UNIVERSITY OF READING

DEPARTMENT OF MATHEMATICS

DATA DEPENDENT TRIANGULAR GRID GENERATION

by

ANDREW JOHN MALCOLM

This thesis is submitted for the degree of

Doctor of Philosophy

DECEMBER 1991

Abstract

Until recently, methods of triangulating a domain relied on geometrical constraints, e.g. on the prescribed position of the nodes on the interior of the region. However for the interpolation of functions on the domain such geometrically generated triangulations can produce poor representations of the underlying data. Grids which can well represent underlying data can be achieved by the use of a data dependent grid generation technique.

To improve the quality of representations, methods are presented which generate data dependent triangulations by reconnecting the edges in the triangulation of a fixed set of nodes, by moving the nodes with a fixed connectivity and by generating nodes and triangles concurrently in a triangulation front procedure.

Results of employing each of these methods independently are presented and a strategy involving combinations of them developed. This leads to a general procedure which produces good data-representing triangulations.

The addition of geometric criteria to the reconnection and repositioning procedures shows that, if required by the numerical solution technique, geometrical constraints, e.g. minimum triangle area, can also be accommodated.

Although this work is carried out in the context of the generation of a grid to well represent initial data it has applications in the more general area of function interpolation.

Acknowledgements

I would like to thank Dr. P.K. Sweby for his suggestions and his supervision during the last three years and his patience during the last three months of writing up.

Many thanks are also given to all my colleagues in the Department of Mathematics at Reading for making the last three years so much fun.

Thanks are also due to Helen Davis for her love and support in the last two years.

I take this opportunity to express my gratitude to my parents for their support over many years.

I acknowledge the financial support provided by the SERC for the past three years and the University of Reading Research Board for the last three months.

Contents

Chapter 1

Introduction

The solutions to many mathematical models of physical processes contain local transient features, for example shocks in the supersonic flow past an aerofoil, or the steep fronts present in semiconductor process modelling. In general such mathematical models can not be solved analytically and so numerical solution techniques must be employed. Early computational techniques were based on finite differences on a regular stencil of points. Such techniques, which are employed on quadrilateral grids, have been supplemented more recently by the finite element procedure which may be employed on triangular grids. Initially numerical solution techniques were based on a fixed computational grid which meant that the fine resolution of transient features required a fine grid over the whole region. For large scale problems this led to very wasteful, computationally expensive, procedures.

Recently another set of techniques has gained in popularity. These are adaptive techniques where the grid on which the numerical scheme is solved evolves in conjunction with the numerical solution of the problem being modelled. The ac-

curacy of the representation of the initial data therefore has a major effect on the quality of the solution of the numerical scheme. These techniques have evolved for both quadrilateral and triangular meshes. The quadrilateral meshes are usually used in conjunction with finite difference schemes, while triangular grids are usually used for finite element, or finite volume, methods. These schemes require a good initial grid to produce accurate results. One technique used to generate initial grids is to start with a globally coarse mesh and to refine the subcells of the mesh where refinement is required. This can lead to very complicated data structures and an increase in the computational complexity of the problem. Other techniques which have since evolved are to cluster the nodes in regions where the initial data has a highly transient feature and if necessary to change the connectivity of a mesh at certain points during the solution procedure.

The aim of this work is to investigate and develop techniques which can be used to generate grids which well represent underlying data and which can therefore be used as initial grids for adaptive techniques. The main area of study is triangular grid generation although some of the techniques developed are applicable to quadrilateral grid generation. Although this work is carried out in the context of the generation of a grid to well represent initial data it has applications in the more general area of function interpolation.

There are three possible components to such techniques which are investigated here. These are the generation of nodes, the connection and reconnection of nodes to form triangles and the movement of nodes. Each of these components is investigated separately. The connection and reconnection is studied first, after which the nodal movement is considered and finally the generation of nodes is studied.

Strategies for combining them to form a complete grid generation procedure are then developed.

The connection, and reconnection, of nodes to form triangles has been studied in the literature as part of the scattered data representation problem (e.g. Franke (1979)) and so results from this research area are introduced and studied. The movement of nodes has been used as a post processor where grids have been constructed and the nodal positions are "smoothed" so that the triangles are nearly equiangular. The generation of nodes has usually been performed largely in a geometric fashion so that the mesh points conform to geometric constraints. Triangulation fronts have been used to introduce nodes which are positioned in order that triangles generated by connecting the nodes are nearly equilateral (e.g. Cavendish (1974)). Refinement has also been used so that nodes are placed at the centre of triangles which have too large an area.

The order in which the strategies and procedures mentioned above are investigated is as follows.

In the next chapter we review the literature concerned with the generation of geometrical grids. These grids can be comprised of either triangles or quadrilaterals. In Chapter 3 we review some of the methods which are currently used for generating data dependent grids. The grids which are considered are again triangular or quadrilateral, although the 1-D case is also studied as this provides information for the nodal positioning on the boundaries of the region.

Chapter 4 is devoted to the presentation of the test functions and the sets of nodal positions on which the procedures will be tested. Also presented in this chapter are the measures which will be used to evaluate the quality of the data

representation by the triangulation and the geometrical aspects of the triangulation.

In Chapter 5 we describe the procedures which can be implemented to generate triangular grids using nodal reconnection. The procedures of Dyn, Rippa and Levin (1990) are presented and then a new technique, based on that proposed by Sweby (1987), is outlined. The effect of combining geometrical constraints into these procedures is also shown.

In Chapter 6 two new procedures for the repositioning of nodes in a triangulation with fixed connectivity are presented. The first of these considers the edges in a triangulation as springs and attempts to position the whole system in equilibrium. The second is based on the use of error estimates to position the nodes so that each triangle in the triangulation has the same error. Again the introduction of geometrical constraints is investigated.

In Chapter 7 methods of generating the nodal positions for an initial triangulation are presented and the use of data dependent criteria for these procedures is investigated. Finally results which show the effect of combining the procedures presented in the previous two chapters with the techniques detailed in Chapter 7 are given.

In the final chapter we summarise the work done in this thesis and suggest possible areas of further research.

Chapter 2

Grid Generation Techniques

In this chapter various techniques of grid generation which have evolved to fill the need to cover a general spatial region with a computational grid are surveyed. The different techniques which generate either quadrilateral or triangular grids are reviewed. The strategies which are described in this chapter use only geometrical constraints to generate the grids, while data dependent grid generation, where the representation of the data upon the grid is also taken into account, is described in Chapter 3.

2.1 Quadrilateral Grid Generation

Until recently numerical schemes used to approximate mathematical models of physical situations were predominantly finite difference schemes which, to produce the required accuracy, utilise a fine uniform mesh of nodes, (Ames (1972)). With the increasing geometrical complexity of regions on which problems are posed, e.g. fluid flow around aircraft bodies and aerofoils, and the increased mathematical complexity of problems to be solved, e.g. high speed fluid flow round an obstacle,

changes in both solution and meshing strategy are required. Owing to large derivatives of the solution variables occurring over small regions in such problems, it is no longer viable to completely cover the whole region with a fine regular mesh to obtain the desired accuracy for the smallest feature. This would lead to an excessive number of points in regions which do not require such a fine grid for accuracy, and the computational cost of solving the problem would be prohibitive.

This problem can be overcome by the use of general quadrilateral grids which have the same inherent structure as uniform square grids, e.g. nodes can be easily labelled and thus neighbours easily found, but the quadrilaterals of the grid do not have to have right angles or straight edges. The main idea behind most quadrilateral grid generation is that for a quadrilateral region in physical (x, y)-space it is possible to produce a transformation into computational (ξ,η)-space onto a uniform grid with nodes at integer values of ξ and η as illustrated in Figure 2.1.

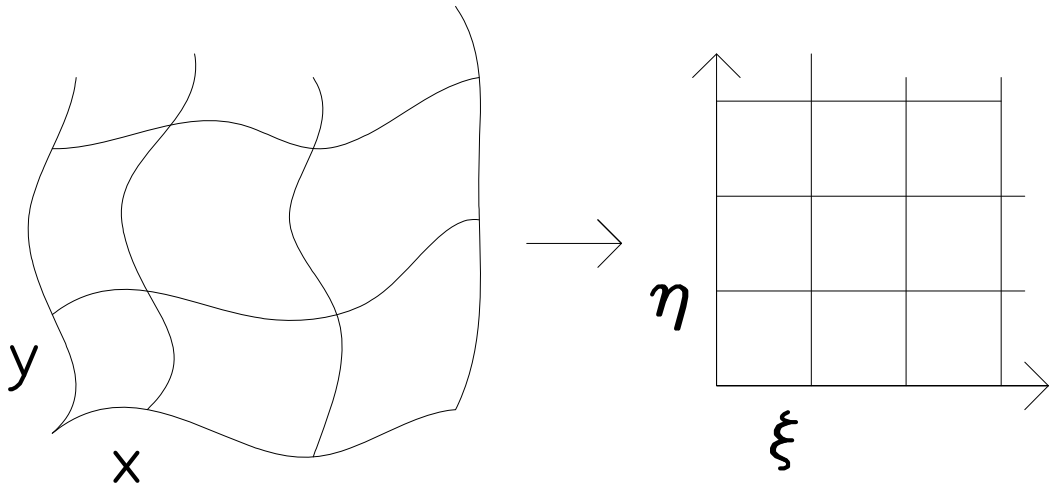

Figure 2.1: Transformation to a regular grid

The advantage of such a principle is that irregular domains may easily be

gridded by transforming them into computational space and overlaying a regular grid onto a now regular domain. This grid can then be transformed back into physical space and the differential equations approximated on it, or alternatively the differential equations themselves can be transformed into (ξ,η)-space and solved there.

There are two main strategies for quadrilateral grid generation; algebraic techniques and differential equation techniques. The differential equation techniques generate the grid as a solution of a differential equation which is used to map between (x, y) and (ξ,η)-space. Since the majority of practical problems are solved on finite regions, elliptic grid generators are appropriate since unlike parabolic and hyperbolic equations they require data to be prescribed on the whole boundary, (Thompson, Warsi and Mastin (1985)).

The simplest elliptic grid generation technique is one in which the boundary nodes of the physical region are specified and then used as boundary data for a pair of Laplace equations

$$\nabla^2 \xi = \xi_{xx} + \xi_{yy} = 0 \tag{2.1}$$

$$\nabla^2 \eta = \eta_{xx} + \eta_{yy} = 0 \tag{2.2}$$

which are then solved over the interior of the region to position the remaining nodes.

In practice the dependent and independent variables in these equations are interchanged to enable x and y to be found in terms of ξ and η. The grids produced by these equations are determined solely by the positioning of the boundary nodes, however control functions can be added to the right-hand side to enable

influence to be asserted over the positioning of the internal nodes, i.e.

$$\nabla^2 \xi = \xi_{xx} + \xi_{yy} = P(x, y) \tag{2.3}$$

$$\nabla^2 \eta = \eta_{xx} + \eta_{yy} = Q(x, y) \tag{2.4}$$

where P and Q are functions which will control the grid features required e.g. smoothness, orthogonality and the positioning of grid lines in regions of interest.

These methods are complicated by the fact that P and Q can be difficult to choose before commencing the grid generation and that calculating their values during the generation can be computationally expensive.

Equations such as (2.3) and (2.4) are discretised using finite differences and the resulting algebraic system solved using iterative methods. An extensive survey of such numerical grid generation techniques has been made by Thompson *et al.*

Algebraic grid generation methods are based on interpolation of physical positions (x, y) on the computational (ξ,η) grid. The coordinates of the boundary nodes are specified and interpolation between them used to determine the interior nodes. The simplest interpolation is linear, which produces equidistant grids, however higher order interpolation can also be used by the introduction of fictitious knots (i.e. they are not nodes of the final grid), their position being used to control the placement of nodes within the region. Transfinite interpolation is used for multi-dimensional problems to blend one dimensional interpolation in the separate computational coordinate directions.

For further details on these methods see Eiseman (1987) or Thompson *et al.* Such methods can be fast but orthogonality, smoothness and the control of grid features are difficult to implement and can make the method computationally expensive.

A different class of grid generators contains those based on variational principles, known as variational methods. Variational methods are such that some measure of the grid quality is represented by an integral involving the computational variables, ξ and η. There can be any number of such integrals associated with the grid, all representing different qualities. Once such integrals have been formulated, it is possible, by the use of the variational Euler equations, to produce equations for the grid positions (x, y) in terms of the computational variables (ξ,η). The integrals can be weighted so as to heighten the effect of particular quality measures, (2.5). The Euler equations are discretised and the resulting difference equations solved iteratively using a technique such as Gauss-Seidel.

Brackbill and Saltzmann (1982) first suggested this method and used the following measures of grid quality, in computational coordinates :-

$$\begin{aligned} \text{Smoothness,} \qquad & I_s = \int_D [(\nabla\xi)^2 + (\nabla\eta)^2] dV \\ \text{Orthogonality,} \qquad & I_o = \int_D [(\nabla\xi.\nabla\eta)^2] dV \\ \text{Weighted volume,} \quad & I_v = \int_D w J dV \end{aligned}$$

where $J = x_\xi y_\eta - x_\eta y_\xi$, is the Jacobian of the transformation, and $w = w(x, y)$ is a weight function used to control distribution of the grid, e.g. to enable clustering to a line or point.

The measure of grid quality to be optimised is then,

$$Q = I_s + \lambda_o I_o + \lambda_v I_v \tag{2.5}$$

where λ_o and λ_v are weights chosen to produce a grid with desirable properties.

Such variational methods are becoming increasingly popular in the field of computational fluid dynamics where the increased complexity of 3-dimensions is

easily within the capacity of modern computers, see e.g. Hawkins and Kightley (1989).

An alternative approach to such variational methods has been advocated by Kennon and Dulikravich (1986). In their approach, once the integrals of grid quality have been formulated, rather than using variational principles to produce the variational Euler equations, the grid functional is discretised directly. The discretisation produced is then solved iteratively as before. The idea of direct minimisation was pursued by Kumar and Kumar (1988) who reformulated the measures of grid quality and solved the equations in a pointwise fashion.

One application is the scattered data-set problem, where at most one data point is required in any quadrilateral, close to its centre. Farmer, Heath and Moody (1991) look at this problem and present a method which gives good results.

In the scattered data set problem, data is given at a small number of points, m, and a quadrilateral grid of size n_i by n_j, $(n_i \times n_j > m)$, is to be fitted to the region; data points should be positioned close to the centre of their surrounding quadrilateral and with at most one data point positioned in each quadrilateral. The criteria of grid smoothness and orthogonality are also included in the functional to be minimised, which has the form,

$$G = f + \sum_{m} \nu(\mathbf{z_m} - \mathbf{c_m})^2 + \sum_{i,j} \mu N_{i,j}(1 - N_{i,j}) \tag{2.6}$$

where :-

- f is a measure of smoothness and orthogonality.
- m is the number of data points with position $\mathbf{z_m}$.
- $\mathbf{c_m}$ is the centre of the quadrilateral in which node m is positioned.

- $N_{i,j}$ is the number of points in the (i,j)th quadrilateral.

The functional is calculated for an initial grid and then mesh points are moved on a local basis until a minimum of the functional is found.

Most of the main techniques of geometrical quadrilateral grid generation have been outlined in this section and in the next section some methods of geometrical triangular grid generation are surveyed.

2.2 Triangular Grid Generation

With the increased complexity and irregularity of domains on which numerical problems are solved, grids comprising solely of triangles have become increasingly popular. This is because it is easier to fit irregular boundaries using triangles rather than quadrilaterals. Triangular grids are now used in many fields including surface representation, Computer-Aided-Design, semi-conductor modelling and fluid dynamics.

There are two main strategies for constructing geometrically based triangular grids. These are :-

1. Given a complete set of nodes, triangulate them,
2. Given a set of boundary nodes, generate interior nodes and triangles at the same time.

When such geometrically based triangulations have been produced they can be post-processed in a geometrical manner. This post-processing can consist of

keeping the nodal connections fixed whilst moving the nodes or changing the connectivity while keeping the nodes fixed.

The first procedure is the oldest and most commonly used; usually a set of nodes is generated by some means, and then a triangulation is produced which possesses some desirable property, such as equiangularity. There are many ways of achieving this, one of the most popular being the Delaunay triangulation technique, (Delaunay (1934)). A Delaunay triangulation is the dual of the Voronoi tesselation (also known as a Dirichlet or Thiessen tesselation), see Dirichlet (1850) or Voronoi (1908), which divides the regions into tiles. Each tile encloses a data point and the tile defines a region of the plane which is nearer to its interior node than to any other node.

The Delaunay triangulation is then constructed by joining nodes whose tiles have common edges (see Figure 2.2). The dual property of the Voronoi tesselation and Delaunay triangulation is such that given a Delaunay triangulation one can construct a Voronoi tesselation and vice-versa (Green and Sibson (1978)).

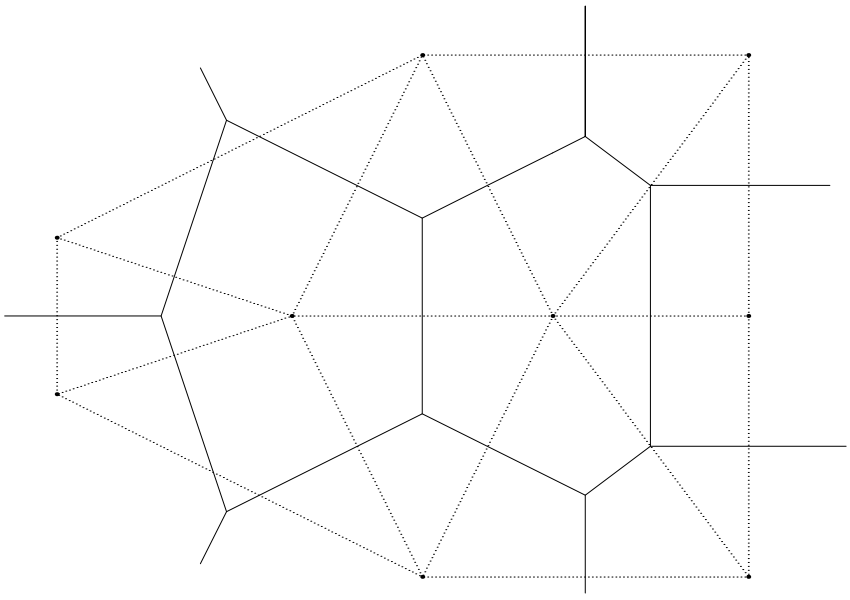

Figure 2.2: A Voronoi tesselation and its associated Delaunay triangulation

Delaunay also demonstrated that this triangulation could be defined in another manner. His formal definition stated that in n-dimensions the circumscribing n-dimensional hypersphere of the n-dimensional grid element contains no points other than those defining the element. In 2-dimensions this means that the circumcircle of a triangle, i.e. the circle which passes through all three vertices of the triangle, contains no other nodes. This definition can however lead to a degenerate case if $n+2$ or more nodes lie on the boundary of the n-dimensional hypersphere; in such a case these nodes can be triangulated by any of the possible connections (see Figure 2.3 for the 2-dimensional case).

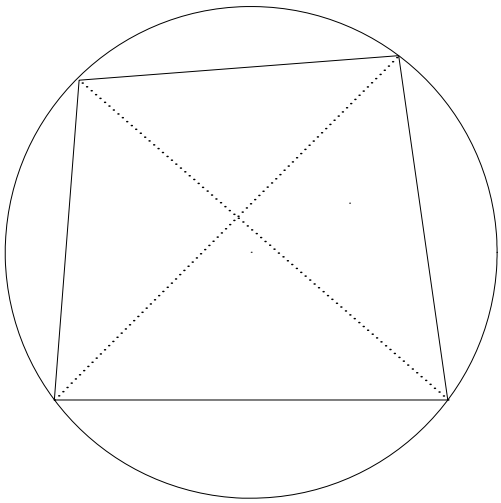

Figure 2.3: Degenerate case of Delaunay

Next some properties of the Delaunay triangulation are described and methods by which Delaunay triangulations can be produced are reviewed.

In Lawson (1977) it is shown that given any initial triangulation it is possible to produce a triangulation with a maximum-minimum angle criterion. This property ensures that in any convex quadrilateral formed by two adjacent triangles, the interior diagonal, see Figure 2.4, is chosen so that the minimum angle in the two resulting triangles is maximised.

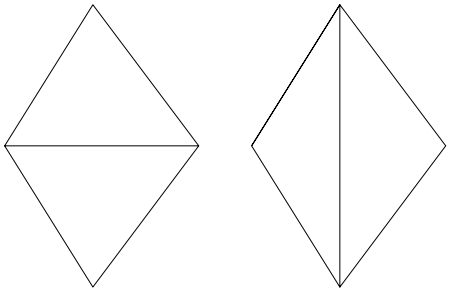

Figure 2.4: The two possible triangulations of a convex quadrilateral

The resultant triangles are said to be "locally equiangular" and if all strictly convex quadrilaterals, (see Figure 2.5), possess this property then the whole triangulation is said to be "globally equiangular".

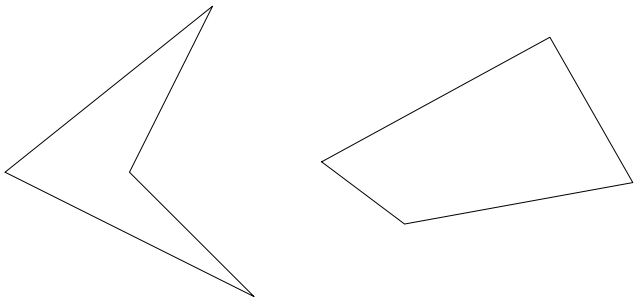

Figure 2.5: Non-convex and convex quadrilaterals

Sibson (1978) proved that a globally equiangular triangulation is unique (subject to degeneracy) and that it is in fact the Delaunay triangulation.

The Delaunay triangulation is widely used due to this "locally equiangular" property, since it is known (see e.g. Strang and Fix (1973)) that triangulations which possess this property result in well-conditioned stiffness matrices for the Finite Element Method applied to self adjoint problems.

Techniques which can be used to generate Delaunay triangulations are now reviewed. There are two main techniques which can be used to produce a Delaunay triangulation from a pre-generated set of nodes; the diagonal swapping technique and the insertion polygon technique, although there is also another

technique for producing Delaunay triangulations which requires only boundary nodes to be supplied and then proceeds to generate interior nodes and triangles in an enrichment manner.

The first technique for use on a pre-generated set of nodes is that of diagonal swapping. This is based on a theorem by Lawson (1972) which states that given any initial triangulation it can be transformed to any other triangulation by a finite number of diagonal swaps, (see Figure 2.4). These diagonal swaps are only applicable if two triangles with a common edge form a convex quadrilateral (see Figure 2.5).

The diagonal swap procedure first requires the generation of artificial nodes to aid in the procedure. These artificial nodes are generated in such a way that they are well distanced from the convex hull, Ω, of the convex region to be triangulated (see Figure 2.6).

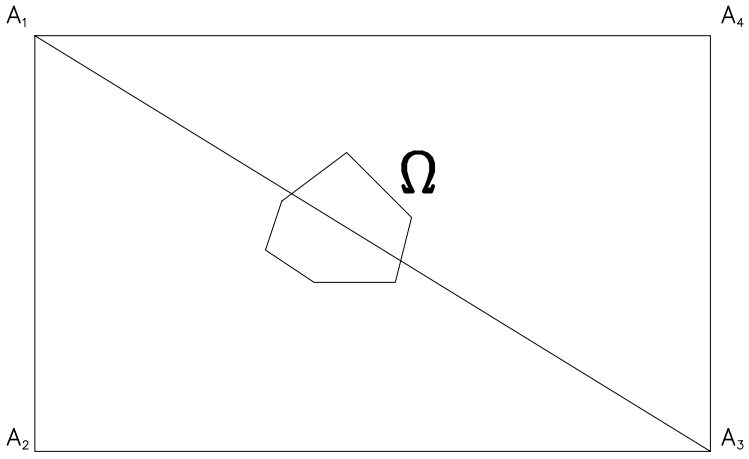

Figure 2.6: Positions of the artificial nodes

The artificial nodes are triangulated and then the Local Optimisation Procedure, as suggested by Lawson (1977), is used. The procedure is as follows:-

(a) The next node in the node list is placed in the triangulation and then connected to the vertices of the existing triangle in which it lies (see Figure 2.7).

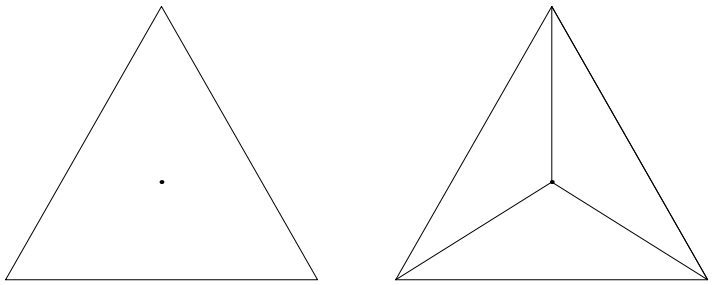

Figure 2.7: Insertion of a node into a triangulation

(b) Convex quadrilaterals, formed by adjacent triangles in the triangulation, are checked so that if swapping the interior diagonals, (see Figure 2.4), increases the minimum angle in the two triangles then they are swapped. This is repeated until no further swaps can be made in the triangulation.

(c) If there are more nodes to be inserted then return to step (a), otherwise all triangles which contain artificial nodes are removed and the remaining triangles form a Delaunay triangulation.

The other method which can be used is the "insertion polygon" method, (Mock (1985)), which is based on the original circumcircle property noted by Delaunay. This again starts with artificial nodes being generated and nodes being inserted into an existing triangulation one at a time. The procedure is as follows:-

(a) For the new node, examine the circumcircle of each existing triangle and flag those triangles for which the node is inside the circumcircle.

(b) The flagged triangles are removed from the triangulation leaving a polygon surrounding the new node. The new node is then connected with neighbouring vertices on the boundary of the polygon to form the new triangles (see Figure 2.8).

(c) The procedure returns to (a) if all nodes have not yet been inserted, otherwise all triangles which contain artificial nodes are removed and the triangles remaining form a Delaunay triangulation.

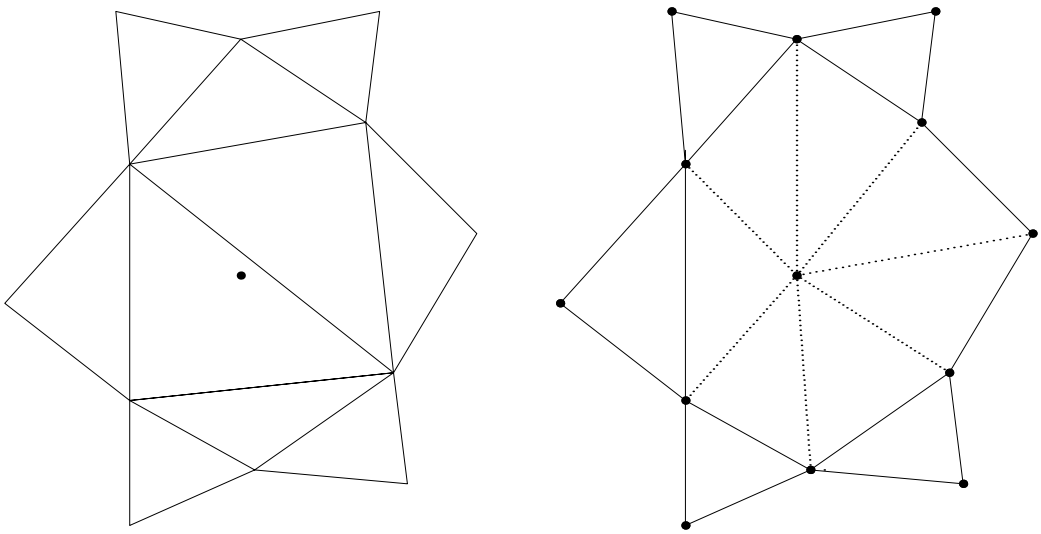

Figure 2.8: Node insertion by the insertion polygon technique

Both of the above methods can be used once a Delaunay triangulation has been produced and extra nodes are to be added. In this case artificial nodes are only required if the extra nodes lie outside the convex hull of the previous triangulation.

For non-convex regions the procedure varies slightly. The convex hull of the non-convex region is found and artificial triangles are produced so that the regions

inside the convex hull but outside the non-convex region are filled by triangles. These triangles are designated as fixed and as such cannot be modified during the node insertion procedure. At the end of the node insertion procedure these artificial triangles are then removed and the resulting triangulation is that of the non-convex region.

The other method of producing a Delaunay triangulation is the enrichment technique, Frey (1987). In this, a set of boundary nodes is generated so that the boundary of the region is well-represented. Interior nodes are then produced and triangulated until a final triangulation satisfies some user defined property, such as the number of triangles in the triangulation. This enrichment approach is usually data dependent, however Frey produced a property which results in Delaunay triangulations.

Frey's method is as follows :-

(a) Having generated a set of boundary nodes, produce a Delaunay triangulation using just these nodes.

(b) Each triangle is then checked, in order of size, largest to smallest, to see if it can be refined. The test that Frey used is that the circumcentre of the triangles circumcircle must be inside the triangle. If such a test is satisfied then the triangle doesn't need refinement. If the triangle fails this test then a new node is placed at the centroid of the triangle, three new triangles being produced (as in Figure 2.7) and the resulting triangulation is then modified using diagonal swapping (as outlined above) so that a Delaunay triangulation is achieved.

(c) Step (b) is then repeated until either:-

1. All triangles have areas less than a specified tolerance, or
2. No triangles fail the test, or
3. A preset number of triangles have been produced;

The procedure then terminates.

This ends our review of the purely Delaunay based generation techniques. Triangular grid generation techniques which produce grids with other properties are now considered.

The Local Optimisation Procedure of Lawson (1977) used in Delaunay diagonal swapping can also be used to reconnect edges according to other criteria, such as that of the Minimum Weight Triangulation(MWT), (also known as the "optimal" triangulation), Watson and Philip (1984). In this procedure the criterion used for diagonal swapping is that the shortest interior edge of a convex quadrilateral must be chosen. This means that the sum of the edge lengths in the triangulation is a minimum. Figure 2.9 shows how this criterion can produce non-Delaunay triangulations. Delaunay's method selects the dotted line, MWT the solid line.

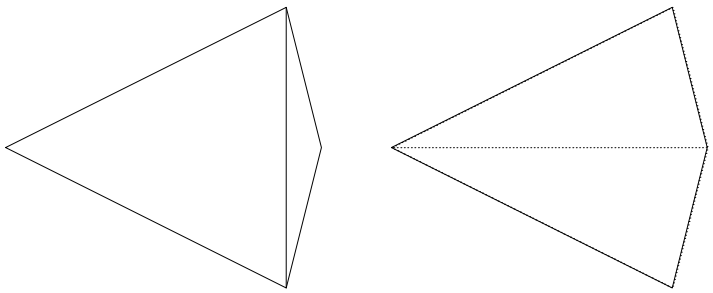

Figure 2.9: MWT and non-MWT triangulations

MWT has the property that near neighbours are connected, however as can

be seen from Figure 2.9, triangles with small angles can be produced.

An alternative set of techniques are those generally known as triangulation front techniques, which can be used either on a pre-generated set of nodes to generate triangles or to generate the nodes concurrently with the triangles.

In the first case the problem is to generate the nodes in a "reasonable" manner. The two main approaches are as follows :-

1) A set of boundary nodes is produced and then, using information given by the user, a random set of interior nodes is generated. The nodes are generated in such a way that the minimum spacing between them can be pre-defined by the user, Cavendish (1974).

2) A regular set of boundary nodes is generated, and then interior nodes are generated by spacing them equally between the pairs of boundary nodes, (Lo (1985)).

Once a complete set of nodes has been generated the procedure to generate the triangles commences. The aim is to produce triangles which are as close to being equilateral as possible. This can be done by examining a measure of the triangle's equiangularity. Many such measures can be constructed, but only two of them are presented here.

The first measure is that defined by Cavendish (1974):-

For a general triangle $\triangle$ABC, let δ be the length of the longest side of $\triangle$ABC, β be the distance between that side and the opposite vertex (see Figure 2.10), and $\gamma_{ABC} = \frac{\delta}{\beta}$.

If $\triangle$ABC is equilateral then $\gamma_{ABC} = \frac{\sqrt{3}}{2}$, so a measure of the deviation, α, from being equilateral, of $\triangle$ABC, is $\alpha = \left|\gamma_{ABC} - \frac{\sqrt{3}}{2}\right|$. Thus the larger the value

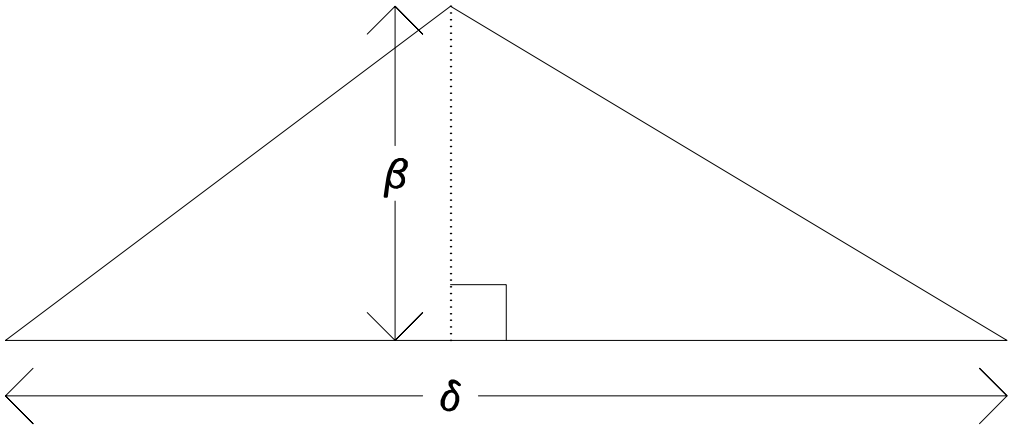

Figure 2.10: Cavendish's measures

of α, the further from being equilateral the triangle is. The decision as to which triangle to form from a base line is based on choosing α as small as possible, while making sure that the triangles which are generated by the new edges do not produce triangles with large values of α.

The second measure is that proposed by Lo (1985) :-

Given all possible triangles that can be chosen, we select that which has the minimum measure α , where for $\triangle$ABC,

$$\alpha = \frac{4\sqrt{3}.\text{ area of}\triangle ABC}{AB^2 + AC^2 + BC^2}. \tag{2.7}$$

The factor of $4\sqrt{3}$ is such that for an equilateral triangle, α =1. Lo also suggested methods for dealing with odd geometrically shaped regions, such as a sector of a circle.

Approaches for generating nodes and triangles concurrently in a triangulation front framework are now presented. These approaches vary in that the number of fronts from which new triangles and nodes can be created is different, see Sadek (1980) and Peraire, Vahdati, Morgan and Zienkiewicz (1987). The two approaches could almost be called "single" and "multiple", and for an explanation of this it

is necessary to look at the two approaches and see how they vary.

The "single" approach is as follows :-

(a) A set of boundary nodes is generated and the front set up.

(b) A new triangle is generated from a baseline in the front. If a new node is used the position of the new node is stored. Once this has been carried out for all base lines on the original front, there is a layer of triangles one triangle thick round the boundary, (see Figure 2.11) and then a new front is set up and step (b) is repeated until the whole region is covered in triangles.

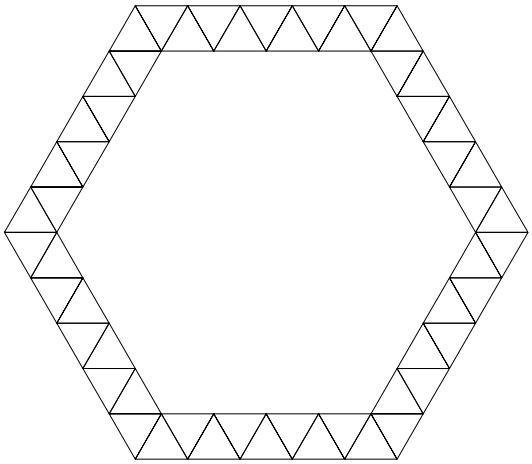

Figure 2.11: Grid after one layer

The positioning of the new nodes is determined by the magnitude of the angle between any two adjacent base lines and the number of triangles that will be formed using the given node on the current level. The general criterion used is to obtain almost equiangular triangles, Sadek (1980). This can be called "single" since only one triangulation front is allowed at one time.

The "multiple" procedure is such that rather than confining triangles to be produced in layers, the front elements can be looked at in any order and if necessary, more than one triangulation front can exist at one time, see Peraire, Vahdati,

Morgan and Zienkiewicz (1987). This splitting of fronts can be caused when a new node should be created near an existing node in the front and rather than having two nodes in very close proximity the triangle is deformed slightly and two fronts are formed (see Figure 2.12).

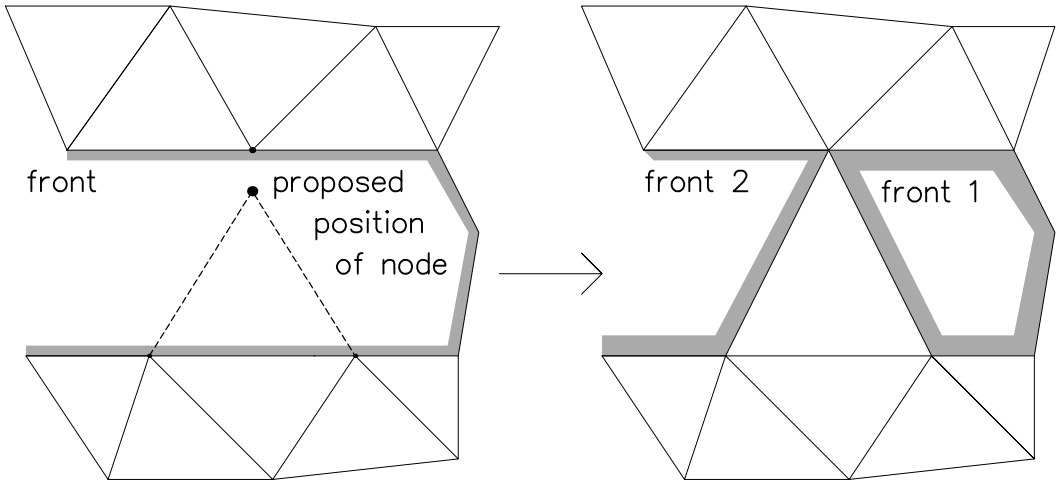

Figure 2.12: Front splitting

Finally a technique which was originally used as a post processor for triangulation front grids but can also be used on any triangulation is presented. This technique is called "smoothing" and is used on completed triangular grids to produce triangles with "better" properties, e.g. equiangularity. This technique was originally used when poor triangles were produced by grid generation and needed modification. The original technique is Laplacian smoothing, Cavendish (1974), whereby each data point is moved to the centroid of the polygon formed by all the triangles surrounding it (see Figure 2.13).

Clearly, this idea can also be treated in a different framework, in which each edge of the triangulation is viewed as a spring with negligible original length and

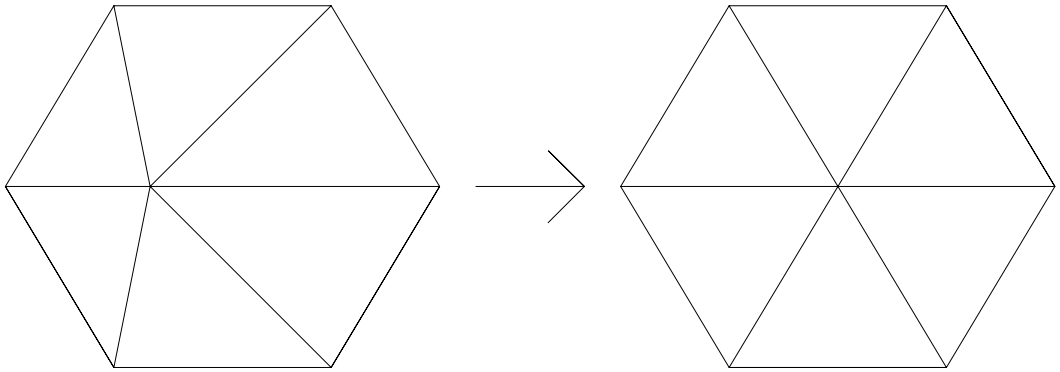

Figure 2.13: Centre node moved by Laplacian smoothing

a spring constant of unity. The complete framework of springs is then iteratively solved to an equilibrium position for all the interior nodes, the boundary nodes being treated as fixed. This approach is pursued in Chapter 6. This idea can also be extended to deal with the placement of boundary nodes, see Thacker, Gonzalez and Putland (1980).

The main methods of producing geometrically based grids have now been outlined and in the next chapter a review is made of how these techniques are modified to produce data dependent grids. Also included in the review are other techniques which have been used to produce data dependent grids.

Chapter 3

Data Dependent

Grid Generation Techniques

In this chapter techniques of grid generation which have evolved to fulfil the need for good representations of functions on computational grids are surveyed. The functions to be represented may be presented as data sets, where if it is wished to obtain values other than at the data points this is done via interpolation. Alternatively one may wish to generate a computational grid which well represents initial data restricted onto it in preparation for use of an adaptive numerical scheme to solve differential equations upon that grid. Such adaptive differential equation solvers permit the computational grid to evolve in tandem with the solution and may themselves be suitable for the initial grid generation, perhaps using an artificial time technique. Alternatively separate grid generation methods may be used, and it is such techniques which are now reviewed, many of which work by improving an existing representation by adjustment of the underlying grid.

There are three different classes of data dependent grid techniques where the shape and position of the grid elements depends not only on the geometry of the region but also on the value of the underlying data at each grid point. The classes are :-

1. Data dependent grid generation where the grid produced depends solely on the data and the data points. This can be split into two categories :- modifying a geometrically generated grid or producing a grid from scratch.

2. Adaptive grid techniques :- a P.D.E is solved and then the grid on which it was solved is adapted so that the grid follows features of the solution.

3. Adaptive P.D.E solvers whereby the P.D.E is solved on a grid and in so doing a new grid is produced on which the new solution is presented, e.g. Moving Finite Elements, Miller and Miller (1981). The grids produced by the techniques in classes 1 and 2 can form initial grids for this class of techniques.

It is primarily the first class which is of interest, however many techniques of this class belong also to class 2.

In this chapter the representation of a function $f(x)$ on a 1-dimensional grid is initially considered and then grid generation in 2-dimensions, using both quadrilaterals and triangles is reviewed.

3.1 1-dimensional Grid Generation

One of the main tools used in 1-D data dependent grid generation is that of equidistribution, De Boor (1973). In this technique, some measure which monitors the representation of the function, $f(x)$, is calculated and is then used to position the nodes. This measure, $w(x)$, which in general is a function of $f(x)$ and/or its derivatives, is also known as a weighting or grading function. The basis of equidistribution is to equally distribute the measure in the intervals between adjacent nodes, i.e.

$$\int_{x_i}^{x_{i+1}} w(x)dx = \text{ constant } = \frac{1}{N}\int_a^b w(x)dx \tag{3.1}$$

where $a = x_0 < x_1 < x_2 < \ldots < x_{N-1} < x_N = b$ are the nodal positions.

Examples of the measures $w(x)$ which can be employed are arc-length, solution gradient or even local truncation error of a solution if used as an adaptive technique.

Equidistribution is used by Pereyra and Sewell (1975) to produce meshes for boundary value problems in O.D.E's, using a measure based on local truncation error of the solution. They discuss two distinct procedures which use equidistribution. The first is the minimisation of the error using a fixed number of nodes, the second is a strategy for inserting extra nodes into an existing grid in order to produce a truncation error less than a specified tolerance in every interval $[x_i, x_{i+1}]$. White (1979) considers a specific solution measure to be equidistributed, namely arc-length, while Ablow and Schechter (1978) introduce what they call a campylotropic coordinate which is based effectively on equidistributing arc-length and a measure of the gradient of the solution curve. They show that using such a

coordinate, a solution can be well approximated by using only a tenth the number of points used previously. Such methods can produce good grids with respect to the solution, e.g. improve approximationion error, but the resulting grids will sometimes possess wildly fluctuating step sizes which can cause inaccuracies in the finite difference approximations. To deal with such a problem Kautsky and Nichols (1980) produced a scheme which not only attempted to equidistribute a weight function but also constrained the ratio of adjacent step sizes.

Carey and Dinh (1985) extended equidistribution work to a more general form by introducing a formula for an "optimal" grading (weight) function for interpolating a function with the error calculated in some specified semi-norm. By the use of Fourier series to obtain an error estimate and a variational approach to optimise the semi-norm, they show that for a piecewise polynomial interpolant of degree k in the H^m-semi-norm, the optimal weight function is :-

$$w = [u^{(k+1)}]^{\frac{2}{2(k-m)+3}}, \tag{3.2}$$

e.g. for linear interpolation, k=1, in the L_2-norm, m=0, this gives

$$w(x) = (u_{xx})^{\frac{2}{5}}. \tag{3.3}$$

This result, however, is only an asymptotic result and as such is most accurate for large numbers of nodes, N.

Equidistribution of mesh points can also be obtained in 1-D by solving a second-order differential equation, (see Anderson (1985)), which may be derived as follows. A positive weight function, $w(x)$, is to be equally distributed over all grid intervals. i.e.

$$\int_{x_i}^{x_{i+1}} w(x)dx = \text{constant} \tag{3.4}$$

or

$$\Delta x_i w_i = \text{ constant}. \tag{3.5}$$

If this process is considered as a transformation from a uniform grid in computational ξ-space to the equidistributed grid in x-space, we have $x(\xi)$, with integer values of ξ defining the nodes. (3.5) can be rewritten as

$$x_\xi w = \text{ constant} \tag{3.6}$$

where the derivative, by the Mean Value Theorem, is evaluated at some interior point of the interval.

The grid is usually obtained by solving a second-order differential equation, which is obtained by differentiating (3.6) with respect to ξ.

$$x_{\xi\xi} w + x_\xi w_\xi = 0. \tag{3.7}$$

This equation is rewritten giving the equation which is solved, using finite differences and an iterative solver, to obtain the mesh point positions in physical space as

$$x_{\xi\xi} + x_\xi \frac{w_\xi}{w} = 0. \tag{3.8}$$

It is also possible to solve (3.6) using simple numerical quadrature to directly find the nodal positions. Let the node number $\xi = \xi_{\max}$ = N-1, where N is the number of nodes, when $x = x_{\max}$ and let $\xi = 0$ at $x = 0$, then

$$\frac{x}{x_{\max}} = \frac{\int_0^\xi \frac{1}{w} d\xi}{\int_0^{\xi_{\max}} \frac{1}{w} d\xi}. \tag{3.9}$$

Either of the above procedures leads to a set of equidistributed nodes with respect to the weight function w.

Other work has considered the least squares approximation of functions in 1-dimension. In this case, rather than use the actual interpolant, a best fit to the curve in the least squares sense is calculated (see Figure 3.1).

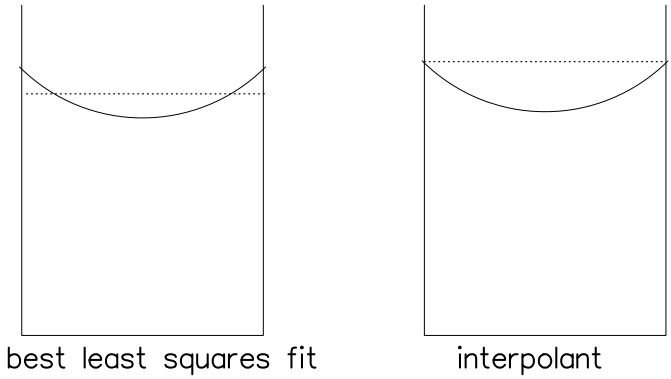

Figure 3.1: Alternative representation strategies : Thick line is function, dotted line is approximation

Chui, Smith and Ward (1977) showed the existence of such approximations for a continuous function and that such a best least squares approximation among discontinuous linears with adjustable nodes is continuous, and Loach and Wathen (1991) introduce algorithms, also with variable nodes, for finding the best least squares approximation, using continuous polynomials, by dynamic programming. Baines and Carlson (1990), Baines (1991) formulated the problem in a variational framework using discontinuous linears with variable nodes and produced an algorithm for finding the best continuous linear least squares approximation almost everywhere.

Having looked at nodal positioning in 1-D, data dependent grid generation in 2-D is considered, first using quadrilaterals, and then in Section 3.3 using triangles.

3.2 Data Dependent Quadrilateral Grid Generation

In this section techniques of generating quadrilateral data dependent grids are reviewed. Many of the techniques presented in this section originated as adaptive P.D.E solvers and are basically post-processors (grid modification procedures), although there are some pure grid generators for solving problems such as function approximation and some of the techniques are extensions of the geometric techniques outlined in Section 2.1.

Coming in the last category is the extension of elliptic grid generation techniques, described in Section 2.1, to a data dependent framework. Anderson (1985) showed that by choosing specific functions P and Q in (2.3) and (2.4), namely

$$P = \frac{\alpha}{J^2}\frac{1}{w_1}\frac{\partial w_1}{\partial \xi} \tag{3.10}$$

$$Q = \frac{\gamma}{J^2}\frac{1}{w_2}\frac{\partial w_2}{\partial \eta} \tag{3.11}$$

where α and γ are constants, $J = x_\xi y_\eta - x_\eta y_\xi$ and w_1 and w_2 are weight functions, then the resulting elliptic grid generator is equivalent to equidistributing a chosen weight function along the curvilinear coordinate lines (see (3.8)). Anderson also shows that a reasonable choice of weight function is one that is gradient based.

In the category of grid modification strategies are nodal movement and grid refinement. Grid refinement is the splitting of a quadrilateral into 4 equally sized, similarly shaped, quadrilaterals. The region in question is usually refined by considering some measure on the original region to check whether it exceeds a specified tolerance. If this is the case, then the region is refined (see Verwer and Trompert (1991)). This can be beneficial for grid generation as efficient data

structures enable the rapid transfer of data from level to level of the grid. However, if a differential equation is to be solved on the mesh then this refinement to non-conforming meshes, where small elements meet the border of large elements (see Figure 3.2), can cause difficulties in the solution procedure, one such problem being that the interpolation to higher levels of grid points can introduce computational inaccuracies.

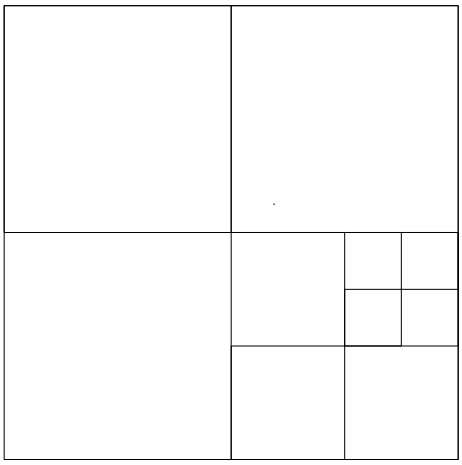

Figure 3.2: Quadrilateral refinement

Nodal movement is another grid modification strategy and it has been employed by Gnoffo (1982) and Catherall (1988), in a manner such that the edges in a grid are treated as springs. The springs are assumed to have negligible original length and the associated spring constant is usually based on some measure of the solution along the edge of the grid element. Catherall (1988) uses the solution gradient along a grid line as a measure of each spring constant. This approach can also be extended to triangular grids, as has been explained in Chapter 2, where smoothing, as employed by Cavendish (1974), is equivalent to a spring constant of unity.

Moving away from post-processors of adaptive P.D.E solutions, the category

of data dependent grid generation is reviewed. One application is the function approximation problem where a function is to be well represented. Farmer, Heath and Moody (1990) look at this problem and present a method which gives good results.

