Mathematical Modelling and Analysis 2005. Pages 467–472
Proceedings of the 10th International Conference MMA2005&CMAM2, Trakai
 ISBN 9986-05-924-0

MULTISCALE MODIFICATION OF SHEPARD'S METHOD FOR MULTIVARIATE INTERPOLATION OF SCATTERED DATA

A.V. MASJUKOV and V.V. MASJUKOV

Tver State University

Zheljabov 33, Tver, 170000, Russia

E-mail: tvergw@slavneft.ru

Abstract. We present a new method of constructing a smooth function of many variables that interpolates data values at arbitrary distributed points. Shepard's method for scattered data interpolation and its quadratic modifications have the advantage of an easy generalization to more than two independent variables. We describe a modified Shepard's method that, without losing the advantage, is self-adjusting to trends of different scales in interpolation data. In the case of interpolation to a uniform mesh (from irregular interpolation nodes) computational complexity of the method is shown to be logarithmic with respect to the overall number of mesh point, independently on the space dimension and the number of interpolation nodes. Accuracy test results for three independent variables are presented.

Key words: Multivariate interpolation, scattered data interpolation, global interpolation, smooth interpolation, surface fitting, multiscale methods

1. INTRODUCTION

The scattered data fitting problem is encountered frequently in applied mathematics. In this paper we address the problem of constructing a function $u \in C^1(\mathbf{R}^m)$ such that, given a set of scattered nodes $\mathbf{x_i} \in \mathbf{R}^m$ with associated values u_i, $u(\mathbf{x_i}) = u_i$ for $i = 1, \ldots, n$.

A quadratic modification of Shepard's method [6] was introduced by Franke and Nielson [2], its further modification was proposed by Renka [5]. It was also shown [5] that Quadratic Shepard is more accurate than triangle-based methods. The primary advantage of Quadratic Shepard is in fitting with a function of three and more variables. However, Quadratic Shepard is a local method, in which an interpolated value is not influenced by all of the data. Therefore, quadratic modifications of Shepard's method require some

artificial setup parameters like a radius of influence or a number of nearest nodes, but there is no algorithm to assign these parameters optimally.

In Section 2 we describe an iterative modification of Shepard's method, referred to further as Iterative Shepard, that is free from artificial setup parameters. Accuracy of the method is proved by test results presented in Section 3. In Section 4 we introduce an efficient numerical implementation of Iterative Shepard in the particular case of interpolation to a uniform mesh (from arbitrary distributed nodes). Section 5 concludes the paper.

Iterations of the method, which we are going to describe, correspond to successive scaling. Therefore, Iterative Shepard is self-adjusting to trends of different scales in interpolation data. In contrast to the multiscale methods presented in [1] and [3], Iterative Shepard doesn't perform successive tessellations of an interpolation domain and compute approximating splines on these tessellations. Thus we preserve the simplicity of the original Shepard's method. Iterative Shepard is a further development of our recent paper [4].

2. Interpolation Algorithm

Interpolating function for given values u_i at nodes $\mathbf{x_i}$ we define by

$$u(\mathbf{x}) = \sum_{k=0}^{K} \sum_{j=1}^{n} \left[u_j^{(k)} \, f\left((\mathbf{x} - \mathbf{x_j})/\tau_k\right) \Big/ \sum_{p=1}^{n} f\left((\mathbf{x_p} - \mathbf{x_j})/\tau_k\right) \right], \qquad (2.1)$$

where $\mathbf{x} \in \mathbf{R}^m$, $u_j^{(k)}$ are the interpolation residuals at k-th step, $u_j^{(0)} \equiv u_j$,

$$u_j^{(k+1)} = u_j^{(k)} - \sum_{q=1}^{n} \left[u_q^{(k)} \, f\left((\mathbf{x_j} - \mathbf{x_q})/\tau_k\right) \Big/ \sum_{p=1}^{n} f\left((\mathbf{x_p} - \mathbf{x_q})/\tau_k\right) \right], \quad (2.2)$$

the weight function f is continuously differentiable,

$$f(\mathbf{x}) \geq 0 \quad \forall\, \mathbf{x}, \qquad f(\mathbf{0}) > 0, \qquad f(\mathbf{x}) = 0 \quad \text{if} \quad \|\mathbf{x}\| > 1, \qquad (2.3)$$

and the scale factor $\tau_k > 0$ decreases from a value τ_0 (that is the first setup parameter of the method) down to

$$\tau_K < r \equiv \min_{i \neq j} \|\mathbf{x_i} - \mathbf{x_j}\|. \qquad (2.4)$$

