THE CONSTRUCTION OF OPTIMAL DRAPE SURFACES WITH CONSTRAINED FIRST AND SECOND DERIVATIVES

by

REINER JUSTIN FOSSATI

submitted in accordance with the requirements for the degree of

DOCTOR OF PHILOSOPHY

in the subject

OPERATIONS RESEARCH

at the

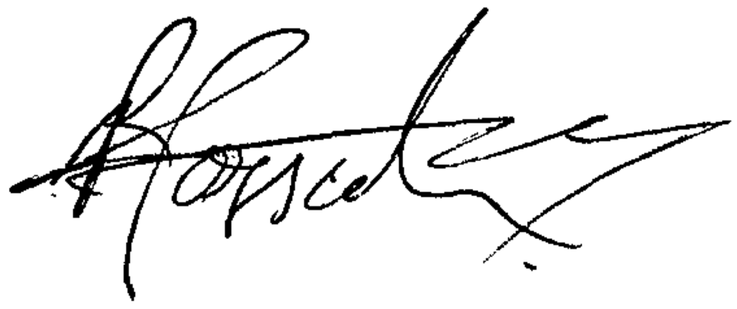
UNIVERSITY OF SOUTH AFRICA

PROMOTER: PROF J S WOLVAARDT

JANUARY 2003

Declaration

I declare that *The Construction of Optimal Drape Surfaces with Constrained First and Second Derivatives* is my own work and that all the sources I have used or quoted have been indicated and acknowledged by means of complete references.



Reiner Justin Fossati

Abstract

The need to construct optimal drape surfaces arises in airborne geophysical surveys where it is necessary to fly a safe distance above the ground and within the performance limits of the aircraft used, but as close as possible to the surface. The problem is formulated as an LP with constraints at every point of a grid covering the area concerned, yielding a very large problem. The objective of the LP is to create as "good" a surface as possible. This formulation is new, as previous methods did not aim to minimise an objective function.

If the desired surface has only slope limitations, the resulting drape surface must be constrained in the first derivative. Such a drape surface is readily constructed using the Lifting Algorithm. It is shown that the Lifting Algorithm is both exact and has great speed advantages. Some numerical results confirming exactness and speed are presented, as is the algorithm's analogy to a flow network method. An enhanced lifting method with a better order of complexity is also proposed and tested numerically.

In most practical situations a drape surface is required which has both first and second derivatives constrained. If only a cut through such a surface is considered, the problem can be solved with relative ease by exploiting its network-like structure. This method forms the basis of one of the preferred heuristics developed later. It was not possible to generalise this method to a full two-dimensional drape surface. A commercially available LP package fares better in finding the optimal solution.

Several heuristic methods were examined. First a general heuristic method based on a lifting approach was developed. This was followed by a method using repeated application of the method used for sections (the Alternating One-dimensional Dual Algorithm ["AODA"]). Three heuristics based on thimbles were also designed. Thimbles are caps whose first and second derivatives are acceptable and which are placed over local infeasibilities in the topography.

The work ends with a chapter comparing the efficiency of various heuristics and comparing the results obtained using a number of test datasets. It was found that heuristic methods provide acceptable drape surfaces and that the choice lies between speed and accuracy, with a previously designed smoothing method being the fastest and the AODA the most accurate and quick enough.

Key terms: Drape surface; Airborne geophysical surveying; Linear programming; Gridding; Thimbles; Heuristic LP methods; Constrained derivatives; Topographic modelling.

Acknowledgements

I would hereby like to acknowledge the assistance of the following people without whose input and commitment the present work would never have come into being:

- My promoter, Prof J S Wolvaardt, for starting me off on this project in the first place, and for his thorough guidance and numerous hours spent in keeping me on the right track.

- The kind people at the Department of Quantitative Management, Unisa, for their moral support, also Prof Chris Swanepoel for making the XA Library available for this work and for his technical input.

- Unisa's subject librarian, Els Ten Krooden, for useful tips about referencing procedures.

- Various persons at Geodass (Pty) Ltd, which has become Fugro Airborne Surveys (Pty) Ltd in the meantime, notably Martin Frere who provided the drape idea many years ago.

- My long-suffering family who have helped me make my way through the quagmire of work, threatening divorce and other unpleasantries, but being there when I really needed them.

January 2003

Contents

Chapter 1
Introduction and Overview

1.1 Description of the Problem

The drape problem arises in the design of geophysical surveys. During such operations an aircraft with measuring instruments is flown along parallel flight lines across an area of interest. In order to obtain the best possible data, the ground clearance should be kept as small as possible. On the other hand, safety dictates a minimum flying height. Data quality is improved if adjacent flight lines are at similar levels. Therefore, in order to obtain the best overall data quality, the flight path should ideally be planned along a smooth surface which can be adhered to by an aircraft flying in the appropriate directions. This is the *drape surface.*

Drape flying is practised by a number of airborne geophysical surveying contractors. However, the drape surfaces which have been constructed and used so far are ones which simply adhere to certain geometric constraints as dictated by aircraft performance. The present work attempts to develop techniques to optimise drape surfaces for given survey areas in the sense that an attempt is made to minimise the accumulated ground clearance while still maintaining flying feasibility. Previous drape surface generation has not addressed the problem with an objective function in mind. The current problem is thus one which has not yet been tackled using Operations Research techniques.

1.2 Overview of the Thesis

This chapter introduces the drape surface problem and gives short descriptions of the chapters to follow, indicating the general train of thought which accompanied the development of various methods for solving the drape surface problem.

Chapter 2 gives a fuller description of the drape surface problem as well as providing an overview of the literature available on the subject. It then gives a mathematical statement of the continuous problem and the function to be minimised. Since topographies are never available as continuous functions, the discrete equivalent of the problem is given, and this is then expressed as a linear program. The chapter ends with a discussion of some practical considerations.

To start with, a simplified version of the problem was considered using constrained first derivatives only. This implies that the aircraft performance with respect to climb and descent rates is taken into account, but that its ability to turn in valleys and drop after flying over the crest of a peak is ignored. This problem can be solved very easily; the derivation of the Lifting Algorithm is given in **Chapter 3**. It is shown that this method has a low order of complexity and that it easily outperforms a standard LP technique. The analogy to a network flow is also given. The chapter concludes by stating the Enhanced Method which has an even lower order of complexity. Because of its extreme simplicity and the fact that its output formed a convenient starting point, the Lifting Algorithm was used as a first pass in all subsequent methods.

In **Chapter 4** another simplified form of the drape problem is considered, this time with both derivatives constrained but restricted to one-dimensional cuts through a surface. The One-dimensional Dual Algorithm is developed which performs favourably when compared to a standard LP package. The exploitation of the special structure of the problem results in a rather parsimonious use of memory. The analogy of a network flow with a given intermediate solution is shown. The One-dimensional Dual Algorithm forms the central computing section of the Alternating One-dimensional Dual Algorithm developed in Chapter 6 for the general problem. This, in turn, is eventually found to be one of the best heuristic methods.

Chapter 5 describes the efforts of generalising the One-dimensional Dual Algorithm of the last chapter to the full two-dimensional problem. It is found that this generalisation is

possible but that the special structure of the problem cannot be exploited to reduce the resources used or increase the speed at which solutions can be found.

Since it was shown in Chapter 5 that a standard LP package could outperform the Dual Algorithm, it was decided to investigate the possibility of using heuristics to solve the drape problem. **Chapter 6** describes the author's previously developed Smoothing Method which convolves the topography with a suitably sized set of weights to produce a feasible drape surface without attempting optimisation. The concept of selectively lifting points is applied, culminating in General Heuristic 5 which produces acceptable results (good answers, competitive speed). The rest of the chapter is devoted to the development of the Alternating One-dimensional Dual Algorithm (AODA) which makes use of repeated applications of the One-dimensional Dual Algorithm developed in Chapter 4. The AODA gives better results than General Heuristic 5.

Chapter 7 addresses the drape problem from a different angle by introducing the concept of thimbles. These are feasible caps which are placed over infeasible peaks in the topography. Once there are no more infeasible peaks, the valleys in between these are made feasible by filling them up using an algorithm that finds the unique lowest height for each valley point. The Basic Thimble Method places thimbles centrally on all infeasible peaks. The chapter also describes the Enhanced Thimble Method which moves the thimbles around in an effort to improve the value of the objective function. Since the Enhanced Thimble Method tended to take rather long, the Simplified Enhanced Thimble Method was developed which simplifies the search for the optimal thimble positions. As expected, the last method is faster but not as accurate as its predecessor.

Chapter 8 compares the algorithms developed in this study and a standard LP method. The exact method developed in Chapter 5 produces correct results but is slow. The heuristics generated in Chapters 6 and 7 fare better. The choice of heuristic is ultimately up to the user who must decide between accuracy (where the AODA is recommended) and speed (for which the Smoothing Method would be preferable).

Chapter 2
Description of the Problem and Literature Study

2.1 Conceptual Description

Airborne geophysical surveys require various measuring instruments to be transported across the area of interest by means of a fixed or rotary wing aircraft. These can include several different instruments; most common are magnetometers, spectrometers and electromagnetic (EM) systems. Magnetometers and spectrometers are usually attached to or contained in the aircraft itself, while EM systems are towed behind it in a "bird".

A survey is generally conducted by flying a series of parallel, equally spaced lines across the area, usually in a direction normal to geological features which are known to exist (or are suspected to exist) below the surface. These parallel lines, also called flight or traverse lines, are typically 50 to 2 500 metres apart and cover the whole area. Once the entire set of traverse lines has been completed, extra lines are flown at right angles to them. These are called tie lines or control lines and have a rather wider spacing than the traverses, usually 5 to 10 times the spacing. The tie lines are eventually used during processing of the data to check the traverse line data at the intersections. The term survey lines will be used below to refer to both traverse (flight) and tie lines.

The quality of the data gathered during the survey is largely dependent on the altitude at which it was collected. The lower the height above the ground, the better the resolution between close lying features. It would indeed be ideal to have the measuring instruments transported along the ground. However, safety considerations dictate that a minimum altitude be maintained by the aircraft. Generally, helicopters (rotary wing platforms) can be made to hug even a rough terrain fairly well and thus adhere to both the safety and resolution requirements. Unfortunately, helicopters cannot always be used for geophysical surveys because they are unsuitable for large areas and/or very long lines.

In the case of fixed wing aircraft the situation is rather different. The rate of climb is limited by the aircraft's performance figures; the maximum rate of controlled descent is obviously also bounded. Although the ambient temperature, pressure, wind conditions and payload all have an effect on the figures, it is usually found that the maximum rate of ascent which can be expected from a particular plane is greater than the maximum rate of controlled descent. The rates of ascent and descent are measured as normal gradients, hence in metres rise per metre of forward displacement.

Problems can now arise between adjacent survey lines. For example, the first line might be flown from south to north along the western edge of the survey area, the next line would then be from north to south at a distance of 250 m to the east, the third line would go from south to north again at a further displacement of 250 m, and so on. Now assume that there was a high cliff perpendicular to the flight direction to be negotiated on several adjoining lines. An experienced pilot would then be able to start his climb fairly late as he approached the cliff. However, when flying the next line he would be flying "down" the cliff and would reach the plain below farther away from the cliff than where he left it on climbing. See Figure 2.1. At a given distance from the cliff, say at A in the figure, the measurements on the approach to the cliff (going left) would be taken at a much lower altitude than those taken when going right. The result would be a set of rather unpleasant corrugations in the final data which could only be removed with difficulty and whose removal would cause degradation of the "good" readings.

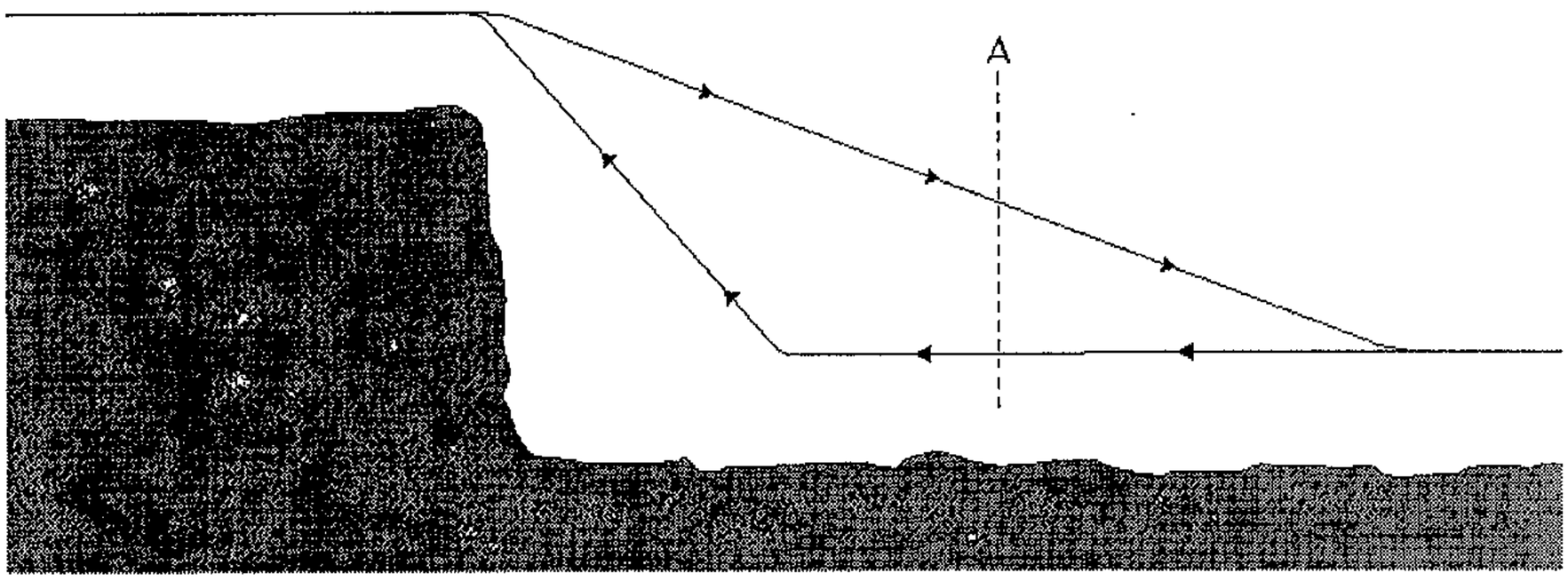


Figure 2.1. Differing slopes flying "up" and "down" a cliff

In the figure above the problem of alternating "high" and "low" survey lines can be avoided by allowing the lines in both directions to follow the same path. In other words, both "up" and "down" lines should follow the upper line in the sketch, ie the "down" path. In this way adjacent survey lines would be essentially at the same altitude at corresponding points (within the limits of the pilot's ability) and there would be hardly any corrugations in the resultant compiled data.

If this concept is extended to the tie lines as well, the generation of a two-dimensional "flyable" surface is required. This is the *drape surface*. The drape surface is generated before the survey is flown. Drape flying further requires that the pilot can manoeuvre the aircraft accurately; this is usually done using some form of differential global positioning system (DGPS). By definition, the drape surface must

- be more than the minimum flying height (normally 80 or 100 metres) above the ground at all points;
- be "flyable" in both directions of both traverse and tie lines by the survey aircraft.

In order to be effective the drape surface should hug the ground as closely as possible. A flat plane positioned at the minimum flying height above the highest point of the survey area would certainly satisfy the defining requirements of a drape surface; however, it would not be in the spirit of accurate airborne surveying.

Apart from the constraints on the slopes of lines (which translate to constraints on the slopes of sections through the drape surface in the directions of flying), constraints on the rate of change of the slope must also be considered. Consider a rock pinnacle which lies directly on a survey line. The section of the drape surface along this line would consist of two segments of equal and opposite slope intersecting above the pinnacle (the solid line in Figure 2.2). The traverse through the drape surface is obviously not differentiable at

the peak and therefore cannot be flown by any fixed wing aircraft. The drape surface must therefore be adjusted to take into account the maximum rate of change of slope which the aircraft can achieve.

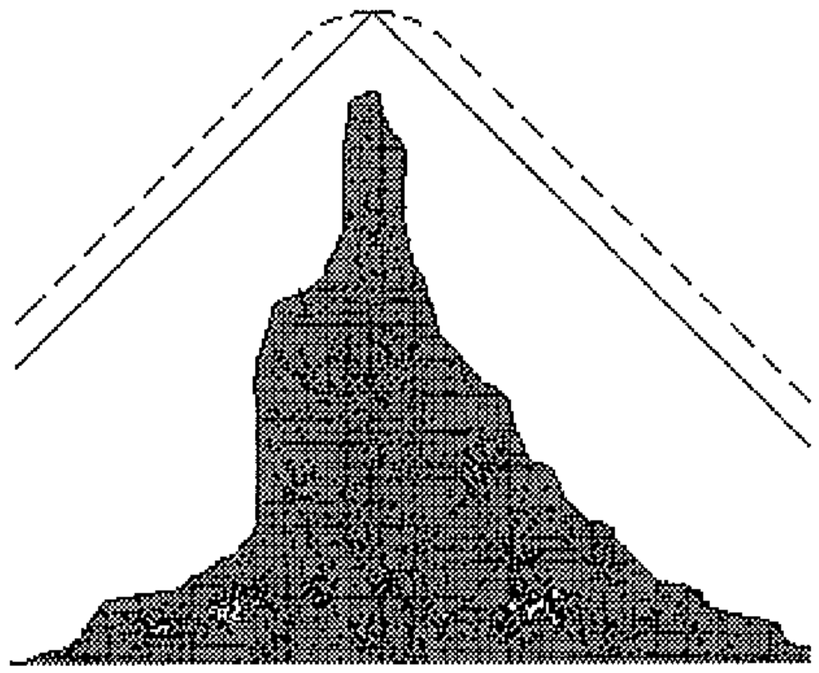


Figure 2.2. Removing the point in the drape surface over a peak

The drape surface must be "rounded out" as shown by the dotted line in Figure 2.2. The rate of change of the slope is measured in metres per metre per metre, or in reciprocal metres (m^{-1}).

Similar rounding has to take place where the flight path makes a valley – it is clearly not possible for the aircraft to change from maximum descent rate to maximum ascent rate instantaneously.

2.2 Survey of Literature

A survey of the existing literature brought to light the following related topics and fields of study:

2.2.1 Modelling of textile deformation in drape situations

The drape surface which we are trying to develop for airborne surveys must be mathematically continuous, as well as being differentiable. It is therefore reasonable to consider the drape surface as if it were a piece of material with some stiffness such as a tablecloth which has been thrown over a model of the topography which seeks as low a position as possible under its own weight. Such a cloth will not develop discontinuities and its shape will be differentiable in the practical case.

Much research has been undertaken and is still ongoing in the field of textile deformation. This relates to both static and dynamic models of cloth movement which are used in computer graphics, computer animation and the textile industry (You, Zhang, Comninos). For example, the researchers just quoted develop a complex model involving in-plane and out-of-plane loads and deformations, and eventually solve the resulting differential equations using a finite difference approximation. As examples they show a tablecloth suspended by its four corners hanging under its own weight, and a similar cloth draped over a square table. The research is aimed at predicting and/or modelling the shape which a cloth will assume under various loads.

Work with a similar aim has been undertaken by Govindaraj and Vallamshettla at the National Textile Center in Philadelphia, PA, USA. They point out that "fabrics are discontinuous structures possessing anisotropic, nonlinear, hysteric and time dependent characteristics… Fabric drape is a complex buckling of planar sheet problem which requires post-buckling response." These researchers, too, point out the complexity and the large number of factors which influence the shape of a cloth surface. They have also applied themselves to the analysis of "fiber plies with seams, stitches and fusings". They model the behaviour of the cloth using finite element methods. Their work is still continuing; in a recent report they point out that "commercial packages of drape are not real drape but texture mapping algorithms". Their current aim is to model complete garments.

The flexibility of a piece of material is determined by its drape coefficient. This is measured in a Cusick drape meter or drape tester. A circular piece of cloth is clamped between two smaller horizontal discs and permitted to hang down under its own weight. The area obtained by projecting vertically the shadow of the draped specimen is expressed as a percentage of the area of the whole specimen (AMS, Lahey, Kahaner).

In all the above textile-related research papers the aim was the prediction of the position (and possibly movement) of a cloth article, usually a tablecloth or an item of clothing. A similar approach to the problem at hand has the following two drawbacks:

- Textiles do not exhibit the properties which would be required of the envisaged drape surface. The draped cloth would have to possess nonrealistic elastic properties. For example, in the drape surface which is being generated for geophysical survey flying, two points a certain distance apart might have a level section of the drape surface between them; on the other hand, they might have the maximum possible height differential which would mean that the drape surface would need to undergo some stretching. However, such stretching would engender a number of reactive forces in the draped material and cause unwanted distortion in the drape surface.

- The intention of generating a drape surface for airborne exploration in this work is to find a surface which is optimal in some way or another (typically by minimising the total ground clearance). The research into the properties of cloth, on the other hand, attempts to predict the fall of the material rather than to influence the nature of this fall and its quantitative properties.

In short, thus, the existing research into "real" drape surfaces such as tablecloths or garments draped over bodies cannot be used as a baseline for a problem which is essentially concerned with generating a surface with an optimal property.

2.2.2 Modelling of deformable surfaces

Much research has been undertaken in the area of modelling the deformation of physical surfaces. This is an extension of the study area mentioned above (tablecloths and textile drapes) to the deformation of other "real-world" structures, often in the medical field.

Tissue structures and shapes are usually acquired by some non-intrusive method, eg tomography, and the three-dimensional surfaces are typically generated from these data using methods such as the Marching Cubes Algorithm (Ward). As a result the structure to be modelled (ie organ) is represented as occupying sections of a number of equal volume elements (voxels). Its surface is determined as described below.

The Marching Cubes Algorithm is a patented method designed by William E. Lorensen and Harvey E. Cline (Dik) which relies on the very simple observation that if one vertex of a voxel lies within the volume of interest (ie the organ) while an adjacent vertex of the same voxel does not, then the surface of the organ must obviously intersect the voxel edge joining the two vertices. It is possible to assign the eight vertices of a voxel to be inside or outside the volume of interest in 256 different ways. These include a fair number of rotations and reflections as well as some trivial cases. For example, if all the vertices of a voxel are within the organ it is (reasonably) assumed that the surface of the organ does not intersect the voxel. On the other hand, if seven vertices are within the organ while the last one lies outside, the surface of the organ must pass through the three edges which meet at the "outside" vertex. As a first pass the midpoints of these three edges are taken to lie on the surface and they describe a triangle which is taken to be a face of the surface. The name "marching cubes" arises from the picture of the cubes moving through the volume of interest in rank and file.

De Bruin *et al* describe a technique called SurfaceNets which they believe provides a better representation of a surface than previously employed techniques such as the Marching Cubes Method. As a typical application they mention the deformation of the

uterus under inflation prior to the surgical removal of growths. The model of the uterus wall is constructed from 2-D image slices (taken in the uninflated state). The method is then used to generate the deformable model and to simulate the inflation.

The shapes of objects found in images can be modelled using a wide variety of techniques. These include **snakes**, energy-minimising curves developed by Kass, Witkin and Terzopoulos (as referenced in Zhang). Snakes are "active contours" (Zhang) in the sense that they are guided by features in the images on which they are drawn. Thus snakes can be used for line or edge detection and can be guided interactively by a user, typically on a graphics screen. Their behaviour is similar to that of piecewise splines and is governed by internal and external forces. The internal forces originate from the shape of the snake, while image features (eg bright spots, lines and so on) constitute the external forces. The concept of planar snakes can be extended to three-dimensional **balloons** (Jakab).

Cohen and other researchers have published numerous papers on the modelling of anatomical structures, the handling of the resulting three-dimensional data and the analysis of the images, also using balloons.

Valuable as all these contributions may be to medicine, computer modelling and a host of other fields, they miss their applicability for generating the drape surfaces which form the subject of the current work. In particular:

- The methods mentioned in this section apply particularly to physical objects with physical properties, especially as far as deformation goes. These elastic properties in a drape surface for flying would cause unwanted deformation. The argument applies in this section in exactly the same way as it did in the previous one.

- Whereas the research described in this section is intended to model the deformation and behaviour of organs and other objects in as realistic a way as possible, the object of the study of drape surfaces for flying is to generate a

surface which is optimal in terms of some metric. The modelling of internal organs does not involve any search for an optimum other than, perhaps, the energy minimisation of snakes and balloons. It is an inherent property of Operations Research methodology that the formulation of the problem invariably involves some form of optimisation. The modelling of deformable surfaces described has a different aim, viz that of predicting the shape of a body after deformation.

- Current methods describing the deformation of physical objects are typically very complex, involve many differential equations and finite element analysis. It is believed that the drape surface problem can be solved in a much simpler way.

- The use of interactive methods to generate drape surfaces would, without doubt, introduce a subjective flavour into any algorithm. Interactive techniques should be avoided – they have great applicability for tracing contours in images but not for generating drape surfaces.

It is a characteristic of Operations Research that where it describes problems also found in other subjects (eg tabling and scheduling problems), the traditional OR approach seeks to optimise a formally defined objective function. This is not common in other fields where the approach is more frequently to simply attain feasibility. We were not surprised to see that related problems are not described by the use of an objective function.

2.2.3 Post-survey drape adjustments

Methods exist which allow level-flown aeromagnetic data to be converted to a drape surface. These typically involve variable continuation methods in which the data as collected in the field are *continued* upwards or downwards on to the drape surface (Schlumberger). The continuation techniques usually involve frequency domain operations which can be tedious and time-consuming. When these operations are applied,

the current representation of the data in the spatial domain is changed to a representation in the frequency domain (using Fourier transforms). In essence the data are now represented as the sum of a number of signals at different frequencies with different amplitudes and different phases. In the frequency domain the continuation operation ("moving" the data upwards or downwards) can be effected quite simply by the application of a so-called transfer function. In the case of upward and downward continuation, the amplitude of each frequency is typically multiplied by a factor (which depends on the frequency). Once the continuation has been done in the frequency domain, a reverse Fourier transform is applied to return the data to the space domain. The Fourier operations are usually not that straightforward as they require the size of the dataset to be padded to a convenient size; this can cause a number of unpleasant edge effects. Even Fast Fourier Transforms (FFTs) are not that fast on large datasets. Worst of all, downward continuation is a process that is often not stable and can cause the calculations to "blow up".

In some cases convolution techniques exist for upward and downward continuation operations. These involve the application of a set of weights to the data grid. Such an operation is straightforward but suitable sets of weights ("convolution operators") are not always available.

Perhaps the most successful level-to-drape converter is the proprietary Compudrape technique by Paterson. This is also available in the Geosoft suite of software (Geosoft). It relies on the so-called chessboard method in which "level-flown data are continued to a series of levels that span the drape range, giving a set like stacked 3-D chessboards; interpolation is carried out vertically" (Society of Exploration Geophysicists [SEG]).

Large sections of the Canadian aeromagnetic coverage have been draped using Compudrape. In particular, this applies to data in the Rocky Mountains where the terrain is very rough (Miles *et al*). Similar techniques were applied by the Montana Bureau of Mines and Geology (Kennelly) and in Colorado (United States Geological Survey [USGS]).

The same methods can be applied for corrections to airborne data which were collected in high wind conditions. These, together with rough terrain, would result in a survey height significantly different from the nominal height. As mentioned previously, such data would exhibit corrugation, and in severe cases would be considerably degraded in the attempt to remove these artifacts.

Pilkington and Thurston describe their experiences with attempting to drape raw data to a drape surface situated at the mean flying height. In their paper they express the opinion that corrections made on a line basis yield better results than if the entire grid were corrected.

Clearly, correction of data by upward and downward continuation from the flown height to the drape height is feasible, and commercial solutions exist for performing this task. Nevertheless, such post-processing cannot replace data which may have got lost and, where possible, surveys should be flown in drape mode in the first place (should this be desired and possible) rather than be draped afterwards.

2.2.4 Actual drape flying

In the past a fair number of surveys have been flown along drape surfaces. Two examples are a high-resolution aeromagnetic survey in northern Alberta, Canada (Best) and another one in Gabon (Peirce *et al*). Although the literature abounds with statements that actual drape surveys have been flown, there is almost no explanation as to how these drape surveys are planned or how the survey lines are generated. The only mention comes from Sander Geophysics Ltd (SGL), a geophysical exploration company.

Bryant *et al* from SGL mention the requirement that adjacent lines should not differ significantly in height even if they are flown in opposite directions or on different days.

They also mention that crossovers between traverse and tie lines should be at essentially the same height.

Sander explains the technique used at SGL in some detail. The method used is identical to the one described above: an accurate digital terrain model (DTM) is obtained, the area of interest is isolated from it, the drape surface is generated taking into account the capabilities of the survey aircraft, and the survey is then performed using a differential global positioning system and a radar altimeter to guide the pilot along the drape surface. Some examples are given which show the excellent results obtained when using the drape flying method.

SGL's system, known as SGDrape, is obviously a proprietary one and therefore not available for dissection. Clearly it does produce flyable drape surfaces and these are workable in practice.

The aim of this document goes well beyond the capabilities of what has been described above and what is embodied in the SGDrape system: instead of just creating a drape surface, the intention is to develop a method which creates the **best** drape surface, ie the surface whose total deviation from the ideal (but unflyable) surface is minimised. It does not appear that this question has ever been addressed in the past. Also, it does not appear that the techniques of LP have ever been brought to bear upon the problem. Once again, this work attempts to bring the philosophy of Operations Research to bear on the problem – the finding of the best drape surface under a large number of constraints which should result in the optimisation of the quality of data obtained when the survey is flown.

2.3 Mathematical Description – Continuous Model

The drape problem can be stated mathematically as follows:

Given a topographical surface D over an area of interest A, we wish to construct a continuous surface Z such that

$$Z \geq D + m$$

for all points of A, where m is the minimum flying height.

We assume that flying will proceed parallel to the Cartesian axes when the area is viewed from above, that the maximum rate of climb/descent is α' and that the maximum absolute rate of change of slope over a peak is β'_u (u for upper), although the actual rate of change over the peak is negative. We also assume that the maximum rate of change of slope in a trough is β'_l (l for lower). Then the slope constraints are

$$\left|\frac{\partial Z}{\partial x}\right| \leq \alpha'$$

and

$$\left|\frac{\partial Z}{\partial y}\right| \leq \alpha'$$

over the entire area A.

The second derivative constraints are

$$\frac{\partial^2 Z}{\partial x^2} \geq -\beta'_u$$

and

$$\frac{\partial^2 Z}{\partial x^2} \leq \beta_l'$$

for peaks and valleys, respectively, in the x direction, and

$$\frac{\partial^2 Z}{\partial y^2} \geq -\beta_u'$$

and

$$\frac{\partial^2 Z}{\partial y^2} \leq \beta_l'$$

for peaks and valleys, respectively, in the y direction.

A drape surface can be considered optimal if it is as "low" as possible. Hence we wish to minimise the difference between the drape surface and the topographical surface, ie

$$\min \int_A (Z - D - m) dA ,$$

and, since m is constant, the objective can be written as

$$\min \int_A (Z - D) dA .$$

2.4 Mathematical Description – Discrete Model

In practice, topographical data are not available as continuous functions but rather as function values $D(x, y)$ for regularly spaced positions of A. This form of data is normally known as a digital terrain model (DTM) or digital elevation model (DEM). In this case the drape surface can be approximated by a raster surface over A with cells coinciding with those of the topographical grid.

Assume square cells of size $h \times h$. The slopes can now be approximated,

$$\left.\frac{\partial Z}{\partial x}\right|_{(x,y)} \text{ by } \frac{Z_{(x+h,y)} - Z_{(x,y)}}{h}$$

and

$$\left.\frac{\partial Z}{\partial y}\right|_{(x,y)} \text{ by } \frac{Z_{(x,y+h)} - Z_{(x,y)}}{h}.$$

In the same manner the second derivatives can be written as

$$\left.\frac{\partial^2 Z}{\partial x^2}\right|_{(x,y)} \approx \frac{\left.\frac{\partial Z}{\partial x}\right|_{(x,y)} - \left.\frac{\partial Z}{\partial x}\right|_{(x-h,y)}}{h} \approx \frac{\frac{Z_{(x+h,y)} - Z_{(x,y)}}{h} - \frac{Z_{(x,y)} - Z_{(x-h,y)}}{h}}{h} = \frac{Z_{(x-h,y)} - 2Z_{(x,y)} + Z_{(x+h,y)}}{h^2}$$

and

$$\left.\frac{\partial^2 Z}{\partial y^2}\right|_{(x,y)} \approx \frac{\left.\frac{\partial Z}{\partial y}\right|_{(x,y)} - \left.\frac{\partial Z}{\partial y}\right|_{(x,y-h)}}{h} \approx \frac{\frac{Z_{(x,y+h)} - Z_{(x,y)}}{h} - \frac{Z_{(x,y)} - Z_{(x,y-h)}}{h}}{h} = \frac{Z_{(x,y-h)} - 2Z_{(x,y)} + Z_{(x,y+h)}}{h^2}$$

The constraints for the discrete model can then be written as

$$\left|Z_{(x+h,y)} - Z_{(x,y)}\right| \leq \alpha' h$$

and

$$\left|Z_{(x,y+h)} - Z_{(x,y)}\right| \leq \alpha' h$$

for the slope constraints, and

$$Z_{(x-h,y)} - 2Z_{(x,y)} + Z_{(x+h,y)} \leq \beta_l' h^2$$

$$-Z_{(x-h,y)} + 2Z_{(x,y)} - Z_{(x+h,y)} \leq \beta_u' h^2$$

$$Z_{(x,y-h)} - 2Z_{(x,y)} + Z_{(x,y+h)} \leq \beta_l' h^2$$

$$-Z_{(x,y-h)} + 2Z_{(x,y)} - Z_{(x,y+h)} \leq \beta_u' h^2$$

for the second derivative constraints. Throughout the text which follows the expression $\alpha' h$ will be replaced by α, while $\beta_u' h^2$ and $\beta_l' h^2$ will be replaced by β_u and β_l respectively.

In the discrete case the objective function becomes

$$\min \sum_A (Z - D) .$$

2.5 Formulation as a Linear Program

The form of the constraints and the objective function described above naturally lead to the formulation of the problem of finding a drape surface for a given topography as a standard linear programming (LP) problem. This takes the following form:

Consider a rectangular survey area A of I rows and J columns whose topographic height $d_{i,j}$ is defined at all points (i, j) where $i = 1,...,I$ and $j = 1,...,J$. Let $z_{i,j}$ be the height of the drape surface to be determined at each point.

The slope constraints between the point (i, j) and its four neighbours, insofar as they exist, can then be written as:

$$z_{i,j} - z_{i+1,j} \leq \alpha$$

$$z_{i+1,j} - z_{i,j} \leq \alpha$$

$$z_{i,j} - z_{i-1,j} \leq \alpha$$

$$z_{i-1,j} - z_{i,j} \leq \alpha$$

$$z_{i,j} - z_{i,j+1} \leq \alpha$$

$$z_{i,j+1} - z_{i,j} \leq \alpha$$

$$z_{i,j} - z_{i,j-1} \leq \alpha$$

$$z_{i,j-1} - z_{i,j} \leq \alpha .$$

Examination of these constraints reveals that each slope actually occurs twice. For example, with $i = 3$ and $j = 6$, the first constraint becomes $z_{3,6} - z_{4,6} \leq \alpha$, while the values $i = 4$ and $j = 6$ in the fourth constraint produce exactly the same inequality. It therefore suffices to only consider the slope constraints which exist between a point and its eastern and southern neighbours.

The constraints for the second derivatives at a point (i, j) are

$$z_{i-1,j} - 2z_{i,j} + z_{i+1,j} \leq \beta_l$$

$$-z_{i-1,j} + 2z_{i,j} - z_{i+1,j} \leq \beta_u$$

for $1 < i < I$ and $1 \leq j \leq J$, and

$$z_{i,j-1} - 2z_{i,j} + z_{i,j+1} \leq \beta_l$$

$$-z_{i,j-1} + 2z_{i,j} - z_{i,j+1} \leq \beta_u$$

for $1 \leq i \leq I$ and $1 < j < J$.

Obviously, these constraints only apply where the subscripts refer to existing grid coordinates, hence the limits as shown. In the text which follows in the rest of this document, terms which refer to any point outside the grid limits are assumed to be zero. This should save the reader the tedium of having to consider the different subscript limits for various similar constraints.

Every point on the drape surface must be at least m above the ground; hence

$$z_{i,j} \geq d_{i,j} + m$$

for all points (i, j).

Since m is constant the above constraints can be simplified to

$$z_{i,j} \geq d_{i,j}.$$

This is equivalent to lifting the drape surface by m at every point after it has been calculated.

The objective is to create as "low" a drape surface as possible, in other words, to minimise the total height of the drape surface. Mathematically, then, we wish to

$$\min \sum_{i,j} (z_{i,j} - d_{i,j} - m) .$$

Since $\sum_{i,j} d_{i,j}$ and m are constant, the objective function can be simplified to:

$$\min \sum_{i,j} z_{i,j} .$$

The LP can now be written in the following standard form with $\leq$ constraints and a maximisation objective function:

$$\max \left(-\sum_{i,j} z_{i,j}\right) \tag{2.1}$$

subject to

$$z_{i,j} - z_{i+1,j} \leq \alpha \qquad i = 1,\ldots,I-1, \quad j = 1,\ldots,J \tag{2.1a}$$

$$z_{i+1,j} - z_{i,j} \leq \alpha \qquad i = 1,\ldots,I-1, \quad j = 1,\ldots,J$$

$$z_{i,j} - z_{i,j+1} \leq \alpha \qquad i = 1,\ldots,I, \quad j = 1,\ldots,J-1$$

$$z_{i,j+1} - z_{i,j} \leq \alpha \qquad i = 1,\ldots,I, \quad j = 1,\ldots,J-1$$

$$z_{i-1,j} - 2z_{i,j} + z_{i+1,j} \le \beta_l \qquad i = 2,...,I-1, \quad j = 1,...,J \qquad (2.1b)$$

$$-z_{i-1,j} + 2z_{i,j} - z_{i+1,j} \le \beta_u \qquad i = 2,...,I-1, \quad j = 1,...,J$$

$$z_{i,j-1} - 2z_{i,j} + z_{i,j+1} \le \beta_l \qquad i = 1,...,I, \quad j = 2,...,J-1$$

$$-z_{i,j-1} + 2z_{i,j} - z_{i,j+1} \le \beta_u \qquad i = 1,...,I, \quad j = 2,...,J-1$$

$$-z_{i,j} \le -d_{i,j} \qquad i = 1,...,I, \quad j = 1,...,J \qquad (2.1c)$$

with

$$\text{all variables} \ge 0.$$

In the LP statement above, constraints (2.1a) are the slope constraints, (2.1b) the second derivative constraints and (2.1c) the height constraints.

The possibility of using other differencing schemes should be addressed here. It is clear that the formulation above uses the so-called forward difference for the first derivative (Shodor Education Foundation). Other possibilities for approximating slopes with difference formulas do exist, eg the three-point and five-point formulas. It was decided not to use these since they do not take into account the central value. Also, a situation is possible in which one of the more intricate formulas does not give an excessive first derivative while the simple forward difference formula does indicate that the slope is too high. Consider the following linear cut through the topography.

Point (i)	1	2	3	4	5	6	7
Height (z_i)	1,0	1,0	2,0	10,0	2,0	1,0	1,0

There is quite a high peak at point 4. In the table below the first derivative (z_i') is calculated using the forward difference (FD) and also using the three-point formula (3P)

$$z_i' \approx \frac{z_{i+1} - z_{i-1}}{2h}$$

where h is the spacing between successive points and assumed to be one in this example.

Point (i)	1	2	3	4	5	6	7
Height (z_i)	1,0	1,0	2,0	10,0	2,0	1,0	1,0
z_i' (FD)	0	1,0	8,0	-8,0	-1,0	0	
z_i' (3P)		0,5	4,5	0	-4,5	-0,5	

If the maximum permissible value of the slope, α, is 5,0, then, clearly, the forward difference slopes are unacceptably large at points 3 and 4, while the three-point formula's slopes are acceptable at all points. The five-point formula suffers from a similar flaw.

Since the slope between the said points **is** too large and thus could constitute a safety hazard, the use of the more elaborate formulas for approximating the slope must be abandoned in favour of the simpler forward difference formula introduced initially. Of course, if three-point and five-point formulas are to be used the changes to the LP model are simple enough to incorporate.