For the function approximation problem, where ideally the function value does not vary too widely over each individual grid cell, a measure of deviation from the average value on each call is required. If $w(x, y)$ is the function to be interpolated then the measure of deviation $v_{i,j}$ on the (i, j)th quadrilateral is

$$v_{i,j} = \int_{cell} [w(x, y) - w_{av}]^2 dxdy \tag{3.12}$$

where w_{av} is defined as

$$w_{av} = \frac{\int_{cell} w(x, y) dxdy}{\int_{cell} dxdy} \tag{3.13}$$

and the functional to be minimised is

$$V = f + \gamma \sum_{i,j} v_{i,j} \tag{3.14}$$

where f is a measure of smoothness and orthogonality.

Once the functional has been calculated for an original grid then the mesh points can be moved. This is done in a local manner, i.e. the point is moved inside some part, Q_b ,of the surrounding patch of quadrilaterals (see Figure 3.3). This is achieved by transforming Q_b onto the unit square and then choosing a point inside the unit square using random numbers. This point is then transformed back onto Q_b and this is taken as the new nodal position.

The new functional is calculated for the new nodal position and if the value of the functional decreases, then the new point is accepted and the procedure is repeated with another mesh point. This continues until no nodal movements are

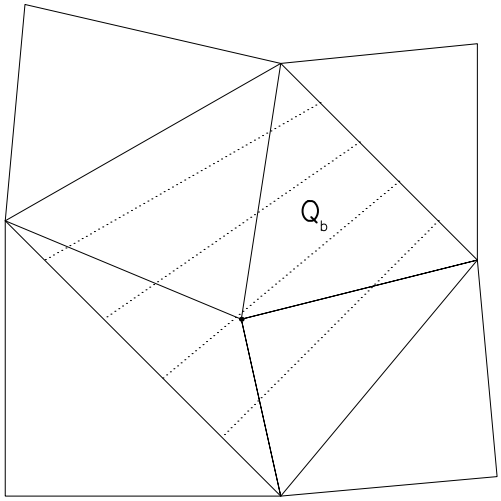

Figure 3.3: Region Q_b formed from a patch of quadrilaterals

accepted, i.e. the functional does not decrease, for a number of consecutive sweeps through all the nodes. This numerical minimisation technique can be improved to find global minima rather than local minima by the use of "simulated annealing", Kirkpatrick, Gellatt and Vecchi (1983), whereby nodal positions which increase the value of the functional are accepted at first but become less likely to be chosen as the procedure continues.

Having reviewed data dependent quadrilateral grid generation, data dependent triangular grid generation is now surveyed.

3.3 Data Dependent Triangular Grid Generation

Many of the techniques used for producing data dependent triangular grids are either modifications of those techniques which produce data dependent quadrilateral grids or extensions of those which produce geometrically based triangular grids. Again, many techniques are based on the adaption of existing grids, al-

though some can be used to generate a data dependent grid from scratch.

As noted in Section 3.2, it is possible to treat the edges in a triangulation as springs with spring constants dependent on some function of the data and then, having produced a system of equations for the positions of the nodes, it is possible to solve them iteratively or in a pointwise fashion, Catherall (1988), Gnoffo (1982).

It is also possible to use enrichment, see Section 2.2, on an existing triangular grid, although this can introduce problems if certain methods of refinement are used. The usual refinement technique on triangles is to introduce a new node at the centroid of the existing triangle, Frey (1987), (see Figure 3.4) but this can then cause the introduction of triangles with small angles even if the original triangle is equilateral, e.g. after 2 refinements of a $(60°, 60°, 60°)$ triangle you get three $(150°, 15°, 15°)$ triangles and six $(105°, 60°, 15°)$ triangles.

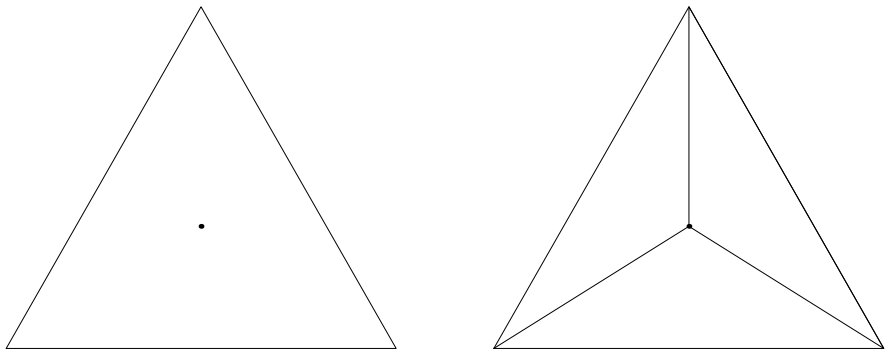

Figure 3.4: Refinement of triangle at the centroid

Another refinement technique is to place a point at the mid-point of each side of the triangle, although this requires local operations to connect the new nodes to neighbouring vertices in existing triangles to produce a valid triangulation, Lohner, Morgan and Zienkiewicz (1986), (see Figure 3.5).

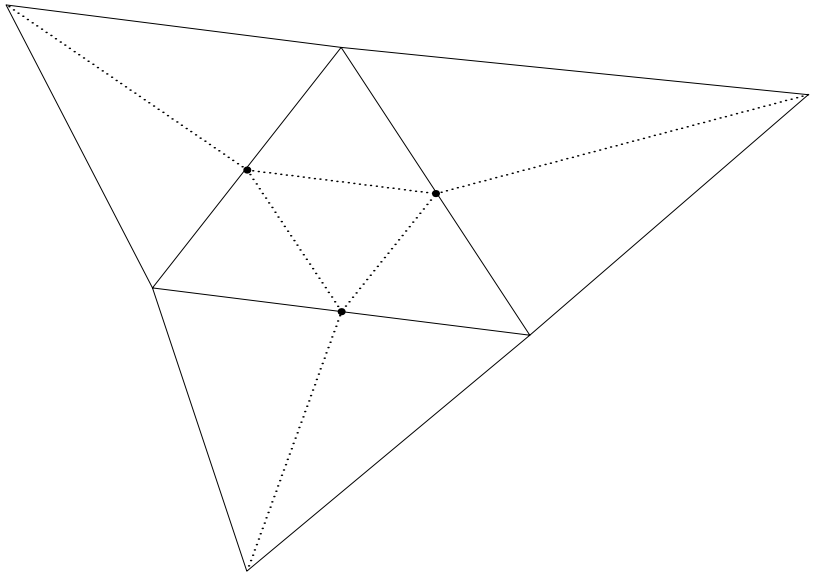

Figure 3.5: Refinement of triangle at edge midpoints

A further method based on refinement is that of Thacker, Gonzalez and Putland (1980), although in this case rather than introducing new nodes into a triangulation, nodes are removed from a fine equilateral mesh as follows :-

1. An extremely fine equilateral mesh is placed over the region.

2. For each node, n_i, the surrounding patch, P, of triangles, T_j, see Figure 3.6, is checked to see if some measure is altered by more than a specified small tolerance if the central node is removed and the patch re-triangulated in a specified manner. If it is not then the central node is removed.

3. If a node is removed then the grid is reformed on the patch in a specified consistent manner, see Figure 3.7, and the next node is checked. Every time a specified number of nodes have been removed then the grid is smoothed using Laplacian smoothing, see Section 2.2, to make the triangles more equiangular. This continues until the measure lies

within specified close limits on each patch of triangles.

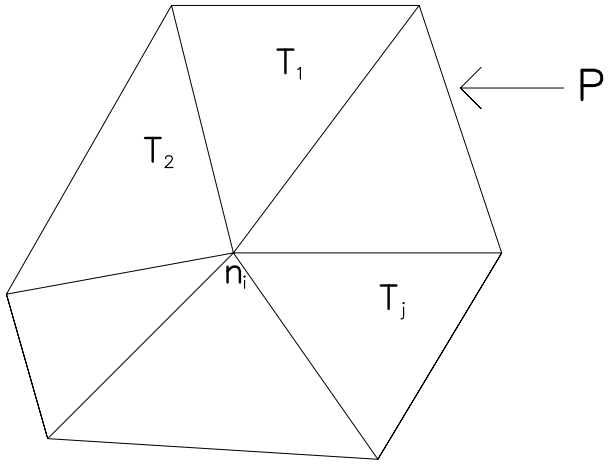

Figure 3.6: Patch of triangles surrounding n_i

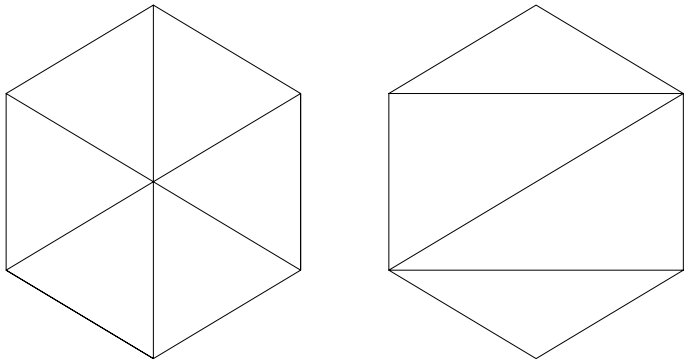

Figure 3.7: Grid reformulation after node deletion

Another method of producing data dependent triangulations is to extend the idea of diagonal swapping, see Section 2.2, to a data dependent framework. In this a cost is assigned either to each triangle or to edges between adjoining triangles. If, by swapping the interior diagonal of convex quadrilaterals, some function of the cost or costs is decreased then the diagonal is swapped. This idea of a triangulation with an associated cost was suggested by Lawson (1972) and employed later by Dyn, Rippa and Levin (1990) to try and improve the results of interpolating an underlying function on a triangular grid using linear interpolants.

Dyn *et al.* present several different methods by which the cost associated with an edge, or a triangle, in a triangulation can be defined. The methods employ the underlying data at the points in two manners. For an edge, the cost associated with the edge is usually defined in terms of the coefficients of the linear interpolants on the triangles with the common edge. The cost on a triangle is usually defined in terms of the data values at the vertices and the vertex positions. Further detail is given in Chapter 5 where their work is considered and extended.

A final procedure is one which does not require an initial, geometrically based grid to be generated before it is reformed in a data dependent manner. Instead a data dependent triangulation is generated directly. Mavriplis (1990) uses a "stretching" factor, which is based on a vector solution measure, to map nodes into a transformed space from physical space. The vector has magnitude, s, and direction θ, with $s > 1$ and $|\theta| < \frac{\pi}{2}$. Mavriplis attempts to position the nodes so that they are approximately equidistant from each other in all space directions on a control surface in transformed space. In physical space, distance, d, is given by :-

$$d^2 = \Delta x^2 + \Delta y^2 \tag{3.15}$$

while in transformed space, distance, δ, is given by :-

$$\delta^2 = \Delta x^2 + \Delta y^2 + \Delta z^2 \tag{3.16}$$

where Δz is defined as

$$\Delta z = (\Delta x \sin\theta - \Delta y \cos\theta)(s - 1) \tag{3.17}$$

These expressions for the transformed directional increments can be used to position the nodes, triangulate them in a Delaunay manner (see Section 2.2) in

transformed space and then transform the triangulation back to physical space.

D'Azevado and Simpson (1989) show that for a convex quadratic function of two variables

$$f(x,y) = \lambda_1 x^2 + \lambda_2 y^2 + cx + dy + e \qquad \lambda_1, \lambda_2 > 0 \tag{3.18}$$

the maximum interpolation error in the l_∞ norm can be minimised by using a transformation

$$x = \sqrt{\frac{\lambda_1}{\lambda_2}}\xi, \qquad\qquad y = \eta \tag{3.19}$$

and then performing a Delaunay triangulation of the points in the transformed (ξ, η)-space and transforming the triangulation back into (x, y)-space. This result can be used on any general strictly convex, or concave, function, since calculating the Hessian of the function at a point gives a local transformation which can be used to position nodes. Rippa (1991) proves that the Delaunay triangulation used on the transformed variables is in fact the optimal triangulation for this problem.

This concludes the overview of grid generation techniques and in the following chapters the application of these techniques, and extensions of them, to the problem under consideration is explored.

Chapter 4

Test Problems, Data Sets and Result Monitors

The main objective of this research is to produce data dependent triangular grids which, when underlying functions are interpolated on them, give better representation of the data than the aforementioned geometric grid generators. To this end it is necessary to be able to test the grid generation methods on a range of test functions and, where grid points must be specified, with different distributions of data points. The test problems considered are chosen so that the performance of the grid generation methods can be evaluated both on underlying functions which are poorly represented by geometric methods and a smooth function which is well represented by any reasonable geometric grid.

Once the data dependent triangulations have been produced it is then necessary to be able to assess the grids produced. These "grid monitors" not only measure the associated interpolation errors but also the geometric properties of the triangulations, thus enabling comparisons with geometrically generated tri-

angulations. These comparisons, for example of angles and aspect ratios, allow checks on statements such as "long thin triangles are bad for interpolation". The monitors, together with the test problems, are introduced in this chapter.

In Section 4.1 the test functions are outlined, described and depicted, while in Section 4.2 the data sets are presented and their origins detailed. Finally, in Section 4.3, the performance monitors are presented and their significance described.

4.1 The Test Problems

The test problems are comprised of sets of nodes and functions. The test functions presented here are taken from various sources. Some are taken from Dyn, Rippa and Levin (1990), who present a number of functions to test the methods of grid generation they employ, some are from D'Azevado and Simpson (1989) while others are those encountered or devised during the course of this research. All the test functions were used in the testing of the techniques produced in later chapters, although not all the functions are shown in the graphical results. The functions may be grouped into classes exhibiting similar features.

The first group of functions includes those functions which change value rapidly in the vicinity of some line but do not have discontinuous derivatives. They can be thought of as smooth ramps.

$$
\begin{aligned}
SR1 &= \frac{(\tanh(9y - 9x) + 1)}{9} \\
SR2 &= \frac{(\tanh(9y + 9x - 9) + 1)}{2} \\
SR3 &= 1 + \tanh(-3g(x, y)) \quad \text{where} \quad g(x, y) = (.595576(y + 3)^2 - x - 1)
\end{aligned}
$$

All these functions are tanh functions. SR1, which has contours parallel to the line $y = x$, see Figure 4.1, was denoted as F2 by Dyn *et al.*, while SR2 has contours at right angles to those of SR1. SR3 has contours which are curves $g(x, y) =$ constant, as shown in Figure 4.2, and was labelled F8 by Dyn *et al.*

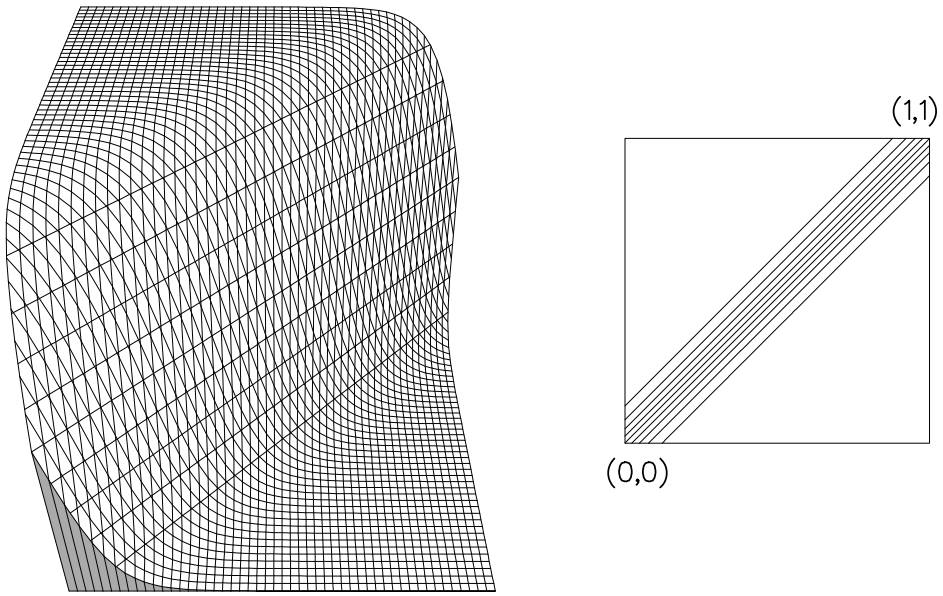

Figure 4.1: Function SR1

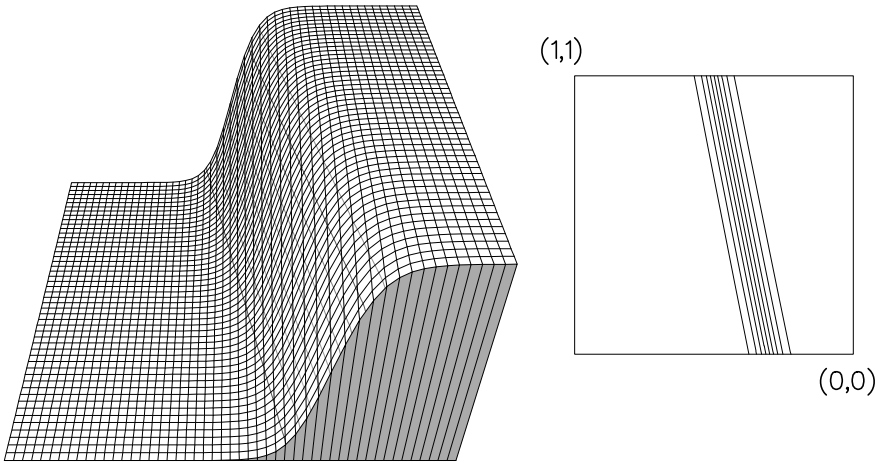

Figure 4.2: Function SR3

Another group of test data functions have "hill" like features whereby the data values rise in a continuous fashion from the edge of the unit square to the centre of the unit square where they attain their maximum value.

$$
\begin{aligned}
SH1 &= \frac{e^{-\frac{81}{16}(x-\frac{1}{2})^2+(y-\frac{1}{2})^2}}{3} \\
SH2 &= \frac{e^{-\frac{81}{4}(x-\frac{1}{2})^2+(y-\frac{1}{2})^2}}{3} \\
SH3 &= \frac{1}{4}\cos^2(\pi((x-\frac{1}{2})^2+(y-\frac{1}{2})^2)) \\
SH4 &= \frac{\sqrt{64-81(x-\frac{1}{2})^2+(y-\frac{1}{2})^2}}{9} - \frac{1}{2}
\end{aligned}
$$

Functions SH1 and SH2 are gaussian hills of different degrees of slope. Function SH3 is a smooth hill centred at $(\frac{1}{2}, \frac{1}{2})$, while function SH4 is part of a sphere above the unit square. Functions SH1, SH2 and SH4 were called F4, F5 and F6 respectively by Dyn *et al*, and all four are depicted in Figures 4.3 to 4.6.

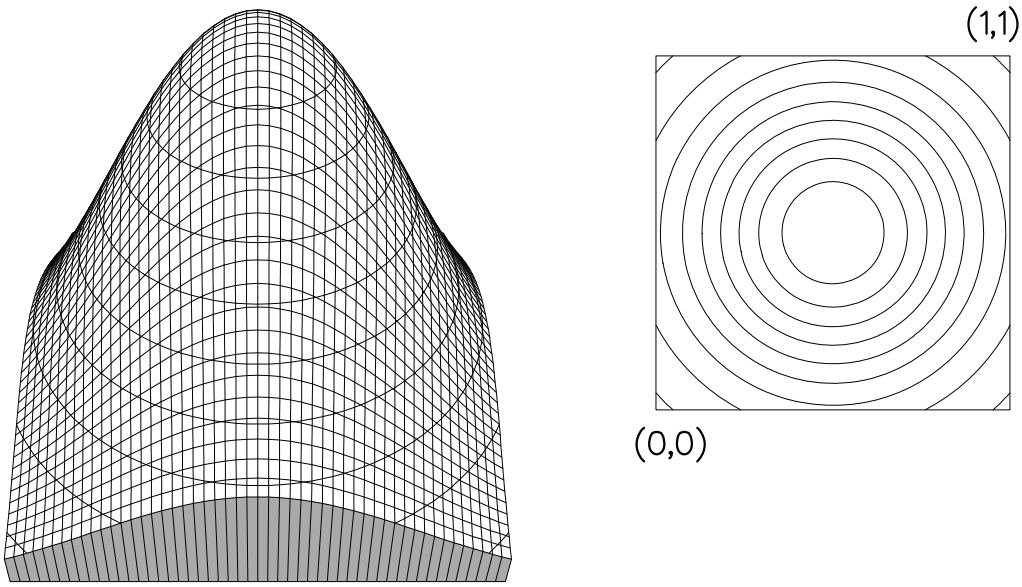

Figure 4.3: Function SH1

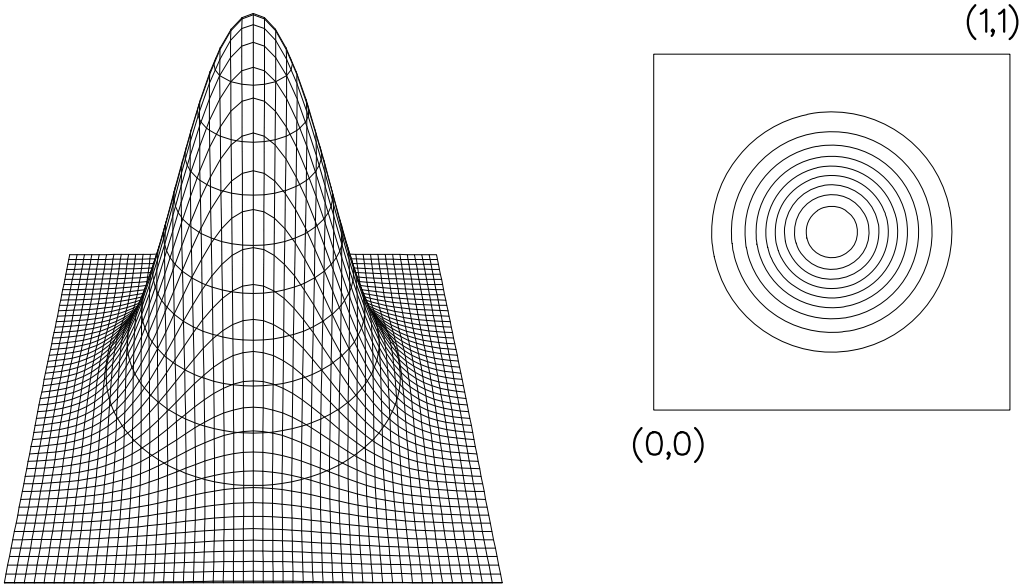

Figure 4.4: Function SH2

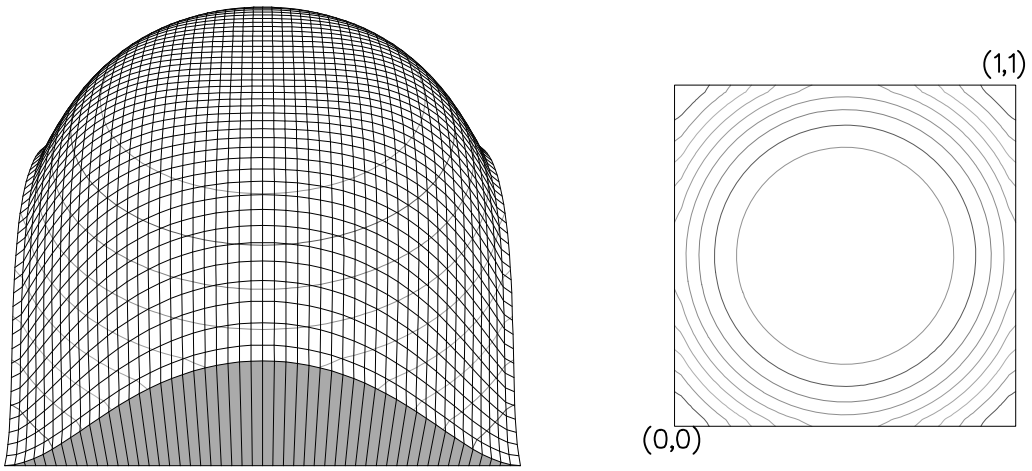

Figure 4.5: Function SH3

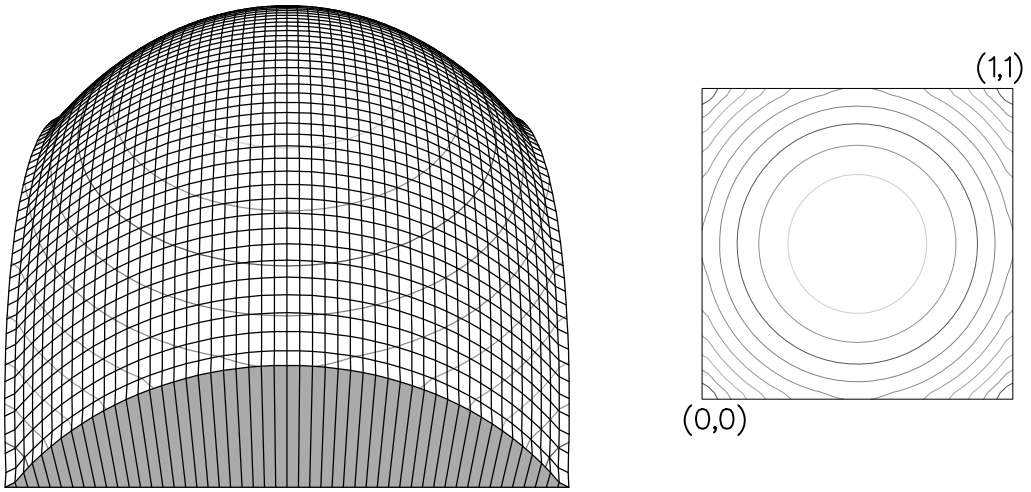

Figure 4.6: Function SH4

The next group of functions consists of those which have three (or more) local extrema on the unit square.

$$\begin{aligned} M1 &= \frac{3}{4}e^{-\frac{1}{4}((9x-2)^2+(9y-2)^2)} + \frac{3}{4}e^{-\frac{1}{49}(9x+1)^2-\frac{1}{10}(9y+1)} \\ &\quad + \frac{1}{2}e^{-\frac{1}{4}((9x-7)^2+(9y-3)^2)} - \frac{1}{5}e^{-((9x-4)^2+(9y-7)^2)} \\ M2 &= \frac{\frac{5}{4}+\cos 5.4y}{6(1+(3x-1)^2)} \end{aligned}$$

Function M1 is 2 "mountains" with a small "hollow", while function M2 is a "saddle", with 2 "mountains" overlooking a "valley". Functions M1 and M2 were called F1 and F3 by Dyn *et al*, and are depicted in Figures 4.7 and 4.8.

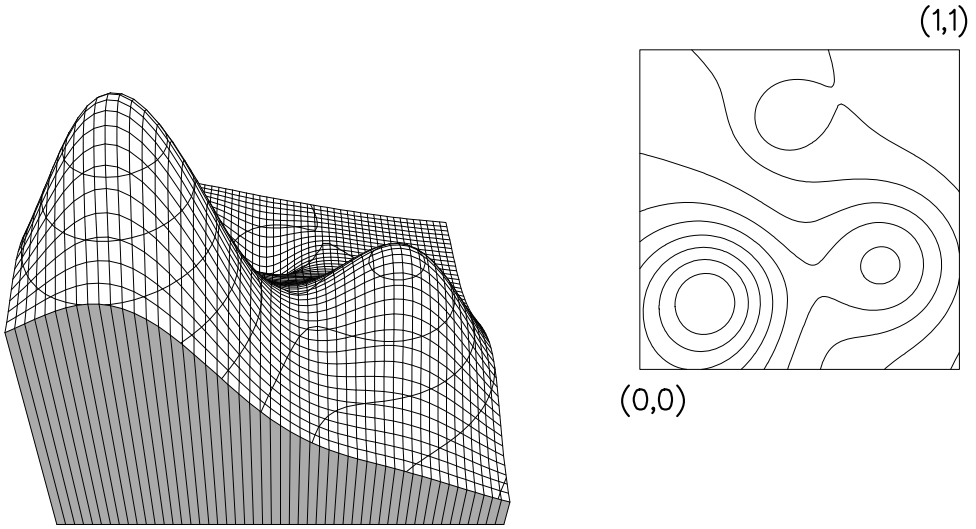

Figure 4.7: Function M1

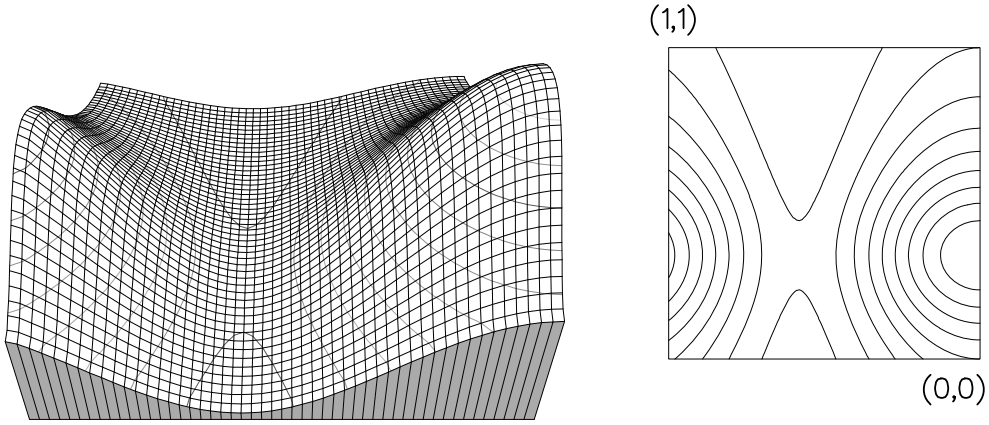

Figure 4.8: Function M2

The next group of functions are those which are purely polynomials in x and y.

$$
\begin{aligned}
P1 &= x^2 + y^2 \\
P2 &= x^2 + 100y^2 \\
P3 &= (1-x)^4 + 5(1-y)^4 \\
P4 &= x^6(1-\frac{y}{2})^6 + y^6(1-\frac{x}{2})^6 \\
&\quad +(1-\frac{x}{2})^6(1-\frac{y}{2})^6 + 1000x^3y^3(1-x)^3(1-y)^3
\end{aligned}
$$

Functions P1 and P2 are 2nd degree polynomials as used by D'Azevado and Simpson (1990), while P3 is a convex polynomial function. Function P4 is a polynomial of degree 12 with 3 raised corners and a small rise in the centre of the unit square, and was called F9 by Dyn *et al.* All four functions are shown in Figures 4.9 to 4.12.

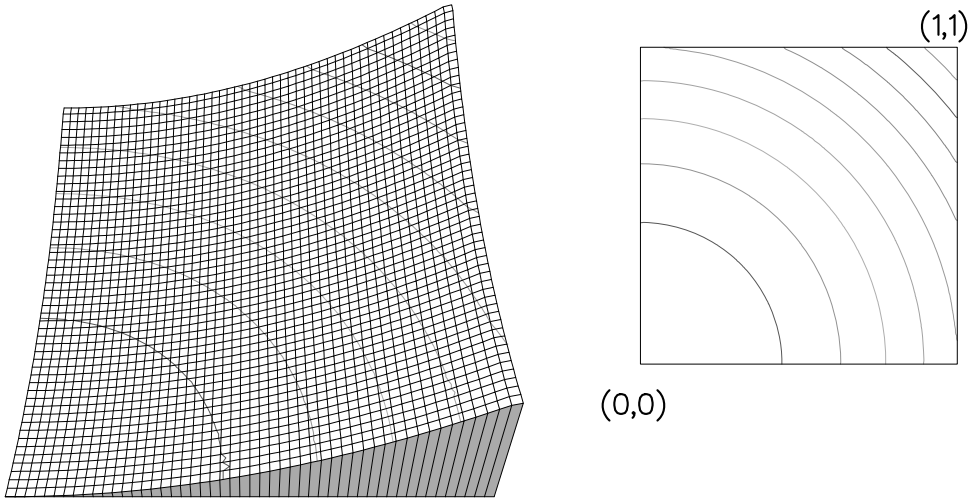

Figure 4.9: Function P1

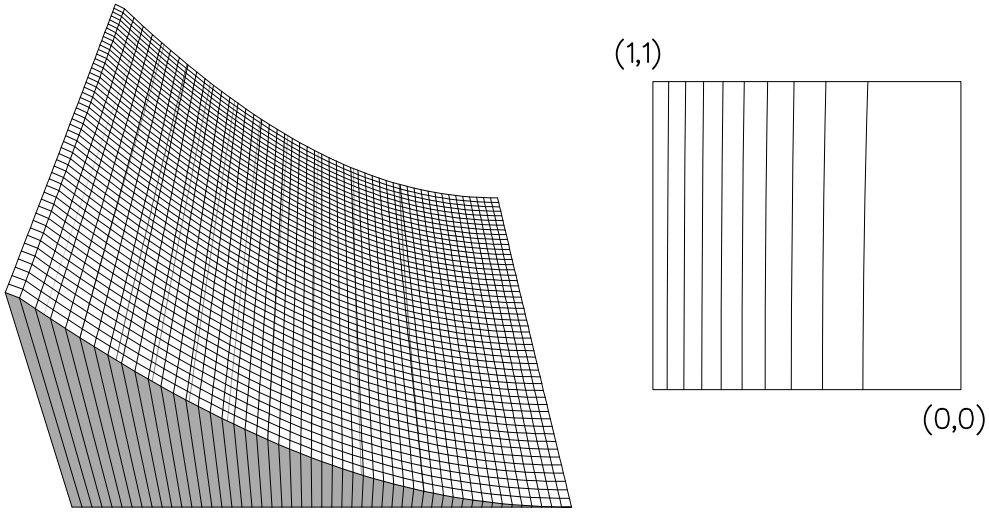

Figure 4.10: Function P2

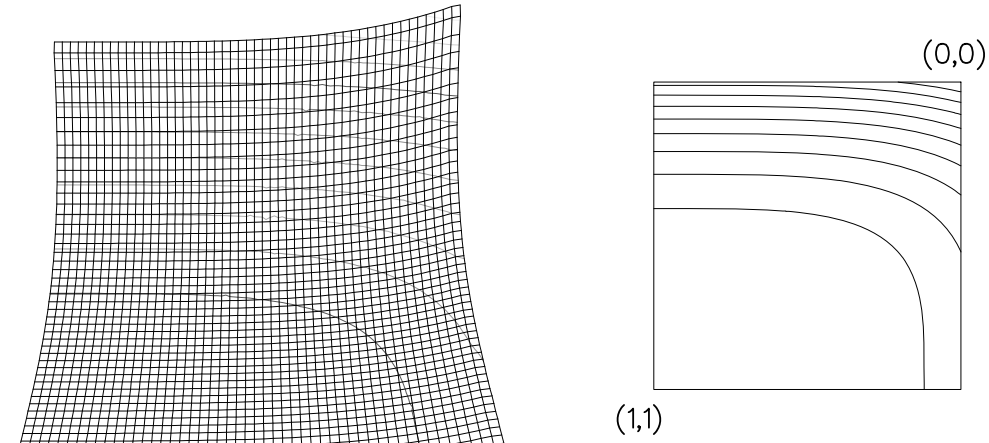

Figure 4.11: Function P3

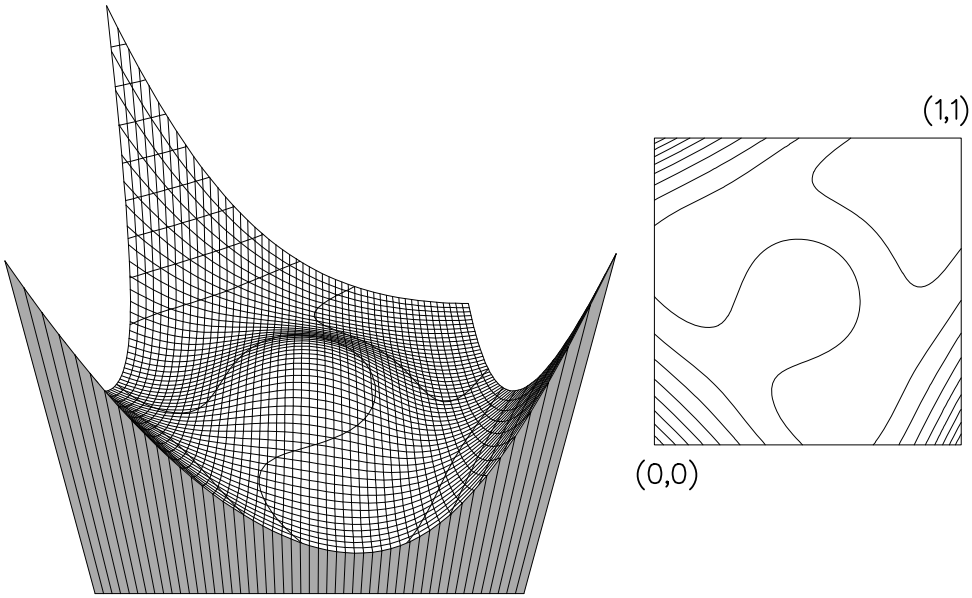

Figure 4.12: Function P4

The final group of functions consists of those with discontinuous derivatives.

These are:-

$$DD1 = \begin{cases} 1 & y - \psi \geq \frac{1}{2} \\ 2(y - \psi) & 0 \leq (y - \psi) < \frac{1}{2} \\ \frac{1+\cos(4\pi r)}{2} & r \leq \frac{1}{4} \\ 0 & \text{otherwise} \end{cases}$$

where
$$\begin{aligned} \psi &= 2.1x - 0.1 \\ r &= \sqrt{((\psi - \frac{3}{2})^2 + (y - \frac{1}{2})^2)}. \end{aligned}$$

$$DD2 = \begin{cases} 1 & x + y > \frac{3}{2} \\ 0 & x + y < \frac{1}{2} \\ x + y - \frac{1}{2} & \text{otherwise} \end{cases}$$

$$DD3 = \begin{cases} \cos^2(\pi s) & s \leq \frac{1}{4} \\ 0 & \text{otherwise} \end{cases}$$

where $$s = \sqrt{(x - \frac{1}{2})^2 + (y - \frac{1}{2})^2}$$

Function DD1 is a flat ramp down to a plane which has a "mountain" on it and was labelled F7 by Dyn *et al.* (see Figure 4.13). Functions DD2 and DD3 are, respectively, a flat ramp and dome sitting on a circular cylinder raised out of a flat plane. Both were suggested by Reeves (1991) and are shown in Figures 4.14 and 4.15 respectively.

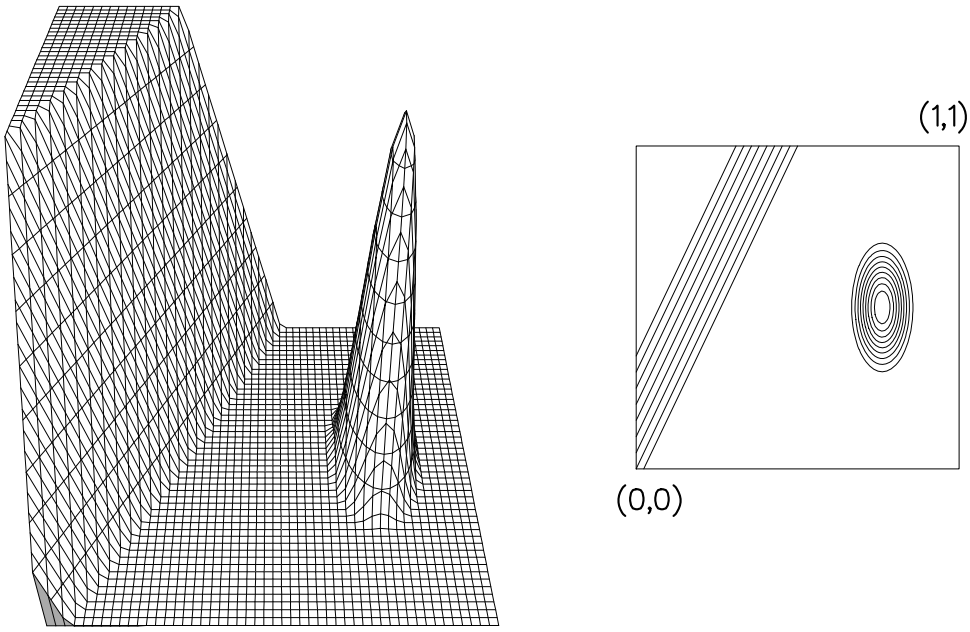

Figure 4.13: Function DD1

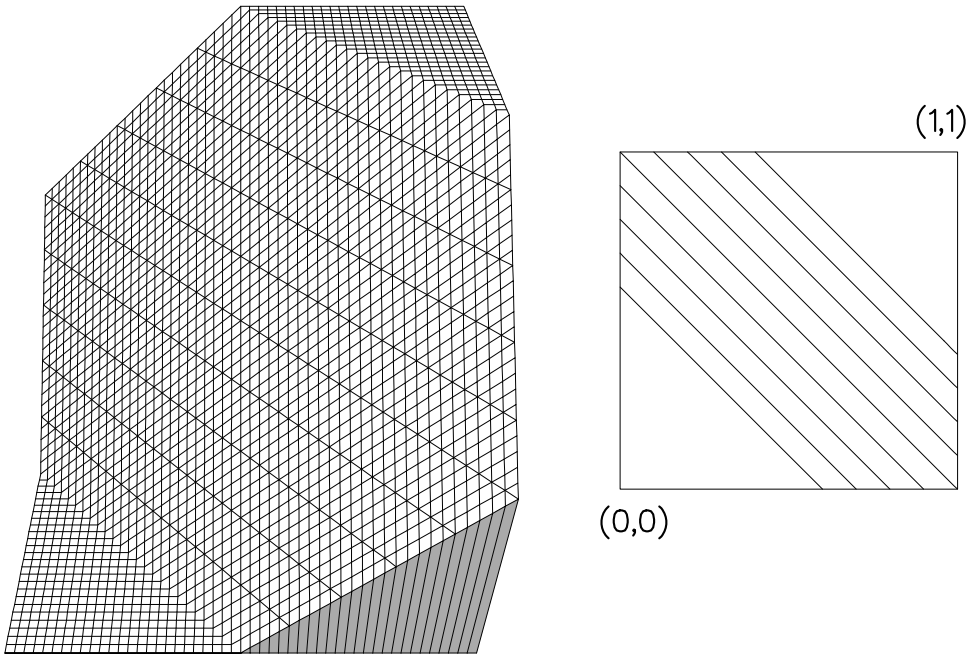

Figure 4.14: Function DD2

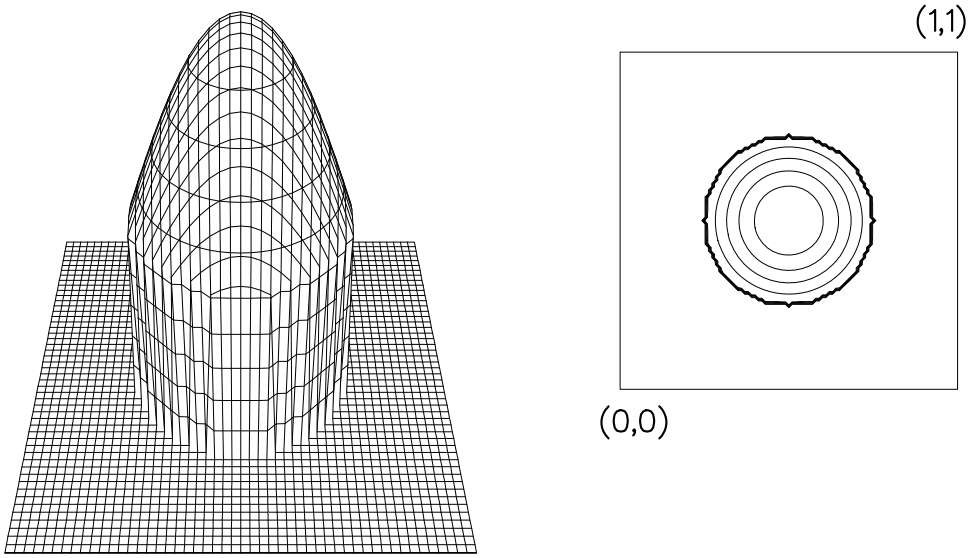

Figure 4.15: Function DD3

Having described all the test functions which were used, the data sets with which these were tested are now detailed.

4.2 Data Sets

The data sets used here to produce original triangulations are either regular sets of points or modifications of the scattered data sets presented by Franke (1979). The regular set of data points, see Figure 4.16, was used to test the effect of reordering the points in a data set. The smaller of Franke's data sets, 33 points,

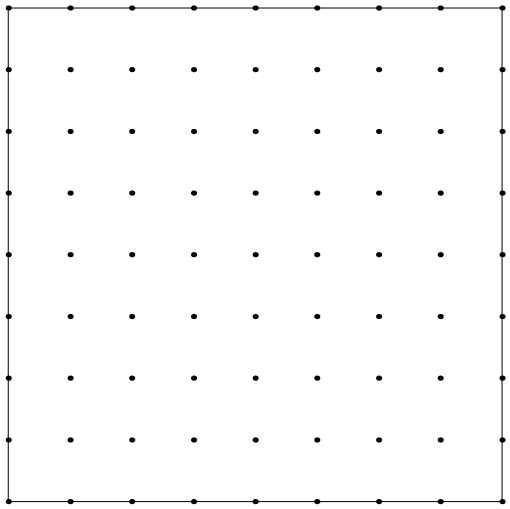

Figure 4.16: 81 Regular data points relative to unit square

is unchanged, see Figure 4.17, while the larger of Franke's data sets is modified,

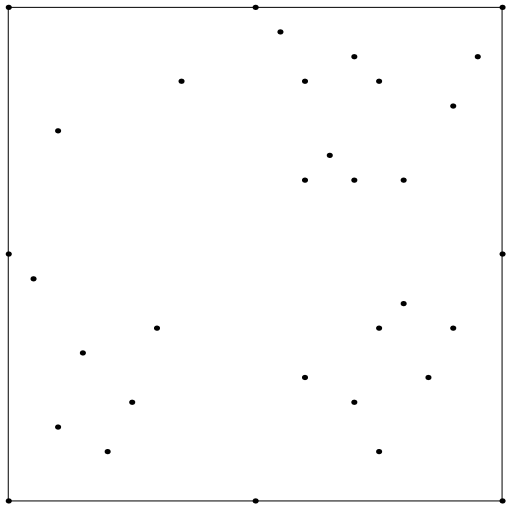

Figure 4.17: 33 data points relative to unit square

see Figure 4.18, so that points outside the unit square or within a small tolerance of the unit square's perimeter are moved onto the perimeter, since, unlike Franke, the aim is to triangulate a region rather than a scattered data set (see Appendix A for lists of the points used in Franke's data sets).

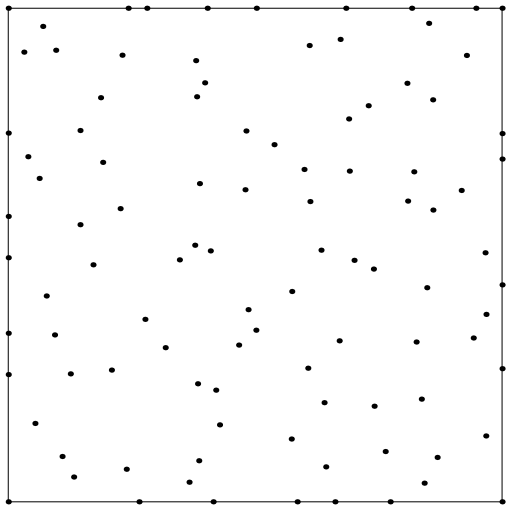

Figure 4.18: 100 data points relative to unit square

For the production of points and triangles from the triangulation front programs one set of vertices was used. This set was the vertices of the unit square.

Having depicted the data sets, the next section presents the performance monitors.

4.3 Result Monitors

There are many ways in which the triangulations produced can be judged to decide if they represent data well and if they possess qualities which are desirable to the user, e.g. for an elliptic problem treated by the Finite Element Method a triangulation with small angles might lead to to a poorly conditioned system (see Strang and Fix (1973)).

The accurate representation of data on the triangulation can be checked in many ways. Dyn, Rippa and Levin (1990) check the data representation by looking at the error between the underlying function and the interpolant at the grids points of a superimposed 33 by 33 regular grid. The values found are then used to measure the quality in three ways, root mean square error (RMS), maximum error and mean error. Other methods of measuring accuracy include looking at the L_2-error of the whole triangulation or looking at the maximum error on each triangle (D'Azevado and Simpson (1989)).

The geometric quality of the triangles and the triangulation can be checked by looking at various monitors which include aspect ratio, skewness, size of angles, uniformity of triangles and number of triangles meeting at a point (see Catherall and Fitzsimmons (1991)). Lo (1985) introduces an alternative aspect ratio to check the acceptability of the triangles which could be produced during his moving front procedure. If the objective of a grid generation procedure is to have a regular grid of close to equiangular triangles then the maximum, minimum and average of these geometrically based quantities can be used to determine the overall quality of the triangulation, or to show which triangles do not conform to such an objective.

Given a triangle, t, with sides of length $s_1 > s_2 > s_3$ and opposite angles of $\theta_1 > \theta_2 > \theta_3$ respectively, the geometric properties of the triangle can be considered in many ways.

Skewness is defined as the departure from equiangularity of a triangle, and is defined for each triangle, t, as

$$sk(t) = \frac{1}{3}\sum_{i=1}^{3}\left|\frac{\pi}{3} - \theta_i\right|.$$

Uniformity (expansion ratio) can be described as follows :- Given a triangle of area A with neighbouring triangles of area B, C and D, the uniformity of the central triangle is defined as

$$er(t) = \max(\frac{A}{B}, \frac{B}{A}, \frac{C}{A}, \frac{A}{C}, \frac{A}{D}, \frac{D}{A}).$$

This gives a measure for each triangle of whether the size of triangles in the grid is smoothly increasing, or decreasing, in the vicinity of the given triangle.

The Aspect Ratio of a triangle is a monitor of the overall shape of a triangle. Two measures of this have been defined. The first, presented by Catherall and Fitzsimmons, is based on a measure presented by Cavendish (1974) in the execution of his triangulation front procedure, while the second is that of Lo. The two measures are as follows:-

Cavendish's Aspect Ratio is defined for a triangle, t, with longest side s_1 and perpendicular distance h from this side to the third vertex of the triangle as

$$\begin{aligned} ar_C(t) &= \frac{\sqrt{3}s_1}{2h} \\ &= \frac{2sin\theta_2 sin\theta_3}{\sqrt{3}sin\theta_1}. \end{aligned}$$

Lo's Aspect Ratio is defined as

$$\begin{aligned} ar_{Lo}(t) &= \frac{4\sqrt{3}\text{area of triangle}}{s_1^2 + s_2^2 + s_3^2} \\ &= \frac{2\sqrt{3}sin\theta_1 sin\theta_2 sin\theta_3}{sin^2\theta_1 + sin^2\theta_2 + sin^2\theta_3}. \end{aligned}$$

The numerical factors are such that a $(60°, 60°, 60°)$ triangle has an Aspect Ratio of 1 in both definitions.

The numerical value of the L_2-error of a triangulation is calculated using the

following formula

$$L_2 - \text{error} = \sqrt{\sum_{i=1}^{t} \int_{T_i} (u(x,y) - F_T)^2 dxdy}. \tag{4.1}$$

The integral is calculated by using 13 point Gaussian quadrature on a triangle. Although this might be regarded as an excessive number of points they are sometimes necessary for the accurate calculation of the error.

All these measures of triangle quality can be presented in either numerical or graphical form. The ideal balance of these monitors will depend on the specific situation and so in general the individual values will be reported.

