Applied Mathematics and Computation 265 (2015) 91–102

Contents lists available at ScienceDirect

Applied Mathematics and Computation

journal homepage: www.elsevier.com/locate/amc

Implicit local radial basis function interpolations based on function values

Guangming Yao [a,*], Jia Duo [b], C.S. Chen [c,d], L.H. Shen [e]

[a] Department of Mathematics, Clarkson University, Potsdam, NY 13699-5815, USA
[b] Department of Mathematics, Heilongjiang Institute of Science and Technology, Heilongjiang, China
[c] Department of Mathematics, University of Southern Mississippi, Hattiesburg, MS, USA
[d] College of Mathematics, Taiyuan University of Technology, Taiyuan, China
[e] Department of Civil Engineering, National Taiwan University, Taiwan

ARTICLE INFO

Keywords:
RBFs
Interpolation
Large-scale
Sparse matrix

ABSTRACT

In this paper we propose two fast localized radial basis function fitting algorithms for solving large-scale scattered data interpolation problems. For each given point in the given data set, a local influence domain containing a small number of nearest neighboring points is established and a global interpolation is performed within this restricted domain. A sparse matrix is formulated based on the global interpolation in these local influence domains. The proposed methods have achieved both low computational cost and minimal memory storage. In comparison with the compactly supported radial basis functions, the proposed fitting algorithms are highly accurate. The numerical examples have provided strong evidence that the two proposed algorithms are indeed highly efficient and accurate. In the two proposed algorithms, we have successfully solved a large-scale interpolation problem with 225,000 interpolation points in two dimensional space.

1. Introduction

During the past two decades, radial basis functions (RBFs) have undergone intensive research and achieved enormous success in various areas of science and engineering such as multivariate data interpolation and approximation, surface reconstruction, computer graphics, numerical solutions of partial differential equations, neural networks, machine learning, etc. The initial development of RBFs was focused on multivariate data interpolation. In 1982, Franke [1] published a review paper evaluating virtually all of the interpolation methods for scattered data sets available at that time. As a result, RBFs have attracted great attention as an effective tool for scattered data interpolation problems. One of the attractive features of RBFs is the simplicity of handling high dimensional scattered data.

Despite the many attractive features of RBFs, most of the commonly used RBFs are globally supported which means the resulting matrix for data reconstruction is dense and could be highly ill-conditioned. For large-scale problems, this implies high computational costs and possible stability issues. For instance, using the general direct solver to fit an RBF with N centers requires $O(N^3)$ flops and $O(N^2)$ storage. These computational costs are not feasible when N is sufficiently large. In many real life applications, the data set can easily go beyond 10,000 which is difficult for RBFs to handle in a reasonable way. In the past, the interpolation of data associated with large-scale problems was typically achieved by decomposing the problem into sub-domains

* Corresponding author. Tel.: +1 315 268 6496.
E-mail address: guangmingyao@gmail.com, gyao@clarkson.edu (G. Yao).

http://dx.doi.org/10.1016/j.amc.2015.04.107

Table 1
Commonly used radial basis functions.

Gaussian:	$\varphi(r) = e^{-cr^2}$, $c > 0$
Inverse multiquadrics:	$\varphi(r) = (r^2 + c^2)^{-1/2}$, $c > 0$
Matern function:	$\varphi(r) = (cr)^n K_n(cr)$, $c > 0$, where K_n is the spherical Bessel function, $n > 0$
Multiquadrics:	$\varphi(r) = (r^2 + c^2)^{1/2}$, $c > 0$
Normalized multiquadrics:	$\varphi(r) = (1 + (cr)^2)^{1/2}$, $c > 0$
Thin-plate spline:	$r^2 \ln(r)$
Polyharmonic:	$r^{2n} \ln(r)$

on which local interpolation problems are solved, often with additional constraints across the domain boundaries so as to assure continuity of the function and perhaps continuity of its derivatives as well [1–3]. One disadvantage of the domain decomposition technique is the requirement of domain discretization which is often very tedious. For high dimensional problems, such an approach is non-trivial. Compactly supported radial basis functions (CS-RBFs) [4] were developed in the mid-1990's to alleviate these difficulties. The use of CS-RBF interpolation generates a sparse interpolation matrix which is desirable for interpolating large-scale problems. However, the convergence rate of CS-RBFs is relatively slow, and it is difficult to achieve high accuracy.