2.6 Modelling Assumptions

Throughout the previous sections the search for optimality has been stressed. It remains for the concepts to be placed into the correct frame of reference.

The entire process of conducting a survey with drape flying involves the following steps in the order given:

- Choice and/or delimiting of survey area

- Choice of line direction
- Obtaining of digital terrain model (DTM)
- Generation of drape surface based on aircraft performance, line direction and DTM
- Flying of survey along drape surface.

It is very important to bear in mind that, while the terrain height is obviously a continuous function of position, the DTM is not. Instead, it merely represents a discretised set of values.

Similarly, the measurements taken along traverse and tie lines in the survey are taken at more or less regular intervals, usually several times a second. Although the acquisition equipment is operating "continuously", the acquisition of an infinitely divisible set of continuous data is not possible. There is currently no digital method for doing this; furthermore, some measurements rely on the acquisition equipment running for a period of time before a reading can be taken. An example of this is a spectrometer where the energies of all particles detected are summed and binned into channels. This cannot be done in an infinitesimally short length of time.

The data are thus acquired at discrete points along the survey lines. However, there is no practical way of determining *a priori* where these points of measurement will lie. It is merely known that the measurement points lie at a spacing of, say, 5 to 7 metres along a survey line. Furthermore, not all measuring instruments work in step – some are free-running, eg radar altimeters, others may be pulsed by a central clocking system. This means that the points of measurement for different instruments will not necessarily coincide. Because of the high density of readings this does not cause any processing

problems; however, it does re-emphasise that the exact location of a reading cannot be determined while planning the survey.

Although every flight line is planned in advance, a variety of factors can cause the aircraft to deviate off this course. These include turbulence, downdraughts, pilot fatigue, manoeuvring and so on. These deviations are usually quite small but they are unpredictable. Once again it is made clear that the exact location of measurements cannot be predetermined.

Since the exact locations of the measurements are unknown, the optimisation of a function defined only at these points is impossible. Instead, the current document discusses methods of finding a drape surface over the area of interest which will ensure feasibility regardless of where the flight lines finally end up. This drape surface is then optimised over the entire terrain using all the information available, ie it should utilise data from all points of the DTM. In this way the final location of the measurements and flight lines is immaterial.

2.7 Practical Considerations

Flight direction. At times it may occur that the coordinate axes in which the topographical grid is defined do not run parallel to the flight directions. In this case the grid should be resampled so that the axes do run parallel to the flight directions. This means that a new DTM is constructed whose axes are correctly positioned. Throughout the rest of this document it is assumed that such resampling has been undertaken where necessary.

Choice of topographical height for a cell. Due to the discrete nature of the DTM each spot height $d_{i,j}$ should be representative of the elevation of all the points in a cell in some meaningful manner. In normal circumstances, the average elevation of all the points in the cell would be satisfactory, but there are cases where this approach can be dangerous.

Consider the case where a tall radio mast has been erected in an otherwise fairly flat countryside. The average height throughout the cell would not differ significantly from that of the flat area, yet the drape surface should be high enough to clear the top of the mast with safety. In the interests of safety each cell in the topographical grid should be assigned the highest spot height within that cell.

Null cells. It may occur that the survey area is not rectangular and that the enclosing rectangle contains areas for which no DTM is available. In practical raster handling software such areas are made to consist of so-called null or dummy cells. This means that the grid values there are not known; it does not mean that the grid values are zero. For clarity and simplicity it will be assumed in this document that no grid under consideration contains any nulls. The methods described can, however, easily be adjusted to handle them. This is done by treating cells which adjoin dummy cells in the same way as cells are treated which are on the edge of the area of interest. Typically, then, the difference formulas will not give a first derivative approximation at (i, j) if $(i, j+1)$ is a null cell. Similarly, the difference formula for the second derivative approximation will fail at (i, j) if either $(i, j-1)$ or $(i, j+1)$ is a null cell.

Area for turns. In practice, an aircraft performing a survey flies a straight line to the edge of the survey area, then executes a turn outside the area and reenters it in a straight line on the next line. Although the ground clearance maintained during the turn is not important for subsequent data processing (although data from the turns are often used), it must nevertheless be possible for the pilot to execute the turn in safety. This would usually mean that the area outside the survey area should also be fitted with a drape surface. Once again, the safety of the aircraft during turns falls beyond the scope of this document; however, the methods discussed can easily be adapted to address the matter.

Chapter 3
The First Order Drape Problem

3.1 General Discussion

As a first approach to a relatively complex problem with very many variables and constraints, it was decided to first solve the drape problem with only the slope and height constraints in place, while ignoring the second derivative constraints for the moment. The solution of this "simplified" problem was also found to be a good starting point for many heuristics.

The primal problem was previously given as equations (2.1a) and (2.1c). In order to appreciate the number of variables and constraints in even this "simplified" problem, consider a DTM with I rows and J columns. The problem of generating a drape surface over this area would require an LP with IJ variables and

$J(I-1)+J(I-1)+I(J-1)+I(J-1)=4IJ-2(I+J)$ slope constraints, and

IJ height constraints,

giving a total of $5IJ-2(I+J)$ constraints.

A typical grid of, for example, 1 000 rows and 1 000 columns would thus require 1 000 000 variables and close on 5 000 000 constraints. Solving a problem of this size using the primal simplex method is a rather daunting task, even given the necessarily well-configured computing machinery and the required time.

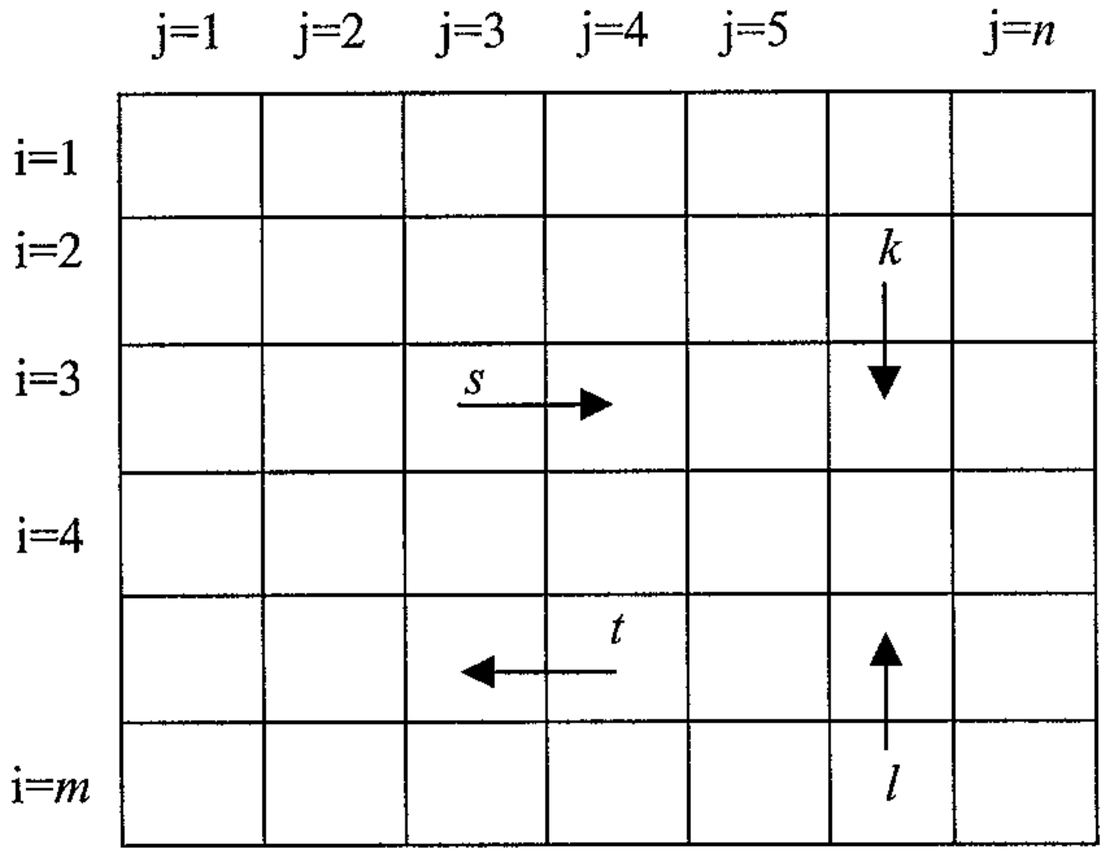
Another way to attack this problem would be to use a dual method. There is some consolation here – the dual problem would have some 5 000 000 variables but "only" 1 000 000 constraints. This is a considerable improvement since the number of iterations for large LPs is typically proportional to the number of constraints [Chvátal, p. 45].

The solution time using a dual method is still excessive and well beyond the reach of current desktop computing machinery. Nevertheless a discussion of the dual of the drape problem is in order since it is necessary for the understanding of the Lifting Algorithm to be described later in this chapter.

3.2 The Dual of the Drape Problem

When an LP problem (the primal) is converted to its dual, there is a one-to-one correspondence between the variables of the primal and the constraints of the dual. Likewise, there is a one-to-one correspondence between the constraints of the primal and the variables of the dual.

Before giving the correspondence between dual variables and primal constraints, the indices used for addressing grid cells should be noted carefully. The first index indicates the row in the grid (increasing to the bottom of the page ["south"]). The second index indicates the column in the grid (increasing to the right ["east"]). The grid indices follow those normally used for matrix operations; they are not Cartesian coordinates in the normal sense of x and y-axes. Fortunately the drape problem does not change under the operation of swopping or reversing the axes. The diagram below indicates the indexing system used.



The arrows inserted are examples of the constraints in the table below. All arrows begin on a higher-lying point and indicate a direction of descent. The following dual variables are used, corresponding to the constraints of the primal:

Constraint	Type	Dual variable
$z_{i,j} - z_{i+1,j} \leq \alpha$	Downhill y	$k_{i,j}$
$-z_{i,j} + z_{i+1,j} \leq \alpha$	Uphill y	$l_{i,j}$
$z_{i,j} - z_{i,j+1} \leq \alpha$	Downhill x	$s_{i,j}$
$-z_{i,j} + z_{i,j+1} \leq \alpha$	Uphill x	$t_{i,j}$
$-z_{i,j} \leq -d_{i,j}$	Height	$w_{i,j}$

The dual problem can now be formulated.

$$\min \alpha \sum_{i,j} (k_{i,j} + l_{i,j} + s_{i,j} + t_{i,j}) - \sum_{i,j} d_{i,j} w_{i,j} \tag{3.1}$$

subject to

$$k_{i,j} - k_{i-1,j} - l_{i,j} + l_{i-1,j} + s_{i,j} - s_{i,j-1} - t_{i,j} + t_{i,j-1} \geq -1$$

with

$$\text{all variables} \geq 0 .$$

To save the reader the tedium of the different limits for the subscripts of the various terms, it is assumed that terms which relate to any point (i, j) outside the grid limits are set to zero. The same applies to slopes which involve such points. For example, in the top left corner of the grid where $i = 1$ and $j = 1$, the constraint

$$k_{1,1} - k_{0,1} - l_{1,1} + l_{0,1} + s_{1,1} - s_{1,0} - t_{1,1} + t_{1,0} \geq -1$$

has $k_{0,1} = l_{0,1} = s_{1,0} = t_{1,0} = 0$ and, in effect, becomes

$$k_{1,1} - l_{1,1} + s_{1,1} - t_{1,1} \geq -1 .$$

There is exactly one constraint in the dual problem for every point in the grid.

A primal method would begin with a drape surface which was feasible and gradually modify this to become optimal. As a starting point one could use a flat surface which goes through the highest point in the topographical surface. Clearly such a surface is higher than the topography at all places and does not violate any slope constraints. This surface can now be depressed downwards where possible, in such a way that the primal constraints are never violated. The image is very much one of a tablecloth being spread over an uneven surface.

Conversely, a dual method would begin with a dual feasible but primal infeasible surface and gradually modify it until optimality was reached (where primal as well as dual feasibility would be achieved). One way of implementing such a method is to begin with a drape surface which is identical to the topography. In areas of rugged terrain the slope constraints might be overstepped and the drape surface will typically not be primal feasible. The surface is now raised (by the smallest amounts necessary) in areas of infeasibility until the entire drape surface is feasible. This will form pyramids around the highest peaks of the topography.

It is such a dual method which is discussed below.

3.3 The Lifting Algorithm

We associate a state variable with each point in the grid. This takes on one of three states: *untouched*, *set* and *fathomed*. A point in the *untouched* state has a drape height equal to

the topographic height and its influence on neighbouring points has not yet been evaluated. A point in the *set* state has its final drape height, but it has not yet been used to adjust the values of other points. A point in the *fathomed* state has its final drape height set and its influence on neighbours has been fully fathomed.

The Lifting Algorithm

Step 1 (*Set up initial surface*): Set $z_{i,j} = d_{i,j}$ for all points (i, j) in the grid. This implies that all the height constraints will be satisfied.

Step 2 (*Set initial state of points*): Set the state of all points (i, j) to *untouched*, except for the highest point in the grid, for which the state is set to *set*. It also becomes the first *current* point.

Step 3 (*Lift neighbours*): Examine each of the *untouched* neighbours of the *current* point. If a point is more than α lower than the *current* point, raise it so that the height differential between it and the *current* point is exactly α. Do not adjust any neighbours of the *current* point whose height differs from that of the *current* point by less than α. All neighbours now move into the *set* state, while the *current* point becomes *fathomed*.

Step 4 (*Find new current point*): Find the highest point (based on the current z-value) which is either *untouched* or *set*. This is the new *current* point. Go to step 3. If all points have been *fathomed* the algorithm terminates (in the next step).

Step 5 (*Termination*): The drape surface has been determined. The objective function value is taken to be the sum of the elevations of all the points of the drape surface.□

In practice the result of this method is such that the drape surface is initially constructed in regions which emanate from the peaks of the topographical surface. Each such region is primal feasible, and the regions eventually join up so that the entire drape surface

becomes feasible. In the final drape surface, each point is either at its original height or it has been raised and maintains a maximum-slope relation with at least one of its higher-lying neighbours.

The Lifting Algorithm starts out with a primally infeasible surface which undergoes modification until it is feasible. If it could be matched step by step by a dual algorithm which starts out with a feasible dual and maintains this dual feasibility until primal feasibility has also been reached, and if complementary slackness holds, then the duality theorem would ensure optimality.

Before proving that this is, in fact, the case, a numerical example is given to facilitate the reader's understanding of the procedure.

3.4 Example of the Lifting Algorithm

The following small example (using only whole numbers for simplicity) may serve to illustrate the Lifting Algorithm. The parameter α , the maximum permissible slope in the drape surface, has been set to 2. The following matrix shows the topographical 3×4 digital terrain model for which a drape surface is to be constructed.

7	10	13	6
4	10	12	7
11	8	6	14

In the following diagrams the *set* state is indicated by an asterisk while the *fathomed* state is shown with two asterisks. Unmarked cells are *untouched*. We begin with all points *untouched* except for the highest, (3;4), which is *set* (its z-value is 14).

7	10	13	6
4	10	12	7
11	8	6	14*

The *current* point is (3;4). Consider its two neighbours, (3;3) and (2;4). Both have an unacceptable height differential with the *current* point. Therefore they are both raised to a height α below the height of point (3;4), ie to a height of 12 units. Point (3;4) has been *fathomed*, and points (3;3) and (2;4) move into the *set* state.

7	10	13	6
4	10	12	12*
11	8	12*	14**

The next highest unfathomed point is (1;3) with a z-value of 13, and it becomes the *current* point. Consider its three neighbours, (1;2), (1;4) and (2;3). The height difference between (1;3) and (2;3) is only one unit and thus acceptable. For the other two neighbours this is not the case; therefore points (1;2) and (1;4) must be raised to a height of 11. Point (1;3) becomes *fathomed*, the three neighbours [(1;2), (1;4) and (2;3)] become *set*.

7	11*	13**	11*
4	10	12*	12*
11	8	12*	14**

The method continues in this fashion until all points have been *fathomed* and the final result is obtained, as shown below.

9**	11**	13**	11**
9**	10**	12**	12**
11**	10**	12**	14**

3.5 Optimality of the Lifting Algorithm

It can be shown that the drape surface generated by the Lifting Algorithm is both feasible and optimal. For this purpose, consider the points (i, j) in the area under consideration and the heights $z_{i,j}$ that are allocated to them by the Lifting Algorithm.

Feasibility requires that both the height constraints and the slope constraints are satisfied. Since the starting values are all set to $d_{i,j}$ and can only be increased, the height constraints ($z_{i,j} \geq d_{i,j}$) are satisfied. The slope constraints can be represented by the requirement that no point has a lower neighbour with height difference exceeding α. Since the algorithm eliminates this possibility for a *current* point before marking it *fathomed* and stops only when all points have been *fathomed*, this is also satisfied.

To prove optimality, it is first shown that each point on the drape surface is either a member of set A or set B where $A = \{(i, j) | z_{i,j} = d_{i,j}\}$ and $B = \{(i, j) | z_{i,j} > d_{i,j}; z_{i,j}(\alpha)\}$. Here $z_{i,j}(\alpha)$ indicates that (i, j) is in a binding relationship with at least one of its neighbours, meaning that the height of that neighbour is exactly α larger than $z_{i,j}$.

The Lifting Algorithm starts with all points in A. It lifts a point only if it has a neighbour the height of which exceeds its own by α or more, and then lifts it to be in a binding

relationship with that point. This moves the point from set A to set B. This happens while the neighbour is the *current* point. When the neighbour becomes *fathomed* the point becomes *set*, and since the height of neither can be changed in later steps of the Lifting Algorithm it holds for the final drape surface. This means that once a point becomes an element of B it remains one. Consequently for the final $z_{i,j}$ values, every point is either in A or B.

Assume the surface is not optimal. Then there must exist a point (i, j) whose height $z_{i,j}$ can be lowered without sacrificing feasibility. This is only possible if it is both higher than its topographic height $d_{i,j}$ and not in a binding relationship with any higher neighbour (if a fixed substructure can be lowered this will imply the existence of such a point at its edge) and therefore (i, j) does not belong to A or B. This contradicts the fact that all (i, j) belong to A or B. The assumption is wrong. The surface is optimal. ■

3.6 Comparison with the Simplex Method

In order to numerically confirm the correctness of the drape surfaces generated using the Lifting Algorithm, it was coded and its results compared to that of commercially available software using the standard simplex method for solving linear programming problems. The commercial software library selected was XA by Sunset Software Technology, San Marino, CA, USA. This suite has remarkable abilities for handling sparse matrices and displays great speed in solving large LP problems encountered by us in practice. It also uses a "crashing" technique which ensures that a feasible solution is reached as soon as possible and thereby reduces the total optimisation time. The table below shows the results obtained for some (smaller) problems. The problems comprise both artificially generated data sets as well as real DTM data gathered during geophysical surveys.

Problem Size		Optimum using XA (standard LP)	Optimum using Lifting Algorithm
Rows × columns	No. points (n)		
4 × 5	20	183,0	183,0
16 × 16	256	4 091,48	4 091,33
32 × 32	1 024	16 487,3	16 487,1
64 × 64	4 096	65 757,1	65 757,2
100 × 100	10 000	161 232	161 233
200 × 50	10 000	161 023	161 024
128 × 128	16 384	263 526	263 529
100 × 200	20 000	321 635	321 639
200 × 200	40 000	643 406	643 414

As was to be expected, the answers obtained using the Lifting Algorithm are the same as those obtained using an exact "traditional" algorithm. There are, of course, some small variations between the answers. These are rounding errors but they are to be expected, considering the very large number of pivot operations usually performed by the LP software and the limited accuracy (about six decimal digits) in the normal computer representation of 32-bit floating point numbers.

3.7 The Equivalent Dual Algorithm

It is instructive to examine the behaviour of the equivalent dual problem during the operation of the Lifting Algorithm described above. For the sake of simplicity the one-dimensional equivalent of the drape surface problem is considered: given a section through a topographical surface we wish to construct a drape line above it whose slope never exceeds α . The primal problem is:

$$\max \; -\sum_{i=1}^{n} z_i$$

subject to

$$z_i - z_{i+1} \le \alpha \qquad i = 1,\ldots,n-1 \qquad \text{(downhill)}$$

$$z_{i+1} - z_i \le \alpha \qquad i = 1,\ldots,n-1 \qquad \text{(uphill)}$$

and

$$-z_i \le -d_i \qquad i = 1,\ldots,n \qquad \text{(height)}$$

and

$$\text{all variables} \ge 0,$$

with n the number of points in the linear array.

The corresponding dual problem is then:

$$\min \alpha \sum_{i=1}^{n-1} (s_i + t_i) - \sum_{i=1}^{n} d_i w_i$$

subject to

$$s_i - s_{i-1} - t_i + t_{i-1} - w_i \ge -1$$

with

all variables ≥ 0.

Once again, terms which relate to any point i outside the limits of the array become zero and are ignored.

Complementary slackness requires that

$$s_i(z_i - z_{i+1} - \alpha) = 0$$

$$t_i(z_{i+1} - z_i - \alpha) = 0$$

$$w_i(z_i - d_i) = 0$$

and

$$z_i(s_i - s_{i-1} - t_i + t_{i-1} - w_i + 1) = 0\,.$$

The Lifting Algorithm starts out by setting $z_i = d_i$ for all points i on the line. The height constraints are satisfied, and, since the heights can only be increased, these constraints remain satisfied throughout. This means that all n primal variables will always be positive. (Even if some heights were zero or negative, the entire topographical surface could be transformed to make all heights positive without changing the essence of the problem.) For complementary slackness to hold, all n dual constraints must consequently be active at every step. They can therefore be written as

$$s_i - s_{i-1} - t_i + t_{i-1} - w_i + 1 = 0\,,$$

that is, as an equality for the one-dimensional case.

3.8 Network Flow Analogy

If the dual is viewed as a network flow problem, the last constraint above can be used as a flow balancing constraint at point i, as shown below. This view is not necessary for

describing the dual or for proving optimality, but its illustrative value is nevertheless large.

Each point on the line is represented as a flow network node. The variables in the dual constraints can be viewed as flows (eg s_i is a flow from node i to node $i+1$); all the constraints together give rise to a network as shown below.

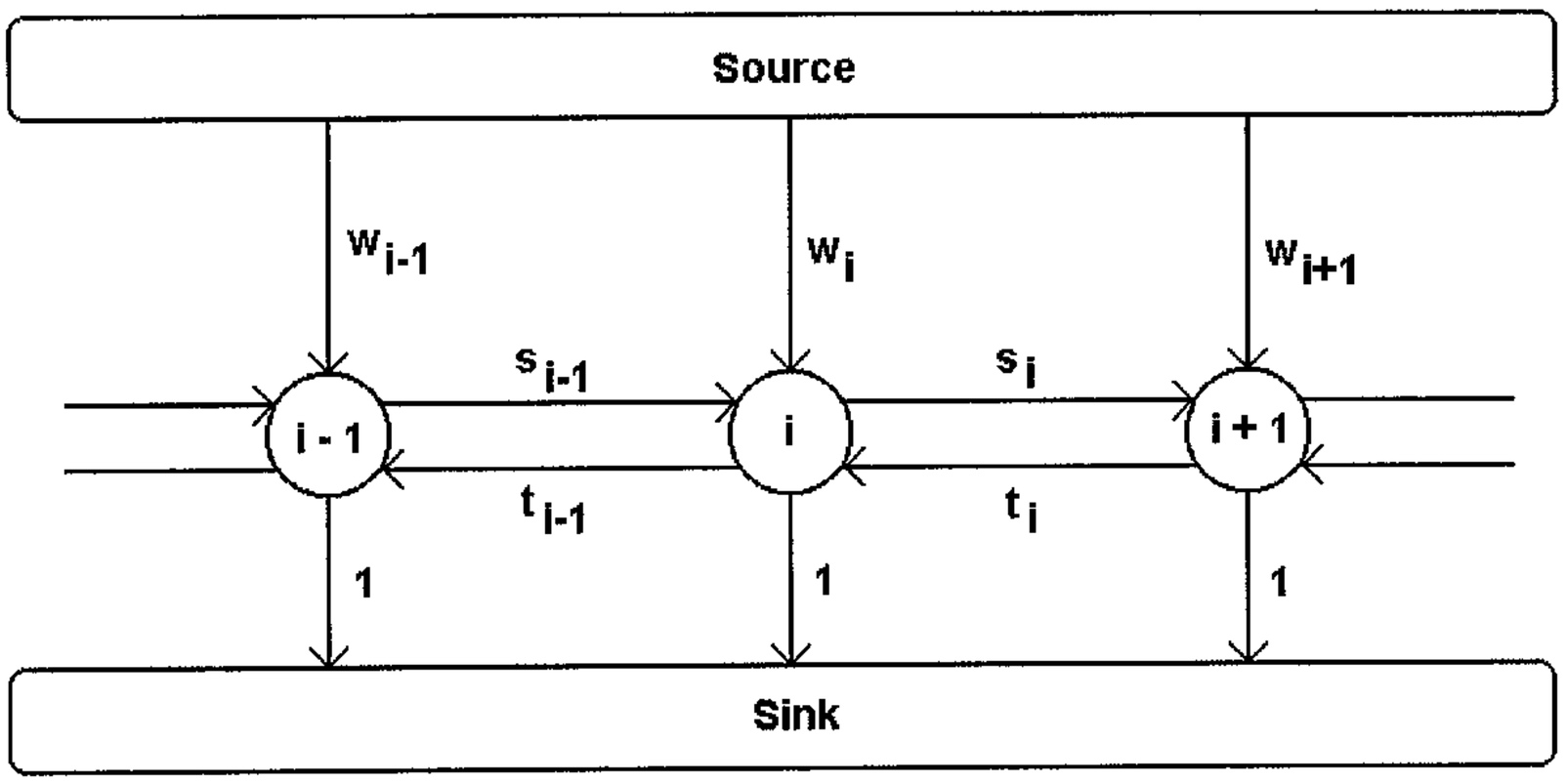


Figure 3.1. Network flow derived from dual problem

Each constraint constitutes a flow balance around one of the nodes in the network. The 1 in the flow balancing constraint is interpreted as an obligatory flow from node i into a common sink. The flow balance therefore also requires that the source supply precisely n units. Complementary slackness (and dual feasibility) are maintained throughout; this means that the flow in the network is always feasible and balanced around every node.

To begin with, the drape surface is set equal to the topographical surface. The equivalent dual solution, based on complementary slackness, has all w-variables set to one, and all others set to zero. This situation is illustrated below in Figure 3.2.

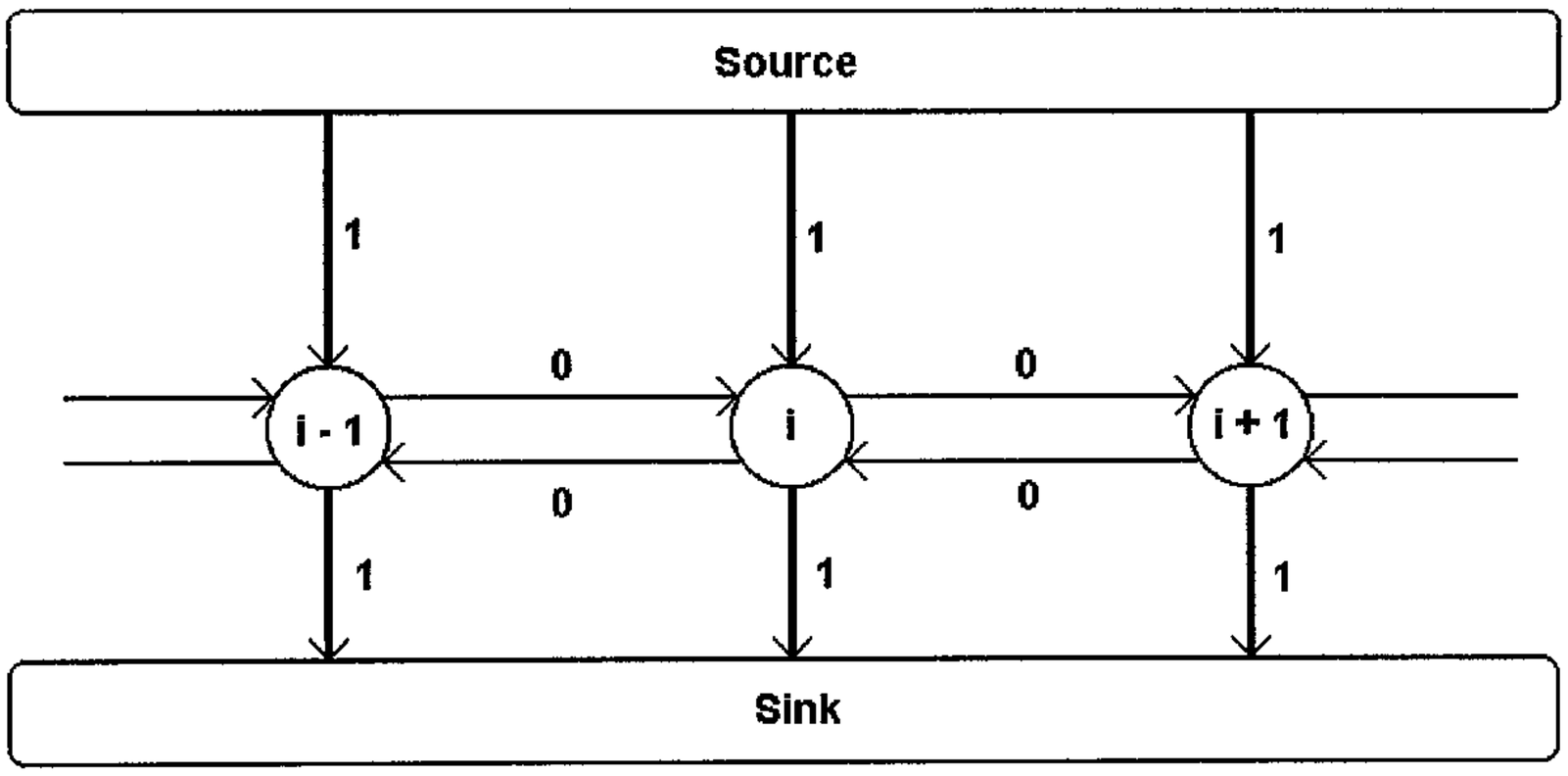


Figure 3.2. Flow corresponding to initial dual-feasible drape surface

It will now be demonstrated that each step of the Lifting Algorithm corresponds to a dual iteration.

Each point remains at its original height until it becomes *set*. This means that $z_i = d_i$ and it follows from $w_i(z_i - d_i) = 0$ that w_i is still basic. When the point becomes *set*, its drape height either remains equal to its topographical height (which does not change the corresponding dual solution), or it is raised to be α below a neighbour. This neighbour must obviously be in the *set* state, and therefore either have a height equal to the original height or in turn be α lower than another neighbour.

In the sketch above assume that the slope in the drape surface from point i to point $i+1$ is too steep ($z_{i+1} - z_i > \alpha$) where point $i+1$ is in the *set* state. The Lifting Algorithm will raise point i until the height differential between points i and $i+1$ is equal to α (ie $z_{i+1} - z_i = \alpha$). Since there is now positive slack in the height constraint the corresponding dual variable, w_i, must become zero (ie $z_i - d_i > 0$, requiring w_i to become zero to satisfy $w_i(z_i - d_i) = 0$). The uphill constraint now has zero slack and the corresponding dual variable, t_i, can assume a positive value (ie $z_{i+1} - z_i - \alpha = 0$, causing

$t_i(z_{i+1} - z_i - \alpha) = 0$ which allows t_i to become positive). To satisfy the flow constraint at i, t_i must be 1. Variable w_i has left the basis and has been replaced by t_i. Moving to node $i+1$, w_{i+1} must increase by one to maintain the flow balance. Note that w_{i+1} may be positive because point $i+1$ is still at its topographical height. The flows are now:

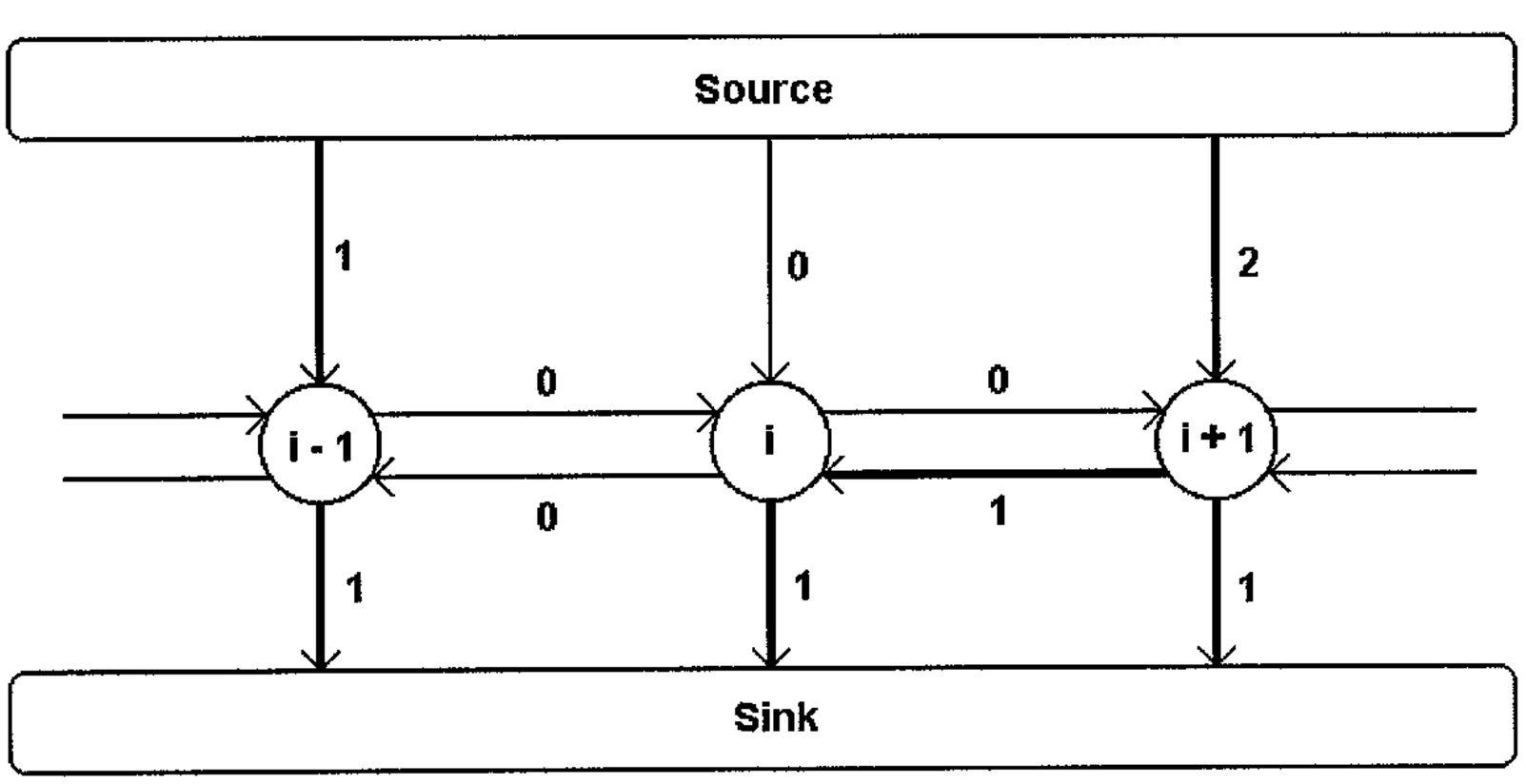


Figure 3.3. Point i is brought into a slope relation with point $i+1$

At this stage, points i and $i+1$ are in the *set* state. Assume that the slope between $i-1$ and i is too large now ($z_i - z_{i-1} > \alpha$). Point $i-1$ thus needs to be lifted. z_{i-1} is lifted to the point where $z_i - z_{i-1} = \alpha$, making $t_{i-1}(z_i - z_{i-1} - \alpha) = 0$ and allowing t_{i-1} to become positive. However, since $z_{i-1} - d_{i-1} > 0$, w_{i-1} must be zero for $w_{i-1}(z_{i-1} - d_{i-1}) = 0$ to hold. This brings t_{i-1} into the basis while w_{i-1} leaves it. To compensate, w_{i+1} and t_i must increase. (This does not threaten complementary slackness.)

The flow is now:

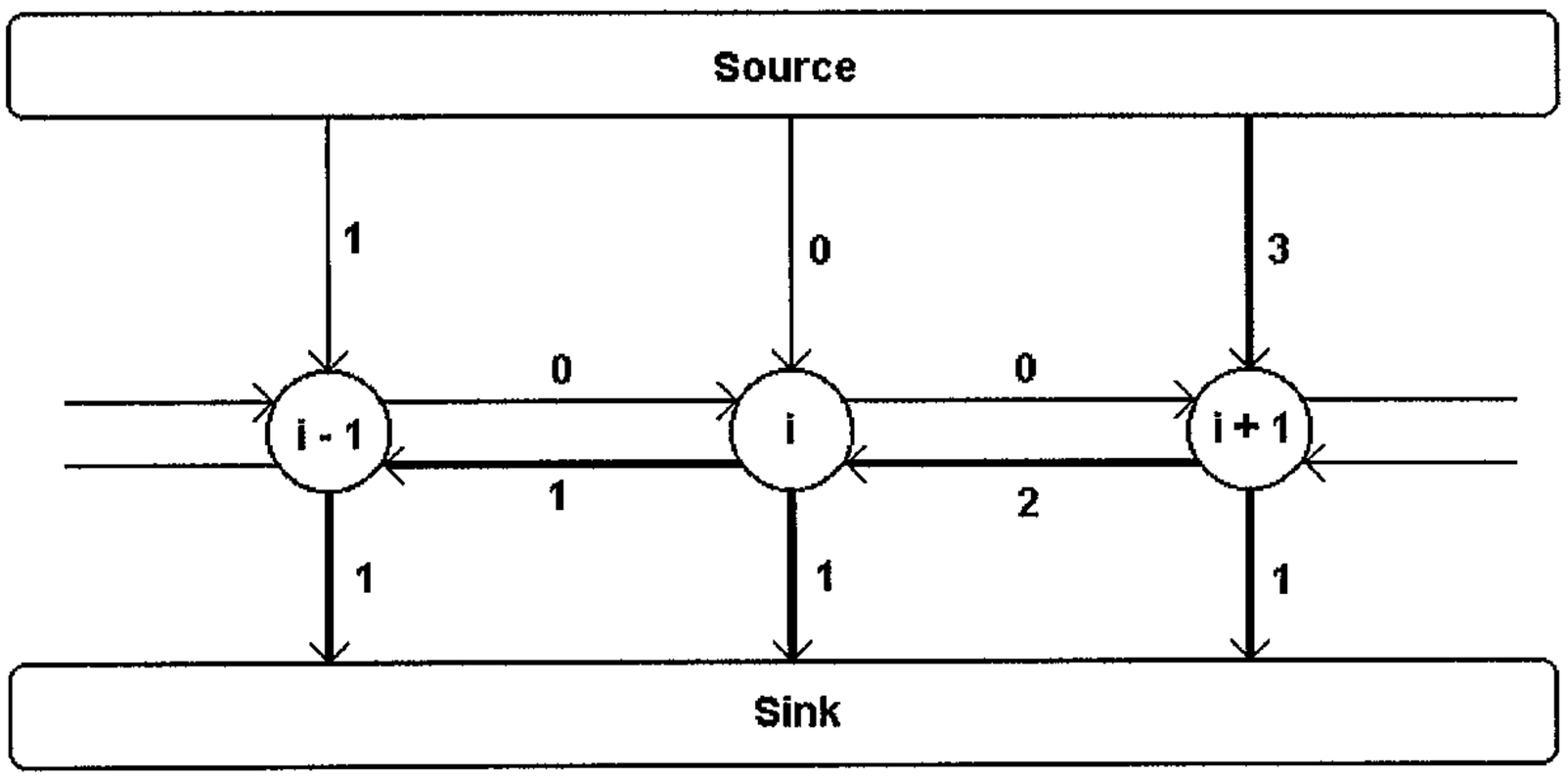


Fig 3.4. Point $i-1$ has also been raised

We have considered the case where the Lifting Algorithm removes an infeasibility to the left of a *set* point and have shown that this is matched by a dual iteration. Similarly, when the Lifting Algorithm removes an infeasibility to the right of a *set* point, a matching dual iteration using s instead of t can be constructed. Since the Lifting Algorithm terminates in a finite number of steps at a feasible primal solution, we have both primal and dual feasibility as well as complementary slackness at this point. The solution is optimal. ■

The extension of the above network flow representation to the two-dimensional problem is trivial, but not so easy to put down on paper. Every interior node is connected to its eastern and western neighbours by s and t arcs respectively, and to its northern and southern neighbours by k and l arcs respectively. Every node is also connected to the common source and sink in the same way as in the one-dimensional case.