Chapter 5

Grid Generation

Using Node Reconnection

In this chapter various criteria which can be used to implement reconnection of the nodes in a triangulation are presented. These criteria all depend on the underlying data function in order to evaluate the cost function upon which the node reconnection is based.

The general idea behind these criteria is that some measure, based on the interpolants or the data values, can be calculated for each edge of a triangulation. It is hoped that, if the sum of these measures is decreased by reconnection, the interpolant on the resultant grid gives a better representation of the underlying data than does the original grid.

The motivation behind some of the data-dependent node reconnection criteria is that a smoothly varying interpolant is best suited for approximating a smoothly varying function. The motivation for the other reconnection criteria is either to extend ideas from 1-dimensional, or geometrical grid generation, into a data

dependent framework. For the criteria based on smoothness, by introducing a measure of smoothness across each edge in the triangulation, based on the interpolants on the triangles adjacent to the edge, the "smoothest" interpolant can be found and the corresponding triangulation may be considered to give the best data representation. This idea was originated by Lee (1982) who introduced the idea of "steepest ascent" of the interpolant on a triangle. Lee used this to show that for a smooth function (i.e. with no derivative discontinuities) it was possible to triangulate a quadrilateral so as to minimise the interpolation error, by minimising some weight function of the steepest ascents of neighbouring triangles. However if a ridge occurs in the function then a better representation is obtained if the weight function is maximised in the vicinity of the ridge.

It is necessary to note that the grids produced which give good results for data representation may be unusable in practical situations due to the very small angles in the triangles generated. Such possible geometrical constraints can be introduced into the nodal reconnection procedure and in Section 5.7 the geometric constraints are detailed and their implementation is discussed. The results of using such additional constraints is shown in graphical and numerical results.

Firstly the notation which will be used is presented and following on from this, the criteria are outlined, and finally the implementation details are presented.

5.1 Notation

Before considering the various criteria which are presented, the notation which will be used is established.

Referring, where applicable, to Figure 5.1 the notation is, following that of Dyn,

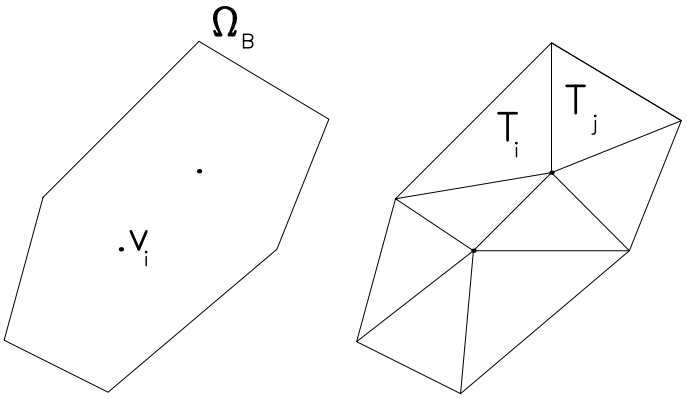

Figure 5.1: Boundary of region and triangles in region

Rippa and Levin (1990) :-

Ω is the region.

Ω_B is the convex hull of Ω.

N is the total number of data points in Ω.

N_B is the number of points on Ω_B.

V is the set of data points,

$$v_i = (x_i, y_i), \quad i = 1, \ldots, N.$$

F is the Data Set. If f is the underlying function being considered then

$$F_i = f(x_i, y_i), \quad i = 1, \ldots, N.$$

The number of triangles, T_i, in a triangulation is

$$t = 2(N-1) - N_B.$$

The number of edges in a triangulation is

$$e = 3(N-1) - N_B.$$

The triangulation, T, is the union of all triangles, i.e.

$$T = \cup_{i=1}^{t} T_i, \qquad T_i \cap T_j = \emptyset, \quad i \neq j.$$

f_T is the interpolating polynomial to the data on the triangulation, with

$$f_T = \cup_{i=1}^{t} f_i,$$

$$f_i = a_i x + b_i y + c_i, (x, y) \in T_i \qquad i = 1, \ldots, t$$

where f_i is the local interpolating polynomial on triangle T_i.

W is the array of the augmented set of data points.

$$W_i = (x_i, y_i, F_i), \quad i = 1, \ldots, N.$$

$\|\mathbf{z}\|$ is the Euclidean norm of the m-vector $\mathbf{z}$, i.e

$$\|\mathbf{z}\| = \sqrt{z_1^2 + z_2^2 + \ldots + z_m^2}.$$

Having presented the notation used in this chapter on nodal reconnection, it is now possible to introduce the criteria used to compare triangulations.

Since different triangulations will be compared a notation is needed to indicate the result of the comparison. Thus if a triangulation T is preferred to T' in the sense of some cost function value (see below), this is denoted by $T < T'$.

The sense in which triangulations are judged is as follows, referring where necessary to Figure 5.2.

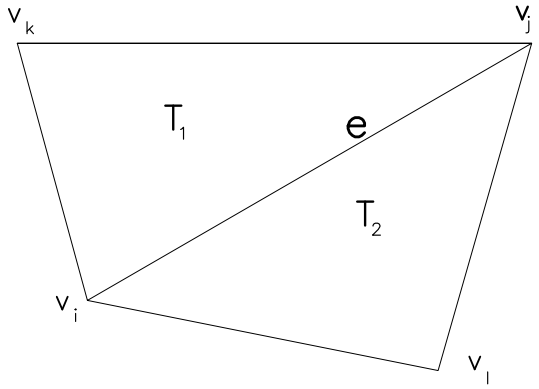

Figure 5.2: Quadrilateral formed by two triangles

Let $e = \overline{v_i v_j}$ be an internal edge of a triangulation T and let T_1 and T_2 be the two triangles sharing the common edge, e.

Suppose that

$$f_1 = P_1(x, y) = a_1 x + b_1 y + c_1 \qquad (x, y) \in T_1$$

$$f_2 = P_2(x, y) = a_2 x + b_2 y + c_2 \qquad (x, y) \in T_2$$

are the linear interpolating polynomials on triangles T_1 and T_2 respectively.

For each interior edge, e of the triangulation T, a real cost function $s = s(f_T, e)$ is assigned, dependent on the criterion being used.

Let $\mathbf{N}$ and $\mathbf{N}'$ be real vectors of size q, with the components ordered in a non-increasing manner. i.e. from largest to smallest. Ordering schemes on R^q are defined such that to say that $\mathbf{N} < \mathbf{N}'$ means that the triangulation T which produced $\mathbf{N}$ is better than the triangulation T' which produced $\mathbf{N}'$, or $\mathbf{N} < \mathbf{N}' \Rightarrow T < T'$.

The ordering schemes considered are:

1. Ordering by the L_1 norm

$$R_1(\mathbf{N}) = \sum_{i=1}^{q} |N_i|$$

   and $\mathbf{N} \leq \mathbf{N}'$ if $R_1(\mathbf{N}) \leq R_1(\mathbf{N}')$.

2. Ordering by the L_2-norm

$$R_2(\mathbf{N}) = \sqrt{\sum_{i=1}^{q} |N_i|^2}$$

   and $\mathbf{N} \leq \mathbf{N}'$ if $R_2(\mathbf{N}) \leq R_2(\mathbf{N}')$.

3. Ordering "lexicographically".

   $\mathbf{N} \leq \mathbf{N}'$ if the vector $\mathbf{N}$ is "lexicographically" not greater than $\mathbf{N}'$.

   i.e. remembering that the vector components are stored in a non-increasing manner, compare $\mathbf{N}$ and $\mathbf{N}'$ component by component.

   if $N_1 < N_1'$ then $\mathbf{N} < \mathbf{N}'$

   if $N_1 > N_1'$ then $\mathbf{N} > \mathbf{N}'$

   otherwise check $N_2, N_2', N_3, N_3', \ldots, N_q, N_q'$.

   if $N_i = N_i', \quad i = 1, \ldots, q \quad$ then $\quad \mathbf{N} = \mathbf{N}'$.

Having outlined methods of determining an ordering for a set of vectors, in the next section the concept of data dependent triangulations is introduced. The definition of an optimal triangulation for a given criterion is presented using the ordering of triangulations previously defined.

5.2 Data Dependent Triangulations

In this section the definition of an "optimal triangulation" is given, following Rippa (1991), and a local optimisation procedure which he used to attempt the generation of such a triangulation is introduced.

Definition 5.1 *An optimal triangulation of a region Ω, given a fixed set of nodes, with respect to a given criterion is the triangulation T^* such that*

$$T^* \leq T$$

for every triangulation T of Ω.

From this definition it can be seen that the optimal triangulation with respect to a criterion need not be unique.

An optimal triangulation of Ω always exists since there are a finite number of triangulations of Ω. However, it may be difficult to obtain this optimal triangulation in practice since it is quite possible to reach a local minimum of the cost function associated with the criterion instead, although this situation can be alleviated by the use of a numerical optimisation technique such as Simulated Annealing (see Section 5.6).

Let T be a triangulation, e an internal edge of T, and Q a quadrilateral formed by the two triangles having e as a common edge. If Q is strictly convex then there are two possible ways of triangulating Q (see Figure 5.3).

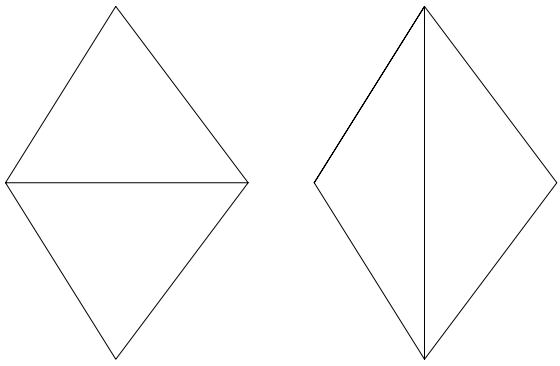

Figure 5.3: Two triangulations of a convex quadrilateral

Definition 5.2 *An edge e is called locally optimal if $T \leq T'$ where T' is obtained from T by replacing e by the other diagonal of Q.*

This leads to

Definition 5.3 *A locally optimal triangulation of Ω is a triangulation T' in which all edges are locally optimal.*

A data dependent triangulation of a fixed set of nodes V, depends on either the data vector F (if it is a scattered data interpolation problem), or the underlying function (if it is a function approximation problem), and so the preferred triangulation cannot be the same for all data sets but varies depending on the data vector or the function.

The optimisation procedure used to construct the triangulations is based on the Local Optimisation Procedure (LOP) suggested by Lawson (1977), which can be written in algorithmic form as follows :-

1. Construct an initial triangulation $T^{(0)}$ of Ω and set $T \leftarrow T^{(0)}$.
2. If T is locally optimal - end the procedure, else go to step 3.
3. Let e be an internal edge of T which is not locally optimal and let Q be the strict convex quadrilateral formed by the two triangles having common edge e. Swap diagonals of Q, replacing e by the other diagonal of Q, therefore transforming T to T'.
4. Set $T \leftarrow T'$ and go to step 2.

This means that after every edge swap occurs, the resulting triangulation is strictly lower in the ordering than the previous one. Since the number of triangulations is finite then the LOP converges after a finite number of edge swaps to a locally optimal triangulation. However it is necessary to mention again that there is no guarantee that the locally optimal triangulation is the globally optimal triangulation.

In the next section the various data dependent criteria under investigation are described.

5.3 Criteria for Producing Triangulations

In this section data dependent criteria for nodal connection are introduced. The data dependent analogies of the geometric triangulation procedures described in Section 2.2 which have the properties of MAX-MIN angle and minimum sum of edge lengths are described. Next the recent work of D'Azevado and Simpson (1989) is presented which uses the Delaunay triangulation in transformed space to obtain a data dependent triangulation. The data dependent criteria categories, nearly C^1 (NC1) and near-planar, from Dyn, Rippa and Levin (1990) are then described followed by a criterion based on equidistribution, (Sweby (1987)). Some illustrative results for all these criteria are presented as they occur throughout the chapter and tables of numerical results follow in Section 5.4.

First, however, some results for the non-data dependent Delaunay and MWT procedures are presented, (see Section 2.2) so that comparisons can be made between them and the data dependent criteria outlined later.

5.3.1 Results for the Geometrical Criteria

We present the Delaunay triangulations of the data sets here to demonstrate the nearly equiangular triangles which are produced by the Max-Min angle property.

Figure 5.4 shows the Delaunay triangulation of the 33 point data set, and Figure 5.5, the Delaunay triangulation of the 100 point data set. Both show large numbers of nearly equiangular triangles in the centre of the region. However it is noted that near to the boundary edges, triangles with small angles appear due to the scarcity of nodes on the boundary.

At this point it is necessary to remind ourselves that we are effectively con-

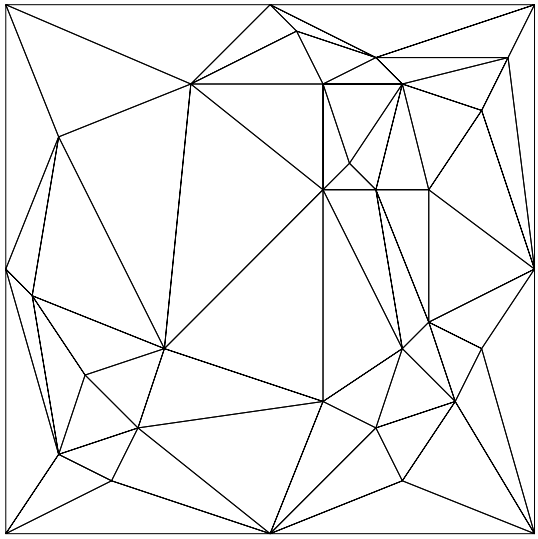

Figure 5.4: The Delaunay Triangulation of the 33 Point Data Set

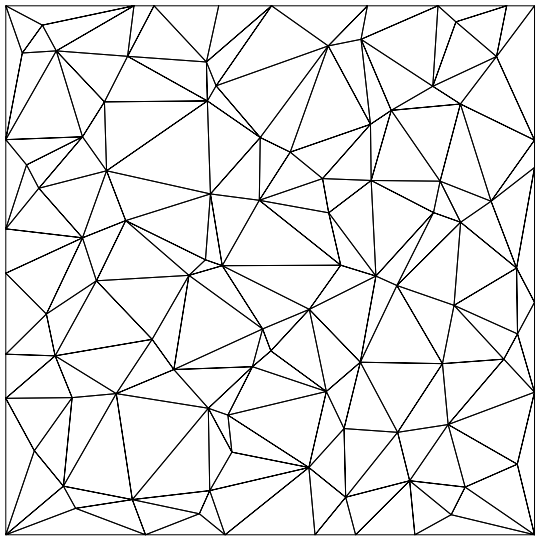

Figure 5.5: The Delaunay Triangulation of the 100 Point Data Set

cerned in this chapter with the triangulation of a *fixed* set of nodes (the scattered data problem) rather than initial grid generation wherein extra nodes can be inserted or nodes moved to improve the function representation. This means that in some cases the poor representations are due to a lack of nodes in a region of interest, rather than a fault of the triangulation; this problem will be addressed in later chapters. Note however that, although the haphazard placement of the nodes within the unit square may seem far from satisfactory, the situation may

well mimic the behaviour when a geometric grid is generated for a more complicated geometry and a function then represented on it. It is therefore still constructive to see the improvements that can be made by nodal reconnection alone.

We also see how Delaunay deals with a regular pattern of nodes, (Figure 5.6), where a regular grid of triangles is produced. Note however, the direction of connection of this regular framework of triangles depends on the order in which the nodes are introduced in the node insertion procedure. Figure 5.7 shows how a change in the ordering of the nodes can produce a totally different pattern of triangles due to the degeneracy property of the Delaunay procedure (the circumcircle of any triangle has 4 nodes on its circumference in the finished grid (see Section 2.2)).

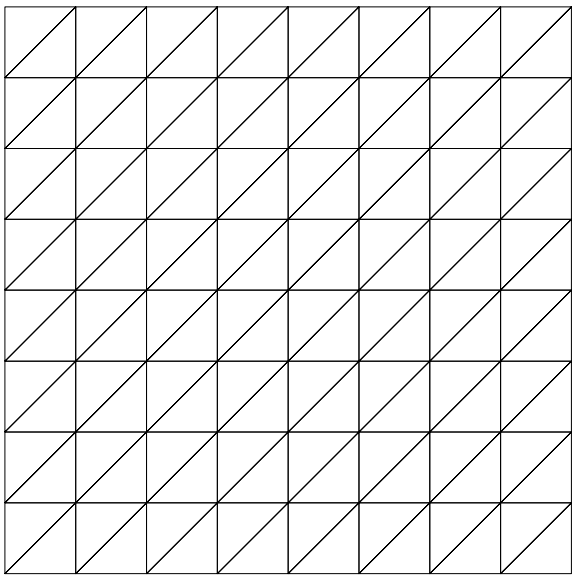

Figure 5.6: The Delaunay Triangulation of the 81 Point Data Set

Figures 5.8 to 5.16 illustrate the failure of the Delaunay triangulation to well represent data. Figures 5.8 to 5.12 show the representation of the underlying function obtained, with linear interpolants on the triangles, when the grid for

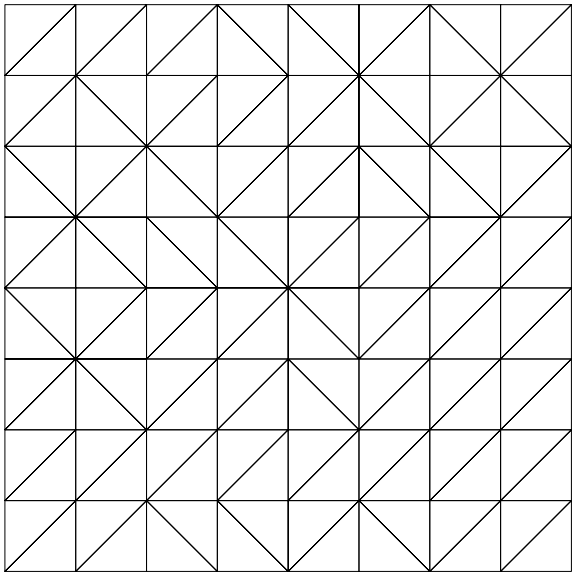

Figure 5.7: The Delaunay Triangulation of the re-ordered 81 Point Data Set

the 33 data point set shown in Figure 5.4 is used. Figure 5.8 shows that for a function with a preferred direction the Delaunay triangulation can produce very poor representations of the region where there is rapid change in function value. This is particularly evident when compared with Figure 4.1, which shows the actual contours.

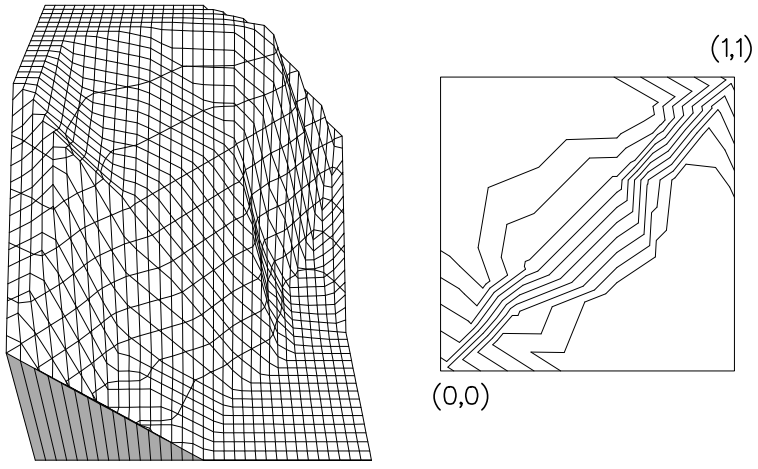

Figure 5.8: Representation of SR1 by Figure 5.4

Figure 5.9 again shows the poor representation in regions of rapid change,

although the poor representation at the ends of the ramp is due in part to the small number of boundary nodes, whilst the poor representation of the mountain is due in part to the lack of points in the region of the mountain. In comparison with Figure 4.13, which shows the actual function and contours, Figure 5.9 shows the smearing of the bottom of the ramp towards the mountain and the differences in the contours in the ramp and mountain. On a finer regular grid these problems are not as pronounced, see Figure 5.14.

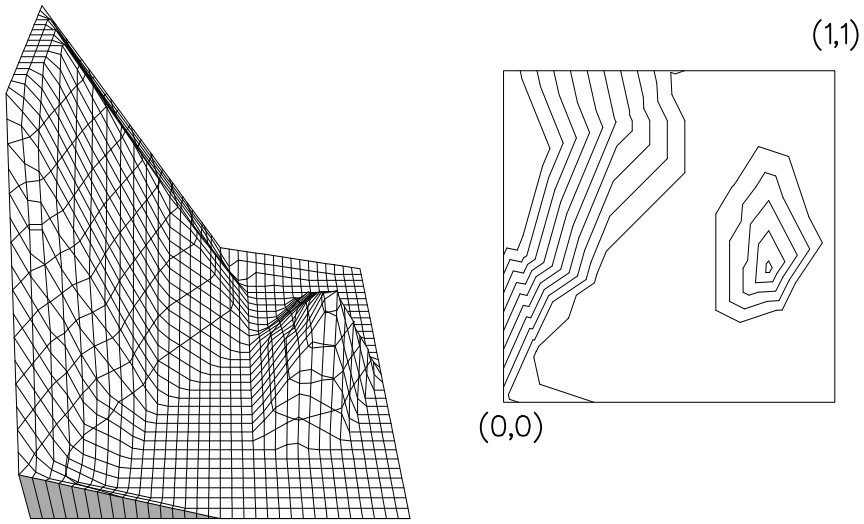

Figure 5.9: Representation of DD1 by Figure 5.4

Figures 5.10 to 5.12 show the representations given by the Delaunay triangulation in Figure 5.4, of some of the other test functions. These can be compared with the actual functions and contours which are shown in Figures 4.3, 4.9 and 4.10 respectively.

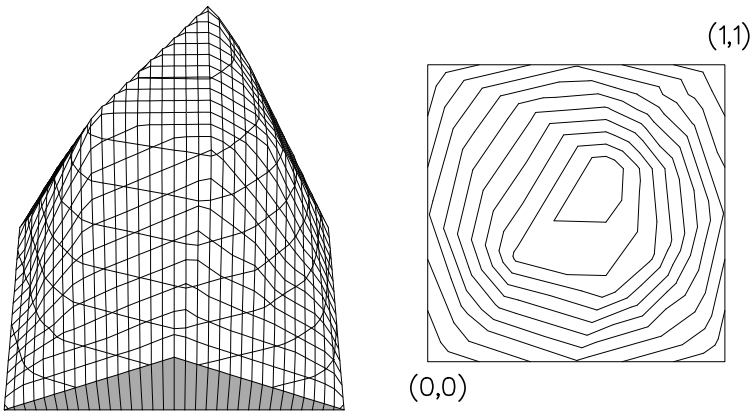

Figure 5.10: Representation of SH1 by Figure 5.4

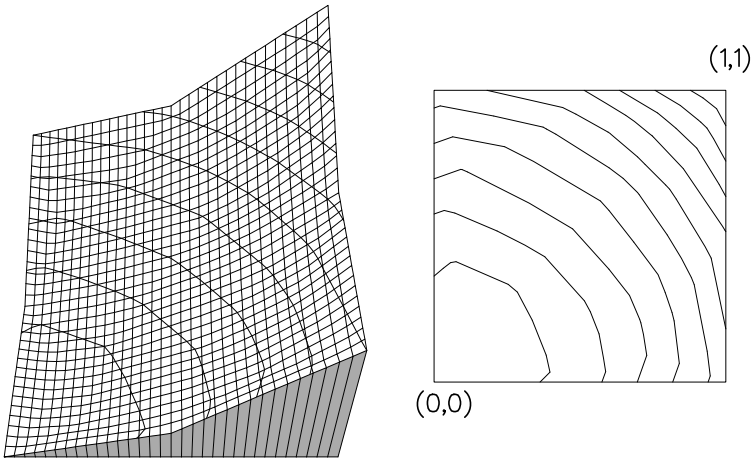

Figure 5.11: Representation of P1 by Figure 5.4

Having looked at representations given by a small number of nodes, we now move on to the larger data sets.

The representation of functions, given by the regularly patterned Delaunay triangulation of 81 points, see Figure 5.6, can be informative. Figure 5.13 shows that if the triangles are aligned with the contours of the function then a good representation of a rapidly changing function is possible, while Figure 5.14 shows that the mountain can be well represented if there are data points near its centre

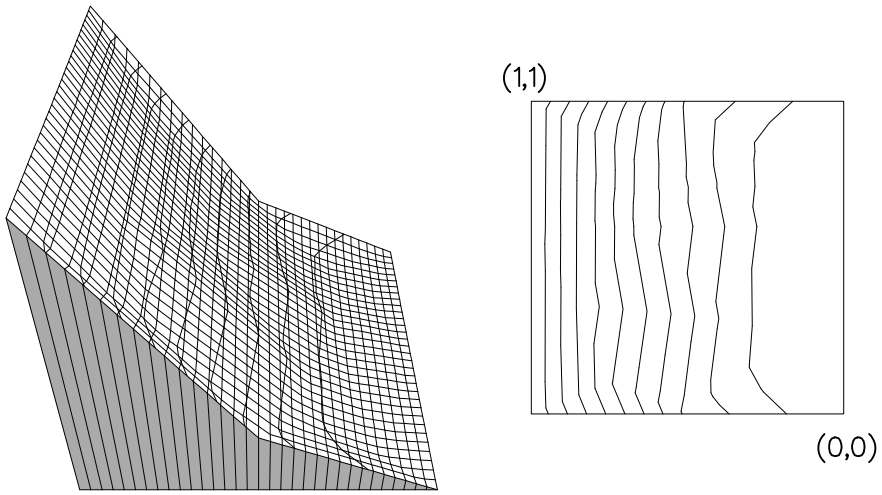

Figure 5.12: Representation of P2 by Figure 5.4

and that the bottom and top of the ramp can be poorly represented if the triangle edges do not align exactly with the functions contours.

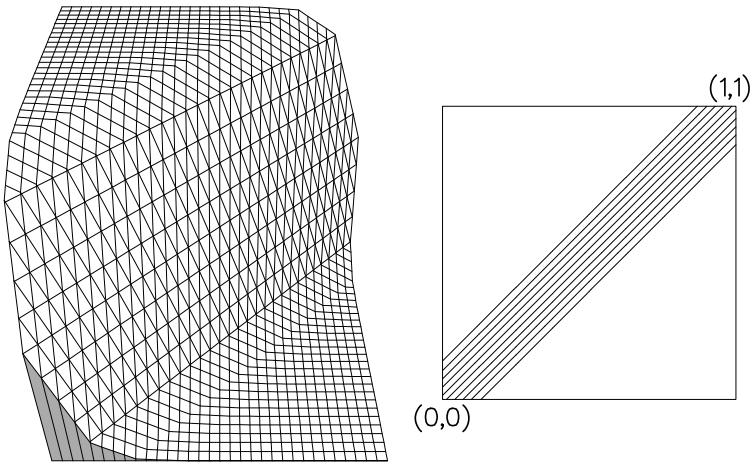

Figure 5.13: Representation of SR1 by Figure 5.6

The results of having more nodes, i.e. 100 compared to 33, in a non-regular pattern can be seen in Figures 5.15 and 5.16. Figure 5.15 shows that even with more triangles in the Delaunay triangulation, see Figure 5.5, a rapidly changing function can still be poorly represented, i.e the contours are still not straight

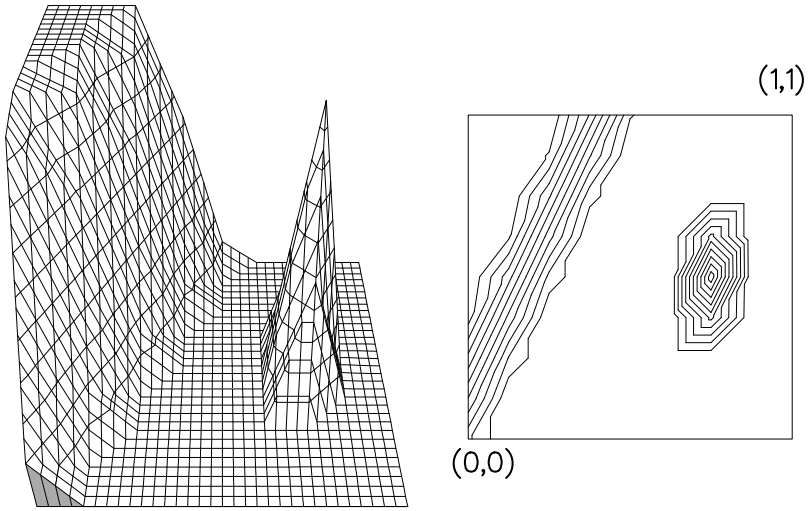

Figure 5.14: Representation of DD1 by Figure 5.6

lines, while Figure 5.16 shows that even with more points in the region of the mountain, unless there is at least one point very close to the centre of the region, the representation of the mountain can still be poor.

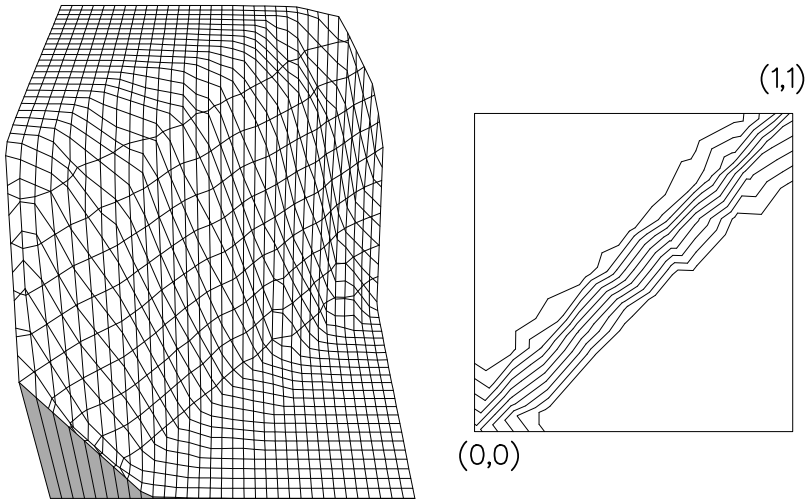

Figure 5.15: Representation of SR1 by Figure 5.5

Having presented the results of the Delaunay triangulation, we now move on to the other geometrical procedure, the Minimum Weight Triangulation (MWT).

The grids produced by the MWT criterion for the 33 and 100 data point

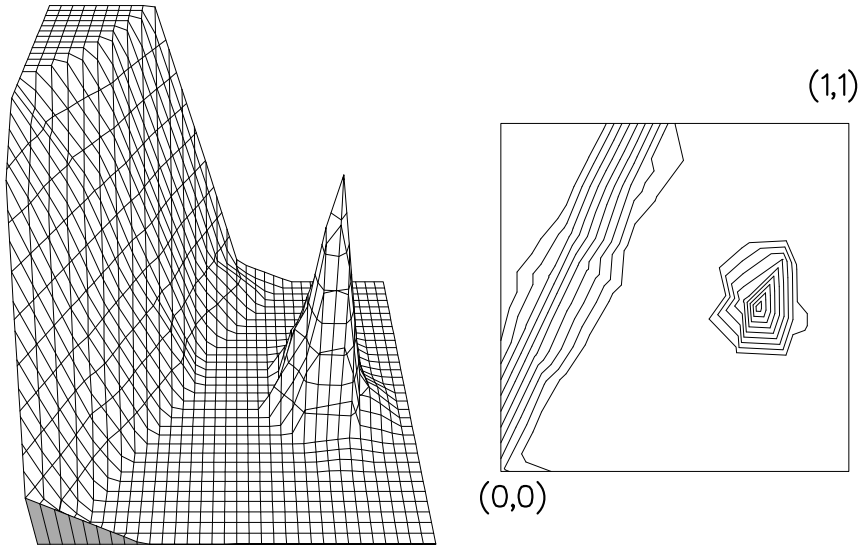

Figure 5.16: Representation of DD1 by Figure 5.5

sets are displayed in Figures 5.17 and 5.18 respectively. These enable us to view the minor differences in triangulations produced by the geometrical criteria. By comparing Figures 5.4 and 5.17, we see that, near the boundaries, the long, thin triangles which are produced by the MWT criterion. While in Figure 5.18, when compared to Figure 5.5, we see long, thin triangles being formed in the centre of the region. The function representations for these grids are very similar to those shown in Figures 5.8 to 5.12 and Figures 5.13 and 5.14, so we will not reproduce them here.

This completes the presentation of the results of the geometric triangulation criteria and we can now introduce the data dependent extensions of these criteria.

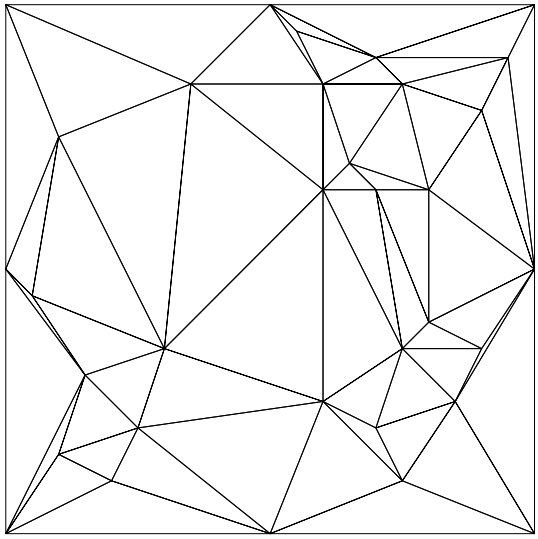

Figure 5.17: The Minimum Weight Triangulation of the 33 Point Data Set

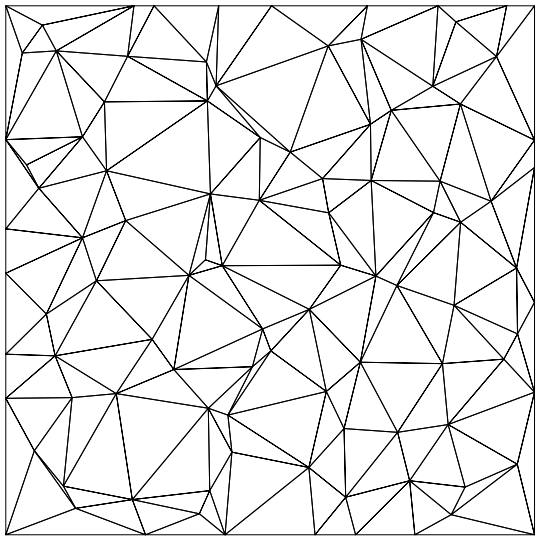

Figure 5.18: The Minimum Weight Triangulation of the 100 Point Data Set

5.3.2 Data Dependent Extensions of the Previous Criteria

The data dependent extensions of the MWT and the MAX-MIN angle property are detailed here, as is the work of D'Azevado and Simpson (1989) who show that for a certain class of functions it is possible to construct an L_p-optimal triangulation by using the Delaunay triangulation in transformed space.

The 3-D MAX-MIN Angle Criterion

This criterion is a data dependent extension of the Delaunay Triangulation. Rather than trying to maximise the minimum angle in a set of triangles with vertices given by V, this criterion attempts to maximise the minimum 3-D angle in a set of triangles with vertices given by W. i.e. the triangles are formed in 3-dimensional space rather than 2-dimensional space. It is possible to think of the Delaunay MAX-MIN angle criterion as the projection of an angle of W onto V. The calculation of the 3-D angle is aided by recalling that

$$\cos(\angle ijk) = \frac{(W_k - W_j, W_i - W_j)}{\|W_k - W_j\|\|W_i - W_j\|}.$$

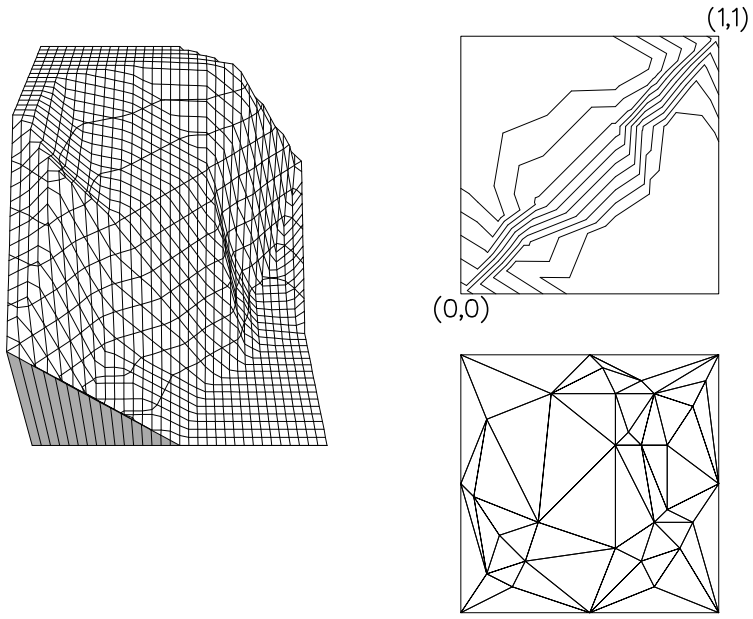

Figure 5.19: The 3-D MAX-MIN Angle Triangulation for SR1 (33 Points)

Comparing Figure 5.4 to the grid in Figure 5.19 we can see that for smoothly varying functions there may be only slight differences between the 3-D MAX-MIN angle triangulation and the Delaunay triangulation. However, looking at the grid in Figure 5.20 we can see that in general it is possible to have major differences in

triangulations. The abysmal representation of the mountain is due to the lack of points near the centre of the mountain and the fact that points on either side of the mountain are on the same triangle. The numerical results in Table 5.3 show that the L_2-error increases with respect to that of the Delaunay triangulation.

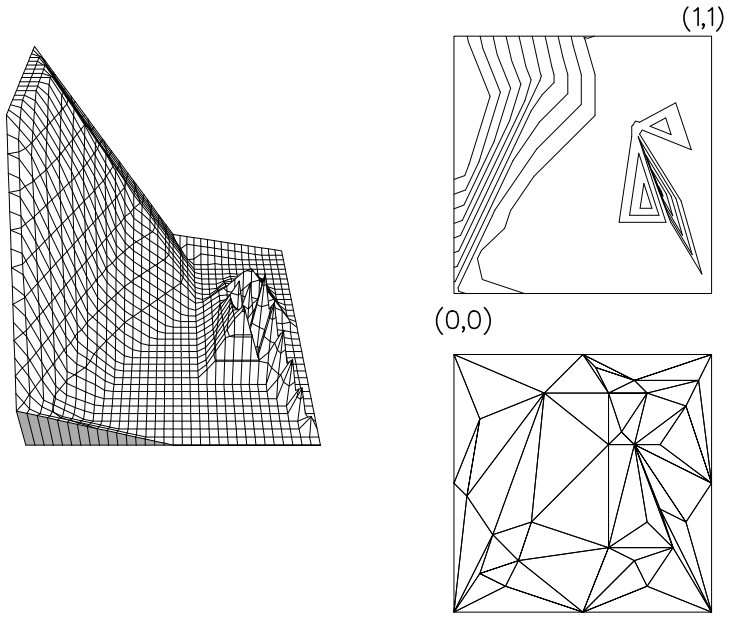

Figure 5.20: The 3-D MAX-MIN Angle Triangulation for DD1 (33 Points)

MWT-3D

This is the data dependent extension by Dyn, Rippa and Levin (1990), of the MWT where attempts are made to minimise the sum of the 3-D lengths of the edges of all the triangles, with vertices given by W, in a triangulation, i.e.

$$\text{Min} \sum_{T_L \in T} \sum_{i,j \in T_L} \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2 + (F_i - F_j)^2}.$$

Hence MWT works on a projection of W onto V.

The MWT-3D triangulation for Function SR1 is unchanged from that of the geometric MWT, but the MWT-3D triangulation for Function DD1, as shown in Figure 5.21, exhibits major differences from that of the geometric MWT, as

shown in Figure 5.17. These differences occur in the ramp, where the triangles are starting to lie along the ramp, rather than across it, thus giving a better representation of the feature, and in the mountain, where again, it is possible to have two nodes on either side of the mountain in a triangle thus creating a break through the mountain.

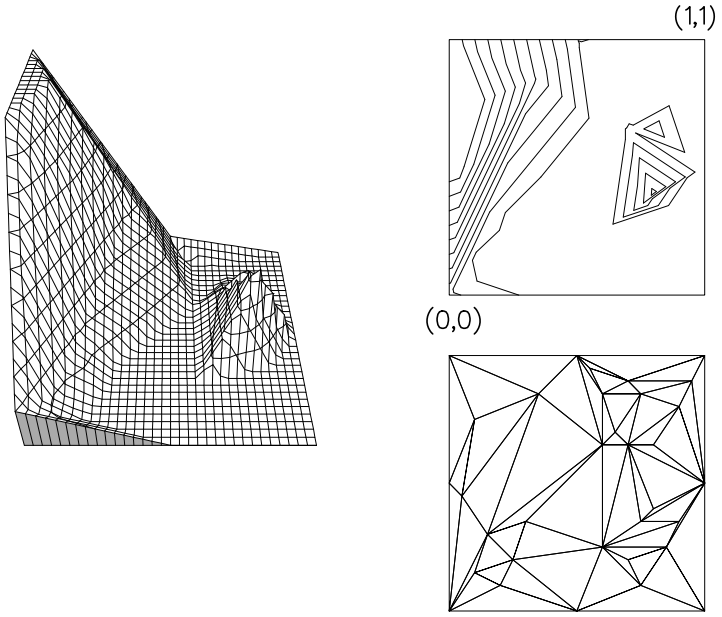

Figure 5.21: The MWT-3D Triangulation for DD1 (33 Points)

These results can be improved by the use of a factor to increase the dependency of the value on the function value, i.e.

$$\text{MWT-3D (mod)} = \text{Min} \sum_{T_L \in T} \sum_{i,j \in T_L} \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2 + \omega(F_i - F_j)^2}.$$

We can see in the figures below how, with large values of ω, this modification can improve the results previously obtained. Figure 5.22 shows that the triangles are starting to lie along the contours of the function, with the result that the contours are bunching closer together. Figure 5.23 shows an even greater improvement of representation than that in Figure 5.21, since, although the mountain is still poorly represented, the ramp is represented in an improved fashion, with less

smearing at the top and bottom. The numerical results show that the addition of the factor ω decreases the errors associated with the representations shown in the figures.

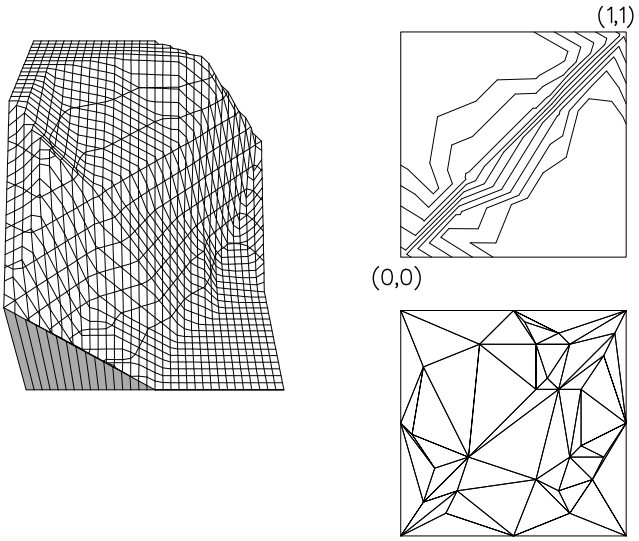

Figure 5.22: The MWT-3D (mod) Triangulation for SR1 with $\omega = 10$ (33 Points)

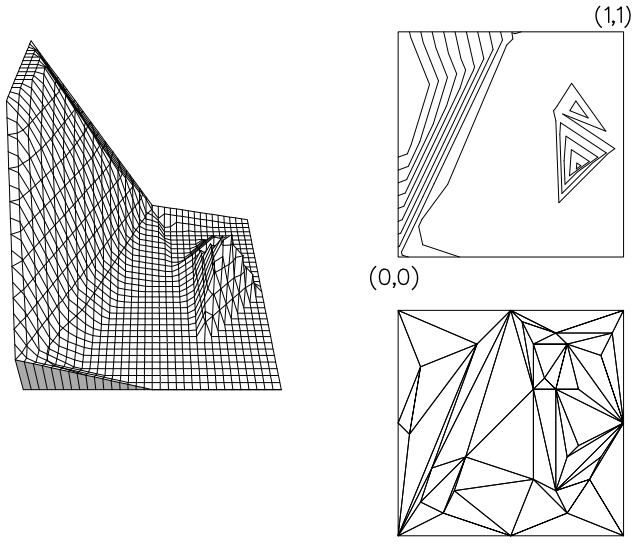

Figure 5.23: The MWT-3D (mod) Triangulation for DD1 with $\omega = 10$ (33 Points)

Having presented the data dependent extensions of geometric criteria it is now possible to introduce a truly data dependent criterion which uses the Delaunay triangulation in a transformed space.

D'Azevado and Simpson's Transformation

The ideas of D'Azevado and Simpson (1989) are presented and an investigation made to see how these ideas can be applied in an node reconnection framework. The problems which may be encountered on a global basis in this work are analysed and the benefits of local techniques are discussed.

D'Azevado and Simpson showed that for a general, strictly convex, bi-variate quadratic polynomial,

$$f(x,y) = \lambda_1 x^2 + \lambda_2 y^2 + \text{lower order terms}, \qquad \lambda_1, \lambda_2 > 0, \tag{5.1}$$

it is possible to introduce a change of coordinate variables such that, in the transformed space, the triangulation which minimises the error in the L_p-norm, $1 \leq p < \infty$, of the interpolant, is a Delaunay triangulation. These results can be extended to all strictly convex, bivariate functions by looking at the Hessian matrix of the function.

In its original form this is a totally global method of grid generation, whereby the triangulation is produced by the use of the Delaunay triangulation on a set of data points, which produces a unique solution (notwithstanding degeneracy).

The basis of the method is that for the function

$$g(x,y) = x^2 + y^2 + \text{lower order terms} \tag{5.2}$$

the optimal triangulation to reduce the error in any norm is the Delaunay triangulation, Rippa (1991). Thus if the function (5.1) is transformed into the function (5.2) in a transformed space, a Delaunay triangulation, (in transformed space) of the points associated with the function (5.1), will produce the optimal connectivity for the non-transformed points.

D'Azevado and Simpson look at the error on each triangle $E(x, y)$ and note that, if linear interpolants are used, the error contour $E(x, y) = 0$ is a circumscribing ellipse of the triangle. The equation of this ellipse can be found and it is then possible to calculate the maximum error on the triangle, which occurs either at the centre of the ellipse, if it lies inside the triangle, or at the mid-point of the longest side of the triangle otherwise. It is possible to transform the ellipses into circles by the use of the transformation

$$\xi = \sqrt{\lambda_1}\, x, \qquad \eta = \sqrt{\lambda_2}\, y.$$

D'Azevado and Simpson demonstrate that decreasing the diameter of the circumcircles in transformed space minimises the maximum interpolation error for (5.2) using linear interpolants. They also show that minimising the diameter of the circumcircles is equivalent to the Delaunay circumcircle property, (see Section 2.2), and thus a Delaunay triangulation in transformed space will give the connectivity for the triangulation with the minimal L_p-error in untransformed space. This has also been shown by Rippa (1991) who proves that the Delaunay triangulation is L_p-optimal for the function $h(x, y) = x^2 + y^2$. This can be seen by comparing Figure 5.24, the triangulation which minimises the L_2-error of the Function P1 $= x^2 + y^2$, with Figure 5.5, the Delaunay triangulation of 100 data points. These triangulations are identical and so Rippa's result is verified (in the case $p = 2$).

Figures 5.25 and 5.26 show the results of using D'Azevado and Simpsons transformation on the Function P2 $= x^2 + 100y^2$. The result of the procedure is to produce a large number of long, thin triangles oriented along the contours, and in so doing it decreases the mean interpolation error and the L_2-error by approximately 50%.

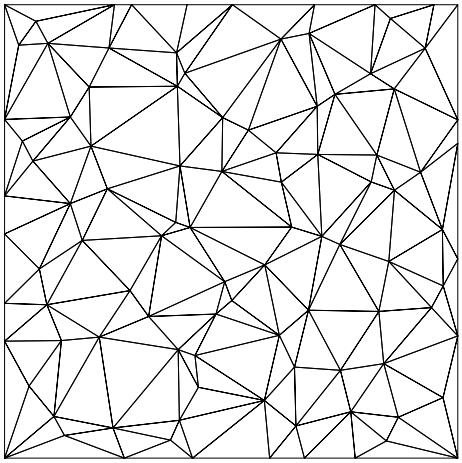

Figure 5.24: The minimal L_2-error Triangulation for P1 (100 Points)

Once again, the triangulations in Figures 5.25 and 5.26 are exactly the same as those which minimise the L_2-error for P2. They also show that long, thin triangles can improve the data representation.

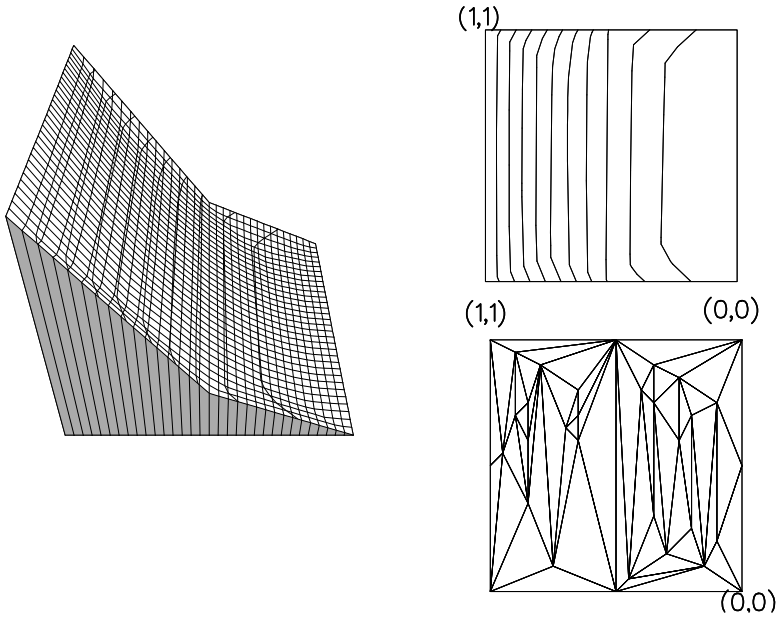

Figure 5.25: The D'Azevado and Simpson Triangulation for P2 (33 Points)

These results work very well for strictly convex functions but if the function is only just convex with values of λ_1 or $\lambda_2 = 0$, then in many cases the data sets will not transform to data sets where the same geometric properties apply, i.e. vertices do not transform to vertices but to interior points, or the transformation

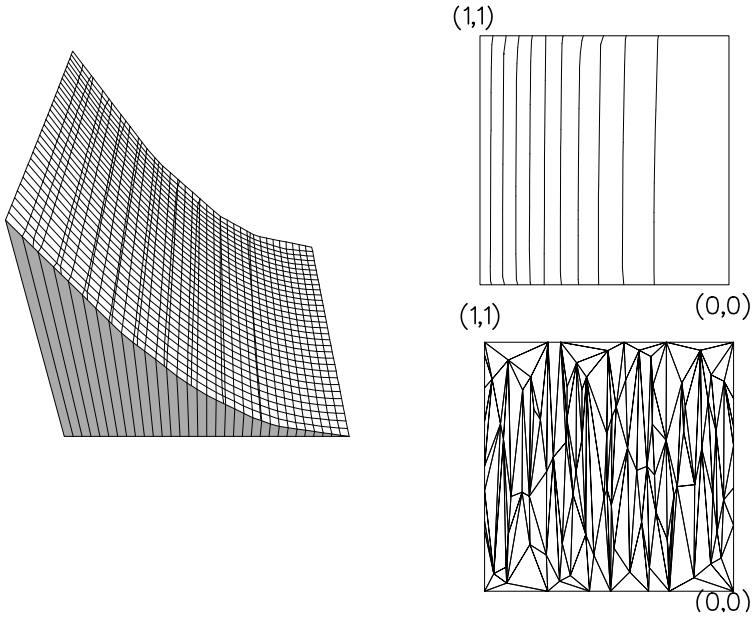

Figure 5.26: The D'Azevado and Simpson Triangulation for P2 (100 Points)

is not injective. For example, for Function P3 $= (1-x)^4 + 5(1-y)^4$ the points $(0,0)$ and $(1,1)$ both transform to $(0,0)$.

