GEOLOGICA ULTRAIECTINA

Mededelingen van de
Faculteit Aardwetenschappen der
Rijksuniversiteit te Utrecht

No. 83

THE SHORTEST PATH METHOD FOR SEISMIC RAY TRACING IN COMPLICATED MEDIA

T.J. MOSER

GEOLOGICA ULTRAIECTINA

Mededelingen van de
Faculteit Aardwetenschappen der
Rijksuniversiteit te Utrecht

No. 83

THE SHORTEST PATH METHOD FOR SEISMIC RAY TRACING IN COMPLICATED MEDIA

DE KORTSTE-ROUTEMETHODE VOOR SEISMISCHE RAYTRACING IN GECOMPLICEERDE MEDIA

CIP-GEGEVENS KONINKLIJKE BIBLIOTHEEK, DEN HAAG

Moser, Tijmen Jan

The shortest path method for seismic ray tracing in
complicated media / Tijmen Jan Moser. - [S.l. : s.n.]. -
(Geologica Ultraiectina. ISSN 0072-1026 : no. 83)
Proefschrift Rijksuniversiteit Utrecht. - Met lit. opg. -
Met samenvatting in het Nederlands.
ISBN 90-71577-37-6
Trefw.: seismologie.

THE SHORTEST PATH METHOD FOR SEISMIC RAY TRACING IN COMPLICATED MEDIA

De kortste-routemethode voor seismische raytracing in gecompliceerde media

(met een samenvatting in het Nederlands)

PROEFSCHRIFT

TER VERKRIJGING VAN DE GRAAD VAN DOCTOR AAN
DE RIJKSUNIVERSITEIT TE UTRECHT, OP GEZAG VAN
DE RECTOR MAGNIFICUS, PROF. DR J.A. VAN GINKEL,
INGEVOLGE HET BESLUIT VAN HET COLLEGE VAN DEKANEN
IN HET OPENBAAR TE VERDEDIGEN OP 22 JANUARI 1992
DES NAMIDDAGS TE 4.15 UUR

DOOR

TIJMEN JAN MOSER

GEBOREN OP 13 JANUARI 1963 TE ZWOLLE

1992

PROMOTORES: PROF. DR A.M.H. NOLET
PROF. DR K. HELBIG

Dit proefschrift werd mogelijk gemaakt met financiële steun van de Stichting voor de Technische Wetenschappen.

Het onderzoek werd uitgevoerd aan het Department of Theoretical Geophysics, Instituut voor Aardwetenschappen, Rijksuniversiteit Utrecht, Postbus 80.021, 3508 TA Utrecht, Nederland.

Durch Wildnisse kam ich, bergauf, talab;
oft ward es Nacht, dann wieder Tag ...

Richard Wagner, Parsifal

Aan mijn grootmoeder
Mevr. J.A. Loggers-van Eyck van Heslinga

Introduction

Knowledge about the propagation of seismic waves plays a central role in seismic modeling, both on a local scale, as in exploration seismics, and on a global scale, as in seismology. When the propagation of a seismic wave is known in a model of the Earth's structure, it can be compared with observations at the Earth's surface. This comparison then yields a judgement on the resemblance of the model with the real Earth. Both seismic tomography (see for instance Nolet, 1987) and seismic migration (see for instance Berkhout, 1982) are methods to reconstruct the Earth's structure from observations at its surface; tomography mainly uses the records of transmitted waves, migration of reflected waves. Both methods contain the computation of the waves in an assumed known Earth model as an essential part.

The propagation of seismic waves, however, is extremely complicated, even in simple media. The waves are deformed by gradual changes in the medium and reflected and refracted by discontinuities. Interference and focusing effects further complicate their shape. A variety of methods has been developed to describe these phenomena: finite difference and finite element methods, integral transform methods and ray tracing methods (see Chin, Hedstrom and Thigpen (1984) for a review).

One important class of methods is formed by the ray methods. Ray paths are the lines along which the seismic energy travels in the high frequency limit (Keller, 1978). They are not physical but merely mathematical idealisations that are of great practical use. When a source point has been indicated, one can calculate rays leaving it in all directions by solving the appropriate differential equations. Additional differential equations describe the amplitude along the ray (Červený, 1987). The assumption that the medium is isotropic, which means that the propagation velocity of the seismic energy is only dependent on the spatial position, simplifies the theory because the ray paths are then perpendicular to the wave fronts. Ray theory in anisotropic media, in which the propagation velocity is also dependent on the direction of the ray, is described by Vlaar (1968).

Generally, the ray path between a source and a receiver is required. There are essentially two methods to solve this two-point ray tracing problem: shooting and bending (Julian and Gubbins, 1977). Shooting is tracing ray paths from the source towards the receiver by solving the ray differential equations and, by adjusting their initial directions, trying to hit the receiver point. Bending is starting with an arbitrary curve between the source and the receiver and perturbing it in such a way that the ray differential equations are satisfied or, equivalently, the travel time along it is extremal. Both methods have their shortcomings: for shooting it may be very difficult to find the correct initial direction, because the trial ray paths may converge very slowly towards the receiver point; for bending it is never certain whether the found extremal travel time curve is only a local extremum or the global extremum generally required. These shortcomings may result in incomplete results or inadmissible computation times.

The extremal travel time property of a ray path, which is the variational equivalent of the ray differential equations, was first formulated by Fermat in 1661 as a least time property. It says that the travel time along a ray path is stationary, that is, it changes only to second order when the ray path is changed. In this century Carathéodory showed that Fermat's principle is indeed a least time principle in the sense that every sufficient small part of a ray has a minimal travel time between its end points (Keller, 1978). It provides an additional

check on a curve whether it is a real ray path or not and, also, gives rise to the application of minimisation techniques for finding seismic rays (Moser, Nolet and Snieder, 1992).

Another shortcoming of the shooting method that is inherent in ray theory is the occurrence of shadow zones, regions that cannot be reached by ray paths from one source point. In shadow zones the high frequency assumption of ray theory breaks down because these zones are reached by waves with finite wave lengths or diffracted waves. Shadow zones appear often in both global seismology and exploration seismics.

In recent years methods have been developed to avoid these difficulties (Vidale, 1988, 1990; Qin *et al.*, 1990; Saito, 1989, 1990; Van Trier and Symes, 1991; Matsuoka and Ezaka, 1990; Podvin and Lecomte, 1991). They construct travel time fields on a rectangular grid of nodes by solving the eikonal equation (Vidale, 1988, 1990; Qin *et al.*, 1990; Van Trier and Symes, 1991) or by using Huygens' principle (Saito, 1989, 1990; Podvin and Lecomte, 1991). In any method the travel times on the grid points are constructed recursively from already known travel times, starting with the source point and then expanding over the whole grid according to schemes that resemble expanding waves. All of the methods construct the travel times from the source point to all other points simultaneously in one run of calculations. This means that the two-point ray tracing problem is solved automatically for receivers at each grid point. Therefore, the grid methods are extremely efficient for the calculation of complete travel time fields. Some of the methods, however, suffer from a lack of robustness in presence of high velocity contrasts where some waves may turn back to the source, thereby conflicting with the propagation scheme of the calculations.

The shortest path method, developed in this thesis, is similar to the above described grid methods but is based on Fermat's principle and network theory. The original idea came from Nakanishi and Yamaguchi (1986) who needed in their study of nonlinear inversion of first arrival times in an island arc structure a ray tracer that could find ray paths and travel times from one source point to a number of receiver points in any velocity model without difficulties. The models they used consist of rectangular cells with a constant velocity in each cell. Nodes are placed on the boundaries of the cells and connections between them are introduced when they are on the same cell (see Chapter 1, Figure 1a). Such a structure of nodes and connections is called a graph. It becomes a network when weights are assigned to the connections. When the weights are chosen equal to the travel time of a seismic wave along it, that is, equal to the ratio of their Euclidian length and the velocity in the cell between the nodes, the shortest paths in the network can be interpreted as approximations to seismic rays, thanks to Fermat's principle. One node can be indicated as a source node and the shortest paths from this node to all other nodes in the network together can be interpreted as the ray field. As a pleasant circumstance, all of the existing methods for the computation of shortest paths in a network construct the shortest paths and travel times along them from one source to the other points of the network simultaneously. Therefore, the resulting travel times are comparable to the travel times as obtained in the grid methods described in the previous paragraph. However, Nakanishi and Yamaguchi (1986) did not investigate the efficiency of the shortest path method nor did they examine other organisations of the networks.

In the last thirty years a huge amount of literature was published on algorithms for computing the shortest paths in a network (Deo and Pang, 1984). The most famous among them is Dijkstra's (1959) algorithm. Gallo and Pallottino (1986) describe modern versions

of the Dijkstra algorithm that are efficient enough to make the shortest path method competitive with other methods for forward seismic modeling. As the network is an abstract mathematical object without explicit relation to the seismic model the shortest path method is very flexible in the sense that it applies for two as well as for three dimensional calculations without any modifications of the algorithm. Moreover, there has not been made any assumption on the spatial distribution of the nodes. This means that any other ordering of the nodes than on cell boundaries, as described above, is possible. Furthermore, the availability of shortest path algorithms whose efficiency do not depend on the weights of the connections between the nodes of the network makes the shortest path method very robust, even in complicated seismic models with large velocity contrasts. The existence of a shortest path between two nodes is never in doubt. Therefore, it is no problem to find the shortest paths to shadow zones; the network has only to be organised in such a way that the shortest path has a physical meaning.

This thesis investigates the properties of the shortest path method for seismic ray tracing. It has a twofold character: it contains parts with a detailed description of the shortest path method and parts with practical geophysical applications. The first chapter, that was published as 'Shortest path calculation of seismic rays' in Geophysics in January 1991 (Moser, 1991), serves as an introduction. The last three sections of the first chapter, 'Shortest path algorithms and their efficiency', 'Constrained shortest paths and reflection seismology' and 'The accuracy of the shortest path method' are introductions to Chapter 2, Chapter 3 and Chapter 4 respectively. The reader who is exclusively interested in geophysical applications may safely skip these chapters and continue with Chapter 5 which contains newly developed ray bending methods that can be applied to refine individual results of the shortest path calculations. Chapter 6 compares the shortest path method with other methods for forward modeling that are used in practice for exploration purposes. The last two chapters show how the shortest path method can be applied to geophysical problems like the earthquake location problem (Chapter 7) and the stochastic analysis of global travel time data (Chapter 8).

The calculations of Chapter 1 were done on a GOULD PN9000 with one CPU, 8 MB memory space and a Unix operating system. In all other chapters, the calculations were performed using one NS32532 processor of an Encore MM510 parallel computer with 2.5 Mflop (comparable to 4.2 Mflop on a SUN Sparc2). Unless otherwise stated, physical quantities like distance, time, velocity and slowness were chosen dimensionless.

CHAPTER 1

Shortest path calculation of seismic rays

Abstract

Like the travelling salesman who wants to find the shortest route from one city to another in order to minimise his time wasted on traveling, one can find seismic ray paths by calculating the shortest travel time paths through a network that represents the Earth. The network consists of points that are linked with neighbouring points by connections as 'long' as the travel time of a seismic wave along it. The shortest travel time path from one point to another is an approximation to the seismic ray between them by virtue of Fermat's principle. The shortest path method is an efficient and flexible way to calculate the ray paths and travel times of first arrivals to all points in the Earth simultaneously. There are no restrictions of classical ray theory: diffracted ray paths and paths to shadow zones are found correctly. Also, there are no restrictions to the complexity or the dimensionality of the velocity model. Furthermore, there are no problems with convergence of trial ray paths towards a specified receiver nor with ray paths with only a local minimal travel time. Later arrivals on the seismogram, caused by reflections on interfaces or by multiples, can be calculated by posing constraints to the shortest paths. The computation time for shortest paths from one point to all other points of the networks is almost linearly dependent on the number of points. The accuracy of the results is quadratically dependent on the number of points per coordinate direction and the number of connections per point.

Introduction

There are two traditional methods to compute seismic ray paths between two points in the Earth: shooting and bending (Julian and Gubbins, 1977). Shooting tries to find ray paths leaving one source point by solving the differential equations that follow from ray theory for different initial conditions until the trial ray arrives at the pre-assigned point. Bending has Fermat's principle as a starting point: it tries to find a ray path between two points by searching the minimal travel time path between them.

Both methods have serious drawbacks. By shooting a fan of rays leaving the source one can obtain an impression of the wave field. However, convergence problems are known to occur frequently, especially in three dimensions. Also, shooting will not find diffracted ray paths or ray paths in shadow zones where ray theory breaks down. With bending one can find every ray path satisfying Fermat's principle, even a diffracted one, but only for one source/receiver pair at a time and it is not certain whether the path has an absolute minimal travel time or only a local minimal travel time. These drawbacks result in the low efficiency of the methods and the incompleteness of their results. The problems are even more severe in three dimensional than in two dimensional ray tracing.

This chapter has been published as:

Moser, T.J., Shortest path calculation of seismic rays, Geophysics, Vol. 56, No. 1, 59-67, 1991.

In this chapter a method is presented that avoids the disadvantages of shooting and bending. It uses an idea of Nakanishi and Yamaguchi (1986) to approximate ray paths by the shortest paths in networks as a starting point. Next, methods are investigated to improve the efficiency of the search for the minimum time path and shows how it is possible to treat reflected arrivals in the same way.

Network theory and shortest paths in networks are an abstract formulation of problems that appear in many different branches of science and technology. They are usually of a discrete nature: there are a finite number of objects and the exact solution of the problem can be found in a finite number of steps. One example is the road map, where the network consists of cities and connections between them: one can ask for the shortest path from one city to another or the shortest closed circuit along all cities, and so on. A wide variety of network theory techniques is available. Among them, the methods based on the theory of shortest paths deserves special attention. See Deo and Pang (1984) for a review of the literature on these subjects.

One advantage of the shortest path method is that all shortest paths from one point are constructed simultaneously. This conclusion follows from the nature of shortest path algorithms: calculating one path costs just as little computation time as calculating all paths. For instance, it can be applied in the simulation of Common Shot Point gathers.

Although a search that ignores the differential equations and even Snell's law does not seem to be efficient at first sight, seismic ray tracing can make direct practical use of the increased efficiency of shortest path algorithms. Efficiency has been improved by several orders of magnitude in the last thirty years by introducing sophisticated data structures (Gallo and Pallottino, 1986). The most efficient one for ray tracing can be selected by a simple comparison of the available algorithms. The shortest path method is designed to find a good approximation to the globally minimal travel time and travel time path. Therefore, it is especially fit for applications in travel time tomography. Later arrivals on a seismogram can only be computed if they can be formulated as constrained shortest paths. For instance, the shortest paths constrained to visit one point of a set of points that form a scatterer or interface approximate diffracted and reflected ray paths.

Finally, the abstract structure of a network has the consequence that there is no notion of dimensionality of the space, so two and three dimensional ray tracing are possible with the same algorithms.

§1 Shortest paths in networks and seismic ray paths

This heuristic section precedes the more theoretical analysis and illustrates the possibilities of networks to represent approximate ray paths. One important property of a seismic ray path is given by Fermat's principle: the ray path is a spatial curve along which the travel time is stationary. The construction of a ray between a seismic source and a receiver can be based on this principle. One could enumerate all curves connecting the source and the receiver and look for the minimum travel time curve. Usually this is done by bending an initial guess of the ray path so that the travel time along it is decreased until stationarity is achieved. The analogy between a seismic ray path and a shortest path in a network provides an alternative way to use Fermat's principle.

In this approach the relevant part of the Earth is represented by a large network consisting of points connected by arcs. Each point or node is connected with a restricted number of

points in its neighbourhood but not with points that lie further away. Therefore, it is possible to travel from one node to another via the connections. The network has indeed a resemblance with a 'three dimensional road map'. Like on a road map the connections between nodes have 'lengths'. This 'length' is to be understood as a weight of the connection. For example, in the application to seismic ray tracing it is the travel time of a seismic wave between the two nodes. In seismic ray tracing the connection will have equal length for both directions along it by virtue of the reciprocity principle. In other applications of network theory the weight can be the electrical resistance in an electrical network, the cost in an economical decision tree or whatever quantity that must be minimised in a discretised medium.

When the 'length' of an arc is chosen equal to the travel time of a seismic wave, one can hope that the shortest travel time path between two nodes approximates the seismic ray path between them. This hope will not be idle when the nodes of the network are distributed in such a way that almost any ray path can be approximated by paths through the network. To this end regular distributions of the nodes and of the connections between them are introduced. This is not only required to give reasonable approximations to seismic ray paths but results also in a considerable saving of memory space. Two organisations of networks are used in this chapter to illustrate the shortest path method. Both illustrations will be given only in two dimensions although the network theory does not impose any restriction on the dimensionality of the space. All quantities are made dimensionless; the horizontal distance x and the depth z range from 0 to 100.

In the first example, taken from Nakanishi and Yamaguchi (1986), the nodes are distributed regularly on the boundaries of rectangular cells in which the propagation velocity of seismic waves is constant (Figure 1). Two nodes are only connected when there is no cell boundary between them. The travel time between two connected nodes is defined as their Euclidian distance multiplied with the slowness of the cell in between. Figures 2a and 2b

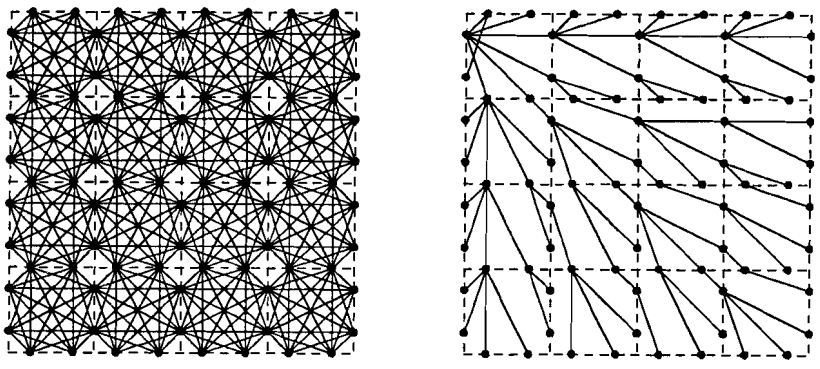

Figure 1. Cell organisation of a network. a) (left) Dashed lines: cell boundaries. Black circles: nodes. Solid lines: connections. b) (right) Shortest paths from one node to other nodes in a homogeneous model.

show the shortest paths in two networks. The model of Figure 2a has a constant velocity of 1.0 so that the shortest paths approximate straight lines. The model of Figure 2b has a (dimensionless) velocity distribution $c = 1.0 + 0.01z$ so that the shortest paths approximate circular ray paths. The cell organisation of networks is particularly suitable for the application to seismic tomography, as in the paper of Nakanishi and Yamaguchi.

The second example uses a rectangular grid of nodes in which each node is connected with the nodes in a neighbourhood of it. The seismic velocity or slowness is sampled at the node locations. The travel time between two connected nodes is defined as their Euclidian distance multiplied by the average slowness of the two nodes. This organisation facilitates the drawing of contours of equal travel times and velocities and, more important, the introduction of interfaces. Figure 3a shows a rectangular grid of 5×5 nodes, each one connected with at most 8 neighbours. In Figure 3b the shortest paths from the upper left node are plotted for a homogeneous model.

In Figure 4a a medium with velocities ranging from 1.0 to 2.0 is represented by a network of 50×50 nodes and at most 48 connections per node. The shortest paths from one node at the left side of the model are plotted. It can be seen that they converge in high velocity regions and try to avoid low velocity zones.

The travel times along shortest paths to the 50 nodes at the right side of the model can be compared with travel times calculated with the shooting method for seismic ray tracing. The ray paths in Figure 4b have been calculated by a numerical solution of the ray equation for 50 fixed initial directions with a fourth-order Runge-Kutta scheme (Stoer and Bulirsch, 1980). No attempt has been made to reach a pre-assigned receiver point; by a standard bisection method this will usually take about 5 times more computation time. The ray paths are discretised into 25 points. The travel times along the ray paths are correct up to 1% when compared to reference rays consisting of 100 points. In this setting the shortest path

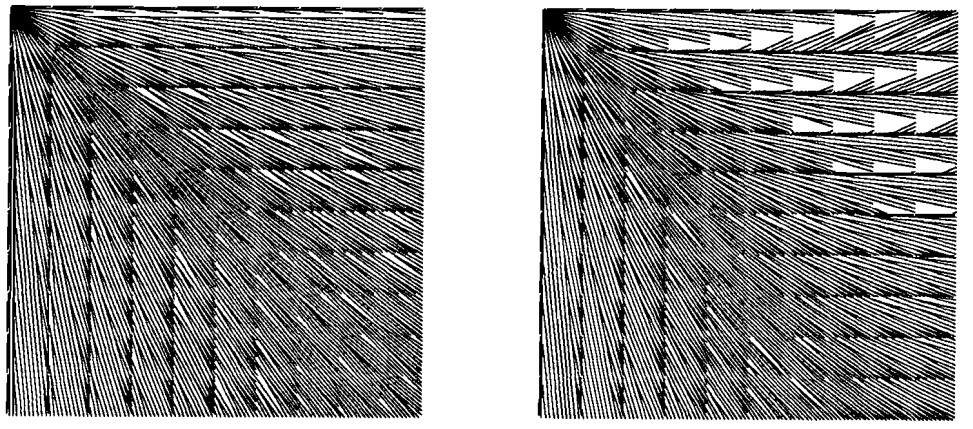

Figure 2. Shortest paths in cell networks with 10×10 *cells and* 10 *nodes per cell boundary. a) (left) Homogeneous model. b) (right) Linear velocity model* $c = 1.0 + 0.01z$.

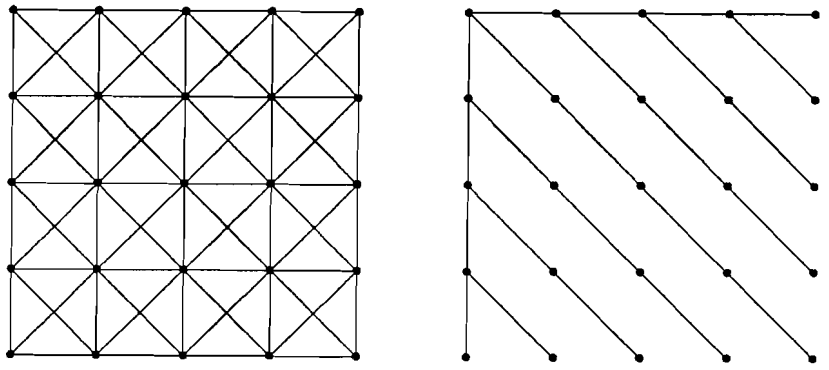

Figure 3. Grid organisation of a network. a) (left) 5 × 5 nodes, each one connected with at most 8 neighbours. b) (right) Shortest paths in a homogeneous model in the network of Figure 3a.

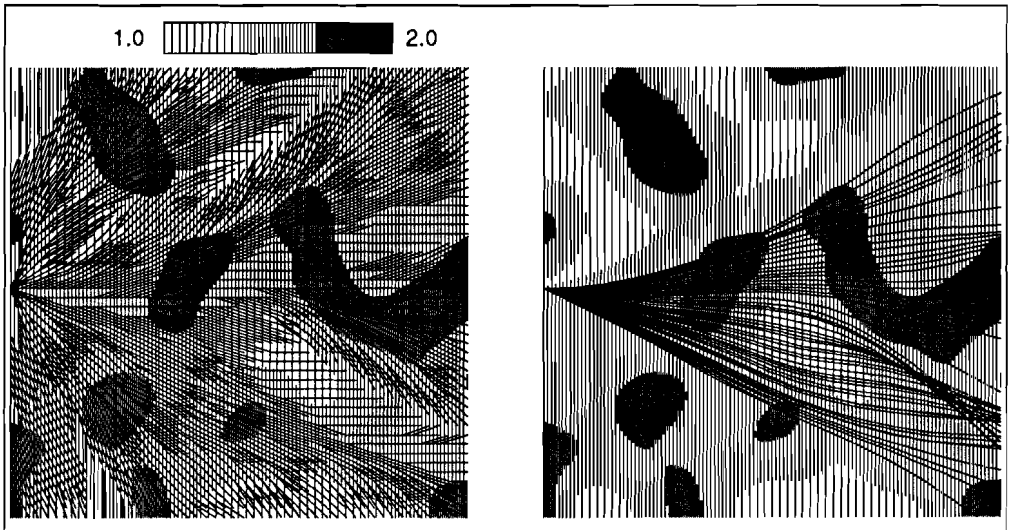

Figure 4. Shortest paths in a smooth medium. a) (left) Shortest paths in a smoothly varying medium covered by a grid of 50 × 50 nodes, each one connected with at most 48 neighbours. b) (right) Ray paths in the same medium as in Figure 4a from the same source location, calculated with a Runge-Kutta shooting algorithm.

calculations and the shooting calculations take roughly the same computation time and the travel times are correct up to 1% for both methods. Although the accuracy and the efficiency may be of the same order, the results of both calculations are different by nature. Intersections of ray paths and triplications in the travel time curves are found correctly by the shooting method but the shortest path method will give only the first arrivals. On the

other hand, the latter method has calculated the shortest paths to the 2450 other points in the same time. Therefore, a careful consideration of purpose, accuracy and efficiency of the ray tracing problem is necessary. The efficiency and accuracy of the shortest path method are given below.

The robustness of the shortest path method comes out in the presence of discontinuities in the velocity field. In Figure 5 a salt dome structure is simulated with two homogeneous layers with velocities 1.0 and 4.0, separated by a curved interface. Refractions can be seen at the top of the salt dome and diffracted paths at the flanks.

Finally, it is clear that both the cell and the grid organisation of the network may not be the best choice to represent a particular velocity model. Especially when the velocity varies rapidly, grid refinement techniques can be applied to make optimal use of the nodes and of the available memory space.

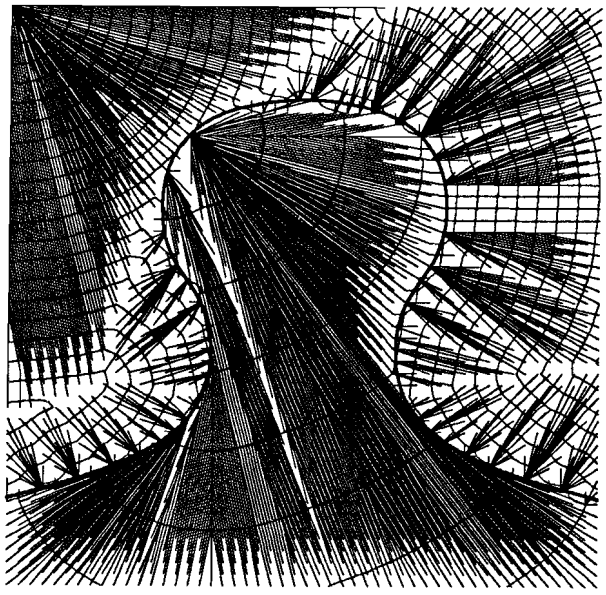

Figure 5. Shortest paths and isochrons in a piecewise smooth medium consisting of two homogeneous layers with velocities 1.0 *(upper layer) and* 4.0 *(lower layer), separated by a curved interface (heavy line) and covered by a grid of* 50×50 *nodes, each one connected with at most* 120 *neighbours.*

§2 Shortest path algorithms and their efficiency

For the description of shortest path algorithms the following notations and definitions are introduced. $G(N,A)$ is a graph, that is a set N containing n nodes together with an arc set $A \subset N \times N$. A network (G,D) is a graph with a weight function $D:N \times N \rightarrow \mathbb{R}$ that assigns a real number to each arc. D can be represented by a matrix (d_{ij}) (Figure 6). For ray tracing purposes it can be assumed that D is symmetric by virtue of reciprocity, $d_{ij} = d_{ji}$, and nonnegative, $d_{ii} = 0$ for $i \in N$ and $d_{ij} \geq 0$ for $i,j \in N$. By convention, $d_{ij} = d_{ji} = \infty$ denotes the case that the nodes i and j are not connected. The forward star of node i, $FS(i)$, is the set of nodes connected with i. All networks representing a velocity model used in this chapter are sparse: it can be assumed that each node is connected only with the nodes in a small neighbourhood of it. This means that the number of elements in the forward stars is bounded by a small number m, with $m << n$ and $m = O(1)$ $(n \rightarrow \infty)$.

A path is a sequence of nodes and connections succeeding each other. The travel time along a path from one node to another is defined as the sum of the weights of the connections of the path. A shortest path is a path with the smallest possible travel time. It may not be uniquely determined, as can be observed from Figure 1. The shortest paths from the source node s to all other nodes calculated with the available algorithms form a so called shortest path tree with its root at s and its branches connecting the other nodes. There is one and only one way to reach a node from the source node through such a tree and there are no loops. One consequence of the tree structure is that the shortest paths are described completely by an array of pointers: $prec(i)$ is the preceding node of i on the shortest path from s to i. The source node s is by definition equal to its own predecessor. A shortest path can be extracted from this array by repeating $\{j := prec(i), i := j\}$ until $i = s$. The travel time along the shortest path from s to i is denoted with $tt(i)$.

The shortest travel times from the point source s obey Bellman's (1958) equations:

$$tt(i) = \min_{j \neq i}(tt(j) + d_{ij}) \quad i,j \in N \tag{1a}$$

subject to the initial condition

$$tt(s) = 0. \tag{1b}$$

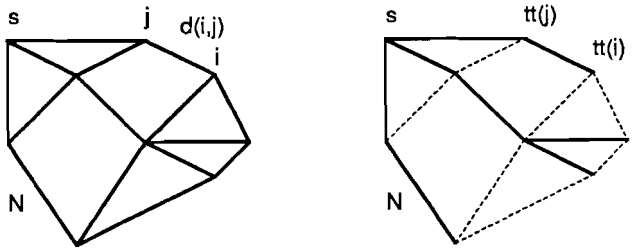

Figure 6. The definition of a network (left) and a shortest path tree (right).

The travel time to a node i is the minimum of the travel times to neighbouring nodes j plus the weight of the connection between both. These equations follow easily from the observation that if $tt(i)$ were not equal to $\min_{j \neq i}(tt(j) + d_{ij})$, there would exist a path with a shorter travel time than $tt(i)$. The node j that minimises $tt(j) + d_{ij}$ is exactly the preceding node of i on the shortest path from s to i: $j = prec(i)$. Bellman's equations suggest a scheme of constructing shortest travel times. All initial travel times are equal to infinity except for the source node, $tt(s) = 0$. It is now possible to repeat the nonlinear recursion

$$tt(i) := \min_{j \in N}(tt(j) + d_{ij}) \tag{2}$$

for all $i \in N$ until no more travel time can be updated. This will certainly happen after many iterations. The shortest paths from one node to all other nodes are then calculated. In fact, the available efficient algorithms calculate all shortest paths from one node simultaneously, although not all that information may be useful.

Dijkstra's (1959) algorithm arranges an order of nodes to be updated so that after exactly n iterations the shortest paths are found. The nodes are divided into a set P of nodes with known travel times and a set Q of nodes with travel times along shortest paths from s, that are not yet known. Initially, P is empty and $Q = N$. The minimum travel time node of Q is s. It has a known travel time ($tt(s) = 0$) so it can be transferred to P. The travel times of all nodes connected with s, all $j \in FS(s)$, are then updated in agreement with (2). An important fact is now that the node in Q with the smallest tentative travel time will no longer be updated. Therefore, it can be transferred to P and the nodes from Q connected with it are again updated. This process of finding the minimum tentative travel time node, transferring it to P and updating its forward star is repeated exactly n times. The complete shortest path tree is then constructed. Thus, Dijkstra's algorithm can be formulated as follows:

Dijkstra's algorithm

1. Initialisation

   $Q := N$ $\qquad$ $tt(i) := \infty$ for all $i \in N$

   $P := \emptyset$ $\qquad$ $tt(s) := 0$

2. Selection

   find $i \in Q$ with minimal travel time $tt(i)$

3. Updating

   $tt(j) := \min\{tt(j), tt(i) + d_{ij}\}$ for all $j \in FS(i) \cap Q$

   transfer i from Q to P

4. Iteration check

   if $P = N$ stop

   else go to 2.

Dijkstra's algorithm is the classical algorithm for the computation of shortest paths but alternatives have been developed that are several orders of magnitude more efficient. To consider the computational efficiency of Dijkstra's algorithm the number of operations can be counted. The initialisation step requires n operations to initiate the travel times. The first time the selection step requires n comparisons. Each iteration the number of required comparisons drops one, because the number of elements of Q decreases by one. The updating costs at most as many operations as there are elements in $FS(i)$, namely m.

Therefore, the total number of operations is:

$$n + (n-1) + (n-2) + \cdots + 1 + m \times n = O(n^2) \quad (n \to \infty),$$

which implies that the computation time is essentially quadratically dependent on the number of nodes.

The selection of the minimum travel time node (step 2) turns out to be the most time consuming step of the algorithm because at each of the n iterations the entire set Q of tentative travel time nodes must be scanned. The scanning could be omitted if the travel times in Q were completely ordered in a waiting list. The minimum travel time node could be found immediately as it would be the first of the waiting list. However, each updating requires the updated node to be shifted to its right position in the waiting list. This costs again $O(n)$ comparisons per iteration. Consequently, a complete ordering does not improve the quadratical dependence of the computation time on the number of nodes.

An alternative was introduced by Johnson (1977) and described by Gallo and Pallottino (1986). First, all nodes in Q with tentative travel times ∞ can be removed from the waiting list as they will never be the smallest. The rest of the nodes in Q are then partially ordered in a so-called 'heap'. A heap is an array of elements $a(i)$, $i = 1, \cdots, n$, such that

$$a(i) \leq a(2i) \tag{3a}$$

and

$$a(i) \leq a(2i+1) \tag{3b}$$

for $i = 1, \cdots, n/2$. Consider for instance the travel times

$$77, 80, \infty, 75, 89, 97, 90, 93, \infty, 70, 101, 99, 87, \infty, 74, 91.$$

After the removal of infinite travel times these can be ordered in a heap as

$$70, 75, 74, 80, 87, 77, 93, 89, 101, 99, 97, 90, 91.$$

The advantages of the heap structure come out when these numbers are represented as a tree (Figure 7). The conditions (3) are visualised in that each number is smaller than or equal to the two numbers just below it. It can be seen that the number of elements in one 'generation' or horizontal layer grows exponentially with the height of the heap. Therefore, there are only $\log n$ generations for a heap of n nodes. The minimum travel time can be found immediately: it is the uppermost number. When some finite travel time is updated, it may violate the conditions (3) so it must ascend or descend to restore the heap structure. Removing the minimum travel time node from the heap and adding an infinite travel time node whose travel time has become finite can also be formulated as descents and ascents. These cost at most $\log n$ operations. The total number of operations for the calculation of a shortest path tree is now

$$\log n + \log(n-1) + \cdots + \log 1 + m \times n = O(n \log n) \quad (n \to \infty),$$

which is much faster than the quadratical computation time of the original Dijkstra algorithm.

The computational times are plotted for four different shortest path algorithms on a series of networks in Figure 8. The calculations were done on a GOULD PN9000 with one CPU, 8 MB memory space and a Unix operating system. Only the shortest path calculation times

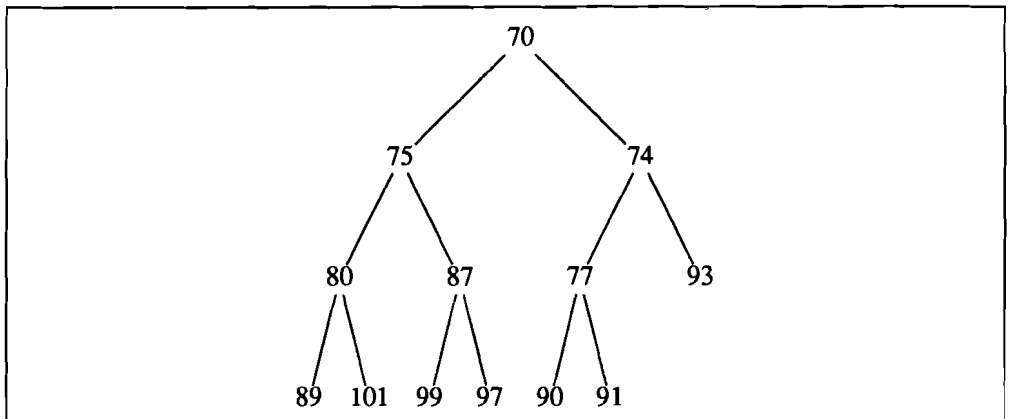

Figure 7. The tree structure of a heap.

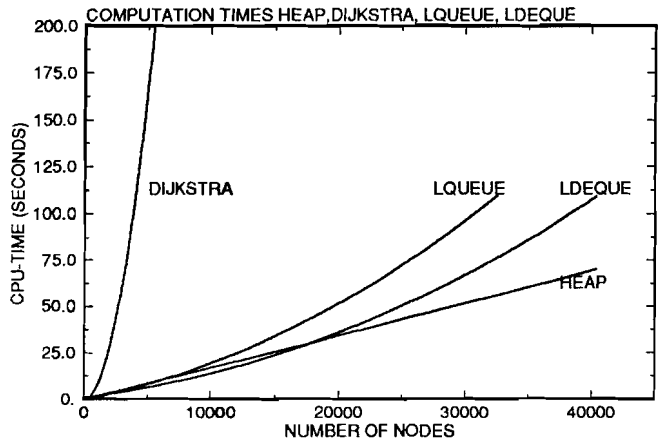

Figure 8. Computation times of various shortest path algorithms as a function of the number of nodes.

are measured, not the input and output of data. The networks are cell networks with 10 nodes per cell boundary and the number of cells in both x and z directions increasing one by one from 1 to 100. For each of the hundred networks one shortest path calculation is done for the four different algorithms: the original Dijkstra algorithm (DIJKSTRA), the Dijkstra algorithm with Q organised as a heap (HEAP) and an alternative of Dijkstra's algorithm with Q organised as a queue (LQUEUE) and as a so called double ended queue (LDEQUE). See Gallo and Pallottino (1986) for details on LQUEUE and LDEQUE. The

original Dijkstra algorithm appears to have indeed a quadratical computational complexity. In the application to seismic ray tracing LQUEUE and LDEQUE are not as fast as is promised in the literature on shortest paths. HEAP is the most efficient algorithm; it has a computation time that is almost linearly dependent on the number of nodes. This is caused by the sparseness of the networks used in ray tracing: the heap size is much smaller than the set of all nodes during the whole process.

§3 Constrained shortest paths and reflection seismology

A restriction to seismic ray tracing with the shortest path method is that only the absolutely shortest paths are found. Later arrivals on the seismogram like reflections and multiples, caused by discontinuities in the spatial velocity distribution, do not travel along the shortest path between the source and the receiver and will not be found by a simple shortest path algorithm. Yet, they are of scientific and economic importance because they contain additional information about the Earth's structure. Therefore, it is necessary to impose a constraint on the shortest paths. This constraint can be formulated as the demand to visit a specified set of nodes that lie on the interface.

The solution to the constrained shortest path problem is as follows. First, all shortest paths from the source node s to all other nodes are calculated with Dijkstra's algorithm. The

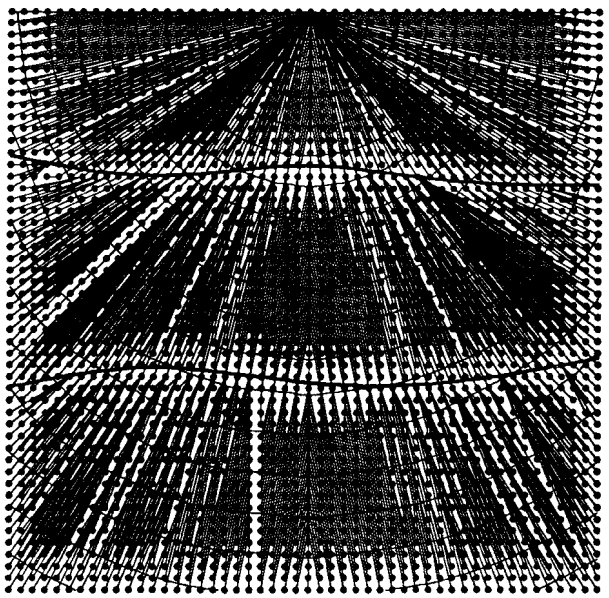

Figure 9. Constrained shortest paths. a) The primary field in a medium with constant velocities 1.0 *(upper layer),* 1.2 *(middle layer) and* 1.4 *(lower layer) and a grid network of* 50 × 50 *nodes, each one connected with* 100 *neighbours.*

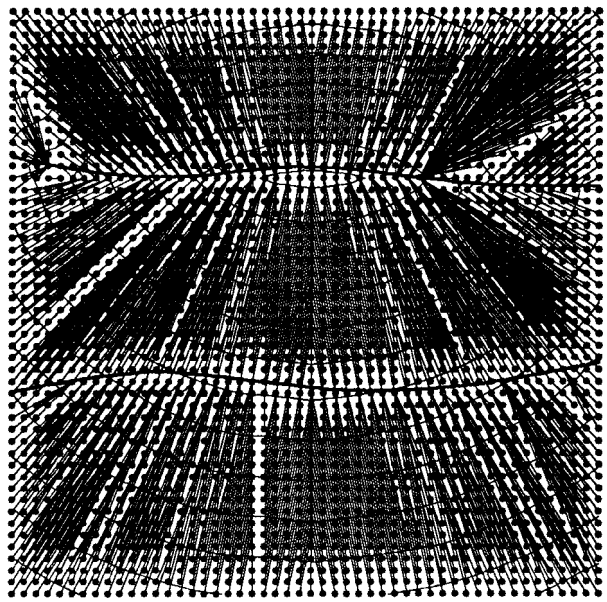

Figure 9. Constrained shortest paths. b) The secondary field, consisting of the second part of paths that are forced to visit the first interface.

travel times of the nodes on the interface are then selected, remembered and ordered in a heap. All other travel times are again set to infinity. The set Q is re-initialised to N and P is re-emptied. Dijkstra's algorithm is then restarted at step 2, the selection step. The resulting travel times are the travel times of shortest paths that are constrained to visit the interface node set.

Consider Figure 9 to see how this works. A model is generated that consists of three homogeneous layers with velocities 1.0, 1.2 and 1.4, separated by curved interfaces (heavy lines). The shortest paths from one source point at the Earth's surface are shown in Figure 9a. It is known *a priori* that points with a non-zero scattering coefficient are real physical scatterers. Nodes on interfaces or, more precisely, nodes with neighbours with a much different velocity are real scatterers so they should 'reflect' shortest paths. Therefore, they are selected and gathered in a new source node set before restarting the algorithm. The result is shown in Figure 9b. The shortest paths in the uppermost layer are reflections on the first interface. They satisfy approximately the law of incidence and reflection. The paths in the second layer are the same paths as the unconstrained shortest paths in Figure 9a. This is no surprise because these paths automatically satisfy the constraint to visit one of the interface nodes. It can be seen that Snell's refraction law holds as far as the network structure permits. The procedure of collecting scattering nodes in a new source node set and restarting the algorithm can be repeated *ad infinitum*, so one can calculate as many multiple reflections as desired. Figure 10 shows the continuation of the constrained shortest paths of Figure 9b, now under the constraint to visit the second interface. The paths are the shortest ones that visit the first interface first and next the second interface. In

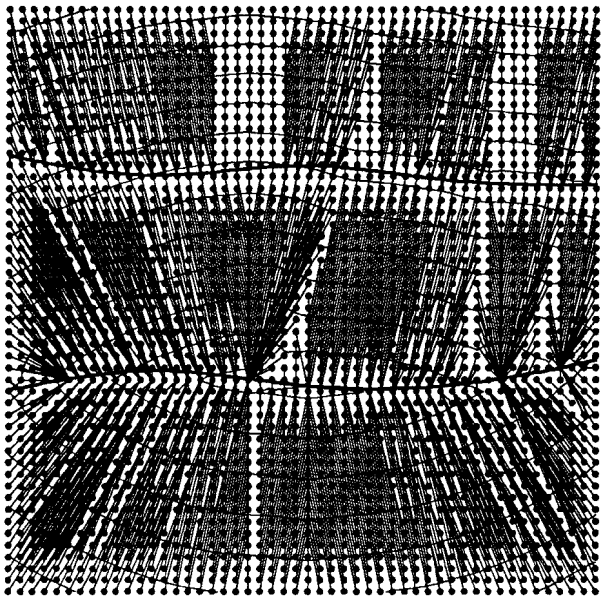

Figure 10. Reflections on the second interface in the model of Figure 9.

the third layer both constraints are again satisfied automatically. The paths in the first and second layers are reflections on the second interface.

The physical significance of the results follows from a combination of Huygens' principle and Fermat's principle. The paths are shortest in travel time between the source and the interfaces and between the interfaces and the receiver; the points on the interfaces, connecting the shortest path segments, act as secondary sources, provided that they are real scatterers.

§4 The accuracy of the shortest path method

By forcing seismic ray paths to follow the connections of a network, one introduces errors in the ray geometry and in the travel time along the ray. These errors are mainly caused by two factors, the space discretisation and the angle discretisation.

It is clear that one error source is the sampling of the velocity field by a finite number of nodes. Rapid variations in the velocity field may be missed by a coarse grid so that the shortest paths consist of line segments that are too long to give a reasonable approximation to the curved ray paths.

A simple example shows that the space discretisation is not the only error source. Consider for instance a homogeneous velocity model covered by a grid network, in which each node is only connected with its 'north', 'east', 'south' and 'west' neighbour. Such a network could imitate the streets of New York with its streets and avenues perpendicular to each other. The shortest paths in two such networks, together with the travel time contours, are

shown in Figure 11. The network of Figure 11a has 10×10 nodes, the network of Figure 11b 50×50 nodes. It is obvious that increasing the number of nodes does not improve the accuracy of the results: in both networks, the travel time contours are straight lines instead of circular wavefronts. One could hope that the shortest paths would approximate the straight ray paths in the homogeneous model, but they are far from unique so a great number of paths have the same (shortest possible) travel time.

Therefore, it can be concluded that the error in the travel time and the ray geometry depend on the number of nodes and the number of connections per node. When the average Euclidian distance between two connected nodes is denoted with δx and the average smallest angle between two connections leaving one node with $\delta\phi$, the travel time error $E = T(exact) - T(approximate)$ obeys an asymptotical relation of the type:

$$E = \alpha_{00} + \alpha_{10}\delta x + \alpha_{01}\delta\phi + \alpha_{20}\delta x^2 + \alpha_{11}\delta x\,\delta\phi + \alpha_{02}\delta\phi^2 + \sum_{i+j>2} \alpha_{ij}\delta x^i \delta\phi^j, \quad (\delta x, \delta\phi \to 0), \tag{4}$$

where α_{ij} are numbers that depend on the variation of the velocity field and the network. It can be shown that

$$\alpha_{00} = \alpha_{10} = \alpha_{01} = 0.$$

This means that the travel times are correct up to the second order in the space and angle discretisation. This result will not be proved here, but it can be illustrated in a linear velocity model. Such a model can be thought to be more or less representative for sufficiently smooth models as they can be approximated by linear velocity regions for small δx. Effects of the discretisation of nonsmooth models are visible at the interfaces in Figures 9 and 10.

The calculations of Figure 2b are done for different space and angle discretisations. The travel times in the velocity field $c = 1.0 + 0.01z$ are computed in a series of cell networks with n_x, the number of cells in the x and z direction, increasing from 2 to 50 with step 1

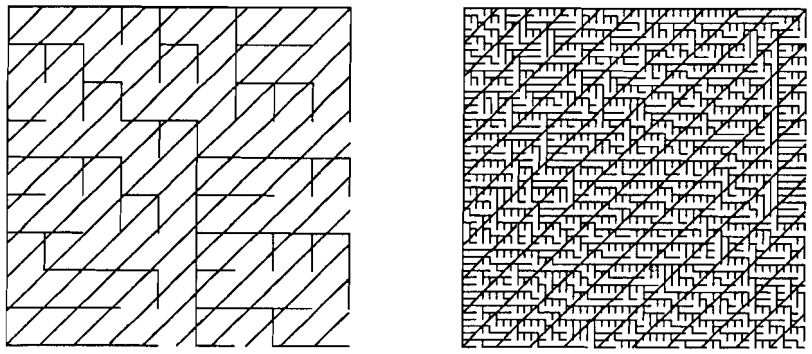

Figure 11. Shortest paths in the streets of New York. a) 10×10 nodes. b) 50×50 nodes.

and nr, the number of nodes per cell boundary, for each n_x increasing from 2 to 15. nr is a measure for $\frac{1}{\delta\phi}$ and n_x for $\frac{1}{\delta x}$. The travel times are compared to the analytical formula for travel times in constant gradient media (Červený, 1987). The absolute difference is averaged over all nodes further away from the source than 10 and divided by the computed travel time. This average relative travel time error is plotted logarithmically in Figure 12. It can be seen that the error is smaller than 0.1% for moderately large networks. This result can be compared with the computational complexity by counting the total number of nodes n as a function of n_x and nr:

$$n = 2nr(n_x + n_x^2).$$

For a cell network of 30×30 cells and 10 nodes per cell boundary the average relative travel time error is 0.0939%. The total number of nodes is 18,600 and the CPU time for this calculation is 31.7 seconds for the LDEQUE algorithm and 32.2 seconds for the HEAP algorithm (see Figure 8). The logarithmic plot of the average relative travel time error (Figure 17) shows that the error curves tend to a $\frac{1}{nr^2}$ relation for $n_x \to \infty$. This points to the fact that $\alpha_{00} = \alpha_{01} = 0$. $\alpha_{10} = 0$ can be illustrated in the same way.

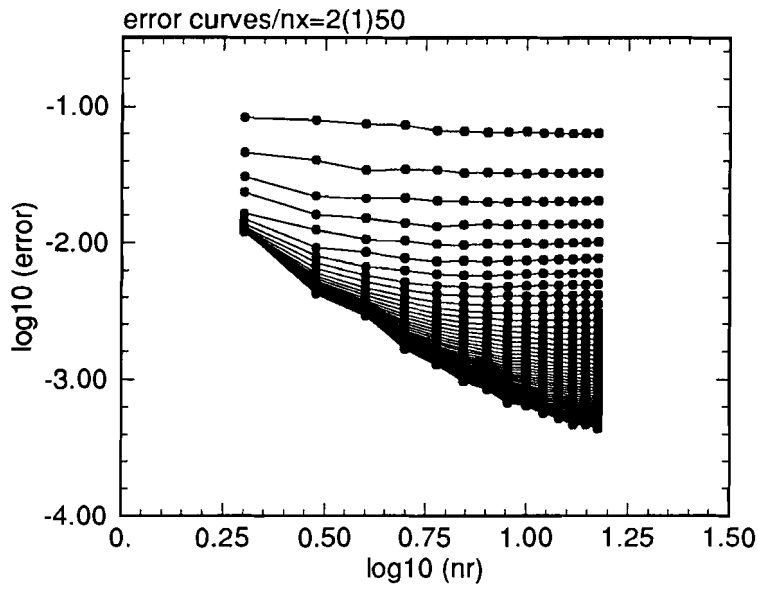

Figure 12. The travel time error as a function of the number of nodes and the number of connections per node in a linear velocity model covered by cell networks.

Conclusions and discussion

The shortest path method is a flexible method to calculate seismic ray paths and shortest travel times. It can be easily coded in FORTRAN because of the general abstract formulation of network theory. There is no need for extra software for complicated structures nor for three dimensions. The method constructs a global ray field to all points in space so there are no problems with the convergence of trial rays towards a receiver. The method finds the absolute minimal travel time path instead of getting stuck in a local minimal travel time path. Later arrivals on the seismograms can be found by an extra run of the shortest path algorithm. The shortest path method has a computation time that is almost linearly dependent on the number of nodes. Its accuracy is quadratically dependent on the number of points per coordinate direction and the number of connections per point. Even when this may not be enough for a few number of rays, the shortest paths are good initial guesses for additional bending. The multiple ray paths that can be seen in Figure 4b cannot be modeled with the shortest path method, at least not by using the suggestions of the section on 'constrained shortest paths and reflection seismology'.

CHAPTER 2

On the computational complexity of shortest path tree algorithms

Introduction

Many methods for forward modeling suffer from efficiency problems. This is due partly to the source receiver configuration partly to the complexity of the velocity model or the existence of large velocity contrasts. These problems cause unacceptable computation times for sometimes trivial results. Therefore, a new method for seismic ray tracing can be justified, among other things, by its computational efficiency. In this chapter the efficiency of ray tracing by means of the shortest path method is analysed. It is shown that algorithms exist with a computation time which is almost proportional to the number of nodes and the number of connections per node in the networks. This implies a proportionality with the volume of the model and with the desired accuracy of the ray paths and the travel times along them (see Chapter 4, 'Accuracy of ray tracing by means of the shortest path method'). A favourable property of all shortest path algorithms is that they construct the shortest paths from one source point to all other points with approximately the same computational effort as to only one receiver point. The fact that all ray paths and the total travel time field are constructed simultaneously makes the shortest path method competitive with the shooting method for seismic ray tracing which can have serious problems to make a series of trial ray paths converge toward the correct receiver. It is also shown that algorithms exist whose computation time does not depend on the weight of the connections between the nodes or, equivalently, the range of the velocity in the model. Other methods whose computation time is dependent on the range of velocities in the model, like the finite difference solution of the wave equation, have problems with large velocity contrasts near fault zones.

The reviews of Deo and Pang (1984) and Gallo *et al.* (1982) on the huge amount of literature on shortest path algorithms are used gratefully. Due to their work it is possible to restrict attention to the special case of sparse undirected networks with nonnegative weights and to concentrate on the application on seismic ray tracing. The explanation of the algorithms follows the ideas argued in the paper of Gallo and Pallottino (1986). Denardo and Fox (1979), Fox (1978) and Dial *et al.* (1979) give insight in the data structures applied in the algorithms.

In the first section notations and definitions are introduced and it is shown how efficient an algorithm must be to compete with other ray tracing methods. The algorithms available are classified in §2. The algorithms are discussed in the sections §3 and §4 using the classification of §2 as a guideline. To confirm and amplify some theoretical evidence, the results of an experiment on networks of realistic sizes are presented in §5. In the last section it is concluded that in general cases the algorithm of Dijkstra implemented with so called heaps works the fastest of all if one takes into account the memory space requirements and the flexibility with regard to irregular seismic velocity fields.

Chapter 2

§1 Definitions and preliminary considerations

§1.1 Notations

The following definitions and notations are used:

A graph $G(N,A)$ is a node set N containing n nodes and an arc set $A \subset N \times N$. A network (G,D) is a graph with a weight function $D:A \rightarrow \mathbb{R}$. This function assigns to each arc $(i,j) \in A$ a real number, the weight, which is the travel time from i to j in the special case of seismic ray tracing. D can be represented by a matrix (d_{ij}). The assumption is made that G is undirected, which means that D is symmetric ($d_{ij} = d_{ji}$). This assumption is due to the reciprocity of travel times with respect to source and receiver. Further, D is nonnegative; $d_{ii} = 0$ for each i and $d_{ij} \geq 0$ for each $i,j \in N$.

a is the number of arcs in A, corresponding with real connections. Pairs of nodes that are not connected get a travel time equal to infinity, $(i,j) \notin A \Rightarrow d_{ij} = \infty$. Infinity is just a large number, far exceeding each possible other quantity.

The forward star of node i, $FS(i) = \{j \in N \mid d_{ij} < \infty\}$, is the set of nodes j connected with i. This term originates from the case in which D is not symmetric and a backward star has to be defined as well. The maximum number of elements of a forward star is denoted with m.

The letter Q is reserved to denote the set of candidate nodes for updating for which shortest travel times will be calculated in future iterations. $q = |Q|$ is the number of such nodes; the vertical bars $|\ |$ are used to denote the number of elements of a set.

s is used to denote the source node from which the shortest path tree is calculated. $prec(i)$ is the preceding node of i, that is the first node on the shortest path from i to s. The travel time of i along that shortest path is denoted as $tt(i)$.

An asymptotic relation which is used to describe the computational complexity CC of an algorithm as a function of some graph related quantity, is denoted with Landau's symbols; $f(x) = O(g(x))$ $(x \rightarrow x_0)$ means that $\left| \frac{f(x)}{g(x)} \right|$ is finite for $x \rightarrow x_0$ and $f(x) = o(g(x))$ $(x \rightarrow x_0)$ means $\lim_{x \rightarrow x_0} \frac{f(x)}{g(x)} = 0$. When $x_0 = \infty$, as is mostly the case, $(x \rightarrow x_0)$ will be skipped.

$\lceil x \rceil$ means the smallest integer which is greater than or equal to x; $\lfloor x \rfloor$ is the greatest integer smaller than or equal to x.

§1.2 Comparison with other ray tracing algorithms

Seismic ray tracing with one source corresponds with the construction of the shortest path tree with the source node as source rather than with the construction of only the shortest path between one pair of nodes. This follows from the fact that finding the shortest path between nodes i and j implies for each existing algorithm finding all shortest paths between i and j. Thus, solving the shortest path problem for two nodes at the extreme ends of a network is the same as constructing the whole shortest path tree for one of the nodes. When one seismic ray is traced, it is a little more effort to go on and trace all rays.

In this way it can be said that ray tracing by network simulation means simultaneous calculation of all rays and the whole travel time field.

In order to compete with other ray tracing algorithms, such as shooting and bending, it must be shown that there is a solution to the shortest path problem which can be reached in a sufficiently small amount of time and number of operations, in particular when the network size grows to infinity. (Infinity in the sense of very large n). In addition it must be shown that with that amount of work a comparable result is obtained. From the preceding it is clear that ray tracing in graphs provides other and more, though less accurate, information than ray tracing by shooting or bending methods which only calculate one ray path each time. It is possible to store the travel times along that ray path but to calculate the whole travel time field (that is, in n nodes) needs n times running the shooting or bending algorithm. The results, however, are more accurate.

To avoid this discussion at this point it would be preferable to find an algorithm which runs an order of magnitude faster than the faster one of shooting and bending to n points in space. Julian and Gubbins (1977) claim that bending is ten times faster than shooting and give an asymptotic relation for the number of operations to calculate one ray, namely $O(k^3)$ if the ray is determined by k points. When n points or nodes are distributed homogeneously in a volume, one line will contain approximately $O(n^{1/3})$. As a consequence, calculating one ray by bending costs $O((n^{1/3})^3) = O(n)$ operations, so calculating all n rays costs $O(n.n) = O(n^2)$ operations. The conclusion is that a shortest path algorithm must run in $o(n^2)$ time to beat other ray tracing algorithms even in the crude complexity estimation above.

§1.3 Topology

In the literature on network theory and shortest path problems much attention is paid to the density and the topology of the network and their implications for the computational complexity. In the application on ray tracing, with D symmetric and nonnegative, two notions are of interest: sparseness and planarity. A network is called sparse if $n < a << \frac{1}{2}n(n-1)$ (n is the number of nodes and a the number of arcs). It is called dense if $n << a < \frac{1}{2}n(n-1)$ and complete if $a = \frac{1}{2}n(n-1)$; each pair of nodes is connected by an arc with a finite weight in the latter case. If $a < n$ the network is not connected and it is possible to point out a pair of nodes between which there does not exist a path at all, let alone a shortest path. A class of networks satisfying $a = g(n)$ for some function g is said to have the same density. In the application on seismic ray tracing $a \leq mn$ for each graph with m a small constant integer ($m \leq 100$). Sparseness means that the forward stars $FS(i)$ are small, which will appear to be crucial.

A network is called planar if it can be mapped on $\mathbb{R}^2$ with possible scaling of the weights, in such a way that no intersection of arcs occurs. The graphs used to represent seismic models are never planar, except from the case where the model is two dimensional and each node is only connected with its direct neighbours. The computation time of some algorithms depends on whether or not the graph is planar.

