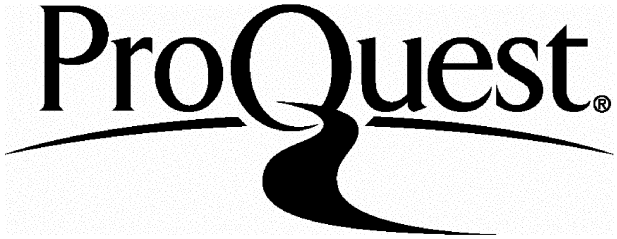
Evaluation of the Fractional Brownian Motion Model of Terrain

David Baxter Treharne Rees

A Thesis
Submitted for the Degree of Doctor of Philosophy
of the University of London

Department of Computer Science
University College London
Gower Street, London WC1E 6BT



Abstract

The goal of this research is to evaluate a commonly applied statistically self-similar model of two-dimensional height fields. Fractional Brownian motion was chosen as such a model, due to the large body of literature supporting its application as an analysis tool and a terrain simulation technique. The measurement process used - variogram analysis - has also inspired a significant body of research detailing its application.

Previous studies have, almost exclusively, used the model in the context of entire datasets. However, the serious application of this technique demands the analysis of local areas over large regions, in order to determine variations in the behaviour of different surfaces. Accordingly, one aspect of the research concentrated on automating analysis, and attaining realistic processing times to make such a study possible.

The constraints that determine fractional Brownian motion behaviour have been tested. The datasets analysed indicate such behaviour to be a poor model of terrain. Thus, doubt is cast upon its application to determine surface roughness, and upon its general use in the characterisation and generation of terrain. However, terrain is shown to be scaling, implying that a valid fractal model of terrain behaviour does exist.

Results indicate that the variogram technique is not ideally suited to determining the fractal dimension of terrain. Such analysis appears vulnerable to periodicity and other behaviour which may occur within the data. Conversely, this behaviour has highlighted poor data and, if errors can be discriminated from natural features, might be used as a quality assessment tool.

Contents

List of Tables

List of Figures

Dedication

To the memory of my mother, SHIRLEY REES

Acknowledgements

I would like to express my appreciation to some of the many people who helped me during the course of this study.

Professor J-P. Muller, my supervisor, for introducing me to techniques by which terrain fractal dimension could be quantified. His knowledge of remote sensing techniques and data availablity have helped immensely during the course of this study, while his constructive criticisms of the thesis made the presentation far more intelligible.

Dr D. Long, for focussing the latter part of the research on those aspects required to confirm the conclusions, and for proof reading of the manuscript. The structure and arguments of the thesis are far more solidly constructed than they might have been as a result of his observations, while the study was brought to a sound conclusion as a result of his efforts.

Dr. G. Schreier, Dr. R. Winter and Dr. W. Markwitz, of DLR Oberpfaffenhofen, have been of great help, in allowing access to the West German and Alpine datasets, and providing the facilities to process the data. I would also like to thank H. Craubner, D. Kosmann and A. Roth for their assistance in processing the data, and everyone, including Paramananda, for my indoctrination into German society!

Dr. P. Otto, for instructing me in the art of optimising image processing programs, and for his essential aid in the comprehension of the various models and statistical descriptions. His mathematical and computer science background were of tremendous help during the early course of the study.

Dr. R. Winder and Mr C. Easteal, for steering the thesis through the trials of administration and submission.

Dr. G. Wilkinson, who invited me to UCL, introduced me to the concept of measurement of fractal dimension of images, and helped formulate my first serious research application.

D. Allison and A. Newton, for applying their specialist knowledge to constructive criticism of the work. To those in Computer Science and Photogrammetry, especially Gordon, for creating an atmosphere in which to live as well as work.

The Science and Engineering Research Council, for their support via a quota award for

this research.

My father and mother, John and Shirley Rees, for supporting me during this period, and my brothers, Paul and Charles, for putting up with a "student" for all this time. To Carolyn, for helping me bear up under the pressure. To John and Jan for intensly serious discussions. Aleks and Hazel, Jenny, Judy and Nigel, and Penny, for providing moral support. Aleks bears the "responsibility" for starting me down the long road leading to this thesis.

Chapter 1

Introduction

1.1 Motivation

The complex behaviour exhibited by terrain differs markedly from the Euclidean primitives (points, lines, boxes, spheres) which underpin common geometric descriptions. Fractal geometry has been cited, within, for example, [Mandelbrot 82], as a means of encapsulating this complexity with very few parameters. The principal advantage of a fractal model lies in the scaling characteristics of the phenomena. When modelling terrain, a common point within the associated fractal descriptions is that some of the parameters remain invariant over a range of scales. In remote sensing applications, a variety of data are available at different resolutions. An analysis tool which can produce equivalent results at different scales would be useful in this context.

One example of a fractal model is fractional Brownian motion (fBm), which is a widely used model of fractal terrain behaviour, serving as the basis of many terrain generation models (for example, [Fournier 82], [Mandelbrot 82], [Saupe 88]), as well as a tool used in studies of terrain behaviour, such as [Culling 87], [Goodchild 82], [Huang 89], and [Mark 84].

Such a model could serve many applications. The measurement and interpolation of terrain regions would allow data compression of height data to be carried out to a specified accuracy, as outlined in [Muller 90a]. The scaling nature of the statistics could be used as the source to a kriging function, detailed by [Muller 90b]. Finally, the association of fractal dimension with surface roughness (in, for example, [Mandelbrot 82], [Pentland 84], and [Voss 88]) could meet some of the requirements of climatic modelling specified by [Rowntree 86].

Previous studies have been unable to fulfill the promise of the model, due to the laborious task of determing the scale ranges within which fractal dimension exhibits distinct behaviour. Recently, a method for the automation of fractal dimension calculation, given in [Yokoya 89], has made feasible the measurement of multiple fractal dimension values over large samples of data. Thus, the evaluation of this technique, in terms of the time taken, behaviour of the model, and usefulness of the results, is essential before fractal dimension quantification is considered a reliable tool for terrain analysis.

1.2 Organisation of the Thesis

The body of the thesis is divided into six main sections, excluding the introduction. A multitude of fractal models exist to describe terrain. Rather than implement them all, or consume study time creating a new model, the second chapter deals with a survey of current techniques, together with a comparison of their results. The models are not all derived from equivalent scaling behaviour, and thus would return non-comparable fractal dimensions from the same datum. Therefore, this section details the basic behaviour from which fractal dimensions are calculated within these models, allowing the studies to be grouped in terms of comparable results.

Techniques had to be found by which distinct scaling behaviour could be detected, and the results of the model could be evaluated. Chapter 3 details this task, surveying the various scaling detection algorithms, and relating the criteria which the selected method satisfied. The fractal model chosen earlier, fractional Brownian motion, had been determined to characterize terrain behaviour based on visual comparisons given in [Mandelbrot 75]. For such a model to be accepted as the basis for data processing techniques, the true validity of its application to terrain should be determined. To achieve this aim, independent linearity measures, together with distribution fit techniques, were employed. These analysis tools are also described in chapter 3.

Initial implementations of the fractal dimension measurement algorithm were far too expensive in computational resources. Testing of the implementation, together with analysis of a wide range of terrain data, would have required longer than the envisaged duration of the study to complete. Therefore, a programme of algorithmic and implementation improve-

ments was undertaken. These stages are detailed in the fourth chapter.

A wide variety of results were obtained from the different datasets. Within chapter 5, these results are presented. From them, the general performance of the linear segmentation method was evaluated, together with the behaviour of the fractal model over large terrain samples. The behaviour of the method was also determined in the context of grids of tiles covering the data, each tile being treated as an independent sample area. Significantly, it was observed that certain regions exhibiting high fractal dimension results were associated with areas of poor terrain representation. In addition, fractal dimension values were found to exceed the expected range of values given by the model. Such extreme ranges were confirmed by surveying the results of other studies.

These results cast doubt on both the fractal model and the analysis technique. Accordingly, the assumptions upon which the model had been based were tested. The model relies on the assumption that a Gaussian distribution should be observed, but tests showed that most of the data examined did not exhibit this characteristic. To confirm this, further tests were carried out, demonstrating that the poor result could not be completely attributed to sampling methods, nor to the terrain representation.

Moreover, the association of increasing fractal dimension with increasing surface roughness proved untrue. Visual and numerical correlation analysis revealed smooth regions exhibiting high fractal dimensions, and vice-versa. Comparisons of the results from several smaller studies showed that visually smooth terrain could not be distinguished from arbitrarily selected terrain on the basis of fractal dimension alone. An investigation of the extreme results allowed possible causes of observed behaviour to be deduced. Furthermore, experimentally generated terrain allowed certain extreme results to be created to order. Analysis of the anomalous behaviour of the model is described in chapter 6.

The concluding chapter summarises the results of the study, using them to justify certain postulates about the fractional Brownian motion model of terrain. Terrain is observed to be scaling, and thus to fit the general notion of fractal behaviour. However, observations given above are at odds with the Brownian model. This casts doubt on the use of the model for terrain interpolation and surface roughness estimation. Certain observations do lead to an alternative application, namely the automated detection and classification of poor quality

data. The future development of such an application is outlined in the final section.

Chapter 2

Determining the Fractal Dimension of Terrain

2.1 Introduction

It was determined that the study should be kept manageable by restricting it, at least initially, to one method of computation of the fractal dimension of a set. To obtain the most appropriate candidate, a survey of a wide spectrum of techniques was undertaken. A large number of measurement techniques are possible using fractal geometry (see [Falconer 90] for examples). By limiting the survey to those models already applied to terrain analysis, the number of techniques was reduced to a manageable quantity. Moreover, these prior applications allowed the behaviour of the models to be compared, sparing the lengthy time required to implement each one.

The techniques identified were found to return different fractal dimensions on certain data sets. This prompted inspection of the models from which the techniques were derived. Therefore, the techniques were analysed and categorised according to the models they embody. The appropriateness of these models to terrain modelling was also considered.

Note that all the models described below share a common point - none of them were derived from known terrain behaviour. In all cases, they originated from pure mathematical theory or models which had originally been applied to different phenomena. With at least some of the models the goal appeared to be to provide a universal description that could be applied to all terrain surfaces. Thus, many prior studies have concentrated on measuring a single fractal dimension value for an entire area, or a whole planetary surface. Regions of

distinct terrain behaviour are anathema to the purpose of these models.

In the current study the contrary view is taken, and the goal is adopted of providing the technology to describe the variability of terrain in terms of scaling systems and thus to classify them by this criterion. By searching for regional behaviour, and examining its link with known terrain measurement quantities, the direction of this research differs from that implied in some of the papers surveyed below, which examine single regions. The advantage of a large literature survey is that these single region results can be compared and contrasted with each other, contributing to the desired examination of regional behaviour.

2.2 Models of Scaling Systems

2.2.1 Scaling of a Euclidean Primitive With Changing Resolution

A *Euclidean primitive* (for example, a line segment) is chosen as a *metric* (the unit of measurement) of a set. This model is based on the observation that as the metric of a set is scaled, the number of primitives that cover the set will alter as a power law of the size of the primitive. Fractal dimension is characterized by

$$N(L) \propto length(L)^{-D}$$

given

$$N(L) = \{min|\ L_s\ |: \forall L_{inst} \in L_s\ ,\ L_{inst}\ is\ an\ instance\ of\ L\ and\ \bigcup L_s \supseteq S\}$$

where L is the Euclidean primitive, $length(L)$ represents the length of a side of the Euclidean primitive, S is the set to be covered, and D is the fractal dimension - the scaling exponent of the set in Euclidean space ($\Re^E, E >= D >= 1$). The application of this method is described in [Voss 88].

The principal advantage of the model lies in its simplicity. Applying the model description to non-fractal geometrical objects produces the expected behaviour. For example, the number of box primitives covering an area will alter at a rate inversely proportional to the square of the edge of the box, i.e. it has a measured dimension of two. A more complex object (such as a deformed surface, but not a volume) may require a more rapid increase in the number of covering boxes with decreasing box size. However, this increase may be less

than the cube of the edge of the box - a fractional result. This links the notion of spatial occupancy with the fractional dimension result.

The model can be applied to sets whose behaviour is known to be *self-similar*. This term literally describes a set as being composed of copies of itself that have been scaled. Such sets exhibit isotropic behaviour, in that the orientation of the metric primitive should have no bearing on the resulting fractal behaviour. In the case of terrain, the behaviour observed is likely to be *statistically self-similar*, since the set is composed of statistically similar rather than exact scaled copies of itself.

2.2.2 Mass Dimension

This model shares some common features with that described above. It involves the scaling of a Euclidean primitive (a square, circle, line or other). Whereas the previous model relies on a covering of a set by such primitives, mass dimension uses this covering to partition the set. Then, if a member of the set exceeds some threshold value within a partition, the partition is said to be *occupied*. Fractal dimension is then calculated by use of the relationship

$$N(L) \propto length(L)^{-D}$$

given

$$N(L) = \sum_{\forall L} T(L)$$

and

$$T(L) = \begin{cases} 1 & \text{if } l \in L,\ l \geq t \\ 0 & \text{otherwise} \end{cases}$$

where L is the Euclidean primitive, $length(L)$ represents the length of a side of the Euclidean primitive, l is a single data sample within L, and t is the threshold data value. Thus, the fractal dimension determines how the number of covering *occupied* primitives varies with the size of the primitives.

Members of L for which the function T is non-zero belong to an *exceedance set*. Multiple threshold values exist for many instances of the function T, resulting in multiple fractal dimensions being found from studies of the same dataset. The term *multifractal* has been coined for this behaviour. The model was originally applied to the properties of radar rain

reflectivities, and weather station location distributions. Applications are extensively detailed in [Gabriel 88] , [Lovejoy 85], [Lovejoy 87], [Lovejoy 89], [Lovejoy 90a], [Lovejoy 90b], [Sarma 90], [Schertzer 87], [Schertzer 89], [Schertzer 91], and [Wilson 91].

Mass dimensions can be usefully applied where *self-affine* behaviour is observed. As with self-similar behaviour, the set under scrutiny appears to be composed of copies of itself. However, self-affinity differs in that the orientation of the metric primitive might produce differing results. Using terrain as an example, similar behaviour of the surface parallel to sea level may be observed at a wide variety of scales. This contrasts with height variability, which increases rapidly until the kilometre scale is reached. Beyond this point, increasing horizontal scales produces little change in height amplitude. By using a thresholding function applied to height, mass dimension avoids such self-affine behaviour, treating the terrain as a set of filled contour areas.

Note that multifractals have been applied to a wide variety of set descriptions, the common point being that multiple fractal dimensions result. However, of all the studies encountered during this survey, only the model given above has been applied to terrain description, and thus this is the only multifractal model considered in this study.

2.2.3 Area/Perimeter Relationships

This model bears a close relationship to the first model suggested in this chapter. However, instead of covering a set with primitives, and measuring how the number of primitives alters with scale, this model compares the scaling of numbers of one primitive with another. In this case, the comparison is between an areal covering and a linear covering.

A scaling relationship holds in that

$$A \propto P^{2/D}$$

where A is the estimated area of the set, and P is the estimated perimeter bounding the area. This method is detailed in [Lovejoy 82], where it was first applied to cloud perimeters.

This model may be used to avoid self-affine behaviour using the example method given above for mass dimension models, but incorporates boundary scaling into the model, in addition to the area scaling used by mass dimension in this application.

2.2.4 Fractional Brownian Motion

All the models detailed above avoid the consideration of self-affine behaviour by limiting their scope to describing the self-similiar aspects of the phenomena, or by using a function to transform the self-affine behaviour to self-similar behaviour. By contrast, self-affinity is inherent in the fractional Brownian motion model, which allows the result of a function to scale independently of the coordinates upon which the function is based. In the case of terrain, the height amplitude may scale independently of the horizontal spacing.

The application of fractional Brownian motion to terrain originated from [Mandelbrot 75]. The method is concerned with the fundamental expectation that for a true Brownian fractal function, the variations in the dataset ΔF for samples at absolute distance Δx apart will be normally-distributed. Moreover, the moments of this distribution will vary as a power law of Δx.

Formally, a function $F(x)$ is a fractal if, for all x and Δx:

$$Pr[\mid \Delta F_{\Delta x} \mid / \parallel \Delta x \parallel^{-H} < y] = erf(y) \tag{2.1}$$

where Pr denotes probability, H is the dimension of the Brown zeroset (which is related to the fractal dimension), and $\mid \Delta F_{\Delta x} \mid$ denotes $\mid F(x + \Delta x) - F(x) \mid$. The distribution $erf(y)$ constrains the behaviour of the increments of the function. With regard to the function 2.1, a reduced Gaussian random-variable distribution is cited by [Mandelbrot 82] in the context of the Brownian function, and the case of the more generalized fractional Brownian function.

Several methods have been devised whereby H may be measured from characteristics of the above model. Where appropriate, they are detailed in the following section.

2.3 Applications of Scaling System Models

2.3.1 Length estimation varying with sample interval

Early work contributing to the recognition of the scaling nature of terrain is provided by [Mandelbrot 67], which details Richardson's work in measuring the perimeters of various countries. By scaling length with resolution, the fractal dimension was determined for each perimeter. Log-log plots of lag versus measured length gave straight line fits over one to two

orders of magnitude, within the range of 10 to 1000 kilometres, resulting in approximate values of 1.5 for the fractal dimension being given in [Mandelbrot 82].

Richardson's analysis of the coastline of Britian gives a fractal dimension value of 1.25. More detailed analysis is documented in [Goodchild 80], which indicates a more complex level of behaviour, with evidence of a break in the scaling system. Within the same scale range as above, the first order-of-magnitude gives a straight line fit with a resulting fractal dimension of 1.15. However, a successive order of magnitude scale range exhibits another distinct linear interval, with the fractal dimension equal to 1.31.

Richardson's measurement method was applied by [Dutton 81] in order to increase cartographic line detail, avoiding the distracting appearance of discrete line segments when datasets are viewed at high magnification. Measurement of United States coastline data resulted in a fractal dimension value of 1.22. However, the author notes that elements of the data exhibit distinct regions of differing fractal dimension, empirically estimated as ranging from 1.1 to 1.3.

Length measures have been used to compare scaling behaviour at differing altitudes. A study detailed within [Goodchild 82] used contour data of Random Island, Newfoundland. From the contours at shoreline, 250 and 500 feet, fractal dimensions of 1.11, 1.19 and 1.31 respectively were recorded. Lake outlines were also analysed, but did not exhibit such uniform scaling, resulting in a short range fractal dimension of 1.8. All measurements indicate an increase in fractal dimension with altitude.

In a later study, contour data was examined by [Culling 87]. In contrast with Goodchild's study, the areas selected were limited to "smooth" soil-covered regions within Southern England. Two distinct ranges of linear scaling behaviour were observed in some samples, a lower fractal dimension being observed for shorter lags. For one of the sample areas, a plot of fractal dimension against contour height is illustrated. The plot may be divided into two distinct sets of values. Below 375 metres, an average dimension value of approximately 1.130 is observed, whereas above this altitude, values decrease to an average of 1.028, the converse of Goodchild's observations. Given the selection of "smooth" terrain data, it is surprising that the range of values observed over the whole study is greater than that recorded by Goodchild, being between 1.059 and 1.532.

In order to test the reliability of length measures, [Muller 86] describes a method by which lines of known fractal dimension were generated and then measured. The study found that values converged to those of a straight line as lag decreased below the generated (or sampling) interval. In addition, a large survey of contour data, from map data at scales from 1/24,000 to 1/1,000,000 was undertaken. Fractal dimensions were calculated for each map scale present, from overlapping scale ranges of lag. Recorded values ranged from 1.03 - 1.24 across datasets. The majority of datasets exhibited a reduction in fractal dimension with map scale, but at smaller scales, self-similarity broke down. Thus, [Muller 86] reached the conclusion that the scale of digitization has an effect on the resulting fractal dimension. Data capture, digitization, and hardware errors, together with the lack of standardisation of cartographic lines (for example, the standard length resolution of measurement of political boundaries), were invoked as an explanation for this behaviour.

Length measures were applied to a different data source in [Keller 87]. Silhouettes of tree and mountain profiles were digitized at various magnifications - the same view but zoomed in, at various standard factors used in 35mm photography. Fractal dimension altered slightly with scale, decreasing from approximately 1.6 to 1.5 over a range of focal lengths of 85mm to 300mm. However, these extremes of scale range encompassed less than one order of magnitude.

2.3.2 Length-Area relationships

Results from a study of the area/perimeter relationship of terrain is given in [Goodchild 82]. Contour lines from the Random Island dataset, used above in a length measure study, were measured at four altitudes (shoreline, 250 feet, 500 feet, and lake outline). The resulting fractal dimensions increased monotonically with altitude, as in the length measure study. However, the rate of increase was far less than that exhibited by length measures.

2.3.3 Mass Dimension

Length measures and length-area measure studies described in [Goodchild 82] indicate a variation in fractal dimension with altitude. In these cases, measurements taken related the length of an outline to a scaling exponent. Mass dimension is distinct, in that its application determines the occupancy of the whole of a Euclidean space by some function

as the measuring interval is altered.

The application of the method to terrain data is described in [Lovejoy 90a]. The process was termed 'functional box counting', and involved the thresholding of terrain to a certain altitude. The resulting plane was uniformly subdivided into boxes. The number of boxes containing terrain exceeding the threshold was noted for each of a series of box sizes, and the fractal dimension of such behaviour was determined from a log-log plot of the number of boxes *occupied* against the length of the box side.

Potentially, thousands of dimension values result, if thresholds are taken at metre altitude intervals. To simplify the process, the application to a 1024 x 1024 km (1km resolution) topographic map of France used thresholds successively decreasing by factors of two, from 3600 metres, to 28 metres. Corresponding fractal dimensions increased from 0.84 (3600m) to 1.92 (28m).

2.3.4 Spectral Analysis

This method is derived from the scaling behaviour of the fractional Brownian model of terrain. Fractional Brownian motion (fBm) deals with the change in behaviour of terrain increments as the point-pair spacing used to calculate the increments is varied. Spectral analysis derives from the relationship between the magnitude of frequencies within a phenomena and the scale of measurement. Given a Gaussian distributed set of terrain increments, the fBm model postulates that the magnitudes of frequencies present will scale as a power law:

$$S(\omega) \propto \omega^{-\beta} \tag{2.2}$$

where

$$D = E + \frac{3-\beta}{2} \tag{2.3}$$

for fluctuations along a path embedded in Euclidean space ($\Re^E, E > 1$), from [Voss 88]. Here, ω represents frequency, and $S(\omega)$ is the resulting power function.

Applications include that detailed within [Forstner 83], where power spectra for terrain profiles were measured, in order to determine quality of interpolation methods. Frequency analysis has also been applied to satellite imagery within [Jones 85]. Both Gaussian and

intermittent (non-Gaussian) fractal processes have been detected in a range of optical, near-infra-red and radar imagery.

Terrain height datasets have also been analysed. [Turcotte 87] used spectral analysis of spherical harmonics to determine the fractal dimension for planetary surfaces. Good linear behaviour, with D = 1.5 for the Earth's surface, was discovered.

A more local analysis is reported in [Huang 89]. One and two dimensional Fast Fourier Transforms were obtained for 1 degree square topographic maps of Arizona. One dimensional analysis returned fractal dimension values of approximately 1.5, and two dimensional analysis a value of 2.1. Significantly, results reported included the effect of using smaller one and two dimensional samples to analyse the same dataset. With decreasing size, the range of fractal dimensions increases. For a single 512 point profile ("grid scale of about seven points per kilometre"), a fractal dimension of 1.58 was recorded, whereas for a series of 32 point profiles covering the original data, a range of 1.31 to 1.86 was recorded. In the two dimensional instance, results exceeded the intuitive range of values for a surface ($2 <= D <= 3$), returning values of $1.90 < D < 2.4$.

2.3.5 Graph of Mean Squared Difference with Scale - the Variogram

As with the spectral technique detailed above, this method is derived from the scaling behaviour of the fractional Brownian model of terrain. In this case, scaling of the terrain increments is determined by measuring change in the variance with scale:

$$< [\Delta Z_{\Delta \boldsymbol{x}}]^2 > \propto || \Delta \boldsymbol{x} ||^{2H} \tag{2.4}$$

where <> denote averages over many samples of $Z_H(\boldsymbol{x})$, H is a scaling parameter with range $0 < H < 1$, and $\Delta \boldsymbol{x}$ is a vector change in position in Euclidean space ($\Re^E, E > 1$). A log-log plot of lag against variance results in a curve with a gradient of $2H$.

Variogram analysis was applied to profiles of terrain located near Oslo, Norway, and was reported in [Frederiksen 83]. Intervals of 15 metres were used, with one profile composed of 500 samples. Linear intervals may be observed in the resulting graph, with a breakdown in such behaviour within the large lags.

An application to two-dimensional datasets was detailed in [Mark 84]. Variograms were

obtained for 17 datasets, 32,000 point pairs being used for each analysis. Scale breaks were detected within the resulting graphs. Analysis of the multiple scale ranges resulted in lower fractal dimension values being recorded for shorter lags. Several of these values were less than 2, lower than the expected range (as in the study [Huang 89] mentioned above).

In addition to the examination of contour data, [Culling 87] also contains details of the application of the semi-variogram to profiles of the regions used within the study. In common with [Mark 84], 17 datasets were used, a single profile being extracted from each contour map. Each profile consisted of approximately 140 samples at 50 metre intervals, making the overall number of samples far smaller than that used in [Mark 84]. The data itself was selected from "smooth" soil covered regions within Southern England. Moreover, the profiles themselves deliberately avoided river valleys and floodplains, as well as major changes of landscape type. Thus, the profiles may be considered as exhibiting "smooth", homogeneous behaviour.

Within many of the semi-variograms, distinct scale ranges were observed, confirming the scale break results reported in [Mark 84]. It appears that, for shorter lags, resulting fractal dimension values are lower than those for larger lags. Short lag dimension values range from 1.08 to 1.27, with larger lag values returning values in the range 1.33 to 1.76. Again, this confirms the behaviour reported in [Mark 84].

However, as with the comparison of contour studies ([Goodchild 82] with [Culling 87]), the fractal dimensions of "smooth" terrain measured using variogram analysis return interesting results. Using the co-dimension addition rule from [Mandelbrot 82], it may be shown that the fractal dimension of a profile embedded in a surface is 1 less than the fractal dimension of the surface itself. This allows the comparison of the results from [Mark 84] and [Culling 87]. For shorter lags, the values reported in [Culling 87] lie within the middle of the range of values given in [Mark 84]. For large lags, the range of values of the studies are almost identical. Results which indicate that "smooth" terrain exhibits fractal behaviour similar to that of a variety of terrain sources appears to contradict the expectations of a variety of literature ([Mandelbrot 82], [Pentland 84], [Voss 88]).

2.3.6 Graph of Mean Absolute Difference with scale

As above, the fractional Brownian motion model was the basis for this method of measurement, which was first proposed in [Pentland 84] and [Pentland 85]. The variation in value of sample value change, ΔF, over various distances $\Delta \boldsymbol{x}$, is used to measure power-law scaling characteristics of the data.

Equation 2.1 implies that $\mid \Delta F_{\Delta \boldsymbol{x}} \mid \mid\mid \Delta \boldsymbol{x} \mid\mid^{-H}$ is normally distributed for a true fractal. Furthermore the power law relation between ΔF and $\Delta \boldsymbol{x}$ should be invariant for a range of $\Delta \boldsymbol{x}$ for fractals, implying that the shape of the distributions for fixed values of $\Delta \boldsymbol{x}$ will remain the same (Gaussian). Thus, the expected value of $\mid \Delta F_{\Delta \boldsymbol{x}} \mid \mid\mid \Delta \boldsymbol{x} \mid\mid^{-H}$ [i.e. $E(\mid \Delta F_{\Delta \boldsymbol{x}} \mid \mid\mid \Delta \boldsymbol{x} \mid\mid^{-H})$] should remain constant for all $\Delta \boldsymbol{x}$. Hence we have:

$$E(\mid \Delta F_{\Delta \boldsymbol{x}} \mid \mid\mid \Delta \boldsymbol{x} \mid\mid^{-H}) = E(\mid \Delta F_{\Delta \boldsymbol{x}=1} \mid) \tag{2.5}$$

The computation of sample change statistics at various space scales will enable H to be deduced. Re-writing equation 2.5 as:

$$E(\mid \Delta F_{\Delta \boldsymbol{x}} \mid) = \frac{\mid\mid \Delta \boldsymbol{x} \mid\mid^{H}}{E(\mid \Delta F_{\Delta \boldsymbol{x}=1} \mid)}$$

and taking logs, we have:

$$\ln(E(\mid \Delta F_{\Delta \boldsymbol{x}} \mid)) = H \ln(\mid\mid \Delta \boldsymbol{x} \mid\mid) - \ln(E(\mid \Delta F_{\Delta \boldsymbol{x}=1} \mid)) \tag{2.6}$$

Now, $\ln(E(\mid \Delta F_{\Delta \boldsymbol{x}=1} \mid))$ is a constant, hence equation 2.6 gives a straight line relationship between $\ln(E(\mid \Delta F_{\Delta \boldsymbol{x}} \mid))$ and $\ln \mid\mid \Delta \boldsymbol{x} \mid\mid$ with gradient H for a true fractal. In common with previous methods, H is determined by computing the gradient of least-squares regressions of the graph $\ln(E(\mid \Delta F_{\Delta \boldsymbol{x}} \mid))$ against $\ln(\mid\mid \Delta \boldsymbol{x} \mid\mid)$ for a linear interval.

This method has been applied to the measurement of image texture within [Pentland 84]. Evidence is given of scaling with image pixel-pair separation. However, the range of values measured was limited, and resulting fractal dimension was calculated by linear regression to the entire graph. In the example detailed, the graph itself appeared sinuous rather than linear, so it is questionable how well the resulting value represented the 'true' fractal dimension of the set. Image segmentation was demonstrated for a variety of scenes, at differing resolutions.

In a further paper, [Pentland 85] applied the above method to the determination of terrain fractal dimension from images, followed by the reconstruction of an image or terrain grid using the resulting fractal dimension.

Pentland's method has also been applied to the segmentation of natural outdoor scenes by [Hyde 85]. In all cases detailed, the graphs of the scaling relationship were distinctly non-linear.

Terrain has been analysed by this method, the results being reported in [Yokoya 88] and [Yokoya 89]. Graphs in these papers exhibited considerably greater linearity than those detailed above. DEMs were segmented into overlapping tiles, with fractal dimension calculated independently for each. These fractal dimensions were then used to interpolate detail within an example DEM.

2.4 Evidence for Varying Fractal Dimension Behaviour

Some studies and simulations of the fractional Brownian motion (fBm) model have put forward the conjecture that it may be applied as a global description of terrain behaviour (see, for example, [Mandelbrot 77], [Mandelbrot 82], and [Turcotte 87]). It has also been proposed within [Burrough 84] that fBm represents a good null hypothesis for terrain description.

With the application of various statistical techniques to more and more data, evidence has emerged of a more complex form of behaviour. Of the studies detailed earlier, many have analysed multiple areas of topography, and a range of fractal dimension values have resulted. Segmentation of data has already been applied to imagery using techniques derived from the fBm model, within [Hyde 85], [Keller 89], [Lam 90], and [Pentland 84].

A survey of fractal dimension values, resulting from a variety of measurement techniques, is detailed in [Burrough 81]. Results are mainly confined to soil content measures and mineral distributions. A wide variety of values are tabulated, ranging from 1.1 to 2.0. Comparable measures, such as soil pH, give differing values in differing regions. Interestingly, the only two measures of a surface, both derived form variogram information, give the same value of D=1.5. Given that one of the datasets is the surface of an airport runway ("one of the smoothest surfaces imaginable in a landscape"), the issue as to what fractal dimension discriminates in a landscape is highlighted.

Variability extends below the surface of terrain. The use of the dividers method in the analysis of cave length scaling is reported within [Laverty 87]. Linear paths within five different caves were examined, and considerable differences were observed, resulting in a range of fractal dimension values of 1.0 to 1.5.

