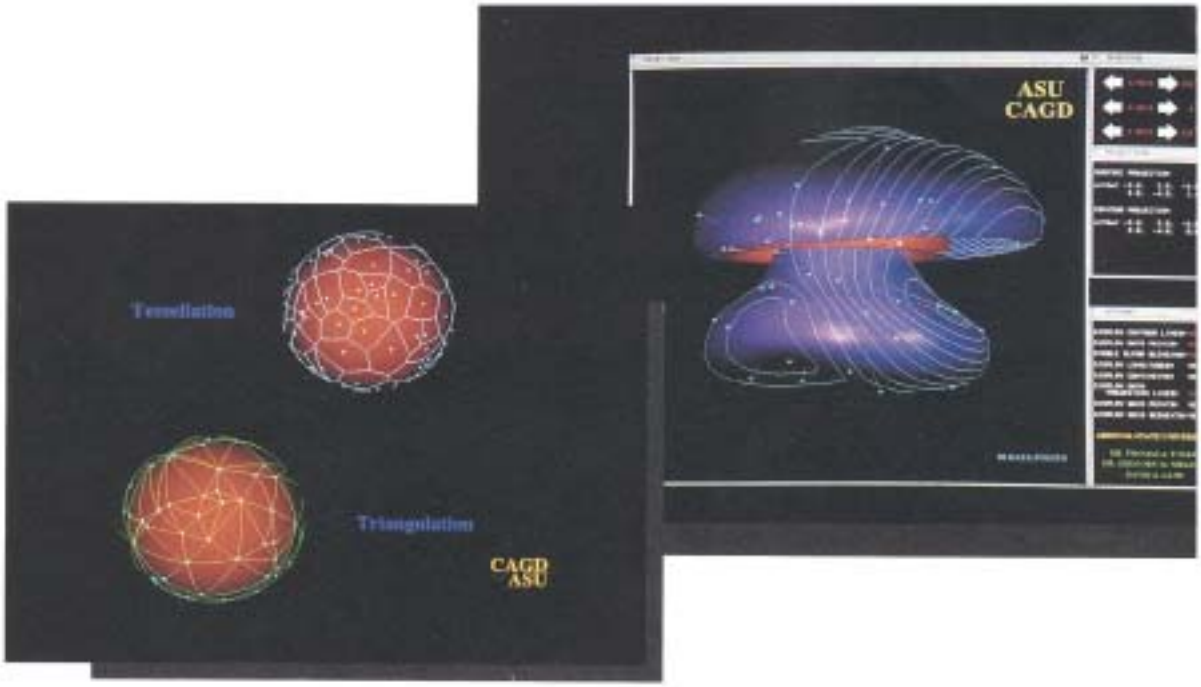
Title
Visualizing and modeling scattered multivariate data

Journal
IEEE Computer Graphics and Applications, 11

Authors
Nielson, G. M.
Foley, T. A.
Hamann, Bernd
et al.

Publication Date
1991

Peer reviewed



Visualizing and Modeling Scattered Multivariate Data

Gregory M. Nielson, Thomas A. Foley,
Bernd Hamann, and David Lane
Arizona State University

Working toward a convenient visualization tool, we've developed mathematical models that let us view scattered volumetric or surface-on-surface data.

In this article we concentrate on visualizing two types of scattered data—volumetric data sampled in a 3D volume and surface-on-surface data sampled on a 3D surface. It would be convenient if scientific data were provided on a uniform grid, but that isn't always the case. Since measurements are often sampled at discrete scattered locations, we developed mathematical models that are defined over the entire domain and interpolate or approximate the given scattered data. We can evaluate the modeling function over a grid, so we can use a conventional "off-the-shelf" visualization tool that applies to data on a uniform grid. We can also compute volumes, gradients, centroids, and other quantities from the model.

When dealing with *volumetric data,* we have a single dependent variable, F, and three independent variables, x, y, and z. We can view the three independent variables as representing a point $p = (x, y, z)$ in a 3D space. In mathematical terms, the modeling problem is to find a trivariate function, $F(p) = F(x, y, z)$, that approximates the relationship implied by a collection of data values, (x_i, y_i, z_i, F_i), $i = 1, \ldots, N$. We make no assumptions about the disposition of the data sites $p_i = (x_i, y_i, z_i)$, $i = 1, \ldots, N$ except that they are distinct.

Sometimes we associate the words "irregular," "unstructured," "arbitrary," and "random" with scattered data. Examples include temperature measurements at various locations in a furnace and mineral concentrations measured at various depths at randomly located well sites.

0272-17-16/91/0500-0047$01.00 ©1991 IEEE

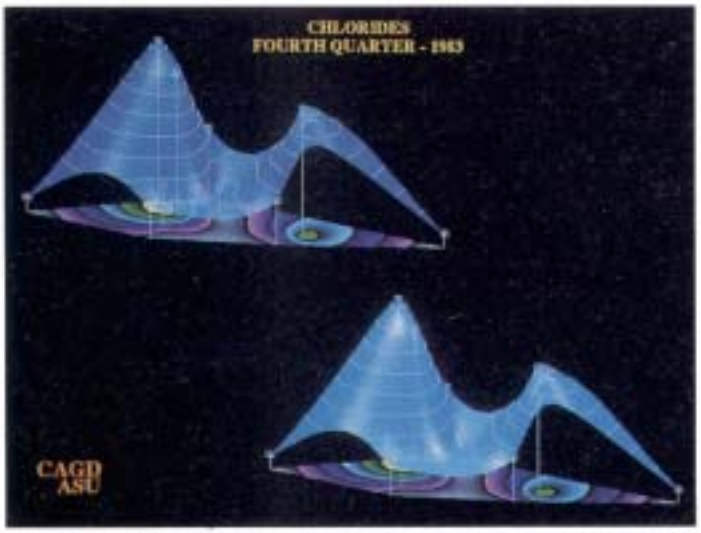


Figure 1. Views of the multiquadric interpolant to sampled data on a planar domain.

