMA J. Numer. Anal.*, 25(1):1–24, 2005.

[46] A.C. Faul and M.J.D. Powell. Krylov subspace methods for radial basis function interpolation. In *Numerical analysis 1999 (Dundee)*, volume 420 of *Chapman & Hall/CRC Res. Notes Math.*, pages 115–141. Chapman & Hall/CRC, Boca Raton, FL, 2000.

[47] M. Fenn and G. Steidl. Fast NFFT based summation of radial functions. *Sampl. Theory Signal Image Process.*, 3(1):1–28, 2004.

[48] M.S. Floater and A. Iske. Multistep scattered data interpolation using compactly supported radial basis functions. *J. Comput. Applied Math.*, 73:65–78, 1996.

[49] M.S. Floater and A. Iske. Thinning algorithms for scattered data interpolation. *BIT*, 38:4:705–720, 1998.

[50] B. Fornberg and G. Wright. Stable computation of multiquadric interpolants for all values of the shape parameter. *Comput. Math. Appl.*, 48(5-6):853–867, 2004.

[51] C. Franke and R. Schaback. Convergence order estimates of meshless collocation methods using radial basis functions. *Adv. Comput. Math.*, 8:381–399, 1998.

[52] C. Franke and R. Schaback. Solving partial differential equations by collocation using radial basis functions. *Appl. Math. Comput.*, 93:73–82, 1998.

[53] 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.*, 119:161–183, 2000.

[54] P. Giesl and H. Wendland. Approximating the basin of attraction of time-periodic ODEs by meshless collocation. Preprint Gttingen/Mnchen, 2006.

[55] P. Giesl and H. Wendland. Meshless collocation: Error estimates with application to dynamical systems. Preprint Gttingen/Mnchen, 2006.

[56] 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*, 24:539–547, 2000.

[57] M.A. Golberg, C.S. Chen, and S. Karur. Improved multiquadric approximation for partial differential equations. *Engineering Analysis with Boundary Elements.*, 18(1):9–17, 1996.

[58] G. Golub and C. van Loan. *Matrix computations*. The Johns Hopkins University Press, 1996. Third edition.

[59] 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(3):853–890, 2000.

[60] M. Griebel and M.A. Schweitzer. A particle-partition of unity method - part II: Efficient cover construction and reliable integration. *SIAM J. Sci. Comput.*, 23:1655–1682, 2002.

[61] M. Griebel and M.A. Schweitzer. A particle-partition of unity method - part III: A multilevel solver. *SIAM J. Sci. Comput.*, 24:377–409, 2002.

[62] P.C. Hansen. Analysis of discrete ill-posed problems by means of the l-curve. *SIAM Review*, 34:561–580, 1992.

[63] P.C. Hansen. Regularization tools: a MATLAB package for analysis and solution of discrete ill-posed problems. *Numerical Algorithms*, 6:1–35, 1994.

[64] P.C. Hansen. The L-curve and its use in the numerical treatment of inverse problems. In P. Johnston, editor, *Computational Inverse Problems in Electrocardiology, Advances in Computational Bioengineering Series*. WIT Press, Southampton, 2000.

[65] 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*, 14:1487–1503, 1993.

[66] R.L. Hardy. Multiquadric equations of Topography and other irregular surfaces. *Journal of Geophysical Research*, 176:1905–1915, 1971.

[67] 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(4):21–33, 1993.

[68] Y.C. Hon. A quasi-radial basis function method for American options pricing. *Comput. Math. Applic.*, 43(3-5):513–524, 2002.

[69] 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*, 56:1931–1948, 2003.

[70] Y.C. Hon and X.Z. Mao. A multiquadric interpolation method for solving initial value problems. *Sci. Comput.*, 12(1):51–55, 1997.

[71] Y.C. Hon and X.Z. Mao. An efficient numerical scheme for Burgers' equation. *Appl. Math. Comput.*, 95(1):37–50, 1998.

[72] Y.C. Hon and X.Z. Mao. A radial basis function method for solving option pricing model. *Int. J. Financial Engineering*, 8(1):31–49, 1999.

[73] Y.C. Hon and R. Schaback. On unsymmetric collocation by radial basis functions. *J. Appl. Math. Comp.*, 119:177–186, 2001.