Two methods used to measure the fractal dimension of terrain profiles were compared, the results being published in [Kazu 91]. A derivative of the box counting method was compared with results from the slope of a Power Spectral Density function. Box counting results exhibited a range of fractal dimensions, from 1.16 to 1.75, whereas the PSD method returned a range of 1.30 to 1.53. Interestingly, analysis of a selected rough profile produced the highest box dimension value, and the second lowest PSD result.

Within [Goodchild 82], a relationship has been demonstrated between measured contour length scaling and altitude. Methods relating to box counting and area/perimeter methods have been used to show that fractal dimension rises (at different rates, depending on the calculation method) with contour altitude. Results suggest a non-homogeneous fBm process, with spatial variation both with position and elevation. It is postulated that the coastline fractal dimension is lower as a result of smoothing processes, such as beach formation. Thus, a process based hypothesis was related to values determined using the fBm model.

An erosional landscape formation model has been proposed by [Newman 90], whereby magnitude of erosion is related to the magnitude of storms. To synthesize the results, the coefficients of a Fourier series experienced rates of decay dependent only on themselves and one lower coefficient. This created a non-linear cascade, which was shown to produce an evolving fractal distribution, even when the initial distribution was not fractal.

Fractal dimension has been related to the age of a surface. Results detailed in [Elliot 89] show the appliance of variance scaling to profile data collected from a glacier environment. Hence, the date of surface formation could be estimated. Fractal dimension appeared to increase with age over a range of three decades, a "counterintuitive result" given that this quantity was applied as a measure of surface roughness.

Other studies have examined process based influences on fBm results. The study by [Culling 87] detailed earlier was expanded in [Culling 88]. Both report the application of fractal measurement techniques to 'smooth' soil covered terrain. Within the sample of

profiles measured, fractal dimension varied, remaining below 1.4 in the case of first linear interval. Culling justified this terrain type as being well modelled by fBm, as the formation process is stated as tending towards a Gaussian surface. Analysis of the profiles confirmed this hypothesis.

The study of a wide variety of regions was reported in [Klinkenberg 92]. The dividers method, box counting, and variogram analysis were applied to 55 USGS DEMs. Of these, seven DEMs were classified by their physiographic region. A large range of values was recorded, the highest being that of coastal plain data. It was proposed that areas of high erosion, such as the coastal region, exhibit the highest fractal dimensions. This need not contradict the results of [Goodchild 82], where variation was apparent within the single region which was examined.

Within [Mark 84], distinct scale breaks were observed in the variograms analysed. Scale breaks have since been reported in [Culling 87], [Culling 88], and [Klinkenberg 92]. Such behaviour was related to the scale at which ridge and valley structures cease to dominate by [Mark 84]. Measured fractal dimension is generally higher for lags of kilometres than for lags of tens or hundreds of metres. It is also noted, within [Goodchild 87] that scaling properties do not always apply, as "the heights of the tallest mountains are not hyperbolic, however, because of obvious limits to growth".

Additional evidence for scale breaks comes from [Elliot 89]. Variance of rock surfaces were measured in a scale range of tens of centimeters to one meter lag. Results indicate between one and three scale breaks being present. Fractal dimension values generally decreased in scale ranges drawn from greater lags. Thus, not only does fractal dimension vary with position and elevation, but also with scale.

It has also been suggested that information complementary to fractal dimension results might be used in data processing. One such measure is lacunarity, which is detailed within [Keller 89]. Several definitions exist - basically, lacunarity is a measure of texture which may be used to distinguish surfaces with the same fractal dimension. The application of this method to image textures aided in the segmentation of regions.

Functional box counting results over terrain in France are given in [Lovejoy 90a]. These demonstrate multiple scaling over a given range, 1 kilometer to 1024 kilometers, at various

threshold altitudes. These results have lead to the modelling of topography using multifractals.

Within multifractal theory, there is the explicit expectation that multiple scaling exponents will be encountered - i.e. the exponent is a function rather than a constant. From [Schertzer 89], this may be modelled by the use of a cascade, whereby exponents may change with position due to multiplicative rather than additive processes. Simulations give visually interesting results.

The power of this model is demonstrated within [Lavallee 92], where it is proposed that only two parameters are required to model the potentially infinite number of scaling exponents - the codimension of the surface (a function rather than a monofractal constant) and the degree of multifractality.

A justification of the formation process was based on a cascade model of turbulence. In the case of topography, fluctuations due to gravity and the velocity of tectonic plates were postulated as the dynamic processes responsible for the cascade.

Analysis of the Deadmann Butte 50 meter DEM resulted in a multifractal index close to the maximum. In addition, the 1 kilometer France DEM mentioned above was re-examined for such multifractal behaviour, and returned similar results to Deadmann Butte. Thus, multifractal behaviour characterised by the two parameters is evident over a large range of scales.

2.5 Selection of Computation Method

If fractal models are to be integrated within a library of image processing techniques, it must have a serious application in this domain. Interpolation and simulation are the only major applications for this model mentioned in any of the above papers. However, [Dutton 81], [Goodchild 82], [Huang 89], [Mark 84], [Yokoya 88] and [Yokoya 89] show variation in fractal dimension with altitude, space, and scale. This reveals the potential for a terrain characterisation and segmentation process, as well as the application of interpolation to areas of homogeneous fractal dimension. In order to achieve these aims, a model must be chosen which allows analysis of area data, while being compatible with current interpolation techniques.

Analysis using Euclidean primitive scaling is mainly confined to linear features, and thus presents problems if a fractal dimension image is to be created for analysis. Area-perimeter methods suffer from a similar disadvantage.

Multifractal models do allow for area analysis. However, the technique used for terrain analysis, functional box counting, produces fractal dimensions for height thresholds. Thus, if applied to segmentation, comparison would be difficult, since it cannot be guaranteed that all sub-sections of a terrain surface would contain similar altitude ranges. Improved techniques were subsequently discovered, but came too late in the progress of the study for their implementation to be considered.

Box dimension has been applied to the analysis of simulated fractal surfaces, the results being detailed within [Keller 89]. The box dimension tends to underestimate the values used to generate such surfaces in the case of high fractal dimension, but modifications are proposed to overcome this deficit in the method.

Of the models described above, fractional Brownian motion dominates the literature available concerning fractal terrain analysis. A wide variety of interpolation methods are associated with this model. Major examples include [Fournier 82], [Miller 86], [Musgrave 89], [Saupe 88], [Saupe 89], and [Voss 88]. Most of the above are widely referenced, being considered key papers in this subject. Notably, all these papers describe terrain creation algorithms based on scaling of variance, with [Saupe 88] and [Voss 88] detailing additional alternative methods.

The selection of fractional Brownian motion as the model to be investigated results in a choice of three measurement methods. Spectral analysis is normally achieved using Fast Fourier Transforms. However, these generally work with data sizes of 2^N on at least one side, whereas a general technique must be applicable to datasets of any size, and in many cases must be based on a non-uniform grid. Moreover, serious application of the method requires a low processing time overhead for these various datasets.

Mean absolute difference scaling can fulfil these requirements, since it compares height differences of point pairs, where only spacing need be known. However, in [Kube 88], Kube and Pentland comment on Pentland's original method when applied to terrain analysis via imagery. Analysis is possible, but the pre-conditions, namely that the surface exhibits purely

Lambertian reflectance, modest slopes, and an absence of occlusion or shadows, remove many areas from consideration.

Variogram analysis exhibits the advantage, as above, of comparing height differences of point pairs from any spatial sampling schema. Computation time is slow, but holds the potential for considerable improvement. An important consideration is that, as detailed above, many of the principal fractal terrain generation techniques rely on variance scaling. Therefore, the variogram analysis method was selected as the tool for use in this study.

Chapter 3

Computation of the Fractal Dimension

3.1 Determination of Fractal Dimension within Variograms

Implementation of the variogram analysis technique required that each stage of information processing be specified. Our method of fractal dimension computation for DEMs is principally derived from that detailed in [Mark 84]. Since it is only valid to determine the fractal dimension over a scale range of constant dimension value, a linear segmentation algorithm (but not the fractal measurement algorithm) due to [Yokoya 88] was used to determine such intervals within log-log plots.

Given the constraints on fractional Brownian motion detailed in equation 2.1, this function exhibits behaviour that gives a similar shaped distribution at different scales. For fractional Brownian motion, $\Delta Z_{\Delta \boldsymbol{x}}$ are the increments of a single valued function $Z_H(\boldsymbol{x})$, the increments having a Gaussian distribution and a variance as in equation 2.4. Taking logs of equation 2.4 we have:

$$\ln(< [\Delta Z_{\Delta \boldsymbol{x}}]^2 >) = \ln(\| \Delta \boldsymbol{x} \|^{2H})$$

and finally, rearranging the above:

$$\ln(< [\Delta Z_{\Delta \boldsymbol{x}}]^2 >) = 2H \ln(\| \Delta \boldsymbol{x} \|) \tag{3.1}$$

Equation 3.1 gives a straight line relationship between $\ln(< [\Delta Z_{\Delta \boldsymbol{x}}]^2 >)$ and $\ln(\| \Delta \boldsymbol{x} \|)$ with gradient $2H$ for a true fractal. Empirical studies in the past have found that fractal components of natural scenes appear to preserve their fractal dimension (and hence H value)

over a variety of ranges in $\| \Delta \boldsymbol{x} \|$. An algorithm to determine these ranges is given later. H is determined by computing the gradient of least-squares regressions of the graph $\ln(< [\Delta Z_{\Delta \boldsymbol{x}}]^2 >)$ against $\ln(\| \Delta \boldsymbol{x} \|)$ for a linear interval.

Finally, the fractal dimension, D, is then obtained from H using the simple relation

$$D = 3 - H \tag{3.2}$$

This is derived by use of the fact that the dimension of the graph of a fractional Brown Line-to-Line function is

$$D_l = 2 - H \tag{3.3}$$

from P352, [Mandelbrot 82].

The co-dimension of a function is its fractal dimension subtracted from its Euclidean dimension:

$$C = E - D$$

For a Line-to-Line function, the particular values from the above equation are

$$C_l = E_l - D_l \tag{3.4}$$

The Line-to-Line function in this case resides in a volume, so E_l is 3. Substituting this, and equation 3.3, into equation 3.4 gives

$$C_l = 3 - (2 - H)$$

$$C_l = 1 + H \tag{3.5}$$

The DEMs examined reside in 3-dimensional space. Thus, the Euclidean dimension for a DEM surface, E_s, is 3. The fractional Brown Line-to-Line graph can be viewed as the intersection of a vertical plane with the DEM surface. Using the co-dimension addition rule ([Mandelbrot 82], P3least

65), C_s for the co-dimension of the surface, and C_p for the co-dimension of the plane, we have:

$$C_l = C_s + C_p \tag{3.6}$$

The co-dimension of the plane, residing in a volume with Euclidean dimension for the plane of $E_p = 3$, is

$$C_p = 3 - D_p$$

$$C_p = 3 - 2 \tag{3.7}$$

Substituting 3.5 and 3.7 into 3.6 gives:

$$(1 + H) = (3 - D_s) + (3 - 2)$$

$$D_s = 3 - H$$

3.2 Automated Determination of Linear Intervals within the Variogram

3.2.1 Behaviour of the functions

Once all obtainable H values for the DEM data have been calculated, the range of scale over which H is invariant must be determined. Several methods have so far been applied. The most commonly cited method involves a visual or least-squares fit of a single line to the entire graph. The majority of the application studies previously cited use this technique, [Pentland 84] being one example.

However, many graphs illustrated within [Mark 84] exhibit scale breaks when analysed manually. Of the example graphs given in the above studies, many exhibit irregularities which bring into question the application of a single linear descriptor. A single regression line might bias the results towards the whole scale range of the study, rather than concentrating on scales of distinct behaviour. A realistic method must be able to detect scale breaks present, while being automated in order to deal with the large quantities of data to be analysed.

[Peleg 84] proposed the calculation of fractal dimension by fitting a line to successive triples of neighbouring points along the graph, and measuring the gradient. This method was applied to expected difference graphs in a study by [Dodd 87], which has shown that human observation cannot distinguish fractal dimensions of generated images that differ by H values of less than 0.06. Given the context of human observation, the above limit seems an appropriate range within which to consider H visually constant. However, the current study deals with fractal dimension as a classification parameter that might be applied to imagery, or digital elevation models. It is possible that such classifications would defy visual interpretation. Thus, such a technique must detect scale breaks by linearity behaviour, rather than homogeneity of a graph segment.

The most promising method to date is applied in [Yokoya 88], which we have adapted. This method is detailed in the following section.

3.2.2 Linear Segmentation Of The Variogram

A measure of linearity put forward in [Yokoya 88] and [Yokoya 89], can be defined as:

$$I = \frac{\sqrt{4\mu_{11}^2 + (\mu_{20} - \mu_{02})^2}}{\mu_{20} + \mu_{02}} \tag{3.8}$$

where $\mu_{ij}(0 \leq i, j \leq 2; i + j = 2)$ represents the central-order second moment of a set of points in a plane.

Let S_n denote a set of n points (where $n \geq 3$) in a plane of fractal plots as

$$S_n = \{p(\| \Delta x \|)\}_{\|\Delta x\| = \|\Delta x\|_{min}}^{\|\Delta x\|_{min}+n-1}$$

and I_n represents the measure of linearity computed for S_n. The largest limit $\| \Delta x \|_{max}$ of scale within which the function is linear is determined by:

$$\| \Delta x \|_{max} = \| \Delta x \|_{min} + n^* - 1$$

where n^* is given by

$$n^* = Min\{n \mid I_{n-1} \leq I_n \,,\; I_n > I_{n+1}; n \geq 4\} \tag{3.9}$$

Note: this is not the equation as given in [Yokoya 88] and [Yokoya 89], which contained the limits:

$$I_{n-1} < I_n \,,\; I_n > I_{n+1}; \tag{3.10}$$

Equations 3.10 will fail to produce a correct solution when two linear intervals, the first of which produces a perfect fit (to the accuracy of the implementation), are presented. To demonstrate this failure case, consider the behaviour of the equation as the end of the first linear interval is approached. Initially, I will gain a measure of 1.0 for the first line. However, when the second line is encountered, I_{n+1} will drop in value, but both I_{n-1} and I_n will remain equal. The test will not detect this as a break in the linearity. Beyond the break in linearity, I will, in many instances, steadily decrease, repeatedly failing the test. Thus, the two linear intervals will be treated as one imperfect linear interval by the original criteria.

3.2.3 Selection of Linear Intervals

As will be detailed later, analysis of the various datasets uncovered a wide variation in linear interval ranges within variograms of DEM regions. Because of this, selection criteria, such as N^{th} linear interval, or interval of best fit, provide results that may not be comparable across the dataset. Moreover, linear intervals over large lag values can become more error prone, since the number of samples contributing to the resulting variance decreases with lag increase. Thus, out of expediency, the first linear interval within a graph was chosen to provide all data in this study.

3.2.4 Determining the Fractal Dimension

Imperfect linear fits are a likely result of most analysis, so a least squares regression was applied to find the coefficients of the best fit line. From equation 3.1, the gradient of the regression line is equivalent to the value $2H$. The resulting fractal dimension may be calculated from the gradient by substituting the value for H within equation 3.2. The Y-axis intercept, C, was retained, to be used in further tests detailed below.

3.3 Accuracy Estimates for Computed Fractal Dimension

3.3.1 Goodness of fit of the interval to sampled data

The above method was a recently proposed tool, modified during the course of this study. Thus, it was necessary to evaluate the results obtained to ensure their validity and accuracy. Since supposedly linear segments are returned, a technique to determine linear fit error was applied.

The R-squared test may be applied to the regression equation

$$Y_{est} = 2HX + C \tag{3.11}$$

to determine the variation of the resulting values as follows:

$$r = \sqrt{\frac{\sum(Y_{est} - \frac{\sum Y}{N})}{\sum(Y - \frac{\sum Y}{N})}}$$

from [Spiegel 72], page 243. The correlation coefficient, r, measures the ratio of explained variation (due to the choice of regression line) to that observed within the samples themselves.

A value of one implies a perfect linear fit, degrading with decreasing value, until the value zero, which implies an extremely poor fit.

3.3.2 The Student T-Test: Statistical Significance

Previous studies, including [Huang 89] and [Mark 84], have recorded the detection of fractal dimension values (resulting from the fractional Brownian model of terrain) which exceed the expected range for a surface. Given the scale of this study, it seemed likely that such results would be repeated. Therefore, an automated method was sought to determine the significance of such extreme values.

The Student t-Test may be applied to such an analysis. Fractal dimension values were obtained from the gradient of a linear interval within the variogram. Thus, the number of points in the graph used to determine the gradient bears on the significance of the result. Accordingly, a variant of the t-Test which incorporated this information was used. From [Spiegel 72], page 247, the t-Test can be evaluated as follows:

$$t = \frac{a_1 - A_1}{\frac{s_{Y.X}}{S_X}} \sqrt{N-2} \tag{3.12}$$

where a_1 is the measured gradient, A_1 is the population mean gradient, $s_{Y.X}$ represents the standard error of Y on X, and S_X represents the standard deviation of X. The number of points in the linear interval is N. The number of degrees of freedom is evaluated as $N-2$, as 2 points can be applied to determine an unambiguous set of linear coefficients.

The resulting value, together with the number of degrees of freedom, was then applied to index a table, representing the t-Test distribution for various degrees of freedom. From this, the significance of the gradient, and thus that of the fractal dimension value, was determined in relation to the population, and the size of the linear interval encountered. Table values were obtained from several publications, principally page 344, [Spiegel 72], to obtain the maximum resolution of significance values, and to allow cross-checking of values.

3.3.3 The Chi-Squared Test: Fit to the Gaussian Distribution

Critical to the behaviour of fractional Brownian motion is the constraint on the distribution of the increments. It can be seen from equation 2.1 that the behaviour of the model is determined by a Gaussian function. The vast majority of fractal interpolation papers surveyed

apply the Gaussian model within their generation algorithms, but, surprisingly, in only one of the fractal terrain analysis papers surveyed was the validity of this distribution tested with respect to the data under scrutiny.

[Culling 87] relates this study, in which profile data obtained from Ordenance Survey contour data of Southern England was examined. The profiles deliberately avoided changes in terrain behaviour and low height variance regions, such as river valleys and floodplains. Thus, certain terrain types were not subject to Gaussian behaviour tests. Results were acquired by plotting the distribution of increments on probability paper. Observation of the linear behaviour on the probablity graph gave three classes of fit. By these criteria, the majority of the 17 profiles analysed exhibited Gaussian behaviour. Within [Burrough 83a], the distribution of soil changes was compared visually with that of a normal distribution, a poor fit being observed. Both of these methods involve some form of visual comparison to obtain a measure of fit. However, given the scale of the present study, an automated method was required.

The Chi-Squared test quantifies the similarity of observed behaviour to that expected from a given distribution. The test is given in [Spiegel 72], page 201, as follows:

$$\xi^2 = \sum_{j=1}^{k} \frac{(o_j - e_j)^2}{e_j} \tag{3.13}$$

where k is the number of samples, o_j represents the observed frequency for a given sample, and e_j is the expected frequency for a given sample.

In order to test for Gaussian behaviour, increments detected were binned into 20 intervals, 10 either side of the increment value 0.0 (the expected increment for a non-skew distribution). Bin intervals were specified by the user, given *a priori* information about increment ranges in the DEM under analysis. Of the resulting frequencies, all those lower than 5 were excluded from further analysis, due to the innacuracies that might result from frequency comparisons at such a coarse resolution.

The expected Gaussian values were generated from a look up table (from the appendix of [Spiegel 72]). The table contained 400 values, representing cumulative area beneath the Normal curve, measured from the mean to a given standard deviation. The values were scaled, so that the total area under the distribution was 1.0. Values up to 4 standard

deviations were included. Thus, each successive value was differed by a percentile of the standard deviation. Effectively, the table contained 800 values, due to symmetry of the distribution about the mean. To give greater resolution with minimal computation, values used were linearly interpolated from the two table entries surrounding a given standard deviation.

To obtain a comparable Gaussian value, the mean and standard deviation of the observed distribution were obtained. The mean was used to offset indexing into the Gaussian array, such that trends observed in the finite input dataset area would be removed. The standard deviation was then used to obtain cumulative area under the curve values in the equivalent positions to those of each observed frequency bin edge. The expected area of one edge of a bin was then subtracted from that of the other edge, so that an unsigned result remained. This was treated as the normalised expected frequency for that bin interval, and, once multiplied by the total observed frequency, could be used in equation 3.13 for comparison.

The Chi-Squared output would change, depending on the specified number of bins used for comparison. To gain some measure of the significance of the result, another table within [Spiegel 72] was used to return significance values for the number of degrees of freedom present (18 in this case).

To test the validity of the t-Test implementation, a Gaussian height field was generated, and the program was applied to this data. Resulting values were well within expected significance limits, considering the number of degrees of freedom present. As an additional test, a Chi-Squared comparison was applied to values within the second point-pair spacing specified, using the distribution obtained from the first point-pair spacing as a reference. Not only would these results aid in checking the implementation, but might also return useful results indicating how distributions change with sampling interval.

Chapter 4

Implementation of Applicable Techniques

4.1 Introduction

A comprehensive evaluation of the behaviour of measured fractal dimension requires analysis of a variety of terrain types. Furthermore, significant terrain sample sizes must be analysed in order to gain reliable values. The largest such evaluation prior to the present work, detailed in [Mark 84], involved the sampling of 32,000 point pairs from each of 17 datasets to construct a variogram. However, the current study involves measurement of local variability, while endevouring to detect well defined ranges of scale invariance within each locale. This implies effective usage of available data.

Application of the above method to the Deadmann Butte dataset (512 by 512 height samples) resulted in a measured time of 36000 CPU minutes (25 CPU days) to analyse the data on a Sun 3/180 workstation. This was totally unacceptable, due both to the constraint upon the number of datasets to be analysed in the study period, and to the vulnerability of such a long running process to machine restarts.

Thus, during the early phases of the study, a significant portion of time was spent optimising the variance algorithm and code. Moreover, intermediate data storage formats were sought, so that the computationally expensive aspects of the analysis need only be performed once.

4.2 Initial Implementation

The process of analysing the fractal dimension of variance statistics may be divided into three broad stages. Initially, the variance must be calculated over all samples for each point spacing required. Secondly, one or more linear intervals must be extracted from the variance data. Finally, the resulting fractal dimensions and original data must be examined to determine their relationship to theoretical expectations.

Measurement of the variance involves use of all input data for computation. In contrast, determination of fractal dimension from the variance and some theoretical fit techniques use comparatively little data, and may be performed in a few CPU minutes on moderate sized datasets. Extensive analysis and comparison was required for the completion of the study. As a result, subsequent programs computed variance values and other information separately from other analysis programs.

Representation of the intermediate data was required. In addition to the variance, the number of point samples used was recorded, to detect poor sample sizes in regions such as coastal boundaries. Variance squared was also stored, to allow for the potential use of error bounds. These triples must be stored for each point-pair spacing, and also for each sub-region of the dataset analysed. The HIPL image processing library [HIPL] includes a storage format, consisting of two dimensional arrays of values, which may in turn be stored in a sequence of frames, all values being of the same type. Accordingly, the two dimensional arrays were assigned to associated positions in the original dataset input, but degraded to the resolution of the tiles used to divide the height data. The triples of values were stored in three successive frames, in the order variance, variance squared, and number of samples. Data resulting from each point-pair spacing was stored in three frame groups, ordered from smallest to largest spacing.

4.3 Structural Improvement of the Initial Implementation

Before embarking on a programme of processing-time optimisations, structural foundations were laid within the program. This helped to ensure its reliability and readability - important given the long operating times experienced, and the mass of modifications which were to be made. Moreover, such structures provided a level of abstraction to allow certain elements

to be shared by the different implementations.

Distinct levels within the hierarchy of processing were identified. At the lowest level, access to the input and output data sets was abstracted. All usage of this memory was routed via four functions, a read and write function for input and for the output storage locations. This allowed input and output data types to be altered with no effect on higher level routines, and also aided in the implementation of the row-by-row input method described below.

At a higher level, the point-pair comparison technique was inherent in the variogram technique itself. No matter what optimisations were carried out, this routine would remain as a core function of the program. Accordingly, a single subroutine was created to perform this task. During the investigation of some of the data sets, it was discovered that values at or below sea level should be ignored, otherwise the flat surface might affect the variance values recorded for coastal regions. The solution involved the incorporation of a simple test which would return a false value if either of the point-pair values lay below a specified height.

One step further up the hierarchy, the method for comparing all point pairs within a specific area (or around a specific point) is encountered. This subroutine varied between the different implementations, but the lower level structure allowed the various algorithms to be easily implemented. In addition, performance increases were gained, as the relegation of point-pair comparison to a single function allowed loops at this level to be expanded, allowing less time to be spent in branching comparisons.

The proposed usage of the final product was in the segmentation of input data sets. This implied that the various regions of the data were to be treated independently. Accordingly, a further level was added to the hierarchy, allowing the operations performed within one area to be expressed as a single function. A simple double loop was then created to apply the method to all tiles in a two dimensional grid covering the input information. The usefulness of this structure was appreciated when row-by-row input was required, as this matched the outer loop structure. The addition of a single read function was all that was required to accommodate the new algorithm at this level.

Finally, all data input and output routines, as well as those that had to read and check command line arguments, were placed in distinct groups of subroutines. A single function was also incorporated to perform all the variance analysis, providing a readable highest level

to the program code. The resulting structure required minimal alteration at the highest and lowest levels. This allowed new implementations to be rapidly created and tested. In addition, the structural abstraction created an environment in which algorithmic optimisation was easily expressed.

4.4 Programming Performance Optimisations

In addition to the more mundane optimisations, two minor alterations to the innermost loop of the variance measurement program were made to improve performance. First, pre-computing the addresses of each row start before analysing a tile avoided a multiplication when calculating addresses in the central loop. Secondly, frequency count variables were internally represented as integers, to improve increment performance.

Comparisons of height differences for constant inter-point spacings results in a circle of samples around each individual sample being selected. The overhead in calculating these positions is considerable. Pre-computation of these circular offsets once for each spacing removes computation from the central loop. Symmetry of the circle implies that it is sufficient to calculate only a quarter of the circle. Moreover, comparisons need only be made for a semi-circle, to avoid bi-directional comparisons of the same points taking place.

Reading the entire dataset into memory to process it might slow execution time, as the image of the program may be frequently swapped to disk. In the current study, input data is arranged in row major order, and must be received from the standard input (i.e. the data may not be re-read). Within a tile, all values are accessed many times. Therefore, the minimum amount of data that can be input at one time is the number of complete rows needed to cover a row of subdivision tiles.

Finally, the use of in-line libraries and optimisation at the compilation stage aided in producing efficient executables.

4.5 Algorithmic Improvement

Improvements detailed in the previous section do reduce the execution time by a constant factor. However, execution time for envisaged datasets still remained too high to consider their analysis.

The brute force algorithm has a time complexity of $O(N^2)$, where N is the number of points within a variogram tile. This estimate is derived from the number of comparisons required - each of the N points must be compared with $O(N^{1/2})$ circles, each composed of on average $O(N^{1/2})$ points.

Limiting analysis to just neighbouring rows and columns, rather than a full circle, would reduce time complexity to $O(N^{3/2})$. This method may be shown to measure non-directional variance. Within the original method, variance was obtained by gaining the expected squared height difference. This expression can be expanded to include a third point (to allow for independent row and column measurements) as follows:

$$(z(x_1) - z(x_2))^2 = ((z(x_1) - z(x_3)) + ((z(x_3) - z(x_2)))^2 \tag{4.1}$$

Expanding the right hand side of the above equation gives:

$$= ((z(x_1) - z(x_3))^2 + ((z(x_3) - z(x_2))^2 + 2(z(x_1) - z(x_3))(z(x_3) - z(x_2)) \tag{4.2}$$

The directional variance components, and the covariance can be defined as:

$$Var(X) = E((z(x_1) - z(x_3))^2) \tag{4.3}$$

$$Var(Y) = E((z(x_3) - z(x_2))^2) \tag{4.4}$$

$$Cov(X,Y) = E(z(x) - z(x + \delta x)) * E(z(y) - z(y + \delta y)) \tag{4.5}$$

Using the expected value of equation 4.1 :

$$E((z(x_1) - z(x_2))^2)$$

equation 4.2 can be altered to reflect the expected value, on the assumption that row and column values within a height array are independent (ie. $x_1 \rightarrow x_3$ and $x_3 \rightarrow x_2$ represent perpendicular vectors). Substituting equations 4.3, 4.4, and 4.5 into the expanded version of equation 4.2 gives:

$$E((z(x_1) - z(x_2))^2) = Var(X) + Var(Y) + 2Cov(X,Y) \tag{4.6}$$

Note that this information returns independent X and Y directional variance for no significant computational cost. To accommodate this, the intermediate storage format was altered to store triples of values (variance, variance squared and frequency) for each of the row, column and co-variance values. However, care must be taken in usage of this information, due to the lower number of sample pairs compared to compute each variance value.

4.6 Spherical Coordinate Systems

The improved performance of the algorithm allows for the analysis of much larger datasets. This imposes an additional consideration - due to the proportion of the planetary surface covered, such data is generally available in geographic projections. Inter-point spacing determination lies at the core of the algorithms above, but geographically projected data exists on a grid with non-uniform spacing. Re-projecting locally to a uniform grid would remove some information, due to the resampling required, and would also take a large amount of time and effort to perform. Accordingly, the variance algorithm was altered to take account of the nature of this data format.

Three programs were written. The first two applied the original and row-column algorithms to uniform grids, to allow comparisons of results from the two methods. A third program was written to apply the row-column algorithm to geographically projected data.

Due to the non-uniform sampling interval, it was no longer possible to collect all the data at one sample-pair spacing at one time. As an alternative, for each point, samples were evaluated at all grid intervals used, and were placed in cumulative bins representing the true surface distances.

Calculating the distance over the surface of a sphere for each point comparison is extremely time-consuming. Computational savings were made by noting that, for the distances under consideration (10km maximum), the planar distance between two points differed insignificantly from the sphere surface distance (less than 0.00005%) compared with errors likely to be present in the sampled data.

A significant saving was also made by noting that within each row of data analysed, the unit spacing between each column was identical, and larger intervals were simply integer

multiples of this value. Moreover, unit row spacing distance is identical for the whole dataset, so this need only be computed once.

Finally, a saving in machine core space could be made, as all the geographically projected datasets used in the study were provided in short integer format. Thus, the programs used to analyse uniformly projected small datasets read in four byte floats, whereas the program applied to the more extensive spheroid data read in two byte height datasets.

4.7 Conclusions

Improvements detailed above, combined with the usage of faster Sun 4 computers, lead to a significant gain in performance. Computation time for the dataset first studied, Deadmann Butte, decreased from 36,000 to 240 CPU minutes. This can be broken down into three independent improvements. Moving to a faster computer architecture resulted in a performance improvement by a factor of approximately three. A similar performance improvement was attributable to the programming improvements. These are constant factors, and are unlikely to vary much between differing datasets.

The most significant improvement resulted from the row-column variance algorithm. In the example case, this resulted in an order of magnitude performance increase. The time complexity improvement, from $O(N^2)$ to $O(N^{3/2})$, lies at the heart of this improvement. Moreover, geographical projection optimisations produced an unknown constant factor improvement in the sphere surface variance implementation.

Without these improvements, the analysis of the larger datasets could not have been seriously contemplated. The German dataset would certainly have taken several CPU years to analyse, longer than the duration of the entire study. Furthermore, without the division of the analysis into variance gathering and fractal analysis stages, the complex behaviour of the results would not have been as rigorously analysed. Thus, the combination of these techniques constituted a crucial element in the production of the results and insights detailed below.

Chapter 5

Observed Behaviour

5.1 Data Examined

In all, six datasets are examined. Of these, four were scrutinized more closely, due to the range of scales covered. It has already been noted by [Mark 84] that a distinct change in behaviour occurs at the "grain" of terrain, the scale at which valleys and ridges exist. Above this scale, terrain variance does not grow very rapidly with sample spacing increase. However, below this scale, variance changes markedly with sample spacing. Thus, an upper limit of inter-point spacing in the order of kilometres has been imposed.