Surface-on-surface data is similar to volumetric data, but the data sites are located on one surface in 3D space. The domain of the relationship is the surface denoted by D. In mathematical terms the data (x_i, y_i, z_i, F_i), where $p_i = (x_i, y_i, z_i) \in D$, are given. The problem is to find a function $F(p) = F(x, y, z)$ defined on D that approximates the relationship implied by this data.

Examples of surface-on-surface data include rainfall measurements taken at various locations on the earth or pressure measurements taken on the wing of an airplane. In the first case, the domain is the earth's surface and in the second, the surface of the wing.

Even though these two kinds of data and their problems look very similar in their mathematical descriptions, we solve them by very different methods. We also visualize the various solutions with methods that are quite different.

Modeling volumetric data

To begin, let's discuss briefly the case of bivariate scattered data. Here we have the data (x_i, y_i, F_i), $i = 1, \ldots, N$, with independent data sites arbitrarily located in some planar domain. Scientists have studied the problem of bivariate scattered data quite extensively. Franke and Nielson[1] described many of the methods that solve this problem. One of the most effective (and also one of the simplest methods to implement) is the multiquadric method Hardy introduced.[1] The modeling function for the multiquadric method is

$$F(p) = \sum_{i=1}^{N} \alpha_i \sqrt{\|p - p_i\|^2 + R^2}$$

where $R^2 > 0$, $p = (x, y)$, and $\|p - p_i\|^2 = (x - x_i)^2 + (y - y_i)^2$.

In the case of interpolation, where it is required that $F(p_i) = F_i$, $i = 1, \ldots, N$, we compute the coefficients α_i by solving the $N \times N$ linear system of constraint equations: $A\alpha = F$, where $A = (a_{ij}) = (\sqrt{\|p_j - p_i\|^2 + R^2})$, $\alpha = (\alpha_1, \alpha_2, \ldots, \alpha_N)^t$, and $F = (F_1, F_2, \ldots, F_N)^t$. Figure 1 shows a plot of this interpolant to 10 points, denoted by boxes connected by line segments to the sampling sites in the x-y plane. The optimal choice of the constant R^2 is an open research problem, but Carlson and Foley[2] recently indicated that the optimal choice depends almost entirely on the dependent data values F_i and only a negligible amount on the number or distribution of the independent data sites $p_i = (x_i, y_i)$. In some cases, the dependent data indicate a smoothly varying relationship. For this situation, a choice of this arbitrary constant is $R^2 = cV$, where c is a constant in the range 0.1 to 1.0 and V is the area of the bounding rectangle for the independent data. For rapidly varying F_i, the parameter R^2 should be relatively small, for example $c = 10^{-4}$ or 10^{-6}. The parameter R^2 should also be small if the number of data points is large, because otherwise, the linear system of equations might be ill-conditioned.

The multiquadric method easily generalizes to multivariate data because it depends only on Euclidean distance, which extends naturally to higher dimensions. Figure 2 shows an example of the 3D multiquadric method. We chose the independent data sites $p_i = (x_i, y_i, z_i)$ at random in a unit cube and took the dependent values $F_i = G(p_i)$ from a given function $G(p)$ whose formula appears in Foley and Lane's work.[3] By using an underlying function to produce the dependent data, we can see how effectively the method reproduces certain shapes and behaviors. The top graph in Figure 2 is the actual test function G, the bottom left is the multiquadric approximation based on 100

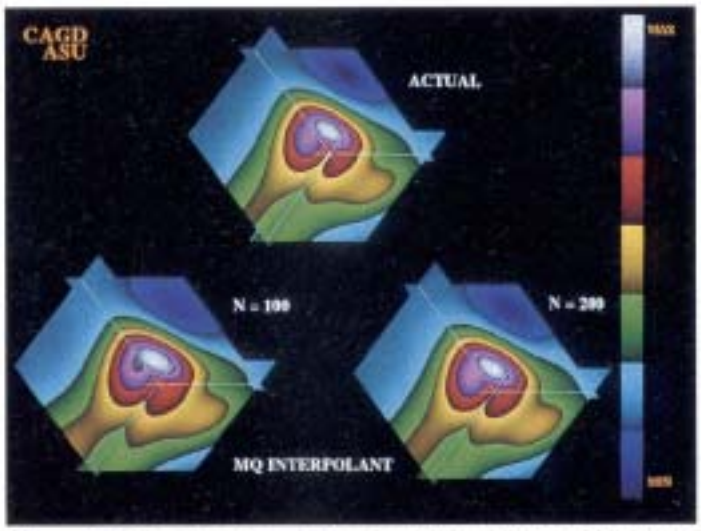


Figure 2. Shaded contour regions on planar slices of the actual function $G(x, y, z)$ (top image), and the multiquadric interpolants using $N = 100$ points (lower left) and $N = 200$ points (lower right).

