CONTOURS TO DIGITAL ELEVATION MODELS: GRID-BASED SURFACE RECONSTRUCTION METHODS

By

Michael B. Gousie

A Thesis Submitted to the Graduate

Faculty of Rensselaer Polytechnic Institute

in Partial Fulfillment of the

Requirements for the Degree of

DOCTOR OF PHILOSOPHY

Major Subject: Computer Science

Approved by the
Examining Committee:

Wm. Randolph Franklin, Thesis Adviser

Mukkai S. Krishnamoorthy, Member

Bruce R. Piper, Member

Charles V. Stewart, Member

Rensselaer Polytechnic Institute
Troy, New York

April 1998
(For Graduation May 1998)

CONTOURS TO DIGITAL ELEVATION MODELS: GRID-BASED SURFACE RECONSTRUCTION METHODS

By

Michael B. Gousie

An Abstract of a Thesis Submitted to the Graduate

Faculty of Rensselaer Polytechnic Institute

in Partial Fulfillment of the

Requirements for the Degree of

DOCTOR OF PHILOSOPHY

Major Subject: Computer Science

The original of the complete thesis is on file
in the Rensselaer Polytechnic Institute Library

Examining Committee:

Wm. Randolph Franklin, Thesis Adviser
Mukkai S. Krishnamoorthy, Member
Bruce R. Piper, Member
Charles V. Stewart, Member

Rensselaer Polytechnic Institute
Troy, New York

April 1998
(For Graduation May 1998)

CONTENTS

LIST OF TABLES

LIST OF FIGURES

ACKNOWLEDGMENT

And gladly wolde he lerne and gladly teche.

– Chaucer [14]

There are many people who have made a positive impact along this journey. Thanks go to my thesis adviser Randolph Franklin, a veritable wellspring of ideas. Thanks also to my committee members Mukkai Krishnamoorthy, who checked up on me periodically to make sure I was making progress, and Bruce Piper, who provided a careful reading of the equations in the thesis. Special thanks to Charles Stewart, who gave me constructive comments and never complained when I slacked off from my teaching duties while in the thesis crunch.

Thinking back over my years at RPI, I'd like to thank the original gang back in the dark days of qualifying exams, especially Jon Campbell and James Bankoski. It was an immense help not to have to struggle alone. The actual graduation of some of the past inhabitants of the Bar & Grill 204 gave me incentive to proceed. Good times were spent with Louis Ziantz, Jim Teresco, Mark Hulber, Wes Turner, Eric Rouchka, Darren Lim, Robin Flatland, Scott Epter, and the entire noon-time basketball gang. The efforts of the very capable and overworked `labstaff` personnel, Nathan Schimke and Roddy Collins, are much appreciated. Special thanks to my buddies Rick Kline, who was always there when I needed him, and Andrew Shapira, who made sure I looked at the brighter side of things. Michael Keable, who called frequently to make sure I was still alive, gets the "Friend from Afar Support Award."

I must also thank my parents who have supported me wholeheartedly in this endeavor. Mom's encouragement and Dad's eternal optimism are appreciated. Finally, a very special thanks to my loving wife Jen, who put her dreams on hold without complaint for these past few years.

ABSTRACT

This thesis presents several new approaches to the problem of creating a terrain surface from contours on a regular grid. Previous solutions often generate surfaces with several kinds of artifacts, most notably terraces. The solutions presented in this thesis minimize the terracing effect while creating surfaces that are globally accurate.

The first technique improves upon previous thin plate approximations. The Intermediate Contours method creates additional contour lines in between the existing contours to alleviate the terracing problem. The Maximum Intermediate Contours method extends this idea to the creation of an initial surface by continually computing intermediate contours. Because peaks can not be computed by intermediate contours, a novel approach for computing such areas using Hermite splines is presented. Gaussian smoothing is applied to smooth the final surface.

The second technique makes use of slope information in an initial surface approximation to create "gradient lines." These gradient lines are paths that follow the steepest slope from local minima to local maxima. An interpolating spline is fit along the paths to create a surface. Thin plate approximation techniques are applied to produce the final, smooth result.

Finally, an interpolating spline method is presented which computes good quality surfaces faster than the aforementioned techniques. Catmull-Rom splines are computed in the horizontal and vertical directions to create an initial surface which is then smoothed with the Gaussian smoothing function.

All of the surfaces created by the new methods are qualitatively analyzed. The generated surfaces are also compared to current US Geological Survey standards. Accuracy and smoothness criteria show that the surfaces computed by the new methods are quantitatively superior than previous techniques.

CHAPTER 1

It is slightly non-trivial, but not terribly non-trivial.

– Mukkai Krishnamoorthy

Introduction

Displaying information via two- or three-dimensional images is becoming increasingly important in an era of ever-improving CAD/CAM systems, scientific visualizations of various kinds, sophisticated computer games, hyper-media, and even virtual reality. Geographic Information Systems, or GIS, are no exception. Data pertaining to the world around us is being collected at increasing rates. As the amount of information multiplies, Geographic Information Systems are called upon to display the myriad data in the context of geographic locations.

Over the years, there have been conflicting and diverse attempts at precisely defining the term Geographic Information System. Although the main theme of spatial data is omnipresent, ever-changing technology has brought about increased robustness. In a fairly recent book, Dangermond gives this overview of a GIS [17]:

> A simple definition is that a GIS is an organized collection of computer hardware, software, and geographic data designed to efficiently capture, store, update, manipulate, and display all forms of geographically referenced information. (p. 11)

In Maguire's opinion [56], which seems to combine most of the earlier definitions, a GIS can be thought of as a melding of three different "views:" map, database, and spatial analysis. The map view enables the system to visually display a map of a geographic location. The database contains information pertaining to the same location. Finally, the spatial analysis view allows the user to find new information about the map in question, such as the size of some area or the elevation

of a particular point. Depending on the user's requirements, all three views may be applicable to a GIS at the same time.

The "map" portion in most GISs are based on a "layering" system. The base layer is a two-dimensional map of some geographic location. Data stored in the database, such as population density, soil type, or utility locations is layered over the base map in different colors, shades, or patterns which serve to differentiate the information. This is how multivariate visualization is achieved in packages such as MapInfo [57] and Arc/Info [26]. In addition, a user may wish to view the geographic location in three dimensions, as can be done in a system such as GRASS (Geographic Resources Analysis Support System)[1]. A three-dimensional surface map of an area is usually stored as a grid of elevation points called a Digital Elevation Model (DEM). Using a DEM, a user can view an area in three dimensions, giving a clearer understanding of the problem. The term Digital Terrain Model (DTM) is also widely used, but often implies other attributes in addition to elevation data.

1.1 Importance of Digital Elevation Models

Aside from the purely aesthetic value of observing terrain in three rather than two dimensions, there are many scientific and practical uses for DEMs, including [11]:

- The need to store digital topographic maps. In an age where almost everything is stored in digital format, it is obvious that maps need to be stored in this manner as well. This allows such maps to be used not only in pure GISs, but in other computer applications, such as games, atlases, and encyclopedias.

- Problems in road design and similar civil and military engineering projects. Viewing a location in three dimensions may make it easier to determine the path of a road through hilly or mountainous terrain, for example.

  In January of 1998, a huge ice storm ravaged upper New York state and parts of Canada. WAMC [84] reported that MapInfo donated software and technical support for the National Guard which was frantically trying to restore power in large portions of New York. The MapInfo software was used to create

maps of specific problem areas, making it easier for crews to locate them. Three-dimensional maps may have helped the crews to visualize the terrain and to determine the kind of equipment necessary to reach remote areas and to complete repairs.

- Realistic display of landforms for such diverse areas as pilot training, weapons guidance, and landscape architecture.
- Statistical analysis and comparison of different kinds of terrain.
- Computing slope maps, aspect maps, and slope profiles that can be used to produce shaded relief maps and aid in other geomorphological problems. For example, DEMs have been shown to be useful in mapping flood areas [55]. These maps can be used to predict which areas may be adversely affected by high water.
- Providing a background for displaying thematic information such as is done in current Geographic Information Systems in two-dimensions.
- The analysis of cross-country visibility. For example, a ski area in Oregon wanted to expand its accessible terrain. Environmental groups were concerned that the new ski trails would adversely impact a hiking trail nearby. A three-dimensional rendering of the ski area as viewed from the hiking trail allowed the designers to incorporate the environmentalists' concerns. As another example, a DEM can be used to help determine how much of the surrounding area will be able to see particular structures, an important consideration in building a housing development near undesirable towers or buildings.
- The elevation data can be replaced by other continuously varying attributes, thus allowing the DEM to represent "surfaces" of items such as travel time, cost, population, etc.

Once data is collected or computed and stored in a DEM, other useful information can be derived, including profiles, volume estimation, contour maps, and

drainage maps, among others [11]. Furthermore, DEMs are not limited to storing elevation values; they can be used to store any kind of spatial data, such as range data collected from any kind of object, measurements of real-world items in three-space, medical imagery, and so forth. In fact, they can be used to store any grid-based phenomena that a user may wish to view in three-dimensions.

1.2 Motivation

There are several factors which motivate this research. The first and foremost aspect is that there exists a dearth of accurate DEMs available to the GIS user and researcher. DEMs that are available are often computed using crude averaging methods giving only a very general idea of the topography of a particular area. The United States Geological Survey (USGS) is currently creating a new database of DEMs; however, this project will take many years.

A second motivation is that DEMs are enormous, making storage of such maps a costly proposition, even with the advances in memory performance. One test case, an uncompressed 800×800 DEM of Mt. Washington in NH, requires about 5 megabytes of memory. An equivalent 800×800 file containing only isolines (contour lines) requires 1.4 megabytes of space. By storing only the isolines in a non-gridded format, the storage requirements can be reduced still further to about 12 kilobytes. Note that memory requirements are relative to the type of terrain represented. Very steep terrain contains more isolines per unit area; flatter terrain contains fewer isolines. In compressed form, the ratio of the space required for the isolines to the space requirements of the DEM is roughly the same; using the `gzip` utility, the initial data file takes about 43 kilobytes of memory, while the DEM requires about 1.7 megabytes. Better compression techniques for elevation data can be found in work done by Franklin [35].

One solution to both problems is to re-create the three-dimensional surface from only as much data as necessary. There exist many elevation-based data sets, notably Digital Line Graphs (DLG) of the USGS, and other scattered data formats. Currently, DLG files are available for most areas of the country. Among other things, a DLG stores terrain elevation information in conjunction with line segments which

form contour lines. Taken together, the isolines form a complete contour map. Such a contour map can be used to interpolate or approximate a surface from the observed elevation values. If there exists a good method for converting such isoline files to accurate DEMs, a given GIS or other mapping system could store many more maps in the same amount of memory. In addition, contour line data is used to model many other phenomena, such as soft tissues in medical imaging. The problem, then, is to produce good quality DEMs from available contour line data.

Computer processors are becoming faster and faster. Therefore, users expect results faster than ever before. A secondary issue is that computing a DEM should take a reasonable amount of time. Some current methods may take several hours on powerful machines to compute a surface. A system that is highly accurate but takes an inordinate amount of time may be less useful to a user than one that runs faster with perhaps a bit more error.

1.3 New Interpolation/Approximation Techniques

There are many techniques that have been used to interpolate or approximate a surface from elevation or, equivalently, range data. However, many of these algorithms are not firmly grounded in mathematical or physical theory. Furthermore, the resulting surfaces are often not quantitatively or qualitatively accurate.

In this thesis, we present two new enhancements to the minimum curvature surface which produce quantifiably better DEMs. The method of minimum curvature is one accepted method for producing DEMs. A partial differential equation (PDE) which models a thin plate being draped over the data set is used to create such a surface. By itself, this technique does not give adequate results, but it forms a solid, theoretically sound basis for our work. We also present two new and more direct techniques which produce similar results but do not need the solution to the partial differential equation and its attendant high computational costs. Figure 1.1 shows a flowchart which portrays the entire DEM construction system and all its possibilities.

The first technique, called Intermediate Contours or IC, produces a better surface by interpolating isoline data to produce more elevation points for subsequent

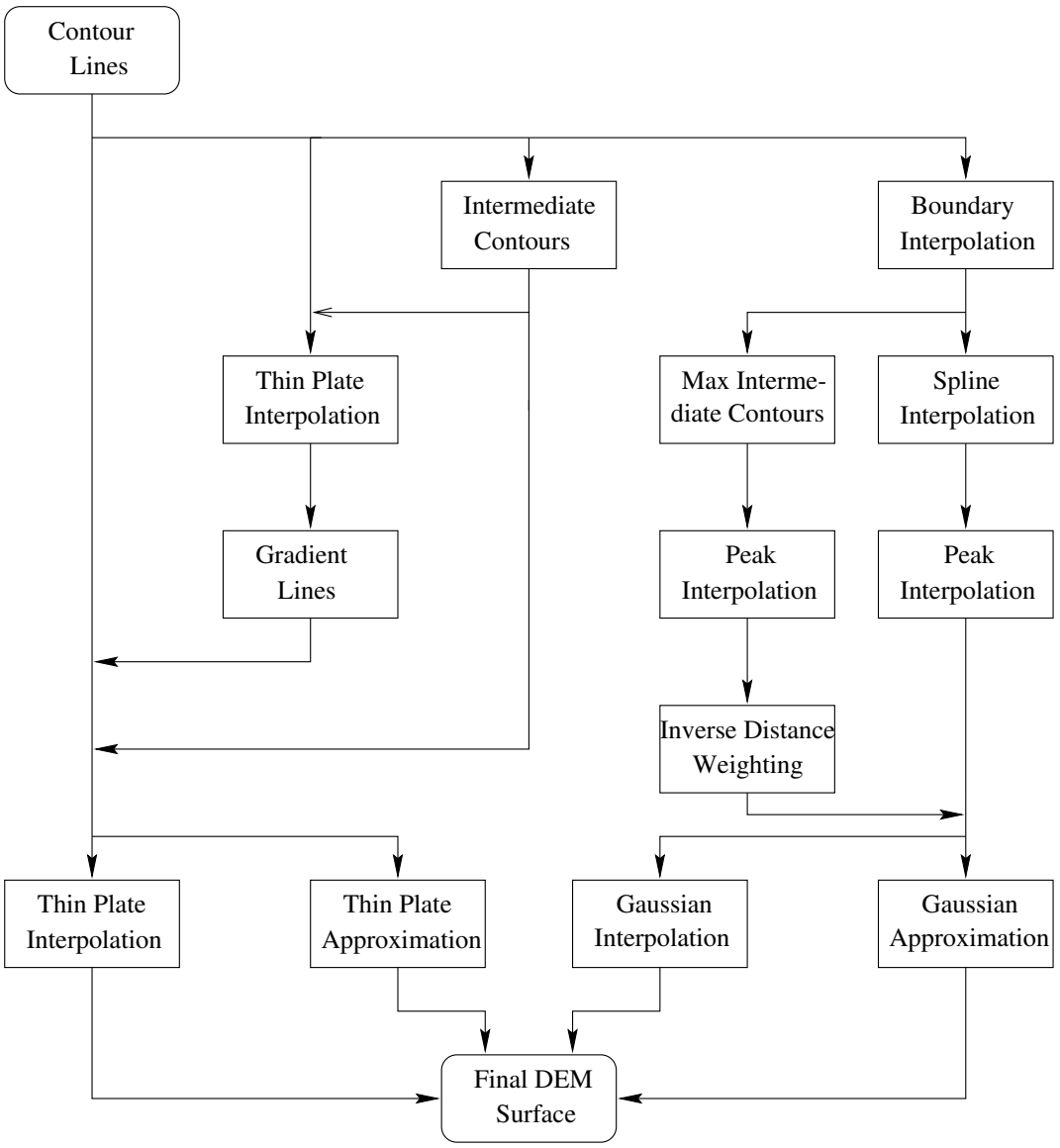

Figure 1.1: Flowchart showing showing all interpolation and approximation alternatives.

thin plate surface computations. These new elevations comprise new contours in between previously known contour lines. The new "intermediate contours" greatly improve the basic minimum curvature surface.

The second technique, uses some information inherent in contour lines to produce "gradient lines." These gradient lines are interpolated paths which follow the steepest slope between local minima and maxima. In so doing, these paths cut across contours, giving better ridge lines and reducing other thin plate inadequacies. Note

that intermediate contours can be used to create a better initial surface used for the gradient lines computation.

Finally, we introduce two methods which compute DEMs directly from contour data, without the need for any thin plate computations. Except for the initial surface computation, both methods follow the same steps to the creation of the final DEM. In the Maximum Intermediate Contours (MIC) method, the maximum number of intermediate contours are generated to produce the initial, approximate surface. The second algorithm employs one-dimensional splines which interpolate across contours to form the initial surface. Neither method can interpolate peaks. These areas are interpolated separately using Hermite splines. Other, small remaining gaps are filled with inverse distance weighting. Finally, a common smoothing function is used to achieve the final surface. Both algorithms are much faster than the thin plate methods while producing good quality surfaces.

1.4 Organization of Thesis

In this chapter, we have given an overview of Geographic Information Systems in general, and discussed the importance of Digital Elevation Models. We have outlined briefly some new methods for computing DEMs from contour line data.

Chapter 2 explains the notion of minimum curvature and the use of the thin plate PDE, and gives a historical review of reconstruction methods. The following two chapters show how the basic thin plate approach can be improved. Chapter 3 gives the details of the Intermediate Contours (IC) method, while Chapter 4 explains the Gradient Lines method. The two methods which directly compute a DEM without the need for thin plate computations, the Maximum Intermediate Contours (MIC) and Fast Spline methods, are given in Chapter 5. In Chapter 6 we discuss how the input files were created and define the tests used to determine a surface's accuracy. The experimental results are shown in Chapter 7. In Chapter 8, we discuss the results and give some conclusions. The research contributions are also reviewed in that chapter, as well as a discussion of future work. Finally, the Appendix contains other methods that were investigated but that did not result in acceptable DEMs.

CHAPTER 2

I could have substituted 1 for r at the beginning,
but then the result would not be as mathematically interesting.

– Boleslaw Szymanski

Review of Surface Reconstruction

Much work has been done in the areas of interpolation/approximation by the AI vision and earth sciences communities. In vision, the problem often is referred to as "surface reconstruction," because sparse elevation or depth data is transformed to a surface which will, hopefully, clearly show the underlying object or objects that the data represents. Similarly, in the earth sciences, elevation data, often in the form of isolines, is used to reconstruct the terrain surface that the data represent. The two problems are extremely similar, but often on a much different scale.

2.1 Surface Reconstruction Using Partial Differential Equations

An important goal in the creation of DEMs from contour (or scattered) data is that the resulting three-dimensional surface be smooth. The surface should not have any unnatural dips, curves, or other strange anomalies. The viewer should not perceive the underlying data points which were used initially to create the surface. Partial Differential Equations (PDEs) can be used to model surfaces subject to certain constraints. One such PDE is the Laplacian, or heat-flow equation:

$$0 = \frac{\partial^2 z}{\partial x^2} + \frac{\partial^2 z}{\partial y^2} \tag{2.1}$$

We can map elevations to temperatures, such that z represents elevations. If we assume that contour lines are fixed at known temperatures and the surface conducts heat uniformly, then each point on the surface will converge to some temperature equilibrium. The temperatures can then be mapped back to elevations. Using finite difference techniques, solving the PDE on a grid results in

$$0 = z_{i-1,j} + z_{i+1,j} + z_{i,j-1} + z_{i,j+1} - 4z_{i,j} \tag{2.2}$$

for elevation values not already known at location i, j. However, this method produces surfaces with severe terracing in between the contours, as shown by Wood and Fisher [89]. A more robust PDE is one that models a thin plate which is draped over the observed data points. It uses the principle of minimum curvature which restricts the amount of bending in the plate. In its pure form, the thin plate will pass through all of the points, resulting in a surface that is a true interpolation of the observed points. In two dimensions, such an interpolant is also called the natural bicubic spline. An advantage of a surface found by this spline is that it belongs to $C(R^2)$; that is, the surface is continuous over the two-dimensional surface [70]. In some cases, it may be desirable to let the thin plate pass *near* the observed points, rather than through them. Such a process, also called a smoothing spline, produces an approximation of the surface. An approximated surface may be more desirable than an interpolated surface due to its smoothness. Indeed, Hutchinson and Gessler conclude that exact interpolation of spatial data using thin plate splines is rarely appropriate [47], and that smoothing is generally needed. Whatever the advantages and limitations of the thin plate interpolation or approximation method, it is useful to note that the techniques are incorporated into several GIS packages [88] such as Surfer [79], GRASS [1], and SphereKit [78].

2.1.1 Thin plate equations

Given N data points, where $i \in \{1..N\}$, the differential equation that models a thin plate is given by:

$$f_i = \frac{\partial^4 z}{\partial x^4} + 2\frac{\partial^4 z}{\partial x^2 \partial y^2} + \frac{\partial^4 z}{\partial y^4} \tag{2.3}$$

where f_i is the force at position i, $x = x_i, y = y_i$, and z is the elevation at (x_i, y_i) [9]. Note that if there is an observed elevation value w_i at (x_i, y_i), then $z_i = w_i$. Integrating twice over the region where the data values are given, R, and rewriting using the notation where f_{xx} means $\frac{\partial^2 f}{\partial x^2}$, a surface *interpolation* of the data can be found by solving:

$$0 = \int_R \int \left(f_{xx}^2 + 2f_{xy}^2 + f_{yy}^2 \right) dx \; dy \tag{2.4}$$

If an *approximation* is desired, then the computed surface must only pass near known values. In such a case, care must be taken so that the surface does not deviate too much from the known data values. Such an approximation can be modeled by adding 2.4 to a function which minimizes the total energy E of a system. One such function [48] is:

$$E^2 = \sum_{i=1}^{n} (z_i - f(x_i, y_i))^2 + \beta^2 \int_R \int \left(f_{xx}^2 + 2f_{xy}^2 + f_{yy}^2 \right) dx \; dy \tag{2.5}$$

where β is a regularizing parameter used to achieve a smoother solution. Choosing a small β results in a close approximation of the data, while choosing a large β results in smoother solution [48].

In practical terms, if the data is in the form of a mesh of points (x_1, y_1) to (x_n, y_n), and the boundaries are ignored, then using finite difference techniques, the solution to the above equation 2.4 is the biharmonic equation [9]:

$$\begin{aligned} 0 &= z_{i-2,j} + z_{i,j-2} + z_{i+2,j} + z_{i,j+2} \\ &\quad +2(z_{i-1,j-1} + z_{i-1,j+1} + z_{i+1,j+1} + z_{i+1,j-1}) \\ &\quad -8(z_{i,j-1} + z_{i-1,j} + z_{i,j+1} + z_{i+1,j}) \\ &\quad +20z_{i,j} \end{aligned} \tag{2.6}$$

where each $z_{i,j}$ represents the elevation at (x_i, y_j). This equation can be regarded as a neighborhood of points, with different weights (coefficients) assigned to current values according to their distance from the center:

$$\begin{array}{ccccc} & & 1 & & \\ & 2 & -8 & 2 & \\ 1 & -8 & 20 & -8 & 1 \\ & 2 & -8 & 2 & \\ & & 1 & & \end{array}$$

The boundary values at both the edges and the corners must also sum to 0. Thus, there are similar equations for all boundary points (See [9] or [82]).

The method chosen to solve the biharmonic equation is iteration. Convergence is achieved when the absolute change in all nodes is less than some ϵ value, typically chosen to be 0.005. For most DLGs, which give elevation values in meters, this has the implication that no computed elevation is changed more than 5mm in one iteration. The most direct, sequential iterative method is the Jacobi method, where finding each new point depends only on values found in the previous iteration [91]:

$$z_i^{p+1} = z_{i-2,j}^p + z_{i,j-2}^p + ... + 20z_{i,j}^p \tag{2.7}$$

where $p =$ iteration.

Somewhat faster convergence can be achieved with the Gauss-Seidel method whereby values of the current iteration, that is $z_{i,j}^{p+1}$, are used whenever possible [91].

As the grid becomes larger, both the Jacobi and Gauss-Seidel methods become prohibitively slow; for a grid of $n \times n$ points, convergence is achieved in time $O(Ln^2)$,

where L is the number of iterations required. In [80], Terzopoulos contends that $L = n^m$, where $m = 4 =$ highest order of partial derivatives, a figure supported by Matheson [58]. This gives us a total time complexity of $O(n^6)$, a rather daunting figure.

A method that speeds up the process by orders of magnitude is the multigrid numerical relaxation approach. In very general terms, the idea is to find values of the original, "fine" grid by reducing the problem to many smaller, "coarse" grids. The solution to the smallest coarse grid is computed quickly and is then used to find the solution of the next finer grid. This process repeats itself until the solution to the original grid is found. The multigrid process is a major theme of Terzopoulos [80] and, to a lesser extent, Smith [77]. Terzopoulos showed that multigrid relaxation techniques significantly increase performance, found to be $O(n^2 \log n)$ by Briggs [10]. However, it is difficult to determine the true convergence factor, so Briggs used heuristics and experimental results (on a rather small sized sample) to reach the above conclusion. An explanation of the multigrid algorithm as applied in general and a specific example (using the Laplacian) is shown in [65]. This has become a very popular method, sprouting many tutorials such as those from Briggs [10] and Rüde [68], among others. Douglas [20] points to tutorials, bibliographies, and software.

2.1.2 Discontinuities

A major source of difficulty in the production of smooth and accurate surfaces from sparse data using the thin plate approach is that of discontinuities in the surface. For example, one kind of discontinuity in terrain may be described as a significant elevation drop between two flatter areas. This is often the case near cliffs or canyons. Such a discontinuity is shown in Figure 2.1, where the vertical bars represent elevations of a profile at regular intervals. The implied discontinuity is in the problem that there is a flat surface connecting the higher elevations and another flat surface connecting the lower elevations, with no clear indication of the desired surface in between. Using the general thin plate method, the resulting surface is shown as the dark line. This curve exhibits behavior that is known as the Gibbs

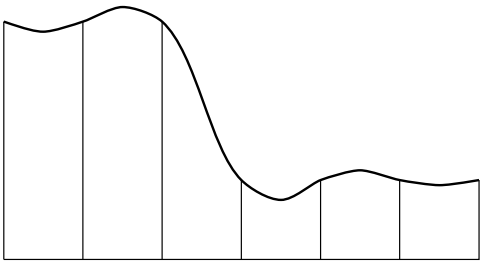

Figure 2.1: Profile showing discontinuity and resulting thin plate surface (dark line)

phenomenon [29]: there is significant "overshoot" on either side of the discontinuity. This is the result of the method attempting to minimize the curvature in areas where the curvature is naturally high. This problem has been handled in various ways (see Section 2.2).

2.1.3 Interpolation Problems Specific to Contour Data

Although the thin plate equation has been used in the surface reconstruction problem, one obstacle is unique when using contour line data as opposed to sparse data. In simplest terms, solving the thin plate equation can be stated as finding the weighted average of a node's neighbors. Consider contour data depicting hilly or mountainous terrain. Furthermore, consider a contour line A with a certain elevation, and a second contour line B which is at the next lower elevation, as shown in Figure 2.2. Typically, contour B will have more data points than contour A, because mountains get smaller (i.e., contours enclose less area) as they get higher. This raises the problem that if one attempts to find the elevation of some point p which lies between A and B, then the number of elevation values whose magnitude is near the elevation of B is greater than the number of elevation values whose magnitude is near the elevation of A. Using Equation 2.4 to find elevation values between two such contours results in a "terracing" effect; that is, there are more lower elevation values than higher values, creating a surface whose average elevation is closer to B than A. This phenomena worsens as the horizontal (flat) distance between successive contour lines increases. This behavior is intuitive, as a thin plate tends to flatness.

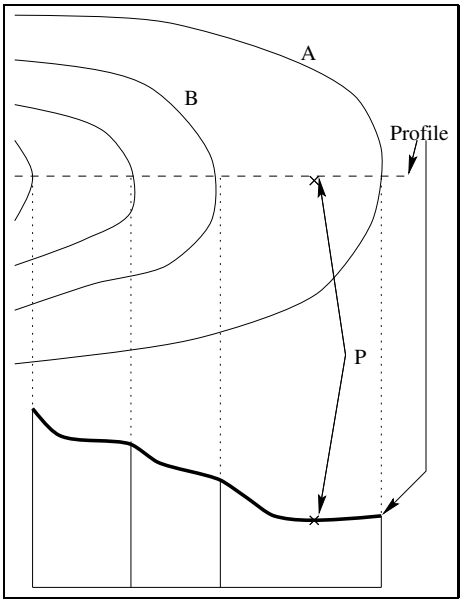

Figure 2.2: Profile showing terracing problem with thin plate surface

A second problem involves the digitization process. Assuming that the base map is accurate, there are obvious problems when superimposing curves onto a regular grid. Curves will often exhibit aliasing problems, meaning that there may be several pixels in a small area depicting a curved portion of a contour. Because all of these pixels come from the same contour, they will all represent the same elevation. It is difficult for any curvature minimizing equation to interpolate such areas smoothly. The result is that there will be tiny flat spots in the areas where such pixels were grouped.

2.2 Survey of Thin Plate Methods

The idea of using a thin plate draped over observed data values to approximate a surface is not new. Early uses tended towards terrain reconstruction, whereas the method was later employed often in machine vision applications. A good overview of some of the early uses of thin plate (and other) splines can be found in a survey by Schumaker [70]. Mathematical descriptions can be found in a paper by Meinguet [59].

2.2.1 Use of Thin Plate Equations in Geology

One of the early attempts at applying the thin plate idea to geological problems was by Briggs [9] who used the notion of minimum curvature to approximate contours from scattered data; Equation 2.4 was taken from this paper. Briggs's work lead Swain to implement a FORTRAN version in 1976 [77]. Franke [31] also applied the thin plate interpolation on scattered data. Dyn [23] described how a bell-shaped basis function can be used to smooth thin plate surfaces, but did not show any results. However, Enriquez et al [25] showed that thin plate interpolation produces maps of high quality if the data are nonequispaced.

Some time later, Sandwell [69] proposed a simpler algorithm, using Green's functions instead of bicubic spline interpolation. His claim is that the method still finds a surface with minimum curvature but is more flexible because slope measurements can be used instead of elevation data. The method has limitations, however; Sandwell notes that it is not particularly efficient. More importantly, the method becomes numerically unstable the larger the ratio of the greatest distance between two points to the least distance between two points becomes. This is a severe limitation, which Sandwell dealt with by ignoring closely spaced points. This may not be a viable solution if such points are accurate and should be included in the solution. This problem may not be as important in Sandwell's case, as he used the method to interpolate satellite data where, because of the nature of such data, ignoring a few data points would not impact the results in a substantial way.

A more recent development is the addition of tension to the biharmonic equation by Smith and Wessel [77]. One problem with the direct application of the minimum curvature method is that Gibbs phenomena may be observed between some data points. Smith and Wessel counteract this by adding tension to the ends of the thin plate, thus "flattening" the surface somewhat. The resulting equation is

$$E = \int_R \int \left(T_{xx} \frac{\partial^2 f}{\partial x^2} + 2T_{xy} \frac{\partial^2 f}{\partial x \partial y} + T_{yy} \frac{\partial^2 f}{\partial y^2} \right) dx \; dy \tag{2.8}$$

where T_{xx}, T_{xy}, and T_{yy} represent horizontal forces per unit vertical length.

The tension parameter has several advantages: it can be adjusted to satisfy

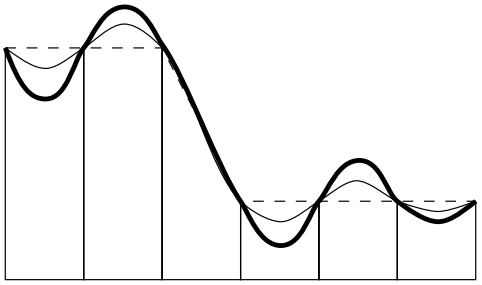

Figure 2.3: Profile showing surfaces with no tension (dark line), some tension (light line), and infinite tension (dashed line).

the user's requirements, the computed surface still honors the observed data, and it is very easy to implement as it is simply changing the weight of four of the nodes in the computational neighborhood, shown as the underlined values:

$$\begin{array}{ccccc} & & 1 & & \\ & 2 & \underline{-8} & 2 & \\ 1 & \underline{-8} & 20 & \underline{-8} & 1 \\ & 2 & \underline{-8} & 2 & \\ & & 1 & & \end{array}$$

The result is that unwanted inflections between sharp elevation changes are minimized, at the expense of localization of the curvature around the data points. This can be seen in Figure 2.3. It was found that this method is helpful in certain situations, such as bathymetric data that contains shelf breaks. The bulge (due to Gibbs phenomena) near the top of the shelf is significantly reduced [77].

Most recently, Powell [64] discusses computational difficulties in interpolating many isolated points and presents an iterative method for up to 10^5 points. The rather small number of points is a major shortcoming of the method.

2.2.2 Use of Thin Plate Equations in Machine Vision

At the same time that research was on-going in the earth-sciences community, similar, relevant work was being done in the area of machine vision and surface reconstruction; see the survey paper of Bolle and Vemuri [4] for a discussion of some of the various methods. Although many results seem to overlap those described above, there appears to be no direct link between the two groups. Much work in

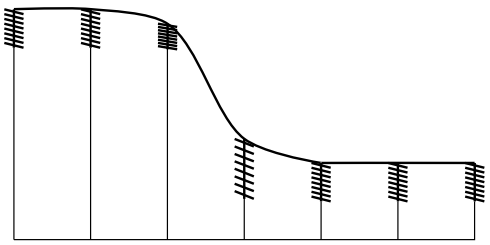

Figure 2.4: Profile showing surface with springs

the vision area was done by Grimson [40, 41] in the early 80's. He presented a theory of visual surface interpolation given stereo range data. He minimized the "quadratic variation" of the surface; this quadratic variation is exactly the thin plate equation, which he solved in a similar manner to Briggs but which resulted in different coefficients [41]:

$$\begin{array}{ccccc} & & 2 & & \\ & 4 & -16 & 4 & \\ 2 & -16 & 40 & -16 & 2 \\ & 4 & -16 & 4 & \\ & & 2 & & \end{array}$$

A surface is computed by using his "gradient projection algorithm," which, for each iteration, computes a scalar value which is added to the current surface approximation.

Although Grimson's algorithm produces a thin plate surface, it is computationally slow and does not handle discontinuities. In [80], Terzopoulos addresses the efficiency problems, by using the multigrid approach to solve essentially the same biharmonic equation as Grimson. However, a smoothness term is added to the quadratic functional, as shown as the β term in Equation 2.5. This term models "springs" at the top of the elevation values, allowing the thin plate to bend in a more natural way, reducing Gibbs effects but creating an approximation instead of interpolation. Figure 2.4 shows how the springs influence the thin plate near a depth discontinuity.

Terzopoulos continued his research in this area with two more works [81, 82]. In these later papers, he addresses the problems of depth and orientation discontinuities

in detail. In simplest terms, he gives a three-fold solution. In areas where there are no discontinuities, the problem reverts to the normal thin plate solution. Where there are depth discontinuities in the surface, defined as occluding contours, the idea is to "break" the plate at the discontinuity, resulting in a piecewise continuous solution. Finally, orientation discontinuities, defined as surface creases, are handled by substituting a flexible membrane for the thin plate. The membrane allows for more bending in such areas. The controlled-continuity stabilizer given in [82] is

$$\begin{aligned} \epsilon(v) &= \int_R \int \rho(x,y) \Big\{ \tau(x,y)(v_{xx}^2 + 2v_{xy}^2 + v_{yy}^2) \\ &\quad + [1 - \tau(x,y)](v_x^2 + v_y^2) \Big\} \, dx \; dy \end{aligned} \tag{2.9}$$

where $\rho(x,y)$ and $\tau(x,y)$ are continuity control functions, the former for depth discontinuities and the latter for orientation discontinuities. The term v_x^2 represents $\left(\frac{\partial v}{\partial x}\right)^2$. By varying the values of ρ and τ, the surface can behave as a membrane at one extreme or as a thin plate in complete tension at the other extreme (the tension parameter is very similar to that reported in [77] later). The surface therefore has differing orders of continuity depending on the presence of discontinuities. A major problem of this approach is that the discontinuities must be known *a priori*. Terzopoulos handles this by comparing data from multiple sensors [4]. Such data is typically not available for the terrain modeler in GIS.

Jou and Bovik [50] point out that multigrid techniques, already quite difficult, become more so when discontinuities are accounted for. More information must propagate between the different grid levels, minimizing performance gains. They suggest using constraint expansion to compute a much improved initial surface while accounting for discontinuities. A better initial approximation can yield significant performance increases. Furthermore, detected intensity edges and the square Laplacian are used to locate discontinuities in the initial surface.

Another approach that deals with the discontinuity problem is given by Sinha and Schunck [48, 74, 75]. They use a two-stage process, where a surface g is first constructed based on the given data. Then g is used to generate regular data which

is used to construct the final surface f [70]. They use a moving least median squares regression (MLMS) method on scattered data to complete the first stage. To find the final surface, they employ a weighted bicubic spline:

$$E = \int_R \int \omega(x,y) \left(f_{xx}^2 + 2f_{xy}^2 + f_{yy}^2 \right) dx\ dy \tag{2.10}$$

where

$$\omega(x,y) = \frac{\beta^2}{1 + \|\rho(x,y)\|^2} \tag{2.11}$$

The smoothing term β is the same as in Equation 2.5, while the value of $\rho(x,y)$ is the gradient, making the weight ω adaptive; it is large when the data is flat (small gradient), allowing the surface to be smoother. In steeper regions (large gradient), ω becomes large, which corresponds to a surface with many steep bends and pitches. The weight is found in the first stage of processing, so efficiency in the second, more computationally intensive stage, is not compromised. Obviously, this too is a surface approximation rather than interpolation because this method allows the surface to deviate from the observed data points.