[74] Y.C. Hon and R. Schaback. Solvability of partial differential equations by meshless kernel methods. to appear in Adv. in Comp. Math., 2005.

[75] Y.C. Hon and T. Wei. A meshless scheme for solving inverse problems of Laplace equation. In *Recent development in theories & numerics*, pages 291–300. World Sci. Publishing, River Edge, NJ, 2003.

[76] Y.C. Hon and T. Wei. The method of fundamental solution for solving multidimensional inverse heat conduction problems. *CMES Comput. Model. Eng. Sci.*, 7(2):119–132, 2005.

[77] Y.C. Hon and Z.M. Wu. Additive Schwarz domain decomposition with a radial basis approximation. *Int. J. Appl. Math.*, 4:81–98, 2000.

[78] 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.*, 60:2183–2201, 2004.

[79] A. Iske. Reconstruction of functions from generalized Hermite-Birkhoff data. In C.K. Chui and L.L. Schumaker, editors, *Approximation Theory VIII, Vol. 1*, pages 257–264. World Scientific, Singapore, 1995.

[80] A. Iske. *Multiresolution methods in scattered data modelling*, volume 37 of *Lecture Notes in Computational Science and Engineering*. Springer-Verlag, Berlin, 2004.

[81] B. Jin and Y. Zheng. Boundary knot method for some inverse problems associated with the Helmholtz equation. *Internat. J. Numer. Methods Engrg.*, 62(12):1636–1651, 2005.

[82] E.J. Kansa. Application of Hardy's multiquadric interpolation to hydrodynamics. In *Proc. 1986 Simul. Conf., Vol. 4*, pages 111–117, 1986.

[83] E.J. Kansa. Multiquadrics - a scattered data approximation scheme with applications to computational fluid dynamics - I. *Comput. Math. Applic.*, 19(8/9):127–145, 1990.

[84] E.J. Kansa. Multiquadrics - a scattered data approximation scheme with applications to computational fluid dynamics - II. *Comput. Math. Applic.*, 19(8/9):147–161, 1990.

[85] E.J. Kansa and Y.C. Hon. Circumventing the ill-conditioning problem with multiquadric radial basis functions: applications to elliptic partial differential equations. *Comput. Math. Appl.*, 39(7-8):123–137, 2000.

[86] N. Kojekine, V. Savchenko, D. Berzin, and I. Hagiwara. Computer graphics applications based on compactly supported radial basis functions toolkit. *Sūrikaisekikenkyūsho Kōkyūroku*, 1288:64–72, 2002. Nonlinear problems in industry, domain decomposition methods and numerical simulations (Japanese) (Kyoto, 2002).

[87] V.D. Kupradze and M.A. Aleksidze. The method of functional equations for the approximate solution of certain boundary value problems. *U.S.S.R. Computational Mathematics and Mathematical Physics*, 4:82–126, 1964.

[88] P. Lancaster and K. Salkauskas. Surfaces generated by moving least squares methods. *Math. Comput.*, 37:141–158, 1981.

[89] E. Larsson and B. Fornberg. A numerical study of some radial basis function based solution methods for elliptic PDEs. *Comput. Math. Appl.*, 46:891–902, 2003.

[90] E. Larsson and B. Fornberg. Theoretical and computational aspects of multivariate interpolation with increasingly flat radial basis functions. *Comput. Math. Appl.*, 49(1):103–130, 2005.

[91] Y.J. Lee, G.J. Yoon, and J. Yoon. Convergence property of increasingly flat radial basis function interpolation to polynomial interpolation. preprint, 2005.

[92] D. Levin. The approximation power of moving least-squares. *Math. Comput.*, 67:1517–1531, 1998.

[93] J. Li. A radial basis meshless method for solving inverse boundary value problems. *Comm. Numer. Methods Engrg.*, 20(1):51–61, 2004.

[94] J. Li. Application of radial basis meshless methods to direct and inverse biharmonic boundary value problems. *Comm. Numer. Methods Engrg.*, 21(4):169–182, 2005.

[95] J. Li and Y.C. Hon. Domain decomposition for radial basis meshless methods. *Numer. Methods for PDEs*, 20:450–462, 2004.