From (2.4) and (2.3) it is easily seen that $u(\mathbf{x})$ satisfies the interpolation conditions. In contrast to other modifications of Shepard's method, current point $\mathbf{x}$ does not appear in the denominators of the relative weights of nodes in (2.1). Thus, Iterative Shepard can produce piecewise polynomial interpolation. In the case of one independent variable ($m = 1$) the following weight function may be used

$$f(x) = \begin{cases} 5(1-|x|)^4 - 4(1-|x|)^5, & |x| < 1, \\ 0, & |x| \geq 1. \end{cases} \qquad (2.5)$$

In the case of three variables ($m = 3$) it corresponds to

$$f(x, y, z) = f(x)f(y)f(z). \tag{2.6}$$

This choice of the weight function makes the interpolant (2.1) doubly continuously differentiable and its third partial derivatives have discontinuities at the interpolation nodes only.

The described algorithm performs a global interpolation if

$$\tau_0 > R \equiv \sup_{\mathbf{x}\in D} \max_{0\le j\le n} \|\mathbf{x} - \mathbf{x_j}\|, \tag{2.7}$$

where D is the interpolation domain. From (2.2) it can be shown that

$$\sum_{j=1}^{n} u_j^{(k)} = 0, \quad k > 0.$$

Therefore if $\tau_0 \gg R$ then in the first step of the algorithm ($k = 0$) we merely subtract the average from the given interpolation values. Further increasing of the setup parameter makes the dependence of the interpolant on τ_0 very weak, since $f\left((\mathbf{x} - \mathbf{x_j})/\tau_k\right)$ comes to $f(\mathbf{0})$, for all j and $\mathbf{x}$, if $\tau_k \to \infty$. If we assign τ_0 according to the largest distance to the nearest node

$$\tau_0 > R_1 \equiv \max\left\{ \max_{0\le i\le n} \min_{0\le j\le n} \|\mathbf{x_i} - \mathbf{x_j}\|, \quad \sup_{\mathbf{x}\in D} \min_{0\le j\le n} \|\mathbf{x} - \mathbf{x_j}\| \right\}, \tag{2.8}$$

then Iterative Shepard becomes a local method. This makes it less accurate but faster ($R_1 < R$). The algorithm is defined completely by a choice of the sequence $\{\tau_k\}$ of scale factors. If we define

$$\tau_k = \tau_0\, \gamma^k, \qquad 0 < \gamma < 1, \tag{2.9}$$

then the only one additional setup parameter γ appears, which determines the rate of scaling. It will be shown in the next section that the method becomes more accurate (interpolant becomes more smooth) while γ is increasing. However for sparse interpolation nodes smaller values of γ may be used in order to decrease the computation time.

Thus Iterative Shepard has two setup parameters, which are not artificial. A reasonable choice of the values of τ_0 and γ can be done based on the distribution of the nodes and an assumption of smoothness of the function to be interpolated. As a rule, if τ_0 is assigned from (2.7) or (2.8) and γ is varied from 0.6 to 0.95, the behaviour of the interpolation function is not changed significantly. This happens because the idea of the multiscale analysis is applied. Since at first iterations large-scale (low-frequency) trends in interpolation data are accounted for and more high-frequency components are residuals for further steps, the method is self-adjusting to data at any reasonable values of the setup parameters.

3. Test Results

Here we compare interpolation errors of the proposed method with errors of Quadratic Shepard in the test suggested by Renka [5]. Following Renka, a uniform random-number generator was used to produce 216 nodes in the unit cube, and the set of interpolation points was taken to be the 20-by-20-by-20 uniform mesh in the unit cube. Renka suggested six functions, which were used to compute interpolation values at random nodes and interpolation errors at the mesh points. We have rejected one of the functions (the sixth) because its first derivatives have singularities just near each vertex of the unit cube and because a global method of smooth interpolation should have greater errors than a local method in this case.