A two-stage approach that also uses thin plate splines is described by Fang and Gossard [27]. The first stage is the formation of a triangular web over scattered data. Boundary and characteristic points must be known beforehand. A thin plate approximation fills in the surface between each of the edges of the triangles.

2.3 Results Using Previous Thin Plate Methods

Although the methods described in Section 2.2 may produce adequate results in some instances, they are not deemed sufficient for computing surfaces from contour data. Except for one test case done by Terzopoulos in [80], all of the methods above used scattered data for their input. However, when using contour line data as input, the basic thin plate approach is not adequate because of the terracing effect that occurs due to the large discrepancy between the number of data points at one elevation and the much lower number of points at the next higher elevation.

For example, consider the synthetic contour map shown in Figure 2.5, a 257×257 raster file containing contours with 20 unit intervals. The normal thin plate interpolation method produces the surface shown in Figure 2.6. One can see easily the terraces between contours and some Gibbs phenomena near the highest contour of the smaller hill. The small, vertical "ripples" in the steep area are the result of digitization errors due to aliasing of the contours' curves. The thin plate method simply can not smooth this out because there are too many observed values which are not allowed to deviate from their initial values.

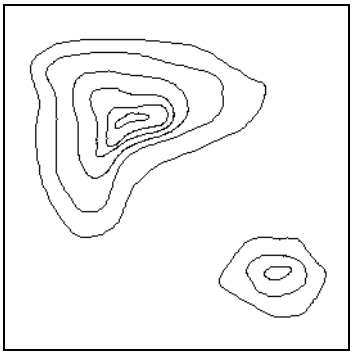

Figure 2.5: A synthetic contour map with interval = 20 units

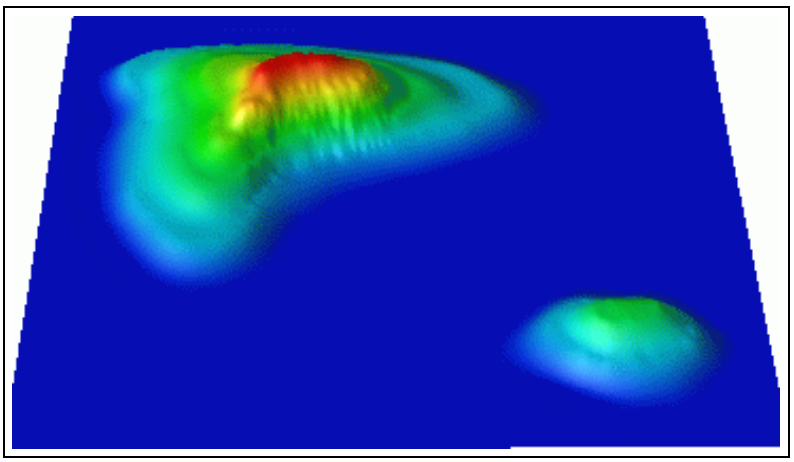

Figure 2.6: Normal thin plate surface interpolation

To reduce the terracing effect, one may allow the surface to deviate slightly from the observed values. If care is taken, the resulting surface fit is close to the contour elevations, and the terracing effect is diminished somewhat. Note also that this method greatly reduces the problems due to digitization errors; the steep sec-

tions are now smooth. Allowing $\beta = 0.5$, which translates to allowing the observed value to deviate by one-half the distance to its newly computed value, the resulting surface for the synthetic data is shown in Figure 2.7. Although the surface is much smoother than the strict interpolation shown previously, some of the details are lost. Furthermore, the method has less effect as the distance between consecutive contours increases.

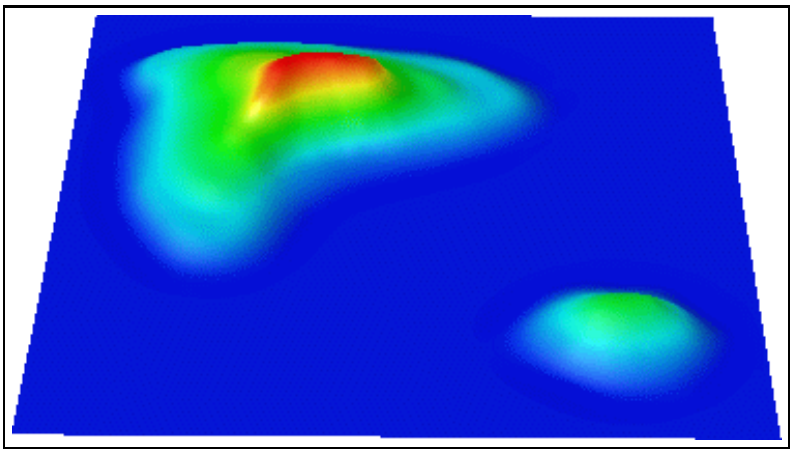

Figure 2.7: Thin plate surface approximation with $\beta = 0.5$

The use of tension has been shown to alleviate some types of terracing problems. It also affords a surface interpolation rather than an approximation. However, tension tends to flatten the tops of peaks and sometimes actually accentuates the terracing problem. The surface resulting from using tension is shown in Figure 2.8. As predicted, the tops of both hills are flat and the contours are easily seen. As the tension is increased, these effects become even more noticeable. Lastly, in areas where consecutive contours are convex, the surface actually tends to be pulled inward, towards the interior of the contours, much like a cloth object would be if stretched taught over two curved shapes.

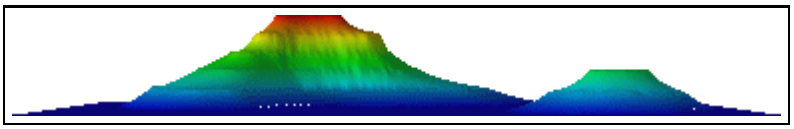

Figure 2.8: Thin plate surface interpolation with tension

2.4 Review of Non-PDF Surface Reconstruction Techniques

There are many other interpolation and approximation methods available for creating terrain surfaces. Whereas the previous section detailed interpolation and approximation techniques based on the physical model of a thin plate, there are many procedures which may be called "manual methods," in the sense that they arise from processes developed by cartographers [85]. As these methods evolved, the computer allowed more data to be used in the reconstructions and more complex procedures to be implemented. Other techniques come from the disciplines of Computational Geometry and Computer Graphics. The survey by Schumaker [70] gives a description of many interpolation and approximation schemes using scattered data. Gold [37] describes methods for use in GIS, as does the paper of Weibel and Heller [86]. The latter also notes some of the problems specific to contour data, and gives a very brief summary of some of the more successful methods. They also note that no interpolation method can be called "best" in all cases. Finally, Franke [30] tested some of the earlier interpolation methods.

Jones [49] outlines several methods for contouring geologic surfaces. The easiest way to calculate a node value is to compute the weighted average of its neighbors. However, this simplistic solution can lead to terracing and furthermore can not compute an elevation that is greater or less than the greatest or smallest data elevations, respectively. Computing peaks from contours is obviously an impossibility. Another method discussed by Jones is one in which functions are fit to selected points, thereby incorporating data trends. New elevations are interpolated from the function at the desired location.

Several approaches rely on first finding a set of data points near the location in question. These natural neighbors are found by Sibson's methods [73]. A simple, popular interpolation using natural neighbors and found in many GIS packages is inverse distance weighting. As described by Watson [85] and Heine [44], the elevation at a particular point p is found by averaging its neighbors, where each neighboring value is weighted inversely proportional to its squared distance from p. For $N = n \times n$ values:

$$F(x,y) = \sum_{i=1}^{n}\sum_{j=1}^{n} \frac{f(x_i,y_j)/d_{i,j}}{\sum_{i=1}^{n}\sum_{j=1}^{n} 1/d_{i,j}} \tag{2.12}$$

In practice, the distances are often squared. For practical and accuracy reasons, the number of data points considered in the summation is usually bounded by some neighborhood window around p [44].

There are several problems to this approach. The surfaces will be discontinuous at data points which tend to form peaks or valleys. Furthermore, the method suffers because of clustering effects. Consider Figure 2.9. It shows two consecutive contours, A and B. The point to be interpolated is shown as p. Because there are more data values along contour A that are closer to p, the interpolated elevation of point p will be biased towards the elevation of that contour [44]. An addition to the basic inverse-distance weighting method is to divide the window into four quadrants [44] with p in the center, as shown in Figure 2.10. An equal number of points from each quadrant is chosen to calculate the value of p, reducing the clustering effect. However, because contours can cross several quadrants, as shown in the figure, the calculated elevation is still prone to error.

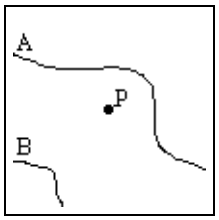

Figure 2.9: Example of clustering problem

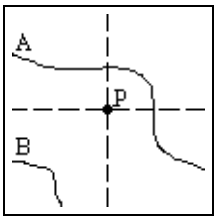

Figure 2.10: Quadrants added to reduce clustering effect

A more complicated weighted averaging method is described by Franke [32]. Weighted local approximations are described by:

$$F(x,y) = \frac{\sum_{k=1}^{n} W_k(x,y)Q_k(x,y)}{\sum_{k=1}^{n} W_k(x,y)} \tag{2.13}$$

where W is a weight function depending on distance and Q is a quadratic

function which must pass through the point. Together, these functions are called the Modified Quadratic Shepard's Method (Schumaker [70] described Shepard's Method as well). Franke added conditions that allow for discontinuities in the surface by detecting faults or creases in the data.

A similar but more robust technique is kriging, where the weights of the neighbors are determined by statistical methods, using distributions found in the sample. This is a method with a long history, dating back to a paper by Krige in 1951. There are many references that describe the method, among them David [18], Jones [49], Dubrule [21], and Oliver and Webster [61]. It is used in several GIS packages [88], including ARC/INFO [26] and DOS SURFER [79], both of which suffer from certain parameter constraints, GEO-EAS and GEOPACK, both from the EPA, and GSLIB from Stanford. The basic formulation is:

$$F(x,y) = \sum_{i=1}^{n} \sum_{j=1}^{n} \lambda_{i,j} f(x_i, y_j) \tag{2.14}$$

where $\lambda_{i,j}$ are the weights and are chosen such that they sum to one and, additionally, minimize the estimation variance. Dubrule notes that kriging and thin plate spline interpolation are essentially equivalent, the difference being that kriging adds the covariance and a degree of trend to the equation. Thus, kriging should be a better interpolation method because it forces the computed surface to more accurately follow trends in the data, a hypothesis that is supported by [21]. This immediately brought about a letter by Philip and Watson [63] who claim that Dubrule's data inherently made the kriging surface more accurate and that the method of kriging, in general, is *not* a more accurate method than some spline methods. Dubrule responded with [22], in which he writes that he was only illustrating that kriging can be better. An interesting excerpt from page 729 is:

> Kriging, by minimizing the estimation variance, is designed to provide estimates which are as close as possible to the actual values. Spline interpolation, by minimizing the total curvature, is designed to provide maps which have nice cosmetic properties.

As discussed in previous chapters, it is obvious that splines alone are not sufficient in creating good surfaces. Similarly, Philip and Watson show that kriging is not always the best interpolating method either. In any case, none of the methods were tested with contour data, which, as shown before, introduce different problems to interpolation methods.

Just as kriging is theoretically better than minimum curvature methods because more information is used in the interpolation, so too is the case with a blended gradient method that Watson espouses [85] because it uses gradient information in addition to elevation data. This method is based partly on a neighborhood-based linear interpolation. A linear interpolation of data points does not preserve continuity across observed values; rather, peaks or pits are formed around them. In order to smooth the surface, the gradients are also computed for each of the neighborhood points. Since there is no surface from which to compute the gradients, these must be estimated using any of the methods shown in [85]. The final surface is found by "blending" the interpolated elevation values and the gradient estimates as a Boolean sum[1]. A Boolean sum is defined as the sum of the elevation values and projected gradients minus their difference. The inclusion of the gradients allows the surface to better follow the slope near the data points, which allows the surface to be more continuous around such points. Much like kriging, the resulting surface is accurate and stable because it is not dependent on only one data set.

Another area of research involves the triangulation of the surface. Keppel [51] triangulates between successive contours to compute a three-dimensional surface. The correctness of the surface depends on the number of triangulation points taken from each of the contours. Furthermore, there is no provision for additional triangulation between contours, which results in planar patches joining each contour. There is also no provision for preserving continuity at the contour itself, yielding abrupt changes in slope at each contour. Christensen [15] triangulates contours to create new elevation lines in between existing contours. This is similar to the central idea of Intermediate Contours, explained in Chapter 3. In Christensen's approach, however, several assumptions are made which may not be viable if real contour data

[1]Watson's term in [85].

is processed. One of the assumptions is that all input contours are distinct and without gaps. Because of digitizing errors, contours may blend together or have other properties that may make it difficult to distinguish one contour from another. Adjacency relationships are also assumed, which may also be difficult to ascertain. Perhaps problems such as these precluded the author from showing any results from large portions of a real contour map.

A Triangulated Irregular Network (TIN) is a system designed by Peuker et al [62] and first implemented by Franklin [33] for DEMs that avoids the redundancies of the rectangular grid method. The TIN is a terrain model employing triangular facets perhaps based on a Delaunay triangulation of the observation points. Because there is no notion of "fineness," a TIN can be made to be more accurate in areas of complex relief, such as ridge or stream lines. A TIN can be created using Franklin's software [34].

Agishtein and Migdal [3] show that Gaussian interpolation is a viable technique, but only if local data is used. Thus, they break the surface into triangles to form a TIN, and then interpolate. Visually, however, their results seem rather unnatural. Garcia et al [36] also used a TIN on contour data, but found that triangles are sometimes horizontal if all three vertices are on the same contour. This technique is used in the ARC/INFO TIN module [88]. A related method is to compute an interpolation using a Voronoi diagram [38]. A test point is inserted into the diagram, and the areas its Voronoi polygon steals from its neighbors are used to weight those neighbors' elevations. The method reduces some discontinuities.

There are few algorithms that explicitly use contour lines as input. The simplest methods find the average between two linearly interpolated profile lines, one profile oriented N-S, the other E-W [67]. However, these orthogonal linear interpolation methods often lead to overestimation because of the same problems that lead to terracing when using thin plate solutions.

Yoeli [90] described a similar method in which he runs four profiles from the desired point until an intersection with a contour line occurs. The distance from the original point is then used as a weight to compute the desired elevation. This method fails in peak or shallow areas, because the computed average can not be

above the highest contour elevation or below the lowest contour elevation.

A better method, called the sequential steepest slope algorithm, initiates an octant search to find the line of steepest descent through the point in question and neighboring contour lines. The final linear interpolation takes place on the selected line, using the elevations found on the two closest contours which intersect the line of steepest descent [54]. Although this method produced adequate results in most cases, additional information is necessary in some valley and ridge areas. If peak elevations are not given, such a linear interpolation is unsufficient, yielding a flat surface to the top of a hill. In such cases, additional processing is required. Clarke, Gruen, and Loon [16] describe a similar method but which instead computes a cubic interpolation along the line of steepest slope. The additional elevation points can improve peak areas that do not include spot heights. Overall, however, they report similar results from the two methods.

The aforementioned algorithms require the direction of steepest slope. The slope is often computed using the spatial derivative as described by Carter [12]. The equation is:

$$Slope^0 = ArcTan\sqrt{\left(\frac{\Delta Z_x}{\Delta X}\right)^2 + \left(\frac{\Delta Z_y}{\Delta Y}\right)^2} \tag{2.15}$$

Although seemingly straightforward, the problem is choosing the Δy (rise) and Δx (run) values. In using gridded data, a natural method is to choose the rise and run values in each of the eight directions. The steepest slope can then be found easily.

One method that fits splines to contours is described by Dierckx et al [19]. They sample the original contours to generate a set of data points which are used as knots in their splines. Interpolating splines are computed from each closed contour and fit to the knots in conjunction with a parameter that determines the closeness of the fit. This method may be difficult to apply to geographic problems because map contours are very large and may not be closed within the area in question. Furthermore, the method does not use all of the contour data points, choosing only a small set as knots for the splines. This will create a surface that, in some areas, be far removed from the underlying data.

Huber [46] describes a method to improve a contour to DEM algorithm by detecting ridges and valleys. Once detected, additional elevation points are interpolated along the ridges or valleys and added to the original data set. The contention is that these points significantly improve the surface interpolation using a wide range of algorithms. However, Huber does not show any resulting surfaces.

Another approach to the interpolation problem is to find an equation that will model the entire surface. Hardy [43] uses this approach, finding an equation from contour data by choosing specific important points such as local maxima, local minima, saddle points, and the like. Although the method may work for simple surfaces, in most cases the approach will find an equation that models the surface poorly because of its generality; most of the original data is ignored. As shown in [43], the derived surfaces show much less detail than the original contour maps, and in many cases, show erroneous contours.

2.5 Summary

While there are many interpolationa and approximation techniques, the computation of accurate surfaces is still an unresolved problem. In general, PDE approaches produce smooth surfaces, but have a tendency to produce terraces. If the terraces are reduced, often other errors are introduced. PDE methods also can be quite slow. On the other hand, PDE methods are based on theory, such as the bending properties of a thin plate. Non-PDE methods are sometimes more ad-hoc, such as inverse-distance weighting or similar weighted-average techniques. As such, they produce generally poor surfaces. Of course there are some non-PDE methods that are more theoretically grounded, such as kriging, which is grounded in statistics. No matter what the method, however, there are very few examples of any technique applied to contour data, which seems odd because of the abundance of such data. Furthermore, in the few cases that do use contour data, there is almost a complete lack of analysis of the resulting surfaces based on quantitative methods and especially comparisons to published USGS DEM data. Without such analyses, it is difficult to establish the merits of many of the aforementioned techniques.

CHAPTER 3

The answer is either m or something else.

– Eva Ma

Thin Plate Surfaces with Intermediate Contours

In Section 2.1.3, we discussed the terracing problem associated with thin plate surfaces. The problem is shown graphically in Figure 2.2. A solution to this problem using thin plate techniques is the subject of this chapter.

3.1 Intermediate Contours

Intuitively, a surface reconstruction algorithm should produce a better surface when there exist more points in the initial data set. This notion is questioned by Eklundh and Mårtensson in [24] in which they show that additional data often adversely affect DEM construction techniques. However, their work focused on the digitizing process applied to contours which were to be used to create DEMs. They report operator errors in this process, yielding contour maps with deficiencies. Of course, these deficiencies are passed to the DEM that is created from such contour maps. However, if additional accurate data can be generated in a contour map through more controlled, programmatic means, then the newly introduced data values should help in producing a better DEM. This is the idea behind the notion of computing intermediate contours. The intermediate contours are generated from existing data before the thin plate surface is computed. The process described in this chapter is shown in Figure 3.1.

A property of contour lines is that, in general, successive lines run approximately parallel to one another. There are several explanations for this:

- We are using contour maps that depict geographic locations. Out of necessity,

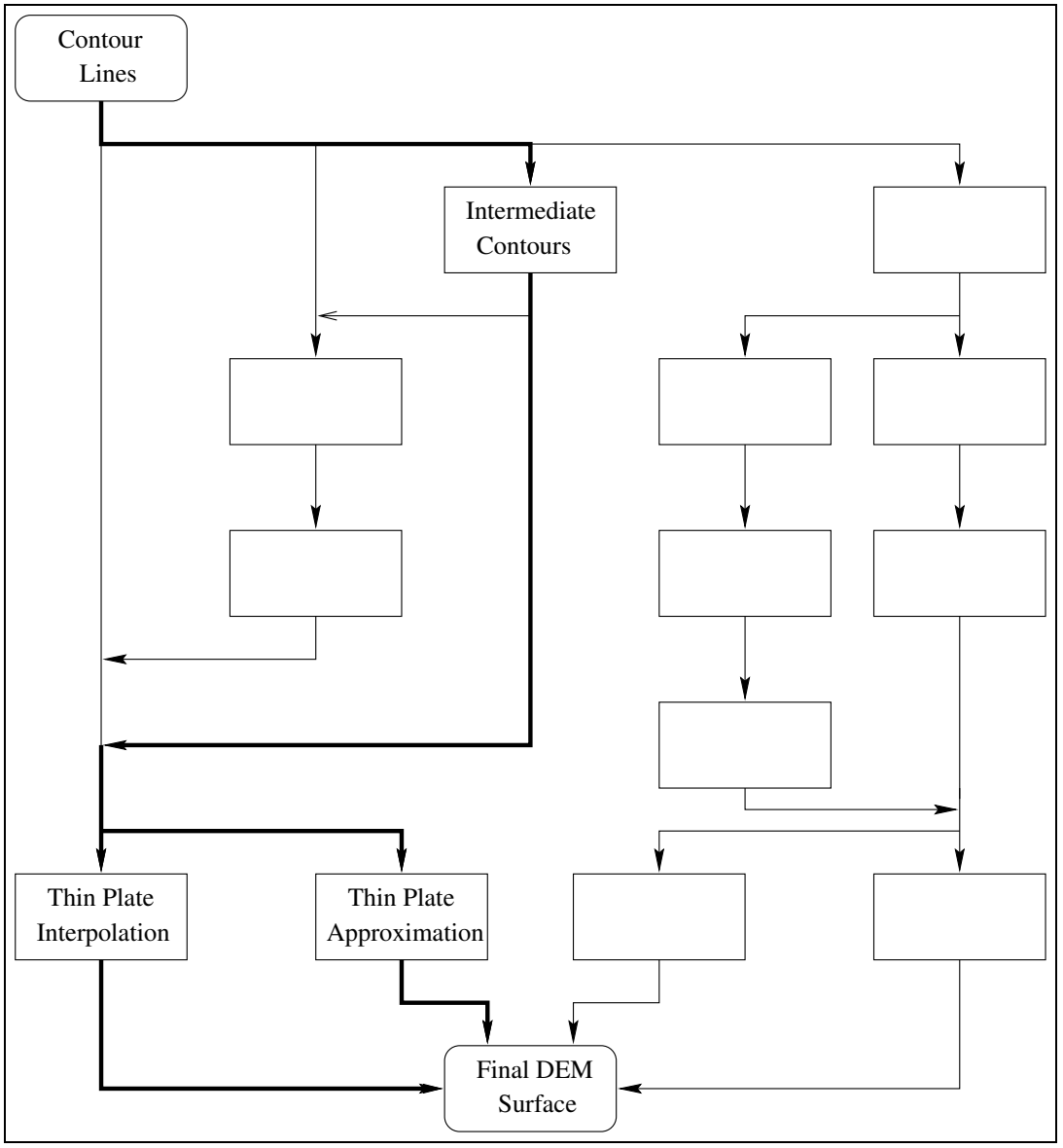

Figure 3.1: Flowchart showing path of Intermediate Contours method(dark line)

such maps are large scale (compared to a contour map of a human heart as is used in medical imaging, for example). Maps typically have contour intervals of between 20 and 100 feet. In such scales, small changes in local topography, can not be adequately represented. It is impossible to find a large boulder or small 15 foot cliff on a map.

- We presently have no means to perform enough precise measurements to com-

pletely model geographic locations. It is therefore impossible to show every nuance in a contour map. Cartographers must infer much of the topography of a region from a scant number of actual measurements. They therefore use a reliable contour to approximate neighboring contours which creates contours that are parallel.

- Although the scale of most maps is fairly large, the contour interval is sufficiently small to model larger phenomena such as large cliffs, canyons, and bowls. Such natural phenomena usually do not occur so abruptly as to change two neighboring contours radically. Even cliffs, for example, usually have a slope curving upwards at the bottom so that contours depict a hill and then gradually form the shape of the cliff as the elevation increases.

Of course, there are instances when successive contours change radically. The bottom of a canyon may be flat with cliff faces rising steeply in many angles. However, only the bottom-most contour of the cliff is very different from the contour of the canyon floor; the next higher contour will look very similar to the first. Similar phenomena can occur at the tops of rounded mountains or hills where the side suddenly falls away.

Because contours run relatively parallel to one another, we can use the information contained in consecutive contours to create a new contour line in between the two original, observed contours. The new contour line has the property that each point on the line is equidistant from the closest point of each of the two neighboring contour lines. Furthermore, the intermediate contour line has an elevation value that is exactly midway between the elevations of the contours on either side. This is similar to what a human cartographer would do when drafting a contour map.

The intermediate contour can give a good approximation of the data values between observed contours. We explained in Chapter 2 that thin plate methods do not work well when contours are spaced far apart, creating terraces in the process. Intermediate contours can be especially helpful in such a situation, filling in data points where there were none previously. The process can be repeated using newly computed intermediate contours as observed data if it is decided that more can be

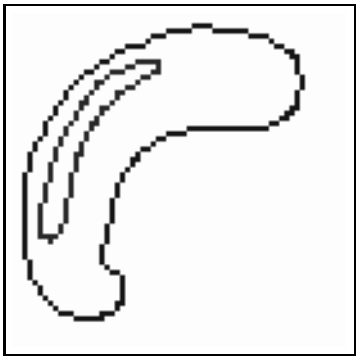

Figure 3.2: A "bean" contour file

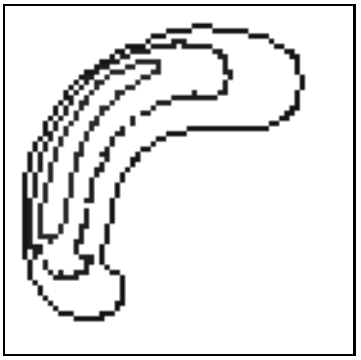

Figure 3.3: Bean with additional intermediate contour

gained by generating even more elevations. Once all of the intermediate contours are obtained, the resulting data set can be interpolated or approximated using the normal thin plate approach or a combination of those cited in Chapter 2. An additional benefit of creating intermediate contours is that convergence of the thin plate methods may occur sooner, thus reducing total computation time.

Consider Figure 3.2. It depicts two contours which form a hill in the shape of a bean. This kind of shape is troublesome to minimum curvature methods because of the large spaces at each end. Intermediate contours can help significantly because elevation values are found in the areas where there were no data points whatsoever, as shown in Figure 3.3.

The key observation to finding a point equidistant from two consecutive contour lines is that the steepest slope from any point on a contour is defined to be perpendicular to the tangent at that point. We compute a good approximation to the perpendicular at a given point by finding the closest neighboring point on the next higher or lower contour. A line is created to join the two points on the consecutive contours, and the midpoint is found easily.

Computing an intermediate contour is done through the following steps:

1. Choose a point P_1 from one contour line A
2. Find the closest point P_2 on contour line B s.t. $B_{elev} > A_{elev}$
3. Determine the midpoint P_{mid} between P_1 and P_2

4. Calculate elevation: $P_{mid}^{elev} = \frac{1}{2}(P_1^{elev} + P_2^{elev})$

We wish to find the closest neighboring contour point such that there is a clear path from the original point to the newly found position. This puts additional constraints on finding the closest point on a higher contour. As the search progresses, if a point with the same elevation is encountered, that is, a point on the same contour line from which we started, the search is terminated in that direction. Similarly, the search is also terminated in the appropriate direction if a contour of lower elevation is encountered.

We employ Bresenham's circle algorithm [29] to find the closest point P_2 from P_1, although this may not be the optimal search technique. From P_1, circles with successively larger radii are generated until the circle contacts a point P_2 which has an elevation value higher than P_1's. The circular search is then terminated and the midpoint computed. In the case that there does not exist a higher contour line nearby, the maximum distance to search can be estimated by computing the average distance between contour lines when the data is initially read in. Bresenham's algorithm only computes one-eighth of the circle, the other arcs being mirror images. If an invalid contour is crossed (one of the additional constraints), the search is discontinued for the arc of the circle in which the point is found. Other methods can be used in place of Bresenham's algorithm to find the closest neighboring point, but it may be difficult to enforce the above constraints. Voronoi diagrams were considered, but the modifications to the algorithm to find points with differing elevations proved very complicated.

The entire algorithm can be repeated using the newly computed contour as observed data lines in successive iterations.

3.2 Problems with Intermediate Contour Generation

A problem arises when using data taken from the natural world. Although successive contours, in general, are parallel, there arise situations where they are not. Figure 3.4 shows one such situation. If we assume that the contour on the right represents a lower elevation, then the algorithm described above behaves predictably and produces an intermediate contour as shown in Figure 3.7. However, if the

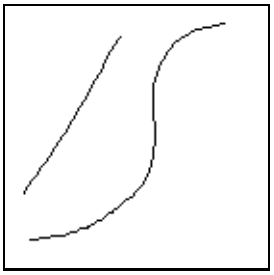

Figure 3.4: Two non-parallel contour lines

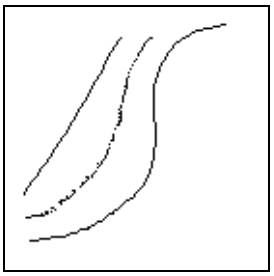

Figure 3.5: Initial intermediate contour; right contour is lower

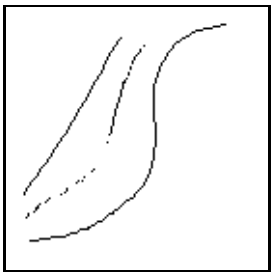

Figure 3.6: Initial intermediate contour; right contour is higher

elevation of the right contour is higher than the left, then the situation shown in Figure 3.8 is produced. Clearly, a good intermediate contour is not computed by the algorithm in this case.

The problem shown in Figure 3.7 can be summarized as follows. In most situations, contours are convex; that is, they curve towards the middle of the hill or mountain. However, Figure 3.8 shows a situation which is similar to a bowl or ravine. In such cases, the curve of the contour is concave, curving outward from the center of the mountain. Finding the intermediate contour by following our algorithm is not sufficient for such areas. In these cases, the intermediate contour can be found after finding the farthest higher point, or, conversely, the closest lower elevation, instead of finding the closest higher contour as was described previously. We employ a second pass of Bresenham's circle algorithm, this time searching for the closest lower contour, to get an approximation of the intermediate contour. The results of this are shown in Figure 3.5 and Figure 3.6. These can be compared to Figures 3.7 and 3.8. Note that the intermediate contour is more complete at the cost of some outliers. The reason for this is that using Bresenham's circle algorithm is not optimal in that it is greedy, terminating as soon as it finds one point. It is

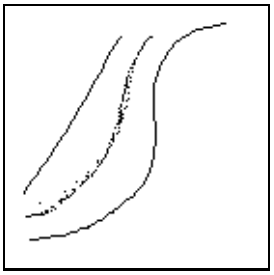

Figure 3.7: Final intermediate contour; right contour is lower

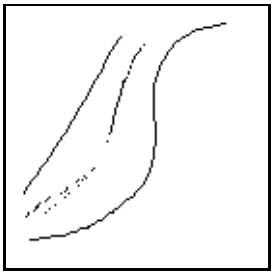

Figure 3.8: Final intermediate contour; right contour is higher

possible, however, that there are additional points that are at the same distance. The direction of the circular search also has an impact on which point is chosen first, which sometimes causes the search from lower to higher contour to have a different midpoint than the search from higher to lower contour. Lastly, a circle on a grid is necessarily not perfectly round, and some pixels must be skipped. One of these skipped pixels may, in fact, be the closest neighbor. Because of these problems, the intermediate contour is viewed as a good *approximation*. In the thin plate processing stage, therefore, these computed points are allowed to change to give us a smooth surface.

3.3 Sample Output

Using the synthetic data from Figure 2.5, two iterations of the intermediate contour method produce contours as shown in Figure 3.9. The method can be run iteratively until the entire surface is computed. However, there may be small gaps in the computed contours because the method assumes that successive, increasing elevation contours have a convex shape, generally circling a local maxima. Thus, in order to reliably fill in any uncomputed areas and to promote a smooth surface, the thin plate approximation is applied to create the final surface. (A different method of filling in the gaps is discussed in Chapter 5.) Figure 3.10 shows the surface computed from the contours shown in Figure 2.5. Because the computed intermediate contours introduce new elevation values into the initial data set, the resulting surface is much more accurate than a thin plate approximation alone.

It is sometimes difficult to ascertain the differences between various reconstruc-

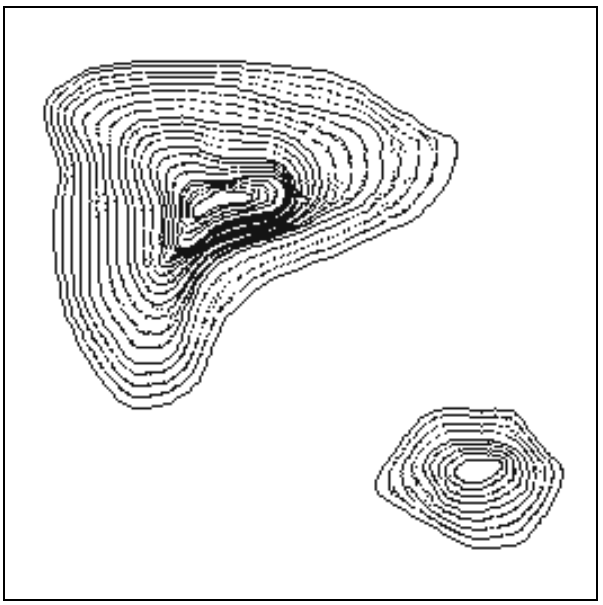

Figure 3.9: Synthetic data with two iterations of intermediate contours

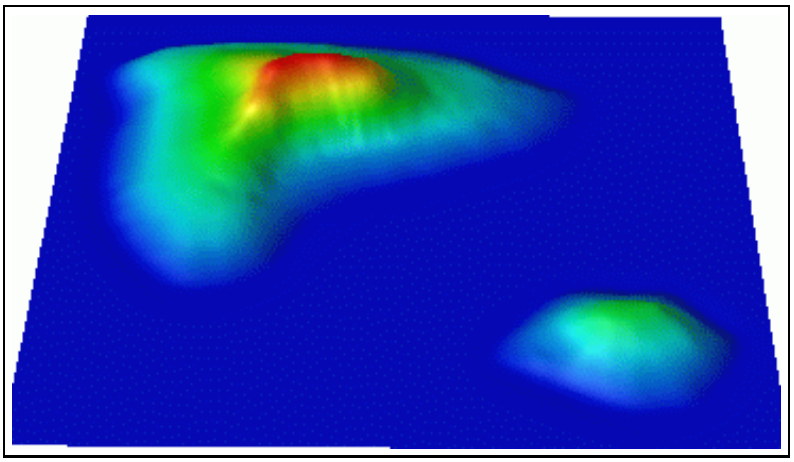

Figure 3.10: Surface approximation using intermediate contours

tion methods using only three-dimensional renderings. Figure 3.11 shows graphically the differences between identical diagonal profiles computed using different methods. A profile computed using the thin plate with springs is compared to the intermediate contour method. The intermediate contour method produces a profile that fits the observed data much better; that is, it does not "sag" between successive contours compared to the thin plate surface.

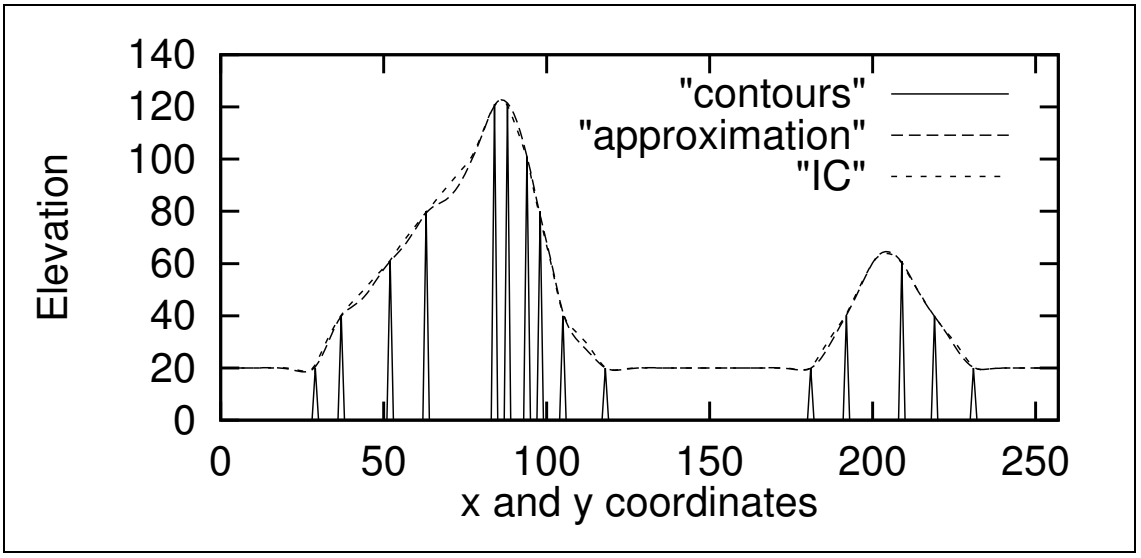

Figure 3.11: Plot of diagonal profile showing results of different reconstruction methods

3.4 Performance of IC Method

The performance of thin plate methods was discussed in Chapter 2. It remains, therefore, to establish the performance of computing intermediate contours. At each grid position p that contains an observed contour value, a circular search is performed to find the closest neighbor. When the neighbor is found, the search terminates. The search also terminates in octants in which a contour with the same elevation as p. A final condition is that the search terminates at some maximum search distance which is defined to be the maximum distance between contours, computed when the data is read in.