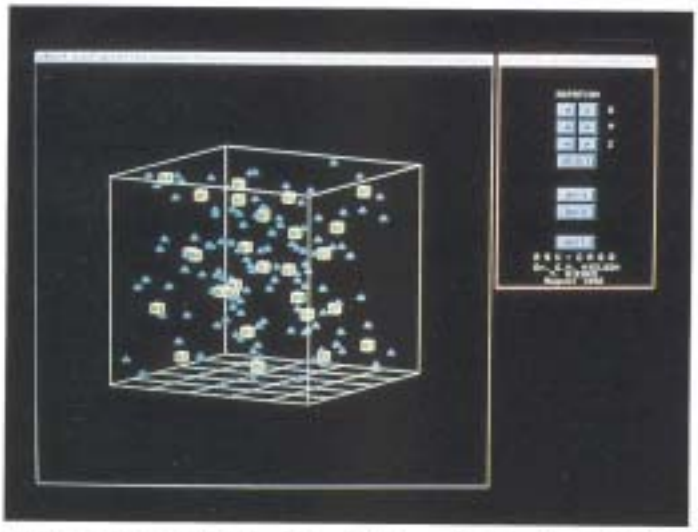


Figure 3. We selected the 27 knots (boxes) to be close to the 125 data sites (pyramids).

data samples, and the bottom right shows the approximation using $N = 200$.

When the number of data points is very large or if the dependent data are noisy, then global methods of interpolation might not be a practical way of modeling the data. An alternative is to find an approximation by the method of least squares. For multiquadrics, the modeling function is

$$F(p) = \sum_{j=1}^{M} \alpha_j \sqrt{\|p - q_j\|^2 + R^2}$$

where $\alpha_1, \ldots, \alpha_M$ are chosen to minimize

$$\sum_{i=1}^{N} \left(\sum_{j=1}^{M} \alpha_j \sqrt{\|p_i - q_j\|^2 + R^2} - F_i \right)^2$$

We can compute the unknown coefficients $(\alpha_1, \alpha_2, \ldots, \alpha_M)^t = \alpha$ by solving the linear system of equations, $A^t A\alpha = A^t F$, where $F = (F_1, F_2, \ldots, F_N)^t$, $A = (a_{ij}) = \sqrt{\|p_i - q_j\|^2 + R^2}$ is an $N \times M$ matrix and A^t is its transpose. This set of equations is often ill-conditioned, so we're usually better off to use the singular value decomposition method applied directly to the rectangular system of equations $A\alpha = F$. To completely define the basis functions, we must specify the knots q_j. We can simply choose these values and distribute them uniformly on a cuberille grid, or voxel grid. This type of grid is the 3D generalization of the 2D rectangular grid. Or we can try to select the knots so that they are distributed somewhat in the same fashion as the data sites. To compute this type of distribution, we use a 3D analogue of the 2D method Franke and McMahon proposed.[4] This iterative algorithm starts with an initial configuration of knots, then moves these values to reduce the distance between the knots and the data sites. We show an example from Dierks' work[5] in Figure 3.

Visualizing volumetric data

Most of the visualization tools available today for volumetric data assume that the data are given over a cuberille grid. This means that the data have the special form

$$(x_i, y_j, z_k, F_{ijk}) \quad i = 1, \ldots, Nx, j = 1, \ldots, Ny, k = 1, \ldots, Nz$$

To obtain cuberille data from scattered data, we can simply evaluate the modeling function to yield these data. That is, we compute

$$F_{ijk} = F(x_i, y_j, z_k) \quad i = 1, \ldots, Nx, j = 1, \ldots, Ny, k = 1, \ldots, Nz$$

Researchers have widely discussed two methods for visualizing cuberille data: the contour or isovalue surface methods and a class of methods based on ray casting, often referred to as "volume rendering" methods.[6,7] Both types are covered in the tutorial by Kaufman[8] and the proceedings edited by Upson.[9] Volume-rendering techniques can reveal a great deal of information about a trivariate relationship. Currently, the main drawback to this class of methods is the tremendous amount of computation required to compute each image. Because of this, we cannot normally use these methods interactively. However, recent developments in hardware and some new algorithms

Figure 4. An isosurface plot of the trivariate multiquadric interpolant using $N = 200$ points.

(developed by Foley, Lane, and Nielson[10]) are changing this situation.

Isovalue surface methods are the trivariate analogues of the widely used topographical contour maps and choropleth temperature maps associated with weather reports. Isovalue surface plots, enhanced with transparency and other features, are very useful for analyzing a trivariate relationship. We show an example in Figure 4. The algorithm used for this plot is based on

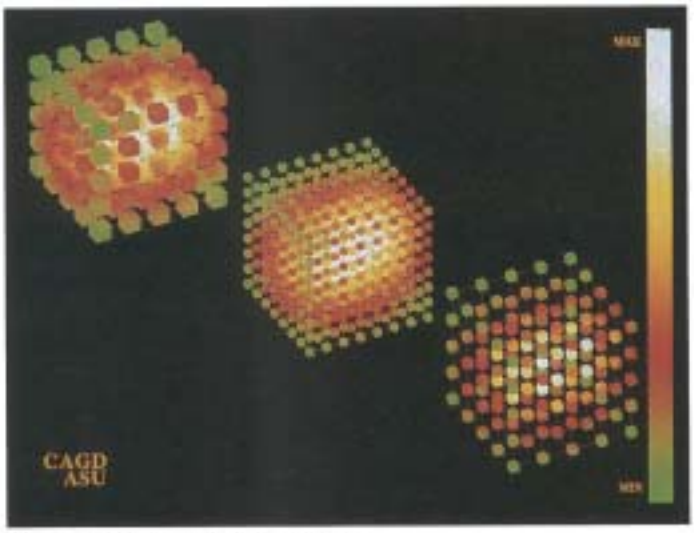


Figure 5. The tiny cubes method.

linear variation over tetrahedra rather than trilinear variation over voxels as used by other methods. We decompose each voxel into 5 (or possibly 6) tetrahedra. Except for certain degenerate cases, a contour segment will consist of a triangle or a planar quadrilateral that is then split into triangles. This yields a surface consisting of triangle facets that we can render with standard rendering techniques.