At smaller inter-point spacings, another change in behaviour is observed. An assumption underlying the application of the variogram technique used in this study is that terrain can be represented by a continuous surface. However, at large scales, surface material strength allows structures that occlude surfaces as viewed through a vertical orthographic projection. The actual scale at which this effect first manifests itself varies with environmental factors, but at sub-metre scales surface occlusion is liable to occur with increasing frequency, as noted in [Wall 89] and [Wall 91].

Availability of data together with quality also affected the choice of data. Accordingly, the study concentrated on the scale range of tens of metres to kilometres. A single dataset was also examined beyond each extreme of the chosen scale range, to investigate the effects of adverse surface behaviour on the algorithm.

Deadmann Butte is a 50 metre spacing grid DEM of size 512 by 512, originating from terrain in Wyoming, USA. The data were available in a floating point format, with height units of feet. The Montagne Sainte Victoire DEM has a 30 metre spacing grid of size 415

by 208, stereo matched from SPOT data of a region near Marseille, France. The data were represented by short integers, in units of metres. The largest dataset at this resolution covers the majority of the Federal Republic of Germany, digitized from NATO contour maps on a one arc-second grid of short integers in metre units, courtesy of DLR Oberpfaffenhofen. Finally, a 3 arc-second (approximately 100 metres) DEM covering the Alpine range (10 million samples) was analysed, courtesy of DLR Oberpfaffenhofen. The data were provided in short integer format, in units of metres.

Of the two extreme scale range datasets, the large scale range was represented by a Mojave Desert disparity map. The data consists of a 4 centimetre spaced grid with a resolution of 64 by 64, taken from helicopter data of the Mojave Desert. The disparity map should not affect analysis of the data, as the algorithm relies on height difference, rather than absolute height values.

To represent kilometre scale data, the USGS North American database was selected. This consists of height samples within a 8400 by 3360 grid with a 30 arc-second spacing, each height being represented by a short integer using units of metres. A significant proportion of the DEM covered sea regions, giving no relief information. To avoid this affecting the resulting variograms, all heights detected below or at zero metres were excluded from further calculations.

5.2 Variogram Behaviour

To determine the performance of the linear segmentation algorithm, it was initially applied to variograms of entire datasets. Despite improvements in the performance of the variogram algorithm detailed above, it would still have taken considerable processing-time to analyse the larger datasets. Therefore, the method was only applied to the Deadmann Butte, Montagne Sainte Victoire, and Mojave Desert datasets. Figure 5.1, figure 5.2, and figure 5.3 show the resulting variograms.

The linear segmentation algorithm was applied to each of the above variograms, to observe its action over wide ranges of scale. Table 5.1 details the linear intervals determined from the variogram data for the Deadmann Butte DEM. Table 5.2 details the linear intervals determined from the variogram data for the Montagne Sainte Victoire DEM. Finally, results

Figure 5.1: Variogram for the whole Deadman Butte DEM

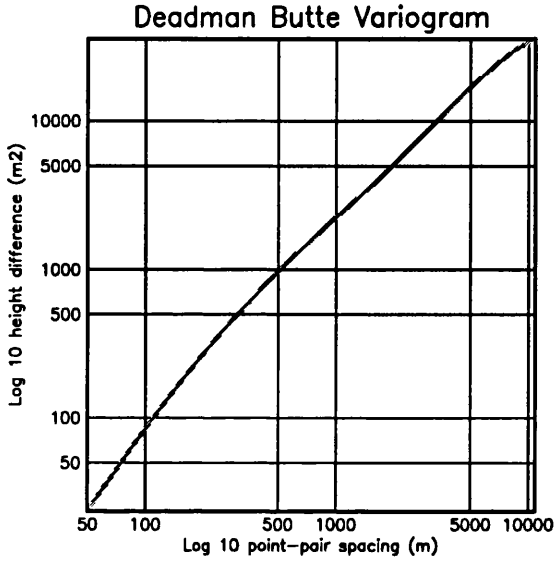


Figure 5.2: Variogram for the whole Montagne Sainte Victoire DEM

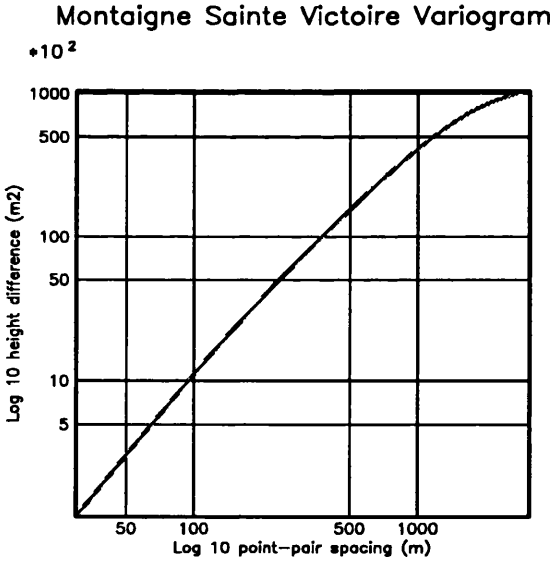
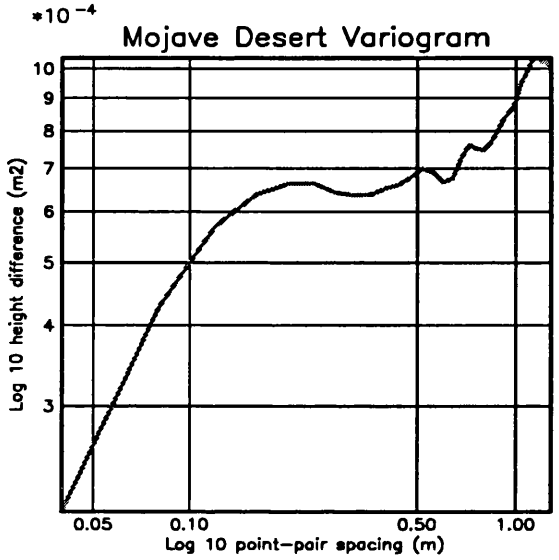


Figure 5.3: Variogram for the whole Mojave Desert disparity map



for the Mojave disparity map are given in Table 5.3.

As can be seen, for the two DEMs, the linear segmentation algorithm produces large linear intervals, together with a collection of smaller intervals in each case. Fractal dimension tends to increase with point spacing. However, if the variograms were to continue beyond half the length or width of the DEM in question, oscillating behaviour could be observed in the curve, leading to extreme fractal dimension values. For Deadmann Butte, the longest

Points	Dx min (metres)	Dx max (metres)	Fractal Dimension
152	50	7600	2.36
6	7600	7850	2.50
7	7850	8150	2.52
5	8150	8350	2.53
44	8350	10500	2.64
16	10500	11250	2.75
10	11250	11700	2.79
11	11700	12200	2.73
7	12200	12350	2.79
5	12550	12750	2.87

Table 5.1: Linear intervals determined for the Deadmann Butte DEM.

Dx min (metres)	Dx max (metres)	Fractal Dimension
30	120	2.13
120	210	2.16
210	300	2.21
300	390	2.24
390	480	2.26
480	570	2.28
570	5670	2.86
5820	6180	3.94

Table 5.2: Linear intervals determined for the Montagne Sainte Victoire DEM.

Dx min (metres)	Dx max (metres)	Fractal Dimension
0.04	0.60	2.82
0.60	0.72	2.62
0.72	1.04	2.66
1.04	1.16	2.48

Table 5.3: Linear intervals determined for the Mojave Desert disparity map.

interval possessed a scale range of 50 metres to 7600 metres, with a fractal dimension of 2.36. The Montagne Sainte Victoire segment ranged from 570 metres to 5670 metres, with a fractal dimension of 2.86. Several other linear intervals were returned for both graphs, but all covered an order of magnitude smaller scale range than the above two.

From analysis of the Mojave disparity map, the linear segmentation algorithm returned four linear intervals, of which the (logarithmically) longest occupied a range of 0.04 metres to 0.6 metres, with a fractal dimension of 2.82. This does not tally visually with a defined linear segment. Thus, within this level of complex behaviour, the validity of the segmentation algorithm breaks down. However, visual inspection of the variogram revealed that manual segmentation would achieve little in any case.

The algorithm does pick up linear intervals around the range of one metre spacings. Fractal dimension appeared to decrease with increasing inter-point spacing, from D=2.82 to D=2.48. The significance of this result should be tempered by the fact that this was the only dataset at this scale included in this study.

In the case of Deadmann Butte, a gradual rise from D=2.36 to D=2.87 was noted. A rise from D=2.13 to D=2.86 was observed withn the Montagne Sainte Victoire results, but the last linear interval exhibited a value for D of 3.94, resulting from the negative gradient of

the variogram. This might be due to the inter-point spacing exceeding half the width (but not half the length) of the DEM within the linear interval. Vertically separated point pairs would, in many cases, have straddled the ridge, and thus a low height difference would have been measured. This result, part of a trend from high variance values recorded at shorter lags, would produce a negative gradient.

5.3 Trends with Decreasing Sample Size

Unfortunately, the linearity variation present within large samples becomes far more apparent in the smaller tile sizes required for segmentation. Multiple scale breaks can occur, even within this limited range. Some of the linear intervals have gradients that give fractal dimensions greater than 3.0. For this reason, longest linear interval was rejected as a criteria for choice of linear interval, since this method produced a large number of anomalous D values in the resulting data. Best fit linear interval was also considered as a selection criterion. However, this also produced anomalous values. A significant point favouring usage of first linear interval is that this selection technique will, if possible, return linear intervals having common scale ranges. The overlap of sample ranges will produce statistics with greater commmonality, allowing the method to be used as a data comparison and segmentation technique. Thus, only first linear interval results are given here.

In order to observe what sorts of trends might characterize differing tile sizes, fractal images were produced using square tiles of sides 16, 32, 64 and 128 as grid elements. These resolutions were applied to both the Deadmann Butte and the Montagne Sainte Victoire DEMs.

Tile size was increased, so that more extensive phenomena might have some bearing on the overall results. However, at the same time, the minimum and maximum sampling intervals were left constant, and the first linear interval was always used in the calculation of the fractal dimension of the tile. Thus, although tile size was increased, the majority of inter-point data used for the final result remained within the same scale range. The resulting fractal dimension ranges of values are given in Table 5.4.

Several trends were noticed in the fractal dimensions of the tiles. Standard deviation increases in a non-uniform manner with increasing tile size, doubling within the range of sizes

DEM	Tile Size	Minimum	Mean	Maximum	Std. Deviation
Victoire	128	2.16	2.20	2.22	0.01445
	64	2.13	2.22	2.36	0.01182
	32	2.08	2.23	2.47	0.00794
	16	1.93	2.23	2.98	0.00757
Wyoming	128	2.24	2.29	2.39	0.00877
	64	2.22	2.29	2.44	0.00587
	32	2.16	2.31	2.72	0.00531
	16	2.04	2.34	3.14	0.00448

Table 5.4: Fractal dimension ranges determined for different sample sizes

considered. Mean fractal dimension decreases by an insignificant amount with increasing tile size, in the order of 1 to 2 percent in both cases. Additionally, the range of fractal dimension values decrease by an order of magnitude. For Montagne Sainte Victoire, a seventeen fold decrease was recorded, converging towards a mean of approximately 2.2. For Deadmann Butte data, a seven-fold decrease was recorded, converging on a mean of approximately 2.3.

5.4 Evaluation of the Row/Column Variance Calculation Method

Within [Klinkenberg 92], row and column fractal dimension results have been shown to yield differing values, due to the process by which the height data was created. Moreover, directional bias is also apparent within certain physiographic provinces of the data itself. With regard to the row/column algorithm applied in the current study, such results have a bearing, as extreme conditions may void the assumptions made in the formulation of the technique. To have confidence in the algorithm, extreme behaviour present within the data was created, and the results examined.

Limiting the scale range of samples considered for linearity behaviour may allow anomalous behaviour to be observed, due to the smaller sample sizes possible. Moreover, a relationship between the linear interval length and the resulting fractal dimension values may become apparent. To investigate this potential behaviour, both the row/column and all variance values algorithms were applied to 50 by 50 tiles covering the Montaigne Sainte-Victoire DEM. The results are tabulated in table 5.5 and table 5.6.

Maximum linear interval lengths of 5 to 21 were chosen. Fewer intervals were selected

Length	Max	Min	Mean	Variance
5	2.53	2.09	2.237913	0.006188
6	2.52	2.09	2.235488	0.006151
7	2.51	2.09	2.234722	0.006091
8	2.50	2.09	2.234417	0.006059
9	2.50	2.09	2.233624	0.006079
10	2.50	2.08	2.232932	0.006098
12	2.49	2.08	2.232493	0.006095
15	2.49	2.08	2.232529	0.006056
21	2.49	2.08	2.233060	0.005990

Table 5.5: Effect of constraining maximum linear interval length determined for the Montagne Sainte Victoire DEM. Data obtained using the all point-pairs algorithm.

Length	Max	Min	Mean	Variance
5	2.50	2.01	2.117373	0.010289
6	2.48	2.01	2.121020	0.010378
7	2.47	2.01	2.124207	0.010588
8	2.46	2.01	2.126961	0.010856
9	2.45	2.01	2.129353	0.011179
10	2.44	2.01	2.131219	0.011401
12	2.44	2.01	2.133312	0.011457
15	2.43	2.01	2.134369	0.011301
21	2.43	2.01	2.134673	0.011255

Table 5.6: Effect of constraining maximum linear interval length determined for the Montagne Sainte Victoire DEM. Data obtained using the Row-Column algorithm.

at the longer scale ranges. Variation in the parameters measured was less dramatic, due to fewer altered samples being included as the constraining maximum linear length approaches the maximum possible value. The maximum length chosen was 21, as in a 50 by 50 tile the maximum variance lag measured had a spacing of 25 samples (a greater spacing would result in anti-correlation artifacts within the variance values), and 4 additional samples were required to determine the start and end of a linear interval.

In both sets of results, highest fractal dimensions recorded decreased with increasing linear length. The decrease was more marked in the Row/Column results, possibly implying that the algorithm is more sensitive to change (fewer variance samples being used). Minimum fractal dimension values changed hardly at all. Within the all point-pairs results, mean and variance of fractal dimension changed slightly (a very small decrease with increasing maximum linear length was observed), whereas the Row/Column results exhibited a more prominent increase of about eight and ten percent respectively, over the range of maximum linear intervals used.

As linear interval length increases, short term variations in fractal dimension are likely to be averaged over. The highest values within the dataset occur at an artifact of the stereo matching process that contains uniform high frequency structures. The contribution of this data will decrease as the scale range increases. This may explain the decrease in extreme fractal dimension values observed above. The increase noted in the mean and variance of row/column data may be due to more general trends. As noted in [Mark 84] and [Klinkenberg 92], it is expected that variogram fractal dimension will increase with scale length, as larger (hundreds of metres to kilometre) lag variance changes less rapidly than shorter (metre to tens of metres) lag variance.

The trends observed are noteable, but the largest mean variation, of 0.02, and the largest maximum value variation, of 0.07, is still smaller than the observed difference between regions of low and high fractal dimension (a change of 0.5 within the German DEM). Thus, in the context of linear interval length sensitivity, the row/column algorithm may be used with justifiable confidence.

Anisotropic behaviour may highlight faults in the algorithm, as row and column data may not be properly integrated to produce a valid result. Accordingly, the row/column

Method	D	First Lag	Final Lag
Row/Column	2.057	1	4
All values	2.200	1	19

Table 5.7: First linear interval fractal dimension values for the entire Montagne Sainte Victoire dataset.

Method	Frequency	Mean D	Variance D	Max D	Min D	Mean L
Row/Column	32	2.14	0.0113	2.43	2.01	8.66
All values	32	2.23	0.00600	2.49	2.08	10.9

Table 5.8: First linear interval fractal dimension values for 50 by 50 tiles of the Montagne Sainte Victoire dataset.

algorithm was applied to the Montaigne Sainte Victoire DEM. This possesses a striking ridge running the length of the DEM, and thus should exhibit global anisotropic behaviour. A variogram of the entire DEM, together with variograms of 50 by 50 tiled segments, were analysed using both row/column results, and those obtained from all point pairs detected within a lag interval. The resulting global fractal dimension values are given in table 5.7, while the tile ensemble statistics are detailed in table 5.8.

In addition, orientation sensitivity was examined by creating artificial 50 by 50 point DEMs, each with a single dominant frequency in one direction only. Thus, no additional information was present to mask non-directional effects, while the resulting periodicity presented a severe test of the performance of the linear segmentation algorithm.

A single sinusoidal waveform was created, at a variety of angles. A wavelength of length 25 samples was first selected, such that full periodic behaviour might be represented by a large number of point pair samples for each lag, while ensuring that change occured over a significant length of the variogram.

It was verified for a single wavelength that results from the algorithm exhibited symmetry in the row and column axes of the DEM, as well as in the diagonals across the DEM. Thus, for all other wavelengths considered, orientations of zero to 45 degrees were examined in 15 degree increments. Additional results were obtained from data with wavelengths of 4, 8 and 16 samples. The resulting fractal dimension values are given in table 5.9.

From the results obtained, the row/column algorithm does appear sensitive to anisotropic behaviour. Where the dominant wavelength of periodicity is small with respect to the tile

	Orientation:	0	15	30	45
Wavelength:					
4	All values	2.98 (16)	2.95 (8)	3.35 (5)	3.34 (5)
4	Row/Column	11.86 (5)	2.64 (8)	2.73 (7)	3.56 (7)
8	All values	2.79 (15)	2.80 (16)	2.81 (16)	2.80 (16)
8	Row/Column	6.38 (9)	2.55 (15)	2.82 (13)	3.21 (13)
16	All values	2.20 (6)	2.18 (6)	2.18 (6)	2.16 (5)
16	Row/Column	4.24 (17)	3.13 (15)	1.92 (8)	1.87 (9)
25	All values	2.16 (10)	2.14 (9)	2.14 (9)	2.14 (9)
25	Row/Column	2.04 (4)	2.00 (4)	1.93 (13)	1.88 (14)

Table 5.9: Fractal dimension values for single frequency artificial 50 by 50 sample tiles, for angles of zero to 45 degrees and wavelengths of 4 to 25 DEM sample spacings. Linear interval lengths appear in parentheses.

	Orientation:	0	15	30	45
Wavelength:					
4	All values	0.999970	0.999679	0.999464	0.999304
4	Row/Column	0.939897	0.991564	0.979121	0.979049
8	All values	0.999859	0.999872	0.999866	0.999865
8	Row/Column	0.970240	0.998529	0.994448	0.993863
16	All values	0.999999	0.999998	0.999999	0.999998
16	Row/Column	0.990936	0.999520	0.999998	0.999995
25	All values	0.999999	0.999999	0.999999	0.999999
25	Row/Column	1.000000	1.000000	0.999999	0.999997

Table 5.10: Goodness of fit (r) of linear intervals from single frequency artificial 50 by 50 sample tiles, for angles of zero to 45 degrees and wavelengths of 4 to 25 DEM sample spacings.

size chosen, both algorithms fail to give sensible results. However, as frequencies approach the scale of the tile, more reasonable results are observed. This is probably due to the linearity algorithm breaking down, as high frequency data would only exhibit short trends, masked by the algorithms' requirement that at least three samples are examined. Poorer linear fits are observed in the high frequency results (see table 5.10), indicating a more irregular variogram. Concerning the implications for the analysis of real data, the high frequency results are unrealistic, as the ridge/valley spacing at which periodicity is likely is in the order of tens of lag spacings in all but the North American DEM.

Within the Montagne Sainte Victoire DEM, such extremes as exhibited by the high frequency test data are not apparent, even though the sample size is smaller and a high frequency artifact is present. When values of the tiled Montagne Sainte Victoire fractal di-

mensions were compared, values from the row/column algorithm differed from corresponding values from all point pairs by a maximum of 0.18 lower, or 0.05 higher. Given that the low frequency test results are more likely to represent terrain periodicity within the scale range of the German and Alpine DEMs, these were considered in evaluating the suitablity of the row/column method with regard to these datasets. The low frequency results, together with the Montagne Sainte Victoire data, exhibit a trend whereby the row/column results generally underestimate the value of the fractal dimension, and produce poorer linearity behaviour.

However, some extreme failure cases are demonstrated by the artificial data, where row/column values may far exceed the results obtained from using all point-pairs. It was noted that these instances occured when waveform orientations were approximately parallel with the measurement direction (i.e. when the discrepancy between perpendicular samples was greatest). Such circumstances would rarely occur in real data. Even if a single dominant frequency were present, other minor amplitude frequencies would lessen the impact of such behaviour. The experimental data tended to confirm this, as, of the hundreds of thousands of values recorded within the German and Alpine DEMs, only eight instances with a fractal dimension greater than three were observed.

Moreover, in the case of the Montagne Sainte Victoire data, the variance of fractal dimensions from row/column results is comparatively high. This implies poorer sampling and/or greater sensitivity to changing data. Linear fits evident in both sets of tiled results were good (r fits of 1.00000 in all cases), so the linearity algorithm behaviour appears acceptable. However, linear interval lengths returned from row/column results are in general shorter than those obtained from all point-pair data. This implies that greater variation is occuring along the length of the row/column variograms, which combined with the short interval length (smaller number of samples from which to obtain the gradient) could explain the greater degree of dispersion.

Results produced from the artificial data exhibit linear intervals with large variability. While no overall relationships were discerned, within the domain of the longer wavelengths (16 and 25 samples), a trend was detected. While row/column results varied with orientation, all the results derived from all point-pairs remained reasonably consistent, confirming the robustness of this method.

The high frequency results have a greater bearing on the analysis of the USA DEM. The results from this dataset do not exhibit the extremes of the test results, indicating that the failure cases in real data are rare. Nevertheless, the USA results should be treated with caution.

Unfortunately, the row/column algorithm had already been applied to the Alpine, German and USA datasets. The Deadmann Butte and Montagne Sainte Victoire DEMs provide case instances for the regional and anomalous behaviour of the fractional Brownian motion model. The larger datasets lend weight to the results, and should ideally be used. While the exact values obtained from the row/column algorithm are in error, the trends may still allow for general comparisons to be made, along the lines of high or low fractal dimension, to illustrate the regional behaviour of fractal dimension. As an example, values within the German DEM for low relief are in general 0.5 higher than the values encountered in high relief regions. This compares with the largest observed difference between algorithms caused by isotropic behaviour, a value of less than 0.2 in the case of the Montagne Sainte Victoire DEM. More extreme differences are apparent in the test data, but over the scales concerned, only a few failures occur which are gross overestimate. Such individual errors may be avoided by using general trends within the results.

5.5 Fractal Dimensions of Tiled Datasets

Terrain surface roughness has been related to the intuitive notion of surface roughness in [Pentland 84]. This was achieved by the visual evaluation of perspective views of artificial fractal height arrays. However, within the literature surveyed, this form of evaluation has never been applied to real terrain data.

To help to highlight any intuitive relationships between terrain and its local fractal dimension, the data sets available to this study were tiled, a fractal dimension being obtained for each tile area. As a result, an image of local fractal dimension was created for each data set, which could then be compared with other terrain measures, as well as visual interpretations of the topography. The Mojave disparity map yielded little data, due to the small size of the whole sample, and so is not detailed here.

The Deadmann Butte and Montaigne Sainte Victoire DEMs were divided into 32 by 32

overlapping tiles (tile spacing of 8 by 8 image elements), for each of which a variogram was calculated. Tiles were overlapped to increase the resolution of the output image, allowing visual comparisons to be made between the structure of the original data and the resulting fractal dimension behaviour. The tile size was chosen to give a comparable number of point-pairs to that used within the study detailed in [Mark 84]. Variance values were produced from all possible point-pairs, since the relatively small size of the data allowed this CPU intensive method to be applied. From these variograms, the first linear interval was detected, and used as the basis for calculating the fractal dimension of the local region. Figures 5.4 and 5.5 show the resulting images.

A wider variation was observed when the method was applied to the Alpine and German DEMs. These were segmented into 50 by 50 tiles, this size being selected to allow the complete covering of the DEMs while maintaining a large sample size. The row-column variance algorithm was applied to these more extensive DEMs, while the larger tile sizes still ensured that a large number of sample pair comparisons were carried out. The tiles did not overlap, reducing the number of tiles, and hence the processing-time required, but still giving a sufficiently detailed result to discern regions of distinct behaviour. The Alpine DEM and resulting fractal dimension image are illustrated in figures 5.6 and 5.7, while the German DEM and associated fractal dimension images are shown in figures 5.8 and 5.9.

To gain a more global picture for many terrain types, the above method was applied to the USGS USA dataset, using 40 by 40 sized tiles. No overlap was required, since the resulting fractal dimension image provides a sufficient resolution for trends to be observed. The result is shown in figure 5.10.

To compare the behaviour of the overall results, a table was created. This detailed mean, standard deviation, and range of recorded fractal dimension values for each dataset analysed using the tiling method. This table is reproduced in table 5.5.

5.6 Fractal Dimension and Other Terrain Measures

As can be seen, there is some limited correlation between the features in the DEM and the resulting fractal values. To test this further, we investigated the relationships that exist between fractal dimension and rudimentary terrain characteristics. To this end, values

Figure 5.4: Deadmann Butte images: Top left - Height intensity (range 1763 - 2302 metres, in steps of 2.105 metres, brightest value representing greatest height), Top right - Lambertian shaded, Lower left - First interval fractal dimension (range 2.16 - 2.72, in steps of 0.00218, brightest value representing greatest value), Lower right - First interval length (range 3 - 14, brightest value representing greatest value)

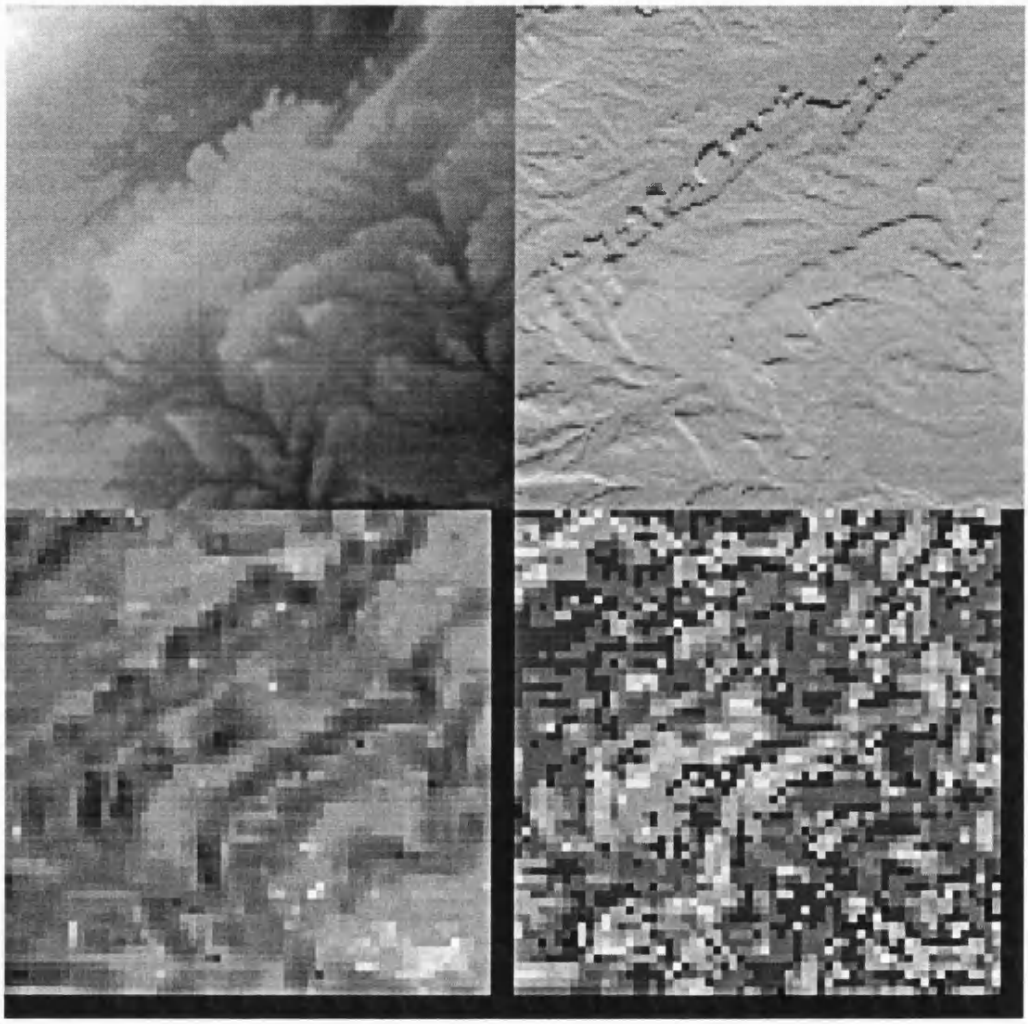


Figure 5.5: Montaigne Sainte Victoire images: Top - Height intensity (range 187.4 - 1016.83 metres, in steps of 3.239 metres, brightest value representing greatest height), Mid - Lambertian shaded, Lower - First interval fractal dimension (range 2.05 - 2.87, in steps of 0.00320, brightest value representing greatest value)

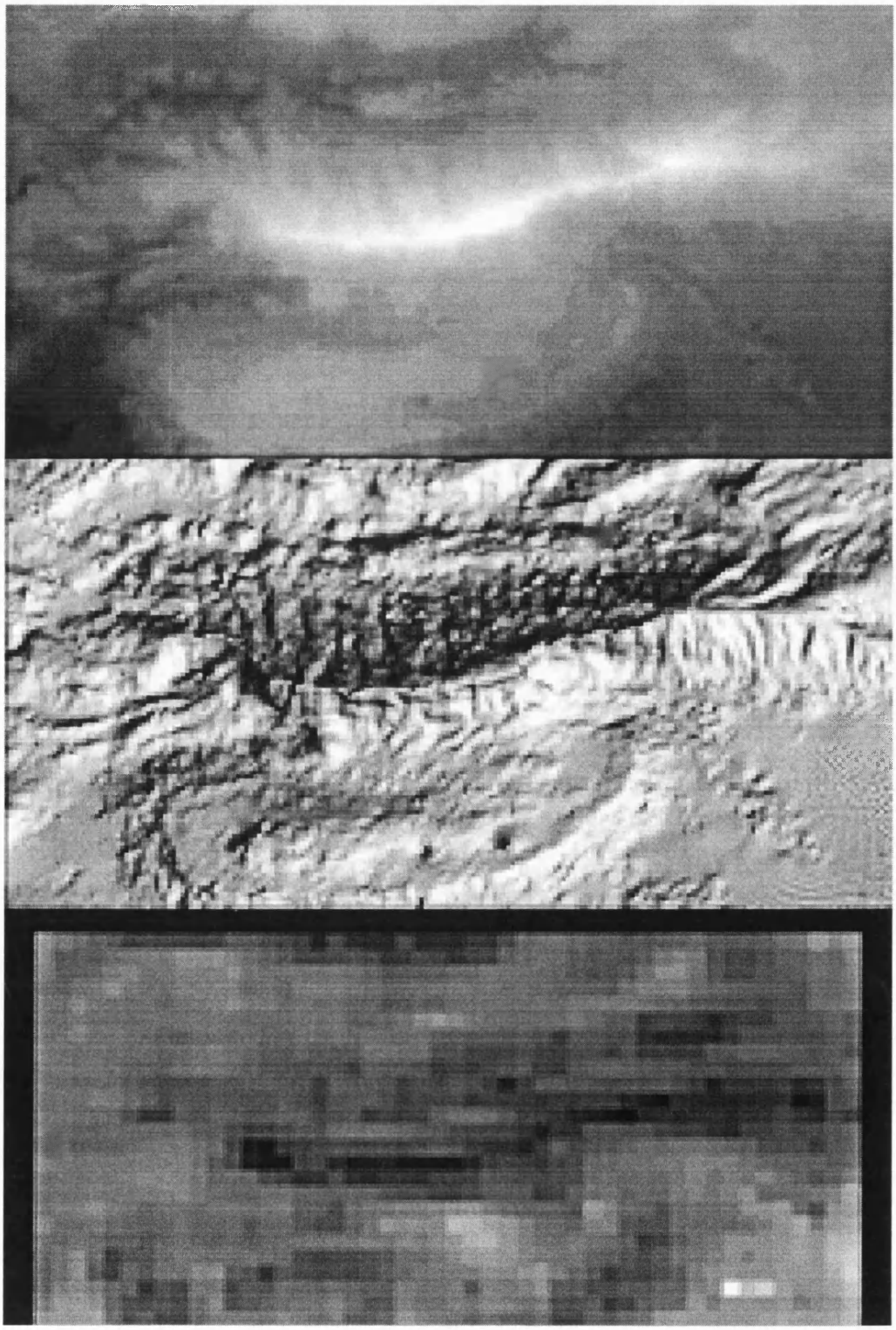


Figure 5.6: Bavarian Alps 3 Arc-Second DEM: Lambertian shaded image

Figure 5.7: Bavarian Alps 3 Arc-Second DEM: First interval fractal dimension image, histogram equalized (range 1.80 to 3.51, brightest value representing greatest value)