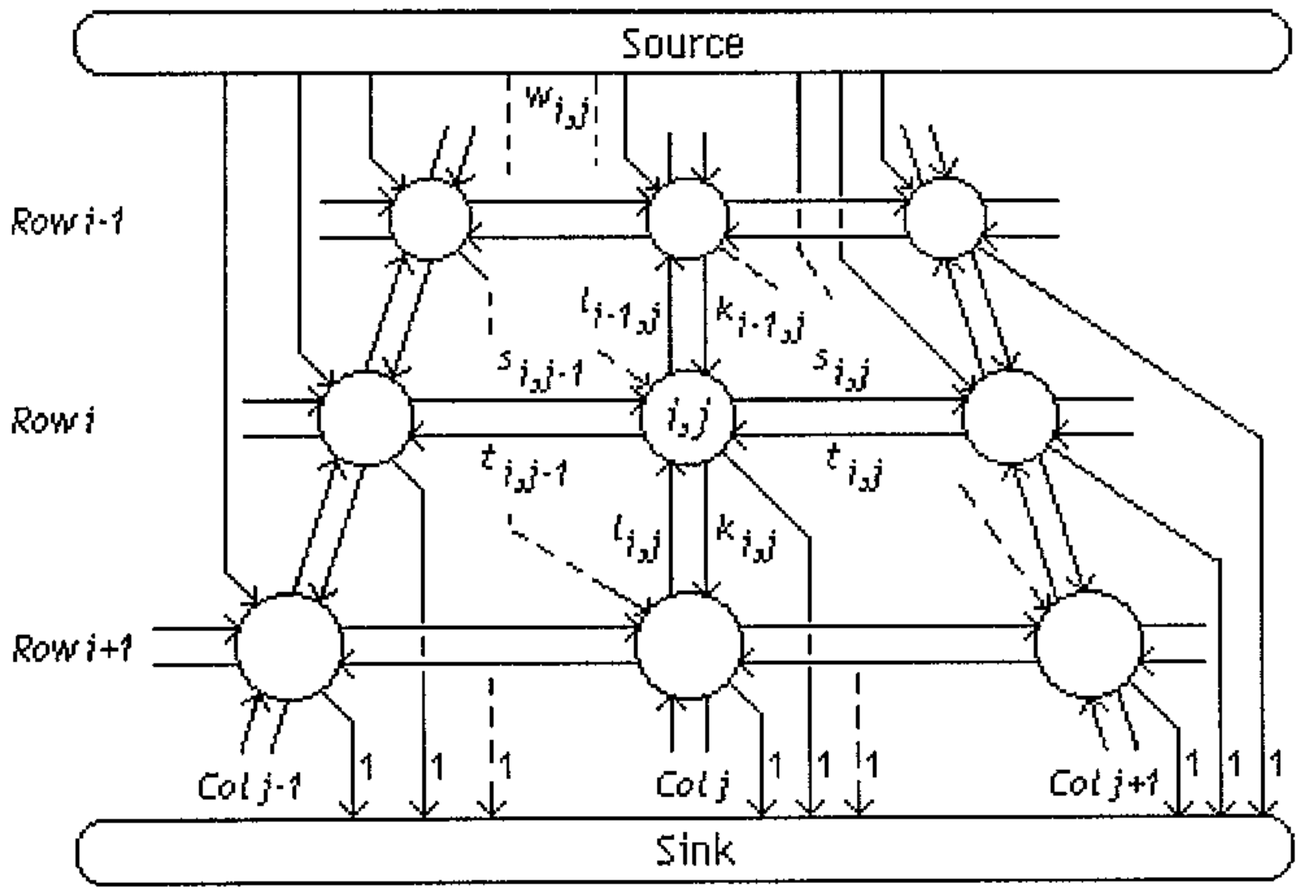


Figure 3.5. Flow diagram for the two-dimensional case

Following the same procedure, it can be shown that every step of the Lifting Algorithm can be matched by an iteration of the dual, maintaining dual feasibility and complementary slackness. The final primal solution is optimal.■

3.9 Enhancement of Method

Although the Lifting Algorithm is a single-pass method, at every iteration the highest unfathomed point in the grid must be found. The complexity of the method is therefore $O(n^2)$ rather than $O(n)$, where n is the number of points in the grid (the product of I and J).

However, it is possible to reach the same primal solution in the following way:

The Enhanced Lifting Algorithm

Step 1 (*Initial surface*): Set $z_{i,j} = d_{i,j}$ for all points (i, j) in the grid. This implies that all the height constraints will be satisfied.

Step 2 (*Set starting point*): Begin with the top left point as the *current* point – here the point (1;1).

Step 3 (*Check height differential on right*): Examine the height difference between the *current* point and its right-hand neighbour, ie $(i, j+1)$. If there is no right-hand neighbour, ignore the rest of this step. If the height difference between the *current* point and the neighbour exceeds α, raise the lower of the two points, so that the difference in question becomes exactly α.

Step 4 (*Check height differential on bottom*): Repeat the process of step 3 with the bottom neighbour of the *current* point, ie $(i + 1, j)$, provided this exists.

Step 5 (*Advance to next point*): Move the *current* point one to the right. If the *current* point is the last one of the current line, move the *current* point to the left-most point of the next line. If the *current* point is the bottom right point of the grid, go on to step 6. Otherwise repeat the procedure for the *current* point from step 3 onwards.

Step 6 (*Start point for reverse pass*): The forward pass has now been completed. Set the *current* point to the bottom right point in the grid, ie (I, J).

Step 7 (*Check height differential on left*): Examine the height difference between the *current* point and its left-hand neighbour, ie $(i, j-1)$. If there is no left-hand neighbour, ignore the rest of this step. If the height difference between the *current* point and the neighbour exceeds α, raise the lower of the two points, so that the difference in question becomes exactly α.

Step 8 (*Check height differential on top*): Repeat the process of step 7 with the top neighbour of the *current* point, ie $(i-1, j)$.

Step 9 (*Advance to next point*): Move the *current* point one to the left. If the *current* point is the first one of the current line, move the *current* point to the right-most point of the previous line. If the *current* point is the top left point of the grid, go on to step 10. Otherwise repeat the procedure for the *current* point from step 7 onwards.

Step 10 (*Termination*): The reverse pass has now been completed and the drape surface has been determined. The objective function value is taken to be the sum of the elevations of all the points of the drape surface.■

This procedure raises the areas to the right of and below peaks in the original topographic surface during the forward pass. During the reverse pass areas to the left and above peaks are also raised into pyramid-shaped sections of the drape surface.

3.10 Example using the Enhanced Method

In this section the same example as presented previously is used. For convenience the initial drape surface (same as topography) is shown again. Again the maximum permissible slope is $\alpha = 2$.

7	10	13	6
4	10	12	7
11	8	6	14

The forward pass begins at the top left cell. Observe that $z_{1,2} - z_{1,1}$ exceeds α. Therefore point (1;1), the lower of the two, needs to be raised so that $z_{1,2} - z_{1,1} = \alpha$. The *z*-values are now:

8	10	13	6
4	10	12	7
11	8	6	14

However, work on the first point is not finished yet. The height difference between the top left point and the one below it is also too large ($z_{1,1} - z_{2,1} > \alpha$). Hence point (2;1) must be raised. The surface is now:

8	10	13	6
6	10	12	7
11	8	6	14

The next point to be considered is (1;2) which currently has $z = 10$. At this point the difference between (1;2) and (1;3) must be adjusted, and (1;2) is raised by one unit.

8	11	13	6
6	10	12	7
11	8	6	14

Notice how this act of raising (1;2) reintroduces an infeasibility between (1;1) and (1;2) whose height differential has now become 3. This infeasibility will disappear during the reverse pass.

The height difference between (1;2) and (2;2) is only 1, and thus in order. Therefore the method continues to point (1;3). Here the height difference between points (1;3) and (1;4) is too large and point (1;4) must be raised. Once the procedure has finished with point (1;4) it advances to the next row, and point (2;1) is considered.

At the end of the forward pass, the grid's z-values are as shown in the following diagram:

8	11	13	11
9	10	12	12
11	9	12	14

The reverse pass begins by considering point (3;4). The height differences with its northerly and westerly neighbours are in order.

The next point in reverse reading order is (3;3). This point's height difference with its left-hand neighbour is too large, and point (3;2) must be raised:

8	11	13	11
9	10	12	12
11	10	12	14

Continuing in this manner we observe that no further adjustments need to be made until the very end. At this step point (1;2) is under consideration and we find the height differential between (1;2) and (1;1) (its left neighbour) exceeds α . Therefore point (1;1) is raised and the algorithm terminates with the solution shown below. This solution is identical to the one generated by the standard lifting algorithm.

9	11	13	11
9	10	12	12
11	10	12	14

3.11 Optimality of the Enhanced Method

After the Enhanced Method has been applied, the drape surface will be feasible. In other words, all height differences between pairs of adjacent points will be α or less. Observe that the process has a complexity of $O(n)$, since the time taken is directly proportional to the number of points in the grid, the size of the problem.

Theorem: The Enhanced Method produces a drape surface which is both feasible and optimal.

Proof: Since the intermediate steps of the Enhanced Method involve surfaces which may be primal infeasible, dual infeasible or both, consider only the surface which is obtained when the Enhanced Method has run to completion.

In this surface every point is in one of two states:

- State A: If a point is in state A it has never been lifted and thus $z_{i,j} = d_{i,j}$ for such a point.

- State B: Any point which is not in state A falls into this group. Points in state B have been lifted from their original topographic height to bring them into a relation of maximum height differential with one of their orthogonal neighbours.

A feasible dual solution can readily be constructed from the drape surface in which all points are in either state A or B, using the network analogy.

Begin by connecting all points in state A to the source. Next, connect all unconnected points (which are all in state B) to one of their respective neighbours with which they have a height differential of α. These connections can be seen as the dual basic variables in the optimal solution.

Once all the connections have been made, the following applies:

- With each connection one point is connected. Hence there are precisely as many dual variables as there are points.

- A feasible (non-negative) flow can readily be constructed such that each tree root (state A point) receives a flow equal to the total number of nodes connected to that root. The total flow is thus equal to the total number of nodes in the network.

Since a feasible primal and dual can be constructed which are in complementary slackness to each other, the solution must be optimal.■

3.12 Example of the Construction of a Dual Solution

To determine the flow in the optimal solution of the example solved in sections 3.4 and 3.8, we observe that $z_{i,j} = d_{i,j}$ for the points (1;3), (2;2) (2;3), (3;1) and (3;4). Therefore $w_{1,3}$, $w_{2,2}$, $w_{2,3}$, $w_{3,1}$ and $w_{3,4}$ are basic in the dual solution.

Next, point (1;2) can be connected to point (1;3) because the height difference between them is exactly α. This connection implies that $t_{1,2}$ is basic. We proceed in this fashion until all nodes are connected. One such dual solution is shown below (Figure 3.6).

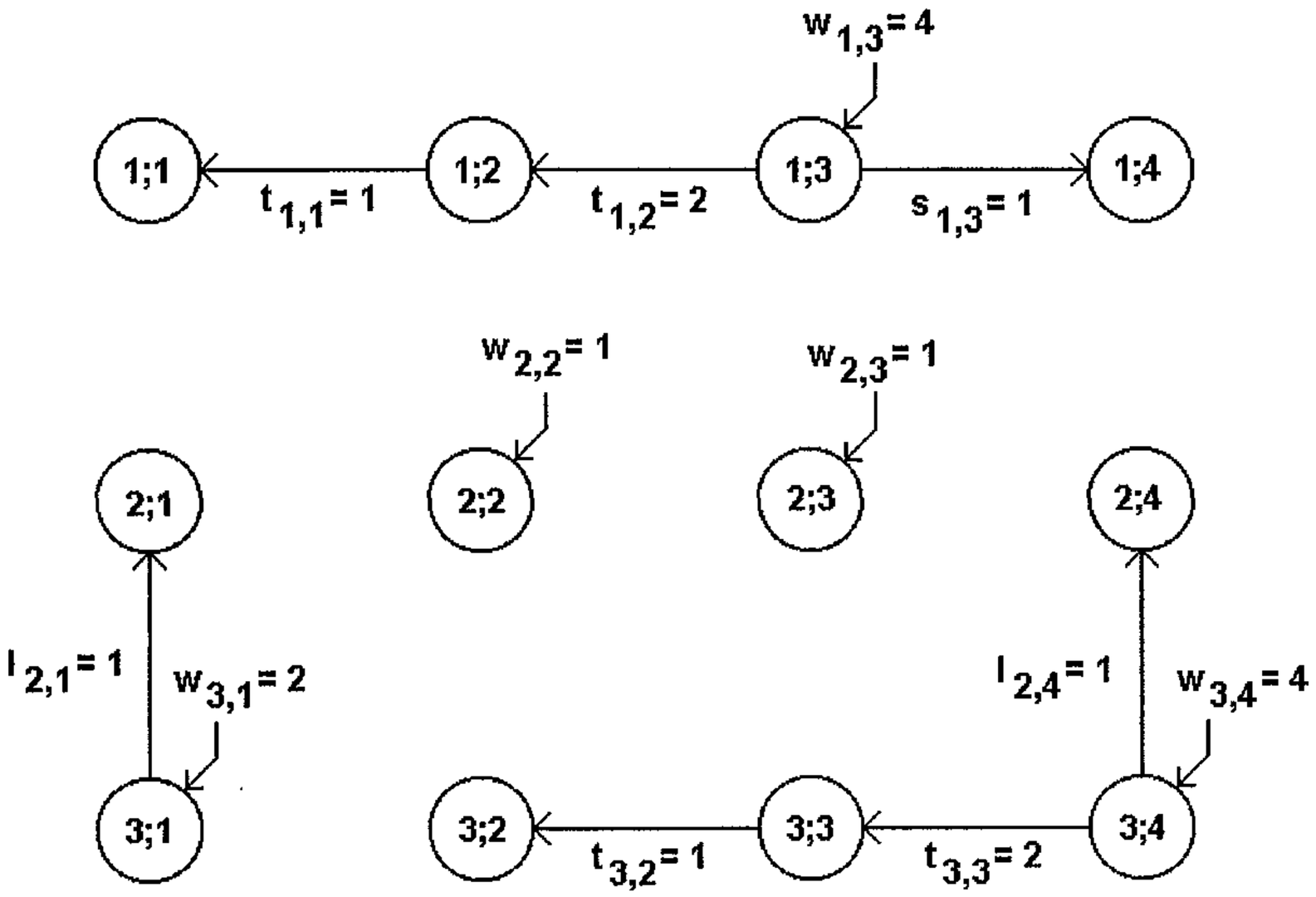


Figure 3.6. Dual network flow for the example

Only those arcs which are basic are shown in the figure, while the unit flows from the individual nodes to the sink have been left out altogether for the sake of clarity. There are exactly 12 dual basic flows; the flow in the network is balanced.

A final check can still be conducted on the problem. The matrix of z-values is

9	11	13	11
9	10	12	12
11	10	12	14

and the corresponding objective function value is $\sum_{i,j} z_{i,j} = 134$.

In the case of the dual, the objective function's non-zero terms are as follows (as per equation 3.1, but with reversed signs):

Dual variable	**Multiplier**	**Dual value**	**Multiplier value**	**Term value**
$w_{1,3}$	$d_{1,3}$	4	13	52
$w_{2,2}$	$d_{2,2}$	1	10	10
$w_{2,3}$	$d_{2,3}$	1	12	12
$w_{3,1}$	$d_{3,1}$	2	11	22
$w_{3,4}$	$d_{3,4}$	4	14	56
$t_{1,1}$	$-\alpha$	1	-2	-2
$t_{1,2}$	$-\alpha$	2	-2	-4
$s_{1,3}$	$-\alpha$	1	-2	-2
$l_{2,1}$	$-\alpha$	1	-2	-2
$t_{3,2}$	$-\alpha$	1	-2	-2
$t_{3,3}$	$-\alpha$	2	-2	-4
$l_{2,4}$	$-\alpha$	1	-2	-2

The sum of the entries in the right-hand column of the table is again 134.

3.13 Practical Results

Both the Lifting Algorithm and the Enhanced Method were coded and tried on a variety of examples. The table below gives execution times for the same datasets already mentioned in Section 3.6, in seconds for the Lifting Algorithm and in microseconds for the Enhanced Method. The more or less constant terms in the columns n^2/t_1 and n/t_2 demonstrate the complexities of $O(n^2)$ and $O(n)$ for the two methods respectively. The ratio indicated by a dash is not shown because of the dominance of measuring errors. All calculations were done on a 100 MHz Pentium with 64 MB of RAM.

Problem size		**Lifting Algorithm**		**Enhanced Method**	
Rows × Columns	**No. Points (*n*)**	**Time (t_1)** (s)	n^2/t_1	**Time (t_2)** (μs)	n/t_2
4 × 5	20	0		114	0,175
16 × 16	256	0		1 096	0,234
32 × 32	1 024	1	-	5 376	0,190
64 × 64	4 096	7	$2{,}4 \times 10^6$	18 678	0,219
100 × 100	10 000	41	$2{,}4 \times 10^6$	40 495	0,247
200 × 50	10 000	41	$2{,}4 \times 10^6$	40 542	0,247
128 × 128	16 384	115	$2{,}3 \times 10^6$	71 107	0,230
100 × 200	20 000	173	$2{,}3 \times 10^6$	92 306	0,217
200 × 200	40 000	709	$2{,}3 \times 10^6$	192 538	0,208

It is of interest to note that the execution time of the Enhanced Method for all the problems above was considerably less than one second. The execution speed for XA varied from a very comparable less than 1 second and 5 iterations for the 4 × 5 problem to a rather longer 6 580 seconds (about 1 hour 50 minutes) and 34 847 iterations for the 200 × 200 problem. Clearly, the Enhanced Method comes into its own as the problem size increases.

Chapter 4
The One-dimensional Drape Problem

4.1 General Discussion

Since the drape problem with only first derivative constraints had been solved very satisfactorily, as explained in the previous chapter, it was felt at first that the same problem with second derivative constraints added should be relatively straightforward too. After much exploring and analysis it became clear that the drape problem with both first and second derivative constraints was much more complex than that of the previous chapter.

In an effort to introduce method and a logical progression into the attempts at finding a solution procedure, it was decided to analyse the one-dimensional problem first. In this way something might be gleaned which would assist in the solution of the more complicated two-dimensional extension. The relaxation of the problem to one defined only in one dimension forms the content of this chapter; conceptually it treats a section through the drape surface. As it turned out later, the algorithm developed for this problem forms the basis of one of the best heuristic methods developed later.

4.2 Mathematical Statement

The one-dimensional drape problem with first and second derivative constraints can be formulated mathematically as follows:

Given a series of n equally-spaced points, each with a topographical elevation d_i, $(i = 1,...,n)$, we wish to find drape heights z_i at these points such that

$$z_i - z_{i+1} \le \alpha \qquad i = 1,\ldots,n-1 \quad \text{(downhill 1st derivative constraints)}$$

$$z_{i+1} - z_i \le \alpha \qquad i = 1,\ldots,n-1 \quad \text{(uphill 1st derivative constraints)}$$

$$z_{i-1} - 2z_i + z_{i+1} \le \beta_l \qquad i = 2,\ldots,n-1 \quad \text{(valley 2nd derivative constraints)}$$

$$-z_{i-1} + 2z_i - z_{i+1} \le \beta_u \qquad i = 2,\ldots,n-1 \quad \text{(peak 2nd derivative constraints)}$$

$$z_i \ge d_i \qquad i = 1,\ldots,n \qquad \text{(height constraints)}$$

and we wish to make the sum of the heights of all the points of the drape surface as small as possible, ie

$$\min \sum_{i=1}^{n} z_i \ .$$

The terms "uphill" and "downhill" are to be interpreted as the tendency of the drape surface with increasing i.

The LP can now be written in the following standard form with $\le$ constraints and a maximisation objective function.

$$\text{Max } (-\sum_{i=1}^{n} z_i) \tag{4.1}$$

subject to

$$z_i - z_{i+1} \le \alpha \qquad i = 1,\ldots,n-1 \tag{4.1a}$$

$$z_{i+1} - z_i \le \alpha \qquad i = 1,\ldots,n-1$$

$$z_{i-1} - 2z_i + z_{i+1} \le \beta_l \qquad i = 2,...,n-1 \qquad (4.1b)$$

$$-z_{i-1} + 2z_i - z_{i+1} \le \beta_u \qquad i = 2,...,n-1$$

$$-z_i \le -d_i \qquad i = 1,...,n \qquad (4.1c)$$

with

$$\text{all variables} \ge 0.$$

In the LP statement above, constraints (4.1a) are the slope constraints, (4.1b) the second derivative constraints and (4.1c) the height constraints.

We note that if $\beta_l \ge 2\alpha$ or $\beta_u \ge 2\alpha$, the first derivative constraints imply that the second derivative constraints are automatically satisfied. Should this situation arise, the relevant second derivative constraints become redundant and can be ignored. The problem therefore simplifies to that of finding only the first derivative drape surface. Since this is straightforward and has been discussed previously, it is assumed throughout the rest of this document that $\beta_l < 2\alpha$ and $\beta_u < 2\alpha$, unless otherwise stated.

4.3 Dual of the One-dimensional Problem

As in the case of the problem with only slope constraints, some insight may be gained by constructing the dual of the drape problem. Every constraint in the primal problem has a corresponding variable in the dual and *vice versa*. Considering the grid point indexed by i, let w_i be the dual variable corresponding to the height constraint $-z_i \le -d_i$. Assign further dual variables as follows:

Constraint	**Type**	**Dual variable**
$z_i - z_{i+1} \leq \alpha$	Downhill slope	s_i
$z_{i+1} - z_i \leq \alpha$	Uphill slope	t_i
$z_{i-1} - 2z_i + z_{i+1} \leq \beta_l$	Valley	u_i
$-z_{i-1} + 2z_i - z_{i+1} \leq \beta_u$	Peak	v_i

There is one dual variable associated with each slope constraint emanating from the current grid point i, one with each second derivative constraint and one with the height constraint at this point.

The dual problem can now readily be formulated as

$$\text{Min} \quad \alpha \sum_i (s_i + t_i) + \beta_l \sum_i u_i + \beta_u \sum_i v_i - \sum_i d_i w_i$$

subject to

$$s_i - s_{i-1} - t_i + t_{i-1} + u_{i-1} - 2u_i + u_{i+1} - v_{i-1} + 2v_i - v_{i+1} - w_i \geq -1 \qquad i = 1,\ldots,n$$

with

all variables ≥ 0.

To save the reader the tedium of the different limits for the subscripts of the various terms, we assume that terms which relate to any point i outside the (linear) grid limits are set to zero. The same applies to slopes and second derivatives which involve such points. For example, at the left-most point of the line where $i = 1$, the constraint

$$s_1 - s_0 - t_1 + t_0 + u_0 - 2u_1 + u_2 - v_0 + 2v_1 - v_2 - w_1 \geq -1$$

has $s_0 = t_0 = u_0 = v_0 = 0$. Also u_1 and v_1 are zero, because they span the three points 0, 1 and 2, of which the first does not exist. There are therefore no peak and valley constraints centred on the first and last points. So the constraint above becomes

$$s_1 - t_1 + u_2 - v_2 - w_1 \geq -1 .$$

There is exactly one constraint for every grid point.

Since the number of constraints in the primal problem is rather larger than the number of variables, it follows that the number of constraints in the dual is smaller than the number of variables. And, since the number of iterations of the simplex algorithm is usually proportional to the number of constraints, a simplex method working on the dual is preferable to one which works on the primal.

A dual method is thus proposed to solve the linear problem. As in the method utilised in solving the problem with only first derivatives, the initial drape surface is set to be the same as the topography of the area. This surface is guaranteed to be dual feasible because the height constraint for every point can be made active, ie all w_i's are set to one while all other dual variables are set to zero. The corresponding primal surface may or may not be feasible, depending on the ruggedness of the terrain. However, primal infeasibilities are easily detected.

The solution method proceeds in the normal fashion of a simplex method: given n points on the line, and thus n dual constraints, the solution at any stage comprises n basic dual variables. At each iteration a primal infeasibility is identified; the corresponding dual variable is chosen to enter the basis, while the leaving variable is the one in the basis which first disappears (reaches zero) as the entering variable grows in size. This method would be no different from the standard textbook method, were it not for the special structure of the problem which can be exploited to simplify and speed up the solution process. A network representation of the dual is used to develop the algorithm.

4.4 Network Flow Representation

Complementary slackness between the primal and dual problems requires that

$$s_i(z_i - z_{i+1} - \alpha) = 0 \qquad \text{(first derivative constraints)}$$

$$t_i(z_{i+1} - z_i - \alpha) = 0$$

$$u_i(z_{i-1} - 2z_i + z_{i+1} - \beta_l) = 0 \qquad \text{(second derivative constraints)}$$

$$v_i(-z_{i-1} + 2z_i - z_{i+1} - \beta_u) = 0$$

$$w_i(z_i - d_i) = 0 \qquad \text{(height constraints)}$$

and

$$z_i(s_i - s_{i-1} - t_i + t_{i-1} + u_{i-1} - 2u_i + u_{i+1} - v_{i-1} + 2v_i - v_{i-1} - w_i + 1) = 0\,.$$

Since z-values will, in general, not be zero and can, in fact, be set to be strictly positive by a simple transformation of the topographical surface, the last equation then requires that

$$s_i - s_{i-1} - t_i + t_{i-1} + u_{i-1} - 2u_i + u_{i+1} - v_{i-1} + 2v_i - v_{i-1} - w_i + 1 = 0$$

for all points. This means that all dual inequality constraints become strict equalities. The above constraint can be rewritten with the terms grouped such that they are all positive:

$$s_i + t_{i-1} + u_{i-1} + u_{i+1} + 2v_i + 1 = s_{i-1} + t_i + 2u_i + v_{i-1} + v_{i+1} + w_i\,.$$

As in the first-derivative solution, each constraint can be seen as a flow balance around a point. A network node represents point i on the line. Viewing the variables in the dual constraints as flows, the constraints can be represented as a network as shown below. The

terms on the left-hand side of the constraint above are outflows from node i while the right-hand side terms embody the inflow.

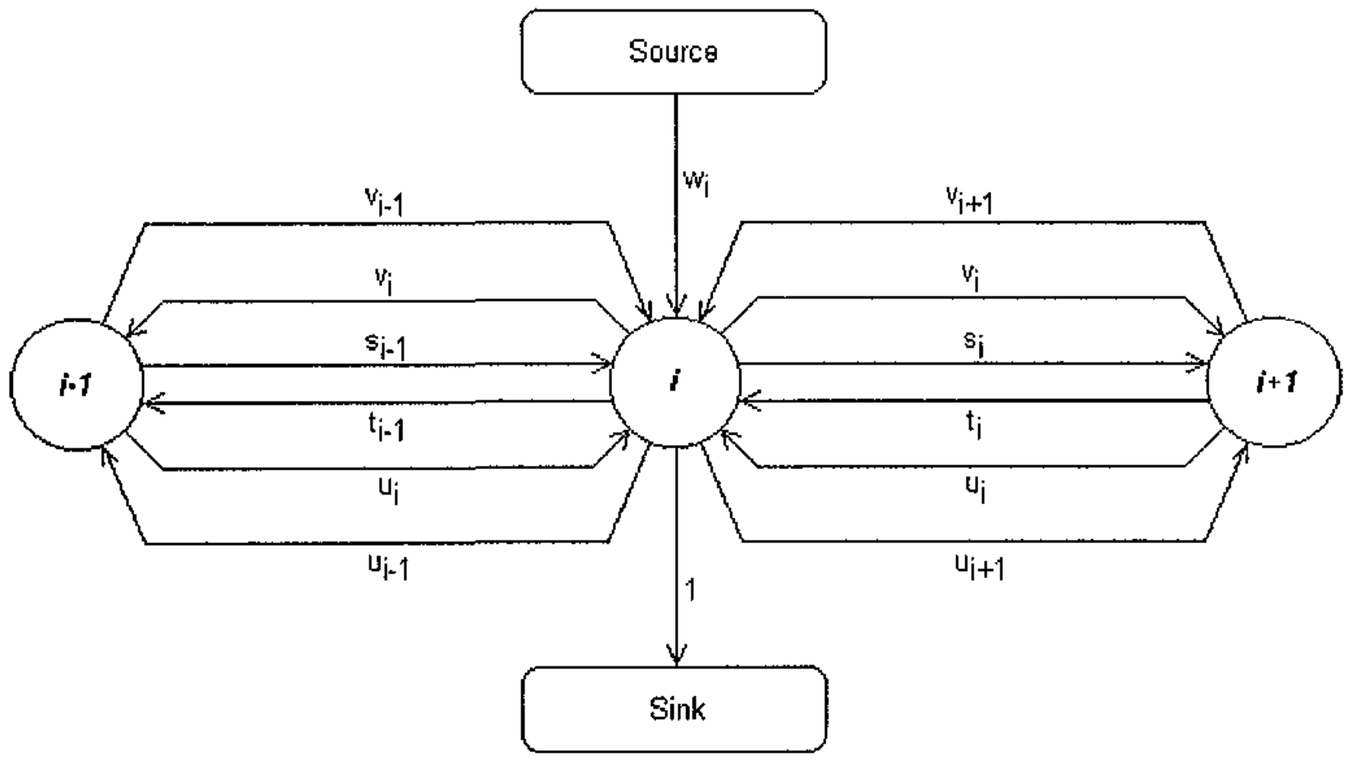


Figure 4.1. Network representation of the dual problem

In the diagram above, only the flows for node i are shown. Flow w_i comes from a limitless source, while the 1 in the constraint is interpreted as an obligatory flow from node i into the sink. Flows s_{i-1} and s_i are straightforward flows between two nodes and going to the right, while t_{i-1} and t_i are similar, but going to the left. The u and v flows are more unusual: observe that there are two arcs marked u_i which start at the neighbours of i and end at i. Although there are two separate flows here, the network analogy requires them to be equal. The same, of course, applies to the v_i arcs, except that these move outwards from node i to its neighbours.

From the rules of complementary slackness, therefore, any feasible solution can be represented as a feasible flow in the corresponding network. Specifically:

- Because there are exactly as many basic dual variables as there are points in the line, there will be at most as many nonzero flows in the analogous network (where the two arcs making up one u_i or v_i flow constitute one flow, of course).

- The existence of a nonnegative flow in s_i requires that the corresponding flow in t_i be zero, and *vice versa*. For a particular i either or both may be zero, but both may not be nonzero.

- The same applies to u_i and v_i for a particular i. Either or both may be zero, but both may not be nonzero.

- A flow balance across the entire network requires the sum of all w flows to be equal to n. Total inflow equals total outflow, and there is a flow of one unit out of every node.

4.5 The Dual Method for the One-dimensional Drape Problem

4.5.1 Description of the algorithm

The One-dimensional Dual Algorithm

Step 0 (*Initial dual solution*): Begin with the primal solution for which $z_i = d_i \quad \forall i$. In other words, the initial solution is a point-for-point match of the topography. The corresponding dual solution has $w_i = 1 \quad \forall i$, and all s_i, t_i, u_i and $v_i = 0$. Clearly, this is a dual feasible solution.

Step 1 (*Finding the entering variable*): Identify a primal constraint which is not satisfied. This might be a z-value which is too low, a slope between two adjacent points which exceeds α, a second derivative at a peak which is greater than β_u, or a second derivative in a trough which exceeds β_l. If all primal constraints are satisfied, the problem has been solved and optimality has been reached; therefore stop. Otherwise, identify the dual variable which corresponds to the first violated primal constraint encountered. The identified dual variable will not be basic (else the primal constraint would not have been violated). It therefore currently has the value zero. This variable must be made to enter the basis.

Step 2 (*Finding the leaving variable*): Identify the variable which must leave the basis. This will be the first variable to reach the value zero as the value of the entering variable increases. In the case of ties, choose any leaving variable.

Step 3 (*Recalculating the flows*): Recalculate the flows in the flow diagram, with the newly entered variable now having a (probably) positive value, while the leaving variable is obviously set to zero.

Step 4 (*Recalculating the drape surface*): Using the dual basic variables in the flow diagram, recalculate the z-values of the drape surface. Go back to step 1 to identify another offending primal constraint. This is done by the subalgorithm shown in Section 4.5.3.■

The above corresponds to a very standard solution method for the problem in question. However, the special structure of the problem (links are only between adjoining nodes) can be exploited to reduce the complexity of the problem considerably, as will be demonstrated below.

Before the special structure is discussed, however, a few words must be said about the identification of the leaving variable (step 2 above), and the recalculation of the drape surface given a particular set of dual basic variables (step 4 above).

4.5.2 Determining the leaving variable

This paragraph gives a more detailed description of step 2 of the One-dimensional Dual Algorithm.

If a particular variable is allowed to enter the basis of the dual problem, the solution of the problem will proceed along some edge of the n-dimensional polyhedron defined by the constraints until another vertex is reached. At this point we have another basic feasible solution, and one of the basic variables will have decreased to zero and can therefore be removed from the basis, leaving n basic variables as before. In general LP terms the determination of the leaving variable requires inverting an $n \times n$ matrix – a calculation with polynomial complexity.

Fortunately, the network representation makes this task a lot simpler. The entering variable is drawn as an arc (or as a pair of arcs in the case of an entering u or v) and given an unknown flow x_1. We consider the flow balance around each of the node(s) which are starting or ending points for the entering arc. In this way concomitant changes induced in other arcs which impinge on these nodes can be expressed in terms of x_1. During the initial stages of the solution (where most of the dual basic variables are w's) an entering arc will only affect the flow in a small number of neighbouring arcs. This allows for rapid progress at the start of the calculation. In the event where too many arcs impinge on a node, extra variables x_2, x_3 etc may be added to represent the new flows in arcs which are affected by the entering variable. At some stage these extra added variables can be eliminated and expressed in terms of x_1.

The method is best illustrated with an example. Consider the following array of five nodes with d-values as indicated:

Node no. (i)	1	2	3	4	5
d_i	8	11	7	6	10

In this example the values of $\alpha = 2$ and $\beta_l = \beta_u = 1$ are used. The initial primal solution has $z_i = d_i$ at all points. The corresponding initial network solution is shown in Figure 4.2.

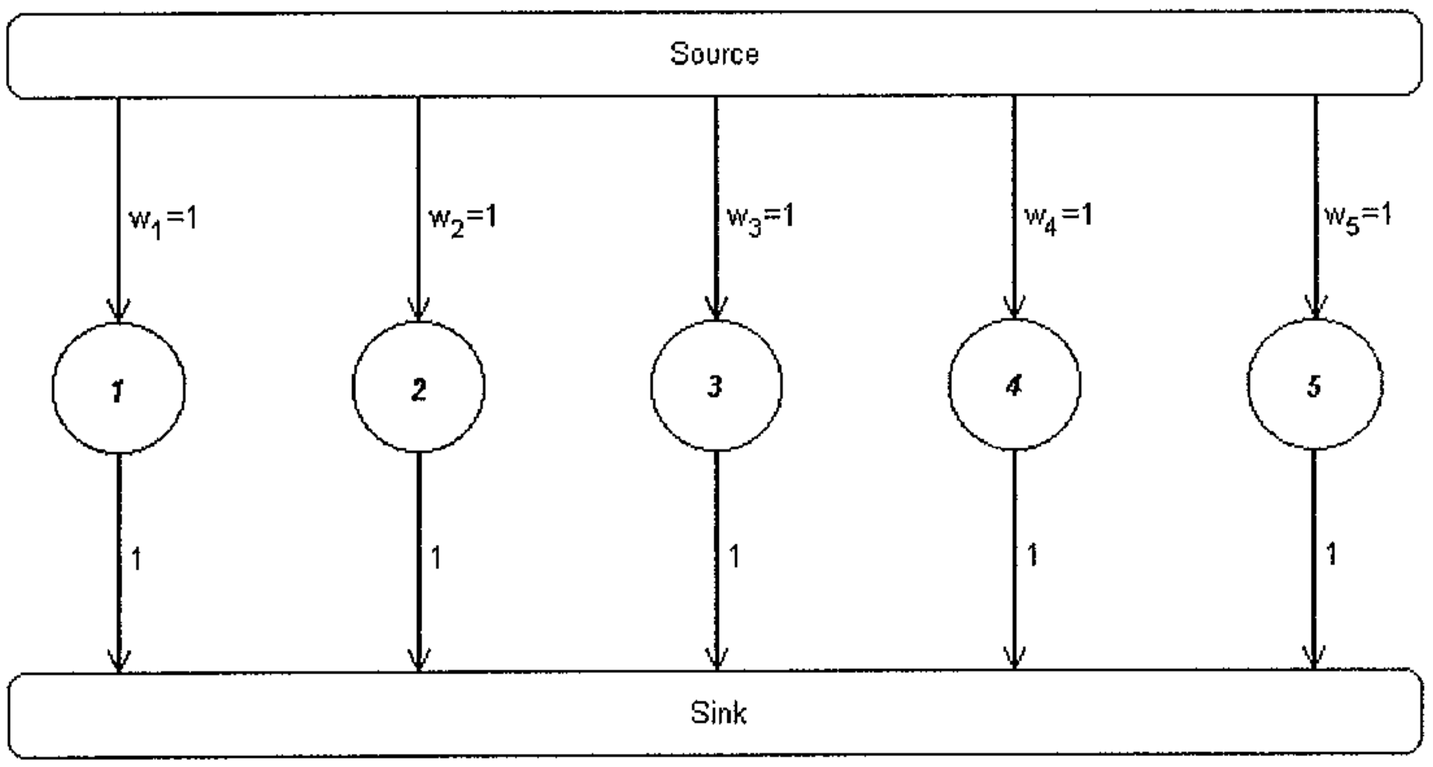


Figure 4.2. Initial flow in the simple example

The first and second derivatives at the nodes are:

Node no. (i)	1	2	3	4	5
$d_i\ (= z_i)$	8	11	7	6	10
1st derivative ($\frac{\partial z}{\partial x} \approx z_{i+1} - z_i$)	3	-4	-1	4	
2nd derivative ($\frac{\partial^2 z}{\partial x^2} \approx z_{i-1} - 2z_i + z_{i+1}$)		-7	3	5	

Note that almost all the derivatives violate the constraints. Only the first derivative between nodes 3 and 4 is acceptable – the quantity -1 in the table. As entering variable the dual variable for any overstepped constraint may be chosen. For this example, choose the second derivative for node 2 (whose absolute value is the largest). The corresponding dual variable is v_2.

If the additional arcs for v_2 are added to the network, flow balances must be adjusted as shown in the diagram below. Assume that the flow in the (new) v_2 arc is x_1. To maintain the flow balance through node 1, the flow in arc w_1 must change from 1 to $1-x_1$. Likewise the flow in arc w_3 must change from 1 to $1-x_1$. For the flow balance around node 2, the flow in w_2 must change from 1 to $1+2x_1$. These changes are illustrated in Figure 4.3.