§1.4 Complexity bound estimation

In the following sections algorithms are to be compared with each other on the ground of the asymptotic relation $O(f(n))$ which predicts the order of magnitude of computation time as a function of n. A historical review like Tarjan's (1978) shows a search for the most efficient algorithms for graph theoretical problems. The question whether there is an absolute lower bound for the theoretical complexity, for instance $O(n)$, will not be treated here. It is more important to note that theoretical complexities are worst case bounds, which refer to extremely unrealistic cases. A worst case bound can be stated when a special (pathological) network can be indicated for which the shortest path algorithm runs that long computation time. Therefore, it is more suitable to give an expected case bound (Noshita, 1985). As this demands a whole machinery of statistical analysis, this will not be done here but it will be shown experimentally if an algorithm with a very bad worst case bound has a reasonable expected case bound.

§1.5 Storage requirement

Apart from computation time, one must take care of the memory space needed to execute an algorithm. It may be possible to design arbitrary efficient algorithms but with an undesirable memory space requirement. Most methods need kn numbers, where $k \leq 5$, stored in arrays, including $tt(n)$ and $prec(n)$. In literature the forward stars of each node or the whole matrix D are often stored. This is not feasible in the application on seismic ray tracing because that would need a real numbers, where $a >> n$, even if $a = O(n)$. The alternative is to make an implicit definition of the forward stars. This is possible and easy when the nodes have been well ordered. The elements of the forward star can then be found by enumerating the neighbouring nodes. Also, when the connections of the forward stars have a constant geometrical shape, standard weights can be stored so that the real weights can be calculated by only one multiplication.

§2 Classification of shortest path tree algorithms

There is no complete agreement in the literature on shortest path algorithms about how to classify them. Most authors, like Deo and Pang (1984), distinguish between labelcorrecting and labelsetting methods, but Gallo and Pallottino (1986) object that some algorithms are labelcorrecting or labelsetting according to the kind of graph. They distinguish between lowest-first search, breadth-first search and depth-first search algorithms. All species are different in the way they treat the set of nodes whose travel times are yet to be changed. Most of them are common so far as they are based on one prototypic algorithm. This algorithm gives the unique solution to the Bellman's (1958) equations:

$$tt(i) = \min_{j \neq i}(d_{ij} + tt(j)), \quad i,j \in N \quad i \neq s. \tag{1}$$

This is a set of nonlinear equations that, subject to the initial condition

$$tt(s) = 0, \tag{2}$$

uniquely determines the travel times $tt(i)$ of all nodes i along shortest paths from source node s. The prototypic algorithm to solve this is as follows (Gallo and Pallottino (1986)):

Algorithm SPT

1. Initialisation

for all $j \in N - \{s\}$
$tt(j) = \infty$
$prec(j) = j$
for $j = s$
$tt(j) = 0$
$prec(j) = s$
set $Q = \{s\}$

2. Selection and updating

for some $i \in Q$
$Q = Q - \{i\}$
for all $j \in FS(i)$ **with** $tt(i) + d_{ij} < tt(j)$
$tt(j) = tt(i) + d_{ij}$
$prec(j) = i$
$Q = Q \cup \{j\}$

3. Iteration

if $Q \neq \emptyset$ **goto 2.**
else stop.

Each time a node j is updated, it gets a new label $tt(j) = tt(i) + d_{ij}$ because $tt(j)(new) < tt(j)(old)$ so $tt(j)(new)$ is a better approximation to the solution of Bellman's equations. Q is the set of candidate nodes for updating so if Q is empty, the process is finished.

§2.1 Labelsetting versus labelcorrecting

According to the treatment of the set Q of candidate nodes for updating an algorithm is called labelsetting or labelcorrecting. When each node after its removal from Q is added to a set of permanently labeled nodes P and will stay there during the rest of the iterations, the algorithm is labelsetting. When there is no such set the algorithm is labelcorrecting; even in the last iteration each node could be relabeled.

A modification of the classification of labeling algorithms of Deo and Pang (1984) is given in Figure 1. In the following each of these methods will be discussed and each will be given an asymptotic relation for its computational complexity.

§2.2 Classification with respect to search methods

In Gallo's and Pallottino's (1986) classification a distinction is made between lowest-first search algorithms, depth-first search algorithms and breadth-first search algorithms, each differing in the way they select a node from the set Q of candidate nodes for updating. When, as in the lowest-first search case, the node i with the shortest travel time is selected and its forward star $FS(i)$ is updated, i itself does not need to be updated any more during the rest of the iterations. For that reason all lowest-first search algorithms are labelsetting, provided that the weight matrix D is nonnegative as in the application on seismic ray tracing; for instance, Johnson (1973) shows that the Dijkstra algorithm is labelsetting when D is nonnegative but labelcorrecting in the general case. Labelcorrecting algorithms which implement Q as a queue are breadth-first search; implementation as a stack is depth-first

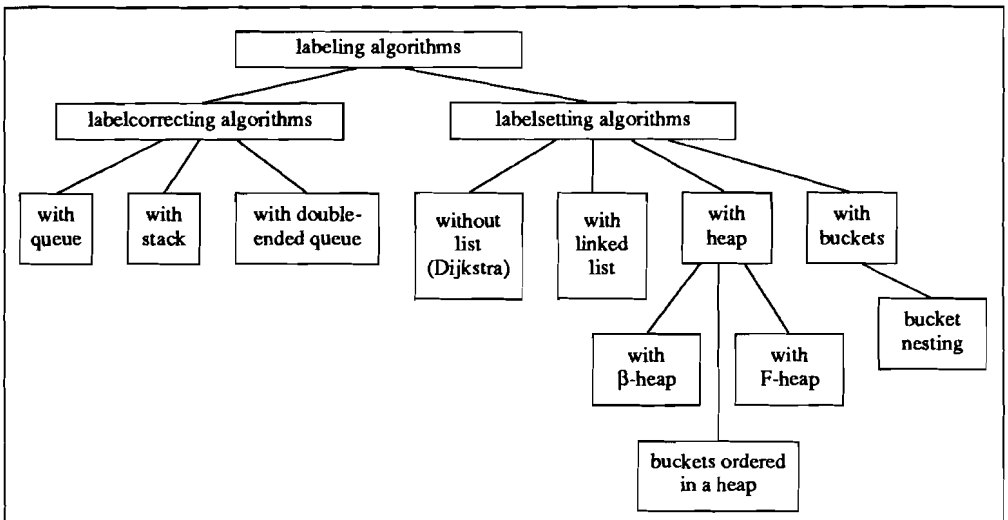

Figure 1. Classification of shortest-path tree algorithms.

search and the use of double-ended queues is a mixture of both.

§3 Labelsetting algorithms

§3.1 Dijkstra's method

In Dijkstra's method (1959), Q is implemented as a totally unordered set of nodes with temporary travel times, yet to be updated. This means that at each iteration the whole Q has to be scanned to find the minimum travel time node. However, for a nonnegative weight matrix D it is certain that the solution is reached in n iterations because each time Q is reduced with exactly one node. The algorithm is as follows:

Algorithm DIJKSTRA

1. Initialisation

for all $j \in N - \{s\}$
  $tt(j) = \infty$
  $prec(j) = j$
for $j = s$
  $tt(j) = 0$
  $prec(j) = s$
set $Q = N$

2. Selection and updating

find $i \in Q$ **such that** $tt(i) = \min_{j \in Q} tt(j)$
for all $j \in FS(i)$ **with** $j \in Q$ **and** $tt(i) + d_{ij} < tt(j)$
  $tt(j) = tt(i) + d_{ij}$

prec(j) = i

Q = Q − {i}

3. Iteration

if Q ≠ ∅ goto 2.
else stop.

Scanning Q at each iteration costs $q = |Q|$ comparisons, updating $|FS(i) \cap Q|$ comparisons and additions. In the first iteration $q = n$, next $n - 1$ and so on until $q = 0$; in each iteration $|FS(i) \cap Q| \leq m$. The result is:

$$CC = \sum_{q=0}^{n} q + nm = \frac{1}{2}n(n+1) + mn = O(n^2). \tag{3}$$

Dijkstra's algorithm requires three n-vectors one for $tt(n)$, one for $prec(n)$ and one for the implementation of Q, in all $3n$ memory places.

§3.2 Linked lists

The preceding argument makes it clear that scanning the whole set Q of temporary labeled nodes to find the minimum labeled node is the most expensive operation. To save time it is necessary to impose some structure on Q. First, exclude from Q all nodes with travel time infinity; these nodes are not to be considered when searching for the minimum travel time node so there is no need looking for them. Only when a node gets a finite label it should be added to Q. Then, as suggested by Yen (1970), Q can be ordered as a linked list in the following way. Let

$$tt(Q(1)) \leq tt(Q(2)) \leq \cdots \leq tt(Q(q)), \tag{4}$$

then define

$head = Q(1)$, $tail = Q(q)$
$s(Q(j)) = Q(j+1)$, for $j = 1, \cdots, q-1$
$p(Q(j)) = Q(j-1)$, for $j = 2, \cdots, q$
$p(Q(1)) = s(Q(q)) = -1$
$p(j) = s(j) = \xi$, when $j \notin Q$.

Two new arrays: $p(n)$, a predecessor array, and $s(n)$, a successor array, are introduced. $p(j)$ is a pointer to the preceding node of j in Q, that is the node with the longest travel time $\leq tt(j)$, and $s(j)$ is a pointer to j's following node in Q, that is the node with the

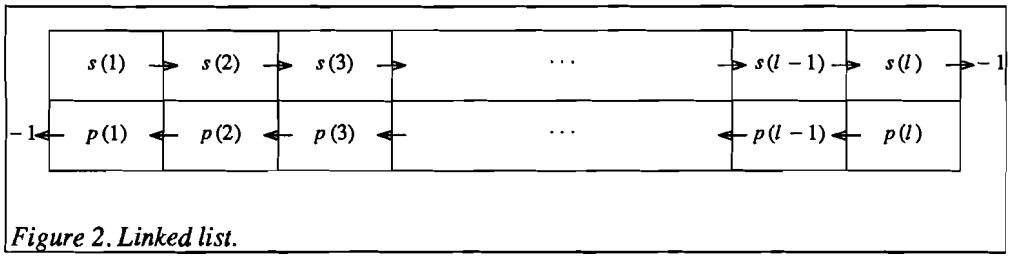

Figure 2. Linked list.

shortest travel time $\geq tt(j)$. $p(j) = -1$ and $s(j) = -1$ mean that j is the head and the tail of the list respectively. ξ is some other non-element symbol: $p(j) = s(j) = \xi \Longleftrightarrow j \notin Q$.

The element of Q with minimum travel time is obtained by simply setting: $i = head$.

Deleting this element i from Q reads: $head = s(i), p(head) = -1, s(i) = p(i) = \xi$.

Adding a node j with former travel time $tt(j) = \infty$ to Q reads $s(tail) = j$, $p(j) = tail$, $s(j) = -1$, $tail = j$ which has to be followed by moving a node j to its right place:

repeat
interchange j and p(j)
correct p and s appropriately
until tt(p(j)) ≤ tt(j)

The whole algorithm thus becomes:

Algorithm DIJKSTRA-LIST
1. Initialisation
for all j ∈ N − {s}
tt(j) = ∞
prec(j) = j
p(j) = s(j) = ξ
for j = s
tt(j) = 0
prec(j) = s
p(j) = s(j) = −1
set Q = {s}
head = tail = s
2. Selection and updating
i = head
remove i from Q
for all j ∈ FS(i) with tt(i) + d_{ij} < tt(j)
tt(j) = tt(i) + d_{ij}
prec(j) = i
if j ∉ Q add j to Q
move j to right place in Q
3. Iteration
if Q ≠ ∅ goto 2.
else stop.

(cf. algorithm DIJKSTRA). The number of iterations of DIJKSTRA-LIST is the same as DIJKSTRA: n. Finding the minimum travel time node of Q costs $O(1)$ operations now and updating $FS(i)$ still $O(m)$. Moving j to its right place costs $O(q)$ operations, but often much fewer, because it will seldom happen that j has to move through the whole Q to find its right place; the tentative travel time of j would then be infinite before the updating and the smallest of Q after the updating. Yet, this is not impossible, so the worst-case bound is $O(n)$. The result is:

$$CC = (1 + m + n)n = O(n^2). \tag{5}$$

The required memory space is four n-vectors, namely $tt(n)$, $prec(n)$, $p(n)$ and $s(n)$. It can be concluded that the use of a list causes no significant acceleration of DIJKSTRA.

§3.3 Heaps

Numerous authors make use of heaps, in order to speed up the search of the minimum travel time node in Q and to maintain the structure of Q after deleting the minimum element and after updating nodes. For technical details see Aho, Hopcroft and Ullman (1974) and Knuth (1968). A heap is defined as an array of real numbers $(a_i)_{i=1}^{h}$ with the property that $a_i \leq a_{2i}$ and $a_i \leq a_{2i+1}$. Heaps can be viewed as binary trees turned upside down with the smallest element on top. For example, the array (5,4,4,6,18,2,7,13,3) can be ordered in a heap as (2,4,3,4,5,7,6,18,13) and in a tree as in Figure 3. The top element of the heap is called the root; for each triple (a_i, a_{2i}, a_{2i+1}) a_i is the parent, a_{2i} and a_{2i+1} are sons. If an element has no sons it is called a leaf.

Some simple properties of heaps can be derived. For a given array $(a_i)_{i=1}^{h}$ $h > 2$ there is no uniquely determined heap. Each element of a heap is less or equal to the elements of the subheap on top of which it is. A heap with k 'generations' has at most $\sum_{i=1}^{k} 2^{i-1} = 2^k - 1$ elements; only the k^{th} generation may not be filled. Consequently, a heap with h elements has $\lceil {}^2\log(h+1) \rceil$ generations. The profit of the use of heaps is based on this property because when some element a_i has been changed and must be moved to its right position at most $\lceil {}^2\log(h+1) \rceil$ descents or ascents are needed.

Let H be a heap with h elements a_i and with one element, a_k, changed in value. Due to this change H may be not a heap any more, so it must be restored as follows:

Algorithm CHEAP

1. Ascend or descend?

if $a_k < a_{\lfloor k/2 \rfloor}$ goto 2.
if $a_k > a_{2k}$ or $a_k > a_{2k+1}$ goto 3.
else stop.

2. Ascent

repeat
interchange a_k and $a_{\lfloor k/2 \rfloor}$

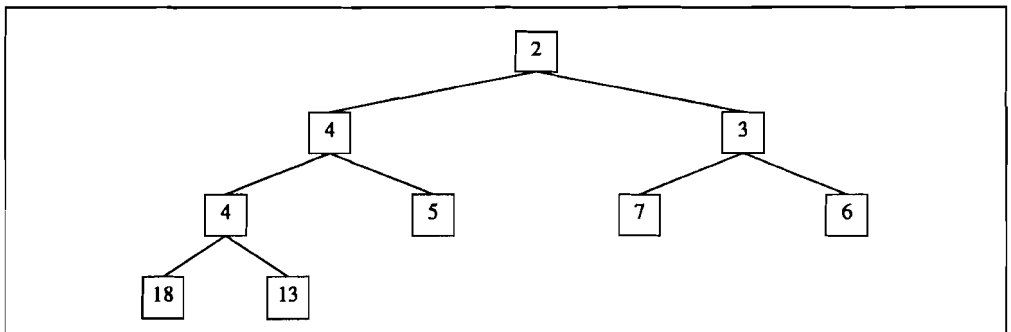

Figure 3. The tree structure of a heap.

until $a_k \geq a_{\lfloor k/2 \rfloor}$
stop.

3. Descent

repeat
interchange a_k **and min**(a_{2k}, a_{2k+1})
until $a_k \leq a_{2k}$ **and** $a_k \leq a_{2k+1}$
stop.

Each step is done provided that $k/2 \geq 1$ and $2k \leq h$ or $2k + 1 \leq h$ respectively; otherwise no comparison is needed. CHEAP costs $O(^2\log h)$ operations.

Adding a new element a_{new} to H reads: set $h = h + 1$, $k = h$, $a_k = a_{new}$ and call CHEAP ($O(^2\log h)$ operations).

Deleting an element a_k from H reads: set $a_k = a_h$, $h = h - 1$, call CHEAP (again $O(^2\log h)$ operations).

Forming a heap from an array of h elements is: choose some element a_j from the array and set $H = (a_j)$; then add successively all elements to H. This costs $\sum_{i=1}^{h} {}^2\log i \leq O(h\ {}^2\log h)$ operations.

Finding the minimum element of H reads: set $a_{\min} = a_1$; this must be followed by deleting a_1 from H ($O(^2\log h)$).

In the application of the heap data structure on shortest path problems the set Q of nodes with temporary travel times less than infinity is implemented as a heap. Two arrays are defined to connect the set of nodes N with Q. $h(i)$ $(i = 1, \cdots, n)$ is the place of node i in the heap, $h(i) = -1$ if $i \notin Q$. $q(i)$ $(i = 1, \cdots, n)$ is the node on place i in the heap, $q(i) = -1$ if $i > q$. This means $h(q(i)) = i$ if $i \leq q$ but not *vice versa*. The shortest path algorithm with the use of heaps is then:

Algorithm HEAP

1. Initialisation

for all $j \in N - \{s\}$
$tt(j) = \infty$
$prec(j) = j$
$h(j) = q(j) = -1$
for $j = s$
$tt(j) = 0$
$prec(j) = s$
$h(s) = 1$
$q(1) = s$
set $Q = \{s\}$

2. Selection and updating

$i = q(1)$
replace $q(1)$ **by** $q(q)$, **set** $k = 1$ **and call CHEAP**
for all $j \in FS(i)$ **with** $tt(i) + d_{ij} < tt(j)$
$tt(j) = tt(i) + d_{ij}$
$prec(j) = i$
if $j \notin Q$ **add** j **to** Q
$k = j$, **call CHEAP**

3. Iteration

if Q ≠ Ø goto 2.
else stop.

The computational complexity is:

$$CC = ({}^2\log q + m + {}^2\log q)\, n = O(n\ {}^2\log q) \leq O(n\ {}^2\log n) \tag{6}$$

Just like DIJKSTRA-LIST four n-vectors are needed: $tt(n)$, $prec(n)$, $h(n)$ and $q(n)$.

§3.4 β-heaps

A generalisation of the heap structure is the β-heap. A β-heap is defined as an array $(a_i)_{i=1}^{h}$ with the property that $a_i \leq a_{\beta i}, a_i \leq a_{\beta i+1}, \cdots, a_i \leq a_{\beta(i+1)-1}$ for each i, provided that each index exists. Each element has at most β sons. The operations on heaps, earlier described, change accordingly. Updates and insertions cost $O({}^{\beta}\log h)$ and deletions cost $O(\beta\ {}^{\beta}\log h)$. Thus, running the shortest path algorithm with β-heaps costs $O(\beta n\ {}^{\beta}\log n)$ operations in the worst case. The function $CC(\beta, n) = \beta n\ {}^{\beta}\log n$ achieves its minimum for $\beta = e = 2.7182818\cdots$ so the most suitable choice of β would be $\beta = 3$. The gain of efficiency by taking 3-heaps instead of 2-heaps is

$$\frac{3n\ {}^3\log n}{2n\ {}^2\log n} = \frac{3}{2}\ {}^3\log 2 \simeq 0.9463949 \tag{7}$$

which is only 5.5%.

Noshita (1985) proposes a $\lfloor {}^e\log\frac{2a}{n} \rfloor$-heap to achieve an expected complexity of

$$O\left(a + \frac{2n\ {}^2\log n\ {}^e\log\frac{2a}{n}}{{}^2\log\ {}^e\log\frac{2a}{n}}\right) \tag{8}$$

which reduces to $O(n\ {}^2\log n)$ if $a = mn$.

D.B. Johnson (1977) manipulates the worst-case bound into $O(n^{1+\varepsilon} + a)$ for networks where $a \geq O(n^{1+\varepsilon})$, $\varepsilon > 0$ constant by taking $\beta = \lceil n^{\varepsilon} \rceil$. However, this is a different case from $a = O(n)$ and besides, $O(n^{1+\varepsilon} + a)$ is worse than $O(n\ {}^2\log n)$ for each $\varepsilon > 0$.

§3.5 F-heaps

A further generalisation of the concept of β-heaps is the Fibonacci heap or F-heap. This is a heap in which no fixed number of sons is demanded for each parent. To describe the structure of an F-heap four n-vectors are introduced. One is a pointer from an element to its parent, two vectors are pointers to its left and its right sibling and the last vector is a pointer to one of its sons. With this system of pointers imposed, deleting an element from the F-heap costs $O({}^2\log h)$ operations; all other actions on heaps cost $O(1)$ operations.

Fredman and Tarjan (1984) show that the Dijkstra method with Q implemented as a F-heap runs in $O(n\ {}^2\log n + a)$ operations which is an improvement of Johnson's (1977) bound $O(a^{(\frac{a}{n}+2)}\ \log n)$ bound. However, these results hold for intermediate networks with

$n << a << n^2$. For seismic applications, as studied in this thesis, $a = mn$ and both bounds reduce to $O(n\ ^2\log n)$.

§3.6 Buckets

A remarkable property of labelsetting algorithms is that nodes with a temporary travel time close enough to the last permanent travel time will no longer be updated. In some sense nodes in Q that lie right before the wave front and close to nodes in P can be transferred to P immediately. This property can be stated more precisely by introducing the following definitions.

(G,D) is a network with a nonnegative symmetric weight matrix D. For the time being it is considered completely connected, that is $A = N \times N$ and $a = |A| = \frac{1}{2}n(n-1)$ so that all weights are finite. $d_{\min} = \min_{(i,j) \in A} d_{ij}$ is the minimum weight of (G,D). $d_{\max} = \max_{(i,j) \in A} d_{ij}$ is the maximum weight of (G,D). Q as in algorithm DIJKSTRA is the set of finite temporary labeled nodes. $F(Q) = \min_{j \in Q}(tt(j))$ is the minimum temporary travel time occurring in Q at some iteration. $B(Q) = \{j \in Q \mid F(Q) \le tt(j) \le F(Q) + d_{\min}\}$ is the subset of nodes in Q at the same iteration with a temporary travel time less than or equal to the minimal travel time in Q plus the minimum weight.

The formulation of the property is now: at some iteration the temporary travel times $tt(j)$ of all $j \in B(Q)$ are equal to the permanent travel times at the end of the algorithm.

Denardo and Fox (1979) give the short proof of this theorem and show that it can be used to develop an shortest path algorithm with computational complexity $O(k\ (1 + \lceil \frac{d_{\max}}{d_{\min}} \rceil)^{\frac{1}{k}} + a\ ^2\log(k+1))$, where k is some fixed integer. This possibility follows from the fact that all elements of $B(Q)$ at some stage of the algorithm can be given a permanent label simultaneously because they will no longer be updated.

The introduction of buckets is based on these ideas. The range of travel times $[0,d_{\max}]$ is divided in disjunct subintervals with equal width $\{tt \mid id_{\min} \le tt < (i+1)d_{\min}\}$ for $i = 0$ to $1 + \lceil \frac{d_{\max}}{d_{\min}} \rceil$). These are called buckets. A node with travel time $tt(j)$ is contained in bucket $i = \lfloor \frac{tt(j)}{d_{\min}} \rfloor$. The bucket itself has the structure of a linked list. To maintain buckets a system of three pointers must be introduced: $b(i)$ $(i = 0, \cdots, 1 + \lceil \frac{d_{\max}}{d_{\min}} \rceil)$ is an arbitrary node of bucket i, not necessarily the minimum travel time node, $b(i) = -1$ if bucket i contains no elements; $s(j)$ $(j = 1, \cdots, n)$ is the next node in the same bucket, $p(j)$ is the preceding node in the bucket, s and $p = -1$ if there are no such nodes. Initially, all buckets are empty, only the zero-th bucket contains the source node s. The shortest path tree algorithm thus becomes:

Algorithm BUCKET

1. Initialisation

for all j $\in$ **N** $-$ **{s}**

tt(j) $= \infty$

prec(j) = j
p(j) = s(j) = −1
for all k ∈ {0,1,2, ⋯ , 1 + ⌈ $\frac{d_{max}}{d_{min}}$ ⌉}
b(k) = −1
for j = s
tt(j) = 0
prec(j) = s
b(0) = s

2. Selection and updating

find lowest non-empty bucket b(k) ≠ −1
for each i ∈ b(k)
for each j ∈ FS(i) with tt(i) + d_{ij} < tt(j)
tt(j) = tt(i) + d_{ij}
prec(j) = i
change b,p,s accordingly
empty b(k)

3. Iteration

if all buckets are empty, stop.
else goto 2.

In cases where the network (G,D) is not completely connected, so that some connections have infinite weights, the above definition of d_{max} has to be modified. As the union of all buckets has to include all permanent travel times, d_{max} must be defined as an upper bound for the travel times at the end of the algorithm. A safe bound is $d_{max} = (n-1) \max_{(i,j) \in A} d_{ij}$ because a shortest path will never contain more than $n-1$ connections. Only when a node gets a temporary travel time less than d_{max} it is put in a bucket.

After this the SPT-algorithm with buckets has computational complexity

$$CC = O(a + 1 + \lceil \frac{d_{max}}{d_{min}} \rceil) = O(max(a, 1 + \lceil \frac{d_{max}}{d_{min}} \rceil)), \qquad (9)$$

because n nodes are considered but only when there are more buckets than nodes $O(1 + \lceil \frac{d_{max}}{d_{min}} \rceil)$ operations are needed. The memory space required is a $\lceil \frac{d_{max}}{d_{min}} \rceil$-vector to list the buckets and four n-vectors for $tt(n)$, $prec(n)$, $p(n)$ and $s(n)$. Buckets are especially fit for the case where all d_{ij} are integers, see Dial *et al.* (1979); only Denardo and Fox (1979) generalise to noninteger d_{ij}'s.

§3.7 Bucket nesting

Denardo and Fox (1979) show that the complexity bound can be reduced by introducing a hierarchy of buckets. A two-level system consists of an array of $1 + \lceil \frac{d_{max}}{c_1 d_{min}} \rceil$ superbuckets with width $c_1 d_{min}$, where c_1 is an integer greater than one, and a virtual array of buckets with width d_{min}. Only when a superbucket contains nonempty buckets the virtual array is made a concrete array of c_1 buckets to be scanned. Such a two-level bucket

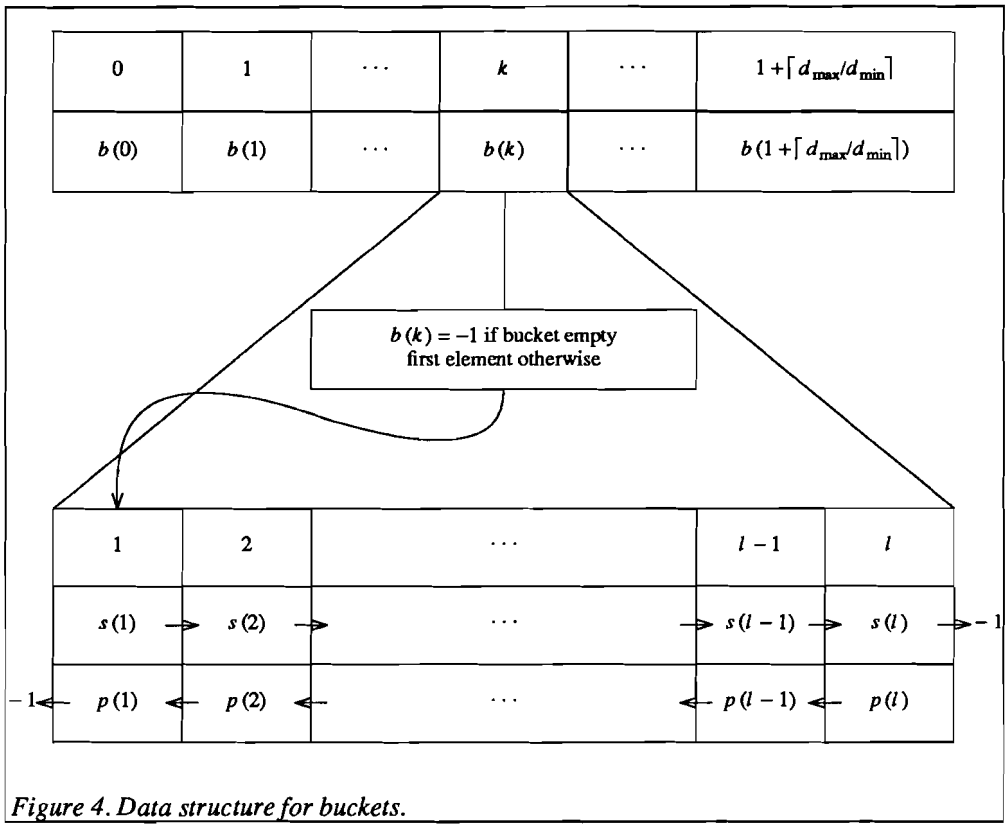

Figure 4. Data structure for buckets.

system demands $1+\lceil \frac{d_{\max}}{c_1 d_{\min}} \rceil + c_1$ buckets altogether. This is minimised for $c_1 = \lceil \sqrt{\frac{d_{\max}}{d_{\min}}} \rceil$. The complexity is then $O(2\sqrt{\frac{d_{\max}}{d_{\min}}} + a)$.

In general, a k-level bucket system consists of a superbucket system, which is to be refined only and only when it contains nonempty subbuckets, which in turn are only refined when containing nonempty subsubbuckets, and so on. The number of buckets is then

$$\lceil \frac{d_{\max}}{d_{\min} \prod_{i=1}^{k-1} c_i} \rceil + 1 + \sum_{i=1}^{k-1} c_i . \tag{10}$$

This is again minimised for $c_i = \lceil \frac{d_{\max}}{d_{\min}} \rceil^{\frac{1}{k}}$, leading to $k \lceil \frac{d_{\max}}{d_{\min}} \rceil^{\frac{1}{k}}$ buckets and a complexity of

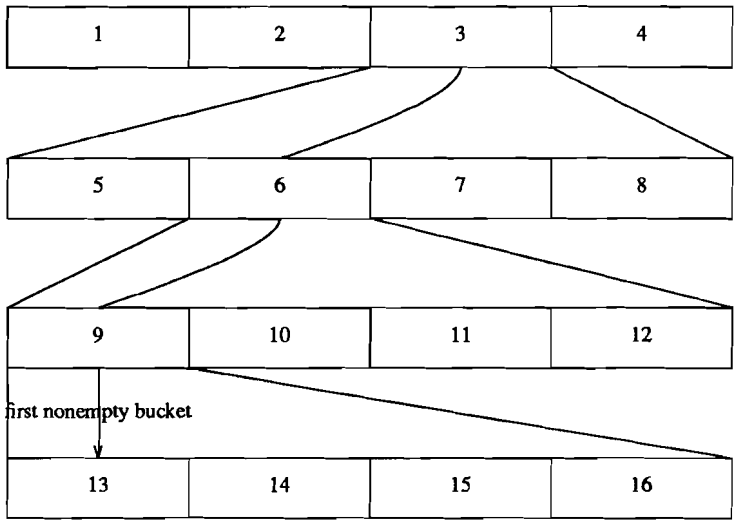

Figure 5. A 4-level bucket hierarchy with 256 virtual and 16 concrete array elements.

$$O(\max(m, k\lceil \frac{d_{\max}}{d_{\min}} \rceil^{\frac{1}{k}})). \tag{11}$$

Finally, this is minimised for $k = {}^{e}\log\lceil \frac{d_{\max}}{d_{\min}} \rceil$. It should be remarked that even with this sophistication the use of buckets may cause problems because the computation time is still dependent on $\frac{d_{\max}}{d_{\min}}$ which can be very large as in seismic models with a large range of slownesses.

§3.8 Combination of buckets and heaps

One way to combine buckets and heaps is to order the elements of each bucket in a heap. However, this is useless because the elements do not need any ordering, due to the theorem in §3.6. The other way is to keep up a heap containing nonempty buckets. This is done by Hansen (1980), who obtains a complexity of $O(a\ {}^{e}\log d_{\max})$. Karlsson and Poblete (1983) reach a complexity of $O(a\ {}^{e}\log {}^{e}\log d_{\max})$ by utilising the data structure provided by van Emde Boas, Kaas and Zijlstra (1977). They recommend their method for the case when $d_{\max} = O(n)$ and refer to the use of simple heaps in other cases.

§4 Labelcorrecting algorithms

When there is no shortest travel time element selected from the set Q of candidate nodes for updating, the shortest path tree algorithm becomes labelcorrecting because a differently selected element could be updated still later on. Depending on the maintenance of Q and the selection of a node, whose forward star is to be explored, the algorithms are depth-first search or breadth-first search. The breadth-first search strategy corresponds to organising Q as a queue. A queue is a linked list with addition of elements allowed only at the end and removal of elements allowed only at the head. Therefore, a queue is also called a FIFO-list (first in, first out). In the prototypic algorithm SPT (§2) selection of some $i \in Q$ with Q as a FIFO-list means deleting Q's head and making its successor the new head. Updating an element of Q means changing nothing in Q. Adding a new node j with former travel time infinity means making j the new tail of Q, the old tail the predecessor of j and j the old tail's successor. These are all $O(1)$ operations.

Depth-first search corresponds to the organisation of Q as a stack. A stack is a linked list where additions and deletions of elements are allowed only at its head. It is also called a LIFO-list (last in, first out). Gallo and Pallottino (1986) note that depth-first search is unnatural in shortest path tree algorithms because the first updated node will be selected last, although it is connected with the source node.

§4.1 Queues

The shortest-path tree algorithm with the queue structure is called Lqueue after Gallo and Pallottino (1986); L stands for list search. This algorithm is an efficient version of the method of Bellman (1958), Ford (1956) and Moore (1957).

Algorithm LQUEUE
1. Initialisation
for all j ∈ N − {s}
tt(j) = ∞
prec(j) = j
p(j) = s(j) = −1
for j = s
tt(j) = 0
prec(j) = s
set Q = {s}
head = tail = s
2. Selection and updating
remove i from Q head
for all j ∈ FS(i) with tt(i) + d_{ij} < tt(j)
tt(j) = tt(i) + d_{ij}
prec(j) = i
if j ∉ Q insert j at Q tail
3. Iteration
if Q ≠ ∅ goto 2.
else stop.

The computational complexity is $O(an)$; the memory space needed is four n-vectors $tt(n)$,

prec (n), $s(n)$, $p(n)$. Although the complexity is $O(an) = O(mn^2)$, which is comparable to Dijkstra's complexity, this is only a worst-case complexity. It appears that Lqueue runs almost $O(n)$ in practice. This efficiency is based on the nature of breadth-first searching: the search is organised in a way which resembles concentric waves travelling away from the source node.

§4.2 Double-ended queues

A double-ended queue, or deque, is a mixture of a stack and a queue. At the first insertion of a node into Q, Q is viewed as a queue so the insertion occurs at the tail of Q. At later updatings the node is inserted at the head so Q is viewed as a stack. The reason for this treatment is that at each time a node j with travel time $tt(j)$ is updated, except the first time when it is updated from infinity, all nodes behind it in the current tree should also be updated. This way their turn will come soon.

D'Esopo and Pape (Pape, 1974) applied the deque structure in their algorithm Ldeque:

Algorithm LDEQUE

1. Initialisation

for all j ∈ N − {s}
tt(j) = ∞
prec(j) = j
p(j) = s(j) = −1
for j = s
tt(j) = 0
prec(j) = s
set Q = {s}
head = tail = s

2. Selection and updating

remove i from Q head
for all j ∈ FS(i) with tt(i) + d_{ij} < tt(j)
if j ∉ Q then
if tt(j) = ∞ insert j at tail Q
else at head Q
tt(j) = tt(i) + d_{ij}
prec(j) = i

3. Iteration

if Q ≠ ∅ goto 2.
else stop.

The computational complexity is in its worst case $O(n2^n)$ but when the network is sparse ($a = O(n)$) and almost planar (mappable on the $\mathbb{R}^2$ without intersection of arcs) the practical complexity is $O(n)$. In fact Gallo and Pallottino (1986) state that Ldeque, apart from the risk of being in a worst case, is the most efficient shortest path algorithm for sparse graphs. To avoid risks other variants like Lqueue can be used.

§5 An empirical comparison of DIJKSTRA, HEAP, LQUEUE and LDEQUE

The computation time of the DIJKSTRA, HEAP, LQUEUE and LDEQUE-algorithms to construct the shortest path tree of one source node are measured in the following experiment with regards to the application on seismic ray tracing. The type of networks used is as in Figure 6. These are characterised by n_x cells in x-direction, n_z cells in z-direction, n_r nodes on every cell boundary and consequently

$$n = n_r(n_x + n_z + 2n_x n_z) \tag{12}$$

nodes in all. A forward star contains $6n_r$ nodes in the inner network, $3n_r$ on its boundaries. The number of finite arcs is $a = 6n_r^2 n_x n_z$. The weights are Euclidian distances. The relation between computational complexity and number of nodes is investigated for each algorithm with n_r constant and $n_x = n_z$ increasing one by one from 1 to 100. This means that each forward star in the whole experiment contains the same number of nodes and $a = O(n)$.

The results are plotted in Figures 7 and 8. Through each series of data points a parabola is drawn, regardless of their real asymptotic relation which is after all not exactly known.

Conclusion

The type of network to represent a seismic velocity field is characterised by extreme sparseness ($a \leq 100n$ mostly), arbitrarily large sizes ($n >> 10^3$), arbitrarily large ranges of weights but scarcely any planarity. According to these qualities the shortest path algorithms can be judged. Doing this one must make a distinction between the theoretical complexity and the practical complexity.

Algorithm Dijkstra without list appears to have a strictly quadratically asymptotic

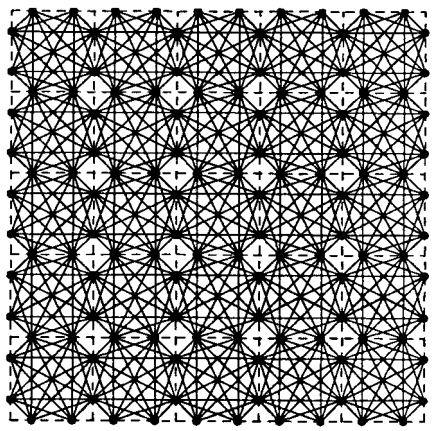

Figure 6. Type of network used for test on computational complexity.

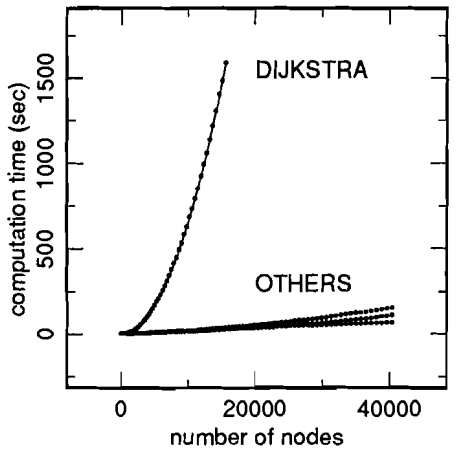

Figure 7. Computation time of four shortest path tree algorithms as a function of the number of nodes.

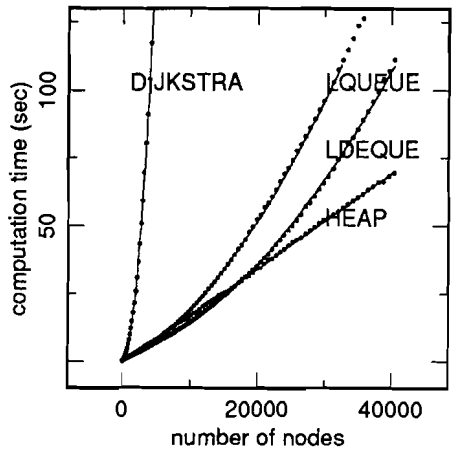

Figure 8. Blow-up of Figure 7.

complexity, just as can be foretold by the theoretical bound. Algorithm Dijkstra-list is slightly better but still not better than quadratic. The application of heaps causes a strong reduction in computation time, $O(n\ ^2\log n)$ in theory, but indistinguishable from $O(n)$ in practice. In the present case this cannot be improved by β-heaps or F-heaps. The $O(a)$ bound is reached theoretically by labelsetting algorithms with buckets.

Labelcorrecting algorithms are characterised by large theoretical bounds (Lqueue $O(an)$, Ldeque $O(n2^n)$), but with reasonable bounds in practice. This reasonability, however, depends strongly on the sparseness and planarity of the network which is disadvantageous for the application under study. Figure 8 shows that Dijkstra with heaps outperforms the labelcorrecting algorithms. Therefore, Heap or one of its improvements should be pointed out as the fastest algorithm in the present network type. It has a secure theoretical bound and often an even better practical bound.

The question whether the use of buckets or bucket nesting should be preferred to heaps leads to the consideration of the memory space needed. In spite of the reductions in storage requirement indicated in §3.6 and §3.7, the dependence on the range of weights and thus on the regularity of the seismic velocity field causes an undesired inflexibility. One could wish to be able to introduce arbitrarily strange or complex models to the ray tracing algorithm, but the use of buckets can imply either a large memory space needed or even a long computation time.

In this way the Dijkstra algorithm with the application of simple heaps can be preferred to all other algorithms because of:

1. a theoretical computational complexity of $O(n\ ^2\log n)$,
2. a storage requirement of $4n$ places,
3. its flexibility with respect to its input model.

CHAPTER 3

Reflection seismology using constrained shortest paths

Introduction

Seismic ray tracing with shortest travel time paths in networks is restricted in that only the absolutely shortest paths are found. These unconditioned shortest paths always represent the first arrivals on a seismogram like direct, refracted or diffracted waves. Later arrivals like reflections and multiples, caused by discontinuities in the spatial velocity distribution, do not travel along the shortest path between a source and a receiver and can therefore not be calculated by a shortest path algorithm. This is easily explained by observing that the shortest path is shorter than or equal to a shortest path satisfying some constraint, the constraint to visit at least one node of a specified set of nodes. Yet, later arrivals are of great scientific and economic importance because they contain additional information on the Earth's structure.

In this chapter, a method is developed to circumvent this difficulty. It is demonstrated that constrained shortest paths are good approximations of reflections and multiples. The conclusion is that each reflection needs one additional execution of the shortest path tree algorithm. The constrained shortest paths from the source node or nodes to all other nodes of the network are then calculated. This is done with the same efficiency as the calculation of shortest paths. The generalisation of the Dijkstra algorithm, needed to solve the constrained shortest path tree problem, enables the solution of a much wider class of problems: the simulation of a so called 'exploding reflector', the calculation of earthquake locations and the migration of zero and nonzero offset data.

The first sections §1, §2 and §3 are preparations to obtain a graph wherein interfaces and interval velocity distributions can be chosen freely and sets of nodes can be pointed out to 'reflect' shortest paths. The constrained shortest path tree problem is presented in §4. The solution is used in section §5 to calculate later arrivals. Section §6 gives the physical and mathematical meaning of constrained shortest paths. The last sections show some other applications of the generalised shortest path procedure.

All examples are given in two dimensions; the extension to three dimensions is straightforward.

§1 Definitions and notations

The general notations and definitions used are the same as in Chapter 2. Additional notations used in this chapter are:

$S \subset N$, the source node set, is the set of nodes for which tentative travel times are specified before the execution of the Dijkstra algorithm.

The nodes are positioned in a rectangular two dimensional grid with n_x nodes in the x-direction and n_z nodes in the z-direction and grid units δx and δz. The grid covers a region of $[0,100] \times [0,100]$ dimensionless distance units in all examples of this chapter. Each node is connected with all nodes in a $[-5,5] \times [-5,5]$-neighbourhood of it except when three nodes are aligned like for instance (0,0), (2,1) and (4,2) (Figure 1). Such a

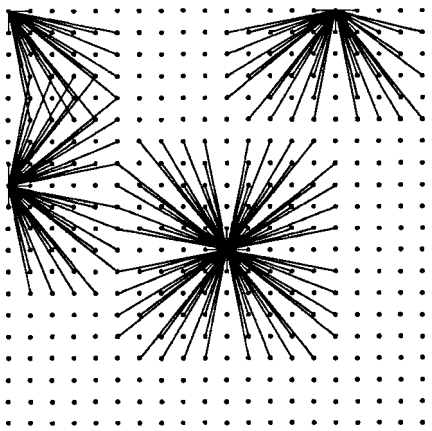

Figure 1. A 20 × 20 grid with forward stars for four different nodes.

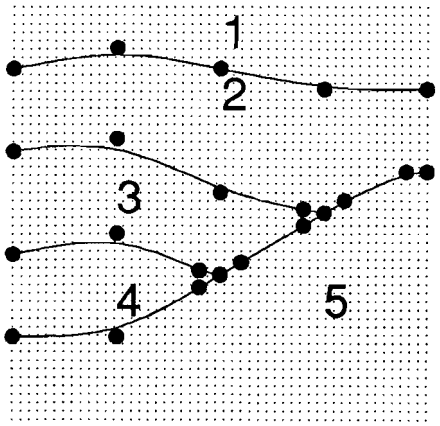

Figure 2. A model of interfaces. The black circles are the Beta-spline support points, the small points are the nodes of the network.

network organisation allows for a simple definition of a discontinuity, namely a curved line, separating the nodes into two different geological layers. The weight of a connection between two nodes i and j, d_{ij}, which is equal to the seismic travel time from i to j, is calculated as

$$d_{ij} = \int \frac{ds}{c}, \tag{1}$$

where the integral is taken along a straight line between i and j and evaluated numerically (see Chapter 6, A practical comparison of the shortest path method with other methods).

The interfaces are constructed with Beta-splines (Barsky, 1988) and are determined by:
1. support points $\vec{V}_i$, where $i = 0,nsupp$ denotes the indices of the points,
2. a Beta-spline curve consisting of *nsupp* continuously differentiable (C^1) segments $\vec{Q}_i(u)$, where $i = 0, \ldots, nsupp$ and $0 \leq u \leq 1$ for each segment,
3. two parameters β_1 and β_2 that determine the shape of the Beta-spline curve.
These notations can be extended to three dimensions. The number of interfaces is neither specified nor limited.

$nc(i), i = 1, \ldots, n$, is the 'colour' of node i, specifying the layer it belongs to.

When later arrivals are computed, $tt(1,i)$ is the travel time of the first arrival, $tt(2,i)$ of the second and so on; $prec(1,i)$ is the preceding node on the shortest path segment, $prec(2,i)$ is the preceding node on the constrained shortest path segment.

§2 The construction of interfaces

In the most general case it is desirable to construct velocity models consisting of continuous layers separated by arbitrarily curved interfaces indicating discontinuous transitions from one layer to another. This can be formulated as a set of functions: $c = c_i(x,z)$ when $(x,z) \in layer(i)$, where i runs over the number of layers; and a set of conditions: $(x,z) \in layer(i)$ if $f_i(x,z) > 0$ (Gjoystdal, Reinhardsen and Åstebol, 1985). A flexible method to construct smooth curves that have the shape of the desired interfaces is modeling with Beta-splines (Barsky,1988) (see Appendix A of Chapter 5, 'Ray bending revisited'). An interface is then defined by $nsupp+1$ support points $\vec{V}_i$. $\vec{V}_0$ and $\vec{V}_{nsupp}$ are the endpoints that are interpolated. One interface can only separate two distinct layers, so in the case of an interface ending on another interface in point $\vec{V}$ three interfaces must be formed, each ending in $\vec{V}$ (Figure 2). The tangents to the interfaces in $\vec{V}$ can be fixed by vicinal points.

§3 Separation of nodes by interfaces

Despite the flexibility of Beta-splines for the construction of interfaces, they provide an explicit parametric form for their shape. This means that points of the interfaces can be generated easily but that an arbitrary node of the network cannot be oriented with respect to the interfaces. Therefore, an array of indices is constructed that refer to the layer to which the node belongs. This is done by 'colouring' the nodes. First, the nodes adjacent to the interfaces are coloured by a so called 'painting algorithm' (Newman and Sproull, 1981). For each calculated point of the Beta-spline $\vec{Q}_i(u)$ the nodes concerned can be found by truncating $\vec{Q}_i(u)$ to the grid size:

$$\begin{matrix} \vec{v} & \vec{v}+(\delta x,0) \\ \vec{v}+(0,\delta z) & \vec{v}+(\delta x,\delta z) \end{matrix} \tag{2}$$

where

$$\vec{n} = (\lfloor \frac{Q_i(x)(u)}{\delta x} \rfloor . \delta x, \lfloor \frac{Q_i(z)(u)}{\delta z} \rfloor . \delta z) \tag{3}$$

These four nodes are the nodes coated by the 'brush' at $\vec{Q}_i(u)$. Then, for each such node it is determined at what side of the interface it lies. A criterion for this is:

$$F = \det\left[\vec{r} - \vec{Q}_i(u), \frac{d\vec{Q}_i}{du}(u)\right] \tag{4}$$

where $\vec{r}$ denotes the position of one of the brushed nodes and 'det' denotes the determinant of the 2×2 matrix with columns $\vec{r} - \vec{Q}_i(u)$ and $\frac{d\vec{Q}_i}{du}(u)$, the tangent vector at $\vec{Q}_i(u)$. When $F = 0$, $\vec{r} - \vec{Q}_i(u)$ and $\frac{d\vec{Q}_i}{du}(u)$ are linearly dependent and the node lies on the tangent. When $F > 0$, it lies at the left side of the tangent, when $F < 0$ it lies at the right side (Figure 3). The four nodes of the brush can then be coloured with the colours of the layer left to the interface or right to the interface.

After treating all interfaces in this way, the nodes adjacent to them are painted. To fill in

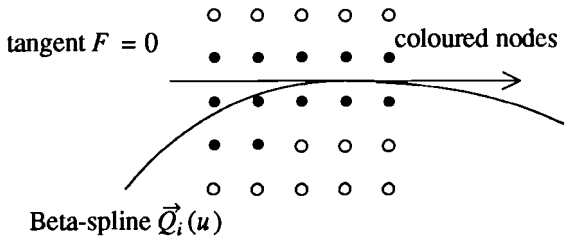

Figure 3. The painting algorithm.

the regions between them the following loop can be executed:

1. For each node i from 1 to n, look whether it is coloured. If this is not the case, consider its four neighbours j. If one is coloured, give i the same colour.
2. Repeat 1. for each node from n to 1.

The idea behind this procedure is that when all nodes neighbouring interfaces are painted, each layer is surrounded by a closed chain of nodes with its colour. The neighbours of uncoloured nodes all have the same colour, if any. An additional check must be done at the triple points of interfaces because the nodes in their neighbourhood may have been coloured more than once.

§4 The constrained shortest path tree problem

The general constrained shortest path tree problem is the problem to find the shortest paths between pairs of nodes, satisfying some constraint. The constraint poses a restriction to the paths available for searching a shortest one. The constrained shortest path is often longer than the unconstrained shortest path; equality occurs only when the shortest path happens to satisfy the constraint.

There are different types of constraints. For example, the paths can be forced to visit all nodes of a subset of the graph. An extreme case of this is the 'traveling salesman problem', the problem to find the shortest path from one node to another visiting all other nodes of the graph. Also, the paths can be forced to visit the nodes of the subset in a predescribed order. One could also fix only the number of nodes to be visited on the way, differing from each other or not. The problem to determine the shortest path not equal to the already known one is the second shortest path problem. For a survey on these problems see for instance the bibliography of Gallo *et al.* (1982).

The constrained shortest path tree problem referred to here is the problem to find the shortest paths from one node to all nodes, each path visiting at least one node of subset of the graph (Figure 4). The solution to this problem proposed here is based on a generalisation of the initialisation step of the Dijkstra algorithm. As is explained in Chapter 1 the Dijkstra algorithm divides the set of nodes N into a set of permanently labeled nodes P and a set of temporarily labeled nodes Q. P contains nodes with known travel times along shortest paths from the source node, Q contains nodes with tentative travel times. Initially, only the source node's travel time is known ($tt(s) = 0$) and all other travel times

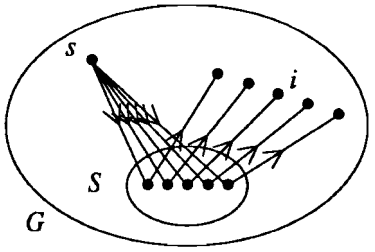

Figure 4. The constrained shortest path problem: given a graph G, a source node s and a subset $S \subset G$, find the shortest paths from s via S to all nodes i.

are unknown and are thus set equal to infinity. Then an iterative procedure is started. In each iteration, all nodes j in Q connected with nodes i in P are updated according to the rule:

$$tt(j) = \min(tt(j), tt(i) + d_{ij}). \tag{5}$$

The minimum travel time node in Q is then transferred to P and the next iteration is done. The procedure ceases when Q is empty because then the travel times to all nodes in the graph are known.

The purpose of the generalisation is to predestinate the set Q before executing the algorithm. Instead of initiating all travel times with infinity one can initiate a specified node set with finite travel times. As there is no longer a unique source node, this set will be called the source node set S. The shortest path algorithm is then:

Algorithm DIJKSTRA-GENERALISED

1. Initialisation

for all j ∈ N − S
tt(j) = ∞
prec(j) = j
for j = S
tt(j) is specified *a priori*
prec(j) = j
set Q = N

2. Selection and updating

find i ∈ Q such that tt(i) = $\min_{j \in Q}$ tt(j)
for all j ∈ FS(i) with j ∈ Q and tt(i) + d_{ij} < tt(j)
tt(j) = tt(i) + d_{ij}
prec(j) = i
Q = Q − [i]

3. Iteration

if Q ≠ Ø goto 2.
else stop.

Note that step 2. and 3. are identical to the original Dijkstra-algorithm. Note also that when

S is organised as a linked list or as a heap, the respective Dijkstra algorithms are generalised in the same way so the efficient algorithm 'Dijkstra with heaps' can be used again.

Whether or not the travel times in S will change during the rest of the procedure depends on their range and distribution. If all finite initial travel times are equal to each other, they will no longer be updated, so all nodes in S act as coupled sources, also called a centroid (Deo and Pang, 1984). On the other hand, if the initial travel times differ from each other, the node with a minimal travel time acts as a pseudo source. When, in presence of such a pseudo source, the travel times along shortest paths from the pseudo source to other nodes in S are greater than their predestinated travel times, the other nodes act as retarded sources. In the other case, when the travel times of the other nodes are greater than or equal to the travel times along the shortest paths to the pseudo source, it is just a real point source.

Full use of this idea of coupled source nodes will be made in the following sections. The application of it on the constrained shortest path problem is as follows. First, the original unconstrained shortest path tree problem with s as source is solved; the result is $tt(1,i)$ and $prec(1,i)$, $i = 1,\ldots,n$. The nodes of subset S that must be visited by the constrained paths are then selected and their travel times eventually ordered in a heap. The travel times of the other nodes are reset to infinity. The nodes of S are then used as a source node set and the algorithm is executed for a second time; the result is $tt(2,i)$ and $prec(2,i)$, $i = 1,\ldots,n$. This solves the problem of finding the travel times of all nodes of N of shortest paths from s via S. The paths themselves are obtained as a composition of the shortest paths of the first and second run of the algorithm which are stored in the arrays $prec(1,n)$ and $prec(2,n)$.

For a proof of correctness of this procedure consider the network (G',D') with an extra node s_2, connected with all nodes j of S and with no other nodes (Figure 5). The weights, $d_{s_2 j}$, $j \in S$, are equal to the travel time along the shortest paths from s_1 to j. For the other nodes and connections, $(G',D') = (G,D)$. Then find the shortest paths from s_2 to all nodes

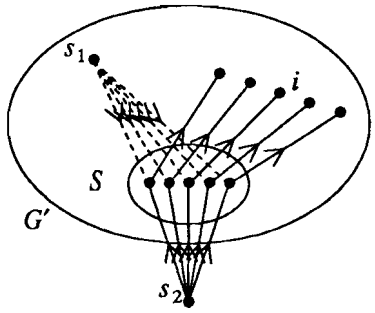

Figure 5. A proof of the constrained shortest path algorithm.

$i \in N$. This procedure is the same as the one outlined above; after the first iteration all nodes j of S are updated with $tt(j) = d_{s_2 j}$ and then S is used as a source set for the rest of the graph. The result is certainly a shortest path tree from s_2 with all paths visiting S.

Consider Figure 6 to see how the algorithm works. A simple network (G,D) is drawn consisting of 7 nodes, 2 on the 'Earth's surface' and 5 on an 'interface' $(n_1, \cdots, n_7)$. The velocity in the first layer is 1.0. The distances are Euclidian distances with $d_{n_1 n_7} = 2.0$ and $d_{n_1 n_2} = 1.0$ and the rest following from Pythagoras' rule. As a consequence of the triangle inequality, the unconstrained shortest paths from n_1 to the other nodes follow the direct connections. Thus:

$$tt(1,n_2) = 1.0, prec(1,n_2) = n_1,$$

$$tt(1,n_3) = \sqrt{1.0^2 + 0.5^2} \approx 1.12, prec(1,n_3) = n_1,$$

$$tt(1,n_4) = \sqrt{1.0^2 + 1.0^2} \approx 1.41, prec(1,n_4) = n_1,$$

$$tt(1,n_5) = \sqrt{1.0^2 + 1.5^2} \approx 1.80, prec(1,n_5) = n_1,$$

$$tt(1,n_6) = \sqrt{1.0^2 + 2.0^2} \approx 2.24, prec(1,n_6) = n_1,$$

$$tt(1,n_7) = 2.0, prec(1,n_7) = n_1. \tag{6}$$

The shortest paths from n_1 satisfying the constraint to visit the set $S = \{n_2, n_3, n_4, n_5, n_6\}$ are now calculated by setting $tt(2,n_1) = tt(2,n_7) = \infty$ and $tt(2,j) = tt(1,j)$ for $j \in S$. The generalised shortest path algorithm is run with the times $tt(2,i)$ as input. The nodes of set S will no longer be updated because $tt(2,j) < tt(2,i) + d_{ij}$ for all $j \in S$ and $i \in N$, so $tt(2,j) = tt(1,j)$ and $prec(2,j) = j$. n_1 can be updated, namely:

$$tt(2,n_1) = \min_{j \in S}(tt(2,j) + d_{n_1 j}) = tt(2,n_2) + d_{n_1 n_2} = 2 \times d_{n_1 n_2} = 2.0 \tag{7}$$

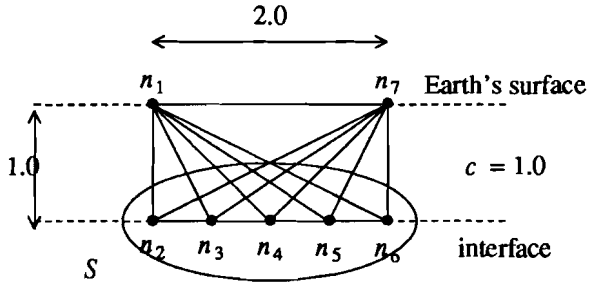

Figure 6. The calculation of constrained shortest paths in a simple network.

and $prec(2,n_1) = n_2$. The same applies for n_7:

$$tt(2,n_7) = \min_{j \in S}(tt(2,j) + d_{n,j}) =$$

$$\min(1.00 + 2.24,\ 1.12 + 1.80,\ 1.41 + 1.41,\ 1.80 + 1.12,\ 2.24 + 1.00) =$$

$$= 2 \times 1.41 = 2.82 \tag{8}$$

and therefore $prec(2,n_7) = n_4$. $tt(2,n_1)$ and $tt(2,n_7)$ cannot be decreased any more, so the process is finished. The travel times along the constrained shortest paths are known at this moment, $tt(2,i)$. The paths themselves are obtained as $prec(2,i)$, $i \in N$ and then $prec(1,j)$, $j \in S$,so (n_1, n_2, n_1), (n_2, n_2, n_1), (n_3, n_3, n_1), (n_4, n_4, n_1), (n_5, n_5, n_1), (n_6, n_6, n_1) and (n_7, n_4, n_1). The last one is the desired reflected ray path from n_1 to n_7.