Interactive viewing of cuberille data

In addition to the volume renderings and isovalue surface plots, we have found it useful to have a collection of interactive methods for viewing the results. We designed and used a variety of methods, but here we describe only a few of the most useful ones.

The first is based on placing in the domain volume objects whose color is determined by the value of F at the location of the object. The objects can be almost anything, but spheres and cubes most readily come to mind. We use cubes for this discussion. The user specifies three resolution parameters: Nx, Ny, and Nz. There will be a total of $Nx \cdot Ny \cdot Nz$ color coded cubes displayed. In addition, the user specifies a value for the parameter M, which controls the amount of open space and consequently the size of the cubes displayed. We let the width, length, and height of each cube be denoted by (Dx, Dy, and Dz). The lower left front corner of each cube is given by the coordinates $X_i = x_0 + (i-1)Dx(M+1)$, $i = 1, \ldots, Nx$, $Y_j = y_0 + (j-1)Dy(M+1)$, $j = 1, \ldots, Ny$, $Z_k = z_0 + (k-1)Dz(M+1)$, $k = 1, \ldots, Nz$, where (x_0, y_0, z_0) is the lower left front corner of the whole domain and $Dx = (x_{max} - x_{min})/(Nx(M+1) - M)$, with Dy and Dz being similar. The function value F_{ijk} and the particular color table used will determine the color used at each vertex. The faces of the cubes are then Gouraud shaded. Each graph of this type requires the display of $6(Nx \cdot Ny \cdot Nz)$ rectangles (of which half are visible). We show example images in Figure 5. Figure 5a illustrates the case where $Nx = Ny = Nz = 5$ and $M = 1$; Figure 5b, the case where $Nx = Ny = Nz = 8$ and $M = 2$; and Figure 5c, the case where $Nx = Ny = Nz = 5$ and $M = 3$.

Similar to the previous method, the next interactive method associates a color with each data location (x_i, y_j, z_k). This color is based on the value of the dependent variable F_{ijk} and the particular color table used. Once we have a color for each vertex, we can entirely color any of the planes parallel to the axes by using linear (Gouraud) shading on each of the rectangles that comprise the plane. $Nx \cdot Ny \cdot Nz$ rectangles are perpendicular to each axis for a total of $3(Nx \cdot Ny \cdot Nz)$ rectangles to be displayed. Of course, if we directly display these rectangles, we will see those on the outer faces. To "see in" we compute an image based on a simple model of transparency for the rectangles. We sort the rectangles by distance from the viewpoint, then display them from back to front using a transparency α-buffer. Marshall Long provided the data for the example in Figure 6. It represents gas concentrations from an acoustically driven forced flow. In Figure 6a, $Nx = 8$ and $t = 0.5$. In Figure 6b, $Nx = 8$ and $t = 0.95$. An aspect of this method not exhibited by these images is that users can vary the transparency factor dynamically so that they can see different levels of the volume of data.

In addition to the volume renderings and isovalue surface plots, we have found it useful to have a collection of interactive methods for viewing the results.

In another method of viewing cuberille data, the user can simultaneously display three rectangular grid data sets, each obtained by taking a slice through the domain by holding one of

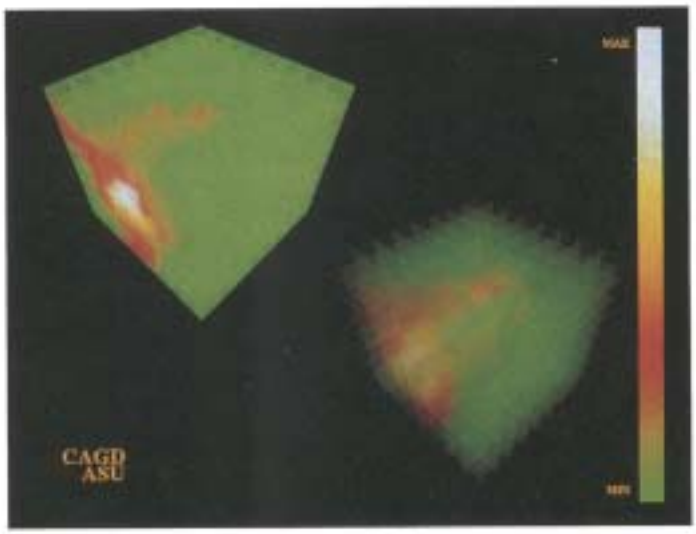


Figure 6. The vanishing cube method.

Figure 7. The surface projection method.

the independent variables fixed. In function notation terms, we simultaneously display some type of graph of the three bivariate relationships, $F_i(y, z) = F(x_i, y, z)$, $F_j(x, z) = F(x, y_j, z)$, and $F_k(x, y) = F(x, y, z_k)$. The user is allowed to interactively vary the fixed (but arbitrary) point (x_i, y_j, z_k). An example of this type of graph appears in Figure 2. In Figure 7 we show another version, one where we used a smooth shaded surface to display the three rectangular grid data sets. These three sets of surfaces could be located anywhere in the image, but we have found it convenient to have each of these graphs located on the face of a cube. We scale the values so that a point on the cube represents the minimum value; the maximum value is one unit in the direction normal to this plane. Also, we have found it useful to use different colors for each of these graphs and to display information indicating the value of (x, y, z) for the display graphs. We accomplish this by displaying three mutually perpendicular planes with colors associated in the proper manner.