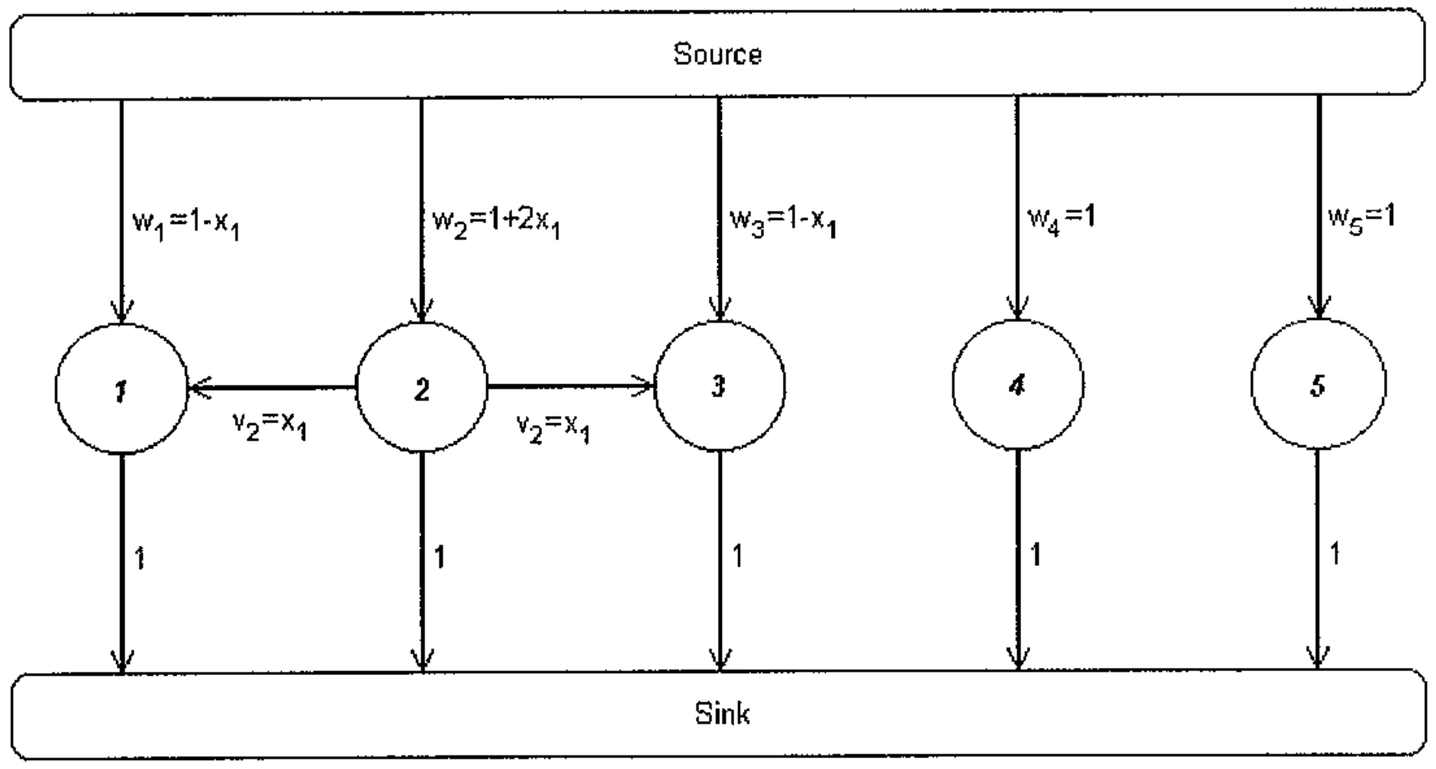


Figure 4.3. Flow changes during the first iteration

The leaving dual variable must now be determined. It will be the first variable to reach zero as the value of x_1 increases. Clearly, w_2 will increase with increasing x_1, and therefore need not be considered. However, both w_1 and w_3 will decrease and reach zero

when $x_1 = 1$. Since there is a tie, one of the two candidates must be chosen to leave the basis; arbitrarily choose w_1. The new flow in the dual problem is shown in Figure 4.4.

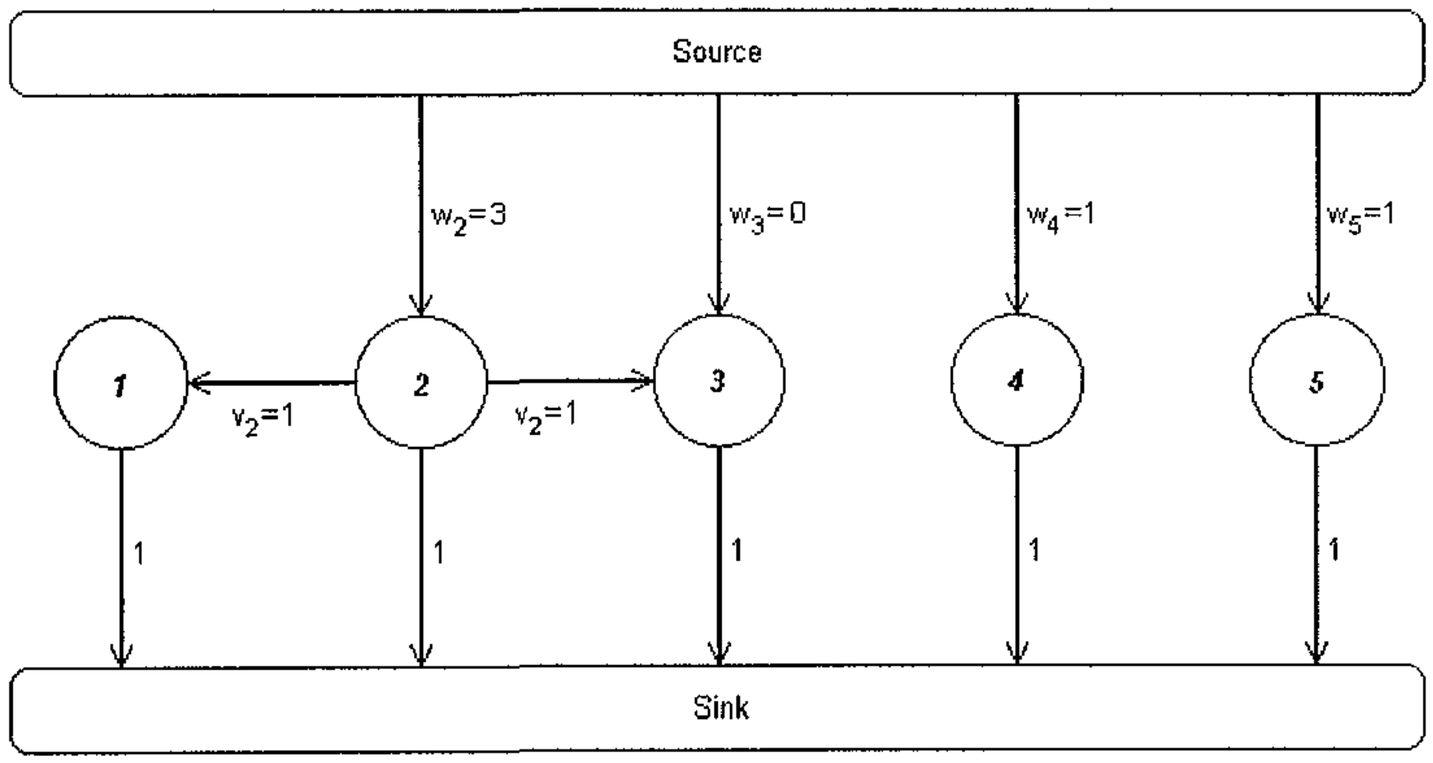


Figure 4.4. The network flow after the first iteration

Note that the operation of determining the leaving variable only involved nodes 1 to 3. The flows beyond the sub-network under consideration (ie involving nodes 4 and 5) are unaffected.

4.5.3 Recalculation of the drape surface

This section details step 4 of the One-dimensional Dual Algorithm.

It is now necessary to recalculate the drape surface so that further constraint violations can be determined in the next iteration.

As noted above, there are exactly n basic dual variables in a system of n nodes. The set of basic dual variables identifies the active constraints in the primal problem, and these can in turn be used to determine the appropriate z-values of the current solution. The method employed is as follows:

Step 4.0: Set all z-values as unknown.

Step 4.1: For all nodes i for which w_i is basic, set $z_i = d_i$.

Step 4.2: If all z-values are known, stop.

Step 4.3: For all basic s_i and t_i for which either z_i or z_{i+1} is known, calculate the unknown of the two z-values by using the difference α.

Step 4.4: For all basic u_i and v_i for which two of the three values z_{i-1}, z_i and z_{i+1} are known, calculate the unknown z-value by using the parameter β_l or β_u.

Step 4.5: Continue iterating through steps 4.2 to 4.4 until no further progress can be made.

Step 4.6: Assign the z-value x_k to a node with unknown z. The subscript k is simply a counter and has no particular significance. Set this z-value as known.

Step 4.7: Continue as in steps 4.2 to 4.4, with the following additional checks:

- If two nodes connected by an s or t in the basis are found, both the corresponding z's are known and one or both of these z-values are expressed in terms of x's, eliminate one of the x-variables and substitute its value throughout the entire system.

- If three nodes connected by a basic u or v are found, all three corresponding z's are known and one or more of these z-values are expressed in terms of x's, eliminate one of the x-variables and substitute its value throughout the entire system.

Go back to step 4.2.■

An example or two may serve to illustrate the procedure above.

The first example shows the regeneration of z-values from the basic dual variables at the end of the first iteration of the 5-point problem above. In this particular example the five basic dual variables are v_2, w_2, w_3, w_4 and w_5.

The w's are used to set the z-values for nodes 2 to 5 to their corresponding d-values, ie 11, 7, 6 and 10. For v_2 we note that both z_2 and z_3 are known while z_1 is not. The active constraint, however, requires that

$$-z_1 + 2z_2 - z_3 = \beta_u$$

or

$$-z_1 + 2\times 11 - 7 = 1$$

which readily gives the missing value: $z_1 = 14$. All z-values are now known.

For the second example consider the following network in which the basic variables are w_1, w_5, u_2, u_3 and u_4.

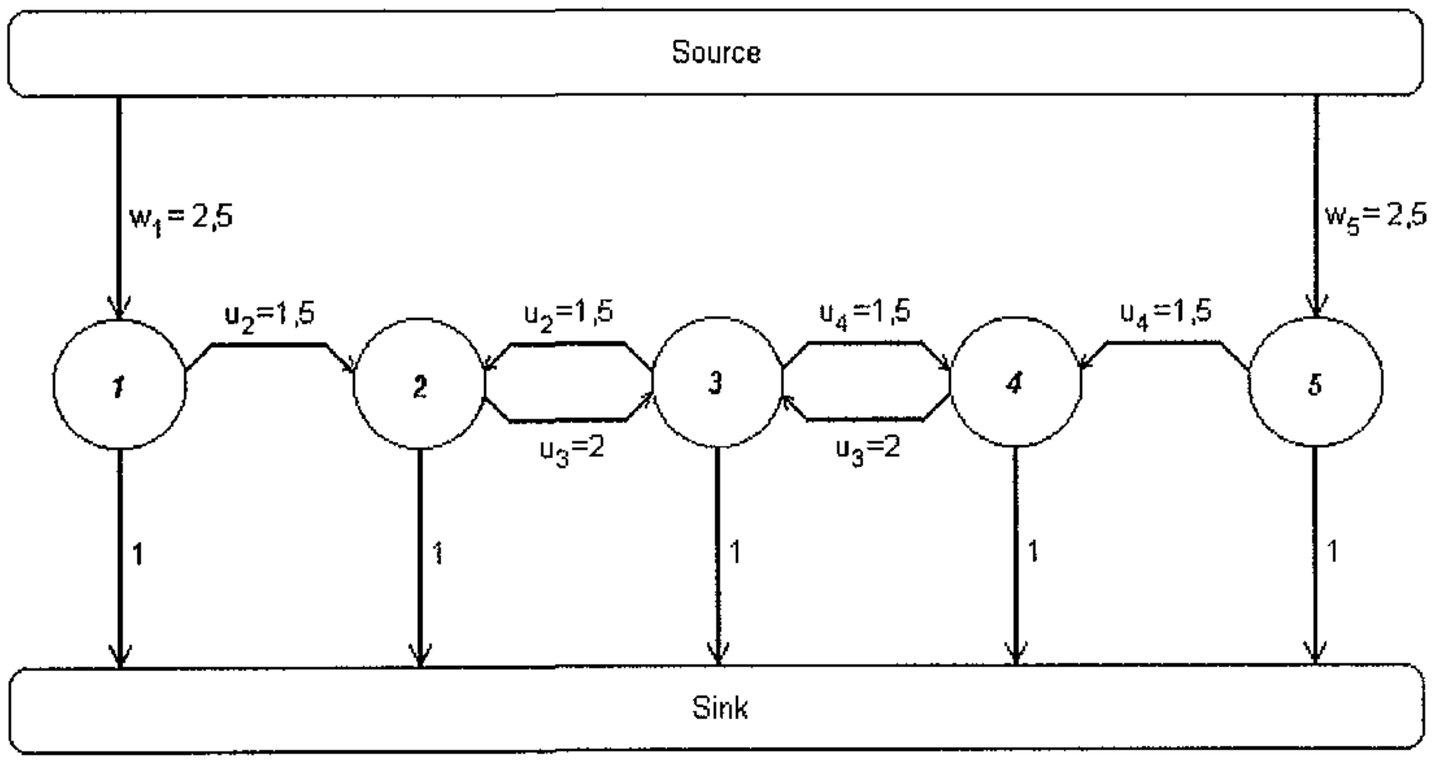


Figure 4.5. Flow diagram for the second example

The drape heights z_1 and z_5 are readily determined since w_1 and w_5 are basic. However, the other z-values cannot be found directly by using the first and second derivative relations because there is no u-constraint for which two z's are known. Hence assume that z_2 is known and that its value is x_1. Now we obtain the following values:

$$z_2 = x_1$$

$$z_3 = \beta_l - z_1 + 2z_2 = \beta_l - z_1 + 2x_1$$

$$z_4 = \beta_l - z_2 + 2z_3 = 3\beta_l - 2z_1 + 3x_1$$

$$z_5 = \beta_l - z_3 + 2z_4 = 6\beta_l - 3z_1 + 4x_1$$

But z_5 is already known. Using this, the value of x_1 can readily be determined from the last equation and back-substituted into all the previous equations. In this fashion all the z's are determined.

4.6 Exploiting the Problem's Structure

It could be argued that the method as described above is none other than a standard simplex algorithm in disguise. Essentially this is true, since the method iterates in the conventional fashion: identification of an entering variable, finding a suitable leaving variable and adjustment of the current solution. Fortunately, the calculation effort required for the algorithm described in this chapter is considerably less.

In the normal dual simplex method the recalculation of the primal solution, as well as the determination of the leaving variable would require manipulating an $n \times n$ matrix. For instance, in the second example above, the following system of equations needs to be solved for $\mathbf{z}$:

$$\mathbf{Az} = \mathbf{b}$$

where

$$\mathbf{A} = \begin{pmatrix} 1 & & & & \\ 1 & -2 & 1 & & \\ & 1 & -2 & 1 & \\ & & 1 & -2 & 1 \\ & & & & 1 \end{pmatrix}$$

$$\mathbf{z} = (z_1, \quad z_2, \quad z_3, \quad z_4, \quad z_5)^t$$

and

$$\mathbf{b} = (d_1, \quad \beta, \quad \beta, \quad \beta, \quad d_5)^t .$$

The One-dimensional Dual Algorithm offers some improvements. In the first place all w-constraints result in simple equalities and the corresponding z-values are thus determined trivially. (There is always at least one active w-constraint. If this were not so, the drape surface would no longer touch the topographic surface and could therefore be lowered.

This would indicate a dual infeasible solution which is precluded by the method which maintains dual feasibility throughout.)

Secondly, it seems that the number of extra variables required here is limited. This is indicated by numerical experience while there is also a (rather loose) mathematical argument for this which is not shown here.

Similar arguments (and numerical experience) suggest that the determination of the leaving variable requires no more than three additional variables (ie the unknown flow represented by the entering variable and the two others).

In practice it was found that the two processes in each iteration which would normally require the inversion of an $n \times n$ matrix now only require the use of an $n \times 3$ matrix. This provides for considerable memory savings for larger problems as well as simplifying the inversion process.

4.7 Comparison of Method to Standard Procedure

In order to establish the efficiency of the method described above, it was coded and its execution time compared to that of commercially available software using the standard simplex method for solving linear programming problems. The commercial software library selected was XA by Sunset Software Technology, San Marino, CA, USA; this suite has remarkable abilities for handling sparse matrices and displays an unprecedented speed in solving LP problems.

The table below lists the wall clock times taken by the One-dimensional Dual Algorithm and XA respectively for a variety of problems.

Size of problem (points)	XA - execution time (s)	One-dimensional Dual Algorithm - execution time (s)
215	0	2
600	10	6
800	17	11
1000	29	17
1200	38	24

The table clearly shows that the proposed method fares quite well against an accepted industry standard. All calculations were done on a 100 MHz Pentium with 64 MB of RAM.

The real benefit of the dual algorithm for the one-dimensional problem, however, lies in its sparing use of memory. The following table lists the approximate number of kilobytes of memory allocated for solving the problem using the two methods above.

Size of problem (points)	XA - memory used (kB)	One-dimensional Dual Algorithm - memory used (kB)
215	93	50
600	386	141
800	574	188
1000	830	234
1200	1150	281

The method's benefit becomes apparent here. Not only does it use less memory than the XA package, its use of memory is a linear function of the number of points in the problem while XA's memory usage increases polynomially.

4.8 Increasing the Speed of the One-dimensional Dual Algorithm

An effective way of reducing the number of iterations used by the dual method for the one-dimensional problem (and hence reducing the time required to solve a particular problem) was to change the initial solution to something which more closely approximated the ultimate drape surface. The inherent problem with setting up such an initial solution for the network method is to ensure that it is dual feasible. Various methods were investigated for constructing "almost feasible" initial surfaces; however, finding the corresponding dual feasible solution in these cases proved a difficult task. Indeed, it was often found that the dual solution, in the few cases in which one could be set up, was itself infeasible and thus not usable.

Fortunately it is possible to use the optimal solution with only first derivatives as the initial solution. This is the solution generated by the Lifting Algorithm or its improved version, the Enhanced Lifting Algorithm. As was shown in the previous chapter, this solution has a feasible dual (and is optimal if only constraints for the first derivative are considered). Furthermore, this solution can be set up in a time which is linearly dependent on the size of the problem.

The initial solution for the One-dimensional Dual Algorithm is set up as follows:

Step 1 (*Initial lifting*): Apply the Lifting Algorithm (or Enhanced Lifting Algorithm) with the appropriate value of α to the DTM grid. The output from this step will generate a drape surface which satisfies all the height and slope constraints.

The dual solution at this stage consists of at least one basic w-variable while the remaining basic variables are w's, s's or t's. However, the basic variables are not explicitly generated by the lifting method. Therefore they need to be regenerated.

Step 2 (*Initialisation for regeneration of basic dual variables*): Each of the n points of the starting drape surface can be in one of two states – *connected* or *unconnected*. Set all points *unconnected*.

Step 3 (*Finding the w-flows*): Examine all points in the drape surface. For all points for which $z_i = d_i$ change the state of point i to *connected*.

Step 4 (*Finding an appropriate point for connection*): Examine all *unconnected* points in the drape surface. If there are no *unconnected* points in the surface, the initial solution for the dual method for the one-dimensional problem has been found. Otherwise find an *unconnected* point i such that either $z_{i-1} = z_i + \alpha$ and $i > 1$ and point $i-1$ is *connected*, or $z_{i+1} = z_i + \alpha$ and $i < n$ and point $i+1$ is *connected*. In other words, find an *unconnected* point which lies α lower than an immediate neighbour which is *connected*.

Step 5 (*Connecting the new point*): Set the state of point i to *connected*. Go back to step 4.■

After the application of this method, the initial dual solution for the problem with n points has n basic dual variables: at least one of these is a *w*-variable, the rest can be w's, s's or t's. The dual network consists of one or more "trees", each fed by a *w*-flow from the source. Nodes which are not connected directly to the source with a *w*-arc are connected to adjoining nodes by s or *t*-arcs.

The following table compares the number of iterations taken by the One-dimensional Dual Algorithm using the topographic surface and the surface from the first derivative drape operation as starting solution respectively:

Size of problem (points)	Iterations starting with topographic surface	Iterations starting with first derivative drape surface
215	256	75
600	960	61
800	1265	92
1000	1591	109
1200	1913	125

For both starting solutions the number of iterations required is a more or less linear function of the problem size; however, beginning with a starting surface which is feasible insofar as the first derivative is concerned, reduces the number of iterations by a factor of about 15.

Chapter 5
The Two-dimensional Drape Problem – Exact Solution

5.1 General

Given the success of the network method for the one-dimensional problem as described in the previous chapter (frugal memory usage, excellent solution speeds, simplicity) the obvious next step is then to generalise the method for the general (two-dimensional) case. Unfortunately, this generalisation was found to be less successful than had been hoped for.

This chapter describes the generalisation of the Dual method for the one-dimensional problem to the two-dimensional problem. It also investigates the ease with which such a problem can be solved, and why the convenient reduction of the order of complexity which was possible in the one-dimensional case cannot be applied here with the same degree of success.

5.2 Mathematical Statement

The mathematical statement of the two-dimensional drape problem with first and second derivative constraints was given previously (section 2.5) but is repeated here for convenience:

Given an array of $m \times n$ equally-spaced points, each with a topographical elevation $d_{i,j}$ ($i = 1,...,m$ and $j = 1,...,n$), we wish to find drape heights $z_{i,j}$ at these points such that

$$z_{i,j} - z_{i,j+1} \le \alpha \qquad i = 1,...,m \qquad j = 1,...,n-1 \quad \text{(downhill W-E 1st derivative)}$$

$$z_{i,j+1} - z_{i,j} \le \alpha \qquad i = 1,...,m \qquad j = 1,...,n-1 \quad \text{(uphill W-E 1st derivative)}$$

$$z_{i,j} - z_{i+1,j} \le \alpha \qquad i = 1,...,m-1 \quad j = 1,...,n \qquad \text{(downhill N-S 1st derivative)}$$

$$z_{i+1,j} - z_{i,j} \le \alpha \qquad i = 1,...,m-1 \quad j = 1,...,n \qquad \text{(uphill N-S 1st derivative)}$$

$$z_{i,j-1} - 2z_{i,j} + z_{i,j+1} \le \beta_l \qquad i = 1,...,m \quad j = 2,...,n-1 \qquad \text{(valley W-E 2nd derivative)}$$

$$-z_{i,j-1} + 2z_{i,j} - z_{i,j+1} \le \beta_u \qquad i = 1,...,m \quad j = 2,...,n-1 \qquad \text{(peak W-E 2nd derivative)}$$

$$z_{i-1,j} - 2z_{i,j} + z_{i+1,j} \le \beta_l \qquad i = 2,...,m-1 \quad j = 1,...,n \qquad \text{(valley N-S 2nd derivative)}$$

$$-z_{i-1,j} + 2z_{i,j} - z_{i+1,j} \le \beta_u \qquad i = 2,...,m-1 \quad j = 1,...,n \qquad \text{(peak N-S 2nd derivative)}$$

$$z_{i,j} \ge d_{i,j} \qquad i = 1,...,m \quad j = 1,...,n \qquad \text{(height constraints)}$$

and we wish to make the total height of the drape surface as small as possible, ie

$$\min \sum_{i,j} z_{i,j} \ .$$

The designations W-E and N-S indicate west-east and north-south respectively and indicate the direction in which a constraint applies, viz to points in adjacent columns or adjacent rows.

The LP can now be written in the following standard form with $\le$-constraints and a maximisation objective function:

$$\max \ (-\sum_{i,j} z_{i,j}) \qquad (5.1)$$

subject to

$$z_{i,j} - z_{i,j+1} \leq \alpha \qquad i = 1,...,m \qquad j = 1,...,n-1 \qquad (5.1a)$$

$$z_{i,j+1} - z_{i,j} \leq \alpha \qquad i = 1,...,m \qquad j = 1,...,n-1$$

$$z_{i,j} - z_{i+1,j} \leq \alpha \qquad i = 1,...,m-1 \quad j = 1,...,n$$

$$z_{i+1,j} - z_{i,j} \leq \alpha \qquad i = 1,...,m-1 \quad j = 1,...,n$$

$$z_{i,j-1} - 2z_{i,j} + z_{i,j+1} \leq \beta_l \qquad i = 1,...,m \qquad j = 2,...,n-1 \qquad (5.1b)$$

$$-z_{i,j-1} + 2z_{i,j} - z_{i,j+1} \leq \beta_u \qquad i = 1,...,m \qquad j = 2,...,n-1$$

$$z_{i-1,j} - 2z_{i,j} + z_{i+1,j} \leq \beta_l \qquad i = 2,...,m-1 \quad j = 1,...,n$$

$$-z_{i-1,j} + 2z_{i,j} - z_{i+1,j} \leq \beta_u \qquad i = 2,...,m-1 \quad j = 1,...,n$$

$$-z_{i,j} \leq -d_{i,j} \qquad i = 1,...,m \qquad j = 1,...,n \qquad (5.1c)$$

with

$$\text{all variables} \geq 0.$$

In the LP statement above, constraints (5.1a) are the slope constraints, (5.1b) the second derivative constraints and (5.1c) the height constraints.

As before, we note that if $\beta_l \geq 2\alpha$ or $\beta_u \geq 2\alpha$, the first derivative constraints imply that those for the second derivative are automatically satisfied. We therefore assume again that $\beta_l < 2\alpha$ and $\beta_u < 2\alpha$, unless otherwise stated.

5.3 Dual of the Two-dimensional Problem

As in the one-dimensional two-derivative case of the previous chapter, every constraint in the primal problem has a corresponding variable in the dual and *vice versa*. Considering the grid point indexed by (i, j), let $w_{i,j}$ be the dual variable corresponding to the height constraint $-z_{i,j} \leq -d_{i,j}$. The complete list of dual variables which are assigned in this fashion is then as shown in the following table.

Constraint	**Type**	**Dual variable**
$-z_{i,j} \leq -d_{i,j}$	Height	$w_{i,j}$
$z_{i,j} - z_{i,j+1} \leq \alpha$	Downhill W-E slope	$s_{i,j}$
$z_{i,j+1} - z_{i,j} \leq \alpha$	Uphill W-E slope	$t_{i,j}$
$z_{i,j} - z_{i+1,j} \leq \alpha$	Downhill N-S slope	$k_{i,j}$
$z_{i+1,j} - z_{i,j} \leq \alpha$	Uphill N-S slope	$l_{i,j}$
$z_{i,j-1} - 2z_{i,j} + z_{i,j+1} \leq \beta_l$	W-E valley	$u_{i,j}$
$-z_{i,j-1} + 2z_{i,j} - z_{i,j+1} \leq \beta_u$	W-E peak	$v_{i,j}$
$z_{i-1,j} - 2z_{i,j} + z_{i+1,j} \leq \beta_l$	N-S valley	$p_{i,j}$
$-z_{i-1,j} + 2z_{i,j} - z_{i+1,j} \leq \beta_u$	N-S peak	$q_{i,j}$

There is one dual variable associated with each slope constraint emanating from the current grid point (i, j), one with each second derivative constraint and one with the height constraint at this point.

The dual problem can now readily be formulated as

$$\text{Min } \alpha \sum_{i,j} (s_{i,j} + t_{i,j} + k_{i,j} + l_{i,j}) + \beta_l \sum_{i,j} (u_{i,j} + p_{i,j}) + \beta_u \sum_{i,j} (v_{i,j} + q_{i,j}) - \sum_{i,j} d_{i,j} w_{i,j}$$

subject to

$$
\begin{aligned}
& s_{i,j} - s_{i,j-1} - t_{i,j} + t_{i,j-1} + k_{i,j} - k_{i-1,j} - l_{i,j} + l_{i-1,j} \\
& + u_{i,j-1} - 2u_{i,j} + u_{i,j+1} - v_{i,j-1} + 2v_{i,j} - v_{i,j+1} \\
& + p_{i-1,j} - 2p_{i,j} + p_{i+1,j} - q_{i-1,j} + 2q_{i,j} - q_{i+1,j} - w_{i,j} \geq -1
\end{aligned}
$$

with

all variables ≥ 0.

To save the reader the tedium of the different bounds for the subscripts of the various terms, we assume that terms which relate to any point (i, j) outside the grid limits are set to zero. The same applies to slopes and second derivatives which involve such points. For example, at the most northwesterly point of the grid where $i = 1$ and $j = 1$, the constraint

$$
\begin{aligned}
& s_{1,1} - s_{1,0} - t_{1,1} + t_{1,0} + k_{1,1} - k_{0,1} - l_{1,1} + l_{0,1} \\
& + u_{1,0} - 2u_{1,1} + u_{1,2} - v_{1,0} + 2v_{1,1} - v_{1,2} \\
& + p_{0,1} - 2p_{1,1} + p_{2,1} - q_{0,1} + 2q_{1,1} - q_{2,1} - w_{1,1} \geq -1
\end{aligned}
$$

has $s_{1,0} = t_{1,0} = k_{0,1} = l_{0,1} = u_{1,0} = v_{1,0} = p_{0,1} = q_{0,1} = 0$. Also $u_{1,1}$, $v_{1,1}$, $p_{1,1}$ and $q_{1,1}$ are zero, because they each span three points of which the first does not exist. Hence the constraint above becomes

$$
s_{1,1} - t_{1,1} + k_{1,1} - l_{1,1} + u_{1,2} - v_{1,2} + p_{2,1} - q_{2,1} - w_{1,1} \geq -1 .
$$

There is exactly one constraint for every grid point.

As in the one-dimensional case, a dual method is used to solve the two-dimensional problem. This begins with all the points in the drape surface at the same height as the topological surface. Once again, this is a dual feasible solution because all the $w_{i,j}$ variables are set to one while all others are zero. In other words, all the height constraints in the equivalent primal problem are active. It is now easy to detect infeasibilities in the

primal – these would initially only be slope or second derivative constraints which had been breached, since all the height constraints are satisfied.

The solution method proceeds in the normal fashion of a simplex method: given n points in the grid, and thus n dual constraints, the solution at any stage comprises n basic dual variables. At each iteration a primal infeasibility is identified; the corresponding dual variable is chosen to enter the basis, while the leaving variable is the one in the basis which first disappears (reaches zero) as the entering variable grows in size. In analogy to the one-dimensional case, an equivalent network representation of the dual can also be envisaged.

5.4 Network Flow Analogy

Complementary slackness between the primal and dual problems requires that

$$s_{i,j}(z_{i,j} - z_{i,j+1} - \alpha) = 0 \qquad \text{(first derivative constraints)}$$

$$t_{i,j}(z_{i,j+1} - z_{i,j} - \alpha) = 0$$

$$k_{i,j}(z_{i,j} - z_{i+1,j} - \alpha) = 0$$

$$l_{i,j}(z_{i+1,j} - z_{i,j} - \alpha) = 0$$

$$u_{i,j}(z_{i,j-1} - 2z_{i,j} + z_{i,j+1} - \beta_l) = 0 \qquad \text{(second derivative constraints)}$$

$$v_{i,j}(-z_{i,j-1} + 2z_{i,j} - z_{i,j+1} - \beta_u) = 0$$

$$p_{i,j}(z_{i-1,j} - 2z_{i,j} + z_{i+1,j} - \beta_l) = 0$$

$$q_{i,j}(-z_{i-1,j} + 2z_{i,j} - z_{i+1,j} - \beta_u) = 0$$

$$w_{i,j}(z_{i,j} - d_{i,j}) = 0 \qquad \text{(height constraints)}$$

and

$$z_{i,j}(s_{i,j} - s_{i,j-1} - t_{i,j} + t_{i,j-1} + k_{i-1,j} - k_{i,j} - l_{i-1,j} + l_{i,j}$$
$$+ u_{i,j-1} - 2u_{i,j} + u_{i,j+1} - v_{i,j-1} + 2v_{i,j} - v_{i,j-1}$$
$$+ p_{i-1,j} - 2p_{i,j} + p_{i+1,j} - q_{i-1,j} + 2q_{i,j} - q_{i+1,j} - w_{i,j} + 1) = 0$$

for all applicable points.

Since z-values will, in general, not be zero and can, in fact, be set to be strictly positive by simply transforming the topographical surface to a new height, the last equation then requires that

$$s_{i,j} - s_{i,j-1} - t_{i,j} + t_{i,j-1} + k_{i-1,j} - k_{i,j} - l_{i-1,j} + l_{i,j}$$
$$+ u_{i,j-1} - 2u_{i,j} + u_{i,j+1} - v_{i,j-1} + 2v_{i,j} - v_{i,j-1}$$
$$+ p_{i-1,j} - 2p_{i,j} + p_{i+1,j} - q_{i-1,j} + 2q_{i,j} - q_{i+1,j} - w_{i,j} + 1 = 0$$

for all points. This means that all dual inequality constraints become strict equalities.

As before, each constraint can be seen as a flow balance around a point. A network node represents point (i, j) on the surface. Viewing the variables in the dual constraints as flows, the constraints can be represented as a network as shown in the sketch below.

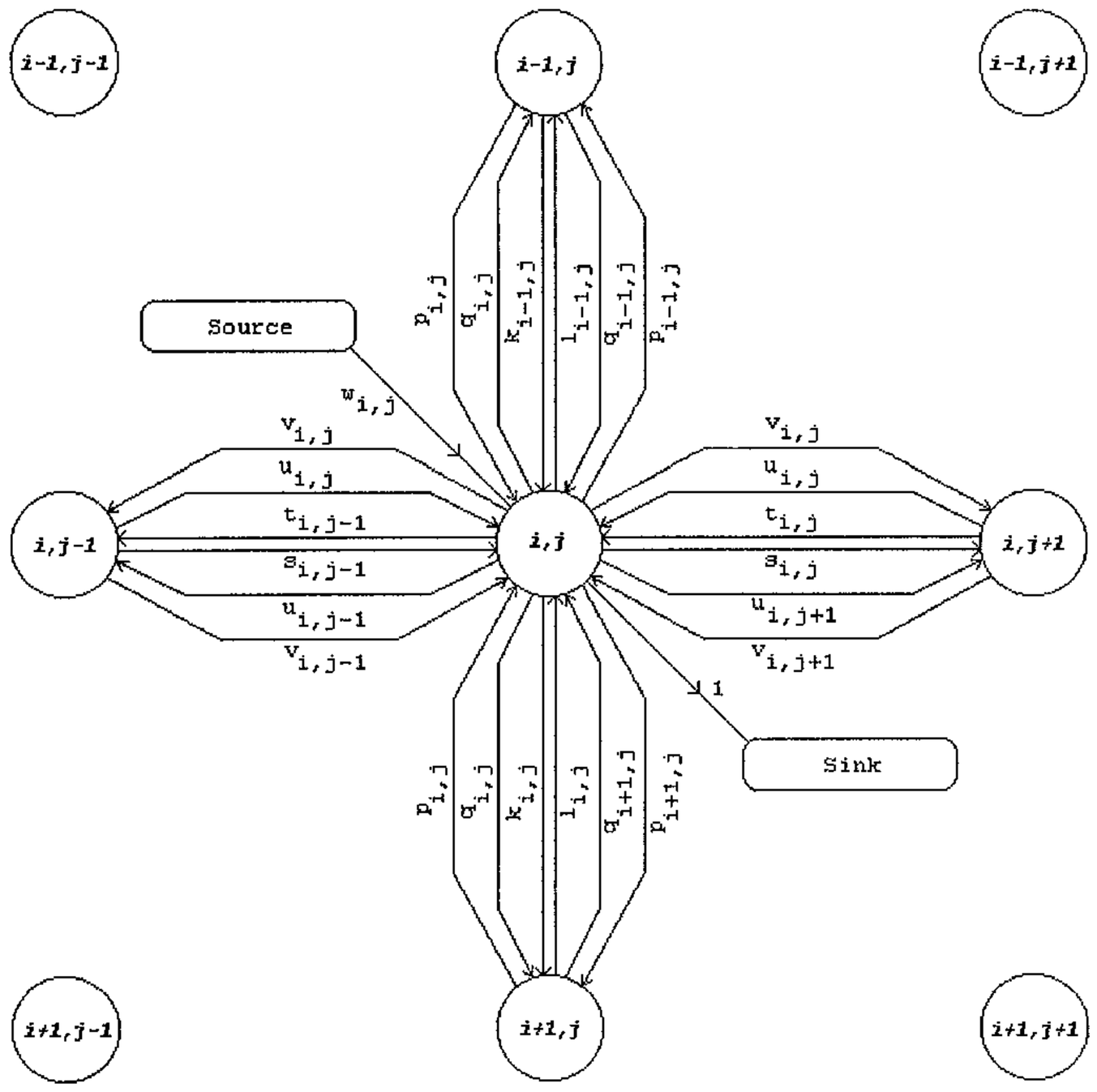


Figure 5.1. Network analogy for the two-dimensional case

The sketch only shows the flows relevant to node (i, j). Similar flows exist at all the other nodes as well. The source is connected to all the nodes in the network by w-arcs while a flow of one unit drops from each node to the communal sink. By analogy to the one-dimensional case each u, v, p and q-flow consists of two equal arcs which emanate from or converge on the central one of three nodes. u and v arcs run west-east, p and q arcs run north-south.

5.5 Solution Method for the Two-dimensional Problem

5.5.1 Description of the algorithm

The dual method for the two-dimensional drape problem

Step 1 (*Initial dual solution*): Begin with the primal solution for which $z_{i,j} = d_{i,j} \quad \forall(i,j)$. In other words, the initial solution is a point-for-point match of the topography. The corresponding dual solution has $w_{i,j} = 1 \quad \forall(i,j)$, and all $s_{i,j}$, $t_{i,j}$, $k_{i,j}$, $l_{i,j}$, $u_{i,j}$, $v_{i,j}$, $p_{i,j}$ and $q_{i,j} = 0$. This is recognisably a dual feasible solution.

Step 2 (*Finding the entering variable*): Identify a primal constraint which is not satisfied. This might be a z-value which is too low, a slope between two adjacent points which exceeds α, or a second derivative at a point which is greater than β_l or β_u. If no such primal constraint can be found, go to step 6. Otherwise identify the dual variable which corresponds to this violated primal constraint. The identified dual variable will not be basic (else the primal constraint would not have been violated). It therefore currently has the value zero. This variable must be made to enter the basis.

Step 3 (*Finding the leaving variable*): Identify the variable which must leave the basis. This will be the first variable to reach the value zero as the value of the entering variable increases. In the case of ties, choose one arbitrarily.

Step 4 (*Recalculating the flows*): Recalculate the flows in the flow diagram, with the newly entered variable now having a (probably) positive value, while the leaving variable is obviously zero.

Step 5 (*Recalculating the drape surface*): Using the dual basic variables in the flow diagram, recalculate the z-values of the drape surface. Go back to step 2 to identify another offending primal constraint.

Step 6 (*Termination*): The optimal drape surface has been generated. The objective function is taken to be the sum of the elevations of all the points of the drape surface. ■

As in the one-dimensional case, this is the standard dual simplex method in a thin disguise. It was hoped to extend the special structure of the one-dimensional case to the two-dimensional problem, ie being able to speedily identify the leaving variable and to rapidly reconstruct a new drape surface from a given dual solution. It was similarly hoped to gain the upper hand on the complexity by being able to show that the memory required was less than for a general linear programming problem.

Some explanation of the various steps of the algorithm is probably necessary. Where appropriate, further details are given in the following sections.

5.5.1 Finding the leaving variable

This paragraph gives a more detailed explanation of step 3 of the dual algorithm.

Before it is possible to determine which variable will leave the basis, it is useful (from a computational point of view) to flag all those dual variables which might be affected by the entering variable coming into the basis. This is done by associating two flags with every point in the grid as well as two flags with every dual variable. The two flags in each case are the *changing* flag and the *fathomed* flag. For points in the grid, the *changing* flag indicates that the drape height of the point may change during the current iteration. For a dual variable the *changing* flag indicates that the flow associated with the current dual variable is likely to change. For grid points the *fathomed* flag indicates that all the possible flows from the current point have been noted. For dual variables the *fathomed* flag shows that all endpoints of the current flow have been determined.

The general philosophy of flagging the points and dual variables in this way is to isolate the area in which flow changes will take place. The marking subalgorithm proceeds as described below.

Step 3.1 (*Initialisation*): Mark all points as *not changing* and *not fathomed*. Similarly, mark all dual variables (flows) as *not changing* and *not fathomed*.

Step 3.2 (*Mark incoming variable*): Mark the incoming dual variable *changing*.

Step 3.3 (*Propagate marked dual variables*): Find all dual variables which are marked *changing* and *not fathomed*. If there are no such variables go to step 3.5. For each of these variables mark as *changing* the endpoints of the flows which they represent. Thus

- For all $w_{i,j}$ variables, mark (i, j) as *changing*.