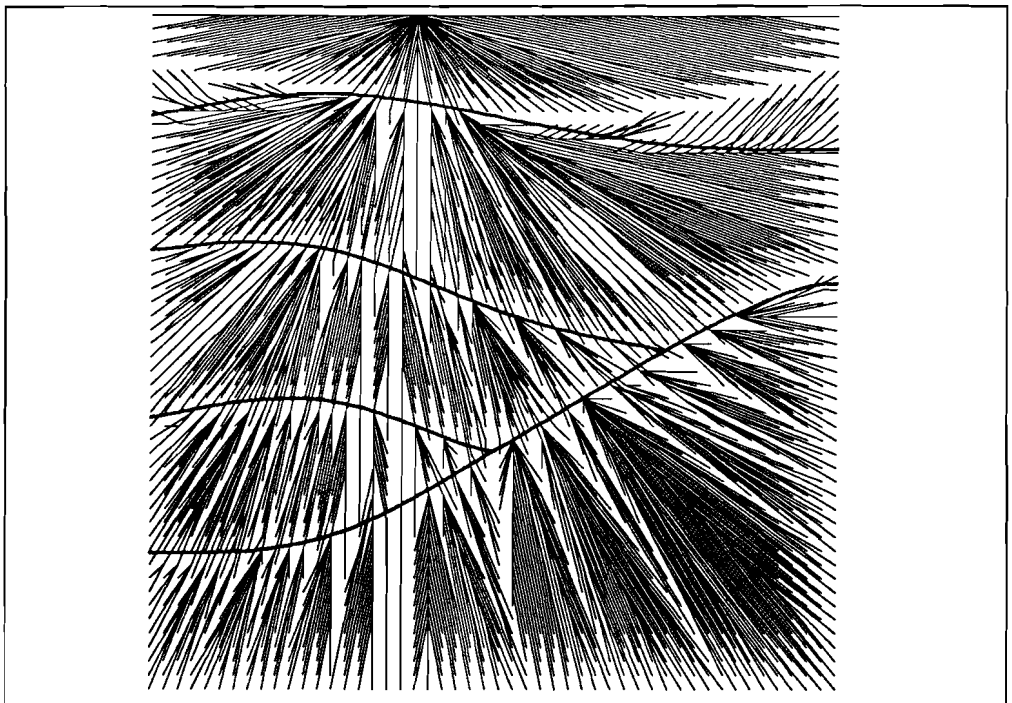

Figure 7. Ray paths of first arrivals.

§5 The calculation of later arrivals

With the preparations of the previous sections, it is now possible to calculate later arrivals on the seismogram. After the shortest paths have been found, one can indicate, manually or otherwise, a set of nodes to scatter the received rays. In particular, one can point out a set of nodes with colour $nc(i) = nr$, each having a neighbour with colour $nc(i) = nl$. When these nodes are used as a source node set, the shortest paths via the interface separating the layers numbered nl and nr are found. Additionally one can point out another set of nodes to scatter again the scattered wave field. It can thus be concluded that the calculation of each new phase or event in the seismogram requires an extra run of the Dijkstra algorithm.

Consider, for example, the model of Figure 2. The grid consists of a grid of 50×50 nodes and is divided in 5 homogeneous layers with velocities $c(1) = 1.0$, $c(2) = 1.5$, $c(3) = 2.0$, $c(4) = 2.5$ and $c(5) = 3.0$. One surface node is chosen as a source and the shortest paths are computed. This is shown in Figure 7. The nodes just below the interface between layers 2 and 3 are selected and chosen to reflect the shortest paths. Note that the interface, constructed by means of Beta-spline curves, is discretised on the grid so that it is no longer smoothly curved. The result is shown in Figure 8. Reflected ray paths are visible in the layers 1 and 2, transmitted ray paths in the layers 3, 4 and 5 and critically refracted rays at

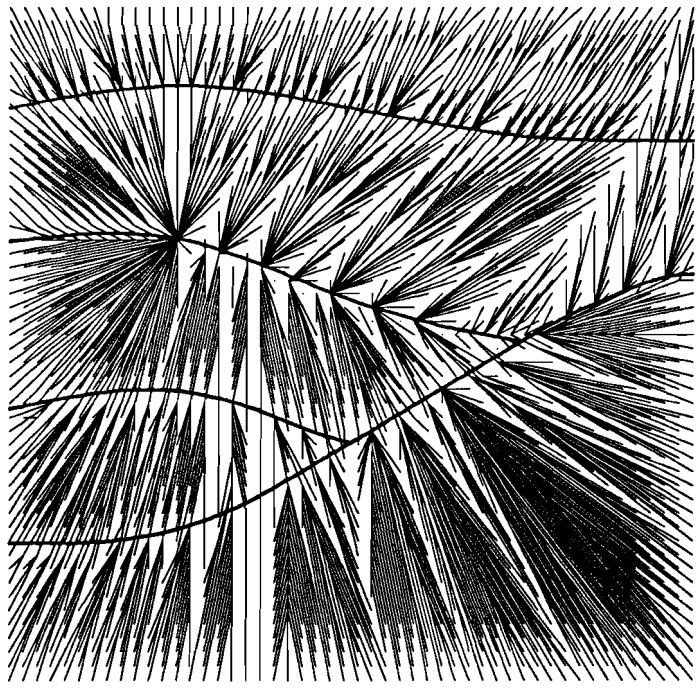

Figure 8. Reflections off the second interface (both upward and downward).

the interface between layers 2 and 5. Note that the transmitted ray paths are exactly equal to the direct rays in Figure 7. This is a manifestation of the fact that shortest paths that accidentally satisfy a constraint are equal to constrained shortest paths. A second reflection is shown in Figure 9. The reflected ray paths from Figure 8 are ordered to reflect again at the interface between layer 1 and 2. Reflections and transmissions can be seen. The latter are again exactly equal to the ray paths in Figure 8.

§6 Physical significance of constrained shortest paths

A careful contemplation is needed when looking at the constrained shortest paths and trying to classify them as a known physical phenomenon.

The physical meaning of unconstrained shortest paths can be based on Fermat's principle in the original formulation as a minimum time principle. The shortest path between two points is a path along which the seismic energy propagates in the high frequency limit. If it has a nonnegligible amplitude, it corresponds to the first observed arrival at the receiver. The first arrivals can be direct waves, refractions or diffractions. Reflected rays are not generally represented by shortest paths because usually there is a better candidate, with a shorter travel time, than the reflected ray path.

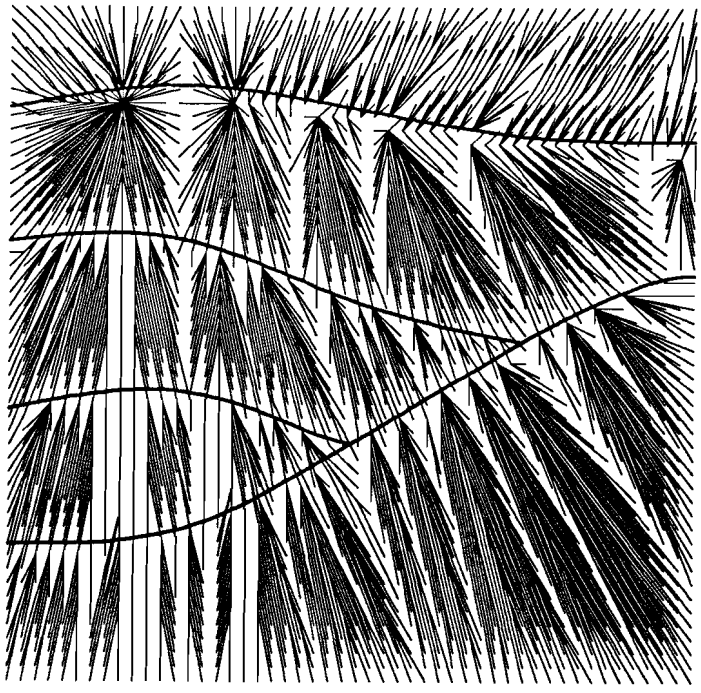

Figure 9. Additional reflections off the first interface.

Generally, constrained shortest paths are not minimum time paths but consist of segments that are themselves shortest paths between their end points. Therefore, Huygens' principle can be used to justify the physical meaning of the constrained shortest path. The first segment has the meaning of a minimum travel time path from the source to the reflection point. According to Huygens' principle, this point then acts as a secondary source from where the second segment departs. The second segment is again a minimum travel time path between the reflection point and the receiver (or the next reflection point).

It is, however, not yet clear if the selected points form a real scatterer, that is if its backscattering coefficient is nonzero and the scattered waves interfere constructively. If it is not a physical scatterer, the constrained shortest path may still obey Huygens' and Fermat's principles but there is no real seismic energy traveling along it. Therefore, one should use the foreknowledge that nodes with a velocity different from the velocity at one or more of its neighbours scatter received energy. Such nodes can be selected, for instance by taking all nodes of one layer facing another layer. The nodes then simulate an interface. This procedure was followed in section §5. More generally, one can choose single points or sets of points and specify that they should scatter energy. The shortest path method then finds along which trajectory and at what time the seismic energy travels; but it does not tell whether the energy is real or phantom.

A restriction to the constrained shortest path method is that when a shortest path already satisfies some constraint, the constrained shortest path does not differ from it. This means that supercritically reflected waves cannot be found, as they are preceded by refracted waves which satisfy the same constraint to visit an interface. Also, triplications in the travel time curve caused by strongly curved interfaces cannot be found unless the interface is split up into smaller parts, each one to be investigated in turn.

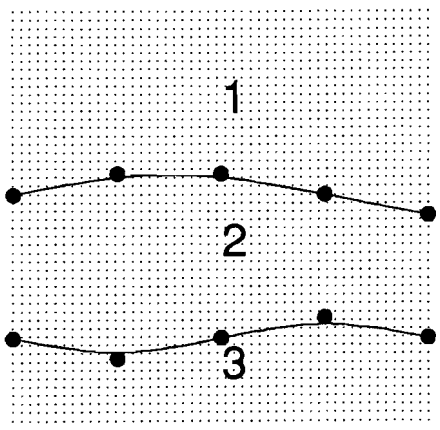

Figure 10.

§7 Exploding reflectors

In the migration of zero-offset data the exploding reflector model plays an important role. In zero-offset data the source and receiver location are supposed to coincide. The ray paths to and from the interface then coincide also. In addition, they intercept the interface at right angles to strike. Therefore, it can be assumed that zero-offset data are caused by virtual explosions at the reflectors, the travel times multiplied by two.

It is evident that this situation can be simulated by setting the travel times of all nodes at the interface to zero and then running the shortest path algorithm. As a pleasant circumstance, there is no restriction to homogeneity of the medium between the interface and the receiver: it could contain other interfaces as well. However, multiple reflections are not well described by the exploding reflector model.

Figure 10 shows a model of three layers 1, 2 and 3 with velocities 1.0, 1.5 and 2.0. In Figure 11 the 'explosion' of the first layer is shown, calculated with the generalised version of the Dijkstra algorithm.

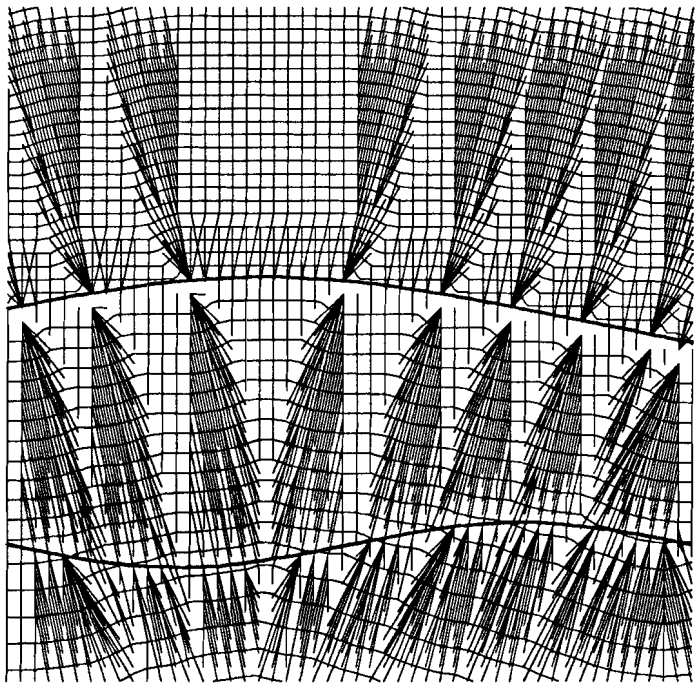

Figure 11. Exploding reflector.

§8 Migration

A common problem in exploration seismics is migration: to reconstruct the positions of the interfaces from reflection travel times and interval velocities. The idea of coupled source nodes can be helpful to do this simultaneously for all measurements. The migration of zero-offset data is the simplest case.

§8.1 Migration of zero-offset data

It is supposed that only zero-offset data at the Earth's surface and interval velocities are given and that the model of interfaces is unknown. The task is to find the real geometry of the interfaces by migrating the surface travel time data. When these are divided by 2, they are exactly the travel times obtained by the exploding reflector model. These can be backpropagated in time; the wavefront at time zero is then equal to the interface. The backpropagation can be done by multiplying all travel times with -1 and then running the shortest path procedure. Yet, the procedure only works when the model is known. This means that at first the shortest zero-offset travel times must be migrated through a medium with the uppermost interval velocity (distribution) everywhere. The other interfaces can then be reconstructed in an iterative process.

Zero offset travel times were computed for the model of Figure 10 and plotted in Figure 12. The task is now to reconstruct Figure 10 with Figure 12 and the knowledge about the interval velocities 1.0, 1.5 and 2.0. With the shortest path method this can be done in two steps.

First, the one-way travel times of the first reflection are multiplied by -1. All surface nodes are used as source nodes with these travel times as input to the shortest path algorithm that is executed in a medium with a constant velocity 1.0. From its output the wavefront at time 0 is considered to be the first interface. Secondly, the travel times of the second reflection are treated in the same way. They are backpropagated in a medium with one already

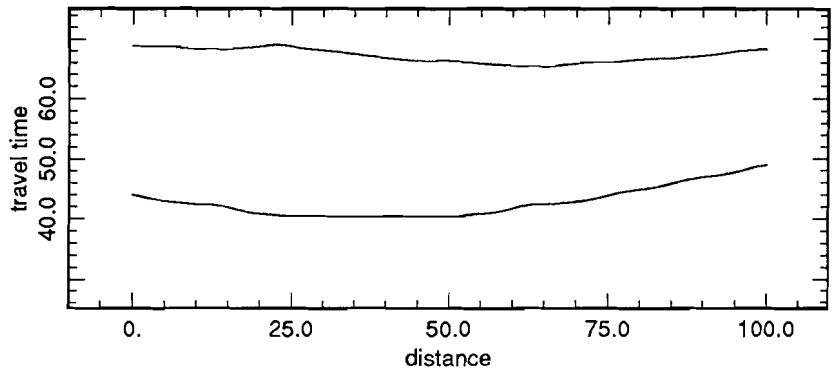

Figure 12. Zero offset travel times for the model in Figure 10.

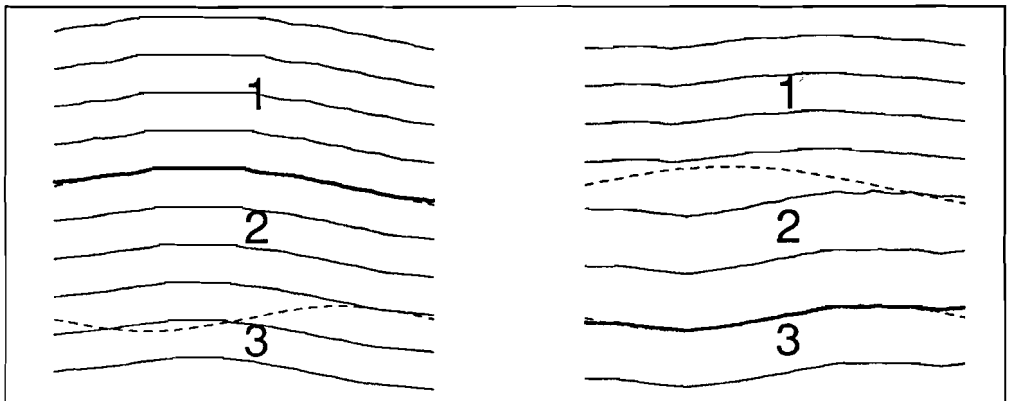

Figure 13. Backpropagated fields from Figure 12. Broken lines: model of Figure 10. Thin solid lines: isochrons. Heavy solid line: isochron at time 0.

known interface and two interval velocities 1.0 and 1.5. The two backpropagated fields are shown in Figure 13 with thin lines, the wavefronts at zero time with heavy lines. The real interfaces from Figure 10 are plotted as well (dashed lines).

§8.2 Migration of non-zero offset data

A little more complicated is the case where source and receiver occupy different positions. Consider, for instance, the common shot point geometry: one source and many receivers. The ray paths and travel times along them to and from the interfaces are no longer equal.

Suppose that the interval velocities and the reflected travel times are known. To reconstruct the uppermost interface, the shortest paths from the source node and the

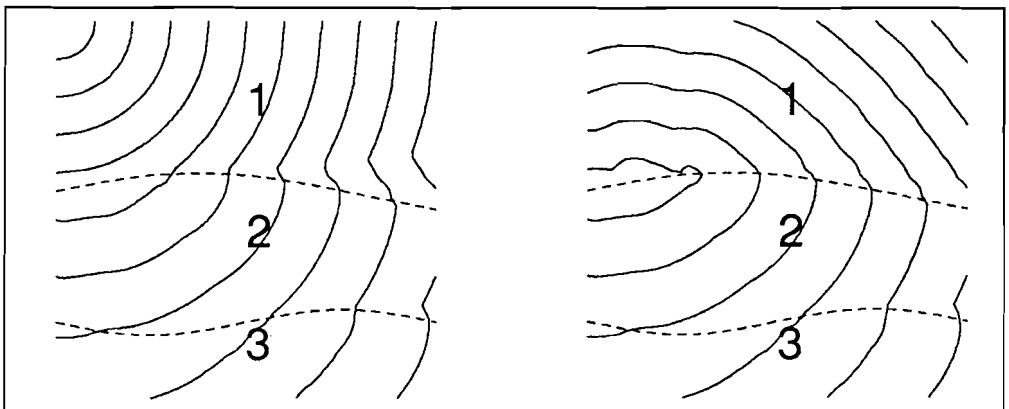

Figure 14. Primary (left) and reflected (right) fields in the model of Figure 11.

backpropagation of the reflected travel times must be found in a medium with the uppermost interval velocity everywhere. The backpropagation is again done by multiplying all travel times by −1 and running the shortest path procedure. The positions where the direct travel times and the backpropagated times sum up to zero coincide then with the first interface. The other interfaces can be again reconstructed in an iterative process. Thus, the migration of each interface demands two runs of the shortest path algorithm.

These ideas are illustrated with the example of §8.1; only the first interface is reconstructed. The primary and reflected fields are shown in Figure 14. The backpropagation of the reflected field, as measured at the Earth's surface, is shown in Figure 15 (left). The backpropagated reflected field and the primary field summed up are

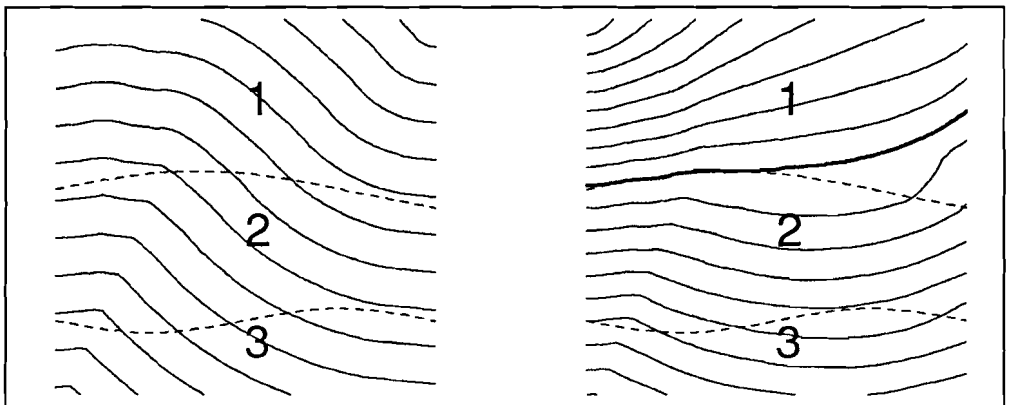

Figure 15. Backpropagated field from travel times of Figure 14 (left) and superposition of primary and backpropagated field (right). Broken lines: model of Figure 10. Thin solid lines: isochrons. Heavy solid line: isochron at time 0.

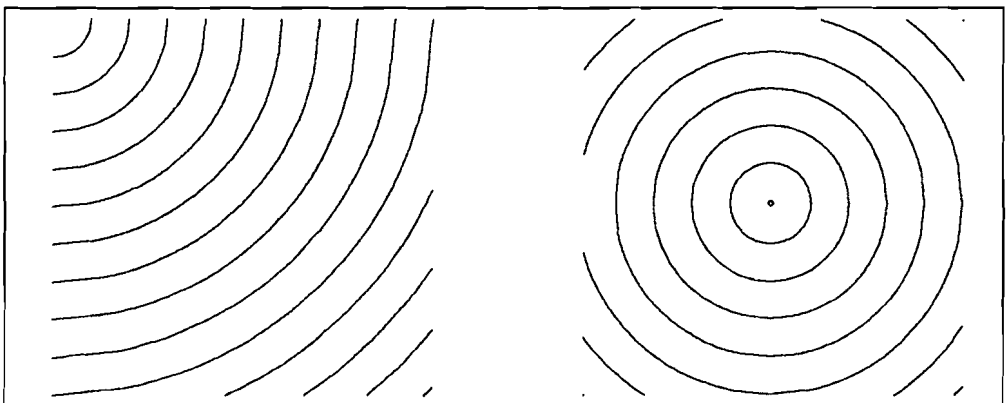

Figure 16. Primary and scattered fields (left and right).

given in the same Figure, right. The thin lines are the summed up wavefronts at nonzero times, the heavy line is the wavefront at zero time, the dashed line is the actual interface to be reconstructed. An uplifting of the calculated interface, visible at the right side, is due to incomplete data sets: some reflections do not reach the surface.

Another illustration isolates this uplifting or 'smiling'. In Figure 16 the primary field (left) is scattered by one diffraction point (right). The backpropagated field (Figure 17, left) now consists of three parts: the upper triangle contains relevant information and the two regions next to it contain circular wavefronts from both ends of the receiver line. The superposition of the primary field and the backpropagated diffracted field (Figure 17, right) has a sharp edge ending in the diffraction point. The heavy line again represents the summed up wavefronts at time zero. The effect of choosing a wrong interval velocity to propagate back is demonstrated in Figure 18.

§9 Earthquake location

The idea of coupled source nodes has another application which is not in the direct context of reflection seismology, namely source location in a known medium from first arrival times. A well known example is earthquake location. The situation is then slightly different from the previous section: the velocity model and a sufficient number of first arrival times from an earthquake to the Earth's surface are known, but the earthquake location and its origin time are unknown and to be found. The number of observations must match at least the number of unknowns (the x and y-coordinate and the origin time of the event and, sometimes, the depth coordinate z), but preferably the problem is formulated as a robust least squares problem by having much more than four observed arrival times. Chapter 7 gives a detailed description of earthquake location by means of the shortest path method, based on least squares minimisation and supplemented with a complete error analysis.

In this section only a very simple two dimensional example is given. Figure 19 (left) shows

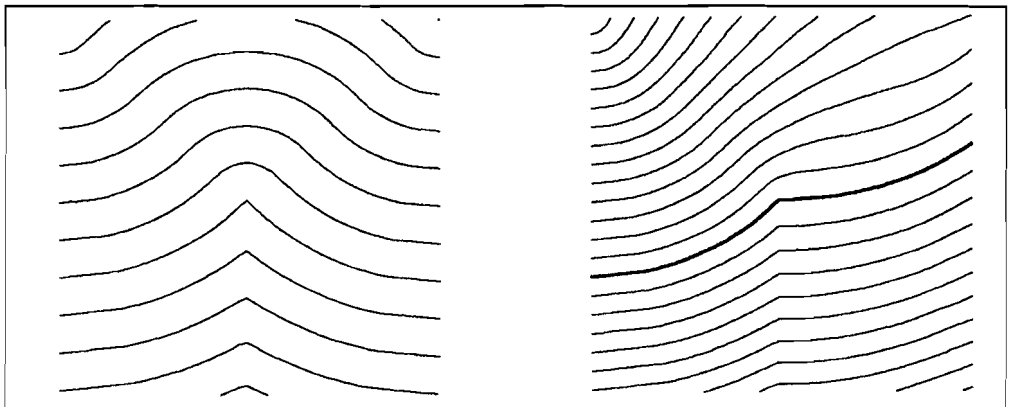

Figure 17. Backpropagated diffracted field (left) and reconstruction of the point diffractor.

the shortest paths from a subsurface source in the linear velocity field $c(x,z) = 1.0 + 0.01z$. The problem is now to reconstruct the source position and the origin time from the first arrival times at the Earth's surface and the known velocity model. Again backpropagation of the travel time measurements is the solution. Three of them are multiplied by -1 and used as source times for the proper nodes. As a result, the medium is separated by the shortest path algorithm into sections each one being reached first by a source node and each one ending in the earthquake location (Figure 19, right). Thus, the earthquake position is found by the intersection of the separations of these sections. The corresponding origin time is obtained by subtracting the travel time from the source to one of the receivers from its first arrival time.

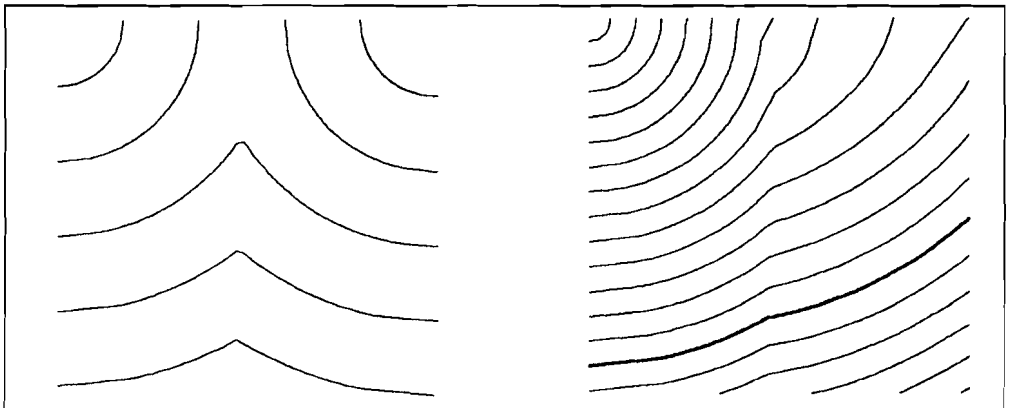

Figure 18. The effect of an incorrect interval velocity in Figure 17.

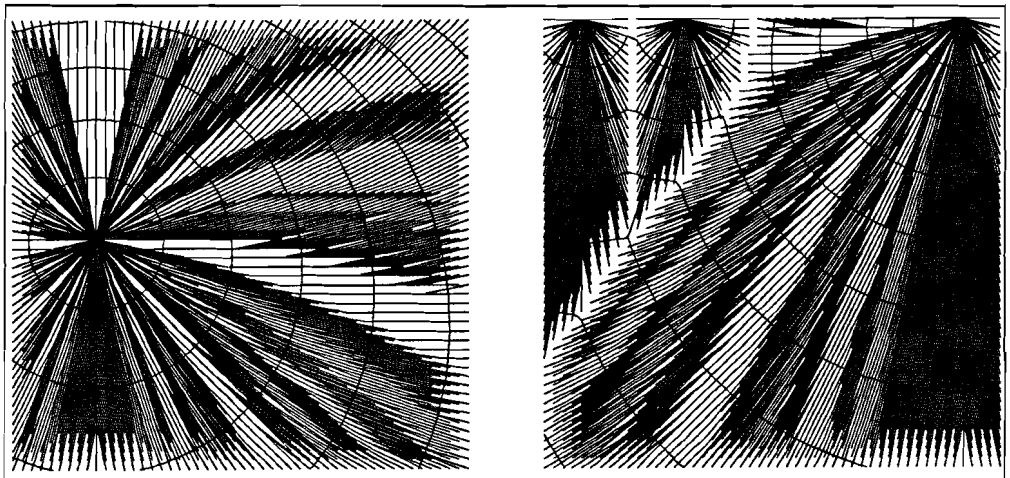

Figure 19. Earthquake location.

Conclusion

The constrained shortest path algorithm can be used to calculate later arrivals in the seismogram. These later arrivals should be interpreted as energy scattered by a specified set of nodes, thus determining their physical meaning. The algorithm is based on a generalisation of the Dijkstra algorithm which predestinates the travel times of chosen nodes. This generalisation allows fitting together single source nodes to more complicated sources. As a byproduct other experiments can be carried out, like migration or the exploding of a reflector.

The question of the accuracy of the found results has not been treated here, because the travel times of the constrained shortest paths are as precise as the times of the shortest paths they consist of. Finally, the computation of a scattered wave requires the same amount of time as computing shortest paths.

CHAPTER 4

Accuracy of ray tracing by means of the shortest path method

Introduction

The discretisation of a seismic velocity model into a network introduces errors in the ray geometry and the travel time along the rays because of the finite number of possible paths and travel times. It can be expected that these errors depend on the characteristic elements of the discretisation.

First, there is the sampling of the velocity field by a finite number of nodes. Whether the sampling is sufficient depends on the variations in the velocity field, the spatial density of nodes and the definition of the weights of the arcs. The travel time along a shortest path is defined as the sum of the weights of the connections it consists of; therefore, the travel time error is directly dependent on the sampling of the model.

The structure of the network itself also influences the accuracy of the ray paths and the travel times. Ray paths are generally smooth curves, but the shortest paths are polygonal. As they can follow only preference directions, a kind of numerical anisotropy is introduced by the network.

To be able to rely on the results of the shortest path calculation, it is necessary to estimate these errors as a function of parameters controlling the network density and structure. The estimated accuracy must then be compared with the computation time, which is dependent on the same parameters. Both the accuracy and the efficiency determine the feasibility of the shortest path calculations in relation to the desired result, which is for instance the shortest paths from one source point to the entire volume, the shortest paths to points in shadow zones or the shortest path between only one source receiver pair.

In this chapter rules are formulated that predict the travel time and ray path errors as functions of network parameters. In the first section both notations and the network types used to test the rules are introduced. §2 shows that part of the error is caused by the nonuniqueness of shortest paths. In §3 an asymptotical relationship is established between the travel time error and two network parameters. This relationship is tested in §4. §5 gives some alternative methods for the improvement of calculated travel times and ray paths, in case they are too inaccurate.

Only two dimensional examples are used, the extension to three dimensions is straightforward.

§1 Notations and definitions

Apart from the general notations and definitions introduced in Chapter 1, the following are used here:

$SP(i,j)$ is the shortest path from node i to node j. Equivalently, $SP(\vec{r}_i, \vec{r}_j)$ is the shortest path from a node i at $\vec{r}_i$ to a node j at $\vec{r}_j$. $SP(i,j)$ is the shortest path in the network (G,D), that is the concatenation of nodes; $SP(\vec{r}_i, \vec{r}_j)$ is its equivalent in the three

dimensional Euclidian space, that is a polygonal path with the edges at the node locations.

$\tilde{T}(i,j)$ or $\tilde{T}(\vec{r}_i, \vec{r}_j)$ is the travel time along $SP(i,j)$ and $SP(\vec{r}_i, \vec{r}_j)$ respectively. This travel time is obtained as the sum of the weights of the segments of the shortest path; the weights are the numerically integrated slownesses along the segments.

$F(\vec{r}_i, \vec{r}_j)$ is the minimum travel time path between $\vec{r}_i$ and $\vec{r}_j$, a spatial curve that minimises $\int_\gamma \frac{ds}{c}$ among all continuous curves γ connecting $\vec{r}_i$ and $\vec{r}_j$. Such a path is referred to as a Fermat path. Existence of a Fermat path is guaranteed by assuming that each nonempty set of candidate curves $\{\gamma | \int_\gamma \frac{ds}{c} \leq M < \infty\}$ is compact and $\int_\gamma \frac{ds}{c}$ a continuous functional of γ (according to Weierstraß, see Funk, 1962). This assumption is valid when the velocity $c(\vec{r})$ is twice continuously differentiable.

$T^*(\vec{r}_i, \vec{r}_j)$ is the travel time along the Fermat path $F(\vec{r}_i, \vec{r}_j)$. $T^*(\vec{r}_i, \vec{r}_j)$ is the smallest possible value of $\int_\gamma \frac{ds}{c}$.

The minimum travel time path or Fermat path is used as a mathematical object to which the shortest path calculations are compared. In physics the Fermat path is the curve along which the seismic energy propagates in the high frequency limit. Therefore, it is reasonable to base the error analysis on the Fermat path and the travel time along it.

$E = \tilde{T} - T^*$ is the travel time error. It depends on the spatial coordinates $\vec{r}_i$ and $\vec{r}_j$, on network- and grid-related parameters and the velocity field and its derivatives. The definitions of $\tilde{T}(\vec{r}_i, \vec{r}_j)$, $T^*(\vec{r}_i, \vec{r}_j)$, $F(\vec{r}_i, \vec{r}_j)$, $SP(\vec{r}_i, \vec{r}_j)$, $SP(i,j)$ and E are all independent of the source configuration. Point sources and coupled sources (see Chapter 3, 'Reflection seismology using constrained shortest paths') can be treated in the same way.

Asymptotic relations, used to describe the travel time error as a function of some network related quantities, are denoted with Landau's symbols; $f(x) = O(g(x))$ $(x \to a)$ means $|\frac{f(x)}{g(x)}|$ is bounded for $x \to a$ and $f(x) = o(g(x))$ $(x \to a)$ means $\lim_{x \to a} \frac{f(x)}{g(x)} = 0$. When $a = 0$, as is mostly the case, $(x \to a)$ is skipped. This is a convention different from Chapter 2.

One can use asymptotic relations to formulate Fermat's principle: let $F(\vec{r}_i, \vec{r}_j)$ be a Fermat path, $\varepsilon > 0$ a small parameter and γ a disturbance curve that vanishes in the source and receiver points $\gamma(\vec{r}_i) = \gamma(\vec{r}_j) = 0$. Both curves are parametrised by a real parameter that increases monotonically along the curve. Addition of two curves is understood as the addition of its coordinates at corresponding values of the curve parameter. The travel time along the disturbed curve $T(F + \varepsilon\gamma)$ then obeys the relation:

$$T(F + \varepsilon\gamma) = T(F) + O(\varepsilon^2) . \qquad (1)$$

This means that the travel time error is generally one order of magnitude smaller than the error in the path geometry. The altitude of a slowly shelving hill can be measured more exactly than the position of its summit. Fortunately, in exploration seismology and travel time tomography, both applications of seismic ray tracing, precise travel times are often more relevant than precise ray paths, which are a mere mathematical idealisation whereas in reality a wider area, the Fresnel zone, influences the travel time. In this chapter an accuracy of 0.1% in the travel time is chosen as a target. The ray geometry error then

follows with one larger order of magnitude.

§1.1 Network description

The networks used in this chapter to test the accuracy of shortest path and travel times along them are of the grid type and the cell type.

In the grid organisation of the network the nodes are positioned in a rectangular two dimensional spatial grid, with n_x nodes in the x direction and n_z nodes in the z direction; the total number of nodes is $n = n_x n_z$. The grid is equidistant in each coordinate direction with coordinate ranges $[x_1, x_2]$ and $[z_1, z_2]$ and grid spacings

$$\delta x = (x_2 - x_1)/(n_x - 1), \quad \delta z = (z_2 - z_1)/(n_z - 1). \tag{2}$$

Each node can be given real coordinates $\vec{r} = (x,z)$ and discrete coordinates (i_x, i_z), $i_x = 0, \ldots, n_x - 1$ and $i_z = 0, \ldots, n_z - 1$, where

$$x = x_1 + i_x \delta x, \quad z = z_1 + i_z \delta z. \tag{3}$$

Each node is connected with all nodes in a $[-nsd, nsd] \times [-nsd, nsd]$-neighbourhood of it, its forward star. An example is given in Figure 1. *nsd* is then the number that determines the size of the forward star; the number of connections with one node (number of standard distances, distances that are computed and stored in memory before executing the shortest path tree algorithm) is bounded above by $4(nsd + 1)nsd$. By convention, $nsd = 0$ is not to describe the case in which there are no connections at all but the case in which a node is connected with four adjacent nodes, its 'north, south, east and west' neighbours. Such a

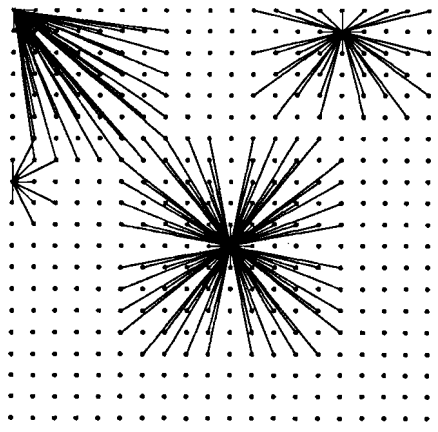

Figure 1. A 20 × 20 grid with the forward stars of four nodes, nsd = 2 (left), 4 (above right), 5 (center) and 7 (above left).

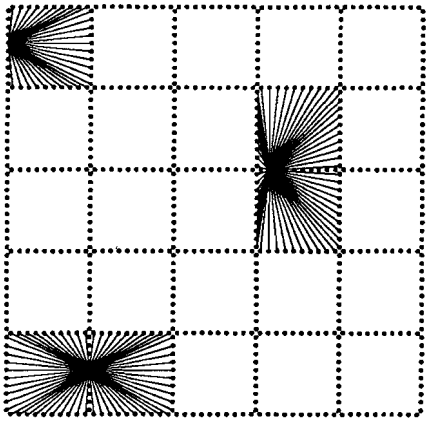

Figure 2. A 5 × 5 cell model with 10 nodes per cell boundary and the forward stars of three nodes.

network will be called the Streets of New York (Figure 3, left). n_x and n_z is used to control the spatial distances allowed, *nsd* is used to control the number of angles allowed. The weight d_{ij} of a connection between nodes i and j is computed as

$$d_{ij} = \int_i^j \frac{\mathrm{d}s}{c} \tag{4}$$

where the integral is taken along a straight line between i and j and evaluated numerically. The choice for the integration rule has a direct consequence for the accuracy of the computed travel time. When d_{ij} is computed with the trapezoidal rule:

$$d_{ij} = \frac{1}{2}\left(\frac{1}{c_i + c_j}\right) || \vec{r}_i - \vec{r}_j || \,, \tag{5}$$

where $|| \, . \, ||$ is the Euclidian norm, the travel time is correct at most up to $O(\delta x^2)$. All grid networks in this chapter use the above definition for the weight of an arc. Other, more advanced, schemes take the variations of the velocity on the nodes between i and j into account (see Chapter 6).

The cell organisation of the network is taken from Nakanishi and Yamaguchi (1986). It is suitable for seismic tomography. The two dimensional medium is divided into rectangular cells with a constant velocity; the nodes are placed on their boundaries (see Figure 2). Two nodes are only connected with each other when there is no cell boundary between them. The cell networks are characterised by n_x cells in the x-direction, n_z cells in the z-direction and n_r nodes per cell boundary. The total number of nodes is $n = n_r(n_x + n_z + 2n_x n_z)$. The parameters n_x and n_z are used to control the spatial distances allowed, n_r is used to control the number of angles allowed. The weight d_{ij} of a connection between two nodes i and j is defined as the Euclidian length of the connection divided by the velocity of the cell between i and j.

§2 Nonuniqueness of shortest paths

The grid organisation of a network allows for a simple examination of the errors that are introduced by discretising a seismic model into a network.

If the shortest path from one node to another in a network representing a homogeneous model would be unique, it would be possible to derive a simple error estimation in the ray geometry. In a network representing a homogeneous model with velocity equal to unity, the weights of the connections are equal to their Euclidian lengths. This means that the triangle inequality

$$|| \vec{r}_1 - \vec{r}_3 || \leq || \vec{r}_2 - \vec{r}_1 || + || \vec{r}_3 - \vec{r}_2 || \tag{6}$$

can be used to construct shortest paths. Now, if the shortest path from i to j, $SP(i,j)$, is uniquely determined, every other path between i and j has a longer travel time. This means that the Fermat path, which is a straight line in a homogeneous model, lies in a close neighbourhood of the shortest path, for instance not further away than the closest non-shortest path. The node spacing is then a measure for the accuracy of the ray geometry. For special types of networks, such as the grid networks or the cell networks, this can be stated precisely with the help of the triangle inequality. In the following, however, it is shown how the nonuniqueness of shortest paths influences the accuracy of the path

geometry and the travel time.

In a homogeneous model, covered by a grid network as described in the previous section, the number of shortest paths from one source node to another node can be derived with the help of recurrence relations. The nodes of the grid are given discrete coordinates $i = \lfloor \frac{x - x_1}{\delta x} \rfloor$ and $j = \lfloor \frac{z - z_1}{\delta z} \rfloor$. The source is positioned at $(i,j) = (0,0)$ and the opposite node has discrete coordinates $(n_x - 1, n_z - 1)$. The number of shortest paths from (0,0) to (i,j) is denoted as $p(i,j)$.

The simplest situation is $nsd = 0$, the Streets of New York (see Figure 3, left). The recurrence relations for $p(i,j)$ are

$$p(i,0) = p(0,j) = 1 \tag{7}$$

$$p(i+1,j+1) = p(i+1,j) + p(i,j+1), \tag{8}$$

both for $i,j \geq 0$. These follow from the observation that to reach $(i+1,j+1)$ along shortest paths, one should travel via $(i,j+1)$ or $(i+1,j)$. The relations can be solved with the help of a generating function $g(x,z) = \sum_{i,j \geq 0} p(i,j)x^i z^j$ $(0 \leq x,z 1)$, a usual procedure from combinatorial theory:

$$g(x,z) = \sum_{i,j \geq 0} p(i,j)x^i z^j$$

$$= 1 + \sum_{i \geq 1} p(i,0)x^i + \sum_{j \geq 1} p(0,j)z^j + \sum_{i,j \geq 1} p(i,j)x^i z^j$$

$$= 1 + \frac{x}{1-x} + \frac{z}{1-z} + z \sum_{\substack{i \geq 1 \\ j \geq 0}} p(i,j)x^i z^j + x \sum_{\substack{i \geq 0 \\ j \geq 1}} p(i,j)x^i z^j$$

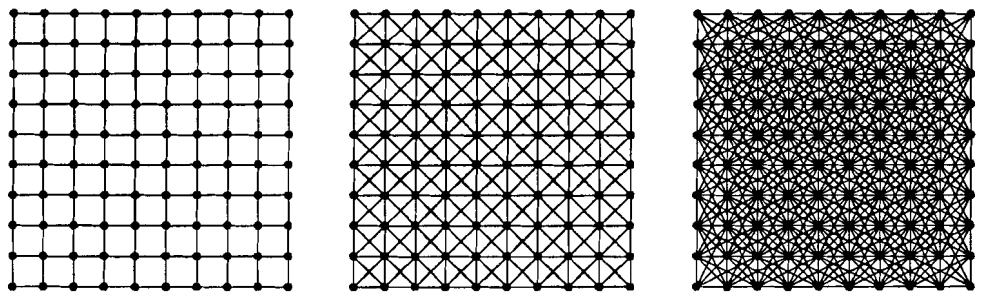

Figure 3. Three networks with 10×10 *nodes and* $nsd = 0$ *(the streets of New York, left),* $nsd = 1$ *(middle) and* $nsd = 2$ *(right).*

$$= 1 + \frac{x}{1-x} + \frac{z}{1-z} + z(g(x,z) - \sum_{j \geq 0} p(0,j)z^j) + x(g(x,z) - \sum_{i \geq 0} p(i,0)x^i)$$

$$= 1 + (x+z)g(x,z) \tag{9}$$

so

$$g(x,z) = \frac{1}{1-x-z} = \sum_{k \geq 0} (x+z)^k = \sum_{k \geq 0} \sum_{i=0}^{k} \binom{k}{i} x^i z^{k-i}$$

$$= \sum_{i,j \geq 0} \binom{i+j}{i} x^i z^j. \tag{10}$$

Therefore, $p(i,j) = \binom{i+j}{i}$. These numbers are arranged as follows:

1	1	1	1	1	1	1	1	1	1
1	2	3	4	5	6	7	8	9	10
1	3	6	10	15	21	28	36	45	55
1	4	10	20	35	56	84	120	165	220
1	5	15	35	70	126	210	330	495	715
1	6	21	56	126	252	462	792	1287	2002
1	7	28	84	210	462	924	1716	3003	5005
1	8	36	120	330	792	1716	3432	6435	11440
1	9	45	165	495	1287	3003	6435	12870	24310
1	10	55	220	715	2002	5005	11440	24310	48620

This is Pascal's triangle containing binomial coefficients, but rotated and scaled into a rectangle. The nodes on the sides of the grid can be reached with a unique shortest path, the shortest paths to the nodes on the diagonal (i,i) $i \geq 1$ are far from unique.

The second case is $nsd = 1$ (Figure 3, middle). The recurrence relations are

$$p(i,0) = p(0,i) = p(i,i) = 1 \tag{11}$$

for $i \geq 0$,

$$p(i+1,j+1) = p(i,j+1) + p(i,j) \tag{12}$$

for $i > j \geq 0$ and

$$p(i+1,j+1) = p(i+1,j) + p(i,j) \tag{13}$$

for $j > i \geq 0$. These numbers are arranged as:

Accuracy

1	1	1	1	1	1	1	1	1	1
1	1	2	3	4	5	6	7	8	9
1	2	1	3	6	10	15	21	28	36
1	3	3	1	4	10	20	35	56	84
1	4	6	4	1	5	15	35	70	126
1	5	10	10	5	1	6	21	56	126
1	6	15	20	15	6	1	7	28	84
1	7	21	35	35	21	7	1	8	36
1	8	28	56	70	56	28	8	1	9
1	9	36	84	126	126	84	36	9	1

This scheme shows two skewed Pascal's triangles fitted together. Now the nodes on the sides and the diagonal of the grid can be reached with a unique shortest path and a strong nonuniqueness appears at the bisections of the diagonal and the sides.

The third case is $nsd = 2$ (Figure 3, right). The recurrence relations are

$$p(i,0) = p(0,i) = p(i,i) = p(2i,i) = p(i,2i) = 1 \tag{14}$$

for $i \geq 0$,

$$p(i+2,j+1) = p(i,j) + p(i+1,j+1) \tag{15}$$

for $0 < j$ and $2j < i$,

$$p(i+2,j+1) = p(i,j) + p(i+1,j) \tag{16}$$

for $0 < j < i < 2j$ and

$$p(i,j) = p(j,i) \tag{17}$$

for $0 < i < j$ because of symmetry along the diagonal. These numbers are arranged in this scheme:

1	1	1	1	1	1	1	1	1	1
1	1	1	2	3	4	5	6	7	8
1	1	1	2	1	3	6	10	15	21
1	2	2	1	3	3	1	4	10	20
1	3	1	3	1	4	6	4	1	5
1	4	3	3	4	1	5	10	10	5
1	5	6	1	6	5	1	6	15	20
1	6	10	4	4	10	6	1	7	21
1	7	15	10	1	10	15	7	1	8
1	8	21	20	5	5	20	21	8	1

By extrapolation of these calculations to higher values of *nsd* it can be conjectured that shortest paths are uniquely determined only for nodes on a few straight lines through (0,0). Between those axes of uniqueness the numbers of possible shortest paths form folded Pascal triangles; the numbers on the bisections of the axes are the greatest and they grow exponentially with the distance from (0,0). Nonuniqueness of shortest paths cannot be avoided in a grid with infinitely many nodes ($n_x = n_z = \infty$) but only a finite number of connections per node ($nsd < \infty$). With a finite number of nodes one can reduce the

nonuniqueness region by increasing the number of arcs per node.

The effect of the nonuniqueness of shortest paths is illustrated in Figures 4 through 8. Figure 4 shows the degenerate case of the shortest paths in the Streets of New York, a homogeneous model covered by a 10×10 grid network with *nsd* = 0, so that each node is connected only with four neighbours. The shortest paths are then forced to follow the directions 0°, 90°, 180° and 270°. Only ray paths that have a favourable orientation with respect to the network are well approximated. The shortest paths from (0,0) to nodes at the upper and left boundaries of the model are uniquely determined and exactly equal to straight lines; the travel times along them are equal to the correct travel times. The nonuniqueness is strongest at the diagonal $(0,0) \rightarrow (n_x - 1, n_z - 1)$; for instance, there are 48620 paths from (0,0) to (9,9). One could hope that the shortest path from the (0,0) to $(n_x - 1, n_z - 1)$ would follow the diagonal as closely as possible so that a further refinement of the path with other methods could be made (Chapter 5, Ray bending revisited). It is not predetermined what paths the shortest path tree algorithm will find; a number of factors influences the outcome. The special choice the algorithm makes depends on the order in which the nodes with tentative travel times are updated (see Chapter 1), which, in turn, depends on the network size and topology but particularly on the nature of the algorithm. Each algorithm, labelsetting, with heaps or buckets, or labelcorrecting, with queues, stacks or double-ended queues, can give other results. Even if the shortest path would follow the diagonal as closely as possible, the travel time along that staircase-like path deviates with a factor $\sqrt{2}$. This is demonstrated by the fact that the travel time along a path (0,0), (0,1), (1,1), (1,2), (2,2) , . . . , $(n_x - 1, n_z - 1)$ is exactly the same as the travel time along the path following the lower left corner (0,0), $(0, n_z - 1)$, $(n_x - 1, n_z - 1)$ or the path via the upper right corner (0,0), $(n_x - 1, 0)$, $(n_x - 1, n_z - 1)$ by step-by-step unfolding the staircase-like path (see Figure 5). However small δx is made, the error remains a factor $\sqrt{2}$ (Figure 6).

Figure 8 shows the shortest paths from (0,0) to the boundary of the model in a homogeneous medium covered by a series of grid networks with increasing *nsd*. The set of available angles for each value of *nsd* is shown in Figure 7. It can be seen that the set of

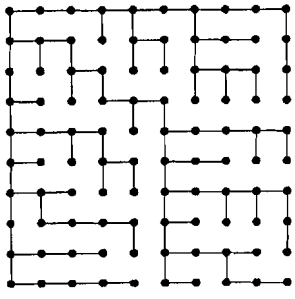

Figure 4. Shortest paths in the streets of New York.

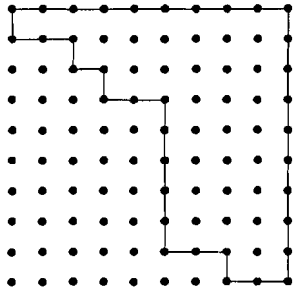

Figure 5. Two shortest paths with the same travel time.

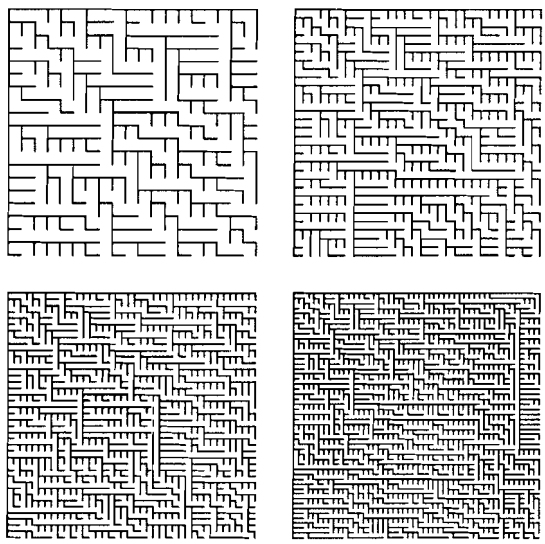

Figure 6. Shortest paths in the streets of New York for n_x = 20, 30, 40 *and* 50.

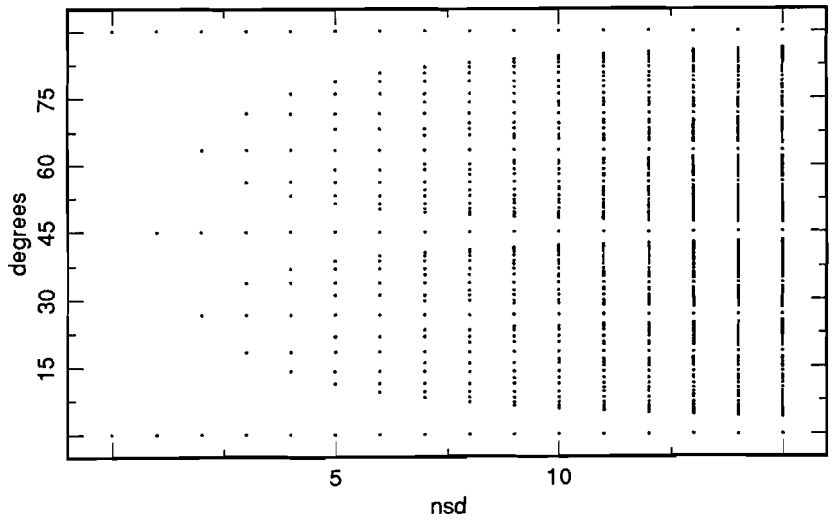

Figure 7. Set of available angles as a function of nsd.

angles is dense in the interval [0°,90°] for $nsd \to \infty$ but not with a uniform density. The distribution shows gaps near 0° and 90° and smaller gaps near 45°, arctg $\frac{1}{2}$ = 63.43497° and arctg 2 = 26.56505°. The travel time error is maximal in these angle ranges and minimal, or zero, where the Fermat paths follow exactly the available directions.

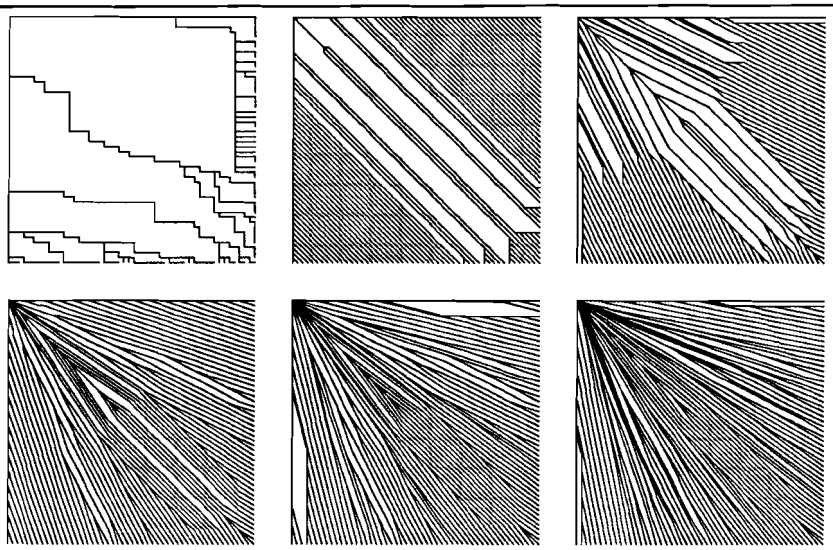

Figure 8. Shortest paths in a homogeneous medium and grids with $n_x = 50$ and (from top left to bottom right) nsd = 0, 1, 2, 3, 4 and 5.

The situation in heterogeneous models is more complicated while the nonuniqueness of shortest paths may be reduced. It is, however, not easy to quantify this reduction. Also, in other types of networks, like the cell networks, the nonuniqueness of shortest paths may not be analysed as easily as in the above case. In any case, the nonuniqueness of shortest paths is a source of errors in the ray paths and the travel times along them even when the weight of each connection is computed correctly.

§3 Asymptotic error analysis

The analysis of §2 suggests an error analysis in terms of the discretisation in coordinates and the discretisation in angles. The error in the ray geometry and in the travel time along the ray, as found by the shortest path method, depends on the number of nodes in each coordinate direction, or the adequacy of the sampling of the velocity field, and on the number of connections per node, or the number of available angles for the shortest paths. In the following sections the discretisation in the angles is denoted by $\delta\phi$ and the discretisation in coordinates by δx. The error $E = \bar{T} - T^*$ can be expanded in a series

$$E(\delta\phi, \delta x) = \sum_{i,j \geq 0} \alpha_{ij} \delta\phi^i \delta x^j = \alpha_{00} + \alpha_{10}\delta\phi + \alpha_{01}\delta x + \cdots, \quad (\delta\phi \to 0,\ \delta x \to 0). \tag{18}$$

It will be shown that $\alpha_{00} = \alpha_{10} = \alpha_{01} = 0$.

§3.1 Angle discretisation

The travel time error $E = \tilde{T} - T^*$ caused by the discretisation in the direction space can be examined by taking $\delta x = 0$ and $\delta\phi > 0$. This means that the node density is infinite, but the number of arcs per node is finite. The velocity field is sampled completely, but the range of possible directions is limited to 0°, $\delta\phi$, $2\delta\phi$, $3\delta\phi$, $\cdots$. This situation is mimicked by homogeneous models like the ones described in §2. In the following it is proved that then $E = O(\delta\phi^2)$.

Suppose that a Fermat path F reaches a node A and follows its way to node B under an angle ϕ. The shortest path cannot follow this way because of the angle discretisation. The best alternative is the path via the node C (see Figure 9). The following definitions are made: D is the orthogonal projection of B on the line AC, ϕ is the angle BAD, $\delta\phi$ is the angle BCD, δs is the distance AB. The velocity is constant and equal to c. $\delta T^* = \frac{\delta s}{c}$ is the exact travel time along the Fermat path from A to B and $\delta\tilde{T}$ is the approximate travel time along the shortest path from A to B, which is the exact travel time along the path ACB. Now,

$$\begin{aligned} AB &= \delta s \qquad BD = \sin\phi\delta s \qquad CD = \frac{\sin\phi}{\mathrm{tg}\delta\phi}\delta s \\ AD &= \cos\phi\delta s \quad BC = \frac{\sin\phi}{\sin\delta\phi}\delta s \quad AC = (\cos\phi - \frac{\sin\phi}{\mathrm{tg}\delta\phi})\delta s \end{aligned} \tag{19}$$

so

$$\delta\tilde{T} = \frac{1}{c}(AC + BC)$$

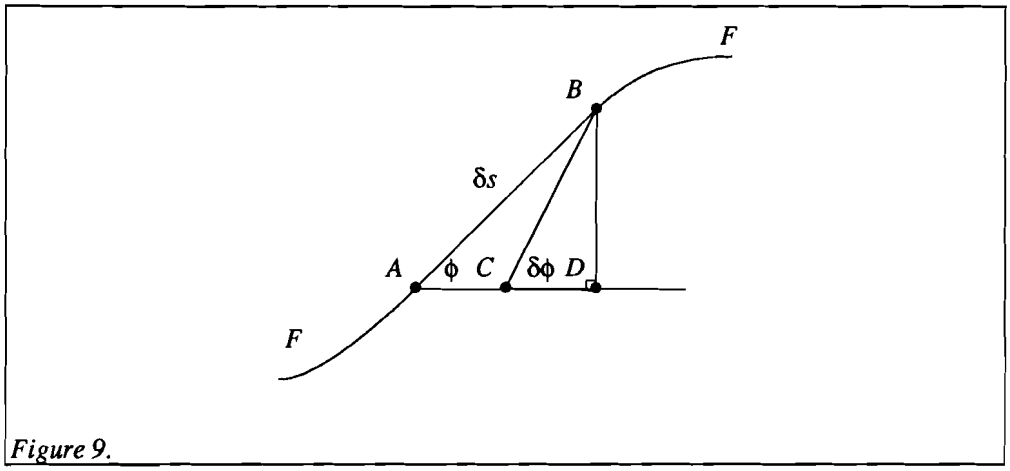

Figure 9.

$$= \frac{1}{c}(\cos\phi - \frac{\sin\phi}{\mathrm{tg}\delta\phi} + \frac{\sin\phi}{\sin\delta\phi})\delta s$$

$$= (\cos\phi + \sin\phi\,(\frac{1-\cos\delta\phi}{\sin\delta\phi}))\delta T^*$$

$$= (\cos\phi + \sin\phi\ \mathrm{tg}\frac{1}{2}\delta\phi)\delta T^*. \tag{20}$$

From this formula, the error estimation of the Streets of New York can be found again by putting $\delta\phi = 90°$. Then

$$\delta\tilde{T} = (\cos\phi + \sin\phi)\delta T^* \tag{21}$$

so that $\delta\tilde{T} = \delta T^*$ for $\phi = 0°$ or $\phi = 90°$, corresponding to the sides of the grid where the shortest paths are unique. The expression $\cos\phi + \sin\phi$ is at most $\sqrt{2}$ for $\phi = 45°$ so that $\delta\tilde{T} = \sqrt{2}\,\delta T^*$ for nodes on the diagonal of the grid, just as is argued in §2.

