Meshfree Approximation Methods with MATLAB

INTERDISCIPLINARY MATHEMATICAL SCIENCES

Published

Vol. 1: Global Attractors of Nonautonomous Dissipative Dynamical Systems
David N. Cheban

Vol. 2: Stochastic Differential Equations: Theory and Applications
A Volume in Honor of Professor Boris L. Rozovskii
eds. Peter H. Baxendale & Sergey V. Lototsky

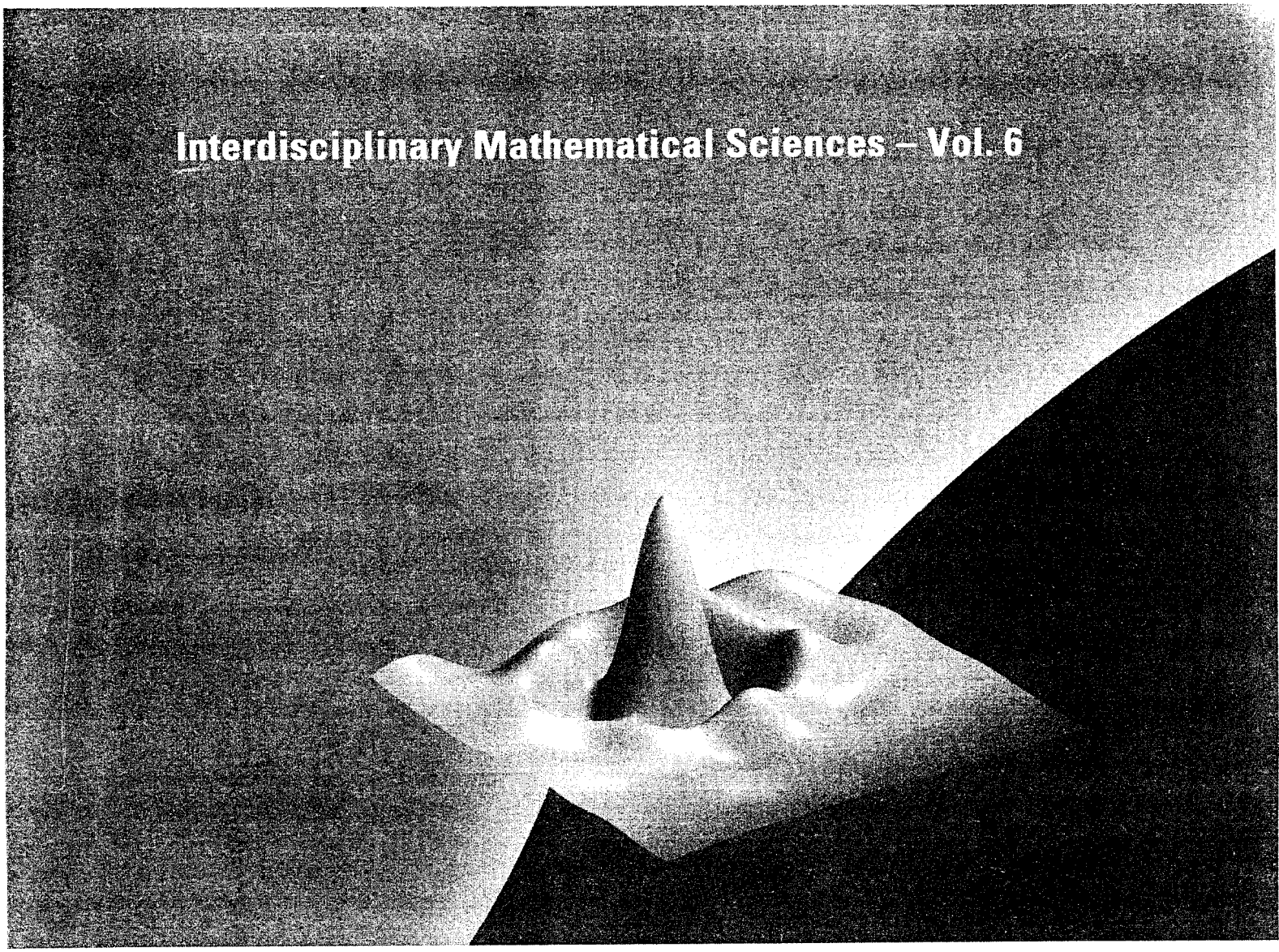
Vol. 3: Amplitude Equations for Stochastic Partial Differential Equations
Dirk Blömker