We can easily find the upper bound performance of this method, but in practice it is much lower but very difficult to quantify. All of the nodes must be visited, which contributes at most n^2. At each node, we search for a neighboring contour using Bresenham's circle algorithm. The maximum estimate for the search is πr^2, where r is at most $\frac{n}{2}$. This gives us a worst-case time of $O(n^4)$ for one iteration. To completely fill the grid requires $\log n$ iterations, resulting in a total of $O(n^4 \log n)$. This upper bound can be reached only with completely unrealistic input data.

Based on our test data, we can compute a more realistic upper-bound. While it is true that there are n^2 total points in the grid, most of them do not represent a

contour location. Using all of our data files, only a minimum of 3% to a maximum of 8% of the nodes contain elevation data which will need further processing. In our test files the maximum number of elevation values is $m = 7n$, far below n^2. For each of the elevation nodes, a search for a neighbor is done. As before, a maximum estimate for the upper bound is πr^2, where r is at most $\frac{n}{2} = d$, making the final estimate πd^2. This upper bound can only occur in the exact center of the grid and only when there are contours on the very edges of the grid. Alternatively, the worst case occurs when one contour is at one boundary and the another contour is at the opposite boundary. In this case, n pixels must be searched, but only in half the grid, giving us the same estimate. More realistically, we need to search only until another contour is touched, which depends on the distance between contours. In our test data, this distance ranges from a low of 70 pixels (in a file where $n = 257$) to a high of 598 pixels (in a file where $n = 900$). The distance depends on the topology of the area, of course, and can change drastically within the same map. Furthermore, portions of the circular search terminate as soon as one of the conditions outlined above is reached. This is very difficult to quantify, however. The resulting worst-case performance is $m\pi d^2 = O(md^2)$ for one iteration. Filling in the entire grid requires $\log d$ iterations for a total of $O(md^2 \log d)$. It is important to note that after each iteration, the distance between contours is halved, making each subsequent iteration twice as fast as the previous. Furthermore, as the contours become more dense, the value of d decreases to a minimum of 1, which will decrease the upper-bound to $\Omega(n^2)$. Similarly, as the density of the contours decreases, m must decrease while d increases to a maximum of n. The bound in this case $O(n^2 \log n)$.

The total performance is thus the time for the generation of intermediate contours plus the time for the thin plate processing:

$$O(md^2 \log d + n^2 \log n) \approx O(n^2 \log n) \tag{3.1}$$

3.5 Summary

In this chapter, we have described the Intermediate Contours method. It creates new, interpolated contours in between existing contour lines. This, in turn,

provides more data to subsequent thin plate processing, alleviating the terracing problem in most areas. The intermediate contours are computed without any significant additional overhead as compared to the thin plate processing. However, it should be noted that the thin plate processing can be rather time consuming. The addition of intermediate contours may take substantial time as well, especially in larger grids.

CHAPTER 4

The top of Everest, two vertical miles above, seemed so impossibly distant that I tried to limit my thoughts to Camp Two, our destination for the day.

– Jon Krakauer[52]

Creating Surfaces with Gradient Lines

One of the major problems with isoline data is the fact that data is oversampled along a contour line and undersampled across the line [86]. Furthermore, it is known that the steepest slope is orthogonal to the tangent at a point on a contour line. If the direction of steepest slope (aspect) can be found at every point, then a path, or "gradient line," can be found joining a local minima to a local maxima. Using the elevations from the contours that the path crosses, a spline can be fitted along the path. Repeating this method for all points will create an interpolated surface which can then be smoothed as desired. The resulting surface does not show the terracing effects due to oversampling because the gradient lines cross contours orthogonally. Figure 4.1 shows the flow of control when computing DEMs using the Gradient Lines method.

4.1 Computing Gradients

In order to create a gradient line, the gradient must first be found at each point. From this, the aspect can be computed easily [76]. Given a function $z = f(x, y)$, the gradient is defined by the vector [53]:

$$\nabla f(x, y) = f_x(x, y)\mathbf{i} + f_y(x, y)\mathbf{j} \tag{4.1}$$

Although this equation can be solved directly, a more accurate result will be

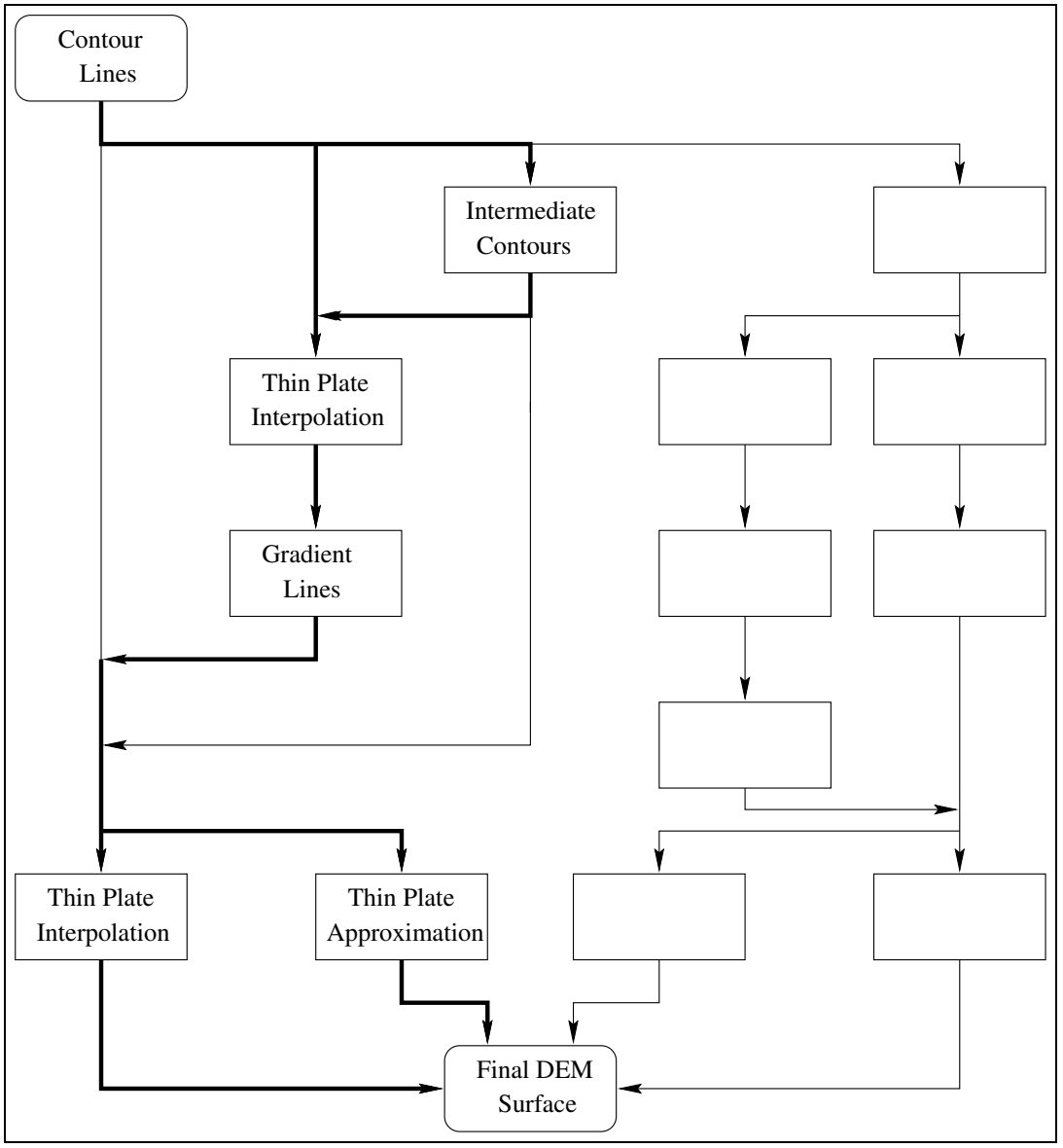

Figure 4.1: Flowchart showing path through gradient lines (dark line)

achieved by differentiating twice. A suitable function is the Laplacian:

$$f(x, y) = f_{xx} + f_{yy} \tag{4.2}$$

Because the Laplacian is a function of x and y, we can substitute it into Equation 4.1 which yields:

$$\nabla f(x, y) = f_x(f_{xx} + f_{yy})\mathbf{i} + f_y(f_{xx} + f_{yy})\mathbf{j} \tag{4.3}$$

One way to solve such an equation is to employ central differences. The three-point central difference equations [39] are:

$$f_{xx} \approx \frac{f(x_{i-1}, y_j) - 2f(x_i, y_j) + f(x_{i+1}, y_j)}{h^2} \tag{4.4}$$

and

$$f_{yy} \approx \frac{f(x_i, y_{j-1}) - 2f(x_i, y_j) + f(x_i, y_{j+1})}{h^2} \tag{4.5}$$

where $h =$ grid spacing.

Because only an approximation to the gradient at any point is desired, the small error term associated with each central difference equation can be safely ignored. Assuming a grid of n^2 points, (x_1, y_1) to (x_n, y_n) and ignoring boundaries s.t. $i, j \in \{3..(n-2)\}$, substituting the central difference equation for each f_{xx} and f_{yy} in the gradient equation yields:[2]

$$\begin{aligned} \nabla f(x,y) &= f_x\Big(\frac{f(x_{i+1}, y_j) + f(x_{i-1}, y_j) + f(x_i, y_{j-1}) + f(x_i, y_{j+1}) - 4f(x_i, y_j)}{h^2}\Big)\mathbf{i} \\ &\quad + \Big[f_y\Big(\frac{f(x_{i+1}, y_j) + f(x_{i-1}, y_j) + f(x_i, y_{j-1}) + f(x_i, y_{j+1})}{h^2}\Big) \\ &\quad - f_y\Big(\frac{4f(x_i, y_j)}{h^2}\Big)\Big]\mathbf{j} \end{aligned} \tag{4.6}$$

Now letting $h = 1$ and expanding,

$$\begin{aligned} &= \Big[f_x(x_{i+1}, y_j) + f_x(x_{i-1}, y_j) + f_x(x_i, y_{j-1}) + f_x(x_i, y_{j+1}) - 4f_x(x_i, y_i)\Big]\mathbf{i} \\ &\quad + \Big[f_y(x_{i+1}, y_j) + f_y(x_{i-1}, y_j) + f_y(x_i, y_{j-1}) + f_y(x_i, y_{j+1}) \\ &\quad - 4f_y(x_i, y_i)\Big]\mathbf{j} \end{aligned} \tag{4.7}$$

The two-point central difference equations [39] are:

[2]The equations for the borders, where $i, j \in \{1, 2, (n-1), n\}$, are similar.

$$f_x \approx \frac{f(x_{i+1}, y_j) - f(x_{i-1}, y_j)}{2h} \tag{4.8}$$

and

$$f_y \approx \frac{f(x_i, y_{j+1}) - f(x_i, y_{j-1})}{2h} \tag{4.9}$$

Substituting the two-point central difference equations into 4.7 and setting h to 1 yields:

$$\begin{aligned} \nabla f(x,y) \;=\; & \frac{1}{2}\Big[f(x_{i+2}, y_j) - f(x_{i-2}, y_j) + f(x_{i+1}, y_{j-1}) - f(x_{i-1}, y_{j-1}) \\ & + f(x_{i+1}, y_{j+1}) - f(x_{i-1}, y_{j+1}) - 4f(x_{i+1}, y_j) + 4f(x_{i-1}, y_j)\Big]\mathbf{i} \\ & + \frac{1}{2}\Big[f(x_{i+1}, y_{j+1}) - f(x_{i+1}, y_{j-1}) + f(x_{i-1}, y_{j+1}) - f(x_{i-1}, y_{j-1}) \\ & - f(x_i, y_{j-2}) + f(x_i, y_{j+2}) - 4f(x_i, y_{j+1}) + 4f(x_i, y_{j-1})\Big]\mathbf{j} \end{aligned} \tag{4.10}$$

Each $f(x_i, y_j)$ represents the elevation at location (x_i, y_j). Thus, the gradient can be computed at any point provided there are valid elevation values found for each neighboring point. Elevation values are estimated by computing an initial thin plate surface; employing intermediate contours can yield a better initial surface and thus produce more accurate gradient values.

4.2 Finding Gradient Lines

Once the gradient is found, a gradient line can be formed. Starting from any grid position P_1, the next point in the gradient line, P_2, is found by moving in the appropriate direction as indicated by the aspect of P_1. Because of the limitations of a regular grid, the actual direction is rounded to one of the eight neighbors of the point. P_1 is then stored and P_2 becomes the starting point to find the next point along the line. When the gradient is found to be zero, a local maxima has been reached and the search terminates. Similarly, the search progresses in the opposite direction until a local minima is reached.

Because each P_i in the line will not lie directly on an observed elevation value, the next step is to find an interpolation of all the unknowns. Each gradient line will pass through some known elevations along contour lines. These elevations can form a basis from which to interpolate by using a one-dimensional spline. The spline used because of its accuracy in passing through each observed data point is the Catmull-Rom spline, which has desirable C^2 continuity [29]. These splines interpolate between two points B and C. Continuity is achieved by using the previous point in the path A and the next point in the path D to guide the interpolation through B and C. Such a spline is computed for each gradient line, resulting in a new "shell" of the surface which is quite accurate, especially along ridge lines. The surface may not be very smooth, however, because there is no direct relationship between individual gradient lines.

Figure 4.2 shows two sets of intermediate contours computed using the synthetic data set. In addition, a portion of the gradient lines can be seen following the steepest slope and crossing contours. This image was generated by computing gradient lines for every tenth grid value instead of every point. It can be seen that the gradient lines join local minima and maxima, ending in local flat areas. Some discontinuities are evident as well; these are the result of inconclusive gradient estimations at certain points. Such gradient values can result when the local initial thin plate surface is flat, such as in an area where a terrace is observed. In order to fill in such gaps and to afford a continuous, smooth surface, the minimum curvature method is employed to produce the final result, as shown in Figure 4.3.

As was done in Chapter 3, a diagonal profile of the synthetic data was generated to compare the results of the Gradient Lines and the thin plate approximation methods. The resulting plot is shown in Figure 4.4.

4.3 Performance

The performance analysis of the Gradient Lines method is, unfortunately, not straightforward. The first stage is to create an initial surface from which to compute gradients. This is done with the thin plate approximation which takes at least $O(n^2 \log n)$. The computation of the gradients is straightforward; it is a

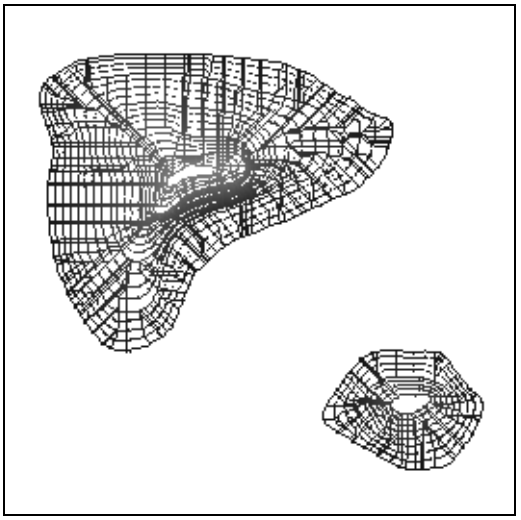

Figure 4.2: Gradient lines and intermediate contours

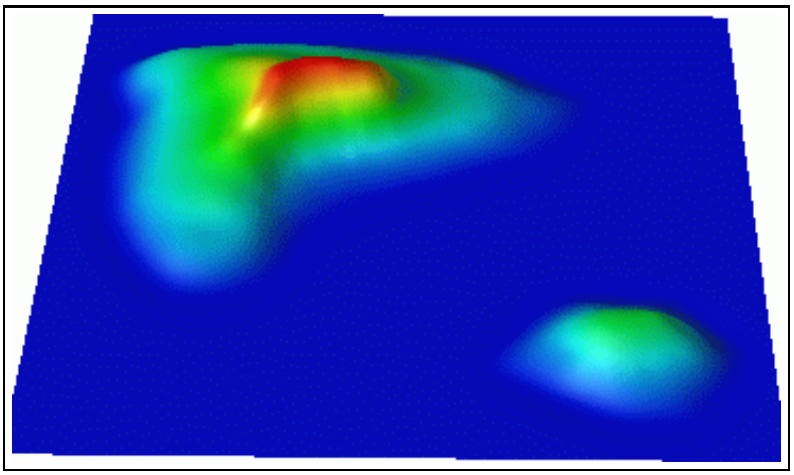

Figure 4.3: Approximated surface using gradient method

constant time operation for each of the grid points for a total of $O(n^2)$. The difficult portion is determining the performance of the calculation of the gradient lines. For each grid point not yet visited, a gradient line is computed. The number of nodes visited along the path is difficult to determine. At most, it is $O(n^2)$, but this can not happen realistically. However, we may assume that a line will reach either from one side to the other of a grid, following either an uphill or downhill trend. Since the path will not be straight, we can make an estimate of O (cn), where c is a small constant. This makes the total gradient lines computation $O(n^3)$. The final step in the method is to smooth the resulting surface, again employing the thin plate

approximation. The complete performance figure is then:

$$O(n^2 \log n + n^3 + n^2 \log n) = O(n^3) \tag{4.11}$$

for a grid of size $n \times n$.

4.4 Summary

In this chapter, we have introduced a new method to alleviate the local terracing problem while creating globally good terrain surfaces. The method involves computing gradient lines which are interpolated paths that follow the steepest slope, an idea similar to what some researchers refer to as "lofting." As such, the method is less arbitrary than, for example, the IC method. The drawbacks to the method are that the gradient computations depend on an initial surface estimate which may not be sufficiently accurate to provide good gradients. A second drawback is that the method increases the computation costs relative to thin plate methods. However, an increase in surface accuracy will usually result.

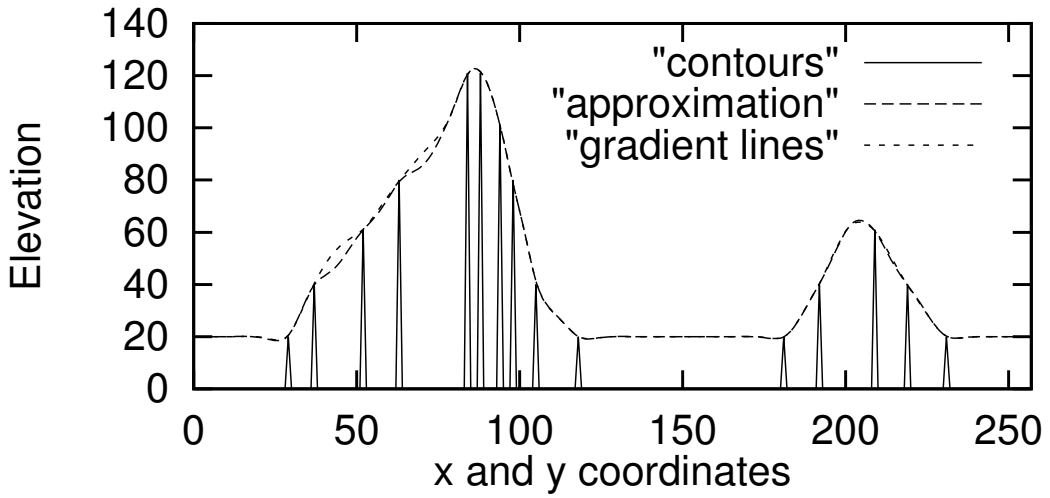

Figure 4.4: Plot of diagonal profile showing results of different reconstruction methods

CHAPTER 5

You can look down in [crevasses] for distances stretching from 100 feet to Hades or China. ...Most of them appear to be bottomless. These are not good things to look at.

– Tom Lloyd, quoted in [71]

Fast Surface Interpolation/Approximation

In this chapter, we discuss two methods which will accurately compute a DEM directly from the initial contours without the need for computationally expensive thin plate methods.

5.1 Maximum Intermediate Contours

In Chapter 3, we explained how intermediate contours can alleviate some of the problems encountered when interpolating or approximating using thin plate methods. We will now show how intermediate contours can be used to create an almost complete surface through iteration. This method is called the Maximum Intermediate Contour (MIC) method. In addition to the intermediate contours, several additional steps are needed to produce the final DEM. The steps are shown in Figure 5.1.

5.1.1 Computing Boundaries

By continually computing intermediate contours, we wish to compute the entire surface. However, if the surface trend is upward toward a boundary, that is, successive contours have higher elevations as they get closer to the edge of the grid, then the last contour in the grid will not have any closest neighbors of greater elevation. The method would not compute any new elevation values from the highest

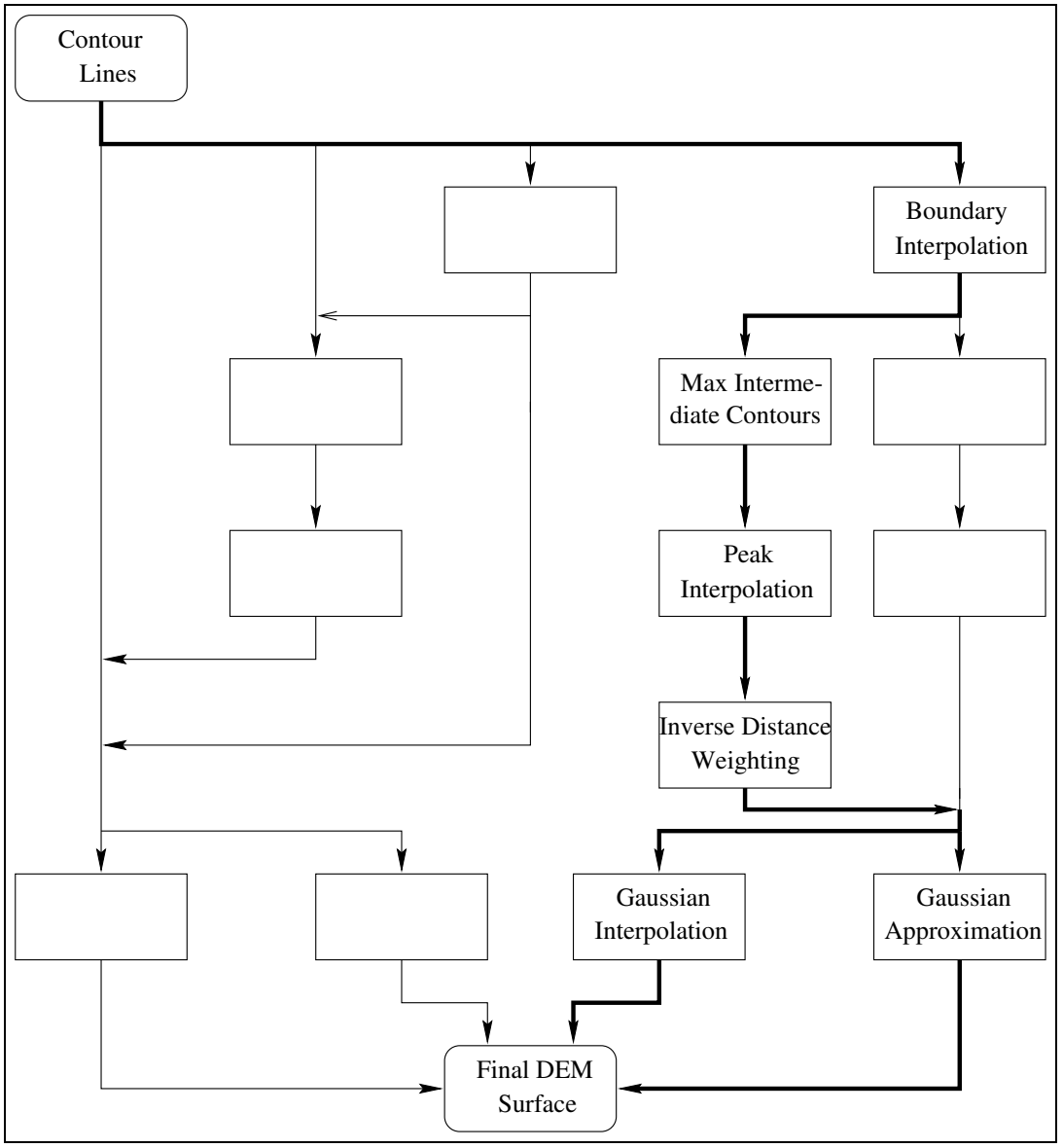

Figure 5.1: Flowchart showing Maximum Intermediate Contours method

contour closest to the edge to the boundary itself. This would leave significant gaps in the surface.

In order to create intermediate contours that extend to the edges of the grid, we compute a boundary rectangle around the perimeter of the grid having a thickness of two pixels.[3] Two cases can occur: If the grid in question is a contour map of an area that is relatively hilly, then there will be some contours that cross the edges of

[3]The reason why *two* pixels are computed will become clear in Section 5.2.

the grid. The solution is to linearly interpolate from one contour to the next along the boundary. The four edges are mapped to a one-dimensional array which is then easily interpolated. That is, for a grid z with n^2 points, an array $Bound$ is created of size $4(n-1)$. $Bound[1]$ to $Bound[n]$ stores the first row of the grid, $z[1][1]$ to $z[1][n]$; $Bound[n+1]$ to $Bound[2n]$ stores the rightmost column of the grid, $z[1][n]$ to $z[n][n]$, and so on for the remainder of the grid in a clockwise fashion. Observed elevation values are noted as such in the array. Values are then interpolated from one observed elevation to the next. Care must be taken to interpolate between the last and first observed elevations in the array. Lastly, the one-dimensional array is mapped back to the two-dimensional grid. The steps are repeated for the second, or inner, boundary defined to be the second pixel from any of the four edges.

In the second case, which is very rare, no contour lines cross the grid edges. If this occurs, the contour with the lowest elevation is found. This elevation is assigned to the two perimeter edges of the grid. While this method is rather arbitrary, the situation arises only if the data is synthetic or if the topology is extremely flat. In both cases, the boundary values should suffice.

5.1.2 Computing Interior Elevations

The generation of intermediate contours was explained in Chapter 3. They are useful for providing additional data for thin plate methods, alleviating the terracing problem. By continually computing intermediate contours, almost the entire surface can be approximated. Figure 5.2 shows the result of applying the maximum intermediate contours to the synthetic contours of Figure 2.5.

Although the surface seems generally good, two problems are immediately apparent. The first is that there are no computed elevations between the boundaries and the lowest contour of both hills. This is due to the fact that both hills have the same elevation for their lowest contours. Since the intermediate contour algorithm looks for contours at the next higher elevation, no new elevations are computed. The second problem that manifests itself is in the small gaps that appear throughout the surface. This problem can actually be broken into two separate situations: peaks and non-peaks. For the same reason that there are no elevations computed between

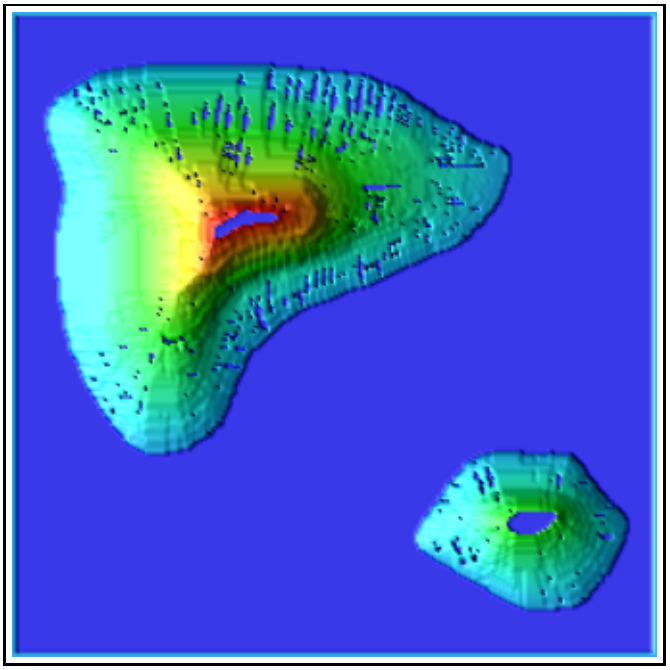

Figure 5.2: Result of applying maximum contours to synthetic data

the boundaries and the lowest contours, there are no values computed for the areas defining the peaks of both hills. This will always occur and is dealt with by one of the methods described in the following sections. The intermediate contour method is also limited by its Bresenham search. The circular search will not always find the closest point because of reasons outlined in Chapter 3. There usually will be small gaps that remain after intermediate contour processing. This is handled by filling in with inverse-distance weighting, described later.

5.1.3 Peak Interpolation

Peak areas are usually relatively small portions of the grid that are bounded by a locally maximum contour. Intermediate contours can not be computed because there are no higher contours to which to compare. The following method also applies to areas between the boundaries and lowest contours of the same elevation as occurs in the synthetic test data.

The elevations in the areas in question are found by computing one-dimensional splines in the horizontal direction. Shmutter and Doytsher [72] discuss another method for computing peaks. We originally believed that the splines were necessary

in at least two directions (horizontally and vertically), but this proved not to be the case. We chose Hermite splines [29] for the interpolation. These splines are well suited for the task because they interpolate between two points P_1 and P_2 by using the direction vectors found at those locations. By taking the direction vectors of both ends into account, a smooth spline is formed that blends in with the surrounding surface.

For the Hermite splines to be as accurate as possible, it is necessary to find good direction vectors for P_1 and P_2. Direction vectors can be found by looking at the point and its neighbors, but the results may not be very accurate for two reasons: contour digitization and/or errors from intermediate contour generation. In the first case, in areas where contours curve, digitized contours are typically more than one pixel wide. If a direction vector is computed from neighbors of a contour point, it is possible that the direction will be computed as 0 because all of the neighbors may reside on the same contour. This result would not be correct. In the latter case, the surrounding area was computed previously by the intermediate contour method. These values are estimates of the elevations in the area and are therefore subject to some error. The solution is to find a previous contour point P_0 and a following contour point P_3. Since the direction of the vector must coincide with the spline, no computation is necessary. A good estimate of the slope at point P_1 can be found from the distance and elevation of P_0; likewise, the slope at P_2 can be found using the information from P_2 and P_3. If the slope is found to be 0 on on both ends, the elevation at P_1 is simply copied for all the points between P_1 and P_2.

The entire peak interpolation algorithm for an $n \times n$ grid is therefore as follows (all observed contour elevations are marked as such previously):

```
for row r = 1 to n
    P1 = 3          // First two columns are boundary values
    while P1 < n − 1
        P1 = search for an uncomputed area starting from P1
        P0 = previous marked contour ≠ P1's contour
        P2 = end of uncomputed area
        P3 = next marked contour (not the same contour as P2)
        slope_left = (P1^elevation − P0^elevation)/(P1x − P0x)
        slope_right = (P2^elevation − P3^elevation)/(P3x − P2x)
        if (slope_left = 0 and slope_right = 0)
            set elevations between P1 and P2 = P1^elevation
        else
            compute Hermite spline between P1 and P2 using
            slope_left and slope_right
        endif
        P1 = P2 + 1
    endwhile
endfor
```

5.1.4 Filling in Gaps

Referring back to Figure 5.2, there will still be small gaps in the surface after executing the peak interpolation procedure. These gaps are typically small and thus powerful interpolation techniques are not required. Furthermore, the surfaces surrounding the gaps are good approximations found by the intermediate contour method. We use inverse distance weighting to fill in the small remaining gaps:

$$z(x,y) = \frac{w_{left} * z(x_{left}, y)}{w_{sum}} + \frac{w_{up} * z(x, y_{up})}{w_{sum}} + \frac{w_{right} * z(x_{right}, y)}{w_{sum}} + \frac{w_{down} * z(x, y_{down})}{w_{sum}} \tag{5.1}$$

where $z(x, y)$ = elevation at location (x, y)

$w_{left} = \frac{1}{(x_{left}-x)^2}$

$w_{up} = \frac{1}{(y_{up}-y)^2}$, etc.

$w_{sum} = w_{left} + w_{up} + w_{right} + w_{down}$

and the subscripts $left$, up, $right$, and $down$ represent the location of the closest known elevation in each of those directions.

This method does not suffer from its usual deficiencies precisely because of the good approximations of the surface surrounding the unknown areas. These areas are filled in accurately and quickly.

5.1.5 Smoothing

The last step in the creation of the DEM is smoothing. In the computation of intermediate contours, there is no provision for ensuring that a new elevation blends in with its neighbors. Similarly, there is no continuity assured in the vertical direction of the peak interpolation procedure. Consequently, it is necessary to apply a smoothing filter over the entire surface to minimize the total curvature and to promote better surface continuity.

In the previous methods described in Chapters 3 and 4, the thin plate computation created the smooth and continuous surface. However, the thin plate method was used to interpolate values, at high computational cost. With the maximum intermediate contour method, the entire surface is already computed, and a function is needed to smooth the result. A good smoothing function is the Gaussian filter. In one dimension, it is defined by:

$$g(x) = e^{-\frac{x^2}{2\sigma^2}} \tag{5.2}$$

where σ represents the width of the Gaussian; a larger σ yields more blurring [48]. We can create a five-point approximation by using the coefficients of the binomial expansion to create a one-dimensional, five-element filter:

$$z(x) = \frac{1}{16}\Big(z(x-2) + 4z(x-1) + 6z(x) + 4z(x+1) + z(x+2)\Big) \tag{5.3}$$

where $z(x)$ represents the elevation at location x. Equations for elevations near the edges of the grid are similar. We arrive at a convolution of the entire surface by first convolving the grid in the horizontal direction by Equation 5.3, and then convolving the surface with the same filter in the vertical direction. In the case of convolving in the vertical direction, $z(x)$ is simply changed to $z(y)$. Convolving in both directions is considered one pass of the filter; the user can choose the number of passes desired. Note that each pass of the filter will smooth the surface by effectively lowering the global elevation. Care must be taken to produce a surface that is smooth while maintaining an acceptable level of accuracy.

We have two versions of the Gaussian smoothing in the maximum intermediate contours method. Notice that in Equation 5.3 the value of $z(x)$ is changed. If one desires an interpolation of the surface from the contour data, then the original contour elevations should not be changed. In what we call the Gaussian interpolating filter, we first check if the elevation to be smoothed, that is, $z(x)$, is marked as an observed contour value. If so, the point is skipped. If it is not marked, the point is smoothed by the filter. If the user wishes a surface approximation, then the value of $z(x)$ is always replaced by the Gaussian filter result. The approximation is usually preferable because it gives a much smoother result.

The final result of all of the steps of the maximum intermediate contour method as applied to the synthetic file can be seen in Figure 5.3.

5.2 Fast Spline Interpolation

In Chapter 4, we discussed the Gradient Lines method which uses Catmull-Rom splines to create an initial estimation of the surface. These splines pass directly through the control points, making them true interpolants. Furthermore, the splines are desirable because they exhibit C^2 continuity, making them very smooth. Once all of the splines are computed, the surface is completed by applying a thin plate

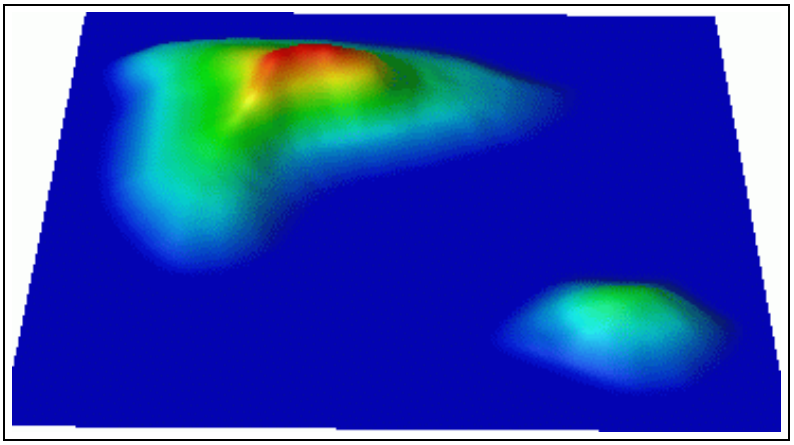

Figure 5.3: Final surface using MIC method

method.

The problem that manifests itself in the construction of gradient lines is the calculation of the gradient which is based on a thin plate surface approximation. If the initial approximation is poor, then the computed gradients will not be accurate, yielding gradient lines that do not necessarily follow the steepest slope. Since that method is only an approximation, we can use the same Catmull-Rom splines in horizontal and vertical directions to get an alternative approximation. The splines alone will form the surface, obviating the need for any computationally expensive thin plate processing. This method, which we call simply the Fast Spline method, is much faster than the Gradient Lines method, although it may be slightly less accurate. The steps of the entire procedure are shown in Figure 5.4.

The heart of the procedure is the Catmull-Rom spline. These splines assume four control points: two endpoints and two interior points. The interpolation occurs between the two interior points. The splines will be applied in the horizontal and vertical directions across the entire grid. There is a very small likelihood that there will be a contour coinciding with a grid boundary, so there will not be any data for the spline to interpolate until the first two contours are crossed. It is therefore necessary to have at least two values computed at the boundary for the spline to interpolate to the edge of the grid. The boundary computation described in Section 5.1.1 handles this problem.