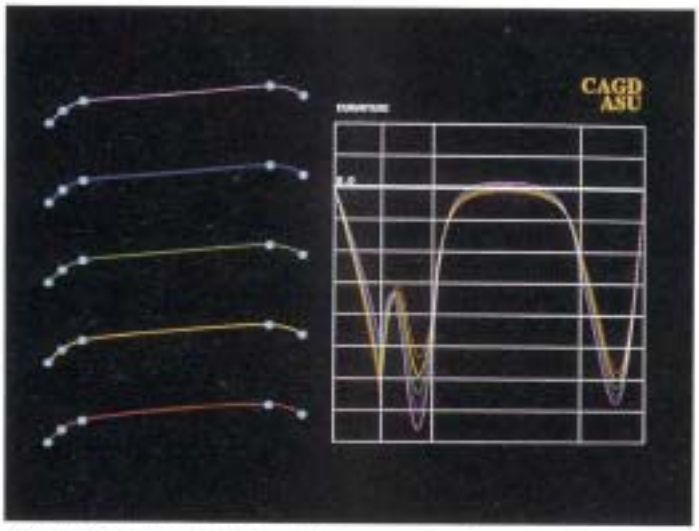


Figure 8. Sequence of curves from convex to nonconvex.

Volume interrogation techniques

Sometimes a relationship is smooth or varies slowly. This makes it difficult to detect certain qualitative changes using standard graphs. As a consequence, we have developed some volume interrogation techniques. To explain these techniques, we describe some methods that have been successful on lower dimensional problems. Figure 8 contains a sequence of curves where the bottom curve is clearly convex and the top curve is clearly nonconvex.

As you can easily see, it is difficult to determine which of the intermediate curves is convex or not based solely on their graphs. In the right image of Figure 8, the curvature has been graphed. From this you can easily determine when this sequence of curves goes from convex to nonconvex. We can extend this idea to surfaces. We can use Gaussian curvature as a texture image. The Gaussian curvature of a parametric surface

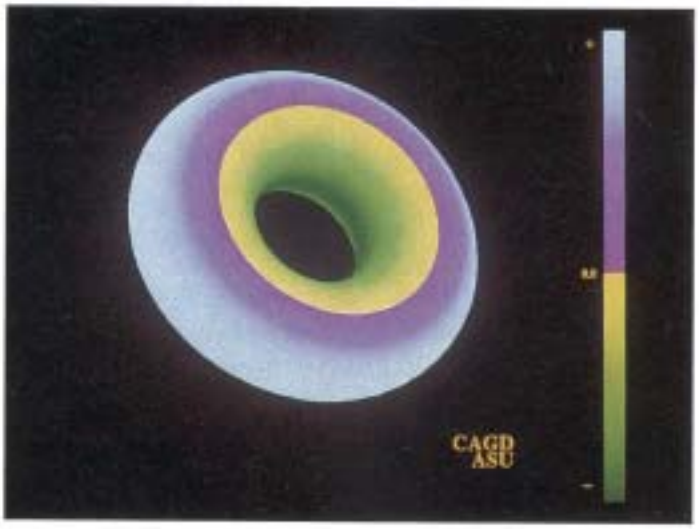


Figure 9. Gaussian curvature used as a texture.

$S(u, v) = (X(u, v), Y(u, v), Z(u, v))$ at a point, p, is $k = k_1 k_2$ where k_1 and k_2 are the principal curvatures at this point. The principal curvatures are the maximum and minimum curvatures of curves lying in planes passing through p and containing the normal at p. An example appears in Figure 9. Not only does the magnitude of the Gaussian curvature reveal quantitative geometric information about the surface, but the sign of this value reveals some interesting qualitative information about the surface. Positive, negative, and zero curvature correspond to elliptic, hyperbolic, and parabolic shapes. This implies that in a local region of positive curvature, we can reorient the surface so as to hold water, but in a region of negative curvature this is impossible.

How do we extend this idea to volumetric data? In the case of a surface given as $(x, y, F(x, y))$, we can compute the principal curvatures, k_1 and k_2, as the eigenvalues of the 2×2 matrix:

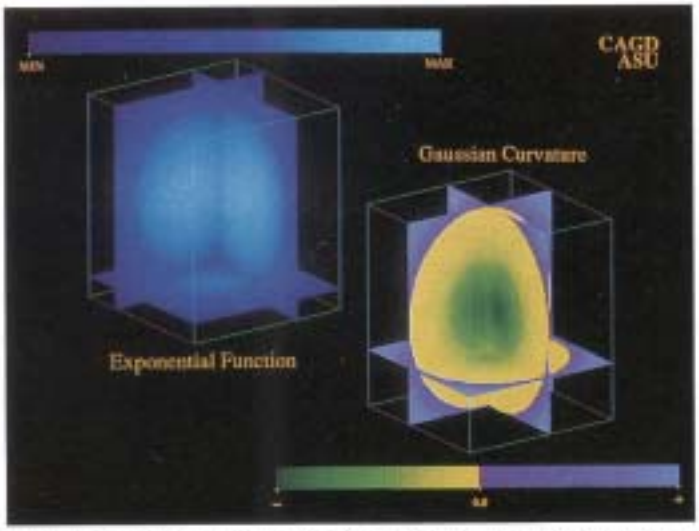


Figure 10. Volume interrogation tool with graph of original function on the left and Gauss-Kronecker curvature on the right.