Vol. 4: Mathematical Theory of Adaptive Control
Vladimir G. Sragovich

Vol. 5: The Hilbert–Huang Transform and Its Applications
Norden E. Huang & Samuel S. P. Shen

Vol. 6: Meshfree Approximation Methods with MATLAB
Gregory E. Fasshauer

Interdisciplinary Mathematical Sciences – Vol. 6



Meshfree Approximation Methods with MATLAB

Gregory E. Fasshauer
Illinois Institute of Technology, USA

Published by

World Scientific Publishing Co. Pte. Ltd.
5 Toh Tuck Link, Singapore 596224
USA office: 27 Warren Street, Suite 401-402, Hackensack, NJ 07601
UK office: 57 Shelton Street, Covent Garden, London WC2H 9HE

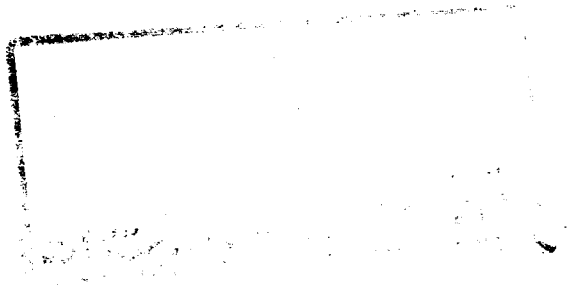
British Library Cataloguing-in-Publication Data
A catalogue record for this book is available from the British Library.

MESHFREE APPROXIMATION METHODS WITH MATLAB
(With CD-ROM)
Interdisciplinary Mathematical Sciences — Vol. 6

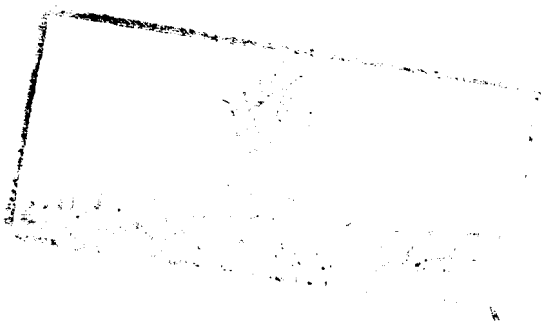
ISBN-13 978-981-270-633-1
ISBN-10 981-270-633-X
ISBN-13 978-981-270-634-8 (pbk)
ISBN-10 981-270-634-8 (pbk)

Printed by Mainland Press Pte Ltd

This book is dedicated to
Inge, Conny, Marc and Patrick.



Preface



Traditional numerical methods, such as finite element, finite difference, or finite volume methods, were motivated mostly by early one- and two-dimensional simulations of engineering problems via partial differential equations (PDEs). The discretization involved in all of these methods requires some sort of underlying computational mesh, *e.g.*, a triangulation of the region of interest. Creation of these meshes (and possible re-meshing) becomes a rather difficult task in three dimensions, and virtually impossible for higher-dimensional problems. This is where *meshfree* methods enter the picture. Meshfree methods are often — but by no means have to be — radially symmetric in nature. This is achieved by composing some univariate basic function with a (Euclidean) norm, and therefore turning a problem involving many space dimensions into one that is virtually one-dimensional. Such *radial basis functions* are at the heart of this book. Some people have argued that there are three "big technologies" for the numerical solution of PDEs, namely finite difference, finite element, and spectral methods. While these technologies came into their own right in successive decades, namely finite difference methods in the 1950s, finite element methods in the 1960s, and spectral methods in the 1970s, meshfree methods started to appear in the mathematics literature in the 1980s, and they are now on their way to becoming "big technology" number four. In fact, we will demonstrate in later parts of this book how different types of meshfree methods can be viewed as generalizations of the traditional "big three".