The algorithm proceeds as follows. Catmull-Rom splines are computed for all

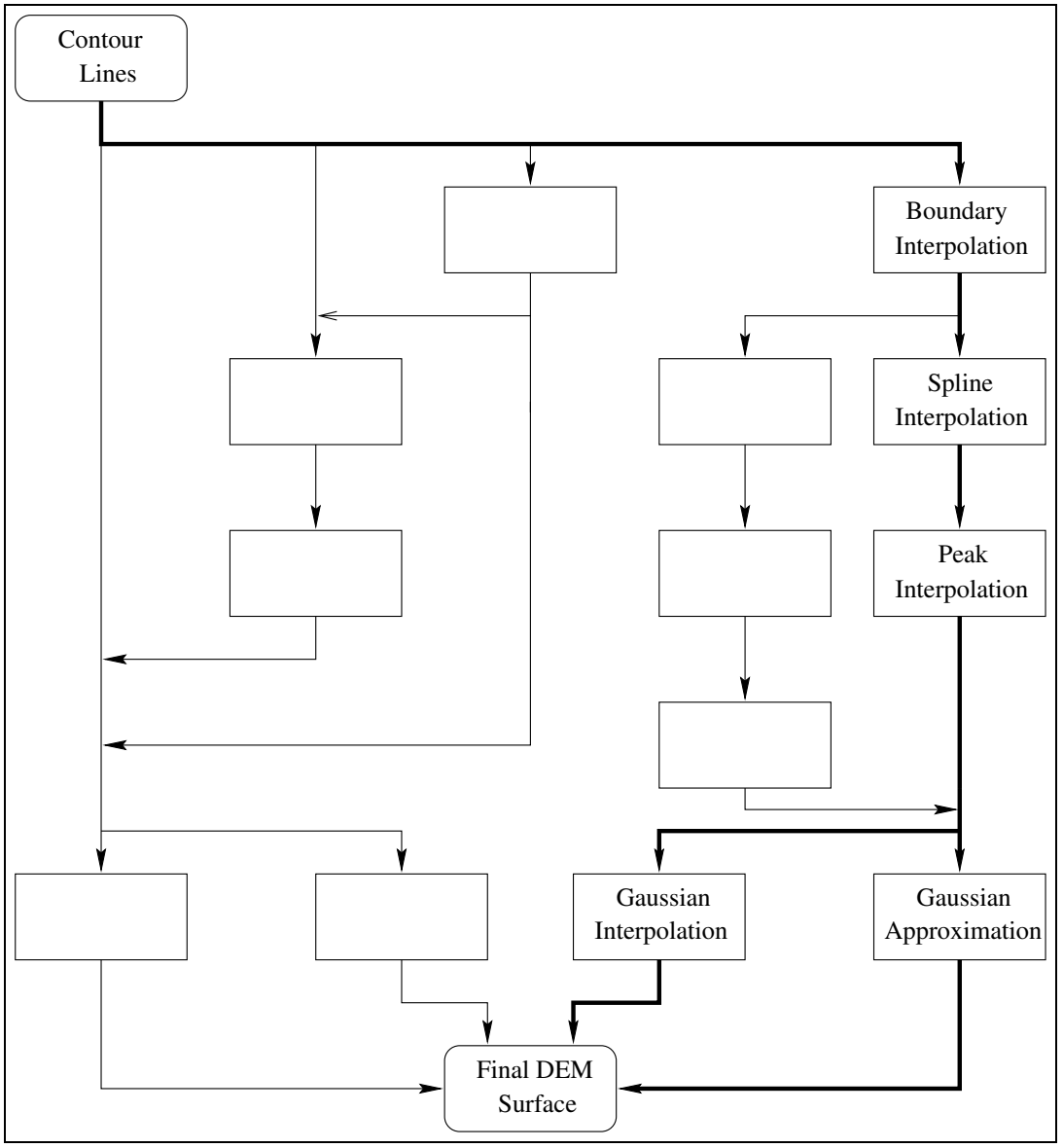

Figure 5.4: Flowchart showing Fast Spline method

vertical and horizontal lines in the grid. Boundary values and contour elevations are used as control points in the splines. The splines are computed for all interior control points except in areas where the two ends have the same elevation. In such a case, the spline may perform poorly, especially if the contours are far apart. The result would be a surface that may be too flat. Once a vertical or horizontal spline is computed, all of the values are marked. All marked values are used as control points in subsequent spline computations.

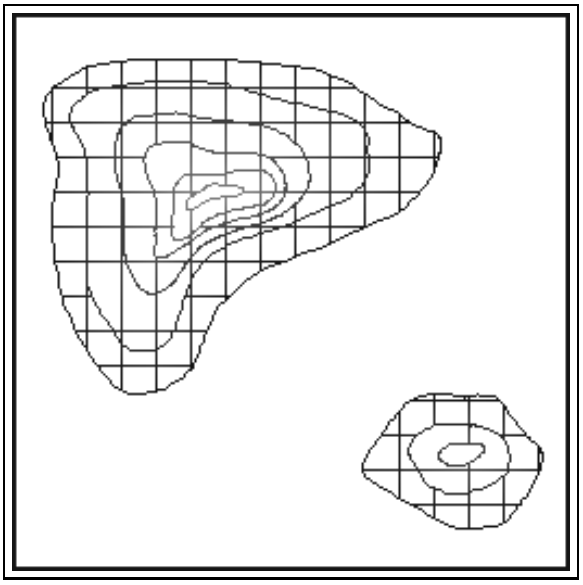

Figure 5.5: Result of four iterations of splines on synthetic data

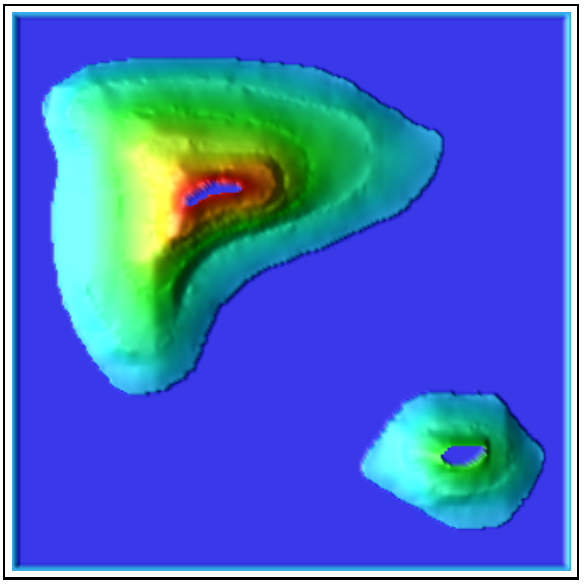

Figure 5.6: Result of applying splines to synthetic data

In order to avoid horizontal or vertical bias, the splines are computed in a lattice fashion. The next spline to be computed is always the middle row or column of the area not yet computed. Thus, the first spline to be computed is the row in the middle of the grid. In a grid of size $n \times n$, this would be row $n/2$. This splits the grid into two horizontal areas. The next spline is the column in the middle of the grid, or column $n/2$. Ignoring the row splines, this splits the grid into two vertical areas. This completes one iteration. In the next iteration, the splines to be computed are in the middle of each of the areas created in the first iteration. Thus, the next splines will be in row $n/4$ and $n - n/4$ in the horizontal direction. The vertical direction is computed analogously. This continues until the entire grid is computed. Figure 5.5 shows the synthetic data after four iterations of the algorithm. Note that there are no values computed between the boundaries and the lowest contours and other areas where the two ends of the spline are at the same elevation. However, after more iterations, splines in the orthogonal direction will fill in all gaps between contours, as shown in Figure 5.6. Only peaks remain to be filled with the possible addition of the area between the boundaries and the lowest contours in the case of synthetic data.

The splines are not used to compute peaks because they tend to produce

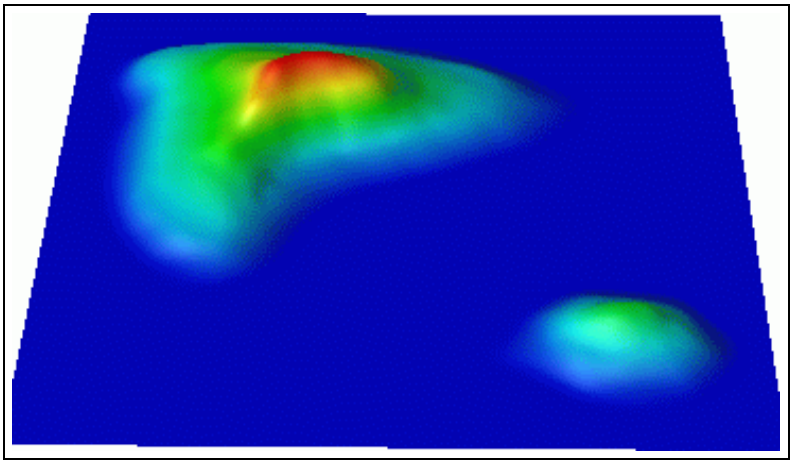

Figure 5.7: Final spline surface from synthetic data

surfaces that are too flat. The same procedure used in the MIC method to compute peaks, described in Section 5.1.3 is employed here as well. The final surface is smoothed using the same Gaussian filters as described in Section 5.3.

5.3 Performance

All of the thin plate methods have the major drawback of high computation time. As discussed in Chapter 2, convergence occurs at best in $O(n^2 \log n)$ time for a grid of size $n \times n$. This result was achieved by computing a convergence factor based on experimental results on very small samples [10]. The convergence for a large grid typical of terrain data may be quite different, putting the performance claim into question. Nevertheless, the methods described in this section show improved computation performance. In what follows, the grid size is assumed to be $n \times n$.

5.4 Performance of MIC Method

The first step in the MIC method is the boundary interpolation. This is only a matter of linearly traversing each edge of the grid. This is done twice to get a thickness of two necessary for the spline interpolation. The length of each edge is n, making the total for the outermost edge $4(n-1)$. The inner boundary, that is 1 pixel from the edge, is $4(n-2)$. The total is therefore $O(n)$.

The second step is the computation of the intermediate contours. The perfor-

mance analysis was discussed in Section 3.4, and is $O(md^2 \log d)$, where m represents the number of contour points and d represents the maximum distance between two successive contours. Note that if m is the entire grid, i.e., n^2, then d becomes 1. Conversely, if d is large, i.e., n, then m must be small.

The third step is the interpolation of peaks using the Hermite spline technique. The splines are computed in only one dimension. This is done quite easily by examining each of the n rows of the grid. While traversing the n columns in a row, we determine if a peak exists by finding the two endpoints of a non-computed area. A peak is found if the endpoints lie on the same contour. The slopes are computed at each end in constant time, and the values interpolated between the endpoints using the Hermite spline. At most, $n - 4$ points must be interpolated, the four points being the already interpolated values on the boundaries at each end. The total time to compute one row is therefore n for searching plus $(n - 4)$ for the spline computation. For the entire grid, the total computation time is therefore $n(n + (n - 4)) = O(n^2)$.

The next stage in the method is to fill in any remaining gaps using inverse distance weighting. Once again, the entire grid must be searched for any non-computed pixels, which is n^2. If such a pixel is found, the inverse weighting operation looks at most $\frac{n}{2}$ pixels in four directions for a total of $2n$. At worst, the total time is $O(n^3)$; however, the remaining gaps are usually only a few pixels wide on each side, making this closer to a constant time operation in practice. Furthermore, most of the grid points will not need any processing in this stage.

The last stage of the method is the Gaussian smoothing. At each point in the grid, the calculation involves that point and four neighboring pixels which is constant time. This is done in the horizontal and vertical directions for a total of $2n^2$. The user can opt for the number iterations g of this method, making the performance $O(gn^2)$. In practice, g is never more than 10, reducing the performance figure to $O(n^2)$.

The worst-case performance evaluation of the MIC method is therefore:

$$O(n + md^2 \log d + n^2 + n^3 + n^2) \approx O(n^3) \tag{5.4}$$

In practice, it is the intermediate contours computation which takes the most time. If we assume that it takes $O(n^3)$ time, it is less computationally efficient than the multigrid thin plate methods, which may be $O(n^2 \log n)$. The MIC method is much faster than the iterative thin plate methods. In any case, the MIC method produces measurably better surfaces than the thin plate methods alone; see results in Chapter 7.

5.5 Performance of Fast Spline Method

The performance evaluation of the Fast Spline method is much simpler than that of the MIC method. The first step in the method is to compute the boundaries. As shown in the previous section, this is $O(n)$.

The next step of this method involves computing Catmull-Rom splines for every row of the input file. The calculation of a spline is linear in the number of values to interpolate. At most, we must interpolate $n - 4$ pixels per row, with the four boundary values already computed. The spline is computed for each of the $n - 4$ rows, for a total of $O(n^2)$.

The smoothing operation is the last step in the Fast Splines method. The Gaussian function is used in the same way as in the MIC method. The performance is the same $O(n^2)$.

The complete performance estimate of the Fast Splines method is therefore:

$$O(n + n^2 + n^2) = O(n^2) \tag{5.5}$$

This is indeed the fastest of all the methods discussed in this thesis. The surfaces produced by this result may exhibit slightly lower accuracy in exchange for this fast processing.

CHAPTER 6

024 — Elevation, in whole meters changed to Elevation, whole meters

– USGS – National Mapping Division

Map Data and Accuracy Measures

The accuracy of a computed terrain surface is, of course, an important consideration when choosing a reconstruction method appropriate for one's needs. It is also impossible to judge the quality of a three-dimensional terrain model without knowledge of the data that was used in its production. In this chapter, we first discuss the type and accuracy of the data used in the production of DEMs. We then explain several accuracy measurements and discuss their relative merits and drawbacks.

6.1 Input Data

We use contour data that has been digitized to a regular grid for all of our input data. A few test files are synthetic, meaning that they were created by hand using a drawing program such as XPAINT. In such cases, contours were drawn with a color representing the elevation. The colors were chosen to represent valid contour intervals. Most of the data comes from USGS sources in the form of Digital Line Graphs (DLGs). A DLG is a digital map data set in vector form. Among other items, such as road and hydrography information, a DLG stores contours as a set of nodes and lines which connect the nodes [83]. The nodes are stored as an (x, y) pair denoting a physical location within the boundary coordinates of the map given at the beginning of the file. The placement for the nodes is usually expressed in meters, but can be feet as well, especially in older maps. The lines are guaranteed to not intersect and always begin and end on distinct nodes. A contour is stored as a set of nodes together with the contour's elevation. The elevation is usually given

in meters, but many maps, especially those that are older, show elevations in feet.

The largest USGS database is one which contains maps of 1:24000 scale. The DLGs stored in this database are derived from published USGS 7.5 minute maps. An important consideration is the fact that the age of these maps can be as old as 50 or more years and therefore may not be as accurate as is possible with more modern mapping techniques. These maps are digitized using automated drafting machines which have a resolution of 0.001 inch and absolute accuracy of between 0.003 and 0.005 inches. An interesting fact is that "the DLG data do not currently carry quantified accuracy statements [83]." This makes the process of determining the accuracy of a surface computed from such data problematic. However, DLGs are checked for correctness for the following items:

1. File fidelity and completeness: the digitized data is visually compared to proof plots which were generated by the drafting machines.

2. Attribute accuracy: each DLG stores not only lines representing contours complete with their associated elevations, but also other information such as roads, railroads, trails, hydrography, pipelines, and the like. Each attribute has an associated code. These codes are checked against a table of valid nodes by software.

3. Topological fidelity: in this perhaps most important category, the guide simply states: "The topological structure of each DLG is fully validated by software." The principle area of validation is in the checking of line crossings and similar errors arising from the digitizing process.

The validation methods listed above apply to current DLG Level 3 maps, the highest USGS quality standard. Future DLGs will also include additional error checking for edge matching and for quality control flags which note the presence of alignment or attribute discontinuities.

The DLGs used in this thesis were gathered from USGS sources on the World Wide Web. Most were retrieved from the public FTP (File Transfer Protocol) site from Xerox:

`ftp://spectrum.xerox.com/pub/map/`

This source contains matching DLGs and DEMs for the same geographic locations, enabling comparisons useful for testing the performance of reconstruction methods. The DLGs found at this site are stored in USGS's "optional" format. The format simply defines the order of presentation of the multifarious attribute data within the file. Several processing steps are necessary in order to extract and convert the raw data from the USGS format to a regular $n \times n$ grid used in the surface reconstruction software:

1. The line segments which form each contour are extracted from the DLG and stored as a set of endpoints and the contour's associated elevation value.

2. Because a DLG encompasses a large geographic area, its size is prohibitive for testing our surface reconstruction algorithms. Thus, a grid size is chosen such that it maps to a smaller, more manageable area of the DLG. The coordinates of the boundaries of this virtual grid are noted. A real grid of the same size is created and initialized to zero.

3. For each of the line segments forming contours found in step 1:

   (a) Digitized contour lines are created from the endpoints of the line segments using Bresenham's line algorithm [29]. Note that aliasing problems may occur at this stage.

   (b) The individual line segments form a contour which is usually quite large. The contour may be wholly visible, partially visible, or completely invisible within the defined boundaries. Therefore, the lines are clipped to the boundaries.

   (c) The clipped line segments with the original coordinates are normalized to fit within the actual grid boundaries.

   (d) The normalized line segments are mapped to the regular grid.

4. The last step is to store the grid in a Portable Gray Map (PGM) file. This format only needs three extra lines of header information which describe the file

as a PGM file, the size of the grid, and the maximum color value. Comments may also be inserted. This format is convenient because such files can be viewed easily using the publicly available graphics visualization program XV [6].

The result of the above steps is a PGM file which contains the elevations of every contour within the defined boundaries. Other areas have elevation zero. For the purposes of this thesis, we have made the grids square. However, this is not a necessity for any of the DEM construction techniques. The final grid has a one-unit resolution, where the units correspond to those used in the processed DLG.

6.2 Accuracy Measures

The accuracy evaluation of computed DEMs is an exercise fraught with danger. We will present several quantitative measures which may be useful in determining the accuracy of a DEM. However, these results may be misleading in some cases, resulting in the conclusion that pure quantitative analysis of a surface may not suffice in determining its accuracy. A qualitative analysis may show errors not easily recognized by quantitative measures. It is therefore desirable to look at several accuracy and smoothness measures to determine if a computed surface meets the user's needs.

6.2.1 Smoothness

One way to quantify the smoothness of a surface is to compute the total squared curvature C^2 of a grid of $n \times n$ points [9]:

$$C^2 = \sum_{i=1}^{n} \sum_{j=1}^{n} (C_{i,j})^2 \tag{6.1}$$

where $C_{i,j}$ is the curvature at the point (x_i, y_j). The curvature is a function of $z_{i,j}$ and can vary depending on the accuracy of the measurement of curvature one would

like to achieve. One such function is:

$$C_{i,j} = z_{i,j-1} + z_{i-1,j} + z_{i,j+1} + z_{i+1,j} - 4z_{i,j} \tag{6.2}$$

Note that this measure of curvature is biased towards flatness; if $C_{i,j} = 0$, the surface is flat. A more accurate function would include more neighbors, such as Equation 2.4. The lower the squared curvature, the smoother the surface.

While the squared curvature (C^2) is useful for comparing surfaces derived from the same input data, it is a sum of the curvature at each pixel and, as such, is of no use in comparing surfaces computed from different data because the measure depends on the number and steepness of the input contours. It is also not useful for assessing the quality of one DEM unless one knows a reliable total curvature estimate of the area. In such cases, a more useful measure is the average absolute curvature:

$$C_{ave} = \frac{1}{n^2} \sum_{i=1}^{n} \sum_{j=1}^{n} |C_{i,j}| \tag{6.3}$$

The closer C_{ave} is to 0, the smoother the surface. This measure is not affected by the number of DEM points and is therefore useful in comparing surfaces computed from different contour maps.

Consider Figures 6.1 and 6.2, surfaces computed from the synthetic data shown in 2.5. Notice the relative smoothness of the two surfaces. The total squared curvature for Figure 6.1, computed using thin plate interpolation, is 8888.46. The total squared curvature of the surface in Figure 6.2, a thin plate approximation, is 1030.73, bearing out the intuition from visual inspection that Figure 6.2 is smoother, especially in the steep portions just slightly up from the center of the image. The average curvature also supports this observation, with $C_{ave} = 0.096$ and $C_{ave} = 0.056$, respectively. Note that the average curvature is quite low, indicating that both surfaces are relatively smooth.

While both measures of curvature assign a numerical quantity to a particular surface, the nature of these functions result in only an approximation of the global

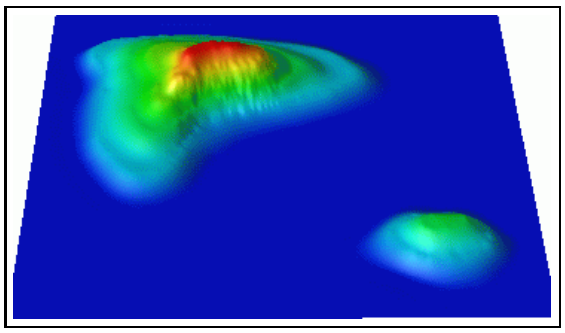

Figure 6.1: Surface computed from synthetic contours using thin plate method

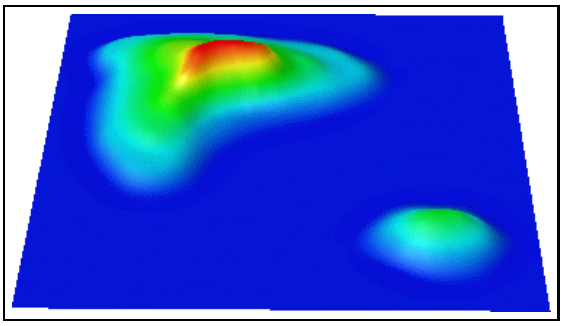

Figure 6.2: Surface computed using thin plate with springs method

smoothness of a surface. Unwanted local inflections may not influence the global smoothness to a perceptible degree. For example, a surface may be deemed smooth by its total squared curvature, but there may be small areas where the surface curvature is undesirable. In order to see such areas more clearly, the relative curvature of each grid point can be computed and shown in a graphical display, using shades of gray or different colors to differentiate the curvature values. The total number of grid points corresponding to each computed curvature value can be plotted so as to compare different surfaces directly. The relative curvatures of Figures 6.1 and 6.2 are shown in Figures 6.3 and 6.4 respectively. The darker the pixel, the higher the curvature. It is important to note that the curvatures depicted in the figures are relative to curvatures *in that file ONLY!* The two figures can NOT be compared to one another. What the Figures do show is the location of the areas with the highest curvatures. Notice that in Figure 6.3, the highest curvatures are concentrated near the steep portion of the hill. In Figure 6.4, the curvature is more evenly spread over the entire surface.

The total number of points with certain absolute curvature values can be plotted as well. This is shown in Figure 6.5. From the graph, one can ascertain that the thin plate interpolation has more points with higher curvature than the thin plate approximation. The maximum curvature of the interpolation is 21.0 (off the scale), while the maximum curvature of the approximation is only 1.9. Thus, the thin plate interpolation has more areas with higher curvature than the approximation, explaining why the total squared curvature and the average curvature are both

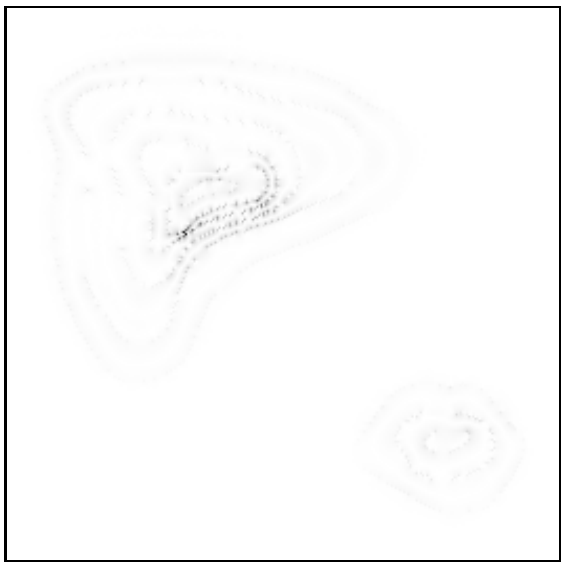

Figure 6.3: Relative curvature of the surface shown in Figure 6.1; Darker pixels = higher curvature

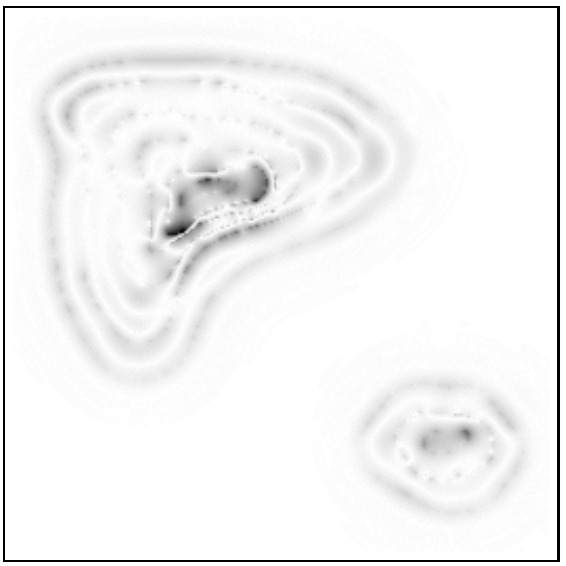

Figure 6.4: Relative curvature of the surface shown in Figure 6.2

larger. Recall that the average curvature of both files is relatively low. This is due to the fact that in both cases, most points have a curvature of essentially zero (both lines at the upper end of the scale in Figure 6.5 converge to a curvature of 0).

6.2.2 Accuracy

Accuracy can be defined as the difference between an observed data value and its corresponding computed value [89]. The usual measurement of accuracy of a DEM is the root mean square error (RMSE) of the surface [66]:

$$RMSE = \sqrt{\frac{1}{n^2}\sum_{i=1}^{n^2}(z_i - w_i)^2} \tag{6.4}$$

where z_i = the interpolated DEM elevation of test point i

w_i = the true (most probable) elevation of test point i

Obviously, one must have an accurate surface with which to compare. The nature of computing surfaces from scattered data implies that there is no previous surface to which comparisons can be made. The RMSE is still useful in that points on the computed surface can be compared to the corresponding observed points on

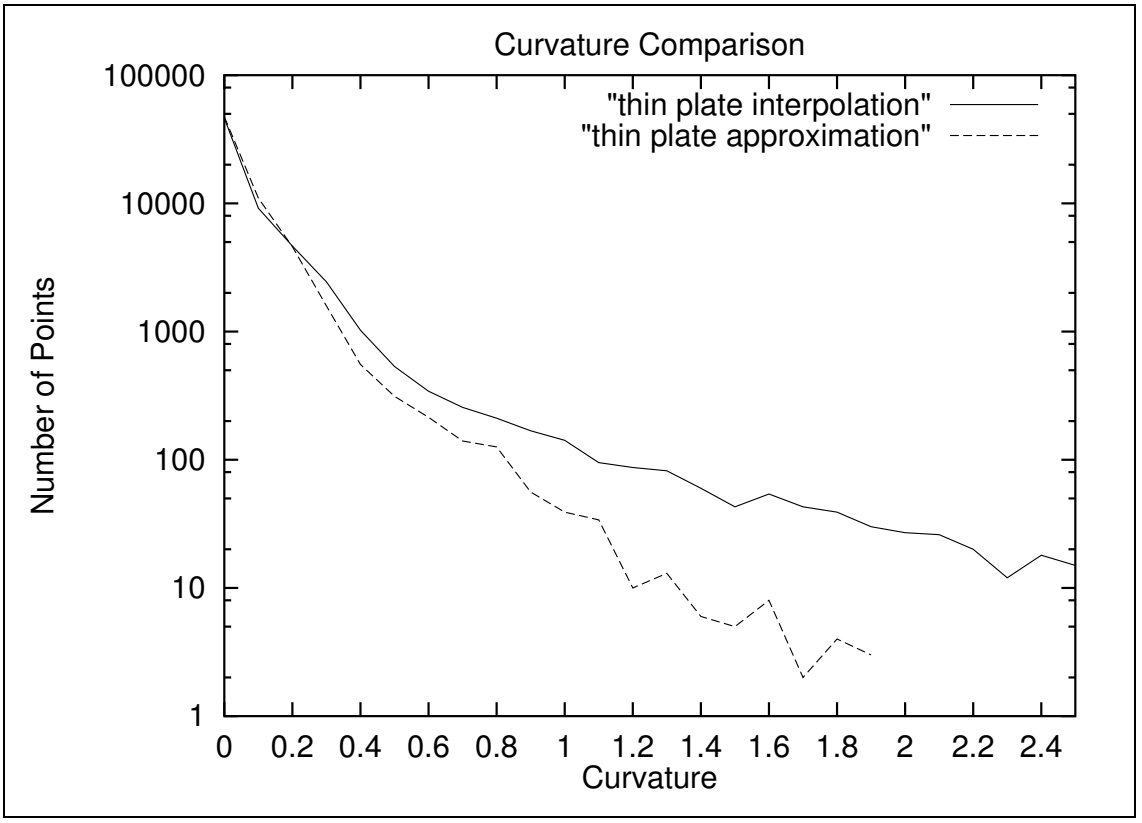

Figure 6.5: Plot of curvature of thin plate interpolation and thin plate approximation

the contour map or in the set of scattered data. We can also compare the computed surface to a USGS published DEM of the same area. However, DEMs have errors as well (see Section 6.3), so this is only useful for comparing different methods on the same data set. When computing a surface approximation, the lower the RMSE the better the surface "fits" the data.

Another method for determining accuracy can be found in [42], where the slopes surrounding a point are compared to ascertain whether the surface is smooth. The paper further gives an algorithm which uses the error information in order to correct any anomalies in the surface.

Using any of the above methods for determining smoothness or accuracy, one can test a system by recovering contours from a DEM, then producing a new surface from these contours. The resulting surface can be compared to the original DEM using any desired method. One problem with this approach is that USGS DEMs have

only 30 meter resolution, which makes the creation of contours from this relatively large-spaced data rather inaccurate.

6.2.3 Qualitative Measures

Purely quantitative measurements of a DEM do not always determine the accuracy of the surface. Often, these measures only give an estimate of the overall quality of the computed surface. However, gross imperfections may exist in a small area of a DEM which otherwise exhibits good behavior. Wood and Fisher [89] have proposed a visualization method which helps detect errors which may not be found using conventional techniques. Although a common method is to display contours with various colors, this is not deemed adequate. A more useful visualization is to display the surface as a shaded relief map, changing contrasts or colors so as to see different aspects of the data. This method allows one to see problems such as terracing. All of our results are displayed using the scientific visualization package DataExplorer from IBM. This package allows the user to view the surface in three dimensions. The image can be rotated easily. Zooming functions are also provided, allowing the user to view specific problem areas. Colors are assigned to elevations, with blue being the lowest and red the highest. Although the user may change the colors, it was found that the default colors, though not realistic, better differentiated the elevations. Images created by DataExplorer are used throughout this thesis.

It is sometimes difficult to determine the fit of a surface relative to the initial contours even with a good visualization package available. In such cases, it is often desirable to plot a profile or cross section of the surface. The plot can be overlaid on the original contour data, showing how well the surface corresponds to the initial data. This kind of plot is also useful for comparing different reconstruction methods. For example, Figure 6.6 shows how the IC method follows the slope of the contours better than the thin plate approximation method (the vertical lines represent the initial contour elevations). This may not be obvious in the images, especially when only grayscale is available.

Computing the slope and aspect of a DEM and plotting the results can also reveal problems in the surface. Laplacian filtering can be used, as well as profile

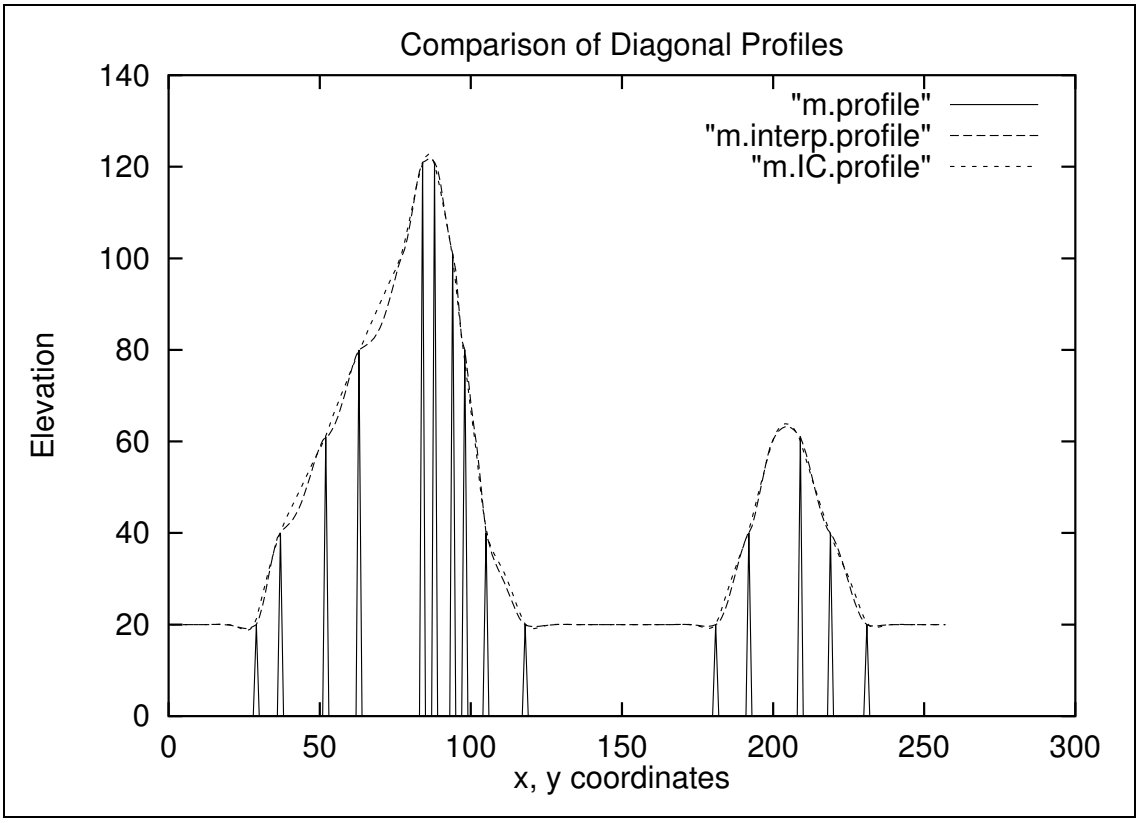

Figure 6.6: Comparison of diagonal profiles from thin plate interpolation and IC surface

and plan convexity, found by computing the second derivative in both the x and y directions. The latter method may be best of all, since it showed some errors that were not clearly represented by any other method in Wood and Fisher's experiments. Finally, it is also possible to visualize RMSE or curvature (see Section 6.2.1, which shows the locations of the areas with the greatest and least accuracy.

6.2.4 Comparisons to USGS Standards

A good decision on the smoothness/accuracy problem can be made based on the standards set forth by the US Geological Survey (USGS) for their own DEMs. There exist 3 quality levels for the standard 1:24000 scale, 7.5- by 7.5-minute, DEMs [13]:

- Level 1 - no point should contain an error over 50 meters, a 7×7 array of points should not be in error by more than 21 meters, and the maximum

RMSE allowed is 15 meters. Furthermore, errors classified as systematic (errors introduced by the creation of the DEM which results in a bias or artifact in the computed surface) are tolerated. Many such systematic errors can occur in the very first stages of DEM production; namely, in the sampling phase [67]. The digitization process may introduce errors (see [5] for more specific information) as well as the problem of superimposing the sampled data onto a regular grid. The latter problem has been addressed by some researchers; see eg. [9] and [77]. Systematic errors that occur in any phase of DEM generation are propagated to the next step.

- Level 2 - no point contains an error greater than two contour intervals and the maximum RMSE allowed is one-half contour interval, not to exceed 7 meters. These DEMs have been smoothed and edited so as to remove systematic errors. This can be done in a variety of ways; for example, by manual editing or by using automatic methods such as described in [42] or [60].

- Level 3 - no point contains an error greater than one contour interval and the maximum RMSE allowed is one-third contour interval, not to exceed 7 meters. These maps have been further processed to insure positional and hypsographic accuracy with respect to planimetric data such as transportation and hypsography.

It is interesting to note that the RMSE is computed using only 20 control points within a DEM. It is therefore adequate to compute the RMSE using the observed data points when comparing a computed surface to the USGS standards because the contours themselves will account for more than 20 points. Visual inspection, often leading to manual editing, is a large component of the USGS process. Most DEMs conform to Level 1 specifications, with newer DEMs conforming to level 2. Even so, DEMs produced from contour data still showed signs of terracing as late as 1991 [2], indicating that surfaces with obvious errors still conform to the standards.

If one assumes that all DEMs are of Level 2 quality, then the maximum RMSE for a 20 unit contour interval can be as high as 10. If the unit is greater than a meter, than the RMSE must be lower than 7. In either case, the RMSE found for our

computed surfaces are well within the USGS criteria. For example, the RMSE for the thin plate approximation on the synthetic data (Figure 6.2), where the contour interval is 20 units and the number of control points is 1975, is no more than 0.687 units (see Chapter 7). Such a low RMSE suggests that it is reasonable to allow the values at the observed points to deviate slightly to achieve a smoother surface which still conforms to USGS standards.