We overcame these global problems by looking at the problem in a local framework. Thus rather than looking at a problem in a global sense and triangulating all the nodes in transformed space, it is necessary to look at two neighbouring triangles which form a convex quadrilateral and then map their vertices to a transformed space and, provided that they form a convex quadrilateral, it is then possible to check the circumcircle or the MAX-MIN angle criterion in the transformed space in case a diagonal swap can be made. The connectivity can then be changed and the modified connectivity used in the untransformed space.

For positive definite functions this procedure works in exactly the same way as the global procedure and for positive semi-definite functions it produces valid triangulations, as can be seen in Figure 5.27, which also shows that the local triangulation procedure can produce triangles oriented along the contours of the

underlying function. This local technique means that for non-strict convex functions it is possible to improve the quality of data representation. However for many functions, such as DD1, even this "local" technique is not guaranteed to work as it is unable to cope with regions where $\lambda_1 = 0$ and $\lambda_2 = 0$, (the ramp) or where λ_1 and λ_2 have different signs (the mountain). In these cases the transformation concept is not a viable proposition.

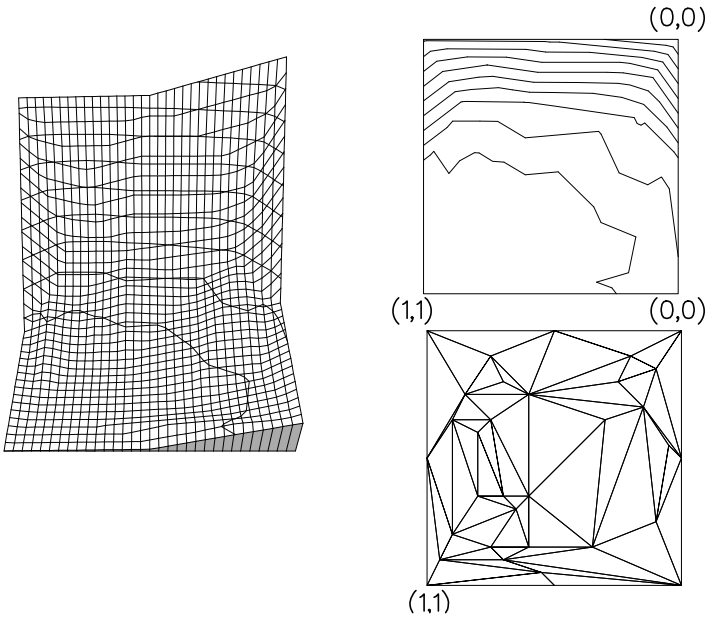

Figure 5.27: The "Local" D'Azevado and Simpson Triangulation for P3 (33 Points)

The next section details truly data dependent criteria which are not extensions of geometric criteria.

5.3.3 Nearly C^1 Data Dependent Criteria

The following criteria, from Dyn, Rippa and Levin (1990), are based on the premise that if the surface produced by the interpolating function is as smooth as possible then the errors in interpolating the underlying function will be reduced,

Lee (1982).

The description of the criterion used will be in the form of letters to abbreviate the name and a number to indicate which vector ordering scheme was applied. The results shown are comparable to those produced by Dyn *et al*; results for all other combinations were obtained but are not shown. Unless specifically shown the grids are oriented with $(0,0)$ in the bottom left hand corner.

Angle Between Normals (ABN)

The Angle Between Normals cost function seeks to measure the angle between the normals to the planes produced by the piecewise linear interpolant in R^3.

Let $\mathbf{n}^{(1)}$ and $\mathbf{n}^{(2)}$ be the normal vectors to the two planes P_1 and P_2 respectively which meet at an edge, i.e.

$$\mathbf{n}^{(i)} = \frac{1}{\sqrt{a_i^2 + b_i^2 + 1}} \begin{pmatrix} a_i \\ b_i \\ -1 \end{pmatrix}.$$

The cost function is the acute angle θ between these two vectors,

i.e. $s(f_T, e) = \theta = \cos^{-1} A$, where

$$A = \frac{a_1 a_2 + b_1 b_2 + 1}{\sqrt{a_1^2 + b_1^2 + 1}\ \sqrt{a_2^2 + b_2^2 + 1}}.$$

This criterion is the closest to the idea of steepest ascent, Lee (1982) and it should provide smoothness even near edges of the domain Ω. The smoothness of the representation can be seen in Figure 5.29 where the triangulation produced using the ABN cost function and the 2-norm for vector ordering for the 33 point data set is displayed. The triangulation comprises of many long, thin triangles

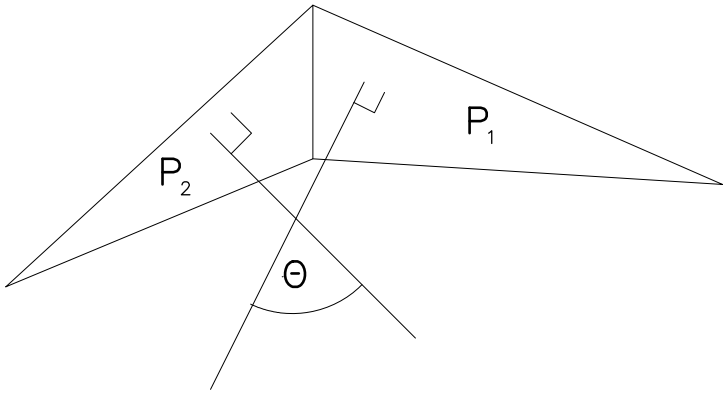

Figure 5.28: Angle θ as Found for Criterion ABN

along the ramp's face and these triangles have the effect of bunching the contours, which are straight (especially when compared to the Delaunay triangulation (see Figure 5.8)). The numerical errors show a 50% decrease in the L_2-error and an even greater decrease in the mean interpolation error.

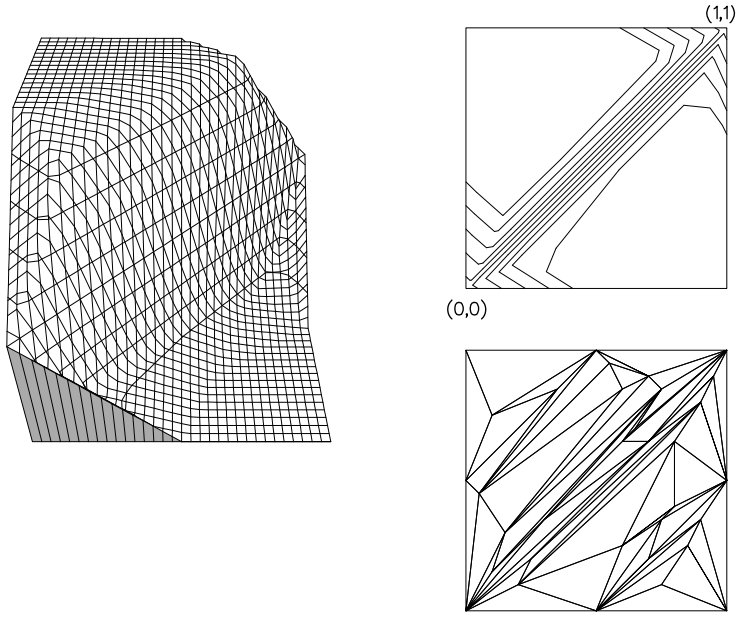

Figure 5.29: The ABN-2 Triangulation for SR1 (33 Points)

This smoothness can also be seen in the ABN-2 triangulation of the 100 data point set for Function SR1 (Figure 5.30). However, even though the contours are

straight and bunched together, the actual triangulation does not comprise of as many long, thin triangles as might be expected. This is due to the order in which the nodes are reconnected and the solution to this problem will be discussed later, in Section 5.5.2. Also, in this case it can be seen that even though there are more nodes on the boundary of the region the representation of the function on the boundary is not as good as might be wished, even though it is improved. This is borne out by the lack of change in the maximum interpolation error which almost certainly means that the error occurs on the boundary. The other numerical errors both decrease by about 50%.

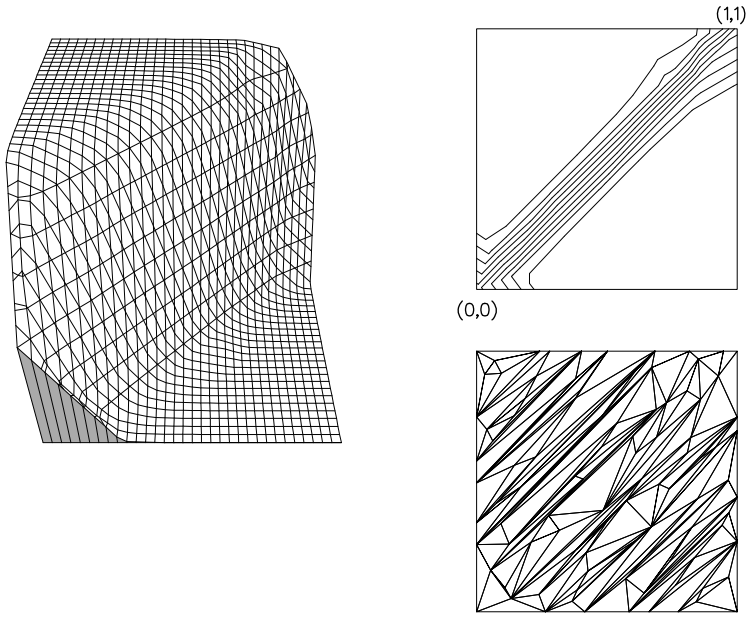

Figure 5.30: The ABN-2 Triangulation for SR1 (100 Points)

Figure 5.31, which shows the ABN-2 triangulation of 33 data points for Function DD1, shows that even with a small number of nodes it is possible to produce a better representation of certain features. The ramp is well represented, with triangles lying on its face, while the representation of the mountain is greatly improved over previous representations, see Figures 5.9, 5.20, 5.21 and 5.23. Figure

5.31 also shows the improved representation near the edges of the domain. The numerical results show that this is one of the best representations of DD1.

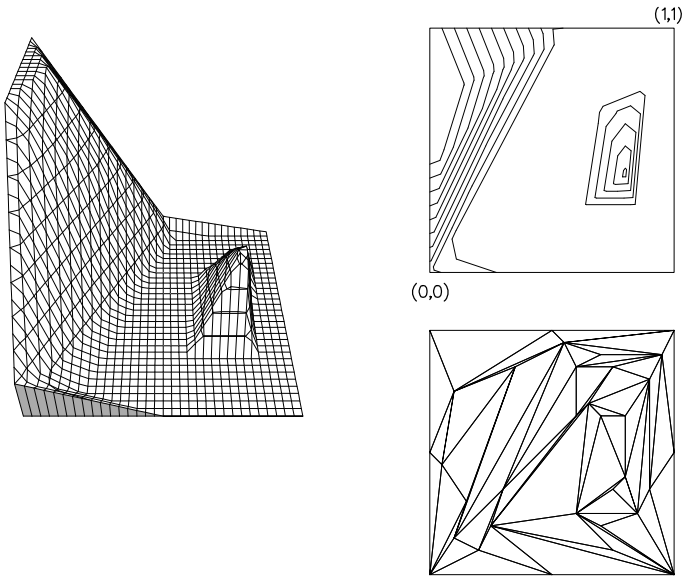

Figure 5.31: The ABN-2 Triangulation for DD1 (33 Points)

Another criterion of Dyn *et al* based on the concept of underlying smoothness is:

Jump in Normal Derivatives (JND)

This cost function is a measure of the jump in the normal derivatives of P_1 and P_2 across the common edge, e, i.e.

$$s(f_T, e) = |n_x(a_1 - a_2) + n_y(b_1 - b_2)|$$

where $\mathbf{n} = \begin{pmatrix} n_x \\ \\ n_y \end{pmatrix}$ is a unit vector in the perpendicular direction of the edge e,

$$\mathbf{n} = \frac{\mathbf{m}}{\|\mathbf{m}\|} \quad \text{where} \quad \mathbf{m} = v_k + (\lambda - 1)v_i - \lambda v_j,$$

$$\lambda = \frac{(v_j - v_i, v_k - v_i)}{(v_j - v_i, v_j - v_i)}.$$

This however does not seem to be related to the idea of steepest ascent and so for smooth functions the Angle Between Normals criterion is better as can be seen in Figures 5.32 and 5.33. Figure 5.32, when compared to Figure 5.29, shows that the different criteria can produce different grids for the same function, e.g. note the extra small triangles in Figure 5.32. The numerical errors show that there is only a slight difference between ABN-2 and JND-1 but the ABN-2 errors are smaller.

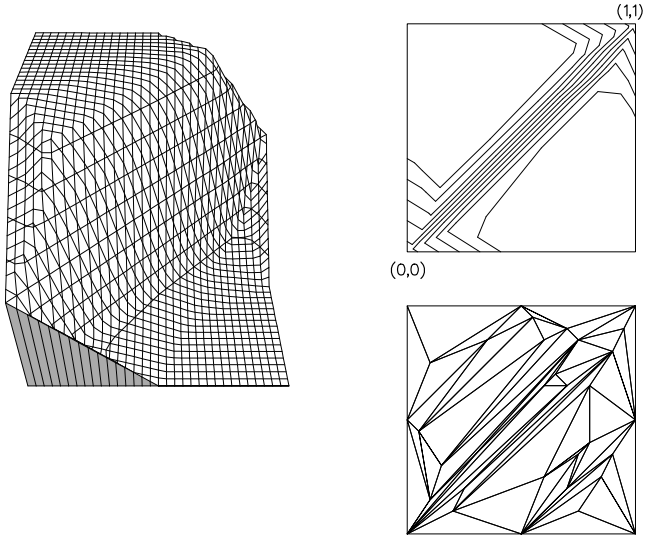

Figure 5.32: The JND-1 Triangulation for SR1 (33 Points)

Figure 5.33 again shows the difference in grids, when compared to Figure 5.31, but also shows that the representations can be very different in regions such as that containing the mountain.

This leads on to the next category of criteria which try to give some measure of how close the planes formed in R^3, by the interpolants on neighbouring triangles, are to being co-planar.

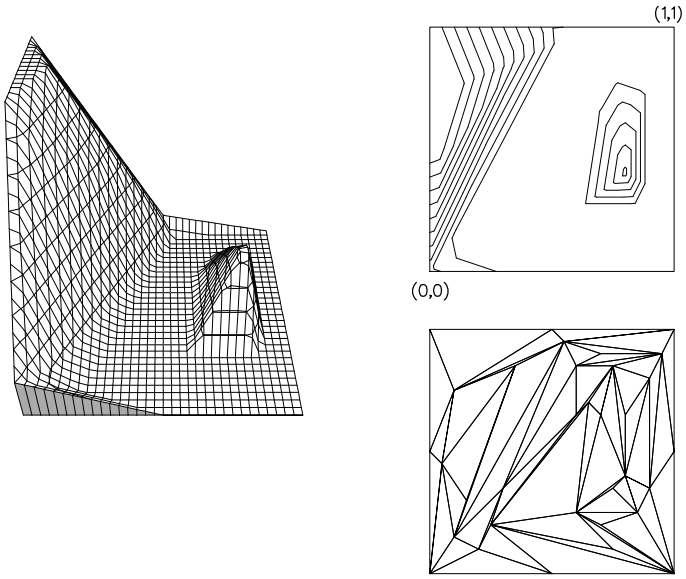

Figure 5.33: The JND-1 Triangulation for DD1 (33 Points)

5.3.4 Near Planar Criteria

These criteria, from Dyn, Rippa and Levin (1990), relate to cost functions defined on interior edges of the triangulation, and attempt to give some measure of how near to being planar two triangles, T_1 and T_2, with a common edge, e, are. Figure 5.34 gives a visual representation of how each of the near planar criteria are calculated.

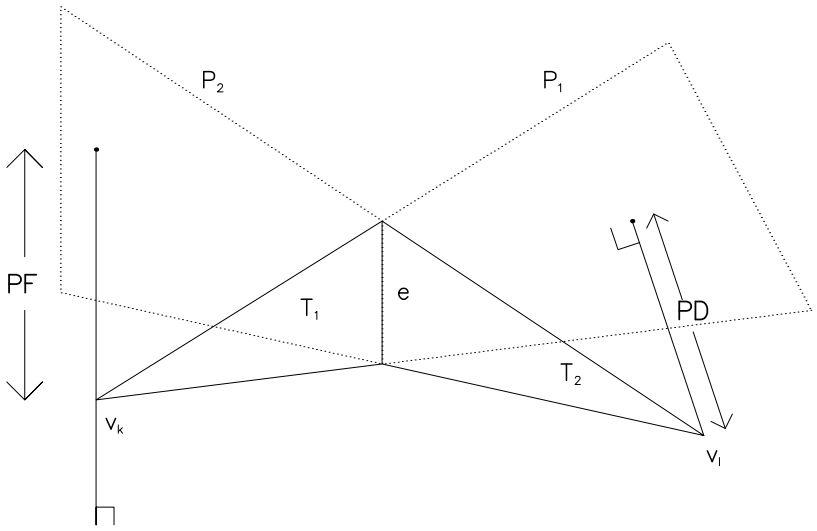

Figure 5.34: Distances as calculated by the near-planar criteria

Plane-fit (PF)

The plane-fit criterion measures the error between the linear interpolants P_1 and P_2 interpolated at the other vertices V_l, V_k respectively, in the quadrilateral Q and the actual function values F_l, F_k respectively.

$$s(f_T, e) = \|\mathbf{h}\|$$

where

$$\mathbf{h} = \begin{pmatrix} |P_1(x_l, y_l) - F_l| \\ \\ |P_2(x_k, y_k) - F_k| \end{pmatrix}.$$

Intuitively this gives a measure of how far from being planar the planes are and operates best on smooth functions which do not have large second derivatives.

The results for this criterion, Figures 5.35 to 5.37, reinforce these intuitive comments. Figure 5.35 shows that for a function with large second derivatives, the representation is poor, and the triangles do not follow the contours. The numerical results highlight this problem.

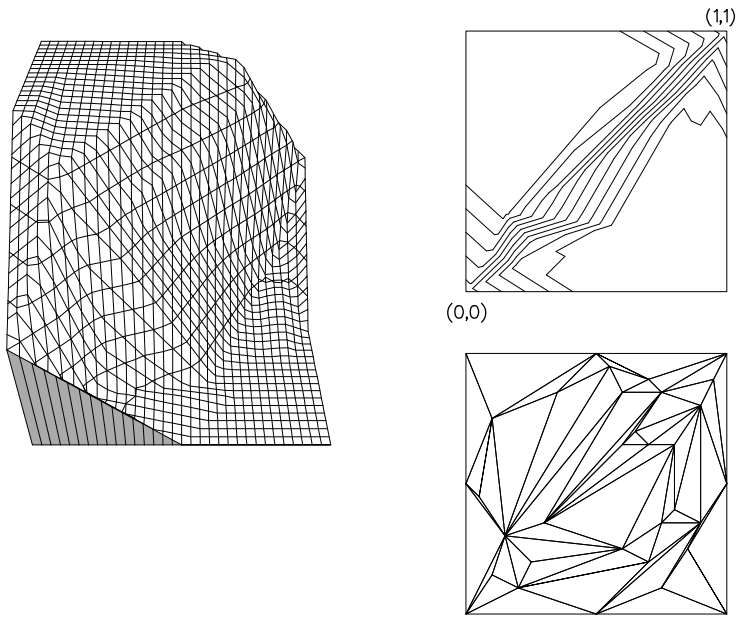

Figure 5.35: The PF-2 Triangulation for SR1 (33 Points)

Figure 5.36 shows that for Function DD1, which has derivative discontinuities in certain regions (the top and bottom of the ramp), the Plane-Fit criterion is unable to cope and smears out the sharp features. The numerical results show an improvement over Delaunay but not as much improvement as for other data dependent criteria.

Figure 5.37 shows that for a smooth function with small second derivatives, i.e. Function SH1, there is an improvement in the function's representation (compared to Figure 5.10).

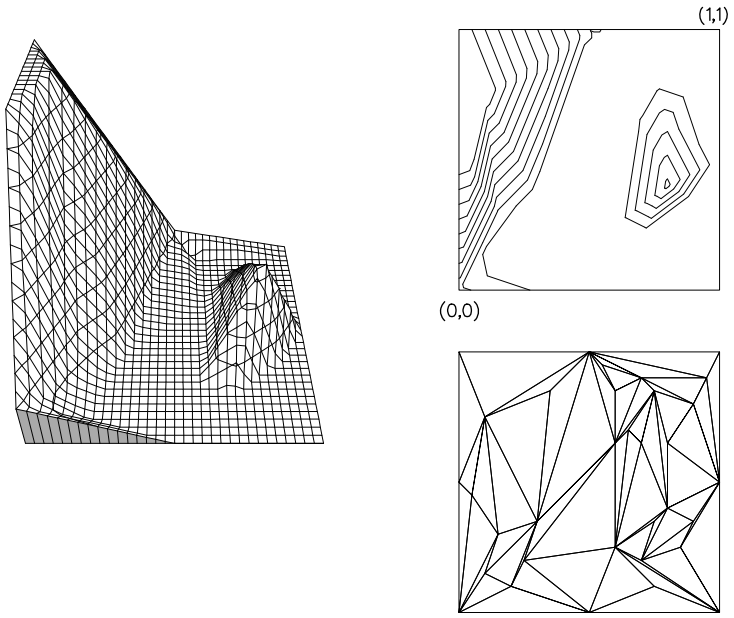

Figure 5.36: The PF-2 Triangulation for DD1 (33 Points)

The other near-planar criterion is outlined below and as can be seen from Figure 5.34, it is an alternative measure of how close to planar two neighbouring triangles are.

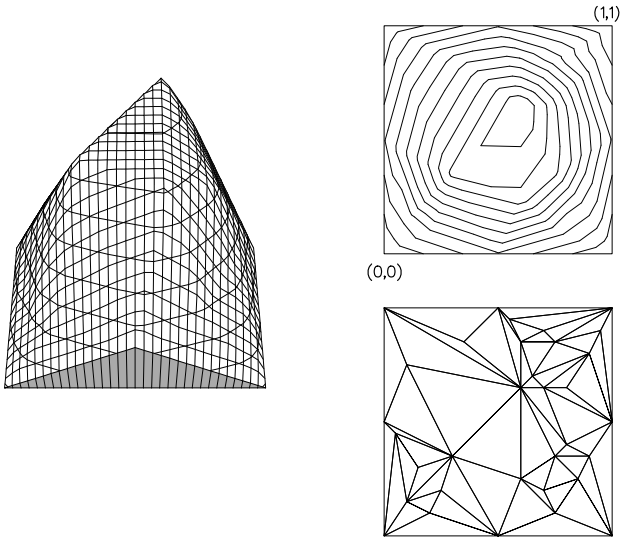

Figure 5.37: The PF-2 Triangulation for SH1 (33 Points)

Plane-Dist (PD)

The Plane-Dist criterion measures the perpendicular distance between the extended planes P_1 and P_2 and the points w_l and w_k respectively.

$$s(f_T, e) = \|\mathbf{h}\|$$

$$\mathbf{h} = \begin{pmatrix} Dist(P_1, F_l) \\ \\ Dist(P_2, F_k) \end{pmatrix}$$

where

$$Dist(P_m, F_n) = \frac{|P_m(x_n, y_n) - F_n|}{\sqrt{a_m^2 + b_m^2 + 1}}.$$

This has the same advantages and disadvantages as the Plane-Fit criterion as can be seen from Figures 5.38 and 5.39. Both figures shows that the Plane-Dist criterion produces different grids to the Plane-Fit criterion, by comparison with Figures 5.35 and 5.36.

In both near planar criteria, the norms used are those which correspond with the vector ordering norm, with the infinity norm corresponding to the lexicographic ordering.

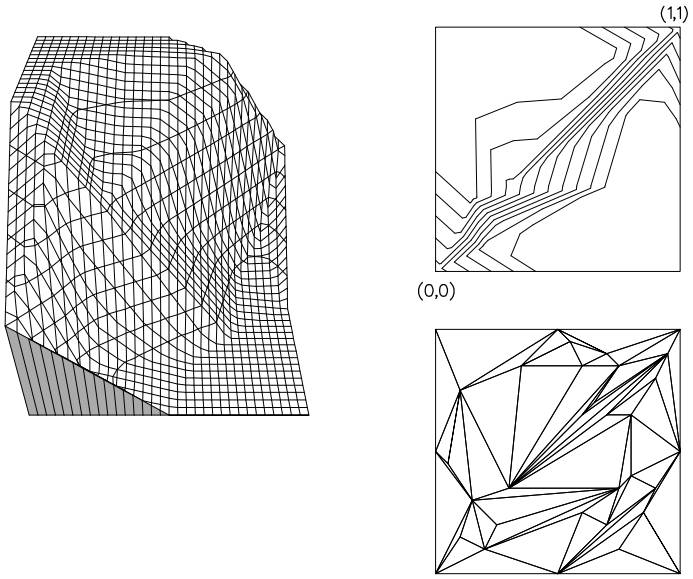

Figure 5.38: The PD-1 Triangulation for SR1 (33 Points)

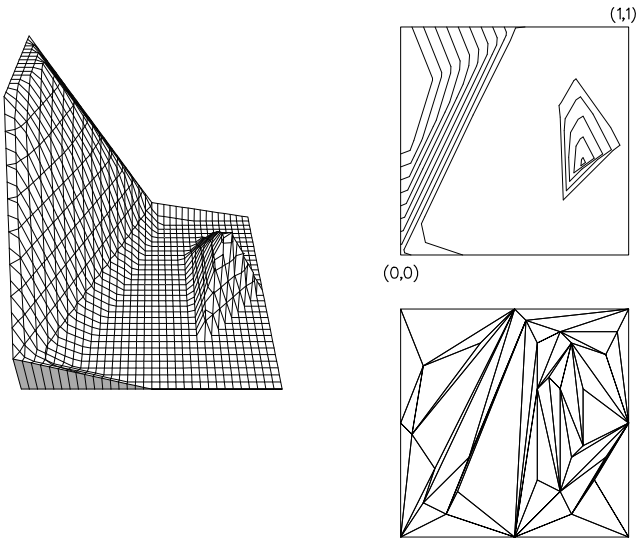

Figure 5.39: The PD-1 Triangulation for DD1 (33 Points)

The final criterion presented is that of equidistribution.

5.3.5 Equidistribution

This criterion seeks to minimise a cost function along the edges in the triangulation by swapping and was first proposed by Sweby (1987). The integral of the weight function, see below, is calculated using quadrature on a small number of

points. The underlying motivation has been outlined in Section 3.1 and is that, in 1-D, the behaviour of the underlying function u can be monitored by looking at the integral of some monitor function w of u over an interval $[a, b]$, see (3.1).

The monitor function, w, is some positive function of u or its derivatives which allows monitoring of its behaviour and can be used to position nodes to minimise errors.

As described in Section 3.1, Carey and Dinh (1985) derived an expression for the "optimal" weighting function, (3.2), which can be used to produce a cost function for nodal reconnection. Using (3.2), taking linear interpolants and the L_2-norm, we see that in 1-D the monitor function, w, is

$$w(x) = (u_{xx})^{2/5}$$

whilst in 2-D, the directional analogue is

$$w = (\cos^2\theta u_{xx} + 2\cos\theta\sin\theta u_{xy} + \sin^2\theta u_{yy})^{2/5}$$

where $$\tan\theta = \frac{y_j - y_i}{x_j - x_i}.$$

The 2-D cost function, S, is then

$$S(f_T, e) = \int_{v_i}^{v_j} w ds,$$

which is calculated on each edge in the triangulation and by using the local optimisation procedure the triangulation which minimises the sum over all the edges is found.

In the test cases, u_{xx}, u_{xy} and u_{yy} could all be evaluated exactly and so calculating $\int_{v_i}^{v_j} w\ ds$ was easily achieved using Gaussian quadrature.

This criterion works well for smooth functions as can be seen in Figure 5.40. The grid produced for Function SR1 contains long, thin triangles parallel to the

actual contours. The numerical errors show the large decrease in the L_2-error and mean interpolation for this representation.

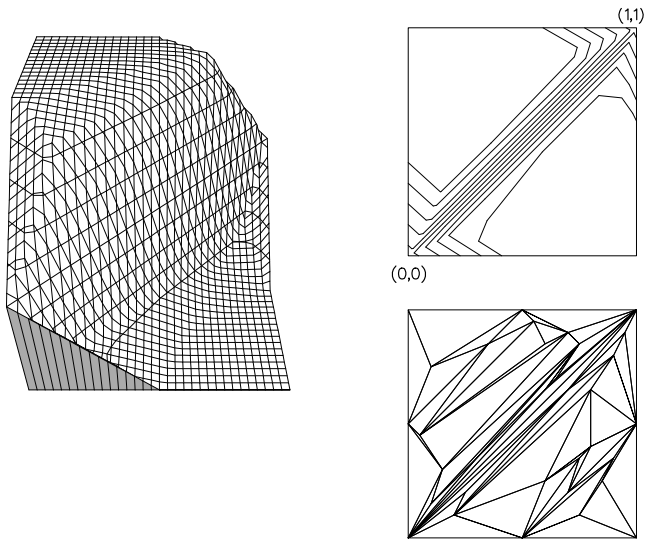

Figure 5.40: The Equidistribution Triangulation for SR1 (33 Points)

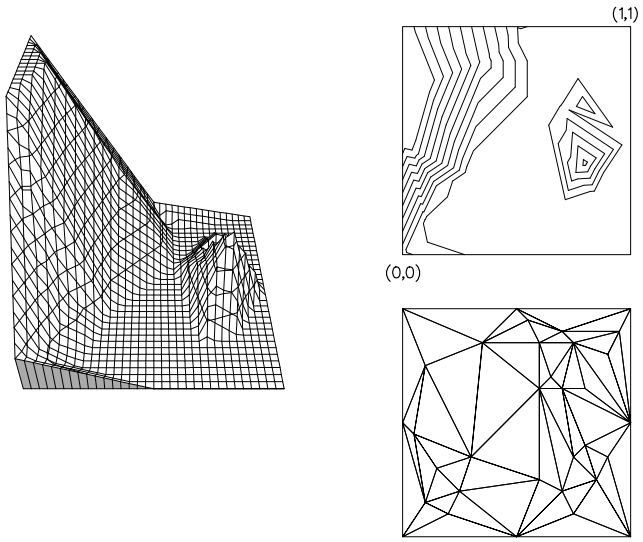

Figure 5.41: The Equidistribution Triangulation for DD1 (33 Points)

For Function DD1 the representation is not as good, since we again come across the problem of two connected nodes being on either side of the mountain. Another problem is that the whole of the region, except the mountain, has a second directional derivative of zero, so the connectivity depends entirely on whether

or not edges with equal cost are swapped or not. The numerical errors highlight this with the increase in the L_2-error.

5.4 Tables of Numerical Results

The tables of numerical results are now presented so that it is possible to see which triangulations have the smallest errors. The tables which follow show the numerical errors and the geometrical measure associated with the different representations already presented. The errors and measures are calculated as outlined in Section 4.3.

Table 5.1 shows the numerical errors for the triangulations of SR1 by the 33 point set in this chapter. In comparison with the Delaunay triangulation the following results stand out; the ABN-2 triangulation shows a significant decrease in the L_2-error and the mean interpolation error. Equidistribution works just as well as ABN-2 for this function; while other of the reconnection criteria improve the results, but not to such a degree as ABN-2 or equidistribution. Table 5.2 shows the geometric measures for SR1 with 33 data points. These show that the Delaunay triangulation has many nearly equiangular triangles, the mean skewness is small, and that the ABN-2 triangulation is comprised of many long, thin triangles, the mean aspect ratio is small.

Errors for SR1 (33 points)				
Method	L_2 – error x 10^{-2}	mean interpolation x 10^{-2}	max interpolation x 10^{-2}	Figure
Delaunay	2.37108	1.60364	7.51542	5.8
ABN-2	1.15505	0.61040	6.38080	5.29
MWT-2D	2.36341	1.60113	7.51542	
MWT-3D (ω=1)	2.35635	1.58545	7.51542	
MWT-3D (ω=10)	2.04891	1.38536	6.68127	5.22
MAX-MIN 3D	2.35838	1.58658	7.51542	5.19
JND-1	1.18125	0.62807	6.38080	5.32
PF-2	1.72252	1.05734	7.14728	5.35
PD-1	2.00543	1.28186	7.73203	5.38
Equidistribution	1.15780	0.61451	6.38080	5.40
An alternative ABN-2	1.60637	0.88416	6.66938	5.44
An alternative PD-1	1.15945	0.61802	6.38080	5.46
Simulated Annealing ABN-2, G1, C_1, f_2	1.15505	0.61040	6.38080	5.51
Simulated Annealing ABN-2, G1, C_1, f_3	1.1728	0.65087	6.38080	5.52

Table 5.1: Numerical errors for SR1 (33 points)

Geometric measures for SR1 (33 points)				
Method	Angles	Skewness	AR (Lo)	Figure
	min, max	mean, max	mean, min	
Delaunay	6.48, 153.4	28.0, 62.3	0.660, 0.188	5.8
ABN-2	0.58, 177.0	53.2, 78.0	0.301, 0.017	5.29
MWT-2D	2.73, 172.9	30.0, 75.2	0.654, 0.067	
MWT-3D (ω=1)	2.72, 172.9	30.4, 75.2	0.646, 0.067	
MWT-3D (ω=10)	2.73, 172.9	4.7, 75.2	0.591, 0.067	5.22
MAX-MIN 3D	6.48, 153.4	28.0, 62.3	0.659, 0.188	5.19
JND-1	0.89, 177.0	49.8, 78.0	0.356, 0.024	5.32
PF-2	2.73, 172.9	44.2, 75.2	0.440, 0.067	5.35
PD-1	1.85, 172.9	44.6, 75.2	0.438, 0.054	5.38
Equidistribution	0.58, 177.0	55.2, 78.0	0.298, 0.017	5.40
An alternative ABN-2	0.58, 177.0	50.2, 78.0	0.342, 0.017	5.44
An alternative PD-1	0.57, 177.9	52.3, 78.6	0.336, 0.016	5.46
Simulated Annealing ABN-2, G1, C_1, f_2	0.48, 177.9	54.0, 78.6	0.300, 0.014	5.51
Simulated Annealing ABN-2, G1, C_1, f_3	0.78, 177.0	48.6, 78.0	0.358, 0.023	5.52

Table 5.2: Geometrical measures for SR1 (33 points)

Table 5.3 shows the numerical errors for DD1 using the 33 point set. When compared to the results for the Delaunay triangulation, the ABN-2 triangulation gives an improvement in results as does the JND-1 triangulation. It can also be noted that some of the triangulations increase the numerical errors e.g. equidistribution and MAX-MIN 3D.

Table 5.4 shows the geometrical measures for DD1 using the 33 point data set. It is possible to see the introduction of long, thin triangles by many of the reconnection procedures.

Errors for DD1 (33 points)				
Method	L_2 – error x 10^{-1}	mean interpolation x 10^{-2}	max interpolation x 10^{-1}	Figure
Delaunay	1.20728	6.86570	6.78551	5.9
ABN-2	1.01215	4.31671	6.73873	5.31
MWT-2D	1.20603	6.89342	6.78551	
MWT-3D (ω=1)	1.22882	6.16923	6.78551	5.21
MWT-3D (ω=10)	1.11246	4.81425	6.78551	5.23
MAX-MIN 3D	1.32738	6.54446	8.81535	5.20
JND-1	1.02777	4.56247	6.73873	5.33
PF-2	1.10078	5.73231	6.73873	5.36
PD-1	1.04093	4.34577	6.78551	5.39
Equidistribution	1.23639	6.60126	6.78551	5.41
An alternative ABN-2	1.01215	4.31671	6.73873	5.45
An alternative PD-1	1.01012	4.26698	6.56733	5.47
An alternative ABN-2	1.01215	4.31671	6.73873	5.48
An alternative PD-1	1.04093	4.34577	6.78551	5.49
Simulated Annealing ABN-2, G1, C_1, f_2	1.00156	4.09598	6.73873	5.53

Table 5.3: Numerical errors for DD1 (33 points)

Geometric measures for DD1 (33 points)				
Method	Angles	Skewness	AR (Lo)	Figure
	min, max	mean, max	mean, min	
Delaunay	6.48, 153.4	28.0, 62.3	0.660, 0.188	5.9
ABN-2	0.33, 178.4	47.8, 79.0	0.382, 0.009	5.31
MWT-2D	2.73, 172.9	30.0, 75.2	0.654, 0.067	
MWT-3D (ω=1)	2.73, 172.9	35.8, 75.2	0.566, 0.067	5.21
MWT-3D (ω=10)	2.40, 173.4	41.2, 75.6	0.485, 0.060	5.23
MAX-MIN 3D	1.24, 176.6	34.8, 77.7	0.572, 0.031	5.20
JND-1	0.33, 178.4	46.8, 79.0	0.393, 0.009	5.33
PF-2	1.68, 172.9	42.7, 75.2	0.463, 0.047	5.36
PD-1	0.24, 179.2	47.4, 79.4	0.365, 0.006	5.39
Equidistribution	6.48, 153.4	30.8, 62.3	0.621, 0.188	5.41
An alternative ABN-2	0.33, 178.4	43.8, 79.0	0.423, 0.009	5.45
An alternative PD-1	1.36, 172.9	43.9, 75.2	0.428, 0.040	5.47
An alternative ABN-2	0.33, 178.4	47.4, 79.0	0.373, 0.009	5.48
An alternative PD-1	1.36, 171.9	46.4, 74.6	0.374, 0.040	5.49
Simulated Annealing				
ABN-2, G1, C_1, f_2	0.33, 178.4	46.4, 79.0	0.399, 0.009	5.53

Table 5.4: Geometric measures for DD1 (33 points)

Table 5.5 shows that for SH1, the reconnection criterion PF-2 improves the numerical errors when compared with those produced by Delaunay.

Table 5.6 shows the numerical errors and geometrical measures for SR1 when the larger data sets are used.

Table 5.7 shows the numerical errors and geometrical measures associated with the Delaunay triangulations of 81 and 100 points.

Table 5.8 shows the improvement that the D'Azevado and Simpson triangulations can make in the the case of P2.

Errors for SH1				
Method	L_2 – error x 10^{-2}	mean interpolation x 10^{-2}	max interpolation x 10^{-2}	Figure
Delaunay (33 points)	2.42616	1.46847	7.67019	5.10
PF-2 (33 points)	2.35113	1.40353	7.47289	5.37
Geometric measures for SH1				
Method	Angles	Skewness	AR (Lo)	Figure
	min, max	mean, max	mean, min	
Delaunay (33 points)	6.48, 153.4	28.0, 62.3	0.660, 0.188	5.10
PF-2 (33 points)	4.76, 164.5	34.2, 68.7	0.587, 0.137	5.37

Table 5.5: Errors and measures for SH1 (33 points)

Errors for SR1				
Method	L_2 – error x 10^{-3}	mean interpolation x 10^{-3}	max interpolation x 10^{-2}	Figure
Delaunay (81 points)	4.32505	2.25362	1.16866	5.13
Delaunay (100 points)	7.57001	4.11860	3.83141	5.15
ABN-2 (100 points)	4.60538	2.03663	3.83141	5.30
Geometric measures for SR1				
Method	Angles	Skewness	AR (Lo)	Figure
	min, max	mean, max	mean, min	
Delaunay (81 points)	45.0, 90.0	20.0, 20.0	0.866, 0.866	5.13
Delaunay (100 points)	10.2, 152.5	23.3, 61.7	0.740, 0.275	5.15
ABN-2 (100 points)	0.04, 179.7	54.0, 79.8	0.304, 0.001	5.30

Table 5.6: Errors and measures for SR1

Errors for DD1				
Method	L_2 – error x 10^{-2}	mean interpolation x 10^{-2}	max interpolation x 10^{-1}	Figure
Delaunay (81 points)	4.44908	1.62930	3.08566	5.14
Delaunay (100 points)	5.87095	2.40301	4.54757	5.16
Geometric measures for DD1				
Method	Angles	Skewness	AR (Lo)	Figure
	min, max	mean, max	mean, min	
Delaunay (81 points)	45.0, 90.0	20.0, 20.0	0.866, 0.866	5.14
Delaunay (100 points)	10.2, 152.5	23.3, 61.7	0.740, 0.275	5.16

Table 5.7: Errors and measures for DD1

Errors for P2				
Method	$L_2 - \text{error}$	mean interpolation	max interpolation	Figure
Delaunay (33 points)	2.00851	1.45028	6.25000	5.12
D'Azevado (33 points)	1.21345	0.74188	6.25000	5.25
Delaunay (100 points)	0.34984	0.28374	1.80730	
D'Azevado (100 points)	0.19832	0.12857	1.80730	5.26

Geometric measures for P2				
Method	Angles	Skewness	AR (Lo)	Figure
	min, max	mean, max	mean, min	
Delaunay (33 points)	6.48, 153.4	28.0, 62.3	0.660, 0.188	5.12
D'Azevado (33 points)	3.01, 170.5	44.6, 73.7	0.429, 0.091	5.25
Delaunay (100 points)	10.2, 152.5	23.3, 61.7	0.740, 0.275	
D'Azevado (100 points)	1.48, 175.5	54.3, 77.0	0.299, 0.043	5.26

Table 5.8: Errors and measures for P2

All the criteria for data dependent nodal reconnection studied have now been presented. However before considering nodal movement criteria it is beneficial to study the implementation details which are presented in the next section together with the differences which can occur due to different implementations; a numerical technique for finding the global minima, namely Simulated Annealing, is also detailed.

5.5 Implementation Details

In this section the different strategies which were used to implement each criterion are examined. The effect of different Local Optimisation Procedures, differences caused by searching the lists of triangles in a different order and the use of a strict or non-strict inequality in the swapping criterion are investigated, as well as the use of Simulated Annealing.

5.5.1 Local Optimisation Procedures (LOP)

A local optimisation procedure (LOP) is used to check the total cost associated with each triangulation and hence choose the triangulation with the lowest cost. However this involves calculating the cost associated with all the edges in the triangulation.

A truly "Local" LOP would check just the costs across the one edge $e = \overline{v_i v_j}$, which is to be swapped, and $e' = \overline{v_k v_l}$, the possible new edge, (see Figure 5.42) and then keep the edge which has the smallest cost.

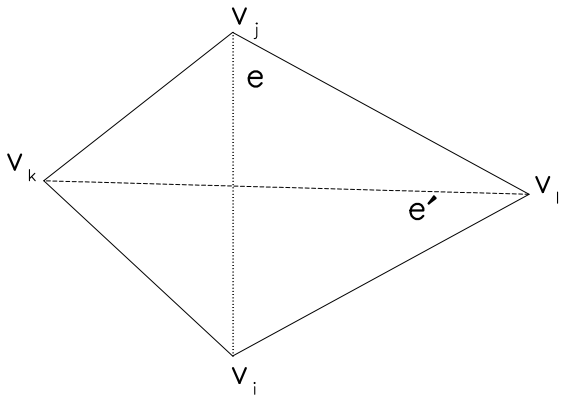

Figure 5.42: Possible triangulations of a convex quadrilateral

However in the "global" LOP used by Dyn, Rippa and Levin (1990), a cost vector **S** is set up so that the effect of changing from e to e' is monitored across

not only the internal diagonals but also the other four edges, $\overline{v_i v_l}$, $\overline{v_i v_k}$, $\overline{v_l v_j}$ and $\overline{v_j v_k}$, (see Figure 5.42). The resulting vectors $\mathbf{S}$ and $\mathbf{S}'$, each with five elements, are then compared using the specified vector Norm.

It should be noted that MWT, MWT-3D, MAX-MIN angle and equidistribution are already "Local" LOP's, and that PF and PD rely on a specified norm since the cost function across an edge is a vector.

The difference in triangulations which result can be seen in Figure 5.43, when it is compared with Figure 5.29. Both the triangulations are produced using the same data dependent criterion and norm, but Figure 5.29 was produced using the "global" approach of Dyn *et al.*, while Figure 5.43 was produced using the "local" approach. We can see that the "global" LOP produces better results than the "local" LOP in this case and also in all the other cases tested. This shows that it is better to consider the effect on the whole triangulation of reconnecting two nodes rather than the effect on the common edge of the triangles.

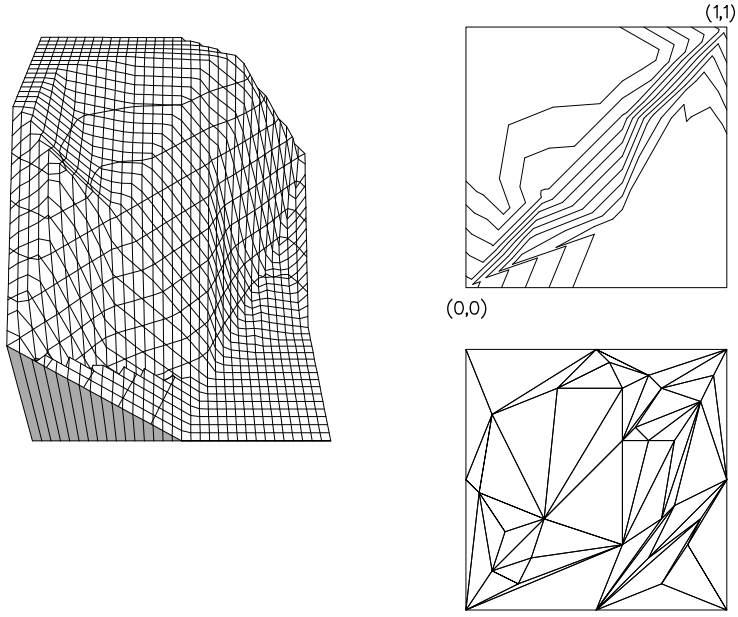

Figure 5.43: The ABN-2 Triangulation of SR1 using the "local" LOP (33 points)

5.5.2 Differences in Triangle Searching

In searching for diagonals to swap, each triangle in the list is taken in turn and neighbouring elements sought (i.e. those with a common edge). On finding a neighbouring element the two are considered as forming a quadrilateral in which, if appropriate, the diagonal can be swapped. If no swapping occurs, another neighbour is sought, or if all neighbours have been considered without swapping occurring then the next triangle is taken as a base. However, if swapping does occur there are two options; either to take one of the newly formed triangles as a base and continue searching for its neighbours or to jump to the next triangle in the list.

The consequences of the different swapping routines can be that totally different grids are produced with almost identical costs. This can be seen by comparing the figures which follow with the corresponding figures earlier in the chapter. All the previous figures have been produced by the procedure whereby one of the newly formed triangles is used as a base. The figures which follow in this subsection use the procedure whereby an unchanged triangle is used as a base after nodal reconnection.

Figure 5.44 shows the effect of the difference in triangle searching for the ABN-2 triangulation of Function SR1. In comparison with Figure 5.29 it can be noted that the representation of the ramp is poorer and that there are not as many long, thin triangles. Figure 5.45 shows that the identical representation can be achieved even with different triangulations (compare with Figure 5.31).

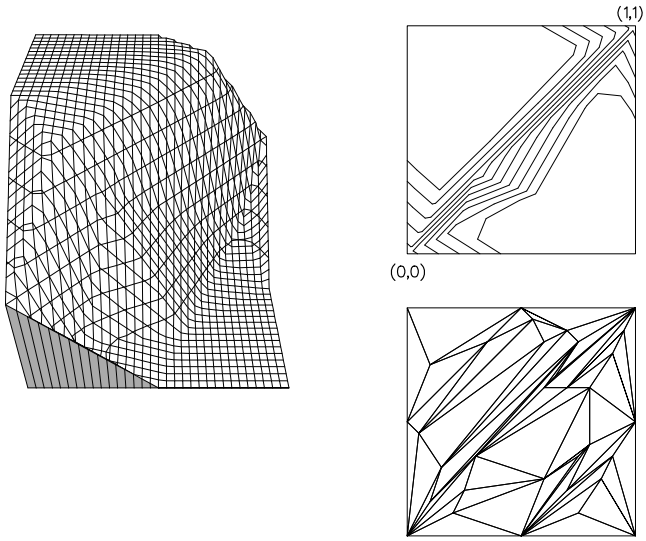

Figure 5.44: An alternative ABN-2 Triangulation of SR1 (33 points)

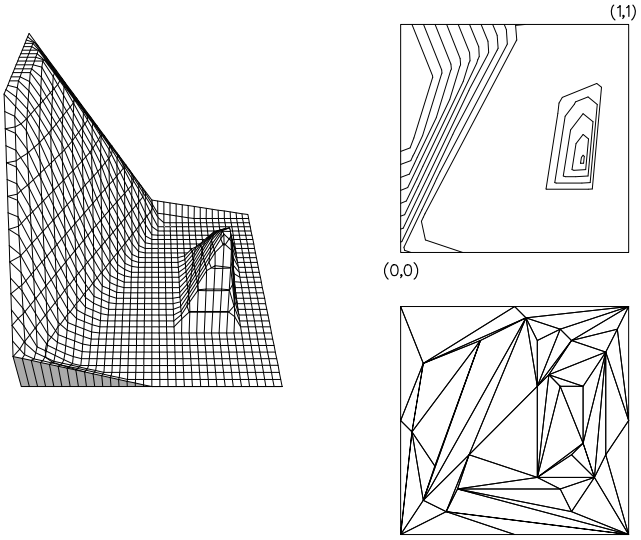

Figure 5.45: An alternative ABN-2 Triangulation of DD1 (33 points)

However, the different triangle searching procedure does not necessarily lead to worse representations as can be seen by looking at Figures 5.46 and 5.47. Figure 5.46 shows the PD-1 triangulation of SR1 and when this is compared to Figure 5.38 the improvement in representation is clear to see as are the increased number of long, thin triangles. Figure 5.47 also shows an improvement in representation when compared to Figure 5.39, the PD-1 triangulation of DD1. The numerical errors associated with both figures show a marked improvement over

those associated with Figures 5.38 and 5.39.