Multivariate meshfree approximation methods are being studied by many researchers. They exist in many flavors and are known under many names, *e.g.*, diffuse element method, element-free Galerkin method, generalized finite element method, *hp*-clouds, meshless local Petrov-Galerkin method, moving least squares method, partition of unity finite element method, radial basis function method, reproducing kernel particle method, smooth particle hydrodynamics method.

In this book we are concerned mostly with the moving least squares (MLS) and radial basis function (RBF) methods. We will consider all different kinds of aspects of these meshfree approximation methods: How to construct them? Are these constructions mathematically justifiable? How accurate are they? Are there ways to implement them efficiently with standard mathematical software packages such

as MATLAB? How do they compare with traditional methods? How do the various flavors of meshfree methods differ from one another, and how are they similar to one another? Is there a general framework that captures all of these methods? What sort of applications are they especially well suited for?

While we do present much of the underlying theory for RBF and MLS approximation methods, the emphasis in this book is not on proofs. For readers who are interested in all the mathematical details and intricacies of the theory we recommend the two excellent recent monographs [Buhmann (2003); Wendland (2005a)]. Instead, our objective is to make the theory accessible to a wide audience that includes graduate students and practitioners in all sorts of science and engineering fields. We want to put the mathematical theory in the context of applications and provide MATLAB implementations which give the reader an easy entry into meshfree approximation methods. The skilled reader should then easily be able to modify the programs provided here for his/her specific purposes.

In a certain sense the present book was inspired by the beautiful little book [Trefethen (2000)]. While the present book is much more expansive (filling more than five hundred pages with forty-seven MATLAB[1] programs, one Maple[2] program, one hundred figures, more than fifty tables, and more than five hundred references), it is our aim to provide the reader with relatively simple MATLAB code that illustrates just about every aspect discussed in the book.

All MATLAB programs printed in the text (as well as a few modifications discussed) are also included on the enclosed CD. The folder `MATLAB` contains M-files and data files of type MAT that have been written and tested with MATLAB 7. For those readers who do not have access to MATLAB 7, the folder `MATLAB6` contains versions of these files that are compatible with the older MATLAB release. The main difference between the two versions is the use of anonymous functions in the MATLAB 7 code as compared to `inline` functions in the MATLAB 6 version. Two packages from the MATLAB Central File Exchange [MCFE] are used throughout the book: the function `haltonseq` written by Daniel Dougherty and used to generate sequences of Halton points; the *kd*-tree library (given as a set of MATLAB MEX-files) written by Guy Shechter and used to generate the *kd*-tree data structure underlying our sparse matrices based on compactly supported basis functions. Both of these packages are discussed in Appendix A and need to be downloaded separately. The folder `Maple` on the CD contains the one Maple file mentioned above.

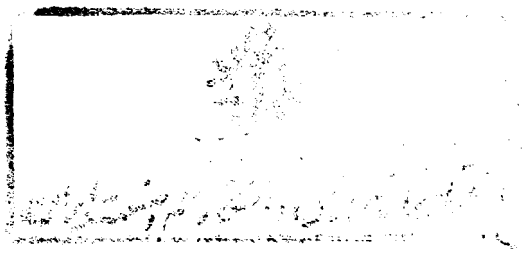
The manuscript for this book and some of its earlier incarnations have been used in graduate level courses and seminars at Northwestern University, Vanderbilt University, and the Illinois Institute of Technology. Special thanks are due to Jon

[1] MATLAB® is a trademark of The MathWorks, Inc. and is used with permission. The MathWorks does not warrant the accuracy of the text or exercises in this book. This book's use or discussion of MATLAB software or related products does not constitute endorsement or sponsorship by The MathWorks of a particular pedagogical approach or particular use of the MATLAB software.

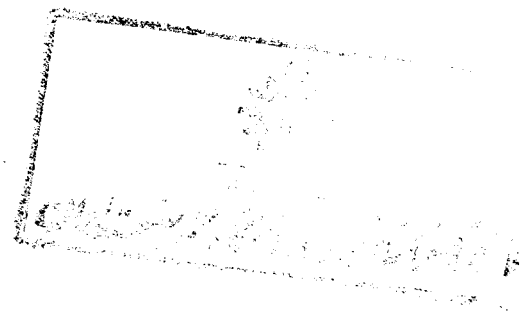
[2] Maple™ is a registered trademark of Waterloo Maple Inc.