DEM	Minimum	Mean	Maximum	Std. Deviation
*Germany	1.06	2.18	5.29	0.20
Alps	1.80	1.99	3.51	0.13
Wyoming	2.09	2.31	2.82	0.09
Victoire	2.05	2.23	2.87	0.09
USA	1.83	2.19	4.60	0.37

Table 5.11: Fractal dimension ranges determined within DEMs (* row/column method used to obtain variance data)

Figure 5.8: Federal Republic of Germany 1 Arc-Second DEM: Aspect image

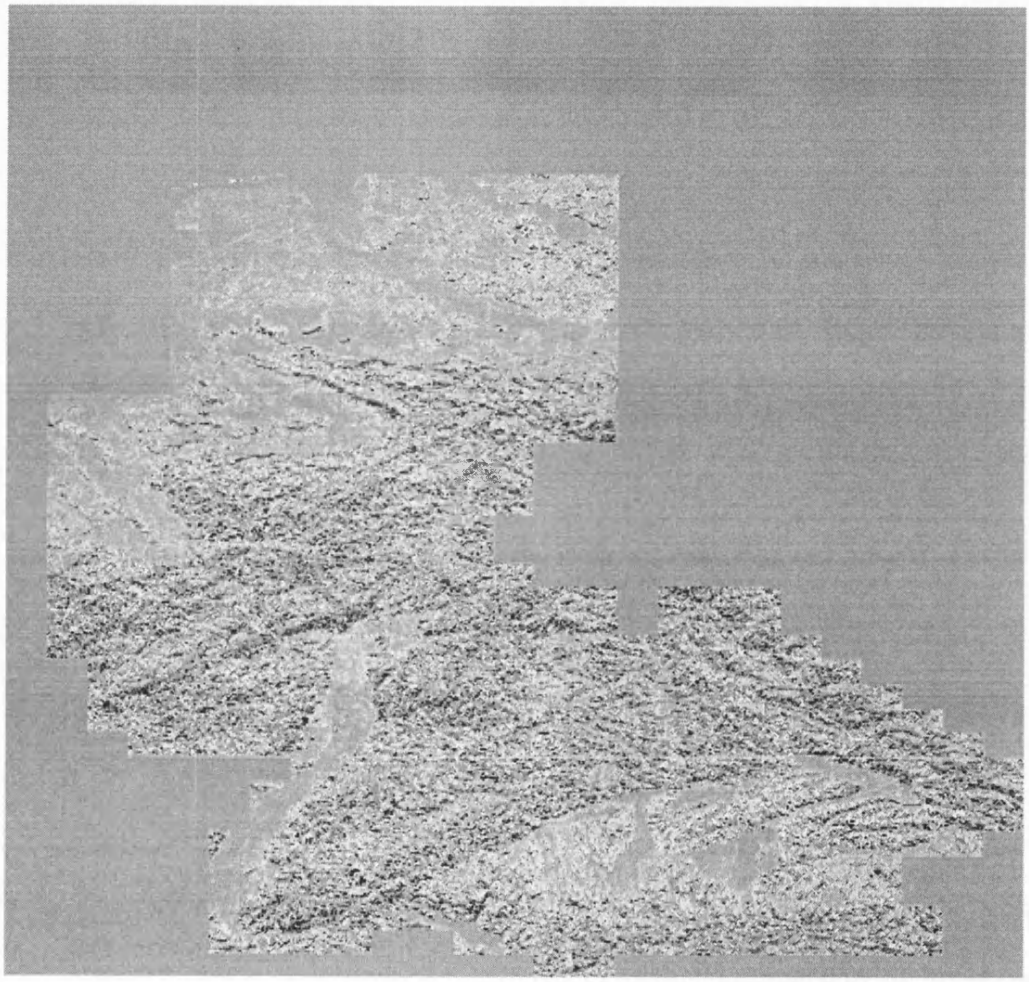


Figure 5.9: Federal Republic of Germany 1 Arc-Second DEM: First interval fractal dimension image, histogram equalized (range 1.06 to 5.29, brightest value representing greatest value)

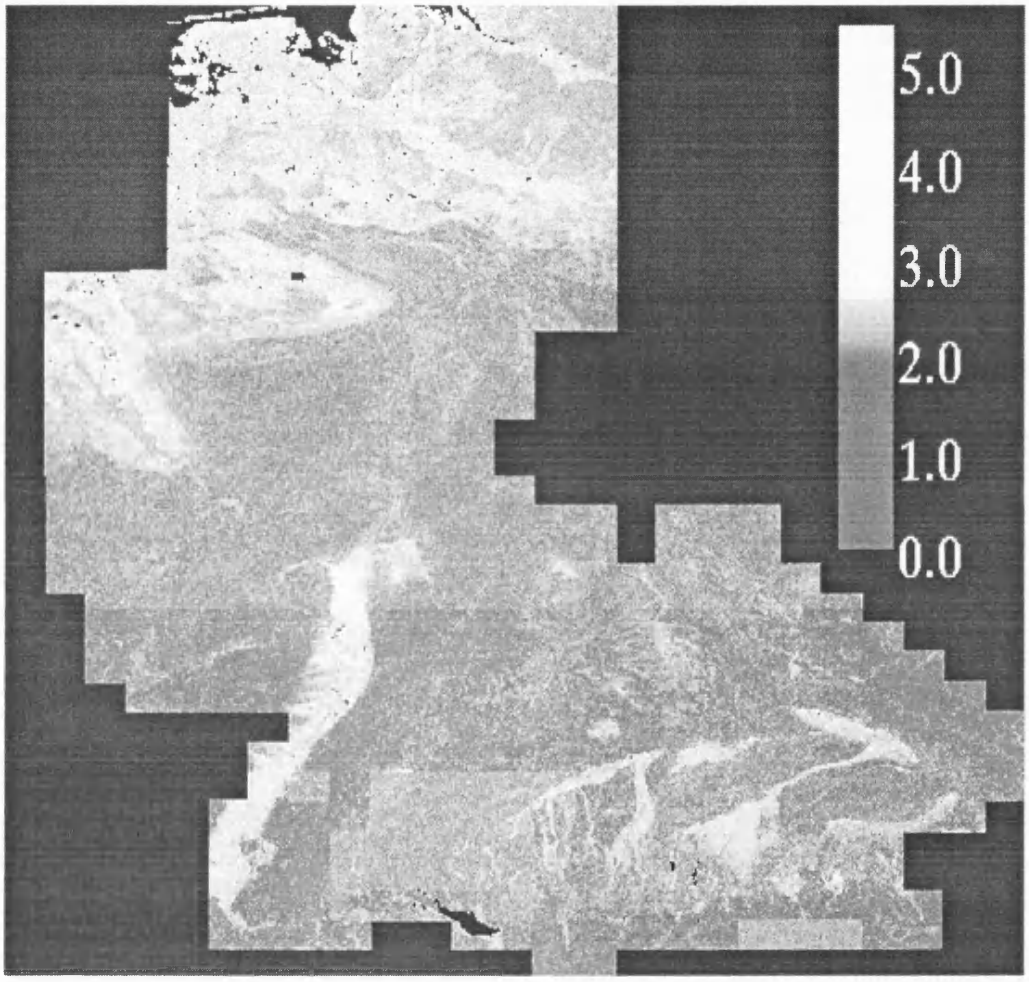
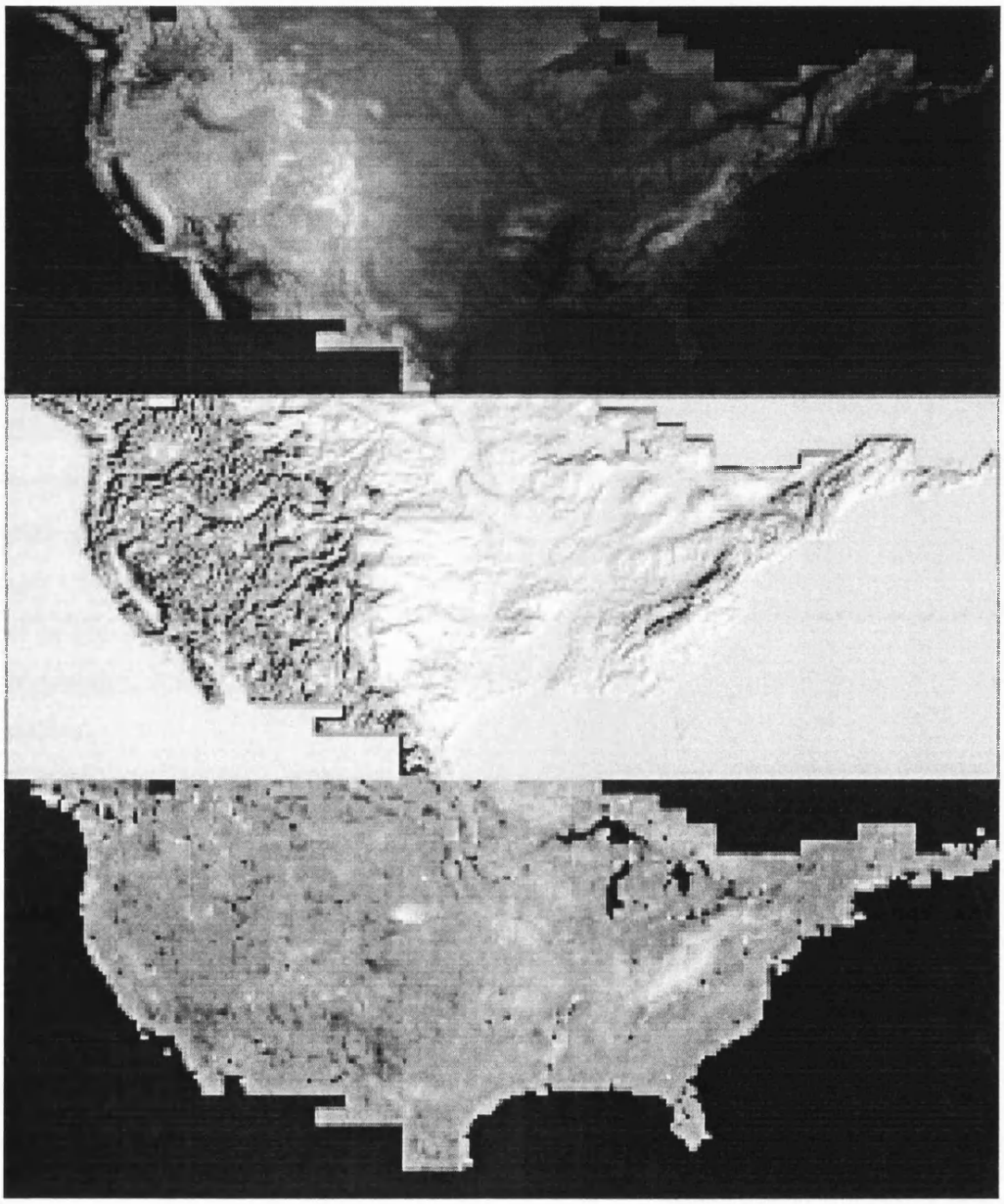


Figure 5.10: USGS USA 30 Second DEM images: Top - Height intensity, brightest value representing greatest height, Mid - Lambertian shaded, Lower - First interval fractal dimension (range 1.83 to 4.60, values in range 2.0 to 3.0 displayed, in steps of 0.00391, brightest value representing greatest value)



resulting from tiled images (32 by 32 sized tiles, non-overlapping) of fractal dimension of the Montagne Sainte Victoire and Deadmann Butte datasets were compared with images, degraded to the same resolution, of height, slope and aspect. The correlation coefficients of fractal dimension with each of the above images were calculated.

Results for the Deadmann Butte DEM suggest that limited correlation exists between these measures. For the Deadmann Butte DEM, correlation of dimension against height gave a value of -0.36. For dimension against slope, a value of -0.40 was recorded. Finally, for dimension against aspect, the result obtained was -0.18. The latter low magnitude value might have been recorded because the terrain features occupy a fraction of a single tile, thus smoothing over any observed effects.

A different picture results from Montaigne Sainte Victoire. Here, a single large feature dominates the DEM, much larger than the sampling tile size. Correlation of dimension against height resulted in a value of -0.44. For dimension against slope, a much more significant value of -0.62 was noted, whereas for the correlation of dimension against aspect gave a value of 0.01, effectively no correlation at all. Thus, both sets of results are consistent in terms of relative values of correlation, while Montaigne Sainte Victoire demonstrates stronger trends due to the dominant feature within the DEM. The absence of correlation with aspect in the case of the Montagne Sainte Victoire data implies that directional behaviour is not present, since the size of the central ridge should allow these effects to manifest themselves.

Correlation results are not given for other datsets, due to the unreliability of the method used to obtain fractal dimension values. However, observation of distinct regional behaviour in the fractal dimension images does indicate some association with terrain geomorphology.

Within sections of the USA DEM, differing fractal behaviour can be observed in distinct geological regions. Regions of the Great Plains exhibit higher fractal dimension values than the Rocky Mountains. Still higher values were observed within the Appalachians range. Regional behaviour is also apparent within the German DEM results, where the Rhine Valley and coastal region are highlighted by fractal dimension results.

DEM	Minimum	Mean	Maximum	Std. Deviation
Germany	0.173078	0.813670	1.000000	0.344601
Alps	0.494155	0.998086	1.000000	0.010518
Wyoming	0.986555	0.999654	0.999996	0.000782
Victoire	0.929434	0.999508	0.999999	0.002978

Table 5.12: R-Squared ranges determined within DEMs

5.7 Reliability of the Segmentation Algorithm

In order to evaluate the effectiveness of the linear segmentation algorithm when used in the context of smaller tile sizes, the R-squared method was applied to West German, Alpine, Deadmann Butte and Montagne Sainte Victoire datasets. The results are given in Table 5.10.

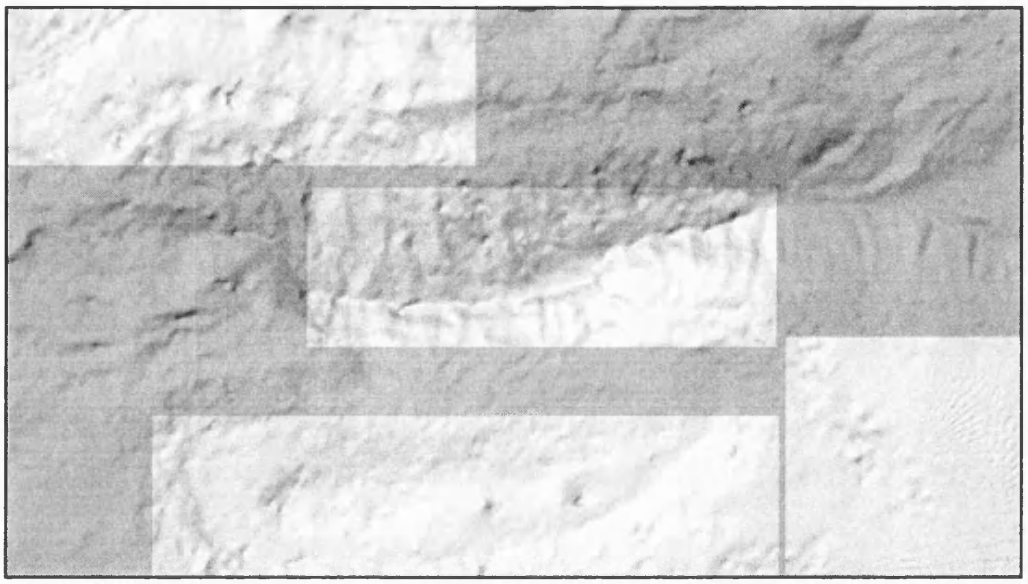
In the case of the last three datasets examined, the mean fit values were extremely good, with low standard deviations. However, the German dataset exhibited much poorer fit values. Prior to the usage of this dataset in the current work, it had been noted that large tracts of the DEM had been poorly digitized, and that low lying areas of the DEM contain very little relief information. This poor data might explain the resulting poor linearity behaviour.

5.8 Regional Behaviour of Data

As can be observed from the figures above, and to some extent within the correlations of fractal dimension with other terrain measures, regions of the DEMs exhibit distinct behaviour. Moreover, this behaviour is sometimes unexpected. For example, high ('rougher') relief can sometimes produce lower fractal dimensions than neighbouring low ('smoother') regions. To examine this behaviour in detail, the Montagne Sainte Victoire DEM was segmented visually on the basis of geomorphological behaviour, producing four smaller regions.

The most striking features - the central ridge and the lower crescent shape - were each represented by separate DEM segments of size 64 rows by 192 columns and 64 rows by 256 columns respectively. An area of high texture was selected to represent the region above the ridge, producing a DEM of 64 by 128 samples. Finally, a low relief area of size 96 by 96 samples containing an artifact of stereo matching was extracted from the lower right

Figure 5.11: Montaigne Sainte Victoire: Lambertian shaded with extracted areas highlighted



hand corner of the DEM, so that anomalous DEM representation effects could be evaluated. Figure 5.11 illustrates the regions used.

A variogram was calculated for each entire segment, using all point pair samples. The crescent region and area of high texture returned first interval fractal dimensions of 2.26 and 2.22 respectively. When measuring the flat artifact, a high fractal dimension of 2.48 was recorded, whereas the ridge produced a value of 2.16.

To examine more local variations, the data regions were segmented into 32 by 32 sample tiles, overlapped with a spacing of 8 samples between neighbouring row and column tile centres. Mean fractal dimensions from this data varied by less than one percent from the corresponding variogram results for entire segments.

Dispersion of the tiled sample fractal dimensions differed between the regions. The crescent and high texture regions exhibited fractal dimension variance of approximately 0.003. The ridge returned a value of 0.0015, half of that from the result above, whereas the flat artifact exhibited a variance of 0.02, virtually an order of magnitude greater than those of other regions. Thus, not only is the mean fractal dimension greater in this region, but the range of variation encountered is far higher than in other segments.

DEM Segment	Constrained	Mean D	Variance of D
crescent	no	2.24	0.00332
	yes	2.25	0.00236
flat artifact	no	2.48	0.0257
	yes	2.49	0.0275
high texture	no	2.24	0.00374
	yes	2.24	0.00342
ridge	no	2.16	0.00152
	yes	2.16	0.00141

Table 5.13: Fractal dimension results for constrained and unconstrained first linear intervals within 32 by 32 tiles

It has been suggested that the linear interval lengths encountered might influence the resulting fractal dimension values. Interval length does indeed vary between regions. Within the entire regional variograms, both the flat artifact and high texture data produce linear interval lengths of eight lag points, while the other two datasets return lengths of 15. This observation is reflected in the tiled data mean interval lengths, though the differences in these values are far less distinct. The ridge returns a much higher linear interval length than other datasets, followed by the crescent region. This is possibly due to the large features present in these segments allowing more uniform scaling of the variance with increasing lag. The other two datsets were selected for their more localised variation, and hence return shorter linear intervals.

To further test the influence of linear interval length with respect to calculated fractal dimension, all tiled values were recalculated for all regions. As above, linear intervals were calculated over all recorded lags. However, once linear behaviour had been detected, the number of lags used to calculated the gradient, and hence the fractal dimension, were limited to five samples. This is half the mean interval length encountered in the ridge segment, and is lower than the mean lengths recorded in any segment. The goal was to highlight any discrepancy due to change in slope within the detected linear interval. Results are given in table 5.13.

In all cases, mean recorded fractal dimension differed by less than one percent. Change in the dispersion of values was more marked. The crescent region exhibited a drop of 30 percent when linear length was constrained. All other regions showed far less change. Interestingly,

DEM Segment	Std. Dev.	Variance	Y-axis Intercept	Slope
crescent	-0.076	-0.044	-0.234*	*0.247
flat artifact	-0.063	-0.096	-0.526*	0.023
high texture	-0.491*	-0.479*	-0.674*	*0.258
ridge	-0.499*	-0.516*	-0.537*	*-0.242

Table 5.14: Regional correlation (r) of first linear interval fractal dimension results with other topographic measures: the standard deviation of the first lag, the variance of the first lag, the variance given by the intercept of the first linear interval in each variogram with log of the DEM grid spacing, and the variance of the slope of the first lag. * indicates a staistically significant correlation using the Student T-Test at a level of t=0.005.

the flat artifact region was the only dataset in which an increase in dispersion was observed, possibly due to the high frequency variation of the stereo matching artifact.

Results for variograms of entire regions, as well as the tiled results for all datasets, indicate some form of correlation between fractal dimension and distinct regional behaviour. However, the results are sometimes the converse of those expected from ([Mandelbrot 82], [Pentland 84], [Voss 88]). As an example, the ridge region returns the lowest fractal dimension, against the observation that it appears to be one of the visually roughest regions selected.

To test this relationship, height variance and standard deviation of single sample spacing lag was measured over identical tile grids to those used to calculate fractal dimension above. In addition, the residual variance was estimated from the first linear interval by calculating the intercept of this line with the log of the DEM grid spacing. Finally, variance of the slope was measured. This gave a measure of undulation, and acted as a check of the method, as the general expectation was that positive correlations with fractal dimension would be observed. This was repeated for each of the four segmented regions within the Montagne Sainte Victoire DEM, after which results were correlated with fractal dimension. Correlations are given in table 5.14.

The crescent and flat artifact regions exhibited low correlation coefficients with first lag calculated variance and standard deviation. However, a significant negative correlation is observed with residual variance. The two other regions, consisting of the ridge structure and the high texture area both exhibit significant negative correlations with all measures. In both cases, correlations with residual variance appear better than those with other measures. The

abscence of correlation with variance and standard deviation may be due to locally smooth data produced by the stereo-matching technique.

Thus, the strength of the relationship between percieved surface roughness and fractal dimension varies on a regional basis. Where the relationship is detectable, in areas of significant texture in the data above, correlation appears to be negative. Thus, over some data, there is an inverse correlation between percieved surface roughness and the associated fractal dimension. No positive correlation was observed with any of the measures of surface roughness. The analysis of slope variance with fractal dimension did return a positive correlation in three of the four test areas, confirming the validity of these results.

The absolute correlation values did not lie very close to one, indicating that a significant amount of information captured by either the fractal dimension or the simple surface roughness measures are not present in each other. Is the fractal dimension, being a more complex measure, capturing a superset of the information present in the simple surface roughness measures, or is there an overlap of interest between the techniques? The latter appears more likely, as fractal dimension captures relative scaling of the increments within the fractional Brownian motion model - and thus is not influenced by the magnitude of the variance (the basis for all the simple roughness measures).

5.9 Detection of Poor Terrain Data

Visual examination of the Montagne Sainte Victoire and West German DEMs has revealed that some regions are poorly represented in the datasets. Within the Montagne Sainte Victoire DEM, the lower left hand corner exhibits a series of high frequency patterns, some radiating from common points. Within the West German DEM, a survey of the data has resulted in the labelling of map sheet areas within which digitization has been poor. In addition, visual inspection of coastal data has revealed regular structures inconsistent with the expected topography.

The highest values within the Montagne Sainte Victoire fractal dimension image (figure 5.5) are located at the region mentioned above. Observing the fractal dimension image of the German DEM (figure 5.9) reveals a series of tiled patterns that contain higher values over their entire area, within the lower third of the dataset. While these regions do not

contain all the highest values within the image, they are distinct, and match the map areas designated as having been poorly digitized.

This implies that certain behaviour within a dataset will often result in increased fractal dimension values, and holds out the possibility of an automated method for quality assesment. However, as is evident within the German DEM, other behaviour might result in high fractal dimension values which saturate the image. Determining the cause of these extreme values would aid in the interpretation of the results presented above, and might possibly be used to distinguish between artifacts of the data representation and the behaviour of the source topography.

Chapter 6

Investigation of Anomalous Behaviour

6.1 Behaviour Contrary to Expectations

As has been shown in table 5.4 and table 5.5, the fractal dimensions recorded within areas of terrain have exceeded the expected range of $2 < D < 3$ (from [Mandelbrot 82], and [Voss 88]). Though these values may have resulted from artifacts of the measurement algorithm, such extreme D values have been noted in other studies using the fBm model. [Mark 84] recorded several variograms of areas returning values less than 2. Spectral analysis, also based on fBm, has been detailed in [Huang 89]. Two dimensional FFTs of Arizona topography have given values of $1.90 < D < 2.4$. Interpolation errors have been mooted as an influence on the results in the former paper. However, in the current study extreme D values were recorded outside areas noted for their poor quality.

Authors of two studies based on scaling spectra, [Hough 89] and [Huang 89], have questioned the effects of sample size on the resulting fractal dimension. Within [Huang 89], Fourier transforms were applied to linear profiles of the topography of Arizona. A 512 point profile was analysed, by applying an FFT to all the points. The data were then divided in two, and FFTs were applied to both halves. This was repeated until a profile length of 32 points was reached. An overall fractal dimension of 1.58 was recorded. However, when the two 256 point profiles were analysed, values of 1.59 and 1.24 were discovered. This diversification continued, the largest range being 1.31 to 1.86 for the 32 point profile strips. In all cases, the average fractal dimension of the profiles of a given length remained approximately

1.5. Results detailed within [Hough 89] tend to confirm the above, since it is suggested that fractal dimension of 1.5 is the limiting value obtained from summing a series of profiles.

In the current study, the analysis was applied to much larger areas, instead of profiles, and used the property of scaling variance derived from the fractional Brownian model. However, analysis of the Deadmann Butte and Montagne Saint Victoire DEMs exhibited similar convergence behaviour to that of [Huang 89]. Table 5.4 details behaviour as sample size is altered.

6.2 Non-Gaussian Behaviour

Fractal dimension has been shown to exceed the expected range of values, both in this and other independent studies, implying that that some terrain surfaces cannot be modelled adequately by fractional Brownian motion. In order to determine the validity of fBm as a model for all terrain behaviour, futher tests were carried out. Fractional Brownian motion was derived from the model of a Brownian process, so the increments of fBm should exhibit Gaussian behaviour ([Mandelbrot 82]). Accordingly, the Chi-Squared test ([Spiegel 72]), which measures the fit of data to a given distribution, was applied using a Gaussian model.

The test was applied to a 20 million point region within the lower left hand side of the German DEM, as well as to the Deadmann Butte, Sainte-Victoire and Alpine DEMs. As with the fractal dimension analysis, the datasets were tiled, Chi-Squared values being returned for each tile.

For comparison, the method was also applied to a two dimensional 256 by 256 height array, tiled into 32 by 32 areas, the data being generated using a recursive subdivision fractal algorithm from [Saupe 88] with Gaussian increments. The fractal dimension specified for this dataset was 2.5. Note that this value is higher than the mean fractal dimensions recorded above. However, this value was used to generate data over the entire scale range of a DEM, whereas the mean values had been compiled from first linear intervals. Examination of the large scale range variograms above revealed that fractal dimension increases with point-pair spacing, so the value of 2.5 was used as an average to represent this entire range.

Table 6.1 details the findings. In all but the fractal data the resulting fit of Gaussian behaviour to recorded distributions was poor. The artificial fractal data exhibited a Chi-

DEM	Minimum	Mean	Maximum	Std. Deviation
Germany	0.61	1174.77	52162.39	2038.34
Alps	0.07	1712.26	24033.67	2129.32
Wyoming	38.35	410.37	2988.52	338.00
Victoire	17.27	223.95	2234.47	229.25
Fractal	6.13	17.49	29.40	5.65

Table 6.1: Chi-Squared fit values to a Gaussian model within the DEMs - 18 degrees of freedom

Squared goodness-of-fit value 67 times better than the mean resulting from the DEM of the Federal Republic of Germany. Better fits were observed for some other datasets, but even in the best case, a comparison of the means revealed that the fractal data gave a thirteen-fold improvement over that of the Montagne Sainte Victoire DEM.

Within [Culling 87], tests were carried out to determine the Gaussian behaviour of the terrain profiles analysed. Visual comparison suggested the majority of profiles exhibited a good fit to a Gaussian distribution. However, given the comparatively small sample sizes used, and the limitation of analysis to a particular class of terrain, the results should be considered a special case, rather than indicative of general terrain behaviour.

6.3 Scaling Distributions

From above, poor results were gained from fitting a Gaussian distribution to the DEMs, as from attempts to fit Gamma and Log-Normal distributions to the data. Thus, fractional Brownian motion appears a poor model from this evidence, but, as has been shown earlier, terrain displays scaling properties within variogram results. By removing 'Brownian' from the terminology, $erf()$ in equation 2.1 can be replaced by any cumulative distribution, provided that the variance varies as a power law of the sample spacing. Rather than search through a multitude of distributions, a technique was selected which involved the comparison of distributions created from an area at different scales. To achieve this, a comparative method was applied independently within each of the tiles within each dataset.

For the first specified point-pair spacing, a 20 bin histogram of increment frequencies was compiled, as in the Gaussian method. However, instead of comparing this histogram with expected Gaussian frequencies, the data was treated as a reference distribution. A

DEM	Minimum	Mean	Maximum	Std. Deviation
Wyoming	11.56	70.49	596.28	37.88
Victoire	10.04	53.04	861.59	44.63
Fractal	1.15	29.41	1371.05	169.13

Table 6.2: Chi-Squared fit values of distributions within tiles within the DEMs - 18 degrees of freedom

second histogram was created using data from the successive point-pair spacing specified. The resulting Chi-Squared value relates to the goodness of fit of the second histogram to the distribution of the first.

The first histogram was treated as a continuous distribution by linearly interpolating the frequencies within the bins. This resulted in a crude representation of the distribution, so Chi-Squared values should be viewed as exaggerating the poorness of fit.

The method was applied to the two smaller datasets, Montagne Sainte Victoire and Deadmann Butte. The overall Chi-Squared behaviour is given in Table 6.2. Given the potential for poorer fit statistics inherent in the distribution representation, it is significant that the results exhibit an order of magnitude improvement over their Gaussian counterparts. The fractal data returns a poorer fit than that detailed in the Gaussian case, illustrating the inaccuracies within the method.

6.4 Data Representation Effects

Of the Gaussian fit results given above, it was noted that the datasets returning the poorest values consisted of quantized heights. Both the German and Alpine DEMs are derived from contour data, and are stored in an integer height format.

To investigate the bearing of such representations on the Chi-Squared values presented above, the reference fractal dataset was rescaled in amplitude, whereafter the height was rounded off to integer accuracy. The Chi-Squared test was then applied to the resulting integer dataset. The results for several different rescaling values are given in Table 6.3.

The Chi-Squared goodness-of-fit appears to diminish with lower amplitude range. This is equivalent to goodness-of-fit diminishing with relative accuracy of representation, since the amplitude range determines the amount of information rounded off from a value held within an integer based environment. Consider that, as the height resolution is reduced, the terrain

	Height		*Chi-Squared*			
Rescale	**Min**	**Max**	**Min**	**Mean**	**Max**	**Std. Dev**
2.0	217	320	80.79	98.68	127.66	9.35
1.0	108	160	283.16	422.48	2093.35	366.94
0.5	54	80	930.32	1115.80	2561.42	238.58
0.25	27	40	2338.98	2626.74	2942.71	132.83

Table 6.3: Chi-Squared fit values within a scaled integer fractal DEM - 17 degrees of freedom

will take on a terraced appearance, and therefore can exhibit a distinctly non-Gaussian distribution of height increments.