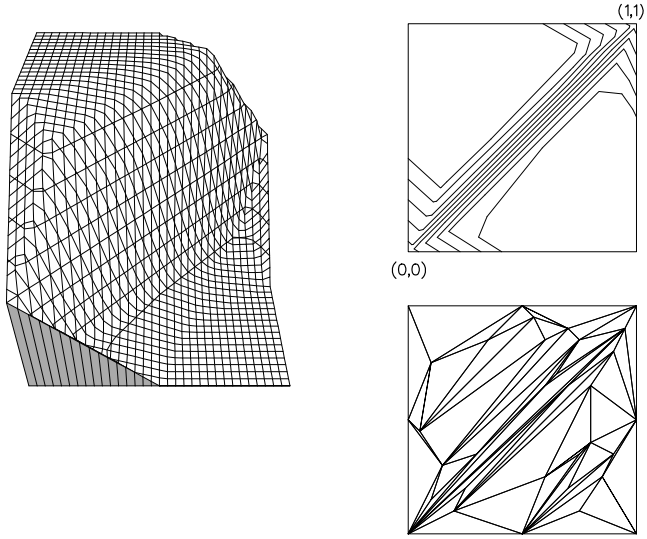

Figure 5.46: An alternative PD-1 Triangulation of SR1 (33 points)

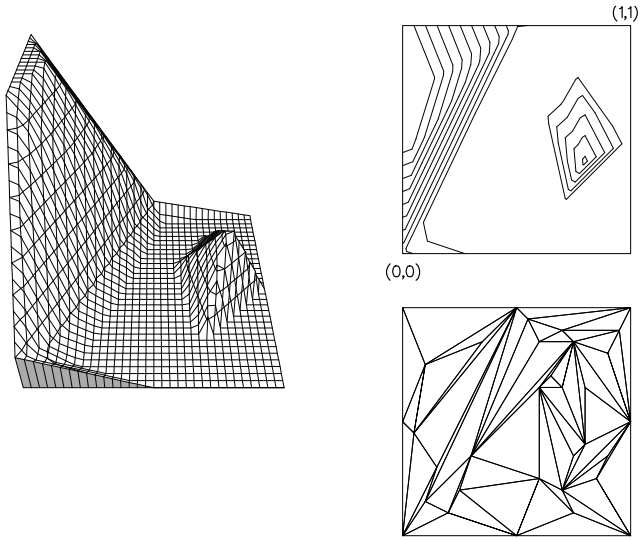

Figure 5.47: An alternative PD-1 Triangulation of DD1 (33 points)

Having described the procedures which can be used to search for the next edge to check, the methods which can be used to decide if nodes should be reconnected are outlined in the next section.

5.5.3 Swapping Strategies

As described in Section 5.2, an ordering on the set of triangulations can be defined such that $T \leq T'$ if $\mathbf{S} \leq \mathbf{S}'$ according to some vector ordering. From this definition it is possible to produce two strategies to change the internal diagonals, e and e' with cost vectors $\mathbf{S}$ and $\mathbf{S}'$ respectively. A sweep has occurred when all the members of the list of triangles have been considered.

The strategies are :-

1. Change from edge e to edge e' if $\mathbf{S}$ is less than or equal to $\mathbf{S}'$ in the ordering, storing the number of swaps on each sweep and the number of exact equalities on each sweep. When all swaps in a sweep are due to equality then change the strategy and change only if $\mathbf{S}$ is strictly less than $\mathbf{S}'$, and keep this strategy until there are no swaps in a sweep when the process terminates.

2. Change from edge e to edge e' if $\mathbf{S}$ is strictly less than $\mathbf{S}'$ in the ordering and continue until there are no swaps in a sweep. Then do one sweep where edge e is changed to edge e' if $\mathbf{S}$ is less than or equal to $\mathbf{S}'$. Then continue with the original strict inequality until there are no more swaps on one sweep.

In many cases, these different swapping strategies produce almost identical triangulations, but in some cases the triangulations are very different. Two such triangulations which can be seen to be different are shown in Figures 5.48 and 5.49 (compare to Figures 5.31 and 5.39 respectively).

The different implementation strategies that can produce various triangula-

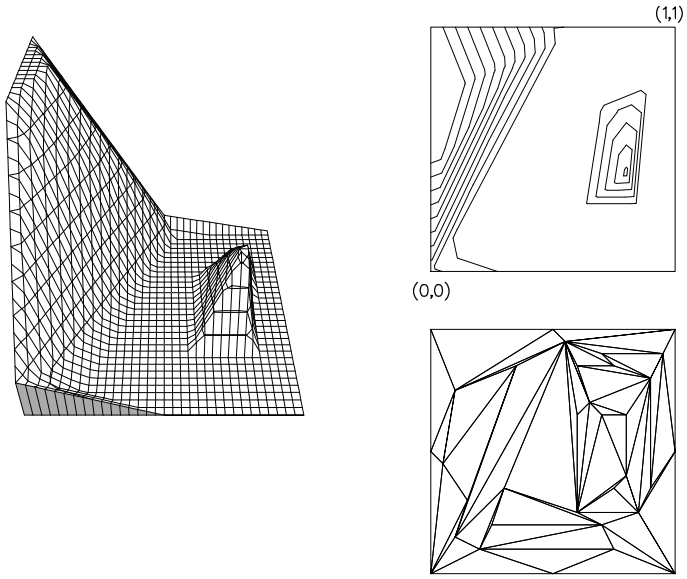

Figure 5.48: An alternative ABN-2 Triangulation of DD1 (33 points)

tions have now been outlined and in the next section a numerical technique which could produce global minima rather than different locally optimal minima is described.

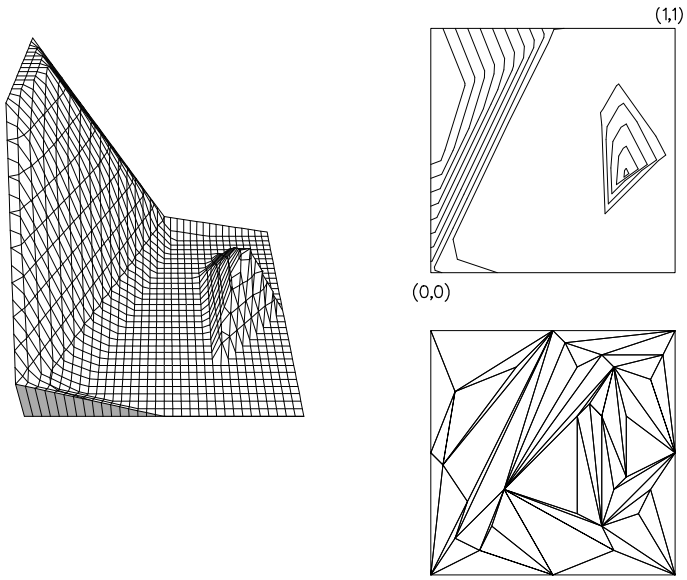

Figure 5.49: An alternative PD-1 Triangulation of DD1 (33 points)

5.6 Simulated Annealing

This is a technique which is used to try and find global extremum in numerical optimisation techniques rather than local extrema (see Figure 5.50).

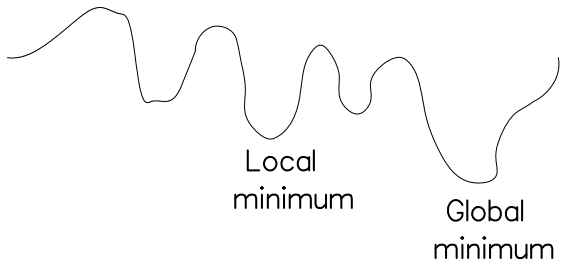

Figure 5.50: Function with global and local extrema

It is based on the idea, from statistical mechanics, that from a highly disordered state, i.e. at a high temperature, it is easier to produce the global extremum. A restatement of this is :- by taking, with a decreasing probability, a step which makes the solution worse, it is possible that a better solution could be achieved than that produced by taking improving steps only.

In statistical mechanics the number of changes of state is directly related to the temperature at which the system is at, i.e. it is more likely to become more chaotic, "worse", at a higher temperature. In the numerical work this is tempered by the fact that not all non-improving swaps will be made, so a random variable is used to introduce the idea that only some swaps will be made. Account is also taken of the length of time, (number of swaps), at the same temperature, as in actuality the system cools, so some criterion to simulate this change in temperature is introduced.

These components for the numerical method are known as :-

1. The annealing schedule, $g(t)$. This is the pseudo-temperature variable which is originally set to a high value so all possible states can be reached from an initial state. It then decreases so more ordered states occur until the pseudo-temperature variable decreases to zero and only "better" states are reached.

2. A randomness function, $f(r)$. This gives a probability that a worse state will be reached at any time.

3. Changing criterion, C_n. This changing criterion is set so that changes in the annealing schedule occur regularly, so that the method does not become computationally expensive.

All three of the above criteria can be user defined.

The procedure is as follows for edge swapping in a triangulation :-

1. Set up an initial triangulation, T_o, with associated cost, I_o.

2. Change the triangulation by swapping a diagonal to produce T_n with associated new cost I_n if

$$I_n < I_o + f(r) \cdot g(t)$$

then accept swap.

3. If swap accepted, set $I_o = I_n$,$T_o = T_n$. Return to stage 2.

4. If C_n is satisfied then move to the next element in the annealing schedule. Reset C_n.

C_n is usually chosen so that when a certain number of edges have been checked or a certain number of swaps made, then the next element in the annealing

schedule is used.

Thus C_n is chosen so that the number of edges checked is A_0, or the number of actual swaps made is A_1, before progressing to the next element in the annealing schedule. i.e.

$$C_1 = (A_0 = 2 \times T, A_1 = T)$$

$$C_2 = (A_0 = T, A_1 = N)$$

where T is the number of triangles in a triangulation and N is the number of interior points.

The choices of randomness function, $f(r)$, with r $\in$ [0,1] are

$$\begin{aligned} f_1(r) &= r \\ f_2(r) &= -log(r) \\ f_3(r) &= 1. \end{aligned}$$

$f_3(r)$ is proposed by Dueck and Scheuer (1990).

The annealing schedule is user, and problem, dependent. The two schedules used had 20 different values :-

$$G1 = (.015, .014, \ldots, .001, 0, 0, 0, 0, 0)$$

$$G2 = (.05, .03, .02, .016, .012, .01, .009, \ldots, .001, 0, 0, 0, 0, 0).$$

A few illustrative results now follow. It was found that using C_2 did not give as good results as C_1, so it seems that the longer before changing the annealing schedule the better. In most cases there was not a large difference between results previously generated and those got by simulated annealing although in a number of cases better representations were found by using Simulated Annealing.

Figure 5.51, which was produced using G1, C_1 and f_2, shows that there can be differences in triangulations, (compare to Figure 5.29), without a significant change in representation.

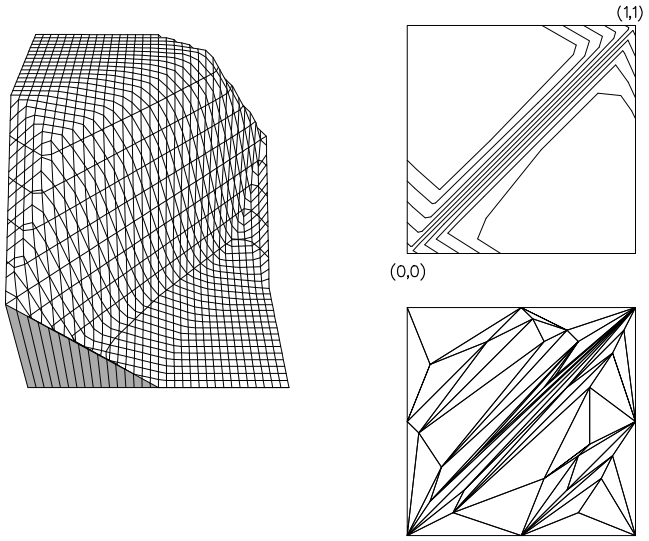

Figure 5.51: A Simulated Annealing ABN-2 Triangulation of SR1 (33 points)

Figure 5.52, which was produced using G1, C_1 and f_3, shows that a better triangulation is not guaranteed, but the result will be very close to any previous result.

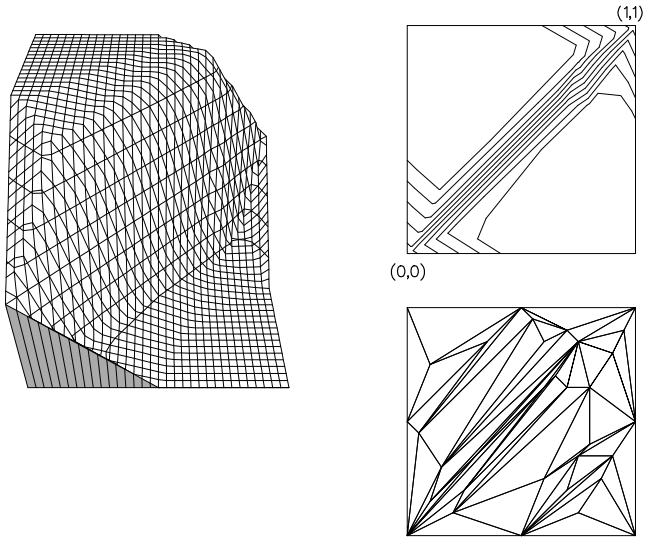

Figure 5.52: A Simulated Annealing ABN-2 Triangulation of SR1 (33 points)

Figure 5.53, which was produced using G1, C_1 and f_2, shows an improved

triangulation which has been produced using the ABN-2 criterion on DD1. The numerical results show this to be the best triangulation for DD1 using the 33 point data set.

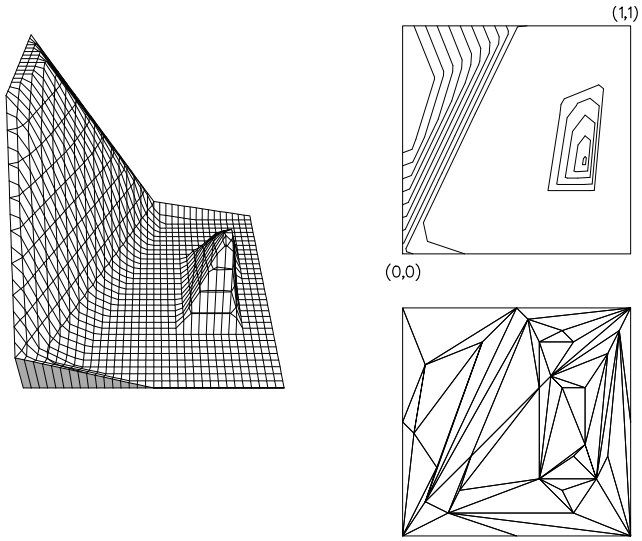

Figure 5.53: A Simulated Annealing ABN-2 Triangulation of DD1 (33 points)

In the next section we outline how geometrical constraints can be introduced into the nodal reconnection procedure.

5.7 Geometrical Constraints

If the triangular grids produced are likely to be used in conjunction with a numerical solution method, then it may be necessary to take into account any constraints that the numerical method introduces. These constraints may be required in order that the system does not become ill conditioned or the accuracy deteriorate. These constraints are usually geometrical in nature and might be that the angles should not be too small, for the conditioning of the solution method, or that the area of any of the triangles should not be too small, again for the conditioning of the numerical method.

These constraints can easily be introduced into a nodal reconnection procedure in the following manner. When two neighbouring triangles, with a common edge, are being considered for a nodal reconnection it is possible to insert a check to ensure that neither of the two newly generated triangles has an area less than the specified tolerance and that none of the angles in the two triangles is below the specified minimum angle.

Usually the choice of these specified tolerances would be dependent on the numerical method being employed, however for illustrative purposes, the tolerances are chosen so that the difference between the geometrically constrained triangulations and the unconstrained triangulations is apparent.

Figure 5.54 shows the effect of using a tolerance of 0.005 for the area and 5° for the minimum angle with the ABN-2 reconnection criterion. The triangulation produced can be compared with that in Figure 5.29 as can the numerical errors and the geometrical measures which are shown in Table 5.9. It is possible to see that the numerical errors are greater than those for the unconstrained triangulation.

Figure 5.55 shows the effect of using a tolerance of 0.005 for the area and 3° for the minimum angle with the ABN-2 reconnection criterion. The triangulation produced can be compared with those in Figure 5.29 and Figure 5.54 as can the numerical errors and the geometrical measures which are shown in Table 5.9. Again the numerical errors are greater than those for the unconstrained triangulation.

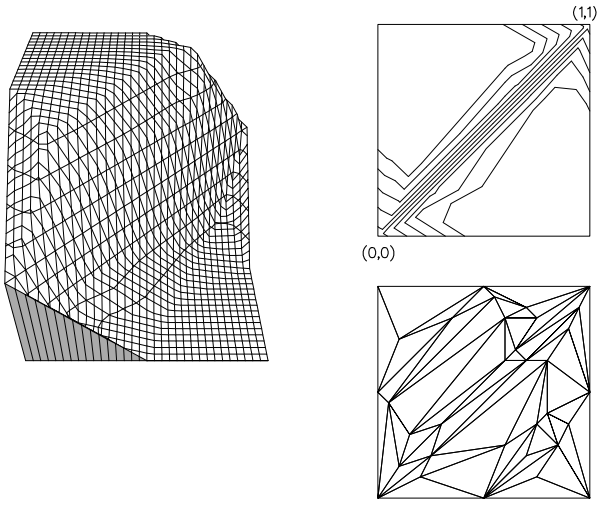

Figure 5.54: An ABN-2 Triangulation of SR1 with geometric constraints (33 points)

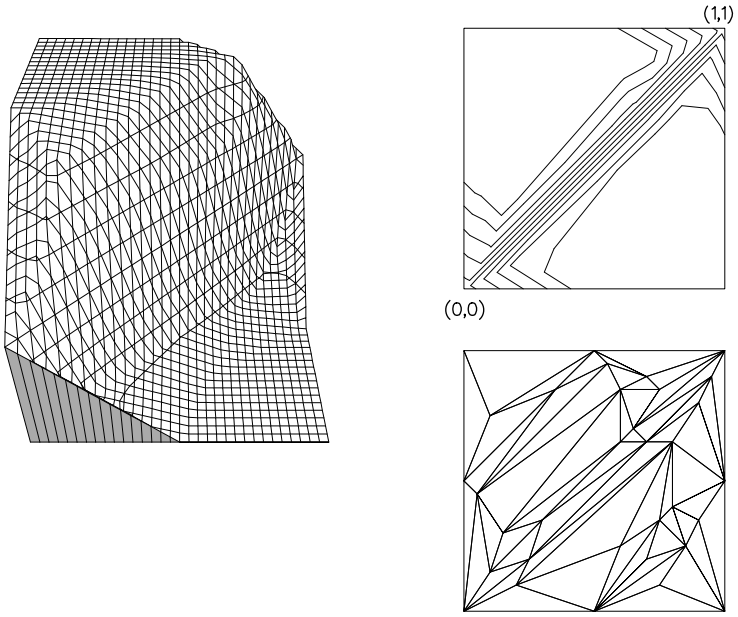

Figure 5.55: An ABN-2 Triangulation of SR1 with geometric constraints (33 points)

Figure 5.56 shows the effect of using a tolerance of 0.005 for the area and 5° for the minimum angle with the ABN-2 reconnection criterion. The triangulation produced can be compared with that in Figure 5.31 as can the numerical errors and the geometrical measures which are shown in Table 5.9. The numerical errors

show a slight worsening over those produced by the unconstrained procedure.

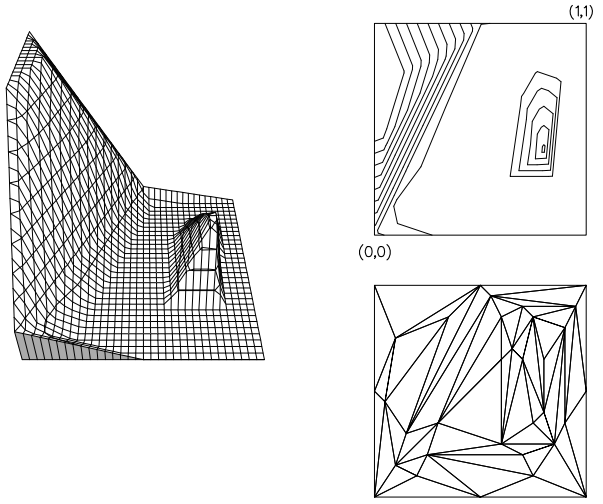

Figure 5.56: An ABN-2 Triangulation of DD1 with geometric constraints (33 points)

Figure 5.57 shows the effect of using a tolerance of 0.002 for the area and 3° for the minimum angle with the ABN-2 reconnection criterion. The triangulation produced can be compared with that in Figure 5.30 as can the numerical errors and the geometrical measures which are shown in Table 5.9. It is possible to

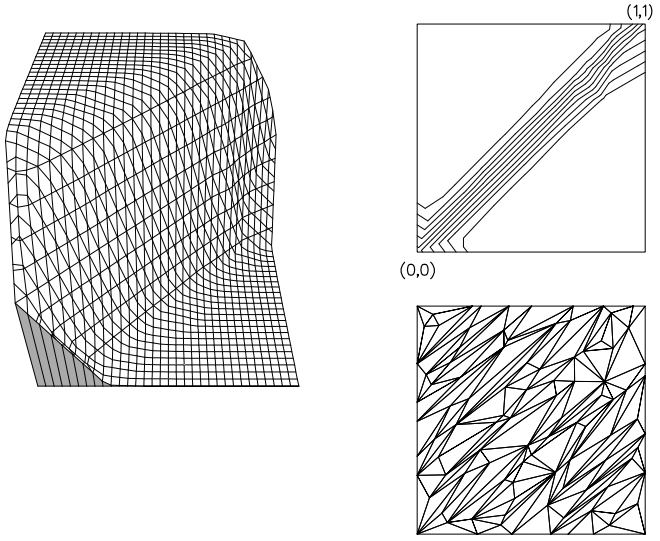

Figure 5.57: An ABN-2 Triangulation of SR1 with geometric constraints (100 points)

see from these results that geometric constraints can be added into the nodal reconnection procedure and that consequently the data representation changes.

This ends the chapter on grid generation by node reconnection and in the next chapter grid generation by nodal movement is described.

Errors for constrained triangulations				
Method	L_2 – error x 10^{-2}	mean interpolation x 10^{-2}	max interpolation x 10^{-2}	Figure
ABN-2 (SR1) (33 points)	1.1550	0.6104	6.3808	5.29
As above with constraints	1.2695	0.7416	6.3808	5.54
As above with constraints	1.2055	0.6821	6.3808	5.55
ABN-2 (SR1) (100 points)	0.4605	2.0366	3.8314	5.30
As above with constraints	0.4838	2.2129	3.8314	5.57
ABN-2 (DD1) (33 points)	10.121	43.167	67.387	5.31
As above with constraints	10.194	43.434	67.855	5.56
Geometric measures for constrained triangulations				
Method	Angles	Skewness	AR (Lo)	Figure
	min, max	mean, max	mean, min	
ABN-2 (SR1) (33 points)	0.58, 177.0	53.2, 78.0	0.301, 0.017	5.29
As above with constraints	5.71, 156.8	42.1, 64.5	0.458, 0.169	5.54
As above with constraints	3.22, 156.8	42.6, 64.5	0.452, 0.095	5.55
ABN-2 (SR1) (100 points)	0.04, 179.7	54.0, 79.8	0.304, 0.001	5.30
As above with constraints	3.47, 169.7	44.6, 73.2	0.436, 0.100	5.57
ABN-2 (DD1) (33 points)	0.33, 178.4	47.8, 79.0	0.382, 0.009	5.31
As above with constraints	5.04, 166.0	40.5, 70.6	0.461, 0.140	5.56

Table 5.9: Errors and measures for constrained triangulations

Chapter 6

Nodal Movement Criteria

In the previous chapter the effect on the representation of a function of reconnecting the nodes of a triangulation, whilst keeping their position fixed, was investigated. A complementary strategy, which is now investigated, is that of keeping the nodal connectivity fixed whilst changing the positions of the nodes of the triangulation. In this chapter various techniques which can be used to implement nodal movement in a triangulation with a fixed connectivity are investigated. The nodal repositioning by these techniques is based on criteria dependent on the properties of the underlying data function to be represented, the aim being to produce a set of data points, with a prescribed connection, which accurately represents the underlying data, (e.g. initial data), when interpolated on the grid.

As mentioned at the start of Chapter 5 it is possible that geometric constraints may have to be taken into account when generating the nodal positions. The possible constraints are the same as those discussed in Section 5.7, the introduction of small angles and small triangles. It is possible to put these into the nodal movement criterion as a check to see if the triangles generated are acceptable.

Given an arbitrary initial grid there are many different methods which might be used to redistribute the nodes towards an "ideal" grid with fixed connectivity. Some methods involve smoothing, e.g. when solving a differential equation, using the equation variables to smooth the grid, Palmerio *et al.* (1990). Others methods strive to approximately equally distribute some measure of interpolation error over all triangles. Here two such strategies which can be used to produce nodal redistribution are outlined. The first is a movement criterion based on treating edges in a triangulation as springs, with spring constants dependent on properties of the underlying function to be interpolated on the grid, see e.g. Catherall (1988), Lohner *et al.* (1986), Eisemann *et al.* (1987). The second procedure used is a movement criterion based on approximately equalising the error of interpolation on each triangle in the triangulation, based on error estimates derived from work by Nadler (1985). This is akin to work done by Sewell (1972).

Both methods involve a local iterative technique, in which local patches of triangles are set in equilibrium according to the corresponding criterion. This process is then repeated over all such patches until global equilibrium is achieved. The structure of this approach can be regarded as similar to a Gauss-Seidel iterative process.

The details of the two different techniques are presented together with some representative results, followed by details of their implementation and the presentation of graphical results to demonstrate the difference on nodal position that the manner of implementation can make. After the main body of results, tables of the numerical errors in function representation achieved by the various techniques are presented.

6.1 Spring Analogy

For this technique the element edges are regarded as being comprised of "springs". The object is to place all of the "springs" in equilibrium; to this end local patches of triangles are considered, each patch being placed in equilibrium. A global iteration over all such patches then achieves the desired result.

Each node, v_i, is surrounded by a patch of triangles, P formed by the p_i surrounding nodes, $v_{i,j},\ j = 1, \ldots, p_i$, with node v_i being connected to node $v_{i,j}$ by edge e_j. See Figure 6.1.

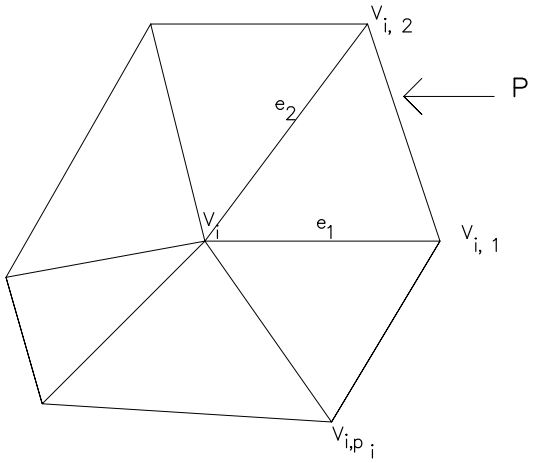

Figure 6.1: Patch surrounding a central node

For this movement criterion, edge e_j has associated with it a spring "constant", k_j, which is dependent on properties of the underlying function. A local equilibrium position for node v_i is found by solving the local spring system which is generated on the patch P.

It was originally envisaged that the "springs" would be given a specified unextended length, (the required minimum inter-nodal spacing), and the equation for the tension, T_j, in the spring, k_j, with this assumption is

$$T_j = \frac{k_j \times ext}{l} \tag{6.1}$$

where

- l is the specified original length.
- ext is the current nodal separation less the specified original length, l.

However, as can be seen in Figure 6.2, where the interior nodes of the Delaunay triangulation shown in Figure 5.5 were redistributed for Function DD1 using $l = 0.01$ and $k_j = 1 + (u_{ss})^{2/5}$, the results produced in some cases were abysmal, with nodes actually moving outside the unit square. This movement of nodes outside the unit square is due to nodes moving very close to each other, i.e. within l of each other. These nodes that are in close proximity then produce compressed springs which try to move the nodes far away and if the spring constants form the right pattern then the central node can be moved out of the patch, and in extreme cases out of the region. Thus it was decided, following Thacker *et al.* (1980) and Palmerio *et al.* (1990), to assume that all springs had negligible length when unstretched.

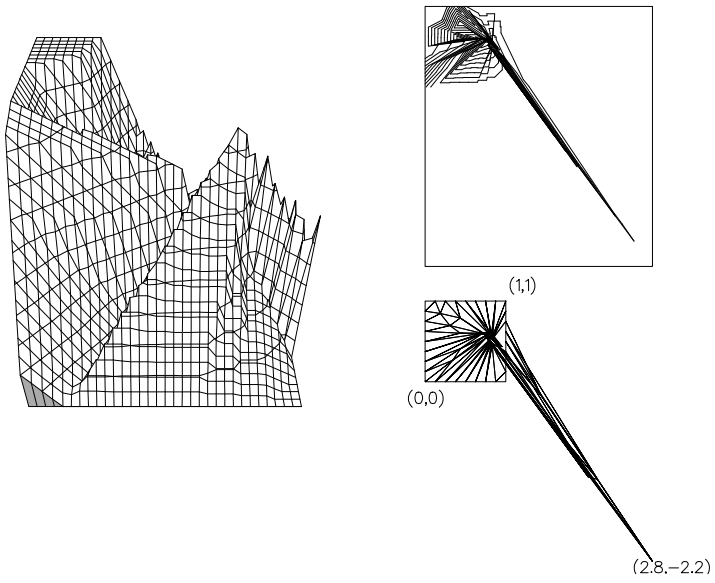

Figure 6.2: Interior Nodes Redistributed using (6.1) for DD1 (81 points)

The result of this is that a pseudo-tension, which does not involve the original length of the spring, is defined as

$$T_j = k_j \times ext. \tag{6.2}$$

Hence the pseudo-tension, T_j, for edge e_j is

$$T_j = k_j \sqrt{(x_i - x_{i,j})^2 + (y_i - y_{i,j})^2} \tag{6.3}$$

which can be separated into x and y components to give

$$T_{j_x} = k_j (x_i - x_{i,j}) \tag{6.4}$$

$$T_{j_y} = k_j (y_i - y_{i,j}). \tag{6.5}$$

Resolving the pseudo-tensions over the whole patch, then gives

$$\sum_{j=1}^{p_i} T_{j_x} = 0 \tag{6.6}$$

which can be used, to find the new position $(\tilde{x}_i, \tilde{y}_i)$ of node v_i, viz

$$\tilde{x}_i = \frac{\sum_{j=1}^{p_i} k_j x_{i,j}}{\sum_{j=1}^{p_i} k_j} \tag{6.7}$$

and similarly for $\tilde{y}_i$. This is similar to the technique for finding the centre of mass of a body with weights k_j at positions $v_{i,j}$.

It is now necessary to decide on which property or properties of the underlying function the spring constant, k_j, is to be based. The original choice was

$$k_j = (u_{ss})^{2/5} \tag{6.8}$$

where u_{ss} is the directional second derivative along the element edge, calculated at the mid-point of edge e_j. This is a two-dimensional analogue of the equidistribution theory of Carey and Dinh, (1985), (see Section 3.1).

However, with k_j defined as in (6.8) this procedure produces excessive clumping of nodes in regions of high curvature and poor representations of smooth regions. Figure 6.3 shows that for Function SR1 which has high curvature near the ramp the nodes cluster there. (In all the remaining figures in this section, (Figures 6.3 to 6.11), a Delaunay triangulation was taken as the original triangulation and only the interior nodes are repositioned).

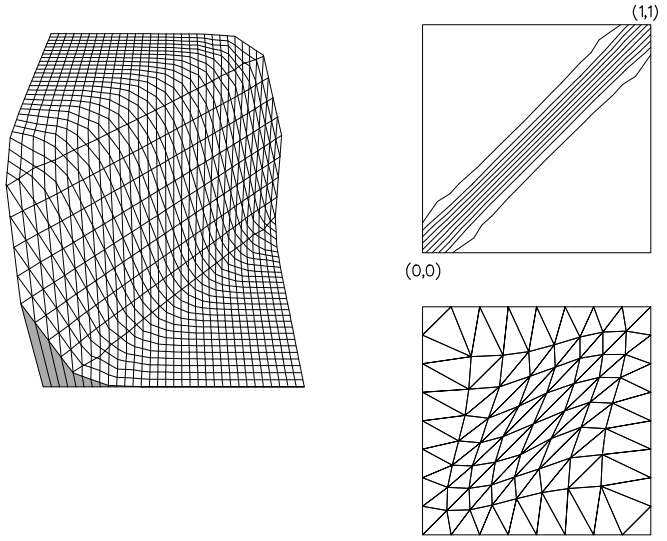

Figure 6.3: Interior Nodes Redistributed using (6.7) and (6.8) for SR1 (81 points)

Although k_j as in (6.8) performs well for Function SR1, for Function DD1 this choice of k_j does not produce acceptable results as there are some points where all the connecting edges except one have spring constants of zero which results in the central node moving to a coincident position with an adjoining node. If either of these coincident connected nodes are moved later in the procedure then the routine fails.

Such shortcomings can however be rectified by choosing

$$k_j = 1 + \alpha \frac{(u_{ss})^{2/5}}{[(u_{ss})^{2/5}]_{max}} \tag{6.9}$$

where $[(u_{ss})^{2/5}]_{max}$ is the maximum value of $(u_{ss})^{2/5}$ over all the edges and α is a parameter chosen to control the weighting given to the equidistribution measure, usually with a value of 10 or 20. Note that for $k_j = 1$ equal spacing (Laplacian smoothing) would result, thus the addition to k_j of unity gives additional control of nodal separation and helps place the nodes in smooth regions where u_{ss} is small.

The effect of Laplacian smoothing, $k_j = 1$, can be seen in Figures 6.4 and 6.5 which show the triangulations and nodal positions which result from Laplacian smoothing of the Delaunay triangulations of 33 and 100 data points respectively (Figures 5.4 and 5.5). The Delaunay triangulation of 81 data points (Figure 5.6)

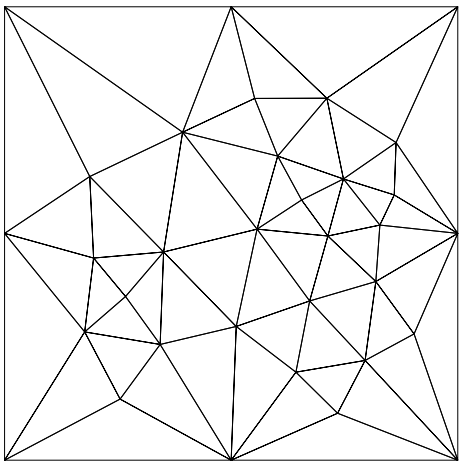

Figure 6.4: The Laplacian smoothed Delaunay Triangulation of 33 data points

is unchanged under Laplacian smoothing due to the regularity of triangles in the grid.

Comparing Figure 6.6, which shows the redistributed nodes for Function SR1 produced by choosing $\alpha = 10$ in (6.9), with Figure 6.3, we can see that such a change in spring constant can produce little effect in some instances. However Figure 6.7 shows the triangulation which can result for Function DD1, again when

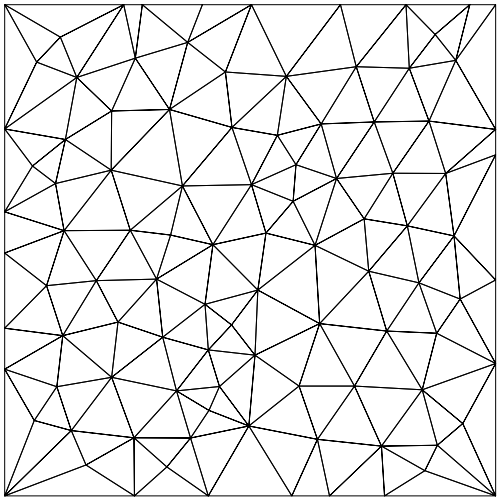

Figure 6.5: The Laplacian smoothed Delaunay Triangulation of 100 data points

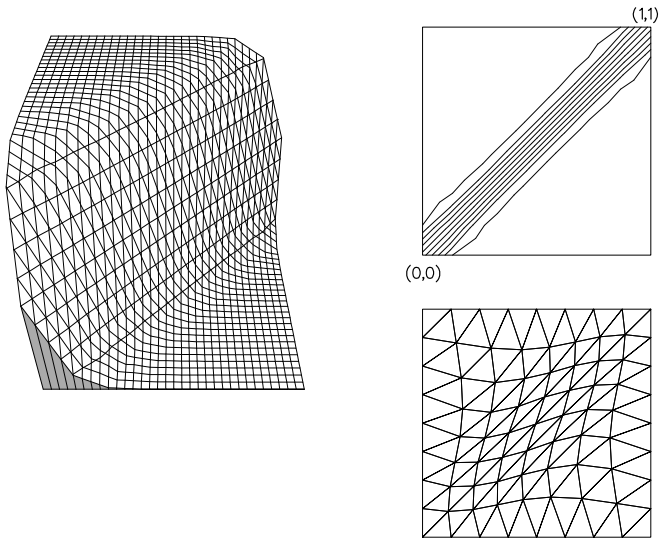

Figure 6.6: Interior Nodes Redistributed using $\alpha = 10$ in (6.9) for SR1 (81 points)

$\alpha = 10$ in (6.9), when the addition of unity halts the introduction of coincident nodes as detailed above. It can be noted that although nodes are now positioned in the "mountain", the ramp has had little influence on the positioning of the nodes. The poor numerical results for this triangulation are due to the poor representation of the ramp and the excessive clumping of nodes near the centre of the mountain, which highlight the need for an alternative choice of spring constant.

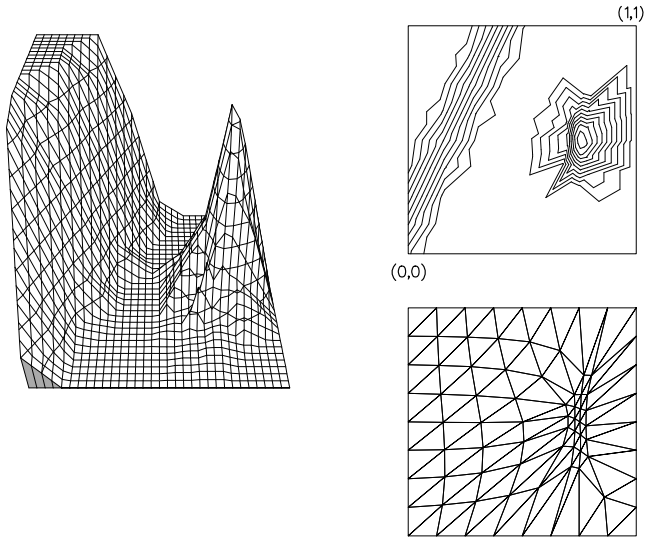

Figure 6.7: Interior Nodes Redistributed using $\alpha = 10$ in (6.9) for DD1 (81 points)

To alleviate such insensitivity to smooth functions an alternative choice for the spring constant was used, based on the first derivative of the function to be represented i.e.

$$k_j = 1 + \beta \frac{|u_s|}{|u_s|_{max}} \tag{6.10}$$

where β is chosen to give a weighting to the gradient (usually 1, 2 or 5). (6.10) is based on the 1-dimensional expression for arc-length and tends to produce movement of nodes to positions of high gradient, with the unit constant again giving a control on nodal separation. The coefficient k_j is calculated for the edge, e_j, joining (x_a,y_a) to (x_b,y_b) by approximating u_s as

$$u_s = \frac{F(x_a, y_a) - F(x_b, y_b)}{\sqrt{(x_a - x_b)^2 + (y_a - y_b)^2}}. \tag{6.11}$$

The main motivation for this is to produce better representation in regions where $u_{ss} \approx 0$ but the function is not smooth (see Figure 6.8, for example). In Figure 6.8, with $\beta = 5$ in (6.10), the nodes can be seen to have remained in the region of the ramp while also congregating slightly in the vicinity of the mountain. The

numerical results show that the triangulation decreases the maximum interpolation error while increasing the mean interpolation error and leaves the L_2-error almost unchanged.

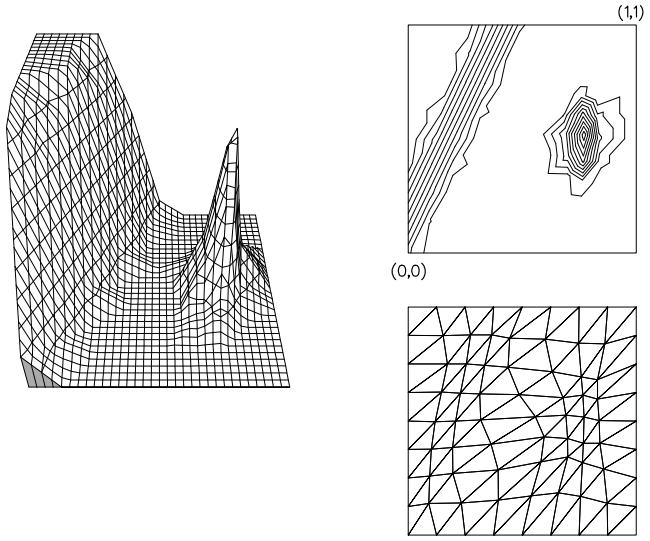

Figure 6.8: Interior Nodes Redistributed using $\beta = 5$ in (6.10) for DD1 (81 points)

However (6.10) does not perform as well as (6.9) in representing rapidly changing functions, (see Figure 6.9). In Figure 6.9, again with $\beta = 5$ in (6.10), the points move to the region of maximum gradient, which is not the region of greatest change of function where the points are required. The numerical results show an increase in all errors compared with those associated with Figures 6.3 and 6.6.

As a result of the difference in representation achieved by using (6.9) and (6.11), a combination of the above weights is used, that is;

$$k_j = 1 + \alpha \frac{(u_{ss})^{2/5}}{[(u_{ss})^{2/5}]_{max}} + \beta \frac{|u_s|}{|u_s|_{max}}. \tag{6.12}$$

The coefficients α and β can then be chosen to reflect the aspect of the function which it is desired to well represent.

Experimentation showed that in order to get acceptable results for all the test

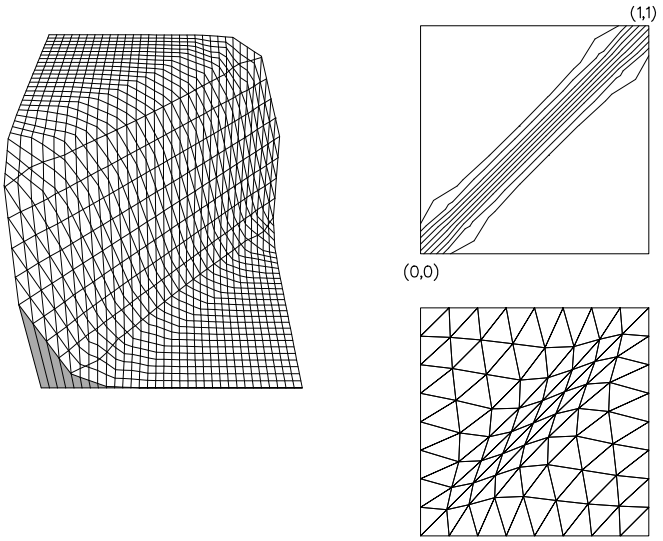

Figure 6.9: Interior Nodes Redistributed using $\beta = 5$ in (6.10) for SR1 (81 points)

functions without utilising any prior knowledge of the function to be represented, values of $\alpha \approx 8 - 10$ and $\beta \approx 2 - 3$ are adequate. The justification for these values is that the equidistribution theory states that the second derivative term is more important and so has a larger weighting, while, if needed, the weighting of the first derivative term must outweigh the contribution of the unity term.

Figures 6.10 and 6.11 show that these values of α and β in (6.12) can result in better representations of the functions. They will not necessarily be the best choice for all functions, as different features would desire different weightings of each component. Figure 6.10 shows the results of choosing $\alpha = 10$ and $\beta = 3$ with Function SR1. The numerical errors show that although it is an improvement over Delaunay, the L_2-, and mean interpolation, errors are larger than those produced by the grid in Figure 6.3.

Figure 6.11 shows the results of choosing $\alpha = 10$ and $\beta = 3$ with Function DD1. In this grid there are nodes near the ramp due to the influence of the gradient weighting but there is clumping of the nodes in the mountain region

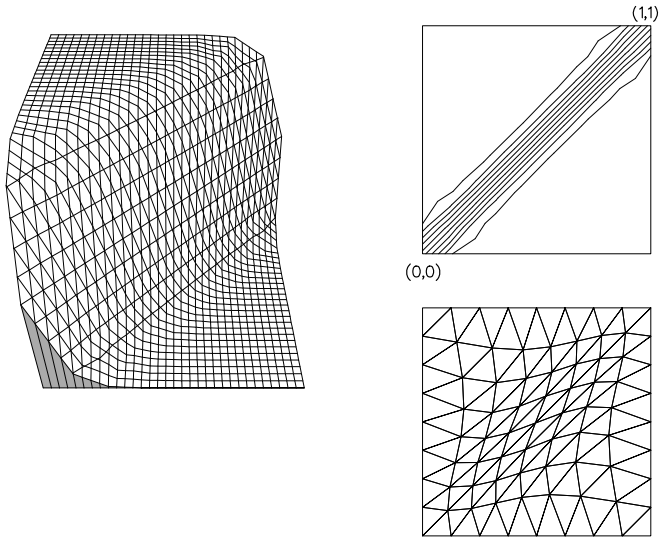

Figure 6.10: Interior Nodes Redistributed using (6.12) for SR1 (81 points)

due to the curvature weighting. The numerical results show small errors when compared to those of the representation given by the grid in Figure 6.7.

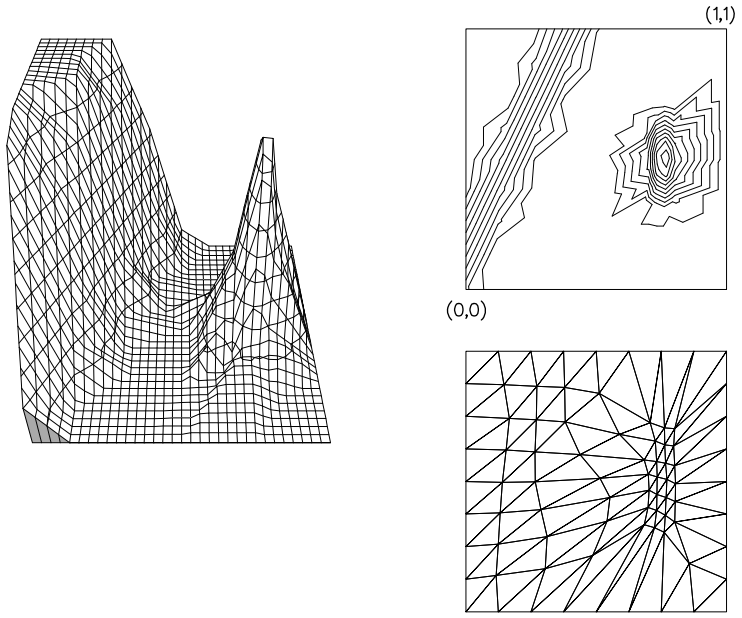

Figure 6.11: Interior Nodes Redistributed using (6.12) for DD1 (81 points)

These results show that the choice of spring constant will vary wildly from function to function, with some requiring more curvature weighting, while others require more gradient weighting. The general values given produce adequate

results in most cases but not necessarily the optimal results.

In the next section the alternative procedure for repositioning the nodes is outlined.

6.2 Error Equalisation

The second movement criterion considered is a global interpolation error equalisation criterion, again treated iteratively in a local manner. This technique is based on expressions for the error of least squares fits on triangles, (Nadler (1985)), which have been modified to produce error estimates for linear interpolants. It is possible to use these expressions to attempt to equalise the error on all triangles by relocating the central node in a patch of triangles.

In this section the following symbols are used :-

P is the number of points connected to the central node of a patch.

E is the error on a triangle.

$A(T_i)$ is the area of triangle T_i, and

$A_o(T_i)$ is the optimal area of triangle T_i.

The formulation is as follows :-

$$E^2 = W_i \approx C_i(\det H) \times [A(T_i)]^3 \times |\det H| \tag{6.13}$$

where

$$H = \begin{pmatrix} u_{xx} & u_{xy} \\ u_{yx} & u_{yy} \end{pmatrix} \tag{6.14}$$

is the Hessian at the centroid of T_i, and C_i is a constant depending on the sign of det H.

Since the error, E, is to be equalised on all triangles, it is acceptable to work with E^2 rather than E.

In order to get the same E^2, say W_{opt}, on each triangle in the patch we use

$$\sum_{i=1}^{P} W_i = P \times W_{opt}, \tag{6.15}$$

then for each triangle in the patch

$$\frac{W_i}{W_{opt}} = \frac{C_i \times [A(T_i)]^3 \times |\det H|}{C_i \times [A_o(T_i)]^3 \times |\det H|} \tag{6.16}$$

where $A_o(T_i)$ is the area which triangle, T_i, should have in order to give an E^2 of W_{opt}, i.e.

$$A_o(T_i) = \left[\frac{W_{opt}}{W_i}\right]^{1/3} \times A(T_i). \tag{6.17}$$

This is valid since the determinant of the Hessian at the centre of the new triangle can be considered as being the same as at the centre of the old triangle, so the determinants, det H, and the constants, C_i, cancel. The determinant is treated as being the same or the problem would become non-linear, i.e. the only information about the new triangle is a line on which one of the vertices can be positioned, which means that the centre of the new triangle is on a parallel line and thus the Hessian cannot be calculated.

A further condition which can be imposed is that the area of a patch should be conserved so that the central vertex remains within the patch.

$$\sum_{i=1}^{P} A(T_i) = \sum_{i=1}^{P} \overline{A_o}(T_i) \tag{6.18}$$

where

$$\overline{A_o}(T_i) = \lambda A_o(T_i) \tag{6.19}$$

and

$$\lambda = \frac{\sum_{i=1}^{P} A(T_i)}{\sum_{i=1}^{P} A_o(T_i)}. \tag{6.20}$$

Using (6.17), (6.18) and (6.20) an expansion factor,

$$\epsilon(i) = \frac{\overline{A_o}(T_i)}{A(T_i)} = \lambda \left(\frac{W_{opt}}{W_i} \right)^{1/3} \tag{6.21}$$

can be defined for triangle T_i, and thought of in the following terms.

$\epsilon(i) > 1$, triangle is enlarged,

$\epsilon(i) < 1$, triangle is shrunk,

$\epsilon(i) = 1$, triangle is unchanged.

This "expansion factor" is used in the procedure to find the new position of the central node of a patch. It gives the equation of a line onto which the node has to move in order to give the optimal area, $\overline{A_o}(T_i)$, for triangle T_i.

The complete procedure is follows and makes use of the following results :-

$$\text{area of triangle} = 0.5 \times \text{base length} \times \text{perpendicular height} \tag{6.22}$$

and

$$D = \frac{au + bv + c}{\sqrt{a^2 + b^2}} \tag{6.23}$$

where D is the perpendicular distance of the point (u, v) from the line $ax+by+c = 0$, as shown in Figure 6.12.

Using these results and (6.21) gives

$$a\tilde{x}_i + b\tilde{y}_i + c = \epsilon(i)(ax_i + by_i + c), \tag{6.24}$$

since the base length remains constant, with $(\tilde{x}_i, \tilde{y}_i)$ the new position of (x_i, y_i). Thus the new point moves on a line parallel to the base of the triangle.

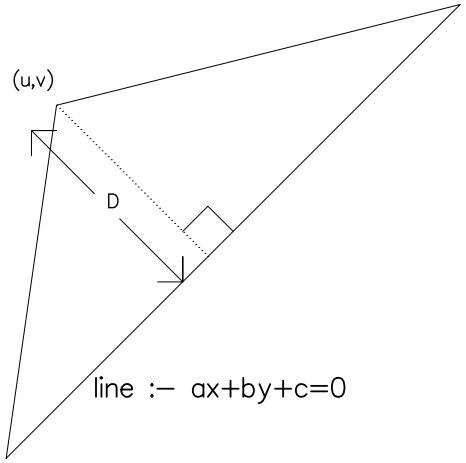

Figure 6.12: Perpendicular distance of a point from a line

Finding the equations from all the triangles in the patch, produces P equations for the position of the new node. The use of some or all of these equations allows the calculation of the new nodal position.