Cherrie, John Erickson, Paritosh Mokhasi, Larry Schumaker, and Jack Zhang for reading various portions of the manuscript and/or MATLAB code and providing helpful feedback. Finally, thanks are due to all the people at World Scientific Publishing Co. who helped make this project a success: Rajesh Babu, Ying Oi Chiew, Linda Kwan, Rok Ting Tan, and Yubing Zhai.

Greg Fasshauer
Chicago, IL, January 2007



Contents



Chapter 1

Introduction

Meshfree methods have gained much attention in recent years, not only in the mathematics but also in the engineering community. Thus, much of the work concerned with meshfree approximation methods is interdisciplinary — at the interface between mathematics and numerous application areas (see the partial list below). Moreover, computation with high-dimensional data is an important issue in many areas of science and engineering. Many traditional numerical methods can either not handle such problems at all, or are limited to very special (regular) situations. Meshfree methods are often better suited to cope with changes in the geometry of the domain of interest (*e.g.*, free surfaces and large deformations) than classical discretization techniques such as finite differences, finite elements or finite volumes. Another obvious advantage of meshfree discretizations is — of course — their independence from a mesh. Mesh generation is still the most time consuming part of any mesh-based numerical simulation. Since meshfree discretization techniques are based only on a set of independent points, these costs of mesh generation are eliminated. Meshfree approximation methods can be seen to provide a new generation of numerical tools. Other traditional numerical methods such as the finite element, finite difference or finite volume methods are usually limited to problems involving two or three parameters (space dimensions). However, in many applications the number of parameters can easily range in the hundreds or even thousands. Multivariate approximation methods present one way to address these issues.

Applications of meshfree methods can be found

- in many different areas of science and engineering via *scattered data modeling* (*e.g.*, fitting of potential energy surfaces in chemistry; coupling of engineering models with sets of incompatible parameters; mapping problems in geodesy, geophysics, meteorology);
- in many different areas of science and engineering via *solution of partial differential equations* (*e.g.*, solution of gas dynamics equations, Boltzmann and Fokker-Planck equations in six-dimensional phase space; problems involving moving discontinuities such as cracks and shocks, multi-scale resolution, large material distortions; elasticity studies in plate and shell bending

problems; applications in nanotechnology);
- in *non-uniform sampling* (*e.g.*, medical imaging, tomographic reconstruction);
- in *mathematical finance* (*e.g.*, option pricing);
- in *computer graphics* (*e.g.*, representation of surfaces from point information such as laser range scan data, image warping);
- in *learning theory*, *neural networks* and *data mining* (*e.g.*, kernel approximation, support vector machines);
- in *optimization*.

Since many of these applications either come down to a function approximation problem, or include function approximation as a fundamental component, we will begin our discussion with — and in fact base a large part of the contents of this book on — the multivariate scattered data interpolation problem.

1.1 Motivation: Scattered Data Interpolation in $\mathbb{R}^s$

We will now describe the general process of scattered data fitting, which is one of the fundamental problems in approximation theory and data modeling in general. Our desire to have a well-posed problem formulation will naturally lead to an introductory example based on the use of so-called *distance matrices*. In the next chapters we will generalize this approach by introducing the concept of a radial basis function.

1.1.1 *The Scattered Data Interpolation Problem*

In many scientific disciplines one faces the following problem: We are given a set of data (measurements, and locations at which these measurements were obtained), and we want to find a rule which allows us to deduce information about the process we are studying also at locations different from those at which we obtained our measurements. Thus, we are trying to find a function $\mathcal{P}_f$ which is a "good" fit to the given data. There are many ways to decide what we mean by "good", and the only criterion we will consider now is that we want the function $\mathcal{P}_f$ to exactly match the given measurements at the corresponding locations. This approach is called *interpolation*, and if the locations at which the measurements are taken do not lie on a uniform or regular grid, then the process is called *scattered data interpolation*.