6.3 Accuracy of USGS DEMs

The RMSE is a major error measure in the quality assessment of a surface. By its nature, the equation needs a basis surface to which to compare. It seems natural to use USGS DEMs for this comparison. In order to understand the RMSE computed from such a comparison, it is essential to understand how the DEM was constructed and how accurate it is.

Burrough [11] claims that DEMs are usually made by sampling stereoscopic aerial photographs using photogrammetric instruments. This has not been the case with the USGS in the past, although the original maps from which they do the sampling may be made this way. Newer DEMs may incorporate this new technology.

The production of USGS DEMs is explained by Rinehart and Coleman [66]. DEMs are created from published DLGs using a system called DLG2DEM. Hypsography information (contours), hydrography information (water bodies), profiles, and spot heights are extracted from a DLG. All of these attributes are used in the creation of the DEM. The gridding process is done by Zycor's CTOG software in the following four steps:

1. Four scan lines representing the 8 neighbors of a grid point are generated. All intersections between the scan lines and features are determined.

2. An initial elevation for a grid point is computed by linearly interpolating along the scan lines. Weights are also determined for each direction incorporating slope information.

3. As was described in Chapter 2, a simplistic linear interpolation will cause terraces. If a sequence of identical elevations is detected, then a new elevation

is determined for the grid location by following the slope information found in the previous step. Weights are recomputed for the new point.

4. The final elevation is determined using a weighted average incorporating the weights found in steps 2 and 3.

The DEMs computed using the above method correspond to Level 1 or Level 2 accuracy. Acevedo [2] shows that typically these DEMs still exhibit terracing and, as of 1991, were still being produced using the same Zycor software. Obviously, care must be taken when comparing one's computed DEM to a USGS DEM.

Another issue to consider is the fact that the DEM is constructed from a DLG. The DLG is also a digitized format in that it is built from paper maps. Even assuming that drawn maps are accurate (which they are not), digitization errors exist [11]. For example, even the lines themselves are problematic. A 1-millimeter drawn line on a 1:100000 scale map covers an area 100 meter wide; a 1-mm line on a 1:24000 map, the scale used in this work, covers an area 24 m wide, which is significant. The general rule is to digitize the middle of the line, but this is difficult in practice. Curves present another problem, as the number of vertices chosen on a particular line impacts its precision. There are even problems regarding distortion arising from the stretch of the paper. Other errors due to digitizing include one contour which actually represents two, very close contours, or one contour which has a split where the two close contours diverge.

Other sources of errors include the age of the maps used in the digitizing process, and the density of observations (i.e., need more observations if area is very irregular). The original maps may have problems in the original measurements, such as the positional accuracy from poor field work or paper shrinkage, simple data entry or operator errors, measurement errors and biases in the particular surveyor him/herself.

Obviously, errors are ubiquitous in the creation of DEMs. Errors can occur at the initial measurement stage in the field, and can be a factor in the creation of paper maps. The errors on the paper maps are then propagated in the process of creating DLGs. In addition, more errors are introduced in the form of paper shrinkage and digitization errors. Finally, the compounded errors are propagated

to the final DEMs, which also suffer from inaccuracies in the interpolation process. Thus, it is almost impossible to quantify the overall accuracy of a DEM, making comparisons to them somewhat suspect.

CHAPTER 7

I have had my results for a long time;
but I do not yet know how I am to arrive at them.

– Karl Friedrich Gauss

Experimental Results

Many experiments were run to test the effectiveness of all of the new methods. The sample data was also used to create surfaces using traditional thin plate methods for comparison purposes. Some data was synthetic, created with the program XPAINT, while most was gathered from USGS sources. Each data set was tested with each new interpolation and/or approximation method. Images were created for each resulting surface using IBM's DATAEXPLORER package. To better see various aspects of the surface, both a top view and either a side or angled view of the surface were created. An unfortunate aspect of this process is the addition of aliasing along the edges of some of the images. The jaggedness of these edges do not reflect the output of the reconstruction techniques. This artificial "raggedness" should be fairly obvious to the reader.

The quantitative results are shown in a table for each data set. This allows direct comparison of the methods for that particular input. The measurements include the total squared curvature (C^2), the average absolute curvature (C_{ave}), the root mean square error of the surface as compared to the original contour data ($RMSE_1$) , and the root mean square error of the surface as compared to a USGS DEM of the same location ($RMSE_2$) , if applicable.

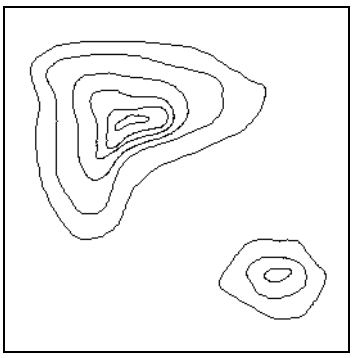

Figure 7.1: Synthetic data

7.1 Results from Synthetic Data

The synthetic data is a grid of size 257×257 containing contours with an interval of 20 units. For convenience, it is shown again in Figure 7.1. The small size of the file makes it particularly conducive for testing purposes. The contours also have sufficient curvature at the southern and eastern ends of the larger hill to cause problems for thin plate methods. The closeness of the contours near the middle of the larger hill exemplify digitizing errors. The flat plain ends abruptly at each hill, testing the continuity properties of the methods.

Although there is no DEM to which to compare, the synthetic data was used to test the relative effectiveness of each method. The simplicity of the contours allows one to easily see the difference between methods. Figures 7.2 and 7.3 show the surface resulting from the application of the thin plate interpolation. As described in Section 2.1.3, this surface exhibits both the terracing problem and the digitizing error problem. The former can be seen easily in Figure 7.2 along the southern and eastern ends of the larger hill, and the south-western area of the smaller hill. The latter problem can be seen along the steep face of the larger hill. Note, too, the flattened peaks in Figure 7.3. Table 7.1 shows the quantitative results of each method. Notice that the total squared curvature of the thin plate interpolation is 8888 and that the average absolute curvature is 0.096. Both of these measurements must improve in order to see fewer digitizing errors.

Figures 7.4 and 7.5 show the surface constructed using the thin plate approximation technique. Notice how the digitization errors are smoothed considerably. Some of the terracing, if it was not too pronounced originally, is also improved; see

Method	C^2	C_{ave}	$RMSE_1$
Thin plate interpolation	8888	0.096	0.00
Thin plate approximation	1031	0.056	0.69
Thin plate under tension	11501	0.068	0.00
IC method	1356	0.062	0.88
Gradient Lines method	980	0.053	0.69
MIC method	804	0.046	1.52
Fast Spline (interp.)	17068	0.094	0.00
Fast Spline (approx.)	852	0.048	1.27

Table 7.1: Results from applying methods to synthetic data

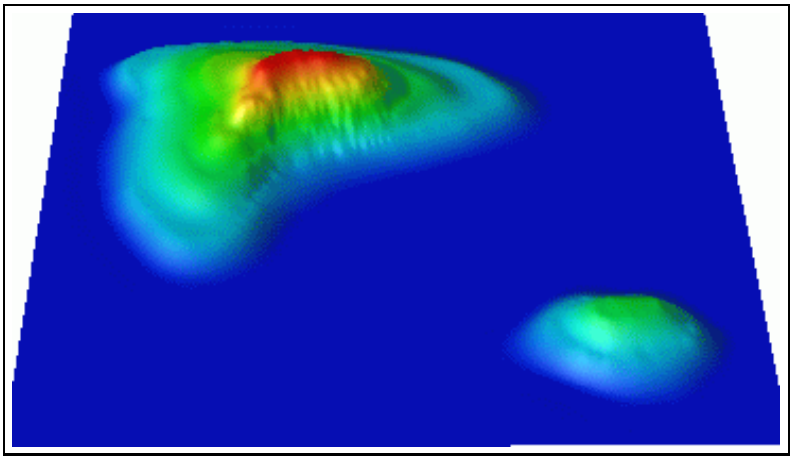

Figure 7.2: Synthetic data: Thin plate interpolation, top view

the south-western edge of the smaller hill. However, the terracing in between the larger contours of the larger hill is still present. The table shows that the total squared curvature dropped to 1031, corresponding to the much smoother surface. In exchange for this smoothness, the $RMSE_1$ rises, but is still very low.

It was hoped that adding tension to the thin plate would reduce the terracing effect. This method can not smooth out digitizing errors because it is a true interpolation technique.[4] Figures 7.6 and 7.7 show the resulting surface. It is obvious that this method does not provide a good surface in areas of high contour curvature. In these areas, the surface acts like a rubber membrane stretched taught over the contours, yielding sharp edges. This is further supported by the high total squared

[4]Allowing tension in a thin plate approximation yields a very smooth surface, but one where the highest elevation of the surface is extremely compromised (i.e., the $RMSE_1$ is very high).

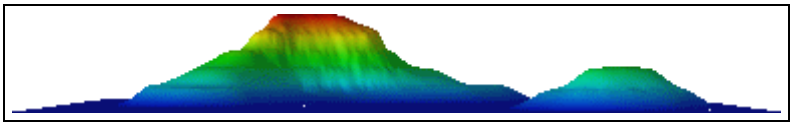

Figure 7.3: Synthetic data: Thin plate interpolation, side view

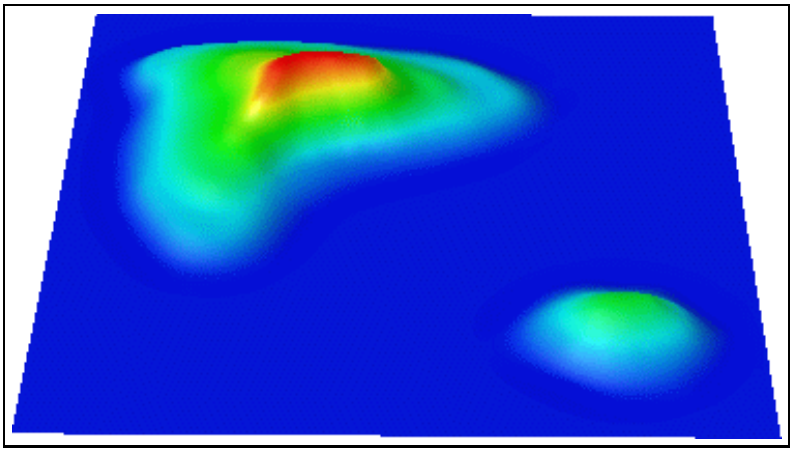

Figure 7.4: Synthetic data: Thin plate approximation, top view

curvature reported in Table 7.1. Note also that the tops of the peaks are completely flattened.

The thin plate approximation has been shown to reduce the digitizing problems. The main focus of the IC method is therefore to reduce the terracing effect. Indeed, Figures 7.8 and 7.9 show a marked improvement in the terracing problem areas while still exhibiting good curvature and $RMSE_1$ measures. The improvement in the terracing is further supported by looking at diagonal profiles as shown in Figure 7.10. Both the thin plate interpolation and approximation show "sags" between some contours, whereas the IC method shows better slope continuity in those areas. The thin plate methods also exhibit some Gibbs phenomena at the junction of the hills and the flat plain. Note that the IC method is shown only using the thin plate approximation to produce the final surface. As shown previously, the thin plate interpolation produces surfaces that are not very smooth.

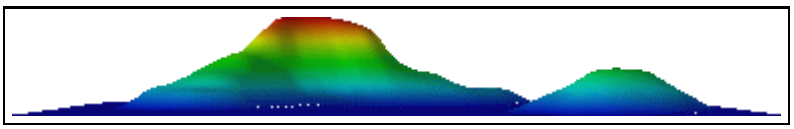

Figure 7.5: Synthetic data: Thin plate approximation, side view

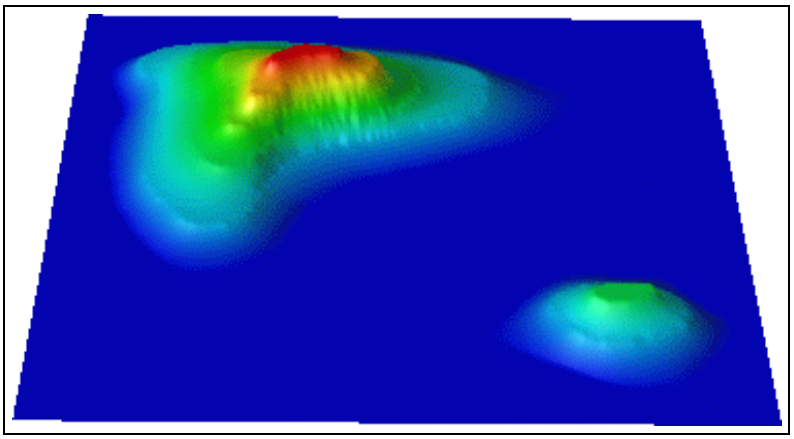

Figure 7.6: Synthetic data: Thin plate under tension, top view

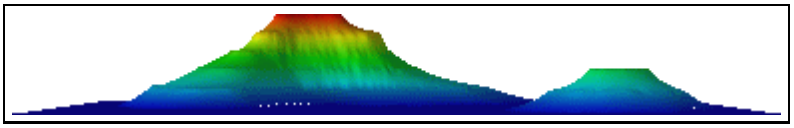

Figure 7.7: Synthetic data: Thin plate under tension, side view

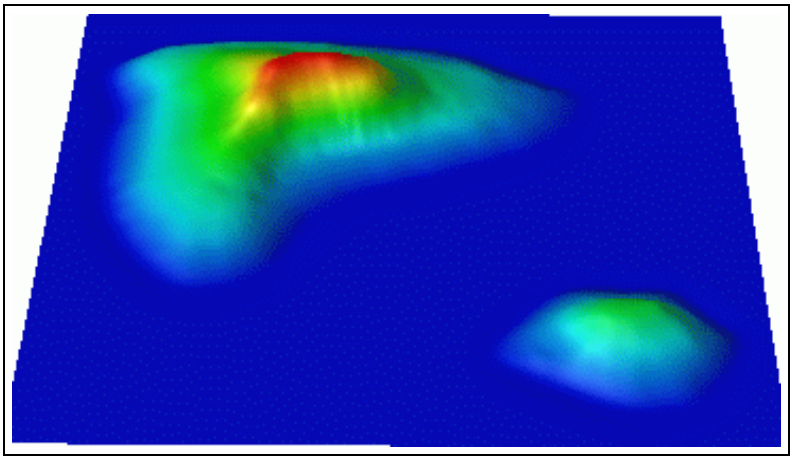

Figure 7.8: Synthetic data: IC method, top view

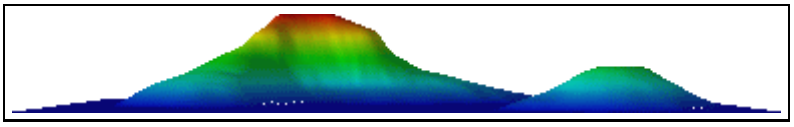

Figure 7.9: Synthetic data: IC method, side view

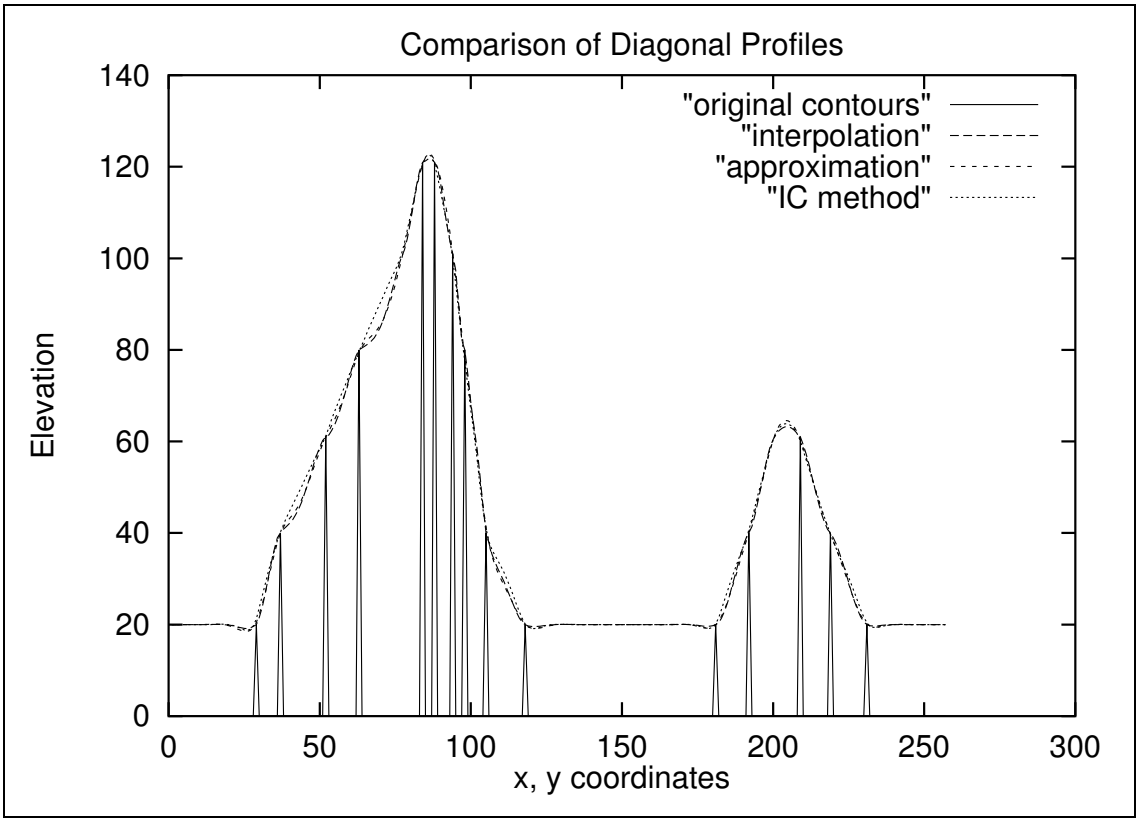

Figure 7.10: Synthetic data: Profiles of thin plate interpolation, thin plate approximation, and IC methods

The Gradient Lines method is another attempt at correcting the terracing problem. In this case, splines that follow the steepest slope are computed. Figures 7.11 and 7.12 show the surface which is very similar to the surface created by the IC method. The curvature is slightly better for the gradient method as shown in the table. The ridgeline is slightly smoother in Figure 7.12. The Gradient Lines method is compared to the IC method in profile form in Figure 7.15.

The MIC method is very similar to the IC method. As a result, the performance of the MIC method is also very much the same. The difference results from five iterations of the Gaussian smoothing function which lowers the total curvature. While the interpolating smoothing function results in a poor surface, the surface produced using the approximating smoothing function results in a very smooth surface. Figures 7.13 and 7.14 show the surface very much like the IC surface, but just a bit smoother. The price of achieving such smoothness is accuracy. Table 7.1

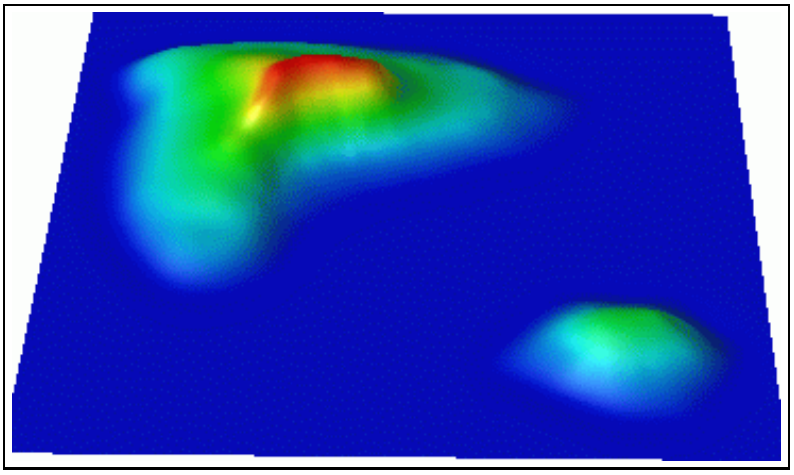

Figure 7.11: Synthetic data: Gradient Lines method, top view

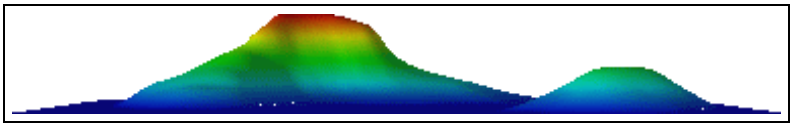

Figure 7.12: Synthetic data: Gradient Lines method, side view

shows very good curvature figures relative to the previous methods. However, the $RMSE_1$ is the highest of all methods. It is still quite acceptable if USGS standards are of interest. The MIC method is also included in the profile plot depicted in Figure 7.15. This method is virtually indistinguishable from the IC method.

The Fast Spline method is the last method tested. Figures 7.16 and 7.17 show the method in conjunction with five iterations of the Gaussian interpolating

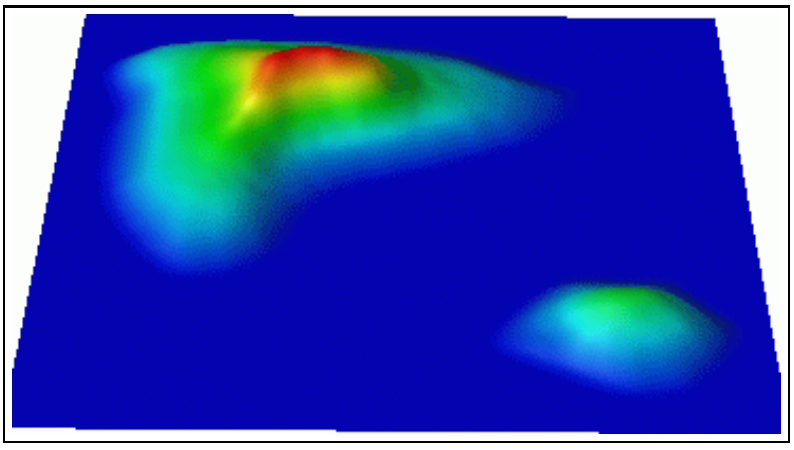

Figure 7.13: Synthetic data: MIC method, top view

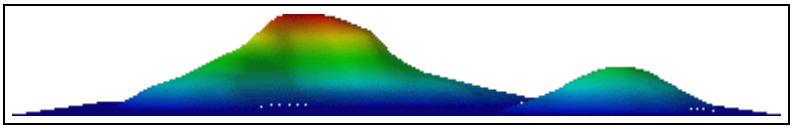

Figure 7.14: Synthetic data: MIC method, side view

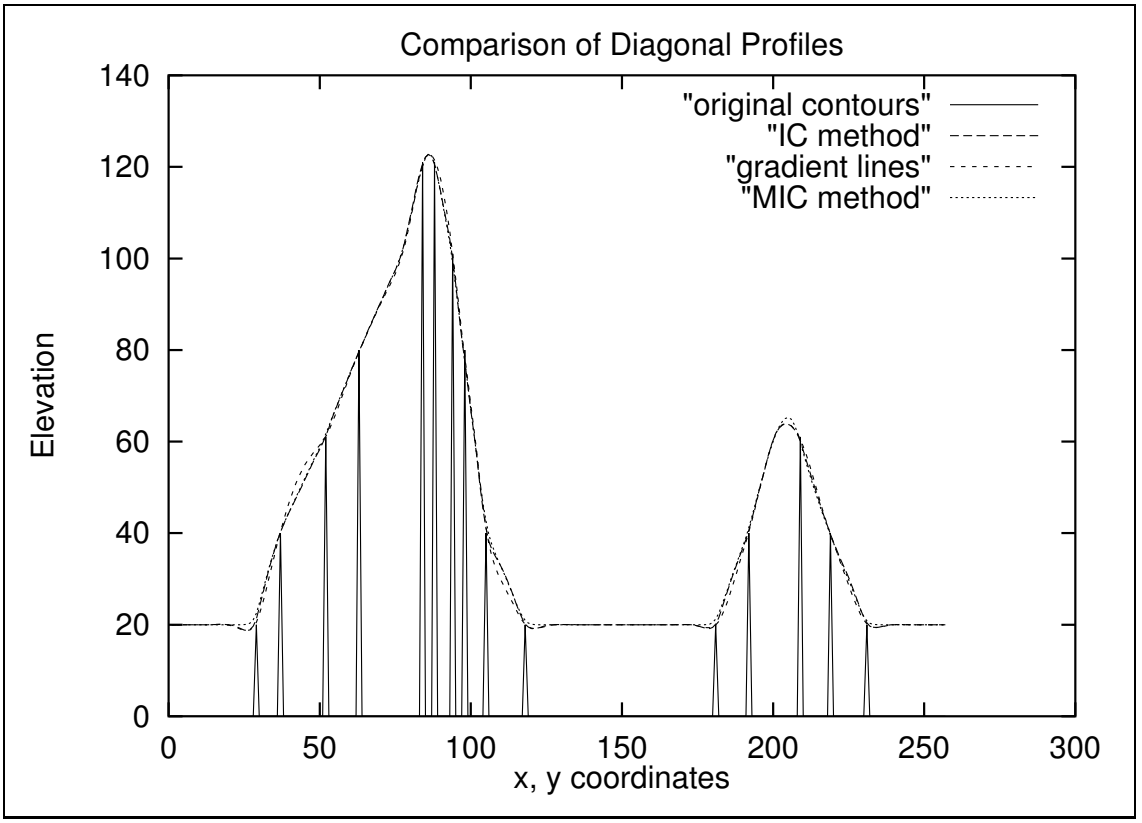

Figure 7.15: Synthetic data: Profiles of IC, Gradient Lines, and MIC methods

smoothing function. Although the ridge line of the large hill is well preserved, the contours are visible because of aliasing errors. The rough areas also explain the poor curvature values shown in the table. The general shape of the surface is quite good, without obvious terraces. The substitution of the approximating Gaussian smoothing function for the interpolating function results in a marked improvement of the surface which is depicted in Figures 7.18 and 7.19. The curvature also improves while the $RMSE_1$ rises, as one would expect. Figure 7.20 shows profiles produced by this method compared to the MIC method. Notice that the splines do not follow the contours quite as well as the MIC method. The splines are computed in the

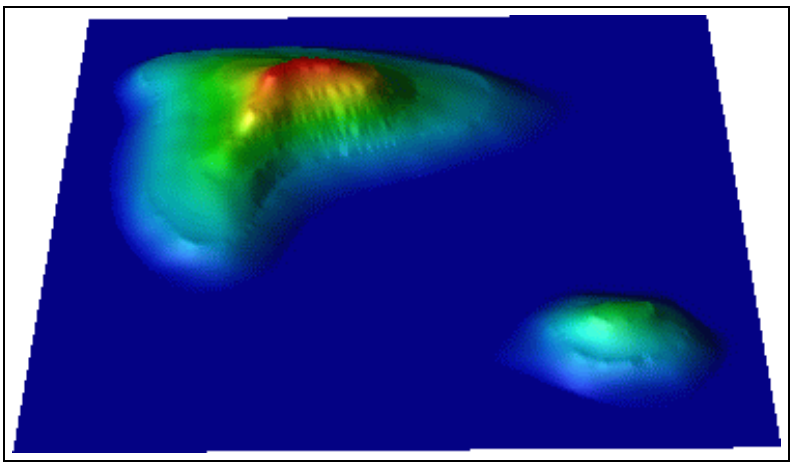

Figure 7.16: Synthetic data: Interpolating Fast Spline method, top view

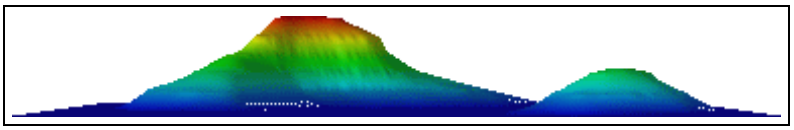

Figure 7.17: Synthetic data: Interpolating Fast Spline method, side view

horizontal and vertical directions, while the plot shows a diagonal profile. This shows the weakness of the Fast Spline method. The slightly poorer surface produced by the Fast Spline method is mediated by the fact that it is the fastest of all the methods. As a final note, both methods show a smoother transition between the hills and the flat plain.

7.2 Results from USGS Data

In this section, we show the results of our algorithms using USGS contour data. Because of the thin plate interpolation's rather poor performance, only the thin plate approximation is used in conjunction with the IC and gradient lines methods. Similarly, the use of the Gaussian interpolating smoothing function produces surfaces with poor curvature. Thus, for the MIC and Fast Spline methods, only the surfaces resulting from the use of the Gaussian approximation smoothing function are shown.

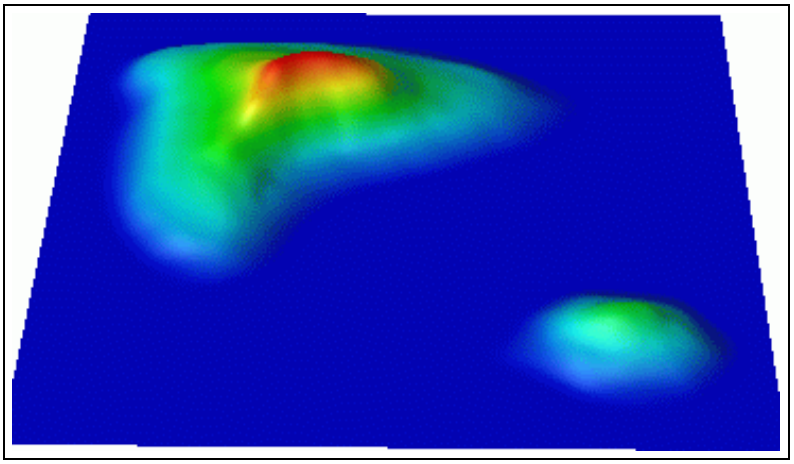

Figure 7.18: Synthetic data: Approximating Fast Spline method, top view

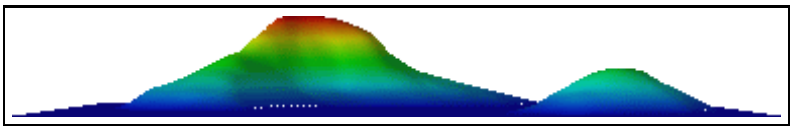

Figure 7.19: Synthetic data: Approximating Fast Spline method, side view

7.2.1 Mount Washington, New Hampshire

The first contour file containing USGS data is taken from Mount Washington, New Hampshire. The original DLG was cropped to include the peak and some of the nearby ravines.[5] The contours are shown in Figure 7.21. The ravine shown in the south-western portion of the map is the famous Tuckerman Ravine where some of the first ski races in the U.S. took place. It is also only of the few areas in the eastern U.S. where avalanches occur. The rasterized contours fill a 800×800 grid. The elevations are in meters and range from 1100 to 1900. The contour interval is 20 meters. This is essential for comparing to USGS RMSE standards. Unfortunately, a USGS DEM of the same area was not available.

Figures 7.22 and 7.24 show the surface resulting from the thin plate interpolation method. The terracing is painfully obvious. The peak is also completely flat. The sharp edges along each contour result in very poor curvatures as shown in Table 7.2. The thin plate approximation, shown in Figures 7.23 and 7.25 are much

[5]The map is rotated one quarter turn in the clockwise direction; that is, North points to the right.

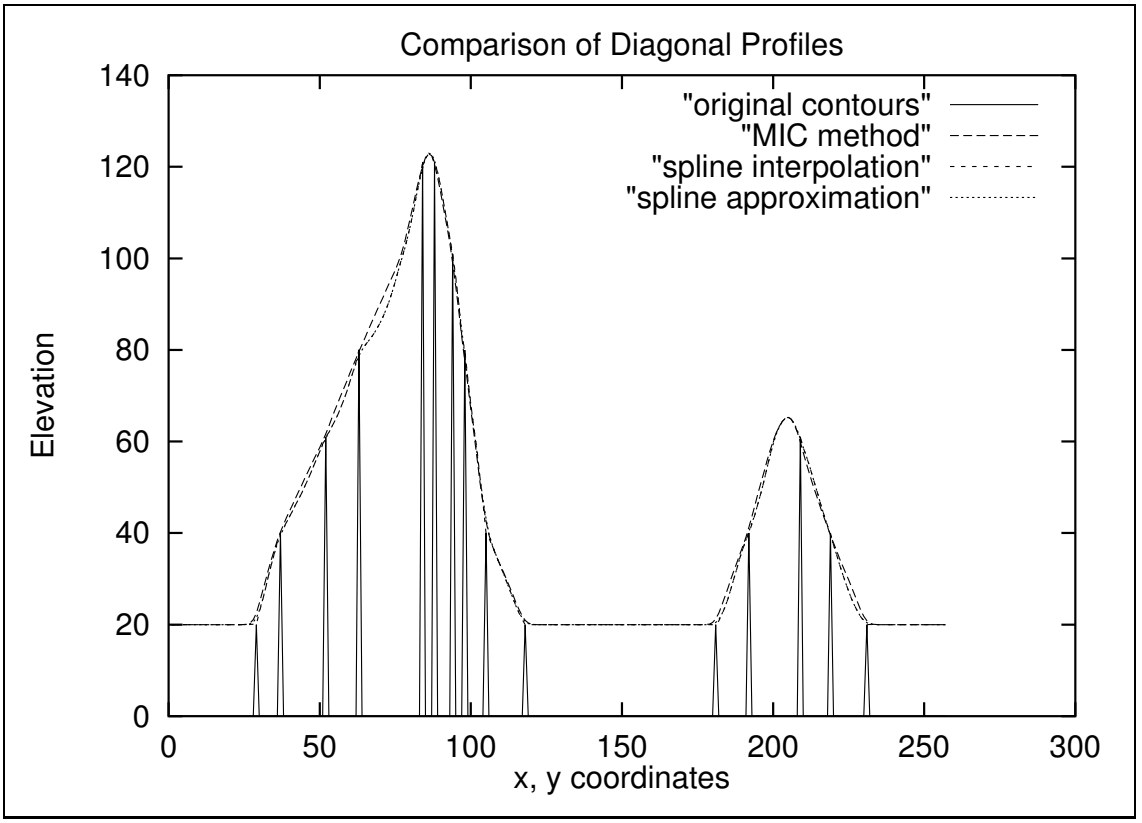

Figure 7.20: Synthetic data: Profiles of MIC, spline interpolation, and spline approximation methods

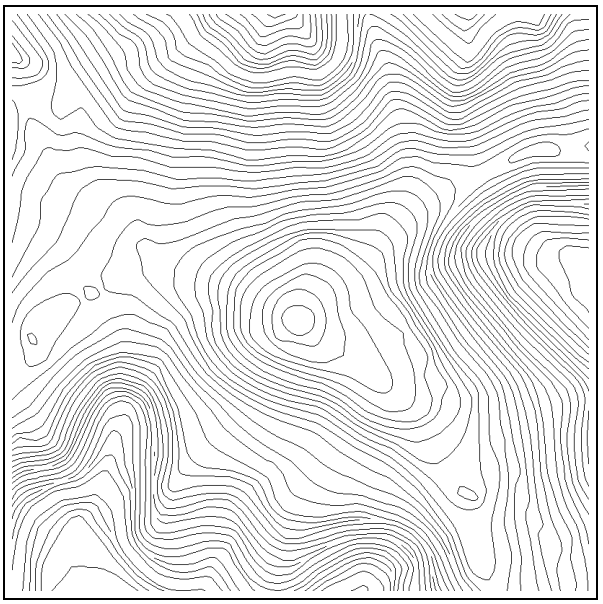

Figure 7.21: Contours of Mt. Washington, NH

Method	C^2	C_{ave}	$RMSE_1$
Thin plate interpolation	153805	0.238	0.00
Thin plate approximation	17001	0.095	0.59
IC method	23279	0.104	1.92
Gradient Lines method	15623	0.083	0.59
MIC method	6376	0.060	1.73
Fast Spline method	8048	0.070	1.53

Table 7.2: Results from applying methods to Mt. Washington contours

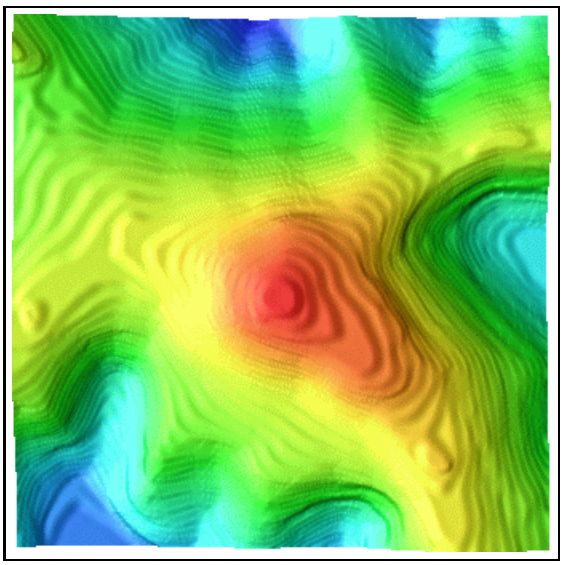

Figure 7.22: Mt. Washington: Thin plate interpolation, top view

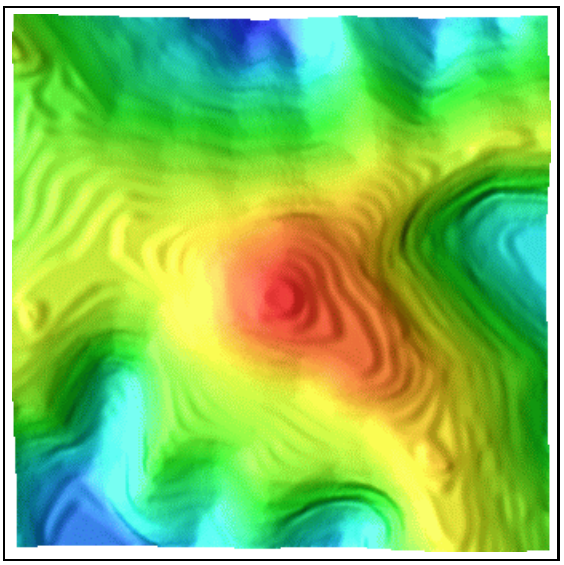

Figure 7.23: Mt. Washington: Thin plate approximation, top view

smoother, but still exhibit very large terraces. While the smoothing of the edges is a large factor in the improved total squared curvature and average curvature, it is ironic that the terraces themselves also contribute to the better quantitative measures. Consider Figure 7.26. It shows the relative curvature of the thin plate approximated surface. The darker shades indicate higher curvature. In flat areas, which represent large terraces, the curvature is very low, contributing very little to the total squared curvature.