Amongst the procedures which can be used to find the nodal position are :-

1. Least squares fit :- since P lines will almost certainly not meet at a point.

2. Using the equations associated with the first triangle in a patch and its neighbour to give the new nodal position: however this does not guarantee that the node will stay inside the patch, especially if $\epsilon(1)$ or $\epsilon(2)$, as defined by (6.21), are greater than 1.

3. Where possible, using the equations associated with triangles j and j+1 in the patch, where $\epsilon(j)$ and $\epsilon(j+1)$ are both less than or equal to 1. If this is not possible, choosing j so that

$$\epsilon(j) + \epsilon(j+1) < \epsilon(k) + \epsilon(k+1), \quad k = 1..p, \quad j \neq k. \tag{6.25}$$

This improves the likelihood of a node remaining inside a patch.

In all cases, however, it is still necessary to check for degenerate triangles (see Figure 6.13). If a degenerate triangle occurs then only a proportion of the move

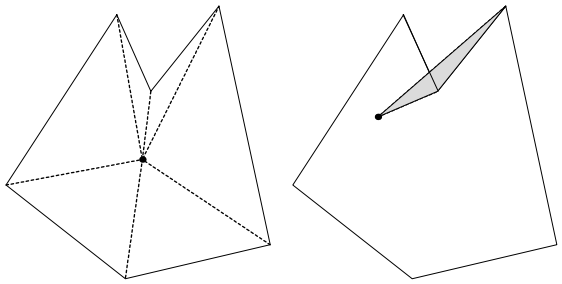

Figure 6.13: Introduction of a degenerate triangle by nodal movement

is made, and another check is made. If it remains degenerate then the node is not moved.

The results which are produced by this procedure are very poor, as can be seen in Figures 6.14 to 6.17. Figure 6.14 shows the representation of DD1 by the 100 point set when the nodes have been moved using the error equalisation procedure. The representation is poor in the region of the mountain and unchanged in the region of the ramp due to the determinant of the Hessian being zero there. The numerical errors also show that this is an inadequate representation.

Figure 6.15 shows the representation of P2 which is produced by the error equalisation criterion on the 100 point set. The numerical errors show that this is an inadequate representation when compared to the Delaunay triangulation.

Figures 6.16 and 6.17 show that with different point sets the representations produced can vary widely. The representation in Figure 6.16 is adequate but the representation in Figure 6.17 is abysmal.

It is also necessary to note that this procedure has no effect on the Function

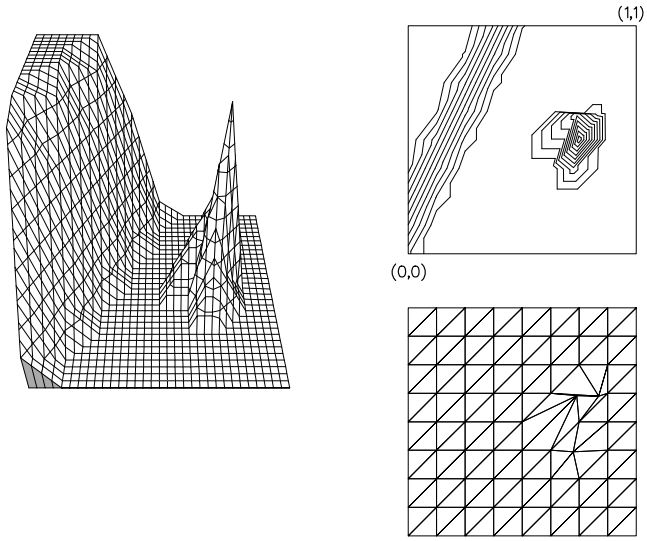

Figure 6.14: Error equalisation triangulation of DD1 (81 points)

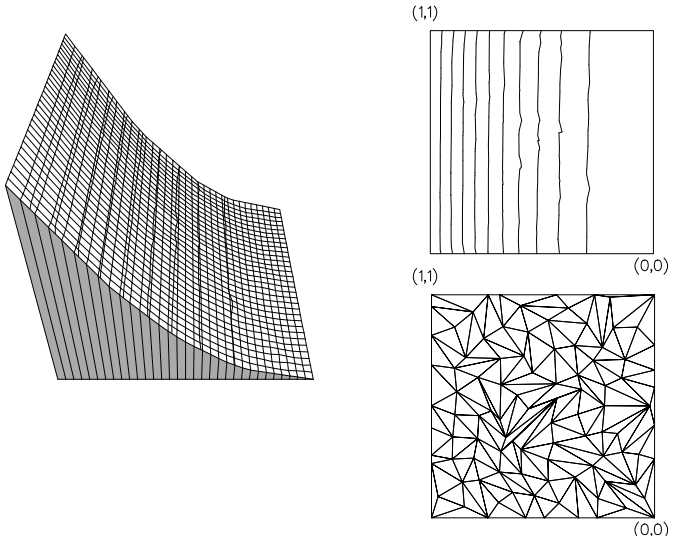

Figure 6.15: Error equalisation triangulation of P2 (100 points)

SR1, as the determinant of the Hessian is always zero for this function so the initial triangulation is unchanged. It is possible to see that this procedure does not improve the representations in the cases mentioned and in almost all cases the procedure produces similar or worse results.

Another problem is the stopping criterion which is very problem dependent and has to be adjusted for each problem and data set. This makes the procedure almost useless as a practical proposition, although a possible modification which

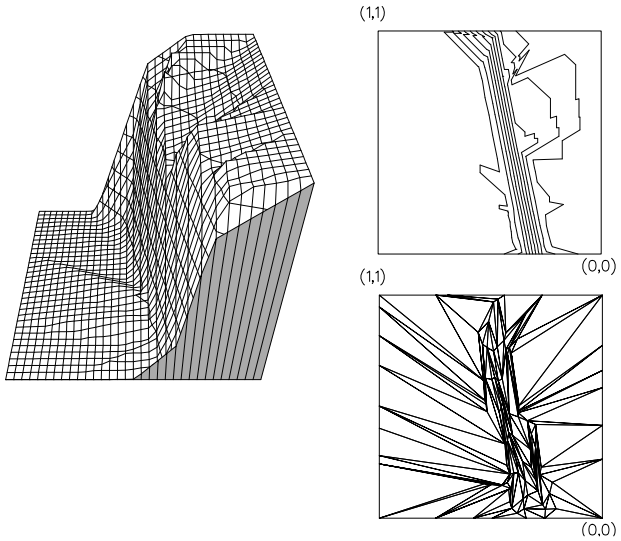

Figure 6.16: Error equalisation triangulation of SR3 (100 points)

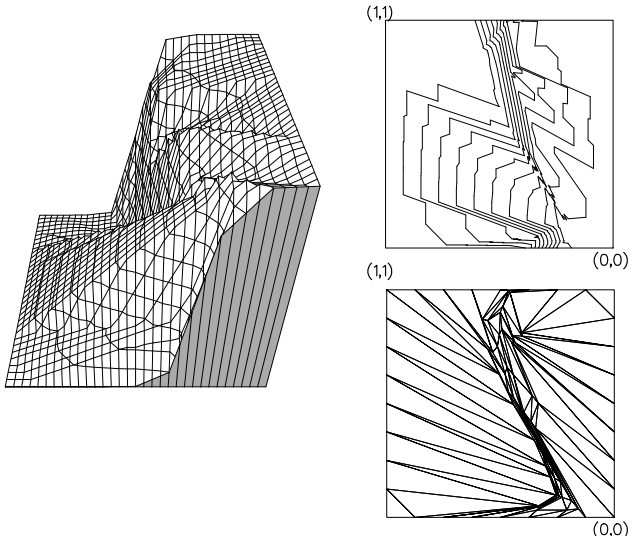

Figure 6.17: Error equalisation triangulation of SR3 (81 points)

could improve the procedure is to work towards equalising the actual errors on each triangle in a patch rather than equalising the bounds on the errors.

6.3 Implementation

The techniques of implementation which were employed in the testing of the criteria are outlined here. First an original triangulation is set up, usually Delaunay,

(although various grids produced by data dependent reconnection criteria are also used, see Section 6.4), the aim being to see if nodal movement depends heavily on the connectivity of the original triangulation.

An iteration technique is used to move the nodes, with each node being moved inside its patch to a new position, and the next node in the set of nodes then being moved. This is analogous to the Gauss-Seidel iteration procedure, as the new position is used in all subsequent moves and all the spring constants are recalculated for each node. An alternative strategy is analogous to the S.O.R method if some proportion, ω, of the calculated move is taken. A complete global method, where all the spring constants are found at the start of a cycle and an n_i by n_i matrix system is solved, where n_i is the number of interior nodes, is not used as it is computationally very expensive to repeatedly solve the resulting matrix system, to convergence, for large numbers of nodes.

Following each sweep through the list of nodes, a check is made to see if the nodes have converged to a steady solution. The check used depends on the criterion in use, e.g. for the spring analogy the check is to calculate the maximum distance moved in each sweep and if this is less than some tolerance, $dtol$, then the method is said to have converged. This tolerance could vary, depending on the function and the length of time the procedure was to take, although it was usually taken to be $O(10^{-3})$. For the error equalisation criterion a note is made of the minimum value of $\epsilon(i)$, the expansion factor for a triangle, over all patches in a sweep. When this minimum value is greater than a given tolerance less than 1, usually about 0.8, then the procedure is stopped as it can be assumed that all triangles are close to the "ideal" size, and the areas have converged to those

required, within a small tolerance.

It is also necessary to look at the treatment of the boundary nodes for both techniques, as in all tests recounted so far, only the interior nodes have been repositioned. For the error equalisation criterion no simple treatment of the boundary nodes exists so they remain fixed, but for the spring analogy a number of treatments exist. Originally the effect of not moving the boundary nodes was investigated, but was found not to be desirable as this created thin triangles round the boundary and caused over-constrained movement of the interior nodes, as can be seen in Figures 6.3 and 6.6 to 6.11, and hence the decision to move the boundary nodes, but to constrain them to move along the boundary was made. Originally this was implemented before the interior nodes were moved, by using the same spring analogy as for the interior nodes, but using only the connections on the boundary. However, this again caused some problems since it was possible that in moving the boundary nodes to their optimal points and keeping the connections the same, degenerate triangles could be produced. This problem can be seen in Figure 6.18 where the shaded area is the triangle which results when only the boundary nodes are moved but the connectivity stays the same. As can be seen the triangle is greatly enlarged, (originally it was the same size as those in the centre of the mesh), and it overlaps other triangles. The same applies to the triangle in the top left corner (which has not been shaded). The representation, contours and numerical results are unreliable in this case due to the overlapping of triangles.

This difficulty could be tackled in two ways :-

1. Include all nodes in a sweep through the list of nodes, but constrain the

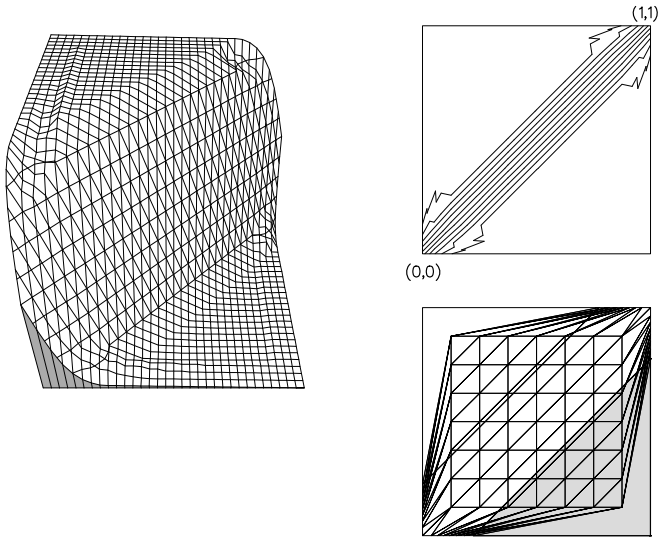

Figure 6.18: Boundary nodes moved only for SR1 (81 points)

vertices of the region to stay fixed and the boundary nodes to remain on the boundary, and then continue until all nodes have moved to their optimal positions, hopefully eliminating the production of degenerate triangles.

2. Once the boundary nodes have been moved to their new positions, re-Delaunay the resulting nodal positions to produce a new initial triangulation (This is computationally the most expensive procedure). This revised nodal connectivity is then employed to calculate the revised positions of the interior points only.

One disadvantage of (2) is that the original connectivity is not preserved and another is that it is computationally very expensive. However with the nodal positions which occur in Figure 6.18 the result of this procedure can be satisfactory (see Figures 6.19 and 6.20). Figure 6.19 shows the resultant Delaunay triangulation from the set of nodes shown in Figure 6.18, while Figure 6.20 shows the resultant positions of the interior nodes when they are moved using the spring

constant, $k_j = 1 + (u_{ss})^{2/5}$. However good the results may be in this case the major drawback is the degenerate triangles which are introduced.

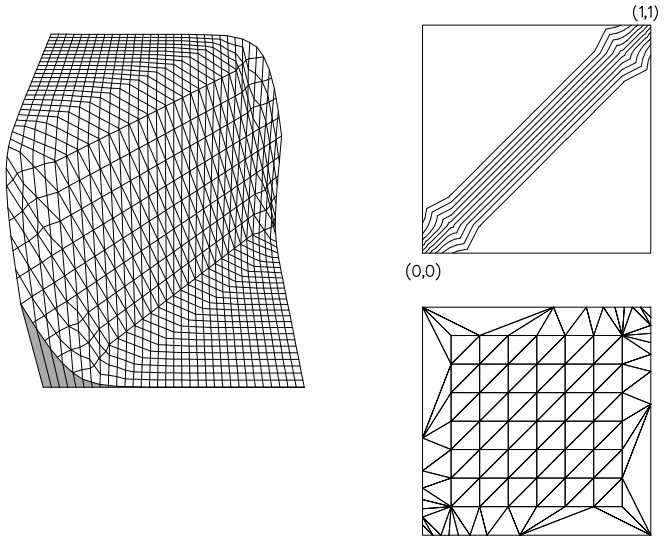

Figure 6.19: Delaunay triangulation of points in Figure 6.18 for SR1 (81 points)

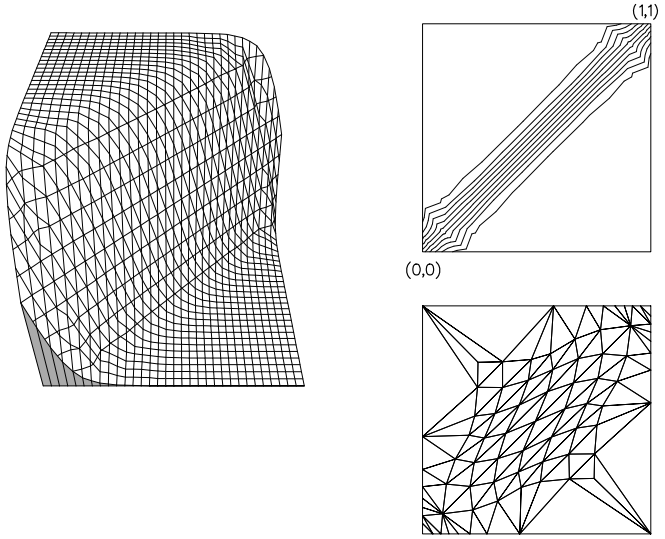

Figure 6.20: Connectivity in Figure 6.19, interior nodes moved for SR1 (81 points)

The advantages of (1) are the lack of extra work to find a new triangulation and the preservation of the original connectivity. The disadvantage of (1) is that if the boundary nodes are constrained to the boundary but all connections in the triangulations must be used as springs, then ghost points must be introduced outside the region (see Figure 6.21). These ghost points are points created solely

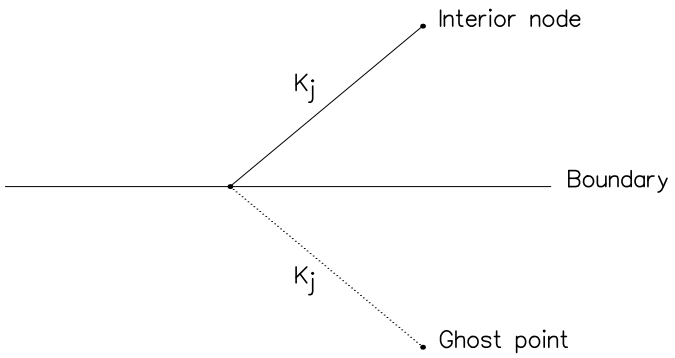

Figure 6.21: Creation of a ghost point

to assist with the calculation of the new positions of the boundary nodes. The procedure is as follows :- if an internal node is connected to a boundary node, then, treating the boundary of the region as a mirror, a ghost point is created outside the region, at the mirror image of the interior point. The resultant ghost point is connected to the boundary node by an imaginary line with spring constant equal to that of the line joining the interior node to the boundary node. In computations the springs which form part of the region boundary have their spring constants multiplied by some factor, δ, ($\delta \approx 5$), following Thacker *et al.* (1980), so that the effect from the internal connections does not become too significant and cause the poor positioning of boundary nodes. The results of using this procedure are shown in Figure 6.22, which was produced for Function SR1 with $\delta = 5$ and values of $\alpha = 10$ and $\beta = 2$ in (6.12). As can be seen from the numerical results in Table 6.2 this has the lowest error of any reconnection procedure for SR1 which uses repositioning only.

Figure 6.23 shows the results when the same parameters are used for Function DD1. Once again the numerical results are not as good as might be expected due

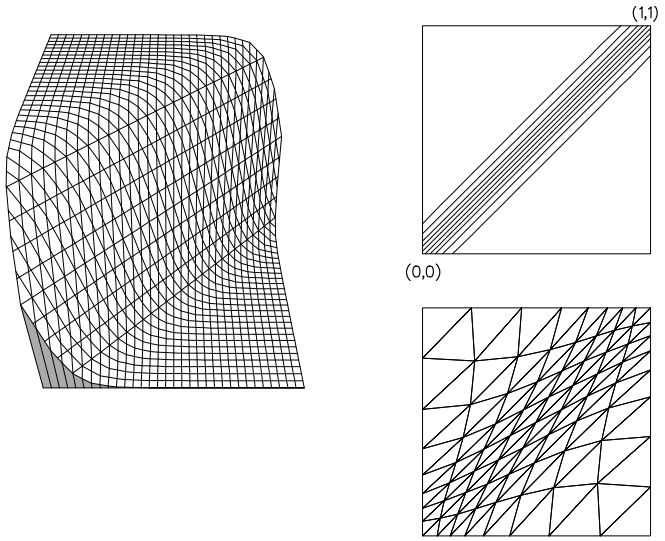

Figure 6.22: All Nodes Redistributed using (6.12) for SR1 (81 points)

to the large weighting on the curvature, though the boundary nodes do actually move towards the ramp.

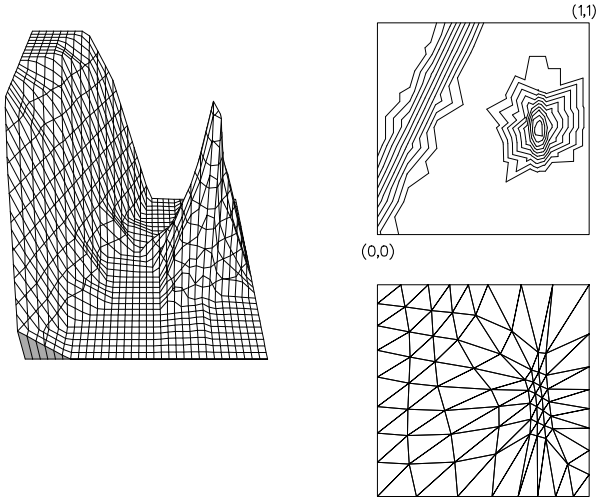

Figure 6.23: All Nodes Redistributed using (6.12) for DD1 (81 points)

The five tables which follow show the numerical errors and geometrical measures associated with the different representations already presented and those associated with ones which will follow in the next section. The errors are calculated as outlined in Section 4.3.

Errors for the Error Equalisation (EE) criterion					
Method	Function, points	L_2 – error x10^{-2}	mean interpolation x10^{-2}	max interpolation x10^{-1}	Figure
Delaunay	P2, 100	34.948	28.374	18.073	
EE	P2, 100	39.972	30.925	18.073	6.15
Delaunay	SR3, 100	8.15303	3.85048	6.42352	
EE	SR3, 100	12.7467	7.60751	6.42352	6.16
Delaunay	SR3, 81	5.96940	2.63085	26.7231	
EE	SR3, 81	38.6385	24.7657	12.3176	6.17

Table 6.1: Errors for the Error Equalisation criterion

Table 6.1 shows that the numerical errors associated with the error equalisation criterion triangulations presented are worse than those of the corresponding Delaunay triangulations.

Table 6.2 shows the numerical errors for the triangulations representing the Function SR1, produced by the set of 81 data points. As can be seen, moving the nodes improves the errors in almost all the triangulations shown in the previous sections in this chapter. The triangulations for which the errors are worse are those which involve the generation of degenerate triangles. As has been previously noted the best errors are produced by movement criteria based on curvature, although the gradient based spring constant did make a slight improvement in the errors. The greatest improvements can be in seen in the entries in the table which show that moving all the nodes including the boundary nodes improves the L_2-error greatly, and also decreases the interpolation errors to less than 25%

Errors for SR1 (81 points)				
Method	L_2 – error x 10^{-3}	mean interpolation x 10^{-3}	max interpolation x 10^{-2}	Figure
Delaunay	4.32505	2.25362	1.16866	5.13
Interior nodes moved	2.59611	1.44170	1.16866	6.3
Interior nodes moved	2.75337	1.53412	1.16866	6.6
Interior nodes moved	3.04027	1.71930	1.16866	6.8
Interior nodes moved	2.68037	1.51198	1.16866	6.10
Boundary nodes moved	5.40282	3.28707	1.53749	6.18
Delaunay of nodes in 6.14	5.20637	2.49211	2.49268	6.19
Nodes of 6.15 moved	3.82267	1.67048	2.43169	6.20
All nodes moved	0.96465	0.60172	0.30352	6.22
Reconnected (ABN-2)	4.32775	2.25362	1.16866	
Moved then reconnected	0.67286	0.35808	0.27266	6.24
Reconnected then moved	1.21347	0.70576	0.44116	6.25

Table 6.2: Errors for SR1 (81 points)

of those of the Delaunay triangulation. Table 6.3 shows the associated geometric measures.

Geometric measures for SR1 (81 points)				
Method	Angles	Skewness	AR (Lo)	Figure
	min, max	mean, max	mean, min	
Delaunay	45.0, 90.0	20.0, 20.0	0.866, 0.866	5.13
Interior nodes moved	17.4, 143.1	26.1, 55.4	0.756, 0.370	6.3
Interior nodes moved	25.4, 129.0	22.8, 46.0	0.810, 0.512	6.6
Interior nodes moved	22.8, 134.0	22.8, 49.3	0.805, 0.462	6.8
Interior nodes moved	23.8, 132.0	23.4, 48.0	0.799, 0.482	6.10
Boundary nodes moved	0.65, 169.1	37.8, 72.7	0.569, 0.020	6.18
Delaunay of nodes in 6.18	11.0, 147.9	23.4, 58.6	0.778, 0.282	6.19
Nodes of 6.19 moved	10.5, 141.0	31.4, 54.0	0.662, 0.296	6.20
All nodes moved	17.0, 145.3	42.1, 56.9	0.546, 0.349	6.22
Reconnected (ABN2)	18.4, 135.0	46.3, 50.0	0.487, 0.433	
Moved then reconnected	0.11, 179.7	66.6, 79.8	0.169, 0.003	6.24
Reconnected then moved	6.62, 166.7	34.9, 71.2	0.522, 0.133	6.25

Table 6.3: Geometric measures for SR1 (81 points)

Errors for DD1 (81 points)				
Method	L_2 – error x10^{-2}	mean interpolation x10^{-2}	max interpolation x10^{-1}	Figure
Delaunay	4.44908	1.62930	3.08566	5.14
Nodes outside region	32.7656	18.7799	8.20206	6.2
Interior nodes moved	9.84769	4.59733	5.24890	6.7
Interior nodes moved	4.44787	2.10183	2.18771	6.9
Interior nodes moved	8.10345	3.82542	4.52270	6.11
All nodes moved	8.70128	4.20088	4.50826	6.23
Reconnected (ABN-2)	4.50187	1.29741	3.08566	
Moved then reconnected	7.26486	2.56037	4.55134	6.26
Reconnected then moved	12.3880	7.34259	4.72496	6.27
Error equalisation	6.13078	2.14163	4.73528	6.14

Table 6.4: Errors for DD1 (81 points)

Table 6.4 shows that for the Function DD1 the nodal movement criteria do not work as well as might be expected. The curvature based spring constants increase the errors associated with the produced triangulations, but the spring constant based solely on gradient produces similar results to the Delaunay triangulation. The triangulation produced by moving all the nodes is not as good as expected due to the spring constant being weighted towards curvature rather than gradient. Table 6.5 shows the geometric measures for DD1 with the 81 point data set. The smallest angle introduced by nodal movement is 6°.

Geometric measures for DD1 (81 points)				
Method	Angles	Skewness	AR (Lo)	Figure
	min, max	mean, max	mean, min	
Delaunay	45.0, 90.0	20.0, 20.0	0.866, 0.866	5.14
Nodes outside region	0.06, 179.9	46.6, 79.9	0.329, 0.001	6.2
Interior nodes moved	6.03, 143.8	23.7, 55.9	0.726, 0.181	6.7
Interior nodes moved	19.9, 116.2	20.4, 37.5	0.818, 0.533	6.9
Interior nodes moved	5.81, 147.8	23.0, 58.5	0.738, 0.174	6.11
All nodes moved	6.54, 125.6	20.0, 43.7	0.779, 0.196	6.23
Reconnected (ABN2)	1.40, 175.2	40.4, 76.8	0.567, 0.038	
Moved then reconnected	0.34, 179.1	41.8, 79.4	0.486, 0.009	6.26
Reconnected then moved	3.83, 169.0	31.9, 72.7	0.583, 0.107	6.27
Error equalisation	2.08, 175.4	21.3, 76.9	0.840, 0.045	6.14

Table 6.5: Geometric measures for DD1 (81 points)

As mentioned at the start of the chapter it is possible to introduce geometric constraints into the nodal movement process. Such constraints were implemented but since nodal movement does not generate small angles or small triangles no results will be shown.

Following on from these nodal movement criteria for an initial triangulation with fixed connectivity, in the next section the consequences of starting with an initial triangulation and then either "moving and reconnecting" or "reconnecting and moving" are investigated.

6.4 Combinations of Movement and Reconnection

If a triangulation of a set of nodes is given (e.g. the result of a triangulation front procedure) then it may be desirable to modify the triangulation (post processing) to better represent some data. The techniques which have been discussed for such a triangulation are nodal movement and nodal reconnection and in this section the different effects of the order of the operations are investigated, i.e. what is the difference in results of "moving and reconnecting" and "reconnecting and moving". It can be seen from the graphical results in the text and the numerical tables at the end of the chapter that the order of these operations has a significant effect on the final grids and representations.

As in the previous sections of this chapter and Chapter 5 the initial triangulation will be a Delaunay triangulation. When the description says "moved" it means that all the nodes are moved using (6.12) with $\alpha = 10$ and $\beta = 2$. This is due to the reasons quoted in Section 6.1. The "reconnection" procedure is the ABN-2 criterion described in Section 5.3.3. This criterion was chosen as it is the nodal reconnection criterion which in general produces the best representations of the underlying data using the grids formed. Another reconnection procedure was also tested, in which a quasi-equidistribution criterion was used, instead of the monitor function being $u_{ss}^{2/5}$, the monitor used was consistent with the criterion used to reposition the nodes. This consistent procedure did not produce results which were as good as those achieved by using ABN-2 on the triangulations produced by moving the nodes.

The form of the descriptions will be as follows :- if the initial procedure has already been presented then the description will be " procedure X was carried out on the triangulation shown in Figure Y", while if the initial procedure has not been previously described the description will be "procedure X was carried out and then procedure Z was applied to the resulting triangulation".

Figures 6.24 to 6.27 show the results associated with the point sets produced in Section 6.3. These figures show the differences that "moving and reconnecting" and "reconnecting and moving" produce.

Figure 6.24 shows the effect on the point set shown in Figure 6.22 of reconnecting the nodes using the ABN-2 criterion. This shows the long, thin triangles produced along the ramps face.

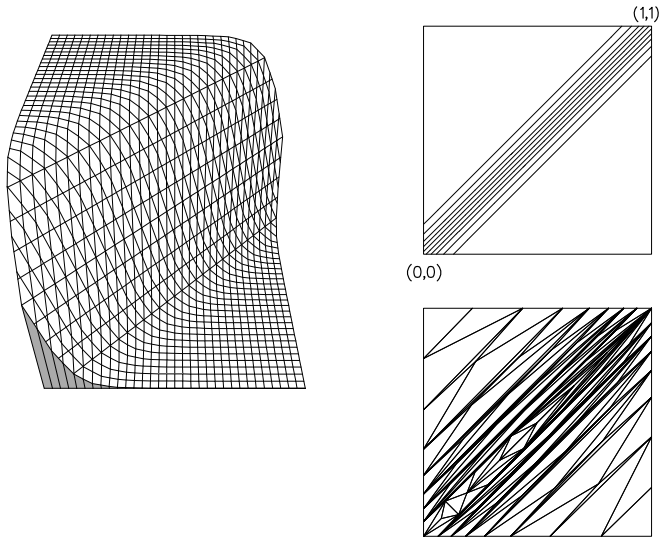

Figure 6.24: Nodes of Figure 6.22 reconnected using ABN-2 for SR1 (81 points)

Figure 6.25 shows the effect of reconnecting the nodes using ABN-2 and then moving them using $\alpha = 10$ and $\beta = 2$ in (6.12). The effect of the ABN-2 reconnection procedure is slight but it changes the connectivity in such a way that the nodes in Figure 6.25 have been pulled towards the top right corner of

the grid.

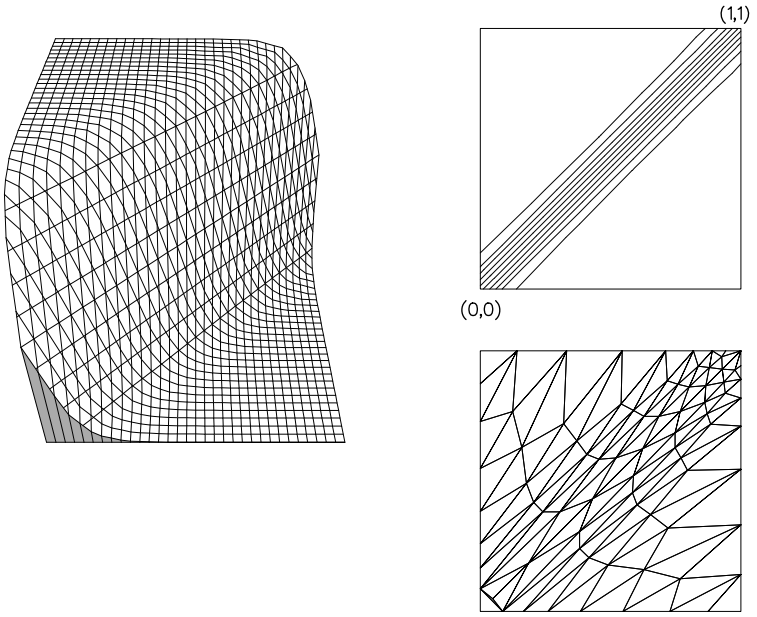

Figure 6.25: Nodes reconnected using ABN-2 and moved using (6.12) for SR1 (81 points)

Figures 6.26 and 6.27 show the differences for Function DD1 and the 81 point data set between "moving and reconnecting" and "reconnecting and moving". Figure 6.26 shows the effect of reconnecting the nodal set shown in Figure 6.23. The ramp is well represented, while the mountains sees an improvement in representation, although the numerical results do not totally bear this out.

Figure 6.27 shows the effect that the initial connectivity can have when the nodes are moved. In this case the nodes are pulled to the mountain, while the ramp becomes smeared due to the pull of all the nodes in the region of the mountain.

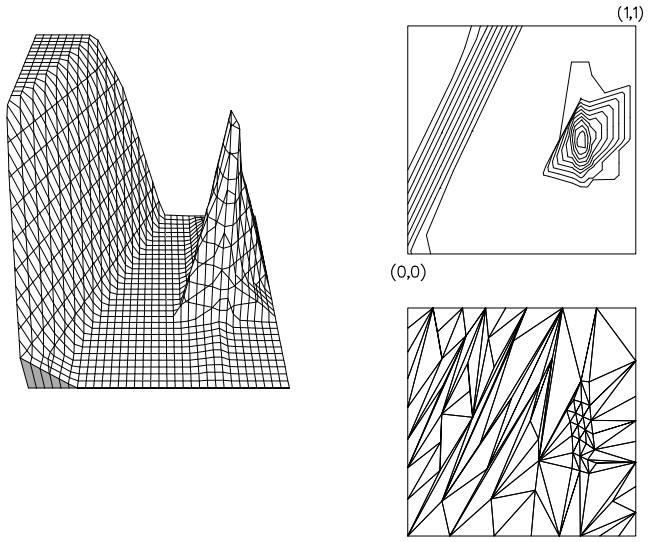

Figure 6.26: Nodes of Figure 6.23 reconnected using ABN-2 for DD1 (81 points)

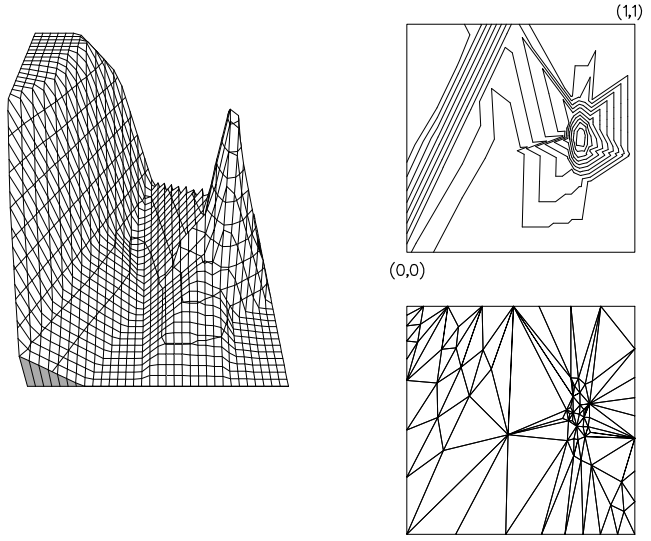

Figure 6.27: Nodes reconnected using ABN-2 and moved using (6.12) for DD1 (81 points)

To show the exact effect of the different procedures, we now present the effects on three different functions, all with the 100 data point set, of the main procedures outlined before :-

- nodal reconnection,
- nodal movement,
- movement then reconnection, and
- reconnection then movement.

Figures 6.28 to 6.31 show the results of these procedures for Function SR1 with the 100 data point set.

Figure 6.28 shows nodes moved towards the centre, and the better representation of the boundary. The numerical errors are considerably improved.

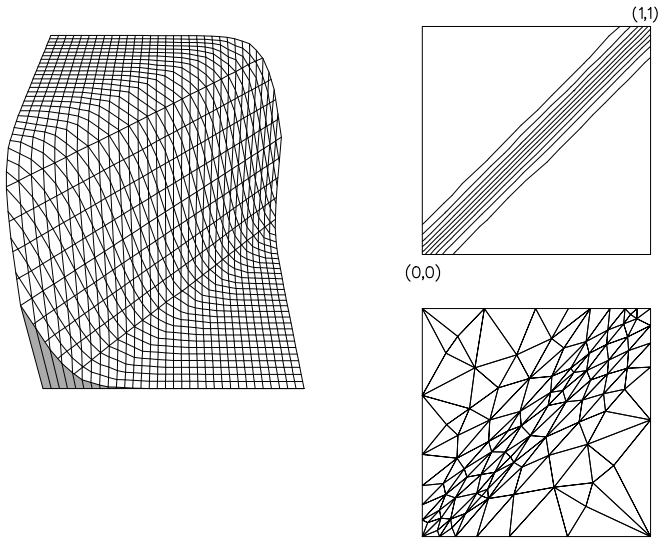

Figure 6.28: Nodes moved for SR1

Figure 6.29 shows the effect of reconnection using ABN-2 only. Long, thin triangles are produced but the apparent poor representation of the boundary is borne out by the maximum interpolation error.

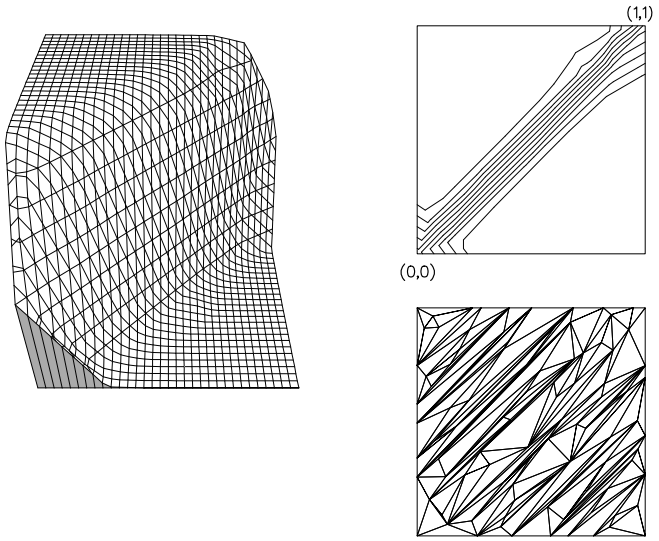

Figure 6.29: Nodes reconnected using ABN-2 for SR1

Figure 6.30 shows the effect of reconnecting the nodes in Figure 6.28 using ABN-2. The representation is much better as can be seen from the numerical results where all the errors are an order of magnitude better than those given by the original Delaunay triangulation.

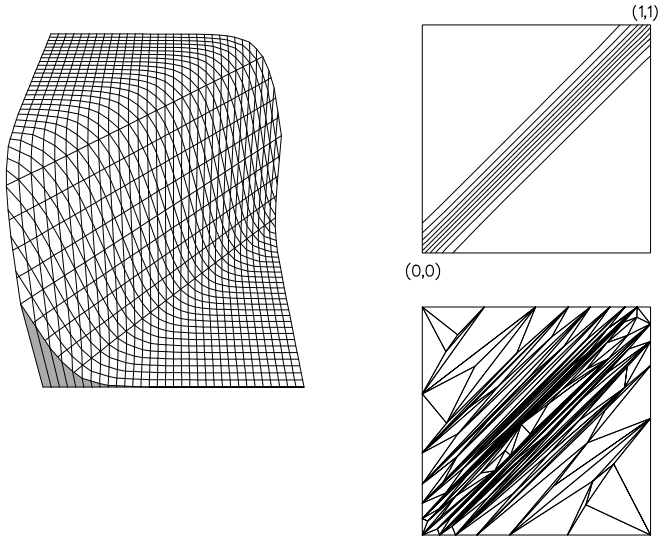

Figure 6.30: Nodes in Figure 6.28 reconnected using ABN-2 for SR1

Figure 6.31 shows the poor representation achieved by starting with the very irregular grid shown in Figure 6.29. The boundary nodes are poorly placed and

the nodal positions are not as expected due to the increased number of connections at certain nodes.

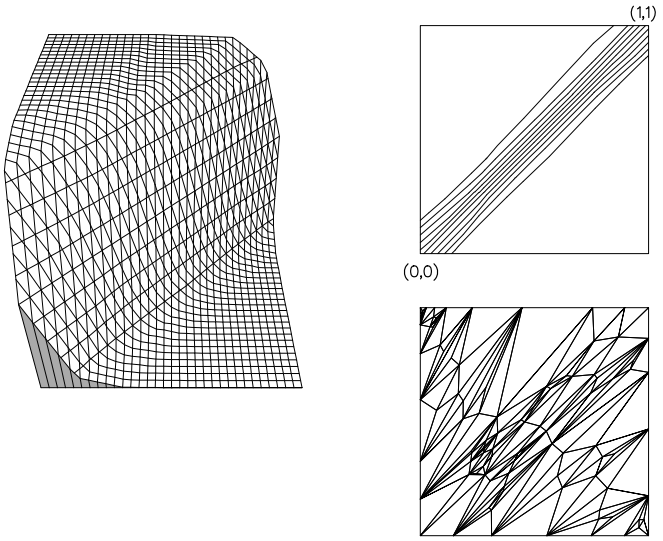

Figure 6.31: Nodes reconnected using ABN-2 and moved using (6.12) for SR1

The overall effect of Figures 6.28 to 6.31 is to show that for SR1, with the 100 data point set, "move and reconnect" gives the best results.

Figures 6.32 to 6.35 show the representation of DD1 with the 100 data point set.

Figure 6.32 shows the effect of moving the nodes. The nodes cluster towards the mountain while some remain near the ramp. Although the maximum interpolation decreases, the L_2-, and mean interpolation, errors increase on those of the Delaunay triangulation.

Figure 6.33 shows the effect of reconnecting the nodes. The ramp is well represented but a triangle is produced which has an edge joining (0,0) to a point on the boundary of the region, $y = 1$. This edge splits the region into two pieces, which has important repercussions. The mountain is also well represented, as can be seen from the numerical errors which show that this is the best triangulation

numerically.

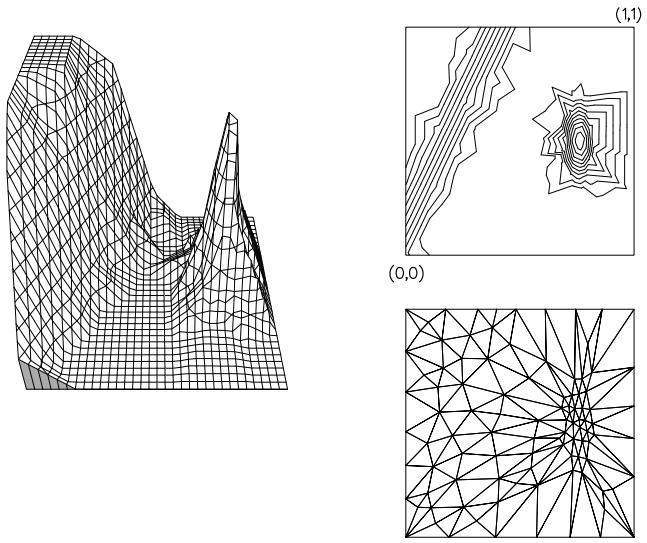

Figure 6.32: Nodes moved for DD1

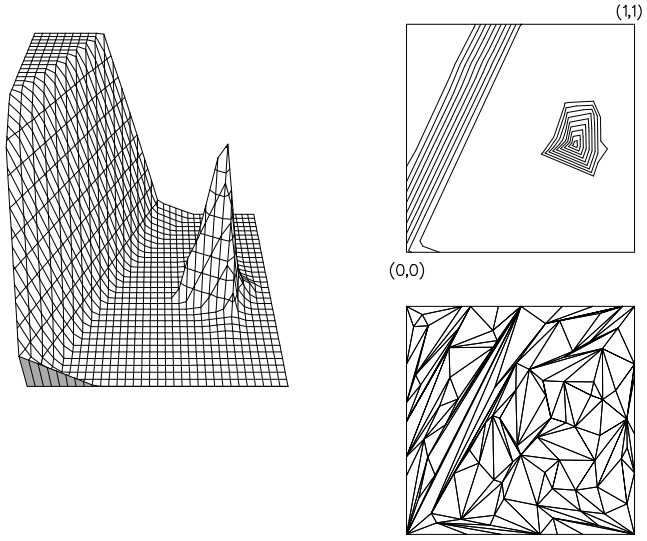

Figure 6.33: Nodes reconnected using ABN-2 for DD1

Figure 6.34 shows the effect of reconnecting the nodes shown in Figure 6.32. The ramp is well represented except at the boundary, and the mountain is well represented except near the boundary, $x = 1$, where the smearing of contours is due to the nodes in the mountain being on the mountains' curved surface rather than at its junction with the plane. The numerical interpolation errors are better than those for the Delaunay triangulation, but the L_2-error is worse.

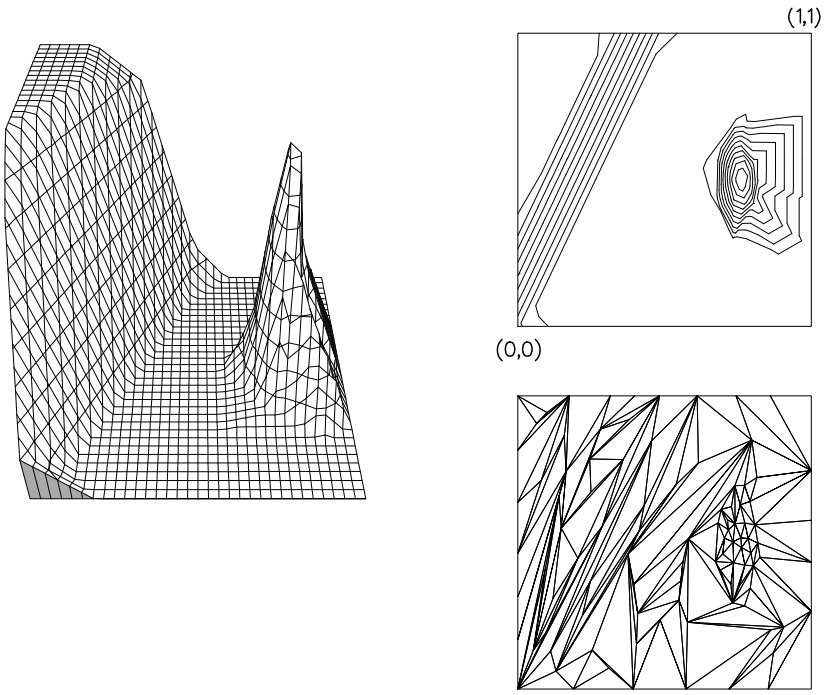

Figure 6.34: Nodes above reconnected using ABN-2 for DD1

Figure 6.35 shows the effect of moving the nodes with the fixed connectivity as shown in Figure 6.33. The importance of the edge splitting the region into two is now apparent. All the nodes to the left of the edge are stuck and can only move in this region, the flat plane near the top of the ramp, while those to the right of the edge are not numerous enough, or do not have the connectivity, to ensure that the mountain is well represented. These defects lead to the resultant, very poor representation where the ramp and mountain almost meet. These defects can also be seen in the numerical results.

Figures 6.32 to 6.35 show that the representation of DD1 can be improved by changing the triangulation, but that "moving and reconnecting" is not always the best procedure (in this case reconnecting using ABN-2 is better).

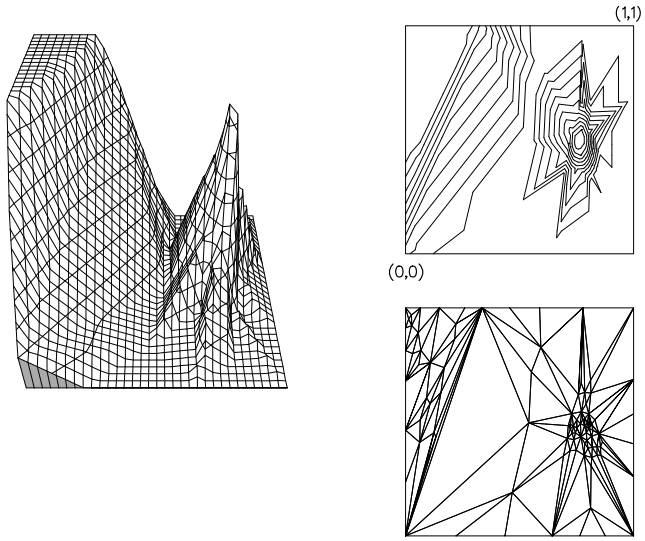

Figure 6.35: Nodes reconnected using ABN-2 and moved using (6.12) for DD1

Figures 6.36 to 6.40 show the effect of all the procedures on the 100 data point set for the Function M1.

Figure 6.36 shows the Delaunay triangulation for 100 points and the representation of M1 using this grid.

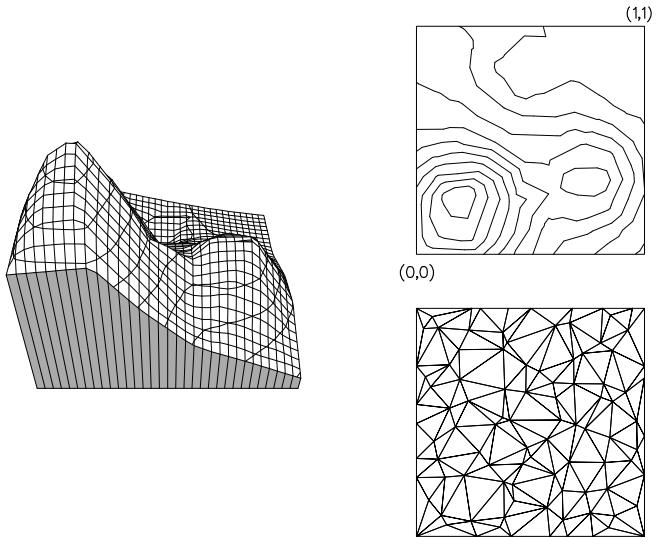

Figure 6.36: Delaunay for M1

Figure 6.37 shows the effect of reconnecting the nodes using the ABN-2 criterion. The numerical results show that this has little effect on the representation.

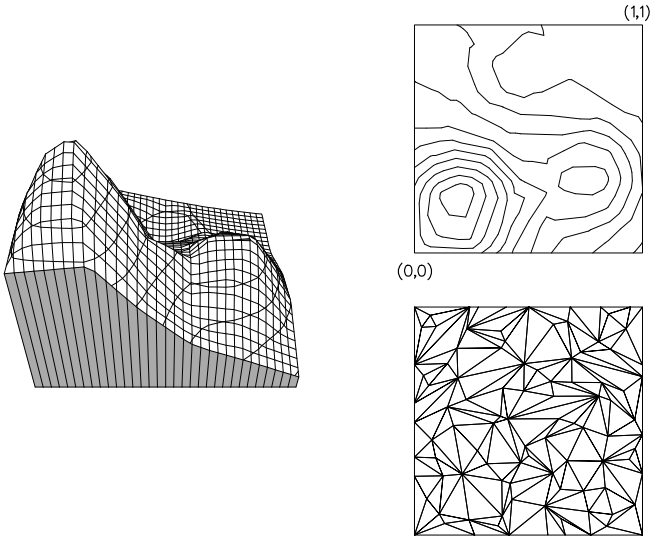

Figure 6.37: Nodes reconnected using ABN-2 for M1

Figure 6.38 shows the nodal positions which result when the nodes are moved while keeping the connectivity shown in Figure 6.36. The numerical errors show a marked improvement, the L_2-error is decreased by almost a half, as is the maximum interpolation error.

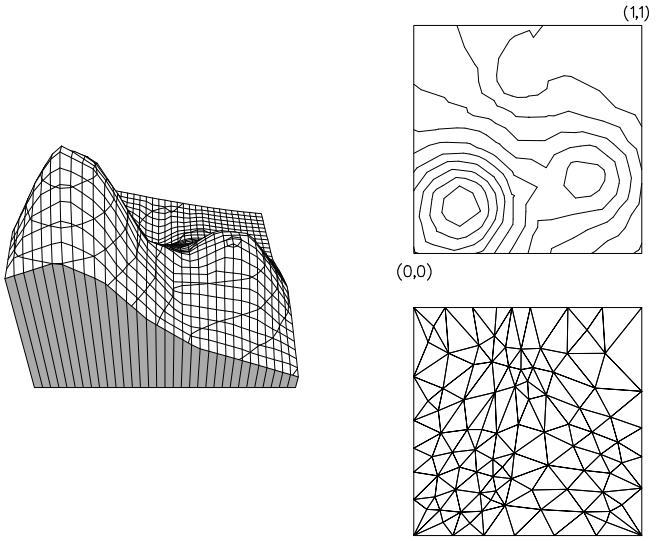

Figure 6.38: Nodes moved for M1

Figure 6.39 shows the effect of moving the nodes to the positions shown in Figure 6.38 and then reconnecting them using the ABN-2 criterion. The nu-

merical errors are the best of those presented, with all the values decreasing by approximately 50 % from those of the Delaunay triangulation.