To give a precise definition we assume that the measurement locations (or *data sites*) are labeled $\boldsymbol{x}_j$, $j = 1, \ldots, N$, and the corresponding measurements (or *data values*) are called y_j. We will use $\mathcal{X}$ to denote the set of data sites and assume that $\mathcal{X} \subset \Omega$ for some region Ω in $\mathbb{R}^s$. Throughout this book we will restrict our discussion to scalar-valued data, *i.e.*, $y_j \in \mathbb{R}$. However, much of the following can be generalized easily to problems with vector-valued data. In many of our later

discussions we will assume that the data are obtained by sampling some (unknown) function f at the data sites, *i.e.*, $y_j = f(\boldsymbol{x}_j)$, $j = 1, \ldots, N$. Our notation $\mathcal{P}_f$ for the interpolating function emphasizes the connection between the interpolant and the data function f. We are now ready for a precise formulation of the scattered data interpolation problem.

Problem 1.1 (Scattered Data Interpolation). *Given data* $(\boldsymbol{x}_j, y_j)$, $j = 1, \ldots, N$, *with* $\boldsymbol{x}_j \in \mathbb{R}^s$, $y_j \in \mathbb{R}$, *find a (continuous) function* $\mathcal{P}_f$ *such that* $\mathcal{P}_f(\boldsymbol{x}_j) = y_j$, $j = 1, \ldots, N$.

The fact that we allow $\boldsymbol{x}_j$ to lie in an arbitrary s-dimensional space $\mathbb{R}^s$ means that the formulation of Problem 1.1 allows us to cover many different types of applications. If $s = 1$ the data could, *e.g.*, be a series of measurements taken over a certain time period, thus the "data sites" $\boldsymbol{x}_j$ would correspond to certain time instances. For $s = 2$ we can think of the data being obtained over a planar region, and so $\boldsymbol{x}_j$ corresponds to the two coordinates in the plane. For instance, we might want to produce a map that shows the rainfall in the state we live in based on the data collected at weather stations located throughout the state. For $s = 3$ we might think of a similar situation in space. One possibility is that we could be interested in the temperature distribution inside some solid body. Higher-dimensional examples might not be that intuitive, but a multitude of them exist, *e.g.*, in finance, optimization, economics or statistics, but also in artificial intelligence or learning theory.

A convenient and common approach to solving the scattered data problem is to make the assumption that the function $\mathcal{P}_f$ is a linear combination of certain *basis functions* B_k, *i.e.*,

$$\mathcal{P}_f(\boldsymbol{x}) = \sum_{k=1}^{N} c_k B_k(\boldsymbol{x}), \qquad \boldsymbol{x} \in \mathbb{R}^s. \tag{1.1}$$

Solving the interpolation problem under this assumption leads to a system of linear equations of the form

$$A\mathbf{c} = \boldsymbol{y},$$

where the entries of the *interpolation matrix* A are given by $A_{jk} = B_k(\boldsymbol{x}_j)$, $j, k = 1, \ldots, N$, $\mathbf{c} = [c_1, \ldots, c_N]^T$, and $\boldsymbol{y} = [y_1, \ldots, y_N]^T$.

Problem 1.1 will be *well-posed*, *i.e.*, a solution to the problem will exist and be unique, if and only if the matrix A is non-singular.

In the univariate setting it is well known that one can interpolate to arbitrary data at N distinct data sites using a polynomial of degree $N-1$. For the multivariate setting, however, there is the following negative result (see [Mairhuber (1956); Curtis (1959)]).

Theorem 1.1 (Mairhuber-Curtis). *If* $\Omega \subset \mathbb{R}^s$, $s \geq 2$, *contains an interior point, then there exist no Haar spaces of continuous functions except for one-dimensional ones.*

In order to understand this theorem we need

Definition 1.1. Let the finite-dimensional linear function space $\mathcal{B} \subseteq C(\Omega)$ have a basis $\{B_1, \ldots, B_N\}$. Then $\mathcal{B}$ is a *Haar space* on Ω if

$$\det A \neq 0$$

for any set of distinct $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N$ in Ω. Here A is the matrix with entries $A_{jk} = B_k(\boldsymbol{x}_j)$.