Although the thin plate approximated surface yields good curvature, there is significant terracing over the entire area. The IC method, shown in Figures 7.27 and 7.29, exhibits much less terracing. However, the total squared curvature is

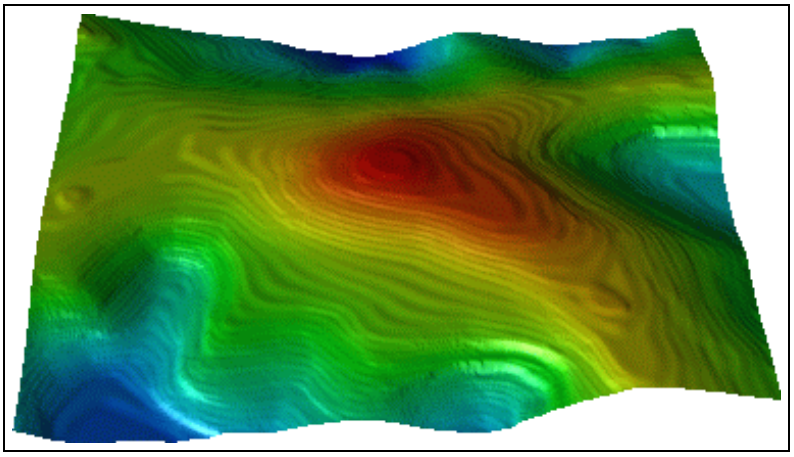

Figure 7.24: Mt. Washington: Thin plate interpolation, angled view

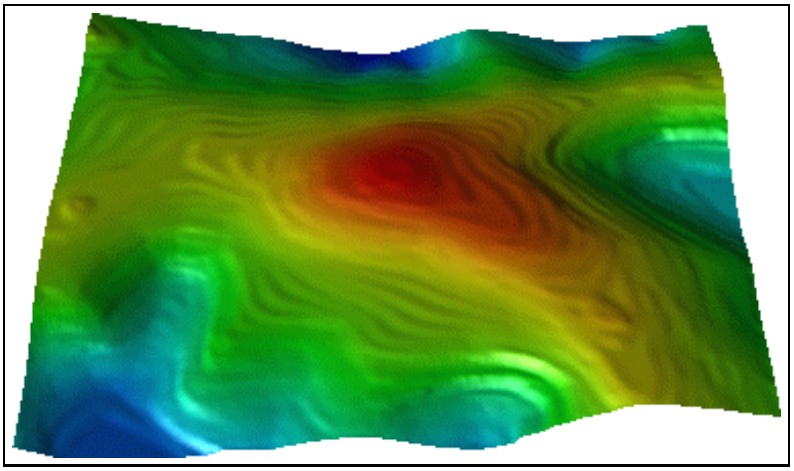

Figure 7.25: Mt. Washington: Thin plate approximation, angled view

actually *higher* than the thin plate approximation! As described in Chapter 3, the intermediate contours may not be quite complete or in precisely the correct location. The intermediate contours generated for the Mt. Washington data is shown in Figure 7.28. In areas where the contours are extremely close, there will be many new elevation points computed by the IC method. Some of those elevations may be erroneous. The final step of the thin plate approximation will smooth out most of these errors, but some will still remain. Figure 7.30 shows the relative curvature of the IC method. The surface is smooth almost everywhere, with a few exceptions in areas where the contours are very close together. The situation is further depicted in Figure 7.31. This plot shows the number of grid points with a given absolute curvature. The thin plate interpolation shows the most points

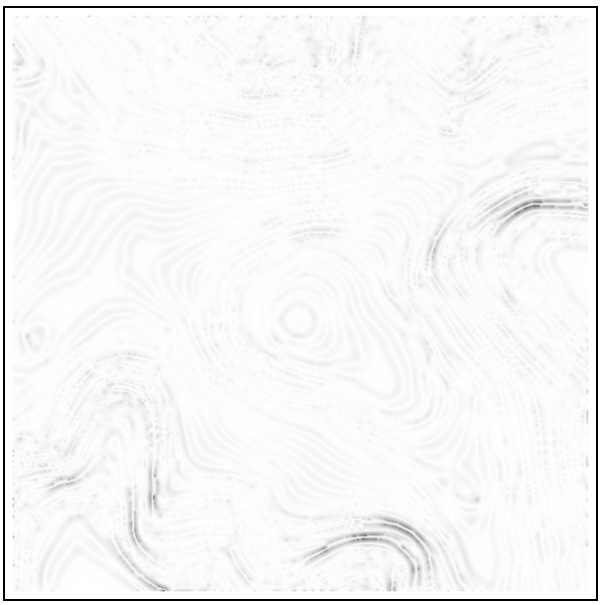

Figure 7.26: Mt. Washington: Relative curvature of thin plate approximation

with a given curvature and the fewest points with a curvature of zero. The thin plate approximation has more points with an absolute curvature of about 0.1. The IC method has more points with a curvature of zero, but a few more with higher curvature. Thus, the IC method can show a surface which is generally smoother at the expense of some poor local curvature. The performance in some local areas creates the somewhat higher total squared curvature of the IC method relative to the thin plate approximation. This also bears out the weakness of some of the quantitative measures, as it seems rather obvious that the IC method produces a generally better surface.

The Gradient Lines method yields a good surface, both qualitatively as shown in Figures 7.32 and 7.34 and quantitatively, as shown in the Table 7.2. The $RMSE_1$ is very low, identical to the thin plate approximation, whereas the total squared curvature and the average absolute curvature are both less. The MIC (Figures 7.33 and 7.35) and Fast Spline (Figures 7.36 and 7.37) methods both fare very well. In both cases, ten iterations of the Gaussian smoothing were used. The curvatures are both very low, about half of the next best method (Gradient Lines). As is to be expected using the Gaussian smoothing, the $RMSE_1$ is slightly higher. However, the

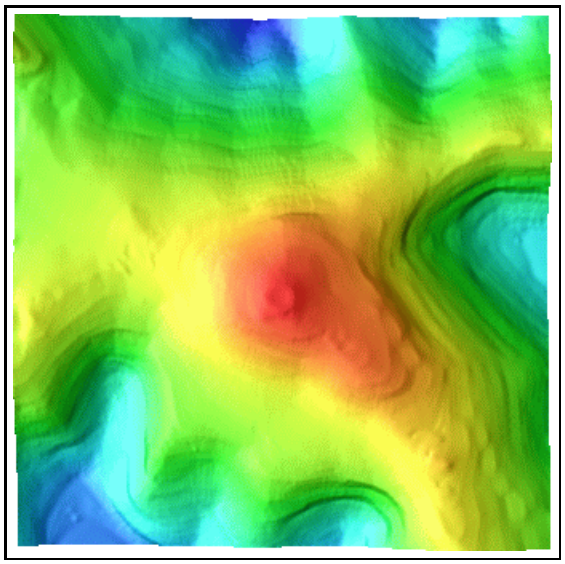

Figure 7.27: Mt. Washington: IC method, top view

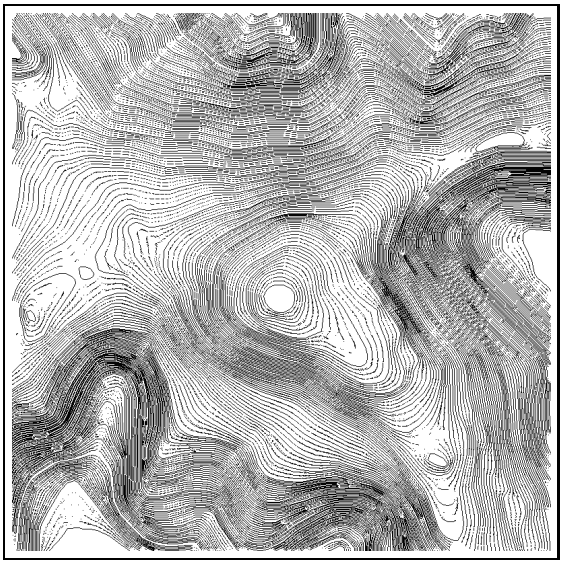

Figure 7.28: Mt. Washington: Intermediate contours

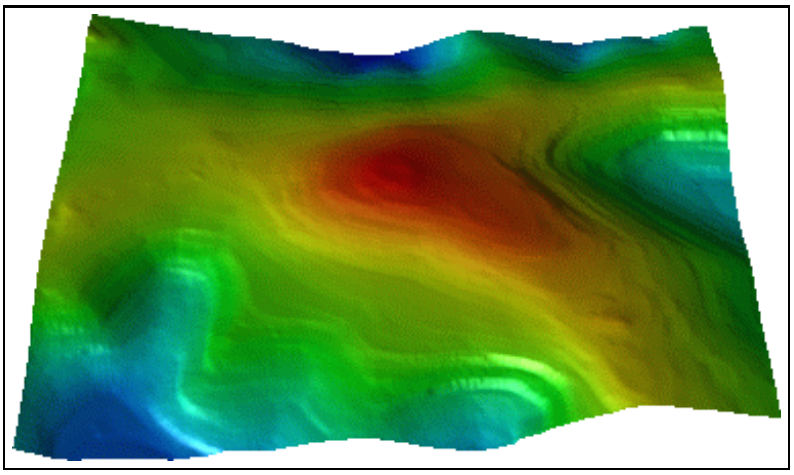

Figure 7.29: Mt. Washington: IC method, angled view

$RMSE_1$ of 1.73 for the MIC method corresponds to an error of only 1.7 meters. The USGS RMSE standards for the highest level DEMs is one-third contour interval, not to exceed seven meters. The contour interval of the Mt. Washington data is 20; a $RMSE_1$ of 1.7 is well below the USGS standards. Figure 7.38 shows the curvatures of all of the methods.

Figure 7.30: Mt. Washington: Curvature using IC method

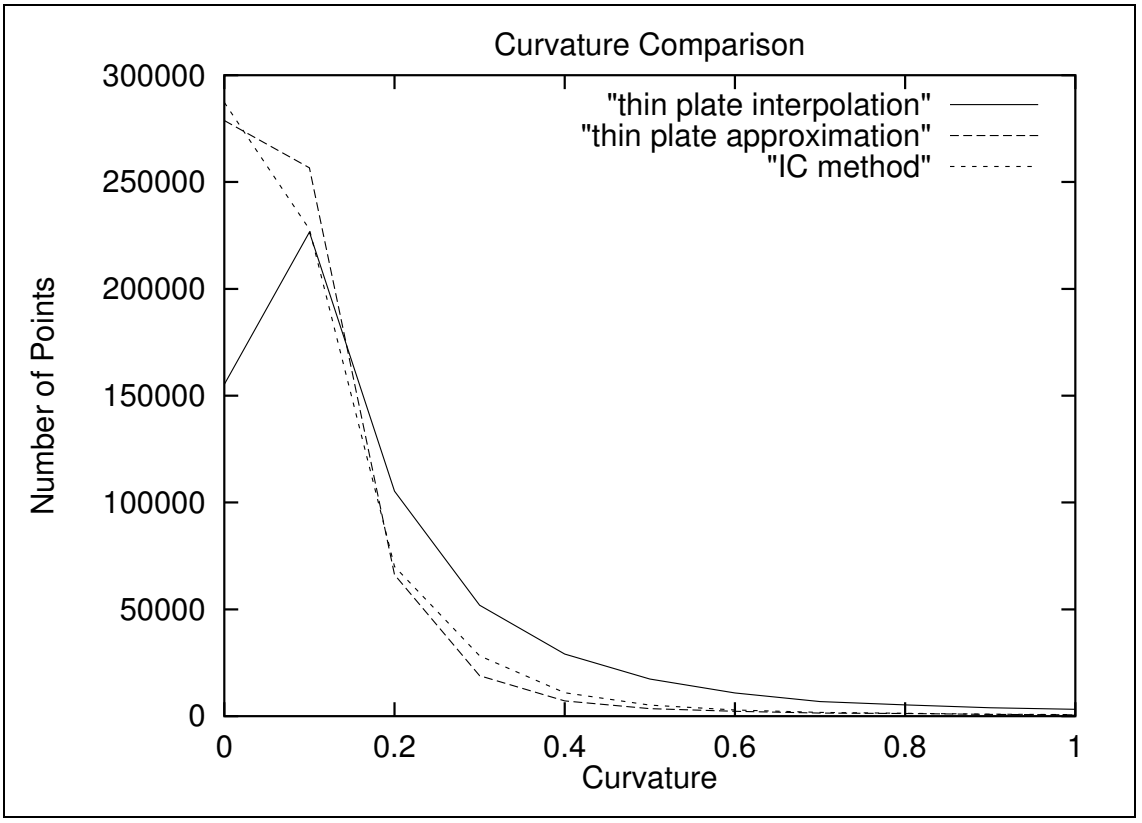

Figure 7.31: Mt. Washington: Plot of curvatures comparing first three methods

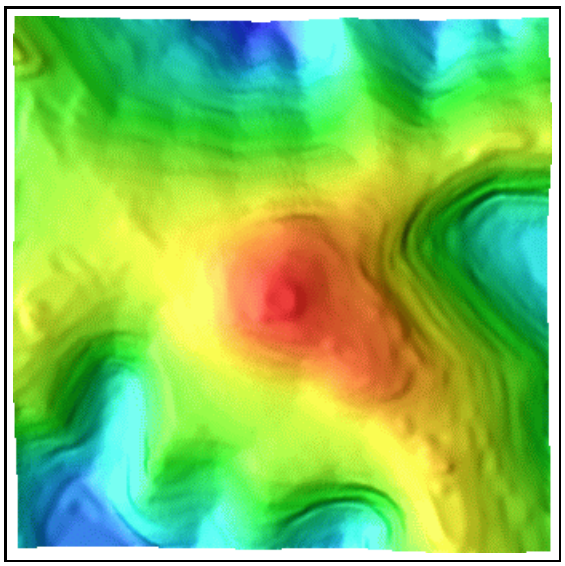

Figure 7.32: Mt. Washington: Gradient Lines method, top view

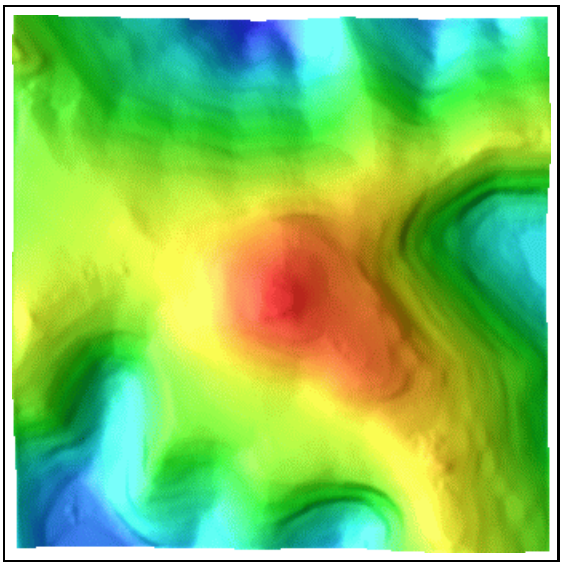

Figure 7.33: Mt. Washington: MIC method, top view

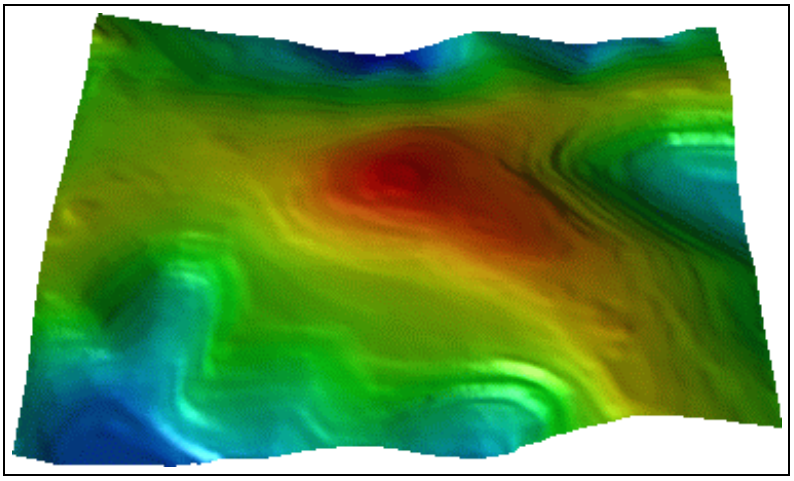

Figure 7.34: Mt. Washington: Gradient Lines method, angled view

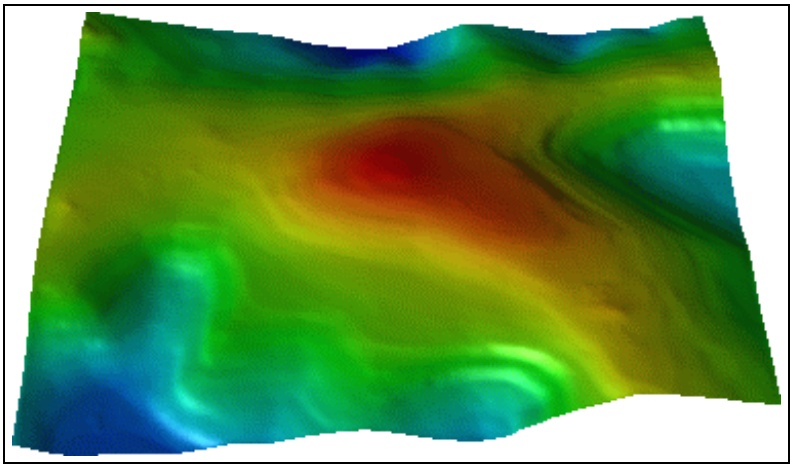

Figure 7.35: Mt. Washington: MIC method, angled view

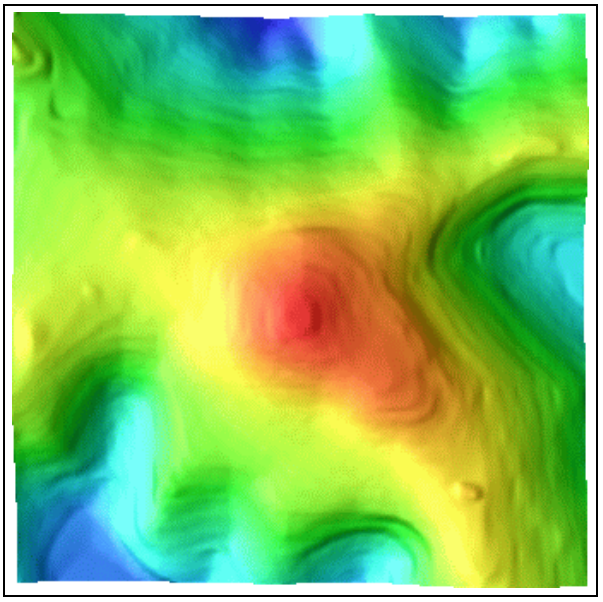

Figure 7.36: Mt. Washington: Fast Spline method with approximating Gaussian smoothing, top view

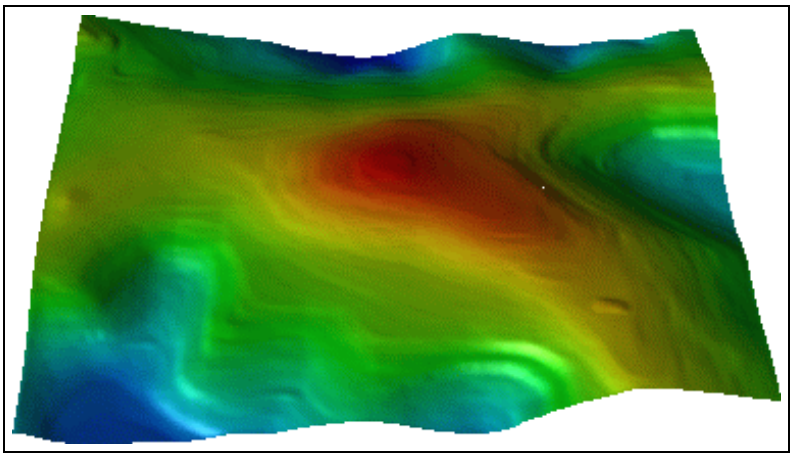

Figure 7.37: Mt. Washington: Fast Spline method with approximating Gaussian smoothing, angled view

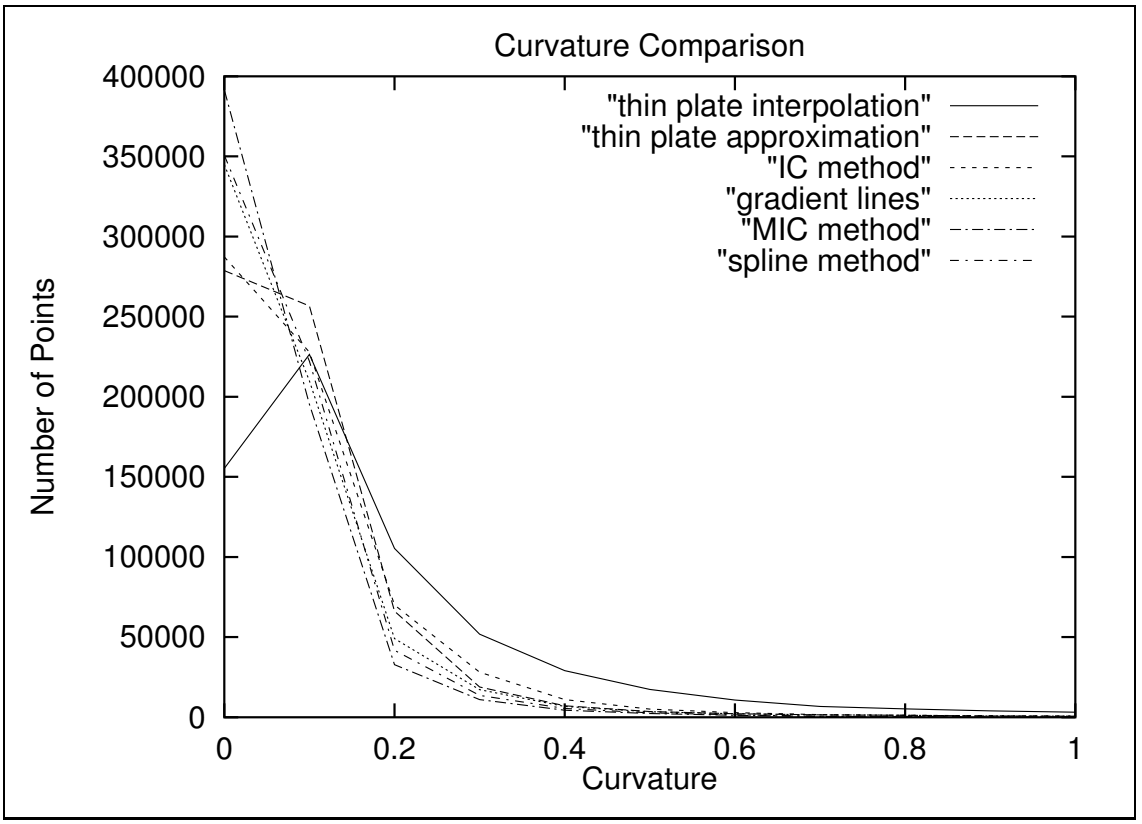

Figure 7.38: Mt. Washington: Plot of curvatures for all methods

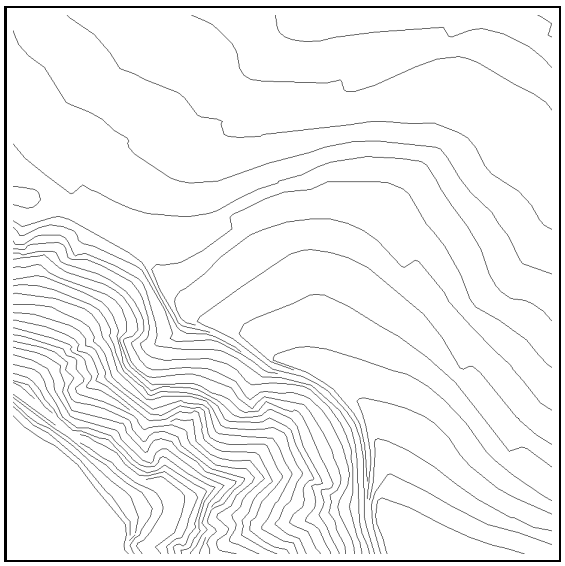

Figure 7.39: Crater Lake: Contours

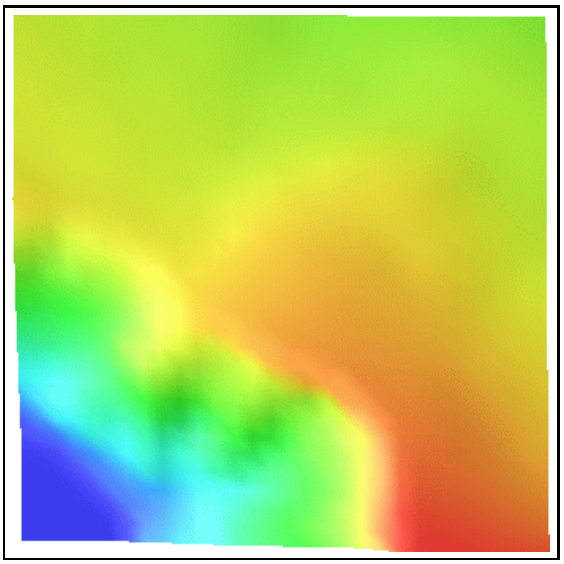

Figure 7.40: Crater Lake: USGS DEM, top view

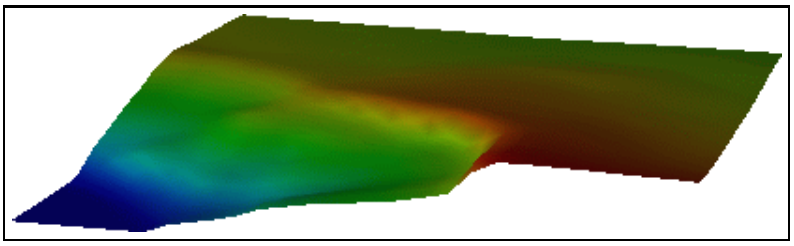

Figure 7.41: Crater Lake: USGS DEM, angled view

7.2.2 Crater Lake, Oregon

This section shows the results of our methods using USGS contour data for Crater Lake, Oregon. The grid size is 900×900, the elevations are in feet ranging from 6200 to 7680, and the contour interval is 40 feet. The horizontal measurements are in meters. The original contours are shown in Figure 7.39. A USGS DEM is available for this contour map. The DEM, which has 30 meter resolution, is shown in Figures 7.40 and 7.41. Recall that $RMSE_2$ in the table refers to the RMSE of the surface compared to the DEM. Furthermore, all methods yield an $RMSE_2$ that is well within the USGS Level 3 quality standards, the highest quality level. The DEM is extremely smooth, but because of its poor resolution, it is not very detailed. The lake is in the lower south-western corner. This is a good test file due to its containing areas of both steepness and flatness.

Method	C^2	C_{ave}	$RMSE_1$	$RMSE_2$
Thin plate interpolation	351121	0.223	0.00	8.75
Thin plate approximation	72987	0.138	1.29	8.69
IC method	93170	0.118	1.93	5.28
Gradient Lines method	72709	0.107	1.29	5.48
MIC method	29089	0.070	4.71	5.13
Fast Spline method	37633	0.082	3.96	5.46

Table 7.3: Results from applying methods to Crater Lake data

Once again we begin by showing the result of the thin plate interpolation in Figures 7.42 and 7.44. The terracing effects are obvious and result in a surface with high curvature as shown in Table 7.3. The thin plate approximation, shown in Figures 7.43 and 7.45, smoothes the surface considerably, as was the case with the previous test files. The area in which the contours are very dense (the south-west corner) is very smooth, while terracing is prevalent in the less steep areas.

As was the case with the Mt. Washington data, two iterations of intermediate contours, shown in Figure 7.46, alleviate the terracing situation. The final IC surfaces are shown in Figures 7.47 and 7.48. Several anomalies present themselves, however. Notice, for example, the small indentation present just south-east of the center of the surface in Figure 7.47. The weakness of the IC method is the failure of the intermediate contours to be generated in all situations. In Figure 7.46, one can see that new intermediate contours were not generated in the area in question. The small indentation is the result of the thin plate approximation filling the area rather poorly. Other small artifacts are observable in the surface and are representative of the same problem. The MIC method was developed partially to help resolve such problem areas. Finally, because the contours were so widely spaced in the input, some small terracing is still visible in the north-west and north-east corners of Figure 7.47.

The presence of these errors account for the curvatures being higher than the thin plate approximation. However, while the $RMSE_1$ is just slightly higher for the IC method, its $RMSE_2$, that is, the RMSE compared to the USGS DEM, is almost 40% lower. This indicates that the surface as a whole is very accurate, at the

expense of some local roughness. We can also inspect a profile from the thin plate approximation and IC methods, shown in Figure 7.49. This graph clear indicates that the IC method follows the contours much better than the other methods with much less of the terracing phenomena.

The resulting surface from the Gradient Lines method is smoother than the result from the IC method as well as the thin plate approximation, as indicated in Table 7.3. The $RMSE_1$ is also lower than for the IC surface, although the $RMSE_2$ is just slightly higher. The surface itself is free of terraces, and looks very good in the areas containing closely spaced contours. However, the surface exhibits some small bumpiness in the flatter areas, most likely due to poor gradient values computed from the initial thin plate surface. Some of these small anomalies can be seen in the profile plot in Figure 7.56.

The MIC method was run with ten iterations of Gaussian smoothing. The surface created by the method, shown in Figures 7.51 and 7.53, are somewhat similar to the IC method, as might be expected. The Gaussian function does a much better job at creating smooth surface, as indicated in the table. Remarkably, the $RMSE_2$ is still very good and is actually lower than the IC method.

Lastly, we come to the Fast Spline method. Recall that this is the fastest method. Some slight terracing is observable in Figures 7.54 and 7.55. There are also a few small anomalies. However, the curvature remains excellent (see Table 7.3) as does the $RMSE_2$. Finally, the profiles shown in Figure 7.56 show that the Fast Spline method follows the contours nearly as well as the more computationally expensive Gradient Lines and MIC methods. Thus, there is a gain in computational speed in return for slightly decreased accuracy.