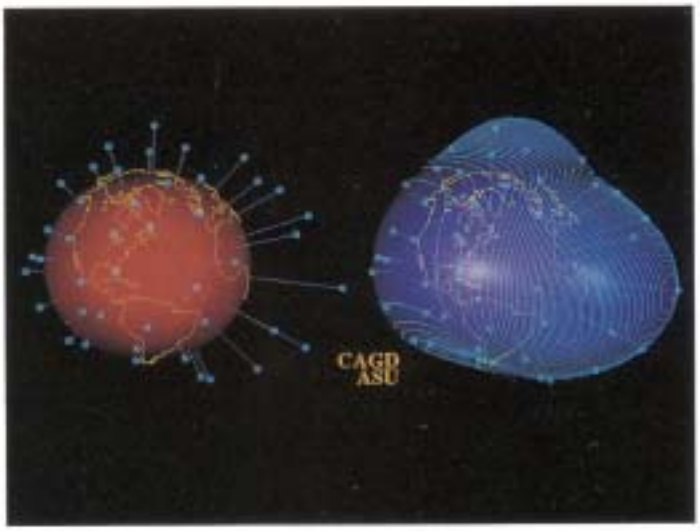


Figure 11. The left image displays data sampled on a sphere and the right image is the transparent surface graph of the modified multiquadric interpolant to this data.

$$G = \frac{1}{N}\begin{pmatrix} F_{xx} & F_{xy} \\ F_{yx} & F_{yy} \end{pmatrix}\begin{pmatrix} 1+F_x^2 & F_xF_y \\ F_yF_x & 1+F_y^2 \end{pmatrix}^{-1}$$

where $N = \sqrt{1 + F_x^2 + F_y^2}$ and the notation F_x and F_{xy} represents partial derivatives. This approach to Gaussian curvature allows an immediate extension to trivariate functions and volumetric data. We can compute the three principal curvatures, k_1, k_2, and k_3, as the eigenvalues of the 3×3 matrix:

$$G = \frac{1}{N}\begin{pmatrix} F_{xx} & F_{xy} & F_{xz} \\ F_{yx} & F_{yy} & F_{yz} \\ F_{zx} & F_{zy} & F_{zz} \end{pmatrix}\begin{pmatrix} 1+F_x^2 & F_xF_y & F_xF_z \\ F_yF_x & 1+F_y^2 & F_yF_z \\ F_zF_x & F_zF_y & 1+F_z^2 \end{pmatrix}^{-1}$$

Curvature[11,12] for a trivariate relationship is now defined as $K = k_1k_2k_3$. Figure 10 illustrates the possibilities of this new interrogation tool. The left image is a graph of the function $F(x, y, z) = \exp(-0.5(x^2 + y^2 + z^2))$. We know a qualitative change takes place for this test function at the surface of the unit sphere. We cannot easily discern this from the graph of the original function, but the graph of our extension to Gaussian curvature reveals this qualitative change quite nicely.

Modeling 3D surface domains

Let's consider the case where the domain is a surface, D, and we have the data (x_i, y_i, z_i, F_i), where $p_i = (x_i, y_i, z_i) \in D$. One of the most useful and interesting instances of the surface-on-surface problem occurs when the domain surface is a sphere. The multiquadric method has a natural extension for this. You can simply take the standard Euclidean distance and replace it by geodesic distance on the sphere. Unfortunately, this "natural extension" does not work well because the first derivatives of the basis functions have discontinuities at points antipodal to the knots. To remedy this situation, Foley[13] used basis functions where the corners have been "rounded off." This leads to a rather effective technique called the modified multiquadric method. Pottmann and Eck[14] described another approach, a spherical multiquadric method based on Hardy and Goepfert's work.[15] Pottmann and Eck used a modeling function of the form

$$F(p) = \sum_{i=1}^{N} \alpha_i \sqrt{1 + R^2 - 2R(p, p_i)}$$

where (p, p_i) represents the scalar or dot product of the points p and p_i that are points on a unit sphere. These two methods yield similar results on many test data sets, with the spherical multiquadric having a slight edge in accuracy and simplicity of implementation. We compute the values α_i by solving a linear system of equations so that $F(p_i) = F_i$ for $i = 1, \ldots, N$. The left image of Figure 11 is a visualization of the given data p_i and F_i on a spherical domain. The lengths of the radial line segments are proportional to F_i, and the segments originate at the sample points p_i on the sphere. The transparent projected surface in the right image of Figure 11 represents the graph of the interpolant Foley described,[13] using the visualization techniques discussed below. Since we can visualize the graph of the function $F(p)$ as a surface over the domain surface, we often refer to the problem as the surface-on-surface problem.

Researchers have developed very few techniques for the more general situation, when the domain D is an arbitrary closed surface. Foley et al.[16] recently developed a domain mapping method. In a nutshell, their method involves mapping the surface domain D to a sphere, solving a corresponding interpolation problem on the sphere, then mapping back to D for a solution. The surface domain D does not need to be convex, but they assumed that it is topologically equivalent to a sphere.

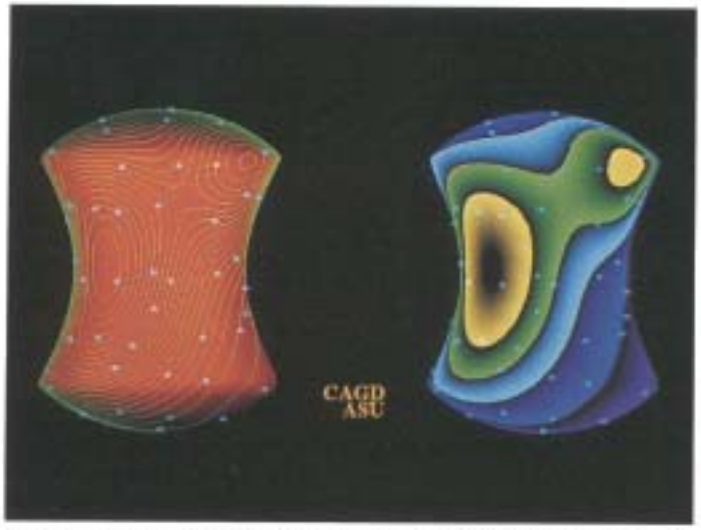


Figure 12. We applied contour curves and shaded contour regions on the apple-core domain of the domain-mapping interpolant to the data denoted by the small boxes.