Table 1 displays interpolation errors of Iterative Shepard (referred as IS) in comparison with Quadratic Shepard (referred as QS). QS errors were taken from the third table in paper [5]. We averaged IS errors over 100 random distributions of interpolation nodes. The weight function was taken as defined by (2.5) and (2.6), $\tau_0 = 4$; the results are presented for three values of γ (0.9, 0.95, 0.99). The columns of the table correspond to Renka's functions F1,..,F5. For convenience, the quotients of IS and QS errors are shown and they are highlighted at $\gamma = 0.99$.

Table 1. Interpolation Errors for a Trivariate 216-Node Test.

Function	F1		F2		F3		F4		F5	
Method	Mean interpolation errors									
QS	.01077	1.00	.00662	1.00	.00614	1.00	.00208	1.00	.00247	1.00
IS(0.9)	.01094	1.01	.00763	1.15	.01120	1.82	.00374	1.80	.00284	1.15
IS(0.95)	.00960	0.90	.00712	1.07	.00843	1.37	.00190	0.91	.00256	1.04
IS(0.99)	.00839	**0.78**	.00723	**1.09**	.00547	**0.89**	.00101	**0.48**	.00198	**0.80**
Method	Maximum interpolation errors									
QS	.2085	1.00	.1476	1.00	.1193	1.00	.0308	1.00	.0463	1.00
IS(0.9)	.1671	0.80	.1264	0.86	.2091	1.75	.0369	1.20	.0608	1.31
IS(0.95)	.1645	0.79	.1184	0.80	.1796	1.51	.0201	0.65	.0459	0.99
IS(0.99)	.1456	**0.70**	.1159	**0.78**	.1196	**1.00**	.0222	**0.72**	.0316	**0.68**

The presented results show that IS accuracy becomes better than QS accuracy while γ is increasing. It should be noted that, although the test functions were chosen by Renka to exhibit a variety of behaviour, they are all smooth well-behaved functions and therefore fail to reflect the erratic nature of many data sets that arise in practice. Data sets encountered in practice may also exhibit much more variation in sparseness of the nodal distribution than node sets used here. Moreover, for very sparse nodes the uncertainty of interpolation (extrapolation) increases and the conception of accuracy becomes meaningless. Our practice of using Iterative Shepard in geophysical predictions made

us sure that $\gamma \approx 0.75$ gives the most credible results, in comparison with other state-of-the-art methods.

4. Efficient Implementation

Frequently interpolation should be performed from scattered nodes to a uniform mesh. Computation time becomes an important criterion in cases of large meshes and large interpolation data sets. Here we describe an efficient numerical implementation arising from a special choice of the weight function of Iterative Shepard.

For a mesh function $a = \{a_i,\, i = 1, \ldots, r\}$, given on a one-dimensional uniform mesh, we define the following operator s_τ such as $s_\tau[a] = b$ if

$$\left(1 + 2\tau^2\right) b_i - \tau^2 \left(b_{i-1} + b_{i+1}\right) = a_i, \quad i = 1, \ldots, r, \quad b_0 = b_1,\, b_{r+1} = b_r. \tag{4.1}$$

It can be shown that operator (4.1) is a difference approximation of the convolution with the filter function

$$(2\tau)^{-1} \exp\left(-|x|/\tau\right).$$

Thus we obtain a scalable filter: its computation time does not depend on the scale factor τ. Operator s_τ^2, which denotes double using of scheme (4.1), approximates the continuously differentiable filter

$$(4\tau)^{-1} \exp\left(-|x|/\tau\right)(1 + |x|/\tau).$$

For multi-dimensional meshes we note that s_τ implies using finite-difference scheme (4.1) in each independent variable, when all remaining variables are fixed. Hence in m variables s_τ^2 approximates the convolution with the function

$$f_\tau(\mathbf{x}) = (4\tau)^{-m} \exp\left(-\tau^{-m} \sum_{l=1}^{m} |x_l|\right) \prod_{l=1}^{m} \left(1 + \frac{|x_l|}{\tau}\right), \tag{4.2}$$

where $x_1, \ldots, x_m$ are coordinates of $\mathbf{x}$.