- For all $s_{i,j}$ and $t_{i,j}$ variables, mark (i, j) and $(i, j+1)$ as *changing*.

- For all $k_{i,j}$ and $l_{i,j}$ variables, mark (i, j) and $(i+1, j)$ as *changing*.

- For all $u_{i,j}$ and $v_{i,j}$ variables, mark $(i, j-1)$, (i, j) and $(i, j+1)$ as *changing*.

- For all $p_{i,j}$ and $q_{i,j}$ variables, mark $(i-1, j)$, (i, j) and $(i+1, j)$ as *changing*.

Then mark all the duals *fathomed* which were treated in this way.

Step 3.4 (*Propagate marked primal variables*): Examine all dual variables in the basis. Mark as *changing* all those for which a flow source or sink point is marked *changing*. In other words

- If $w_{i,j}$ is in the basis and (i, j) is marked as *changing*, mark $w_{i,j}$ as *changing*.

- If $s_{i,j}$ is in the basis and (i, j) or $(i, j+1)$ is marked as *changing*, mark $s_{i,j}$ as *changing*.

- If $t_{i,j}$ is in the basis and (i, j) or $(i, j+1)$ is marked as *changing*, mark $t_{i,j}$ as *changing*.

- If $k_{i,j}$ is in the basis and (i, j) or $(i+1, j)$ is marked as *changing*, mark $k_{i,j}$ as *changing*.

- If $l_{i,j}$ is in the basis and (i, j) or $(i+1, j)$ is marked as *changing*, mark $l_{i,j}$ as *changing*.

- If $u_{i,j}$ is in the basis and $(i, j-1)$, (i, j) or $(i, j+1)$ is marked as *changing*, mark $u_{i,j}$ as *changing*.

- If $v_{i,j}$ is in the basis and $(i, j-1)$, (i, j) or $(i, j+1)$ is marked as *changing*, mark $v_{i,j}$ as *changing*.

- If $p_{i,j}$ is in the basis and $(i-1, j)$, (i, j) or $(i+1, j)$ is marked as *changing*, mark $p_{i,j}$ as *changing*.

- If $q_{i,j}$ is in the basis and $(i-1, j)$, (i, j) or $(i+1, j)$ is marked as *changing*, mark $q_{i,j}$ as *changing*.

Go back to step 3.3.

Step 3.5: The marking phase has been completed. The algorithm can now move on to finding the leaving variable.■

For finding the leaving variable a flow of θ is postulated for the entering flow in the dual basis. All points marked *changing* as well as all dual flows marked in this way are considered, and resultant flows are determined in all *changing* arcs in terms of θ. As a result all *changing* arc flows are now expressed in the form

$$f_{i,j} = f_{i,j}^0 + g_{i,j}\theta$$

where $f_{i,j}$ is the new value of one of the dual variables (ie one of $w_{i,j}$, $s_{i,j}$, $t_{i,j}$, $k_{i,j}$, $l_{i,j}$, $u_{i,j}$, $v_{i,j}$, $p_{i,j}$ or $q_{i,j}$) while $f_{i,j}^0$ is its current (old) value and $g_{i,j}$ is a coefficient whose sign can be negative, zero or positive. To find the leaving variable, only those dual variables $f_{i,j}$ need be examined which have a negative $g_{i,j}$. As leaving variable that $f_{i,j}$ is chosen for which $g_{i,j} < 0$ and $\left|\frac{f_{i,j}^0}{g_{i,j}}\right|$ is minimised.

5.5.2 Recalculating the flows

This paragraph gives a more detailed explanation of step 4 of the dual algorithm.

Recalculating the flows is quite straightforward using the intermediate results calculated in the last step.

- The leaving variable is given the value zero and is removed from the basis.
- The entering variable takes on the minimum value of θ determined in the previous step.
- All other basic variables are updated using the relation

$$f_{i,j} = f_{i,j}^0 + g_{i,j}\theta \ .$$

5.5.3 Recalculating the drape surface

This paragraph gives a more detailed explanation of step 5 of the dual algorithm.

The input to this step of the method is the set of basic dual variables; each of these refers to an "active" flow in the network analogy. This information must be used to determine the heights of the points in the drape surface. The algorithm as described below is used.

Step 5.1 (*Initialise flags*): Associate a flag with each point in the drape surface. This flag is set to indicate that the drape height of a point has been calculated, else it is reset. Reset the flags of all points which have the *changing* flag set. We associate a number of ancillary variables, x_i, with each point. In the beginning all these variables are set to zero and none of them is in use.

Step 5.2 (*Handle w-flows*): For all $w_{i,j}$ in the basis, set $z_{i,j} = d_{i,j}$ and set the flag for point (i, j). Reset the *changing* flag for $w_{i,j}$. These are the points which lie on the topography.

Step 5.3 (*Resolve dual variables for two points*): Examine all dual variables which describe flows between two points ($s_{i,j}$, $t_{i,j}$, $k_{i,j}$ and $l_{i,j}$) for which the *changing* flag is set.

- For those which have the point flag set for one of the endpoints (but not the other), calculate the height of the unknown endpoint using the difference in height of α from the known point. Set the point flag for the other endpoint. Reset the *changing* flag for this dual variable.

- For those which have both endpoint flags set, eliminate one of the ancillary x-variables and substitute this value for all occurrences of the applicable x throughout the system. Reset the *changing* flag for this dual variable.

- For those which have neither endpoint flag set, do nothing.

Step 5.4 (*Resolve dual variables for three points*): Examine all dual variables which describe flows affecting three points ($u_{i,j}$, $v_{i,j}$, $p_{i,j}$ and $q_{i,j}$) for which the *changing* flag is set.

- For those which have the point flag set for two of the points affected by the dual variable, calculate the height of the third point using the value of β_u or β_l. Set the point flag for the third point. Reset the *changing* flag for this dual variable.

- For those which have the point flag set for all three affected points, eliminate one of the ancillary x-variables and substitute this value for all occurrences of the applicable x throughout the system. Reset the *changing* flag for this dual variable.

- For those which have the point flag set for one of the points affected by the dual variable, assign an (as yet) unknown height of x_k to one of the points affected by the dual variable, where the subscript k is simply a counter. Calculate the height of the third point affected in terms of this unknown value. Set the point flags for both of the unknown points. Reset the *changing* flag for the dual variable.

- For those which have no point flags set for the affected drape points, do nothing.

Step 5.5 (*Check for termination*): If there are still dual variables with the *changing* flag set or points whose point flag is not set, go back to step 5.3. Else all heights in the drape surface have been determined.■

5.6 Numerical Results

As is often found, the translation of a relatively straightforward algorithm from human-intelligible form to a form in which the necessary manipulations can be carried out by machine brought with it more than the usual number of programming difficulties. Some of these will be mentioned shortly:

- The problem of rounding is ever present in many programs, and the implementation of the two-dimensional dual algorithm is no exception. Without due regard to rounding, many iterations (often an infinite number) can be spent uselessly trying to bring into feasibility a constraint which is essentially feasible already, but where the machine's resolution is simply unable to represent the required quantity accurately. This problem is most easily circumvented by decreasing the values of α, β_l and β_u by a small amount when adjusting z-values. A value adjusted by 0,001 % was found to work well. In this way a "corrected" and feasible constraint will not become marginally infeasible on a successive iteration.

- All two-dimensional arrays, eg those representing the structures of information relating to the grid points, were linearised. This technique (which is implemented by many compilers anyway) has the advantage that the array can be scanned more quickly than if double indices were used (eg to find a flag set for one of the points).

- In a problem containing n grid points, the algorithm calls for n dual variables as well. Hence n structures to describe these dual variables are also required. However, any one iteration always requires manipulating one extra variable – the dual variable which enters the basis. This extra structure was effectively accommodated in the $(n+1)$ st position during an iteration, then transferred to the slot in the array of dual variable structures at the position just vacated by the leaving variable at the end of the iteration.

- The algorithm as described requires both primal and dual representations of the problem. The primal representation associates a structure with each grid point which contains not only the original and the current drape heights but also a list of possible arcs of which the current point forms an end point. Likewise each structure associated with a dual variable contains not only the type of arc and flow that the dual variable represents but also the grid point(s) which it applies to. It goes without saying that the housekeeping of these structures causes considerable headaches. Extreme care is required when updating such a system.

The solution method was coded successfully after overcoming the problems mentioned above, and the following results were obtained. Unfortunately, the execution times became quite prohibitive with increasing problem size, even for relatively small problems. All calculations were performed on a Pentium with 64 MB of RAM.

Size of problem	**XA**			**Dual Algorithm**		
	Iterations	**Memory (bytes)**	**Time (s)**	**Iterations**	**Memory (bytes)**	**Time (s)**
12 × 12	193	151 748	1	417	57916	2
30 × 30	215	951 680	1	293	619 984	2
50 × 50	313	2 702 180	4	349	2 521 304	3
60 × 60	3 647	3 965 516	80	8 926	4 206 364	17 439
64 × 64	7 209	4 690 496	218	11 305	5 047 940	141 151

The table above compares the performance of the network method described above (and programmed as described) with the commercial XA package mentioned previously. Although the network method may be slightly less memory-hungry for smaller problems, the execution time compares rather poorly with XA's performance, especially for the larger problems. It would appear that the time taken rises exponentially with the size of the problem. Therefore, generating a drape surface for a topographic grid of, say, even 100 × 100 cells would probably consume time in the range of months or years, and the method would certainly not be of any practical use for problems of this size and larger.

Two factors are responsible for the generally poor performance of the method for the two-dimensional case. They are

- the necessity of recalculating both the primal and dual representations at each iteration (the general method requires only the primal or dual matrix) and

- the apparent impossibility of exploiting the special structure of the problem to create a matrix with one very small dimension to represent the flow, as was done in the one-dimensional case.

If the second of these two limitations could be removed (as was done in the one-dimensional case) the unacceptable speed of the method would readily be brought into acceptable limits, even for far larger problems.

5.7 A Numerical Example

The following example has been included to indicate the complex paths which are eventually generated by the network method for even a simple problem of only 6 rows and 7 columns.

The original topographical grid had the values shown below.

10	12	11	16	12	10	6
11	9	6	8	4	7	7
16	15	13	9	10	6	5
10	8	15	9	7	11	13
12	13	16	13	10	8	7
9	7	8	10	12	5	8

The optimal solution was found to be the following, using $\alpha = 2$ and $\beta_l = \beta_u = 1$.

13	15	16	16	15	13	11
14,7	14,5	14,76	14,1	13,07	12,03	12
16	15	14,53	13,2	12,13	12,06	13
16,3	16,3	15,3	13,3	12,2	12,1	13
16,2	16,6	16	14,4	12,6	11,8	12
15,1	15,9	15,7	14,5	12,5	10,75	10

The equivalent dual network is as follows:

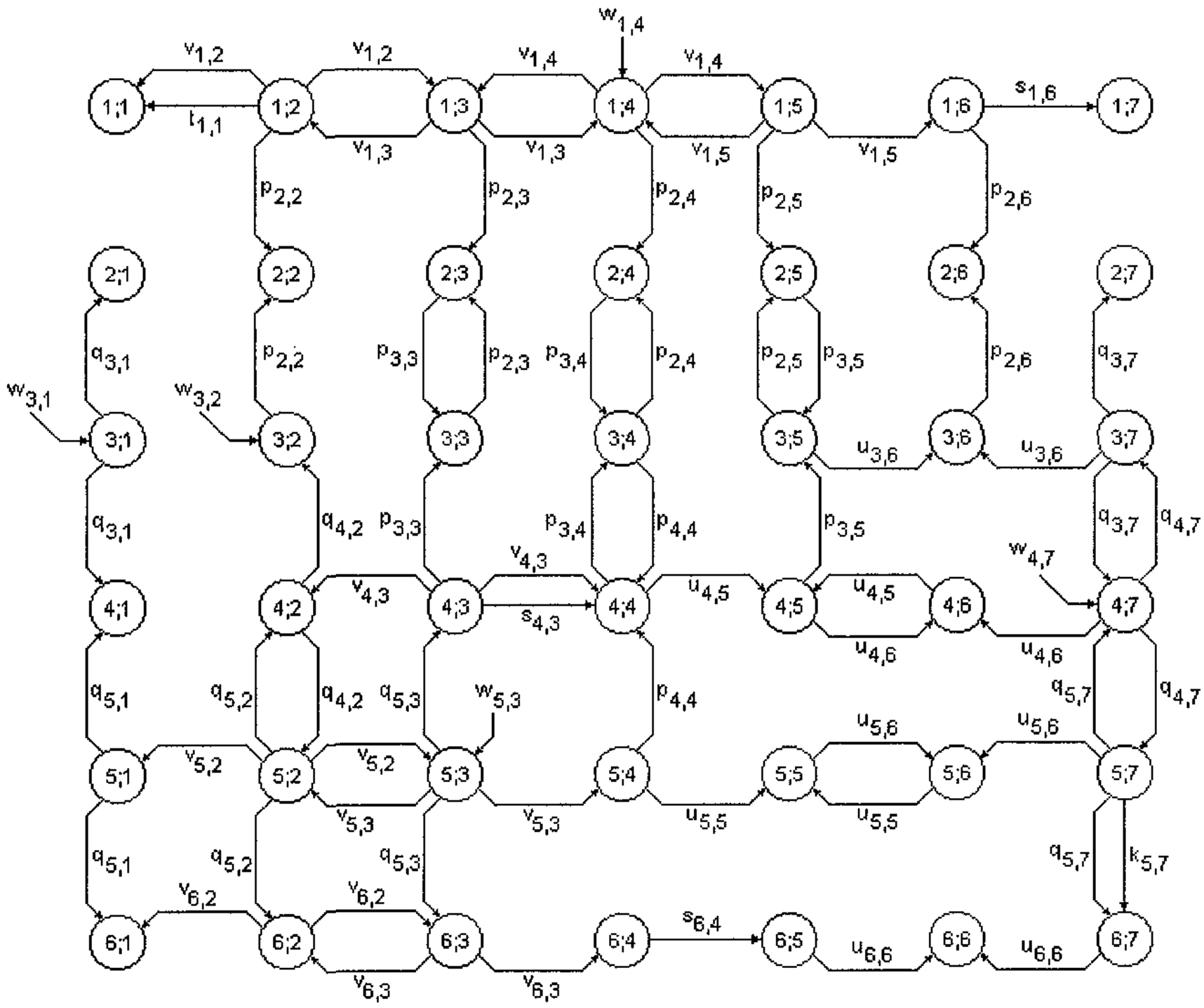


Figure 5.2. Flow representation of the optimal solution

5.8 Shortcomings of the Proposed Method

It was found in most examples, as in the one above too, that the optimal solution could rarely be broken into separate networks. Had this been the case, the problem would obviously have been much simpler. As it stands, though, the introduction of one extra arc into the basis requires changes to be effected in all existing arcs in the basis in order to determine the one which is to leave it. It is this calculation which is prohibitive since the number of arcs to be considered is exactly equal to the number of points in the grid.

In the table above comparing the execution times for a few (small) problems, the generally accepted technique applied by XA beat the proposed method most convincingly. The poor performance of the network method is deemed to be due to the two factors mentioned previously, the inability to sufficiently constrain the number of columns in the matrix representing the network flow, and the necessity of carrying along both primal and dual representations of the problem and using them respectively to find the leaving variable and the entering variable. Clearly, a method such as XA has the advantage here because it does not require switching between representations.

Chapter 6
The Two-dimensional Drape Problem – General Heuristic Methods

6.1 General

After the lack of success with the proposed network method based on the dual of the problem, and in view of the fact that it was deemed very unlikely that an exact method would be found which would solve the drape problem in a reasonable amount of time, it was decided to consider heuristic methods.

It has been indicated previously (see Chapter 2) that the drape surface has to be constructed in such a way that it is "flyable", regardless of where the flight (and tie) lines eventually come to land. Also, it is not possible to determine *a priori* the points at which measurements will be made, so that optimality in terms of measurement positions is impossible to obtain by design. Hence, optimality in the formulation of the problem refers to an optimal objective function as calculated at points of the DTM grid.

In a practical sense, the problem is to find a "good" flyable surface rather than to find the best one. The best one cannot be realised in practice anyway, and the results of the previous chapter indicate that the time taken to calculate it is far too long to be of any practical use.

This chapter will deal with some heuristic methods. These proceed from quite basic intuitive methods to algorithms which incorporate lookahead in order to establish the best sequence of raising points. Lastly, a heuristic is presented which makes use of the repeated application of the exact method developed in Chapter 4. Some methods suffered from convergence problems, but the results obtained in other cases were found to be very close to optimal.

6.2 Smoothing Method

When the problem of generating a drape surface occurred, it was first and foremost a problem of generating a feasible surface. In this sense the problem was divorced from the traditional Operations Research approach since there was no aspect of optimisation present. The importance of the following factors was realised, and it was decided that they could not be compromised under any circumstances:

- The separation between the drape surface and the topography may never be less than the flying height.

- The slope of the drape surface in the flight directions may never exceed the given maximum slope (α).

- The second derivative of the drape surface along the flight directions over peaks and in valleys may never exceed $\beta_{\prime\prime}$ and $\beta_{\prime}$ respectively.

These requirements are, of course, exactly the same as those which form the constraints in the standard linear programming formulation of the problem.

It has previously been noted that the first two requirements above are easily met by the first derivative method (Lifting Algorithm) which also runs in polynomial time. Given such a solution which would have pyramid-shaped peaks over the peaks in the topography and valleys with inverted pyramid shapes over the ravines of the DTM, the notion suggests itself that these peaks and valleys could be "rounded out" to adhere to the second derivative requirement. This is the principle behind the Smoothing Method. Fortunately, rounding out of V-shaped valleys in the drape surface will always raise the surface and therefore not overstep the height constraints. In contrast to this, rounding out a peak causes the drape surface to be lowered, bringing the rounded peak closer to the topographic surface. This creates an infeasibility (see Figure 6.1).

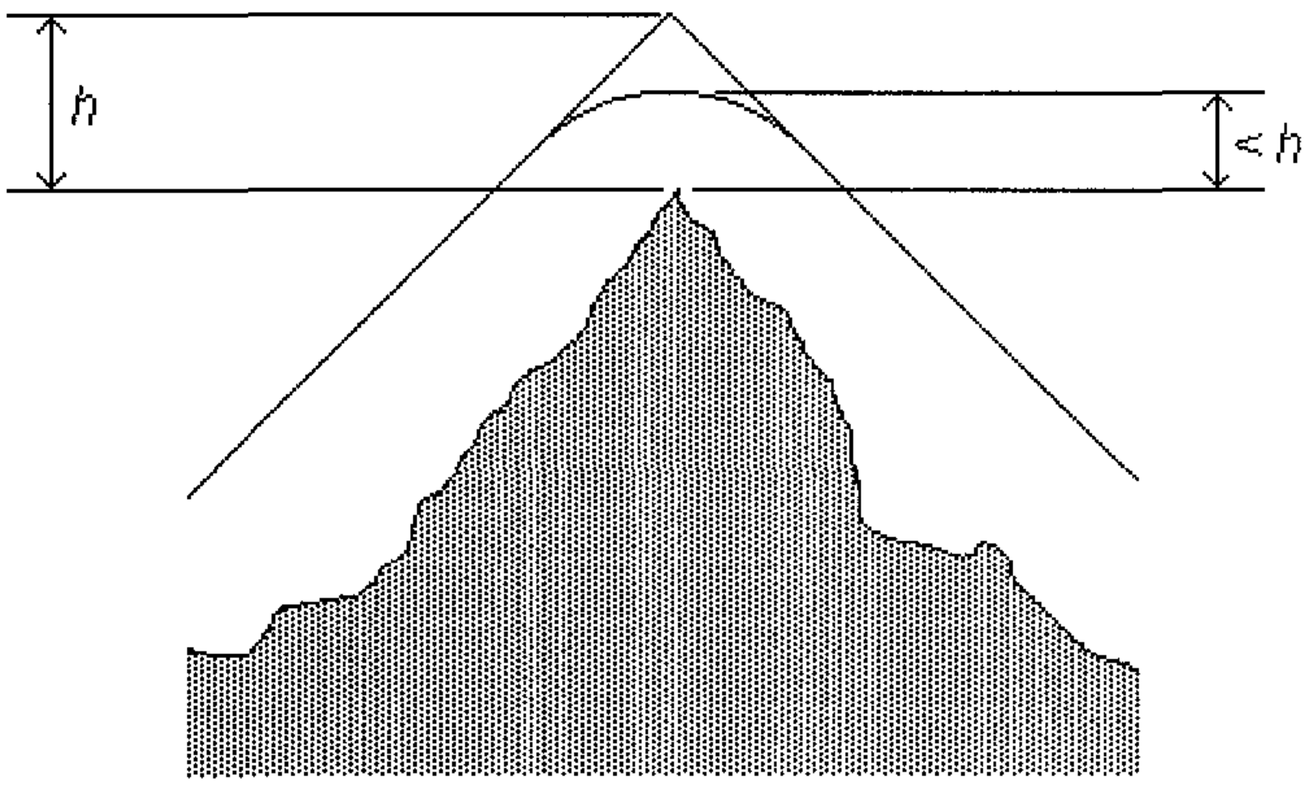


Figure 6.1. Rounding out a peak in the drape surface creates an infeasibility in the height constraint

Nevertheless, this problem can be avoided by lifting the entire drape surface once all required roundings have been completed. The drape surface is raised so that the smallest ground clearance is acceptable to the height constraints.

Various methods of rounding were considered. In the interests of simplicity it was decided to apply a simple smoothing filter over the entire surface. This smoothing filter is merely a convolution filter consisting of a set of equal weights. The size of the filter then determines the ultimate maximum second derivative of the drape surface.

In order to explain the process more fully, consider a linear cut through the topography for which a drape surface is to be generated. Begin by applying the Lifting Algorithm of Chapter 3. After this, there can no longer be any sections of the drape surface whose first derivative exceeds α .

The second derivative may still exceed the constraints, usually at peaks and troughs. Consider a peak which is so sharp that the first derivative drape surface falls off with maximum slope on either side (see Figure 6.2). The greatest possible transgression of the second derivative limits occurs in this case. If the height of the drape surface peak is z_0, the height of the points of the drape surface around the peak will be

$$\ldots,\ z_0 - 3\alpha\ ,\ z_0 - 2\alpha\ ,\ z_0 - \alpha\ ,\ z_0,\ z_0 - \alpha\ ,\ z_0 - 2\alpha\ ,\ z_0 - 3\alpha\ ,\ldots$$

as shown in Figure. 6.2 below.

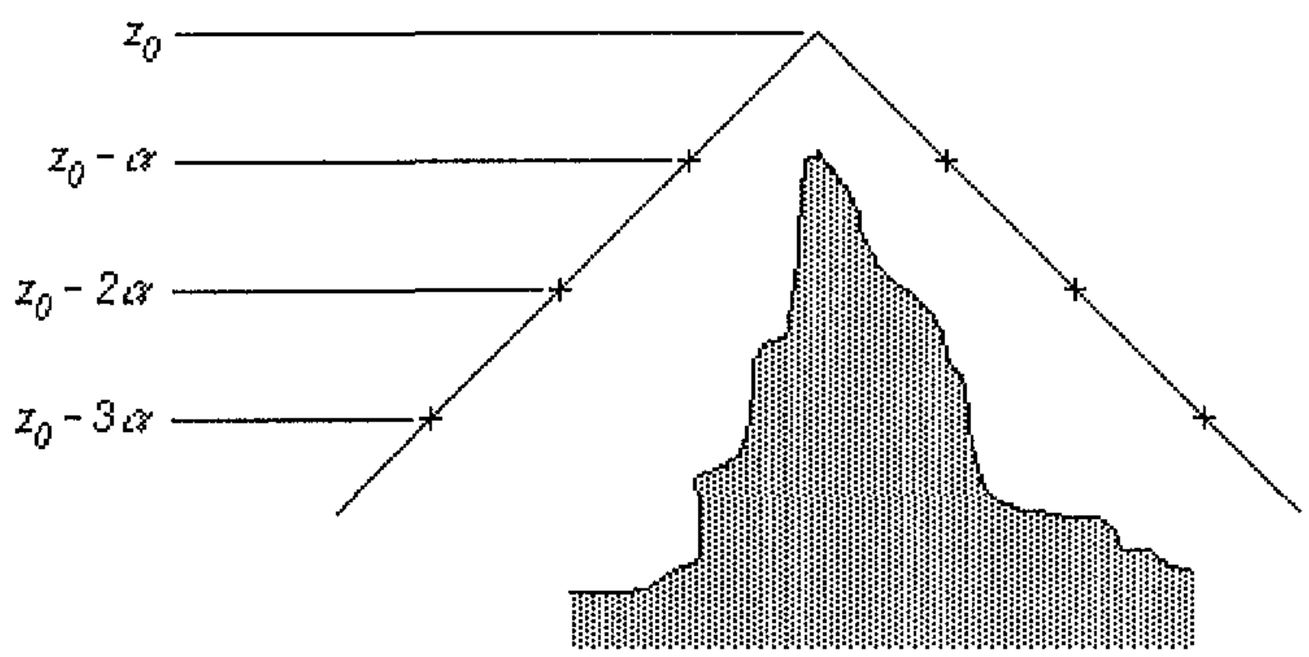


Figure 6.2. Maximum slope around a peak after applying the Lifting Algorithm

Now consider the new values of the drape surface after various convolution filters have been applied. Applying a set of three equal weights results in the values in the table below.

				Peak			
Original height	$z_0 - 3\alpha$	$z_0 - 2\alpha$	$z_0 - \alpha$	z_0	$z_0 - \alpha$	$z_0 - 2\alpha$	$z_0 - 3\alpha$
Height after convolution	$z_0 - 3\alpha$	$z_0 - 2\alpha$	$z_0 - \alpha$	$z_0 - \frac{2}{3}\alpha$	$z_0 - \alpha$	$z_0 - 2\alpha$	$z_0 - 3\alpha$

Only at the peak is the drape surface lower than the Lifting Algorithm height.

If a set of five equal weights is applied, the following heights are obtained.

				Peak			
Original height	$z_0 - 3\alpha$	$z_0 - 2\alpha$	$z_0 - \alpha$	z_0	$z_0 - \alpha$	$z_0 - 2\alpha$	$z_0 - 3\alpha$
Height after convolution	$z_0 - 3\alpha$	$z_0 - 2\alpha$	$z_0 - \frac{7}{5}\alpha$	$z_0 - \frac{6}{5}\alpha$	$z_0 - \frac{7}{5}\alpha$	$z_0 - 2\alpha$	$z_0 - 3\alpha$

The three central points are lowered in this case.

If seven equal weights are applied, the results are as in the table below.

				Peak			
Original height	$z_0 - 3\alpha$	$z_0 - 2\alpha$	$z_0 - \alpha$	z_0	$z_0 - \alpha$	$z_0 - 2\alpha$	$z_0 - 3\alpha$
Height after convolution	$z_0 - 3\alpha$	$z_0 - \frac{16}{7}\alpha$	$z_0 - \frac{13}{7}\alpha$	$z_0 - \frac{12}{7}\alpha$	$z_0 - \frac{13}{7}\alpha$	$z_0 - \frac{16}{7}\alpha$	$z_0 - 3\alpha$

For filters of larger size the same method is applied. The sketch below shows the successive roundings generated by the above filters.

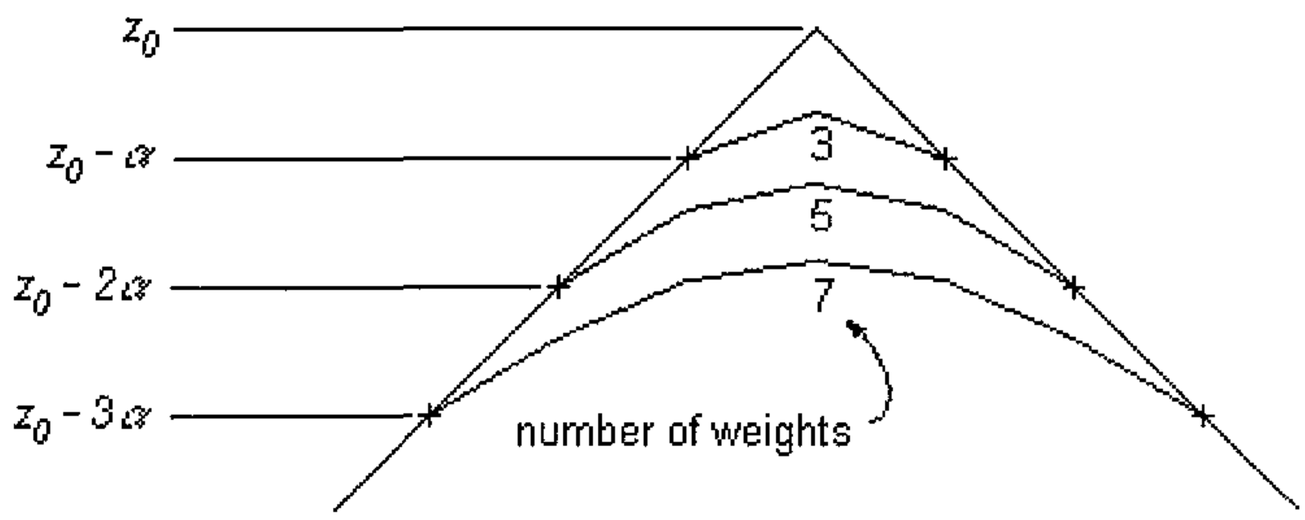


Figure 6.3. Rounding achieved with convolution filters of 3, 5 and 7 equal weights respectively

For the case of maximum slopes falling off to either side of a peak (as shown in Figure 6.2), the following formula gives the new height of the central peak after convolution of the one-dimensional acceptable surface with a filter of length f,

$$z_0' = z_0 - \frac{f^2 - 1}{4f}\alpha .$$

The new height of off-centre points is given by the formula

$$z_j = z_{-j} = z_0 - \frac{f^2 + 4j^2 - 1}{4f}\alpha ,$$

where j is the number of grid points from the peak. Clearly $j < \frac{f-1}{2}$, since the "smoothed" points on the flat sides of the peak will not be moved by convolution ($\frac{f-1}{2}$ is half the filter length).

It is now necessary to determine what size filter will result in all second derivatives of the peaks of the drape surface being smaller than or equal to β_u.

First consider the second derivative of the central three points of a smoothed peak. These points will have respective heights of z_{-1}, z_0 and z_1 with $z_{-1} = z_1$. The second derivative at the peak is then

$$\frac{\partial^2 z}{\partial x^2} \approx \frac{-z_{-1} + 2z_0 - z_1}{2} = \frac{2z_0 - 2z_1}{2} = z_0 - z_1 .$$

Applying the formulae above results in the following expression for the second derivative at the peak.

$$\begin{aligned} \frac{\partial^2 z}{\partial x^2} &\approx (z_0 - \frac{f^2 - 1}{4f}\alpha) - (z_0 - \frac{f^2 + 3}{4f}\alpha) \\ &= \frac{\alpha}{f} \end{aligned}$$

Similarly, if we consider three adjacent points off the peak on the smoothed drape surface, $j-1$, j and $j+1$, these will have heights of z_{j-1}, z_j and z_{j+1} respectively. The second derivative at z_j with $j \neq 0$, is then

$$\begin{aligned} \frac{\partial^2 z}{\partial x^2} &\approx \frac{-z_{j-1} + 2z_j - z_{j+1}}{2} \\ &= \frac{-[z_0 - \frac{f^2 + 4(j-1)^2 - 1}{4f}\alpha] + 2[z_0 - \frac{f^2 + 4j^2 - 1}{4f}\alpha] - [z_0 - \frac{f^2 + 4(j+1)^2 - 1}{4f}\alpha]}{2} \\ &= \frac{\alpha}{f} \end{aligned}$$

Hence the second derivative of a smoothed drape surface is $\frac{\alpha}{f}$ at both the peak and the off-peak points. To ensure that the smoothing produces a drape surface with an acceptable second derivative we therefore need

$$\frac{\alpha}{f} \leq \beta_u$$

ie,

$$f \geq \frac{\alpha}{\beta_u}.$$

Therefore, using a filter of at least $\frac{\alpha}{\beta_u}$ weights will guarantee an acceptable drape surface.

The Smoothing Method can thus be stated succinctly as follows:

- Generate an initial drape surface satisfying the first derivative constraints by using the Lifting Algorithm described in Chapter 3.

- Determine the size of the smoothing filter to be used. This will be the larger of $\frac{\alpha}{\beta_u}$ and $\frac{\alpha}{\beta_l}$. Round this value up to the next odd integer. This becomes the size of the filter, f.

- Smooth the drape surface by convolving it with a square filter of the size determined in the last step. This is equivalent to performing the operation

$$z'_{i,j} = \frac{1}{f^2} \sum_{q=-g}^{g} \sum_{p=-g}^{g} z_{i+p,j+q}$$

for applicable points. In this equation, g is half the filter length, rounded down, ie

$$g = \frac{f-1}{2}.$$

Note that this operation requires two arrays for the storage of the grids, one for the grid with acceptable first derivative, the other for the convolved grid.

- Determine the minimum distance between the convolved grid and the original topographic surface. Note that this distance may, in fact, be negative – this would occur at peaks where the rounded drape surface drops so low that it intersects the topography. Raise the entire convolved grid by an amount which will cause this minimum clearance distance to be equal to the minimum flying height. A feasible drape surface has now been constructed.■

The Smoothing Method described above is easily implemented and it executes in polynomial time. The results are, however, quite far from optimal since the smoothing operation also smoothes peaks and valleys which are already within the second derivative specification. Also, if the values of β_l and β_u are different, either peaks or valleys will be smoothed unnecessarily.

Probably the greatest drawback of the Smoothing Method is that the entire rounded surface is lifted, a process which adds considerably to the drape surface height, especially in places where this extra lift would be totally unnecessary.

Some results for the Smoothing Method are shown below, together with the optima calculated using XA. All calculations were done on a 233 MHz Pentium 2 with 64 MB of RAM. The "% difference" column gives the percentage deviation of the heuristic solution from the optimum.

Problem size	Optimum (XA)		Smoothing Method		% difference
	Solution	Time (s)	Solution	Time (s)	
10 × 10	21 172	< 1	21 363	< 1	0,902
25 × 25	130 148	< 1	131 689	< 1	1,184
50 × 50	498 997	6	514 634	< 1	3,134
10 × 10	21 191	< 1	21 363	< 1	0,812
10 × 10	21 224	< 1	21 353	< 1	0,608
25 × 25	130 525	2	131 484	< 1	0,735
52 × 52	542 995	53	552 268	< 1	1,708
60 × 60	729 221	479	731 763	< 1	0,349
75 × 125	1 827 595	69	1 888 205	< 1	3,316
150 × 250	7 304 665	1526	7 458 641	< 1	2,108

In the light of the considerable savings in time, the Smoothing Method produces eminently usable results.

6.3 General Heuristics

This section contains the description of a number of methods of varying success which were developed to solve the drape problem heuristically. After the first general heuristic, the subsequent methods were developed in an attempt to improve performance and prevent unwanted results such as divergence or a lifted surface. They eventually culminated in General Heuristic 5, a method whose results are eminently usable. The methods are all based on the principle that a primal infeasibility can eventually be removed if one of the points which is part of this infeasibility is lifted high enough. Successive applications of this principle should eventually result in a feasible drape surface. In the methods which are described in this section the order in which points are raised is varied, depending on a number of approaches based mainly on intuition.

6.3.1 General Heuristic 1

The success of the Lifting Algorithm for constructing drape surfaces which are feasible as far as the first derivative goes (as described in Chapter 3) suggested that a similar approach might be used to attack the general problem.

A simple iterative procedure which was devised stipulated that the drape surface be generated in much the same way as the single derivative drape surface described in Chapter 3. The notion on which this heuristic is based, is that violated constraints be fixed iteratively until the entire drape surface is primally feasible. Again, points on the drape surface may only be raised.

General Heuristic 1 begins by setting the drape surface equal to the topography, ie $z_{i,j} = d_{i,j}$ for all points (i, j). Since the method involves only lifting operations, no height constraint is ever violated.

The method proceeds by examining pairs of adjacent points or sets of three in-line points; all these points are on the drape surface under construction. For each pair of points the height differential is calculated. If this value is too large, ie if the slope between the two points is too steep, the lower of the two is raised far enough to bring the slope of the line between the two points to its maximum allowed limit.

For example, given the two points (i, j) and $(i+1, j)$ in the same column, at which the current drape surface heights are $z_{i,j}$ and $z_{i+1,j}$ respectively, three cases may exist: the first point may be so low with respect to the second that the slope between them is too large, the slope between the two points may be within acceptable limits, or the first point may be too high. In the first and the last of these cases the lower of the points is raised; the table below summarises the operation.

Value of slope $s = z_{i,j} - z_{i+1,j}$	**Action**	**New value of** $z_{i,j}$	**New value of** $z_{i+1,j}$
$s < -\alpha$	raise $z_{i,j}$	$z_{i+1,j} - \alpha$	$z_{i+1,j}$
$-\alpha \le s \le \alpha$	none	$z_{i,j}$	$z_{i+1,j}$
$\alpha < s$	raise $z_{i+1,j}$	$z_{i,j}$	$z_{i,j} - \alpha$

The case of three successive points in one row or column is handled as follows: If the second derivative between the three points exceeds β_l or β_u, the lowest lying of the three points is raised far enough to bring the second derivative between the three points to feasibility. For example, assume that the three points under consideration are $(i-1, j)$, (i, j) and $(i+1, j)$. The table below lists the situations which may occur, together with the action which is taken in each case.

Value of 2nd derivative $d = -z_{i-1,j} + 2z_{i,j} - z_{i+1,j}$	**Case**	**Action**	**New value of** $z_{i-1,j}$	**New value of** $z_{i,j}$	**New value of** $z_{i+1,j}$
$d < -\beta_l$	1	raise $z_{i,j}$	$z_{i-1,j}$	$\frac{z_{i-1,j} + z_{i+1,j} - \beta_l}{2}$	$z_{i+1,j}$
$-\beta_l \le d \le \beta_u$	2	none	$z_{i-1,j}$	$z_{i,j}$	$z_{i+1,j}$
$\beta_u < d$	3a	raise $z_{i-1,j}$	$2z_{i,j} - z_{i+1,j} - \beta_u$	$z_{i,j}$	$z_{i+1,j}$
	3b	raise $z_{i+1,j}$	$z_{i-1,j}$	$z_{i,j}$	$2z_{i,j} - z_{i-1,j} - \beta_u$

The actions in the table are shown in Figure 6.4 below. Note that case 3a is simply a mirror image of case 3b where the left-hand point is raised instead of the right-hand one.