[96] L. Ling and E.J. Kansa. Preconditioning for radial basis functions with domain decomposition methods. Math. Comput. Modelling (to appear), 2004.

[97] L. Ling and E.J. Kansa. A least-squares preconditioner for radial basis functions collocation methods. *Adv. Comput. Math.*, 23(1-2):31–54, 2005.

[98] W.R. Madych and S.A. Nelson. Bounds on multivariate polynomials and exponential error estimates for multiquadric interpolation. *J. Approx. Theory*, 70:94–114, 1992.

[99] J.C. Mairhuber. On Haar's theorem concerning Chebychev approximation problems having unique solutions. *Proc. Am. Math. Soc.*, 7:609–615, 1956.

[100] V. Maz'ya and G. Schmidt. On approximate approximations using Gaussian kernels. *IMA Journal of Numerical Analysis*, 16:13–29, 1996.

[101] G. Meinardus, G. Nürnberger, Th. Riessinger, and G. Walz. In memoriam: the work of Lothar Collatz in approximation theory. *J. Approx. Theory*, 67(2):119–128, 1991.

[102] J.M. Melenk and I. Babuska. The partition of unity finite element method: Basic theory and applications. *Comput. Meths. Appl. Mech. Engrg.*, 139:289–314, 1996.

[103] 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. In *International Conference on Shape Modeling and Applications (SMI 2001)*, pages 89–98. Computer Society Press, 2001.

[104] C.T. Mouat. *Fast Algorithms and Preconditioning Techniques for Fitting Radial Basis Functions*. PhD thesis, Mathematics Department, University of Canterbury, Christchurch, New Zealand, 2001.

[105] F.J. Narcowich, R. Schaback, and J.D. Ward. Multilevel interpolation and approximation. *Appl. Comput. Harmon. Anal.*, 7:243–261, 1999.

[106] 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.*, 24:175–186, 2006.

[107] Z. Nashed. Applications of wavelets and kernel methods in inverse problems. In *Integral methods in science and engineering*, pages 189–197. Birkhäuser Boston, Boston, MA, 2006.

[108] J.Y. Noh, D. Fidaleo, and U. Neumann. Animated deformations with radial basis functions. In *ACM Virtual Reality and Software Technology (VRST)*, pages 166–174, 2000.

[109] Y. Ohtake, A. Belyaev, and H.-P. Seidel. A multi-scale approach to 3D scattered data interpolation with compactly supported basis functions. In *SMI '03: Proceedings of the Shape Modeling International 2003*, page 292, Washington, DC, USA, 2003. IEEE Computer Society.

[110] D. Potts and G. Steidl. Fast summation at nonequispaced knots by NFFTs. *SIAM J. Sci. Comput.*, 24(6):2013–2037 (electronic), 2003.

[111] M.H. Protter and H.F. Weinberger. *Maximum Principles in Differential Equations*. Englewood Cliffs, 1967.

[112] P. Reuter, I. Tobor, C. Schlick, and S. Dedieu. Point-based modelling and rendering using radial basis functions. In *Proceedings of ACM Graphite 2003*, pages 111–118, 2003.

[113] G. Roussos and Brad J.C. Baxter. Rapid evaluation of radial basis functions. *J. Comput. Appl. Math.*, 180(1):51–70, 2005.

[114] S. Saitoh. Applications of reproducing kernels to best approximations, Tikhonov regularizations and inverse problems. In *Advances in analysis*, pages 439–445. World Sci. Publ., Hackensack, NJ, 2005.

[115] R. Schaback. Error estimates and condition numbers for radial basis function interpolation. *Adv. Comput. Math.*, 3:251–264, 1995.

[116] R. Schaback. On the efficiency of interpolation by radial basis functions. In A. LeMéhauté, C. Rabut, and L.L. Schumaker, editors, *Surface Fitting and Multiresolution Methods*, pages 309–318. Vanderbilt University Press, Nashville, TN, 1997.

[117] R. Schaback. Convergence analysis of methods for solving general equations. In A. Kassab, C.A. Brebbia, E. Divo, and D. Poljak, editors, *Boundary Elements XXVII*, pages 17–24. WITPress, Southampton, 2005.