As $\delta x = 0$, δT^* can be chosen arbitrarily small equation (20) can be integrated along the whole Fermat path yielding the error E:

$$E(\delta\phi) = \tilde{T} - T^* = \int_F (\cos\phi + \sin\phi\ \mathrm{tg}\frac{1}{2}\delta\phi - 1)\mathrm{d}T^*. \tag{22}$$

This integral can be bracketed as

$$0 \le E(\delta\phi) \le \max_{0\le\phi\le\delta\phi}(\cos\phi + \sin\phi\ \mathrm{tg}\frac{1}{2}\delta\phi - 1)\int_F \mathrm{d}T^*. \tag{23}$$

Let $\varepsilon(\phi, \delta\phi) = \cos\phi + \sin\phi\ \mathrm{tg}\frac{1}{2}\delta\phi - 1$, then $\varepsilon(\phi, \delta\phi)$ vanishes for $\phi = 0°$ and $\phi = \delta\phi$ and $\varepsilon(\phi, \delta\phi)$ is positive between $\phi = 0°$ and $\phi = \delta\phi$. To find its maximum, put

$$\frac{\partial\varepsilon}{\partial\phi} = -\sin\phi + \cos\phi\mathrm{tg}\frac{1}{2}\delta\phi = 0. \tag{24}$$

This is solved for $\phi = \frac{1}{2}\delta\phi$, so

$$\varepsilon_{\max}(\delta\phi) \equiv \max_{0\le\phi\le\delta\phi}\varepsilon(\phi, \delta\phi) = \cos\frac{1}{2}\delta\phi + \sin\frac{1}{2}\delta\phi\ \mathrm{tg}\frac{1}{2}\delta\phi - 1. \tag{25}$$

The substitution of the series expansions of cos, sin and tg for $\delta\phi \to 0$ gives:

$$\varepsilon_{\max}(\delta\phi) = \frac{1}{4}(1 - \frac{1}{2!})\delta\phi^2 + \frac{1}{16}(\frac{1}{3} - \frac{1}{3!} + \frac{1}{4!})\delta\phi^4 + O(\delta\phi^6) \tag{26}$$

The combination of (23) and (26) gives the final result:

$$E(\delta x = 0, \delta\phi > 0) = O(\delta\phi^2). \tag{27}$$

§3.2 Coordinate discretisation

The travel time error $E = \tilde{T} - T^*$ due to the discretisation in the coordinate space can be derived from the error characteristics of numerical integration schemes (Stoer and Bulirsch, 1980). The situation now to be examined is a finite node density ($\delta x > 0$) and a continuum of angles ($\delta\phi = 0$). The shortest paths are allowed to follow all directions, but the lengths of their segments are of the order of δx. There are no finite networks that can simulate such a situation. The shortest paths to be found contain finite numbers of nodes, but their positions are free because for the connections between nodes all directions must be allowed. A shortest path is thus a polygonal path that minimises

$$\tilde{T} = \sum_{i=1}^{k} \int_{i-1}^{i} \frac{ds}{c}, \tag{28}$$

where the nodes are numbered from 0 to k and their positions denoted with $\vec{r}_i = (x_i, z_i)$ (see Chapter 5). The integral is taken along the straight line between $\vec{r}_{i-1}$ and $\vec{r}_i$ and evaluated numerically. In this chapter the trapezoidal rule is used

$$\int_{i-1}^{i} \frac{ds}{c} = \frac{1}{2}\left(\frac{1}{c_{i-1}} + \frac{1}{c_i}\right) || \vec{r}_i - \vec{r}_{i-1} || , \tag{29}$$

where c_i is the seismic velocity at $\vec{r}_i$. Two assumptions are made: 1. the velocity field is twice differentiable and 2. there is no clustering of nodes. The clustering of nodes is excluded by demanding that $|| \vec{r}_i - \vec{r}_{i-1} || = O(\delta x)$ or, more precisely:

$$\alpha\delta x \le || \vec{r}_i - \vec{r}_{i-1} || \le \beta\delta x \tag{30}$$

for $i = 1, \ldots, k$ and two finite non-zero numbers $0 << \alpha < \beta << \infty$.

In this case,

$$\tilde{T} = \sum_{i=1}^{k} \frac{1}{2}\left(\frac{1}{c_{i-1}} + \frac{1}{c_i}\right) || \vec{r}_i - \vec{r}_{i-1} || \tag{31}$$

is minimal if and only if

$$\frac{\partial \tilde{T}}{\partial \vec{r}_i} = 0, \; i = 0, \ldots, k. \tag{32}$$

Now,

$$\frac{\partial \tilde{T}}{\partial \vec{r}_i} =$$

$$= \frac{1}{2}\frac{\partial}{\partial \vec{r}_i}\left[\left(\frac{1}{c_{i+1}} + \frac{1}{c_i}\right) || \vec{r}_{i+1} - \vec{r}_i || + \left(\frac{1}{c_i} + \frac{1}{c_{i-1}}\right) || \vec{r}_i - \vec{r}_{i-1} || \right] =$$

$$= \frac{1}{2}\left[|| \vec{r}_{i+1} - \vec{r}_i || \frac{\partial}{\partial \vec{r}}\left(\frac{1}{c}\right)(\vec{r}_i) - \left(\frac{1}{c_{i+1}} + \frac{1}{c_i}\right)\frac{\vec{r}_{i+1} - \vec{r}_i}{|| \vec{r}_{i+1} - \vec{r}_i ||} + \right.$$

$$+ \| \vec{r}_i - \vec{r}_{i-1} \| \frac{\partial}{\partial \vec{r}} (\frac{1}{c})(\vec{r}_i) + (\frac{1}{c_i} + \frac{1}{c_{i-1}}) \frac{\vec{r}_i - \vec{r}_{i-1}}{\| \vec{r}_i - \vec{r}_{i-1} \|}]. \tag{33}$$

Therefore,

$$\frac{\partial}{\partial \vec{r}} (\frac{1}{c})(\vec{r}_i) = \frac{2}{\| \vec{r}_{i+1} - \vec{r}_i \| + \| \vec{r}_i - \vec{r}_{i-1} \|} \times \tag{34}$$

$$\times [\frac{1}{2} (\frac{1}{c_{i+1}} + \frac{1}{c_i}) \frac{\vec{r}_{i+1} - \vec{r}_i}{\| \vec{r}_{i+1} - \vec{r}_i \|} - \frac{1}{2} (\frac{1}{c_i} + \frac{1}{c_{i-1}}) \frac{\vec{r}_i - \vec{r}_{i-1}}{\| \vec{r}_i - \vec{r}_{i-1} \|}].$$

In this expression (34)

$$\frac{1}{2} (\frac{1}{c_{i+1}} + \frac{1}{c_i}) \frac{\vec{r}_{i+1} - \vec{r}_i}{\| \vec{r}_{i+1} - \vec{r}_i \|} \tag{35}$$

is the second order approximation of $\vec{p} = \frac{1}{c} \frac{d\vec{r}}{ds}$ at $i + \frac{1}{2}$. Similarly,

$$\frac{1}{2} (\frac{1}{c_i} + \frac{1}{c_{i-1}}) \frac{\vec{r}_i - \vec{r}_{i-1}}{\| \vec{r}_i - \vec{r}_{i-1} \|} \tag{36}$$

is the second order approximation of $\vec{p}$ at $i - \frac{1}{2}$. The whole expression at the right hand side of (34) is thus equal to the second order approximation of $\frac{d\vec{p}}{ds}$ at i.

This means that minimising $\tilde{T}$ is the same as solving the trapezoidal approximation to the standard ray tracing equations:

$$\frac{d\vec{r}}{ds} = c\vec{p}, \quad \frac{d\vec{p}}{ds} = \vec{r} \tag{37}$$

with the boundary conditions $\vec{r}(0) = \vec{r}_0$ and $\vec{r}(S) = \vec{r}_k$, where S is the total arc length along the ray.

Keller (1974) and Pereyra, Lee and Keller (1980) show that (34) has an isolated solution, provided that the discretisation is fine enough. Because of Fermat's principle, the error in the travel time $E = \tilde{T} - T^*$ is then of the second order in $\max_{1 \le i < k} \| \vec{r}_{i+1} - \vec{r}_i \| = O(\delta x)$ so that

$$E(\delta x > 0, \delta \phi = 0) = O(\delta x^2). \tag{38}$$

This result, (38) and (27), confirms the statement made at the beginning of this section, namely that

$$E(\delta \phi, \delta x) = \sum_{i,j \ge 0} \alpha_{ij} \delta \phi^i \delta x^j = \alpha_{00} + \alpha_{10} \delta \phi + \alpha_{01} \delta x + \cdots, \quad (\delta \phi \to 0, \ \delta x \to 0), \tag{39}$$

with $\alpha_{00} = \alpha_{10} = \alpha_{01} = 0$.

§4 Test of the error characteristics

The asymptotical relationship for the error in the travel time can be tested most easily in the cell organisation of the networks because $\delta\phi$ and δx can be changed independently from each other in the cell organisation by changing its parameters, whereas the parameters of the grid organisation interact unfavourably (a larger forward star contains more nodes, so a finer angle discretisation means a coarser coordinate discretisation).

The travel time error is tested in a linear velocity field:

$$c(\vec{r}) = c_0 + (\nabla c\,, \vec{r}), \tag{40}$$

where c_0 is the seismic velocity at $\vec{r} = \vec{0}$, ∇c the constant velocity gradient and (.,.) the standard Euclidian inner product. A linear velocity field can be considered representative for a general heterogeneous model because such a model can be approximated locally by linear velocity regions. The linear velocity field tests both the angle discretisation and the coordinate discretisation. The ray paths are circular arcs and the travel time along them, from a source point $\vec{r}_0$ to a receiver point $\vec{r}$, is

$$T(\vec{r}, \vec{r}_0) = \tag{41}$$

$$= \frac{1}{||\nabla c||} \operatorname{arccosh} \frac{||\nabla c||^2 ||\vec{r} - \vec{r}_0||^2 - |(\nabla c\,, \vec{r} - \vec{r}_0)|^2 + c(\vec{r})^2 + c(\vec{r}_0)^2}{2c(\vec{r})c(\vec{r}_0)},$$

provided that $c(\vec{r}_0) > 0$ and $c(\vec{r}) > 0$ (Červený, 1987).

The travel times in the velocity field $c = 1.0 + 0.01z$ are computed in a series of cell networks with $n_x = n_z$, the number of cells in the x- and z-direction, increasing from 2 to 50 with step 1 and n_r, the number of nodes per cell boundary, increasing from 2 to 15, for

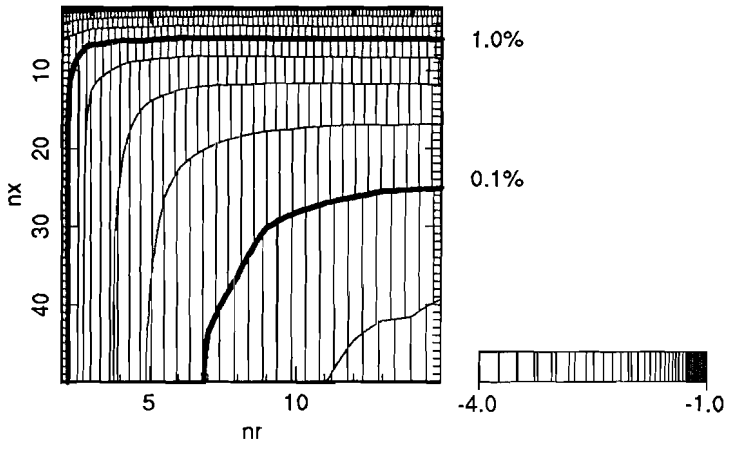

Figure 10. Contourplot of $^{10}\log(E)$ *as a function of* n_r *and* n_x.

each n_x. The networks cover a region of $[0,100] \times [0,100]$. Each cell has a constant velocity, equal to (40) evaluated at the middle of the cell, and the weight of a connection between two nodes is the quotient of their Euclidian distance and the velocity. This is not exactly the trapezoidal approximation of an integral over the slowness, but asymptotically the results are identical. The angle and coordinate discretisation is related to n_x and n_r by $n_x = O(\frac{1}{\delta x})$ and $n_r = O(\frac{1}{\delta \phi})$.

The travel times are computed from the first node of the network, which has coordinates $(0.0,100.0/2n_x n_r)$, to all other nodes. The travel times to nodes that lie closer than 10.0 distance units to the source node are not used for the comparison, because the discretisation in the cell organisation has an unfavourable effect on some of the nodes near the source nodes. The relative travel time error is then defined as

$$E = \frac{|\tilde{T} - T^*|}{T^*}, \tag{42}$$

averaged over all nodes that lie further away from the source node than 10.0. $\tilde{T}$ is the travel time, computed with the shortest path method, T^* is the travel time, computed from (41). The results are shown in Figures 10 and 11. In Figure 10 $^{10}\log(E)$ is contoured as a function of n_r and n_x. Two heavy lines separate the regions for which the error is larger than 1.0% (upper left part), between 1.0% and 0.1% (central part) and below 0.1% (lower right part). For a combination $n_x = n_z = 30$ and $n_r = 10$ the average relative travel time error is below 0.1%. This network has 18,600 nodes and the computation time for one shortest path tree is 32.2 s for the HEAP algorithm (Chapter 2).

Figure 11 shows the error curves as a function of $^{10}\log(n_r)$ for different n_x (left) and *vice versa* (right). In both cases the curves converge to a straight line with a slope of approximately –2. This illustrates the asymptotical relationship for the travel time error

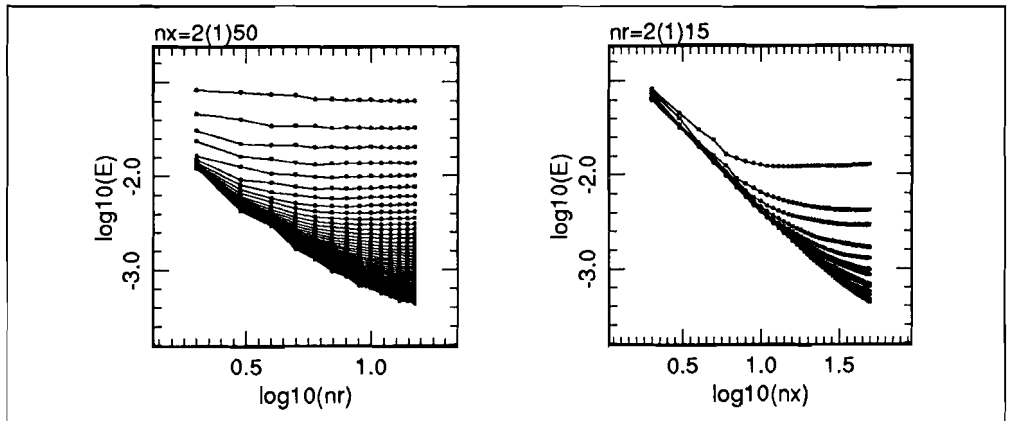

Figure 11. Error curves of $^{10}\log(E)$.

(39). An important conclusion to be drawn from this analysis is that an increase in precision is obtained only by increasing both n_x and n_r. For a given n_x there is one most efficient n_r at a given level of precision. Obviously, an intelligent choice of n_x and n_r is decisive in obtaining an efficient algorithm.

§5 Expedients

In spite of the error estimation of §3 the error in the travel time is in general not very small for moderate numbers of nodes or connections per node. If it is desirable but too expensive to reach the target of 0.1% uniform precision, the results of cheaper calculations could be extrapolated (§5.1). If only a few very accurate ray paths are required, additional bending can be used (§5.2). When the velocity field changes rapidly, one can calculate the travel time along the node connections with a higher order of precision (§5.3).

§5.1 Extrapolation

The asymptotic relationship of §3 can be used to extrapolate the calculations on finite networks to much denser networks or perhaps to a continuum of coordinates and angles.

Let δp be chosen in such a way that $\delta p = A\,\delta x = B\,\delta\phi$ with constant A and B. Then it follows from (39) that the travel time can be expanded in a series in δp

$$\tilde{T} = T^* + Q_1\delta p + Q_2\delta p^2 + Q_3\delta p^3 + \cdots \tag{43}$$

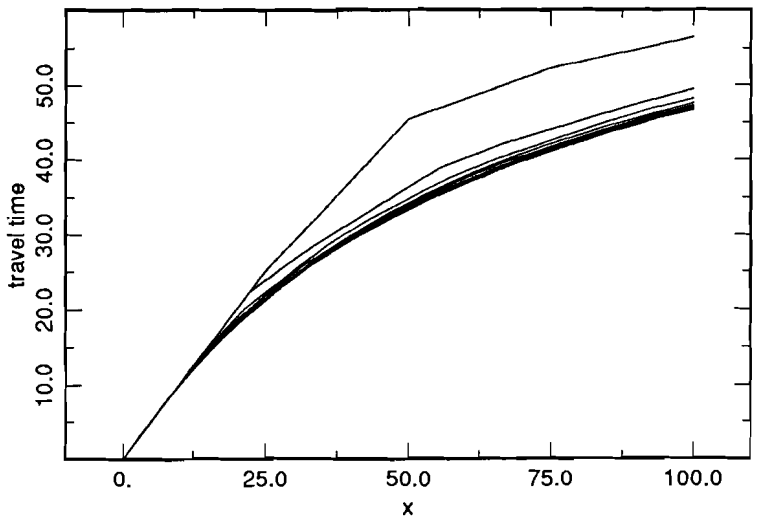

Figure 12. Extrapolation of travel time curves. $n_x = n_z$ *range from* 5 *(top curve) to* 50 *with step* 5. *nsd* $= \lfloor \sqrt{n_x} \rfloor$.

with $Q_1 = 0$ and T^* and Q_i $(i > 1)$ unknown. For $\delta p = 0$ the error is zero and the travel times are calculated exactly (see Bender and Orszag, 1984). $\delta p = 0$ corresponds to the ideal, but not realisable, case of an infinitely dense network with an infinite node density and an infinite number of connections per node. The idea of extrapolation is now to calculate $\tilde{T}$, the travel time along some shortest path, for a set of discretisations that tend to zero. A polynomial is fitted to these calculations. The evaluation of the polynomial at zero, T_0, is then a much better approximation for T^*.

A set of N discretisations can be defined for a fixed combination δp of δx and $\delta\phi$ by $\alpha_i \delta p$ $(i = 1, \ldots, N)$, where α_i are decreasing numbers $\alpha_i \to 0$. A suitable choice is δp, $(\frac{1}{2})\delta p$, $(\frac{1}{2})^2\delta p$, $\cdots$, $(\frac{1}{2})^{N-1}\delta p$. For each discretisation $\alpha_i \delta p$ a travel time $\tilde{T}_i$ is calculated by running the shortest path algorithm in the corresponding network. The polynomial has then order $N - 1$ and its coefficients $T_0, Q_1, \cdots, Q_{N-1}$ are determined by the calculated travel times $\tilde{T}_i$. They are found by solving a linear system of equations:

$$\begin{aligned}
\tilde{T}_1 &= T_0 + Q_1\alpha_1\delta p + Q_2\alpha_1^2\delta p^2 + \cdots + Q_{N-1}\alpha_1^{N-1}\delta p^{N-1} \\
&\cdots \\
\tilde{T}_i &= T_0 + Q_1\alpha_i\delta p + Q_2\alpha_i^2\delta p^2 + \cdots + Q_{N-1}\alpha_i^{N-1}\delta p^{N-1} \\
&\cdots \\
\tilde{T}_N &= T_0 + Q_1\alpha_N\delta p + Q_2\alpha_N^2\delta p^2 + \cdots + Q_{N-1}\alpha_N^{N-1}\delta p^{N-1}.
\end{aligned} \tag{44}$$

The unknowns are $Q_1\delta p, Q_2\delta p^2, \cdots, Q_{N-1}\delta p^{N-1}$ and T_0, everything else is known. T_0 can then be solved by Cramer's rule. For $\alpha_i = (\frac{1}{2})^{i-1}$ and $N = 3$, this becomes:

$$T_0 = \frac{\det\begin{bmatrix} \tilde{T}_1 & 1 & 1 \\ \tilde{T}_2 & \frac{1}{2} & \frac{1}{4} \\ \tilde{T}_3 & \frac{1}{4} & \frac{1}{16} \end{bmatrix}}{\det\begin{bmatrix} 1 & 1 & 1 \\ 1 & \frac{1}{2} & \frac{1}{4} \\ 1 & \frac{1}{4} & \frac{1}{16} \end{bmatrix}} = \frac{\tilde{T}_1 - 6\tilde{T}_2 + 8\tilde{T}_3}{3} \tag{45}$$

which is a weighted average of the three calculated travel times $\tilde{T}_1$, $\tilde{T}_2$ and $\tilde{T}_3$. T_0 is always a linear combination of $\tilde{T}_1, \cdots, \tilde{T}_N$ so it is not difficult to compute.

(45) is a practical form of the extrapolation procedure to work with because the error in $\tilde{T}$ is halved each time and generally it is too expensive to calculate more than three travel times.

Figure 12 illustrates the extrapolation of a series of travel time curves. The velocity field is given by $c = 1.0 + 0.1z$ and the region $[0,100] \times [0,100]$ is covered by a series of grid

networks with increasing node density and arc density. The number of nodes in both coordinate directions, $n_x = n_z$, ranges from 5 to 50 with step 5 so that the total number of nodes ranges from 25 to 2500. The forward star sizes are chosen so that the ratio between the number of nodes and the number of connections per node is approximately constant: $nsd = \lfloor \sqrt{n_x} \rfloor$. The travel time curves for $z = 0$ and $0 \le x \le 100$ are shown in Figure 12. They converge to the exact travel time, given by (41).

The travel times from (0,0) to (100,0) are tabulated in Table 1.

The extrapolated travel time from (0,0) to (100,100) can be calculated from, for instance, $(n_x, nsd) = (5,2), (10,3)$ and (20,4) by formula (45):

$$T_0 = \frac{\tilde{T}_1 - 6\tilde{T}_2 + 8\tilde{T}_3}{3}$$

$$= \frac{56.4703 - 6 \times 49.5346 + 8 \times 47.6541}{3}$$

$$= 46.8318, \tag{46}$$

which is a much better approximation of T^* than 47.6541. The combination $(n_x, nsd) = (10,3), (20,4)$ and (40,6) yields the even more accurate result: $T_0 = 46.2079$.

The extrapolated travel times T_0, calculated in this way, also obey a kind of asymptotics. The idea of extrapolation is meant to eliminate lower-order terms in favour of higher order terms. A hypothetical case is when the series for the calculated travel times $\tilde{T}_i$ ends after a finite number (M) of terms:

$$\tilde{T}_i = T_0 + Q_1 \alpha_i \delta p + \cdots + Q_M \alpha_i^M \delta p^M \tag{47}$$

so that $Q_{M+1}, Q_{M+2}, \cdots$ are zero. When there are now $M + 1$ different calculations of $\tilde{T}$ and a polynomial of order M is fitted to them, the extrapolated travel time T_0 is exactly T^* no matter what choice is made for the numbers α_i. This means that the first until the M-th order effects are removed and only the zero-th order effect remains.

Table 1. Travel time extrapolation.

n_x	nsd	$\tilde{T}$
5	2	56.4703
10	3	49.5346
15	3	48.2872
20	4	47.6541
25	5	47.2965
30	5	47.1052
35	5	46.9649
40	6	46.8767
45	6	46.8088
50	7	46.7438
exact value	T^*	46.2488

In more realistic cases the series does not end and extrapolation removes as many non-zero-th order terms as there are travel times calculated; higher order effects remain. For three travel times calculated each time, the extrapolated values can be expected to be correct to $O(\delta p^3)$.

§5.2 Additional bending

It is seldom necessary to compute the travel times to all points in a model with uniform accuracy. More often, the travel times of only a small number of rays are required with an error less than 0.1%. The shortest paths calculated in a coarse grid may then be used as initial guesses for additional bending. The bending procedure is described in Chapter 5. The advantage of the shortest path method with regards to bending is that the shortest paths are safe initial guesses that likely converge to the Fermat path for the ray with the absolute minimum travel time.

Figure 13 shows the additional bending of a shortest path. The velocity field is $c = 1.0 + 0.01z$, the grid has 20×20 nodes and $nsd = 3$. The small points are the nodes of the network, black circles are the support points of the Beta-splines at subsequent iterations (see Chapter 5). The travel time from (0,0) to (100,100) is improved from 96.4784 to 96.2473 in two iterations of the additional bending procedure with a computation time of 1.62 s. The exact value is 96.2424 (from formula (41)), so the accuracy is improved from 0.2% for the shortest path calculation to 0.005% for the additional bending.

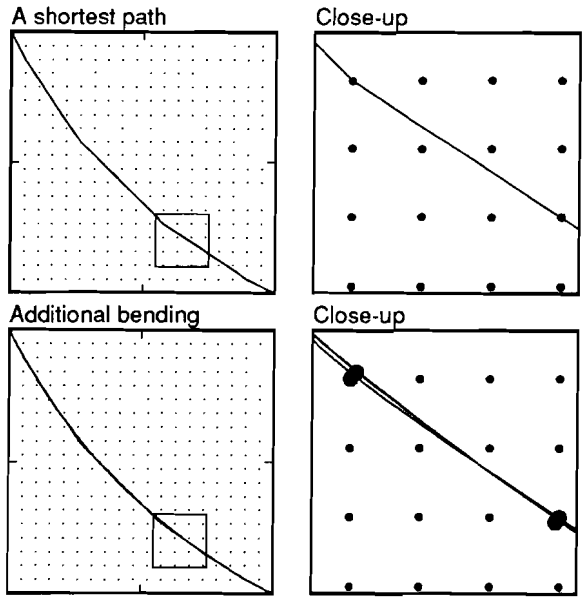

Figure 13. Additional bending.

§5.3 Higher order integration

When the velocity field changes rapidly from point to point, higher order integration can be used to diminish the error due to space discretisation. In §3.2 it is derived that the trapezoidal rule for integration (29) causes a quadratical dependence of this error (38). A higher order integration rule is Simpson's rule:

$$\int_{x_1}^{x_3} f(x)\mathrm{d}x = \frac{1}{3}h(f(x_1) + 4f(x_2) + f(x_3)) + O(h^5 f^{(4)}(\xi)). \tag{48}$$

This rule requires three equidistant points x_1, x_2 and x_3 with distances h. ξ is an unknown point between x_1 and x_3. A parabola is fitted through these points and integrated to give an approximation to the exact integral. However, nodes on shortest paths are generally not equidistant, nor are they positioned on one line, but rather on a polygonal path. Therefore, a generalisation of Simpson's rule must be used. The polygonal path is rectified and the arc length s along it serves as abscissa, while the slowness $\frac{1}{c}$ serves as ordinate. For each trio of nodes $k, j = prec(k), i = prec(prec(k)) = prec(j)$ (see Figure 14), a parabola is fitted to the slownesses in the three nodes and integrated from the second to the third to give a better approximation to the travel time between them.

This works as follows. Let $s_1 = ||\vec{r}_j - \vec{r}_i||$ and $s_2 = s_1 + ||\vec{r}_j - \vec{r}_k||$ and c_i^{-1}, c_j^{-1} and c_k^{-1} the slownesses at i, j and k. The parabola has the form $c^{-1} = a_0 + a_1 s + a_2 s^2$ and is known in three points, so there are three equations for a_0, a_1, a_2:

$$\begin{aligned} c_i^{-1} &= a_0 \\ c_j^{-1} - c_i^{-1} &= a_1 s_1 + a_2 s_1^2 \\ c_k^{-1} - c_i^{-1} &= a_1 s_2 + a_2 s_2^2. \end{aligned} \tag{49}$$

a_1 and a_2 follow from Cramer's rule:

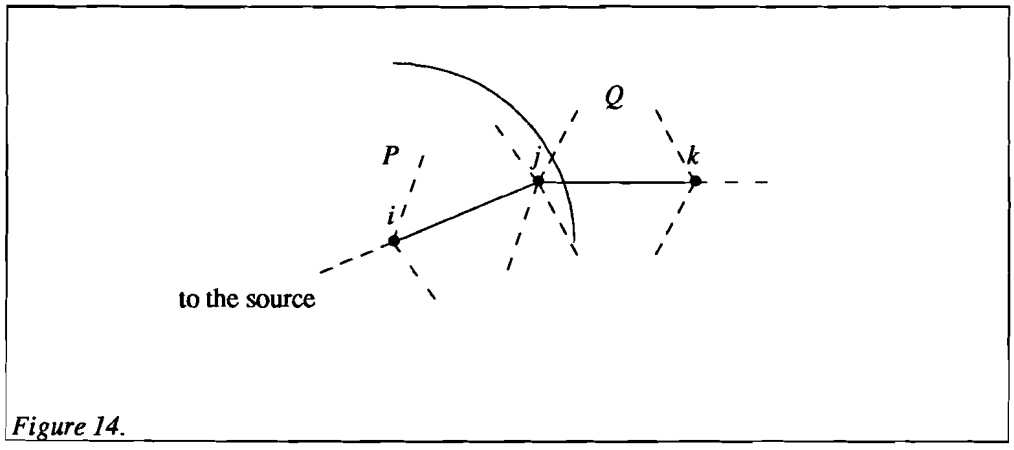

Figure 14.

$$a_1 = \frac{(c_j^{-1} - c_i^{-1})s_2^2 - (c_k^{-1} - c_i^{-1})s_1^2}{s_1 s_2^2 - s_2 s_1^2} \quad a_2 = \frac{(c_j^{-1} - c_i^{-1})s_2 - (c_k^{-1} - c_i^{-1})s_1}{s_1 s_2^2 - s_2 s_1^2}. \tag{50}$$

The travel time from j to k along the shortest path is then:

$$I_{jk} = \int_{s_1}^{s_2} c^{-1} \mathrm{d}s = (a_0 s_2 + \frac{1}{2} a_1 s_2^2 + \frac{1}{3} a_2 s_2^3) - (a_0 s_1 + \frac{1}{2} a_1 s_1^2 + \frac{1}{3} a_2 s_1^3). \tag{51}$$

This formula can be used in two stages of the execution of the shortest path tree algorithm. One possibility is to implement it in each updating step of the algorithm. An updating step consists of giving new values to the travel times of the nodes k from the forward star of the last permanently labeled node j according to:

$$tt(k) = \min(tt(k), tt(j) + d_{jk}) \tag{52}$$

(see Chapter 2). The trapezoidal rule calculates d_{jk} as (29), the higher order rule as I_{jk} (51). When the tentative travel times are updated with the higher order rule, the shortest paths may differ from the paths computed with the trapezoidal rule. As the shortest path tree is rather insensitive to small changes in the weights of the arcs, this will rarely happen. Therefore, it is generally more efficient to take the second possibility, namely first to compute the shortest path tree with the trapezoidal rule and then improving the travel times with a higher order interpolation scheme.

The higher order interpolation rule is tested in a velocity model $c = 1.0 + 0.01z$ with spatial extensions $0 \le x \le 100$ and $0 \le z \le 100$ and a network with $n_x = 50$ and $nsd = 5$. The trapezoidal rule travel times, the higher order rule travel times and the exact travel times are calculated and tabulated in Table 2 for a subgrid of 3×3 points. $\tilde{T}(Trap)$ is the travel time along a shortest path to (i_x, i_z) according to the trapezoidal rule, $\tilde{T}(Simpson)$ is the travel time along the same path according to Simpson's rule and T^* is the exact travel time from formula (41). Almost all travel times are indeed improved by the higher order rule.

Table 2.

i_x	i_z	$\tilde{T}(Trap)$	$\tilde{T}(Simpson)$	T^*
9	9	23.8483	23.8470	23.8189
9	29	48.6767	48.6490	48.6350
9	49	70.5860	70.5239	70.4289
29	9	56.2774	56.2535	56.2147
29	29	65.2351	65.1848	65.1790
29	49	80.0816	80.0372	80.0149
49	9	90.4564	90.4116	90.3477
49	29	89.1913	89.1474	89.1215
49	49	96.3095	96.2472	96.2424

Conclusion

The main conclusion of the analysis of this chapter is that the error in the travel times computed by the shortest path method depends quadratically on both the angle discretisation and the coordinate discretisation. The error in the ray geometry is one order of magnitude larger than the travel time error due to Fermat's principle. With the help of the asymptotical relationship between the travel time error and the network parameters it is possible to estimate and control the accuracy in relation to the computation time. Also, extrapolation of the results to finer networks is possible, due to the asymptotical relationship. A standard uniform error of 0.1% in the travel time can be reached for moderately complicated velocity models with networks of moderate size. When this may be not enough for a selected set of ray paths, additional bending can be applied to improve the travel time and the ray path. In fact, the shortest path method is especially fit to be combined with the bending method, because for adequate networks the shortest paths are safe initial guesses that are guaranteed to lie close to the global minimum travel time path.

CHAPTER 5

Ray bending revisited

Abstract

Two improvements on conventional ray bending are presented in this chapter. First, gradient search methods are proposed to find the locations of successive points on a ray such that the travel time between its endpoints is minimal. Since only the integration of the travel time along the whole ray is involved, such methods are inherently more stable than methods that use first and second derivatives of the ray path to solve the ray tracing equations. Secondly, it is shown that interpolation between successive points on a ray is generally necessary to obtain sufficient precision in the quadrature but also advantageous in terms of efficiency when Beta-splines are used.

Velocity discontinuities are easy to handle in the proposed bending algorithm. The target for the travel time precision is 1 part in 10^4, which can be considered necessary for many applications in seismic tomography. With the proposed method this precision was reached for a constant gradient velocity model using the Conjugate Gradients algorithm on a 5-point Beta-spline in single precision arithmetic. Several existing methods for parametrisation and minimisation failed to produce this target on this simple model even with computing times that are an order of magnitude larger. Calculations in a spherical Earth yield the required precision within feasible computation times. Finally, alternative search directions, that can be obtained from an approximate second order expansion of the travel time, accelerate the convergence in the first few iterations of the bending process.

Introduction

The steadily increasing ability to map the lateral heterogeneity of the Earth's crust and deeper regions not only makes stricter demands on the precision of seismic observations such as travel times; inversion procedures for such data, notably the more recent tomographic methods (Nolet, 1987), are also more exacting on the precision and efficiency of the algorithms used to solve the forward problem. This chapter takes a new look at the old problem of seismic ray tracing with a special emphasis on precision without loss of efficiency.

To find the ray geometry and travel time for a seismic ray between two fixed points A and B in the Earth there are essentially two different computational approaches (Aki and Richards, 1980, ch. 13): in the method of shooting rays are computed that leave point A in different directions by solving the ray tracing equations until one ray happens to arrive in B. In the bending method some initial guess of the ray is perturbed while A and B are kept fixed until it satisfies the ray equations or minimises the travel time.

One advantage of the method of ray bending over the method of ray shooting is that the

---

This chapter will be published as:

Moser, T.J., Nolet, G., and Snieder, R., Ray bending revisited, Bull. of the Seism. Soc. of Am., Volume 82:1.

bending method yields the travel time of the diffracted ray, even if point B is in the shadow zone where no 'physical' rays arrive from A. Of course, the term 'physical' is somewhat misleading since in the real Earth diffracted seismic energy does arrive in B and may have been observed as an arrival time. In tomographic interpretations such observations must be discarded when the direct problem is solved by shooting rather than bending. Also, when the diffracted wave arrives before the physical ray, it is likely that the fastest of the two is observed. Unless this is taken into account, tomographic studies are likely to yield models that are biased towards high velocities (Wielandt, 1987).

Bending methods have been introduced in seismology by Wesson (1971), Chander (1975) and Julian and Gubbins (1977). Two variants exist of the bending method. In the original method (Julian and Gubbins, 1977), of which the most elaborate version is by Pereyra, Lee and Keller (1980), the ray tracing equations:

$$\frac{\mathrm{d}}{\mathrm{d}s}\left(\frac{1}{c(\vec{r})}\frac{\mathrm{d}\vec{r}}{\mathrm{d}s}\right) - \nabla\left(\frac{1}{c(\vec{r})}\right) = 0 \tag{1}$$

for a ray $\vec{r}$, parametrised by the arc length s, in a velocity field $c(\vec{r})$, are approximated by discretising the ray path into $k+1$ points $(\vec{r}_0, \vec{r}_1, \cdots, \vec{r}_k)$ and by using finite differences for the derivatives. The resulting system of equations is then linearised and solved to find the perturbation that must be applied to some initial guess of the ray in order to reduce the right hand side of the ray equations to 0. Since the linearised system is approximate, several iterations are needed for convergence.

In the second variant, pioneered by Um and Thurber (1987) and Prothero, Taylor and Eickemeyer (1988), one attempts to minimise directly the travel time as a functional of the ray curve γ:

$$T(\gamma) = \int_{\gamma} \frac{\mathrm{d}s}{c} \rightarrow Min \tag{2}$$

This second variant of ray bending has one distinct advantage over the finite-difference approach. The precision of the finite difference approach is limited by the precision of numerical differentiation, which is inherently more unstable than numerical integration. Therefore, the linearised system in the first variant contains small errors. Since, in our experience, the system is ill-conditioned in the presence of a complicated velocity model, these small errors prohibit convergence to the desired accuracy or may even cause a catastrophic divergence. Such stability problems do not occur when T is minimised directly as in the second variant of ray bending.

Um and Thurber (1987) represent the ray by a number of points, connected by straight line segments. In this chapter such representations are referred to as 'polygonal paths'. In the method presented by Um and Thurber the precision of the integration increases with the number of segments. For efficiency reasons the number of segments grows with the number of iterations. Travel time accuracies of a few parts in 10^4 are obtained using considerable CPU time (see Table 3 of this chapter). All computations reported in this chapter were performed using one NS32532 processor of an Encore MM510 parallel computer with 2.5 Mflop. The method is not suitable for media with velocity discontinuities, nor does it work well with predominantly constant velocity zones. These drawbacks do not exist with a similar method developed by Prothero *et al.* (1988), at the expense, however, of computation times that are larger by a factor 2.5-5.5.

Ray bending revisited

In this chapter two improvements on the method of Um and Thurber (1987) are investigated: ray parametrisations in terms of Beta-splines rather than polygonal paths; and a global rather than a 3-point perturbation scheme to minimise the travel time T. In contrast with Prothero *et al.* (1988) who use the Simplex method for minimisation (Nelder and Mead, 1965), it is shown that the smoothness of the travel time allows for the use of the derivatives of T in minimisation techniques such as the Conjugate Gradients method (Fletcher and Reeves, 1965). The Beta-spline curve representation (Barsky, 1988) is chosen instead of trigonometric polynomials because of its greater flexibility.

Instead of a gradient method for minimising the travel time, one can also use second-order minimisation methods that use information that is present in the Hessian matrix. The last section of this chapter shows that, using an approximate expression for the inversion of the Hessian, this leads to a stable ray bending method.

The presented method has been compared with other bending methods only qualitatively. A comparison with respect to efficiency is only partly useful because of the different convergence behaviour of the methods in the presence of strong heterogeneities or discontinuities. The presented method has no problems with such situations unlike most other methods. For example, Julian and Gubbins (1977) encounter problems in complicated velocity models due to the ill-conditioning of their linearised system of equations and Um and Thurber (1987) have difficulties with constant velocity zones due to their definition of the ray perturbation. A quantitative analysis is given to show the advantages of Beta-splines with regard to polygonal paths both in efficiency and accuracy.

This study was motivated by the need to obtain more precise travel times and yet retain efficiency in the computations to enable three dimensional models to be incorporated in delay-time studies. In seismic tomography the number of data (rays) is notoriously large. A precision of 0.1-0.2 seconds in measured arrival times of 12 minutes (P) or 20 minutes (PKP) is required. Thus, ray tracing algorithms should aim at a precision of about 1 part in 10^4. This shall be used as the target precision in this chapter.

The first section introduces the parametrisation of a ray path by the coordinates of successive points on it and the travel time and its gradient along it as multivariate functions. The second section, on minimisation techniques using derivatives, shows how Conjugate Gradients can be used for ray bending. The Beta-spline curve representation is introduced in the third section. In the fourth section it is shown that bending in the presence of discontinuities causes no problems. The section on higher-order terms gives a search direction obtained from a second order expansion of the travel time, which leads to a faster convergence in the first bending iterations. The proposed bending method is applied to a spherical Earth in the last section.

With the exception of the last section this chapter uses dimensionless quantities for distance, time, velocity and slowness. All velocity fields, except in the last section, are sampled on a 50×50 grid and bilinearly interpolated. The range of the fields is 0 to 100 in both the x- and z-directions.

§1 The travel time function and its derivatives

According to Fermat's principle, the gradient ∇T of the travel time T, considered as a functional on the set of continuous curves γ connecting a specified source-receiver pair, vanishes for a seismic ray path. With exception of some degenerate ray perturbations, such as shifting a finite ray segment over a velocity discontinuity, $T(\gamma)$ is a smooth functional. This suggests the use of $\nabla T(\gamma)$ for finding the minimal travel time path. The variation of the travel time along an initial guess path due to small perturbations in its geometry, for which $\nabla T(\gamma)$ is a measure, gives an indication about how to change the guess path in order to search for a minimum of T.

For computational purposes the paths under consideration can be discretised into polygonal paths consisting of $k+1$ points in a 2- or 3-dimensional space, numbered from 0 to k and connected by straight line segments. Their successive x- and z-coordinates or x-, y- and z-coordinates form a set of $n = 2(k+1)$ or $3(k+1)$ parameters that are stored in an n-vector γ:

$$\gamma = (x_0, z_0, x_1, z_1, \cdots, x_k, z_k). \tag{3}$$

In the rest of this chapter the dimension will be chosen 2 and asymptotical relations for the efficiency and accuracy will be given in terms of n. The extension to three dimensions is trivial.

The travel time along such a path, $T(\gamma)$, can be approximated with the trapezoidal rule:

$$T(\gamma) = \sum_{i=1}^{k} \frac{1}{2}\left(\frac{1}{c_i} + \frac{1}{c_{i-1}}\right)\sqrt{(x_i - x_{i-1})^2 + (z_i - z_{i-1})^2}, \tag{4}$$

where c_i is the seismic velocity at (x_i, z_i). It is a function from $\mathbb{R}^n$ to $\mathbb{R}$ that can be evaluated with $O(n)$ arithmetic operations. A Taylor expansion in γ gives:

$$T(\gamma + \delta\gamma) = T(\gamma) + (\nabla T(\gamma), \delta\gamma) + O(||\delta\gamma||^2), \quad (||\delta\gamma|| \to 0), \tag{5}$$

where $\delta\gamma = (\delta x_0, \delta z_0, \cdots, \delta x_k, \delta z_k)$ is a small perturbation on γ, $(.,.)$ is the standard inner product on the n-dimensional Euclidian space and $||.|| = \sqrt{(.,.)}$ is the norm. The gradient of T in the n-dimensional point γ is an n-vector as well:

$$\nabla T(\gamma) = \left(\frac{\partial T}{\partial x_0}, \frac{\partial T}{\partial z_0}, \frac{\partial T}{\partial x_1}, \frac{\partial T}{\partial z_1}, \cdots, \frac{\partial T}{\partial x_k}, \frac{\partial T}{\partial z_k}\right). \tag{6}$$

For ray bending purposes the end points of the polygonal paths must be kept fixed:

$$\delta x_0 = \delta z_0 = \delta x_k = \delta z_k = 0, \tag{7}$$

so that the corresponding components of the gradient do not need to be computed. The other components can be calculated in several ways, the simplest of which is numerical differentiation of (4). A more reliable differentiation will be given in the section 'Interpolation with Beta-splines'. The central difference formula for numerical differentiation gives:

$$T(\gamma + \delta\gamma) - T(\gamma - \delta\gamma) = 2(\nabla T(\gamma), \delta\gamma) + O(||\delta\gamma||^3), \quad (||\delta\gamma|| \to 0), \tag{8}$$

where $\delta\gamma$ consists of suitable increments δx_i and δz_i (see Stoer and Bulirsch (1980), section 5.4.3). One component of $\nabla T(\gamma)$ can be calculated by changing the appropriate point of the

polygonal path with δx_i or δz_i and $-\delta x_i$ or $-\delta z_i$, taking the difference in travel time along both perturbed paths and dividing this by $2\delta x_i$ or $2\delta z_i$. Such numerical differentiation has a discretisation error which is $O(||\delta\gamma||^2)$ ($||\delta\gamma|| \to 0$) which is more accurate than for instance forward differences. However, $||\delta\gamma||$ cannot be chosen too small, because of the round-off error that comes in when the nearly equal numbers $T(\gamma+\delta\gamma)$ and $T(\gamma-\delta\gamma)$ are subtracted. For each component $T(\gamma+\delta\gamma)-T(\gamma-\delta\gamma)$ needs to be calculated only for the two segments in which $\gamma+\delta\gamma$ and $\gamma-\delta\gamma$ are different. Thus, both the travel time and its gradient can be computed with $O(n)$ operations.

Figure 1 illustrates the smoothness of the travel time functional in a simplified example of a three-point polygonal path using the trapezoidal approximation of the travel time. The gradient, computed from the perturbed paths, gives a correct indication of the direction in which the middle point should be changed in order to decrease the travel time.

§2 Minimisation techniques using derivatives

Since the travel time $T(\gamma)$ and its gradient $\nabla T(\gamma)$ can be easily calculated, minimisation techniques using derivatives can be used for bending an initial guess path γ_0 towards the minimum travel time path. The Conjugate Gradients method is used here because of its modest memory space requirements, namely only four n-vectors, and because of its superior convergence properties. Other methods are Newton's method, that needs $O(n^2)$ memory space, and the Steepest Descent method, that has convergence problems in the presence of a long narrow valley in the travel time function. See Stoer and Bulirsch (1980) for Conjugate Gradients and Newton's method and Conte and de Boor (1980) for the shortcomings of the Steepest Descent method.

In the simple but unrealistic case that the travel time is a quadratic function in γ (equation (9)), the Conjugate Gradients method leads to the minimal travel time path in at most n steps:

$$T(\gamma) = c + (b, \gamma) + \frac{1}{2}(\gamma, A\gamma). \tag{9}$$

Here c is a constant real number, b an n-vector, A a symmetric $n \times n$-matrix. The gradient of the quadratic $T(\gamma)$ is $b + A\gamma$ and the minimum path satisfies $\nabla T(\gamma) = b + A\gamma = 0$.

Starting with γ_0, the Conjugate Gradients method generates a series of paths $\gamma_1, \gamma_2, \cdots$ such that

$$T(\gamma_1) = \min_{u_0} T(\gamma_0 + u_0 \nabla T(\gamma_0)) \tag{10}$$

$$T(\gamma_2) = \min_{u_0, u_1} T(\gamma_1 + u_0 \nabla T(\gamma_0) + u_1 \nabla T(\gamma_1))$$

$$\cdots$$

$$T(\gamma_{j+1}) = \min_{u_0, u_1, \cdots, u_j} T(\gamma_j + u_0 \nabla T(\gamma_0) + \cdots + u_j \nabla T(\gamma_j)).$$

The first path γ_1 can be obtained by a simple line minimisation, that is the minimisation of

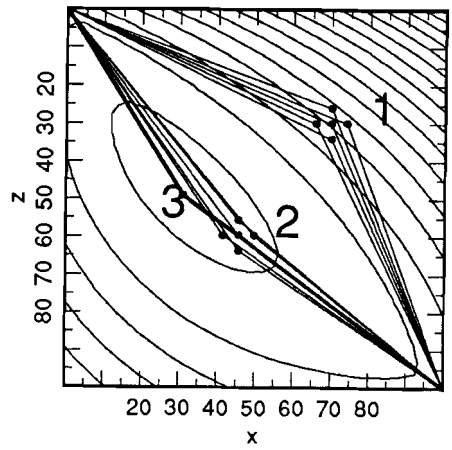

Figure 1. Contours of the travel time along polygonal paths of three points ((0,0),(x,z),(100,100)) *as a function of the position of the second point, computed with the trapezoidal rule. The contours are for travel times from* 100.0 *to* 170.0 *with an increment of* 5.0. *Two guess paths (1 and 2) are shown, together with the perturbations used for the numerical gradient calculation with* $\delta x = \delta z = 4.0$. *Path 3 is the minimal trapezoidal travel time path consisting of three points. The velocity field is given by* $c(x,z) = 1.0 + 0.01z$ *and, as for all figures, sampled on a* 50×50 *grid and bilinearly interpolated. The minimal trapezoidal travel time is* 98.6549, *whereas an analytic solution gives* 96.2424.

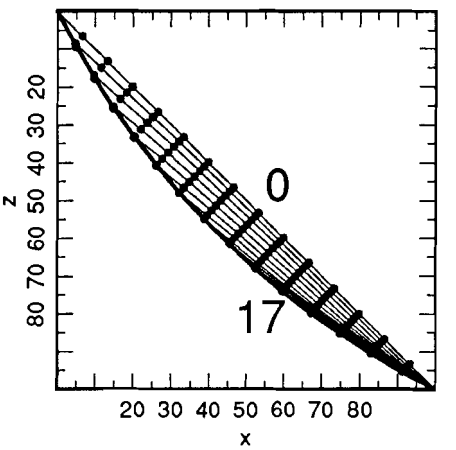

Figure 2. Successive line minimisations of the Conjugate Gradients method from an initial straight guess path consisting of 16 *equidistant points* ((0,0), $\cdots$,(100,100)). *The velocity field is given by* $c(x,z) = 1.0 + 0.01z$. *After* 3.8 *seconds,* 17 *iterations,* 235 *travel time function evaluations and using a tolerance equal to the machine precision the travel time is* 96.2970 *(exact value:* 96.2424*).*

T along the line $\gamma_0 + u_0 \nabla T(\gamma_0)$, where γ_0 and $\nabla T(\gamma_0)$ are fixed vectors and u_0 is a scalar parameter. The second path γ_2 must be obtained by minimising T in the plane $\gamma_1 + u_0 \nabla T(\gamma_0) + u_1 \nabla T(\gamma_1)$, otherwise the result of the first minimisation would be spoiled. The $j+1$-st path γ_{j+1} is found by minimising T in a $j+1$-dimensional subspace of $\mathbb{R}^n$ spanned by the vectors $\nabla T(\gamma_0), \cdots, \nabla T(\gamma_j)$ which are mutually orthogonal. The very expensive $j+1$-dimensional minimisation is avoided by keeping track of another series of vectors $p_0, \cdots, p_j$ such that

$$p_0 = -\nabla T(\gamma_0) \tag{11}$$

$$p_j = -\nabla T(\gamma_j) + \frac{(\nabla T(\gamma_j), \nabla T(\gamma_j))}{(\nabla T(\gamma_{j-1}), \nabla T(\gamma_{j-1}))} p_{j-1}.$$

The minimisations on the right hand side of (10) can then be replaced by line minimisations:

$$T(\gamma_{j+1}) = \min_{\nu} T(\gamma_j + \nu p_j). \tag{12}$$

Only four n-vectors need to be stored in memory, namely γ_j, $\nabla T(\gamma_j)$, $\nabla T(\gamma_{j-1})$ and p_j. For details see Stoer and Bulirsch (1980) or Press *et al.* (1986).

In general, the travel time function is not quadratic, which is immediately clear when it has, for instance, more than one stationary point. The vectors $\nabla T(\gamma_0), \cdots, \nabla T(\gamma_j)$ are not exactly orthogonal and the Conjugate Gradients method may take more steps than n. Press *et al.* (1986) argue that there are $O(n^2)$ function evaluations necessary before finding the minimal point of a quadratic function because it is given by $O(n^2)$ coefficients. Following this argument, one can estimate the total number of operations to find the minimum ray

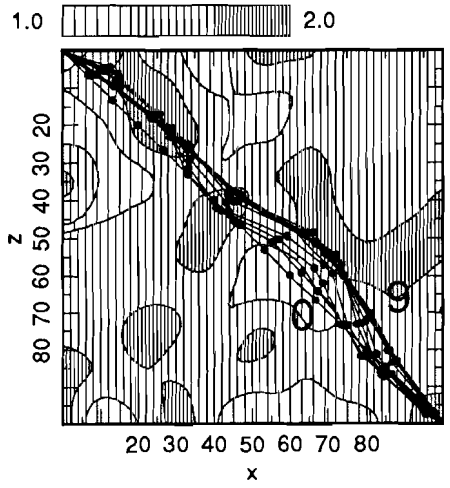

Figure 3. Bending with Conjugate Gradients from an initial straight guess path of 16 *points in a complicated medium with velocities smoothly varying from* 1.0 *to* 2.0. *After* 1.9 *seconds,* 9 *iterations,* 116 *travel time function evaluations and using a tolerance equal to the machine precision the travel time is* 85.7733 *(no analytic solution known).*

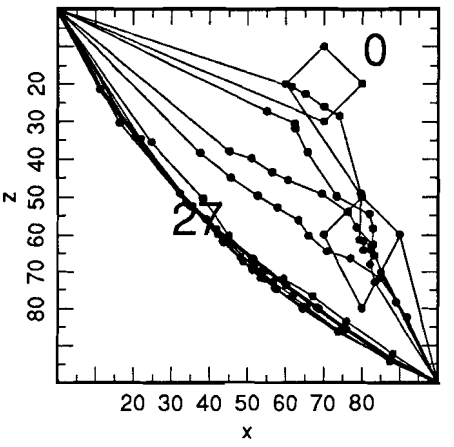

Figure 4. Bending with Conjugate Gradients from an unrealistic initial polygonal guess path with two loops consisting of 12 *points. The velocity field is given by* $c(x,z) = 1.0 + 0.01z$. *After* 4.3 *seconds,* 27 *iterations,* 328 *travel time function evaluations and using a tolerance of* 0.0001, *the travel time is* 96.6638 *(exact value:* 96.2424*). Only every third iteration is shown.*

path as $O(n^3)$, although this is a very coarse rule of thumb that makes no account of the structure of the matrix A nor of the function being not quadratic. In any case, the Conjugate Gradients method cannot diverge because each time a nonzero gradient has been found the line minimisation results in a new travel time which is smaller than the old one and a zero gradient means that a stationary point has been found. Figure 2 shows the performance of the Conjugate Gradients bending of a polygonal path in a linear velocity medium, Figure 3 in a random medium.

In the ideal case of exact arithmetic the process will converge to a stationary travel time path. However, round-off errors in the calculation of the travel time and its derivatives will cause small deviations from the stationary path. Therefore, a tolerance must be defined as a stopping criterion. One possible stopping criterion is the decrease in travel time after an iteration step. If it is smaller than some predescribed tolerance, the process can be stopped. This tolerance can be chosen as small as the computational precision of T. Another possibility is the change in position of the minimum point after an iteration. This tolerance should not be chosen smaller than the square root of the computational precision of T (Press *et al.*, 1986).

A completely different error source is the approximation of continuous curves with polygonal paths and the travel time along them with the trapezoidal rule. In exact arithmetic, the approximation of the travel time along a continuous curve with the trapezoidal rule is correct up to $O(\frac{1}{k^2})$ $(k \to \infty)$. When k is too small, the polygonal path with a minimal trapezoidal travel time can be completely wrong because it may have missed rapid variations in the velocity field. One way out is to refine it by doubling the number of nodes. The integration is then more accurate but the cost of this improvement is to increase the number of arithmetic operations by a factor of roughly 8 because of the efficiency bound of $O(n^3)$. An alternative for the computation of T will be given in the next section.

Several strategies are possible for choosing an initial guess, stopping criteria and refining. In this chapter the convergence of the relative change in travel time to the machine precision of 1.0×10^{-8} is used as a stopping criterion, unless otherwise stated. Refining is omitted because the interpolation with Beta-splines (next section) is a more general way to enhance the accuracy. An example of an initial guess path with two loops (Figure 4) shows that unrealistic guesses are no problem using Conjugate Gradients. Including higher-order terms in the Taylor expansion of the travel time (5) causes the initial guess to converge rapidly towards the neighbourhood of a minimum path (see section on 'Higher-order terms'). It is by no means clear whether the resulting path is a global or only a local minimum. All that can be said is that an initial guess path will converge to the minimal travel time path that is closest to it in some sense. Also, in the case of multipathing, it is not *a priori* clear which one of the paths will be selected. However, advanced techniques from graph theory (Moser, 1991) provide good initial guesses and guarantee that they are close to the global minimum.

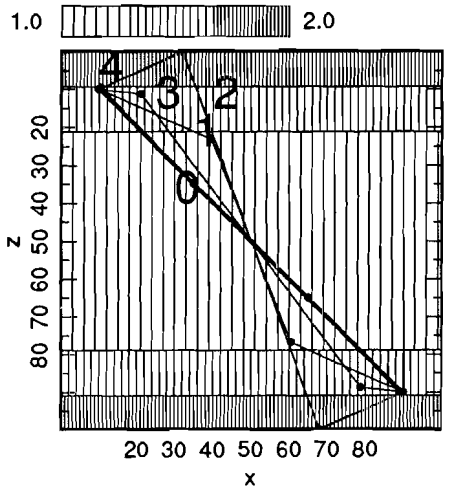

Figure 5. Conjugate Gradients bending of a straight polygonal path of 4 *points* ((10,10),(35,35),(65,65),(90,90)) *in a velocity field given by* $c(x,z) = \frac{(z-50)^2}{2500} + 1$ *whose inverse is a concave function around* $z = 50$. *The successive positions of the second point are given with numbers* $0, \cdots, 4$. *After* 0.5 *seconds,* 9 *iterations,* 148 *travel time function evaluations and using a tolerance equal to the machine precision the travel time is* 69.0294.

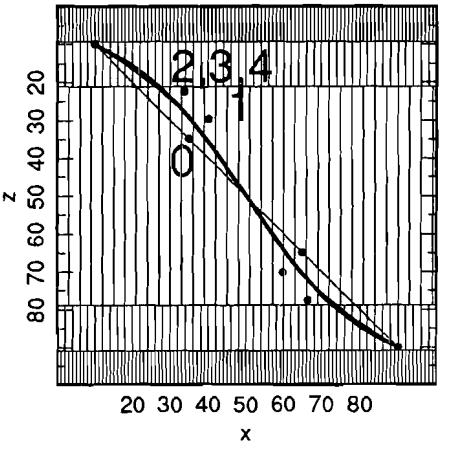

Figure 6. Conjugate Gradients bending of Beta-splines in the same situation as in Figure 5 ($k = 3$, $m = 10$, *see text). The successive positions of the second point are given with numbers* 0, 1, 2. *After* 1.9 *seconds,* 4 *iterations,* 60 *travel time function evaluations and using a tolerance equal to the machine precision the travel time is* 94.1967 *(no analytic solution but* $k = 8$ *and* $m = 10$, *which can be considered as more accurate, gives* 94.2376*).*

§3 Interpolation with Beta-splines

There are two important reasons for interpolation between the points of the polygonal paths.