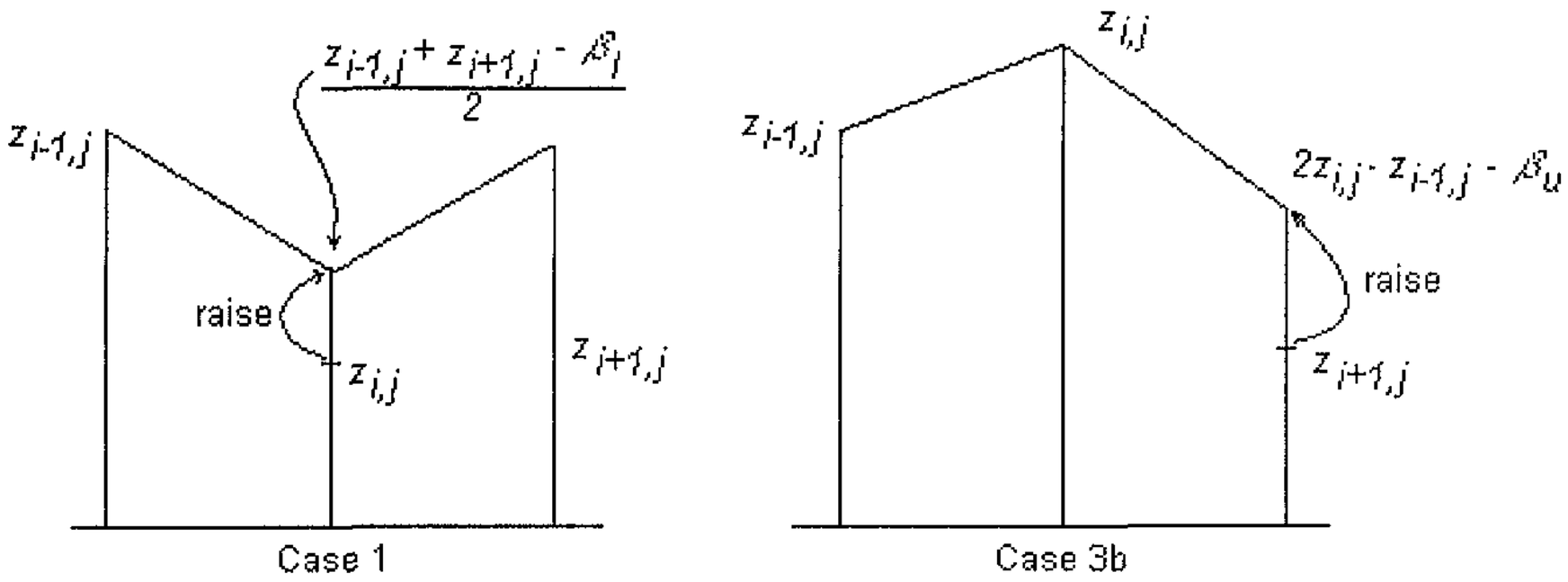


Figure 6.4. Raising the lowest point to obtain a feasible second derivative

General Heuristic 1

Step 1 (*Create initial drape surface*): Set the initial drape surface to the DTM, ie $z_{i,j} = d_{i,j}$ for all points (i, j).

Step 2 (*Forward pass*): Make the top left point the current point. Compare its height with its right-hand neighbour (if this neighbour exists). If the height differential exceeds α, lift the lower of the two points until the slope between the two is acceptable. Perform the same operation with the point's lower neighbour. Next, consider the point's relation to its left and right neighbours (insofar as these exist). If the three points form a valley whose second derivative exceeds β_l, raise the central point far enough for the second derivative to be in kilter. If the three points form a peak which is too pointy, raise the lower of the two neighbours of the current point until the second derivative over the peak is β_u. Perform the same operation using the top and bottom neighbours of the current point. Continue with the next point, moving to the right and advancing line by line (standard reading order) until the lower right-hand corner of the grid has been reached.

Step 3 (*Reverse pass*): Make the lower right point the current point. Perform the slope check and adjustment operation with the point's left-hand neighbour, then with its upper neighbour. Then consider the point together with its left and right neighbours and check and adjust the second derivative. Do the same using the point and its upper and lower neighbours. Continue with the next point, moving to the left and upwards line by line (reverse reading order) until the top left-hand corner has been reached.

Step 4 (*Iterate*): If any changes were made during the forward or reverse passes (steps 2 or 3), go back to step 2. Else the iterative part of the method is finished; go to step 5.

Step 5 (*Adjust overall height*): Raise the entire generated drape surface by the minimum flying height. Stop.■

Some results obtained with General Heuristic 1 are shown in the table below, compared to the optimal results obtained using XA, all performed on a 233 MHz Pentium 2 with 64 MB of RAM.

Problem size	Optimum (XA)		General Heuristic 1			% difference
	Solution	Time (s)	Solution	Iterations	Time (s)	
10 × 10	21 172	< 1	21 172	2	< 1	0,0
25 × 25	130 148	< 1	130 149	2	< 1	0,001
50 × 50	498 997	6	523 595	49	1	4,930
10 × 10	21 191	< 1	21 195	11	< 1	0,019
10 × 10	21 224	< 1	21 288	14	< 1	0,302
25 × 25	130 525	2	134 306	23	< 1	2,897
52 × 52	542 995	53	Lift off	748	2	-
60 × 60	729 221	479	Lift off	141	1	-
75 × 125	1 827 595	69	Lift off	159	1	-
150 × 250	7 304 665	1526	7 325 158	37	1	0,281

Although General Heuristic 1 is straightforward to apply it suffers from two drawbacks:

- There is no guarantee that it will converge. However, it has been found in practice that a final solution is always found. Hence, in practical terms, convergence is not a problem.

- No heuristic necessarily attains optimality, and it was found here that the drape surface can "lift off" the topography altogether and hover in space high above the actual surface to be draped. Clearly, such a drape surface may be primally feasible but is not optimal since it can be lowered. In the rest of this document *lift off* refers to this situation where the method has converged but the drape surface hovers above the topography.

6.3.2 General Heuristic 2

General Heuristic 1 described above rules rather arbitrarily that it is always the lowest of the three points which form a second derivative infeasibility between them, which is the one to be raised. Certainly, in the case of valley constraints being exceeded, this selection makes sense. In the case of peak infeasibilities, there is no real reason for choosing the lowest of the three points; obviously, the choice will fall upon one of the two outer points.

Indeed, the amount of lift incurred to bring a peak infeasibility to a feasible state is exactly the same, regardless of whether the lower or the higher end point is lifted. Consider the three points in question having drape heights of $z_{i-1,j}$, $z_{i,j}$ and $z_{i+1,j}$ respectively, such that $-z_{i-1,j}+2z_{i,j}-z_{i+1,j}\geq\beta_u$. Assume that $z_{i-1,j}$ is lower than $z_{i+1,j}$. In order to make this peak feasible using General Heuristic 1 of the previous section, $z_{i-1,j}$ would need to be increased to $2z_{i,j}-z_{i+1,j}-\beta_u$, an increase of

$2z_{i,j} - z_{i+1,j} - \beta_u - z_{i-1,j}$. If, instead, the other extremity were raised, $z_{i+1,j}$ would have to be lifted to $2z_{i,j} - z_{i-1,j} - \beta_u$, an increase of $2z_{i,j} - z_{i-1,j} - \beta_u - z_{i+1,j}$. This figure is the same as the one obtained for raising the first point. In other words, it might be conjectured that raising the higher of the two extremities would give the same overall drape surface.

Implementing this approach resulted in General Heuristic 2. This is identical to General Heuristic 1, except that the higher of the two extremities is raised in the case of peak infeasibilities, instead of the lowest of the three points being lifted.

Results obtained with General Heuristic 2 as compared with the optimal results from XA are shown in the table below. Again, a 233 MHz Pentium 2 with 64 MB of RAM was used in the calculations.

Problem size	**Optimum (XA)**		**General Heuristic 2**			**% difference**
	Solution	**Time (s)**	**Solution**	**Iterations**	**Time (s)**	
10 × 10	21 172	< 1	21 172	2	< 1	0,0
25 × 25	130 148	< 1	130 149	2	< 1	0,001
50 × 50	498 997	6	Diverges	-	-	-
10 × 10	21 191	< 1	Diverges	-	-	-
10 × 10	21 224	< 1	Diverges	-	-	-
25 × 25	130 525	2	Diverges	-	-	-
52 × 52	542 995	53	Diverges	-	-	-
60 × 60	729 221	479	Diverges	-	-	-
75 × 125	1 827 595	69	Diverges	-	-	-
150 × 250	7 304 665	1526	Diverges	-	-	-

Where General Heuristic 1 exhibited the unwanted symptom of lift off, General Heuristic 2 suffered from divergence. The objective function simply increased without

limit and the method never converged. As a heuristic, this method therefore has only limited value.

6.3.3 General Heuristic 3

An alternative to the straightforward General Heuristics 1 and 2 is General Heuristic 3. In this case the forward and reverse passes are each broken up into two distinct operations. The first of these checks and adjusts the first derivative values while the second operation takes care of the second derivative values. (General Heuristic 1 examined both derivatives in both directions at every point before moving on to the next point.)

The table below lists the results obtained for General Heuristic 3, compared to those obtained using XA. Again, these were obtained on a 233 MHz Pentium with 64 MB of memory.

Problem size	Optimum (XA)		General Heuristic 3			% difference
	Solution	Time (s)	Solution	Iterations	Time (s)	
10 × 10	21 172	< 1	21 172	2	< 1	0,0
25 × 25	130 148	< 1	130 149	2	< 1	0,001
50 × 50	498 997	6	587 261	39	1	17,688
10 × 10	21 191	< 1	21 195	11	< 1	0,019
10 × 10	21 224	< 1	21 295	14	< 1	0,335
25 × 25	130 525	2	140 460	33	< 1	7,612
52 × 52	542 995	53	Lift off	54	< 1	-
60 × 60	729 221	479	Lift off	63	< 1	-
75 × 125	1 827 595	69	Lift off	168	1	-
150 × 250	7 304 665	1526	Lift off	287	8	-

As with General Heuristic 1, convergence was found in each instance but in many cases the solution was a surface hanging in the air above the DTM and unconnected to it. In general it seemed to work reasonably well for small examples but lifted off the terrain for larger models.

6.3.4 General Heuristic 4

From the relatively unimpressive performance of the previously described methods and in view of the consequences of lifting different sides of a peak (as discussed below) it is clear that a rudimentary rule such as "always lift the lowest point" or "always lift the other point" is insufficient for obtaining acceptable results. Also, the order in which various adjustments are made would not appear to have any great influence on the success of the method. Consequently, an attempt was made to widen the view of the drape surface at an infeasibility.

The three points which straddle a peak infeasibility (ie one whose second derivative exceeds $\beta_{"}$) are clearly not the only ones to be taken into account when trying to smooth out the peak. Consider the following cut along a line, where the topography consists of a single peak closer to one end of the line's space than the other. The peak is at position i, the topography is slightly higher at $i-1$ than at $i+1$. See Figure 6.5 below.

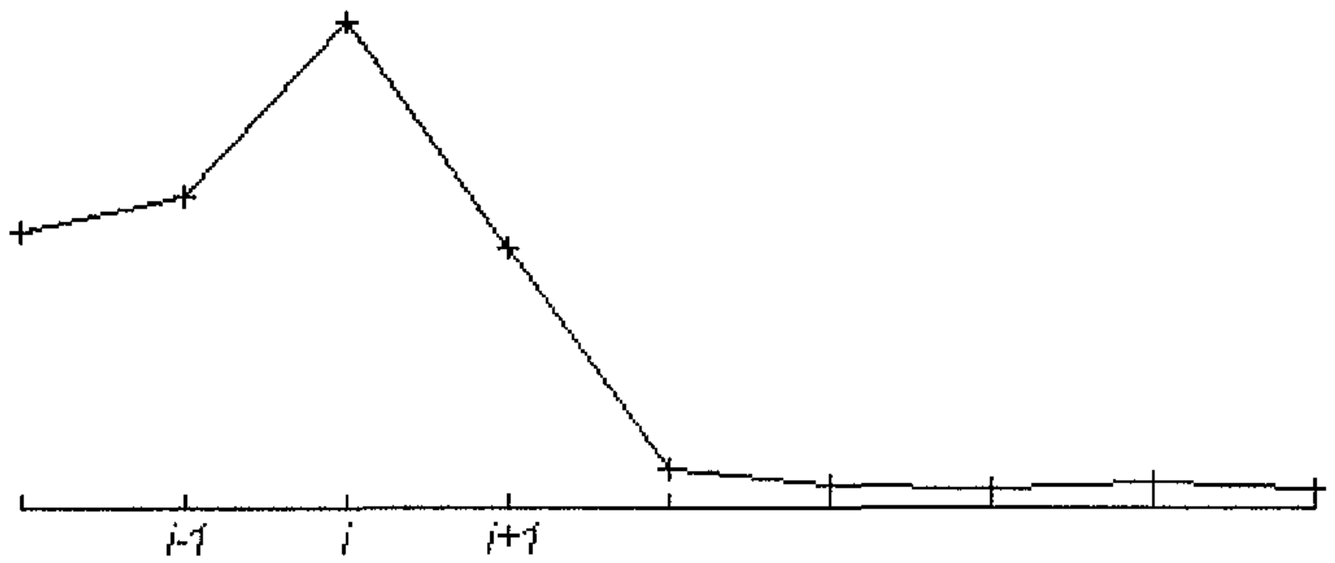


Figure 6.5. Example of topography with single peak on side

Initially the drape surface is set to the DTM. Hence the initial surface in the figure above also has a peak as shown. We assume that the peak at i has a second derivative which exceeds $\beta_{\prime\prime}$. General Heuristic 1 now dictates that the point at $i+1$ be lifted until the second derivative at point i is acceptable. This is shown in Figure 6.6 below.

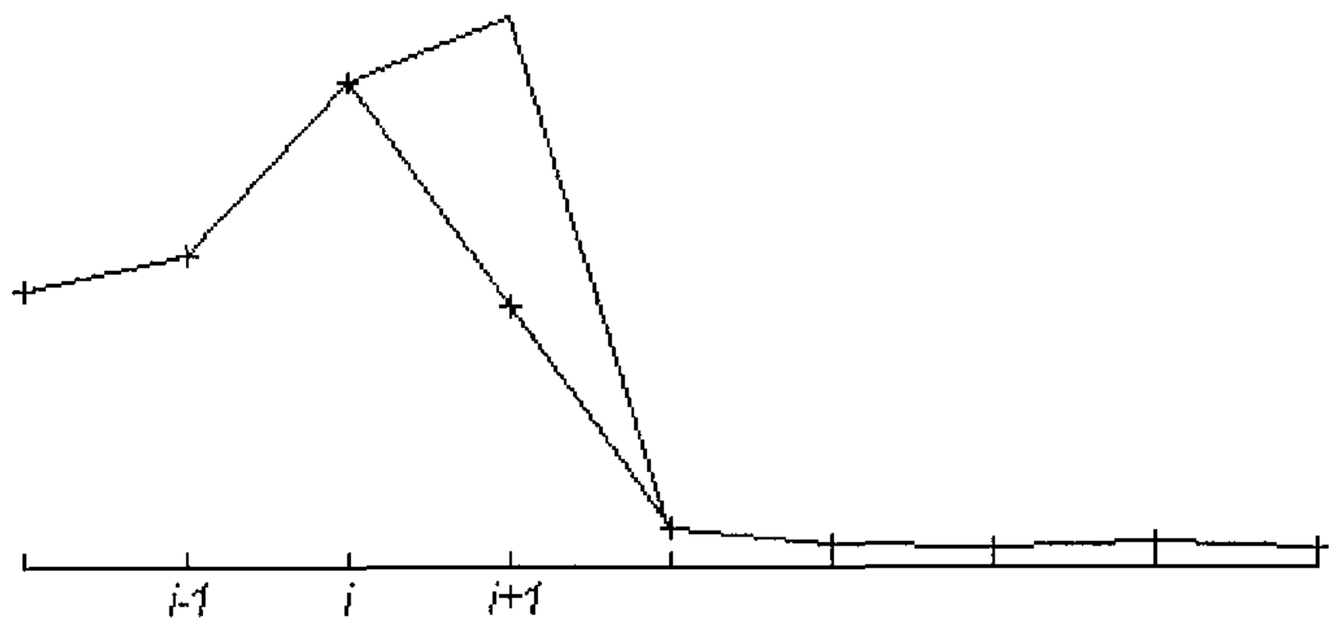


Figure 6.6. The infeasibility at point i has been removed

Observe how the raising of point $i+1$ has caused an infeasible peak to appear at precisely that point. The next iteration of the heuristic will fix this – see Figure 6.7 below;

it does, however, cause another problem at point $i+2$.

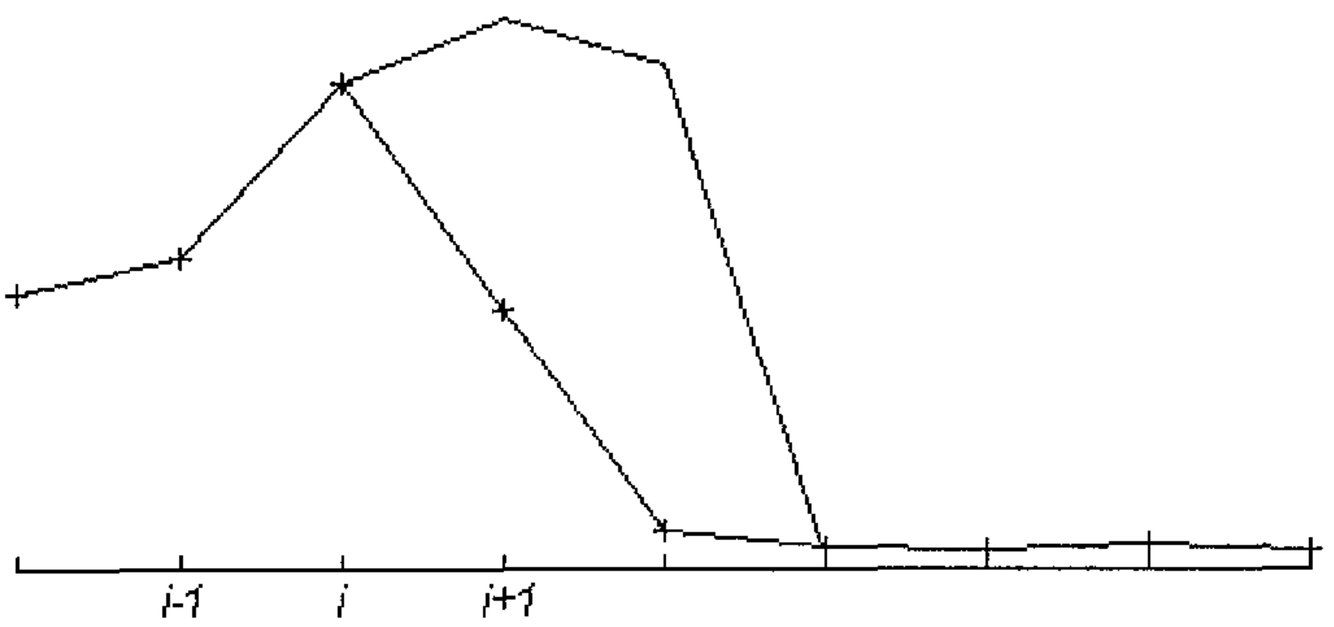


Figure 6.7. Point $i+2$ has been lifted to remove the infeasibility at point $i+1$

The method continues in this fashion until the situation shown in Figure 6.8 is reached. In this figure the shaded area indicates the total lift which was applied to generate the drape surface. (The surface shown is not feasible because of the sharp valley crease which exists where the lifted surface rejoins the more or less flat topography at the far right.)

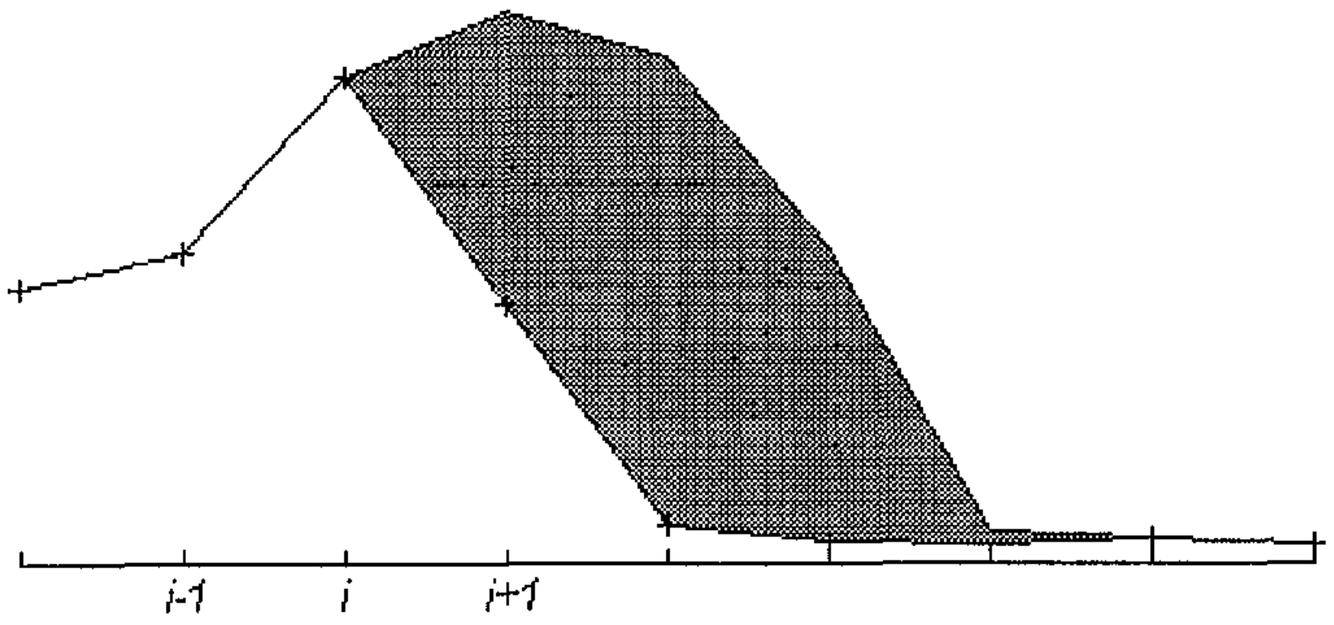


Figure 6.8. Semi-completed drape surface obtained when point $i+1$ is the one lifted to resolve the peak infeasibility

If, on the other hand, the peak infeasibility is addressed by lifting point $i-1$ instead of $i+1$, the drape surface as shown in Figure 6.9 is obtained.

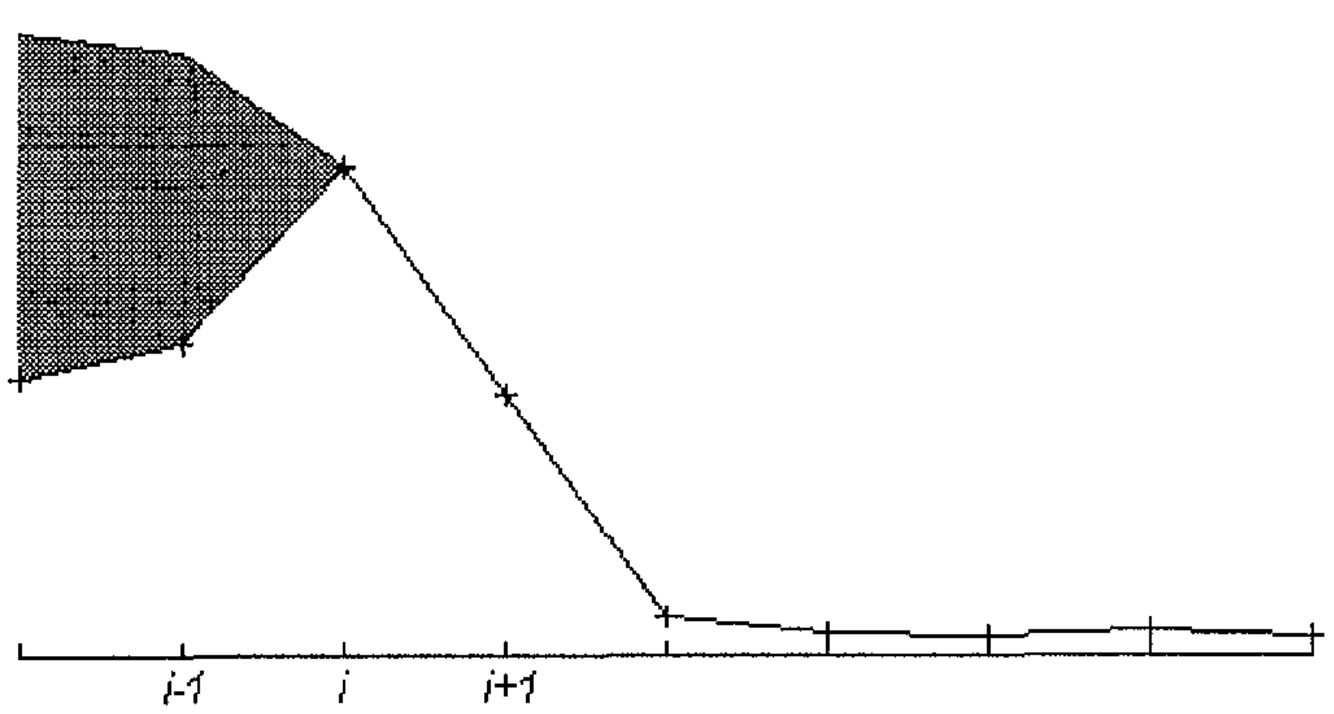


Figure 6.9. Semi-completed drape surface obtained when point $i-1$ is the one lifted to resolve the peak infeasibility

In Figures 6.8 and 6.9 above, the shaded area represents the total lift applied in each case, had the problem been a continuous one. In the discrete case the total lift is the sum of the differences in height between the drape surface and the topography at each of the abscissas. However, the shaded area is easier to visualise and illustrates the total discrete lift. The example shows that rather different increases in the objective function can be achieved depending on how the lifting operation is sequenced. Similarly, the choice of "always lift the lowest point" as opposed to "always lift the second-lowest point" will result in a better drape surface in some cases than in others.

There is thus some merit in "looking ahead" to see what the cumulative effect will be of lifting one side of a peak infeasibility rather than the other, before actually embarking on this operation. For example, in the case above, it was more advantageous to lift the drape

surface to the left of the peak rather than to its right. The main reason for this would appear to be the fact that there are fewer grid points to the left of the topographical peak than to its right. Indeed, in a totally symmetrical situation, the drape surface should be lifted on that side of the peak which is closer to an edge. In the more general case that side of the surface should be lifted which will dive below the topography more rapidly (or run out of the grid). Clearly, these are fairly harsh generalisations which cannot possibly apply in all cases; valid counterexamples are easily constructed. Nevertheless, this approach can be applied in a heuristic method.

In the proposed lookahead method, the drape surface is constructed from the highest point downwards. The reason for this approach is that higher peaks are far more likely to influence the shape of the drape surface lower down than *vice versa*. The method thus repeatedly hunts out the highest infeasibility and adjusts its surroundings before proceeding to the next one.

The method considers linear cuts through the area to be draped – first in the x-direction and then in the y-direction. Initially the method examined the points on either side of the infeasible peak and chose to raise the point on the side which had the shortest downward run. Consider the cut shown in Figure 6.10.

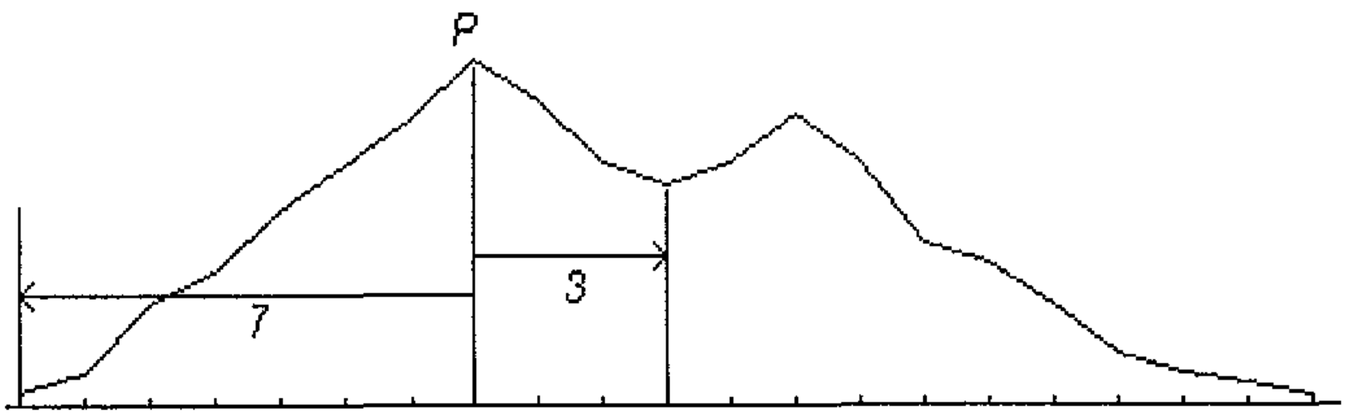


Figure 6.10. Determining which side of a peak to lift

In the example above, point P represents a point whose second derivative is too large. The downward run to the left of the peak is seven units before the line runs out. To the right there is a run of three units before the current drape surface rises again. If we lifted the left side we would have to lift a total of seven points, while only three points would need lifting if the right-hand neighbour of P were raised. Therefore the algorithm would decide to lift the right-hand neighbour.

Originally the notion described above was implemented as follows: the highest point whose second derivative was infeasible (in either the x or the y-direction) was determined. Clearly, if no such point could be found, the solution was feasible. The infeasibility in either direction was then removed by lifting the neighbour on the side with the shortest run. Then the hunt proceeded for the next point with a peak infeasibility.

When this method was coded it occasionally went into an infinite loop; sometimes it was found that the drape surface had lifted right off the topography. An example of this behaviour is given below.

Consider an intermediate stage of the generation of the drape surface with the following values. The value of β_i is once again assumed to have the value 1,0.

	19,0		
	19,0	19,0	19,0
19,0	19,2	20,0	19,9
		18,5	

The shaded square has a peak infeasibility in the north-south direction. The shortest run is to the north, hence its northern neighbour (19,0) must be lifted. This results in the values in the next sketch.

	19,0		
	19,0	20,5	19,0
19,0	19,2	20,0	19,9
		18,5	

In the last sketch above there is now a peak infeasibility in the east-west direction at the shaded square. We assume that the shortest run is to the west, hence the left-hand neighbour of the shaded cell is the one to be lifted. The following situation is then obtained.

	19,0		
	21,0	20,5	19,0
19,0	19,2	20,0	19,9
		18,5	

This results in the shaded cell (with value 21,0) having a north-south infeasibility. Assume that the shortest run for lifting is in a southerly direction. This means that the value of the cell's southern neighbour, viz 19,2, must be increased. The grid below is obtained after lifting.

	19,0		
	21,0	20,5	19,0
19,0	22,0	20,0	19,9
		18,5	

In the last figure there is an east-west peak infeasibility at the shaded cell. Assume that the shortest run is to the east this time. Raising the eastern neighbour of the shaded cell results in the diagram below.

	19,0		
	21,0	20,5	19,0
19,0	22,0	24,0	19,9
		18,5	

There is now an infeasibility in exactly the same place where one was previously encountered. The last shaded cell which initially was assumed to touch the peak has lifted off the DTM completely and taken several other points with it. (In practical problems the crucial steps shown above are often interspersed with other steps which attempt to iron out other infeasibilities.)

Clearly, this method required some modification.

A system of marking the points was subsequently added. A point is marked as fixed when all its infeasibilities in both directions have been resolved. The method begins with the

drape surface the same as the first derivative feasible solution. All points are set as unmarked.

The method proceeds by repeatedly finding the highest unmarked point in the drape surface. Before this point is marked, the peak second derivative between it and its left and right neighbours is checked. If this is too large, the neighbour on the side with the shortest run is lifted high enough to bring the second derivative to an acceptable value. The same process is also applied to the northern and southern neighbours. The originally unmarked central point is now marked as fixed.

The decision to use the point on the side with the shortest run is only applied if both neighbours of the point under consideration are unmarked. If one of the neighbours is marked as fixed, the other neighbour is raised. If both neighbours are fixed, the current point is in a valley and this is smoothed out in the valley stage of the algorithm.

Whenever a point is raised, the method checks that its neighbours are still within the first derivative limits. If this is no longer the case, the neighbours which lie too low are also raised in order to maintain first derivative feasibility.

General Heuristic 4

Step 1 (*Initial surface*): Generate a drape surface which is feasible in the first derivative.

Step 2 (*Initial marking*): Ensure that all points are unmarked (not marked as fixed).

Step 3 (*Set current point*): Find the highest unmarked point in the grid. This becomes the current point. If there is no such point, go to step 12.

Step 4 (*Find horizontal second derivative*): Check whether the second derivative from the current point's left neighbour over the current point to the current point's right neighbour is feasible. If this value is below $-\beta_u$ go to step 5. Else go to step 9.

Step 5 (*Both neighbours unmarked*): If either the left or the right neighbour (or both) of the current point are marked, go to step 6. If both points are unmarked, count the number of points lying to the left of the current point before the drape surface rises (or comes to an end). Do the same on the right-hand side. If the right-hand run is shorter, raise the right-hand neighbour of the current point until the second derivative at the current point is feasible. Alternatively raise the left-hand neighbour. Go to step 9.

Step 6 (*Left neighbour unmarked*): If the left neighbour is marked, go to step 7. Else only the right neighbour is marked. Raise the left neighbour until the second derivative over the current point is feasible. Go to step 9.

Step 7 (*Right neighbour unmarked*): If the right neighbour is marked, go to step 8. Else only the left neighbour is marked. Raise the right neighbour until the second derivative over the current point is feasible. Go to step 9.

Step 8 (*Both neighbours marked*): Both neighbours are marked. This is a valley. Do nothing. Go to step 9.

Step 9 (*Vertical operation*): Repeat steps 4 to 8 using the upper and lower neighbours instead of those on the left and right.

Step 10 (*Adjust first derivative*): Check whether the first derivative of the current point with respect to its neighbours is in order. Where it is not, raise the lower-lying of the two.

Step 11 (*Mark current point*): Mark the current point as fixed. Go back to step 3.

Step 12 (*Find infeasible valley*): All the peaks in the grid are now feasible. Find a point in the grid where the second derivative exceeds β_l. This becomes the current point. If no such point can be found, go to step 14.

Step 13 (*Smooth out valley*): Raise the current point until the second derivative in both directions is feasible. Note that this could cause the second derivative at other points to become too large again. Go back to step 12.

Step 14 (*Termination*): The entire grid is feasible. Stop.■

The table below shows results obtained with General Heuristic 4. It also shows the optimal results which were obtained using XA. The results were calculated on a 233 MHz Pentium 2 with 64 MB of memory.

Problem size	**Optimum (XA)**		**General Heuristic 4**			**% difference**
	Solution	**Time (s)**	**Solution**	**Iterations**	**Time (s)**	
10 × 10	21 172	< 1	21 172	1	< 1	0,0
25 × 25	130 148	< 1	130 150	1	1	0,002
50 × 50	498 997	6	Diverges	-	-	-
10 × 10	21 191	< 1	21 203	1	< 1	0,057
10 × 10	21 224	< 1	21 413	1	< 1	0,891
25 × 25	130 525	2	Diverges	-	-	-
52 × 52	542 995	53	Diverges	-	-	-
60 × 60	729 221	479	Diverges	-	-	-
75 × 125	1 827 595	69	Diverges	-	-	-
150 × 250	7 304 665	1526	Diverges	-	-	-

The table above shows that General Heuristic 4 produced quite good results for most smaller datasets. In the case of larger areas its performance was not found to be very good. It often suffered from the divergence which had also been noticed in previously described heuristic methods.

6.3.5 General Heuristic 5

One of the drawbacks encountered with General Heuristic 4 which was described in the previous section was that the peaks which were generated in the drape surface above peaks in the topography tended to be skewed or lopsided, and therefore rather higher than they needed to be. This is shown in the figure below (Figure 6.11).

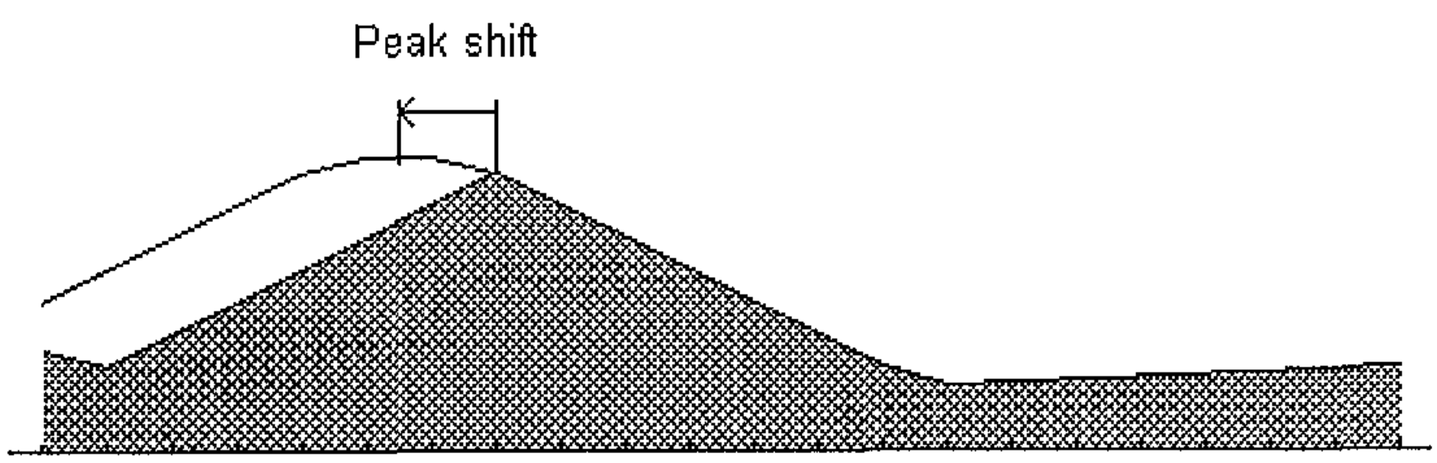


Figure 6.11. Peak in drape surface has shifted from position of peak in topography

General Heuristic 5 places the drape surface peaks more symmetrically onto the topographical peaks. The change here is effected for the case where both neighbours of the current point (which has a second derivative which is too small) are unmarked (not fixed). The method attempts to place a symmetrical hat over the current point. See Figure 6.12.

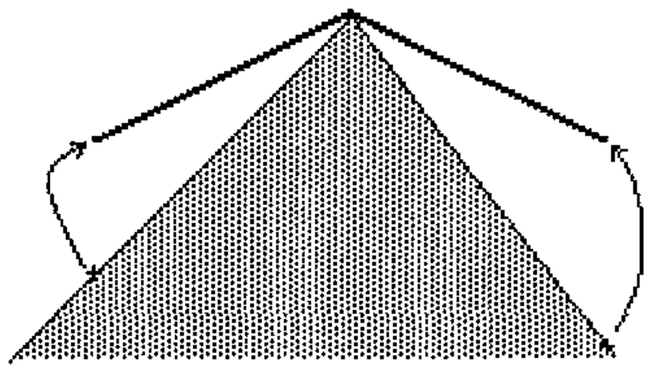


Figure 6.12. Symmetrical lifting around an infeasible peak

In the case where the adjustment of one neighbour to the drape surface would actually require it to be lowered, the hat is brought to rest on that neighbour while its other side is lowered to compensate. This is shown in Figure 6.13.

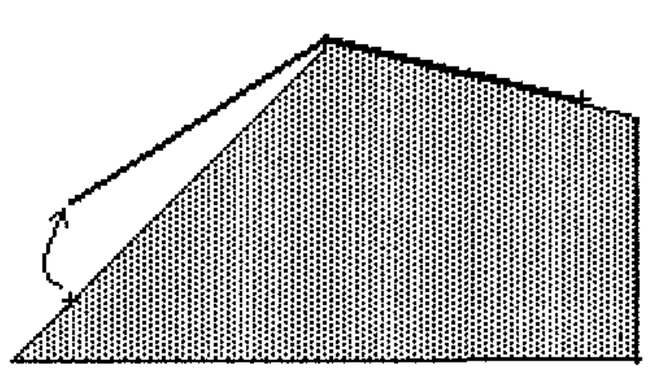


Figure 6.13. Symmetrical placement of hat is not possible – right-hand side rests on existing surface

General Heuristic 5 thus does not cause peak shifting as does the method described in the previous chapter. This, in turn, leads to a better objective function value.

The table below lists the results obtained using General Heuristic 5 and compares them to the optimal results obtained using XA. Again, a 233 MHz Pentium 2 with 64 MB of RAM was used.

Problem size	Optimum (XA)		General Heuristic 5			% difference
	Solution	Time (s)	Solution	Iterations	Time (s)	
10 × 10	21 172	< 1	21 172	1	< 1	0,0
25 × 25	130 148	< 1	130 150	1	< 1	0,002
50 × 50	498 997	6	499 038	1	1	0,008
10 × 10	21 191	< 1	21 193	1	< 1	0,009
10 × 10	21 224	< 1	21 235	1	< 1	0,052
25 × 25	130 525	2	130 625	1	1	0,077
52 × 52	542 995	53	552 188	1	< 1	1,693
60 × 60	729 221	479	732 166	1	1	0,404
75 × 125	1 827 595	69	1 827 969	1	6	0,020
150 × 250	7 304 665	1526	7 305 840	1	172	0,016

The results obtained using General Heuristic 5 compare rather favourably with the others mentioned in this chapter. Where the Smoothing Method produced a deviation of 1,486 % from the optimal value of the objective function – already no mean achievement – General Heuristic 5 has reduced this deviation to 0,228 % - almost a whole order of magnitude. The time taken by these two methods is very much in the same order. The latter method is thus a very serious contender in the race for the best heuristic.