In comparing the results with the Chi-Squared values from terrain data, the mean and standard deviation were considered, but not the range. This is due to all the DEMs analysed presenting far larger numbers of tiles than those present in the fractal DEMs, resulting in a much larger range of values for a given standard deviation.

Upon comparing the statistics, the Deadmann Butte and Montagne Sainte Victoire DEMs exhibited a slightly better fit to a Gaussian distribution than the integer fractal DEM with a rescale value of 1.0. The statistics exhibited by the German and Alpine DEMs approximated those of integer fractal DEM to which a rescale value of 0.5 had been applied.

Can the poor Chi-Squared fits be solely accounted for by the integer DEM representation format? This question can be answered by comparing the amplitudes of the terrain DEMs with the fractal DEM which gives the closest Chi-Squared fit. The amplitude ranges of both the Deadmann Butte and Montagne Sainte Victoire DEMs are in the order of hundreds of metres. These DEMs have twice the height and width resolution of the fractal DEM, but the range of heights is still far greater than the amplitude range of 108 to 160 in the relevant fractal DEM. Moreover, the amplitude ranges for the region of the German DEM (a significant fraction of which is covered by the Bavarian Alps) and the entire Alpine DEM far exceed those of the relevant fractal DEM (54 to 80 height units). The areas of these DEMs are far larger, but there is significant local variation within the mountainous regions, certainly within a 50 by 50 sample tile area. In all the above cases, the amplitude ranges of the fractal DEM were far lower than those of terrain DEMs returning similar Chi-Squared fit values. Therefore, poor height representation could only result in a small part of the poor Gaussian behaviour observed.

It has been shown that degrading of height information does produce a markedly poorer fit to a Gaussian distribution. This might partially explain the poor statistics evident from the terrain data, due to the integer representation of heights. However, even when allowances are made for the crude comparison process, the disparity in information content between the comparable fractal and terrain DEMs implies that much of the poor fit results from reasons other than representation. The different methods by which the data was obtained (stereo matching, contour digitization) appear to have no obviously different effect on the final results. Thus, these values appear to reflect, to some extent, the behaviour of the original terrain.

6.5 Extreme Behaviour

Results detailed above give an overview of the performance of the algorithm. However, one of the most interesting aspects of the study was the extreme values of fractal dimension recorded.

In order to determine what conditions might bring about such values, a category of ten lowest and ten highest fractal dimension values for each dataset was devised. Other statistics detailed above were scrutinised to determine any common trends within the selected category. Analysis of the German dataset was limited to the region used within the Chi-Squared study detailed above.

For the sake of clarity, results given here are limited to those measures which displayed interesting variation and common behaviour within the above category. Hence, Chi-Squared results are not tabulated for certain datasets, there being little observable variation within the category, or with respect to the data as a whole. The significance values of extreme fractal dimension tiles remained constant at 0.2 for both the Deadmann Butte and Montagne Sainte Victoire DEMs, and so are not included in those tables. Behaviour of the German and Alpine DEM results is noteable, in that all the low fractal dimension values exhibit high significances, of 1.0 or 2.0, whereas the high fractal values are associated with low significances, of 0.2. Given the simplicity of this behaviour, it is not included in the tables below.

These significance values do not reveal any additional information about the extreme

Dimension	Interval	Fit	Min Var	Max Var	Chi-Sqr
1.813239	1 to 3	0.999080	213.420166	2846.716553	510.836731
1.819214	1 to 3	0.999088	172.316666	2268.836914	186.100510
1.829951	1 to 3	0.999293	146.417130	1886.909302	401.126495
1.832942	1 to 3	0.999101	190.635544	2436.227051	408.431366
1.835660	1 to 3	0.999320	231.859131	2951.826660	675.626892
1.836324	1 to 3	0.999741	86.753311	1108.907593	1417.340210
1.839062	1 to 3	0.999515	117.537300	1488.586182	412.227295
1.852824	1 to 3	0.999567	395.397888	4862.352539	576.461121
1.854115	1 to 3	0.999875	162.345230	2001.159180	1173.600586
1.855368	1 to 3	0.999812	44.584766	547.322144	1723.084473
3.193812	2 to 7	0.895255	0.189474	0.123736	4770.053223
3.007798	3 to 6	0.173078	36.705170	36.692497	5827.432617
2.968778	2 to 5	0.700527	0.456340	0.492999	5209.797363
2.962713	3 to 6	0.852744	5.381865	5.728531	1592.757568
2.934121	2 to 5	0.899836	0.263850	0.303344	4134.422852
2.932996	2 to 8	0.780182	0.085106	0.110221	5582.726074
2.923555	1 to 3	0.986832	0.006801	0.008079	2401.000000
2.923555	1 to 3	0.986832	0.006801	0.008079	2401.000000
2.909325	1 to 3	0.995218	14.321877	17.531122	1413.509277
2.866672	1 to 4	0.998520	0.029262	0.042624	4721.810547
2.873380	3 to 5	0.985626	1.251205	1.421697	3963.972900

Table 6.4: German DEM results for extreme fractal dimension values. Innacurate dimension values due to the measurement method may be present.

behaviour. The low fractal dimension results have a higher significance simply because they lie closer to the mean fractal dimension values, and hence the mean gradient values. This result does reinforce an observation that can be made of other results, namely that the distribution of fractal dimension is skewed towards the lower end of the range of values.

Other results are given in Table 6.4 for the region within the West German dataset. Alpine dataset extremes are detailed within Table 6.5, while Deadmann Butte and Montagne Sainte Victoire results are given in Tables 6.6 and 6.7 respectively.

Several trends can be observed in the tabulated results. The fit of the segmented lines to the curve, using the r-squared technique, exhibited much better behaviour in low fractal dimension regions than in areas of high fractal dimension, for all datasets. Differences of one to two orders of magnitude are common, such that a .98 fit in a high dimension region may contrast with a .9998 fit in a low dimension area. This might be in part due to the magnitude of the variance, which is unsurprisingly high in low dimension regions (a high rate

Dimension	Interval	Fit	Min Var	Max Var	Chi-Sqr
1.799880	1 to 3	0.998908	3091.478516	42391.070312	287.551788
1.803636	1 to 3	0.999432	4042.750977	55268.648437	683.292053
1.804442	1 to 3	0.999140	78.324211	1065.586792	1968.432495
1.815153	1 to 3	0.999103	1301.251709	17288.066406	1312.400635
1.815734	1 to 3	0.999068	3279.202148	43497.492188	236.905869
1.820297	1 to 3	0.999143	2062.871338	27110.791016	1215.468750
1.821610	1 to 3	0.999308	270.474365	3550.312744	2886.481689
1.822660	1 to 3	0.999266	4378.076172	57311.027344	431.781189
1.827070	1 to 3	0.999617	82.519920	1074.321899	715.219543
1.828199	1 to 3	0.998992	205.807251	2654.884277	8376.829102
3.509964	3 to 9	0.984510	30.379713	10.940397	6631.872070
3.160047	1 to 9	0.981234	0.008407	0.004454	9443.107422
3.098428	2 to 5	0.987297	0.013500	0.011288	9411.242188
3.060320	1 to 6	0.912286	0.018684	0.015492	9375.458008
3.039480	1 to 4	0.915919	0.017214	0.015421	9379.430664
3.003891	1 to 3	0.678874	0.030644	0.030426	9276.415039
2.981971	1 to 4	0.933156	0.036731	0.038527	9229.061523
2.980433	1 to 3	0.987834	0.033594	0.035106	9252.723633
2.979924	5 to 9	0.494155	0.037253	0.038752	9276.415039
2.978834	1 to 7	0.825226	0.019109	0.020556	9367.515625

Table 6.5: Alpine DEM results for extreme fractal dimension values. Innacurate dimension values due to the measurement method may be present.

Dimension	Interval	Fit	Min Var	Max Var	Chi-Sqr
2.086399	2 to 16	0.999730	45.956738	1940.182373	198.571259
2.124001	1 to 11	0.999562	33.828468	2079.547607	780.563843
2.125277	1 to 11	0.999594	32.241425	1978.740967	763.578430
2.125938	2 to 16	0.999844	92.557266	3330.846924	96.372009
2.126696	2 to 16	0.999748	43.020390	1551.683838	272.791138
2.128166	1 to 10	0.999559	35.590118	1835.537476	938.870850
2.132844	2 to 16	0.999838	24.383333	863.931030	445.397339
2.134481	2 to 14	0.999779	23.755320	649.713318	228.078323
2.134596	1 to 12	0.999654	32.794418	2239.998779	621.305603
2.137105	1 to 8	0.998933	9.185100	321.316071	231.364731
2.820068	5 to 8	0.997694	28.062695	33.340252	225.460846
2.753156	2 to 13	0.988483	9.250532	26.138044	244.121460
2.750664	3 to 13	0.991873	14.225455	31.991796	240.304184
2.739344	3 to 8	0.998279	12.350524	20.870682	259.816498
2.734863	4 to 14	0.992401	54.695343	113.660446	181.784317
2.713719	4 to 14	0.989984	35.260120	78.864616	438.689484
2.710602	2 to 8	0.997194	7.795745	18.076223	261.451416
2.706285	3 to 7	0.997474	11.368312	19.027840	279.078217
2.703056	2 to 8	0.997796	7.786170	18.374392	264.031372
2.697920	2 to 7	0.996657	7.662943	16.882076	288.302277

Table 6.6: Deadman Butte DEM results for extreme fractal dimension values

Dimension	Interval	Fit	Min Var	Max Var	Chi-Sqr
2.051306	2 to 16	0.999813	919.323730	44812.378906	112.417931
2.059366	2 to 14	0.999828	931.737915	34470.710938	51.211155
2.060585	2 to 16	0.999848	919.571655	43352.492188	106.749207
2.061679	2 to 16	0.999837	1122.306763	53215.921875	31.316360
2.062613	2 to 16	0.999868	989.055847	46709.160156	241.568939
2.062964	2 to 16	0.999861	984.173035	47117.878906	138.150146
2.064031	2 to 16	0.999879	991.466858	47116.917969	197.325195
2.065624	2 to 16	0.999867	400.538116	18964.263672	72.590553
2.067153	2 to 16	0.999844	965.260986	44245.367187	119.887390
2.067378	2 to 16	0.999865	1024.467896	48153.312500	75.880836
2.866877	4 to 9	0.929434	53.161541	69.276825	1464.377197
2.730331	5 to 8	0.996785	106.759422	138.452408	670.375122
2.726726	2 to 7	0.970531	13.652305	26.663595	231.396088
2.714793	2 to 7	0.955324	16.999468	34.157894	117.475861
2.643512	2 to 10	0.979504	32.538830	108.563858	1294.340210
2.629479	2 to 8	0.994887	12.080674	33.213539	343.032654
2.609717	2 to 9	0.962287	18.264008	56.765228	88.924744
2.608801	2 to 8	0.988364	16.436703	47.351494	158.333679
2.574014	2 to 7	0.998533	12.150000	35.088097	329.287628
2.569688	2 to 8	0.975963	17.125887	54.568920	114.604645

Table 6.7: Montagne Sainte Victoire DEM results for extreme fractal dimension values

of change of variance implies a low fractal dimension value, so variance values can possibly range to higher extremes in such areas). Most of the fit values found in extreme dimension regions were good. Thus, the linear segmentation algorithm appears to function well in many extreme cases.

As noted above, variance values appear typically high in areas of low fractal dimension, whereas low variance values are associated with high fractal dimensions. This may imply that the trend continues at sub-sample scales, since a comparison of variances found at the start of scale ranges reveals a large disparity between high and low dimension areas. The disparity must close, if a physically meaningful surface is to result, so the rate of change of variance has, in the long run, to continue within the broad rates observed above. Thus, while the exact linear behaviour may not continue to sub-sample scales, the trend indicated by such behaviour will probably exist at smaller sampling spacings.

One interesting aspect of the variance behaviour can be observed within the two larger DEMs. Within both the German and Alpine DEMs, high fractal dimension areas exhibit extremly low variance values. Recall that both these datasets are held to integer accuracy. However, the majority of the variance values selected by the above criteria return sub-integer results. Since the variance was obtained from an area, this implies that the majority of the area exhibits no relief, and probably consists of quantized steps of metre increments.

Chi-Squared values tend to exhibit better behaviour in regions of lower dimension, at least within the German and Alpine datasets. This could be due in part to the variance values found within these regions of distinct behaviour. Earlier, it was postulated that higher variance within an integer representation implied that more elevation information was present from which a distribution might be determined. Though this may be the case in such extreme categories as low and high fractal dimension, examination of the individual categories reveals limited correlation between variance magnitude and Chi-Squared fit. As an example, within the German data, a variance of approximately 36 metres squared is associated with a Chi-Squared fit poorer than any other value within the table, including entries exhibiting variances less than 0.01 metres squared.

6.6 Postulated Causes

Extreme results may have been caused by errors inherent in the row/column sampling technique. However, the existence of such values in other studies, together with the values detected in this study using alternative methods warrants further investigation.

Periodicity is a potential cause of such extreme results. Consider the behaviour of variance within such terrain. At inter-point lags approaching half the ridge spacing, variance will increase as normal. However, as lag approaches the ridge spacing, the rate of variance increase will diminish. Beyond this point, up until 1.5 times the ridge spacing, variance can actually decrease. This will result in a value of D greater than 3, given that the area variogram gradient ($2H$, from equation 2.4) is related to the fractal dimension by $D = 3 - H$. Moreover, as spacing increases still further, variance will increase at a high rate. This can in turn lead to a very low D value, possibly lower than 2.

[Mark 84] identifies the "grain" of topography as being the spacing of alternating ridges and valleys (i.e. a spacing of kilometres). At spacings smaller than the grain, terrain exhibits non-stationarity, fluctuating within a large range of values. However, at larger spacings, the effects described above come into effect. Height appears to be constrained within a narrow band of values, with the higher frequency ridges and valleys producing a large proportion of the measured variance. As inter-point distance increases, the surface appears increasingly flat, at least until the effects of continental masses make their presence felt.

Note that the fractal dimensions obtained from the German and Alpine DEMs exhibit greater anomalous behaviour than the Wyoming and Marseilles DEMs. Indeed, the average fractal dimension of the Alpine dataset is less than 2.0. Both the German and Alpine DEMs were derived from contour data, and close examination revealed tiers of heights in some of the flatter regions. This behaviour may be dominant in the extremely high fractal dimension statistics detailed above. Such quantization might have a bearing on the resulting values. However, this type of behaviour can be observed in terrain, through agricultural structures, and due to tectonic and discrete water erosion effects.

Unfortunately, this implies that although the value for D might remain within the legal range of $2 < D < 3$, the contribution of periodicity or other factors can result in a change in variance that does not correctly represent the topography present. Periodicity can occur due

to tectonic formations, within sand dune patterns, or due to man-made structures, as well as within artifacts resulting from the measurement method. Thus, fBm might not be capable of modelling all terrain types, and, since such terrain types can form a minor contribution to any area measure, this imposes limits on the evaluation of any terrain region.

6.7 Artificially Generated Extreme Values

In order to gather some additional evidence supporting the existence of extreme values of fractal dimension reported above, some of the characteristics that might result in such extremes were artificially created.

The first experiment involved the generation of a sinusoidal DEM. To replicate the experimental conditions under which anomalous data was produced by the large DEMs, a 50 by 50 sample grid was generated, and was treated as a one arc-second DEM. A waveform with a period of 33 arc seconds (approximately 1 kilometre) was the sole contribution to the amplitude of the DEM. The method used to analyse the data was virtually identical to the process applied to German DEM, the only difference being that the DEM was positioned at latitude zero such that row and column dimensions were approximately equal.

Only one linear interval was returned from this analysis. It ranged from bin one to bin four (from 40 metres to 120 metres lag) and returned a fractal dimension of 1.940003. Close examination of the variogram revealed a sharp rise in variance up to bin interval five, followed by oscillating behaviour involving similar values to those within bins four and five.

In a further experiment, the effect of poor sampling was investigated. A 50 by 50 sample grid was generated, but in this case, the waveform generated had a frequency of 0.91 the sampling interval of the grid. In this case, the variance was drawn from all point-pairs within a uniform grid. This method was selected in order to observe whether it was possible to create anomalous values when making best use of the data present. The resulting linear intervals are given in Table 6.8.

Several other frequencies were analysed, with similar or less extreme results. However, the goal of the experiment was simply to demonstrate that certain datasets can predictably produce extreme fractal dimension values. These results show that both extremely high and low fractal dimension values can result from the measurement techniques applied in the

Dx min	Dx max	Fractal Dimension	Fit
1	6	3.12	0.999987
6	13	3.15	0.999995
14	21	2.82	1.000000
21	24	2.16	0.999999

Table 6.8: Linear intervals determined for a poorly sampled artificial DEM.

study. Moreover, it has been shown that the linear interval selection criteria justified earlier present no guarantee of a certain range of fractal dimension values.

6.8 Counter-Intuitive Smooth and Rough Comparisons

A more subtle behavioural characteristic can be observed in figure 5.9. Large structures, associated with hydrological activity, are clearly visible. The Rhine valley is highlighted in the lower left of the image, and the flat coastal region in the upper third of the image is distinct. This illustrates an unexpected result - namely that the 'smooth' terrain regions within the DEM exhibit fractal dimensions that are significantly higher than other, 'rougher' terrain. The Bavarian Alps, at the base of the image, are virtualy indistinguishable from surrounding terrain, and return fractal dimensions far lower than those of the Rhine valley. The Sainte-Victoire and Deadmann Butte data sets support this result, while the Alpine data set, composed almost entirely of mountainous terrain, exhibits low D values overall. In addition, other evidence of anomalous behaviour can be found in independent studies.

As has been noted in the literature survey, studies prior to the current research have detailed results which, when compared with each other, yield interesting conclusions. Analysis of length measures within [Goodchild 82] resulted in a range of 1.11 to 1.31 for a range of contour altitudes (a value of 1.8 was recorded for lake outlines). In a later study, contour data was examined by [Culling 87], and areas selected were limited to "smooth" soil covered regions within Southern England. However, the range of values, 1.059 to 1.532, encompasses the range of all contour values in the earlier study. This is despite there being no mention of terrain smoothness/roughness being a selection criteria in the earlier study.

This pattern is repeated in behaviour derived from variograms. Comparison of results from [Mark 84] and [Culling 87] reveals that, for shorter lags, the values reported

in [Culling 87] lie within the middle of the range of values given in [Mark 84]. For large lags, the range of values of the studies are almost identical.

Of the variety of results detailed in [Burrough 81], two were obtained by use of the variogram method, and both gave the same value of D=1.5. Given that one of the datasets represented the surface of an airport runway ("one of the smoothest surfaces imaginable in a landscape"), the result demonstrates that surface roughness and fractal dimension need not always be highly correlated.

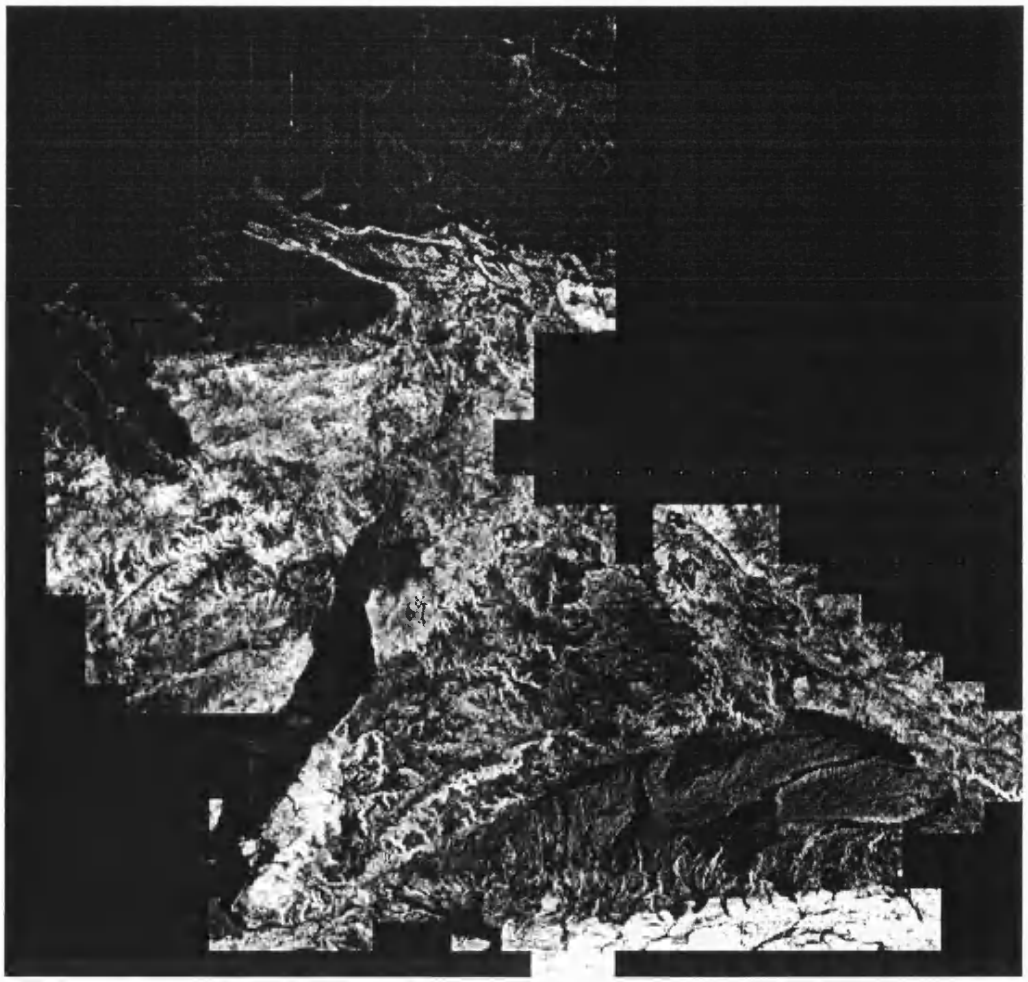
Comparing these results, it is difficult to distinguish smooth terrain from a variety of terrain types. Moreover, in the current larger study, there appears to be an inverse correlation between fractal dimension and roughness measures (first lag variance and standard deviation of height), as well as an inverse correlation with empirical roughness expectations. This seems to contradict the expectations presented in a variety of literature ([Mandelbrot 82], [Pentland 84], [Voss 88]).

Such behaviour may result in studies producing conclusions contrary to expectations. Results detailed in [Elliot 89] showed the application of variance scaling to profile data collected from a glacier environment. The date of surface formation could be estimated, and related to the fractal dimension. Fractal dimension appeared to increase with age over a range of three decades, a "counterintuitive result" given that this quantity was applied as a measure of surface roughness.

A resolution of this discrepancy may lie in the magnitude of the variance. Note that most artificial fractal terrain exhibits a significant height amplitude. It would be perfectly valid to generate a fractal terrain where all amplitudes were small with respect to the length scales of the area. Such a surface would appear smooth to the eye, even though it might have a high fractal dimension. Within the fractional Brownian motion model, the relative scaling of the increments is considered, rather than the magnitude of the height variation present. Figure 6.1 illustrates the variance of the first lag used (40 metres) in the computation of the associated fractal dimensions (reduced in range to enhance feature recognition). Subjectively, this appears to bear a greater resemblance to expected surface roughness than the fractal dimension image. Thus, fractal dimension alone does not uniquely characterize surface roughness of a DEM. Only in combination with the magnitude of the variances or frequencies

present can one approach a quantitative description.

Figure 6.1: Federal Republic of Germany 1 Arc-Second DEM: First interval variance image (range originally 0.0 - 19155.44, constrained to 0.0 - 100.0 metres squared)



Chapter 7

Conclusions

7.1 Overview

A method has been devised and implemented, whereby the parameters of a fractal model can be evaluated for the covering of a terrain area. The techniques allow for the unsupervised completion of this task, and performance improvements make possible its application to large data sets. These improvements are outlined below, together with an evaluation of the extent to which the algorithm returned usable results.

A wide variety of results have been obtained by the use of these techniques. From these, a series of conclusions concerning the behaviour of fractional Brownian motion have been reached. In some cases, evidence from independent sources has supported these conclusions. The observed behaviour, together with possible causes, is summarised below.

This greater understanding of the relationship between the model and terrain behaviour has undermined some previously envisaged applications, while suggesting new applications of the techniques. The final section accounts for the rejection or acceptance of certain applications, while outlining a methodology by which such techniques could be applied.

7.2 Performance of the Algorithm

The goal of constructing a method for automated measurement, returning the fractal dimension of terrain has been attained. The fractional Brownian motion model was selected, applied using variogram analysis techniques, due to its popularity, both with regard to terrain measurement and terrain interpolation (an envisaged application). A linear segmentation algorithm has been adapted to return valid regions within the variogram of distinct scaling

behaviour.

The time performance of the algorithm has been improved, from $O(N^2)$ to $O(N^{3/2})$, allowing the application of the method to a variety of data sets. Implementation and hardware improvements also aided in decreasing the amount of CPU time consumed. In a timed study, analysis of the Deadmann Butte DEM decreased from 36,000 to 240 CPU minutes.

Unfortunately, it has been determined that the row/column technique can produce extreme errors, though such cases are found to be rare in real data. More significantly, the method tends to underestimate the fractal dimension of most terrain samples in the evaluation detailed earlier. This may explain why the mean fractal dimension of the Alpine dataset falls below 2.0. This does not preclude the use of the results in a comparative study where only high and low classes of values are considered, but the reliability of individual values is suspect.

Splitting the functionality into two sections, the CPU intensive variance calculation and the linearity analysis, provided the ability to perform high cost operations once. Results could then be analysed rapidly and repeatedly, important when developing a new technique.

In practice, the performance of the linear segmentation algorithm was acceptable. With regard to the Alpine, Deadmann Butte and Montagne Sainte Victoire DEMs, mean r-squared fits suggested extremely good linearity. The technique failed when presented with the complex behaviour of sub-metre terrain, but linear behaviour was not detected by visual examination of the data. Poorer fit results were returned from analysis of the German DEM. It was hypothesized that this was due to poor data being present. While this connection has not been proven, independent examination of the DEM did reveal large tracts of poor data.

7.3 Observed Behaviour of fractional Brownian motion

Good linearity was observed in the range of tens of metres to kilometres. Both variograms computed at this scale were dominated by large single linear intervals, with some attendant smaller intervals - all an order of magnitude shorter. The longest linear intervals extended for one to two orders of magnitude scale range. Fractal dimension decreases approaching metre scales from centimetre scales, and from kilometre scales to tens of metres. This might imply a minimum of fractal dimension at metre scales.

As variogram sample size is decreased, and the number of variograms is increased, fractal dimension ranges increase, until the expected range of $2 < D < 3$ is surpassed. Images resulting from the application of the method to DEMs show visual correlation between fractal dimension values and the structure of the source terrain. This is confirmed by the correlation coefficients of fractal dimension with slope and height measures. Significantly, both these comparisons exhibit an inverse correlation. Increasing fractal dimension has been associated with increasing surface roughness, but in the current study, increasing slope and height (associated with regions of increasing roughness) is correlated with decreasing fractal dimension. Moreover, an inverse correlation between fractal dimension and surface roughness measures (first lag variance and standard deviation of height) has been observed over different terrain types. Correlations are significant, but not close to -1.0, implying that the inverse correlation is a trend, within which a complex relationship exists.

The fractional Brownian motion model involves the scaling of a distribution. As the name implies, this model is derived from behaviour due to Brownian motion, and therefore should exhibit Gaussian behaviour. However, examination of terrain samples failed to reveal the presence of this distribution. Such adverse behaviour could not have resulted from the analysis technique, which returned excellent results for an artificially-generated Gaussian DEM. The poor results may have been due to the representational accuracy of the terrain. However, conversion of the Gaussian DEM to integer format with a given accuracy still resulted in comparatively better fit values than for real terrain.

By divorcing the model from its roots, any distribution can be scaled to produce the desired behaviour. Rather than test many standard models, distributions were produced from the terrain data at the shortest sampling interval, and compared with distributions gained from larger sampling intervals. Distributions were represented by 20 bins, and so should have given poorer fit values than the Gaussian distribution values above (the Gaussian distribution being represented by 800 bins). In spite of this, fit results were a significant improvement over the Gaussian counterparts. Therefore, it was concluded that most terrain exhibited non-Gaussian characteristics.

A noteable discovery was that the technique seems to reveal artifacts caused by poor DEM creation. Such poor behaviour was not readily apparent in slope, aspect or variance

behaviour other than by visually examining shaded DEMs at a magnification of close to one. Some of the poor behaviour had been noted from slow, detailed examination, and, in the case of the German DEM, map sheets of poor quality had been noted by the originators of the data. This information was visually correlated with high fractal dimension values in the German and Montagne Sainte Victoire DEMs.

However, these results did not always represent the highest values within the DEM, which were sometimes associated with natural structures. Examples include the Rhine Valley, German coastal regions, and various lakes. It was noted that all these regions contained poorly represented terrain (low variation relief, held to integer accuracy). Thus, high fractal dimension values appeared to result from poor terrain representation, which stems from a variety of causes. Unfortunately, as has been shown in the case of the Appalachians range (North America), certain naturally occuring structures that contain periodicity (tectonic effects, sand dunes) can also produce such behaviour.

Surveys of the larger datasets produced fractal dimensions outside the expected range ($2 < D < 3$). This is in spite of the fact that far larger samples were used than those that produced anomalous values earlier in this study. Errors may have been introduced by the processing techniques applied, but it is uncertain as to how many of the extreme values are accounted for by this. High values in the German and Alpine datasets almost invariably indicated very low relief changes, apparently composed of tiered heights (due to the integer representation). The linear segmentation algorithm appeared to function well in most cases - mean linear fit values recorded were good for most of the data sets analysed. However, examination of the extreme fractal dimension results indicated that a minority of cases produced extremely poor fits.

Low fractal dimension values were associated with extremely rapid rises in variance. A conjecture that periodicity could cause such results was tested by analysing an artificial data set. The results demonstrated that fractal dimensions less than two could be produced by certain surface behaviour. Further tests showed that fractal dimensions greater than three could result from poor sampling. While these results highlighted the limitations of the measurement technique, it should be noted that both data sets could represent valid terrain surfaces. In addition, the scaling of frequencies or variances shown could represent

both rapid rises in true terrain variability, or the behaviour due to the presence of periodic structures. Thus, the fractional Brownian motion model cannot be used to represent all terrain phenomena unamibiguously.

The mean, limits and distribution of the fractal values recorded show a bias towards the lower end of the range of values. This is in line with fractal dimensions estimated for the Earth's surface. Mean values recorded for most of the data sets ranged from about 2.2 to 2.3. One exception, the Alpine DEM, exhibited a mean value just lower than 2.0. This might be due to the underestimation of fractal dimension by the row/column measurement techniqe, as well as due to the inter-point spacing approaching the scale of terrain ridge/valley structures, since periodic structures have been shown to produce such behaviour in this study.

It has also been shown that, contrary to expectations, fractal dimension does not uniquely determine the observed surface roughness of terrain. In a large proportion of the regions studied, areas with high observed surface roughness return low fractal dimension values and vice versa. It has been noted that, in the case of the West German dataset, measured local variance appears to represent a much closer relationship to observed surface roughness. From the comparison of other studies, it was not possible to distinguish between smooth terrain and arbitrarily selected terrain on the basis of fractal dimension results. This has important consequences for the multitude of fractal terrain generation algorithms, which have generally treated fractal dimension as a roughness parameter.

The misnomer of surface roughness estimation might have arisen from comparisons of generated fractal terrains. Within [Mandelbrot 82], chapter 28, a fractal dimension of 2.5 is considered too rough to model the surface of the Earth. It is proposed that lowering the fractal dimension will produce a "less unsmooth" relief. This fallacy is also implied in [Voss 88], where generated surfaces of increasing fractal dimension are illustrated. The surface with $D = 2.8$ is described as "unrealistically rough". As postulated above, the recreation of these surfaces with amplitudes of close to zero would produce surfaces with a smooth appearance, but the same fractal dimension. To avoid confusion, roughness and smoothness descriptions should not be attributed to fractional Brownian motion, at least in the context of terrain surface analysis.