First, consider the behaviour of trapezoidal travel time minimisation with polygonal paths in concave slowness regions. The slowness distribution, $\sigma(x,z) = \frac{1}{c(x,z)}$, is concave in a region $D \subset \mathbb{R}^2$ when for all $(x_1, z_1), (x_2, z_2) \in D$ and $0 \leq \lambda \leq 1$

$$\sigma(\lambda) \geq \lambda\sigma_1 + (1-\lambda)\sigma_2 \tag{13}$$

where $\sigma(\lambda) = \sigma(\lambda x_1 + (1-\lambda)x_2, \lambda z_1 + (1-\lambda)z_2)$, $\sigma_1 = \sigma(x_1, z_1)$ and $\sigma_2 = \sigma(x_2, z_2)$. In such a region the linear interpolation between the slownesses at two points (x_1, z_1) and (x_2, z_2), the right hand side of (13), is smaller than or equal to the slowness itself, the left hand side of (13). Therefore, the trapezoidal rule, which is an integral over the linearly

interpolated slowness, is an underestimation of the exact travel time, which is an integral over the real slowness. This means that a minimisation algorithm will always repel the points of the polygonal paths out of the concave slowness region. The result is one segment of the polygonal path 'bridging' the region of concavity and a completely incorrect travel time.

Concave slowness regions appear in all realistic models; a common example is the low velocity zone. Consider for instance the velocity distribution

$$c(x,z) = \frac{(z-50)^2}{2500} + 1, \tag{14}$$

whose inverse, the slowness, is concave in a zone around $z = 50$ where c has its minimum, namely $(1 - \frac{1}{3}\sqrt{3})50 \leq z \leq (1 + \frac{1}{3}\sqrt{3})50$. The subsequent iterations of the Conjugate Gradients bending of an initial polygonal path with points ((10,10),(35,35),(65,65),(90,90)) are shown in Figure 5. It can be seen that the second and third point, initially at (35,35) and (65,65), are removed from the concave region and eventually coincide with the source point (10,10) and the receiver point (90,90). The trapezoidal travel time is then equal to the travel time along a straight ray path in a homogeneous model with a velocity equal to $c(10,10) = 1.64$.

One possible solution to this problem is to use equidistributing paths for which the spacing between the points is adapted to the variations in the velocity field (Pereyra *et al.*, 1980). However, this requires extra computation time whereas interpolation results in a considerable saving of time as will be shown below.

A second reason to interpolate polygonal paths is the observation that seismic ray paths are continuous curves. Two points on a ray with a small difference in travel time will have a small difference in spatial position. Therefore, the successive points of a polygonal path are not completely free to move with respect to each other and constraints should be formulated for the minimisation of $T(\gamma)$ in order to reduce the part of the space $\mathbb{R}^n$ wherein the minimum is sought. However, it is a delicate question how to formulate such continuity constraints so that the omission of parts of $\mathbb{R}^n$ compensates the overhead of the extra computation. A good alternative is then to use fewer points and interpolate between them. This reflects the fact that not all the points that are involved in the numerical integration for computing the travel times need to be perturbed independently from each other. In other words, the number of points to be perturbed can be smaller in magnitude than the number of points to be integrated over.

§3.1 Beta-splines

The Beta-spline curve representation (Barsky, 1988; see also Appendix A, 'Beta-splines') is a modification of the cubic B-spline curve (Newman and Sproull, 1979) with several properties that are in favour of the ray bending procedure, described in the previous section.

Its parametric representation allows the ray path or its approximations to be multivalued in each coordinate direction so that there is no principal restriction to the curve: it can make loops or oscillations if necessary. On the other hand, its convex hull property ensures complete control on the behaviour of the curve because each segment of the Beta-spline

curve lies in the convex hull of its four support points. Other curve interpolations or representations that do not obey this property, like the minimal curvature spline (Reinsch, 1967) or the trigonometric polynomial, can show undesired wanderings or oscillations (Gibbs' phenomenon, see Courant and Hilbert, 1967). Generally, the Beta-spline curve does not pass through its support points but it can be forced to do so by making three successive support points coincident, for instance the end points of the ray or points on interfaces. Another property, the local control property, makes the recomputation of one Beta-spline segment possible after the perturbation of one support point without recomputing the rest of the curve. This property is of great advantage in the calculation of the travel time and its gradient because they can be evaluated with the same efficiency as in the case of polygonal paths. The behaviour of the Beta-spline curve is further controlled by two shape parameters β_1 and β_2 that describe the continuity of the first and second derivatives at the joints of the segments. $\beta_1 = 1$ and $\beta_2 = 0$ is the most natural case when both derivatives are continuous. The Beta-splines are equal to cubic B-splines in this setting. For $\beta_1 = \beta_2 = 0$ the Beta-spline curve consists of straight line segments between the support points; therefore, both cubic B-splines and polygonal paths are special cases of the Beta-spline representation of the seismic ray.

Like the points specifying polygonal paths (3), the $k + 1$ support points $\vec{V}_i$ of the Beta-spline curve are collected in an n-vector γ where n is again $2(k + 1)$ in two dimensions:

$$\gamma = (\vec{V}_0, \vec{V}_1, \cdots, \vec{V}_k). \tag{15}$$

The travel time along a Beta-spline can be calculated as accurately as desired, again with the trapezoidal rule:

$$T(\gamma) = \sum_{i=1}^{k} \sum_{j=1}^{m} \frac{1}{2}\left(\frac{1}{c_{ij}} + \frac{1}{c_{i,j-1}}\right) || \vec{Q}_i(u_j) - \vec{Q}_i(u_{j-1}) || , \tag{16}$$

where $\vec{Q}_i(u)$, $(i = 1, \cdots, k, 0 \le u \le 1)$ is a point on the ith curve segment, m is the number of points $\vec{Q}_i(u_j)$ to be integrated over on that segment and $|| \, . \, ||$ is the Euclidian norm. c_{ij} is the seismic velocity at $\vec{Q}_i(u_j)$. An appropriate choice for u_j is $u_j = \frac{j}{m}$.

It is also possible to invoke higher-order schemes for integration like Simpson's rule. The accuracy of the integration (16), considered as the inverse of the error, is $O(m^2 n^2)$ and can be increased as much as is desired by increasing m, without changing the number of support points. The price for the more accurate integration is $O(mk)$ or $O(mn)$ operations for one travel time calculation.

The gradient of the travel time function can be calculated numerically as before by moving each successive support point $\vec{V}_i$ in each coordinate direction over a fixed increment and recomputing the travel time only along the four changed segments. However, the convergence of the bending is strongly dependent on the definition of the increment. An alternative is the formal differentiation of the travel time along a Beta-spline curve with respect to one support point (see Appendix B, 'The gradient of the travel time along a Beta-spline'). Such a differentiation is more stable because it can be expressed as an integral along the four segments over quantities that are known analytically. The only remaining differentiation is the calculation of the gradient of the velocity field which is singular only at discontinuities. In the next section it will be shown that this singularity causes no problems.

Table 1.

increment	iterations	result	error
10.0	8	96.2483	0.0062%
1.0	9	96.2444	0.0021%
0.1	8	96.2444	0.0021%
0.01	7	96.2444	0.0021%
0.001	10	96.2449	0.0026%
0.0001	14	96.2597	0.0180%
0.00001	61	96.7315	0.5100%
formal diff.	10	96.2443	0.0020%
exact value		96.2424	

Table 1 shows the travel times in the velocity field $c(x,z) = 1.0 + 0.01z$ along a Beta-spline with minimal travel time as found with the Conjugate Gradients bending where the gradient is computed with numerical differentiation with a fixed increment $\delta x = \delta z$ along the ray and with the formal differentiation, described above and in Appendix B. In all cases the initial guess path consists of 11 equidistant points from (0,0) to (100,100) ($k = 10$) and there are 10 integration points on each segment ($m = 10$). The computation time per iteration is roughly the same for all results, namely about 1.2 seconds. This example shows that the numerical differentiation is stable for a range of increments from $\delta x = \delta z = 0.001$ to 10.0 and that the formal differentiation amounts to the same as numerical differentiation with an optimal choice for the increment.

In any case, the number of arithmetic operations for finding one gradient vector is $O(mn)$,

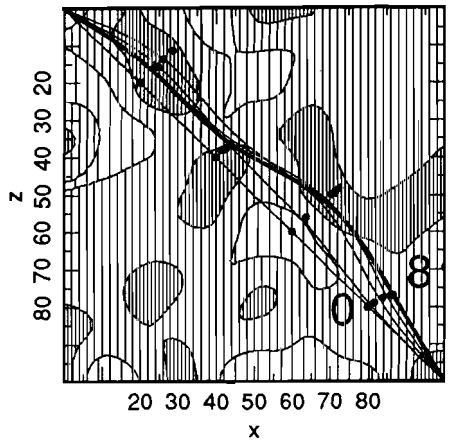

Figure 7. Conjugate Gradients bending of Beta-splines in the velocity field of Figure 3 ($k = 5$, $m = 10$). After 5.9 seconds, 8 iterations, 112 travel time function evaluations and using a tolerance equal to the machine precision the travel time is 89.0627 (no analytic solution known, but $k = 25$ and $m = 50$, which can be considered as much more accurate, gives 88.9772).

thanks to the local control property. Following the rule of thumb that the minimisation of a function requires $O(n^2)$ evaluations of it, one can estimate that the total number of operations to find the minimum ray path consisting of $k+1$ support points and each segment interpolated with a Beta-spline curve evaluated at m points is $O(mk^3)$ or $O(mn^3)$. This can be much less than the minimisation with polygonal paths because a much smaller number of parameters n can be chosen. In practice, a minimisation of the trapezoidal travel time along polygonal paths can be done much more efficiently with the same accuracy or much more accurate with the same computational effort. One can divide n by a factor α and multiply m with α so that the accuracy of the trapezoidal travel time (16) is the same, as $m \times n$ is the same, but the minimisation is α^2 times more efficient in the latter case. However, it must be kept in mind that the number of support points n must be large enough for the Beta-spline curve to accommodate the variations in the velocity field. When n is too small, the continuity constraints are too strict; the travel time accuracy along candidate paths may be acceptable for a large m but the candidate paths do not have enough degrees of freedom to approach the minimum travel time path.

The accuracy and efficiency of minimisation with polygonal paths and Beta-splines are summarised in Table 2.

The advantages of Beta-splines can be illustrated with the example of Figure 2. An initial guess path consisting of $k+1$ points, equally distributed from (0,0) to (100,100), is bent towards the minimum path in the linear velocity field $c(x,z) = 1.0 + 0.01z$, each time using the machine precision as a stopping criterion. The results are given in Table 3.

The first calculation of the travel time from (0,0) to (100,100) with $k = 64$ and $m = 1$ has been compared to the bending method of Um and Thurber (1987). In both cases, no refining as described by Um and Thurber has been applied. The second row of Table 3 shows that the method of Um and Thurber needs an excessive number of iterations to reach only a poor result. The third calculation with $k = 8$ and $m = 8$ shows an enormous gain in

Table 2.

	computation time	accuracy
polygonal paths	$O(n^3)$	$O(n^2)$
Beta-splines	$O(mn^3)$	$O(m^2n^2)$

Table 3.

Velocity field: $c(x,z) = 1.0 + 0.01z$	Initial guess: $k+1$ equidistant points from (0,0) to (100,100)					
	k	m	iterations	function eval.	comp.time	result
1. polygonal path ($\beta_1 = \beta_2 = 0$)	64	1	67	795	58.0	96.2463
2. idem, Um and Thurber-method	64	1	200	-	75.1	96.8554
3. Beta-spline ($\beta_1 = 1, \beta_2 = 0$)	8	8	6	82	5.3	96.2464
4. Beta-spline ($\beta_1 = 1, \beta_2 = 0$)	32	8	17	234	59.1	96.2429
5. Beta-spline ($\beta_1 = 1, \beta_2 = 0$)	5	10	4	57	2.9	96.2514
exact value						96.2424

efficiency compared to the first calculation with polygonal paths of 65 points while preserving the precision. The fourth calculation with $k = 32$ and $m = 8$ is the other way around: with the same computation time as in the first calculation, one obtains a much higher accuracy. After some experience it appears that, in this example, $k = 5$ and $m = 10$ is the most efficient choice to get an acceptable error of 1 part in 10^4.

Figure 6 shows the Conjugate Gradients bending of Beta-splines in the concave slowness field of Figure 5. As a result of the interpolation between the support points, there is no 'bridging' effect like with the bending of polygonal paths although a more accurate integration still results in a larger travel time. Also, the minimal travel time path is much smoother with the same number of parameters, thanks to the Beta-spline representation. Figure 7 shows the solution of the bending of a Beta-spline in the velocity field of Figure 3.

§4 Discontinuities

The occurrence of discontinuities in the velocity field is a problem with classical bending methods (Um and Thurber, 1986; Thurber and Ellsworth, 1980; Prothero *et al.*, 1988).

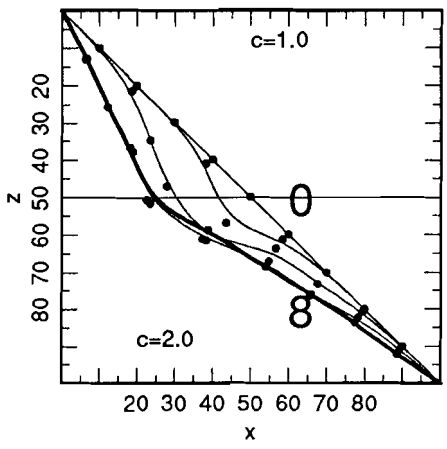

Figure 8. Conjugate Gradients bending of Beta-splines ($k = 10$, $m = 5$) in a discontinuous velocity field ($c = 1.0$ for $z \leq 50.0$, $c = 2.0$ for $z > 50.0$). After 6.1 *seconds,* 8 *iterations,* 130 *travel time function evaluations and using a tolerance equal to the machine precision the travel time is* 100.897 *(exact value obtained from bisection:* 100.941*).*

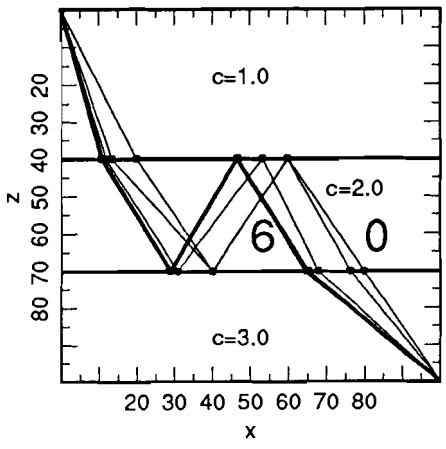

Figure 9. Constrained Conjugate Gradients bending of Beta-splines ($\beta_1 = \beta_2 = 0$, $k = 5$, $m = 1$) in a discontinuous velocity field ($c = 1.0$ for $z \leq 40.0$, $c = 2.0$ for $40.0 < z \leq 70.0$ and $c = 3.0$ for $70.0 < z$). After 0.08 *seconds,* 6 *iterations,* 181 *travel time function evaluations and using a tolerance equal to the machine precision the travel time is* 109.326 *(equal to exact value obtained from bisection).*

Even without modifications the Conjugate Gradients bending of Beta-splines has no major problems with discontinuities as is shown in Figure 8. The minimal travel time is found with an error smaller than 0.1%. However, the corresponding ray path does not refract at the discontinuity although three coincident support points could realise this. It is probably due to the computational error in the numerical integration or differentiation that such a configuration of support points does not occur.

Discontinuities can be handled more correctly by posing constraints on the paths for which the travel time is to be minimised. The candidate paths are Beta-spline curves that behave as differentiable curves in smooth velocity regions, like in the previous section, but with threefold support points that are constrained to lie on the interface. In this way one can calculate different phases on the seismogram like reflections, multiples or reverberations.

The constraints can be easily incorporated by inserting an extra transformation between the support points γ and the travel time $T(\gamma)$ along the path that they define. Suppose that there is one interface in the model, given by a relation $z = I(x)$ or $x = I_1(u)$ and $z = I_2(u)$ with u a variable parameter. γ is then transformed as follows:

$$\gamma' = (x'_0, z'_0, x'_1, z'_1, \cdots, x'_k, z'_k) \tag{17}$$

where $x'_i = x_i$ and $z'_i = z_i$ when i is a free support point and

$$x'_i = x_i, \quad z'_i = I(x_i) \tag{18}$$

or

$$x'_i = I_1(f(x_i)), \quad z'_i = I_2(f(x_i)), \tag{19}$$

when i is to lie on the interface. $f(x_i)$ is a function to transform the x-coordinate of the support point into a suitable parameter u that, in turn, produces coordinates of a point on the interface. z_i has become a dummy variable that plays no role in the minimisation process.

Figure 9 shows the constrained Conjugate Gradients bending of a Beta-spline in a three layered model.

§5 Higher-order terms

Although the Conjugate Gradients bending has no problems with initial guess paths that are far away from a minimal travel time path (see Figure 4), information from a higher-order expansion of the travel time functional may provide a faster convergence. Consider, for instance, the calculations of Figure 2 where a relatively large number of iterations was necessary to find the ray path in a rather simple velocity field. It was argued earlier (see Table 3) that such a slow convergence is caused by a ray path with too many points and the neglect of the velocity variations between the points. Yet, it can be expected that taking into account the second derivatives of the travel time function enhances the convergence, even when Beta-splines are used. This follows from the observation that the minimum of a quadratic function (9) can be found at once by inversion of the Hessian, the matrix of second derivatives of the travel time. A non-quadratic function can be minimised in an iterative procedure by calculating or estimating the inverse Hessian and using it for finding a better approximation of the ray path. Such procedures are described extensively in Stoer and Bulirsch (1980).

In the case that the Hessian is calculated the minimisation of a function of n variables indeed requires $O(n^3)$ arithmetic operations for each iteration. For an almost quadratic function, there are relatively few iterations needed so the total number of operations is again $O(n^3)$. In the case that the inverse Hessian is estimated and updated each iteration, n iterations are needed, each one requiring $O(n^2)$ operations (see Press *et al.*, 1986). Therefore, no profit is necessarily to be expected when using second order terms instead of only first order terms in the travel time expansion.

Another problem is the numerical differentiation for calculating second derivatives which can be very unstable, causing errors in the computed Hessian. These errors can be catastrophic when the Hessian, which is usually ill-conditioned, is inverted. In such situations, which frequently occur in complicated velocity models, the convergence is destroyed completely.

It is possible, however, to simplify the expressions for the second derivatives using some assumptions with regard to the derivatives of the slowness field along the ray and the ray itself. Appendix C gives a derivation of an expression, equation (C25), that can be used as a search direction in the Conjugate Gradients bending. It is a numerically efficient and stable approximation of the direction that would be obtained from the inversion of the Hessian. It gives a much better direction for the line minimisation, at least at the first iterations of the bending. In the vicinity of the ray path it is difficult to compute numerically so that the first order search is to be preferred in further iterations. Table 4 is an extension of the calculations of Figure 2 and Table 3. Starting with a Beta-spline consisting of 6 equidistant points from (0,0) to (100,100) and 10 interpolated points on each segment ($k = 5$, $m = 10$) in the velocity field $c(x,z) = 1.0 + 0.01z$, the Conjugate Gradients bending gives a travel time of 96.2514 in 5 iterations when the gradient is calculated with the formal differentiation derived in Appendix B (first column). The second order search direction introduced above gives almost the same value, 96.2510, in the same number of iterations but with a much faster convergence in the first iterations. The successive iterations of the first and second order search bending are shown in Figure 10.

The problem of the efficiency of the second order search is still a subject of research.

Table 4.

iteration	1st order search	2nd order search
0	98.0287	98.0287
1	96.5979	96.2614
2	96.2576	96.2523
3	96.2514	96.2510
4	96.2514	96.2510
5	96.2514	96.2510
exact value	96.2424	96.2424

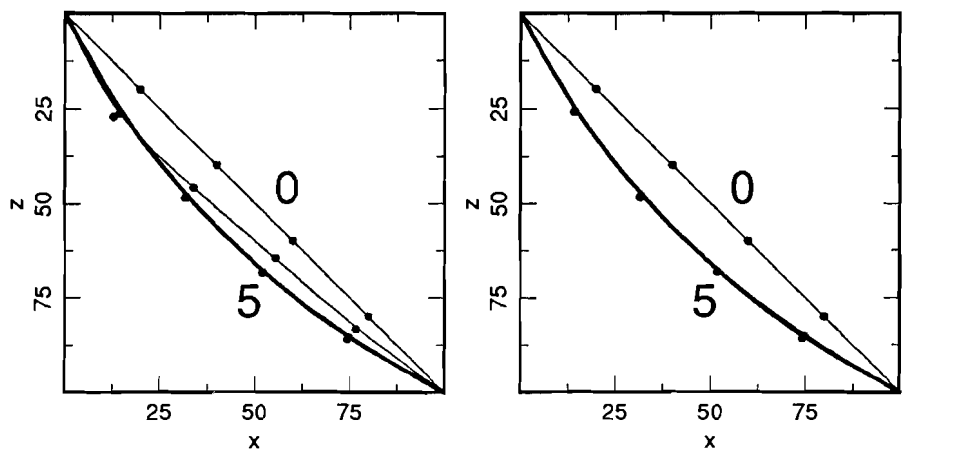

Figure 10. Bending of Beta-spline curves ($\beta_1 = 1$, $\beta_2 = 0$, $k = 5$, $m = 10$) *with first-order search direction (left) and second-order search direction (right). The velocity field is the same as in Figure 2,* $c(x,z) = 1.0 + 0.01z$. *The travel times of the successive iterations are tabulated in Table 4.*

§6 Bending in a spherical Earth

In this section the Conjugate Gradients bending is applied to a spherical Earth with discontinuities. The computations were done for a two dimensional slice through the Earth; three dimensional computations would involve a gradient of larger dimension but are not expected to give significantly different results. The primary aim was to see whether the spherical geometry, with discontinuities, poses any special problems and to investigate the trade-off between precision and computing time.

The curves that are used are the cubic B-spline variant of Beta-splines ($\beta_1 = 1$, $\beta_2 = 0$). Their support points or nodes are specified by spherical coordinates r and θ. To investigate the precision, rays are traced in a spherically symmetric Earth (the PEMC model of Dziewonski, Hales and Lapwood, 1975) for which the travel times are computed with a precision of 0.01 seconds. The initial guess rays are obtained by randomly perturbing the exact ray both in the radial and in the latitudinal (θ) direction. The perturbations Δr and $\Delta\theta$ correlate from ray node to ray node according to the following formula:

$$\Delta r_i = 0.7\,\Delta r_{i-1} + 0.3\,PERT \times (RAND - 0.5) \qquad (1)$$

where *PERT* determines the rms amplitude of the perturbation (which is about 0.16 × *PERT*) and *RAND* is the pseudo random number with uniform distribution on [0,1] as computed by a Fortran library function. An identical algorithm determines the perturbation $\Delta\theta_i$. Figure 11 shows the first 100 values of Δr_i for *PERT* = 100 km. The nodes obtained by adding the perturbations to the coordinates of the initial ray are used as the support points for the Beta-spline interpolation in the first iteration. Some experience showed that a step size of h = 20 km is the optimal choice for the numerical computation of the gradient

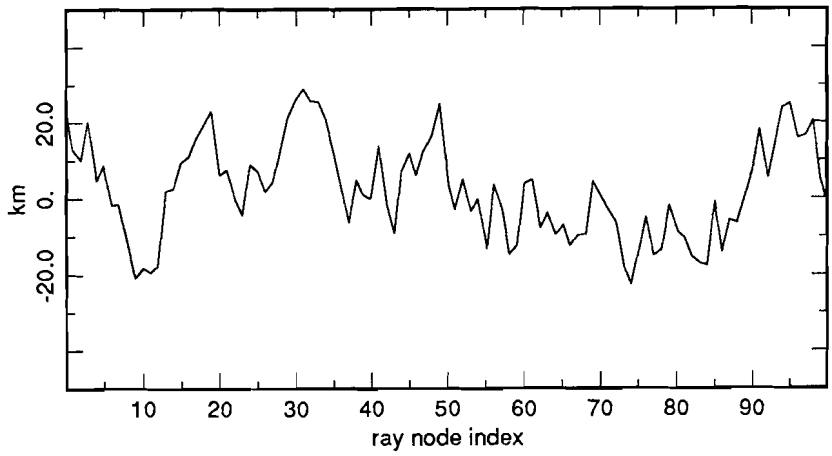

Figure 11. Perturbations of the first 100 *nodes of the exact ray, used to produce an initial guess for ray bending.*

with the central difference formula:

$$\frac{\partial T}{\partial r_i} \approx \frac{T(\cdots, r_i + h, \cdots) - T(\cdots, r_i - h, \cdots)}{2h}$$

$$\frac{1}{r}\frac{\partial T}{\partial \theta_i} \approx \frac{T(\cdots, \theta_i + h/r_i, \cdots) - T(\cdots, \theta_i - h/r_i, \cdots)}{2h}.$$

Discontinuities were handled by including a threefold node on the discontinuity, ensuring that the interpolated curve actually passes through the point. Such interface points are only perturbed in the θ-direction. Since a small shift along a discontinuity with a velocity contrast gives a much larger change in T than a similar shift of a node within a continuous velocity field, a reduced h (of 0.2 km) was used for the computation of $\frac{1}{r}\frac{\partial T}{\partial \theta}$ on interface points.

Computations were performed for 6 values of the ray parameter p: 5,7,...,10 s/deg. With these slownesses rays arrive between 21° and 86° epicentral distance, with turning point depths ranging from 650 to more than 2500 km and travel times ranging between 394 and 763 seconds. The main results are summarised in Figure 12.

The decrease in travel time in the results of two subsequent line minimisations was used as a stopping criterion. Whenever this decrease was less than 0.01 sec the iterations were stopped. The resulting travel time was in all cases within 0.17 sec of the exact T with the average over all 6 slownesses close to 0.1 sec. The error grows slightly as the initial ray is perturbed further from the correct ray (Figure 12 right).

The computation time is of the order of 10 seconds per ray if interface nodes in the crust and upper mantle are perturbed from the initial ray with equal rms perturbations (Figure 12 left, black circles). It is more realistic to assume that the ray near the endpoints is quite

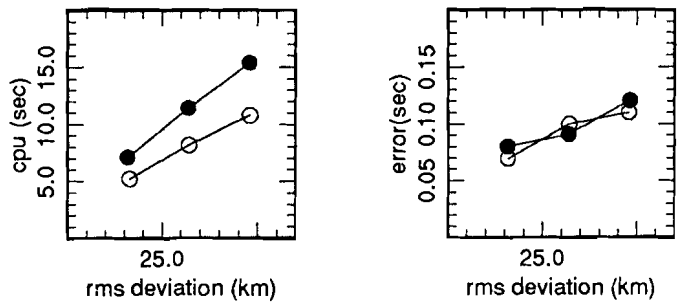

Figure 12. Computation time (left) and travel time error (right) in the final solutions of the ray bending for different deviations of the initial guess from the exact ray. Open circles refer to the case of unperturbed points at interfaces, black circles represent the calculations with perturbations on all points (see text).

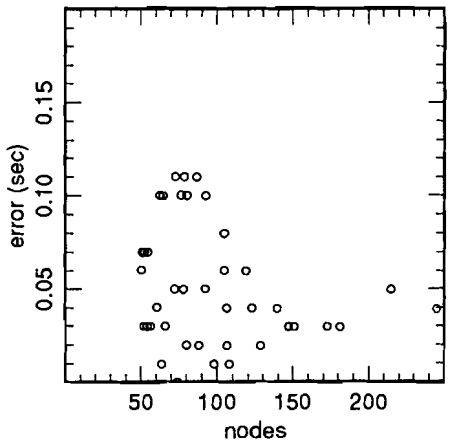

Figure 13. Travel time error of the final solutions of the ray bending as a function of the number of nodes along the path.

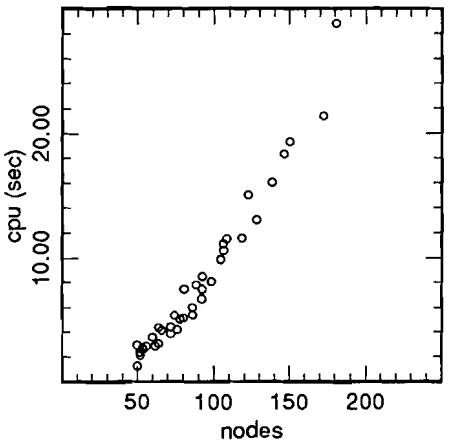

Figure 14. The computation time of the ray bending as a function of the number of nodes along the path.

close to the initial ray. This situation is approximated by leaving the interface points unperturbed, in which case the computation time decreases by 2-5 seconds (Figure 12 left, open circles) while the error is the same (Figure 12 right). The computation time grows roughly linearly with the rms amplitude of the deviation of the initial ray.

In these computations a node spacing on the rays was used of about 100 km. The computation of the gradient at interface points becomes numerically unstable if the distance to the next node is much closer than the average spacing but a simple removal of such nodes was sufficient to improve the accuracy.

Additional characteristics of the algorithm were tested using different node spacings and a constant *PERT* of 100 km. Figure 13 shows that the node spacing is an important variable in determining the final error. The few errors larger than or equal to 0.1 sec all belong to long rays with a small ray parameter and few (<100) nodes. In view of the rather conservative stopping criterion (0.01 sec), it can be inferred that it is the spline interpolation that is not accurate enough to shape the ray if there are not enough nodes. This inference was confirmed by letting one of these computations go on until the decrease in T was below the machine precision, which did not improve the error. In laterally heterogeneous Earth models the node spacing should match the wavelength of the heterogeneities to obtain an acceptable precision in T. Figure 14 shows how the computation time increases as a function of the number of nodes. The behaviour is nonlinear, especially when only few nodes are used and the limited precision is reached within a few iterations only.

Plotting the computed rays and the exact rays shows that the largest errors are near the turning point. The maximum ray location error for a bad initial ray (PERT=300 km, p=8 s/deg) was about 20 km.

Conclusions

The two most important points presented in this chapter are the use of the Conjugate Gradients method for ray bending and the representation of the ray paths by Beta-splines. Minimising the travel time along a curve is to be preferred to fitting the ray equation because it is more stable and cannot diverge. The use of information about the derivative of the travel time along a curve with respect to perturbations of it is suggested by Fermat's principle. Therefore, the Conjugate Gradients method for minimisation using derivative information is suitable because it requires the storage of only four vectors of the size of the storage of the ray itself and it employs the structure of the travel time function. For the first few iterations it can be advantageous to use a pseudo second order search direction, obtained from an approximation of the inversion of the Hessian of the travel time. The points of the ray path should be interpolated during the minimisation process, first because it avoids inaccuracies in concave slowness regions and secondly because interpolation results in both a considerably higher accuracy and efficiency. Beta-splines are particularly suitable as an interpolation scheme because they have several properties, like the local control and convex hull properties, that are in favour of the ray bending and because they provide extra control on the curves by means of the shape parameters. Ray bending in the presence of discontinuities in the velocity field can be done by transforming the location parameters of a point of a ray in such a way that it is forced to lie on the discontinuity. There are no restrictions to the initial guess path that can be chosen because divergence is impossible. However, it is not clear whether the resulting minimal travel time path is a global minimal path or only a local minimal path.

Appendix A, Beta-splines

This appendix summarises the theory of Beta-splines; more details can be found in Barsky (1988). A Beta-spline curve is a curve consisting of segments that are joined together continuously or with a continuous derivative and that is determined by two shape parameters. A point on a Beta-spline curve segment $\vec{Q}_i(u)$ can be regarded as a weighted

average of four support points $\vec{V}_{i-2}$, $\vec{V}_{i-1}$, $\vec{V}_i$ and $\vec{V}_{i+1}$:

$$\vec{Q}_i(u) = \sum_{r=-2}^{1} b_r(\beta_1, \beta_2, u)\vec{V}_{i+r}, \quad \text{for } 0 \le u \le 1 . \tag{A1}$$

The four weighting factors $b_r(\beta_1, \beta_2, u)$ are cubic polynomials in u with coefficients determined by β_1 and β_2:

$$b_r(\beta_1, \beta_1, u) = \sum_{g=0}^{3} c_{gr}(\beta_1, \beta_2)u^g, \quad \text{for } 0 \le u \le 1 \text{ and } r = -2, \cdots, 1 \tag{A2}$$

These coefficients are chosen so that the joint between the i^{th} and $(i+1)^{st}$ curve segments satisfies continuity conditions:

$$\vec{Q}_{i+1}(0) = \vec{Q}_i(1) \tag{A3}$$

$$\frac{d\vec{Q}_{i+1}}{du}(0) = \beta_1 \frac{d\vec{Q}_i}{du}(1)$$

$$\frac{d^2\vec{Q}_{i+1}}{du^2}(0) = \beta_1^2 \frac{d^2\vec{Q}_i}{du^2}(1) + \beta_2 \frac{d\vec{Q}_i}{du}(1)$$

By imposing these conditions β_1 and β_2 come out as shape parameters because they control the curve in a global manner when the support points are fixed. The parameter β_1 measures the discontinuity of $\frac{d\vec{Q}}{du}$ at a joint: $\beta_1 = 1$ for C^1 curves; the parameter β_2 determines the tension of the curve: for positive β_2 the curve is attracted towards its support points, for

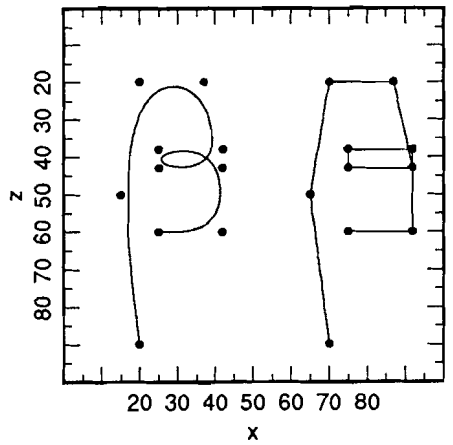

Figure 15. Beta-spline curves with $\beta_1 = 1$, $\beta_2 = 0$ (left) and $\beta_1 = \beta_2 = 0$ (right). The end points are interpolated automatically.

negative β_2 it is pushed away. $\beta_1 = 1$ and $\beta_2 = 0$ indicates continuity of first and second derivatives so produces the smoothest curves. The coefficients $c_{gr}(\beta_1, \beta_2)$ obtained by imposing the previous conditions are:

$$\begin{array}{llll} c_{0,-2}=\dfrac{2\beta_1^3}{\delta} & c_{0,-1}=\dfrac{4\beta_1^2+4\beta_1+\beta_2}{\delta} & c_{0,0}=\dfrac{2}{\delta} & c_{0,1}=0 \\ c_{1,-2}=-\dfrac{6\beta_1^3}{\delta} & c_{1,-1}=\dfrac{6\beta_1(\beta_1^2-1)}{\delta} & c_{1,0}=\dfrac{6\beta_1}{\delta} & c_{1,1}=0 \\ c_{2,-2}=\dfrac{6\beta_1^3}{\delta} & c_{2,-1}=3\dfrac{-2\beta_1^3-2\beta_1^2-\beta_2}{\delta} & c_{2,0}=3\dfrac{2\beta_1^2+\beta_2}{\delta} & c_{2,1}=0 \\ c_{3,-2}=-\dfrac{2\beta_1^3}{\delta} & c_{3,-1}=2\dfrac{\beta_1^3+\beta_1^2+\beta_1+\beta_2}{\delta} & c_{3,0}=-2\dfrac{\beta_1^2+\beta_1+\beta_2+1}{\delta} & c_{3,1}=\dfrac{2}{\delta} \end{array} \tag{A4}$$

where

$$\delta = 2\beta_1^3 + 4\beta_1^2 + 4\beta_1 + \beta_2 + 2 .$$

The following properties and advantages with regard to other curve representations can be summed up:

1. Parametric representation. This allows a curve to be multivalued in each coordinate direction without causing ambiguity. In practice this means that loops are possible.

2. Convex hull property. The weight functions $b_r(\beta_1, \beta_2, u)$ satisfy:

$$\sum_{r=-2}^{1} b_r(\beta_1, \beta_2, u) = 1, \quad \text{for all } u. \tag{A5}$$

The implication of this fact is that $\vec{Q}_i(u)$ lies in the convex hull of the support points $\vec{V}_{i-2}$, $\vec{V}_{i-1}$, $\vec{V}_i$ and $\vec{V}_{i+1}$. This means that unexpected and undesired oscillations or wanderings are impossible. One can easily control the curve by its support points and its shape parameters. This also means that interpolation can be forced by simply setting $\vec{V}_{i-1} = \vec{V}_i = \vec{V}_{i+1}$. The tangents are then specified by $\vec{V}_{i-1} - \vec{V}_{i-2}$ and $\vec{V}_{i+2} - \vec{V}_{i+1}$. The endpoints $\vec{V}_0$ and $\vec{V}_n$ can be interpolated automatically by projecting indices -1, -2 and $n + 1$ on 0 and n respectively.

3. Local control. Shifting one support point $\vec{V}_i$ causes only changes to the curve segments $\vec{Q}_{i-1}(u)$, $\vec{Q}_i(u)$, $\vec{Q}_{i+1}(u)$ and $\vec{Q}_{i+2}(u)$. This provides an easy way to recompute the whole curve if one point is perturbed. Therefore, $\vec{Q}_i(u)$ can be calculated very efficiently, namely in $O(1)$ time, regardless the number of points.

4. Application of tension to the curve. By varying β_2 one can attract or repel the curve from the support points, each obeying the convex hull property (at least for not too negative β_2). The original polygonal path defining the Beta-spline curve is found back for $\beta_1 = \beta_2 = 0$. The concept of shape parameters can be extended to continuously varying shape parameters along the curve.

5. Generalization to 3D. There is no restriction to the dimension of the space in which the curve is drawn; exactly the same formulation applies. Properties 1 to 4 remain valuable, also in three dimensions.

Most of these properties are illustrated by Figure 15: at the left side a Beta-spline with

$\beta_1 = 1$ and $\beta_2 = 0$ and at the right side with $\beta_1 = 0$ and $\beta_2 = 0$; the endpoints are interpolated.

Appendix B, The gradient of the travel time along a Beta-spline

The change of travel time along a Beta-spline $\vec{Q}(u)$ due to change of one support point $\vec{V}_i$ is

$$\frac{\partial T}{\partial \vec{V}_i} = \frac{\partial}{\partial \vec{V}_i} \int_S^R \frac{ds}{c}$$

$$= \frac{\partial}{\partial \vec{V}_i} \int_S^R \frac{|| \vec{Q}'(u) ||}{c(\vec{Q}(u))} du$$

$$= \int_S^R \frac{\partial}{\partial \vec{V}_i} \left(\frac{|| \vec{Q}'(u) ||}{c(\vec{Q}(u))} \right) du$$

$$= \int_S^R \left(\frac{1}{c(\vec{Q}(u))} \frac{\partial || \vec{Q}'(u) ||}{\partial \vec{V}_i} + || \vec{Q}'(u) || \frac{\partial}{\partial \vec{V}_i} \frac{1}{c(\vec{Q}(u))} \right) du \,. \qquad \text{(B1)}$$

The prime $'$ denotes the derivative with respect to u. The first term of (B1) accounts for the extension of the ray path, the second term for the change in velocity along the path. The differentiation of the first term is carried out as follows:

$$\frac{\partial || \vec{Q}'(u) ||}{\partial \vec{V}_i} = \frac{\vec{Q}'(u)}{|| \vec{Q}'(u) ||} \frac{\partial \vec{Q}'(u)}{\partial \vec{V}_i}$$

where $\frac{\partial \vec{Q}'(u)}{\partial \vec{V}_i}$ is a 2×2 (or 3×3 in three dimensions) identity matrix which is multiplied with a scalar factor equal to the derivative with respect of u of one of the polynomials $b_r(u)$. If the Beta-spline $\vec{Q}(u)$ is considered as segments $\vec{Q}_j(u)$, then

$$\frac{\partial \vec{Q}'_j(u)}{\partial \vec{V}_i} = b'_{i-j}(u)$$

for $j = i - 1$ to $i + 2$ and otherwise

$$\frac{\partial \vec{Q}'_j(u)}{\partial \vec{V}_i} = 0 \,.$$

The second term of (B1) can be differentiated as follows

$$\frac{\partial}{\partial \vec{V}_i} \frac{1}{c(\vec{Q}(u))} = \nabla \left(\frac{1}{c} \right) (\vec{Q}(u)) \frac{\partial \vec{Q}(u)}{\partial \vec{V}_i} \,,$$

where $\frac{\partial \vec{Q}(u)}{\partial \vec{V}_i}$ is a 2×2 identity matrix again, this time multiplied with one of the

polynomials $b_r(u)$ themselves:

$$\frac{\partial \vec{Q}_j(u)}{\partial \vec{V}_i} = b_{i-j}(u)$$

for $j = i - 1$ to $i + 2$ and otherwise

$$\frac{\partial \vec{Q}_j(u)}{\partial \vec{V}_i} = 0 .$$

The result is:

$$\frac{\partial T}{\partial \vec{V}_i} = \tag{B2}$$

$$= \sum_{j=i-1}^{i+2} \int_0^1 \left(\frac{1}{c(\vec{Q}_j(u))} \frac{\vec{Q}'_j(u)}{\| \vec{Q}'_j(u) \|} b'_{i-j}(u) + \| \vec{Q}'_j(u) \| \nabla\left(\frac{1}{c}\right)(\vec{Q}_j(u)) b_{i-j}(u)\right) du.$$

The usual integration by parts of the first term of (B2) is omitted here in order to avoid second derivatives of the ray path.

Appendix C, Higher-order terms in the travel time expansion

The purpose of this appendix is to derive an expression for a search direction, using a second order expansion in the travel time along a curve with respect to perturbations on it. The travel time along a curve $\vec{r}$, parametrised by the arc length s, in a slowness field $\sigma(\vec{r}) = \frac{1}{c(\vec{r})}$ is:

$$T = \int \sigma(\vec{r}) ds. \tag{C1}$$

Let an initial guess path joining the source and the receiver be given by $\vec{r}_0(s_0)$. The source is located at $\vec{r}_0(0)$ while the length of the path is S_0. Let $\vec{\xi}(s_0)$ be a perturbation perpendicular on $\vec{r}(s_0)$ in such a way that

$$(\vec{\xi}, \dot{\vec{r}}_0) = 0 \tag{C2}$$

and

$$\vec{\xi}(0) = \vec{\xi}(S_0) = 0. \tag{C3}$$

The dot $\dot{}$ denotes the derivative with respect to s_0. This perturbation has two effects on the travel time, namely one due to the change in the arc length and one due to the sampling of a different velocity region. For the arc length this gives to second order

$$\frac{\partial s}{\partial s_0} = 1 + (\dot{\vec{r}}_0, \dot{\vec{\xi}}) + \frac{1}{2}((\dot{\vec{\xi}}, \dot{\vec{\xi}}) - (\dot{\vec{r}}_0, \dot{\vec{\xi}})^2), \tag{C4}$$

and for the slowness

$$\sigma(\vec{r}_0 + \vec{\xi}) = \sigma(\vec{r}_0) + (\vec{\xi}, \nabla\sigma(\vec{r}_0)) + \frac{1}{2}(\vec{\xi}\vec{\xi}:\nabla\nabla\sigma(\vec{r}_0)), \tag{C5}$$

where : stands for the dyadic product. Inserting this into (C1) and (C4) gives

$$T(\vec{\xi}) = T_0 + T_1(\vec{\xi}) + T_2(\vec{\xi}), \tag{C6}$$

where T_n is the travel time perturbation of the n-th order in $\vec{\xi}$:

$$T_0 = \int\sigma(\vec{r}_0)\mathrm{d}s_0, \tag{C7.a}$$

$$T_1 = \int\left\{(\vec{\xi}, \nabla\sigma(\vec{r}_0)) + \sigma(\vec{r}_0)(\dot{\vec{r}}_0, \dot{\vec{\xi}})\right\}\mathrm{d}s_0, \tag{C7.b}$$

$$T_2 = \int\left\{\frac{1}{2}(\vec{\xi}\vec{\xi}:\nabla\nabla\sigma(\vec{r}_0)) + (\dot{\vec{r}}_0, \dot{\vec{\xi}})(\vec{\xi}, \nabla\sigma(\vec{r}_0)) + \frac{1}{2}\sigma(\vec{r}_0)((\dot{\vec{\xi}}, \dot{\vec{\xi}}) - (\dot{\vec{r}}_0, \dot{\vec{\xi}})^2)\right\}\mathrm{d}s_0. \tag{C7.c}$$

Now perturb $\vec{\xi}$ with an extra perturbation $\delta\vec{\xi}$ which is one order of magnitude smaller. The travel time is stationary for a finite curve change $\vec{\xi}(_0)$ when it does not change to first order by the extra small perturbation $\delta\vec{\xi}(_0)$. Applying integration by parts using (C3), one can write the perturbations in T_1 and T_2 due to the perturbation $\delta\vec{\xi}$ as

$$\delta T_1 = \int(\nabla\sigma - \frac{\mathrm{d}}{\mathrm{d}s_0}(\sigma\dot{\vec{r}}_0), \delta\vec{\xi})\mathrm{d}s_0, \tag{C8.a}$$

$$\delta T_2 = \tag{C8.b}$$

$$= \int\left[((\vec{\xi}, \nabla\nabla\sigma) - (\vec{\xi}, \nabla\sigma)\ddot{\vec{r}}_0 + (\dot{\vec{r}}_0, \dot{\vec{\xi}})\nabla\sigma - \sigma\ddot{\vec{\xi}} - (\dot{\vec{r}}_0, \nabla\sigma)\dot{\vec{\xi}} + \sigma(\dot{\vec{r}}_0, \dot{\vec{\xi}})\ddot{\vec{r}}_0), \delta\vec{\xi}\right]\mathrm{d}s_0.$$

It has been used that $(\dot{\vec{r}}_0, \delta\vec{\xi}) = 0$ and $\delta\vec{\xi}(0) = \delta\vec{\xi}(S_0) = 0$.

Note that δT_1 vanishes when $\frac{\mathrm{d}}{\mathrm{d}s_0}(\sigma\dot{\vec{r}}_0) = \nabla\sigma$, that is when $\vec{r}_0(s_0)$ is a true ray. However, $\vec{r}_0(s_0)$ is in general not a ray and a finite deformation $\vec{\xi}(s_0)$ is needed to make the travel time stationary. This is the case when $\delta T_1 + \delta T_2$ does not depend on arbitrary small perturbations $\delta\vec{\xi}(s_0)$ which happens when

$$\sigma\ddot{\vec{\xi}} + (\nabla\sigma, \dot{\vec{r}}_0)\dot{\vec{\xi}} - (\nabla\sigma, \dot{\vec{r}}_0)\dot{\vec{\xi}} - \sigma\ddot{\vec{r}}_0(\dot{\vec{r}}_0, \dot{\vec{\xi}}) + \ddot{\vec{r}}_0(\nabla\sigma, \vec{\xi}) - (\nabla\nabla\sigma, \vec{\xi}) = \nabla\sigma - \sigma\ddot{\vec{r}}_0. \tag{C9}$$

This is a second order differential equation that needs to be solved for $\vec{\xi}$ with the boundary conditions (C3). The right hand side of this expression is the deviation from the ray equation for the reference curve $\vec{r}_0(s_0)$.

Constraining the perturbation to be perpendicular to the reference curve eliminates one degree of freedom and makes the computation more efficient without loss of generality. In order to take full advantage of this it is convenient to transform (C9) to ray-coordinates. Define two unit vectors $\hat{q}_1$ and $\hat{q}_2$ in such a way that $(\dot{\vec{r}}_0, \hat{q}_1, \hat{q}_2)$ form an orthonormal basis. It is assumed here that the unit vectors do not rotate around the reference curve, that is

$$(\dot{\hat{q}}_1, \hat{q}_2) = (\hat{q}_1, \dot{\hat{q}}_2) = 0. \tag{C10}$$

The perturbation $\vec{\xi}$ is perpendicular to the reference curve so that

$$\vec{\xi} = q_1\hat{q}_1 + q_2\hat{q}_2. \tag{C11}$$

In order to convert (C9) to ray coordinates the derivatives of $\hat{q}$ are needed. From the orthogonality $(\dot{\vec{r}}_0, \hat{q}_1) = (\dot{\vec{r}}_0, \hat{q}_2) = (\hat{q}_1, \hat{q}_2) = 0$ and the normalisations $(\vec{r}_0, \vec{r}_0) = (\hat{q}_1, \hat{q}_1) = (\hat{q}_2, \hat{q}_2) = 1$ one derives by direct differentiation that

$$(\dot{\hat{q}}, \dot{\vec{r}}_0) = -(\hat{q}, \ddot{\vec{r}}_0), \tag{C12}$$

$$(\dot{\hat{q}}, \hat{q}) = 0, \tag{C13}$$

$$(\dot{\vec{r}}_0, \ddot{\vec{r}}_0) = 0, \tag{C14}$$

where $\hat{q}$ denotes either $\hat{q}_1$ or $\hat{q}_2$. The projection of $\dot{\hat{q}}_1$ on the basis vectors gives

$$\dot{\hat{q}}_1 = (\dot{\hat{q}}_1, \hat{q}_1)\hat{q}_1 + (\dot{\hat{q}}_1, \hat{q}_2)\hat{q}_2 + (\dot{\hat{q}}_1, \vec{r}_0)\dot{\vec{r}}_0. \tag{C15}$$

With (C10), (C12) and (C13) this leads to

$$\dot{\hat{q}}_1 = -(\ddot{\vec{r}}_0, \hat{q}_1)\dot{\vec{r}}_0. \tag{C16}$$

The differentiation of (C16) and use of (C10) gives

$$\ddot{\hat{q}}_1 = -(\frac{d^3\vec{r}_0}{ds_0^3}, \hat{q}_1)\dot{\vec{r}}_0 - (\ddot{\vec{r}}_0, \hat{q}_1)\ddot{\vec{r}}_0. \tag{C17}$$

Let the perpendicular component of an arbitrary vector $\vec{v}$ be denoted by

$$\vec{v}_\perp \equiv \vec{v} - (\vec{v}, \dot{\vec{r}}_0)\dot{\vec{r}}_0 = (\vec{v}, \hat{q}_1)\hat{q}_1 + (\vec{v}, \hat{q}_2)\hat{q}_2. \tag{C18}$$

By differentiation of (C11) one finds, using (C16) and (C17),

$$\dot{\vec{\xi}} = \dot{q}_1\hat{q}_1 + \dot{q}_2\hat{q}_2 - \left\{q_1(\ddot{\vec{r}}_0, \hat{q}_1) + q_2(\ddot{\vec{r}}_0, \hat{q}_2)\right\}\dot{\vec{r}}_0, \tag{C19.a}$$

$$\ddot{\vec{\xi}}_\perp = \ddot{q}_1\hat{q}_1 + \ddot{q}_2\hat{q}_2 - \left\{q_1(\ddot{\vec{r}}_0, \hat{q}_1) + q_2(\ddot{\vec{r}}_0, \hat{q}_2)\right\}\ddot{\vec{r}}_0. \tag{C19.b}$$

It follows from (C14) that

$$\ddot{\vec{r}}_0 = (\ddot{\vec{r}}_0, \hat{q}_1)\hat{q}_1 + (\ddot{\vec{r}}_0, \hat{q}_2)\hat{q}_2. \tag{C20}$$

Projecting (C9) on the unit vectors $\hat{q}_1$ and $\hat{q}_2$ perpendicular to the reference curve one obtains with (C19) and (C20) that

$$\frac{d}{ds_0}(\sigma\frac{dq_1}{ds_0})\hat{q}_1 + \frac{d}{ds_0}(\sigma\frac{dq_2}{ds_0})\hat{q}_2 - (q_1\hat{q}_1\hat{q}_1:\nabla\nabla\sigma + q_2\hat{q}_1\hat{q}_2:\nabla\nabla\sigma)\hat{q}_1 \tag{C21}$$

$$-(q_1\hat{q}_1\hat{q}_2{:}\nabla\nabla\sigma + q_2\hat{q}_2\hat{q}_2{:}\nabla\nabla\sigma)\hat{q}_2 - ((\nabla\sigma)\downarrow\ddot{\vec{r}}_0 + \ddot{\vec{r}}_0(\nabla\sigma)\downarrow, q_1\hat{q}_1 + q_2\hat{q}_2) = \nabla\downarrow\sigma - \sigma\ddot{\vec{r}}_0,$$

where it is used that $(\dot{\vec{r}}_0, \nabla f) = \frac{df}{ds_0}$. By projecting this equation on $\hat{q}_1$ and $\hat{q}_2$ respectively, one finds using $\nabla\nabla\sigma - \frac{2}{\sigma}(\nabla\sigma)(\nabla\sigma) = -\sigma^2\nabla\nabla\frac{1}{\sigma}$ and after some algebraic manipulations that

$$\frac{d}{ds_0}(\sigma\frac{dq_i}{ds_0}) + \sigma^2\sum_{j=1}^{2}\hat{q}_i\hat{q}_j{:}\nabla\nabla(\frac{1}{\sigma})q_j + (\frac{1}{\sigma})\sum_{j=1}^{2}(\hat{q}_i\hat{q}_j + \hat{q}_j\hat{q}_i){:}(\nabla\sigma)(\sigma\ddot{\vec{r}}_0 - \nabla\sigma)q_j =$$

$$(\hat{q}_i, \sigma\ddot{\vec{r}}_0 - \nabla\sigma)\,. \tag{C22}$$

This equation constitutes two coupled linear differential equations in q_1 and q_2 that need to be solved subject to the boundary condition $q_i(0)=q_i(S_0)=0$. Note that this expression is different from the expression (13.15) of Aki and Richards (1980). The reason for this difference is that they ignored the change in the path length when the ray is perturbed, that is, they used $\frac{\partial s}{\partial s_0} = 1$ rather than (C4). Furthermore, expression (C22) is in ray coordinates whereas Aki and Richards (1980) consider a three dimensional Cartesian coordinate system.

The ray estimate can be updated by solving (C22) recursively. When the solution has converged to the true ray, the right hand side of (C22) vanishes and the solution does not change anymore. If one applies (C22) recursively, it may be advantageous to solve a simplified version of (C22) rather than the exact equations (C22). When the reference curve is near the true ray, $(\hat{q}, \sigma\ddot{\vec{r}}_0 - \nabla\sigma) \approx 0$. Therefore, it is reasonable to ignore this term in the left hand side of (C22). Furthermore, it may not be possible to compute the second derivative of the velocity $\nabla\nabla(\frac{1}{\sigma})$ with great numerical accuracy. Ignoring these second derivatives gives

$$\frac{d}{ds_0}(\sigma\frac{dq_i}{ds_0}) \approx (\hat{q}_i, \sigma\ddot{\vec{r}}_0 - \nabla\sigma). \tag{C23}$$

Note that in this approximation the components q_1 and q_2 are decoupled. Equation (C23) can be integrated in closed form. One can make a further approximation and ignore the derivative of the slowness in the left hand side of (C23):

$$\ddot{q}_i \approx \frac{1}{\sigma}(\hat{q}_i, \sigma\ddot{\vec{r}}_0 - \nabla\sigma). \tag{C24}$$

This equation can be integrated to give

$$q_i(s) \approx -(1 - \frac{s}{S_0})\int_0^s s'F_i(s')ds' - s\int_s^{S_0}(1 - \frac{s'}{S_0})F_i(s')ds', \tag{C25}$$

where $F_i(s) = \frac{1}{\sigma}(\hat{q}_i, \sigma\ddot{\vec{r}}_0 - \nabla\sigma)$.

CHAPTER 6

A practical comparison of the shortest path method with other methods

Introduction

The shortest path method for seismic ray tracing has some attractive characteristic properties in comparison to other methods. Among its advantages there is the global result: the shortest paths from one source point to all other possible receiver points and the travel times along them, are calculated simultaneously so that complete knowledge of the travel time field is obtained. There is not only a complete coverage of ray paths in regions where classical ray theory is valid but also in shadow zones where no classical ray arrival can be found. The shortest paths always correspond to the first arrivals on the seismogram in some point, whatever physical meaning they may have. This distinguishes the shortest path method from methods that may not necessarily find the first arrival but that allow *a priori* knowledge about the physical meaning of the arrival. However, the complete coverage of ray paths can be less useful and therefore costly when only few receiver points are involved. Another advantage of the shortest path method is robustness: it can handle arbitrarily complicated models and arbitrarily large velocity contrasts, without getting 'stuck'. It always gives an answer that is mathematically and physically meaningful because of Weierstraβ' existence theorem for extremals and Fermat's minimum travel time path principle. Other methods can suffer from numerical instabilities due to the approximate solving of ray equations or from assumptions on the behaviour of the ray paths. In spite of the robustness the discretisation introduced by the network structure causes inaccuracies in the computed results. This discretisation effect can be separated into two parts: errors due to the space discretisation, the sampling of the velocity field on a finite number of points, and errors due to the angle discretisation, the finite number of connections per node. Although the accuracy can be estimated from asymptotic relationships for the two types of discretisation, it is limited by memory space requirements and computation time. Some of these properties interact with each other, thus mixing up the advantageous and disadvantageous effects. In order to investigate the performance of the shortest path method, it should, therefore, be compared with methods that have appeared to be reliable in practical use.

In the practice of exploration seismics the required accuracy in the computed travel times is 0.1%; this percentage is used as a target in this chapter. In contrast to other methods, the computation time to find the travel times of a few ray paths within this precision can be enormous for the shortest path method. On the other hand, the shortest path method has then computed the travel times to all points in space in the same time and with the same accuracy. It may be more efficient to calculate shortest paths on a coarse grid and then perform an additional bending on the selected ray paths for which the shortest paths are guaranteed safe initial guesses. A careful consideration of the purpose of the ray tracing problem is, therefore, necessary.

This chapter compares the shortest path method with three other methods for forward modeling. The first comparison is with industrial ray tracing software, the second with a

finite difference programme for acoustic wave propagation. In the third comparison the additional bending is tested by comparison with analytical solutions (see Chapter 5). In all examples the networks are of the grid type because this type allows for an easy definition of interfaces (see Chapter 3) and links up with the other methods. Grid refinement techniques, that increase the node density in presence of rapid velocity variations, are not taken into account.

During August 1990, the shortest path method was tested at the Koninklijke/Shell Exploratie en Productie Laboratorium (KSEPL) at Rijswijk, The Netherlands, on a complicated geological model (referred to as model CS673) and compared with ray tracing software developed at the KSEPL (referred to as the Shell ray tracer). This chapter contains parts of the report 'The comparison of the shortest path method with industrial ray tracing software' (Moser, 1990). Detailed information about both the model CS673 and the Shell ray tracer is confidential, so a decisive comparison with the shortest path method is not possible. Also, the computation times of Shell software and the shortest path method have not been compared for reasons of confidentiality. However, an attempt has been made to investigate to what extent the results of the shortest path method in an unsophisticated form do agree with the results from the Shell ray tracer. The aim of this investigation was in the first place to test the software for the shortest path method on a realistic, highly demanding structure such as can be expected in exploration seismics. It should be emphasised that there is no precise yardstick to judge the precision of a three dimensional ray tracer in media of arbitrary complexity and it may sometimes be impossible to track down the cause of discrepancies between the two methods, especially where this was hindered by the confidential nature of the Shell software. The discrepancy between the first version of the shortest path algorithm and the Shell ray tracer was as large as 1%. In many cases this difference was simply due to the fact that the two programmes select different rays (the Shell ray tracer does not handle shadow zones or head waves). However, it was found that several aspects of the shortest path method could easily be improved. Other recognised sources of discrepancy are the different ways of describing the location of the interface (which in fact causes the two algorithms to compute two slightly different models) and the inaccuracy of the regularisation procedure for travel times at the Earth's surface, that is applied after the Shell ray tracer.