Note that existence of a Haar space guarantees invertibility of the interpolation matrix A, *i.e.*, existence and uniqueness of an interpolant of the form (1.1) to data specified at $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N$ from the space $\mathcal{B}$. As mentioned above, univariate polynomials of degree $N-1$ form an N-dimensional Haar space for data given at $x_1, \ldots, x_N$.

The Mairhuber-Curtis theorem tells us that if we want to have a well-posed multivariate scattered data interpolation problem we can no longer fix in advance the set of basis functions we plan to use for interpolation of arbitrary scattered data. For example, it is not possible to perform unique interpolation with (multivariate) polynomials of degree N to data given at arbitrary locations in $\mathbb{R}^2$. Instead, the basis should depend on the data locations. We will give a simple example of such an interpolation scheme in the next subsection.

Proof. [of Theorem 1.1] Let $s \geq 2$ and assume that $\mathcal{B}$ is a Haar space with basis $\{B_1, \ldots, B_N\}$ with $N \geq 2$. We need to show that this leads to a contradiction.

We let $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N$ be a set of distinct points in $\Omega \subset \mathbb{R}^s$ and A the matrix with entries $A_{jk} = B_k(\boldsymbol{x}_j)$, $j, k = 1, \ldots, N$. Then, by the definition of a Haar space, we have

$$\det A \neq 0. \tag{1.2}$$

Now, consider a closed path P in Ω connecting only $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$. This is possible since — by assumption — Ω contains an interior point. We can exchange the positions of $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ by moving them continuously along the path P (without interfering with any of the other $\boldsymbol{x}_j$). This means, however, that rows 1 and 2 of the determinant (1.2) have been exchanged, and so the determinant has changed sign.

Since the determinant is a continuous function of $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ we must have had $\det = 0$ at some point along P. This contradicts (1.2). □

1.1.2 *Example: Interpolation with Distance Matrices*

In order to obtain data dependent approximation spaces as suggested by the Mairhuber-Curtis theorem we now consider a simple example. As a "testfunction" we employ the function

$$f_s(\boldsymbol{x}) = 4^s \prod_{d=1}^{s} x_d(1 - x_d), \qquad \boldsymbol{x} = (x_1, \ldots, \boldsymbol{x}_s) \in [0,1]^s.$$

This function is zero on the boundary of the unit cube in $\mathbb{R}^s$ and has a maximum value of one at the center of the cube. A simple MATLAB script defining f_s is given as Program C.1 in Appendix C.

We will use a set of uniformly scattered data sites in the unit cube at which we sample our testfunction f_s. This will be accomplished here (and in many other examples later on) by resorting to the so-called *Halton points*. These are uniformly distributed random points in $(0,1)^s$. A set of 289 Halton points in the unit square in $\mathbb{R}^2$ is shown in Figure 1.1. More details on Halton points are presented in Appendix A. In our computational experiments we generate Halton points using the program `haltonseq.m` written by Daniel Dougherty. This function can be downloaded from the MATLAB Central File Exchange (see [MCFE]).

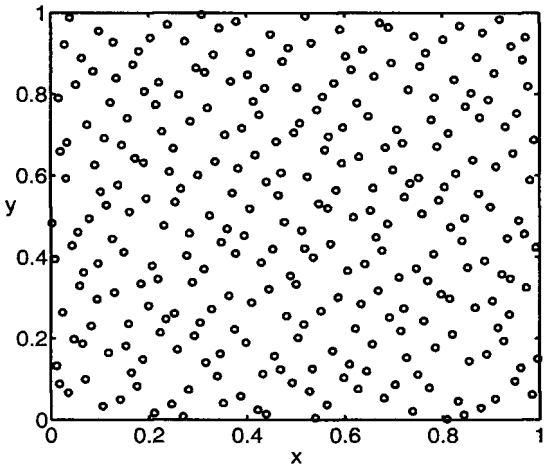


Fig. 1.1 289 Halton points in the unit square in $\mathbb{R}^2$.