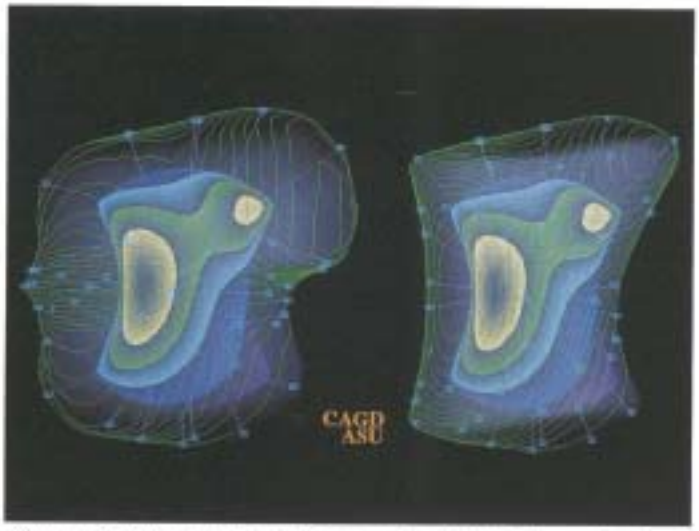


Figure 13. Transparent surface graphs of the interpolant contoured in Figure 12. The left image uses a normal projection and the right image uses a radial projection.

With the exception of implicitly defined surfaces, a closed surface D is generally defined by a mapping $B(p)$ from a simpler domain A onto D. Foley et al.[16] gave special attention to the situation where A is a planar rectangle and $B(p)$ is a periodic parametric mapping, and the case where A is a sphere and $B(p)$ is a radial projection. They also considered what happens when only discrete points on D are given and D is not known explicitly. In this case, users can form interpolants to data sampled on implicitly defined surfaces. Users can also easily apply the domain mapping technique to any domain D that is a trivariate deformation of the previous cases, assuming that the deformation is a one-to-one and onto transformation.

In Figures 12 and 13 we show different visualization techniques (described in the next section) of the domain mapping technique with the modified reciprocal multiquadric method on a spherical domain.[17] The apple core-shaped surface in Figure 12 is the domain D. The two plots contain contour curves and color-blended contour regions. A color-blended graph depicts additional information in that the color varies linearly with the value of the function from one contour to the next. We show a graph of the interpolating function $F(p)$ by projecting the transparent surface radially in the right image of Figure 13. We projected it in a direction normal to the domain D in the left image of Figure 13. The line segments connecting the domain D with the transparent surface indicate the sample locations p_i on the domain and the relative magnitudes of the values F_i.

Visualizing 3d surface domains

We can visualize the graph of a function defined over an arbitrary surface domain by drawing contour or isovalue curves on the surface (see the example in Figure 12). We based our approach to computing these contours on a triangulation of the domain surface D. In the case of the domain mapping method, this triangulation is inherited from a triangulation of the unit sphere. We evaluate the modeling function at each vertex of the triangulation and assume linear variation over each triangle, which implies that the contours for each triangle will be line segments. The collection of all of these line segments yields piece-wise linear approximations to the contour curves. By increasing the resolution of the triangulation of the surface domain, we can achieve a smoother and closer approximation, but this also increases computation and display costs. We can use another graphical approach and render the regions bounded by contours with a distinct color.

Contour plots, although often effective, are not always the best way to analyze a function because they do not clearly indicate its geometric shape. Standard methods for graphing univariate and bivariate functions use distance to indicate the value of the dependent variable. We can use the same idea for a surface-on-surface graph. We can interpret the surface graphs in Figure 1 as projecting a distance $F(x, y)$ perpendicularly from the point (x, y) in the plane. Likewise, the transparent surface graph in the left image of Figure 13 results from projecting a distance proportional to $F(p)$ in a direction perpendicular or normal to p in the surface domain. Unfortunately, for nonconvex domains this type of surface graph can have self-intersections, which makes it difficult to obtain very much geometrical information about this function. In the right image of Figure 13, we used a radial projection from the center of the domain. In general, however, for convex surface domains we prefer the normal projection. We computed the transparent surface graphs in Figures 11 and 13 as the 3D points

$$G(p) = p + N(p)L\left(\frac{F(p) - F_{\min}}{F_{\max} - F_{\min}}\right)$$

Figure 14. A hypersurface projection graph.

where p varies over the surface domain D, L is a positive scale factor, F_{min} and F_{max} are the minimum and maximum values of $F(p)$, and the direction of the projection at p is the unit vector $N(p)$. The direction vector $N(p)$ for the normal projection in the left image of Figure 13 is the outward normal vector to D of unit length at point p. The radially projected transparent surface graph in the right image of Figure 13 uses $N(p) = (p - c)/|p - c|$, where c is the center of the bounding box containing the domain D. For the surface-on-surface graph on the sphere in Figure 11, both of these techniques yield the same results.

Another visualization tool for surface-on-surface data is a method we recently developed. We call it the *hypersurface projection graph*. (Some of our ideas for this method came from discussions with Helmut Pottmann and Hans Hagen.) We based this graph on the following: The graph of a bivariate function $F(x, y)$ with some planar domain D consists of the collection of 3D points $(x, y, F(x, y))$, where (x, y) is in D. We often render this graph by displaying a network of 2D lines based on the projection of the 3D points (see Figure 1). In the case of surface-on-surface data, we have a graph consisting of points in 4D space, $(x, y, z, F(x, y, z))$, where $(x, y, z) \in D$. When we project these points (or their rotated versions) to 3D space using a parallel projection, we obtain points on a 3D surface. We have found it useful to simultaneously display three different projections:

- $(F(x, y, z), y, z)$
- $(x, F(x, y, z), z)$
- $(x, y, F(x, y, z))$

We show an example of this method in Figure 14.