One aspect that can be improved easily is the forward star definition with a two point trapezoidal calculation of the travel time between two neighbour points (see Chapter 4). Multipoint integration schemes for the calculation of the weights are very costly unless they take advantage of the grid organisation of the network. By invoking sophisticated line integration algorithms in rectangular grids, it is possible to enhance the accuracy without loss of efficiency.

The claim that the shortest path method finds the correct travel times in shadow zones is justified by a comparison with the finite difference programme for acoustic wave propagation of Marquering (1991). The first arrival times on the finite difference seismograms agree within 0.5% with the shortest path travel times.

A final test shows the combination of the shortest path method and additional bending. A complicated three dimensional model, consisting of flat, dipping and curved interfaces, is covered by a relatively coarse grid of nodes for the shortest path calculations which are improved by the bending procedure (see Chapter 5). The additional bending travel times are then compared with travel times obtained from a bending procedure on intersections of

the ray paths with the interfaces in which the travel time is given by analytical formulae.

Because of the variety of methods presented, this chapter has a rather mixed character, with a lot of references to previous Chapters. In the first section the definitions, computational complexity and error characteristics of the shortest path method are shortly summarised with respect to the grid organisation of the networks. §2 gives a summary of the comparison with the Shell ray tracing software. The advanced line integration is given in §3; it is included in this chapter because it applies only to the grid organisation. In section §4 the shortest path method is compared with the finite difference programme. The last section contains the additional bending in a complicated three dimensional model.

§1 Definitions

The networks used in this note are of the grid type. The three numbers n_1, n_2 and n_3 are denoted as an integer three-vector: $\vec{n} = (n_1, n_2, n_3)$. The total number of nodes is $n = n_1 \times n_2 \times n_3$. The grid is equidistant in each coordinate direction with grid spacings dx, dy and dz, denoted as $\vec{d} = (dx, dy, dz)$. When the coordinate ranges are $[x_1, x_2]$, $[y_1, y_2]$ and $[z_1, z_2]$, then $\vec{n}$ and $\vec{d}$ are related by

$$dx = (x_2 - x_1)/(n_1 - 1),\ dy = (y_2 - y_1)/(n_2 - 1),\ dz = (z_2 - z_1)/(n_3 - 1). \qquad (1)$$

For a two dimensional grid in the x and z direction $dy = 0$ and $n_2 = 1$. (See Figure 1). The distance, time, velocity and slowness are given by dimensionless quantities except for section §2, where respectively km, sec, km/sec and sec/km are used. Each node has discrete coordinates (i_1, i_2, i_3) and real coordinates as follows:

$$x = x_1 + (i_1 - 1)dx,\ y = y_1 + (i_2 - 1)dy,\ z = z_1 + (i_3 - 1)dz. \qquad (2)$$

Each node is connected with the nodes in a rectangular neighbourhood of it, its forward star. The neighbourhood is defined by three integers $\vec{f} = (f_1, f_2, f_3)$ in such a way that a

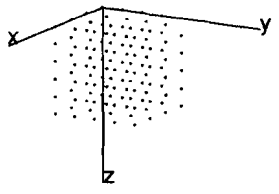

Figure 1. A rectangular grid of nodes with 4 nodes in the x-direction, 5 nodes in the y-direction and 6 nodes in the z-direction ($\vec{n}$ = (4,5,6)).

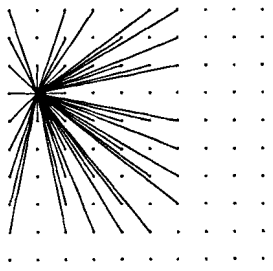

Figure 2. A two dimensional grid with the forward star of one node ($\vec{f}$ = (5,0,5)).

node (i_1, i_2, i_3) is connected with another node (j_1, j_2, j_3) if

$$|i_1 - j_1| \leq f_1, \quad |i_2 - j_2| \leq f_2, \quad |i_3 - j_3| \leq f_3 \tag{3}$$

and if there is no other node on the line segment connecting (i_1, i_2, i_3) and (j_1, j_2, j_3) (Figure 2). The last condition is satisfied when $|i_1 - j_1|$, $|i_2 - j_2|$ and $|i_3 - j_3|$ have no common divisors (see Chapter 3).

The velocity field is sampled on the node locations. The weight of a connection between two nodes i and j is defined in the simplest form as its Euclidian length multiplied with the average of the slownesses at the two node locations:

$$w(i,j) = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2 + (z_i - z_j)^2}\,\frac{1}{2}\left(\frac{1}{c_i} + \frac{1}{c_j}\right), \tag{4}$$

where $w(i,j)$ is the weight, (x_i, y_i, z_i) is the location of node i and c_i is the velocity on that location. The travel time along a path is defined as the sum of the weights of the connections it consists of. This definition (4) corresponds to the trapezoidal rule for numerical integration. It is a rather crude one because it disregards the velocity variations between the nodes, which can lead to errors for large forward stars; on the other hand, it is efficient because it requires only one addition and one multiplication per connection. Section §3 gives a more advanced definition of the weight.

The shortest path algorithm that is used is Dijkstra's (1959) algorithm with the set of tentative travel times ordered as a heap. This algorithm and its computational complexity

$$CC = O(mn \log n), n \to \infty, \tag{5}$$

where m is the number of connections per node, is described in Chapter 2.

The accuracy of the shortest path method as a function of δx, the space discretisation, and $\delta\phi$, the angle discretisation, is derived in Chapter 4.

$$E = \alpha_{20}\delta x^2 + \alpha_{11}\delta x\,\delta\phi + \alpha_{02}\delta\phi^2 \tag{6}$$

plus higher order terms in δx and $\delta\phi$. α_{20}, α_{11} and α_{02} are parameters that depend on the network type and the velocity field but not on δx and $\delta\phi$. In the grid organisation of the network δx is proportional to dx, dy and dz and $\delta\phi$ to $\frac{1}{m}$ or $\frac{1}{f_1 f_2 f_3}$. dx, dy and dz are directly related to the number of nodes and $f_1 f_2 f_3$ to the number of connections per node. The accuracy can thus be fixed by choosing appropriate values for $\vec{n}$ and $\vec{f}$. The price for such a choice can then be estimated from the computational complexity bound.

§2 Comparison with industrial ray tracing software

§2.1 Description of the model CS673 and the Shell ray tracer

In this section the shortest path method is tested on the model CS673, that consists of four linear velocity regions separated by curved interfaces. The calculations have been compared with the Shell ray tracer. Distances are given in kilometers, times in seconds, velocities in kilometers per second. The coordinate ranges are

$$4.0 \leq x \leq 16.0, \tag{7}$$

$$3.0 \leq y \leq 18.0, \tag{8}$$

$$0.0 \leq z \leq 8.0. \tag{9}$$

The velocity functions are

$$c_1(z) = 1.7 + 0.72\,z \tag{10}$$

$$c_2(z) = 2.0 + 0.30\,z \tag{11}$$

$$c_3(z) = 1.4 + 1.50\,z \tag{12}$$

$$c_4(z) = 2.2 + 0.34\,z \tag{13}$$

for the uppermost, second, third and bottom layer respectively.

The model CS673 has been covered by a number of three dimensional rectangular grids with different coordinate ranges and numbers of nodes. The velocity field is sampled at the node locations by selecting the layer to which the node belongs and evaluating the appropriate linear velocity function.

Two source locations were selected, labeled 802 and 807. 802 is a deep source and has coordinates

$$\vec{s}_{802} = (10.0,7.5,6.0) \tag{14}$$

and 807 is shallow:

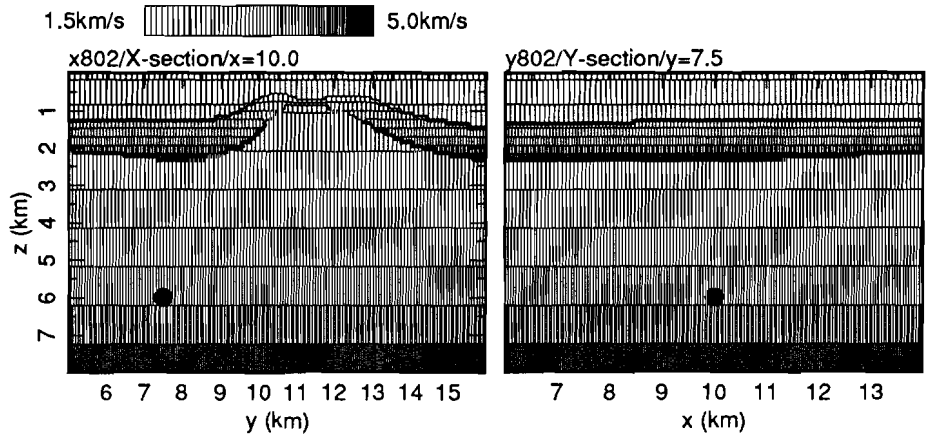

Figure 3. x- and y-sections of the model CS673 at the source point $\vec{s}_{802}$, located at (10.0,7.5,6.0) (black circle).

$$\vec{s}_{807} = (10.0,10.0,2.5). \tag{15}$$

x- and y-sections of the model through these locations are shown in Figures 3 and 4. The first interface, at a depth of 0.5km, is hardly visible due to its small velocity contrast.

The Shell ray tracer calculates ray paths in three dimensional models with linear velocity regions separated by curved interfaces like the model CS673, see Figures 3 and 4. The ray path segments in the linear velocity regions and the travel times along them are calculated exactly from analytic formulae (see Chapter 4, formula (41), or Červený, 1987). The segments are linked together at the interfaces by Snell's law. From both source points, $\vec{s}_{802}$ and $\vec{s}_{807}$, a fan of rays has been shot towards the Earth's surface. The travel times along them are coded with r802 and r807 see respectively. The fans consist of 100×100 rays with initial directions ranging from $-75°$ to $75°$ in x- and y-direction, where $0°$ correspond to a ray directed vertically upward. Reflections and refractions have been discarded, only transmitted ray paths have been taken into account. In most cases this corresponds to the calculation of first arrivals. The Shell ray tracer is classical in the sense that it only finds solutions to the ray equations and does not find diffractions or arrivals in shadow zones. Using the Shell ray tracer it may also prove difficult to provide a regular or even a complete coverage of ray paths. Convergence problems may occur, when a specified configuration of receiver points at the Earth's surface must be reached. In order to avoid these problems, the fans of rays have been shot upward to the Earth's surface, resulting in an irregular grid of endpoints, with large voids or shadow zones (Figure 5). The ray end points have then been regularised onto the rectangular grids. This regularisation has been done by fitting bicubic polynomials to Shell travel times that are in a close neighbourhood of a grid point. The resulting travel times are coded as r802_reg for the r802 times and r807_reg for the r807 times. The loss of travel time accuracy due to the regularisation is 0.1% for $\vec{s}_{802}$ and

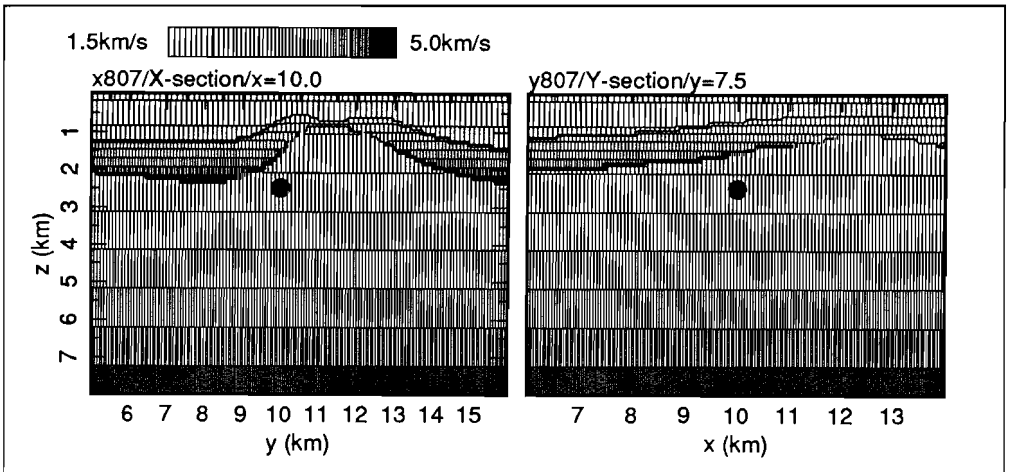

Figure 4. x- and y-sections of the model CS673 at the source point $\vec{s}_{807}$, located at (10.0,10.0,2.5) *(black circle).*

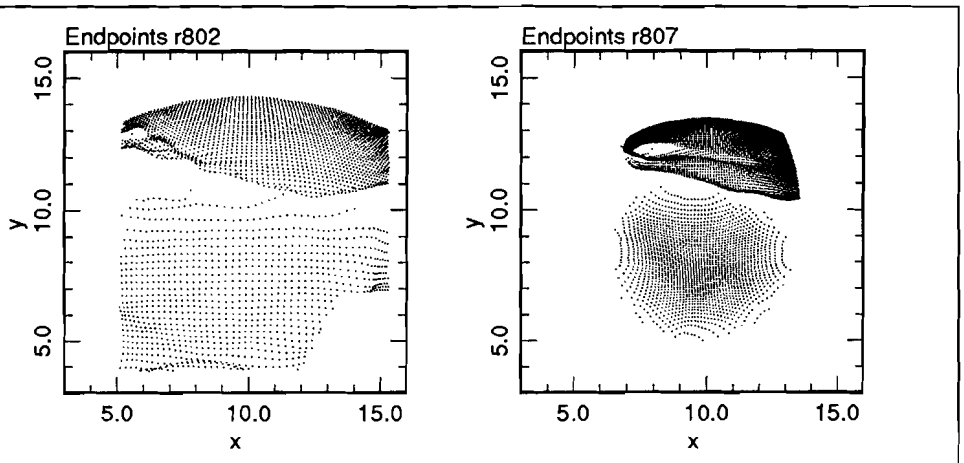

Figure 5. Endpoints of the Shell ray paths at the Earth's surface for both source points.

0.25% for $\vec{s}_{807}$ (Moser, 1990).

§2.2 Tests

The regularised travel times of the Shell ray tracer have been compared with shortest path travel times in two networks with different numbers of nodes and connections per node. The relative difference between the shortest path travel times T_{SPT} and the regularised Shell travel times T_{SR} is defined as

$$\frac{|\, T_{SPT} - T_{SR} \,|}{T_{SPT}}, \tag{16}$$

when T_{SR} exists and ∞ otherwise. For each test three plots are shown: the contours of the shortest path travel times, the contours of the relative differences and their distribution.

The source point 802 has a large depth coordinate compared to the interfaces. The shortest paths lie mainly in the fourth layer and information about the complicated structure between the first and the third interface is summarised in their last parts. A shadow zone is clearly visible at the Earth's surface from this source point. The model CS673 has been covered by a rectangular grid with

$$\vec{n} = (81,111,81) \tag{17}$$

and coordinate ranges

$$x \in [6.0,14.0], \tag{18}$$

$$y \in [5.0,16.0], \tag{19}$$

$$z \in [0.0,8.0]. \tag{20}$$

The grid spacings are

$$\vec{d} = (0.1,0.1,0.1). \tag{21}$$

The source point's coordinates are (10.0,7.5,6.0) so its discrete coordinates are

$$\vec{s} = (41,26,61). \tag{22}$$

The forward star is defined by $\vec{f} = (3,3,3)$. The experiment has been coded as t802_1. Figure 6 shows the contours of the travel times of t802_1 (left) and the relative differences with r802_reg (right), Figure 7 shows the distribution of the differences. The other source point, 807, has been chosen at a smaller depth level so that the salt dome structures are just between the source and the receivers. It has been shown (Moser, 1990) that the part of the model below the source point is not illuminated by rays that reach the surface so that it can be discarded, which means that the rest of model can be sampled much more densely with the same number of nodes.

The depth range has been reduced to

$$z \in [0.0,2.5], \tag{23}$$

so that a much denser depth sampling is possible with the same effort. The grid spacing is

$$\vec{d} = (0.1,0.1,0.05)\,, \tag{24}$$

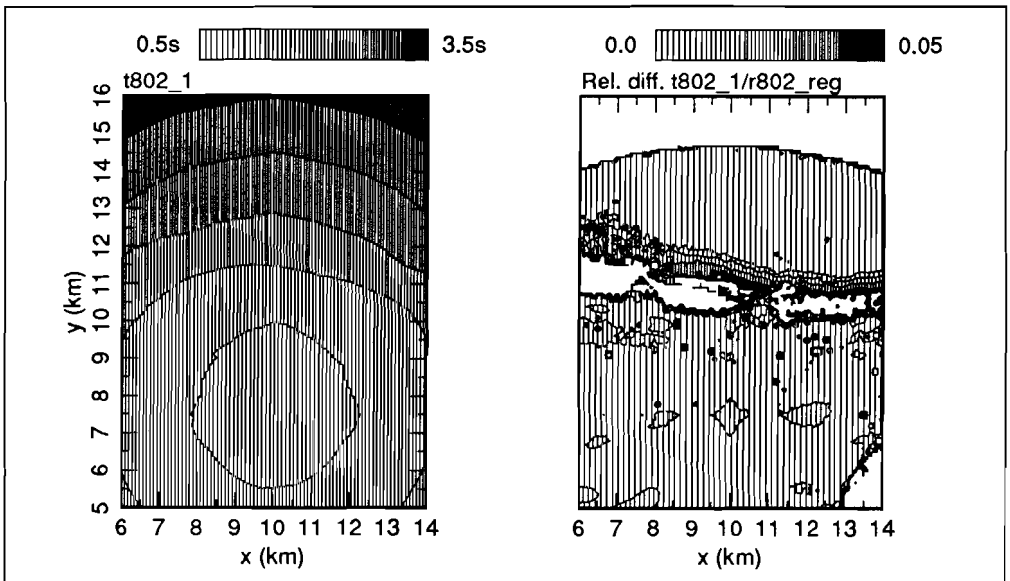

Figure 6. The shortest path travel times for the t802_1 test (left) and the relative difference with the regularised Shell travel times of the source point 802 (right). $\vec{n} = (81,111,81)$, $\vec{d} = (0.1,0.1,0.1)$, $\vec{f} = (3,3,3)$.

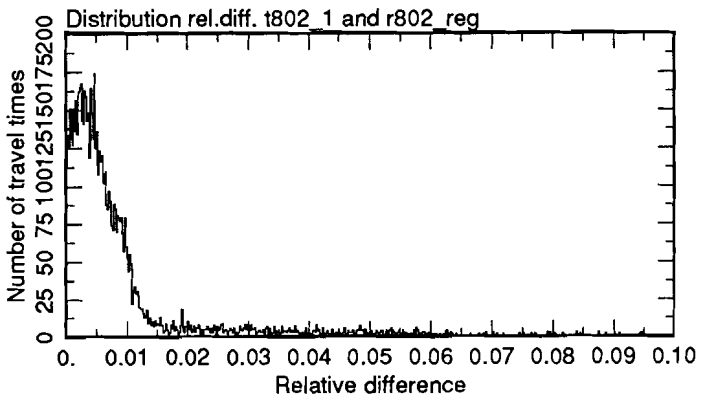

Figure 7. Distribution of the relative difference of the shortest path travel times of t802_1 and the regularised Shell travel times of the source point 802.

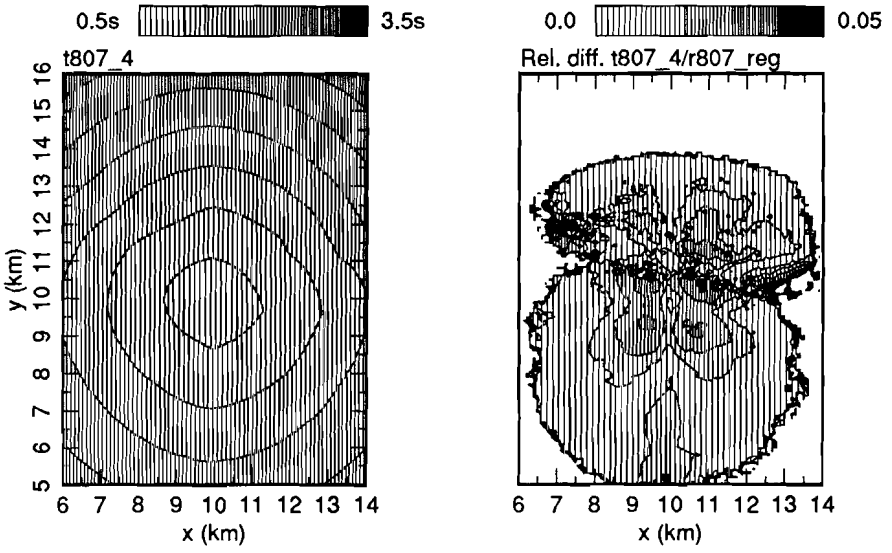

Figure 8. The shortest path travel times for the t807_4 test (left) and the relative difference with the regularised Shell travel times of the source point 807 (right). $\vec{n}$ = (81,111,51), $\vec{d}$ = (0.1,0.1,0.05), $\vec{f}$ = (5,5,3).

so that the number of nodes is

$$\vec{n} = (81,111,51) \tag{25}$$

and the discrete coordinates of the source are

$$\vec{s} = (41,51,51). \tag{26}$$

The forward star is defined by $\vec{f} = (5,5,3)$ and the travel times are coded as t807_4. Its travel time contours are shown in Figure 8 (left), the relative differences with r807_reg in Figure 8 (right) and the distribution of the differences in Figure 9.

A two dimensional test on a slice of the model CS673 enables the plotting of the shortest paths so that their properties are discernible, for instance, if they represent classical ray arrivals or diffractions or refractions. The source point $\vec{s}_{807}$ has been used for such a test through an x-section at x=10.0km. The two dimensional grid is specified by

$$\vec{n} = (1,101,101), \tag{27}$$

$$x \in [10.0,10.0], \tag{28}$$

$$y \in [8.0,12.0], \tag{29}$$

$$z \in [0.0,2.5], \tag{30}$$

so

$$\vec{d} = (0.0,0.04,0.025). \tag{31}$$

The points of this grid are linked together to form a network that is characterised by

$$\vec{f} = (0,5,5). \tag{32}$$

Figure 10 shows the x-section through the model CS673 at x = 10.0km (x807_10) and the shortest paths are shown in Figure 11 (p807_10).

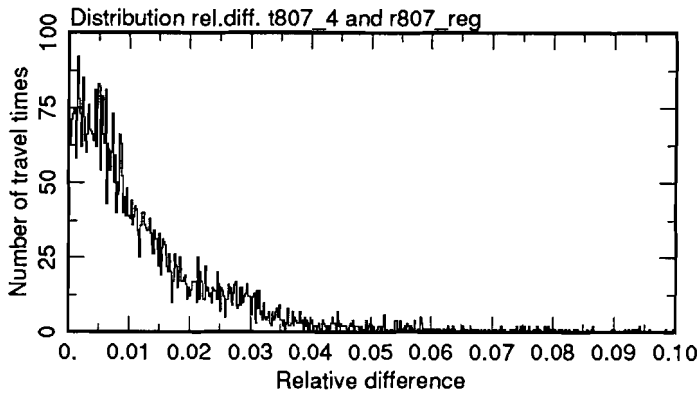

Figure 9. Distribution of the relative difference of the shortest path travel times of t807_4 and the regularised travel times of the source point 807.

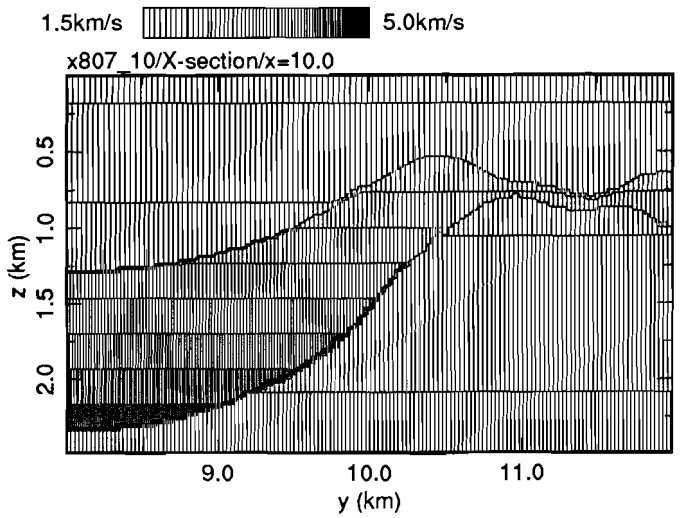

Figure 10. x-section through source location 807 with 101 *nodes in the y- and z-direction.*

§2.3 Discussion

The tests described in §2.2 show a considerable discrepancy between the shortest path travel times and the travel times computed by the Shell ray tracer. In both tests most of the travel times of both methods fail to agree within the desired 0.1%. A variety of effects is due to this disagreement.

First, there is the error in the shortest path calculations due to the discretisation in space and angle. To investigate these errors a comparison was made of the shortest path calculations with travel times that are precise within machine precision in a simple two dimensional model that is representative for two dimensional slices of the model CS673, namely consisting of two linear velocity regions separated by a curved interface (Figure 12). The travel times from one source point to a rectangular grid of receiver points in this model can be calculated with arbitrary precision by means of the bending procedure described in Chapter 5 because the travel times within the layers can be evaluated analytically (See Chapter 4, formula (41), or Červený, 1987). The velocity field consists of two layers with velocity functions $c_1 = 1.0 + 0.01z$ for the uppermost layer and $c_2 = 2.0 + 0.02z$ for the bottom layer. The two layers are separated by the parabola through the points (0.0,70.0), (50.0,40.0) and (100.0,70.0), the dashed line in Figure 12. A source point is located at $(x,z) = (50.0,0.0)$. The model is covered with a 99×99 grid of nodes. $\vec{f} = (3,0,3)$, so each node is connected with nodes in a $[-3,3] \times [-3,3]$ neighbourhood of it. The travel time has been calculated for each grid point above the interface with the analytical formula, below the interface with the bending procedure. This procedure gives a value T^* for the travel time within machine precision for well behaved interfaces. However, it does not find refracted ray paths in contrast with the shortest path method. The relative travel time error

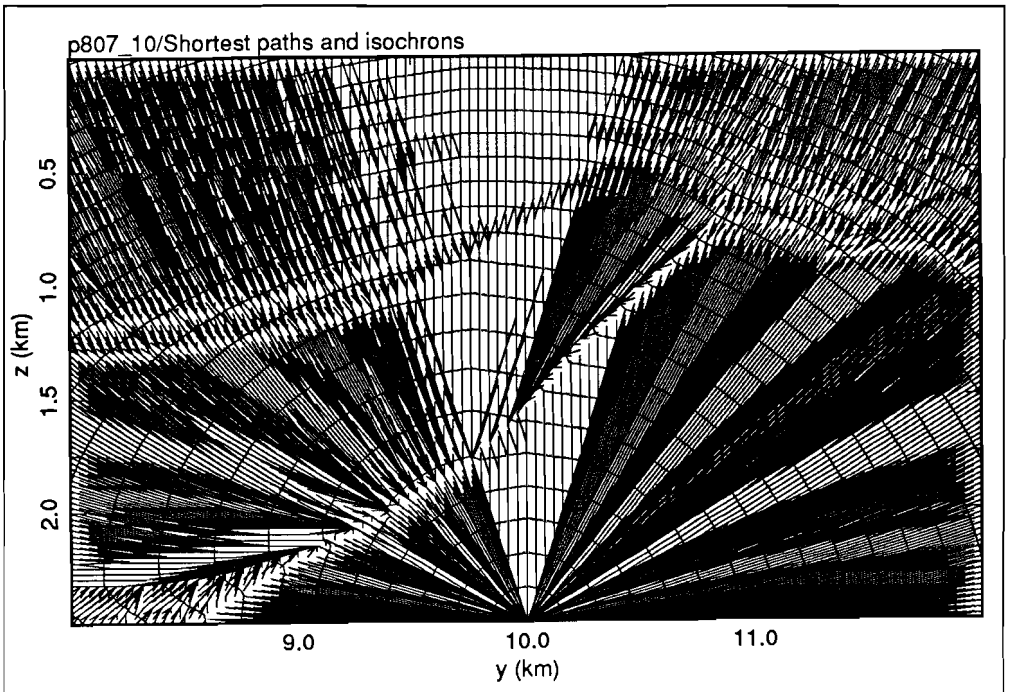

Figure 11. Shortest paths in the model of Figure 10 together with travel time contours.

is computed as

$$\frac{|\ T^* - T_{SPT}\ |}{T^*}. \tag{33}$$

It is shown in Figure 12 that the error in this model is mainly below 0.4% for the 99×99 grid. The largest errors occur at the interface, due to discretisation of the interface and the refracted shortest paths. A series of tests on slightly different velocity fields, with other gradients, with other shapes of the interface and also with a low velocity layer below the interface, showed the same results: the travel time errors are mainly below 0.4% for the 99×99 grid; at the interfaces they are larger due to the discretisation and the refracted shortest paths. Moreover, the relative errors do not accumulate along the shortest paths; therefore, it can be expected that the same conclusions can be drawn for models with more interfaces.

An important conclusion from these tests, in view of the comparison with the Shell computations, is that the travel time difference can be imputed only for approximately 0.4% to the shortest path calculations, even with the unsophisticated forward star definition (4). These tests also reveal that refracted ray paths can result in large differences between the shortest path calculation and the classical ray tracing. The model CS673 has a high velocity zone between two low velocity zones, as can be seen in Figure 10 and from the ray paths of Figure 11. Refracted ray paths at the contrast between the high velocity layer and

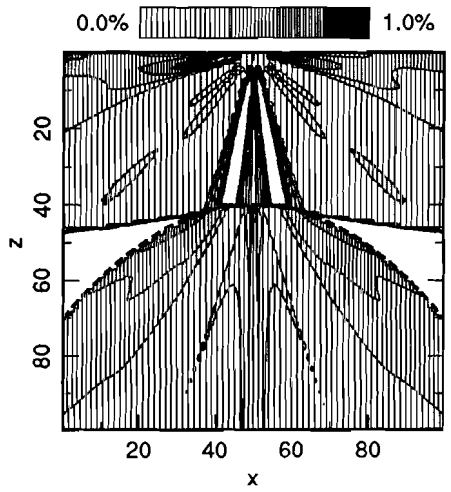

Figure 12. Relative travel time errors in % of the shortest path travel times in a model consisting of two layers with velocities $c = 1.0 + 0.01z$ *(uppermost layer) and* $c = 2.0 + 0.02z$ *(bottom layer), separated by a curved interface (dashed line).* $\vec{n} = (99,1,99)$ *and* $\vec{f} = (3,0,3)$.

the bottom layer are clearly visible. They reach the part of the Earth's surface between approximately $y = 10.0$ and $y = 11.0$ as first arrivals or, equivalently, as shortest paths. This region corresponds to the region where large differences occur in the test t807_4.

Another part of the discrepancy between the shortest path travel times and the Shell travel times can be explained by regularisation. The regularisation, which is necessary for the comparison of the shortest path method with the Shell ray tracer, can account for an error of about 0.1% for the 802 source point and 0.25% for the 807 source point (Moser, 1990).

The remaining part of the discrepancy, about 0.5%, must be hidden in the Shell ray tracer.

§3 Accurate line integration

The accuracy of the shortest path calculations with a forward star definition as in section §1, with a two point trapezoidal approximation of the travel time between two connected nodes, is still too low (test in §2.3), especially in presence of discontinuities in the velocity field. The reason for this is illustrated in Figure 13. Figure 13a shows the same calculations as in Figure 12 but with a larger forward star, $\vec{f} = (5,0,5)$. From the asymptotic relationship, given in Chapter 4 and §1 of this chapter, it could be expected that the errors are much smaller than with the forward star defined by $\vec{f} = (3,0,3)$ because there are more connections per node. This is indeed the case in the uppermost layer of the model where the error is reduced to less than 0.2% almost everywhere. In almost the whole lower part, however, the error is more than 1.0%.

The two point trapezoidal approximation of the travel time along long connections, like the

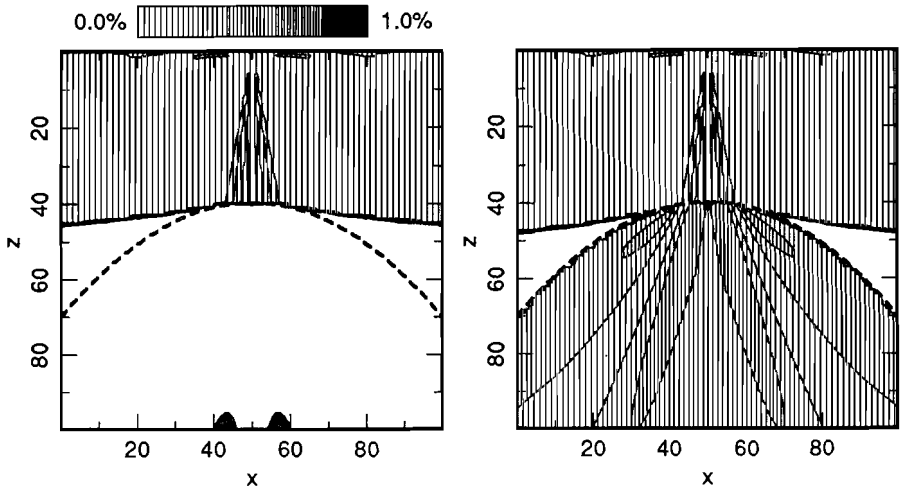

Figure 13. Same as Figure 12 but with $\vec{f} = (5,0,5)$. *a) (left) Two point trapezoidal approximation of the weights of the connections. b) (right) Multipoint approximation.*

ones in Figure 2, is apparently too inaccurate when the velocity changes rapidly from node to node. However, this does not contradict the asymptotic relationship for the travel time error because the increase of the forward star, necessary to refine the angle discretisation, introduces longer connections and thus a less adequate coordinate discretisation. Another effect of the two point trapezoidal approximation is that when the velocity at the nodes between the end nodes of a connection is smaller than the velocity at the end nodes themselves, the shortest path algorithm favours this connection and results in an underestimation of the travel time. This situation is similar to the one outlined in Chapter 5, where a minimisation of the trapezoidal travel time along a polygonal path in a concave slowness region implies that the points of the path are repelled out of the region, giving a too small travel time. Just as in Chapter 5 interpolation of the slowness along the connections is the solution to this problem.

The slowness along a connection can be interpolated by generating new points on the straight line between the end nodes and evaluating it at these points. This has been done in Figure 13b, using the same grid and forward stars as in Figure 13a. It is clear that interpolation between the end points of the connections improves the travel times considerably. The optimal number of extra points to be generated for the interpolation depends on the number of grid lines between the end points. When the slowness field between the nodes is defined by linear interpolation of the slowness values at the node locations, it does not change rapidly between them, so there are as many intermediate points needed as there are grid lines between the end nodes. An extended trapezoidal integration scheme gives then a good approximation for the travel time along a straight line through the slowness field, which is obtained from linear interpolation between the grid points. Such a procedure has been used to produce Figure 13b. Figure 14 offers an

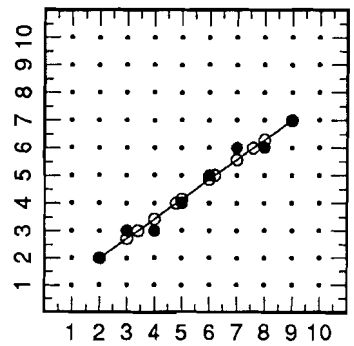

Figure 14. Line integration in a discrete grid.

illustration. Suppose that there is a two dimensional rectangular equidistant 10×10 grid of nodes and that the weight of the connection between the node on (2,2) and the node on (9,7) is required. The two point trapezoidal approximation uses only the slowness values at the end nodes (2,2) and (9,7) and the Euclidian distance between them and discards the slowness variations on the nodes in between. The extended trapezoidal rule, described above, uses, apart from the slownesses at the end nodes, the slownesses at the positions given by open circles.

In the version described above the computation time for the shortest algorithm with the extended trapezoidal rule is inadmissibly large; it is 8 times larger for the results of Figure 13b than for Figure 13a. However, Mitchell and Dickof (1990) show that the generation of intermediate points between two points in a discrete grid, necessary for the line integration, can be done much faster when only integer operations are used instead of floating-point operations. They distinguish between parametric schemes and nonparametric schemes; a parametric scheme uses the arc length along the line as a parameter, the nonparametric schemes are based on Bresenham's line drawing algorithm (Newman and Sproull, 1981). This algorithm is used in computer graphics to draw lines that consist only of the points of a raster. The line from (2,2) to (9,7) in Figure 14, drawn with Bresenham's algorithm, consists of the heavy black nodes. When the raster or grid is made fine enough, the points are not visibly different from the straight line between the end points. Both the open circles and the black nodes in Figure 14 can be found by using only integer operations; the black nodes because they lie on node locations, the open circles because their coordinates are rational numbers with the same denominator so that the necessary division can be done afterwards and only once. Mitchell and Dickof (1990) give two integer-arithmetic integration schemes, one based on the open circles and one on the black nodes . The first scheme is as accurate as the floating point scheme but it is nine times faster on their computer. The second scheme is somewhat less accurate but 19 times faster.

In Figure 15 the floating-point scheme for the calculation of the weight in a forward star is compared with the second integer-arithmetic scheme, referenced above. In the floating-point scheme the intersections of the connection between the end nodes and the grid lines are computed with floating-point operations and then used in the extended trapezoidal rule. The integer-arithmetic scheme generates the black nodes by integer manipulations,

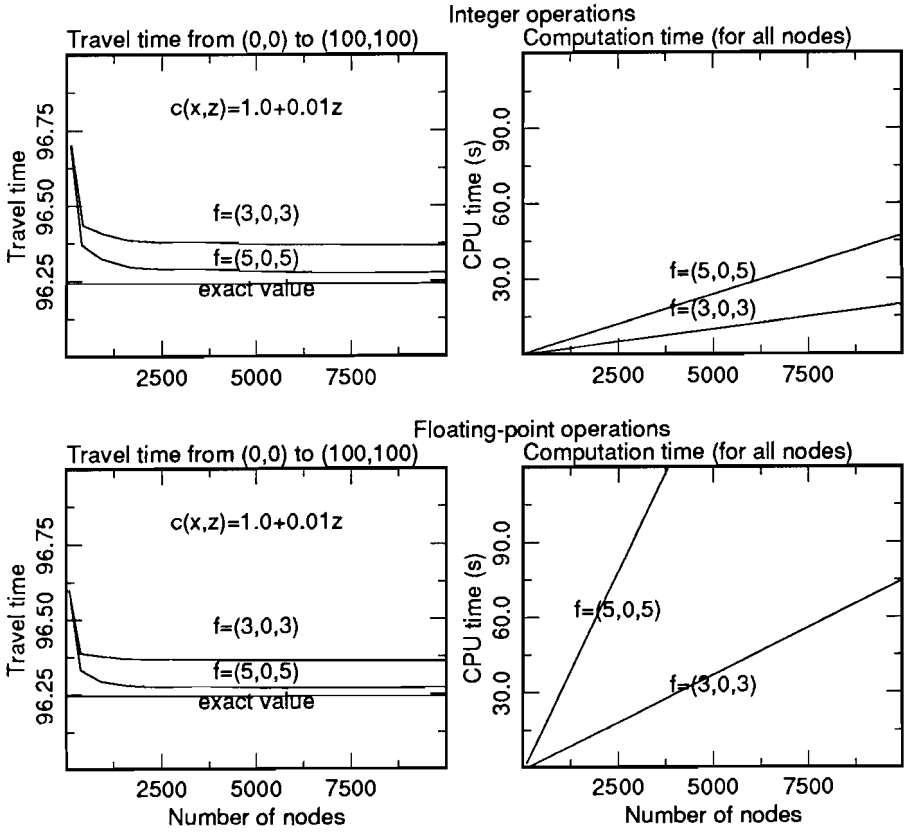

Figure 15. Comparison between the integer-arithmetic and the floating point integration schemes.

averages the slownesses at these points and multiplies this with the Euclidian distance to give the result. The calculations in Figure 15 are done for a series of grids with an increasing number of nodes and forward stars defined by $\vec{f} = (5,0,5)$ and $\vec{f} = (3,0,3)$ respectively. The velocity field is given by $c(x,z) = 1.0 + 0.01z$ and the computed travel time from (0.0,0.0) to (100.0,100.0) (real coordinates) is plotted. The exact value is 96.24237. The computed travel times are shown at the left, the computation times for the shortest paths from (0.0,0.0) to all n nodes are shown at the right; the above two plots refer to the integer arithmetic, the two plots below to the floating-point arithmetic. The plots show that the accuracy is approximately the same for both schemes but that the computation times are much smaller for the integer calculations.

This means that the line integration by using only integer operations is indeed much more efficient without loss of precision. This scheme is used in sections §4 and §5 and Chapters 7 and 8.

§4 Comparison with the finite difference solution of the wave equation

The claim that the shortest path method is able to find the correct travel times also in shadow zones, where classical ray methods break down, can be supported by the finite difference method for the solution of the acoustic wave equation. In this section the shortest path method is compared with the finite difference programme of Marquering (1991) (based on Mufti, 1990) on one example that shows wave propagation in a shadow zone. It is shown that these waves have an observable amplitude, so that the shortest path arrival time has a real physical meaning.

A two dimensional region of $[0.0,1.019] \times [0.0,1.019]$ is covered by a 101×101 rectangular grid of nodes. The velocity is chosen equal to 1.0, except for a circularly shaped low velocity zone, centered at (50,70) (discrete coordinates) and with a radius of 20 grid spacings, where the velocity is equal to 0.5.

This model is hit from below by a plane wave with an initial source function

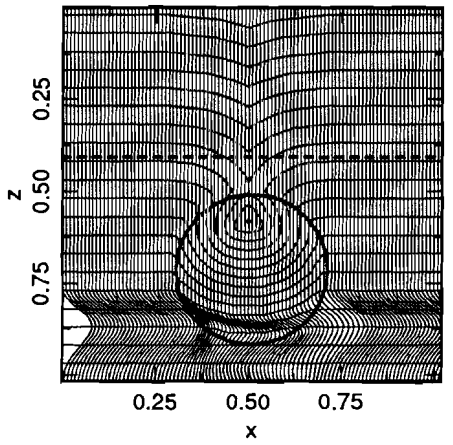

Figure 16. Snapshot of acoustic wave computed with finite differences at $tt = 0.255$. *The amplitude is given by the horizontal displacement of the vertical lines. The horizontal lines are the isochrons of the shortest path calculations, increment* 0.05 *time units,* $tt = 0.0$ *at the bottom. The velocity is equal to* 1.0, *except for the low velocity zone* ($c = 0.5$), *shown with the heavy circle. The dashed heavy line is the line on which the seismograms of Figure 20 are collected.*

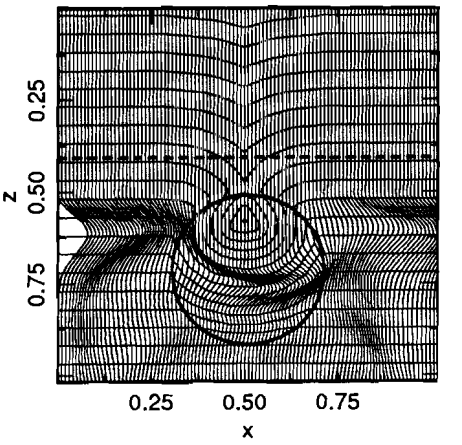

Figure 17. Same as Figure 16 at $tt = 0.510$.

$$f(t) = AH(t)t^2 e^{-\alpha t}, \tag{34}$$

where $A = 1000.0$, $\alpha = 30.0$ and $H(t)$ the Heavyside unit step function. The finite difference scheme is of the fourth order. It has a time step of 5.097×10^{-3} and there are approximately 6 points per wave length. Figures 16, 17, 18 and 19 show four successive snapshots of this wave, the first one just after reaching the low velocity zone and the others half-way up, right after and far after the low velocity zone. The low velocity zone is given by the heavy circle, the amplitude of the wave by the horizontal displacement of the vertical lines. The isochrons, computed by the shortest path method, are superposed on these plots for time steps of 0.05 time units, starting with $tt = 0.0$ at the lower boundary of the model. The shortest paths are calculated on a grid of 101×101 nodes with forward stars given by $\vec{f} = (5,0,5)$. The dashed line contains receiver points where seismograms are collected. These are shown in Figure 20 (vertical lines) together with the shortest path travel time curve (heavy horizontal line). The first arrival time of the finite-difference wave is defined as the first time on which they reach a fixed amplitude. This amplitude is chosen in such a way that the difference between the shortest path time and the finite-difference time vanishes for $x \leq 0.20$ and $x \geq 0.80$, where the arriving wave is a plane wave, so that the shortest path travel time is exact.

The snapshots show the expected agreement between the finite difference wave fronts and the shortest path isochrons. Of course, the shortest path method cannot account for variations in amplitude due to focusing or interference, certainly not in the shadow zone. Yet, the first arrivals on the seismograms, defined above, agree within about 0.5% with the shortest path travel times. The seismograms show a nonnegligible amplitude of the first arrival in the shadow zone at about $tt = 0.7$. This first arrival corresponds to the wave which is diffracted by the low velocity circle. The ray theoretical arrival is also visible at the seismogram at about $tt = 1.0$. The discrepancy between the ray theoretical first arrival

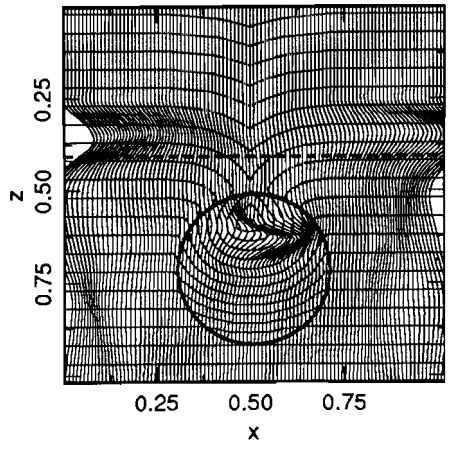

Figure 18. Same as Figure 16 at tt = 0.764.

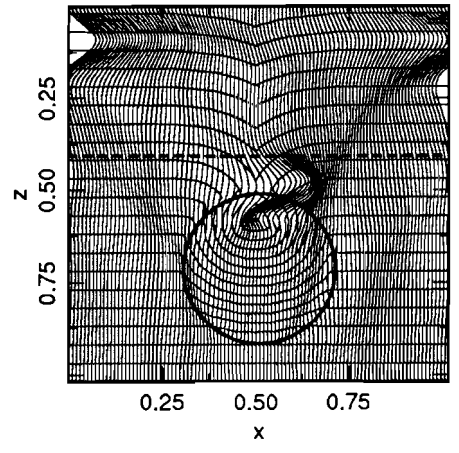

Figure 19. Same as Figure 16 at tt = 1.019.

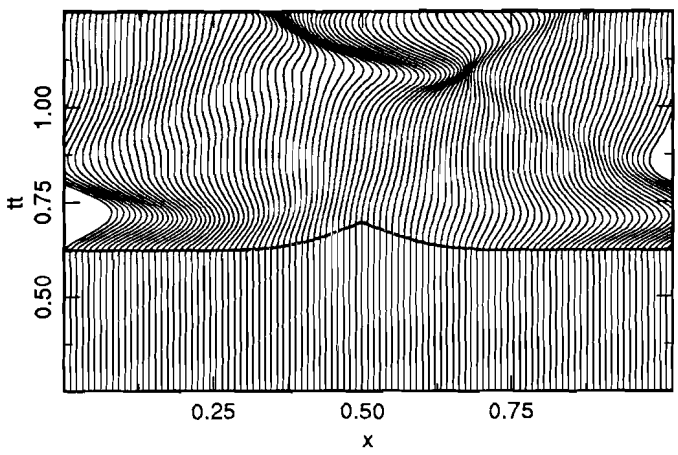

Figure 20. Finite difference seismograms (vertical lines) and shortest path travel times (horizontal heavy line) at z = 0.408.

time and the physical first arrival time gives rise to the so-called 'Wielandt effect' in seismic tomography (Wielandt, 1987).

The finite difference computations to produce these snapshots and the seismograms cost about one hour CPU time, the shortest path calculations about 46 seconds. However, the results obtained within these computation times are totally different: the shortest path method gives only the shortest path to and the first arrival time at each point of the grid while the finite difference method also provides information on amplitude and phase of the first and all later arrivals.

§5 Comparison with bending

By increasing the number of nodes and the number of connections per node and using the advanced line integration (§3) for the definition of the weights of the connections it is certainly possible to produce arbitrarily accurate results. However, the computational effort is then far out of proportion; also, the results are obtained for the whole volume of nodes, which is far too much. When only relatively few ray paths and travel times are required within the 0.1% accuracy mentioned in the introduction of this chapter, it may be better to compute shortest paths on a coarse grid and use them as initial guesses for additional bending.

In this section ray paths and travel times are computed in a complicated three dimensional model consisting of linear velocity layers, separated by flat, dipping and curved interfaces. The model is first covered by a coarse rectangular grid of nodes. The shortest paths and their travel times are computed in this grid from a source point at the bottom of the model to all other nodes. For a subset of randomly selected nodes at the Earth's surface the ray paths and travel times are improved with additional bending (as in Chapter 5, Section on

discontinuities, Figure 8). The result is then compared with the bending procedure on nodes that are fixed to the interfaces and with travel times computed analytically (as in the same Chapter, same section, Figure 9).

The three dimensional model has a coordinate range of $[0.0,100.0] \times [0.0,100.0] \times [0.0,100.0]$ and consists of four different layers with velocities

$$c_1(x,y,z) = 1.0 + 0.01x + 0.01z, \tag{35}$$

$$c_2(x,y,z) = 1.5 - 0.01y + 0.02z, \tag{36}$$

$$c_3(x,y,z) = 2.0 - 0.01x + 0.03z, \tag{37}$$

$$c_4(x,y,z) = 2.5 + 0.01y + 0.03z. \tag{38}$$

The Earth's surface is at $z = 0.0$. The layers are separated by three interfaces given by

$$z = 25.0 + 0.1(x - 50.0) - 0.1(y - 50.0) \tag{39}$$

between c_1 and c_2,

$$z = 50.0 + 5.0 \cos\left(\frac{2\pi x}{40.0}\right) \cos\left(\frac{2\pi y}{40.0}\right) \tag{40}$$

between c_2 and c_3 and

$$z = 75.0 \tag{41}$$

between c_3 and c_4 (Figure 21).

The model has been covered by a rectangular grid of nodes with $\vec{n} = (26,26,26)$. The source point has been chosen at the bottom of the model at

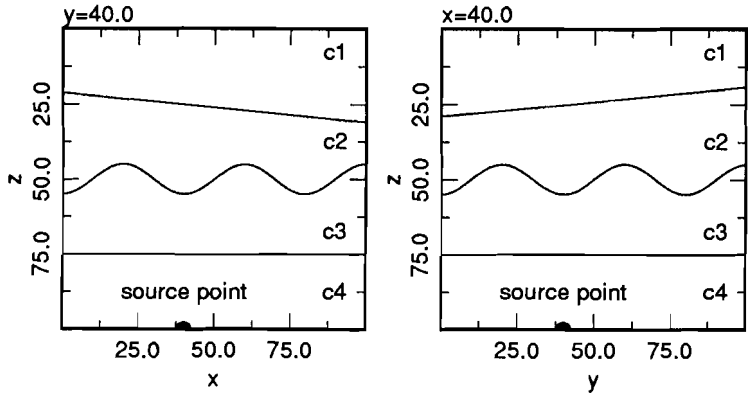

Figure 21. x- and y-sections of the test model for additional bending.

$$\vec{s} = (40.0,40.0,100.0). \tag{42}$$

The forward star is defined by $\vec{f} = (3,3,6)$ with a finer angle discretisation in the z-direction than in the x- and y-direction in order to give more accurate results for the travel times at the Earth's surface. Its weights are computed with the advanced line integration (§3).

The additional bending procedure has been applied on 400 randomly chosen shortest paths to the Earth's surface. The points of the shortest paths serve as support points of initial guesses that consist of Beta-splines ($\beta_1 = 1.0$, $\beta_2 = 0.0$). 25 intermediate points on each segment of the Beta-spline have been chosen in order to enable an accurate integration ($m = 25$ in the notation of Chapter 5). The slowness, given by the above formulae (35) to (38), has been integrated along the Beta-spline without attempt to fix the support points on the interfaces (as in Chapter 5, Figure 8). The gradient of the travel time as a function of the coordinates of the support points is evaluated numerically with an increment equal to 0.5. The tolerance is chosen equal to the machine precision. These parameters controlling the bending process are kept fixed for all receiver points.

The shortest path travel times and the additional bending travel times have been compared with the travel times obtained from the bending procedure described in Chapter 5, Figure 9, in which the support points are allowed to move only on the interfaces and the travel time between two successive support points is computed analytically. This bending procedure, referred to as 'exact', gives travel times within machine precision because the travel times along all candidate paths are obtained from summations of analytically known travel times.

274 of the 400 randomly chosen grid points at $z = 0.0$ occupy different positions. These 274 are used as input for the bending procedure. The shortest path travel times for the 274 receiver nodes agree on average within 1.5% with the exact bending times, which is favourable for the coarse grid ($\vec{n} = (26,26,26)$) that has been used.

In order to get an accuracy of 0.1% for all additional bending times, it appears to be necessary to change the parameters controlling the bending procedure (β_1, β_2, m, the increment for numerical differentiation and the tolerance) from receiver point to receiver point. This is caused by the fact that the convergence of the additional bending procedure, in contrast with the shortest method, is dependent on the complexity of the velocity model. Therefore, its efficiency varies from point to point. With the parameters kept fixed, the additional bending travel times agree on average within 0.5% with the 'exact' travel times; 105 of the 274 arrival times agree within 0.1%. For other choices of the control parameters all travel times agree within 0.1% with the 'exact' bending travel time, at the cost of longer computation times. Figure 22 shows a contour plot of the shortest path travel times at $z = 0.0$. The black circles denote the 105 nodes of the 274 selected nodes for which the additional bending travel time agrees within 0.1% with the 'exact' bending travel time. The size of the black circles is proportional to the relative errors in percents.

The computation time for the shortest paths from the source point to all other points is 400.0 seconds, each additional bending with the controlling parameters given above costs a few seconds.

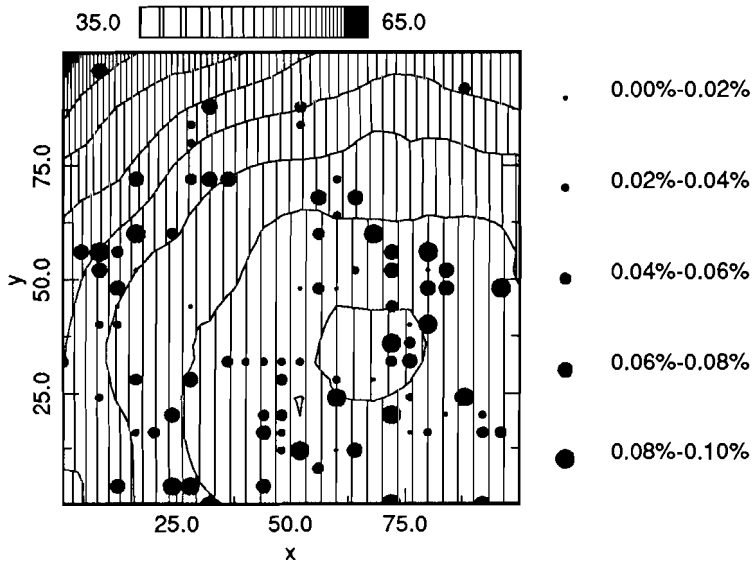

Figure 22. Contour plot of shortest path travel times at z = 0.0. 105 receiver points are selected and used for additional bending (see the text). They are denoted by the black circles with a radius proportional to the error of the additional bending travel time.

Conclusion

The shortest path method has been compared with other methods for forward seismic modeling.

The comparison with the Shell ray tracing software shows the travel times to agree within only 1.0% instead of the required 0.1%. The differences can be explained partly by the shortest path error characteristics (0.4%), partly by the necessary regularisation of the Shell times onto the rectangular grid of shortest path times (0.1% to %0.25) and partly by the observation that some of the shortest paths are refracted or diffracted ray paths in contrast to the transmitted ray paths of the Shell computations. The remaining discrepancy could not be further investigated due to the confidential nature of the Shell software. Also, the efficiency of the shortest path method cannot be compared with the Shell software for the same reasons. Yet, the comparison has given suggestions to improve the shortest path travel times by taking into account the velocity variations along the connections of the network.

Section §3 shows that both the efficiency and the accuracy can be enhanced tremendously for the shortest path calculations on grid networks by invoking advanced line integration schemes that replace the floating-point manipulations by integer-arithmetic manipulations.

Practical comparison with other methods

The claim that travel times in shadow zones are found correctly by the shortest path method, when their corresponding amplitude is physically observable, can be supported by finite difference calculations.

The last section shows that the combination of the shortest path and the additional bending methods allows for an efficient and robust calculation of the ray paths and the travel times along them. For a correct choice of the parameters controlling the bending process, the accuracy of the travel times is below 0.1%.

CHAPTER 7

Hypocenter determination in strongly heterogeneous Earth models using the shortest path method

Abstract

The shortest path method (Moser, 1991) for the calculation of seismic ray paths and travel times along them can be applied directly in the hypocenter location method proposed by Tarantola and Valette (1982). It uses the analogy between seismic rays in the Earth and shortest paths in networks to construct simultaneously first arrival times from one point to all other points of a three-dimensional grid in a fast, robust way in Earth models of arbitrary complexity. Doing this for all stations of a seismic array one can find the hypocenter location by minimising the difference between the observed and the calculated travel times at the stations over the three-dimensional grid.

The concept of probability density functions allows then for a fully nonlinear examination of the uncertainties in the hypocenter location due to uncertainties in the travel time data, numerical errors in the calculated travel times and, to a limited extent, incomplete knowledge about the Earth model. The result is a three-dimensional contour map of regions of equal confidence for the earthquake location. The method becomes especially attractive when more than one event recorded by the same array is studied because the calculation of the travel times, which is relatively the most time consuming operation, has to be done only once.

The method is applied to the location of an event that occurred on January 19 1989 in Israel.

Introduction

Finding the location of earthquakes requires the calculation of the travel times between an earthquake hypocenter and the recording stations, using a velocity model between hypocenter and station. Therefore, ray tracing is an important element in the problem of locating earthquakes. Ray tracing in horizontally layered models does not pose many computational problems; however, tomographic studies of the crust beneath seismic arrays (Thurber, 1983) often lead to complicated three-dimensional models. It is proposed here to use the advantages of the shortest path ray tracing algorithm (Moser, 1991) in locating earthquakes in complicated media and calculating location errors.

With a known velocity model, the earthquake location problem has four unknowns, namely the hypocentral coordinates $x_0 = (x_0, y_0, z_0)$ and the origin time t_0, or $h_0 = (x_0, t_0)$. The unknowns may be estimated from the measured first arrival times τ_i at K stations of a seismic array with coordinates $x_i = (x_i, y_i, z_i)$, $i = 1, \cdots, K$. An estimate $\hat{h}_0 = (\hat{x}_0, t_0)$

This chapter has been submitted for publication as:

Moser, van Eck and Nolet, submitted to the Journal of Geophysical Research, 1991, copyright by American Geophysical Union

of the hypocentral parameters will result in travel time residuals $r_i(\hat{\boldsymbol{h}}_0)$ for each station:

$$r_i(\hat{\boldsymbol{h}}_0) = \tau_i - t_0 - T(\hat{\boldsymbol{x}}_0;\boldsymbol{x}_i) \tag{1}$$

where $T(\hat{\boldsymbol{x}}_0;\boldsymbol{x}_i)$ is the calculated travel time from $\hat{\boldsymbol{x}}_0$ to $\boldsymbol{x}_i$, computed by ray tracing in a known velocity model. Hypocentral estimates are found by minimising a misfit criterion, $g(\hat{\boldsymbol{h}})$, which is a function of this travel time residual (see for example Anderson, 1982):

$$g(\boldsymbol{h}_0) = f(r_1(\boldsymbol{h}_0), \cdots, r_K(\boldsymbol{h}_0)). \tag{2}$$

The computed travel times are in general non-linear functions of the hypocenter estimate and, consequently, the location problem is also non-linear. Generally, this problem is linearised and solved with iterative methods, such as Gauss-Newton, conjugate gradient and damped least squares (Buland, 1976; Lee and Stewart, 1981; Pavlis, 1986). These methods have in common that they require travel time derivatives near an estimated hypocenter and that they are susceptible to instabilities when the problem is ill-conditioned, for instance when the hypocenter location is far outside the array (Thurber, 1985; Hirata and Matsu'ura, 1987).