The most important fact, perhaps, about General Heuristic 5 is that the method converged for all examples which it was tried on. It also never generated a surface which had "lifted off" the topographic surface.

6.4 The Alternating One-dimensional Dual Algorithm (AODA)

The method described in this section, the Alternating One-dimensional Dual Algorithm (AODA), has its origins in the Dual Algorithm for the one-dimensional problem which was found to work exceptionally well in Chapter 4. It was hoped that this success would yield a successful heuristic when applied in the two axial directions in turn.

As before, the method begins by generating a drape surface which is equal to the DTM. If the slopes between all pairs of adjacent points and the second derivatives across every set of three points are all within the acceptable limits, the problem has already been solved. In the general case there will, of course, be points where some of these limits are exceeded.

The method now proceeds by examining each line of points in turn. Each such cut is treated as a one-dimensional drape problem and solved using the exact method which was developed in Chapter 4. Once the feasible drape surface along this cut has been determined, it is substituted for the grid line in the two-dimensional drape surface. For each line the relevant section through the current drape surface is taken to be the topography. This means that no point in the drape surface is ever lowered and that the height constraints are therefore never overstepped. Once all the lines parallel to the x-axis have been examined in turn, the method then examines all the lines running perpendicular to these in the same manner. It then proceeds to consider the first set of lines again and continues to iterate in this fashion until no further changes are necessary. If any section line under consideration does not contain any violations of the required constraints, it is skipped.

The table below shows the results obtained using the AODA compared with the optima obtained using XA. The number of iterations is the number of times every cut in both the x and y directions was examined before feasibility was obtained. Although it is probably not possible to prove that the method will always converge, it certainly did so in all cases

to which it was applied. The results in the table were obtained using a 233 MHz Pentium 2 with 64 MB of memory.

Problem size	Optimum (XA)		AODA			% difference
	Solution	Time (s)	Solution	Iterations	Time (s)	
10 × 10	21 172	< 1	21 172	2	< 1	0,0
25 × 25	130 148	< 1	130 148	2	< 1	0,0
50 × 50	498 997	6	499 045	4	1	0,010
10 × 10	21 191	< 1	21 193	3	< 1	0,009
10 × 10	21 224	< 1	21 230	4	< 1	0,028
25 × 25	130 525	2	130 569	4	< 1	0,034
52 × 52	542 995	53	550 648	16	4	1,409
60 × 60	729 221	479	730 164	19	6	0,129
75 × 125	1 827 595	69	1 827 817	4	0	0,012
150 × 250	7 304 665	1526	7 305 129	44	6	0,006

The results produced by the AODA compare very favourably with the optimal values obtained by using an exact method. Except for the case of the 52 × 52 problem all deviations from the optimum are rather less than 0,13 % and therefore represent eminently usable approximations in practice.

The tabulated results in this chapter will be discussed in more detail in the last chapter (Chapter 8) in which the various methods will be weighed up against each other.

Chapter 7
The Two-dimensional Drape Problem – Heuristic Methods Based on Thimbles

7.1 Introduction

With many of the heuristics developed and described in the last chapter, the heart of the problem of generating an acceptable drape surface lay in "fattening" peaks across which the second derivative was out of bounds in such a way that the resultant second derivative was satisfactory. The problem can be seen as a sort of water bed or even a jelly-like surface which covers the topography. Adjusting the drape surface over a peak by moving it to one side would cause the drape surface on that side to rise while it would be lowered on the opposite side. Figure 7.1 shows a qualitative example on a linear drape surface. Two possible drape surfaces are shown. At the leftmost peak the solid line drape surface lies to the left of the dotted one, while it lies to the dotted one's right at the central peak. The water bed effect is achieved by considering what would happen at the valleys A and B in the figure when the drape peak on the left is moved from the solid to the dotted position. This would cause the valley at A to rise while the one at B would be lowered. Whether the solid or the dotted drape is better (ie of better solution quality) is hard to gauge; there is probably not much to choose between the two. The improvement which is gained by lowering the surface at B will be offset by raising it in valley A.

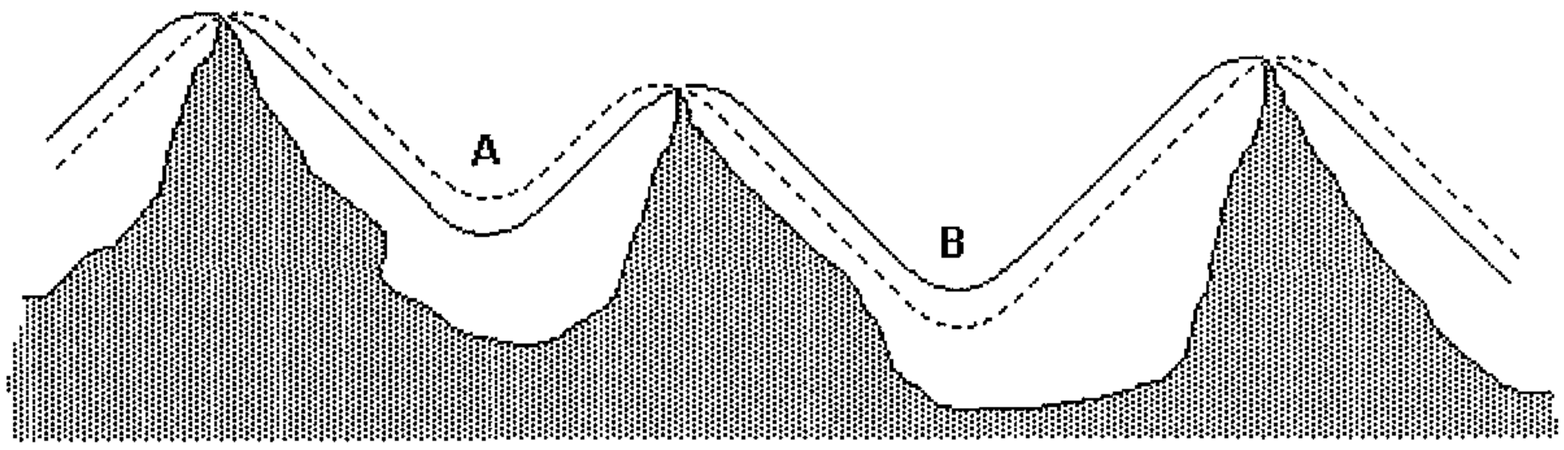


Figure 7.1. Two possible drapes over a simple linear surface to illustrate the "water bed" effect

When this illustration is extended to the three-dimensional case it becomes even harder to picture. Adjusting the drape surface over a particular peak could cause other vertical movements over valleys which would be hard to establish analytically, especially if the drape surface actually started to touch the ground when it was lowered in places.

7.2 The Thimble Approach

Following on the discussion above, it will be seen that the drape surface will require a cap of a constant shape to be placed upon each peak of the topography whose second derivative falls outside the bounds of the problem. Such a cap is called a *thimble*, because of its shape along an axis. Clearly the thimble will eventually intersect the topography or extend beyond the edges of the area under consideration.

The general idea with thimble methods is that one or more thimbles are placed over the peaks of the topography, as appropriate. Where the sides of two thimbles run into each other a valley will be formed. Valleys where the second derivative is too large are subsequently filled up using the method of section 7.4. The result is then a feasible drape surface which is not necessarily optimal but should be reasonably good.

7.3 Shape and Placing of Thimbles

Each cut through a thimble (along one of the coordinate axes) would look as follows if we were considering a continuous system (see Figure 7.2).

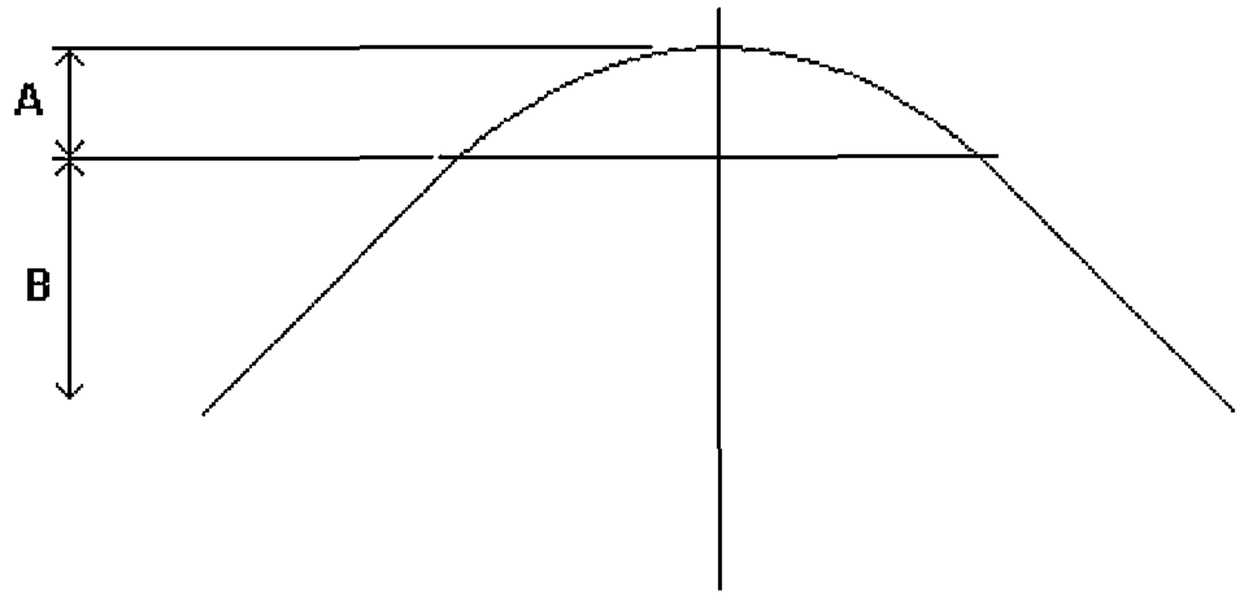


Figure 7.2. A section through a continuous thimble in an axial direction

The top of the section through the thimble is parabolic, this is indicated by the letter A in the figure. In this section the second derivative's value lies at β_u. In section B the drape line is straight and has a slope of $\pm\alpha$, the maximum permissible first derivative in the drape line.

In three dimensions the continuous thimble is not, as might be expected, a solid of revolution produced by rotating the figure above around its vertical axis. Instead, its lower section is the frustum of a pyramid, while the upper portion looks a little like a squared-off paraboloid. The shape is shown isometrically in Figure 7.3 below.

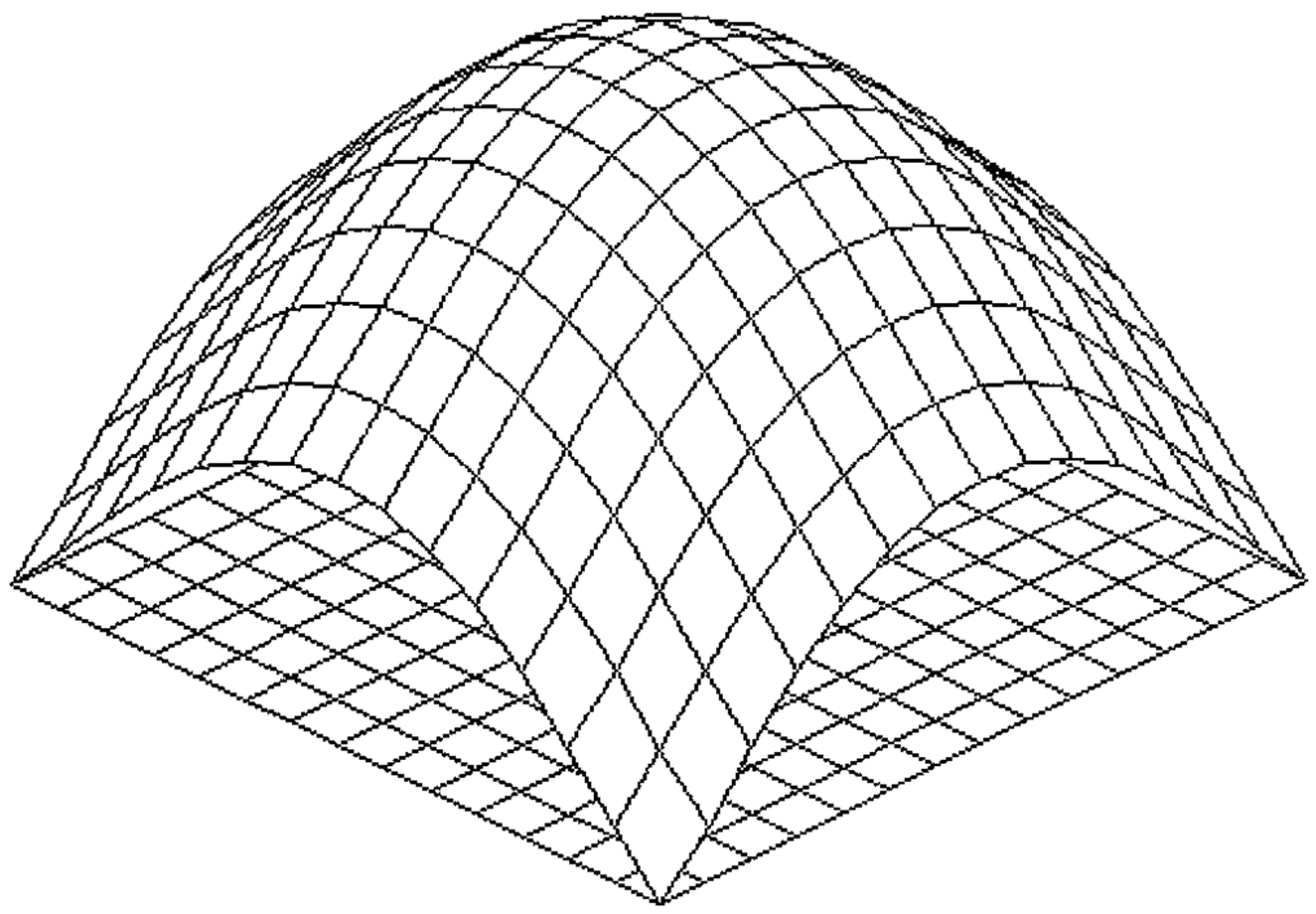


Figure 7.3. Isometric depiction of a continuous thimble

In the case of discrete models (which are the only ones which are of practical use in this problem), discrete thimbles are constructed by using only the values at the grid points. Figure 7.4 below shows two different discrete linear thimbles obtained from different positions of the continuous thimble.

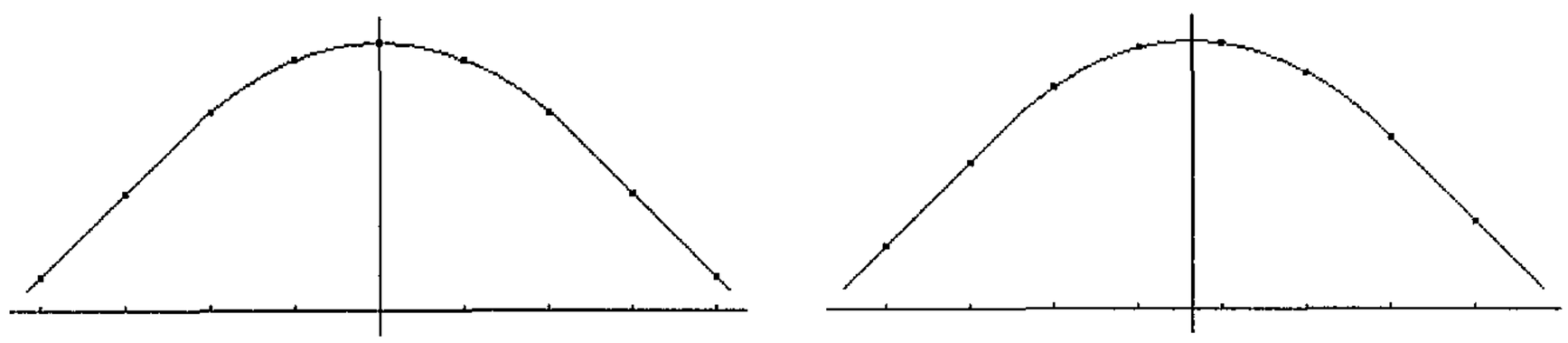


Figure 7.4. Discrete linear thimbles sampled from a continuous thimble. On the left the grid coincides with the peak, on the right it does not.

Variation can be achieved in the thimble methods by altering the way in which the thimble is placed upon a peak. It can be placed centrally on the peak or off to the side; in the latter case the thimble's apex need not be at a grid point. The figure below (7.5) shows a centrally located thimble on the left, and one which is off-centre and whose apex does not coincide with a grid point on the right.

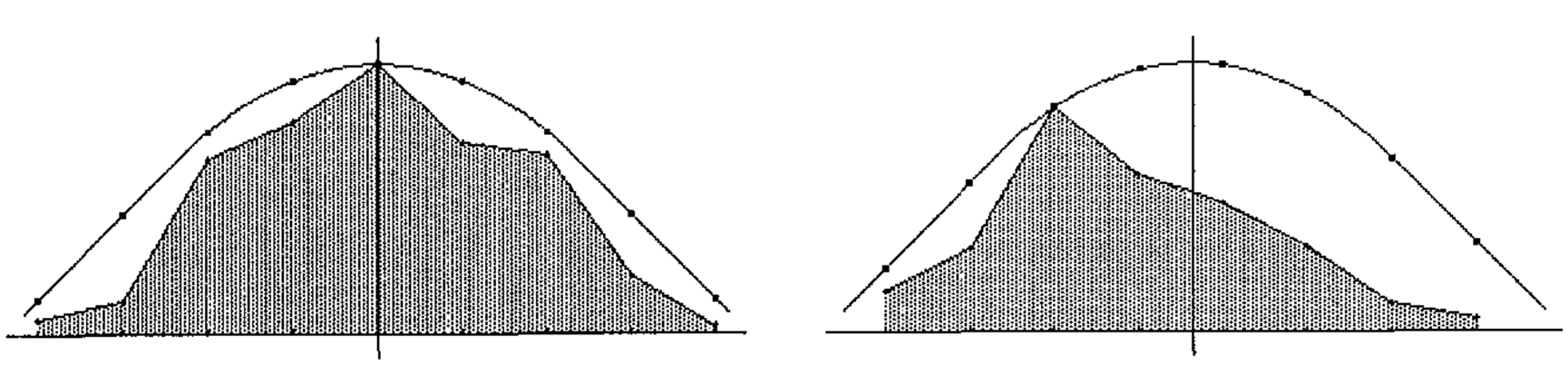


Figure 7.5. Central and off-peak placement of a thimble

These different methods of placing the thimbles form the differences between the various heuristic methods which were investigated. These are discussed from Section 7.5 onwards.

7.4 Treatment of Valleys

Once the requisite number of thimbles has been placed (according to the method selected) the drape surface may contain areas in which the valley second derivative exceeds β_1. The areas in question might either be naturally existing valleys in the topography which had not been covered by thimbles, or they might be areas between thimbles where the steep sides of the placed thimbles create a ravine at their meeting place.

The handling of the ravine-like valleys in the drape surface requires some discussion. Assume for the purposes hereof that all peak infeasibilities have been eliminated using thimbles and that there are no violated slope constraints.

7.4.1 The One-dimensional Iterative Approach

Consider an axial cut through a valley at whose lowest point the second derivative is larger than β_l. The idea is that an inverted thimble is dropped into the valley, keeping its axis vertical all the time, until it cannot go any lower. This is illustrated in the two sketches in Fig. 7.6 below.

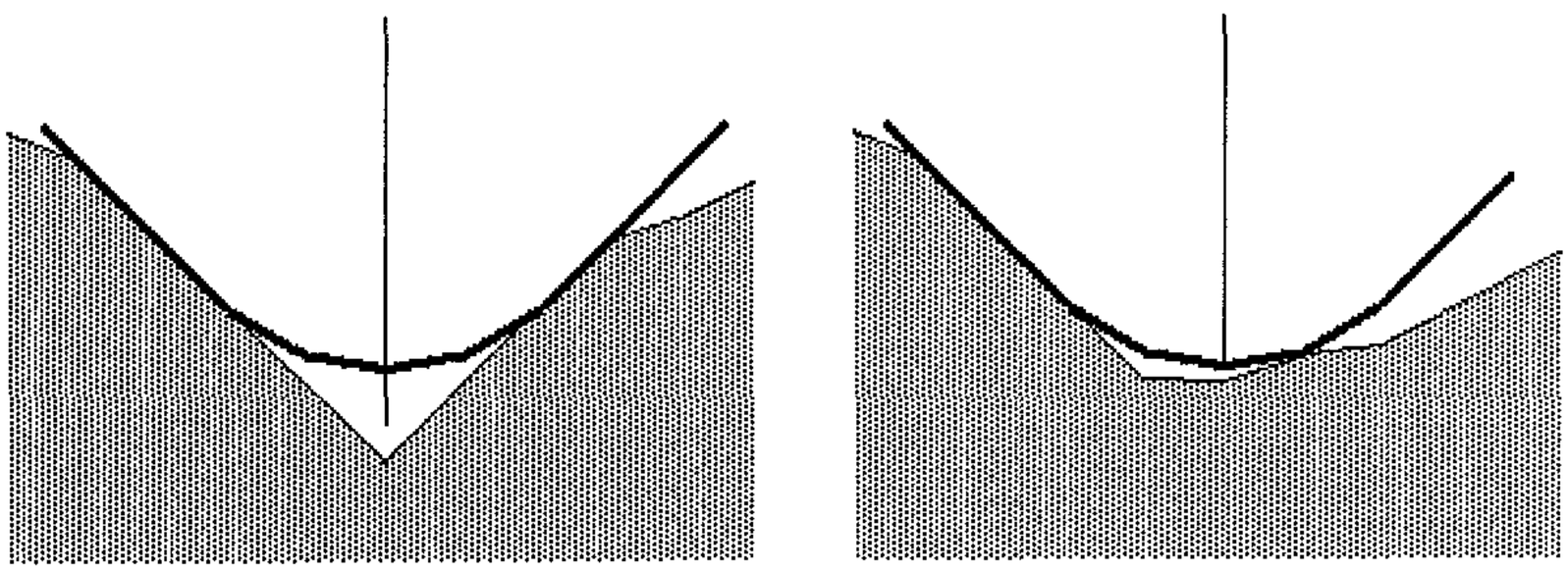


Figure 7.6. Two examples of dropping a thimble into a valley with a second derivative which is too large

The first method tried for "rounding out" the valleys consisted of sequentially finding points at which the second derivative was too high, and then raising them until the value was acceptable. Unfortunately, this method will not necessarily converge in a finite number of steps. For example, consider the simple case of five points making up a ravine whose sides have a slope $\alpha = 2.0$ and for which we need to find a drape surface with $\beta_l = 1.0$. The table below shows successive lifting operations in terms of the method described above, one iteration per row. Points at which the second derivative exceeds β_l

are shaded. The point on the drape surface which is raised to reach the next iteration is shown in bold.

Iteration	z_{-2}	z_{-1}	z_0	z_1	z_2
1	7,0	5,0	**3,0**	5,0	7,0
2	7,0	**5,0**	4,5	5,0	7,0
3	7,0	5,25	**4,5**	5,0	7,0
4	7,0	**5,25**	4,625	5,0	7,0
5	7,0	5,3125	**4,625**	5,0	7,0
∞	7,0	5,5	5,0	5,5	7,0

The table shows the interchange of infeasibility at z_{-1} and z_0 respectively, with the second derivative infeasibility at z_1 becoming steadily worse. Although the adjustments which are made gradually become smaller, the complete removal of the infeasibilities still takes an infinite number of iterations. For a machine with finite accuracy, the value at z_1 will eventually be adjusted when the second derivatives at z_{-1} and z_0 reach the value of β_l. After the adjustment of z_1 the entire process of adjusting z_{-1} and z_0 is repeated, and so on. The last row in the table shows the final feasible solution for this valley.

The process may be finite on a real machine but its use is extremely limited practically. What is required is a method which removes infeasibilities without introducing others in the process. The *chain method* achieves this.

7.4.2 The One-dimensional Chain Method

The Chain Method attempts to model the drape curve over a deep valley as if it were a chain suspended between "feasible" anchors on either side. The idea is reminiscent of the chains which are hung between vertical pillars in museums, banks and other buildings to provide a (not very effective) barrier. The chain itself consists of rigid segments linked together in such a way that the second derivative at any linkage is limited to be β_I.

The method of hanging a chain begins by identifying a point where the second derivative exceeds β_I. The two segments responsible for this infeasible second derivative are replaced by two segments forming an angle equivalent to the maximum permissible second derivative. As the chain is extended, more segments are added to this structure, also using locked joints. Adjacent segments with locked joints are treated as one unit, and only the end points of the assembly are considered.

Again consider the example given above. In the discussion which follows, x_0 refers to the grid coordinate while z_0 is the height of the chain or drape surface at x_0. We begin with the following values.

Iteration	z_{-2}	z_{-1}	z_0	z_1	z_2
1	7,0	5,0	3,0	5,0	7,0

The value z_0 is infeasible. Hence z_0 must be increased to 4,5 so that the second derivative is exactly β_I; this results in a V-shaped formation with a fixed angle. The result is identical to that of the first operation in the method of Section 7.4.1 above.

Iteration	z_{-2}	z_{-1}	z_0	z_1	z_2
2	7,0	5,0	**4,5**	5,0	7,0

As before, we now have two infeasible points. This is shown by the shading. We also have a chain of two segments centred on x_0 as shown in bold.

The infeasible arrangement at x_{-1} is now replaced with a maximum second derivative joint, and the three segments incorporating x_{-1} and x_0 together now form a chain. This chain is suspended between z_{-2} and z_1, and the following boldfaced values are obtained for the heights of the two internal points of the chain.

Iteration	z_{-2}	z_{-1}	z_0	z_1	z_2
3	7,0	**5,333**	**4,667**	5,0	7,0

Observe how this arrangement does not introduce any extra infeasibilities along the chain. The infeasibility at x_1 still remains, so the chain must be extended to the right. The chain now consists of four segments and the three boldfaced points and is suspended between z_{-2} and z_2, as shown in the final table below.

Iteration	z_{-2}	z_{-1}	z_0	z_1	z_2
4	7,0	**5,5**	**5,0**	**5,5**	7,0

The Chain Method has managed to smooth out the valley in just a few short steps.

In order to apply the Chain Method mechanically, a function is needed with which to determine the z-value of every point on the chain, given the z-values of the anchor points and the number of points in the chain. The following development (with increasing numbers of points in the chain) is instructive. In all cases, let n be the number of points in the chain, while z_0 and z_{n+1} are the heights of the anchors.

n	Heights of chain points			
1	$\frac{1}{2}(z_0+z_2)-\frac{1}{2}\beta_l$			
2	$\frac{1}{3}(2z_0+z_3)-\beta_l$		$\frac{1}{3}(z_0+2z_3)-\beta_l$	
3	$\frac{1}{4}(3z_0+z_4)-\frac{3}{2}\beta_l$	$\frac{1}{4}(2z_0+2z_4)-2\beta_l$	$\frac{1}{4}(z_0+3z_4)-\frac{3}{2}\beta_l$	
4	$\frac{1}{5}(4z_0+z_5)-2\beta_l$	$\frac{1}{5}(3z_0+2z_5)-3\beta_l$	$\frac{1}{5}(2z_0+3z_5)-3\beta_l$	$\frac{1}{5}(z_0+4z_5)-2\beta_l$

From the pattern above the general expression for the ith chain height can be inferred, where i is a counter which increases from 1 (the chain point closest to z_0) to n (the point closest to the anchor at the other end). The height of the ith point of the chain is given by

$$z_i = \frac{n+1-i}{n+1}z_0 + \frac{i}{n+1}z_{n+1} - \frac{i(n+1-i)}{2}\beta_l .$$

The first two terms in the expression above form a linear combination of z_0 and z_{n+1}, while the third term accounts for the (symmetrical) quadratic deviation from this linear chord.

The correctness of the above expression can be demonstrated by evaluating the second derivative at an arbitrary point i of the chain, as well as at either end.

At an arbitrary point i the second derivative of the chain is given by

$$
\begin{aligned}
z_{i-1} - 2z_i + z_{i+1} &= \frac{n+1-(i-1)}{n+1} z_0 + \frac{i-1}{n+1} z_{n+1} - \frac{(i-1)(n+1-(i-1))}{2} \beta_l \\
&\quad - 2\left[\frac{n+1-i}{n+1} z_0 + \frac{i}{n+1} z_{n+1} - \frac{i(n+1-i)}{2} \beta_l \right] \\
&\quad + \frac{n+1-(i+1)}{n+1} z_0 + \frac{i+1}{n+1} z_{n+1} - \frac{(i+1)(n+1-(i+1))}{2} \beta_l \\
&= \frac{n+1-i+1-2n-2+2i+n+1-i-1}{n+1} z_0 \\
&\quad + \frac{i-1-2i+i+1}{n+1} z_{n+1} \\
&\quad + \frac{-in+n+i^2-i-2i+2+2in+2i-2i^2-in-n+i^2+i}{2} \beta_l \\
&= \beta_l .
\end{aligned}
$$

In the above the coefficients of the terms for the two endpoints are zero, while the coefficient of β_l becomes one.

Considering the leftmost point of the chain, point 1, the second derivative is given by

$$
\begin{aligned}
z_0 - 2z_1 + z_2 &= z_0 - 2\left[\frac{n+1-1}{n+1} z_0 + \frac{1}{n+1} z_{n+1} - \frac{1(n+1-1)}{2} \beta_l \right] \\
&\quad + \frac{n+1-2}{n+1} z_0 + \frac{2}{n+1} z_{n+1} - \frac{2(n+1-2)}{2} \beta_l \\
&= \frac{n+1-2n+n-1}{n+1} z_0 + \frac{-2+2}{n+1} z_{n+1} + \frac{2n-2n+2}{2} \beta_l \\
&= \beta_l .
\end{aligned}
$$

For the rightmost point of the chain, point n, the second derivative is given by

$$
\begin{aligned}
z_{n-1} - 2z_n + z_{n+1} &= \frac{n+1-(n-1)}{n+1} z_0 + \frac{n-1}{n+1} z_{n+1} - \frac{(n-1)(n+1-(n-1))}{2} \beta_l \\
&\quad - 2\left[\frac{n+1-n}{n+1} z_0 + \frac{n}{n+1} z_{n+1} - \frac{n(n+1-n)}{2} \beta_l \right] \\
&\quad + \frac{n+1-(n+1)}{n+1} z_0 + \frac{n+1}{n+1} z_{n+1} - \frac{(n+1)(n+1-(n+1))}{2} \beta_l \\
&= \frac{n+1-n+1-2n-2+2n+n+1-n-1}{n+1} z_0 \\
&\quad + \frac{n-1-2n+n+1}{n+1} z_{n+1} \\
&\quad + \frac{-2n+2+2n}{2} \beta_l \\
&= \beta_l .
\end{aligned}
$$

The expressions for the second derivative at the endpoints of the chain give identical results, as was to be expected.

It goes without saying that chains always follow parabolic paths. No chain will ever have two adjacent segments with the same slope, and it is never necessary to hang a chain which extends beyond the parabolic region.

7.4.3 The Two-dimensional Chain Method

The construction of a two-dimensional chain requires some discussion. Initial attempts to generate a valley-shaped thimble considered each point of the drape surface in turn. In this case, an infeasibility in one of the axial directions would cause a point to be lifted.

An infeasibility in the other direction would then often lift its neighbours, thus causing the first infeasibility which had been removed, to reappear. Various schemes of marking the points as feasible in one or the other direction were tried, but the housekeeping became difficult, and, although no "lift-off" occurred, the process lacked elegance.

The problem of generating a two-dimensional chain to fill up between a set of given thimbles turned out to be quite simple in the end. The method applied is given below.

- On every linear cut in the one axial direction ensure that no valley infeasibilities exist by hanging chains as described above. Raise the drape surface in the valleys to coincide with the newly hung chains.

- Apply the same technique in the other axial direction, using the drape surface which was generated in the previous step. Again raise the drape surface where this lies below any newly hung chains.

- Repeat the above steps until there are no more valley infeasibilities.■

The above method was found to be both simple and quick. The correctness of the method can be demonstrated as follows.

Lemma: For a given set of thimbles, the Two-dimensional Chain Method produces the lowest possible value for every point on the drape surface.

Proof: After all necessary thimbles have been placed, the drape surface may have valley infeasibilities in the form of steep ravines or pyramidal dents. No point on the current drape surface may be lowered; on a thimble, points are at their lowest feasible height and elsewhere they are on the ground. Existing infeasibilities are to be removed by applying the Two-dimensional Chain Method.

By considering each linear section individually and by hanging chains where required, the values of all the affected points are raised. Since a chain is constructed to be at the limits of feasibility, the surface resulting from a pass in one direction is as low as possible. There are no lower feasible points and the current drape surface may be considered as valid a starting point as the ground level. When hanging chains in the orthogonal direction, the same reasoning applies. With each further hanging of chains the same reasoning still applies. This means that when feasibility is finally achieved, every point on the drape surface is as low as it can ever be. ■

Since the method described above fills out the valleys between thimbles and peaks in an optimal and unique fashion, *the drape problem reduces to the problem of finding the best placement of the thimbles*.

7.5 The Basic Thimble Method

As in most methods described so far, the Basic Thimble Method begins with a drape surface from which any excessive slopes have been removed, ie all first order infeasibilities have been handled using the Lifting Algorithm.

The simplest form of thimble method works iteratively by finding the highest peak infeasibility in the surface and covering this infeasible peak with a centrally located thimble. (As before, a peak infeasibility is a peak for which the second derivative in either axial direction exceeds β_u in magnitude.) This process continues until all infeasible peaks have been covered by thimbles. The valleys which exist between peaks and for which the second derivative exceeds β_l are then filled up using the Two-dimensional Chain Method described above.

When a thimble is placed over the current drape surface under construction, all points which are lower than the thimble are raised while those whose height already causes them to lie above the thimble are not adjusted. Hence the thimble has no further effect once it dips below the current surface. Figure 7.7 below shows how the surface is raised to the height of the thimble.

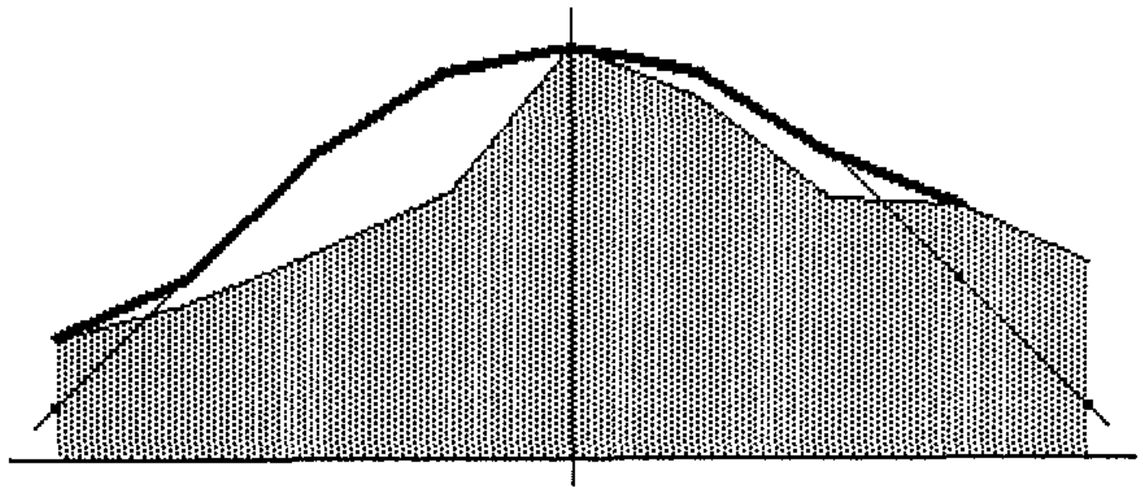


Figure 7.7. Only those points of the thimble which lie above the current surface affect the raising of the surface. The heavy line indicates the raised surface.

The mathematical implementation is as follows:

First consider a cut along an axial direction. For a thimble whose centre is at an abscissa of x_0 and which has a peak of z_0, the (continuous) thimble close to x_0 will be parabolic in shape (with a constant second derivative of $-\beta_u$) and will change to a straight line when the slope of the parabola reaches $\pm\alpha$. The table below gives the coordinates of the thimble at grid points.

***x*-coordinate**	***y*-coordinate**
x_0	z_0
$x_{\pm 1}$	$z_0 - \frac{1}{2}\beta_u$
$x_{\pm 2}$	$z_0 - 2\beta_u$
$x_{\pm 3}$	$z_0 - \frac{9}{2}\beta_u$
$x_{\pm 4}$	$z_0 - 8\beta_u$

A point at a distance of d grid points from x_0 will have a height coordinate of $z_0 - \frac{1}{2}d^2\beta_u$. The transition from the parabola to the straight line will take place when the slope between two successive points on the parabolic segment of the thimble exceeds α. If we consider points lying to the right of the centre of the thimble (which will have positive *d*), the height differential between two adjacent points, viz d and $d+1$, will be

$$[z_0 - \frac{1}{2}d^2\beta_u] - [z_0 - \frac{1}{2}(d+1)^2\beta_u] = \frac{1}{2}(2d+1)\beta_u .$$

Hence, if $\frac{1}{2}(2d+1)\beta_u > \alpha$, the thimble must have a straight edge. The transition occurs for $d = \pm(\frac{\alpha}{\beta_u} - \frac{1}{2})$. Consequently, the transition point is at $d+1 = \pm(\frac{\alpha}{\beta_u} + \frac{1}{2})$. If this expression is integral, all points from the peak to $d+1$ will lie on a parabola and all points farther than $d+1$ away from the peak will lie on a straight line. In the general case the expression is not integral, and the transition point lies between d and $d+1$. Then all points from the peak to d will be on a parabola and all points farther away from the thimble peak will be on a straight line. Bear in mind that in this case the slope between d and $d+1$ is already larger than α, hence d is the last point on the parabola. In general, therefore, the transition occurs at $\pm\left\lfloor \frac{\alpha}{\beta_u} + \frac{1}{2} \right\rfloor$, ie the largest integer smaller than or equal to the continuous transition value.

A one-dimensional thimble is described as follows, where $d_0 = \left\lfloor \frac{\alpha}{\beta_u} + \frac{1}{2} \right\rfloor$ is the transition distance from the peak.

$$z = \begin{cases} z_0 - \frac{1}{2} d_0{}^2 \beta_u - \alpha(x_0 - x - d_0) & \text{if} \quad x \le x_0 - d_0 \\ z_0 - \frac{1}{2}(x_0 - x)^2 \beta_u & \text{if} \quad x_0 - d_0 \le x \le x_0 + d_0 \\ z_0 - \frac{1}{2} d_0{}^2 \beta_u - \alpha(x - x_0 - d_0) & \text{if} \quad x \ge x_0 + d_0 \end{cases} \qquad 7.1$$

The case of the two-dimensional thimble is similar. However, when calculating the height of the thimble at a given distance from its peak it must be borne in mind that the thimble does not have circular symmetry and that the Euclidean distance from the point under consideration to the peak cannot be used as it stands. Instead, assuming that the thimble's peak is at (x_0, y_0) and that the point under consideration is at (x_i, y_j), the height of the thimble at the latter point is determined using a two-stage process, illustrated in Figure 7.8.