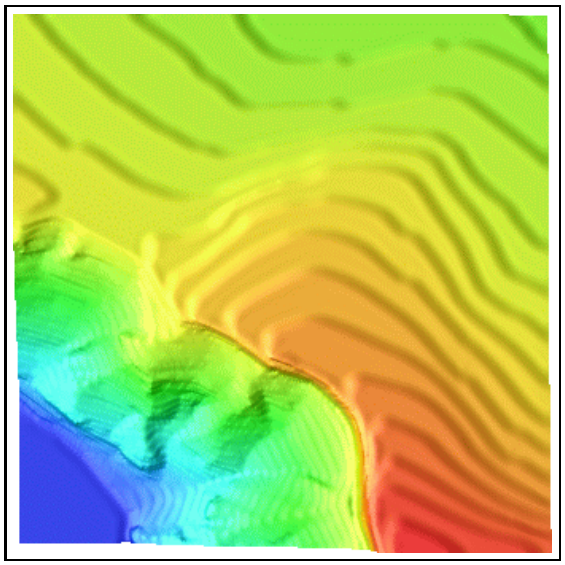

Figure 7.42: Crater Lake: Thin plate interpolation, top view

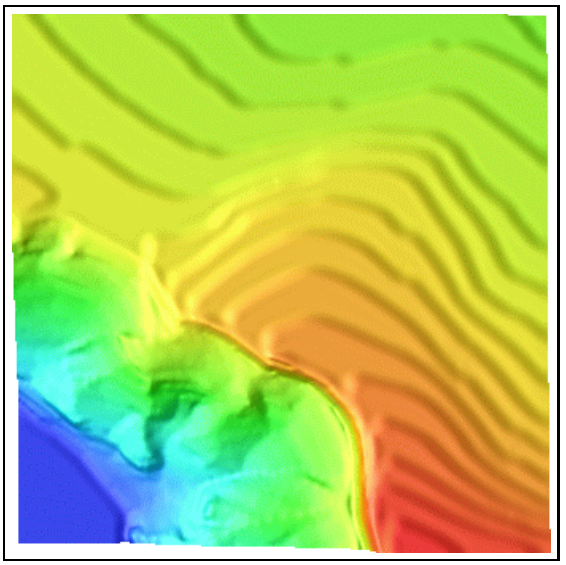

Figure 7.43: Crater Lake: Thin plate approximation, top view

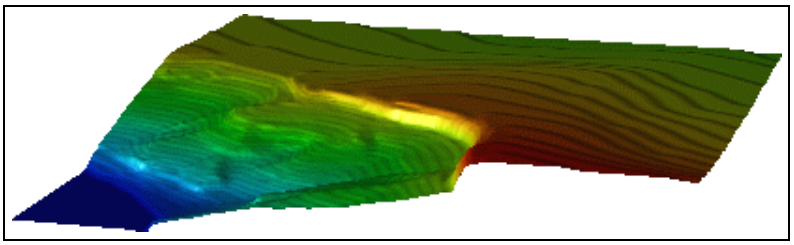

Figure 7.44: Crater Lake: Thin plate interpolation, angled view

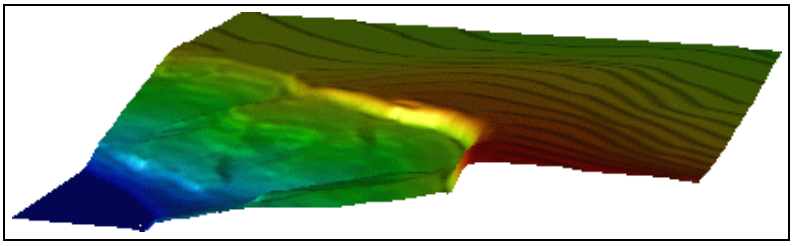

Figure 7.45: Crater Lake: Thin plate approximation, angled view

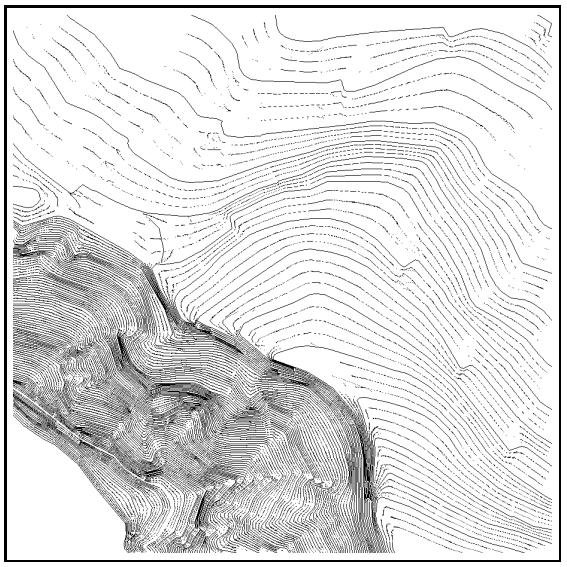

Figure 7.46: Crater Lake: Two iterations of intermediate contours

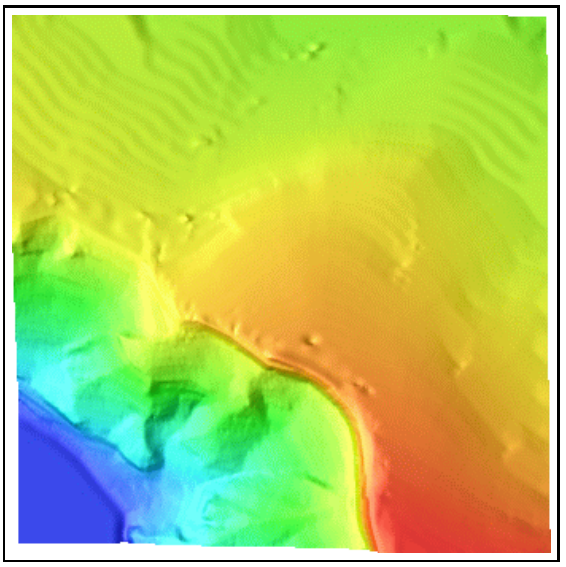

Figure 7.47: Crater Lake: IC method, top view

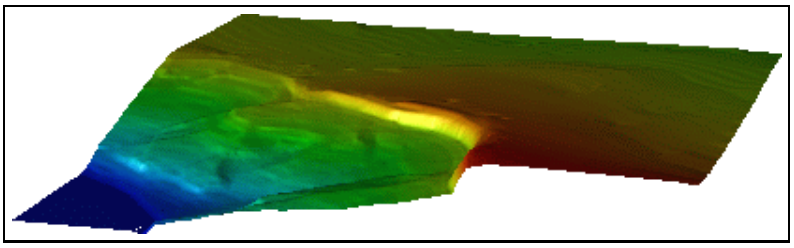

Figure 7.48: Crater Lake: IC method, angled view

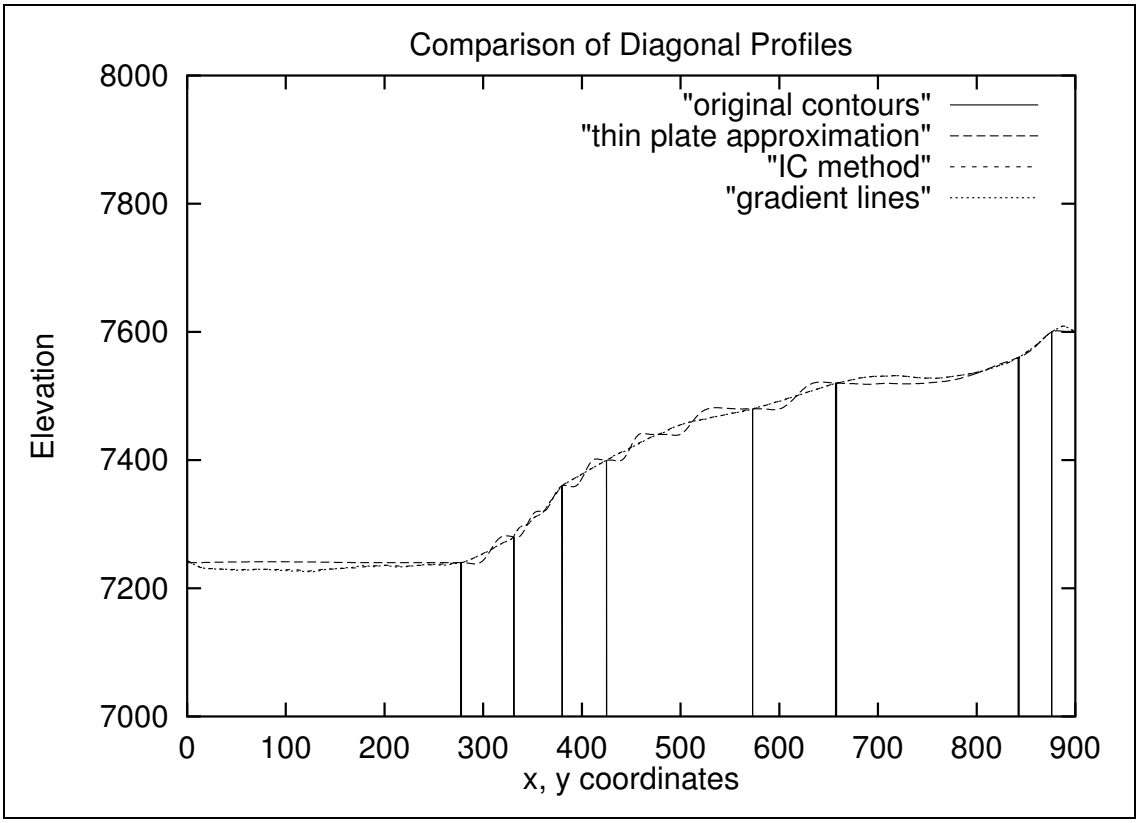

Figure 7.49: Crater Lake: Plot of profiles for first three methods

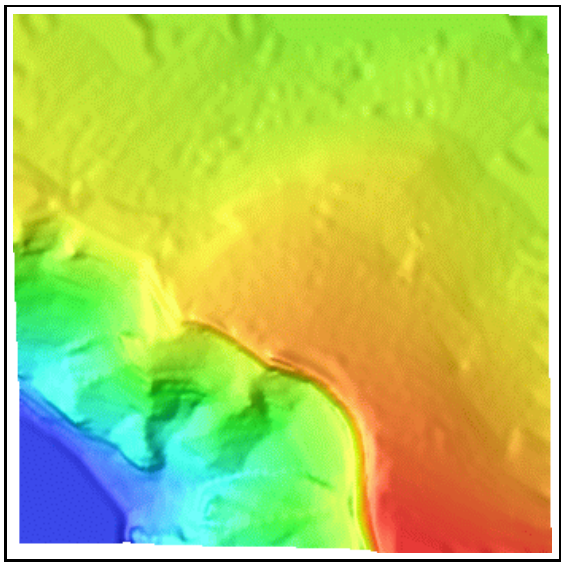

Figure 7.50: Crater Lake: Gradient lines method, top view

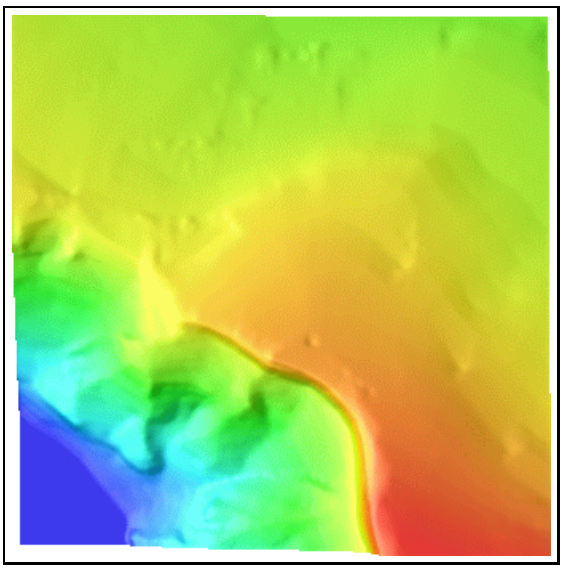

Figure 7.51: Crater Lake: MIC method, top view

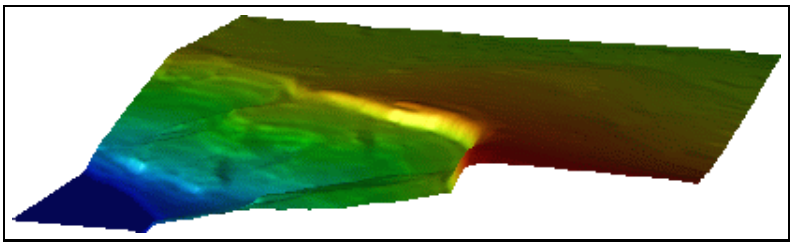

Figure 7.52: Crater Lake: Gradient lines method, angled view

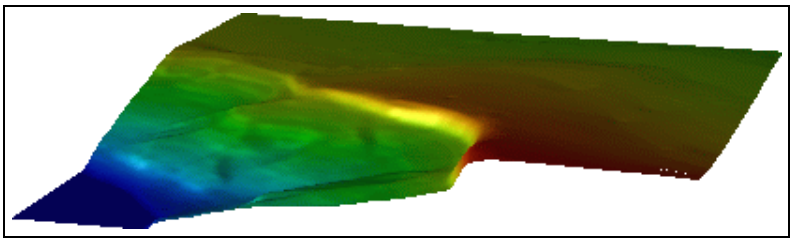

Figure 7.53: Crater Lake: MIC method, angled view

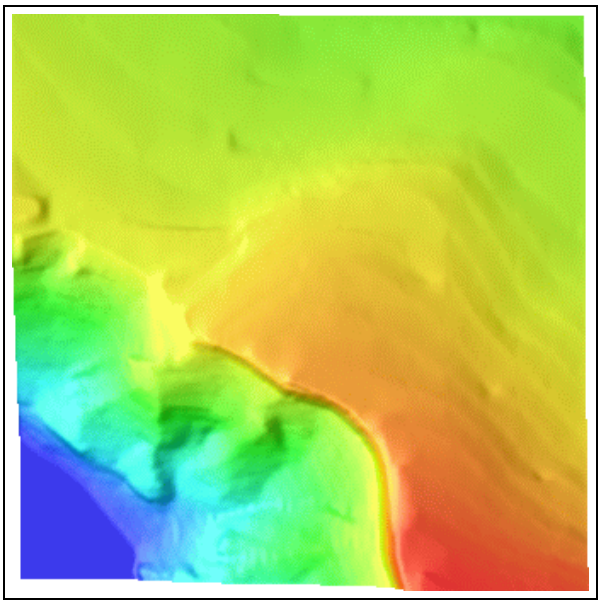

Figure 7.54: Crater Lake: Fast Spline method, top view

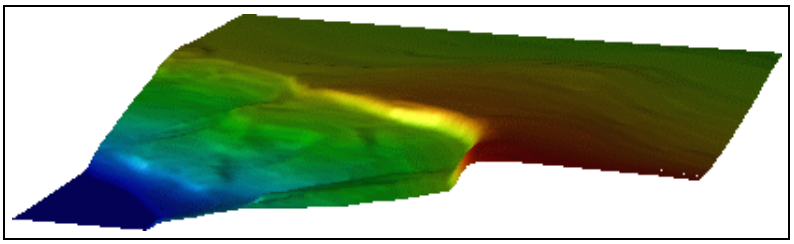

Figure 7.55: Crater Lake: Fast Spline method, angled view

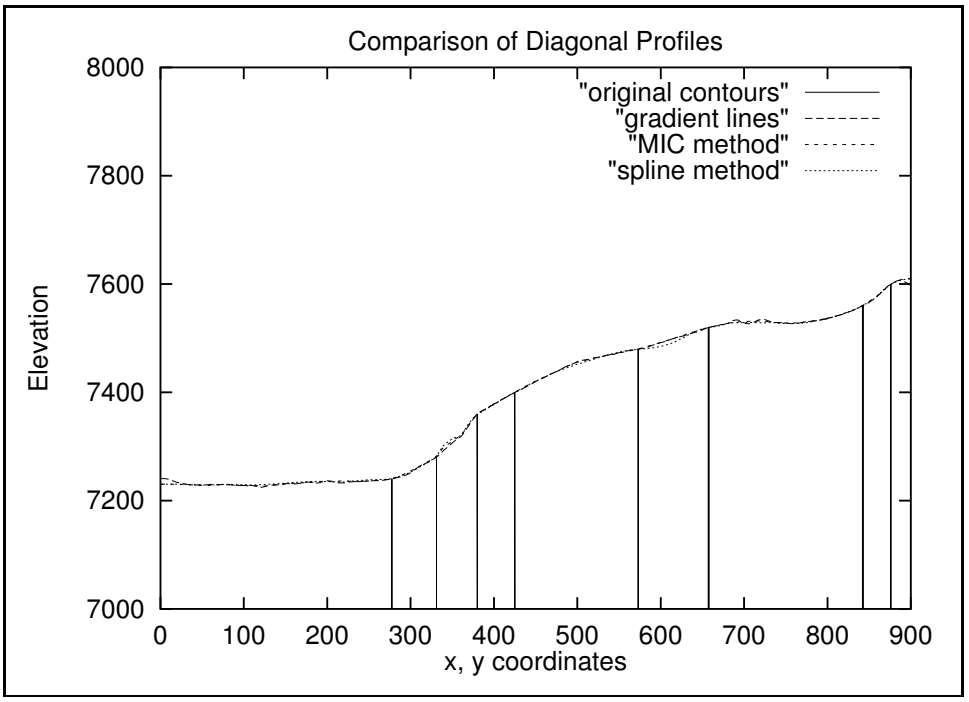

Figure 7.56: Crater Lake: Plot of profiles for last three methods

Method	C^2	C_{ave}	$RMSE_1$	$RMSE_2$	# Errors
Thin plate approximation	45628	0.080	0.058	16.539	40
IC method	41228	0.059	0.458	15.973	8
Gradient Lines method	25894	0.052	0.068	16.425	13
MIC method	12905	0.035	1.483	15.968	5
Fast Spline	29685	0.057	1.240	16.034	6

Table 7.4: Results from applying methods to Bountiful Peak file.

7.2.3 Bountiful Peak, Utah

Bountiful Peak is the area of our last data set. Like the Crater Lake data, it is derived from a USGS DLG, this time from a map representing Bountiful Peak in Utah. For this map, we chose a 2100×2100 grid to test the robustness of our methods. The elevation is in feet, ranging from a low of 6720 to a high of 8700. The contour interval is 40 feet. The horizontal units are in meters. The USGS DEM is shown in Figure 7.57. Just as with the Crater Lake DEM, it is smooth, but has only 30 meter resolution. The original contour data is shown in Figure 7.58. Notice how the contours are more evenly spaced, except in the very southern portions of the map.

Because all of the previous input files have shown that the thin plate interpolation does not produce a good surface, the method is not used in conjunction with the Bountiful Peak data. The thin plate approximation is used as a baseline, and it returns predictable results as shown in Figures 7.59 and 7.60. The contours are obvious once again. Note the curvatures and RMSE values in Table 7.4. The $RMSE_2$ is especially high, although it still falls into the Level 2 USGS DEM quality category. Recall that the maximum RMSE for a Level 3 DEM is one-third contour interval, which in this case is 13.33 feet. The table also contains an additional column which gives the number of grid points that exceed one contour interval (40 feet), which is the maximum allowed error for individual points in a USGS Level 3 DEM. It is possible that the DEM boundaries do not match exactly the DLG boundaries which could account for the rather high $RMSE_2$. However, the comparison of the $RMSE_2$ is still valid across the same input data.

The IC method certainly gives a qualitatively better surface, as shown in

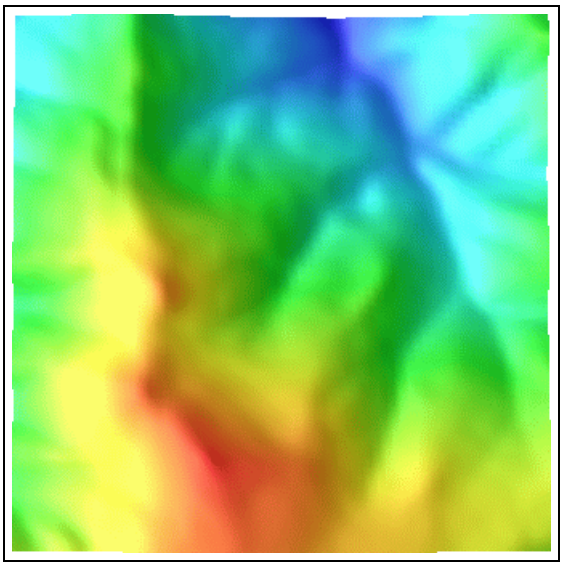

Figure 7.57: Bountiful Peak: USGS DEM

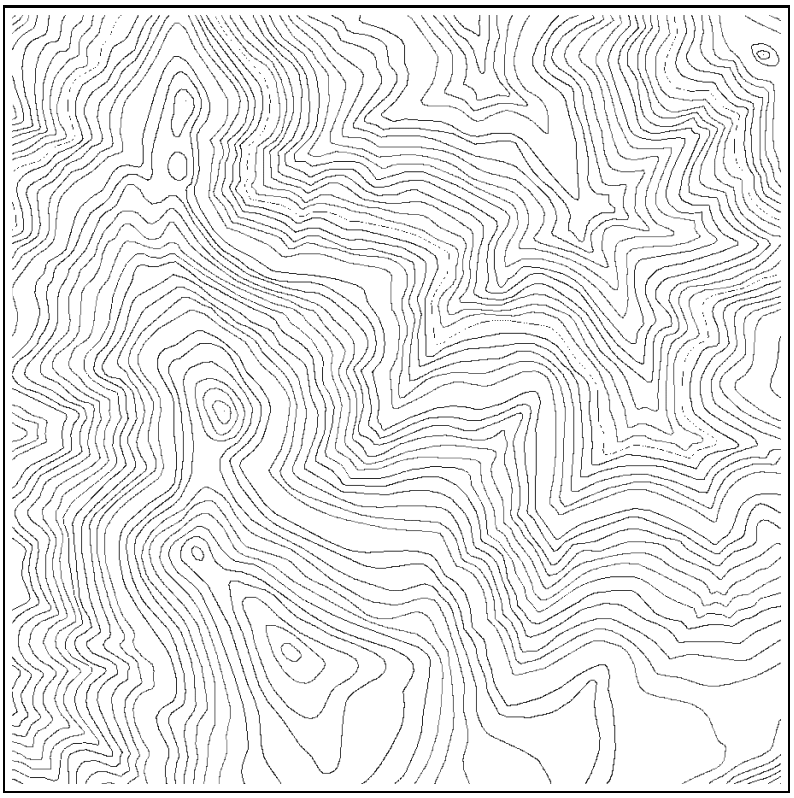

Figure 7.58: Bountiful Peak: Contours

Figures 7.62 and 7.63. Some slight terracing exists in the southern areas where the contours are widely spaced. Compared to the thin plate approximation, the curvature of the IC surface is only slightly improved, as is the $RMSE_2$. The $RMSE_1$ rises somewhat. The profiles, shown in Figure 7.64 are virtually indistinguishable.

Figures 7.65 and 7.66 show the surface resulting from the Gradient Lines method. The surface is generally good, as indicated by its good curvature and $RMSE_1$. However, in areas where the thin plate approximation fails (in the terraced areas in the south), the Gradient Lines method fails as well because the gradients are based on the initial approximated surface.

The MIC method fares very well indeed on this input file. The surfaces are shown in Figures 7.67 and 7.68. Again, the Gaussian smoothing is probably responsible for the excellent curvature. Note that the $RMSE_1$ is slightly higher.

At last, we come to the final test. Figures 7.69 and 7.70 show the surface created by the Fast Spline method. Although the total squared curvature is almost doubled that of the MIC method, it is only slightly higher than the Gradient Lines method. The $RMSE_1$ and $RMSE_2$ figures are also in between the two other methods. The surface has a slightly more terraced look. Again, the accuracy suffers somewhat in exchange for faster computation.