The described algorithm for computing the convolution (multivariate and scalable) with function (4.2) can be used in Iterative Shepard as a special choice of the weight function

$$f(\mathbf{x}) = \exp\left(-\sum_{l=1}^{m} |x_l|\right) \prod_{l=1}^{m} \left(1 + |x_l|\right). \tag{4.3}$$

It should be noted that function (4.3) does not meet the last of conditions (2.3). In this case the interpolant of Iterative Shepard is represented by an infinite series. Theorems of its convergence and differentiability can be proved similarly to the same theorems of our another method [4].

Finally, we rewrite Iterative Shepard for a uniform mesh in the form

$$u(\mathbf{x}) = \sum_{k=0}^{\infty} w^{(k)}(\mathbf{x}),\ w^{(k)}(\mathbf{x}) = S_k\left[g_1^{(k)}\right](\mathbf{x}),\ g_0(\mathbf{x}) = \sum_{j=0}^{n} \delta(\mathbf{x}-\mathbf{x_j}),$$
$$g_1^{(k)}(\mathbf{x}) = \sum_{j=1}^{n} \frac{u_j^{(k)}}{S_k\left[g_0(\mathbf{x})\right]|_{\mathbf{x}=\mathbf{x_j}}} \delta(\mathbf{x}-\mathbf{x_j}), \quad u_j^{(k+1)} = u_j^{(k)} - w^{(k)}(\mathbf{x_j}), \tag{4.4}$$

where interpolant $u(\mathbf{x})$ is a mesh function, $S_k = s_\tau^2$ at $\tau = \tau_k$, scale factor sequence is defined by (2.9), $\delta(\mathbf{x}) = 0$ if $\mathbf{x} \neq \mathbf{0}$, $\delta(\mathbf{0}) = 1$, in assumption that each interpolation node coincides with some mesh point (fine mesh).

While applying operator S_k at the uniform mesh in the m-dimensional cube with N mesh points, system (4.2) is solved $2N^{1-1/m}$ times. Therefore S_k requires $10mN$ multiplications. If the scale factor is decreased from the cube edge down to the mesh step, then we must perform $K = \lceil \log_{1/\gamma} N^{1/m} \rceil$ iterations. Note that the denominators in (4.4) are computed at interpolation nodes only. Nevertheless, if they are computed similarly (not directly by summing over nodes), the total multiplication count is approximately

$$20N \log_{1/\gamma} N \approx 20\gamma(1-\gamma)^{-1} N \ln N.$$

Remarkably, the computation time does not depend on n or m. The storage requirement is that a copy of the mesh is stored.

5. Concluding Remarks

The accuracy of our method was shown in a trivariate test. An efficient numerical implementation for interpolation to a uniform mesh was constructed, the computation time of which is $O(N \ln N)$, where N is an overall number of mesh points, regardless of the space dimension and of the number of interpolation nodes. Our multiscale method, by its nature, adjusts itself to data. The method described here has an obvious generalization to a smoothing method, which is more appropriate when the data values are inaccurate. It is only necessary to stop iterations before the interpolation conditions are satisfied.

References

[1] E. Arge, M. Dhlen and A. Tveito. Approximation of scattered data using smooth grid function. *J. Comput. Appl. Math.*, **59**, 191 – 205, 1995.

[2] R. Franke and G. Nielson. Smooth interpolation of large sets of scattered data. *Int. J. Numer. Metods Eng.*, **15**, 1691 – 1704, 1980.

[3] S. Lee, G. Wolberg and S.Y. Shin. Scattered data interpolation with multilevel b-splines. *IEEE Trans. on Visualization and Computer Graphics*, **3**, 228 – 244, 1997.

[4] A. Masjukov and V. Masjukov. Interpolation technique based on the iterative scalable smoothing. *Matematicheskoe modelirovanie*, **17**, 46 – 56, 2005. (in Russian)

[5] R. Renka. Multivariate interpolation of large sets of scattered data. *ACM Trans. on Mat. Software*, **14**, 139 – 148, 1988.

[6] D. Shepard. A two dimensional interpolation function for irregularly-spaced data. *ACM National Conference*, 517 – 524, 1968.