This ill-conditioned problem may be circumvented by minimising $g(\boldsymbol{h}_0)$ for three or two parameters while keeping the others fixed. This can be done either by minimising $g(\boldsymbol{h}_0)$ as a function of the epicenter, (x_0, y_0), and origin time, t_0, for discrete steps in hypocentral depth, z_0, (Rowlett and Forsyth, 1984) or by minimising $g(\boldsymbol{x}_0)$ for discrete steps in t_0. Alternatively, one may solve the ill-conditioned problem by not using derivatives, for example, with the simplex method (Rabinowitz, 1988) or localised grid searches (Sambridge and Kennett, 1986).

Grid ray tracing methods, such as the finite-difference method proposed by Vidale (1988) and the shortest path method (Moser, 1991), provide robust ray tracers with reasonable effort. Robust, because they are able to find global minima and handle complicated media, including shadow zones, without extra computational effort. As they construct complete travel time fields simultaneously, they can be applied to the earthquake location problems with great benefit (Nelson and Vidale, 1990).

This chapter combines a method proposed by Tarantola and Valette (1982) to solve the non-linear location problem, and the shortest path ray tracing method of Moser (1991). A complementary procedure to evaluate location errors due to errors in both the observations and the velocity model is also presented. The method is illustrated and tested with the location of a small event near the Jordan-Dead Sea fault.

§1 The shortest path method

The shortest path method for seismic ray tracing is originally due to Nakanishi and Yamaguchi (1986); variant that is efficient enough for applications in three-dimensional models was developed by Moser (1991). It is based on Fermat's minimum travel time principle for seismic ray paths and the algorithms that have been developed in network theory for the calculation of shortest paths between nodes of a network. In order to avoid confusion this chapter consistently uses the term 'network' for the system of nodes and 'array' for the system of seismic stations.

The Earth model is represented by a network consisting of a large set of points (or nodes) and connections between points that are close in space. Each connection is given a weight

equal to the travel time of a seismic wave along it. The shortest path between two points is then defined as a path following those connections for which the sum of the weights is as small as possible. According to Fermat's principle this shortest path is an approximation of the seismic ray between the two points.

The advantages of the shortest path method for seismic ray tracing are its efficiency, robustness and global result. The global result, a consequence of the nature of shortest path algorithms, means that the algorithm constructs the shortest paths from one source point to all other points of the network simultaneously. The computation time grows approximately linearly with the number of nodes. On the other hand, the number of nodes is directly related to the accuracy of the computed travel times. The shortest path method is robust because it always gives the first arrival time of wave energy, no matter how complicated the medium; it even yields a result when classical ray theory breaks down, as in shadow zones. By increasing the density of nodes and connections between them, the shortest paths can be brought arbitrarily close to the global minimum travel time paths between the source and the receivers. There are no problems with convergence toward the ray path or with local minima.

The networks used in this chapter are based on a three-dimensional rectangular grid of nodes, covering a region M in the Earth, with positions given by Cartesian coordinates x,y,z (Figure 1). Each node is connected with the nodes in its direct surroundings (Figure 2). The velocity model is sampled on the node locations and linearly interpolated between them. The weight of a connection between two nodes is computed as the numerically integrated slowness along a straight line. One node is declared to be a source point and other nodes are receiver points. The Dijkstra algorithm with the heap implementation (Dijkstra, 1959; Moser, 1991) is then used to find the shortest paths and the travel times along them from the source node to the receiver nodes. In the more general case where the source point or the receiver points are not coincident with node locations, a simple extrapolation procedure is used to find the correct travel time.

Figure 3 shows the shortest paths in a two-dimensional grid covering a model with two constant velocity layers from the source point in the second layer to receiver points at the top of the model. Such a situation simulates an earthquake, where the source point is at the hypocenter and the receiver points are at the station locations.

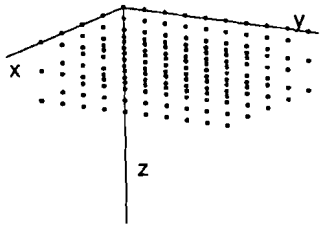

Figure 1. Three-dimensional rectangular grid of $5 \times 10 \times 3$ *nodes.*

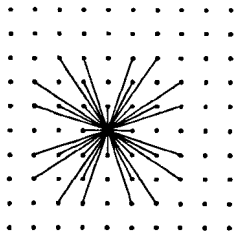

Figure 2. Two-dimensional grid of nodes with the connections of one node.

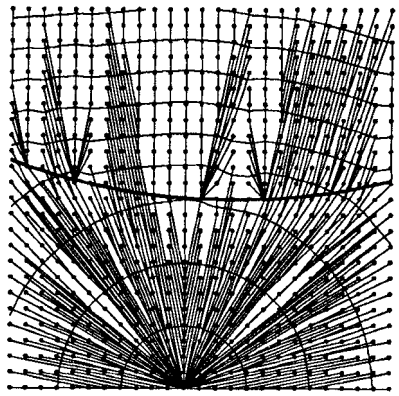

Figure 3. Two-dimensional 25×25 *grid, covering a model with two horizontal layers with velocities* 1.0 *and* 2.0, *separated by a curved interface (heavy line). The shortest paths from one node at the bottom of the model are shown (thin lines) together with the isochrons (contours).*

§2 Earthquake location using the shortest path method

Tarantola and Valette (1982) formulated the earthquake location problem in the language of probability theory in which the likelihood of the hypocenter being located in a given volume is obtained by integration over a joint probability density function. This approach is suitable for location error analysis and has, therefore, been used by the present authors. A short summary of their idea with respect to the earthquake location problem is presented here for completeness.

A probability density function is a multivariate function $f(\boldsymbol{x})$ that assigns probabilities to regions A in its domain space by:

$$P(\boldsymbol{x} \in A) = \int_A f(\boldsymbol{x}')\mathrm{d}\boldsymbol{x}'. \tag{3}$$

All knowledge about the data $\boldsymbol{d}$ and the parameters $\boldsymbol{p}$ can be given in the form of joint probability density functions, functions of both $\boldsymbol{d}$ and $\boldsymbol{p}$. Three joint probability density functions are introduced: 1. the *a priori* joint probability density function $\rho(\boldsymbol{d}, \boldsymbol{p})$, that describes the *a priori* knowledge of the data $\boldsymbol{d}$ and the parameters $\boldsymbol{p}$, 2. the null information function $\mu(\boldsymbol{d}, \boldsymbol{p})$, that describes total ignorance, and 3. a joint probability density function describing the theoretical relationship between the data and the parameters, $\theta(\boldsymbol{d}, \boldsymbol{p})$. These functions can be combined in a joint probability density function, not necessarily normalised:

$$\sigma(\boldsymbol{d}, \boldsymbol{p}) = \frac{\rho(\boldsymbol{d}, \boldsymbol{p})\theta(\boldsymbol{d}, \boldsymbol{p})}{\mu(\boldsymbol{d}, \boldsymbol{p})}. \tag{4}$$

Normalised, this *a posteriori* probability density function $\sigma(\boldsymbol{d}, \boldsymbol{p})$ represents the complete state of information. The inverse problem is solved by integration over the data:

$$\sigma_p(\boldsymbol{p}) = \int \sigma(\boldsymbol{d}, \boldsymbol{p}) d\boldsymbol{d}. \tag{5}$$

This result shows a probability density function acting on the parameter space. For example, the probability that $\boldsymbol{p}$ lies in region A, given $\rho(\boldsymbol{d}, \boldsymbol{p})$, $\theta(\boldsymbol{d}, \boldsymbol{p})$ and $\mu(\boldsymbol{d}, \boldsymbol{p})$, is:

$$P(\boldsymbol{p} \in A) = \int_A \sigma_p(\boldsymbol{p}') d\boldsymbol{p}'. \tag{6}$$

Consequently, in Tarantola and Valette's (1982) formulation the solution to the inverse problem is an *a posteriori* probability density function $\sigma_p(\boldsymbol{p})$ that requires the evaluation of integrals of the form $\int \sigma_p(\boldsymbol{p}') d\boldsymbol{p}'$. No assumption is made about linearity or non-linearity. To apply this formulation to hypocenter location the functions $\rho(\boldsymbol{d}, \boldsymbol{p})$, $\theta(\boldsymbol{d}, \boldsymbol{p})$ and $\mu(\boldsymbol{d}, \boldsymbol{p})$ are specified below.

In earthquake location the *a priori* information on the data $\boldsymbol{t}$ is independent of the *a priori* information about the hypocenter parameters $\boldsymbol{h}_0$. Therefore, the joint probability density function $\rho(\boldsymbol{d}, \boldsymbol{p})$ can be written as:

$$\rho(\boldsymbol{t}, \boldsymbol{h}_0) = \rho_d(\boldsymbol{t}) \rho_p(\boldsymbol{h}_0), \tag{7}$$

where $\boldsymbol{t}$ are the true arrival times at the stations. In its simplest form, the probability density function $\rho_p(\boldsymbol{h}_0)$ can be defined by assuming that the hypocenter location, $\boldsymbol{x}_0$, lies within the region M that is covered by the grid of nodes:

$$\rho_p(\boldsymbol{h}_0) = \begin{cases} 1 \text{ when } \boldsymbol{x}_0 \in M \\ 0 \text{ else} \end{cases}. \tag{8}$$

The other probability density function in equation (7), $\rho_d(\boldsymbol{t})$, describes the *a priori* knowledge about the measurements in terms of the probability that the true arrival time at station i is t_i instead of the measured τ_i. In the rest of the chapter it is assumed that the true arrival times are measured with a Gaussian error distribution around τ_i:

$$\rho_d(\boldsymbol{t}) = \exp\left[-\frac{1}{2} \sum_{i=1}^{K} \sum_{j=1}^{K} w_{ij}^{\rho} (t_i - \tau_i)(t_j - \tau_j) \right]. \tag{9}$$

w_{ij}^{ρ} are the elements of $(\boldsymbol{C}^{\rho})^{-1}$, where $\boldsymbol{C}^{\rho}$ is the *a priori* variance matrix of the observations.

The theoretical relationship between the calculated travel times, $T(\boldsymbol{x}_0; \boldsymbol{x}_i)$, and the parameters $\boldsymbol{h}_0 = (\boldsymbol{x}_0, t_0)$ can be written as a conditional Gaussian probability density function

$$\theta(\boldsymbol{t} \mid \boldsymbol{h}_0) = \exp\left[-\frac{1}{2} \sum_{i=1}^{K} \sum_{j=1}^{K} w_{ij}^{\theta} (t_i - T(\boldsymbol{x}_0; \boldsymbol{x}_i) - t_0)(t_j - T(\boldsymbol{x}_0; \boldsymbol{x}_j) - t_0) \right]. \tag{10}$$

$\theta(\boldsymbol{t} \mid \boldsymbol{h}_0)$ describes the probability that the true arrival times $\boldsymbol{t}$ for a given hypocenter $\boldsymbol{h}_0$ are found correctly using calculated travel times $T(\boldsymbol{x}_0; \boldsymbol{x}_i)$. w_{ij}^{θ} are the elements of $(\boldsymbol{C}^{\theta})^{-1}$, where $\boldsymbol{C}^{\theta}$ is the *a priori* variance matrix of the modeling. The assumption of a Gaussian distribution for both $\rho(\boldsymbol{t}, \boldsymbol{h}_0)$ and $\theta(\boldsymbol{t} \mid \boldsymbol{h}_0)$ is explained in the next section on location error analysis.

The probability density function $\mu(\boldsymbol{t}, \boldsymbol{h}_0)$, which describes total absence of *a priori*

knowledge, is assumed to be constant and equal to one.

The combination of equations (7), (8), (9) and (10) and substitution into equation (5) yields:

$$\sigma_p(\boldsymbol{h}_0) = \int \rho_p(\boldsymbol{h}_0)\rho_d(t)\theta(t \mid \boldsymbol{h}_0)\mathrm{d}t \tag{11}$$

$$= \rho_p(\boldsymbol{h}_0)\int \exp\left[-\frac{1}{2}\sum_{i=1}^{K}\sum_{j=1}^{K} w_{ij}^{\rho}(t_i - \tau_i)(t_j - \tau_j)\right] \times$$

$$\exp\left[-\frac{1}{2}\sum_{i=1}^{K}\sum_{j=1}^{K} w_{ij}^{\theta}(t_i - T(\boldsymbol{x}_0;\boldsymbol{x}_i) - t_0)(t_j - T(\boldsymbol{x}_0;\boldsymbol{x}_j) - t_0)\right]\mathrm{d}t.$$

The subscript p in $\sigma_p(\boldsymbol{h}_0)$ is dropped in the rest of the derivations. In their appendix Tarantola and Valette (1982) show that the integration over true travel time t in equation (11) can be carried out analytically with the result:

$$\sigma(\boldsymbol{h}_0) = \rho_p(\boldsymbol{h}_0)\exp\left[-\frac{1}{2}\sum_{i=1}^{K}\sum_{j=1}^{K} w_{ij}(\tau_i - T(\boldsymbol{x}_0;\boldsymbol{x}_i) - t_0)(\tau_j - T(\boldsymbol{x}_0;\boldsymbol{x}_j) - t_0)\right], \tag{12}$$

where w_{ij} are the elements of $(\boldsymbol{C}^{\rho} + \boldsymbol{C}^{\theta})^{-1}$.

The *a posteriori* probability density function for $\boldsymbol{x}_0$ can be found separately by integrating $\sigma(\boldsymbol{h}_0)$ over the origin time interval $[-\infty, \infty]$, or:

$$\sigma(\boldsymbol{x}_0) = \int_{-\infty}^{\infty} \sigma(\boldsymbol{h}_0)\mathrm{d}t_0 = \alpha \exp\left[-\frac{1}{2}g(\boldsymbol{x}_0)\right], \tag{13}$$

where α is a normalisation constant and $g(\boldsymbol{x}_0)$ is the misfit criterion of equation (2) which is here:

$$g(\boldsymbol{x}_0) = \sum_{i=1}^{K}\sum_{j=1}^{K} w_{ij} r_i(\boldsymbol{x}_0) r_j(\boldsymbol{x}_0) \tag{14}$$

with

$$r_i(\boldsymbol{x}_0) = \tau_i - \frac{\sum_{k=1}^{K} w_{ik}\tau_k}{\sum_{k=1}^{K} w_{ik}} - T(\boldsymbol{x}_0;\boldsymbol{x}_i) + \frac{\sum_{k=1}^{K} w_{ik} T(\boldsymbol{x}_0;\boldsymbol{x}_k)}{\sum_{k=1}^{K} w_{ik}}. \tag{15}$$

The maximum likelihood estimation for the hypocenter, $\hat{\boldsymbol{x}}_0$, is given by the location of the global maximum of $\sigma(\boldsymbol{x}_0)$:

$$\sigma_{\max} \equiv \max_{\boldsymbol{x}_0 \in M} \sigma(\boldsymbol{x}_0) = \sigma(\hat{\boldsymbol{x}}_0), \tag{16}$$

or, equivalently, the global minimum of $g(\boldsymbol{x}_0)$:

$$g_{\min} \equiv \min_{\boldsymbol{x}_0 \in M} g(\boldsymbol{x}_0) = g(\hat{\boldsymbol{x}}_0). \tag{17}$$

Similarly, the *a posteriori* probability density function for t_0 may be obtained by

integrating (12) over the space coordinates:

$$\sigma(t_0) = \int_M \sigma(\boldsymbol{h}_0)\mathrm{d}\boldsymbol{x}_0, \tag{18}$$

which is maximised by the maximum likelihood estimate for the origin time, t_0 (see appendix):

$$t_0 = \frac{\sum_{i=1}^{K}\sum_{j=1}^{K} w_{ij}(\tau_i - T(\boldsymbol{x}_0;\boldsymbol{x}_i))}{\sum_{i=1}^{K}\sum_{j=1}^{K} w_{ij}}. \tag{19}$$

Finally, the probability density function for the hypocenter location (13) can be normalised by using the assumption that the hypocenter is located within M (equation (8)) or:

$$\sigma(\boldsymbol{x}_0) = \frac{\exp\left[-\frac{1}{2}(g(\boldsymbol{x}_0) - g_{\min})\right]}{\int_M \exp\left[-\frac{1}{2}(g(\boldsymbol{x}_0') - g_{\min})\right]\mathrm{d}\boldsymbol{x}_0'}. \tag{20}$$

The denominator and the numerator in (20) have been multiplied with a factor $\exp(\frac{1}{2}g_{\min})$ in order to avoid underflow in the computer manipulations.

In this chapter the hypocentral location estimate, $\hat{\boldsymbol{x}}_0$, is found in two computational steps. First, a simple search over the three-dimensional grid for the minimum of $g(\boldsymbol{x}_0)$ will give the node with the global minimum. With reasonable assumptions on the smoothness of the velocity model the actual minimum, for which $g(\boldsymbol{x}_0) = g_{\min}$, will lie in a close neighbourhood of this node. Therefore, the actual minimum is found in a second step through a few iterations of Conjugate Gradients minimisation with a numerical approximation of the gradient of $g(\boldsymbol{x}_0)$ (Press *et al.*, 1986).

The disadvantage of the elegant solution in equation (20) when using traditional three-dimensional ray tracing is that the evaluation of integrals over possible hypocentral locations $\boldsymbol{x}_0$ requires travel time calculations over many different paths, which is generally not viable. The shortest path method, however, is ideally suited for this formulation, as travel times are evaluated at all grid points simultaneously.

As a spin-off of the shortest path method calculations, the angle at which the ray to a particular station leaves the source area becomes available even for complicated heterogeneous media. This is important for accurate focal mechanism solutions. Similarly, the azimuth measured at a station can be corrected for path distortions. This may be important in location procedures which use the azimuth of the incoming rays.

§3 Location error analysis

The location uncertainty information is contained in the probability density function $\sigma(\boldsymbol{x}_0)$ with no *a priori* assumptions about its form. Conventionally, $\sigma(\boldsymbol{x}_0)$ is obtained by assuming that the hypocenter location problem is linear and Gaussian so that the error or confidence region contours are ellipsoidal. However, as shown by Tarantola and Valette (1982) and in the example given in the next section, this is generally not valid.

The *a priori* probability density function, $\rho(t, \boldsymbol{h}_0)$, and the theoretical relation, $\theta(t \mid \boldsymbol{h}_0)$, have both been defined as Gaussian distributions. For $\rho(t, \boldsymbol{h}_0)$ this has been done for reasons of simplicity and may be questionable. Often used probability density functions for $\rho(t, \boldsymbol{h}_0)$ are Jeffreys' distribution (Jeffreys, 1936) or the exponential distribution (Anderson, 1982, Tarantola, 1987). However, the Gaussian distribution in the theoretical relationship, $\theta(t \mid \boldsymbol{h}_0)$ can be defended with the central limit theorem of statistics which states that the travel time, computed by summing over many connections, will approach a Gaussian distribution irrespective of the distribution in the individual connections. The advantage of Gaussian distributions is that the observational errors and the modeling errors can be combined into one covariance matrix (Tarantola, 1987, p. 58) and that the integral in equation (11) can be derived analytically. However, implementation of other distributions is possible, too.

The *a priori* covariance matrix $\boldsymbol{C}^{\rho}$ is assumed diagonal:

$$(\boldsymbol{C}^{\rho})_{ij} = \begin{cases} (\sigma_i^{\rho})^2 & \text{when } i = j \\ 0 & \text{else} \end{cases}, \tag{21}$$

where the variance $(\sigma_i^{\rho})^2$ is defined as the expectation value of the squared error $\delta\tau_i^2$ in the arrival time τ_i.

The theoretical covariance matrix $\boldsymbol{C}^{\theta}$ contains the variances $(\sigma_i^{\theta})^2$ of the computed travel times $T(\boldsymbol{x}_0;\boldsymbol{x}_i)$ on its diagonal and the covariances on the off-diagonal entries. One possible definition of these covariances is to relate them to the mutual distance of the stations they belong to:

$$(\boldsymbol{C}^{\theta})_{ij} = \sigma_i^{\theta}\sigma_j^{\theta}\exp\left[-\frac{1}{2}\frac{||\boldsymbol{x}_i - \boldsymbol{x}_j||^2}{\Delta^2}\right], \tag{22}$$

where Δ is some correlation length. The error δT_i in the travel time $T(\boldsymbol{x}_0;\boldsymbol{x}_i)$ is the sum of two contributions:

$$\delta T_i = \delta T_i^{model} + \delta T_i^{spt},$$

where δT_i^{model} is the error in the travel time introduced by incomplete knowledge of the Earth model and δT_i^{spt} is the error in the shortest path calculation. An analysis of the errors from incomplete knowledge about the Earth's velocity model makes use of Fermat's principle. The error δT_i^{model} in the travel time along a ray path γ in the true Earth model c, due to small deviations δc from c, can be expressed to the first order as an integral along γ (Nolet, 1987):

$$\delta T_i^{model} = \int_\gamma \delta(\frac{1}{c})ds = -\int_\gamma \frac{\delta c}{c^2}ds + O(\delta c^2), \quad (\delta c \to 0). \tag{23}$$

This means that travel time errors due to small model perturbations can be estimated easily. A conservative upper bound for this estimation is (Pavlis, 1986):

$$|\delta T_i^{model}| \leq \max_\gamma |\frac{1}{c+\delta c} - \frac{1}{c}| \int_\gamma ds, \tag{24}$$

which is the ray length multiplied by the maximum slowness perturbation. Equation (23) is invalid for large perturbations, as δT^{model} contributions from changes in the ray geometry may not be neglected anymore.

The errors in the shortest path calculations, δT_i^{spt}, can be estimated (Moser, 1991) and are usually an order of magnitude smaller than the modeling errors δT_i^{model} and the observed arrival time errors $\delta\tau_i$. Systematic errors due to the shortest path method can be disregarded because of the centered quantities in equation (15). Thus, σ^θ is dominated by insufficient knowledge of the subsurface velocities.

With $\sigma_i{}^\rho$ and $\sigma_i{}^\theta$ estimated, the weighting coefficients $w_{ij} = (C^\rho + C^\theta)^{-1}$ can be computed and, consequently, the probability density function $\sigma(x_0)$ of equation (20). In particular, the probability $P(\xi)$ that the hypocenter is located within a region where $\sigma(x_0) \geq \xi\sigma_{max}$ is given by:

$$P(\xi) \equiv P(\sigma(x_0) \geq \xi\sigma_{max}) = \int_{\sigma \geq \xi\sigma_{max}} \sigma(x_0')dx_0', \tag{25}$$

where $0 \leq \xi \leq 1$. When $\sigma(x_0)$ has no maxima other than the global maximum, which is always the case in linearised earthquake location, the region defined by $\sigma(x_0) \geq \xi\sigma_{max}$ has minimum volume among all regions with the same associated probability $P(\xi)$. It is, therefore, a generalisation of the confidence ellipsoids from linearised theory and can thus be associated with a confidence region (Buland, 1976).

In order to find the confidence region for a fixed probability the correct σ-contour has to be found, which means that the formula (25) should be inverted. This can be avoided by simultaneously evaluating (25) for a range of contours (for instance, ξ = 0.0 to 1.0 with step 0.01) and keeping a list of the results. It is then easy to invert this list into a list of contours corresponding to confidences (for instance, 0% to 100% with step 10%). The integrals appearing in (25) can be evaluated accurately by skipping the regions for which $\sigma(x_0)$ is below the machine precision and isolating the region for which it is computationally nonzero.

§4 Illustrative example

As explained in the previous sections, the shortest path method is especially fit to be combined with the hypocenter location method of Tarantola and Valette (1982). In this section the combined methods are applied to the location of a small earthquake (M_L = 2.7), that occurred on January 19, 1989 on one of the branches of the Jordan fault in Israel. The event choice was more or less an arbitrary one, only restricted by the requirement to have stations around the epicenter.

Table 1. ARRIVAL TIMES, REFERENCE MODEL AND HYPOCENTER

P-Arrival times for the January 18, 1989 Earthquake				
Station	x	y	z	$t(P)$
	(km)	(km)	(km)	(sec)
MML	188.84	204.84	-0.51	54.5
KRPI	202.80	206.00	0.00	54.8
BLVR	199.00	222.00	-0.14	55.4
GLH	211.23	235.53	-0.34	57.9
ATZ	175.51	247.59	-0.51	59.5
CRI	153.86	231.99	-0.43	60.0
ZNT	152.78	182.89	-0.31	60.1
JVI	182.46	149.03	-0.69	61.7

Reference crust model for Israel (I1)	
depth	P-velocity
(km)	(km/sec)
0-2.1	3.5
2.1-12.7	5.7
12.7-28.2	6.4
28.2-inf	7.9

Hypocenter location according to IPRG Bulletin				
	x_0^{bull} (km)	y_0^{bull} (km)	z_0^{bull} (km)	t_0^{bull} (sec)
	194.0	207.0	21.0	50.8
	δx_0^{bull} (km)	δy_0^{bull} (km)	δz_0^{bull} (km)	δt_0^{bull} (sec)
	1.4	0.9	2.0	0.13

Table 2. EXAMPLES

Example	Model	Grid	$\delta\tau$	δT			Δ	Figure
				δT^{spt}	$\delta(\frac{1}{c})$	δT^{model}		
			(sec)	(sec)	(sec)	(sec/km)	(sec)	(km)
1	I1	$10\times20\times5$	0.3	0.16	0.0	0.0	10.0	
2	I1	$30\times60\times15$	0.3	0.042	0.0	0.0	10.0	4a,b
3	I1	$40\times80\times20$	0.3	0.028	0.0	0.0	10.0	
4	I2	$30\times60\times15$	0.3	0.042	0.003	0.10–0.28	10.0	
5	I3	$30\times60\times15$	0.3	0.042	0.03	1.01–2.75	10.0	
6	I1	$30\times60\times15$	0.3	0.042	0.0	0.0	10.0	5a,b
7	I4	$30\times60\times15$	0.3	0.042	0.0	0.0	10.0	7
8	I4	$30\times60\times15$	0.3	0.042	0.0025	0.11–0.27	10.0	8

Table 3. RESULTS

Hypocenter location by means of the shortest path method								
Example	Hypocenter solution				Difference with Bulletin			
	x_0	y_0	z_0	t_0	$x_0 - x_0^{bull}$	$y_0 - y_0^{bull}$	$z_0 - z_0^{bull}$	$t_0 - t_0^{bull}$
	(km)	(km)	(km)	(sec)	(km)	(km)	(km)	(sec)
1	194.6	206.1	19.4	50.67	0.6	-0.9	-1.6	-0.13
2	193.8	206.5	18.5	50.74	-0.2	-0.5	-2.5	-0.06
3	194.6	206.3	18.9	50.89	0.6	-0.7	-2.1	0.09
4	193.6	206.4	18.4	50.73	-0.4	-0.6	-2.6	-0.07
5	192.7	205.6	18.5	50.61	-1.3	-1.4	-2.5	-0.19
6	192.9	211.9	16.1	51.47	-1.1	4.9	-4.9	0.67
7	193.2	205.8	16.1	51.35	-0.8	-1.2	-4.9	0.55
8	193.1	205.9	16.2	51.35	-0.9	-1.1	-4.8	0.55

All relevant data used in this example have been extracted from the bulletin (IPRG, Seismological Bulletin no. 8, event 16) of the Israel Seismograph Station Network (ISSN) and are summarised in Table 1.

Distances are given in kilometers, times in seconds. The data include the P-wave arrival times at 8 stations, the (Cartesian) station coordinates and the laterally homogeneous layered crustal model (I1), used in the ISSN location procedure. The travel time observation errors have been chosen conservatively, namely $\delta\tau = 0.3$sec, based on the assumption of a bad signal to noise ratio for small seismic events. However, different values of $\delta\tau$ are easily implemented without changing the essence of this experiment.

Also presented in Table 1 is the ISSN hypocenter location which is referred to as the Bulletin's solution and used as a reference location hereafter. Shapira (1983) describes the ISSN location algorithm that searches for the optimal epicenter at a sequence of depths using Geiger's method. In the ISSN procedure the depth with a minimum residual is chosen as the optimal hypocenter and the location errors were obtained from the covariance matrix calculated by the Singular Value Method. The location errors, denoted by δx_0^{bull}, δy_0^{bull}, δz_0^{bull} and δt_0^{bull}, refer to confidence intervals of 84%. All shortest path calculations were performed on a three-dimensional grid covering the larger part of Northern Israel, referred to as the region M and bounded by $x = [150, 215]$, $y = [145, 250]$, $z = [-2, 25]$, where the numbers indicate kilometers in the Israel Geographical Coordinate System. The x-coordinate increases eastward, the y-coordinate northward and the z-coordinate with depth. The lower bound for the z-coordinate is set to -2km in order to account for station elevations; the topography is simplified to a plateau at 2km above sea level with seismic velocities equal to the velocities in the first layer of the crustal model of Table 1. The velocity field is defined by the velocities on each grid node and by linear interpolation between them.

The optimum grid spacing with respect to computational effort, accuracy and an adequate modeling of strong heterogeneities is investigated in the first three examples. The computation time for the shortest paths from one node to all other n nodes of a network is proportional to $n \log n$, which is approximately linear in practice (Moser, 1991). The accuracy in the resulting travel times is quadratically proportional to the number of nodes in each coordinate direction and to the number of connections per node (Moser, 1991). The errors in the shortest path calculations, δT^{spt}, are estimated by comparing the shortest path

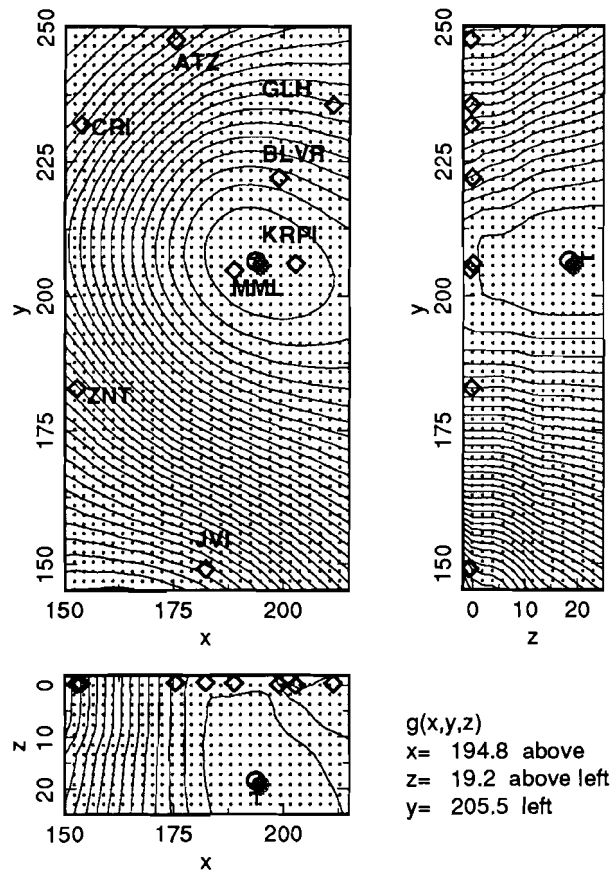

Figure 4. Earthquake location in the reference model of Israel I1, covered by a $30 \times 60 \times 15$ *grid of nodes (small points). See text, Tables 2 and 3, Example 2. (a)* z*-,* x*- and* y*-sections of the model, through the minimum misfit node which is denoted by a black circle. The open circle is the improved hypocenter location after a few iterations of Conjugate Gradients minimisation. The stations of the seismic array are depicted by diamonds. Also shown are contours of the misfit function* $g(\boldsymbol{x}_0)$*, ranging from* 1.0 *with step* 1.0. *The Bulletin's hypocenter solution is given by the cross, whose cross-beams have lengths equal to the error bars given in Table 1.*

travel times with travel times computed by the bending method of Moser, Nolet and Snieder (1991), both from the ISSN earthquake location to all grid points at the Earth's surface in the reference model I1. The travel times of the bending method can be assumed exact for this purpose. The differences between these two travel times were centered to the average difference over all receivers because the residuals (15) contain only centered quantities. In Table 2 the examples 1, 2 and 3 show the resulting error estimation δT^{spt} for

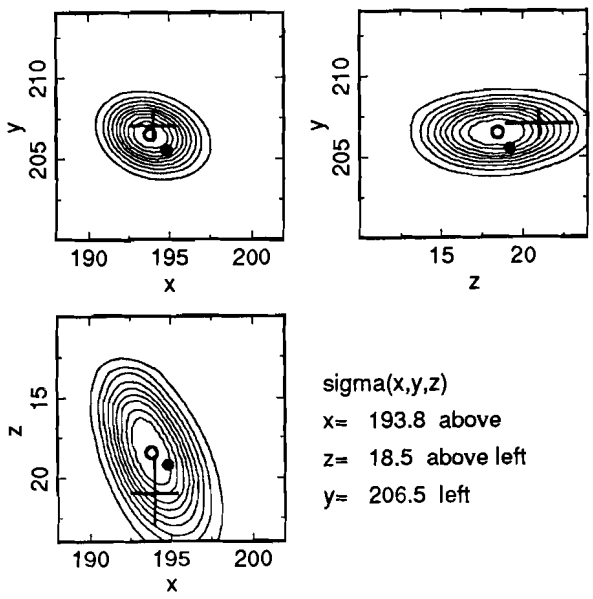

Figure 4. (b) Confidence regions: blow-ups of z-, x- and y-sections of the model, through the improved hypocenter, shown with an open circle. The contours are contours of equal probability density $\sigma(x_0)$, corresponding to 10% probability for the inner contour to 90% for the outer contour. Also shown are the minimum misfit node (black circle) and the Bulletin's hypocenter solution.

grids with $10 \times 20 \times 5$, $30 \times 60 \times 15$ and $40 \times 80 \times 20$ nodes, corresponding to grid spacings of about 5.5km, 2.0km and 1.5km respectively.

The hypocenter solutions for the three examples are not much different (Table 3), except for a discrepancy in the depth coordinate.

For all following experiments, the medium size grid of $30 \times 60 \times 15$ nodes was chosen where δT^{spt} is one order of magnitude smaller than $\delta\tau$, and details of a heterogeneous velocity model can still be modeled without excessive computation times. The misfit function $g(x_0)$ and the *a posteriori* probability density function $\sigma(x_0)$ of the second example are shown in Figures 4a and 4b.

In the next two examples, 4 and 5, the influence of random velocity fluctuations superposed on the reference crust model of Israel (I1) is investigated. The fluctuations range from –0.1km/sec to 0.1km/sec in the model I2 of Example 4 and from –1.0km/sec to 1.0km/sec in the model I3 of Example 5 and both have a spatial correlation length of about 20.0km. The resulting hypocenters are shown in Table 3; only the large fluctuations of model I3 (Example 5) have an effect larger than 1km on the epicenter solution. Example 4, with the small fluctuations of model I2, shows hardly any shifting of the solution. When the fluctuations are interpreted as uncertainties $\delta(\frac{1}{c})$ in the reference model, the theoretical

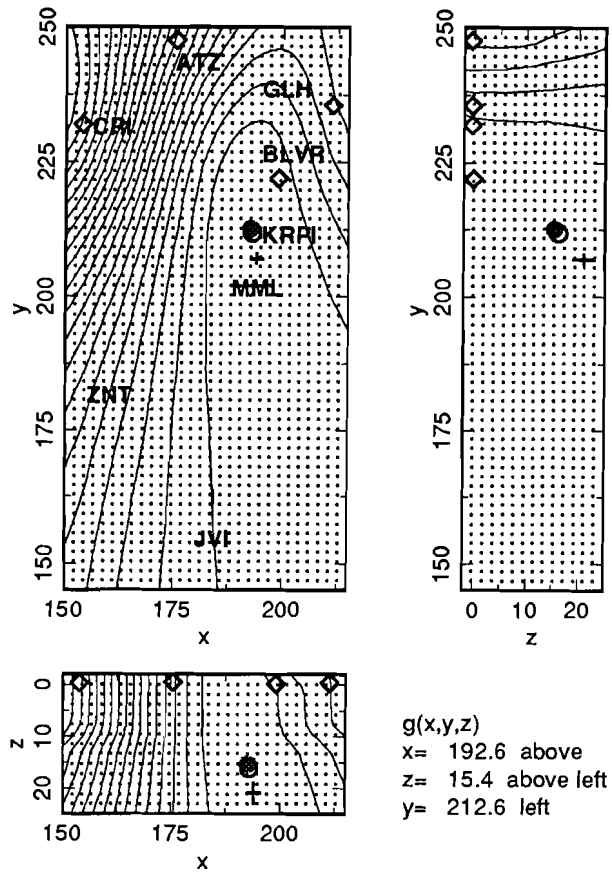

Figure 5. Earthquake location in an ill-conditioned situation (Tables 2 and 3, Example 6). Plot symbols and contours are the same as in Figure 4. (a) Misfit function $g(x_0)$ through the minimum misfit node.

errors δT^{model}, due to these uncertainties, can be estimated by means of formula (24). They will vary from station to station, as they depend on the length of the shortest path. Table 2 shows the range of errors δT^{model} for the eight stations for the examples where the theoretical modeling error is included ($\delta(\frac{1}{c}) \neq 0$). In Example 4, using the model I2 with fluctuations of ±0.1km/sec (or $\delta(\frac{1}{c}) = 0.003$sec/km), this results in travel time errors that are comparable to the arrival time error $\delta\tau$. However, in model I3, where the fluctuations are much larger (±1.0km/sec or $\delta(\frac{1}{c}) = 0.03$sec/km) they are one order of magnitude larger. This means that in the latter case the confidence regions are much larger. Thus,

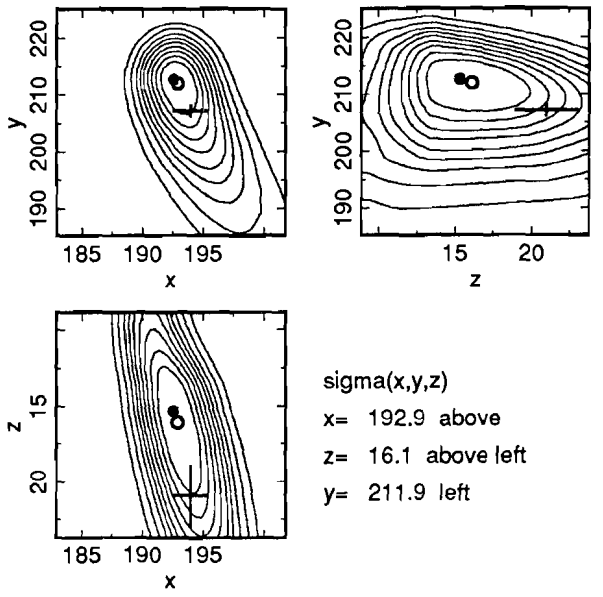

Figure 5. (b) Confidence regions through the hypocenter solution.

although the hypocenter solution is rather insensitive to changes in the velocity model, confidence regions can change drastically.

The next example (Example 6) illustrates the capability of the shortest path method applied to earthquake location to obtain clearly non-Gaussian probability density functions in a traditionally ill-conditioned location problem (Thurber, 1985). The same event is located in the reference model of Israel (I1) but with only the stations north of the epicenter, so that it is far outside the seismograph station array. One can see from Figures 5a and 5b that the confidence regions are deformed and do not show any resemblance with the ellipsoidal regions from linearised theory.

In the last two examples (Table 2, Example 7 and 8, Figure 7 and 8) the ISSN layered, laterally homogeneous velocity model is replaced by a laterally inhomogeneous model, I4, based on refraction observations in the area (Ginzburg *et al.*, 1979; Ginzburg and Folkman, 1980). This model includes dipping interfaces, a velocity gradient in the basement layer and a sharp lateral inhomogeneity for the Faraa-Carmel fault, an active fault branch of the Jordan-Dead Sea fault which is the main left-lateral strike-slip fault in the area (Figure 6). In Example 7, the velocity model I4 is assumed to be exact, that is, no theoretical modeling errors are included. A more realistic estimation of the model uncertainty ($\delta(\frac{1}{c}) = 0.0025$sec/km) is chosen in Example 8. In both cases non-Gaussian probability density functions are obtained and the confidence regions become larger when uncertainties in the velocity model are included.

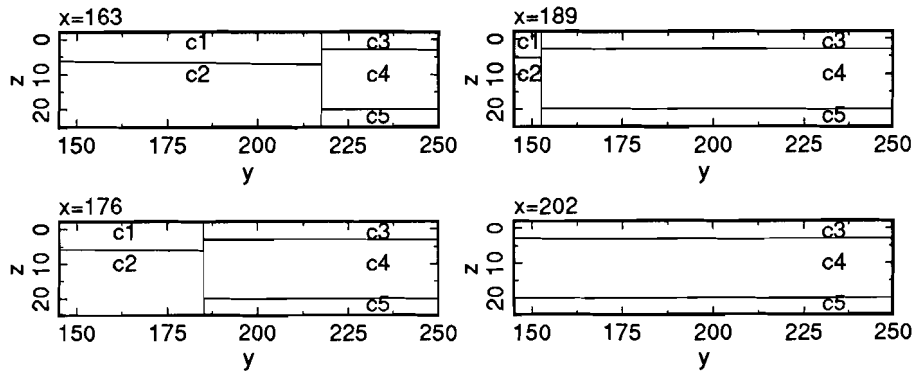

Figure 6. More realistic model of northern Israel (see text: model I4). x-sections at $x = 202$km, $x = 189$km, $x = 176$km *and* $x = 163$km *(south-north direction).* $c1 = c3 = 4.2$km/sec. $c2$ *and* $c4$ *are linear functions of* z *for each* x,y *- location, ranging from* 6.15km/sec *at the first interface to* 6.59km/sec *at the second.* $c5 = 8.0$km/sec. *The interfaces have different forms for two regions, separated by a vertical fault, directed from the north-west to the south-east and given by* $y = -2.5x + 625.0$. *In the north-east part both interfaces are horizontal, at depths of* 3.0km *and* 20.0km *respectively. In the south-west part the first interface is slowly dipping and has the form* $z = -0.04x + 0.0114y + 11.1425$, *while the second interface is below the model and has the form* $z = 0.112x - 0.008y + 10.20$.

Discussion and conclusions

In this chapter it is shown that the shortest path method for seismic ray tracing can be applied advantageously to seismic event location in strongly heterogeneous media. The method is robust, finds the global minimum and provides a measure of reliability in the form of confidence regions. This approach uses the formulation presented by Tarantola and Valette (1982), which states the inverse problem in terms of joint probability density functions.

The advantage of the formulation of Tarantola and Valette (1982) is that a probability density function for the hypocenter location is found, that is, both the most likely location and its uncertainty. The disadvantage is that the probability density function is obtained by integration over a large number of possible locations, requiring an enormous number of travel time calculations. This is not feasible for traditional ray tracing methods in heterogeneous media. It is feasible, however, with the shortest path method which provides ray tracing from one source point to the entire model within reasonable computing time. Moreover, for a given grid and a given station configuration, travel time computations for each grid and station point can be stored. This means that in practical applications of the presented location procedure one computation of all travel times will be sufficient and each earthquake location procedure will be a simple search through travel time tables. The location on one of the grid points can be improved by using a minimisation procedure on the misfit, which is extended by interpolation of the travel times between the grid points.

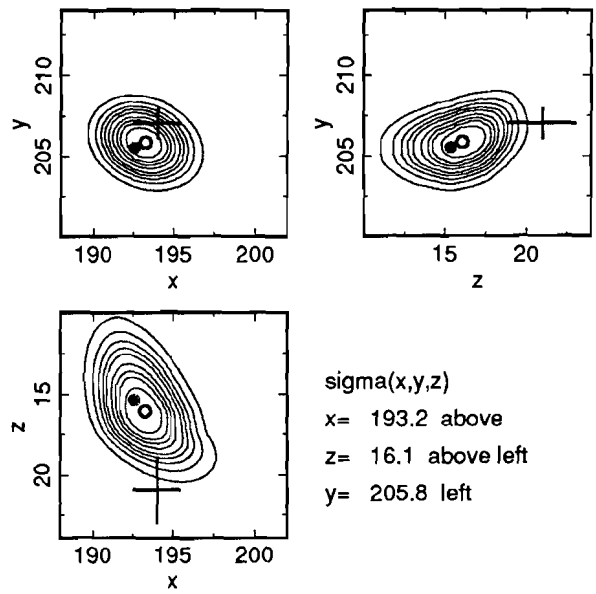

Figure 7. Confidence regions for the hypocenter location in the more realistic model I4, shown in Figure 6. Theoretical modeling errors are not included. See text, Tables 2 and 3, Example 7. Plot symbols and contours are the same as in Figure 4b.

Confidence regions can be determined by integrating the obtained probability density function for the location.

Appendix, origin time estimate

The *a posteriori* probability density function for the origin time is obtained as the integral of $\sigma(\boldsymbol{h}_0)$ in equation (12) over M:

$$\sigma(t_0) = \int_M \sigma(\boldsymbol{h}_0)\mathrm{d}\boldsymbol{x}_0 = \int_M \exp\left[-\frac{1}{2}\sum_{ij} w_{ij}(\tau_i - T(\boldsymbol{x}_0;\boldsymbol{x}_i) - t_0)(\tau_j - T(\boldsymbol{x}_0;\boldsymbol{x}_j) - t_0)\right]\mathrm{d}\boldsymbol{x}_0. \quad \text{(A1)}$$

The origin time can be obtained from a maximum-likelihood estimation:

$$\frac{\partial\sigma(t_0)}{\partial t_0} = 0 \quad \text{(A2)}$$

or:

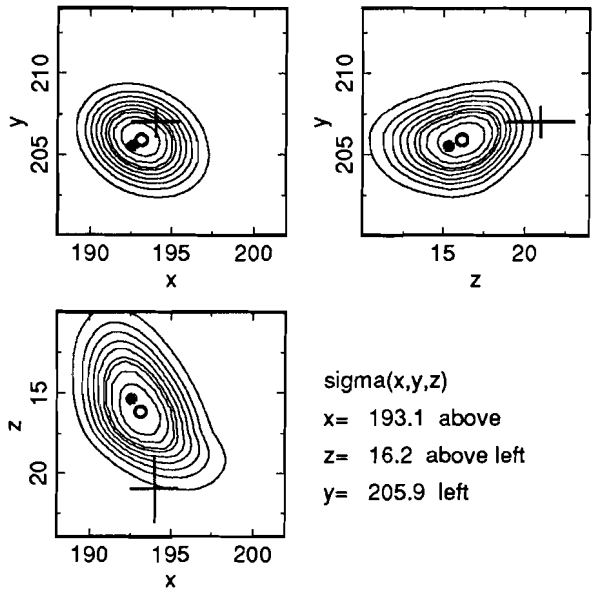

Figure 8. Same as Figure 7, including theoretical modeling errors due to an uncertainty in the slowness of $\delta(\frac{1}{c}) = 0.0025$sec/km.

$$\frac{\partial\sigma(t_0)}{\partial t_0} = \frac{\partial}{\partial t_0}\int_M \sigma(\boldsymbol{h}_0)\mathrm{d}\boldsymbol{x}_0 = \int_M \frac{\partial}{\partial t_0}\sigma(\boldsymbol{h}_0)\mathrm{d}\boldsymbol{x}_0 =$$

$$= \int_M \left[-\frac{1}{2}\sum_{ij} w_{ij}\left[(\tau_i - T(\boldsymbol{x}_0;\boldsymbol{x}_i) - t_0)\frac{\partial(\tau_j - T(\boldsymbol{x}_0;\boldsymbol{x}_j) - t_0)}{\partial t_0} + \right.\right.$$

$$\left.\left. + \frac{\partial(\tau_i - T(\boldsymbol{x}_0;\boldsymbol{x}_i) - t_0)}{\partial t_0}(\tau_j - T(\boldsymbol{x}_0;\boldsymbol{x}_j) - t_0)\right]\right] \times$$

$$\times \exp\left[-\frac{1}{2}\sum_{ij} w_{ij}(\tau_i - T(\boldsymbol{x}_0;\boldsymbol{x}_i) - t_0)(\tau_j - T(\boldsymbol{x}_0;\boldsymbol{x}_j) - t_0)\right]\mathrm{d}\boldsymbol{x}_0 =$$

$$= \int_M \left[\frac{1}{2}\sum_{ij} w_{ij}\left[(\tau_i - T(\boldsymbol{x}_0;\boldsymbol{x}_i) - t_0) + (\tau_j - T(\boldsymbol{x}_0;\boldsymbol{x}_j) - t_0)\right]\right]$$

$$\exp\left[-\frac{1}{2}\sum_{ij} w_{ij}(\tau_i - T(\boldsymbol{x}_0;\boldsymbol{x}_i) - t_0)(\tau_j - T(\boldsymbol{x}_0;\boldsymbol{x}_j) - t_0)\right]\mathrm{d}\boldsymbol{x}_0 = 0. \qquad \text{(A3)}$$

Since the exponential factor is strictly positive, the integral vanishes only if:

$$\sum_{ij} w_{ij}\left[(\tau_i - T(\boldsymbol{x}_0;\boldsymbol{x}_i) - t_0) + (\tau_j - T(\boldsymbol{x}_0;\boldsymbol{x}_j) - t_0)\right] = 0, \tag{A4}$$

so that

$$t_0 = \frac{\sum_{ij} w_{ij}\left[(\tau_i - T(\boldsymbol{x}_0;\boldsymbol{x}_i) + \tau_j - T(\boldsymbol{x}_0;\boldsymbol{x}_j)\right]}{2\sum_{ij} w_{ij}} = \frac{\sum_{ij} w_{ij}(\tau_i - T(\boldsymbol{x}_0;\boldsymbol{x}_i))}{\sum_{ij} w_{ij}} \tag{A5}$$

is the maximum likelihood estimate of the origin time.

CHAPTER 8

Stochastic analysis of arrival times using the shortest path method

Introduction

Recent papers (Frankel and Clayton, 1986; Gudmundsson, Davies and Clayton, 1990; Petersen, 1990; Gudmundsson and Clayton, 1991; Müller, Roth and Korn, 1991) study the relationship between the stochastic properties of the Earth's structure and the statistical characteristics of travel time delays. Frankel and Clayton (1986) use finite-difference simulations of wave propagation in random media to predict the variation of the travel time as a function of parameters describing the random media. Gudmundsson *et al.* (1990) use ray theory in a randomly disturbed Earth to produce travel time delays. They then invert the travel time delays from the ISC catalogue to yield statistical models of the heterogeneity of Earth's mantle. For instance, they conclude that the upper mantle contains small-scale features, with a correlation length smaller than 300km, and that the lower mantle contains variations in the velocity of 0.1% on length scales of about 1000km.

However, Gudmundsson *et al.* (1990) explicitly do not take into account the effect of ray bending or of the associated effect of wave front healing. Wave front healing is a consequence of Huygens' principle: a new wave front is the envelope of the wave fronts of individual imaginary sources on an earlier wave front. It means that delays in the wave front due to the passage of a local low velocity region are healed in the sense that they disappear eventually. The negligence of wave front healing leads to the so-called 'Wielandt effect' in seismic tomography (Wielandt, 1987, see also Chapter 6, §4). This effect causes the velocities in a tomographic image to be biased toward high values.

In this chapter a short survey is given of the effect of wave front healing on the statistical properties of the travel times in random media. The media analysed cover a rectangular, beam-shaped, region and are generated by a random perturbation of a homogeneous reference slowness. This random perturbation is filtered and scaled in order to give the desired smoothness and variance. Shortest path calculations have been done for two different source configurations: a plane wave hitting one of the small sides of the beam and a point source at the same side. In this chapter the term 'wave front' has the meaning of the isochron of the first arrival time; only travel times are involved in the analysis, not wave forms. The shortest path travel times are compared with the Fermat approximation of the travel time, that is the integral over the perturbed slowness along a reference ray in the unperturbed medium. The Fermat approximation of the travel time delay is a linear approximation that does not take into account the wave front healing effect. The travel times of the shortest paths and of the Fermat approximation are calculated at a range of propagation distances and used for a statistical analysis. The random character of the media is somewhat questionable because only one random number generator has been used; strictly speaking, one should repeat the calculations for a large number of different random media with the same statistical properties. This chapter uses the interpretation of randomness of Aki and Richards (1980, Chapter 13). In any case, the calculations are a test of the Fermat approximation and the wave front healing effect.

The tests have been done mainly because the shortest path method is especially suitable for the massive calculations of entire wave fields. The results have a preliminary character; no attempt is made at this stage to draw conclusions about the consequences for a stochastic analysis of global travel time data.

§1 Method

This chapter uses the abbreviations summarised in Table 1.

The first two quantities, *varsp* and *clsp*, describe the statistical properties of the random medium. They are used as an input for the travel time delay calculations. The other six quantities, *avtp*, *vartp*, *cltp*, *avtpf*, *vartpf* and *cltpf*, are the results.

§1.1 Construction of slowness field

A rectangular region with extensions $0 \le x \le 100$, $0 \le y \le 100$ and $0 \le z \le 400$ is covered by a grid with $\vec{n} = (32,32,128)$ (notation from Chapter 6). The reference model is homogeneous with a slowness $u(x,y,z) \equiv 1.0$. The construction of the random perturbation $\delta u(x,y,z)$ on this reference field is as in Frankel and Clayton (1986). A random number generator is used to assign a random number to each grid point. The random field is then Fourier transformed to the three dimensional wave number domain:

$$\delta u(x,y,z) \leftrightarrow \delta U(k_x, k_y, k_z), \tag{1}$$

where the double arrow denotes a Fourier transform pair. The routine for n-dimensional Fourier transforms is taken from Press *et al.* (1986). In the three dimensional wave number domain, the field is filtered with a Gaussian filter in order to get the desired degree of smoothness and scaled in order to get the desired variance. This Fourier transformed, filtered and scaled random field is then used for two purposes. First, it is back transformed and added to the reference slowness field to give the perturbed slowness field in which the shortest path and ray calculations take place. Secondly, it is multiplied with its complex conjugate (denoted with the asterisk *) and back transformed to give the autocorrelation function of the slowness perturbation:

$$AC(x,y,z) \leftrightarrow \delta U(k_x, k_y, k_z)\delta U^*(k_x, k_y, k_z). \tag{2}$$

In general, the autocorrelation function AC is a function of the coordinates of two points

Table 1.

varsp	***var***iance of the *s*lowness ***p***erturbation
clsp	***c***orrelation ***l***ength of the *s*lowness ***p***erturbation
avtp	***av***erage ***t***ravel time ***p***erturbation, calculated with the shortest path method
vartp	***var***iance in the ***t***ravel time ***p***erturbation, calculated with the shortest path method
cltp	***c***orrelation ***l***ength in the ***t***ravel time ***p***erturbation, calculated with the shortest path method
avtpf	***av***erage ***t***ravel time ***p***erturbation, ***F***ermat approximation
vartpf	***var***iance of the ***t***ravel time ***p***erturbation, ***F***ermat approximation
cltpf	***c***orrelation ***l***ength of the ***t***ravel time ***p***erturbation, ***F***ermat approximation

which are given by six variables (x_1, y_1, z_1) and (x_2, y_2, z_2), but in this chapter it is assumed that it is homogeneous, so that it depends only on the differences of the coordinates, $AC(x,y,z) = AC(x_2 - x_1, y_2 - y_1, z_2 - z_1)$, and isotropic, so that it depends only on one variable, $AC(x,y,z) = AC(\sqrt{x^2 + y^2 + z^2}) = AC(r)$ (Sobczyk, 1985). The variance of the slowness perturbation (*varsp*) is then equal to the autocorrelation evaluated at zero, scaled by the number of nodes:

$$varsp = \frac{AC(0)}{n_1 n_2 n_3}. \tag{3}$$

The correlation length of the slowness perturbation (*clsp*) is defined as (after Sobczyk, 1985):

$$clsp = \frac{1}{AC(0)} \int_0^L AC(r)\mathrm{d}r. \tag{4}$$

In Sobczyk's definition $L = \infty$, but as the autocorrelation is given on a finite rectangular grid and is also symmetric with respect to the center of the grid (Press *et al.*, 1986), the integral in (4) is evaluated only to that point.

§1.2 Shortest path calculation and Fermat approximation

The delays in the arrival times that are due to the slowness perturbations described in the previous section and that have been calculated with the shortest path method, have been compared with the delays as derived from the Fermat approximation. In the case of the shortest path calculations the travel times in the disturbed medium are computed with the shortest path method and compared with the travel times in the undisturbed medium. As the undisturbed medium is homogeneous in the examples given in this chapter, the latter travel times are found directly. The Fermat approximation of the disturbed travel time is defined as the travel time in the disturbed medium along the reference ray path in the undisturbed medium. In the case of a homogeneous reference model the Fermat approximation of the delay is just the integral over the slowness perturbation along a straight line.

The initial conditions have been chosen in two different ways. The first option is a plane wave hitting the model boundary at $z = 0.0$ at time $t = 0.0$. The second one is a point source with discrete coordinates (16,16,1) and initial time $t = 0.0$.

The shortest path calculations have been done with forward stars defined by $\vec{f} = (3,3,5)$ and with the weights of the connections evaluated by the advanced line integration, described in (Chapter 6, §3). With such definitions the shortest path travel times are accurate enough for the purposes of this chapter (Chapters 4 and 6), because the travel time perturbations under study are of the order of several percent.

Figures 1 and 2 show a two dimensional version of the calculations of this chapter. In Figure 1, a two dimensional slowness field is shown with slownesses between 0.5 and 1.5, a variance of 3.2×10^{-2} and a correlation length of about 20, sampled on a 50×50 grid. Such a variance is very large compared to the variances used in the three dimensional computations of §2; therefore, the differences between the Fermat delay times and the shortest path delay times are strongly amplified. The plot at the left shows the distortion of

a plane wave by the slowness perturbation, the right plot shows a spherical wave, excited by a point source, both calculated with the shortest path method. In both cases, it is clearly visible, due to the large slowness contrast, that the distortion is not permanent but that the waves heal themselves. The wave front healing is most apparent at the left plot along the line at about $x = 25.0$.

The travel times at $z = 100.0$ are shown in Figure 2 for both waves. The shortest path travel times, shown by a solid line, are systematically shorter than the Fermat travel times, shown by a dashed-dotted line. Also, the variance of the shortest path travel times is smaller than the variance of the Fermat travel times. Both effects are caused by wave front healing; wave front healing results in an earlier arrival of the wave and large oscillations in the wave front are rectified.