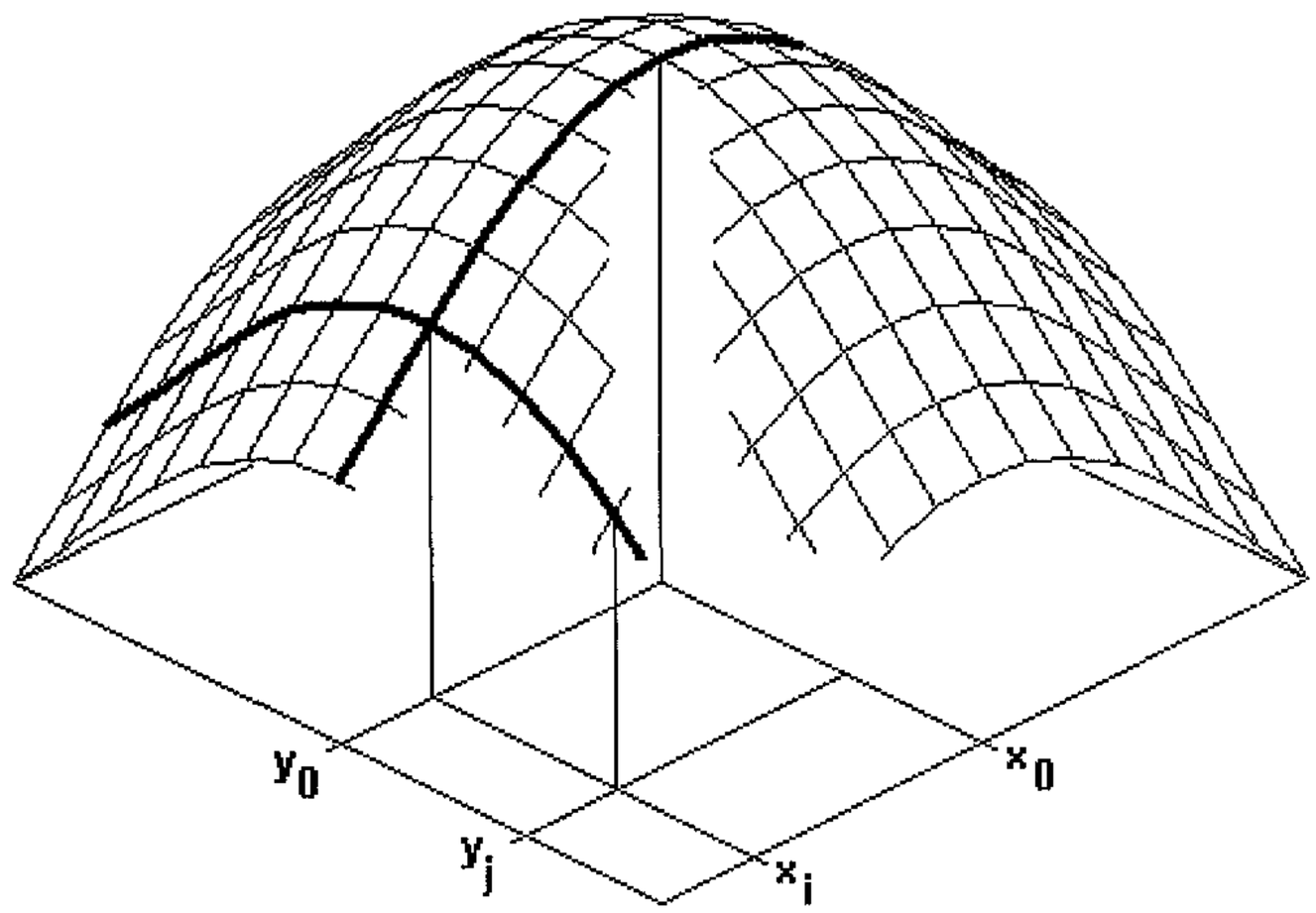


Figure 7.8. Determining the height of a point on the thimble

Using the expressions above, the height of the thimble at the point (x_i, y_0) is first determined from the thimble's peak position, (x_0, y_0), and the thimble parameters α and β_u. Once the thimble's height at (x_i, y_0) is known, a similar procedure is used on the thimble section at $x = x_i$. On this section the peak lies at (x_i, y_0) and the height to be calculated is at the point (x_i, y_j). Note that the two heavily outlined thimble sections (and for that matter any other parallel sections) all have the same shape.

The two-phase thimble height calculation above is obviously commutative. Instead of proceeding from the thimble's peak (x_0, y_0) to the section peak (x_i, y_0) to the point of interest (x_i, y_j), the calculation could equally easily have taken the path from (x_0, y_0) to the "other" intermediate peak (x_0, y_j) to (x_i, y_j).

Once enough thimbles have been placed upon the topography to eliminate any existing peak infeasibilities (ie points of the drape surface where the second derivative in either axial direction exceeds β_u), the drape surface is either feasible or it may contain areas where the second derivative along an axial cut exceeds β_l. These are valleys which are "too deep for their width". The Two-dimensional Chain Method described in section 7.4 is now used to round out the valleys.

The Basic Thimble Method

Step 1 (*Initialise drape surface*): Set the drape surface equal to the topographic surface, ie set $z_{i,j} = d_{i,j}$ for all points (i, j).

Step 2 (*Find next peak infeasibility*): Find the highest point of the drape surface for which a peak infeasibility exists. If no such point can be found, go to step 4.

Step 3 (*Place thimble*): Place a thimble over the infeasible peak found in the previous step. This will raise the drape surface as described previously in this chapter. Go back to step 2.

Step 4 (*Go to second phase*): No further peak infeasibilities exist. Go to step 5.

Step 5 (*Hang west-east chains*): Analyse all sections of the drape surface parallel to the x-axis. Hang chains where valley infeasibilities are found.

Step 6 (*Hang north-south chains*): Analyse all sections of the drape surface parallel to the y-axis. Hang chains where valley infeasibilities are found.

Step 7 (*Check for termination*): If no chains were added during the last two applications of steps 5 and 6, go to step 8. Else go back to step 5.

Step 8 (*Terminate*): A feasible drape surface has been found. Stop. ■

The table below gives some results obtained using the Basic Thimble Method. (The input datasets used in this chapter are the same as the ones used in the previous chapter.) All calculations were performed on a 233 MHz Pentium 2 with 64 MB of RAM.

Problem size	Optimum (XA)		Basic Thimble Method		% difference
	Solution	Time (s)	Solution	Time (s)	
10 × 10	21 172	< 1	21 173	< 1	0,005
25 × 25	130 148	< 1	130 162	< 1	0,011
50 × 50	498 997	6	499 246	< 1	0,050
10 × 10	21 191	< 1	21 205	< 1	0,066
10 × 10	21 224	< 1	21 255	< 1	0,146
25 × 25	130 525	2	130 769	3	0,187
52 × 52	542 995	53	552 113	1	1,697
60 × 60	729 221	479	732 181	1	0,406
75 × 125	1 827 595	69	1 829 084	1	0,081
150 × 250	7 304 665	1526	7 308 897	30	0,058

The results obtained are quite acceptable, considering the rather small difference between the optimum and the heuristic solutions, as well as the speed of the Basic Thimble Method.

7.6 The Enhanced Thimble Method

The Basic Thimble Method assumes a unique position for every thimble and no attempt is made to place the thimbles optimally over peaks.

It is obviously worth considering different positions for the thimbles. This might cause local worsening of the drape surface which might, however, act to the benefit of the entire drape surface.

It must be pointed out that the variation in the placing of thimbles changes the entire scope of the problem and reduces its complexity by orders of magnitude. In the general LP case the problem has as many variables as there are points in the grid. The modified method now becomes a problem which has vastly fewer variables, to wit only the x and y-coordinates of the centre of each thimble. Once these are known, calculating the corresponding drape surface using the methods described above is a purely mechanical operation. While the original LP had a large number of variables, the Enhanced Thimble Method has twice as many variables as there are thimbles. The variables are the offsets of the drape surface thimbles from their central position in both the x and y directions. For example, the 150 × 250 dataset requires 37 500 variables when formulated as an LP, while only 1 944 variables are needed when using thimbles.

Essentially, the Enhanced Thimble Method identifies all the thimbles which need to be placed upon the topography to generate a drape surface which has no peak infeasibilities. These thimbles are then moved around their central positions, and a new drape surface is generated for each case. The combination of thimble positions which ultimately generates the best drape surface is considered to be the optimum solution generated by this heuristic.

The Enhanced Thimble Method

Step 1 (*Initial drape surface*): Use the Lifting Algorithm to create an initial drape surface from the topography so that all slope constraint violations are eliminated. This step is not absolutely essential, but tends to decrease the overall computation time.

Step 2 (*Find next peak infeasibility*): Find the highest peak infeasibility in the drape surface. If none can be found, go to step 6. Otherwise proceed to step 3.

Step 3 (*Find objective with central thimble position*): Place the thimble centrally over the peak infeasibility. Calculate the sum of all the z-values in this drape surface, even though it may still contain infeasible areas. Retain this value as the current best value of the drape sum. Restore the drape surface to its state before the thimble was added.

Step 4 (*Move thimble around*): Now try the thimble in a square array of positions around the peak infeasibility's position. These positions should be equally spaced in both axial directions by a small search increment; the edge of the square array need be no larger than twice the maximum search distance. See Section 7.6.1 below. For each thimble position, determine the sum of all the z-values in the drape surface (even though it may contain infeasibilities) and compare it to the best value obtained so far. Remember which thimble position gave the best value. Remove the thimble before trying it in the next position.

Step 5 (*Place thimble in optimal position*): Place the thimble permanently in the best position found so far. Go back to step 2.

Step 6 (*Fix valleys*): There are no more peak infeasibilities. Round out the infeasible valleys using the Two-dimensional Chain Method described previously. This generates a completely feasible drape surface.■

Before continuing, a few words are necessary about the maximum search distance mentioned in step 4 above.

7.6.1 Calculating the maximum search distance

This section gives a more detailed description of the maximum search distance mentioned

in step 4 of the Enhanced Thimble Method.

The maximum search distance is the smallest orthogonal distance from the centre of the thimble which will cause the point at that part of the thimble to be lower than the lowest point in the drape surface. In other words, moving the maximum search distance away from the centre of the thimble guarantees the thimble to have moved below the existing drape surface.

The placing of the thimble now also requires calculation of its peak's height before the drape surface can be determined. Let the infeasible peak be at (x_p, y_p) and let the thimble's desired position be (x_t, y_t). The height of the drape surface at an arbitrary point (x_i, y_j) is now found as follows. Use formula 7.1 and the separation in the x-direction between the peak and the thimble, $|x_t - x_p|$, to determine the height of the thimble at (x_t, y_p) (see section 7.5), then apply the same method using the separation in the y-direction, ie $|y_t - y_p|$. The height at the thimble centre (x_t, y_t) is now known. Now use the axial separations $|x_t - x_i|$ and $|y_t - y_j|$ to calculate the height of the thimble at (x_t, y_t).

We observe that (x_p, y_p) coincides exactly with a grid point, ie x_p and y_p are integral. In contrast to this, (x_t, y_t) need not coincide with a grid point.

The table below gives some results obtained using the Enhanced Thimble Method, all obtained on a 233 MHz Pentium 2 with 64 MB of RAM.

Problem size	Optimum (XA)		Enhanced Thimble Method		% difference
	Solution	Time (s)	Solution	Time (s)	
10 × 10	21 172	< 1	21 172	1	0,0
25 × 25	130 148	< 1	130 149	9	0,001
50 × 50	498 997	6	499 047	209	0,010
10 × 10	21 191	< 1	21 194	2	0,014
10 × 10	21 224	< 1	21 236	3	0,057
25 × 25	130 525	2	130 563	121	0,029
52 × 52	542 995	53	550 538	1 101	1,389
60 × 60	729 221	479	729 861	1 710	0,088
75 × 125	1 827 595	69	1 827 877	3 001	0,015
150 × 250	7 304 665	1526	7 305 273	112 451	0,008

The time taken by the Enhanced Thimble Method exceeds that taken by the exact method in all cases. However, the results looked promising and therefore an effort was made to improve the speed.

7.7 The Simplified Enhanced Thimble Method

In an effort to shorten the execution time of the Enhanced Thimble Method while still allowing the thimbles to be moved around their central positions on the peaks, the Simplified Enhanced Thimble Method was devised.

In the previous method there were d^2 possible positions to be evaluated for each thimble, d being the number of search positions in both the x and y directions. For the method of this section it was decided to attempt to search in the two directions separately, ie first to find the minimum attained by varying the x position, then using this "best" x position and varying the y position to find the best overall position. Consequently the number of positions to be tried in this method is only $2d$.

The table below gives details of the results obtained using the Simplified Enhanced Thimble Method compared to the exact results obtained using the XA Library. All calculations were done on a 233 MHz Pentium 2 with 64 MB of RAM.

Problem size	**Optimum (XA)**		**Simplified Enhanced Thimble Method**		**% difference**
	Solution	**Time (s)**	**Solution**	**Time (s)**	
10 × 10	21 172	< 1	21 172	< 1	0,0
25 × 25	130 148	< 1	130 161	< 1	0,010
50 × 50	498 997	6	499 222	6	0,045
10 × 10	21 191	< 1	21 201	< 1	0,047
10 × 10	21 224	< 1	21 248	< 1	0,113
25 × 25	130 525	2	130 677	3	0,116
52 × 52	542 995	53	550 961	19	1,467
60 × 60	729 221	479	730 607	31	0,190
75 × 125	1 827 595	69	1 828 897	73	0,071
150 × 250	7 304 665	1526	7 308 401	2 520	0,051

The results obtained using the Simplified Enhanced Thimble Method are very close to the optimum in most cases although not as good as those obtained with the Enhanced Thimble Method. However, the wall clock times required to obtain these results are also of the same order of magnitude as the times needed to generate exact solutions. A comparison of the various thimble methods with other heuristics is given in the next chapter.

Chapter 8
Conclusions

8.1 Measures

The previous chapters have described a number of different methods for solving the drape problem. In this chapter the efficiency of the various methods is compared.

In comparing methods some reference is needed. In this case this is obviously the performance of an off-the-shelf method; here we use the XA Library.

The following factors were deemed important when comparing methods:

- *Speed*: The time taken to arrive at a solution is probably the single most important factor in assessing the usefulness of a method. In drape problems the time taken to produce a feasible solution is very much a function of the problem size. For the purposes of comparison, problem size is taken to be the number of points in the DTM grid.

- *Accuracy*: This factor is clearly important when evaluating heuristic methods, since accuracy is usually sacrificed in favour of speed or less lavish use of resources.

- *Resources*: The resources hogged by a computation are important when evaluating the method. It can readily occur that too much memory is required when a particular method is attempted.

The number of iterations performed in reaching the final solution was not considered as a point of comparison. For the most part, a user wishing to produce a drape surface for a

given DTM is not concerned with the number of iterations required to perform this process.

8.2 Simplified Problems

In Chapter 3 the Lifting Algorithm was developed. This was later improved upon by the Enhanced Method which was also discussed there. Both of these are exact methods for producing a drape surface for which only the first derivative is constrained. In all examples tried using these methods, the calculation was performed in seconds while an industry-standard solution procedure (the XA Library) took appreciably longer. As was explained, this difference is due to the differing complexities of the two methods (see Section 3.13). As far as all three criteria go, the Lifting Algorithm is certainly preferable to the full-blown LP method. *Accuracy* is not an issue since the Lifting Algorithm is an exact algorithm. As far as *resources* go, the Lifting Algorithm only requires storage for the grid values, the LP method of necessity needs more.

In the case of the one-dimensional drape problem which was discussed in Chapter 4, the One-dimensional Dual Algorithm which was developed there also compares quite favourably with the standard LP method. In terms of *speed* the execution times are of the same order of magnitude, with the One-dimensional Dual Algorithm always being faster. *Accuracy* is not an issue again because the method is exact. The dual algorithm has the upper hand when it comes to *resources*, though, because its memory requirements increase more or less linearly with the problem size while XA's increase polynomially (see Section 4.7).

If $\beta_l \geq 2\alpha$ and $\beta_u \geq 2\alpha$, ie if the first order derivative constraints are rather more stringent than the second order constraints, then the second order derivative constraints can be ignored and the problem of generating a drape surface can be tackled using the Lifting Algorithm. This will allow vast time savings.

The methods developed for solving reduced problems have distinct speed and resource advantages over standard LP methods.

8.3 Exact Methods

Various drape problems were solved using both the industry standard XA Library and the Dual Method for the two-dimensional drape problem. The findings were discussed in Chapter 5. In terms of the criteria listed above, the following comments can be made:

- *Speed*: For small problems XA and the Dual Method are comparable. As the input grid becomes larger, say above 3 000 points, the Dual Method performs badly when compared to XA. A grid of some 4 000 points took over 39 hours to run, the same grid was solved by XA in under four minutes.

- *Accuracy*: Other than small rounding errors which would be expected when the same problem is solved using different methods, the results obtained using the two methods were the same. This was to be expected.

- *Memory*: For very small problems the Dual Method uses less memory than XA. As the problem size increases, however, XA uses less memory than the Dual Method.

It must be borne in mind that the speed of reaching the solution as well as the memory usage of the Dual Method are dependent on the actual grid data. Rough terrain in the input grid can require more iterations before a feasible solution is reached.

Conclusion: The Dual Method produces usable results but requires a faster machine and more memory than XA. It is thus not of practical use.

8.4 Heuristic Methods

Most of the heuristic methods developed and discussed in Chapters 6 and 7 had good speed properties and, in general, the accuracy obtained was quite acceptable.

The memory requirements for the heuristic methods were very modest, often not much more than the memory required to store one or two grids of the DTM grid size. This left speed and accuracy as the most important factors to be considered when comparing methods.

The same input datasets were used in all evaluations of heuristic methods.

The table below summarises the results obtained when various methods were applied to the 150 × 250 problem, the largest test dataset so far.

Method	**Time (s)**	**% difference**
Smoothing Method (6.2)	< 1	2,108
General Heuristic 5 (6.3.5)	172	0,016
Alternating One-dimensional Dual Algorithm (6.4)	6	0,006
Basic Thimble Method (7.5)	30	0,058
Enhanced Thimble Method (7.6)	112 451	0,008
Simplified Enhanced Thimble Method (7.7)	2 520	0,051

In the table above all times are given in seconds. The exact method using XA reaches the optimum in 1 526 seconds. The percentage difference in the table is the deviation of the obtained solution from the optimum. The graph below shows a plot of accuracy versus execution time for this particular problem.

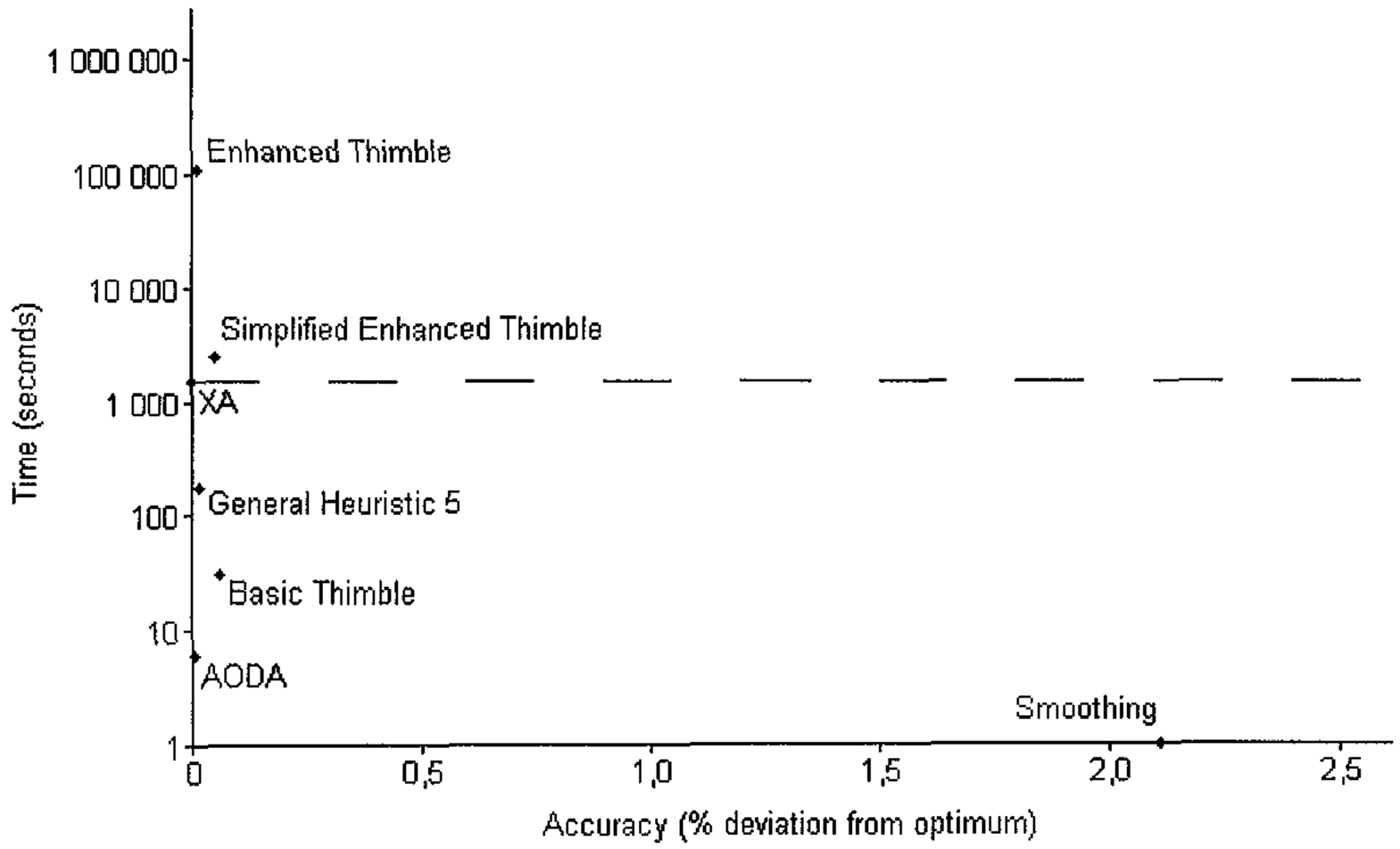


Figure 8.1. Plot of execution time versus accuracy for various heuristic methods for the 150 × 250 problem

The scale of the time axis is logarithmic to accommodate the wide range of values. The graph also shows the performance of XA. Any points lying above the dotted line correspond to heuristic methods whose execution speed is slower than that of the exact method. Clearly, such methods have no practical use and can be ignored.

A further comparison can be made by comparing an average of the execution times for various methods for the five largest standard datasets, as well as the percentage deviation from the optimum. The table below lists these results.

Method	**Time (s)**	**% difference**
Smoothing Method (6.2)	< 1	1,643
General Heuristic 5 (6.3.5)	36	0,442
Alternating One-dimensional Dual Algorithm (6.4)	3	0,318

Method	Time (s)	% difference
Basic Thimble Method (7.5)	7	0,486
Enhanced Thimble Method (7.6)	23 677	0,306
Simplified Enhanced Thimble Method (7.7)	529	0,379

The exact solutions were obtained by XA in an average of 426 seconds. The graph below shows a plot of accuracy versus execution time for the five problems.

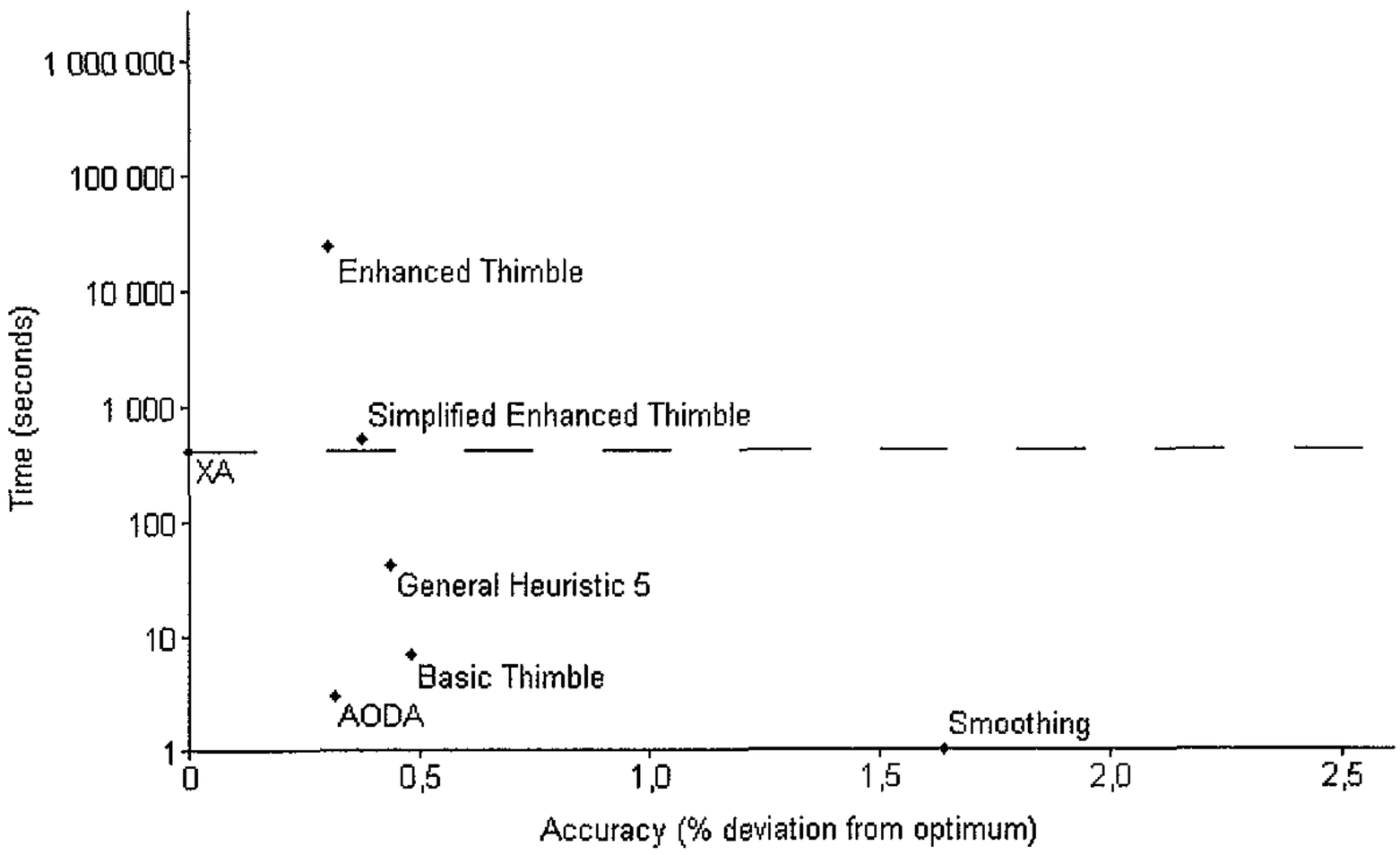


Figure 8.2. Plot of execution time versus accuracy for various heuristic methods averaged for five datasets

In order to make a solid comparison between the various viable methods, a further set of several large problems was generated. The results obtained are given in the table below, together with averages for the various columns.

Problem	XA	Smoothing		Gen Heur 5		AODA	
	Time (s)	Time (s)	% diff	Time (s)	% diff	Time (s)	% diff
300×100a	3 738	< 1	1,033	106	0,068	13	0,026
300×100b	3 024	<1	1,075	111	0,050	9	0,016
300×100c	4 123	1	1,013	106	0,096	15	0,025
150×250a	10 348	< 1	0,592	178	0,017	71	0,024
150×250b	20 699	< 1	0,439	180	0,034	175	0,059
250×150a	9 113	< 1	0,747	178	0,042	27	0,027
250×150b	10 947	< 1	0,677	177	0,065	49	0,034
150×150a	569	< 1	0,823	51	0,007	2	0,001
150×150b	293	< 1	0,671	55	0,002	1	0,001
150×150c	386	< 1	0,883	49	0,003	1	0,001
200×200a	1 258	< 1	0,976	207	0,005	3	0,001
200×200b	1 353	< 1	0,819	211	0,005	4	0,001
Average	**5 488**	**1**	**0,812**	**134**	**0,033**	**31**	**0,018**

Problem	Basic Thimble		Simplified Enh TM	
	Time (s)	% diff	Time (s)	% diff
300×100a	43	0,170	2 459	0,096
300×100b	40	0,201	2 361	0,122
300×100c	36	0,207	2 065	0,109
150×250a	102	0,047	7 071	0,057
150×250b	83	0,102	5 570	0,119
250×150a	133	0,085	9 063	0,077
250×150b	116	0,125	8 329	0,112
150×150a	9	0,040	660	0,034
150×150b	7	0,013	394	0,012
150×150c	8	0,024	502	0,021
200×200a	22	0,025	1 880	0,022
200×200b	23	0,029	1 968	0,027
Average	**52**	**0,089**	**3 527**	**0,067**

The above results are shown diagrammatically in the graph below.

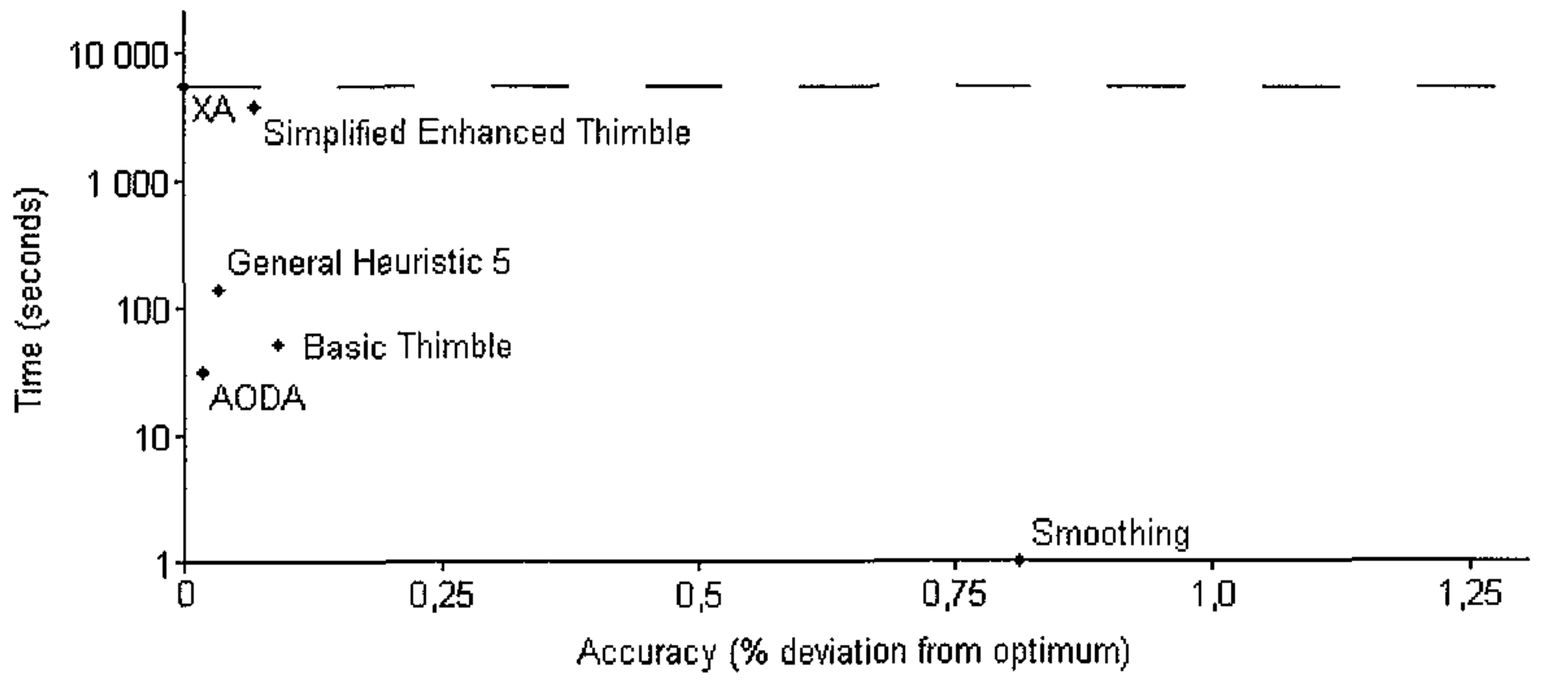


Figure 8.3. Plot of execution time versus accuracy for various viable heuristic methods averaged for twelve large datasets

The heuristic method to be used for generating a drape surface is now a matter of personal choice. However, the Alternating One-dimensional Dual Algorithm (AODA) is faster and more accurate than the Basic Thimble Method and General Heuristic 5. The point on the plot for the AODA lies to the left of and lower than the points indicating these other methods on all graphs.

The choice of method thus falls between the AODA and the Smoothing Method. Of these, the former is more accurate while the latter is faster.

8.5 Final Remarks

Before making a choice of method the user should consider the implications of accuracy. The practical impossibility of a pilot adhering to a predefined flight path (in both the vertical and horizontal directions) has been mentioned before. In the light of this observation it would not appear to be worthwhile to apply an accurate algorithm, given that the time for solving a problem is likely to be quite prohibitive.

In practical cases, therefore, a choice of heuristic should be made, preferably either the AODA or the Smoothing Method. Where the speedy generation of drape surfaces is required, the Smoothing Method might be preferable. A case where speed might be necessary is simulation software which could illustrate to a prospective client how a drape surface might vary with aircraft performance. In this example a large number of differing drape surfaces could be required, and they would have to be generated as quickly as possible. On the other hand, if the user applies the AODA he can rest secure in the knowledge that he probably has as good a drape surface as he will ever need in practice.

8.6 Further Research

Although it is felt that the work contained in this document represents a coherent approach to the problem of drape surfaces, avenues for further research can always be found. Other heuristic techniques might be possible, perhaps using simulated annealing or genetic algorithms. Further work on drape surfaces might comprise modelling the performance of an aircraft more carefully by taking into account varying temperature and pressure as well as the aircraft's steadily reducing mass as the fuel is consumed. Clearly, such models would require linking specific sections of the survey to particular times of the day as well as introducing extra requirements regarding the payload. The current work has attempted to model the survey conditions without imposing almost unforeseeable restrictions.

References

AMS Systems Pte Ltd, 2001, *Laboratory Testing Equipment* [Online]. Available: http://www.amshq.bizland.com/TEX_JH_laboratory.htm [Accessed 30 March 2002].

Bardinet, E., Cohen, L.D. and Ayache, N., 1995, *A parametric deformable model to fit unstructured 3D data* [Online]. Available: http://citeseer.nj.nec.com/bardinet95parametric.html [Accessed 30 March 2002].

Best, M.E., Abercrombie, H.J. and Peirce, J.W., 1999, *Interpreted Faulting Patterns in Northeast Alberta using High Resolution Aeromagnetic Data* [Online], Can. J. Expl. Geoph. Available: http://www.gedco.com/papers/mb-athabasca.htm [Accessed 30 March 2002].

Bryant, J., Coffin, B., Ferguson, S. and Imray, M., 1997, *Horizontal and Vertical Guidance for Airborne Geophysic*s [Online], Can. Aeronautics and Space J., **43**, no. 2, 126-134, Available: http://www.casi.ca/97abst2.htm [Accessed 30 March 2002].

Chvátal, Vašek (1983). *Linear Programming*. New York: W.H. Freeman and Company.

Cohen, L.D. and Cohen, I., 1991, *Finite Element Methods for Active Contour Models and Balloons for 2D and 3D Images* [Online]. Available: http://citeseer.nj.nec.com/cohen91finite.html [Accessed 30 March 2002].

Cohen, L.D., 1996, *Auxiliary Variables and Two-step Iterative Algorithms in Computer Vision Problems* [Online]. Available: http://citeseer.nj.nec.com/cohen96auxiliary.html [Accessed 30 March 2002].

De Bruin, P.W., Vos, F.M., Frisken-Gibson, S.F., Post, F.H. and Vossepoel, A.M., 2001, *Improving Mesh Quality of Extracted Surfaces using SurfaceNets* [Online]. Available:

http://www.cg.its.tudelft.nl/wwwsv/publications/debruin2000a.pdf [Accessed 30 March 2002].

Dik, J., n.d., *Marching cubes* [Online]. Available: http://www.students.cs.uu.nl/people/jedik/Methods/Surface_fitting/Marching_cubes.htm [Accessed 30 March 2002].

Geosoft Inc., 2000, Tough Terrain – Tested Techniques, *Geosoft News* [Online], 4 – 5. Available: http://www.geosoft.com/support/newsletters/pdfs/00_1_news&views.pdf [Accessed 30 March 2002].

Govindaraj, M. and Vallamshettla, A., 1999, *Physically Based Fabric Drape Models for Computer-Aided Design*, National Textile Center Annual Report: November 1999, I98-P02.

Govindaraj, M. and Vallamshettla, A., n.d., *Physically Based Fabric Drape Models for Computer-Aided Design* [Online]. Available: http://www.ntcresearch.org/current/year9/I98-P02.htm [Accessed 30 March 2002].

Jakab, M., 1996, *Refinement Using Deformable Contour Models* [Online], Available: http://www.spl.harvard.edu:8000/pages/papers/mri_seg_tina/node9.html [Accessed 30 March 2002].

Kahaner, D.K., 1999, *ATIP96.069: Australian Engineering Mathematics Conference (AEMC'96)* [Online]. Available: http://www.cs.arizona.edu/japan/www/atip/public/atip.reports.96/atip96.069r.html [Accessed 30 March 2002].

Kennelly, P.J., 2000, *Three Dimensional Representations of Aeromagnetic and Isostatic Residual Gravity Surfaces with Geology in Montana* [Online]. Available: http://www.mbmg.mtech.edu/pdf/gis-dmtposter.pdf [Accessed 30 March 2002].

Lahey, T.J., 1998, *The Drape Coefficient: A method for characterising draping behaviour* [Online]. Available: http://www.cgl.uwaterloo.ca/~tjlahey/cgl_talks/oct26_98/drapeCoeff.html [Accessed 30 March 2002].

Peirce, J.W., Goussev, S., McLean, R. and Marshall, M., 1999, *Aeromagnetic interpretation of the Dianongo Trough HRAM survey, onshore Gabon*, CSEG Conference 1999 [Online]. Available: http://www.cseg.org/events/meetings/abstracts/1999/ps_ch11.pdf [Accessed 30 March 2002].

Miles, W., Pilkington, M. and Dumont, R., 2000, *Levelling Aeromagnetic surveys to a Common Datum*, Canadian Geological Survey [Online]. Available: http://gdcinfo.agg.emr.ca/gdc/levelling/index_e.html [Accessed 30 March 2002].

Pilkington, M. and Thurston, J.B., 2001, *Draping Corrections for Aeromagnetic Data: Line- versus Grid-based Approaches*, Exploration Geophysics, **32**, 95-101.

Sander, L., 1997, *Pre-planned drape surfaces: a new planning tool* [Online]. Available: http://www.sgl.com/News/LUTALK-4.html [Accessed 30 March 2002].

Schlumberger, 2002, *Oilfield Glossary* [Online]. Available: http://www.glossary.oilfield.slb.com [Accessed 30 March 2002].

Shodor Education Foundation, Computational Science Education Reference Desk, 2002, Numerical Derivatives [Online]. Available: http://www.shodor.org/cserd/Resources/Algorithms/NumericalDifferentiation/index.php [Accessed 4 January 2003].

Society of Exploration Geophysicists, 2000, *Gravity and Magnetics Terms for the 4th Edition of the SEG encyclopedic Dictionary* [Online]. Available: http://www.seg.org/comm-info/grav_mag/gm_dict.html [Accessed 30 March 2002].

Sunset Software Technology, 2001, *XA Optimization Library manual*. San Marino, CA, USA: Sunset Software Technology.

United States Geological Survey, 2000, *Colorado Aeromagnetic Data Processing* [Online]. Available: http://geology.cr.usgs.gov/pub/open-file-reports/ofr-00-0042/colo_proc.htm [Accessed 30 March 2002].

You, L., Zhang, J.J. and Comninos, P., 1999, *Cloth Deformation Modelling Using a Plate Bending Model*, http://citeseer.nj.nec.com/cache/papers/cs/17976/http:zSzzSzwscg.zcu.czzSzwscg99zSzpaperszSzG19.final.pdf/cloth-deformation-modelling-using.pdf, WSCG'99 Conference, Feb 11. 1999; 7th International Conference in Central Europe on Computer Graphics, 9-12 Feb 1999, Plzen, Czech Republic.

Ward, M., 1999, *Overview of the Marching Cubes Algorithm* [Online]. Available: http://www.cs.wpi.edu/~matt/courses/cs563/talks/march_cub.html [Accessed 30 March 2002].

Zhang, L., 2001, *Active Contour Model, Snake* [Online], Available: http://www.cs.unr.edu/~lzhang/snake/snake.doc [Accessed 30 March 2002].