Visually categorised artifacts, tiered heights in regions of low amplitude (and thus, in

an integer environment, poor information content) and artificially created poorly sampled data all exhibit high fractal dimensions in their first linear intervals. Artificially generated periodic behaviour exhibits a low fractal dimension within the resulting first linear interval. From [Mandelbrot 82], P353, values of $H < 0.5$ (i.e. $D > 2.5$) indicate antipersistence, implying that correlation between local increments is negative. This tends to borne out, at least in the case of heights quantized by integer representation. Increments in regions returning values of $H > 0.5$ exhibit persistence. In the case of values of $D > 3.0$, correlation might be too high, as demonstrated in cases of periodicity.

From above, three mis-conceptions appear commonly within fBm terrain decription literature. First, linearity behaviour is far more complex than any prior papers have acknowledged. Initially, the model implied that terrain scaled uniformly over an infinite range of scales. Subsequent visual analysis in [Mark 84] revealed a series of scale ranges, one up to the order of kilometres, and at least one beyond this spacing distance. The current study applied a more stringent automated linear segmentation algorithm, which has produced many linear intervals within the decametre to kilometre scale range. However, study of the variograms reveals that these intervals are not absolutely straight. It can be concluded that there are a large number of approximately linear intervals, a few stretching over order of magnitude scale ranges, while the majority are of insignificant length.

The second mis-conception is that surface roughness is associated with fractal dimension. This may be true in certain circumstances, but has been shown to be false over the majority of the data sets used in this study, as well as by the comparison of other studies. The final mis-conception is that terrain increments generally exhibit Gaussian behaviour. This has been shown to be untrue in the majority of tile segments examined in the study.

Development of applications of the model has been rapid in the area of computer graphics. In the original paper proposing the application of fBm to terrain description ([Mandelbrot 75]), a principal justification was the visual similarity between massive computer graphic simulations and real terrain. This tradition has continued, with computer graphics papers rarely citing studies of real terrain. Conversely, terrain studies have often cited computer graphics papers as a justification for the application of the model. As a result, dissemination of the behaviour of fBm has relied on opinions derived from simulations of the model, as well as

actual measurement and validation of the terrain behaviour. Thus, aspects of the model have been repeatedly divorced from the realities of topographic descriptions, giving rise to the mis-conceptions detailed above.

So, what information does the fractal dimension of a fractional Brownian motion model represent? The terrain types that result in anomalous values of fractal dimension tend to exhibit some form of structure (for example, parallel ridges and tiered heights). Thus, fractional Brownian motion appears to return more consistent results within regions of disorder, where no structure is apparent. From the behaviour detailed within [Mandelbrot 82], fractal dimension may be better thought of as representing the correlation of disparate samples within a region. However, the model fails to distinguish consistently between anti-persistence due to highly chaotic behaviour, and anti-persistence due to distinct regions of behaviour being present. Thus, this fractal model will have to be used in conjunction with other measures, in order to identify uniquely the various characteristics of the terrain that are present.

7.4 Applications of the Method

Many of the characteristics attributed to fractional Brownian motion appear absent under close scrutiny, at least as far as general terrain analysis is concerned. In most cases, the fractal dimension does not appear to be related to surface roughness, nor does it exhibit a one-to-one mapping with the surface characteristics of terrain. Such results cast doubt on the current usage of fractional Brownian motion within terrain data compression and interpolation algorithms.

Conversely, at its present stage of development, the technique can be used to highlight areas of poor quality within terrain data. This study has shown that some anomalous effects due to correctly represented terrain are predictable. Very high fractal dimension values are common amongst regions where values are held to poor accuracy, while periodicity can produce both high and low extremes.

Unfortunately, periodic behaviour occurs within natural structures, such as tectonic folds, sand dunes, or due to man-made structures (for example, ploughed fields). Such structures return both high fractal dimensions (the Appalachian mountain range), as well as extremely

low values (the Alpine range). These results need not be contradictory, since the two regions were measured at different scales, and, as has been postulated earlier, differing scale ranges can produce opposite extremes where periodic behaviour is concerned. The *a priori* detection of such behaviour, and the identification of associated structures in the true terrain surface could leave residual behaviour associated solely with poor terrain representations.

Analysis of the behaviour observed within regions of extreme fractal dimension has been detailed above. A crude classification of the manner in which data is poorly represented has been attempted. This classification could be refined in conjunction with alternative terrain measures (for example, variance and scale). As a result, not only could automated quality assessment of terrain be attempted, but the way in which terrain fails such a test could be automatically determined.

Appendix A

Paper presented at IGARRS '90

SURFACE ROUGHNESS ESTIMATION USING FRACTAL VARIOGRAM ANALYSIS

A copy of a paper presented to the Stereo Matching and Digital Terrain Mapping Session, IGARRS '90, Washington DC. United States.

D. Rees and J-P.Muller
Department of Photogrammetry and Surveying,
University College London,
Gower Street,
London WC1E 6BT.
INTERNET: drees@uk.ac.ucl.cs

A.1 Abstract

Fractal dimension is a potentially valuable means of quantifying the roughness of an entire DEM, and/or its sub-regions. However, it has found few applications, due to the fact that it is computationally expensive, and that the intervals of constant fractal dimension had to be determined manually.

A technique has been developed whereby linear sections within a variogram, and hence scale ranges of constant fractal dimension, may be determined automatically.

A study is being conducted to determine the usefulness of fractal dimension as a tool for segmentation of terrain types by their associated quantified roughness. Variograms for grids

of small tiles covering the DEM have been obtained, from which linear intervals, and hence fractal dimension, have been automatically determined.

The method has been applied to a stereo matched DEM, and to the USGS USA 30 second DEM. The DEMs has been segmented, and comparisons of fractal behaviour with the terrain measured are detailed.

Keywords: DEM Segmentation, Fractal Dimension, Variogram.

A.2 INTRODUCTION

Fractal techniques have gradually gained a reputation as a way of rendering visually realistic terrain. In particular, much attention has been focused on fractional Brownian motion ([Fournier 82]; [Mandelbrot 77] ; [Mandelbrot 82]). Despite the fact that this process has not been derived from models of terrain formation, it still persists as a useful application of fractal geometry to terrain datasets. Fractals are basically spatial distributions or patterns which possess self-similarity so that there exists a statistical equivalence between small-scale and large-scale fluctuations in these patterns. Many patterns observed in real world data (point distributions, curves and surfaces) appear to be of self-similar fractal form, such as coastlines which appear to be similar at different scales.

Fractals are characterized by their fractional dimension D, which gives a measure of the change in the properties of a phenomena with scale. In a pure mathematical model, fractal curves maintain these properties over infinite ranges of scale, though in real data, finite limits have been observed. Thus, fractal models of real phenomena include inner and outer cutoff scales, which determine the limits within which similar scaling behaviour is dominant. More than one interval can exist within a wide scale range, so discrete behaviour changes over large scale changes have been recorded.

Early evidence that terrain had scaling properties came from Benoit Mandelbrot [Mandelbrot 67] who interpreted an earlier study by Richardson. Richardson had been investigating the measurement of length of international boundaries and coastlines. From the results, he concluded that boundary length was dependent on the resolution with which it was measured, increasing as a function of the inverse of the measurement interval. Mandelbrot related Richardson's data to the fractal dimension D, showing a scaling of boundary length with measurement

interval could be related by a function with a constant exponent over wide ranges of scale, of up to two orders of magnitude.

Fractional Brownian motion was applied to contour data in [Goodchild 82]. The model was further studied in [Mark 84], which used the variogram of elevation differences between points a known distance apart. Data sampled using the GESTALT photomapper from United States Geological Survey 30 metre Digital Elevation Models (DEMs) was used.

Breaks in the linearity of the graph were detected visually. All but one of the variograms exhibited linear behaviour over limited ranges of scale. Some exhibited distinct linear behaviour of several ranges of scale within the same variogram.

Our aim is to automate this process, such that regional variation of fractal dimension may be detected. This is only feasible if variograms are computed for arbitrarily many portions of the DEM, and from this, fractal dimension is determined in an unsupervised manner.

Applications of these segmented DEMs include (a) data compression of regional and global data-sets [Muller 90a]; (b) simulation of sub-pixel scattering effects [Muller 90c]; (c) estimation of kriging interpolation local functions [Muller 90b].

A.3 COMPUTATION OF FRACTAL DIMENSION

The application of fractional Brownian motion to terrain originated from [Mandelbrot 75]. Fractal behaviour is determined by defining a function applicable to a phenomena, where the function exhibits 'invariant' behaviour over a range of scales. When applied to two dimensional functions of natural stochastic systems, in this case height within a coordinate system, a variety of functions may potentially be used to determine fractal properties.

Our method of fractal dimension computation for DEMs is derived from [Mark 84], which uses a variant on Brownian motion known as fractional Brownian motion. As it is only valid to determine the fractal dimension over a scale range of constant dimension value, a linear segmentation algorithm (but not the fractal measurement algorithm) due to [Yokoya 88] and [Yokoya 89] is used to determine such intervals over log-log plots.

Formally, a random function $Z_H(\boldsymbol{x})$ exhibits pure fractional Brownian motion if for all $\boldsymbol{x}$ and $\Delta\boldsymbol{x}$:

$$Pr[\mid \Delta Z_{\Delta\boldsymbol{x}} \mid / \mid\mid \Delta\boldsymbol{x} \mid\mid^{-H} < y] = erf(y) \tag{A.1}$$

where "Pr" denotes probability, H is the dimension of the Brown zeroset (which is trivially related to the fractal dimension), and $\mid \Delta Z_{\Delta x} \mid$ denotes $\mid Z_H(x + \Delta x) - Z_H(x) \mid$.

This function exhibits behaviour that gives a similar shaped distribution at different scales. For fractional Brownian motion, $\Delta Z_{\Delta x}$ are the increments of a single valued function $Z_H(x)$, the increments having a Gaussian distribution and a variance as A.2.

$$< [\Delta Z_{\Delta x}]^2 > \propto \| \Delta x \|^{2H} \tag{A.2}$$

where <> denote averages over many samples of $Z_H(x)$, H is a scaling parameter with range $0 < H < 1$, and Δx is a vector change in position in Euclidean space ($\Re^E, E > 1$).

The power law relation between $\Delta Z_{\Delta x}$ and Δx should be invariant for all Δx for true fractals (which have constant H at all space scales) implying that the shape of the distributions for fixed values of Δx will remain the same (Gaussian). Thus, computation of height change statistics at various space scales will enable H to be derived. Taking logs of equation A.2 we have:

$$\ln(< [\Delta Z_{\Delta x}]^2 >) \propto \ln(\| \Delta x \|^{2H})$$

and finally, rearranging the above to give:

$$\ln(< [\Delta Z_{\Delta x}]^2 >) \propto 2H \ln(\| \Delta x \|) \tag{A.3}$$

Equation A.3 gives a straight line relationship between $\ln(< [\Delta Z_{\Delta x}]^2 >)$ and $\ln \| \Delta x \|$ with gradient $2H$ for a true fractal. Empirical studies in the past have found that fractal components of natural scenes appear to preserve their fractal dimension (and hence H value) over a variety of ranges in Δx. An algorithm to determine these ranges is given later.

H is determined by computing the gradient of least squares regressions of the graph $\ln(< [\Delta Z_{\Delta x}]^2 >)$ against $\ln(\| \Delta x \|)$ for a linear interval.

Finally, the fractal dimension, D, is then obtained from H using the simple relation

$$D = 3 - H \tag{A.4}$$

A.4 LINEAR SEGMENTATION OF THE VARIOGRAM

A measure of linearity put forward by [Yokoya 88] and [Yokoya 89] was defined as:

$$I = \frac{\sqrt{4\mu_{11}^2 + (\mu_{20} - \mu_{02})^2}}{\mu_{20} + \mu_{02}} \tag{A.5}$$

where $\mu_{ij}(0 \leq i, j \leq 2; i + j = 2)$ represents the central-order second moment of a set of points in a plane.

Let S_n denote a set of n points (where $n \geq 3$) in a plane of fractal plots as

$$S_n = \{p(\| \Delta x \|)\}_{\|\Delta x\| = \|\Delta x\|_{min}}^{\|\Delta x\|_{min}+n-1}$$

and I_n represents a measure of linearity computed for S_n. The largest limit $\| \Delta x \|_{max}$ of scale within which the function is linear is determined by:

$$\| \Delta x \|_{max} = \| \Delta x \|_{min} + n^* - 1$$

where n^* is given by

$$n^* = Min\{n \mid I_{n-1} \leq I_n, I_n > I_{n+1}; n \geq 4\} \tag{A.6}$$

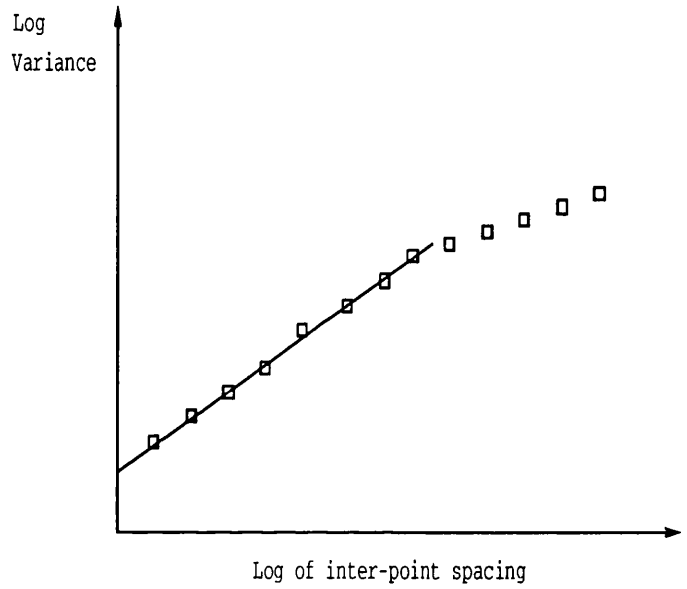
Note: this is not the equation as given in [Yokoya 88] and [Yokoya 89], which contained the erroneous limits:

$$I_{n-1} < I_n \ , \ I_n > I_{n+1}; \tag{A.7}$$

Given a graph with two perfect sections of linearity, I will gain a measure of 1.0 for the first line. However, when the second line is encountered, I_{n+1} will drop in value, but both I_{n-1} and I_n will remain equal. The test will not detect this as a break in the linearity. Beyond the break in linearity, I will steadily decrease, repeatedly failing the test. Thus, the two perfect linear intervals will be treated as one imperfect linear interval by the original criteria.

In figure A.1 a variogram of a DEM shows distinct linear intervals. To determine the extent of the first linear interval, $\| \Delta x \|_{max}$ must be found. The linearity measure for all points up to and including the current point are show in figure A.2, where the y axis shows I calculated using equation A.5. As each new I value is calculated, equation A.6 is used to determine when the maximum I value is reached. After this point, the linear fit to that

Figure A.1: Example variogram with first linear interval shown



section of the variogram decreases, such that the current linear interval has been exceeded. Thus, the first linear interval of the variogram, shown in figure A.1 is now calculated.

Once a linear interval is determined, equation A.3 is used as the basis of a least squares fit algorithm, to determine attributes such as the fractal dimension of the interval.

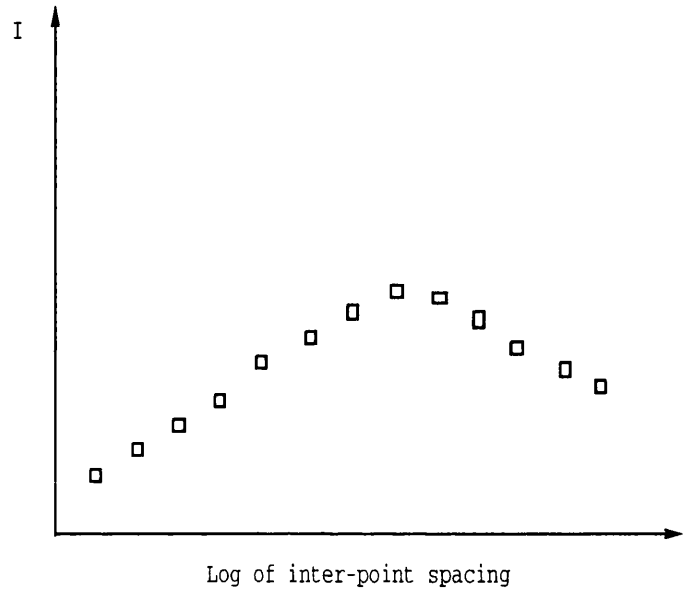
A.5 APPLICATION TO DEMS

In order for segmentation to be applied to DEMs, fractal dimension must be calculated locally over the entire DEM. To achieve this, we divide the DEM into a matrix of equal sized tiles. These tiles are then treated as complete DEMs within their own right, with a variogram being produced for each.

We have used a tile size of at least 32 by 32 DEM grid elements, giving in the order of a thousand comparisons for each inter-point spacing required. To derive the variance data, all points are compared with all others for small tiles. A larger sample justifies use of a row column algorithm (see [Rees 90a]), which speeds up computation.

Two DEMs were considered, to give an idea of how the technique performed at different scales. The larger of the two covered a area of 9 363 132 square kilometres, resulting in the

Figure A.2: Linear fit values for the example variogram



use of spherical coordinates to analyse the data. In order to gain an equivalent comparison over the entire DEM, variances were binned at kilometre intervals. In addition, a maximum point-pair spacing was guaranteed for the entire DEM, such that all linear intervals were limited in the range of scales used.

Producing graphs and finding linear intervals by hand is an impossible task for this volume of data. Hence, the method of automatic linear segmentation outlined above was used. From theory, we expect fractal dimension to remain constant over a significant range of scales. When the inter-point spacing is plotted against expected absolute difference on a log-log graph, equation A.3 implies that a linear interval with gradient H will result.

However, when using the linear segmentation method, we may encounter two or three distinct linear intervals for the tile size given above. This leads to the problem of which interval should be chosen to represent the fractal dimension for that particular area. In other studies, where the method has been applied to more local terrain datasets, the first linear interval has been found to give the most consistent results. It avoids the complexity of comparing intervals which do not represent exactly the same scale range. Moreover, the method allows for potential interpolation of the data, as output of the first linear interval can

be fed directly into a fractal interpolation program, as applied in [Allison 89] and [Rees 90b].

A.6 RESULTS

To observe any intuitive relationships between terrain, and its local fractal dimension over a broad area, the USA 30 second DEM was segmented into 40 by 40 tiles, for each of which a variogram was calculated. Tiles were not overlapped, as the resulting fractal dimension was of sufficient detail such that trends could be observed. From these variograms, the first linear interval was detected, and used as the basis for calculating the fractal dimension of the local region. The DEM consists of 8400 by 3360 height samples, giving a resultant tiled image with a resolution of 210 by 84. However, not all points represent an area within the land mass of the United States, so such tiles are given a null fractal dimension value. Figure A.3 shows the resulting image.

In order to examine more local effects, the method was applied to a stereo matched DEM of Montaigne Sainte Victoire, near Aix-en-Provence. This consisted of 415 by 208 height samples, equi-spaced at 30 metre intervals. This was segmented into 32 by 32 tiles. Figure A.4 shows the results. In this figure, overlapping tiles with a grid pitch of 8 samples are used to give a better visual impression of the changing fractal dimension. However, for all analysis, non-overlapping tiles were used, such that redundant information was not processed, and time constaints were served.

As can be seen, there is some limited correlation between the features in the DEM and the resulting fractal values. To test this further, we investigated the relationships that exist between fractal dimension and rudimentary terrain characteristics. To this end, values resulting from tiled images (40 by 40 sized tiles, non-overlapping) of fractal dimension of the dataset were compared with images, degraded to the same resolution, of height, slope and aspect. The correlation coefficients of fractal dimension with each of the above images were calculated.

For Montaigne Saint Victoire, correlation of dimension against height resulted in a value of 0.44. For dimension against slope, a much higher value of 0.62 was noted. Both the above are extremely significant, with $P < 0.01$, whereas for the correlation of dimension against aspect, a value of 0.01, was observed, giving effectively no correlation at all.

Figure A.3: USGS USA 30 Second DEM images: Top - Height intensity, Top mid - Lambertian shaded, Lower mid - First interval fractal dimension (range -24.97 to 4.39, values in range 2.0 to 3.0 displayed)

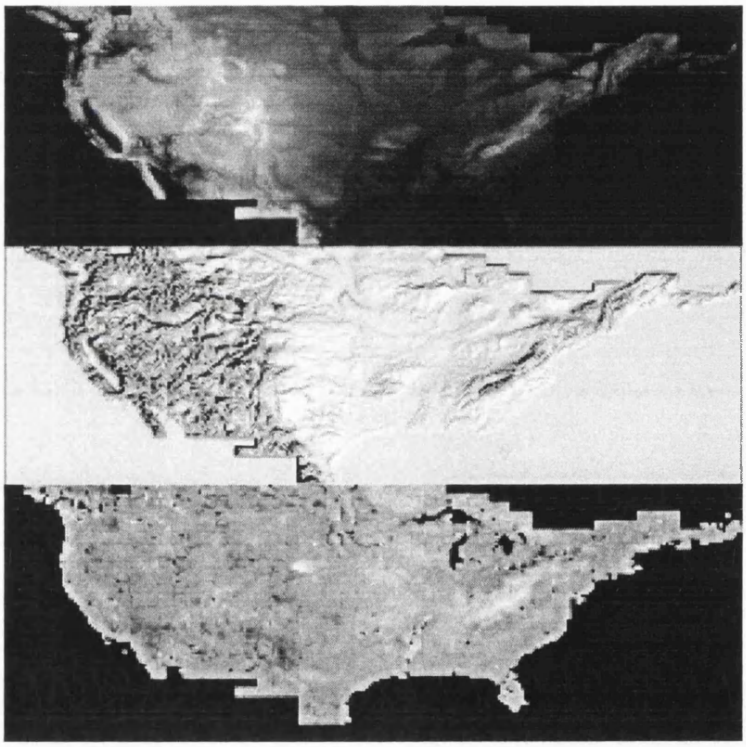
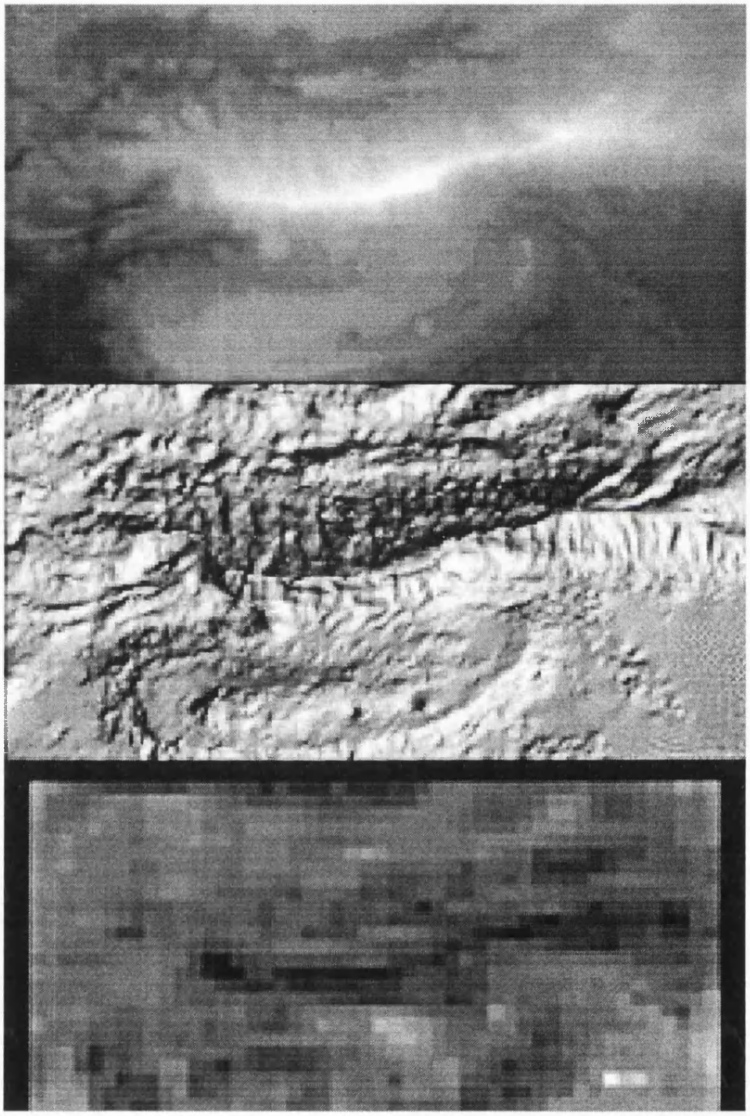


Figure A.4: Montaigne Sainte Victoire stereo-matched DEM images: Top - Height intensity, Middle - Lambertian shaded, Bottom - First interval fractal dimension (range 2.05 to 2.87).



For the USGS USA 30 second DEM, different behaviour was recorded. The correlation of dimension against height gave the best result, a value of 0.45, and a significance level $P < 0.01$. For the other two comparisons, negligable values resulted, with significance level $P > 0.1$. For dimension against aspect, a correlation coeficient of 0.03 was recorded, and for dimension against slope, an even lower value of 0.01 resulted.

A most surprising result was that of the range of fractal dimension values encountered. For Montaigne Sainte Victoire, a range of 2.05 to 2.87 was recorded. However, for the USA DEM, the minimum value measured was -24.97, whereas the maximum value measured was 4.39. Measurement of the areas producing these values using 60 by 60 sized tiles yielded similar results. However, the vast majority of values did lie within the range $2 < D < 3$, as can be appreciated by comparing the normal and histogram equalised results in figure A.3.

Using our row/column algorithm, a time complexity of $O(N^{3/2})$ results, an improvement on the $O(N^2)$ behaviour of the standard method. On a Sun 4/60, calculation of the variance values for the USA dataset occupied 180 CPU minutes. In addition, calculation of the linear segments, and the resulting fractal dimensions took a further 2 CPU minutes.

A.7 CONCLUSIONS

Given the size of the dataset under study, the time of three hours taken to aquire the segmentation information seems acceptable.

Correlation between fractal dimension and aspect gave negligible results for both samples. However, correlation with slope gave a marked result for the Sainte Victoire DEM, and negligable correlation for the USA. This is probably due to the tile size being larger than most features exhibited by the USA DEM, averaging over any slopes. For Sainte Victoire, a tile may occupy a small proportion of a slope on the central ridge, resulting meaningful average slope for the whole tile.

Correlation between fractal dimension and height gave an intuitively correct result. It is to be expected that 'rougher' surfaces would occur more commonly in mountainous regions. However, the correlation is not absolute, suggesting that fractal dimension has a role in segmentation that cannot be supplanted by height alone.

The most surprising result of the study is that of fractal dimension exceeding the expected

range of $2 < D < 3$. Such values have now been obtained using different algorithms, and different datasets. We are at present determining conditions under which these values can occur.