In the following section the plane wave and the point source travel times, as computed with the shortest path method, are selected for 32 equidistant z-levels that coincide with the nodes in the grid, from $z = 9.5$ to $z = 400.0$ with step 14.5. For each z-level the travel times in the reference medium are then subtracted from these perturbed travel times in order to give the travel time delays or perturbations. The travel time perturbations are used to evaluate *avtp*, *vartp* and *cltp*. The Fermat approximation of the travel time perturbations is obtained at the same z-level by integration of the slowness perturbation along the appropriate straight lines. It is used, in its turn, to evaluate *avtpf*, *vartpf* and

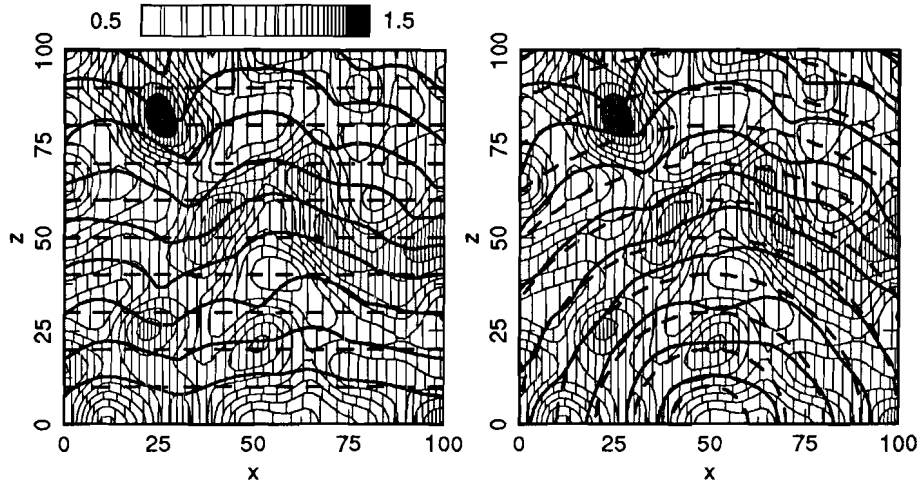

Figure 1. Two dimensional example of wave propagation in a random medium. The contours and shading denote the slowness distribution, the heavy solid lines are the isochrons as computed by the shortest path method, the dashed lines are the isochrons in the reference medium with slowness equal to 1.0. *a) (left) Plane wave. b) (right) Point source.*

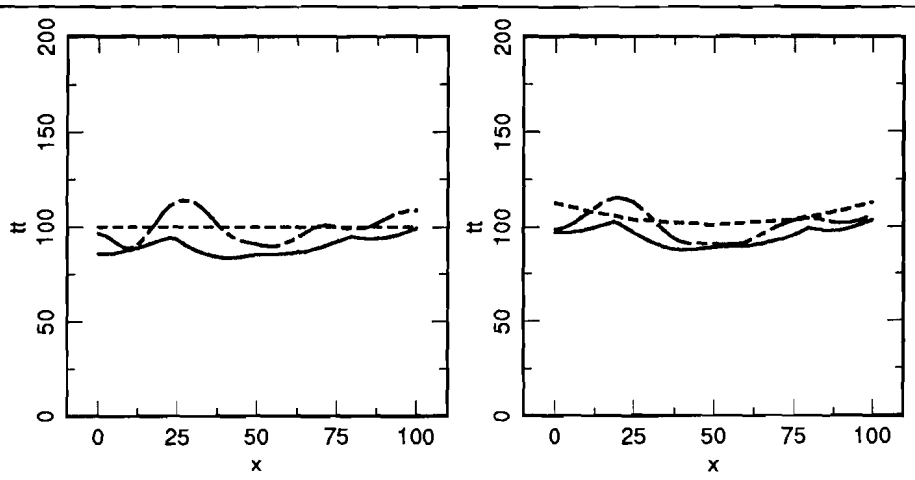

Figure 2. Arrival times at $z = 100.0$ *for the waves in Figure 1. The heavy solid lines are the arrival times as computed by the shortest path method, the dashed-dotted lines are the arrival times according to the Fermat approximation and the dashed lines are the arrival times in the reference medium. a) (left) Plane wave travel times. b) (right) Point source travel times.*

cltpf.

avtp and *avtpf* are obtained by simply averaging the travel time perturbation over all nodes at the z-level. The calculation of *vartp*, *cltp*, *vartpf* and *cltpf* is analogous to the calculation of *varsp* and *clsp*, described in §1.1 the only difference is that the Fourier transforms are now two dimensional and the scale factor for the autocorrelation function is $n_1 n_2$.

§2 Results

This section presents the results of the travel time calculations on 10 different random media. These are characterised by two values of *clsp*, the correlation length of the slowness perturbation, and ten values of *varsp*, the variance of the slowness perturbation, given in Table 2. The variances are chosen in such a way that the standard deviations in the slowness perturbation, equal to the square root of the variances, increase linearly.

In each of these 10 media a plane wave calculation and a point source calculation has been done by means of the shortest path method. The resulting travel times have been selected at 32 different z-levels. These travel times have been used to produce values for *avtp*, *vartp* and *cltp*. The quantities *avtpf*, *vartpf* and *cltpf* are obtained from the Fermat approximation of the travel time perturbation at the same z-levels.

Figures 3 to 8 show results of the plane wave version, Figures 3 to 5 for the smoother medium (*clsp* = 11.37), Figures 6 to 8 for the rougher medium (*clsp* = 6.573).

Table 2.

clsp	standard deviation (= $\sqrt{varsp}$) in %
6.57	0.4637
6.57	0.9274
6.57	1.3910
6.57	1.8547
6.57	2.3184
11.73	0.4921
11.73	0.9843
11.73	1.4765
11.73	1.9688
11.73	2.4609

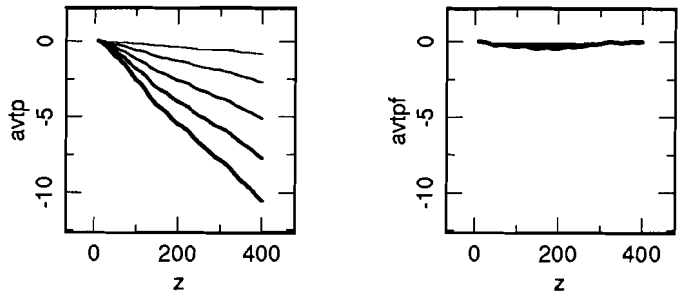

Figure 3. avtp (left) and avtpf (right) as a function of z, plane wave version, clsp = 11.37. *The thickness of the lines is proportional to the variances of the slowness perturbations (varsp) given in Table 2.*

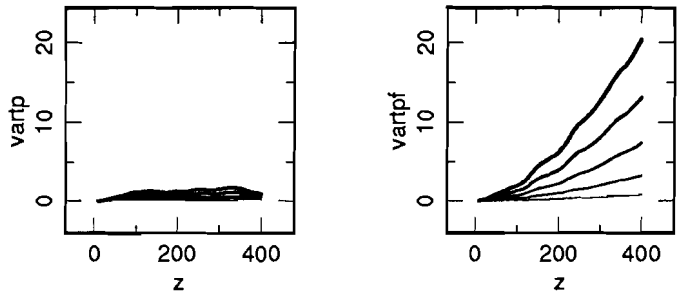

Figure 4. vartp (left) and vartpf (right) as a function of z, plane wave version, clsp = 11.37.

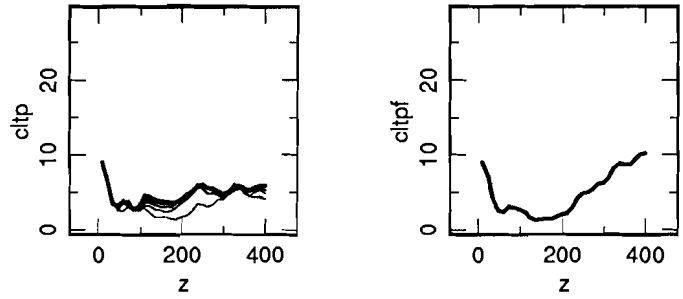

Figure 5. cltp (left) and cltpf (right) as a function of z , plane wave version, clsp = 11.37.

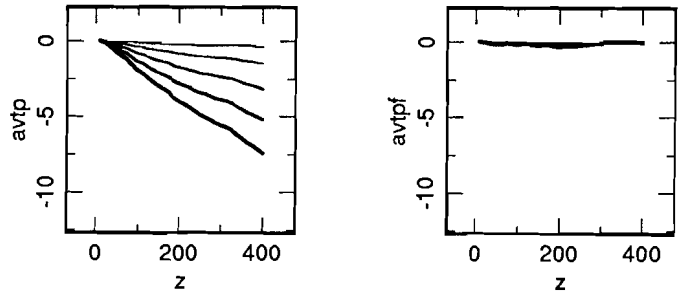

Figure 6. avtp (left) and avtpf (right) as a function of z , plane wave version, clsp = 6.573.

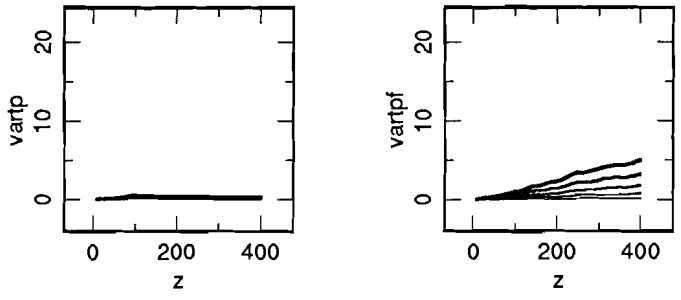

Figure 7. vartp (left) and vartpf (right) as a function of z , plane wave version, clsp = 6.573.

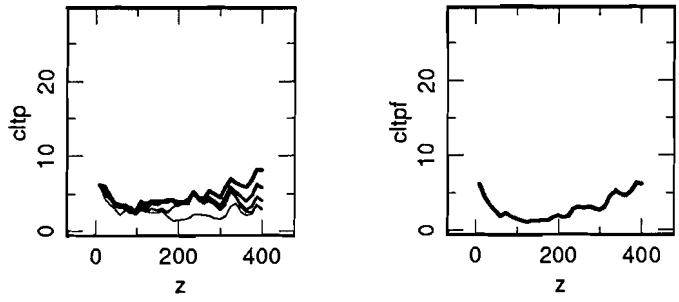

Figure 8. cltp (left) and cltpf (right) as a function of z, plane wave version, clsp = 6.573.

Figures 9 to 14 show results of the point source version of the calculations, Figures 9 to 11 for the smoother medium (*clsp* = 11.37), Figures 12 to 14 for the rougher medium (*clsp* = 6.573).

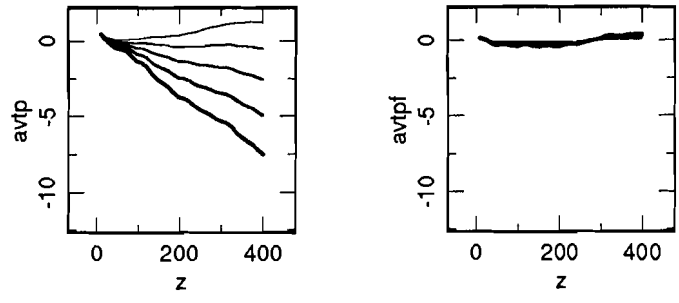

Figure 9. avtp (left) and avtpf (right) as a function of z, point source version, clsp = 11.37. *The thickness of the lines is proportional to the variances of the slowness perturbations (varsp) given in Table 2.*

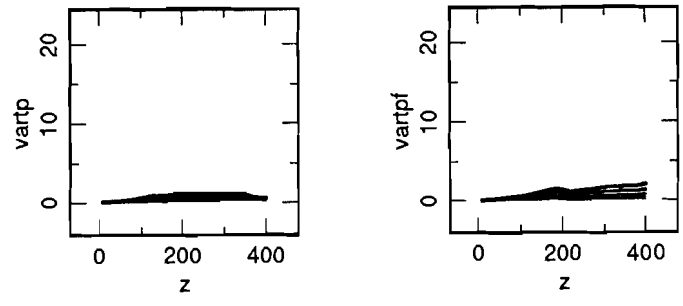

Figure 10. vartp (left) and vartpf (right) as a function of z, point source version, clsp = 11.37.

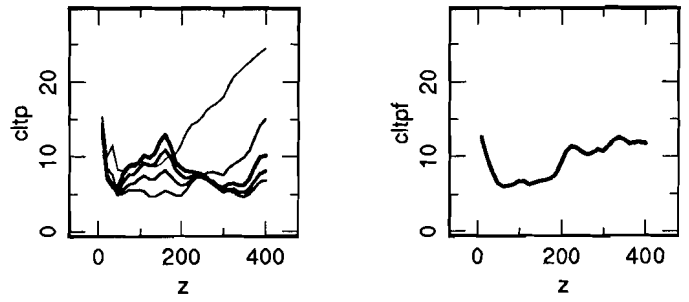

Figure 11. cltp (left) and cltpf (right) as a function of z, point source version, clsp = 11.37.

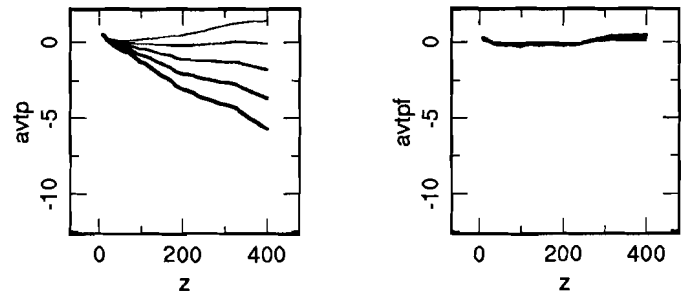

Figure 12. avtp (left) and avtpf (right) as a function of z, point source version, clsp = 6.573.

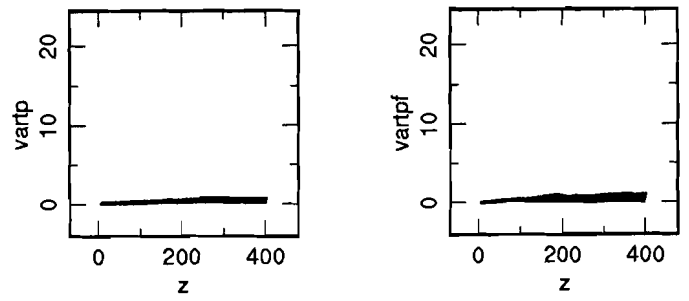

Figure 13. vartp (left) and vartpf (right) as a function of z, point source version, clsp = 6.573.

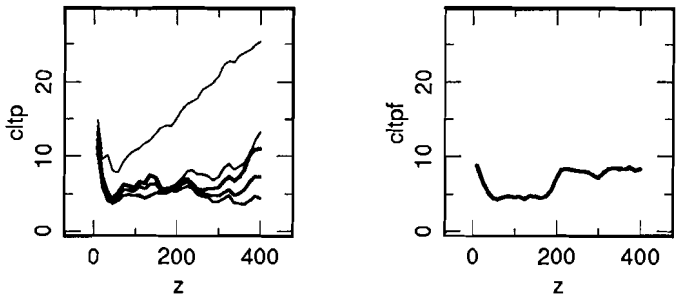

Figure 14. cltp (left) and cltpf (right) as a function of z, point source version, clsp = 6.573.

§3 Discussion

The main difference between the Fermat approximation of the travel time delay and the shortest path approximation is that the first one is linearised, or first order, in the slowness perturbation and that the second one is a numerical approximation of the complete, nonlinear, effect of the slowness perturbation on the travel times. This nonlinear effect is manifest in the wave front healing.

§3.1 Average travel time perturbation

The plots of the average travel time perturbations, *avtp* and *avtpf*, Figures 3, 6, 9 and 12, show an obvious difference between the Fermat delays and the shortest path delays. The averaged Fermat delays are very small for all calculations, do not strongly depend on the z-coordinate and increase with increasing slowness perturbation *varsp*. This is likely, because the slowness perturbation is chosen in such a way that the average slowness on the whole grid is equal to one. The Fermat approximation of the travel time perturbation is an integral of the slowness perturbation along a straight line, so when it is averaged over all nodes at a z-level it cannot expected to be large. The fact that *avtpf* is apparently not exactly zero at $z = 400.0$ may be caused by the calculation procedure: integration of the slowness perturbation along straight lines to $z = 400.0$ and then summation of these integrals over all nodes at $z = 400.0$. Such a procedure is slightly different from averaging the slowness perturbation over the whole grid.

The shortest path delays behave totally differently. In all cases, they decrease monotonically with z and with *varsp*. Apart from the point source calculations in the media with the smallest *varsp*, shown in Figure 9 and 12 by the thinnest lines, all of the shortest path delays are negative. This means that the rays arrive earlier than predicted by the Fermat approximation, which is a consequence of the wave front healing. Also, the delay accumulates along the ray and increases with the amplitude of the slowness variations.

§3.2 Variance in the travel time perturbation

The variance in the Fermat approximation of the travel time perturbation *vartpf* increases monotonically with the variance of the slowness perturbation *varsp* and almost monotonically with z (Figures 4, 7, 10 and 13). These effects can be expected because *vartpf* is the integral of the slowness perturbation, squared and summed over all nodes at a z-level, so that a larger amplitude in the slowness variations necessarily implies a larger *vartpf*. The increase of *vartpf* with z is not quite monotonical as can be seen in Figure 7 (plane wave version) but especially in Figures 10 and 13 (point source version). In any case, the Fermat delays oscillate more and more along the ray paths and with increasing slowness perturbations.

This is not the case with the shortest path delays as is clearly visible in Figures 4, 7, 10 and 13. The variances in the shortest path delay times vary with z but not monotonically; in the plane wave calculations, they are even very small and almost constant compared to *vartpf*. These effects are clearly caused by wave front healing. Parts of the wave front that stay behind because they traveled through high slowness regions, are overtaken by the neighbouring parts of the wave front. As the new wave front is the envelope of individual wave fronts of imaginary sources on the old wave front, gaps are healed and the variance remains bounded. In both the plane wave version and point source version of the calculations it appears that the wave fronts can restore themselves from distortions caused by the slowness perturbations but that they are deformed permanently in the Fermat approximation.

§3.3 Correlation length of the travel time perturbation

The correlation length of the travel time delays is more difficult to interpret. In the Fermat approximation the correlation lengths *cltpf* are equal for all slowness perturbations. This is caused by the fact that the Fermat delays for different slowness perturbations are simply scaled versions of each other, due to the linearity of the Fermat approximation. The variation of *cltpf* with z is rather accidental; the smoothness of the Fermat travel times at different z-levels depends on the smoothness of the medium behind the wave front which is assumed to be uniform.

The correlation lengths of the shortest path delays show a rather chaotic behaviour, apart from the point source calculation in the medium with the smallest perturbation. It appears that the shortest path travel times do not become smoother after the wave front healing. This can be understood from Figures 1 and 2. The wave front healing, as seen in Figure 1 at $(x,z) = (25.0,75.0)$, left plot and right plot, means that the parts of the wave front that travel faster than the delayed part propagate towards each other and make a kink. Such a kink, also visible in Figure 2 at $x = 20.0$ in both plots, results in high frequent components in the Fourier transform of the travel time delays and, therefore, in a small correlation length. Between the kinks the shortest path travel time delays seem to be smoother than the Fermat delays.

§3.4 The effect of the smoothness of the medium

The calculations in §2 have been done for media with two different degrees of smoothness given by correlation lengths *clsp* = 6.573 and *clsp* = 11.37. This may not be enough to study the effect of the smoothness on the travel time perturbations but there are conclusions that can be drawn. The averaged shortest path delay *avtp*, for instance, appears to be more negative in the smoother medium, given by *clsp* = 11.37, than in the rougher medium, given by *clsp* = 6.573 (compare Figure 3 with 6 and Figure 9 with 12), even when it is taken into account that the values of *varsp* are not equal for different *clsp* (Table 2). It seems that wave front distortion and healing is larger in a smoother medium. Also, the variance of the travel time perturbation is larger in the smoother medium for both the shortest path and the Fermat approximation (*vartp* and *vartpf*). For the shortest path travel times this may be explained by the fact that a smoother medium allows for more wave front distortion and, therefore, wave front healing.

§3.5 Difference between the plane wave and the point source version

The plane wave version and the point source version of the calculations are fundamentally different. In the point source version there is only one source point and the travel time delays in the shortest path and the Fermat approximation are evaluated for the same source receiver pair. In the plane wave version, on the other hand, the distortion of the plane waves implies that a receiver point does not necessarily lie on the same (x,y) location as the source point from where the ray path originates. This means that the Fermat delay and the shortest path delay are generally not for the same source receiver pair. Yet, the plane wave calculations make sense because they still account for the error that is made by the Fermat approximation. Another point of view is that the point source version gives the results only for one individual wave. When the point source happens to lie in a strongly anomalous slowness region, the conclusions about the statistical properties of the delays may be less valuable. The plane wave version provides better statistics because it gives the results of the attributions of many different sources, although they interact with each other. A third way to interpret the difference between the plane wave version and the point source version is that the point source version accounts for phenomena near the source and the plane wave version for the far field.

Despite the different character of the two versions the results show no apparent differences. The shortest path delay times in the medium with the smallest perturbation are positive (Figures 9 and 12), but this may be caused by the position of the source in a high slowness region. Only the variances of the Fermat delays are much larger for the plane wave version (Figures 4 and 10). In all plots it appears that the above conclusions about the statistical properties of the delays (§3.1 to §3.4) apply for the plane wave version as well as for the point source version of the calculations but that the effects are smaller in the latter version.

References

Aho, A.V., Hopcroft, J.E., and Ullman, J.D., *The design and analysis of computer algorithms*, Addison-Wesley, Reading, MA, 1974.

Aki, K., and Richards, P., *Quantitative Seismology: Theory and Methods*, W.H. Freeman and Company, 1980.

Anderson, K.R., Robust earthquake location using M-estimates, *Phys. of the Earth and Plan. Int.*, 30, 119-130, 1982.

Barsky, B.A., *Computer graphics and geometric modeling using Beta-splines*, Computer & science workbench, Springer-Verlag, Berlin, 1988.

Bellman, R., On a routing problem, *Quarterly of Applied Mathematics*, 16, 88-90, 1958.

Bender, C.M., and Orszag, S.A., *Advanced mathematical methods for scientists and engineers*, McGraw-Hill, 1984.

Berkhout, A.J., *Seismic migration, Imaging of acoustic energy by wave field extrapolation, A. Theoretical aspects*, and *B. Practical aspects*, Developments in solid Earth geophysics, Second revised and enlarged edition, 1982.

Buland, R., The mechanics of locating earthquakes, *Bull. of the Seism. Soc. of Am.*, 66, 173-187, 1976.

Červený, V., Ray tracing algorithms in three-dimensional laterally varying layered structures, in *Seismic Tomography, with applications in global seismology and exploration geophysics*, Ed. G. Nolet, D. Reidel Publishing Company, 99-133, 1987.

Chander, R., On tracing seismic rays with specified end points, *J. Geoph.*, 41, 173-177, 1975.

Chin, R.C.Y., Hedstrom, G., and Thigpen, L., Numerical Methods in Seismology, *J. of Comp. Phys.*, 54, 18-56, 1984.

Conte, S.D., and De Boor, C., *Elementary Numerical Analysis*, McGraw-Hill, 1980.

Courant, R., and Hilbert, D., *Methoden der Mathematischen Physik*, Dritte Auflage, Springer, 1967.

Denardo, E.V., and Fox, B.L., Shortest-route methods: 1. Reaching, pruning, and buckets, *Operations Research*, 27, 161-186, 1979.

Deo, N., and Pang, C., Shortest-path algorithms: taxonomy and annotation, *Networks*, 14, 275-323, 1984.

Dial, R., Glover, F., Karney, D., and Klingman, D., A computational analysis of alternative algorithms and labeling techniques for finding shortest path trees, *Networks*, 9, 215-248, 1979.

Dijkstra, E.W., A note on two problems in connexion with graphs, *Numer. Math.*, 1, 269-271, 1959.

Dziewonski, A.M., Hales, A.L., and Lapwood, E.R., Parametrically simple Earth models consistent with geophysical data, *Phys. of the Earth and Plan. Int.*, 10, 12-48, 1975.

Fletcher, R., and Reeves, C.M., Function minimisation by conjugate gradients, *The Computer Journal*, 7, 149-154, 1965.

Ford, L.R., *Network flow theory*, Report No. P-293, Rand Corporation, 1956.

Fox, B.L., Data structures and computer science techniques in operations research, *Operations Research*, 26, 686-717, 1978.

Frankel, A., and Clayton, R.W., Finite Difference Simulations of Seismic Scattering:

Implications for the Propagation of Short-Period Seismic Waves in the Crust and Models of Crustal Heterogeneity, *J. of Geophys. Res.*, Vol. 91, No. B6, 6465-6489, 1986.

Fredman, M.L., and Tarjan, R.E., Fibonacci heaps and their uses in improved network optimisation algorithms, *Proceedings of the 25th annual IEEE Symposium on Foundations of Computer Science*, 1984.

Funk, P., *Variationsrechnung und ihre Anwendung in Physik und Technik*, Springer Verlag, 1962.

Gallo, G., Pallottino, S., Ruggeri, C., and Storchi, G., *Shortest paths: A bibliography*, Rapporto 27.82, Consiglio Nazionale delle Ricerche - Progetto Finalizzato Informatica - SOFMAT, Rome, 1982.

Gallo, G., and Pallottino, S., Shortest path methods: A unifying approach, *Mathematical Programming Study*, 26, 38-64, 1986.

Ginzburg, A., Makris, J., Fuchs, K., Prodehl, C., Kaminski, W., and Amitai, U., A seismic study of the crust and upper mantle of the Jordan-Dead Sea rift and their transition toward the Mediterranean sea, *J. Geophys. Res.*, 84, 1569-1582, 1979.

Ginzburg, A., and Folkman, Y., The crustal structure between the Dead Sea rift and the Mediterranean, *Earth Plan. Science Lett.*, 511, 181-188, 1980.

Gjoystdal, H., Reinhardsen, J.E., and Åstebol, K., Computer representation of complex 3D geological structures using a new 'solid modeling' technique, *Geoph. Prosp.*, 33, 1195-1211, 1985.

Gudmundsson, O., Davies, J.H., and Clayton, R.W., Stochastic analysis of global travel time data: mantle heterogeneity and random errors in the ISC data, *Geophys. J. Int.*, 102, 25-43, 1990.

Gudmundsson, O., and Clayton, R.W., A 2-D synthetic study of global traveltime tomography, *Geoph. J. Int.*, 106, 53-65, 1991.

Hansen, P., An $O(m.\log D)$ algorithm for shortest paths, *Discrete Applied Mathematics*, 2, 151-153, 1980.

Hirata, N., and Matsu'ura, M., Maximum-likelihood estimation of hypocenter with origin time eliminated using nonlinear inversion technique, *Phys. of the Earth and Plan. Int.*, 47, 50-61, 1987.

Jeffreys, H., On travel times in seismology, *Bur. Centr. Seism. Trav. S.*, 14, 3-86 (reprinted in *'The collected papers of Sir Harold Jeffreys'*, 2, Gordon and Breach, London, 1973), 1936.

Johnson, D.B., A note on Dijkstra's shortest path algorithm, *Journal of the ACM*, 20, 385-388, 1973.

Johnson, D.B., Efficient algorithms for shortest paths in sparse networks, *Journal of the ACM*, 24, 1-13, 1977.

Julian, B.R., and Gubbins, D., Three-Dimensional Seismic Ray Tracing, *J. Geoph.*, 43, 95-113, 1977.

Karlsson, R.G., and Poblete, P.V., An $O(m\log\log D)$ algorithm for shortest paths, *Discrete Applied Mathematics*, 6, 91-93, 1983.

Keller, H.B., Accurate difference methods for nonlinear two-point boundary value problems, *SIAM J. Numer. Anal.*, 11, 2, 1974.

Keller, J.B., Rays, waves and asymptotics, *Bull. of the Am. Math. Soc.*, 84, 5, 727-749, 1978.

Knuth, D.E., *The art of computer programming, Vol.3: sorting and searching*, Addison-Wesley, Reading, MA, 1968.

Lee, W.H.K. and Stewart, S.W., Principles and applications of microearthquake networks, in: *Advances in Geophysics*, Academic Press, New York, 130-139, 1981.

Marquering, H.A., *A finite-difference simulation of acoustic waves: a tool for investigating the effect of diffractions on the picking of the arrival time*, unpublished M.Sc. thesis, Utrecht, 1991.

Matsuoka, T., and Ezaka, T., Ray tracing using reciprocity, *Soc. of Exploration Geophysicists 1990 Annual Meeting, Expanded Abstracts*, 1028-1031, 1990.

Mitchell, J.R., and Dickof, P., A comparison of line integral algorithms, *Computers in Physics* , vol. 4, no. 2, 166-172, 1990.

Moore, E.F., The shortest path through a maze, *Proceedings of the International Symposium on Theory of Switching (1957)*, Harvard University Press, Cambridge, MA, 285-292, 1959.

Moser, T.J., Nolet, G., and Snieder, R., Ray bending revisited, *Bull. of the Seism. Soc. of Am.*, to appear in 1992, also Chapter 5 of this thesis.

Moser, T.J., Shortest path calculation of seismic rays, *Geophysics*, 56, 59-67, 1991, also Chapter 1 of this thesis.

Moser, T.J., *The comparison of the shortest path method with industrial ray tracing software*, Internal Report, Utrecht, 1990.

Mufti, I.R., Large-scale three-dimensional seismic models and their interpretive significance, *Geophysics*, 43, 1166-1182, 1990.

Müller, G., Roth, M., and Korn, M., Seismic-wave traveltimes in random media, submitted to *Geoph. J. Int.*, 1991.

Nakanishi, I., and Yamaguchi, K., A numerical experiment on nonlinear image reconstruction from first-arrival times for two-dimensional island arc structure, *J. Phys. Earth*, 34, 195-201, 1986.

Nelder, J.A., and Mead, R., A simplex method for function minimisation, *The Computer Journal*, 7, 308-313, 1965.

Nelson, G.D., and Vidale, J.E., Earthquake locations by 3-D finite-difference travel times, *Bull. of the Seism. Soc. of Am.*, 80, 395-410, 1990.

Newman, W.M., and Sproull, R.F., *Principles of Interactive Computer Graphics*, second edition, Computer science series, McGraw-Hill, 1981.

Nolet, G., Seismic wave propagation and seismic tomography, in *Seismic Tomography, with applications in global seismology and exploration geophysics*, ed. G. Nolet, D. Reidel Publishing Company, 1-23, 1987.

Noshita, K., A theorem on the expected complexity of Dijkstra's shortest path algorithm, *Journal of Algorithms*, 6, 400-408, 1985.

Pape, U., Implementation and efficiency of Moore-algorithms for the shortest route problem, *Math. Programming*, 7, 212-222, 1974.

Pavlis, G.L., Appraising earthquake hypocenter location errors: a complete, practical approach for single-event locations, *Bull. of the Seism. Soc. of Am.*, 76, 1699-1717, 1986.

Pereyra, V., Lee, W.H.K., and Keller, H.B., Solving two-point seismic ray tracing problems in a heterogeneous medium, part 1, A general adaptive finite difference method, *Bull. of the Seism. Soc. of Am.*, 70, 1, 79-99, 1980.

Petersen, N.V., Inverse kinematic problem for a random gradient medium in geometric optics approximation, *Pure Appl. Geophys.*, 132, 417-437, 1990.

Podvin, P., and Lecomte, I., Finite difference computation of traveltimes in very contrasted velocity models: a massively parallel approach and its associated tools, *Geophys. J. Int.*, 105, 271-284, 1991.

Press, H.P, Flannery, B.P., Teukolsky, S.A., and Vetterling, W.T., *Numerical Recipes, The Art of Scientific Computing*, Cambridge University Press, 1986.

Prothero, W.A., Taylor, W.J., and Eickemeyer, J.A., A fast, two-point, three-dimensional ray tracing algorithm using a simple step search method, *Bull. of the Seism. Soc. of Am.*, 78, 3, 1190-1198, 1988.

Qin, F., Olsen, K.B., Luo, Y., and Schuster, G.T., Solution of the eikonal equation by a finite difference method, *Soc. of Exploration Geophysicists 1990 Annual Meeting, Expanded Abstracts*, 1004-1007, 1990.

Rabinowitz, N., Microearthquake location by means of nonlinear simplex procedure, *Bull. of the Seism. Soc. of Am.*, 78, 380-384, 1988.

Reinsch, C.H., Smoothing by Spline Functions, *Numerische Mathematik*, 10, 177-183, 1967.

Rowlett, H., and Forsyth, D.W., Recent faulting and microearthquakes at the intersections of the Vema fracture zone and the Mid-Atlantic ridge, *J. of Geoph. Res.*, 89, B7, 6079-6094, 1984.

Saito, H., 3-D ray tracing method based on Huygens' principle, *Soc. of Exploration Geophysicists 1990 Annual Meeting, Expanded Abstracts*, 1024-1027, 1990.

Saito, H., Traveltimes and raypaths fo first arrival seismic waves: computation method based on Huygens' principle, *Soc. of Exploration Geophysicists 1989 Annual Meeting, Expanded Abstracts*, 244-247, 1989.

Sambridge, M.S., and Kennett, B.L.N., A novel method of hypocenter location, *Geoph. J. R. Astr. Soc.*, 87, 679-697, 1986.

Seismological Bulletin No.8., *Earthquakes in and around Israel during 1989*, Institute for Petroleum Research and Geophysics, Report Z1/567/79(72), 1990.

Shapira, A., *A guide for using program LME83*, IPRG Report Z1/567/79(16), 1983.

Sobczyk, K., *Stochastic wave propagation*, Fundamental studies in engineering 6, Elsevier, 1985.

Stoer, J., and Bulirsch, R., *Introduction to Numerical Analysis*, Springer-Verlag New York, Inc., 1980.

Tarantola, A., *Inverse problem theory. Methods for data fitting and model parameter estimation*, Elsevier, Amsterdam, 1987.

Tarantola, A., and Valette, B., Inverse problems = quest for information, *J. Geoph.*, 50, 159-170, 1982.

Tarjan, R.E., Complexity of combinatorial algorithms, *SIAM Review* , 20, 457-491, 1978.

Thurber, C.H., Earthquake location and three-dimensional crustal structure in the Coyote Lake area, Central California, *J. of Geoph. Res.*, 88, 8226-8236, 1983.

Thurber, C.H., Nonlinear earthquake location: theory and examples, *Bull. of the Seism. Soc. of Am.*, 75, 779-790, 1985.

Thurber, C.H., and Ellsworth, W.L., Rapid solution of ray tracing problems in heterogeneous media, *Bull. of the Seism. Soc. of Am.*, 70, 1137-1148, 1980.

Um, J., and Thurber, C.H., A fast algorithm for two-point seismic ray tracing, *Bull. of the Seism. Soc. of Am.*, 77, 972-986, 1987.

Van Emde Boas, P., Kaas, R., and Zijlstra, E., Design and implementation of an efficient priority queue, *Math. Systems Theory*, 10, 99-127, 1977.

Van Trier, J., and Symes, W.W., Upwind finite-difference calculation of traveltimes, *Geophysics*, 56, 812-821, 1991.

Vidale, J., Finite-difference calculation of travel times in 3D, *Geophysics*, 55, 521-526, 1990.

Vidale, J., Finite-difference calculation of travel times, *Bull. of the Seism. Soc. of Am.*, 78, 2062-2076, 1988.

Vlaar, N.J., Ray theory for an anisotropic inhomogeneous elastic medium, *Bull. of the Seism. Soc. of Am.*, 59, 2053-2072, 1968.

Wesson, R.L., Travel-time inversion for laterally inhomogeneous crustal velocity models, *Bull. of the Seism. Soc. of Am.*, 61, 729-746, 1971.

Wielandt, E., On the validity of the ray approximation for interpreting delay times, in *Seismic Tomography, with applications in global seismology and exploration geophysics*, ed. G.Nolet, D. Reidel Publishing Company, 1-23, 1987.

Yen, J.Y., *A shortest path algorithm*, Ph.D. dissertation, University of California, Berkeley, 1970.

Summary

This thesis is concerned with the shortest path method for seismic ray tracing. The shortest path method is based on Fermat's minimum travel time principle and the theory on shortest paths in networks. To construct seismic ray paths in a complicated velocity model the relevant part of the Earth is covered by a large number of nodes that are connected when they are in a close neighbourhood of each other. Such a structure is mathematically referred to as a graph. It is turned into a network when weights are assigned to each connection. When the weights are chosen equal to the travel time of a seismic wave along them, the shortest paths in the network, defined as sequences of connections with minimum total weights, can be used to approximate seismic ray paths. The travel times along them can be used as approximations of the travel times between source and receiver pairs.

The shortest path method is a flexible method to calculate seismic ray paths and shortest travel times. The abstract formulation of the network theory allows for two dimensional as well as for three dimensional computations without any modifications of the software. The shortest path algorithm that appears to be the most suitable for seismic ray tracing is Dijkstra's algorithm. This algorithm works with two sets of nodes: a set with nodes for which the shortest path travel time is definitely known and a set nodes for which the travel times have only tentative values. The first set is empty at the beginning of the algorithm and is iteratively extended by adding nodes for which the tentative travel times are made definite. The fastest version of Dijkstra's algorithm, applied to ray tracing, is a version in which the tentative travel times are ordered as a heap. The computation time for this version is proportional to $n \log n$, where n is the number of nodes. The Dijkstra algorithm constructs the shortest paths and the travel times along them simultaneously to all nodes of the network. For seismic ray tracing this means that the method constructs a global ray field to all points in space, so there are no problems with the convergence of trial rays towards a receiver as with the shooting method. Also, the shortest path method finds the absolute minimal travel time path instead of getting 'stuck' in a local minimal travel time path, which is a major problem with the bending method.

One can also ask for constrained shortest paths, namely paths that are constrained to visit a specified set of nodes. The algorithm to solve this constrained shortest path problem can be used to calculate later arrivals in the seismogram when the specified nodes are chosen on an interface or scatterer. The constrained shortest path algorithm is based on a generalisation of the Dijkstra algorithm which consists in predestinating the travel times of chosen nodes. This generalisation allows for fitting together single source nodes to more complicated sources. Each additional reflection requires one extra run of the generalised Dijkstra algorithm. As a byproduct other experiments can be carried out like migration or the exploding of a reflector.

The accuracy of the shortest path method appears to be dependent on the number of nodes and the number of connections per node. It is shown that the travel time error dependency on both the coordinate discretisation and the angle discretisation is quadratic. The error in the ray geometry is one order of magnitude larger than the travel time error due to Fermat's principle. With the help of this relationship between the travel time error and the network parameters it is possible to estimate and control the accuracy in relation to the computation time. Also, extrapolation of the results to finer networks is possible due to this error relationship. A standard uniform error of less than 0.1% in the travel time can be reached

for moderately complicated velocity models with networks of moderate size. When this may be not enough for a selected set of ray paths, additional bending can be applied to improve the travel time and the ray path. In fact, the shortest path method is especially fit to be combined with the bending method because for adequate networks the shortest paths are safe initial guesses that generally lie close to the global minimum travel time path.

The additional bending method is based on three ingredients: 1. the ray paths are parametrised by their coordinates (preferably the coordinates of the nodes of the shortest path that has to be improved); 2. the Conjugate Gradients method for minimisation is used to find the minimum travel time path; 3. Beta-splines are used to provide an arbitrarily accurate integration along the candidate paths. The Conjugate Gradients method for minimisation using derivative information is suitable because it requires the storage of only four vectors of the size of the storage of the ray itself and it employs the structure of the travel time function. For the first few iterations it can be advantageous to use a pseudo second order search direction obtained from an approximation of the inversion of the Hessian of the travel time. The points of the ray path should be interpolated during the minimisation process, first because it avoids inaccuracies in concave slowness regions and secondly because interpolation results in both a considerably higher accuracy and efficiency. Beta-splines are particularly suitable as an interpolation scheme because they have several properties, like the local control and convex hull properties, that are in favour of ray bending and because they provide extra control on the curves by means of the shape parameters. Ray bending in the presence of discontinuities in the velocity field can be done by transforming the location parameters of a point of a ray in such a way that it is forced to lie on the discontinuity. With the additional bending method there are no restrictions to the initial guess path that can be chosen because divergence is impossible, but it is not clear whether the resulting minimal travel time path is a global minimal path or only a local minimal path.

The shortest path method has been compared with other methods for forward seismic modeling. First, it has been compared with ray tracing software developed at Shell. This comparison shows the travel times to agree within only 1.0% instead of the required 0.1%. The differences can be explained only partly by the shortest path error characteristics and other known effects; the remaining discrepancy could not be further investigated due to the confidential nature of the Shell software. Also, the efficiency of the shortest path method cannot be compared with the Shell software for the same reasons. Yet, the comparison has given suggestions to improve the shortest path travel times by taking into account the velocity variations along the connections of the network. Both the efficiency and the accuracy can be enhanced tremendously for the shortest path calculations on grid networks by invoking advanced line integration schemes that replace the floating-point manipulations by integer-arithmetic manipulations. The claim that travel times in shadow zones are found correctly by the shortest path method, when their corresponding amplitude is physically observable, can be supported by finite difference calculations. It is shown that the combination of the shortest path and the additional bending methods allows for an efficient and robust calculation of the ray paths and the travel times along them. For a correct choice of the parameters controlling the bending process the error in the travel times is below 0.1%.

As an application of the shortest path method it is shown that it can be used advantageously for seismic event location in strongly heterogeneous media. It is combined with a

formulation of the inverse problem in terms of joint probability density functions. The combined method is robust, finds the global minimum of the misfit and provides a measure of reliability in the form of confidence regions.

As a second application the effect of wave front healing on the statistical properties of the travel times in random media has been investigated. The shortest path travel times are compared with the Fermat approximation of the travel time, that is the integral over the perturbed slowness along a reference ray in the unperturbed medium. The Fermat approximation of the travel time delay is a linear approximation that does not take into account the wave front healing effect. The conclusions are that the shortest path travel time delays are systematically smaller than zero, while the Fermat delays are zero on average. Furthermore, it appears that the variance in the shortest path delays is much smaller than can be expected from the Fermat approximation.

Samenvatting

Kennis over de voortplanting van seismische golven speelt een centrale rol in seismisch modelleren, zowel op een locale schaal, zoals in de exploratieseismiek, als op een globale schaal, zoals in de seismologie. Wanneer de voortplanting van seismische golven bekend is in een model van de aardstructuur, kan deze vergeleken worden met observaties aan de aardoppervlakte. Zulk een vergelijking leidt dan tot een uitspraak over de gelijkenis tussen het model en de werkelijke aardstructuur.

De voortplanting van seismische golven is echter zeer gecompliceerd, zelfs in eenvoudige media. De golven worden vervormd door graduele overgangen in het medium en teruggekaatst en gebroken door discontinuïteiten. Interferentie- en focusseringseffecten compliceren hun vorm nog verder. Er is een grote hoeveelheid methoden ontwikkeld om deze golfverschijnselen te beschrijven, waarvan seismische *ray tracing* er één is. *Ray tracing* maakt gebruik van seismische stralen, lijnen waarlangs de (hoogfrequente) seismische energie zich voortplant. Deze stralen zijn alleen een wiskundige idealisatie, zonder physische betekenis, en blijken van groot practisch nut.

Dit proefschrift heeft betrekking op de kortste-routemethode voor seismische *ray tracing*. De kortste-routemethode is gebaseerd op het minimumreistijdprincipe van Fermat en op de theorie van kortste routes in netwerken. Voor de constructie van seismische stralen in een gecompliceerd snelheidsmodel wordt het betreffende gedeelte van de Aarde bedekt met een groot aantal punten, die met elkaar verbonden zijn als ze dicht bij elkaar liggen. Zo'n structuur van punten en verbindingen wordt een graaf genoemd. Het wordt een netwerk als bovendien gewichten worden toegekend aan de verbindingen. Wanneer die gewichten worden gedefinieerd als de reistijd van een seismische golf langs de verbinding kunnen de kortste routes in het netwerk, gedefinieerd als rijen opeenvolgende verbindingen met minimaal totaal gewicht tussen de eindpunten, gebruikt worden om seismische stralenpaden te benaderen. De reistijden erlangs kunnen evenzo gebruikt worden om de reistijden tussen bron- en ontvangerparen te benaderen.

De kortste-routemethode is een flexibele methode om seismische stralen en reistijden te berekenen. De abstracte formulering van de netwerktheorie maakt zowel twee- als driedimensionale berekeningen mogelijk zonder aanpassingen van de algorithmen. Het algorithme van Dijkstra blijkt het geschiktst te zijn voor seismische *ray tracing*. Dit algorithme definieert twee verzamelingen punten: een verzameling van punten waarvoor de reistijd langs kortste routes definitief bekend is en een verzameling van punten waarvoor de reistijden alleen nog voorlopige waarden hebben. De eerste verzameling is leeg bij het begin van het algorithme en wordt in een iteratief proces uitgebreid door punten toe te voegen uit de tweede verzameling, waarvan de voorlopige reistijden definitief geworden zijn. De snelste versie van het Dijkstraälgorithme, toegepast op *ray tracing*, is een versie waarbij de tweede verzameling, van reistijden met nog slechts een voorlopige waarde, geordend is als een zogenaamde *heap*. De rekentijd voor deze versie is evenredig met $n \log n$, met n het aantal punten in het netwerk. Het Dijkstraälgorithme construeert de kortste routes en de reistijden erlangs tegelijk naar alle punten van het netwerk. Dit betekent voor seismische *ray tracing* dat de kortste-routemethode een globaal stralenveld construeert naar alle punten in het model, zodat er geen problemen zijn met convergentie van zoekstralen in de richting van een gespecificeerde ontvanger, zoals dit wel het geval is bij de *shooting*-methode voor seismische *ray tracing*. Daarbij vindt de kortste-

routemethode het pad met de absoluut minimale reistijd, zonder te blijven steken in een locaal minimum, zoals vaak gebeurt bij de *bending*-methode.

Men kan ook vragen naar geconditioneerde kortste routes, dat wil zeggen routes die gehouden zijn om een gespecificeerde deelverzameling van punten in het netwerk te bezoeken. Het algorithme dat dit geconditioneerde kortste-routeprobleem oplost kan gebruikt worden om latere aankomsten op het seismogram te berekenen wanneer de deelverzameling een grensvlak of verstrooier voorstelt. Het geconditioneerde kortste-routealgorithme is gebaseerd op een generalisatie van het Dijkstraälgorithme, erin bestaande dat de reistijden van gespecificeerde punten vooraf gepredestineerd worden. Deze generalisatie maakt het mogelijk om verschillende puntbronnen bij elkaar te voegen tot gecompliceerdere bronnen met een ruimtelijke uitgebreidheid. Iedere nieuwe reflectie vereist dan een extra uitvoering van het Dijkstraälgorithme. Als bijproduct kunnen andere experimenten uitgevoerd worden, zoals migratie of een zogenaamde exploderende reflector.

De accuratesse van de kortste-routemethode blijkt afhankelijk te zijn van het aantal punten in het netwerk en het aantal verbindingen per punt. Er wordt bewezen dat de reistijdfout quadratisch afhankelijk is van zowel de ruimtediscretisatie als de hoekdiscretisatie. Tengevolge van het Fermatprincipe is de fout in de stralengeometrie een orde van grootte groter dan de reistijdfout. Met hulp van de relatie tussen de reistijdfout en de netwerkparameters kan de precisie geschat en gecontroleerd worden, dit in relatie tot de bijhorende rekentijd. Ook kunnen de resultaten van grovere netwerken naar fijnere netwerken geëxtrapoleerd worden. Een standaard uniforme fout in de reistijd van minder dan 1 promille kan bereikt worden in matig gecompliceerde snelheidsmodellen met netwerken van niet al te grote afmetingen. Wanneer dit niet voldoende mocht zijn, kan zogenaamde additionele *bending* worden toegepast om de reistijden en het stralenpad te verbeteren. De kortste-routemethode is zelfs bij uitstek geschikt om met zo'n additionele *bending*-methode gecombineerd te worden, omdat de kortste routes veilige beginschattingen zijn die dicht bij het globaal minimale-reistijdpad liggen als het netwerk maar toereikend is.

De additionele *bending*-methode is gebaseerd op drie bestanddelen: 1. de stralenpaden worden geparametriseerd door hun coördinaten (liefst de coördinaten van de kortste routes die verbeterd moeten worden) 2. de geconjugeerde-gradiëntenmethode voor minimalisatie wordt gebruikt om het minimale-reistijdpad te vinden 3. *Beta-splines* worden gebruikt opdat een willekeurig nauwkeurige integratie langs de candidaatpaden mogelijk is. De geconjugeerde-gradiëntenmethode voor minimalisatie van functies met gebruik van hun afgeleiden is geschikt voor dit doel omdat slechts vier vectoren, ieder met afmetingen gelijk aan de afmeting van de straal zelf, hoeven worden opgeslagen en omdat de structuur van de reistijdfunctie effectief uitgebuit wordt. Het kan voor de eerste paar iteraties voordelig zijn om een pseudo tweede-ordezoekrichting te gebruiken, die verkregen wordt door de inversie van de Hessiaan van de reistijdfunctie te benaderen. De punten van het stralenpad moeten tijdens het minimalisatieproces geinterpoleerd worden, ten eerste omdat anders fouten ontstaan wanneer de inverse snelheid een concave functie is, en ten tweede omdat interpolatie tegelijk een veel nauwkeuriger resultaat en een veel kortere rekentijd oplevert. *Beta-splines* zijn geschikt als interpolatieschema, omdat ze een aantal eigenschappen hebben, zoals de locale controle en de convexe-omhullende-eigenschap, dat gunstig uitvalt voor *bending* en ten tweede omdat ze een extra controle verschaffen op de stralenpaden

door de vormparameters. *Bending* in aanwezigheid van discontinuïteiten in het snelheidsveld kan door de positie van de punten op de straal zodanig te transformeren dat ze gedwongen worden om op de discontinuïteit te liggen. De *bending*-methode legt geen beperkingen op aan de beginstraal, omdat divergentie van het proces onmogelijk is, maar er kan niet met zekerheid bepaald worden of het resultaat een globaal of slechts een locaal minimum-reistijdpad is.

De kortste-routemethode is vergeleken met andere methoden voor voorwaartse modellering. Ten eerste is een vergelijking gemaakt met programmatuur ontwikkeld bij de Shell. Die vergelijking liet zien dat de reistijden van beide methoden slechts binnen 1 procent met elkaar overeenkomen, in plaats van met de gewenste 1 promille. De verschillen kunnen slechts gedeeltelijk verklaard worden door de foutenkarakeristiek van de kortste-routemethode en nog wat andere bekende effecten; het overige gedeelte kon niet verder onderzocht worden door het confidentiële karakter van de Shell-programmatuur. Om dezelfde reden zijn de rekentijden niet met elkaar vergeleken. Toch heeft de vergelijking belangrijke suggesties opgeleverd voor de verbetering van de kortste-routemethode, namelijk door het in aanmerking nemen van de snelheidsvariaties langs de verbindingen van het netwerk, in plaats van alleen aan de eindpunten. Voor de kortste-routemethode toegepast op rechthoekige grids van punten kunnen zowel de efficiëntie als de accuratesse enorm verhoogd worden door het toepassen van geavanceerde lijnintegratieschema's, die in plaats van drijvende-puntoperaties alleen integer-operaties gebruiken. De bewering dat aankomsttijden van seismische golven in schaduwzones, wanneer ze een waarneembare amplitude hebben, correct gevonden worden door de kortste-routemethode wordt ondersteund door eindige-differentieberekeningen. Er wordt getoond dat de combinatie van de kortste-routemethode en additionele *bending*-methoden een efficiënte en robuuste berekening van de stralenpaden en de reistijden erlangs mogelijk maakt. De reistijdfout is minder dan 1 promille voor een juiste keuze van de parameters die het *bending*-proces bepalen.

Bij wijze van toepassing is er aangetoond dat de kortste-routemethode op een voordelige manier gebruikt kan worden om aardbevingen te localiseren in sterk heterogene media. Hiervoor is de kortste-routemethode gecombineerd met een formulering van inverse problemen in termen van kansdichtheidfuncties. De gecombineerde methode is robuust, vindt de best passende locatie en geeft een idee van de betrouwbaarheid van de oplossing in de vorm van confidentiegebieden.

Als tweede toepassing van de kortste-routemethode is het effect van golffrontherstel op de statistische eigenschappen van reistijden in willekeurig verstoorde media bestudeerd. De kortste-routereistijden zijn vergeleken met reistijden volgens de Fermatbenadering, dat wil zeggen de integraal over de verstoring van de inverse snelheid langs een referentiestraal in het onverstoorde medium. De Fermatbenadering van de reistijdvertraging is een lineaire benadering die geen rekening houdt met het herstel van een golffront nadat dit een anomale snelheidsstructuur heeft gepasseerd. De conclusies zijn dat de kortste-routereistijdvertragingen systematisch kleiner zijn dan nul, in tegenstelling tot de Fermatvertragingen, die altijd gemiddeld nul zijn. Voorts blijkt dat de variantie in de kortste-routereistijdvertragingen veel kleiner is dan kan worden verwacht op grond van de Fermatbenadering.

Dankbetuiging - Acknowledgements

Mijn beide promotores, Prof. Dr A.M.H. Nolet en Prof. Dr K. Helbig, ben ik erkentelijk voor hun steun en bijdrage aan dit werk.

Vooral de samenwerking met Guust Nolet, eerst tijdens mijn doctoraalonderzoek en daarna gedurende vier jaar promotieonderzoek, was zeer stimulerend. Door hem heb ik geleerd dat verschillende golflengten niet destructief hoeven te interfereren. Ik dank hem voor vele uren vruchtbare discussies en de tijd en moeite die hij nam om mij van verkeerde gedachten af te helpen. Ik ben erg benieuwd naar dat 19e eeuwse roeitochtje op de rivier, thee onder de parasol op een vlekkeloos gemaaid gazon en Shakespeare toe.

Professor Helbig wil ik bedanken voor zijn begeleiding gedurende de periode Augustus 1988 tot Augustus 1989, maar vooral voor zijn steun tijdens het tot stand komen van het proefschrift.

De onvergetelijke bijdrage van Roel Snieder aan het onderzoek en met name aan Appendix C van Hoofdstuk 5 waardeer ik ten zeerste. Het was mij een genoegen om met hem samen te werken, prikkelende ideeën uit te wisselen en bij hem thuis af te wassen.

Ik ben dankbaar voor de samenwerking met René Bastians, waardoor ik meer inzicht heb gekregen in mijn programmatuur. De vele keren dat ik per trein afreisde naar Brabant en een verrukkelijke lunch genoot zal ik niet snel vergeten.

I would like to thank Prof. Dr E. Eisner for his discussion and the suggestions worked out in Chapter 6 that greatly enhanced the value of my research.

Systeembeheerder Joop Hoofd ben ik dankbaar voor het geven van hulp en raad, maar vooral voor het uitlezen van tapes, als die door mijn toedoen onleesbaar waren geworden.

I am grateful to Dr M. Joswig for his hospitality in Bochum and his helpful discussions on earthquake location.

Van mijn collegae wil ik vooral mijn voormalige kamergenoten Suzan van der Lee en Filip Neele bedanken voor het critisch lezen van het artikel dat ten grondslag ligt aan Hoofdstuk 5. De overige collegae dank ik voor de prettige sphaer, een goede voedingsbodem voor hoogwaardig wetenschappelijk onderzoek.

Prof. Dr J.K. Lenstra ben ik dankbaar voor zijn bijdrage in het begin van het onderzoek, waardoor ik snel de geschiktste algorithmen voor het berekenen van kortste routes in netwerken op het spoor kwam.

Van de geophysicastudenten wil ik vooral Henk Marquering bedanken voor het doctoraalonderzoek dat hij bij mij gedaan heeft en dat een bijdrage tot Hoofdstuk 6 geleverd heeft. De overige studenten dank ik voor de gezelligheid gedurende vele kantinebezoeken en borrels, maar vooral voor de leerzame referaten die ze voor mij als colloquiumvoorzitter gehouden hebben.

De moeder van de Geophysicaäfdeling, onze secretaresse Mevr. D. Pattynama, dank ik voor haar nooit aflatende zorg en toewijding. Ik ben haar erkentelijk voor haar adviezen met betrekking tot mijn geestelijk welzijn, maar vooral voor haar smakelijke bami en fondue.

Drs H.P.G.M. den Rooyen van het Koninklijke/Shell Exploratie en Productie Laboratorium dank ik voor zijn bijdrage aan het onderzoek in Augustus 1990 in Rijswijk en het regelen

van rekentijd op de CRAY. Ook een woord van dank aan T. Stavinga en Ed Duin van de Rijks Geologische Dienst te Haarlem voor hun gastvrijheid en medewerking.

Ir A.J.P. Sanders en zijn opvolger Dr W. Boontje van de Stichting voor de Technische Wetenschappen dank ik voor hun medewerking aan het onderzoek.

Prof. Dr A. van der Sluis wil ik graag bedanken voor de tijd die hij voor mij beschikbaar heeft gemaakt en zijn raadgevingen, die leidden tot Hoofdstuk 4.

Wim Spakman ben ik erkentelijk voor het beschikbaar stellen van zijn plotprogramma, P, voor adviezen en discussies.

I am especially grateful to Prof. Dr Ph.L. Toint and his Ph.D. students Didier Burton and Christophe Sebudandi for their hospitality during my visits to Namur and their helpful discussions about network theory and ray tracing.

Torild Van Eck ben ik dankbaar voor zijn medewerking aan Hoofdstuk 7.

De Heer en Mevrouw Top-van Eyck van Heslinga ben ik zeer erkentelijk voor de steun die zij mij verleend hebben.

Van mijn familie wil ik Oma, Tante Irene, Liesje en Pimmetje bedanken voor hun voortdurende morele ondersteuning, niet alleen gedurende mijn promotietijd, maar ook tijdens mijn studententijd. Oma voor het vele malen ter beschikking stellen van de studeerkamer, maar vooral voor alle dingen die niet genoemd kunnen worden. Tante Irene voor het luisteren naar het Urker Mannenkoor. Liesje voor de kennismaking met vele nieuwe vrienden. Pimmetje voor zijn verstrooiende computerspelletjes.

Mijn beide paranymphen, Drs H.P. Flier en Mej. A.F. de Korte, ben ik erkentelijk voor hun hulp.

Het nauwkeurig nalezen van het manuscript van het proefschrift door Dr C.G. de Koster en zijn vrouw stel ik ten zeerste op prijs.

Drs B.J. Michels, voormalig student, verdient een woord van dank voor zijn niet aflatende aandacht voor mijn physieke welzijn.

Graag wil ik mijn huisgenoten van Schalkwijkstraat 4bis bedanken voor hun steun en gezelligheid, vooral gedurende de laatste maanden. Harm-Jaap ben ik erkentelijk voor het leren roken van een Heerenpluim en pannekoeken, maar dan eten. Bianca voor het geven van welgemeende adviezen, het braden van mijn biefstuk en het automatisch poetsen der tanden. Amelia voor het uitleggen van televisieseries, als ik de draad kwijt was. Jurjen voor het uitleggen van elementaire geologische begrippen.

Plato ben ik dankbaar voor haar hemelse liefde, en de hemelse Suzanne voor het timmeren met haar hamertje.

Curriculum vitae

Schrijver dezes behaalde in Juni 1982 het diploma Gymnasium β aan het Gymnasium Celeanum te Zwolle. In hetzelfde jaar werd de studie Natuurkunde met bijvak Geophysica aangevangen aan de Rijksuniversiteit Utrecht. Na het behalen van het propaedeutisch examen in Mei 1984 werd de studie voortgezet als doctoraalstudie Geophysica met bijvakken Natuurkunde en Wiskunde. Deze werd in November 1987 voltooid met een doctoraalonderzoek onder begeleiding van Prof. Dr A.M.H. Nolet. Dit onderzoek had als onderwerp de stralentheorie en bestond uit een literatuuronderzoek en de ontwikkeling van raytracingalgorithmen gebaseerd op grafentheorie. De suggesties die dit onderzoek opleverde zijn uitgewerkt tijdens een aanstelling als Onderzoeker in Opleiding aan het Department of Theoretical Geophysics te Utrecht van Januari 1988 tot en met December 1991, onder begeleiding van Prof. Dr A.M.H. Nolet en Prof. Dr K. Helbig.

Contents