An Evaluation of Three Methods for Arbitrarily Located Data Modeling

Marius Dorian Zaharia
Computers Department, University POLITEHNICA Bucharest
E-mail: zaharia@cs.pub.ro

Abstract: The paper describes three methods to model scalar scattered valued functions defined over 3D domains and specified through samples forming 3D unstructured data sets. These methods are based on interpolation/approximation techniques. They are suitable to be used in scientific visualization applications, including those in the field of economic sciences. The implementation and evaluation of those methods (from the point of view of algorithm efficiency and approximation errors) are also emphasized.

Keywords: data modeling, scattered data, scientific visualization, least square method.

1. Introduction

A lot of application fields are processing scattered data. These categories of spatially distributed data usually result from experimental measurements. The sampling process follows an unstructured grid over a tri-dimensional domain or over a surface of the 3D space. These data could equally result from scientific applications, modeling applications or simulations. The scattered data are characteristic to economic applications as well, for example the spatial distribution map of the total income (or the distribution of unemployment levels) across different regions of a country is a scattered data collection.

Typically, a set of k scattered data points $\{(x_{i1}, x_{i2}, \ldots, x_{in}, f_i) \mid 1 \leq i \leq k\}$ could be considered to describe a functional dependence:

$$F(x_{i1}, x_{i2}, \ldots, x_{in}) = f_i$$

where $x_{i1}, x_{i2}, \ldots, x_{in}$ are independent variables and f is the dependent variable. Usually n=2 in case of bivariate data or n=3 in case of trivariate data.

Sometimes it is useful to determine a function that approximates the irregularly distributed data samples. In computer graphics applications, ([2]) various interpolation or approximation functions (Bezier approximation surfaces or different types of spline surfaces (B-splines, NURBS)) have the shape completely defined by a small number of control points and satisfy in the same time some continuity constraints. Other approximation methods of irregularly distributed data are based on quadratic functions, some of these methods are described in [3] and [5].

The present paper describes and evaluates the implementation of three methods for scattered data modeling. These methods are particularly useful in scientific visualization applications, in order to render the correlation between the scattered data points. The evaluations were made in case of bivariate data sets containing 3D points (x_i, y_i, z_i) where $z_i = f(x_i, y_i)$, $1 \le i \le k$. The unknown function f will be approximated by a polynomial function Q(x, y). The methods could be easily generalized in case of trivariate data.

2. Description of the methods

The simplest method is due to *Shepard* [5]. Given the data points $P_1, P_2, \ldots, P_k$ and the corresponding values f_i ($1 \le i \le k$) and P an arbitrary point in the definition domain then:

$$Q(P) = \frac{\sum_{i=1}^{k} \frac{f_i}{\left\| P - P_i \right\|^2}}{\sum_{i=1}^{k} \frac{1}{\left\| P - P_i \right\|^2}}, \qquad for \qquad P \notin \{P_1, \ldots, P_k\}$$

and $Q(P_i)=F_i$, for $1 \le i \le k$

The notation $|P-P_i|$ corresponds to the Euclidean distance between P and P_i.

The second method is described by Nielson ([5]) who called it the *modified quadratic Shepard method (MQS)*. The principle is that every data point $P_i(x_i, y_i)$ has an associated basis function $B_i(x,y)$. This function depends directly on the Euclidean distance between P(x,y) and $P_i(x_i,y_i)$. The global approximation is a linear combination of the basis functions. If k is the number of data points then:

$$Q(x, y) = \sum_{i=1}^{k} h_i B_i(x, y)$$

The basis functions are quadratic polynomials:

$$B_i(x, y) = a_{i0}(x - x_i)^2 + a_{i1}(y - y_i)^2 + a_{i2}(x - x_i)(y - y_i) + a_{i3}(x - x_i) + a_{i4}(y - y_i) + f_i$$

The coefficients a_{ik} ($0 \le k \le 4$) are obtained applying the least square method i.e. through the minimization of the functional:

$$E = \sum_{\substack{j=1 \\ j \neq i}}^{k} (B_i(x_j, y_j) - f_j)^2$$

This is done by solving the 5*5 linear system:

$$\frac{\delta E}{\delta a_i} = 0, \qquad for \qquad 0 \leq i \leq 4$$

by direct Gauss elimination.

The global approximation function is given by:

$$Q(P) = \frac{\sum_{i=1}^{k} B_i(P) W_i(P)}{\sum_{i=1}^{k} W_i(P)}$$

The weight $W_i(P)$ depends on the "radius of action" of the basis function $B_i(x,y)$ and is given by:

$$W_i(P) = \frac{\rho(R_w, \|P - P_i\|)}{R_w \|P - P_i\|}$$

where:

$$\rho(a,b) = \begin{cases} 0 & if \quad b > a \\ b - a & if \quad b \leq a \end{cases}$$

that is the weight is non null if and only if P and P_i are placed at a distance smaller than R_w.

The lastly implemented method called *the multiquadric method* looks after a modeling function having the form:

$$Q(P) = \sum_{i=1}^{k} C_i \sqrt{R^2 + \|P - P_i\|^2}$$

where the coefficients C_i are found as the solution of the linear system:

$$A*[C_1, C_2, \ldots C_k]^T = [f_1, f_2, \ldots, f_k]^T$$

The k order square matrix A has the elements:

$$A_{ij} = \sqrt{R^2 + \|P_i - P_j\|^2}$$

and R is a user defined parameter (0 in the present implementation).