We can observe that there are a number of fast fitting algorithms and fast computational methods available for tackling the large-scale interpolation problems. We refer readers to references [5–8].

Our purpose in this paper is to propose two localized interpolation techniques to reconstruct surfaces based on the given interpolation points. Such localized methods are different from CS-RBFs. In our approach, global RBFs are used on each local influence domain. A global interpolation is performed on each influence domain and then a sparse matrix is formulated to link all of the influence domains together. The RBFs with strong convergence rates such as Gaussian, multiquadrics (MQ), normalized MQ, inverse MQ, and Matern functions are used as the basis functions. The sub-optimal shape parameter of these RBFs can be obtained by using the technique of the so called leave one out cross validation (LOOCV) [9,10]. For every given data point, we need to search for neighboring evaluation points to form the influence domain so that the functions' values at the evaluation points can be approximated. Our proposed approaches have the features of fast computation, small memory storage, and high accuracy for large-scale fitting problems. The highly efficient kd-tree algorithm [11] is used in the local method to search for nearest neighbors of the evaluation points among the interpolation points.

2. Radial basis functions interpolation

RBFs have been applied to solve many problems in science and engineering. Among these problems, scattered data interpolation is one of the earliest topics for applications of RBFs. Let $\Omega \subset \mathbf{R}^n$, and $\mathbf{x}_i \in \Omega$. We are given $(\mathbf{x}_i, f(\mathbf{x}_i)) \equiv (\mathbf{x}_i, f_i)$ as known coordinate pairs. Let $\{\mathbf{s}_j\}_{j=1}^N \subset \Omega$ denote the evaluation or test points at which we seek to interpolate f by $\hat{f}$, such that $\hat{f}(\mathbf{s}_j) \approx f(\mathbf{s}_j)$, $j = 1, \ldots, N$, with $\hat{f}(\mathbf{s}_k) = f(\mathbf{s}_k)$ whenever $\mathbf{s}_k = \mathbf{x}_k$.

The interpolation of the multivariate function f can be constructed by linear combinations of univariate interpolation functions with Euclidean norm $\|\cdot\|$. Popular invariant functions include the radial basis functions whose value at any point $\mathbf{x} \in \mathbf{R}^n$ depends only on the distance from the fixed point $\mathbf{c}$ and can be written as

$$\phi(\mathbf{x}) = \phi(\|\mathbf{x} - \mathbf{c}\|),$$

where $\mathbf{c}$ is the center of the radial basis function ϕ. Some commonly used RBFs are listed in Table 1.

The interpolants $\hat{f}$ to function f can be obtained by

$$\hat{f}(\mathbf{x}) = \sum_{j=1}^{N} \alpha_j \phi(\|\mathbf{x} - \mathbf{x}_j\|), \tag{1}$$

in which all of the interpolation points $\mathbf{x}_i$, $i = 1, 2, \ldots, N$ are chosen as the centers of the basis functions, and α_i, $i = 1, 2, \ldots, N$ are unknown real coefficients, which can be obtained by interpolation:

$$f(\mathbf{x}_i) = \sum_{j=1}^{N} \alpha_j \phi(\|\mathbf{x}_i - \mathbf{x}_j\|), \quad i = 1, 2, \ldots, N. \tag{2}$$

The resulting system of linear equations can be written in a matrix/vector format

$$A\alpha = \mathbf{f},$$

where $\boldsymbol{\alpha} = [\alpha_1, \alpha_2, \cdots, \alpha_N]^T$, $\mathbf{f} = [f_1, f_2, \cdots, f_N]^T$, and

$$A = \begin{bmatrix} \phi(\|\mathbf{x}_1 - \mathbf{x}_1\|) & \cdots & \phi(\|\mathbf{x}_1 - \mathbf{x}_n\|) \\ \vdots & \cdots & \vdots \\ \phi(\|\mathbf{x}_n - \mathbf{x}_1\|) & \cdots & \phi(\|\mathbf{x}_n - \mathbf{x}_n\|) \end{bmatrix}, \tag{3}$$