Appendix B

Program Listings

B.1 VarianceAll

```
/*********************************************
 VarianceAll

 Program to read in a HIPs
 uniformly spaced grid DEM and output
 variance values in a HIPs file format.
 Outputs variance, variance variance
 and number of point pairs used.
 This is repeated for each Dx distance.
 This version calculates the above values
 for all combinations in the specified
 inter-point range.

 David Rees 16/8/1989 */

#include        <stdio.h>
#include        <math.h>
#include        "Hipl_format.h"

float   *dem, *ExpectationVals;                 /*Storage for dem in, and expectations out*/
struct  header h;                               /*File in and out header*/
int     radx[512], rady[512], RadMax;           /*Storage for circle coordinates, a distance radius about a point,*/
                                                /*and the amount of values stored in the arrays*/
int     Radius;                                 /*The radius of the circle about a point to be examined*/
int     XNumbSquares, YNumbSquares;             /*The size of the side of area to be examined*/
int     XSpaceSquares, YSpaceSquares;           /*The interval between the center of each square*/
int     InitialDx, LastDx, PlusDx;              /*First and last radius size and increment*/
long    FrameSize;                              /*The size of each frame of expectations to be output*/
int     Frames, NewRows, NewCols;               /*Number of frames, with row and column dimensions*/
int     StartSquareRow, StartSquareCol; /*Precalculated start positions*/
float   *left_edge_ptr[10000];                  /*Precalculate left edge of current rectangle*/

/*Statistical variables*/
        float   Threshold;                      /*The threshold height below which no points will be considered*/

/*****************************************************/
main(argc, argv)
int argc;
char *argv[];

{
        Read_in_DEM();
        Input_Arguements(argc,argv);
        Get_Frame_Memory();
        Get_All_Expectations();
        Write_Expectations(argc,argv);
}/*End of main*/


/******SetUp Routines**********/

Input_Arguements(argc, argv)
/*Input the commmand line arguements*/
int argc;
char *argv[];

{
int argn, find_string();

        if ((argn=find_string("-x",argc,argv)) != -1)
        {
                sscanf(argv[argn+1],"%d",&XNumbSquares);
                if (XNumbSquares > h.cols) {fprintf(stderr,"X side size is larger than image size.\n");exit(1);}
                if (XNumbSquares <= 0   ) {fprintf(stderr,"X side size is zero or less.\n");exit(1);}
        }
        else
                XNumbSquares = h.cols;

        if ((argn=find_string("-y",argc,argv)) != -1)
        {
                sscanf(argv[argn+1],"%d",&YNumbSquares);
                if (YNumbSquares > h.rows) {fprintf(stderr,"Y side size is larger than image size.\n");exit(1);}
                if (YNumbSquares <= 0   ) {fprintf(stderr,"Y side size is zero or less.\n");exit(1);}
        }
        else
                YNumbSquares = h.rows;

        if ((argn=find_string("-r",argc,argv)) != -1)
        {
                sscanf(argv[argn+1],"%d",&XSpaceSquares);
                if (XSpaceSquares > h.cols) {fprintf(stderr,"X interval size is larger than image size.\n");exit(1);}
                if (XSpaceSquares <= 0   ) {fprintf(stderr,"X interval size is zero or less.\n");exit(1);}
        }
        else
                XSpaceSquares = XNumbSquares;

        if ((argn=find_string("-c",argc,argv)) != -1)
        {
                sscanf(argv[argn+1],"%d",&YSpaceSquares);
                if (YSpaceSquares > h.rows) {fprintf(stderr,"Y interval size is larger than image size.\n");exit(1);}
```

```
                if (YSpaceSquares <= 0   ) {fprintf(stderr,"Y interval size is zero or less.\n");exit(1);}
        }
        else
                YSpaceSquares = YNumbSquares;

        if ((argn=find_string("-i",argc,argv)) != -1)
        {
                sscanf(argv[argn+1],"%d",&InitialDx);
                if (InitialDx > h.rows && InitialDx > h.cols) {fprintf(stderr,"Minimum pair spacing is larger than the DEM.\n");exit(1);}
                if (InitialDx <= 0   ) {fprintf(stderr,"Minimum pair spacing is zero or less.\n");exit(1);}
        }
        else
                InitialDx = 1;

        if ((argn=find_string("-l",argc,argv)) != -1)
        {
                sscanf(argv[argn+1],"%d",&LastDx);
                if (LastDx > h.rows && LastDx > h.cols) {fprintf(stderr,"Maximum pair spacing is larger than the DEM.\n");exit(1);}
                if (LastDx <= 0   ) {fprintf(stderr,"Maximum pair spacing is zero or less.\n");exit(1);}
                if (LastDx < InitialDx) {fprintf(stderr,"Maximum pair spacing is less than minimum pair spacing.\n");exit(1);}
        }
        else
                LastDx = InitialDx;

        if ((argn=find_string("-p",argc,argv)) != -1)
        {
                sscanf(argv[argn+1],"%d",&PlusDx);
                if (PlusDx > h.rows && PlusDx > h.cols) {fprintf(stderr,"Increment of pair spacing is larger than the DEM.\n");exit(1);}
                if (PlusDx <= 0   ) {fprintf(stderr,"Increment of  pair spacing is zero or less.\n");exit(1);}
        }
        else
                PlusDx = 1;

        if ((argn=find_string("-t",argc,argv)) != -1)
                sscanf(argv[argn+1],"%f",&Threshold);
        else
                Threshold = 0.0;
}

/*Get the frames to store the expectation values in*/
Get_Frame_Memory()
{
    /*Three frames per DX value.
        One for the variance, another for variance squared,
        and finally one for number of points analysed*/
        Frames = ((LastDx - InitialDx)/PlusDx + 1)*3;
        NewRows = ((h.rows - YNumbSquares) / YSpaceSquares) + 1;
        NewCols = ((h.cols - XNumbSquares) / XSpaceSquares) + 1;
        FrameSize = NewRows*NewCols;
        if ( (ExpectationVals=(float*)calloc(Frames*FrameSize, sizeof(float))) == NULL)
                {fprintf(stderr,"Can't get  new frames storage.\n");exit(1);}
}

/*.....*/
/*****routines to find DEM expected values*****/
/*.....*/

/*Work out all Expectation values, In order of Dx size, then square by square*/
Get_All_Expectations()
{
int     ExpRows, ExpCols;
float   Variance,VarianceVariance,NumberPointPairs;
int     FrameCount;

        FrameCount = 0;
        for (Radius = InitialDx; Radius <= LastDx; Radius=Radius + PlusDx)
        {
/*For all the chosen radii...*/
/*set up the circle size for pixel pair differences*/
                DifferenceArray();
                for (ExpRows = 0; ExpRows < NewRows; ExpRows++)
                {
                        StartSquareRow = ExpRows * YSpaceSquares;
                        for (ExpCols = 0; ExpCols < NewCols; ExpCols++)
                        {
                                StartSquareCol = ExpCols * XSpaceSquares;
                                GetDiffs(&Variance, &VarianceVariance, &NumberPointPairs);
                                SetExpValue(ExpCols,ExpRows,FrameCount*3,Variance);
                                SetExpValue(ExpCols,ExpRows,FrameCount*3+1,VarianceVariance);
                                SetExpValue(ExpCols,ExpRows,FrameCount*3+2,NumberPointPairs);
                        }
                }
                FrameCount++;
        }
}


/*Get the expected values for a particular area of the DEM*/
GetDiffs(Variance,VarianceVariance,NumberPointPairs)
float   *Variance, *VarianceVariance, *NumberPointPairs;
```

```
{
int     ximage, yimage;
float   Variancedifference, Variancevariancedifference, numbpoints;
float   variance, variancevariance;
int     numberofpoints;

        variance = 0.0;
        variancevariance = 0.0;
        numbpoints = 0.0;

        for (yimage = 0; yimage < YNumbSquares; yimage++)
        {
                left_edge_ptr[yimage] = dem + (((long)StartSquareRow + (long)yimage) * (long)h.cols + (long)StartSquareCol);
        }

        for (yimage = 0; yimage < YNumbSquares; yimage++)
        {
                for (ximage = 0; ximage < XNumbSquares; ximage++)
                {
                        CalcEforPoint(ximage, yimage, &Variancedifference, &Variancevariancedifference, &numberofpoints);
                        if (numberofpoints != 0)
                        {
                                variance += Variancedifference;
                                variancevariance += Variancevariancedifference;
                                numbpoints += numberofpoints;
                        }
                }
        }
        *Variance = variance / numbpoints;
        *VarianceVariance = variancevariance / numbpoints;
        *NumberPointPairs = numbpoints;
}

/*Calculate the X,Y coordinates that lie on a circle of Radius*/
DifferenceArray()
{
int x,y;
double sqrt(), length, square, radlow, radhih;
        radlow = Radius - 0.5;
        radhih = Radius + 0.5;
        RadMax = 0;
        for (x = 0; x <= Radius; x++)
        {
                for (y = 0; y <= Radius; y++)
                {
                if ((y != 0 && x == Radius) || (x != 0 && y == Radius) || ((x < Radius) && ( y < Radius)))

                {
/*Compare the vector length of the coordinate with the difference Radius*/
                        square = x*x + y*y;
                        length = sqrt(square);
                        if (length >= radlow && length < radhih)
                        {
                                RadMax++;
                                radx[RadMax] = x;
                                rady[RadMax] = y;
                        }
                }
                }
        }
}

/*Calculate the total absolute difference from points in a circle around (x,y)*/
CalcEforPoint(x , y, Variance, VarianceVariance, numberp)
int x, y, *numberp;
float *Variance, *VarianceVariance;
{
        register float variance = 0.0, variancevariance = 0.0;
        register int number = 0;
        register int            radloop, xadd, yadd, xsub, ysub, CurrentRadius;
        float       vardiff, varvardiff, centerpix, DEMValue();
        int thisradx, thisrady;
        int OnePixelDiff();

        CurrentRadius = Radius;
        centerpix = DEMValue(x,y);
        if (centerpix > Threshold)
        {
                for (radloop = 1; radloop <= RadMax; radloop++)
                {
                        thisradx = radx[radloop];
                        thisrady = rady[radloop];
                        xadd = x + thisradx;
                        yadd = y + thisrady;
                        xsub = x - thisradx;
                        ysub = y - thisrady;
                        if ( OnePixelDiff(xadd, yadd, centerpix, &vardiff, &varvardiff) )
                        {
                                variance += vardiff;
                                variancevariance += varvardiff;
```

```
                                        number++;
                                }
/*Now for the other three quadrants*/
                                if ( OnePixelDiff(xsub, yadd, centerpix, &vardiff, &varvardiff) )
                                {
                                        variance += vardiff;
                                        variancevariance += varvardiff;
                                        number++;
                                }
                        }
/*Now for the exceptions - (N,0),(0,N),(-N,0),(0,-N)*/
                        xadd = x + CurrentRadius;
                        if ( OnePixelDiff(xadd, y, centerpix, &vardiff, &varvardiff) )
                        {
                                variance += vardiff;
                                variancevariance += varvardiff;
                                number++;
                        }
                        yadd = y + CurrentRadius;
                        if ( OnePixelDiff(x, yadd, centerpix, &vardiff, &varvardiff) )
                        {
                                variance += vardiff;
                                variancevariance += varvardiff;
                                number++;
                        }
                }
                *numberp = number;
                *Variance = variance;
                *VarianceVariance = variancevariance;
}


int
OnePixelDiff(xa ,ya ,centerpixvalue, vardiff, varvardiff)
int     xa, ya;
float   centerpixvalue, *vardiff, *varvardiff;
{
register        float   val, valsquared, *ValuePtr;
/*float   DEMValue();*/

        if ((xa > -1) && (xa < XNumbSquares) && (ya > -1) && (ya < YNumbSquares))
        {
/*              val=DEMValue(xa,ya);*/
                ValuePtr = left_edge_ptr[ya] + (long)xa;
                val = *ValuePtr;
                if (val > Threshold)
                {
                        valsquared = (centerpixvalue - val);
                        valsquared *= valsquared;
                        *vardiff = valsquared;
                        valsquared *= valsquared;
                        *varvardiff = valsquared;
                        return(1);                      /* success */
                }
        }
        return(0);              /* failure */
}

float DEMValue(x,y)
int     x,y;
{
float   *ValuePtr;

        ValuePtr = left_edge_ptr[y] + (long)x;
        return(*ValuePtr);
}

/*.....*/
/**********Subroutines for manipulating HIPs files**********/
/**********/

Read_in_DEM()
/*Read in a DEM, placing the header in a global variable h,
  and the data in a location returned in pointer dem.*/
{
long    InFrameSize;

        fread_header(0,&h);
        if (h.num_frame != 1)
                {fprintf(stderr,"There must only be one frame.\n");exit(1);}
        if (h.pixel_format != PFFLOAT)
                {fprintf(stderr,"Must be float format\n");exit(1);}

        InFrameSize = h.rows*h.cols;

        if ( (dem=(float*)calloc(InFrameSize,sizeof(float))) == NULL)
                {fprintf(stderr,"Can't get store\n");exit(1);}

        pread(0,dem,InFrameSize*sizeof(float));
}
```

```
Write_Expectations(argc,argv)
/*Input the commmand line arguements*/
int argc;
char *argv[];

/*Write a HIPs file whose header is in variable h, and whose data
  is pointed to by pointer ExpectationVals*/
{
        h.num_frame=Frames;
        h.rows = NewRows;
        h.cols = NewCols;

        update_header(&h,argc,argv);

        update_desc(&h," Expectation, Variance, Variance variance, and point pair count values for areas of the DEM.");

        write_header(&h);
        write(1,ExpectationVals,Frames*FrameSize*sizeof(float));
}


SetExpValue(x,y,frame,value)
int     x,y,frame;
float   value;
{
float   *ValuePtr;

        ValuePtr = (float*)((long)ExpectationVals+((long)frame*(long)FrameSize+(long)y*(long)NewCols+(long)x)*sizeof(float));
        *ValuePtr = value;
}

/*
find_switch (c,argc,argv)
char c;
int argc;
char *argv[];
{   int i;
    for (i=0;i<argc;i++) if (*argv[i] == '-' && *(argv[i]+1) == c && *(argv[i]+2) == '\0') return(TRUE);
    return (FALSE);
}
*/
int find_string (s,argc,argv)
char *s;
int argc;
char *argv[];
{       int i;
        for (i=0;i<argc;i++) if (strcmp(s,argv[i]) == 0) return(i);
        return(-1);
}
```

B.2 VarianceRowCol

```
/***********************************************
 VarianceRowCol

 Program to read in a HIPs
 regular spaced grid DEM and output
 variance values in a HIPs file format.
 Output consists of frames of X variance, X variance
 variance (for standard deviation of data),
 Y variance, Y variance variance, CoVariance,
 CoVariance variance, and frequency counts
 for each of X,Y and Covariance number of point
 pairs used.
 This is repeated for each Dx distance.
 This version uses a faster means of
 calculating values, by means of only
 rows and columns.

  David Rees 16/5/1989 */

#include        <stdio.h>
#include        <math.h>
#include        "Hipl_format.h"

float   *dem, *ExpectationVals;                 /*Storage for dem in, and expectations out*/
struct  header h;                               /*File in and out header*/
int     radx[512], rady[512], RadMax;           /*Storage for circle coordinates, a distance radius about a point,*/
                                                /*and the amount of values stored in the arrays*/
int     Radius;                                 /*The radius of the circle about a point to be examined*/
int     XNumbSquares, YNumbSquares;             /*The size of the side of area to be examined*/
int     XSpaceSquares, YSpaceSquares;           /*The interval between the center of each square*/
int     InitialDx, LastDx, PlusDx;              /*First and last radius size and increment*/
long    FrameSize;                              /*The size of each frame of expectations to be output*/
int     Frames, NewRows, NewCols;               /*Number of frames, with row and column dimensions*/
int     StartSquareRow, StartSquareCol; /*Precalculated start positions*/
float   *left_edge_ptr[10000];                  /*Precalculate left edge of current rectangle*/

/*Statistical variables*/
        float   Threshold;                      /*The threshold height below which no points will be considered*/

/*****************************************************/
main(argc, argv)
int argc;
char *argv[];

{
        Read_in_DEM();
        Input_Arguements(argc,argv);
        Get_Frame_Memory();
        Get_All_Expectations();
        Write_Expectations(argc,argv);
}/*End of main*/


/******SetUp Routines**********/

Input_Arguements(argc, argv)
/*Input the commmand line arguements*/
int argc;
char *argv[];

{
int argn, find_string();

        if ((argn=find_string("-x",argc,argv)) != -1)
        {
                sscanf(argv[argn+1],"%d",&XNumbSquares);
                if (XNumbSquares > h.cols) {fprintf(stderr,"X side size is larger than image size.\n");exit(1);}
                if (XNumbSquares <= 0   ) {fprintf(stderr,"X side size is zero or less.\n");exit(1);}
        }
        else
                XNumbSquares = h.cols;

        if ((argn=find_string("-y",argc,argv)) != -1)
        {
                sscanf(argv[argn+1],"%d",&YNumbSquares);
                if (YNumbSquares > h.rows) {fprintf(stderr,"Y side size is larger than image size.\n");exit(1);}
                if (YNumbSquares <= 0   ) {fprintf(stderr,"Y side size is zero or less.\n");exit(1);}
        }
        else
                YNumbSquares = h.rows;

        if ((argn=find_string("-r",argc,argv)) != -1)
        {
                sscanf(argv[argn+1],"%d",&XSpaceSquares);
                if (XSpaceSquares > h.cols) {fprintf(stderr,"X interval size is larger than image size.\n");exit(1);}
                if (XSpaceSquares <= 0   ) {fprintf(stderr,"X interval size is zero or less.\n");exit(1);}
        }
        else
                XSpaceSquares = XNumbSquares;
```

```
        if ((argn=find_string("-c",argc,argv)) != -1)
        {
                sscanf(argv[argn+1],"%d",&YSpaceSquares);
                if (YSpaceSquares > h.rows) {fprintf(stderr,"Y interval size is larger than image size.\n");exit(1);}
                if (YSpaceSquares <= 0    ) {fprintf(stderr,"Y interval size is zero or less.\n");exit(1);}
        }
        else
                YSpaceSquares = YNumbSquares;

        if ((argn=find_string("-i",argc,argv)) != -1)
        {
                sscanf(argv[argn+1],"%d",&InitialDx);
                if (InitialDx > h.rows && InitialDx > h.cols) {fprintf(stderr,"Minimum pair spacing is larger than the DEM.\n");exit(1);}
                if (InitialDx <= 0    ) {fprintf(stderr,"Minimum pair spacing is zero or less.\n");exit(1);}
        }
        else
                InitialDx = 1;

        if ((argn=find_string("-l",argc,argv)) != -1)
        {
                sscanf(argv[argn+1],"%d",&LastDx);
                if (LastDx > h.rows && LastDx > h.cols) {fprintf(stderr,"Maximum pair spacing is larger than the DEM.\n");exit(1);}
                if (LastDx <= 0    ) {fprintf(stderr,"Maximum pair spacing is zero or less.\n");exit(1);}
                if (LastDx < InitialDx) {fprintf(stderr,"Maximum pair spacing is less than minimum pair spacing.\n");exit(1);}
        }
        else
                LastDx = InitialDx;

        if ((argn=find_string("-p",argc,argv)) != -1)
        {
                sscanf(argv[argn+1],"%d",&PlusDx);
                if (PlusDx > h.rows && PlusDx > h.cols) {fprintf(stderr,"Increment of pair spacing is larger than the DEM.\n");exit(1);}
                if (PlusDx <= 0    ) {fprintf(stderr,"Increment of  pair spacing is zero or less.\n");exit(1);}
        }
        else
                PlusDx = 1;

        if ((argn=find_string("-t",argc,argv)) != -1)
                sscanf(argv[argn+1],"%f",&Threshold);
        else
                Threshold = 0.0;
}

/*Get the frames to store the expectation values in*/
Get_Frame_Memory()
{
    /*Four frames per DX value. One for  difference,
        one for the variance, another for variance squared,
        and finally one for number of points analysed*/
       /*This allows the fractal dimension according to
       Mark&Aronson and/or Pentland to be calculated*/
       Frames = (LastDx - InitialDx + 1)*9;
       NewRows = ((h.rows - YNumbSquares) / YSpaceSquares) + 1;
       NewCols = ((h.cols - XNumbSquares) / XSpaceSquares) + 1;
       FrameSize = NewRows*NewCols;
       if ( (ExpectationVals=(float*)calloc(Frames*FrameSize, sizeof(float))) == NULL)
                {fprintf(stderr,"Can't get  new frames storage.\n");exit(1);}
}

/*Work out all Expectation values, in order of Dx size, then square by square*/
Get_Ali_Expectations()
{
int     ExpRows, ExpCols;
float   XExpectedVariance, XExpectedVarianceVariance, ExpectedDifferenceX;
float   YExpectedVariance, YExpectedVarianceVariance, ExpectedDifferenceY;
float   ExpectedCoVariance, ExpectedCoVarianceVariance, XNumberPointPairs;
float   YNumberPointPairs, CovNumberPointPairs;
int     FrameCount;

        FrameCount = 0;
        for (Radius = InitialDx; Radius <= LastDx; Radius=Radius + PlusDx)
        {
                for (ExpRows = 0; ExpRows < NewRows; ExpRows++)
                {
                        StartSquareRow = ExpRows * YSpaceSquares;
                        for (ExpCols = 0; ExpCols < NewCols; ExpCols++)
                        {
                                StartSquareCol = ExpCols * XSpaceSquares;
                                GetDiffs(&XExpectedVariance, &XExpectedVarianceVariance,
                                        &YExpectedVariance, &YExpectedVarianceVariance,
                                        &ExpectedCoVariance, &ExpectedCoVarianceVariance,
                                        &XNumberPointPairs, &YNumberPointPairs,
                                        &CovNumberPointPairs);
                                SetExpValue(ExpCols,ExpRows,FrameCount*9,XExpectedVariance);
                                SetExpValue(ExpCols,ExpRows,FrameCount*9+1,XExpectedVarianceVariance);
                                SetExpValue(ExpCols,ExpRows,FrameCount*9+2,YExpectedVariance);
                                SetExpValue(ExpCols,ExpRows,FrameCount*9+3,YExpectedVarianceVariance);
                                SetExpValue(ExpCols,ExpRows,FrameCount*9+4,ExpectedCoVariance);
                                SetExpValue(ExpCols,ExpRows,FrameCount*9+5,ExpectedCoVarianceVariance);
                                SetExpValue(ExpCols,ExpRows,FrameCount*9+6,XNumberPointPairs);
```

```
                                        SetExpValue(ExpCols,ExpRows,FrameCount*9+7,YNumberPointPairs);
                                        SetExpValue(ExpCols,ExpRows,FrameCount*9+8,CovNumberPointPairs);
                                }
                        }
                        FrameCount++;
                }
}

/*Get the expected values for a particular area of the DEM*/
/*The difference, the variance, and the variance of the variance,
  for confidence limits*/
GetDiffs(XExpectedVariance, XExpectedVarianceVariance,
                YExpectedVariance, YExpectedVarianceVariance,
                ExpectedCoVariance, ExpectedCoVarianceVariance,
                XNumberPointPairs, YNumberPointPairs, CovNumberPointPairs)
float   *XExpectedVariance, *XExpectedVarianceVariance;
float   *YExpectedVariance, *YExpectedVarianceVariance;
float   *ExpectedCoVariance, *ExpectedCoVarianceVariance;
float   *XNumberPointPairs, *YNumberPointPairs, *CovNumberPointPairs;
{
register        Int     ximage, yimage, CurrentRadius;
register        float   numbpoints, centerpix, variance;
float   Xtotalvariance, Xtotalvariancevariance, Ytotalvariance, Ytotalvariancevariance;
float   differencex, differencey, totalcovariance, totalcovariancevariance;
int     xnumbpoints, ynumbpoints, covnumbpoints, xpointon, ypointon;
int     OnePixelDiff();
float   DEMValue(), difference;

        CurrentRadius = Radius;
        Xtotalvariance = 0.0;
        Xtotalvariancevariance = 0.0;
        Ytotalvariance = 0.0;
        Ytotalvariancevariance = 0.0;
        totalcovariance = 0.0;
        totalcovariancevariance = 0.0;
        xnumbpoints = 0.0;
        ynumbpoints = 0.0;
        covnumbpoints = 0.0;


        for (yimage = 0; yimage < YNumbSquares; yimage++)
        {
                left_edge_ptr[yimage] = dem + (((long)StartSquareRow + (long)yimage) * (long)h.cols + (long)StartSquareCol);
        }

        for (yimage = 0; yimage < YNumbSquares - CurrentRadius; yimage++)
        {
                for (ximage = 0; ximage < XNumbSquares - CurrentRadius; ximage++)
                {
                        centerpix = DEMValue(ximage,yimage);
                        if (centerpix > Threshold)
                        {
                                xpointon = 0; ypointon = 0;
                                if ( OnePixelDiff(ximage + CurrentRadius, yimage, centerpix, &difference) )
                                {
                                        xpointon = 1;
                                        differencex = difference;
                                        variance = differencex * differencex;
                                        Xtotalvariance = Xtotalvariance + variance;
                                        Xtotalvariancevariance = Xtotalvariancevariance + variance*variance;
                                        xnumbpoints++;
                                }
                                if ( OnePixelDiff(ximage, yimage + CurrentRadius, centerpix, &difference) )
                                {
                                        ypointon = 1;
                                        differencey = difference;
                                        variance = differencey * differencey;
                                        Ytotalvariance = Ytotalvariance + variance;
                                        Ytotalvariancevariance = Ytotalvariancevariance + variance*variance;
                                        ynumbpoints++;
                                }
                                if (xpointon == 1 && ypointon == 1)
                                {
                                        variance = differencex * differencey;
                                        totalcovariance = totalcovariance + variance;
                                        totalcovariancevariance = totalcovariancevariance + variance*variance;
                                        covnumbpoints++;
                                }
                        }
                }
        }

        *XNumberPointPairs = xnumbpoints;
        if (xnumbpoints != 0)
        {
                *XExpectedVariance = Xtotalvariance / xnumbpoints;
                *XExpectedVarianceVariance = Xtotalvariancevariance / xnumbpoints;
        }
        *YNumberPointPairs = ynumbpoints;
        if (ynumbpoints != 0)
```

```
        {
                *YExpectedVariance = Ytotalvariance / ynumbpoints;
                *YExpectedVarianceVariance = Ytotalvariancevariance / ynumbpoints;
        }
        *CovNumberPointPairs = covnumbpoints;
        if (covnumbpoints != 0)
        {
                *ExpectedCoVariance = totalcovariance / covnumbpoints;
                *ExpectedCoVarianceVariance = totalcovariancevariance / covnumbpoints;
        }
}

/*.....*/
/**********Subroutines for manipulating HIPs files**********/
/*.....*/

Read_in_DEM()
/*Read in a DEM, placing the header in a global variable h,
  and the data in a location returned in pointer dem.*/
{
long    InFrameSize;

        fread_header(0,&h);
        if (h.num_frame != 1)
                {fprintf(stderr,"There must only be one frame.\n");exit(1);}
        if (h.pixel_format != PFFLOAT)
                {fprintf(stderr,"Must be float format\n");exit(1);}

        InFrameSize = h.rows*h.cols;

        if ( (dem=(float*)calloc(InFrameSize,sizeof(float))) == NULL)
                {fprintf(stderr,"Can't get store\n");exit(1);}

        pread(0,dem,InFrameSize*sizeof(float));
}

Write_Expectations(argc,argv)
/*Input the commmand line arguements*/
int argc;
char *argv[];

/*Write a HIPs file whose header is in variable h, and whose data
  is pointed to by pointer ExpectationVals*/
{
        h.num_frame=Frames;
        h.rows = NewRows;
        h.cols = NewCols;

        update_header(&h,argc,argv);

        update_desc(&h," Expectation, Variance, Variance variance, and point pair count values for areas of the DEM.");
        update_desc(&h," Horizontal and vertical grid comparisons only.");

        write_header(&h);
        write(1,ExpectationVals,Frames*FrameSize*sizeof(float));
}


/*.....*/
/**********routines to find DEM expected values**********/
/*.....*/


int OnePixelDiff(xa ,ya ,centerpixvalue, diff)
int     xa, ya;
float   centerpixvalue;
register float *diff;
{
float   val;
float   DEMValue();

        if ((xa > -1) && (xa < XNumbSquares) && (ya > -1) && (ya < YNumbSquares))
        {
                val=DEMValue(xa,ya);
                if (val > Threshold)
                {
                        *diff = (centerpixvalue - val);
                        return(1);                      /* success */
                }
        }
        return(0);              /* failure */
}

float DEMValue(x,y)
int     x,y;
{
float   *ValuePtr;

        ValuePtr = left_edge_ptr[y] + (long)x;
        return(*ValuePtr);
```

```
}

SetExpValue(x,y,frame,value)
int         x,y,frame;
float       value;
{
float   *ValuePtr;

            ValuePtr = (float*)((long)ExpectationVals+((long)frame*(long)FrameSize+(long)y*(long)NewCols+(long)x)*sizeof(float));
            *ValuePtr = value;
}

/*
find_switch (c,argc,argv)
char c;
int argc;
char *argv[];
{       int i;
        for (i=0;i<argc;i++) if (*argv[i] == '-' && *(argv[i]+1) == c && *(argv[i]+2) == '\0') return(TRUE);
        return (FALSE);
}
*/
int find_string (s,argc,argv)
char *s;
int argc;
char *argv[];
{           int i;
            for (i=0;i<argc;i++) if (strcmp(s,argv[i]) == 0) return(i);
            return(-1);
}
```