Remarks

For the most part, we have only covered one type of basis function for the mathematical model, namely, the multiquadric. Although the multiquadric method generally produces excellent results on smooth test data, we suggest that you apply other methods to the data and compare the results. (Franke and Nielson discussed many other methods.[1]) We are currently comparing and evaluating a number of methods for interpolating volumetric, scattered data. We'll report on this later. We are also developing new methods that take advantage of some structure in the data, such as data sampled at varying depths in arbitarily located wells. ❑

Acknowledgments

We acknowledge the support of the U.S. Department of Energy under contract DE-FG02-87ER25041 to Arizona State University and NATO under research contract RG 0097/88. We wish to thank the members of the CAGD group at ASU for their help and support.

References

1. R. Franke and G. Nielson, "Scattered Data Interpolation and Applications: A Tutorial and Survey," in *Geometric Modeling: Methods and Their Application*, H. Hagen and D. Roller, eds., Springer, Heidelberg, Germany, 1990.
2. R.E. Carlson and T.A. Foley, "The Parameter R^2 and Multiquadric Interpolation," *Computers and Mathematics with Applications*, Vol. 21, No. 9, 1991, pp. 29-42.
3. T.A. Foley and D. Lane, "Visualization of Irregular Multivariate Data," *Proc. Visualization 90*, IEEE Computer Society Press, Los Alamitos, Calif., 1990, pp. 247-254.
4. J.R. McMahon and F. Franke, "Knot Selection for Least Squares Thin Plate Splines," to be published in *SIAM J. Statistical and Scientific Computing*, 1991.
5. T. Dierks, "Analysis and Visualization of Scattered Volumetric Data," master's thesis, Arizona State University, Tempe, AZ, 1990.
6. M. Levoy, "Display of Surfaces from Volume Data," *IEEE CG&A*, Vol. 8, No. 3, May 1988, pp. 29-37.
7. R.A. Drebin, L. Carpenter, and P. Hanrahan, "Volume Rendering," *Computer Graphics* (Proc. Siggraph), Vol. 22, No. 4, Aug. 1988, pp. 65-74.
8. A. Kaufman, *Volume Visualization*, IEEE Computer Society Press, Los Alamitos, 1990.
9. *Proc. Chapel Hill Workshop on Volume Visualization*, C. Upson, ed., Chapel Hill, NC, Dept. Computer Science, University of North Carolina, Chapel Hill, May 1989.
10. T.A. Foley, D. Lane, and G. Nielson, "Towards Animating Ray-traced Volume Visualization," *Visualization and Computer Animation J.*, Vol. 1, No. 1, Aug. 1990, pp. 2-8.
11. M.J. Laszlo, "Techniques for Visualizing 3D Manifolds," in *Proc. Visualization 90*, IEEE Computer Society Press, Los Alamitos, Calif., 1990, pp. 342-352.
12. J. Thorpe, *Elementary Topics in Differential Geometry*, Springer, Heidelberg, Germany, 1979.
13. T.A. Foley, "Interpolation to Scattered Data on a Spherical Domain," in *Algorithms for Approximation II*, M. Cox and J. Mason, eds., Chapman and Hall, London, 1990, pp. 303-310.
14. H. Pottmann and M. Eck, "Modified Multiquadric Methods for Scattered Data Interpolation over a Sphere," *Computer Aided Geometric Design*, Vol. 7, No. 1-4, June 1990, pp. 313-322.
15. R.L. Hardy and W.M. Goepfert, "Least Squares Prediction of Gravity Anomalies, Geoidal Undulations, and Deflections of the Vertical with Multiquadric Harmonic Functions," Geophysical Research Letters, Vol. 2, 1975, pp. 423-426.
16. T.A. Foley, et al., "Interpolation of Scattered Data on Closed Surfaces," *Computer Aided Geometric Design*, Vol. 7, No. 1-4, June 1990, pp. 303-312.

Gregory M. Nielson is a professor of computer science and adjunct professor of mathematics at Arizona State University, where he teaches and performs research in computer graphics, computer-aided geometric design, and scientific visualization. He is also a participatory guest scientist at Lawrence Livermore National Laboratory.

Nielson received his PhD from the University of Utah in 1970. He is a member of ACM, IEEE Computer Society, and the IEEE Computer Society Technical Committee on Computer Graphics.

Thomas A. Foley is an associate professor in the computer science department at Arizona State University and a visiting researcher at Lawrence Livermore National Laboratory. He has also taught at California Polytechnic University in San Luis Obispo and the University of Nevada. His current teaching and research interests are computer-aided geometric design, multivariate data fitting, and scientific visualization.

Foley received his PhD from Arizona State University in 1979. He is a member of the IEEE Computer Society and the IEEE Computer Society Technical Committee on Computer Graphics.

Bernd Hamann is a PhD candidate in the computer science department at Arizona State University. His research interests include computer-aided geometric design and scientific visualization.

Hamann did his undergraduate work in computer science at the Technical University of Braunschweig, Germany.

David Lane is a PhD candidate in the computer science department at Arizona State University and a research associate in ASU's Computer-Aided Geometric Design Group. His dissertation topic is the representation and visualization of scattered multivariate data, and his research interests include computer-aided geometric design and scientific visualization.

Lane received his BS and MS in computer science from the University of Nevada, Las Vegas. He is a member of ACM Siggraph, SIAM, IEEE Computer Society, and the IEEE Computer Society Technical Committee on Computer Graphics.

Nielson can be reached at the Department of Computer Science, Arizona State University, Tempe, AZ 85287-5406. His e-mail address is nielson@enuxva.eas.asu.edu.