[118] R. Schaback. Convergence of unsymmetric kernel-based meshless collocation methods. Preprint, 2006.

[119] R. Schaback. Limit problems for interpolation by analytic radial basis functions. Preprint, to appear in J. of Computational and Applied Mathematics, 2006.

[120] R. Schaback. Recovery of functions from weak data using unsymmetric meshless kernel-based methods. Preprint, 2006.

[121] R. Schaback. Unsymmetric meshless methods for operator equations. Preprint, 2006.

[122] R. Schaback and H. Wendland. Using compactly supported radial basis functions to solve partial differential equations. In C.S. Chen, C.A. Brebbia, and D.W. Pepper, editors, *Boundary Element Technology XIII*, pages 311–324. WitPress, Southampton, Boston, 1999.

[123] R. Schaback and H. Wendland. Approximation by positive definite kernels. for the IDOMat2001 proceedings (invited lecture), 2002.

[124] R. Schaback and H. Wendland. Kernel techniques: From machine learning to meshless methods. *Acta Numerica*, 15:543–639, 2006.

[125] B. Schölkopf and A.J. Smola. *Learning with Kernels*. MIT Press, Cambridge, 2002.

[126] M.A. Schweitzer. *A Parallel Multilevel Partition of Unity Method for Elliptic Partial Differential Equations*, volume 29 of *Lecture Notes in Computational Science and Engineering*. Springer, 2003.

[127] J. Shawe-Taylor and N. Cristianini. *Kernel Methods for Pattern Analysis*. Cambridge University Press, 2004.

[128] 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(3):285–291, 2004.

[129] 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(3):467–474, 2004.

[130] E. Trefftz. Ein gegenstück zum ritzschen verfahren. In *2. Int. Kongr. f. Techn. Mechanik, Zrich*, pages 131–137. 1926.

[131] T. Wei, Y.C. Hon, and Y.B. Wang. Reconstruction of numerical derivatives from scattered noisy data. *Inverse Problems*, 21:657 – 672, 2005.

[132] H. Wendland. Piecewise polynomial, positive definite and compactly supported radial functions of minimal degree. *Adv. Comput. Math.*, 4:389–396, 1995.

[133] H. Wendland. Local polynomial reproduction and moving least squares approximation. *IMA Journal of Numerical Analysis*, 21:285–300, 2001.

[134] H. Wendland. Fast evaluation of radial basis functions: Methods based on partition of unity. In C.K. Chui, L.L. Schumaker, and J. Stöckler, editors, *Approximation Theory X: Wavelets, Splines, and Applications*, pages 473–483. Vanderbilt University Press, 2002.

[135] H. Wendland. *Scattered Data Approximation*. Cambridge University Press, 2005.

[136] H. Wendland and C. Rieger. Approximate interpolation with applications to selecting smoothing parameters. *Numer. Math.*, 101:643–662, 2005.

[137] S.M. Wong, Y.C. Hon, and M.A. Golberg. Compactly supported radial basis functions for shallow water equations. *Appl. Math. Comput.*, 127(1):79–101, 2002.

[138] S.M. Wong, Y.C. Hon, and M.A. Golberg. Compactly supported radial basis functions for the shallow water equations. *Appl. Math. Comput.*, 127:79–101, 2002.

[139] 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. *Comput. Math. Appl.*, 37(8):23–43, 1999.

[140] J.H. Wu and L. Kobbelt. Efficient spectral watermarking of large meshes with orthogonal basis functions. In *The Visual Computers (Pacific Graphics 2005 Proceedings)*, volume 21 (8-10), pages 848–857, 2005.

[141] Z.M. Wu. Hermite–Birkhoff interpolation of scattered data by radial basis functions. *Approximation Theory Appl.*, 8/2:1–10, 1992.

[142] Z.M. Wu. Multivariate compactly supported positive definite radial functions. *Adv. Comput. Math.*, 4:283–292, 1995.

[143] J. Yoon. Spectral approximation orders of radial basis function interpolation on the Sobolev space. *SIAM J. Math. Anal.*, 33(4):946–958 (electronic), 2001.

[144] X. Zhou, Y.C. Hon, and J. Li. Overlapping domain decomposition method by radial basis functions. *Appl. Numer. Math.*, 44:241–255, 2003.