3. Results evaluation

For evaluation it was considered the case of the "gaussian hill" a surface patch given by its explicit equation:

$$z = G(x, y) = 4.0 * e^{-\frac{x^2+y^2}{2.0}}, \qquad where \qquad -4.0 \le x, y \le 4.0 \qquad (1)$$

The parameters that were considered to influence the evaluation process are:

npct = the number of datapoints

ntots = the total number of points computed on the surface. Every such point is a vertex of the polygonal mesh that was rendered instead of the approximating surface.

ampp = the amplitude of the perturbation. The coordinates of every data point are generated starting from a point P placed precisely on the gaussian surface (1) and changing the coordinates of P with small amounts a, where |a|<ampp.

Table 1 shows the total z-deviation (measured on z axis) of the three approximating surfaces relative to the exact gaussian (ampp was considered 0) the total z-deviation per computed point and the running time per computed point (tpp). The test case had npct=16 and ntots=6400.

	total z_dev	z_dev per point	time per point (ms)
Shepard	1156.65	0.176	0.23
MQS	841.006	0.12	3.205
Multi_quad	477.99	0.072	0.034

Table 1

Table 2 shows the total z-deviation per computed point, as function of the perturbation amplitude. In that case npct=256 and ntots=6400.

ampp	0.0	0.2	0.4	0.6	0.8	1.0	2.0
Shepard	0.11	0.19	0.27	0.350	0.442	0.533	0.966
MQS	0.09	0.152	0.243	0.320	0.414	0.509	0.957
Multi_q	0.0032	0.096	0.200	0.295	0.394	0.499	0.962

Table 2

For results visualization, the data points (x,y) are placed on a perturbed npct=n*n rectangular grid, the dependent variable z has also a random variation relative to the value given by (1). The resulted surface is represented by two families of curves, this time the sampling grid has ntots=nc*nc regularly distributed points. The same polygonal mesh is rendered using a Gouraud shading technique. The surface is

illuminated by two greenish spot sources placed near the horizontal plane (z=0) and a third ambient source placed exactly over the highest point of the gaussian. Figure 1 shows a visualization of an approximating MQS surface.

The implementation was done on an Intel Pentium/200MHz processor under a Windows 95 operating system, using Visual C++ environment and Open-GL for results visualization.

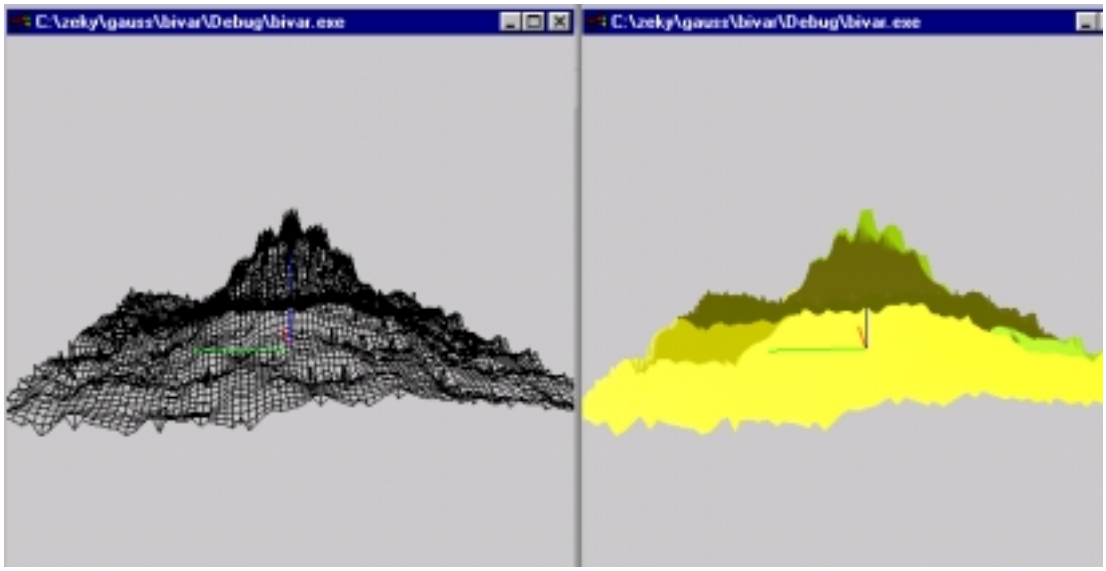

Figure 1 – MQS npct=100, ntots=2500, ampp=0.6

References

[1] I. Dancea, Programarea Calculatoarelor Numerice, pag. 104-107, Editura Dacia, 1973.

[2] G. Farin, Curves and Surfaces for Computer Aided Geometric Design, Academic Press, 1988.

[3] H. Hagen, Visualization of large data sets, in Scientific Visualization Advances and Challenges, pag. 187-198, Academic Press, 1994.

[4] V. Iorga, B. Jora, Programare Numerica, Editura Teora, 1996.

[5] G. M. Nielson, Scattered Data Modeling, IEEE Computer Graphics and Applications, pag. 60-70, January 1993.

[6] W. Schroeder, K. Martin, B. Lorensen, The Visualization Toolkit, an Object Oriented Approach to 3D Graphics, Prentice Hall, 1998.

[7] M. Woo, J. Neider, T. Davis, OpenGL Programming Guide, Silicon Graphics Inc., 1997.