As explained in the previous subsection we are interested in constructing a (continuous) function $\mathcal{P}_f$ that interpolates the samples obtained from f_s at the set of Halton points, *i.e.*, such that

$$\mathcal{P}_f(\boldsymbol{x}_j) = f_s(\boldsymbol{x}_j), \qquad \boldsymbol{x}_j \text{ a Halton point.}$$

As pointed out above, if $s = 1$, then this problem is often solved using univariate polynomials or splines. For a small number of data sites polynomials may work satisfactorily. However, if the number of points increases, *i.e.*, the polynomial degree grows, then it is well known that one should use splines (or piecewise polynomials) to avoid oscillations. The simplest solution is to use a continuous piecewise linear spline, *i.e.*, to "connect the dots". It is also well known that one possible basis for the space of piecewise linear splines interpolating data at a given set of points in $[0,1]$ consists of the shifts of the absolute value function to the data sites. In other words, we can construct the piecewise linear spline interpolant by assuming $\mathcal{P}_f$ is of the form

$$\mathcal{P}_f(x) = \sum_{k=1}^{N} c_k |x - x_k|, \qquad x \in [0,1],$$

and then determine the coefficients c_k by satisfying the interpolation conditions

$$\mathcal{P}_f(x_j) = f_1(x_j), \qquad j = 1, \ldots, N.$$

Clearly, the basis functions $B_k = |\cdot - x_k|$ are dependent on the data sites as suggested by the Mairhuber-Curtis theorem. The points x_k to which the basic function $B(x) = |x|$ is shifted are usually referred to as *centers*. While there may be circumstances that suggest choosing these centers different from the data sites one generally picks the centers to coincide with the data sites. This simplifies the analysis of the method, and is sufficient for many applications. Since the functions B_k are (radially) symmetric about their centers x_k this constitutes the first example of *radial basis functions*. We will formally introduce the notion of a radial function in the next chapter.

Of course, one can imagine many other ways to construct an N-dimensional data-dependent basis for the purpose of scattered data interpolation. However, the use of shifts of one single basic function makes the radial basis function approach particularly elegant.

Note that we distinguish between *basis* functions B_k and the *basic* function B. We use this terminology to emphasize that there is one basic function B which generates the basis via shifts to the various centers.

Coming back to the scattered data problem, we find the coefficients c_k by solving the linear system

$$\begin{bmatrix} |x_1 - x_1| & |x_1 - x_2| & \ldots & |x_1 - x_N| \\ |x_2 - x_1| & |x_2 - x_2| & \ldots & |x_2 - x_N| \\ \vdots & \vdots & \ddots & \vdots \\ |x_N - x_1| & |x_N - x_2| & \ldots & |x_N - x_N| \end{bmatrix} \begin{bmatrix} c_1 \\ c_2 \\ \vdots \\ c_N \end{bmatrix} = \begin{bmatrix} f_1(x_1) \\ f_1(x_2) \\ \vdots \\ f_1(x_N) \end{bmatrix}. \tag{1.3}$$

As mentioned earlier, for higher space dimensions s such a data dependent basis is required. Thus, even though the construction of piecewise linear splines in higher space dimensions is a different one (they are closely associated with an underlying computational mesh), the idea just presented suggests a very simple generalization of univariate piecewise linear splines that works for any space dimension.

The matrix in (1.3) above is an example of a *distance matrix*. Such matrices have been studied in geometry and analysis in the context of isometric embeddings of metric spaces for a long time (see, *e.g.*, [Baxter (1991); Blumenthal (1938); Bochner (1941); Micchelli (1986); Schoenberg (1938a); Wells and Williams (1975)] and also Chapter 10). It is known that the distance matrix based on the Euclidean distance between a set of distinct points in $\mathbb{R}^s$ is always non-singular (see Section 9.3 for more details). Therefore, we can solve the scattered data interpolation problem we posed on $[0,1]^s$ by assuming

$$\mathcal{P}_f(\boldsymbol{x}) = \sum_{k=1}^{N} c_k \|\boldsymbol{x} - \boldsymbol{x}_k\|_2, \qquad \boldsymbol{x} \in [0,1]^s, \tag{1.4}$$