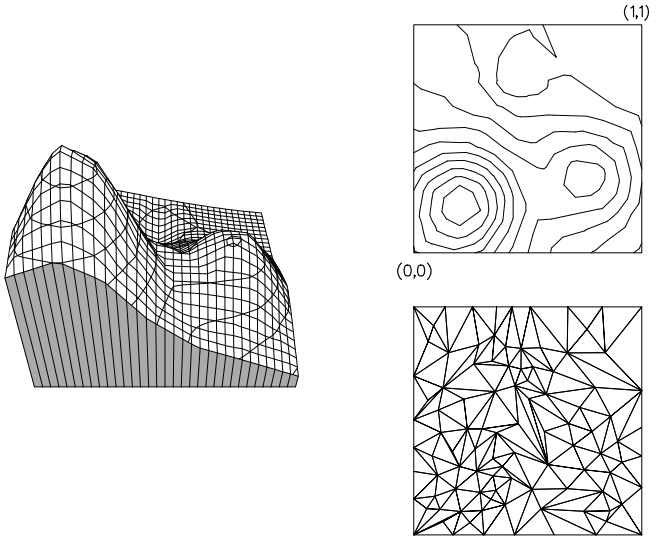

Figure 6.39: Nodes moved then reconnected using ABN-2 for M1

Figure 6.40 shows the effect of moving the nodes with the connectivity shown in Figure 6.37. The numerical results are an improvement over those associated with the Delaunay triangulation but are not as good as those produced by some of the other procedures listed here.

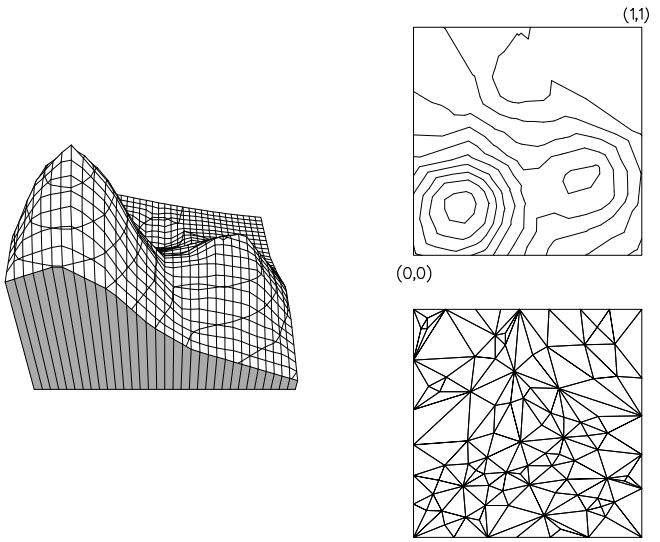

Figure 6.40: Nodes reconnected using ABN-2 then moved for M1

Figures 6.36 to 6.40 show that for a general function, the procedures outlined previously can improve the representation of data when interpolated on the triangulation. In this case, the best procedure is moving the nodes and then reconnecting, but most of the other procedures also produce an improvement in the representation.

Errors for SR1 (100 points)				
Method	L_2 – error x 10^{-3}	mean interpolation x 10^{-4}	max interpolation x 10^{-3}	Figure
Delaunay	7.57001	41.1860	38.3141	5.15
Reconnected (ABN-2)	4.60538	20.3663	38.3141	6.29
Moved	1.16079	7.30275	4.40141	6.28
Moved then reconnected	0.50770	2.88055	4.05630	6.30
Reconnected then moved	2.96845	14.8279	18.2749	6.31

Table 6.6: Errors for SR1 (100 points)

Geometric measures for SR1 (100 points)				
Method	Angles	Skewness	AR (Lo)	Figure
	min, max	mean, max	mean, min	
Delaunay	10.2, 152.5	23.3, 61.7	0.740, 0.275	5.15
Reconnected (ABN-2)	0.04, 179.7	54.0, 79.8	0.304, 0.001	6.29
Moved	9.39, 150.9	30.9, 60.6	0.635, 0.272	6.28
Moved then reconnected	0.03, 179.9	64.5, 79.9	0.194, 0.0008	6.30
Reconnected then moved	1.51, 172.8	46.1, 75.2	0.340, 0.046	6.31

Table 6.7: Geometric measures for SR1 (100 points)

Tables 6.6 to 6.11 show that the procedures outlined previously can actually produce differences in the numerical errors produced by the interpolants on the grids. They also show that there is no specified procedure which will improve the representation in all cases, but that one or more of the procedures will usually improve the representation in all cases.

Errors for DD1 (100 points)				
Method	L_2 – error x 10^{-2}	mean interpolation x 10^{-2}	max interpolation x 10^{-1}	Figure
Delaunay	5.87095	2.40301	4.54757	5.16
Reconnected (ABN-2)	4.91634	1.50004	3.47278	6.33
Moved	7.48263	3.60339	3.72416	6.32
Moved then reconnected	6.30996	2.00391	3.72416	6.34
Reconnected then moved	14.0425	8.71242	4.57597	6.35

Table 6.8: Errors for DD1 (100 points)

Geometric measures for DD1 (100 points)				
Method	Angles	Skewness	AR (Lo)	Figure
	min, max	mean, max	mean, min	
Delaunay	10.2, 152.5	23.3, 61.7	0.740, 0.275	5.16
Reconnected (ABN-2)	0.06, 179.8	45.3, 79.9	0.431, 0.001	6.33
Moved	6.18, 151.5	22.3, 61.0	0.744, 0.185	6.32
Moved then reconnected	0.05, 179.6	46.6, 79.8	0.428, 0.001	6.34
Reconnected then moved	0.80, 177.7	34.5, 78.4	0.560, 0.020	6.35

Table 6.9: Geometric measures for DD1 (100 points)

Errors for M1 (100 points)				
Method	L_2 – error x 10^{-2}	mean interpolation x 10^{-2}	max interpolation x 10^{-2}	Figure
Delaunay	2.76383	1.57275	14.8171	6.36
Reconnected (ABN-2)	2.66476	1.46380	14.6194	6.37
Moved	1.44762	1.01554	6.97824	6.38
Moved then reconnected	1.31325	0.89145	5.51289	6.39
Reconnected then moved	1.82925	1.21916	6.54900	6.40

Table 6.10: Errors for M1 (100 points)

Geometric measures for M1 (100 points)				
Method	Angles	Skewness	AR (Lo)	Figure
	min, max	mean, max	mean, min	
Delaunay	10.2, 152.5	23.3, 61.7	0.740, 0.275	6.36
Reconnected (ABN2)	3.97, 169.4	31.5, 72.9	0.621, 0.098	6.37
Moved	16.1, 135.2	16.6, 50.2	0.852, 0.450	6.38
Reconnected, then moved	7.60, 164.4	24.4, 69.6	0.742, 0.157	6.39
Moved, then reconnected	2.17, 174.9	24.5, 76.6	0.753, 0.050	6.40

Table 6.11: Geometric measures for M1 (100 points)

The next chapter looks at some of the procedures which can be used to generate triangular grids if the only nodal information prescribed is the coordinates of the vertices of the region.

Chapter 7

Triangulation Front Techniques

In this chapter we look at a set of techniques which can be used to generate triangular meshes which can be used on problems where the vertices of the region are given. The aim is to create a valid triangulation with as many nodes and triangles as are necessary to cover the entire region whilst giving a desired resolution of the features of the underlying data. Boundary nodes between the vertices are first generated and the interior nodes can then either be created before any triangles are generated or can be generated in conjunction with the triangles themselves. The purpose of such techniques is to generate nodes in regions where they are required, with no prior knowledge of where they should be placed.

As mentioned in Section 5.7 it may be necessary to introduce geometric constraints into the grid generation procedure. However it was decided that for the generation of the nodes using the triangulation front procedure it would not matter if the triangles were too small or had small angles as long as the nodes were in the correct region. Investigation showed that post processing of the grids generated by the grid generation procedure was sometimes necessary, and it was at

this stage of the procedure that the geometrical constraints could be introduced.

There are many different ways of going about this strategy and these methods can be split into two families :- "interior nodes first" and "interior nodes and triangles concurrently". The first family to be described is the "interior nodes first".

7.1 Interior Nodes First

In this family of methods the vertices of the region are known and the boundary nodes are then positioned according to some criterion. This criterion can be equi-spacing (Lo (1985)), based on some background function for nodal spacing (Cavendish (1974)); or some other choice. Once the boundary nodes are positioned then the interior nodes can be generated. Lo places the nodes so that they are regularly spaced along the line joining corresponding nodes on opposite edges of the region (thus for the unit square, Lo's procedure produces a regular pattern of nodes). Cavendish places the interior nodes by using the background function for nodal spacing. The region is overlaid with rectangles, whose size depends on the background function, and in a systematic ordering of rectangles, a new node is randomly placed in a rectangle. The node is placed randomly in order that a regular pattern of nodes is not automatically generated.

In Figure 7.1 the region is denoted by the dotted line and Cavendish's rectangles are shown. The boundary nodes can also be seen. If the first rectangle taken was the third up on the right then a new node, A, would be placed at random in that rectangle. If the new node, A, is too close, according to the background function, to an existing node then it is discarded. This procedure continues un-

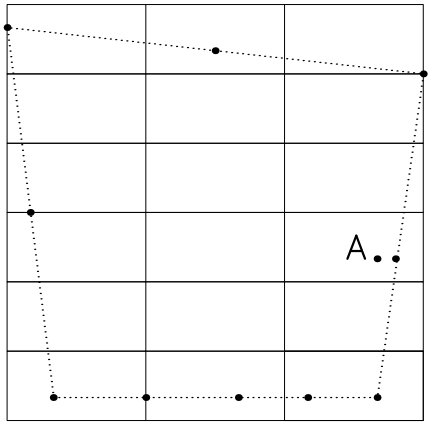

Figure 7.1: Insertion of point in Cavendish's procedure

til a new node is placed in the rectangle or five attempts have been rejected in which case no node is positioned. When all the rectangles have been considered and nodes placed, the resulting nodal distribution is accepted. In Figure 7.1 the maximum number of internal nodes allowed is 18 (one in each rectangle).

Once a complete set of nodes has been generated the procedure for generating the triangles commences. In this procedure the generation of triangles is performed in a geometrical manner (Lo), or in such a way that the sizes of the triangles are dependent on the background data in some form (Cavendish). The triangles are generated in the following fashion. The boundary edges, taken in any anti-clockwise order, form the original front. One edge is chosen from the front and the list of nodes is checked to find all the nodes to the left of the front within a specified distance from the midpoint of the edge. These nodes are checked in order of distance from the midpoint of the edge, with a measure of how close the resulting triangle is to equiangular being calculated. The measures of triangles which could be subsequently generated are also calculated. These measures are the aspect ratios of Lo and Cavendish (Section 4.3). Provided no very long, thin triangles would be generated in later steps the new triangle is accepted and the

front updated.

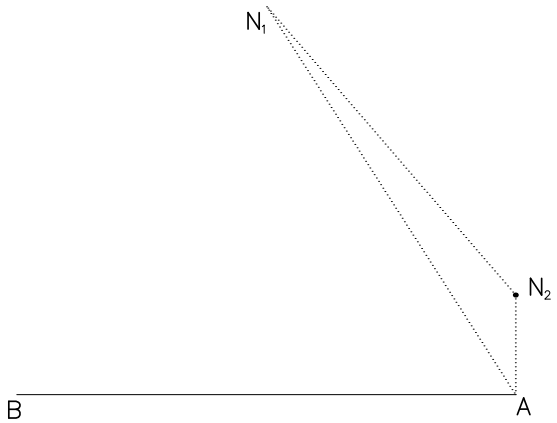

Figure 7.2: How the choice of N_1 can create a long thin triangle

Figure 7.2 shows that if node N_1 is chosen from edge AB in the front then further on in the process, $\triangle AN_1N_2$ will also be formed although it has a very poor measure. The actual procedure is that $\triangle ABN_2$, which does not have as good a measure as $\triangle ABN_1$, but has a better measure than $\triangle AN_1N_2$ would be formed. The result of this is that later on in the procedure $\triangle ABN_1$ will be generated and the net effect is that the sum of the measures of the triangles generated is greater than that of the pair of triangles which is not generated.

The process terminates when the front has no elements in it. A subsequent smoothing step (Laplacian smoothing, Section 6.1) is then employed so that the resulting triangles are more nearly equiangular than those originally generated. The next section details the family of methods which produce "nodes and triangles concurrently".

7.2 Nodes and Triangles Concurrently (Geometrically)

In this section we investigate methods where the boundary nodes are equally spaced around the boundary and the positions of the interior nodes are dependent on the geometric method of generation of the triangles. This can be done in two ways which are detailed below. In all the work that follows, a front is defined as being the collection of edges, taken in an anti-clockwise ordering, which is used to generate new triangles.

7.2.1 Layered

This approach was first presented by Sadek (1980). The boundary of the region is treated as the original front, with the boundary nodes being on layer 1. Nearly equiangular triangles are then generated and any nodes formed during this process which are connected to two nodes on layer n are members of layer $n+1$. Figure 7.3 shows the procedure during the generation of the first layer of triangles. The shaded triangles are about to be generated from the two edges marked 'x'. The numbers indicate which layer each node is on. This production of triangles continues until no edges with nodes on layer n are members of the front, with the restriction that until this occurs no triangles may be generated by an edge with two nodes on layer $n+1$. This process is repeated until there are no edges in the front. The whole procedure is called layered as the triangulation is built up in layers one triangle thick from the boundary into the interior of the region.

The nodes are generated in the following manner :- if the internal boundary

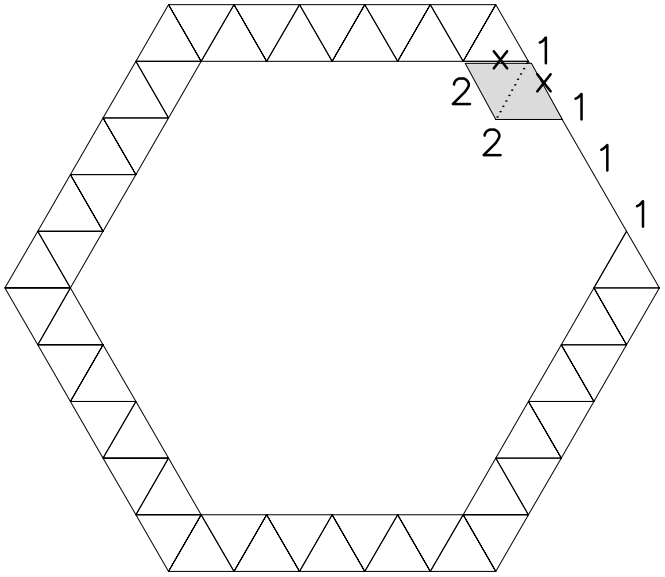

Figure 7.3: Layered triangle generation

angle is $\Phi°$, then the number of triangles generated is the nearest integer to $\frac{3\Phi}{180}$. This results in the generation of nearly equiangular triangles. When the number of triangles to be generated has been found the interior angle is divided into that many equal angles and using the fact that all the triangles should be similar it is possible to position the new node, or nodes. Figure 7.4 shows the procedures which are used to produce the relevant number of triangles and nodes. If the

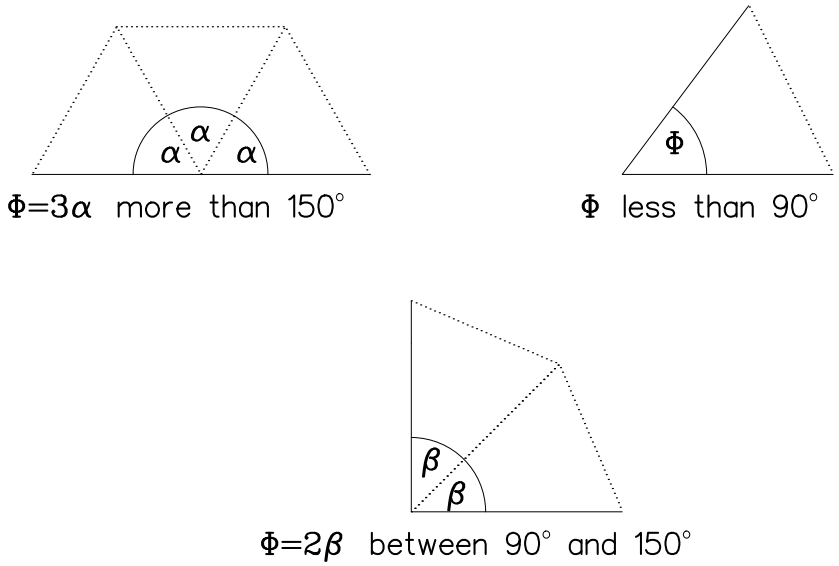

Figure 7.4: How angle Φ dictates number of triangles produced

interior angle is small then, if the size and shape of the triangle which fills the space is acceptable, the gap is closed.

7.2.2 Unlayered

In this approach the order in which triangles and nodes are generated is not restricted by the need to generate layers. Deljouie-Rakhshandeh (1990) used this approach, and generates the triangles in a slightly different manner to Sadek (1980). The number of triangles which are generated is either one, if the interior angle is small, or two, if the angle is larger. In the case of one triangle being produced, (closing), no new node is generated, whilst if two triangles are generated the new node is positioned at the midpoint of the positions where two equilateral triangles would be formed by the enclosing edges. Figure 7.5 shows how a new

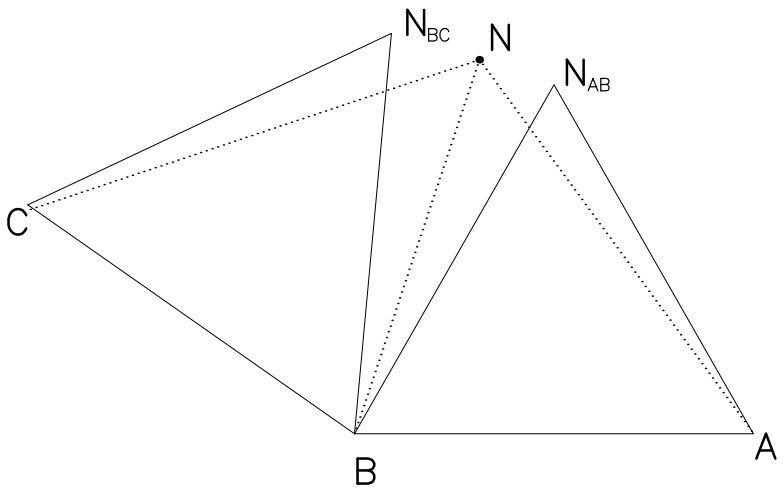

Figure 7.5: Generation of two triangles

point N is created from the edges AB and BC. An equilateral triangle generated from edge AB, shown by solid lines, would generate point N_{AB}, while an equilateral triangle generated from edge BC, shown by solid lines, would generate N_{BC}.

N is then taken as the midpoint of N_{AB} and N_{BC} and the two triangles formed are those which are denoted by the dotted lines. A check is kept of the proximity of existing nodes to new nodes and if they are too close then they coalesce. This method can lead to the generation of more than one front since the order of node generation is not fixed to a layered structure. This front splitting can be seen in Figure 7.6.

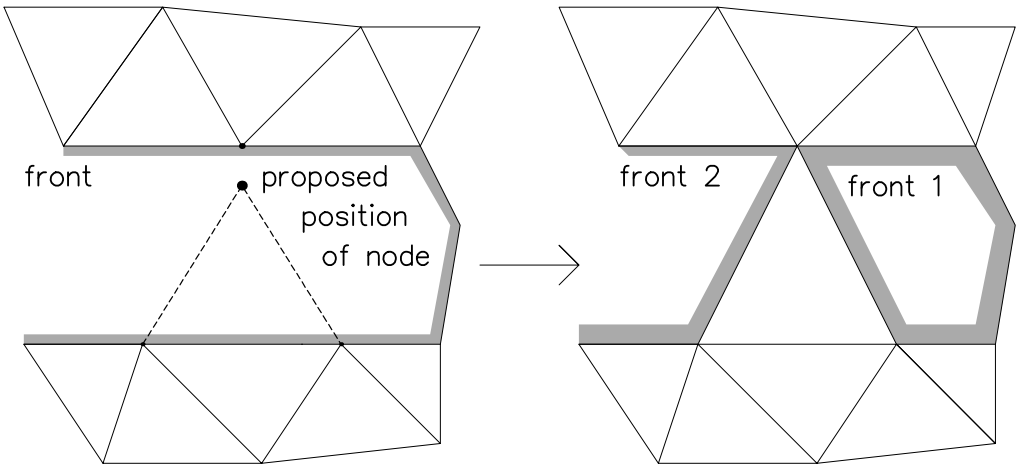

Figure 7.6: Splitting of a front

We now move on to look at the data dependent version of the "nodes and triangles concurrently" procedure since the "interior nodes first" method does not lend itself to a truly data dependent approach, as the positioning of the nodes for data dependent criteria is usually based on the value of a measure calculated along an edge in a triangle which has been generated.

7.3 Nodes and Triangles Concurrently (Data Dependent)

In this procedure all the grid generation information is stored on a nodally sparse background grid (usually triangular) and values over the rest of the region are found by interpolation. This is usually used as a stage of an adaptive solution technique where the values of the solution at the last time interval are used to generate the grid on which the solution at the next time interval will be calculated. The values which are stored on the background grid are :-

- Nodal spacing, δ.
- Stretching parameter, s.
- Direction of stretching, α.

These parameters are connected as shown in Figure 7.7.

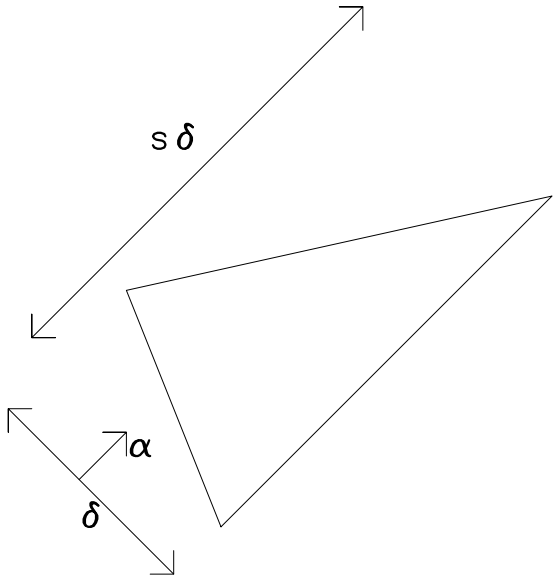

Figure 7.7: How the background parameters relate to each other

Palmerio *et al.* (1990) and Peraire *et al.* (1987) use this method and their procedure is as follows :-

1. Position the boundary nodes such that they satisfy the nodal spacing measure.
2. Choose an edge, AB, to start the generation and find M, its midpoint. The edge AB can be either the smallest edge in the front or the edge with the smallest value of one of the parameters.
3. Find the values of the parameters at M, δ_M, s_M and α_M.
4. Rotate AB so that α_M lies along the x_1-axis and scale down the x_1 coordinates by s_M.
5. Determine δ_1 according to the sizes of δ_M and AB. i.e. if δ_M is too large, or too small, when compared to AB, then δ_1 is set as some multiple of AB; otherwise it is set as δ_M.
6. Construct C at a distance δ_1 from each of A and B.
7. Find all the active nodes within a specified distance of C, and order then in order of distance from C (nearest first).
8. Put C at the top of this list unless $\mathrm{A}N_1 < 1.5 \times \delta_1$ and $\mathrm{B}N_1 < 1.5 \times \delta_1$.
9. The required point is the first point, N_i, such that the interior of $\triangle$ ABN_i does not contain another node and such that the line MN_i does not intersect an existing side on the front.
10. The new element is formed, the coordinates are re-transformed and the front is updated.

11. This procedure continues until there are no edges left in the front at which point the procedure terminates having produced a valid triangulation.

No conditions are made regarding small angles, since it may be necessary to generate small angles to satisfy the background information. A post-processing smoother, based on the solution gradient between points, can be employed to reposition the nodes in the generated triangulation.

Difficulties which have been reported (see e.g. Thomasset (1981)) are that the algorithms may be comparatively fragile and may fail if the boundary nodes are clustered closely together due to the generation of large numbers of triangles. This can be countered by using a "layered" version of the procedure. The other problem is that the program may reach a state in which it cannot generate any more triangles in the interior of the region. This can be countered by changing the values of the proximity tolerances until further triangles can be generated.

Having detailed the triangulation front procedures which have been previously used we now proceed to outline the procedures we used to implement a triangulation front generation program.

7.4 Implementation of Geometric Grid Generation

In order to eventually produce a data dependent triangulation front program we decided to investigate what difficulties might be encountered by originally programming a geometrically based triangulation front program.

The geometrical procedure adopted was as follows :-

1. The boundary nodes were positioned at equi-spaced intervals along the boundary.

2. Find the smallest edge in the front, AB, and create an equilateral triangle, $\triangle ABN_1$, with AB as base and to the left of the front, with new point, N_1. Repeat for both its neighbouring edges, AC and BD, creating points N_2 and N_3 respectively (see Figure 7.8).

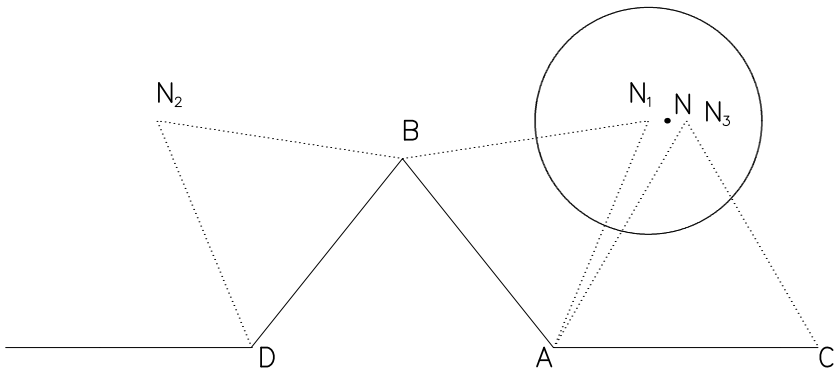

Figure 7.8: Generation of new nodes from edges

3. If either, or both, of N_2 and N_3 are close, i.e. within a prescribed distance $dtol$, of N_1 the corresponding edge is flagged. In Figure 7.8 the distance $dtol$ is shown by the circle.

4. The new point, N, is then the midpoint of the points created by flagged edges.

5. Checks are made to see to see if the new node and triangles are acceptable. These checks include :-

- N is close, less than $ntol$, to an existing node in the front. If this occurs then the triangle is generated and the front splits into two, see Figure 7.6.
- N lies inside an existing triangle. Reject N.
- Existing points lie inside the triangles which are generated. Reject N.
- The edges of the triangles created do not cut any edges in the front. If they do, reject N.
- N is too close to an edge in the front while not being close to its endpoints (see Figure 7.9). Reject N.

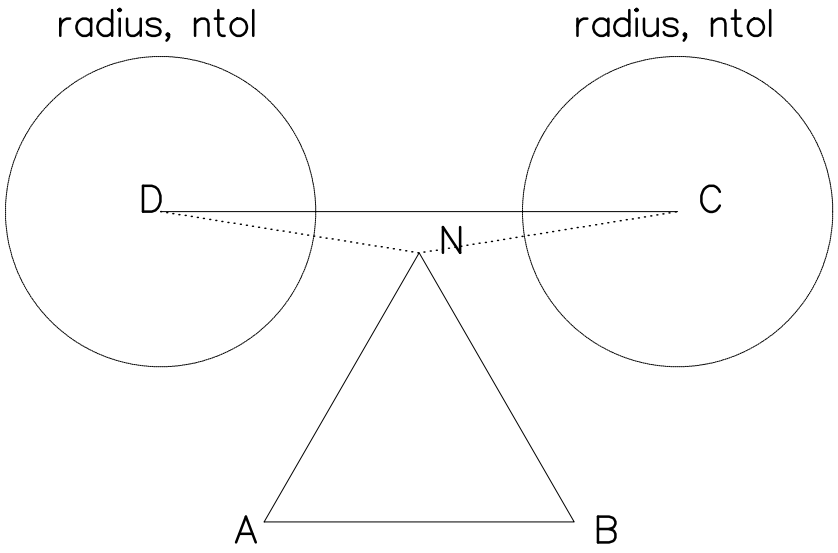

Figure 7.9: N is acceptable but $\triangle$ NCD is not

- Any of the new triangle edges are too close to an existing node (see above). Reject N.

6. If N passes all of these checks it is accepted, the relevant triangles are created concurrently and the front is updated.

7. If small interior angles had been generated by the introduction of the new triangles then these angles were "closed", (see Figure 7.10), provided the triangle generated did not fail any of the checks in the list above.

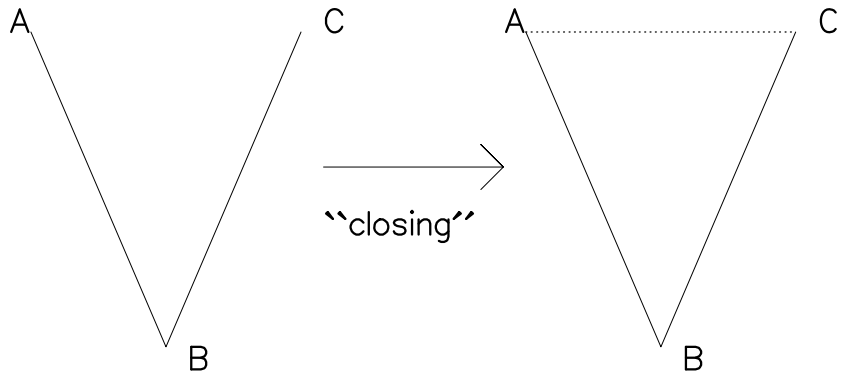

Figure 7.10: Closure of $\angle ABC$ by joining A to C

8. Return to step 2 and repeat the procedure until no edges exist in any generation front.

This outlines the basic procedure and the problems which were encountered during its implementation are detailed in the next section.

7.4.1 Difficulties

In this section the problems which were encountered during the programming and testing of the geometrical triangulation front program are detailed.

1. Taking the smallest edge first will create other small edges so that the generation will start again with the newly created edges as bases and

so if the process continues, the triangles created will spill across the middle of the region and the effect of larger edges will be lost. This can be solved by taking a layered approach to the problem.

2. Tolerances for distances. What is a good choice of $dtol$ and $ntol$?

3. Tolerances for small angles. How small has an angle to be before it is "closed"?

4. If the smallest edges are too small then too many small triangles are generated, while a too large edge can create excessively large triangles. This can be avoided by not using an edge as a generating edge if it is "too large" or "too small". These are usually set as some proportion of the average separation of the boundary nodes.

It was thus necessary to give values to the various tolerances which would decide the final form of the triangulation produced.

The choice of tolerances for distances was a matter of checking which tolerance produced the most equiangular triangulation for the region under consideration. This choice might change for other regions. Finally the choice came down to a proportion of the separation of the boundary nodes. The differences in triangulations produced can be seen in Figures 7.11 to 7.13. The triangulation to be generated was a geometrical triangulation comprising of as many equilateral or nearly equilateral triangles as possible. The production of small angles was to be avoided in this case.

The choice of distance tolerance in Figure 7.11 was rejected since the area of the triangles changed appreciably from the boundary to the centre and small angles were produced. These comments are reinforced by the geometrical measures

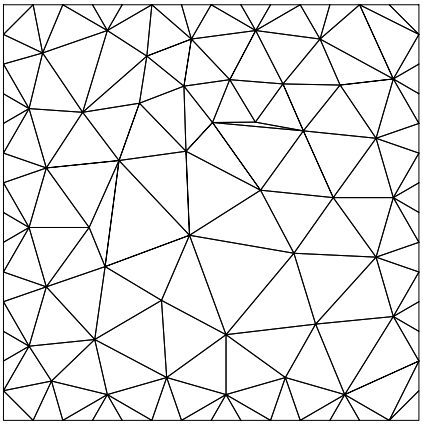

Figure 7.11: $dtol$ = average separation of boundary nodes

presented in Table 7.2 which show the smallest angle as 5° and the maximum ratio of adjacent triangle areas is over 14.

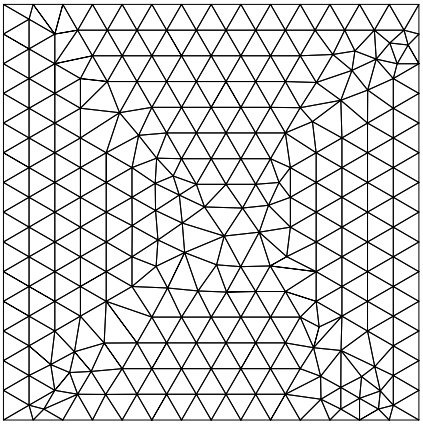

Figure 7.12: $dtol$ = half average separation of boundary nodes

The triangulation shown in Figure 7.12 contains few long, thin triangles and has the most nearly equiangular triangles, the smallest angle is 20°and the maximum skewness in a triangle is 47°. It is thus the most desirable choice of proximity tester.

The triangulation shown in Figure 7.13 has produced a large number of long, thin triangles, the smallest angle is 8°, and the triangles are not of similar size, the maximum uniformity is almost 9, so the proximity tester was rejected.

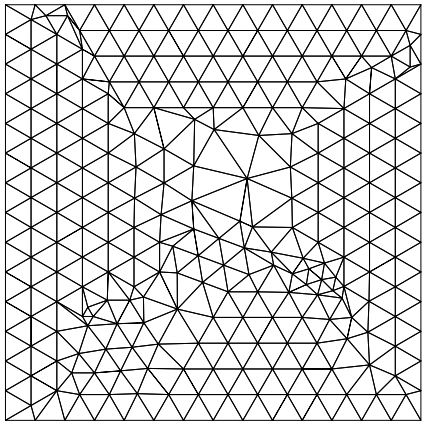

Figure 7.13: $dtol$ = a third of average separation of boundary nodes

The consequences of these results is that the tolerance chosen for the proximity indicator is half the average distance between the boundary nodes. The result of using this tolerance for a different set of boundary nodes is shown in Figure 7.14.

The geometrical triangulations produced had differing numbers of nodes and triangles which are shown in Table 7.1. The geometric measures for the geometric triangulations in Figures 7.11 to 7.14 are shown in Table 7.2 and they highlight which triangulation, and thus which proximity tester, is the best.

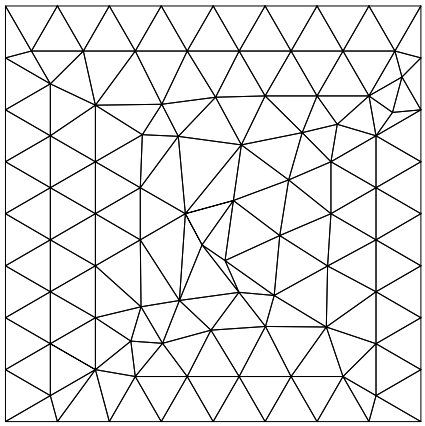

Figure 7.14: Geometrically generated triangulation

The results of the geometrical triangulation front program have been presented and the problems encountered outlined and we now progress to the data dependent triangulation front procedure.

The number of nodes and triangles in geometric triangulations				
Figure	7.11	7.12	7.13	7.14
Nodes	99	265	271	96
Boundary nodes	56	56	56	32
Triangles	140	472	484	158

Table 7.1: The number of nodes and triangles in geometric triangulations

Geometric measures for geometric triangulations				
Figure	Angles	Skewness	AR (Lo)	Uniformity
	min, max	mean, max	mean, min	mean, max
7.14	14.3, 148.0	6.57, 58.7	0.947, 0.317	1.506, 6.120
7.12	19.7, 130.0	4.07, 46.9	0.971, 0.494	1.237, 2.366
7.11	5.25, 168.8	16.1, 72.5	0.864, 0.112	2.366, 14.14
7.13	7.85, 156.2	5.11, 64.1	0.952, 0.217	1.516, 8.830

Table 7.2: Geometric measures for geometric triangulations

7.5 Data Dependent Triangulation Fronts

Having investigated geometrical meshes, and discovered some of the difficulties in implementation, we proceeded with the implementation of data dependent triangulation front procedures. There were two possible implementations :- using a background grid and data dependent background variables, or looking at some measure on the edges of a triangle (akin to the equidistribution nodal reconnection criterion). These choices of data dependent criteria are outlined below.

7.5.1 Background Data

For the first implementation, the choice of background variables had to be decided first. Eventually the choice was $\delta = 1$, and s and α depended on the Hessian, H, of the function at each grid point. (This choice of δ meant we did not mind how large the triangles became).

$$H = P^T \Lambda P$$

where

$$H = \begin{pmatrix} u_{xx} & u_{xy} \\ u_{yx} & u_{yy} \end{pmatrix}$$

$$\Lambda = \begin{pmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{pmatrix}$$

and P is a permutation matrix.

s and α were calculated using

$$s = \sqrt{\left|\frac{\lambda_1}{\lambda_2}\right|}$$

and

$$\alpha = \tan^{-1} \frac{u_{xx} - \lambda_1}{u_{xy}}$$

the latter being the direction of the eigenvector associated with λ_1. This choice of parameters arose from the nodal reconnection work of D'Azevado and Simpson (1989) where the coordinate transformation depended on the square root of the ratio of the functions' second derivatives.

Geometrical grid generation can be thought of as this procedure with $s = 1$ and $\alpha = 0$.

Difficulties

The difficulties which occurred with these parameters are listed below.

- The size of s has to be checked and an upper bound set, since if it becomes too large the points generated might lie outside the region. This problem also occurs if $\lambda_2 = 0$, as s is then not defined.

- The calculation of α can also be difficult in certain cases, e.g. where $u_{xy} = 0$.

- If large values of s occur and the front splits into two, there is no guarantee that any node generated will lie in the un-triangulated region enclosed by the front containing the generating edge.

We now move to the other data dependent triangulation front procedure which was tested.

7.5.2 Cost of Triangle

The other data dependent implementation was an extension of the nodal reconnection criterion, equidistribution. In this procedure some measure was calculated on each edge in the front, the edge with the smallest measure was used as a base and an equilateral triangle was created. The cost of this triangle, the sum of the measures, was calculated and the node moved until the cost of the triangle was within specified tolerances. If this did not occur before the triangle became too big or too small, no node was positioned and the edge with the next smallest measure was considered. "Nodes and triangles concurrently" geometrical grid generation can be thought of as this procedure with each edge automatically having a cost of 1 and the tolerance being 3.

The boundary nodes for this procedure were placed by equidistributing the measure on each edge over the whole boundary.

Difficulties

The difficulties encountered in this procedure are detailed below.

- Calculating the triangle cost tolerances. If this is done by looking at the cost on boundary edges then it is possible that if the costs increase enough inside the region, then the triangles in the interior will become smaller and smaller until the program generates too many exceedingly small triangles and never terminates. If no triangles are generated or they become too small then the tolerances need to be changed.

- The effect of closing small angles to produce triangles has to be studied to make sure that the cost of the triangle is within the tolerances allowed.

7.6 Use of Generated Nodal Sets

In previous chapters the effect of reconnecting a fixed set of nodes and of repositioning the nodes with a fixed connectivity has been investigated. However both of these require an initial set of nodes. These nodes have until now been randomly generated or generated in a geometric manner. However for a truly data dependent grid generation procedure the set of nodes should have been generated in a data dependent manner. The set of nodes generated by a triangulation front procedure have been produced in a data dependent manner but the position of the nodes may be non-optimal due to the local nature of the generation procedure. This can be corrected by the use of a global movement procedure such as was investigated in Chapter 6. However this movement may have resulted in a non-optimal connectivity and a local reconnection of the connectivity will result in a good data dependent grid.

The triangulations generated by the procedures can hence be used as the basis for a data dependent grid procedure where the nodes are generated, repositioned according to a data dependent criterion and then reconnected using a data dependent criterion.

In the next section we present the results which show the differences in triangulations which the different data dependent triangulation front implementations produce. Also presented are the results of using the produced triangulations as an initial grid for a complete data dependent triangulation procedure.

7.7 Results

In the section that follows, the results of the triangulation front procedures are presented. The generated grids are then used as the initial part of a data dependent grid generation procedure in which the nodes are repositioned first and then reconnected afterwards. To aid the clarity of results the three triangulations generated for each function, triangulation front, nodal repositioning and nodal reconnection are presented together as sets of three pictures. After each set of pictures the numerical results are presented in a table, so a direct comparison between the grids generated is possible. Unless specifically indicated, the grids are oriented with $(0, 0)$ in the bottom left hand corner.

7.7.1 Background Grid

The results shown here are for the background grid data dependent triangulation front program. The distance tolerances were all set at half the average spacing between nodes, while the maximum value of s allowed was 20.

The results are shown for many of the functions in Chapter 4. The results are presented here in the following fashion :- The triangulation produced by the triangulation front program is shown first, the result of moving the nodes using the method outlined in Section 6.1 is then shown and the last picture shows the result of reconnecting the moved nodes using the ABN-2 method as outlined in Section 5.3.3. The numerical results are presented in the form of the errors first and then the geometrical measures.

The first results are those for the function P2.

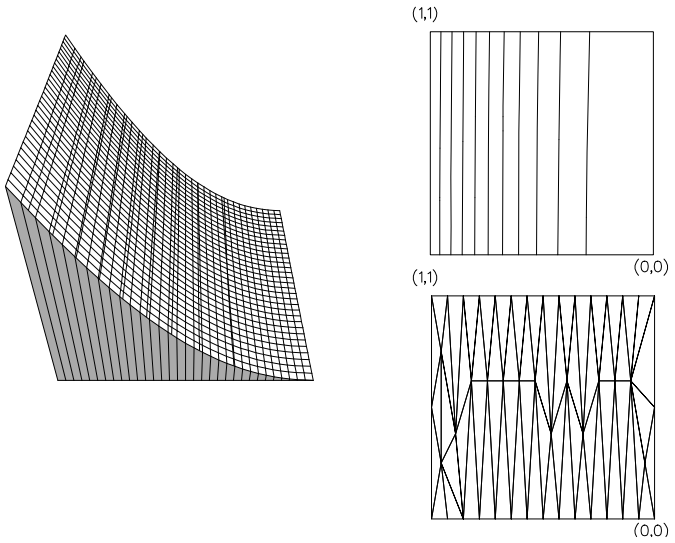

Figure 7.15: Triangulation produced by triangulation front for P2

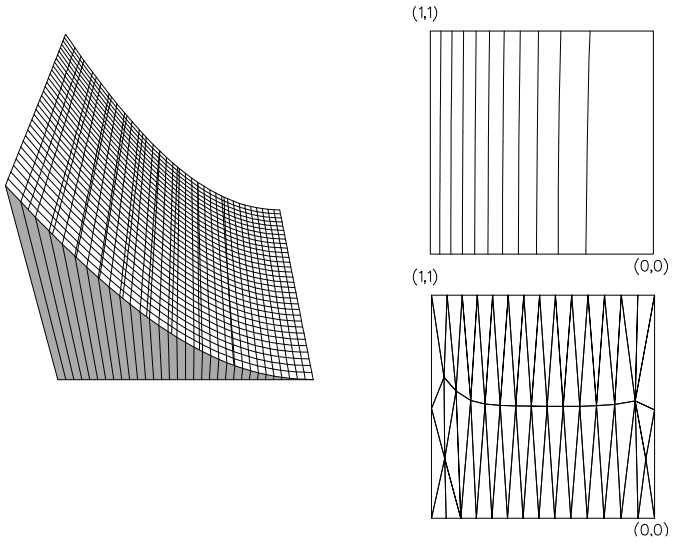

Figure 7.16: Result after nodal movement on Figure 7.15

The triangulation produced by the triangulation front program contains long, thin triangles while the nodes move to a central position after the nodal movement as expected. The triangulation produced by reconnecting produces slightly smaller errors than either of the preceding triangulations as can be seen in Table 7.3.

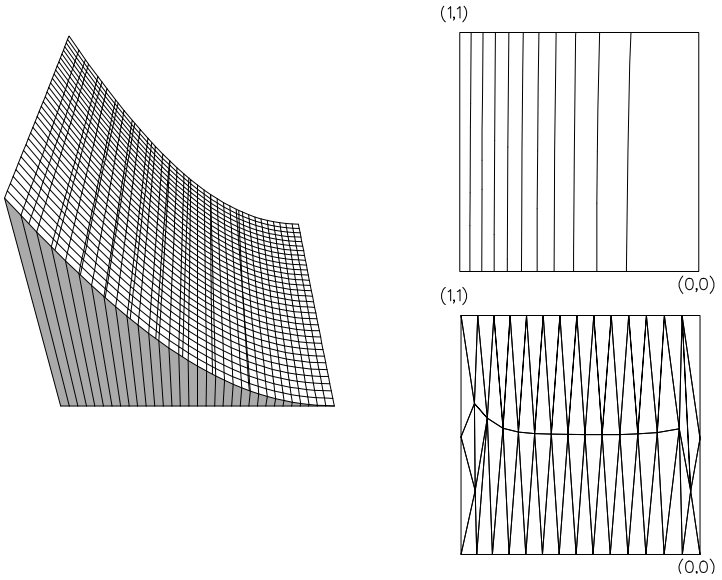

Figure 7.17: Result of reconnecting nodes in Figure 7.16

Errors for P2				
Method	L_2 – error	mean interpolation	max interpolation	Figure
Triangulation front	0.1234	0.1108	0.3151	7.15
Nodes moved	0.1137	0.1038	0.2326	7.16
Nodes reconnected	0.1125	0.1030	0.2158	7.17
Geometric measures for P2				
Method	Angles	Skewness	AR (Lo)	Figure
	min, max	mean, max	mean, min	
Triangulation front	6.52, 163.7	43.3, 69.2	0.264, 0.155	7.15
Nodes moved	6.84, 162.3	37.6, 68.2	0.263, 0.178	7.16
Nodes reconnected	4.10, 169.2	39.1, 72.8	0.247, 0.103	7.17
Number of boundary nodes			32	
Number of nodes			47	
Number of triangles			60	

Table 7.3: Errors and measures for P2

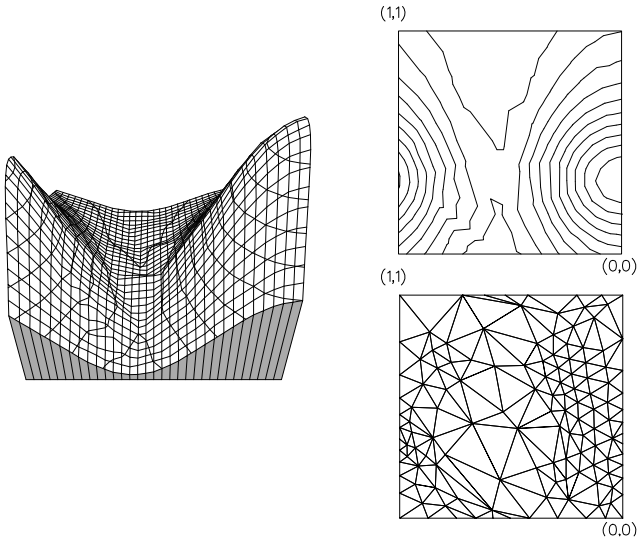

Figure 7.18: Triangulation produced by triangulation front for M2

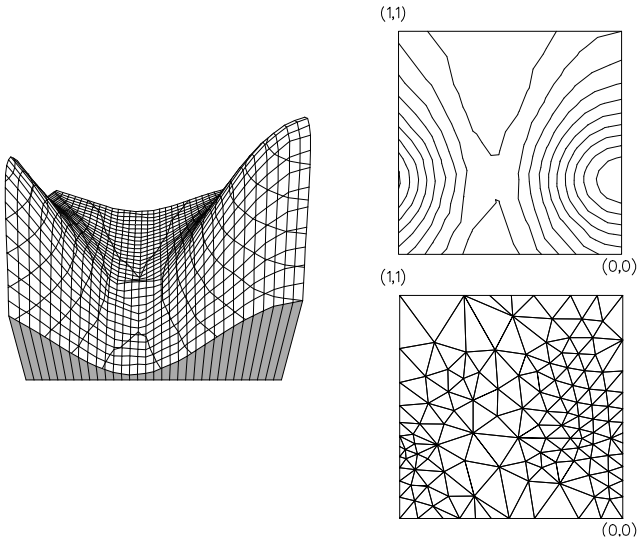

Figure 7.19: Result after nodal movement on Figure 7.18

The grid for M2 produced by the triangulation front contains small triangles where the function changes rapidly and large triangles where it is flat. After the nodal movement the nodes are redistributed so that all the rapidly changing regions are well provided with points. The grid produced by nodal reconnection aligns the triangles with the contours and reproduces the valley between the hills better. These comments are reinforced by the results shown in Table 7.4.

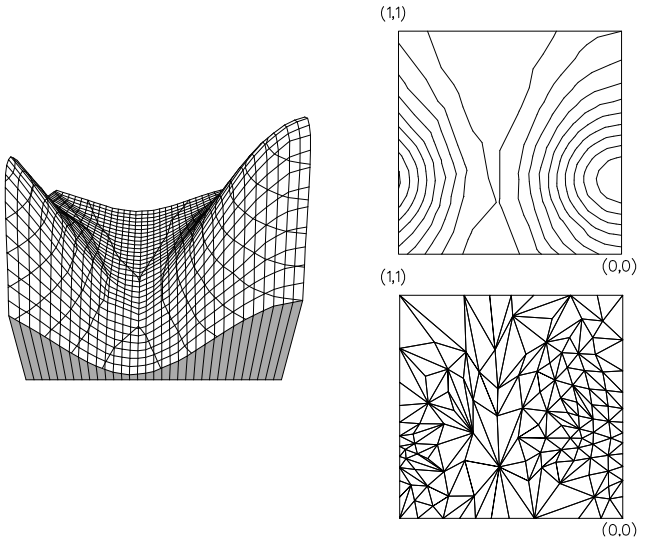

Figure 7.20: Result of reconnecting nodes in Figure 7.19

Errors for M2				
Method	L_2 – error	mean interpolation	max interpolation	Figure
	10^{-3}	10^{-3}	10^{-2}	
Triangulation front	5.9695	3.3736	3.0790	7.18
Nodes moved	3.3177	2.1831	2.2649	7.19
Nodes reconnected	2.3605	1.6704	1.2339	7.20
Geometric measures for M2				
Method	Angles	Skewness	AR (Lo)	Figure
	min, max	mean, max	mean, min	
Triangulation front	5.77, 164.2	15.6, 69.5	0.860, 0.144	7.18
Nodes moved	18.9, 137.8	12.6, 51.8	0.908, 0.418	7.19
Nodes reconnected	4.90, 166.0	26.2, 70.6	0.735, 0.125	7.20
Number of boundary nodes			40	
Number of nodes			131	
Number of triangles			220	

Table 7.4: Errors and measures for M2

The next results are those for the Function SR1.