B.3 VarianceSphereRowCol

```
/***********************************************
 VarianceSphereRowCol

 Program to read in a Geographically
 projected DEM and output
 variance values in a HIPs file format.
 Output consists of frames of X variance, X variance
 variance (for standard deviation of data),
 Y variance, Y variance variance, CoVariance,
 CoVariance variance, and frequency counts
 for each of X,Y and Covariance number of point
 pairs used.
 This is repeated for each Dx distance.
 This version uses a faster means of
 calculating values, by means of only
 rows and columns.

  David Rees 16/9/1989 */

#include        <stdio.h>
#include        <math.h>
#include        <pixrect/pixrect_hs.h>
#include        "Hipl_format.h"

#define         MaxArray        10000
#define         TEST_HEIGHT     MaxArray
#define         TEST_WIDTH      MaxArray

float   *ExpectationVals;                       /*Storage for dem in, and expectations out*/
struct  header h;                               /*File in and out header*/
int     radx[512], rady[512], RadMax;           /*Storage for circle coordinates, a distance radius about a point,*/
                                                /*and the amount of values stored in the arrays*/
int     Radius;                                 /*The radius of the circle about a point to be examined*/
int     XNumbSquares, YNumbSquares;             /*The size of the side of area to be examined*/
int     XSpaceSquares, YSpaceSquares;           /*The interval between the center of each square*/
float   InitialDx, LastDx, PlusDx;              /*First and last radius size and increment*/
long    FrameSize;                              /*The size of each frame of expectations to be output*/
int     Frames, NewRows, NewCols;               /*Number of frames, with row and column dimensions*/
int     StartSquareRow, StartSquareCol; /*Precalculated start positions*/
short   *left_edge_ptr[TEST_HEIGHT];            /*Precalculate left edge of current rectangle*/
float   Xvar[MaxArray], Yvar[MaxArray], Xvarvar[MaxArray];
float   Yvarvar[MaxArray], Covar[MaxArray], Covarvar[MaxArray];
long    Xnumb[MaxArray], Ynumb[MaxArray], Conumb[MaxArray];
                                                /*Storage for the values for one tile*/
float   pi = 3.1415927;
int     MaxUseArray;                            /*Maximum used value for the above arrays*/
int     AllSpaceSquares;                        /*Maximum spacing, in terms of grid squares*/

short   *dem;                           /*Storage for DEM*/
float   Threshold;                      /*The threshold height below which no points will be considered*/
int     Increment, LatitudeOffset, LongitudeOffset;
                                        /*Units and start corner of the array of heights*/
                                        /*Assumption - the heights are equispaced, distance
                                          Increment in angular units apart.*/
long    ReadSize, ReadInLocation, FirstReadSize;
                                        /*Read in file buffered. Location and
                                          size of buffer*/
int     ReadInRasterFile;               /*Rasterfile DEM input flag*/

/*************************************************/
main(argc, argv)
int argc;
char *argv[];

{
        Input_File_Arguements(argc,argv);
        Read_in_DEM();
        Input_Arguements(argc,argv);
        Get_Frame_Memory();
        Get_All_Expectations();
        Write_Expectations(argc,argv);
}/*End of main*/


/******SetUp Routines**********/

Input_File_Arguements(argc,argv)
int argc;
char *argv[];

{
int argn, find_string();

        if ((argn=find_string("-rasterfile",argc,argv)) != -1)
                ReadInRasterFile = TRUE;
        else
                ReadInRasterFile = FALSE;
}

Input_Arguements(argc, argv)
```

```
/*Input the commmand line arguements*/
int argc;
char *argv[];

{
int argn, find_string();

        if ((argn=find_string("-x",argc,argv)) != -1)
        {
                sscanf(argv[argn+1],"%d",&XNumbSquares);
                if (XNumbSquares > h.cols) {fprintf(stderr,"X side size is larger than image size.\n");exit(1);}
                if (XNumbSquares <= 0   ) {fprintf(stderr,"X side size is zero or less.\n");exit(1);}
        }
        else
                XNumbSquares = h.cols;

        if ((argn=find_string("-y",argc,argv)) != -1)
        {
                sscanf(argv[argn+1],"%d",&YNumbSquares);
                if (YNumbSquares > h.rows) {fprintf(stderr,"Y side size is larger than image size.\n");exit(1);}
                if (YNumbSquares <= 0   ) {fprintf(stderr,"Y side size is zero or less.\n");exit(1);}
        }
        else
                YNumbSquares = h.rows;

        if ((argn=find_string("-r",argc,argv)) != -1)
        {
                sscanf(argv[argn+1],"%d",&XSpaceSquares);
                if (XSpaceSquares > h.cols) {fprintf(stderr,"X interval size is larger than image size.\n");exit(1);}
                if (XSpaceSquares <= 0   ) {fprintf(stderr,"X interval size is zero or less.\n");exit(1);}
        }
        else
                XSpaceSquares = XNumbSquares;

        if ((argn=find_string("-c",argc,argv)) != -1)
        {
                sscanf(argv[argn+1],"%d",&YSpaceSquares);
                if (YSpaceSquares > h.rows) {fprintf(stderr,"Y interval size is larger than image size.\n");exit(1);}
                if (YSpaceSquares <= 0   ) {fprintf(stderr,"Y interval size is zero or less.\n");exit(1);}
        }
        else
                YSpaceSquares = YNumbSquares;

        if ((argn=find_string("-i",argc,argv)) != -1)
        {
                sscanf(argv[argn+1],"%f",&InitialDx);
                if (InitialDx <= 0.0  ) {fprintf(stderr,"Minimum pair spacing is zero or less.\n");exit(1);}
        }
        else
                InitialDx = 1.0;

        if ((argn=find_string("-l",argc,argv)) != -1)
        {
                sscanf(argv[argn+1],"%f",&LastDx);
                if (LastDx <= 0.0   ) {fprintf(stderr,"Maximum pair spacing is zero or less.\n");exit(1);}
                if (LastDx < InitialDx) {fprintf(stderr,"Maximum pair spacing is less than minimum pair spacing.\n");exit(1);}
        }
        else
                LastDx = InitialDx;

        if ((argn=find_string("-p",argc,argv)) != -1)
        {
                sscanf(argv[argn+1],"%f",&PlusDx);
                if (PlusDx <= 0.0   ) {fprintf(stderr,"Increment of  pair spacing is zero or less.\n");exit(1);}
        }
        else
                PlusDx = 1.0;

        MaxUseArray =(int)((LastDx - InitialDx) / PlusDx);
        if (MaxUseArray > MaxArray) MaxUseArray = MaxArray;

        if ((argn=find_string("-t",argc,argv)) != -1)
                sscanf(argv[argn+1],"%f",&Threshold);
        else
                Threshold = -40000.0;

/*Lower left hand corner coordinates - angular coordinates.
  Units in terms of 1/Increment fraction of one circumference.
  ie Increment = 360 gives units of degrees.
     Increment = 21600 gives units of minutes.*/

        if ((argn=find_string("-inc",argc,argv)) != -1)
                sscanf(argv[argn+1],"%d",&Increment);
        else
                Increment = 1;

        if ((argn=find_string("-lat",argc,argv)) != -1)
        {
                sscanf(argv[argn+1],"%d",&LatitudeOffset);
                /*Image lines start at northernmost point, so want the
```

```
                offset from the top of the image from the North Pole.
                Thus take offset from equator + image height, and
                subtract that from 90 degrees, all in Increment units.*/
                LatitudeOffset = Increment/4 - (LatitudeOffset + h.rows);
        }
        else
                LatitudeOffset = 0;

        if ((argn=find_string("-long",argc,argv)) != -1)
                sscanf(argv[argn+1],"%d",&LongitudeOffset);
        else
                LongitudeOffset = 0;
}

/*Get the frames to store the expectation values in*/
Get_Frame_Memory()
{
long    lnFrameSize;
    /*Nine frames per DX value.
        One for the variance, another for variance squared,
        and finally one for number of points analysed,
        repeated for x,y and covariance.*/
        Frames = MaxUseArray * 9;
        NewRows = ((h.rows - YNumbSquares) / YSpaceSquares) + 1;
        NewCols = ((h.cols - XNumbSquares) / XSpaceSquares) + 1;
        FrameSize = NewRows*NewCols;

        if ( (ExpectationVals=(float*)calloc(Frames*FrameSize, sizeof(float))) == NULL)
                {fprintf(stderr,"Can't get  new frames storage.\n");exit(1);}

        lnFrameSize = h.cols*YNumbSquares;

        if ( (dem=(short*)calloc(lnFrameSize,sizeof(short))) == NULL)
                {fprintf(stderr,"Can't get store\n");exit(1);}

        ReadSize = h.cols * YSpaceSquares;

        FirstReadSize = h.cols * (YNumbSquares - YSpaceSquares);

        ReadInLocation = h.cols * (YNumbSquares - YSpaceSquares);

        pread(0,dem,FirstReadSize*sizeof(short));
}

/*Work out all Expectation values, in order of Dx size, then square by square*/
Get_All_Expectations()
{
int     ExpRows, ExpCols;
register        int     FrameCount;

        if (YSpaceSquares < XSpaceSquares)
                AllSpaceSquares = YSpaceSquares / 2;
        else
                AllSpaceSquares = XSpaceSquares / 2;

        for (ExpRows = 0; ExpRows < NewRows; ExpRows++)
        {
                Read_Next_Lines();
                StartSquareRow = ExpRows * YSpaceSquares;
                for (ExpCols = 0; ExpCols < NewCols; ExpCols++)
                {
                        StartSquareCol = ExpCols * XSpaceSquares;
                        GetDiffs();
                        /*Write the binned variances out to frame store*/
                        /*Note bins 0 and MaxUseArray + 1 are used to store values
                          under and over the limits the user specifies*/
                        for (FrameCount = 0; FrameCount < MaxUseArray; FrameCount++)
                        {
                                SetExpValue(ExpCols,ExpRows,FrameCount*9,Xvar[FrameCount+1]);
                                SetExpValue(ExpCols,ExpRows,FrameCount*9+1,Xvarvar[FrameCount+1]);
                                SetExpValue(ExpCols,ExpRows,FrameCount*9+2,Yvar[FrameCount+1]);
                                SetExpValue(ExpCols,ExpRows,FrameCount*9+3,Yvarvar[FrameCount+1]);
                                SetExpValue(ExpCols,ExpRows,FrameCount*9+4,Covar[FrameCount+1]);
                                SetExpValue(ExpCols,ExpRows,FrameCount*9+5,Covarvar[FrameCount+1]);
                                SetExpValue(ExpCols,ExpRows,FrameCount*9+6,(float)Xnumb[FrameCount+1]);
                                SetExpValue(ExpCols,ExpRows,FrameCount*9+7,(float)Ynumb[FrameCount+1]);
                                SetExpValue(ExpCols,ExpRows,FrameCount*9+8,(float)Conumb[FrameCount+1]);
                        }
                }
        }
}

Read_Next_Lines()
/*Read successive lines of the DEM into the program buffer*/
{
        bcopy(dem + ReadSize, dem, FirstReadSize * sizeof(short));
        pread(0, dem + ReadInLocation, ReadSize * sizeof(short));
}

GetDiffs()
```

```
/*Get the expected values for a particular area of the DEM*/
/*The difference, the variance, and the variance of the variance,
  for confidence limits*/
{
register        int     ximage, yimage, xpointon, ypointon;
register        float   numbpoints, centerpix, variance, differencex, differencey;

float   difference;
int     OnePixelDiff();
float   DEMValue();
float   CalcLongitudeDist(), CalcLatitudeDist(), LongitudeDist, LatitudeDist, CovarDist;
int     ArrayLatDist, ArrayLongDist, ArrayCovarDist, CurrentRadius;
int     Radius;
float   Xtotalvariance,Xtotalvariancevariance,Ytotalvariance,Ytotalvariancevariance;
float   totalcovariance,totalcovariancevariance;
long    xnumbpoints,ynumbpoints,covnumbpoints;

        /*For each tile, zero the variance bins*/
        for (Radius = 0; Radius < MaxUseArray; Radius++)
        {
                Xvar[Radius] = 0.0; Yvar[Radius] = 0.0; Xvarvar[Radius] = 0.0;
                Yvarvar[Radius] = 0.0; Covar[Radius] = 0.0; Covarvar[Radius] = 0.0;
                Xnumb[Radius] = 0; Ynumb[Radius] = 0; Conumb[Radius] = 0;
        }

        /*Set up the edge of tile rows pointers*/
        for (yimage = 0; yimage < YNumbSquares; yimage++)
        {
                left_edge_ptr[yimage] = dem + (((long)yImage) * (long)h.cols + (long)StartSquareCol);
        }

        /*For all point spacings*/
        for (CurrentRadius = 1; CurrentRadius < AllSpaceSquares; CurrentRadius++)
                for (yimage = 0; yimage < YNumbSquares - CurrentRadius; yimage++)
                {
                        /*For one row, all the longitude and latitude distances along rows and
                          columns will be the same - BIG time saving - calculate once, then
                          calculate the appropriate array indexing values to store variances and
                          counts and use in the row loop*/
                        LongitudeDist = CalcLongitudeDist(0,CurrentRadius,yimage + StartSquareRow + LatitudeOffset);
                        ArrayLongDist = (int)((LongitudeDist - InitialDx) / PlusDx + 0.5) + 1;
                        if (ArrayLongDist > MaxUseArray) ArrayLongDist = MaxUseArray + 1;
                        else if (ArrayLongDist < 1) ArrayLongDist = 0;

                        LatitudeDist = CalcLatitudeDist(yimage, yimage +CurrentRadius);
                        ArrayLatDist = (int)((LatitudeDist - InitialDx) / PlusDx + 0.5) + 1;
                        if (ArrayLatDist > MaxUseArray) ArrayLatDist = MaxUseArray + 1;
                        else if (ArrayLatDist < 1) ArrayLatDist = 0;

                        CovarDist = (float)sqrt((double)LongitudeDist * (double)LongitudeDist
                                        + (double)LatitudeDist*(double)LatitudeDist);
                        ArrayCovarDist = (int)((CovarDist - InitialDx) / PlusDx + 0.5) + 1;
                        if (ArrayCovarDist > MaxUseArray) ArrayCovarDist = MaxUseArray + 1;
                        else if (ArrayCovarDist < 1) ArrayCovarDist = 0;

                        /*Load array values into local variables to save time*/
                        Xtotalvariance = Xvar[ArrayLongDist]; Ytotalvariance = Yvar[ArrayLatDist];
                        Xtotalvariancevariance = Xvarvar[ArrayLongDist];
                        Ytotalvariancevariance = Yvarvar[ArrayLatDist];
                        totalcovariance = Covar[ArrayCovarDist];
                        totalcovariancevariance = Covarvar[ArrayCovarDist];
                        xnumbpoints = Xnumb[ArrayLongDist]; ynumbpoints = Ynumb[ArrayLatDist];
                        covnumbpoints = Conumb[ArrayCovarDist];

                        /*For all columns in the tile*/
                        for (ximage = 0; ximage < XNumbSquares - CurrentRadius; ximage++)
                        {
                                centerpix = DEMValue(ximage,yimage);
                                /*If the value to be compared is greater than the minimum altitude*/
                                if (centerpix > Threshold)
                                {
                                        /*Calculate the variances in latitude, longitude and covariance directions*/
                                        xpointon = 0; ypointon = 0;
                                        if ( OnePixelDiff(ximage + CurrentRadius, yimage, centerpix, &difference) )
                                        {
                                                xpointon = 1;
                                                differencex = difference;
                                                variance = differencex * differencex;
                                                Xtotalvariance = Xtotalvariance + variance;
                                                Xtotalvariancevariance = Xtotalvariancevariance + variance*variance;
                                                xnumbpoints++;
                                        }
                                        if ( OnePixelDiff(ximage, yimage + CurrentRadius, centerpix, &difference) )
                                        {
                                                ypointon = 1;
                                                differencey = difference;
                                                variance = differencey * differencey;
                                                Ytotalvariance = Ytotalvariance + variance;
                                                Ytotalvariancevariance = Ytotalvariancevariance + variance*variance;
                                                ynumbpoints++;
```

```
                                        }
                                        if (xpointon == 1 && ypointon == 1)
                                        {
                                                variance = differencex * differencey;
                                                totalcovariance = totalcovariance + variance;
                                                totalcovariancevariance = totalcovariancevariance + variance*variance;
                                                covnumbpoints++;
                                        }
                                }
                        }
                        /*Put the temporary values back in the array entries*/
                        Xvar[ArrayLongDist] = Xtotalvariance; Yvar[ArrayLatDist] = Ytotalvariance;
                        Xvarvar[ArrayLongDist] = Xtotalvariancevariance;
                        Yvarvar[ArrayLatDist] = Ytotalvariancevariance;
                        Covar[ArrayCovarDist] = totalcovariance;
                        Covarvar[ArrayCovarDist] = totalcovariancevariance;
                        Xnumb[ArrayLongDist] = xnumbpoints; Ynumb[ArrayLatDist] = ynumbpoints;
                        Conumb[ArrayCovarDist] = covnumbpoints;
                }

        /*Have all figures for a tile - average the totals, if any data there*/
        for (Radius = 0; Radius < MaxUseArray; Radius++)
        {
                xnumbpoints = Xnumb[Radius];
                if (xnumbpoints != 0)
                {
                        Xvar[Radius] = Xvar[Radius] / (float)xnumbpoints;
                        Xvarvar[Radius] = Xvarvar[Radius] / (float)xnumbpoints;
                }
                ynumbpoints = Ynumb[Radius];
                if (ynumbpoints != 0)
                {
                        Yvar[Radius] = Yvar[Radius] / (float)ynumbpoints;
                        Yvarvar[Radius] = Yvarvar[Radius] / (float)ynumbpoints;
                }
                covnumbpoints = Conumb[Radius];
                if (covnumbpoints != 0)
                {
                        Covar[Radius] = Covar[Radius] / (float)covnumbpoints;
                        Covarvar[Radius] = Covarvar[Radius] / (float)covnumbpoints;
                }
        }
}

float   CalcLatitudeDist(y1,y2)
int     y1,y2;
{
float   EquatorRadius = 6378137.0, PolarRadius = 6356752.3;
float   AvDistance = 6367444.7;
float   result;
int     ydist;

        if (y1 > y2)
                ydist = y1 - y2;
        else
                ydist = y2 - y1;
/*circumf = Avdist * 2 * pi = 2pi*r.
  frac of circ = ydist/inc*/
        result = 2.0 * ((float)ydist) * AvDistance / ((float)Increment) * pi;
        return(result);
}

float   CalcLongitudeDist(x1,x2,y)
int     x1,x2,y;
{
float   EquatorRadius = 6378137.0, PolarRadius = 6356752.3;
float   AvDistance = 6367444.7;
float   result;
int     xdist;

        if (x1 > x2)
                xdist = x1 - x2;
        else
                xdist = x2 - x1;

/*circumf = Equator radius* 2 * pi = 2pi*r.
  frac of circ = xdist/inc
  Half the width to cancel the multiplication by 2 in 2*PI*r.*/
        result = 2.0 * ((float)xdist) * EquatorRadius / ((float)Increment) * pi;
/*Have the equitorial radius. Now must scale for the particular latitude*/
        if (y > (Increment/4)) y = Increment/2 - 1 - y;
        result = result * (float)sin(2.0 * (double)(y) / ((float)Increment) * (double)pi);
        return(result);
}


/*.....*/
/*************Subroutines for manipulating HIPs files**************/
/*************************************************************/
```

```
Read_in_DEM()
/*Read in a DEM, placing the header in a global variable h,
  and the data in a location returned in pointer dem.*/
{
        if (ReadInRasterFile)
        {struct rasterfile raster_header;
                fprintf(stderr,"Reading in rasterfile format\n");
                if (read(0,&raster_header,sizeof(struct rasterfile)) != sizeof(struct rasterfile))
                        {fprintf(stderr,"Error reading rasterfile headers\n");exit(1);}
/*              if (raster_header.ras_magic != RAS_MAGIC || raster_header.ras_depth != 16 || raster_header.ras_type != RT_STANDARD)
        Ignore rasterfile type for the moment, as Upstairs produces non standard types*/
                if (raster_header.ras_magic != RAS_MAGIC || raster_header.ras_depth != 16)
                        {fprintf(stderr,"The rasterfile is not a 16 bit non runlength encoded pixrect\n");exit(1);}
                if (raster_header.ras_maptype != RMT_NONE && raster_header.ras_maplength > 0)
                        {char dummy[768];read(0,dummy,raster_header.ras_maplength);}

                h.cols=raster_header.ras_width;
                h.rows=raster_header.ras_height;
                h.pixel_format=PFSHORT;
                h.num_frame=1;
        }
        else
        {
                fprintf(stderr,"Reading in HIPs format\n");
                fread_header(0,&h);
                if (h.num_frame != 1)
                        {fprintf(stderr,"There must only be one frame.\n");exit(1);}
                if (h.pixel_format != PFSHORT)
                        {fprintf(stderr,"Must be short format\n");exit(1);}
        }
}


Write_Expectations(argc,argv)
/*Input the commmand line arguements*/
int argc;
char *argv[];

/*Write a HIPs file whose header is in variable h, and whose data
  is pointed to by pointer ExpectationVals*/
{

        init_header(&h,"David's Geoid RowCol variogram data","???",1,"???",NewRows,NewCols,8,0,PFFLOAT,"");
        h.num_frame=Frames;
        h.rows = NewRows;
        h.cols = NewCols;

        update_header(&h,argc,argv);

        update_desc(&h," Variance, Variance variance, and point pair count values for areas of the DEM.");
        update_desc(&h," The above repeated for horizontal, vertical and covariance comparisons.");
        update_desc(&h," Horizontal and vertical grid comparisons only.");

        write_header(&h);
        write(1,ExpectationVals,Frames*FrameSize*sizeof(float));
}


/*.....*/
/*****************routines to find DEM expected values************/
/*""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""*/


int OnePixelDiff(xa ,ya ,centerpixvalue, diff)
int     xa, ya;
float   centerpixvalue;
register float *diff;
{
float   val;
float   DEMValue();

        if ((xa > -1) && (xa < XNumbSquares) && (ya > -1) && (ya < YNumbSquares))
        {
                val=DEMValue(xa,ya);
                if (val > Threshold)
                {
                        *diff = (centerpixvalue - val);
                        return(1);                      /* success */
                }
        }
        return(0);              /* failure */
}

/*Abstract data types from program - always return float
  no matter what the DEM format*/
float DEMValue(x,y)
int     x,y;
{
short   *ValuePtr, Value;
```

```
        ValuePtr = left_edge_ptr[y] + (long)x;
        Value = *ValuePtr;
        return((float)Value);
}


float FixDEMValue(x,y)
int    x,y;
{
short   *ValuePtr, Value;

        ValuePtr = dem + (long)y * (long)TEST_WIDTH  + (long)x;
        Value = *ValuePtr;
        return((float)Value);
}

SetDEMValue(x,y,Value)
int         x,y;
float       Value;
{
short       *ValuePtr;

        ValuePtr = dem + (long)y * (long)TEST_WIDTH  + (long)x;
        *ValuePtr = (short)Value;
}

SetExpValue(x,y,frame,value)
int         x,y,frame;
float       value;
{
float   *ValuePtr;

        ValuePtr = (float*)((long)ExpectationVals+((long)frame*(long)FrameSize+(long)y*(long)NewCols+(long)x)*sizeof(float));
        *ValuePtr = value;
}

/*Command line arguement subroutines*/
/*
find_switch (c,argc,argv)
char c;
int argc;
char *argv[];
{    int i;
     for (i=0;i<argc;i++) if (*argv[i] == '-' && *(argv[i]+1) == c && *(argv[i]+2) == '\0') return(TRUE);
     return (FALSE);
}
*/
int find_string (s,argc,argv)
char *s;
int argc;
char *argv[];
{       int i;
        for (i=0;i<argc;i++) if (strcmp(s,argv[i]) == 0) return(i);
        return(-1);
}
```

Bibliography

[Allison 89] Allison D. (1989): *Extraction of Planetary Manned Landing Sites From Surface Microstructure Using Fractal Image Analysis*, Internal report - Dept. Of Photogrammetry and Surveying, University College London.

[Burrough 81] Burrough P. A. (1981): *Fractal dimensions of landscapes and other environmental data*, Nature, 294, 240-242, 19 November 1981.

[Burrough 83a] Burrough P. A. (1983): *Multiscale sources of spatial variation in soil. 1. The application of fractal concepts to nested levels of soil variation*, Journal of Soil Science, 1983, 34, 577-597.

[Burrough 84] Burrough P. A., *Fakes, facsimilies and facts: fractal models of geophysical phenomena*, Proc. of IBM Conference (ed. S Nash), London, 150-169, (March 1984).

[Culling 87] Culling W. E. H., Datko M. (1987): *The Fractal Geometry of the Soil-Covered Landscape*, Earth Surface Processes and Landforms, 12, 369-385, 1987.

[Culling 88] Culling W. E. H., Datko M. (1987): *Dimension and Entropy in the Soil-Covered Landscape*, Earth Surface Processes and Landforms, 13, 619-648, 1987.

[Dutton 81] Dutton, G. H. (1981): *Fractal Enhancement of Cartographic Line Detail*, The American Cartographer, Vol 8, No 1, 23-40, 1981.

[Dodd 87] Dodd N. (1987): *Multispectral Texture Synthesis Using Fractal Concepts*, IEEE Pattern Analysis and Machine Int., PAMI-9, 5, 703-707.

[Elliot 89] Elliot J. K. (1989): *An investigation of the change in surface roughness through time on the foreland of Austre Okstindbreen, North Norway*, Computers and Geosciences, 15, 2, 209-217.

[Falconer 90] Falconer, k.j. (1990): *Fractal Geometry : Mathematical Foundations and Applications*, John Wiley and Sons, Chichester, England.

[Forstner 83] Forstner W. (1983): *On the Morphological Quality of Digital Elevation Models*, Proc. of the Int. Colloquium on "Mathematical Aspects of DEMs", Dept of Photogrammetry, Royal Institute of Technology, Stockholm, Sweden, 18-20 April 1983.

[Fournier 82] Fournier A., Fussell D. and Carpenter L. (1982): *Computer Rendering of Stochastic Models*, Comm. ACM, 25, 6, 371-384.

[Frederiksen 83] Frederiksen P., Jacobi O., Kubik K. (1983): *Measuring Terrain Roughness by Topological Dimension*, Proc. of the Int. Colloquium on "Mathematical Aspects of DEMs", Dept of Photogrammetry, Royal Institute of Technology, Stockholm, Sweden, 18-20 April 1983.

[Gabriel 88] Gabriel P., Lovejoy S., Schertzer D., Austin G. L. (1988): *Multifractal Analysis of Resolution Dependence in Satellite Imagery*, Geophysical Research Letters, Vol 15, No 12, 1373-1376, November 1988.

[Goodchild 80] Goodchild M. F. (1980): *Fractals and the Accuracy of Geographical Measures*, Mathematical Geology, Vol 12, No 2, 85-98.

[Goodchild 82] Goodchild M. F. (1982): *The Fractional Brownian Process as a Terrain Simulation Model* Proceedings, Thirteenth Annual Pittsburg Conference on Modelling and Simulation, vol 13, 1133-1137.

[Goodchild 87] Goodchild M. F., Mark D. M. (1987): *The Fractal Nature of Geographic Phenomena*, Annals of the Association of American Geographers, 77, 2, 265-278.

[Gupta 89] Gupta V. K., Waymire E. (1989): *Statistical Self-Similarity in River Networks Parameterized by Elevation*, Water Resources Research, 25, 3, 463-476.

[HIPL] *Human Information Processing Lab*, Department of Psychology, New York University, NY 1003, United States of America.

[Hough 89] S.E.Hough, *On the Use of Spectral Methods for the Determination of Fractal Dimension*, Geophysical Research Letters, Vol 94. No B6, 7491-7495, June 10, 1989. 5/89.

[Huang 89] Huang J., Turcotte D. L. (1989): *Fractal Mapping of Digitized Images: Application to the Topography of Arizona and Comparisons With Synthetic Images*, Journal of Geophysical Research, Vol 94, No B6, 7491-7495, June 10, 1989.

[Hyde 85] Hyde J., Fullwood J. A. and Corrall D. R. (1985): *An Approach to Knowledge-Driven Segmentation*, Image and Vision Computing, 3, 4, 198-205.

[Isacks 88] Isacks B. L. (1988): *Uplift in the Central Andean Plateau and Bending of the Bolivian Orocline*, Journal of Geophysical Research, Vol 93, No B4, 3211-3231, April 10, 1988.

[Jones 85] Jones J. G., Thomas R. W. (1985): *Texture Analysis of Natural Features in Satellite Imagery Using a Fractal Based Model*, Remote Sensing Society / British Pattern Recognition Association One-day Workshop on Image Understanding in Remote Sensing, University College London, England, 10 July 1985, 2, 1-21.

[Jones 89] Jones J. G., Thomas R. W., and Earwicker P. G. (1989): *Fractal properties of computer generated and natural geophysical data*, Computers and Geosciences, 15, 2, 227-235.

[Julesz 81] Julesz B. (1981): *Textons, the Elements of Texture Perception and their Interactions*, Nature, 290, 12 March, 91-96.

[Kazu 91] Kazuhiko O. (1991): *Fractal dimensions of terrain profiles*, Journal of Terramechanics, 28, 2/3, 155-165.

[Keller 87] Keller J. M., Crownover R. M., Chen R. Y. (1987): *Characteristics of Natural Scenes Related to the Fractal Dimension*, IEEE Trans. Pattern Analysis and Machine Int., PAMI-9, 5, 621-627.

[Keller 89] Keller J. M., Chen S., Crownover R. M. (1989): *Terain Description and Segmentation through Fractal Geometry*, Computer Vision, Graphics and Image Processing, 45, 150-166.

[Klinkenberg 92] Klinkenberg B., Goodchild M. F. (1992): *The Fractal Propoerties of Topography: A Comparison of Methods*, Earth Surface Processes and Landforms, 17, 217-234.

[Kube 88] Kube P., Pentland A. P. (1988): *On the Imaging of Fractal Surfaces*, IEEE Trans. Pattern Analysis and Machine Intelligence, PAMI-10, 5, 704-707.

[Lam 90] Lam N. S. (1990): *Description and Measurement of Landsat TM Images Using Fractals*, Photogrammetric Engineering and Remote Sensing, 56, 2, 187-195, February 1990.

[Lavallee 92] Lavallee D., Lovejoy S., Schertzer D., Ladoy P. (1992): *Non-linear Variability of Landscape Topography: Multifractal Analysis and Simulation*, Fractals in Geography, ed. L. De Cola, Prentice Hall, New York. (to appear).

[Laverty 87] Laverty J. P. (1987): *Fractals in Karst*, Earth Surface Processes and Landforms, 12, 475-480.

[Lewis 87] Lewis J. P. (1987): *Generalized Stochastic Subdivision*, ACM Transactions on Computer Graphics, Vol 6, No 3, 167 - 190, July 1987.

[Lovejoy 82] Lovejoy S. (1982): *Area Perimeter Relations for Rain and Cloud Areas*, Science 216, 185.

[Lovejoy 85] Lovejoy S., Schertzer D. (1985): *Scale and Dimension Dependence in Remote Sensing: Analysis and Simulation* One-day Workshop on Image Understanding in Remote Sensing, University College London, England, 10 July 1985.

[Lovejoy 87] Lovejoy S., Schertzer D., Tsonis A. A. (1987): *Functional Box-Counting and Multiple Elliptical Dimensions in Rain*, Science, Vol 235, 1036-1038, 27 February 1987.

[Lovejoy 89] Lovejoy S., Schertzer D. (1989): *Scale Invariance and Multifractals in the Atmosphere*, Pergamon Encyclopaedia of the Environment, 1989.

[Lovejoy 90a] Lovejoy S., Schertzer D. (1990): *Our Multifractal Atmosphere: A Unique Laboratory for Non-linear Dynamics*, Physics in Canada, 46, 4.

[Lovejoy 90b] Lovejoy S., Schertzer D. (1990): *Fractals, Raindrops and Resolution Dependence of Rain Measurements*, Journal of Applied Meteorology, 29, 11, 1167-1170, November 1990

[Mandelbrot 67] Mandelbrot, B. B. (1967): *How long is the Coast of Britian? Statistical Self-Similarity and Fractional Dimension* Science 155, 636 - 638.

[Mandelbrot 75] Mandelbrot, B. B. (1975): *Stochastic Models of the Earth's Relief, the Shape and the Fractal Dimension of the Coastlines, and the Number-Area Rule for Islands* Journal of the Nat Acad Sci USA 72, No 10, 3825-3828, October 1975.

[Mandelbrot 77] Mandelbrot B. B. (1977): *Fractals, Form, Chance and Dimension*, W. H. Freeman, San Francisco.

[Mandelbrot 82] Mandelbrot B. B. (1982): *The Fractal Geometry of Nature*, W. H. Freeman, San Francisco.

[Mandelbrot 85] Mandelbrot B. B. (1985): *Notes on Fractal Curves and Surfaces*, Fractals: Basic Concepts, Computation and Rendering, Course notes, SIGGRAPH 85.

[Mark 84] Mark D. M., Aronson P. B., *Scale-Dependent Fractal Dimensions of Topographic Surfaces: An Empirical Investigation, with Applications in Geomorphology and Computer Mapping* Journal of Mathematical Geology, Vol 16, No 7, 1984.

[Miller 86] Miller G. S. P. (1986): *The Definition and Rendering of Terrain Maps*, SIGGRAPH 86, Computer Graphics, 20, 4, 39-47.

[Muller 86] Muller J-C (1986): *Fractal Dimension and Inconsistencies in Cartographic Line Representations*, The Cartographic Journal, Vol 23, No 2, 123-130, December 1986.

[Muller 90a] Muller J. P., Eales P. (1990): *Requirement Specification for Global Topography for EOS*, IGGARS '90, 1411 - 1414, Vol. II, WP-3.

[Muller 90b] Muller J. P. et al. (1990): *Quality Assessment of Complete SPOT-DEMs with Digital Multi-Source Data*, IGGARS '90, 1971 - 1974, Vol. III, THA-9.

[Muller 90c] Muller J. P., et al. (1990a): *Application of Monte Carlo ray-tracing and automatically stereo-matched SPOT-DEMs to understanding LANDSAT-TM images* IGGARS '90, 1779 - 1780, Vol. III, THA-3.

[Musgrave 89] Musgrave F. K., Kolb C. E., Mace R. S. (1989): *The Synthesis and Rendering of Eroded Fractal Terrains*, SIGGRAPH 89, Computer Graphics, 23, 3, 41-50.

[Newman 90] Newman W. I. and Turcotte D. L. (1990): *A cascade model for fluvial geomorphology*, International Geogphysics Journal, 100, 433-439.

[Peleg 84] Peleg S., Naor J., Hartley R., Avnir D. (1984): *Multiple Resolution Texture Analysis and Classification*, IEEE Trans. Pattern Analysis and Machine Int., PAMI-6, 4, 518-523.

[Pentland 84] Pentland A. P. (1984): *Fractal-Based Description of Natural Scenes*, IEEE Trans. Pattern Analysis and Machine Int., PAMI-6, 6, 661-674.

[Pentland 85] Pentland A. P. (1985): *On Describing Complex Surface Shapes*, Image and Vision Computing, 3, 4, 153-162.

[Rees 90a] Rees D., Muller J.P. (1990): *Automated Terrain Segmentation Using Fractal Variogram Analysis*, Under preparation for International Journal of Remote Sensing.

[Rees 90b] Rees D., Muller J.P. (1990): *Terrain Interpolation via Fractal Feature Enhancement*, Submitted to SIGGRAPH '90.

[Rowntree 86] Rowntree P. R., Sangster A. B. (1986): *Remote Sensing Needs Identified in Climate Modelling Experiments with Hydrological and Albedo Changes in the Sahel*, ESA SP-248. (Proc. ISLSCP Conference, Rome, Italy, 2-6 December 1985).

[Sarma 90] Sarma G., Wilson J., Schertzer D., Lovejoy S. (1990): *Universal Multifractal Cascade Models of Rain and Clouds*, Conference on Cloud Physics, San Francisco, 24-27 July 1990.

[Saupe 88] Saupe D. (1988): *Algorithms for Random Fractals*, The Science of Fractal Images, 71-136. Springer-Verlag

[Saupe 89] Saupe D. (1989): *Random Fractals in Image Synthesis*, Fractals and Chaos, British Computer Society Displays Group Seminar, Birkbeck College, London, December 1989.

[Schertzer 87] Schertzer D., Lovejoy S. (1987): *Physical Modelling and Analysis of Rain and Clouds by Anisotropic Scaling Multiplicative Processes*, Journal of Geophysical Research, 92, D8, 9693-9714, August 20, 1987.

[Schertzer 89] Schertzer D., Lovejoy S. (1989): *Non-linear Variability in Geophysics: Multifractal Simulations and Analysis*, Fractals' Physical Origin and Properties, ed. L. Pietronero, Plenum Press, New York, 49-79.

[Schertzer 91] Schertzer D., Lovejoy S., Lavallee D., Schmitt F. (1989): *Universal Hard Multifractal Turbulence: Theory and Observations*, Non-linear Dynamics of Structures, eds Sagdeer R. Z., Frisch U., Moiseer S., Erkhim N., World Scientific 1991.

[Spiegel 72] Spiegel M. R. (1972): *Theory and Problems of Statistics (SI edition)*, Schaum's Outline Series in Science, McGraw-Hill.

[Turcotte 87] Turcotte D. L. (1987): *A Fractal Interpretation of Topography and Geoid Spectra on the Earth, Moon, Venus and Mars*, Procs. of the Seventeenth Lunar and Planetary Science Conf., Part 2, Journal of Geophysical Research, Vol 92, No B4, E597-E601, March 30, 1987.

[Voss 88] Voss R. F. (1988): *Fractals in Nature: From Characterization to Simulation*, The Science of Fractal Images, 21-70. Springer-Verlag.

[Wall 89] Wall S. D., Farr T. G., Muller J-P. Lewis P., Leberl F. W. (1989): *Measurement of Surface Microtopgraphy: Helicopter-Mounted Film Cameras and two Stereo-Matching Techniques are used to Estimate the RMS Roughness of a Lava Flow*, IGARSS '89, (12th Canadian Symposium on Remote Sensing), III, 1173-1176.

[Wall 91] Wall S. D., Farr T. G., Muller J-P. Lewis P., Leberl F. W. (1991): *Measurement of Surface Microtopgraphy: Helicopter-Mounted Film Cameras and two Stereo-Matching Techniques are used to Estimate the RMS Roughness of a Lava Flow*, Photogrammetric Engineering and Remote Sensing, August 1991.

[Wilson 91] Wilson J., Schertzer D., Lovejoy S. (1991): *Continuous Multiplicative Cascade Models of Rain and Clouds*, Non-linear Variability in Geophysics, eds. Schertzer D., Lovejoy S., 185-207, 1991.

[Woronow 81] Woronow A. (1981): *Morphometric Consistency with the Hausdorff-Besicovich Dimension*, Mathematical Geology, Vol 13, No 3, 1981.

[Yokoya 88] Yokoya N., Yamamoto K., Funakubo N. (1988): *Fractal-Based Analysis and Interpolation of 3D Natural Surface Shapes and Their Application to Terrain Modelling*, ETL Technical Report, 1-24.

[Yokoya 89] Yokoya N., Yamamoto K., Funakubo N. (1989): *Fractal-Based Analysis and Interpolation of 3D Natural Surface Shapes and Their Application to Terrain Modelling*, Computer Vision, Graphics, and Image Processing, 46, 284-302.