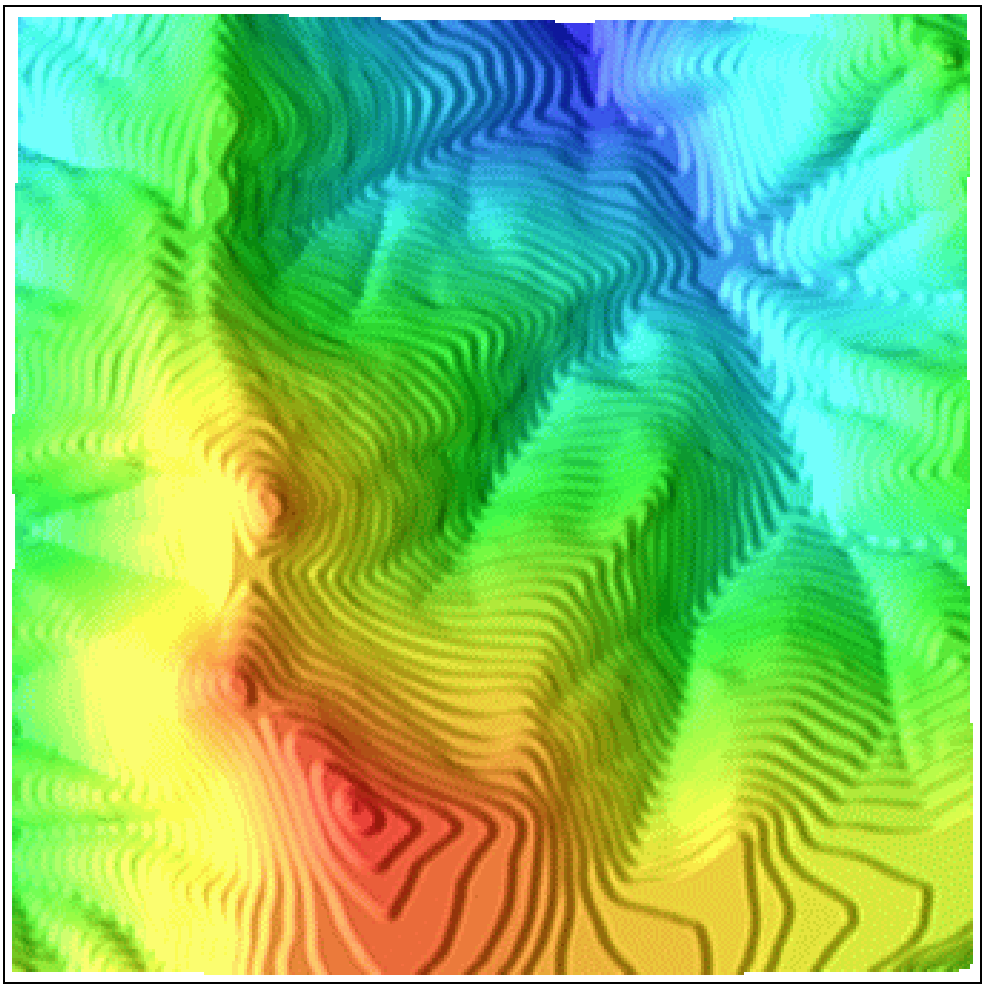

Figure 7.59: Bountiful Peak: Thin plate approximation, top view

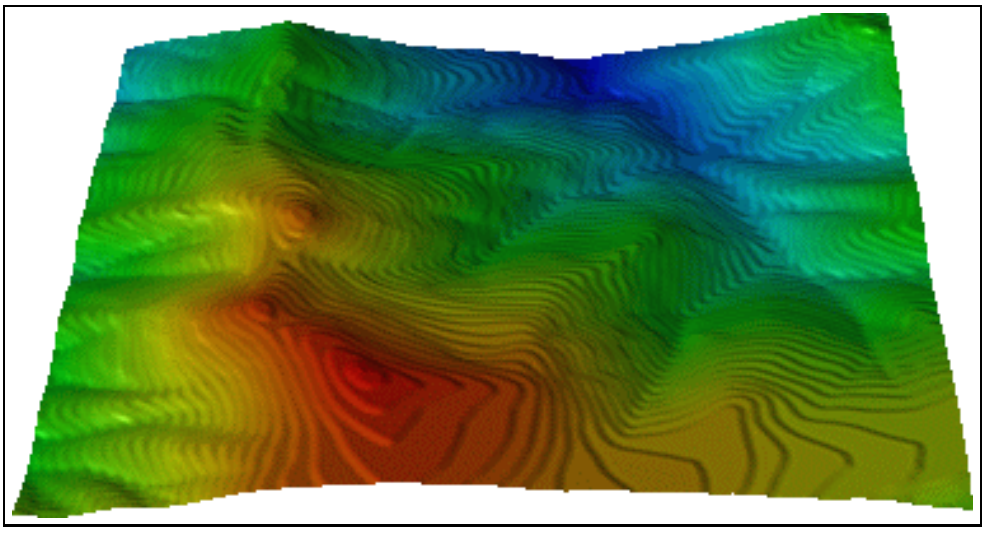

Figure 7.60: Bountiful Peak: Thin plate approximation, angled view

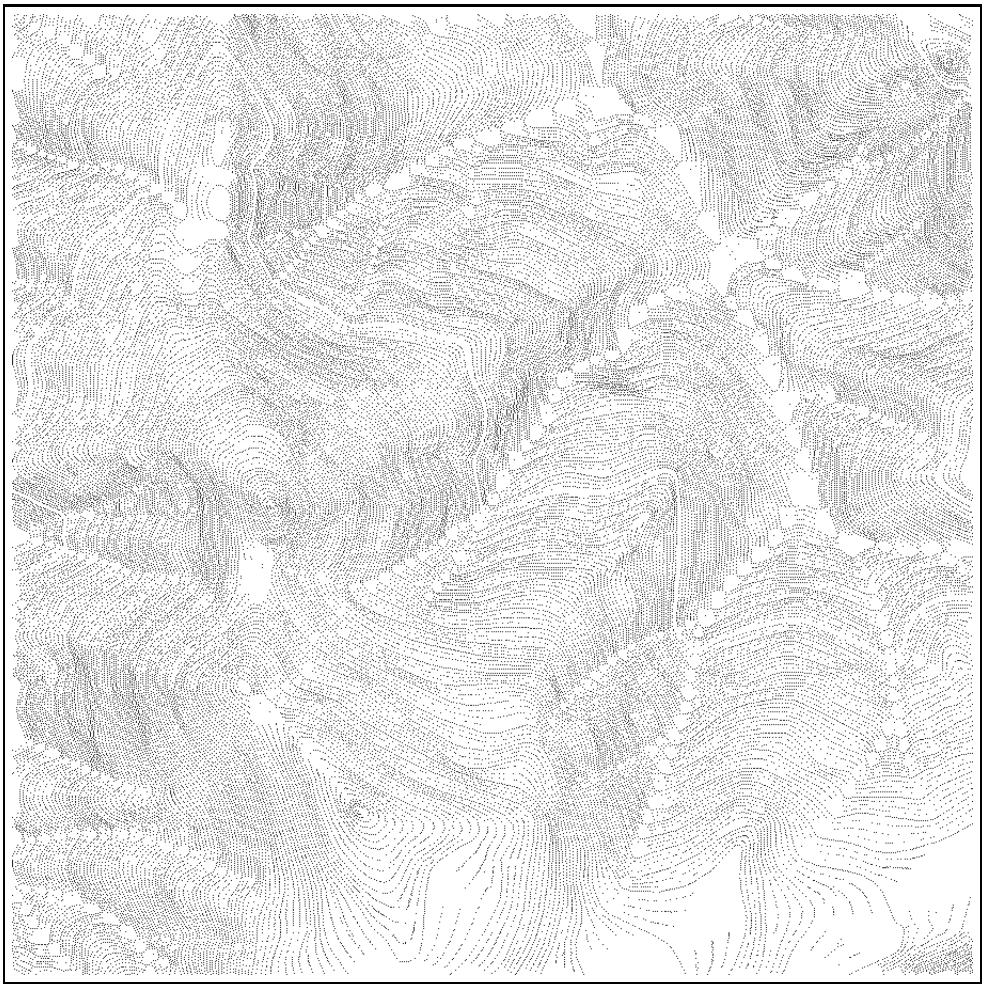

Figure 7.61: Bountiful Peak: Two iterations of intermediate contours

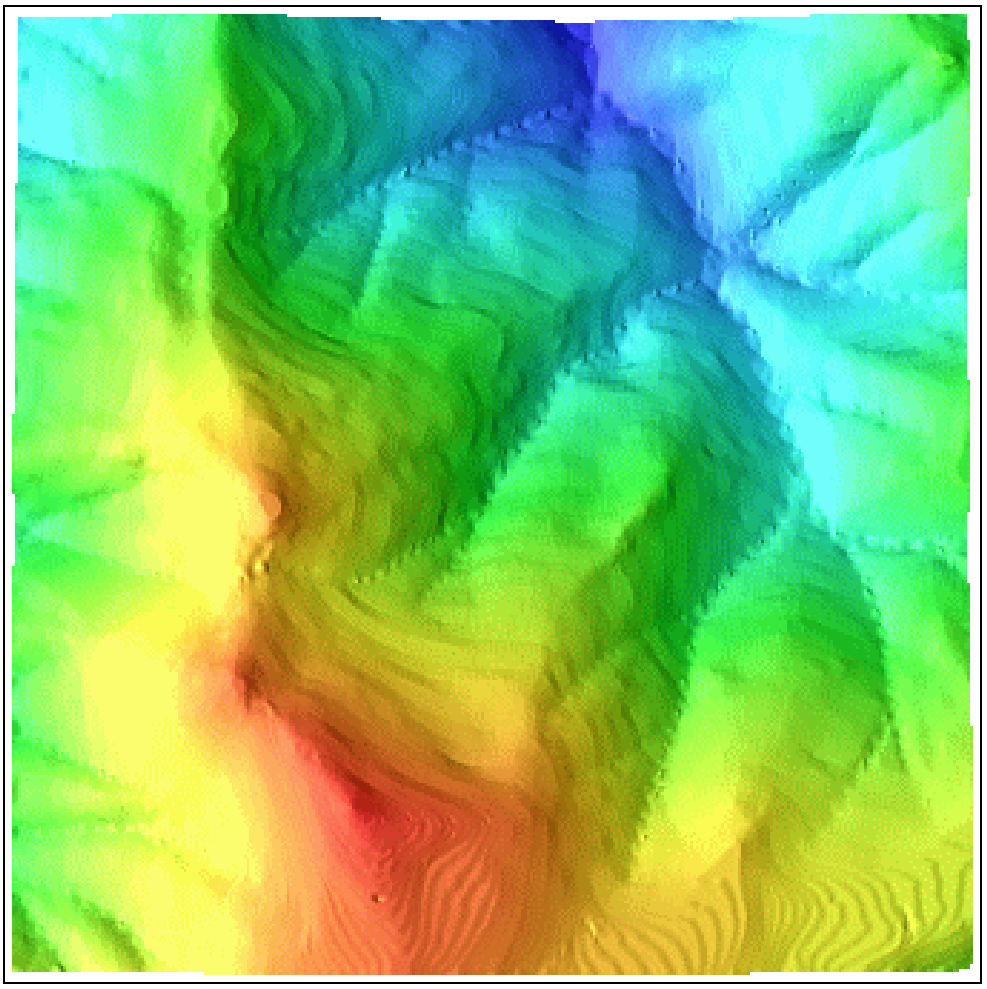

Figure 7.62: Bountiful Peak: IC method, top view

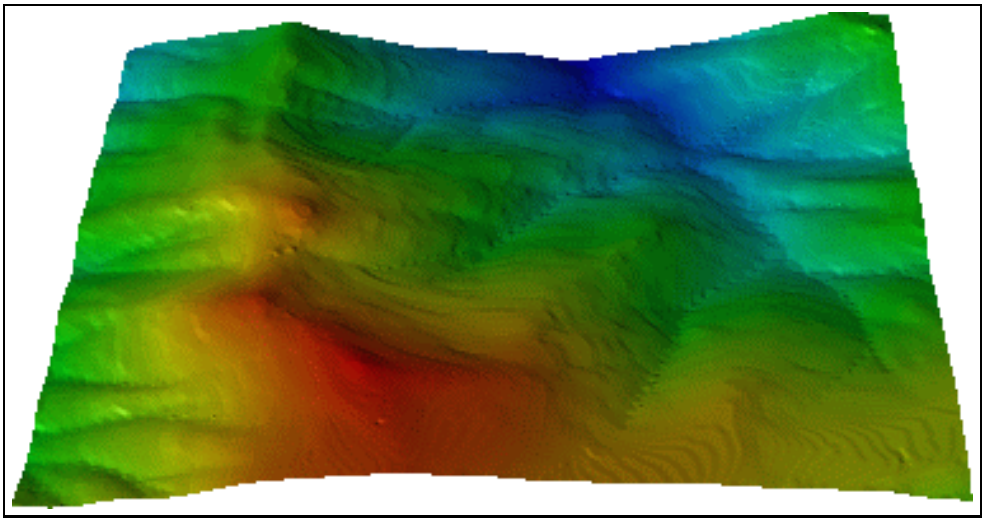

Figure 7.63: Bountiful Peak: IC method, angled view

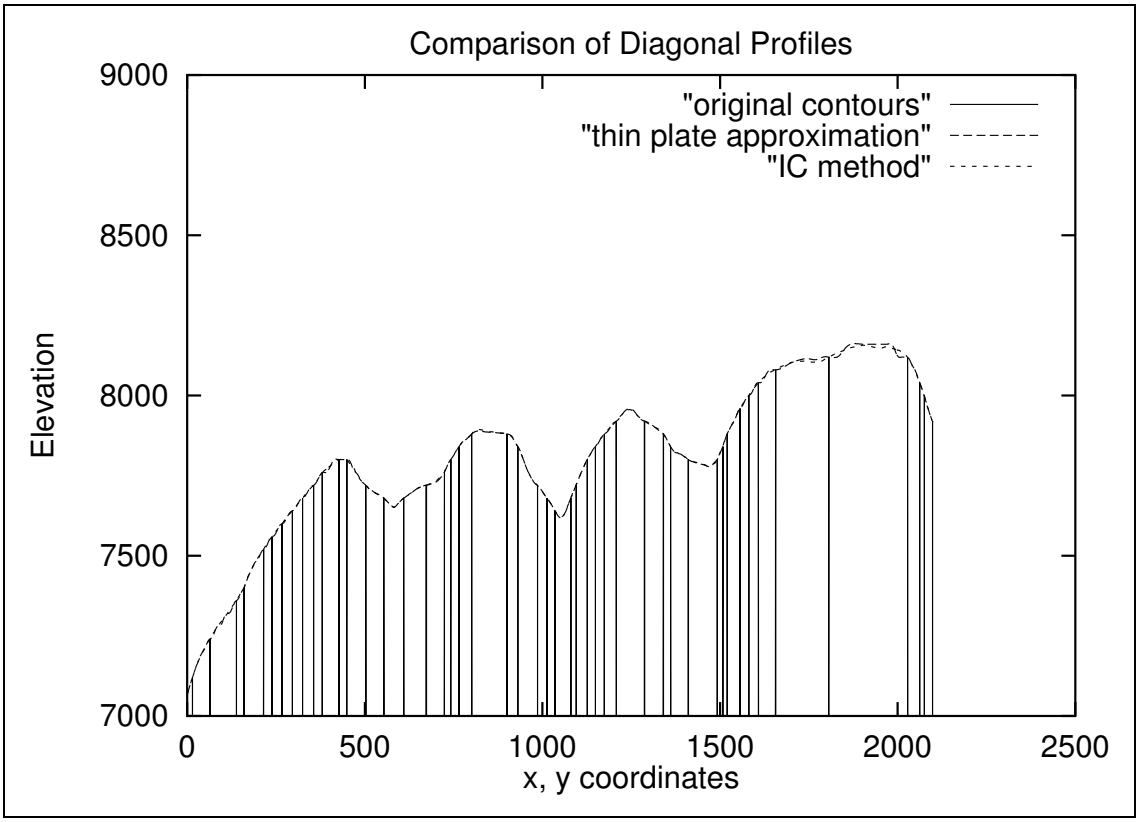

Figure 7.64: Bountiful Peak: Profiles of thin plate approximation and IC methods

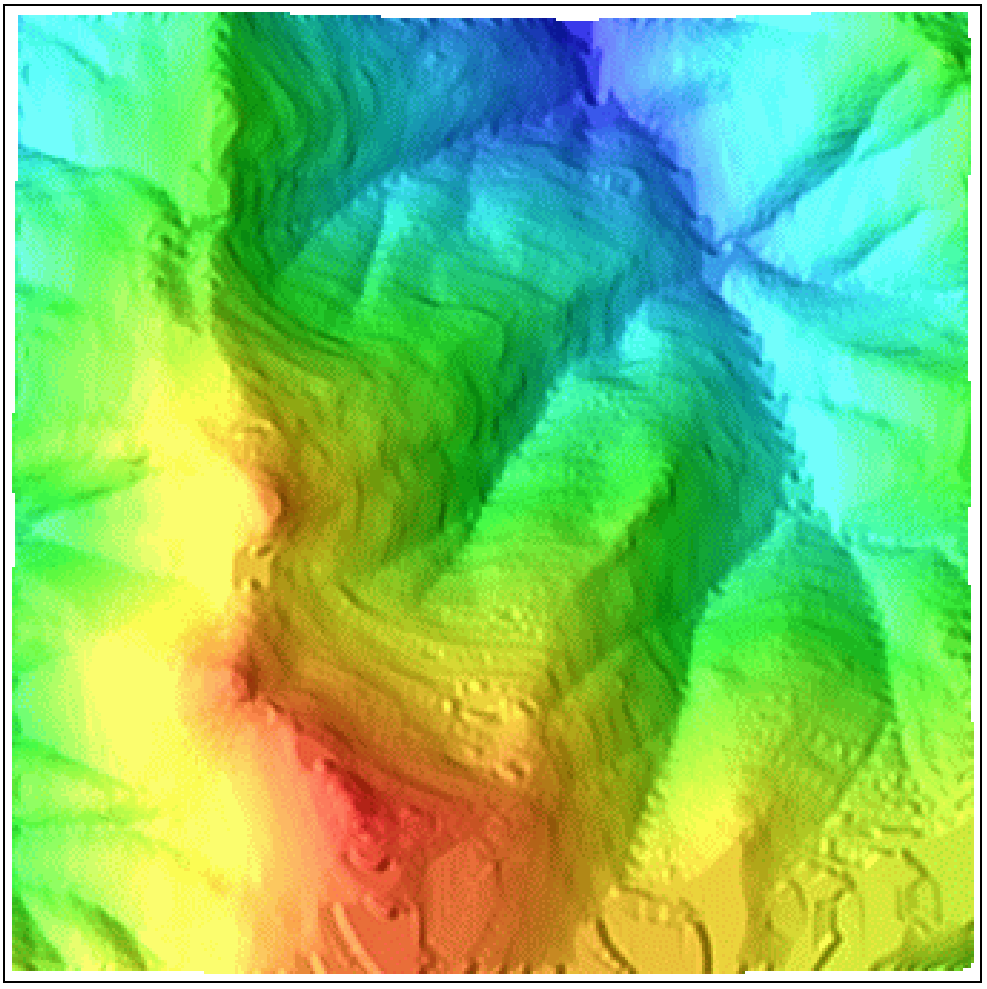

Figure 7.65: Bountiful Peak: Gradient lines method, top view

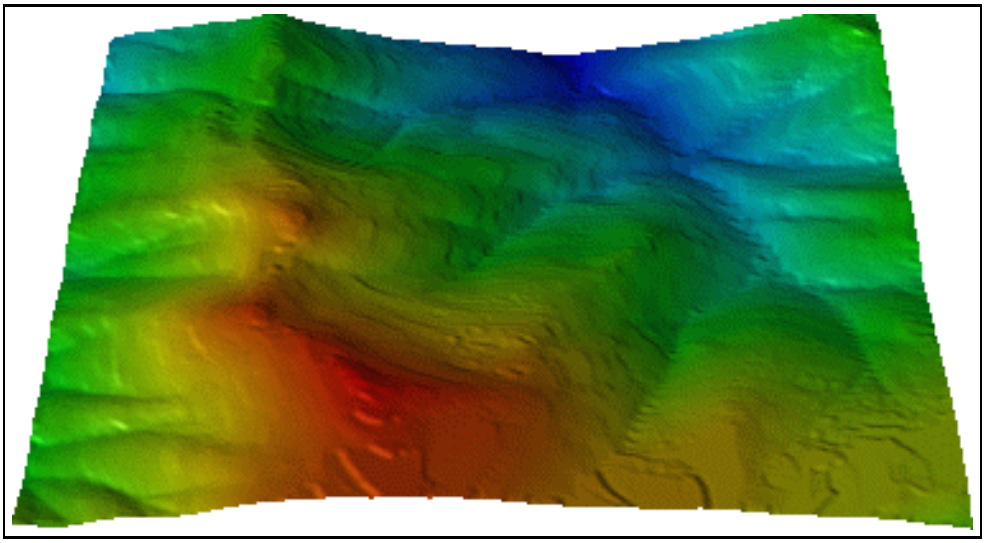

Figure 7.66: Bountiful Peak: Gradient lines method, angled view

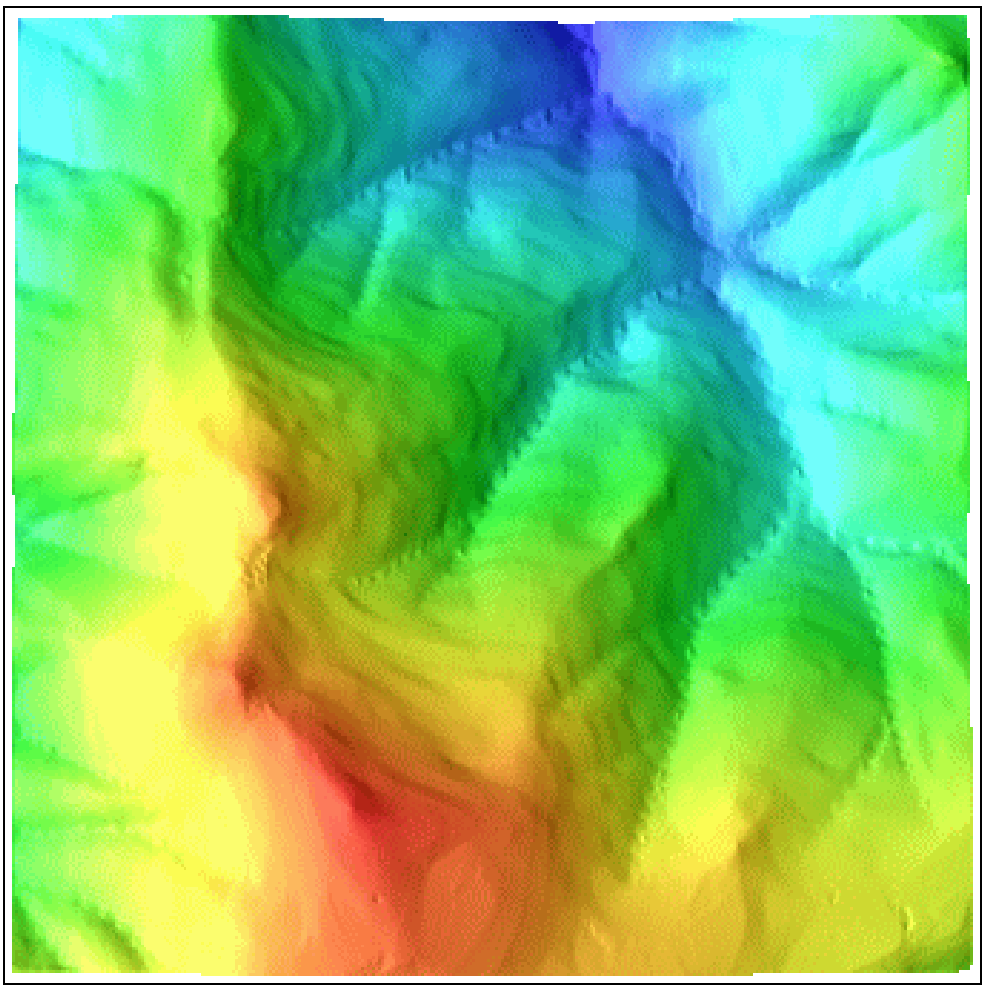

Figure 7.67: Bountiful Peak: MIC method, top view

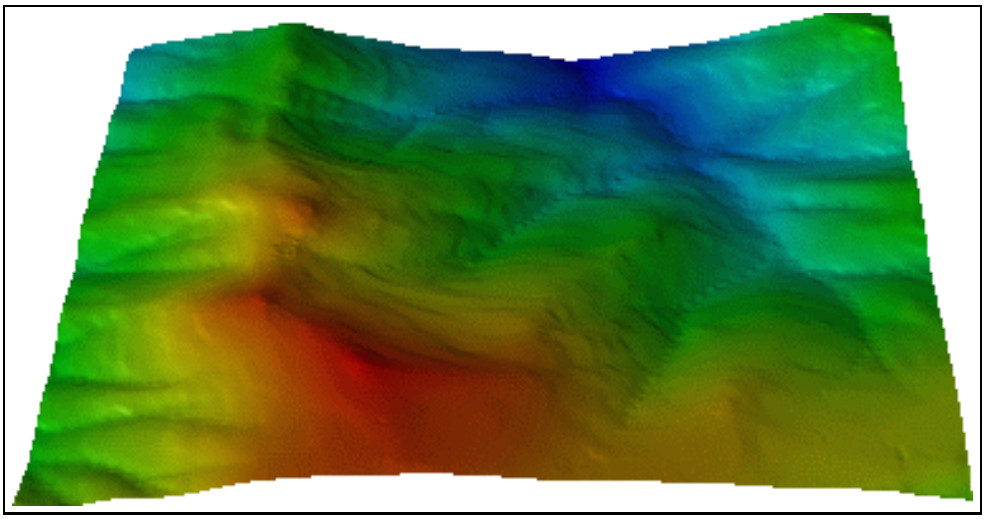

Figure 7.68: Bountiful Peak: MIC method, angled view

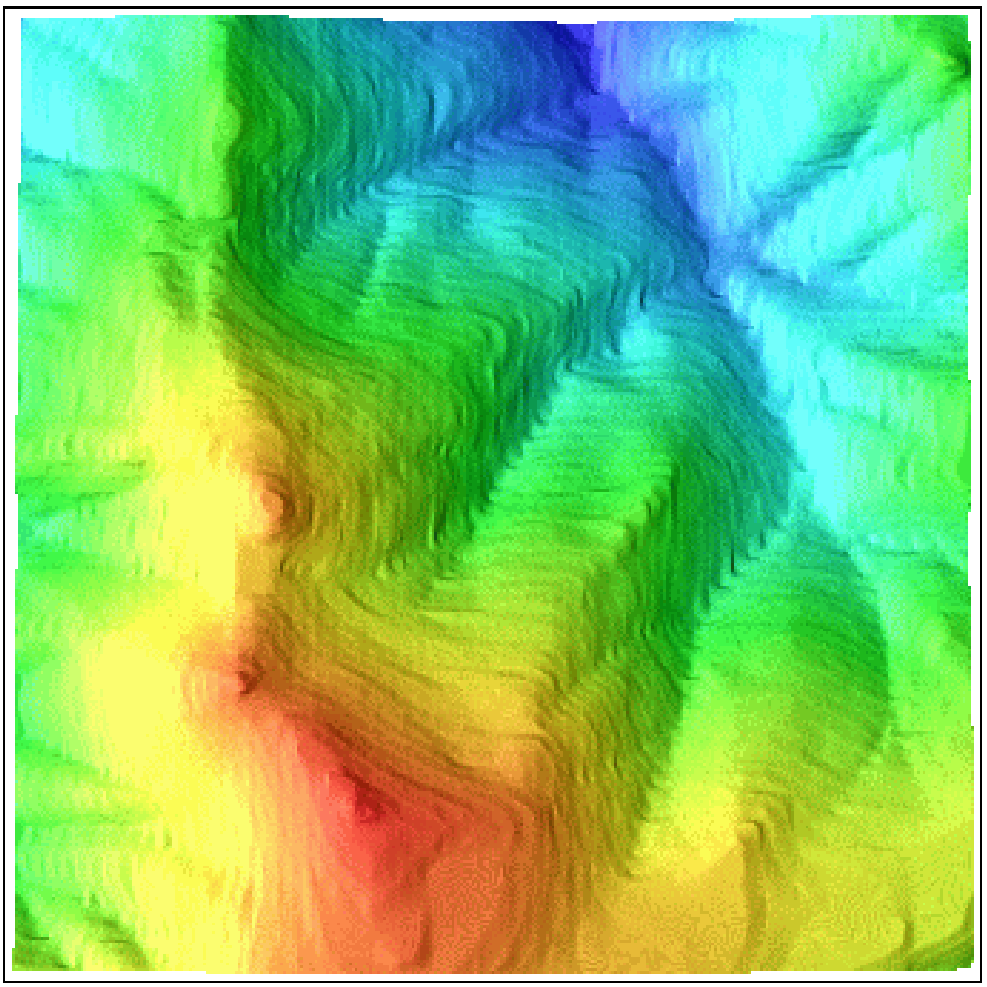

Figure 7.69: Bountiful Peak: Fast Spline method, top view

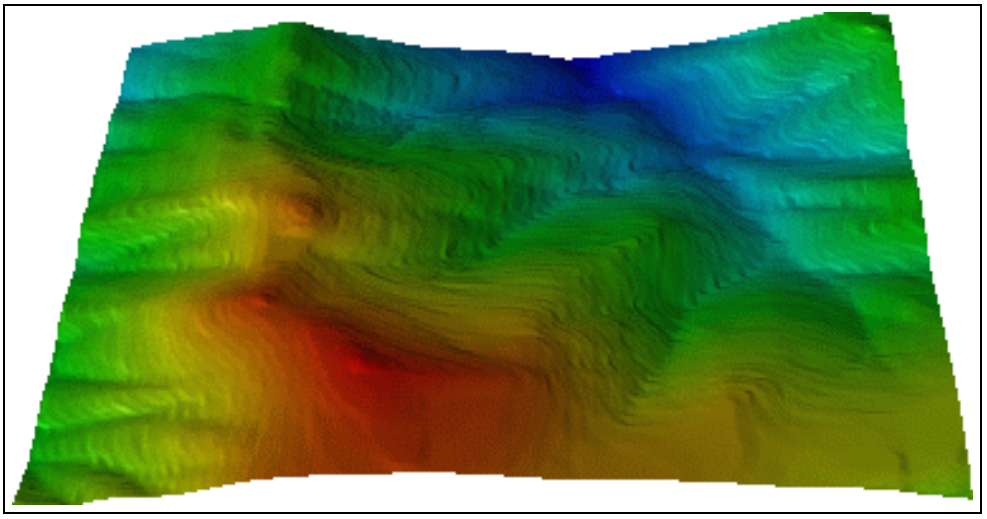

Figure 7.70: Bountiful Peak: Fast Spline method, angled view

CHAPTER 8

All the words created cognitive dissonance.

– Charles Stewart

Research Contributions and Future Work

This chapter summarizes the research contributions contained in this thesis. It will also explore the relative merits and drawbacks of each of the new surface generation methods. We conclude with a discussion of problems and directions for future work.

8.1 Summary of Research Contributions

This thesis discussed the importance of Digital Elevation Models (DEMs) in the Geographic Information Systems (GIS) community, and the relative scarcity of accurate data. While there have been many approaches to DEM and general surface reconstruction, most techniques still produce surfaces with qualitative and quantitative defects. We first presented the method of surface reconstruction using the thin plate model, and concluded that the surfaces so constructed, while smooth, are not very accurate and often exhibit unnatural terracing. We presented four new methods which all provide better surfaces than thin plate methods alone. All of the methods were tested with synthetic and USGS data and analyzed for smoothness and accuracy.

The first to be explored was the Intermediate Contours, or IC, method. The general approach is similar to that reported by Christensen [15], but the computation is done in a much different manner. The IC method creates new contours in between existing contours in order to fill in good elevation approximations in areas where thin plate methods fail. Once the intermediate contours are generated, the thin plate approximation is applied to compute the final, smooth surface. This method fared

very well with the Crater Lake data and not as well with the Mt. Washington and Bountiful Peak data. This is not surprising, as the Crater Lake map exhibits large, flat portions which are prone to extreme terracing. Its computational performance is on par with thin plate methods. Thus, one may conclude that this method is a good choice if the input map contains contours with large, horizontal spacing.

The Gradient Lines method was the second to be discussed. In this new method, paths are created which join local minima and maxima. The paths follow the steepest slope. The potential weakness of this method is that the slope is found via gradients derived from an initial surface. The initial surface is computed using the thin plate approximation technique which is prone to terracing in many circumstances. Thus, an accurate gradient may not always be found, limiting the exactness of the gradient line fit. Once the path is determined, Catmull-Rom splines are used to interpolate which are very accurate and very smooth, having C^2 continuity. The final step in the process is to smooth the global surface with the thin plate approximation. This method proved to be the best of all new methods in regard to $RMSE_1$. The total squared curvature was also better than the IC method for all of the USGS data. Where it fared poorest in terms of total squared curvature was on the Crater Lake data. This makes intuitive sense because this file has the largest flat areas, which in turn causes problems for the thin plate approximation. Therefore, there are flat areas which do not return an accurate gradient estimate. This method is a good choice overall, but especially for files that do not have large flat areas. Its performance of $O(n^3)$ is rather high in exchange for its accuracy.

It was recognized that the IC method had potential but that the thin plate approximation portion did little to contribute to its performance. The MIC method is an outgrowth of the method that needs no thin plate processing. Intermediate contours are generated until the surface is almost completely computed. A new method for computing peaks using Hermite splines was presented. Inverse-distance weighting, normally not a good choice for interpolating, was shown to be a good choice for the small gaps remaining in the surface. Finally, Gaussian smoothing is used to create the final surface. This new method created surfaces that always exhibited the lowest total squared curvature, while still maintaining acceptable $RMSE_1$

and $RMSE_2$ values. In fact, the $RMSE_2$ was lowest for both applicable USGS files. Although it is difficult to quantify, its performance of roughly $O(n^2 \log n)$ is on par with the best thin plate algorithms while producing smoother and more accurate surfaces.

Finally, a new technique employing Catmull-Rom splines in only the horizontal and vertical directions was examined. By interleaving the splines horizontally and vertically, bias in the final surface is eliminated. The splines perform very well, and in conjunction with the Gaussian smoothing filter, generate surfaces that had consistently good curvature measures which were only slightly higher than the MIC method, while returning $RMSE_1$ values that were better. The $RMSE_2$ values were just above the MIC method. Thus, the method performs admirably on all of the files, and has a cost of only $O(n^2)$. In real terms, a surface can be generated by the Fast Spline method in only minutes, compared to any of the other methods which may take hours. For a good surface with an acceptable level of errors, the Fast Spline method is a good choice. For most real-time applications, this method would probably prove adequate for displaying surfaces.

8.2 Future Work

As work progressed on this thesis, it became apparent that many techniques can be employed to reconstruct surfaces from contour data. This thesis explores only a few possibilities. Some general directions for future work involve fitting different portions of the various algorithms together. For example, once a gradient lines surface is computed, instead of creating the final surface with the thin plate approximation, the techniques in the MIC method could be employed. This may generate a smoother surface which would be computationally more efficient.

There are many specific problems which may be addressed in the new methods:

- IC and MIC methods: the intermediate contours are generated by finding the closest neighbor using Bresenham's circle algorithm. This algorithm is not optimal in that it may miss the closest neighbor because of the nature of digitizing in a circular fashion. It is also not very computationally efficient.

A better breadth-first search could be found that would enhance eliminate or enhance both problems.

- Gradient Lines method: while this method is theoretically good, the obvious drawback to this method is its reliance upon the thin plate approximation surface for the gradients. Since it is known that such an approximation is flawed, especially in flatter areas, the gradients must be flawed in those areas as well. Perhaps the use of the Fast Spline method or some other direct method would enhance performance. A second problem is that due to the nature of a regular grid, the gradient direction must be rounded to one of node's eight neighbors.

- Fast Spline method: although smooth surfaces are generated by this method, the $RMSE_1$ and $RMSE_2$ are somewhat compromised due at least in part to the fact that splines are run only in two directions. The addition of diagonal splines may make the surface more accurate while adding little computational cost.

Another research direction is in the mechanics of the surface reconstruction system and the display of output surfaces. At present, the software is command-line driven and very difficult to use, especially for a non-expert. A GUI which would provide a better interface would enhance the system considerably. Furthermore, an especially valuable feature would be the display of surfaces, not only when they are complete, but also as they are computed. For example, it would be interesting to see the result of intermediate contours before any subsequent processing is done. This might show the relative merits or drawbacks of each part of the system, allowing the user to fine-tune the overall surface reconstruction. It may also allow the designer to determine the problem areas in the methods.

It is apparent that current accuracy and smoothness measures are not adequate. New methods for determining the validity of a surface should be explored. This may take the form of looking at additional information in the surface, such as the slope, or other statistical measures to compare surfaces to existing DEMs.

Finally, an interesting method that might apply to the creation of surfaces is

conformal mapping. A conformal map is a transformation of a plane that preserves angles. Contours would be mapped to a surface for which accurate interpolation techniques exist. Once the interpolation is done, an inverse transformation would map the surface back to the original plane.

LITERATURE CITED

[1] *GRASS 4.1 User's Reference Manual*. US Army Corps of Engineers Construction Engineering Research Laboratories (USACERL), Champaign, IL, 1993. `http://www/cecer.army.mil/grass/GRASS.main.html`.

[2] Acevedo, W. First assessment of u.s. geological survey 30-minute dem's: A great improvement over existing 1-degree data. In *Proceedings of the ACSM-ASPRS Annual Convention* (1991), vol. 2, American Congress on Surveying and Mapping, American Society for Photogrammetry and Remote Sensing, pp. 1–12.

[3] Agishtein, M. E., and Migdal, A. A. Smooth surface reconstruction from scattered data points. *Computers and Graphics 15*, 1 (1991), 29–39.

[4] Bolle, R. M., and Vemuri, B. C. On three-dimensional surface reconstruction methods. *IEEE Transactions on Pattern Analysis and Machine Intelligence 13*, 1 (January 1991), 1–13.

[5] Bolstad, P. V., Gessler, P., and Lillesand, T. M. Positional uncertainty in manually digitized map data. *International Journal of Geographical Information Systems 4*, 4 (1990), 399–412.

[6] Bradley, J. xv 3.10a — interacitve image display for the X window system. ftp://ftp.cis.upenn.edu/pub/xv/, 1994.

[7] Breen, D. E., House, D. H., and Getto, P. H. A physically-based particle model of woven cloth. *The Visual Computer 8* (1992), 264–277.

[8] Breen, D. E., House, D. H., and Wozny, M. J. Predicting the drape of woven cloth using interacting particles. *Computer Graphics 28* (July 1994), 365–372.

[9] Briggs, I. Machine contouring using minimum curvature. *Geophysics 39*, 1 (February 1974), 39–48.

[10] Briggs, W. L. *A Multigrid Tutorial*. SIAM, 1987.

[11] Burrough, P. A. *Principles of Geographic Information Systems for Land Resources Management*. Clarendon Press, Oxford, 1986.

[12] Carter, J. R. Some effects of spatial resolution in the calculation of slope using the spatial derivative. In *Proceedings of the ACSM-ASPRS Annual Convention* (1990), vol. 1, pp. 43–52.

[13] Caruso, V. M. Standards for digital elevation models. In *Proceedings of the ACSM-ASPRS Annual Convention* (1987), vol. 4, American Congress on Surveying and Mapping, American Society for Photogrammetry and Remote Sensing, pp. 159–166.

[14] Chaucer, G. The canterbury tales. In *The Riverside Chaucer*, L. D. Benson, Ed., 3 ed. Houghton Mifflin, 1987.

[15] Christensen, A. H. J. Fitting a triangulation to contour lines. In *Auto–Carto 8:Eighth International Sumposium on Computer-Assisted Cartography* (Baltimore, Maryland, 1987), N. R. Chrisman, Ed., pp. 57–67.

[16] Clarke, A. L., Gruen, A., and Loon, J. C. The application of contour data for generating high fidelity grid digital elevation models. In *Auto–Carto 5:Fifth International Sumposium on Computer-Assisted Cartography* (1982), J. Foreman, Ed., pp. 213–223.

[17] Dangermond, J. What is a geographic information system (gis)? In *Geographic Information Systems (GIS) and Mapping – Practices and Standards*, A. Johnson, C. Pettersson, and J. Fulton, Eds. American Society for Testing and Materials, Philadelphia, PA, 1992, pp. 11–17.

[18] David, M. *Geostatistical Ore Reserve Estimation.* Elsevier Scientific Publishing Company, Amsterdam, Oxford, New York, 1977.

[19] Dierckx, P., Suetens, P., and Vandermeulen, D. An algorithm for surface reconstruction from planar contours using smoothing splines. *Journal of Computational and Applied Mathematics 23* (1988), 367–388.

[20] Douglas, C. C. MGNet. `http://www.mgnet.org/`, 1997.

[21] Dubrule, O. Comparing splines and kriging. *Computers & Geosciences 10*, 2-3 (1984), 327–338.

[22] Dubrule, O. Reply: Comparing splines and kriging. *Computers & Geosciences 12*, 5 (1986), 729–730.

[23] Dyn, N., Levin, D., and Rippa, S. Surface interpolation and smoothing by "thin plate" splines. In *Approximation Theory IV*, C. K. Chui, L. L. Schumaker, and J. D. Ward, Eds. Academic Press, 1983, pp. 445–449.

[24] Eklundh, L., and Mårtensson, U. Rapid generation of digital elevation models from topographic maps. *International Journal of Geographical Information Systems 9*, 3 (1995), 329–340.

[25] Enriquez, J. O. C., Thomann, J., and Goupillot, M. Applications of bidimensional spline functions to geophysics. *Geophysics 48*, 9 (September 1983), 1269–1273.

[26] ESRI. Arc/info. http://www.esri.com/.

[27] Fang, L., and Gossard, D. C. Reconstruction of smooth parametric surfaces from unorganized data points. In *Curves and Surfaces in Computer Vision and Graphics III* (1992), vol. 1830, Society of Photo-Optical Instrumentation Engineers (SPIE), pp. 226–236.

[28] Fasshauer, G. E. Adaptive least squares fitting with radial basis functions on the sphere. In *Mathematical Methods for Curves and Surfaces*, M. Daehlen, T. Lyche, and L. L. Schumaker, Eds. Vanderbilt University Press, 1995, pp. 141–150.

[29] Foley, J. D., van Dam, A., Feiner, S. K., and Hughes, J. F. *Computer Graphics: Principles and Practice*, 2 ed. Addison Wesley, 1990.

[30] Franke, R. Scattered data interpolation: Tests of some methods. *Mathematics of Computation 38* (1982), 181–200.

[31] Franke, R. Smooth interpolation of scattered data by local thin plate splines. *Computing and Mathematics with Applications 8* (1982), 273–281.

[32] Franke, R., and Nielson, G. M. Surface approximation with imposed conditions. In *Surfaces in Computer Aided Geometric Design*, Robert E. Barnhill and Wolfgang Boehm, Ed. North-Holland, 1982, pp. 135–146.

[33] Franklin, W. R. Triangulated irregular network program. ftp://ftp.cs.rpi.edu/pub/franklin/tin73.tar.gz, 1973.

[34] Franklin, W. R. Triangulated irregular network program. ftp://ftp.cs.rpi.edu/pub/franklin/tin.tar.gz, 1994.

[35] Franklin, W. R. Compressing elevation data. In *Symposium on Spatial Data Handling* (Orono, Maine, 1995).

[36] Garcia, Antonio Bello, e. a. A contour line based triangulating algorithm. In *Proceedings of the 5th International Symposium on Spatial Data Handling* (Charleston SC USA, 1992), P. Bresnahan, E. Corwin, and D. Cowen, Eds., vol. 2, IGU Commission of GIS, pp. 411–423.

[37] Gold, C. M. Surface interpolation, spatial adjacency and gis. In *GIS: Three Dimensional Applications in Geographic Information Systems*, J. Raper, Ed. Taylor and Francis, 1989, ch. 3, pp. 21–35.

[38] Gold, C. M., and Roos, T. Surface modelling with guaranteed consistency — an object-based approach. In *IGIS'94: Geographic Information Systems*, T. Roos, Ed., Lecture Notes in Computer Science No. 884. Springer-Verlag, 1994, pp. 70–87. http://www.gmt.ulaval.ca/homepages/gold/papers/ascona.html.

[39] Greenspan, D., and Casulli, V. *Numerical Analysis for Applied Mathematics, Science, and Engineering*. Addison-Wesley, 1988.

[40] Grimson, W. A computational theory of visual surface interpolation. *Phil. Trans. R. Lond. B 298* (1982), 395–427.

[41] Grimson, W. An implementation of a computational theory of visual surface interpolation. *Computer Vision, Graphics, and Image Processing 22* (1983), 39–69.

[42] Hannah, M. J. Error detection and correction in digital terrain models. *Photogrammetric Engineering and Remote Sensing 47*, 1 (1981), 63–69.

[43] Hardy, R. L. Multiquadric equations of topography and other irregular surfaces. *Journal of Geophysical Research 8* (1971), 1905–1915.

[44] Heine III, G. W. A controlled study of some two-dimensional interpolation methods. *Computer Oriented Geological Society Computer Contributions 2*, 2 (August 1986), 60–72.

[45] Homer. *The Odyssey*. The New American Library, 1937.

[46] Huber, M. Contour-to-dem: A new algorithm for contour line interpolation. In *Joint European Conference and Exhibition on Geographical Information Proceedings* (Basel, 1995), vol. 1, JEC-GI, pp. 221–222. http://lbdsun.epfl.ch/ huber/ctdjec.html.

[47] Hutchinson, M. F., and Gessler, P. E. Splines – more than just a smooth interpolator. *Geoderma 62* (1994), 45–67.

[48] Jain, R., Kasturi, R., and Schunck, B. G. *Machine Vision*. McGraw Hill, New York, 1995.

[49] Jones, T. A., Hamilton, D. E., and Johnson, C. R. *Contouring Geologic Surfaces with the Computer*. Van Norstand Reinhold Company Inc., New York, 1986.

[50] Jou, J.-Y., and Bovik, A. C. Improving visible-surface reconstruction. In *Proceedings IEEE Conference on Computer Vision and Pattern Recognition* (1988), IEEE, pp. 138–143.

[51] Keppel, E. Approximating complex surfaces by triangulation of contour lines. *IBM Journal of Research and Development 19* (1975), 2–11.

[52] Krakauer, J. *Into Thin Air*. Villard, 1997.

[53] Larson, R. E., and Hostetler, R. P. *Calculus*, 3 ed. D.C. Heath, Lexington, Massachusetts, 1986.

[54] Leberl, F. W., and Olson, D. Raster sanning for operational digitizing of graphical data. *Photogrammetric Engineering and Remote Sensing 48*, 4 (April 1982), 615–627.

[55] Leenaers, H., and Okx, J. P. The use of digital elevation models for flood hazard mapping. *Earth Surface Processes and Landforms 14*, n6-7 (1989), 631–640.

[56] MaGuire, D. J. An overview and definition of gis. In *Geographical Information Systems: Principles and Applications*, D. J. Maguire, M. F. Goodchild, and D. W. Rhind, Eds., vol. 1. Longman Scientific & Technical, 1991, ch. 1, pp. 9–20.

[57] MapInfo. MapInfo Corp. `sales@mapinfo.com`.

[58] Matheson, L. R., and Tarjan, R. E. Analysis of multigrid algorithms on massively parallel computers: Architectural implications. *Journal of Parallel and Distributed Computing 33* (February 1996), 33–43.

[59] Meinguet, J. Multivariate interpolation at arbitrary points made simple. *Journal of Applied Mathematics and Physics (ZAMP) 30* (1979), 292–304.

[60] Norvelle, R. F. Using iterative orthophoto refinements to correct digital elevation models (dem's). In *Proceedings of the ACSM-ASPRS Annual Convention* (1992), vol. 4, American Congress on Surveying and Mapping, American Society for Photogrammetry and Remote Sensing, pp. 347–355.

[61] Oliver, M. A., and Webster, R. Kriging: a method of interpolation for geographical information systems. *International Journal of Geographical Information Systems 4*, 3 (1990), 313–332.

[62] Peuker, T. K., Fowler, R. J., Little, J. J., and Mark, D. M. The triangulated irregular network. In *Proceedings of the DTM Symposium of the American Society of Photogrammetry-American Congress on Survey and Mapping* (St. Louis, Missouri, 1978), pp. 24–31.

[63] Philip, G. M., and Watson, D. F. Comment on "comparing splines and kriging". *Computers & Geosciences 12*, 2 (1986), 243–245.

[64] Powell, M. Thin plate spline interpolation in two dimensions. In *CTAC97, The eighth biennial Computational Techniques and Applications Conference* (1997), Computational Mathematics Group of Australian and New Zealand Industrial and Applied Mathematics Society (ANZIAM), a Division of the Australian Mathematical Society.

[65] Press, W. H., Teukolsky, S. A., Vetterling, W. T., and Flannery, B. P. *Numerical Recipes in C*, 2 ed. Cambridge, 1993.

[66] Rinehart, R. E., and Coleman, E. J. Digital elevation models produced from digital line graphs. In *Proceedings of the ACSM-ASPRS Annual Convention* (1988), vol. 2, American Congress on Surveying and Mapping, American Society for Photogrammetry and Remote Sensing, pp. 291–299.

[67] Robinson, G. J. The accuracy of digital elevation models derived from digitised contour data. *Photogrammetric Record 14* (1994), 805–814.

[68] Rüde, U. *Mathematical and Computational Techniques for Multilevel Adaptive Methods*. SIAM, 1993.

[69] Sandwell, D. T. Biharmonic spline interpolation of geos-3 and seasat altimeter data. *Geophysical Research Letters 14*, 2 (1987), 139–142.

[70] Schumaker, L. L. Fitting surfaces to scattered data. In *Approximation Theory II*, G. G. Lorentz, C. Chui, and L. L. Schumaker, Eds. Academic Press, 1976, pp. 203–268.

[71] Sherwonit, B. *To the Top of Denali: Climbing Adventures on North America's Highest Peak*. Alaska Northwest Books, 1990.

[72] Shmutter, B., and Doytsher, Y. Conversion of contours. In *Auto–Carto 9:Ninth International Sumposium on Computer-Assisted Cartography* (1989), pp. 245–254.

[73] Sibson, R. A brief description of natural neighbor interpolation. In *Interpreting Multivariate Data*, Barnet, Ed. Wiley, 1981, pp. 21–36.

[74] Sinha, S. S., and Schunck, B. G. Surface approximation using weighted splines. In *Proceedings IEEE Conference on Computer Vision and Pattern Recognition* (1991), IEEE, pp. 44–49.

[75] Sinha, S. S., and Schunck, B. G. A two-stage algorithm for discontinuity-preserving surface reconstruction. *IEEE Transactions on Pattern Analysis and Machine Intelligence 14*, 1 (1992), 36–57.

[76] Skidmore, A. K. A comparison of techniques for calculating gradient and aspect from a digital elevation model. *International Journal of Geographical Information Systems 3*, 4 (1989), 323–334.

[77] Smith, W. H. F., and Wessel, P. Gridding with continuous curvature splines in tension. *Geophysics 55*, 3 (1990), 293–305.

[78] SphereKit. `http://ncgia.ucsb.edu/pubs/spherekit`, 1996. National Center for Cartographic Information and Analysis.

[79] Surfer. Golden Software Inc.

[80] Terzopoulos, D. Multilevel computational processes for visual surface reconstruction. *Computer Vision, Graphics, and Image Processing 24* (1983), 52–96.

[81] Terzopoulos, D. The role of constraints and discontinuities in visible-surface reconstruction. In *Proceedings of the Eighth International Joint Conference on Artificial Intelligence* (August 1983), pp. 1073–1077.

[82] Terzopoulos, D. The computation of visible-surface representations. *IEEE Transactions on Pattern Analysis and Machine Intelligence 10*, 4 (1988), 417–438.

[83] United States Department of the Interior: U.S. Geological Survey. Data users guide 1: Digital line graphs from 1:24,000-scale maps, 1990.

[84] WAMC. National Public Radio News broadcast, January 1998.

[85] Watson, D. *Contouring: A Guide to the Analysis and Display of Spatial Data*. Pergammon Press, Oxford, NY, 1992.

[86] Weibel, R., and Heller, M. Digital terrain modelling. In *Geographical Information Systems: Principles and Applications*, D. J. Maguire, M. F. Goodchild, and D. W. Rhind, Eds., vol. 1. Longman Scientific & Technical, 1991, ch. 19, pp. 269–297.

[87] Weil, J. The synthesis of cloth objects. *Computer Graphics 20*, 4 (August 1986), 49–54.

[88] White, J. C. Posting to GIS newsgroup, June 1994.

[89] Wood, J. D., and Fisher, P. F. Assessing interpolation accuracy in elevation models. *IEEE Computer Graphics and Applications* (March 1993), 48–56.

[90] Yoeli, P. Computer executed production of a regular grid of height points from digital contours. *The American Cartographer 13*, 3 (1986), 219–229.

[91] Young, D. M. *Iterative Solution of Large Linear Systems*. Academic Press, New York, 1971.

APPENDIX A

Trying is the first step towards failure.
–Homer Simpson

...since the mind of Odysseus has not wholly failed in you,
there is hope for the future, and I tell you that you will succeed.
– Homer[45]

Failed Methods

This appendix discusses, briefly, some methods that were examined and implemented, but which did not compute adequate surfaces or otherwise provide improvements to reconstruction techniques.

A.1 Pre-processing to Improve Performance

The thin plate methods are known to be computationally expensive. Straight iterative methods may perform as poorly as $O(n^6)$ for a $n \times n$ grid. Although multigrid methods speed the process to $O(n^2 \log n)$ it is difficult to convert algorithms to the process. An observation regarding the thin plate method in general is that it takes quite a while for non-data points to achieve a magnitude similar to the initial data points. Thus, we computed an $O(n^2)$ initial surface by computing points along rows of the grid based on the last known contour elevation. This resulted in a very terraced initial surface that would be smoothed by the thin plate method. The actual result was that the performance was not significantly improved, and that the resulting surfaces were sometimes less accurate.

A.2 Cloth as a Model

An interesting idea was to use cloth as a material to drape over the contours instead of a thin plate. The idea grew out of the papers from Weil [87] and Breen et al [7, 8]. Due to the nature of cloth, it would be able to provide a true interpolation without exhibiting Gibbs phenomena. However, if cloth were to be draped over the contours, it would naturally sag due to gravity, which is incorporated in Breen's model. The idea was to compute the surface upside-down, and let gravity act in an inverted fashion on the surface between contours. Unfortunately, it was not possible to control the "gravity" so as to create a realistic surface. The result was that in areas where thin plate methods typically produce terraces, the cloth model produced upward-facing bulges.

A.3 Pruning Input Contours

Eklundh and Mårtensson [24] suggest that, due to digitization errors, it is helpful to actually eliminate some of the points in the contour data. The decision as to which points to eliminate is somewhat problematic; one idea is to simply remove every n^{th} point along a given contour. The resulting surfaces are very much like the thin plate approximation albeit with less control over the $RMSE_1$.

A.4 Increasing Grid Resolution

A similar idea to the pruning of input contours is to increase the grid resolution. Because many points in close proximity on the same contour can result in flat spots in the surface using thin plate techniques, increasing the resolution would spread out such points. The thin plate would then have additional area in which to flex to create a smoother surface. However, we found that the increased resolution exacerbated the terracing problem by increasing the distance between consecutive contours.

A.5 Weighted Thin Plate Interpolation

One of the deficiencies of the thin plate interpolation or approximation is that there is no provision for preserving slope continuity across the contour. By

definition, the steepest slope is orthogonal to the tangent of a point on a contour. We attempted to modify the thin plate equations to take this fact into account by weighting points along the orthogonal somewhat higher than the rest of the points in the computational neighborhood. Our attempt failed but the approach seems interesting.

A.6 Adaptive Fitting

Fasshauer suggests that some data points on a surface contribute more curvature errors than others [28]. The general scheme is to produce a surface, then discard data points which account for the highest curvatures. The surface is recomputed and the cycle repeated until the desired accuracy is achieved. In applying this to our methods, a thin plate surface was computed, and the data points with the most curvature were discarded. The surface only showed modest improvement until many of the offending (high curvature) points were discarded. Since the points that showed poor curvature tended to be clustered in the same location, many points were deleted from the same contour line which, in effect, deleted the entire contour in that area. This created a surface which therefore ignored significant portions of some contours which resulted in very poor $RMSE_1$ figures.