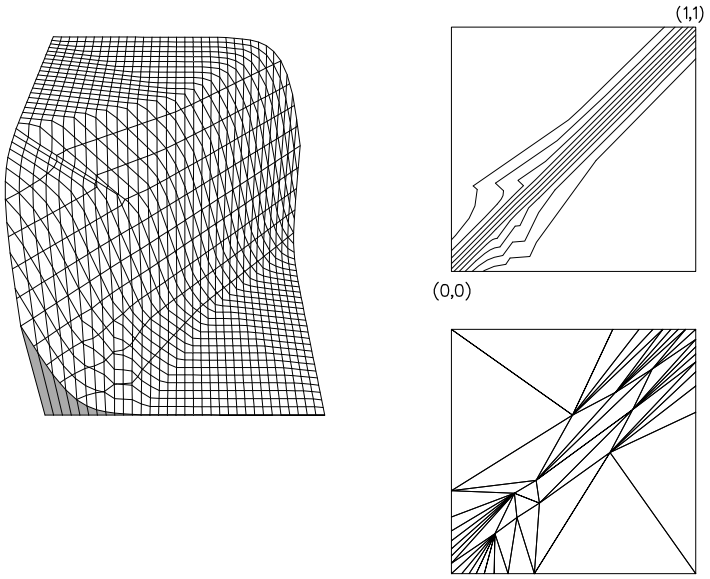

Figure 7.21: Triangulation produced by triangulation front for SR1

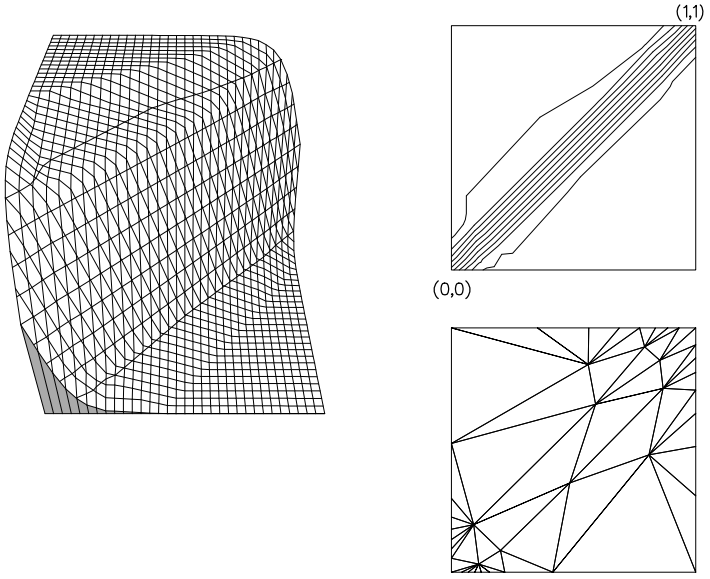

Figure 7.22: Result after nodal movement on Figure 7.21

The grid produced by the triangulation front contains long, thin triangles along the contours. The movement of the nodes separates the nodes and reduces the number of long, thin triangles. The triangulation produced by reconnecting the nodes again aligns the triangles with the contours although there are minor faults in the contours produced. The numerical results are shown in Table 7.5.

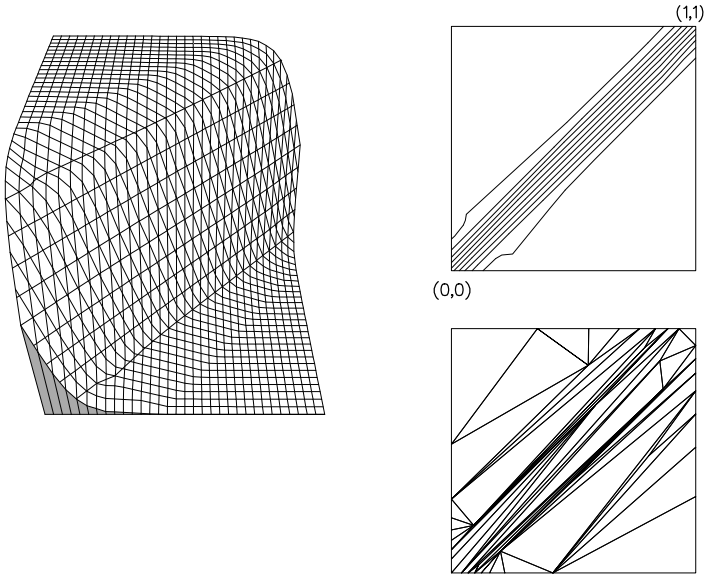

Figure 7.23: Result of reconnecting nodes in Figure 7.22

Errors for SR1				
Method	L_2 – error	mean interpolation	max interpolation	Figure
	10^{-3}	10^{-3}	10^{-2}	
Triangulation front	7.3280	4.7033	3.2832	7.21
Nodes moved	6.7710	4.2036	1.9292	7.22
Nodes reconnected	3.4772	1.9163	1.2681	7.23

Geometric measures for SR1				
Method	Angles	Skewness	AR (Lo)	Figure
	min, max	mean, max	mean, min	
Triangulation front	3.15, 164.2	41.3, 69.5	0.356, 0.095	7.21
Nodes moved	8.64, 154.7	31.8, 63.2	0.611, 0.222	7.22
Nodes reconnected	0.29, 178.9	48.0, 79.2	0.315, 0.009	7.23

Number of boundary nodes	32
Number of nodes	42
Number of triangles	50

Table 7.5: Errors and measures for SR1

The final two sets of figures are both associated with the Function DD1 and show the difference that the number of boundary nodes can make to a final triangulation after movement and reconnection.

The next results are those for the Function DD1 with an average of 9 points per boundary.

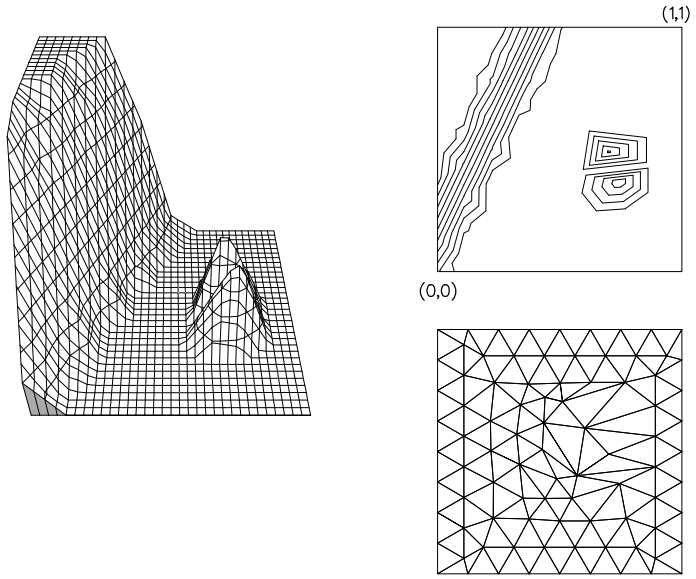

Figure 7.24: Triangulation produced by triangulation front for DD1

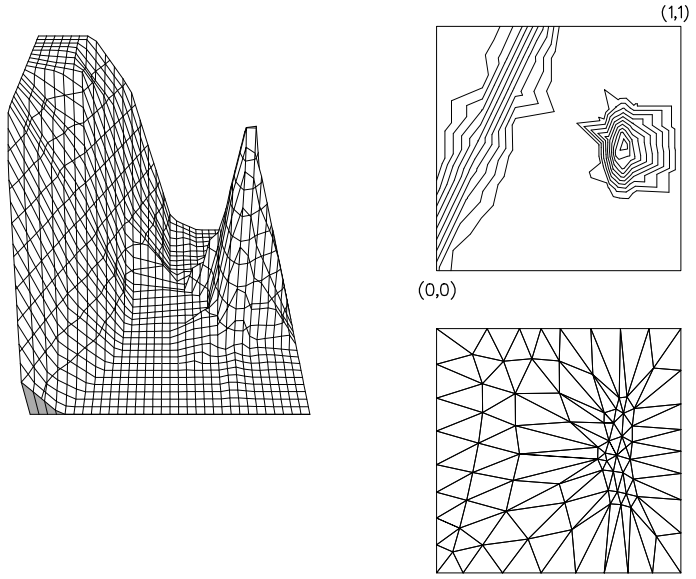

Figure 7.25: Result after nodal movement on Figure 7.24

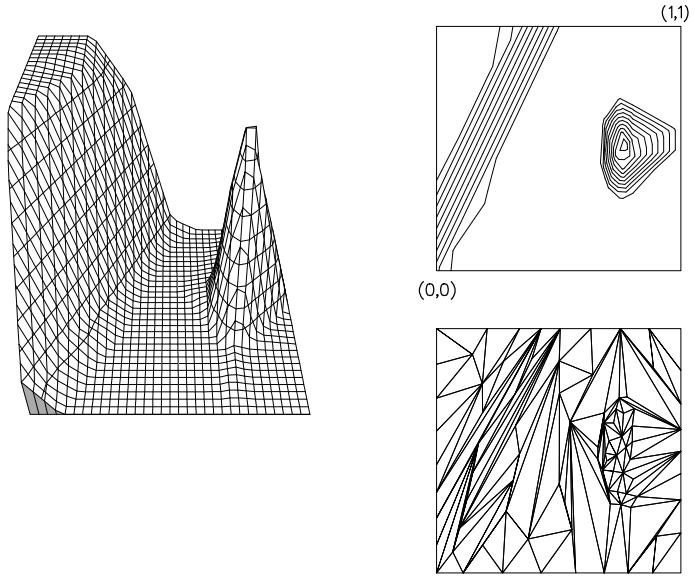

Figure 7.26: Result of reconnecting nodes in Figure 7.25

Errors for DD1				
Method	L_2 – error	mean interpolation	max interpolation	Figure
	10^{-2}	10^{-2}		
Triangulation front	10.424	3.3043	0.7495	7.24
Nodes moved	7.8217	3.7625	0.4785	7.25
Nodes reconnected	5.4309	1.6040	0.4767	7.26

Geometric measures for DD1				
Method	Angles	Skewness	AR (Lo)	Figure
	min, max	mean, max	mean, min	
Triangulation front	17.0, 145.6	7.54, 57.1	0.935, 0.346	7.24
Nodes moved	6.58, 143.4	20.5, 55.6	0.764, 0.197	7.25
Nodes reconnected	0.13, 179.7	44.3, 79.8	0.445, 0.003	7.26

Number of boundary nodes	32
Number of nodes	88
Number of triangles	142

Table 7.6: Errors and measures for DD1

The next results are those for the Function DD1 with an average of 13 points per boundary.

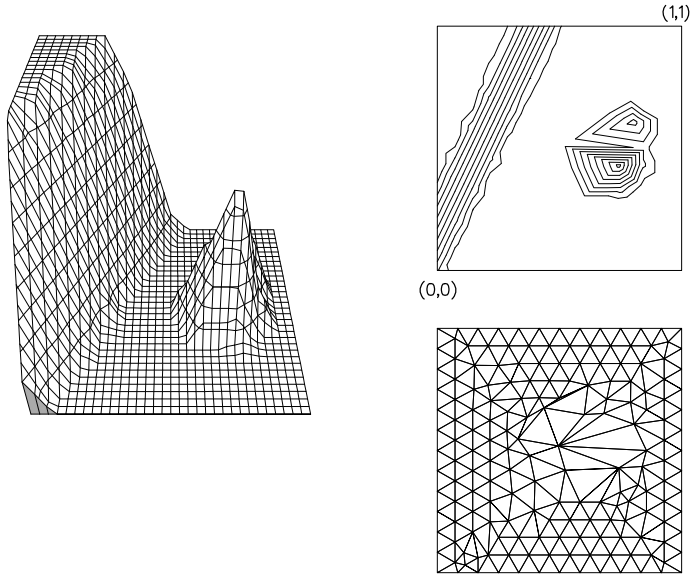

Figure 7.27: Triangulation produced by triangulation front for DD1

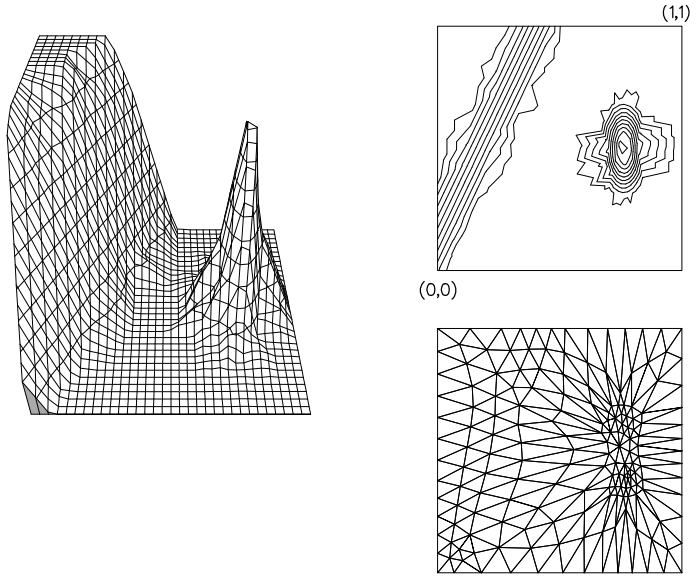

Figure 7.28: Result after nodal movement on Figure 7.27

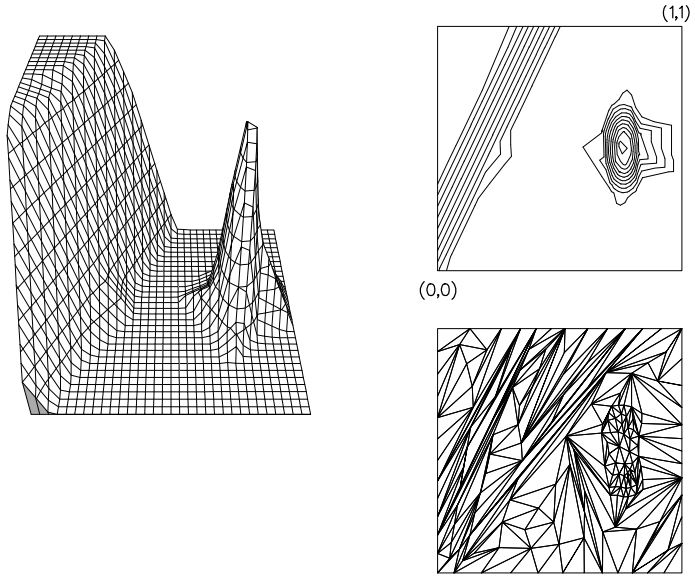

Figure 7.29: Result of reconnecting nodes in Figure 7.28

Errors for DD1				
Method	L_2 – error	mean interpolation	max interpolation	Figure
	10^{-2}	10^{-2}		
Triangulation front	9.5242	2.6033	0.8515	7.27
Nodes moved	5.0812	2.1580	0.3447	7.28
Nodes reconnected	3.7058	1.1217	0.3239	7.29

Geometric measures for DD1				
Method	Angles	Skewness	AR (Lo)	Figure
	min, max	mean, max	mean, min	
Triangulation front	2.90, 171.9	6.43, 74.6	0.944, 0.073	7.27
Nodes moved	10.1, 147.4	21.2, 58.3	0.756, 0.269	7.28
Nodes reconnected	0.02, 179.9	45.4, 79.9	0.439, 0.0006	7.29

Number of boundary nodes	48
Number of nodes	175
Number of triangles	300

Table 7.7: Errors and measures for DD1

The two sets of results show the differences that can occur with different numbers of boundary nodes. In both cases the results have the same form but the degree of misrepresentation varies. The triangulation produced by the triangulation front program poorly represents the data function by either splitting, or almost splitting, the mountain in half due to the stretched triangles near the mountain. The movement of the nodes moves the nodes towards the mountain but in so doing the representation of the ramp degenerates. The triangulation produced by reconnecting the nodes restores the representation of the ramp and significantly improves the representation of the mountain. The numerical results for both the sets are shown in Tables 7.6 and 7.7.

It is necessary to report that this procedure fails for the Function SR3 due to the combination of the positioning of the boundary nodes and the magnitude and direction of s. For the other functions, valid triangulations are produced which represent the underlying functions moderately well but the effect of moving and reconnecting the nodes is minimal.

Having presented the results produced by the background grid we now show the grid produced by the triangle cost procedure.

7.7.2 Cost of Triangles

Before the results of using the cost of triangles triangulation front procedure are presented it is first necessary to actually detail the measure which will be used to generate the cost of each edge in the triangulation. The measure actually used was analogous to that used in the spring analogy nodal reconnection procedure. The cost of a triangle edge was calculated as the length of the edge multiplied by

the spring constant k_j given in (6.12) with $\alpha = 10$ and $\beta = 2$. This complements the nodal movement criterion. The results for this implementation are presented in the same form as the results for the background grid implementation. The triangulation front produced triangulation is shown first, followed by the triangulation produced by repositioning the nodes and finally the triangulation produced by nodal reconnection is shown. The numerical results are then shown in a table which shows the errors associated with the triangulations and the geometrical measures associated with the same triangulations.

This is the set of results for the Function P2 generated with 9 nodes per boundary.

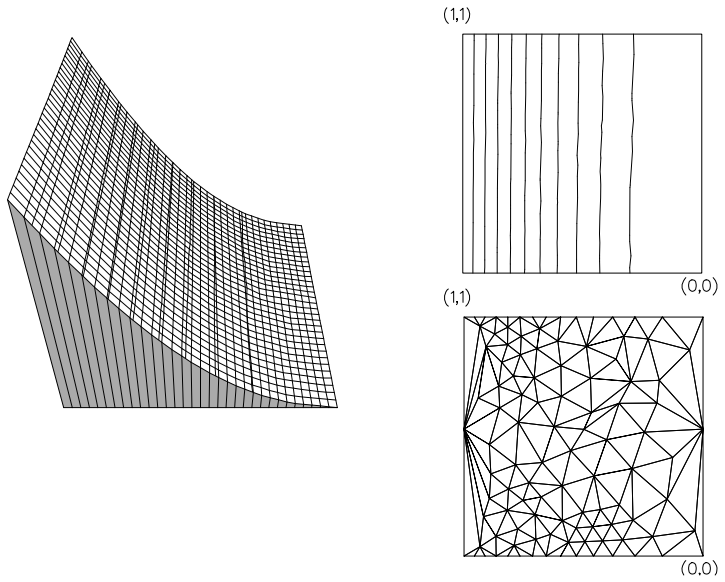

Figure 7.30: Triangulation produced by triangulation front for P2

Figures 7.30 to 7.32 show the triangulations generated for the Function P2. Figure 7.30 shows the poor triangulation produced by the triangle cost triangulation front program. The results of Rippa (1991) and D'Azevado and Simpson prove that the triangulation should be comprised of long, thin triangles. The triangulation shown here has almost no long, thin triangles in the direction of the

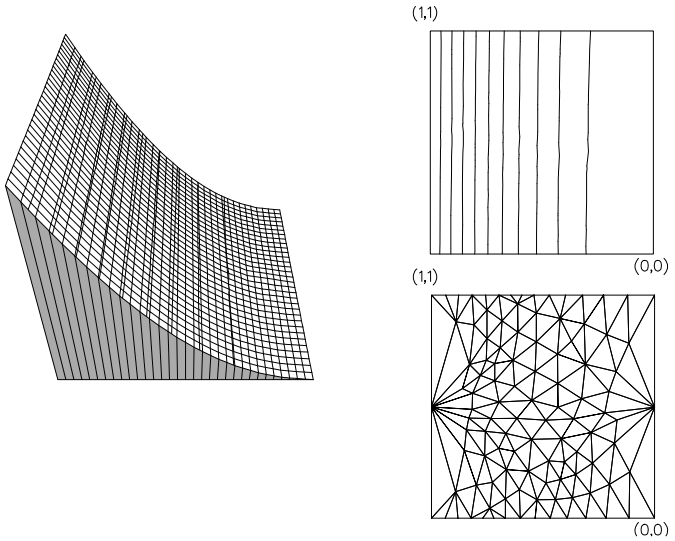

Figure 7.31: Result after nodal movement on Figure 7.30

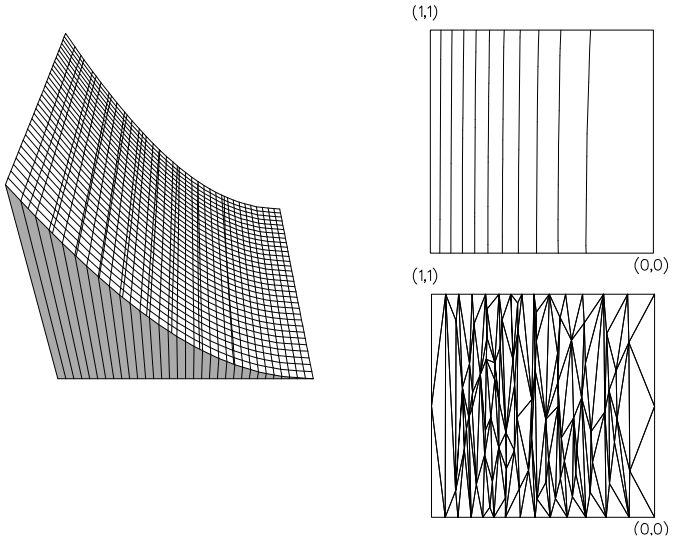

Figure 7.32: Result of reconnecting nodes in Figure 7.31

contours. The result of moving and then reconnecting the nodes is to introduce long, thin triangles but even this does not significantly improve the errors. This can be seen by comparing the numerical errors in Table 7.3 with those in Table 7.8. Even though the triangulation produced by the triangle cost program has twice as many nodes and three times as many triangles as the background grid produced triangulation it does not manage to attain the same L_2-, or maximum interpolation, error as the background grid program produced grid.

Errors for P2				
Method	L_2 – error	mean interpolation	max interpolation	Figure
Triangulation front	0.3119	0.2288	1.4508	7.30
Nodes moved	0.2572	0.1862	1.1639	7.31
Nodes reconnected	0.1239	0.0897	0.3688	7.32

Geometric measures for P2				
Method	Angles	Skewness	AR (Lo)	Figure
	min, max	mean, max	mean, min	
Triangulation front	4.59, 164.6	15.0, 69.7	0.858, 0.132	7.30
Nodes moved	12.1, 137.7	17.6, 51.8	0.835, 0.338	7.31
Nodes reconnected	1.02, 177.4	57.3, 78.2	0.283, 0.024	7.32

Number of boundary nodes	32
Number of nodes	106
Number of triangles	178

Table 7.8: Errors and measures for P2

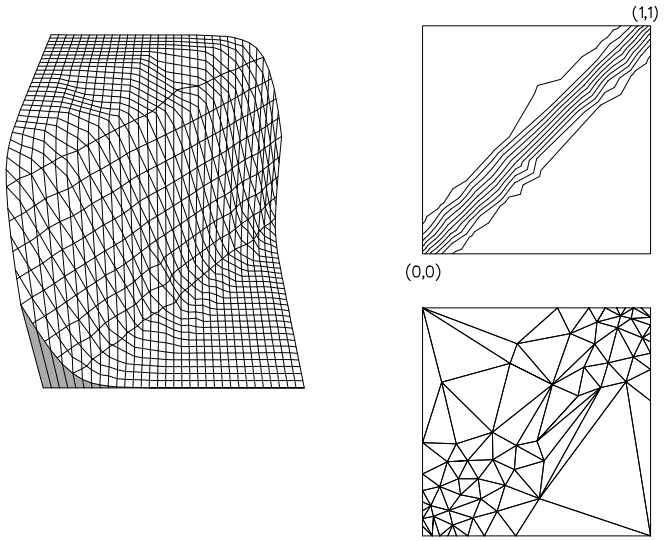

Figure 7.33: Triangulation produced by triangulation front for SR1

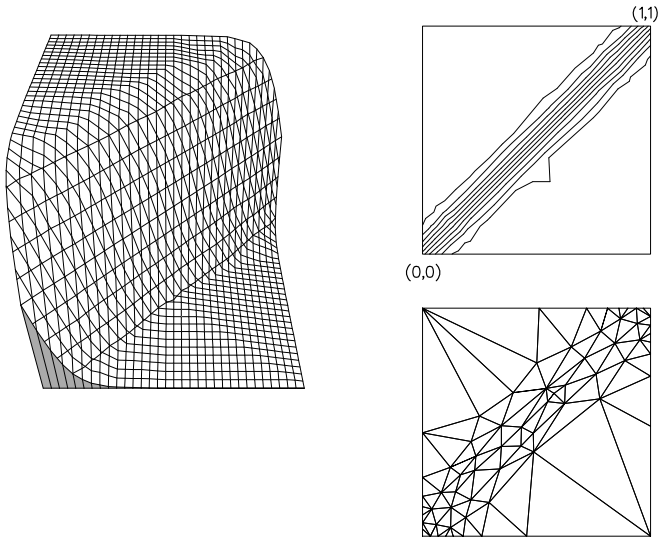

Figure 7.34: Result after nodal movement on Figure 7.33

Figures 7.33 to 7.35 show the grids for SR1. They show the regular triangles produced by the triangulation front and the long, thin triangles produced after the nodal movement and reconnection. Table 7.9 shows the improvement in the errors created by the long, thin triangles. It can be noted that the triangle cost grid has twice as many triangles and nodes as the background grid generated mesh.

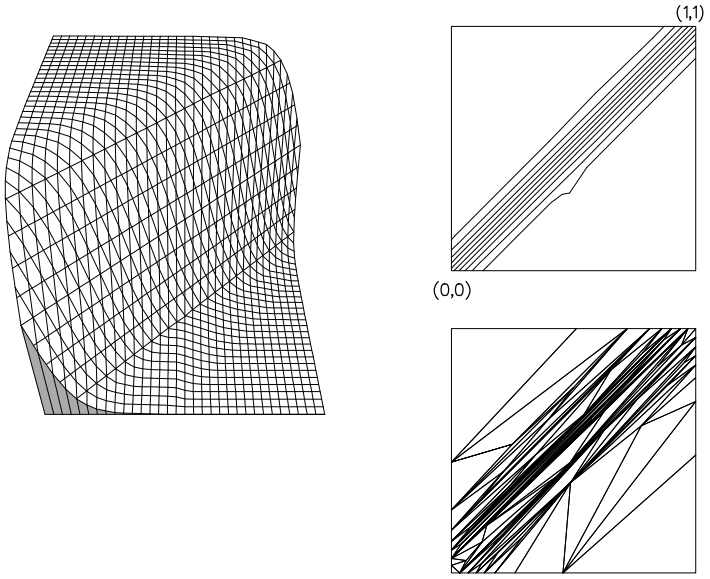

Figure 7.35: Result of reconnecting nodes in Figure 7.34

Errors for SR1				
Method	L_2 – error	mean interpolation	max interpolation	Figure
	10^{-3}	10^{-3}	10^{-2}	
Triangulation front	4.8407	2.9298	2.0421	7.33
Nodes moved	5.2418	3.5206	1.7417	7.34
Nodes reconnected	2.1561	0.9181	1.1746	7.35

Geometric measures for SR1				
Method	Angles	Skewness	AR (Lo)	Figure
	min, max	mean, max	mean, min	
Triangulation front	7.56, 151.9	18.5, 61.2	0.809, 0.214	7.33
Nodes moved	13.1, 135.0	21.5, 50.0	0.783, 0.365	7.34
Nodes reconnected	0.10, 179.6	63.8, 79.7	0.175, 0.003	7.35

Number of boundary nodes	32
Number of nodes	71
Number of triangles	108

Table 7.9: Errors and measures for SR1

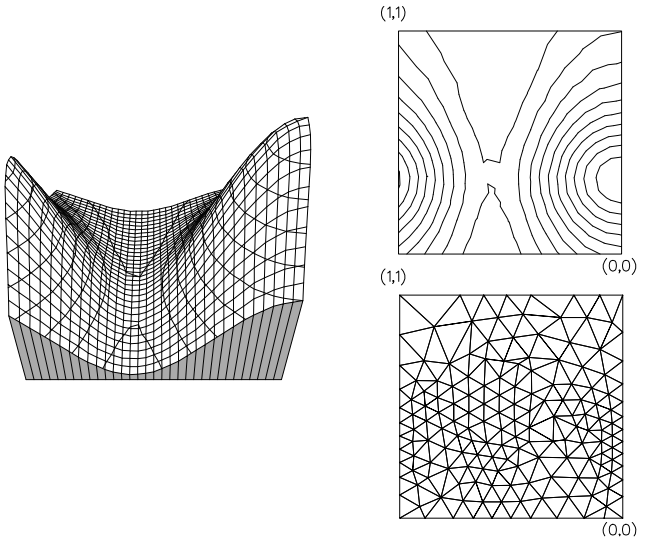

Figure 7.36: Triangulation produced by triangulation front for M2

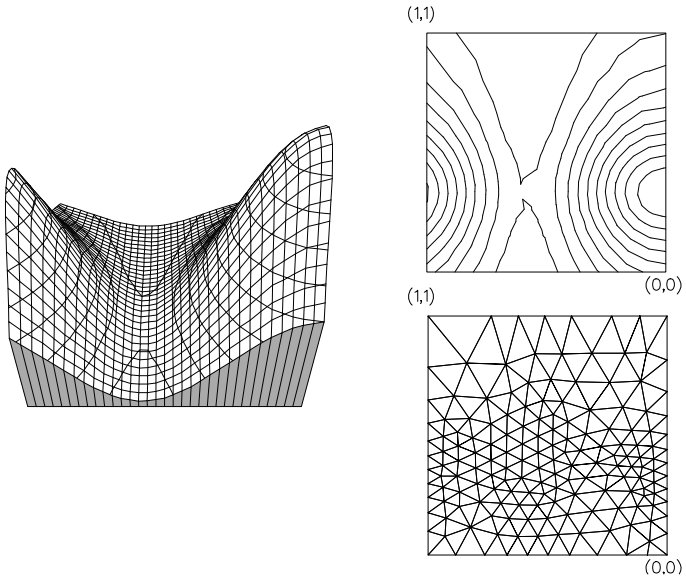

Figure 7.37: Result after nodal movement on Figure 7.36

Figures 7.36 to 7.38 show the results of the three procedures. The numerical results in Table 7.10 show that the operations of moving and reconnecting the nodes produced by the triangulation front program improve the errors of the representations of the underlying function.

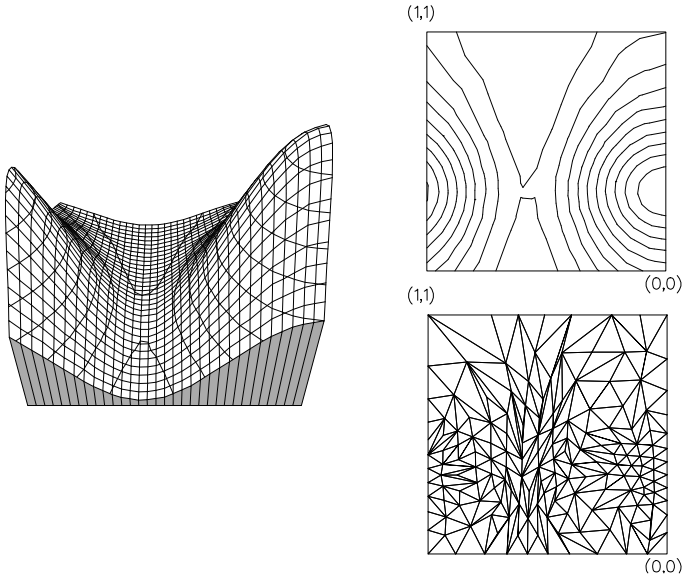

Figure 7.38: Result of reconnecting nodes in Figure 7.37

Errors for M2				
Method	L_2 – error	mean interpolation	max interpolation	Figure
	10^{-3}	10^{-3}	10^{-3}	
Triangulation front	1.5093	1.2038	5.5684	7.36
Nodes moved	1.4902	1.2381	4.8659	7.37
Nodes reconnected	1.2682	1.0268	4.3529	7.38
Geometric measures for M2				
Method	Angles	Skewness	AR (Lo)	Figure
	min, max	mean, max	mean, min	
Triangulation front	7.28, 136.7	7.3, 51.2	0.952, 0.427	7.36
Nodes moved	29.4, 99.5	7.9, 26.4	0.959, 0.731	7.37
Nodes reconnected	1.85, 175.9	28.6, 77.3	0.704, 0.041	7.38
Number of boundary nodes			40	
Number of nodes			167	
Number of triangles			292	

Table 7.10: Errors and measures for M2

This procedure actually worked for the Function SR3 although it produced poor results, but did not work for the Functions DD1, SH1 and SH2. The results for the functions which are not shown are similar to those shown here, e.g. they include many nearly equilateral triangles.

The conclusion of comparing the two procedures is that the triangle cost procedure requires more triangles to triangulate the same region with the same boundary nodes as the background grid procedure.

We now move on to consider how the introduction of geometrical constraints can be implemented in the data dependent grid generation procedure.

7.8 Geometric Constraints

As has been mentioned before, the purpose of introducing a data dependent triangulation front program was to apply the grids generated as the initial triangulation of a data dependent grid generation procedure. The main purpose of the triangulation front was to position nodes where they are required to give a desirable resolution. This might mean that there should be a large number of nodes in a small region thus generating small triangles while the stretching factors might require the introduction of small angles. For these reasons when geometric constraints are introduced, they are not applied in the triangulation front program at any point. They are however applied to the latter stages of the data dependent procedure when the nodes are moved and reconnected. Such constraints were applied but in all the cases tested the numerical errors showed that the solutions were not as good as those produced by the unconstrained procedure.

We have now detailed some of the techniques which can be applied to the prob-

lem of triangular grid generation. In the next chapter the work is summarised, recommendations on the strategies implemented are presented and possible areas of further study are detailed.

Chapter 8

Summary and Further Work

We now summarise the work of this thesis and the possible areas where the work may be taken further. In Chapters 2 and 3 we introduced and described the methods currently used to find geometric and data dependent grids. The different techniques which are used for the generation of both quadrilateral and triangular grids were reviewed.

Chapter 4 presented the functions and data sets upon which the methods detailed in the later chapters would be tested. The measures which would be used to judge the quality of a triangulation were also detailed. Some of these measures were calculations of some error in the representation of the underlying function while others were based on the geometrical properties of the triangulation.

In Chapter 5 we detailed the data dependent techniques which can be used to implement nodal reconnection in a triangulation. Techniques based on the interpolating polynomials on neighbouring triangles were presented, as was a technique based on the the cost of a measure on a line joining two nodes.

Chapter 6 detailed the data dependent methods which can be used to reposi-

tion the nodes in a triangulation while keeping a fixed connectivity. The extension of the spring analogy technique to data representation was detailed and results were presented. A technique based on error estimates was also outlined.

In Chapter 7, a procedure to generate the positions of nodes in a data dependent manner was outlined, based on the triangulation front technique which generates triangles concurrently with nodes. The result is a data dependent triangulation, however it was discovered that improved representation could be achieved using smoothing and reconnection on this grid. The use of such triangulations as the basis for a complete data dependent triangulation procedure was investigated and results were presented.

By studying the results in Chapters 5-7 it is possible to make recommendations on the choice of procedures which might improve data representation on triangulations. The results in Chapter 5 indicate that for most functions, for a given set of nodes, nodal reconnection results in an improvement in the representation of the underlying data. Thus it is possible to recommend nodal reconnection as part of a grid generation procedure. Of the methods tested in this thesis the best overall choice of nodal reconnection criterion was the Angle Between Normals criterion with the vectors ordered using the 2-norm, ABN-2.

The motivation behind nodal movement is that the data points might be poorly positioned and that by moving them to areas of interest, i.e. transient solution features, the data, or solution, will be better represented. By reviewing the results in Chapter 6 it is possible to see that in most cases the effect of repositioning nodes while maintaining a fixed connectivity results in improvements in the function representation. The procedure that is most effective is the spring

analogy criterion. Although it is possible to modify the parameters in (6.12) if some prior knowledge of the function is available, the general choice of parameters is acceptable. Nodal movement can therefore also be recommended as part of a grid generation procedure.

Since nodal movement will be, to some degree, constrained by the connectivity, it is generally advisable to perform a reconnection after such movement, as was illustrated in Chapter 6.

In Chapter 7 the results show that the background grid procedure for generating the positions of nodes in a data dependent manner produces reasonable nodal positions for the functions on which the procedure was tested. The procedure is therefore recommended as one which will produce a data dependent set of nodal positions with a data dependent nodal connectivity. It is obvious that such a technique should be used as the original step in a data dependent grid generation procedure and, with the addition of the techniques recommended in Chapter 6, a complete strategy can be developed. This strategy, which will result in better representations for most functions, is to generate the nodes in a data dependent manner, then to reposition them using a data dependent criterion and finally to reconnect the nodes using a data dependent technique.

Although this work is carried out in the context of the generation of a grid to well represent initial data it has applications in the more general area of function interpolation.

Having presented the recommendations for the grid generation procedure it is now possible to outline possible avenues for further work. One avenue of research is to make a further investigation of other choices of spring constant and

background grid variables. Although the suggested choice of parameters in (6.12) produces reasonable results it was shown in Chapter 6 that if the parameters are specifically chosen for each function, the results can be improved. A similar situation exists with regard to the choice of variables for the background grid triangulation front procedure. The automation of such choices is a possible area for further work.

Other avenues for research are to study the effect of refining the grid at some stage during the grid generation procedure, and to search for acceptable criteria for refining the grids. It would also be of interest to investigate the possible application of the methods presented here to quadrilateral grid generation and their extension to 3-dimensional grid generation.

References

C.M. Ablow and S. Schechter, (1978), *Campylotropic Coordinates*, J. Comp. Phys. **27**, pp 351-362

W.F. Ames, (1972), *Nonlinear Partial Differential Equations in Engineering, vol II*, Academic Press

D.A. Anderson, (1985), *Constructing Adaptive Grids with Poisson Grid Generation*, in: J. Hauser and C. Taylor, eds., Numerical Grid Generation in Computational Fluid Dynamics (Pineridge, Swansea, U.K.) pp 125-136

M.J. Baines, (1991), *Algorithms for Best Piecewise Discontinuous Linear and Constant L_2 Fits to Continuous Functions with Adjustable Nodes in One and Two Dimensions*, SIAM J. Sci. Stat. Comput., Submitted

M.J. Baines and N.N. Carlson, (1990), *On Best Piecewise Linear L2 Fits with Adjustable Nodes*, University of Reading, Numerical Analysis Report, 6/90

J.U. Brackbill and J.S. Saltzmann, (1982), *Adaptive Zoning for Singular Problems in Two Dimensions*, J. Comp. Phys. **46**, pp 342-368

G.F. Carey and H.T. Dinh, (1985), *Grading Functions and Mesh Redistribution*, SIAM J. Num. Anal. **22**, pp 1028-1040

D. Catherall, (1988), *A Solution-Adaptive-Grid Procedure for Transonic Flows Around Aerofoils*, RAE Technical Report 88020

D. Catherall and C.J. Fitzsimons, (1991), *On the Measurement of Grid Quality in CFD Applications*, RAE Technical Report, Preprint

J.C. Cavendish, (1974), *Automatic Triangulation of Arbritrary Planar Domains for the Finite Element Method*, Int. J. Num. Meth. Engng. **8**, pp 679-696

C.K. Chui, P.W. Smith and J.D. Ward, (1977), *On the Smoothness of Best L_2 Approximants from Nonlinear Spline Manifolds*, Math. Comp. **31**, pp 17-23

E.F. D'Azevado and R.B. Simpson, (1989), *On Optimal Interpolation Triangle Incidences*, SIAM J. Sci. Stat. Comput. **10**, pp 1063-1075

C. De Boor, (1973), *Good Approximations by Splines with Variable Knots*, in: Spline Functions and Approximation Theory. Int. Ser. Num. Meths. **21**, Basel: Birkhauser

B. Delaunay, (1934), *Sur la sphere vide*, Bull.Acad.Sci. USSR (VII): Classe Sci. Mat. Nat., pp 793-800

K Deljouie-Rakhshandeh, (1990), *An Approach to the Generation of Triangular Grids Possessing Few Obtuse Triangles*, Int. J. Num. Meth. Engng. **29**, pp 1299-1321

G.L. Dirichlet, (1850), *Uber die Reduction der Positiven Quadratischen Formen mit drei Unbestimmten Ganzen Zahlen*, J. Reine Angew. Math. **40**, pp 209-277

G. Dueck and T. Scheuer, (1990), *Threshold Accepting: A General Purpose Optimization Procedure Appearing Superior to Simulated Annealing*, J. Comp. Phys. **90**, pp 161-175

N. Dyn, S. Rippa and D. Levin, (1990), *Data Dependent Triangulations for Piecewise Linear Interpolation*, IMA J. Num. Anal. **10**, pp 137-154

P.R. Eiseman, (1987), *Adaptive Grid Generation*, Computer Methods in App. Mech and Engng. **64**, pp 321-376

P.R. Eiseman and G. Erlebacher, (1987), *Grid Generation for the Solution of Partial Differential Equations*, ICASE report 87-57. NASA Langley,

C.L. Farmer, D.E. Heath and R.O. Moody, (1991), *A Global Optimisation Approach to Grid Generation*, SPE 21236, Reservoir Simulation Symposium, Anaheim, Feb 17-20

R. Franke, (1979), *A critical comparison of some methods for interpolation of scattered data*, Report NPS-53-79-003, Naval Postgraduate School

R. Franke, (1982), *Scattered Data Interpolation: Tests of Some Methods*, Math. Comp. **38**, pp 181-200

W.H. Frey, (1987), *Selective Refinement: A New Strategy for Automatic Node Placement in Graded Triangular Meshes*, Int. J. Num. Meth. Engng. **24**, pp 2183-2200

P.A. Gnoffo, (1982), *A Vectorized, Finite Volume, Adaptive-Grid Algorithm for Navier-Stokes*, in: J.F. Thompson, ed., Numerical Grid Generation (North-Holland, Amsterdam) pp 819-835

P.J. Green and R. Sibson, (1978), *Computing Dirichlet Tesselations in the Plane*, Comp. J. **21**, pp 168-173

I.R. Hawkins and J.R. Kightley, (1989), *Grid Adaption for Turbulent Flow Problems*, in: Proceedings of the Conference on Laminar and Turbulent Flows, Pineridge Press

J. Kautsky and N.K. Nichols, (1980), *Equidistributing Meshes with Constraints* SIAM J. Sci. Stat. Comput. **1**, pp 499-511

S.R. Kennon and G.S. Dulikravich, (1986), *Generation of Computational Grids Using Optimization*, AIAA Journal. **24**, pp 1069-1073

S. Kirkpatrick, C.D. Gellatt and M.P. Vecchi, (1983), *Optimisation by Simulated Annealing*, Science, **220**, pp 671-680

A. Kumar and N.S. Kumar, (1988), *A New Approach to Grid Generation Based on Local Optimisation*, in: S. Sengupta *et al.*, eds., Numerical Grid Generation in Computational Fluid Mechanics 1988 (Pineridge Press, Swansea, U.K.) pp 177-184

C.L. Lawson, (1972), *Transforming Triangulations*, Discrete Mathematics **3**, pp 365-372

C.L. Lawson, (1977), *Software for C^1 Interpolation*, in: J.R. Rice ed., Mathematical Software III. (New York: Academic Press) pp 161-194

G.S. Lee, (1982), *Piecewise Linear Approximation of Multivariate Functions*, Bell System Tech. J. **61**, pp 1463-1486

S.H. Lo, (1985), *A New Mesh Generation Scheme for Arbitrary Planar Domains*, Int. J. Num. Meth. Engng. **21**, pp 1403-1426

P.D. Loach and A.J. Wathen, (1991), *On the Best Least Squares Approximation of Continuous Functions using Linear Splines with Free Knots*, IMA J. Num. Anal. **11**, pp 393-410

R. Lohner, K. Morgan and O.C. Zienkiewicz, (1986), *Adaptive Grid Refinement for the Compressible Euler Equations*, in: I. Babuska *et al.* eds., Accuracy Estimates and Adaptive Refinements in Finite Element Computations (John Wiley and Sons Ltd) pp 281-297

D.J. Mavriplis, (1990), *Adaptive Mesh Generation for Viscous Flows Using Delaunay Triangulation*, J. Comp. Phys. **90**, pp 271-291

K. Miller and R.N. Miller, (1990), *Moving Finite Elements, Part 1*, SIAM J. Num. Anal. **18**, pp 1019-1032

M.S. Mock, (1985), *Tetrahedral Elements and the Scharfetter-Gummel Method*, in: NASECODE IV, Proceedings of the Fourth International Conference on Numerical Analysis of Semiconductor Devices and Integrated Circuits (Boole Press)

E.J. Nadler, (1985), *Piecewise Linear Approximation on Triangulations of a Planar Region*, PhD thesis, Brown University

B. Palmerio, L. Fezoui, C. Olivier and A. Dervieux, (1990), *On TVD Criteria for Mesh Adaption for Euler and Navier-Stokes Calculations*, INRIA report 1175,

J. Peraire, M. Vahdati, K. Morgan and O. C. Zienkiewicz, (1987), *Adaptive Remeshing for Compressible Flow Computations*, J. Comp. Phys. **72**, pp 449-466

V. Pereyra and E.G. Sewell, (1975), *Mesh Selection for Discrete Solution of Boundary Problems in Ordinary Differential Equations*, Num. Math. **23**, pp 261-268

C.P. Reeves, (1991), *Moving Finite Elements and Overturning Solutions*, PhD thesis, University of Reading

S. Rippa, (1991), *Long and Thin Triangles can be Good for Linear Interpolation*, Preprint

E.A. Sadek, (1980), *A Scheme for the Automatic Generation of Triangular Finite Elements*, Int. J. Num. Meth. Engng. **15**, pp 1813-1822

E. G. Sewell, (1972), *Automatic Generation of Triangulations for Piecewise Polynomial Approximation*, PhD Thesis, Purdue University

R. Sibson, (1978), *Locally Equiangular Triangulations*, Comp. J. **21**, pp 243-245

G. Strang and G.S. Fix, (1973), *An Analysis of the Finite Element Method*, Prentice-Hall Inc

P.K. Sweby, (1987), *Data-Dependent Grids*, University of Reading, Numerical Analysis Report, **7/87**

W.C. Thacker, A. Gonzalez and G.E. Putland, (1980), *A Method for Automating the Construction of Irregular Grids for Storm Surge Forecast Models*, J. Comp. Phys. **37**, pp 371-387

F. Thomasset, (1981), *Implementation of Finite Element Methods for Navier-Stokes Equations*, Springer-Verlag, Berlin

J.F. Thompson, Z.U.A. Warsi and C.W. Mastin, (1985), *Numerical Grid Generation, Foundations and Applications*, North-Holland, Amsterdam

J.G. Verwer and R.A. Trompert, (1991), *Local Uniform Grid Refinement for Time-Dependent Partial Differential Equations*, Report NM-R9105, CWI, Amsterdam

G. Voronoi, (1908), *Nouvelles Applications des Parametres Continus a la Theorie des Formes Quadratiques. Deuxieme Memorie: Recherches sur les Parralleloedres Primitifs*, J. Reine Angew. Math. **134**, pp 198-287

D.F. Watson and G.M. Philip, (1984), *Survey-Systematic Triangulation*, Computer Vision, Graphics and Image Processing. **26**, pp 217-223

A.B. White, (1979), *On Selection of Equidistributing Meshes for Two-Point Boundary-Value Problems*, SIAM J. Num. Anal. **16**, pp 472-502

Appendix A

Sets of Data Points

In this Appendix the sets of data points mentioned in Chapter 4 are presented. In A.1 the 33 data point set is shown, while in A.2, the 100 data point set is given.

A.1 33 Data Points

This is the set of 33 data points as presented by Franke (1979).

(x,y)	(x,y)	(x,y)
$(1.0, 1.0)$	$(0.0, 1.0)$	$(0.0, 0.0)$
$(1.0, 0.0)$	$(0.6, 0.7)$	$(0.7, 0.2)$
$(0.1, 0.3)$	$(0.7, 0.7)$	$(0.3, 0.8)$
$(0.6, 0.2)$	$(0.8, 0.1)$	$(0.9, 0.3)$
$(0.1, 0.8)$	$(0.9, 0.9)$	$(0.8, 0.3)$
$(0.5, 1.0)$	$(0.2, 0.2)$	$(1.0, 0.5)$
$(0.6, 0.8)$	$(0.8, 0.2)$	$(0.5, 0.0)$

$(0.8, 0.7)$	$(0.6, 0.9)$	$(0.0, 0.5)$
$(0.2, 0.1)$	$(0.3, 0.3)$	$(0.1, 0.1)$
$(0.1, 0.5)$	$(0.7, 0.9)$	$(0.8, 0.8)$
$(0.9, 0.8)$	$(0.8, 0.4)$	$(0.7, 0.7)$

A.2 100 Data Points

This set of 100 data points is a modification of that presented by Franke (1979). All the points outside the unit square have been moved onto the boundary; the four points nearest to the vertices of the unit square have been moved to the vertices and any point within 0.03 of the boundary was moved onto the boundary. This gives more nodes on the boundary and all the nodes are on, or inside, the unit square.

(x,y)	(x,y)	(x,y)
(1.00000, 1.00000)	(0.00000, 1.00000)	(0.00000, 0.00000)
(1.00000, 0.00000)	(0.48557, 0.38914)	(0.00000, 0.25769)
(0.00000, 0.49436)	(0.03954, 0.69934)	(0.03158, 0.91077)
(0.13242, 0.05013)	(0.12544, 0.25930)	(0.07676, 0.41711)
(0.06265, 0.65522)	(0.26456, 0.00000)	(0.20890, 0.26688)
(0.17147, 0.48017)	(0.19092, 0.68788)	(0.23046, 0.90465)
(0.36632, 0.03970)	(0.38324, 0.23896)	(0.34668, 0.49030)
(0.38732, 0.64452)	(0.37954, 0.89381)	(0.41498, 0.00000)
(0.42000, 0.22625)	(0.47926, 0.63243)	(0.39778, 0.84897)
(0.58487, 0.00000)	(0.60639, 0.27093)	(0.57413, 0.42594)

(0.59901, 0.67338) (0.60970, 0.92424) (0.66169, 0.00000)

(0.63965, 0.20083) (0.70012, 0.48907) (0.69090, 0.66978)

(0.67189, 0.93661) (0.77369, 0.00000) (0.74104, 0.19366)

(0.73960, 0.47142) (0.82145, 0.66851) (0.80766, 0.84768)

(0.84246, 0.03805) (0.83669, 0.20831) (0.84781, 0.43356)

(0.91757, 0.63074) (0.92799, 0.90423) (1.00000, 0.26958)

(1.00000, 0.43961) (1.00000, 0.69415) (0.05399, 0.15867)

(0.00000, 0.34140) (0.09587, 0.91465) (0.00000, 0.57829)

(0.00000, 0.74702) (0.10903, 0.09186) (0.09345, 0.33816)

(0.14519, 0.56156) (0.14528, 0.75241) (0.06956, 0.96324)

(0.23916, 0.06623) (0.27673, 0.36960) (0.22668, 0.59406)

(0.18676, 0.81856) (0.24262, 1.00000) (0.38577, 0.08345)

(0.31791, 0.31241) (0.37766, 0.51998) (0.38129, 0.82038)

(0.28035, 1.00000) (0.42777, 0.15610) (0.46636, 0.31751)

(0.40920, 0.50850) (0.48123, 0.75110) (0.40273, 1.00000)

(0.57300, 0.12724) (0.50139, 0.34777) (0.61069, 0.60847)

(0.53806, 0.72352) (0.50262, 1.00000) (0.64278, 0.07078)

(0.67040, 0.32598) (0.63336, 0.50963) (0.68956, 0.77586)

(0.68377, 1.00000) (0.76353, 0.10214) (0.82590, 0.32358)

(0.80866, 0.60916) (0.72906, 0.80228) (0.81710, 1.00000)

(0.86840, 0.09020) (0.94185, 0.33185) (0.85996, 0.59101)

(0.85963, 0.81448) (0.85128, 0.96950) (0.96706, 0.13341)

(0.96763, 0.37953) (0.96570, 0.50444) (1.00000, 0.74599)

(0.94715, 1.00000)