APPLICATION OF SIMULATED ANNEALING TO SOME SEISMIC PROBLEMS

By

Danilo Rubén Velis

Geofísico, Universidad Nacional de La Plata, Argentina

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF

THE REQUIREMENTS FOR THE DEGREE OF

DOCTOR OF PHILOSOPHY

in

THE FACULTY OF GRADUATE STUDIES

DEPARTMENT OF EARTH AND OCEAN SCIENCES

We accept this thesis as conforming

to the required standard

.....

.....

.....

.....

.....

.....

THE UNIVERSITY OF BRITISH COLUMBIA

August 1998

© Danilo Rubén Velis, 1998

In presenting this thesis in partial fulfilment of the requirements for an advanced degree at the University of British Columbia, I agree that the Library shall make it freely available for reference and study. I further agree that permission for extensive copying of this thesis for scholarly purposes may be granted by the head of my department or by his or her representatives. It is understood that copying or publication of this thesis for financial gain shall not be allowed without my written permission.

Department of Earth and Ocean Sciences
The University of British Columbia
129-2219 Main Mall
Vancouver, Canada
V6T 1Z4

Date:

Abstract

Wavelet estimation, ray tracing, and traveltime inversion are fundamental problems in seismic exploration. They can be finally reduced to minimizing a highly nonlinear cost function with respect to a certain set of unknown parameters. I use simulated annealing (SA) to avoid local minima and inaccurate solutions often arising by the use of linearizing methods. I illustrate all applications using numerical and/or real data examples.

The first application concerns the 4th-order cumulant matching (CM) method for wavelet estimation. Here the reliability of the derived wavelets depends strongly on the amount of data. Tapering the trace cumulant estimate reduces significantly this dependency, and allows for a trace-by-trace implementation. For this purpose, a hybrid strategy that combines SA and gradient-based techniques provides efficiency and accuracy.

In the second application I present SART (SA ray tracing), which is a novel method for solving the two-point ray tracing problem. SART overcomes some well known difficulties in standard methods, such as the selection of new take-off angles, and the multipathing problem. SA finds the take-off angles so that the total traveltime between the endpoints is a global minimum. SART is suitable for tracing direct, reflected, and headwaves, through complex 2-D and 3-D media. I also develop a versatile model representation in terms of a number of regions delimited by curved interfaces.

Traveltime tomography is the third SA application. I parameterize the subsurface geology by using adaptive-grid bicubic B-splines for smooth models, or parametric 2-D functions for anomaly bodies. The second approach may find application in archaeological and other near-surface studies. The nonlinear inversion process attempts to minimize the rms error between observed and predicted traveltimes.

Table of Contents

List of Tables

List of Figures

Acknowledgment

I would like to express my deep thanks to my supervisor Tadeusz Ulrych for his guidance and support. He has been both a mentor and a friend to me throughout my years at UBC.

I thank Laurence Lines and Toshifumi Matsuoka for helpful comments about ray tracing. I also thank Michael Bostock, Richard Pawlowicz, and the members of the committee, for their excellent comments and suggestions that helped to improve the quality of this work.

A note of gratitude goes to Xingong Li and David Aldridge, who let me use some of their codes.

For continuing support, I am very grateful to Facultad de Ciencias Astronómicas y Geofísicas, Universidad Nacional de La Plata, Argentina.

A special thanks to my former supervisor Alberto Comínguez, for introducing and guiding me into the seismic data processing arena, and to the graduate students and colleagues at the Department of Applied Geophysics, Universidad Nacional de La Plata, for their friendship and support.

Thanks to the other current and earlier Ph.D. students and staff at the Department of Earth and Ocean Sciences, for their commitment to create a pleasant and dynamical environment at the department, and for those unforgettable soccer Friday afternoons.

I am especially in debt to my parents, brothers, and friends, who are responsible for a great part of what I have achieved in my life.

Most of all, I wish to thank my wife Susana and my son Juan Manuel for their unconditional love, patience and understanding.

To the memory of my mother... I miss you.

Chapter 1

Introduction

Thus, in our reasoning we depend very much on 'prior information' to help us in evaluating the degree of plausibility in a new problem. This reasoning process goes on unconsciously, almost instantaneously, and we conceal how complicated it really is by calling it 'common sense'.

E.T. Jaynes – *Probability Theory: The Logic of Science*

1.1 Introductory remarks

In general, geophysical inverse problems ultimately reduce to minimizing an error, objective, or cost function with respect to a set of unknown model parameters:

$$\text{Cost Function(Model)} = \text{Minimum} \tag{1.1}$$

Most of these problems are nonlinear and very difficult to solve using standard optimization techniques. The difficulties arise not only because of the nonlinearities involved in such optimization problems, but also because frequently the unknown parameters, which represent physical quantities, must satisfy a set of constraints. These are used to incorporate a priori geophysical or geological information on the model parameters. Generally these constraints are simply bounding ranges (e.g. a velocity must be greater than zero), but often, they represent additional nonlinear functions and relationships. Consequently, the problem is usually cast as a constrained nonlinear optimization one. Other difficulties arise from the fact that often the forward problem does not lend itself to be easily treated

with standard techniques that are based on local gradient directions or derivatives (e.g. the cost function has no analytical form and is nondifferentiable).

Nonlinear optimization problems are often solved iteratively. Given an initial guess, the current solution is updated until no further improvement in the cost function is observed. Here multimodality plays a key role for the success of the algorithm at hand, for local minima may prevent it from finding the desired solution. This point represents one of the most important concerns in solving equation (1.1).

That the minimization of a multimodal cost function using local techniques (e.g. steepest descent) leads to the solution which is closest to the starting model is a commonly known fact. On the contrary, the simulated annealing (SA) method is not much affected by the initial guess. As a result, the global minimum of the cost function can be obtained. Figure 1.1 depicts the contrasting behavior between SA and a local algorithm in minimizing a multimodal cost function. Local algorithms are "greedy" in the sense that they only make downhill moves, whereas in the SA, uphill moves are not forbidden. This allows SA to avoid local minima.

1.2 Simulated annealing applications

SA applies to a wide variety of applications. From circuit design [WS92] and optimal protein states conformation [BB89], to econometric problems [GFR94] and optimal deployment of missile interceptors [BJS88]. Van Laarhoven and Aarts describe more traditional applications [vLA88], while Sen and Stoffa focus on geophysical applications [SS95]. The first application in Geophysics was made by Rothman in 1985 [Rot85, Rot86] for estimating residual statics corrections. Subsequently, the method has found several applications like coherence optimization [LBT89], transmission tomography [AV90], waveform inversion [SS91, LHS95], reflection inversion [PL93], resistivity inversion [SBS93], earthquake

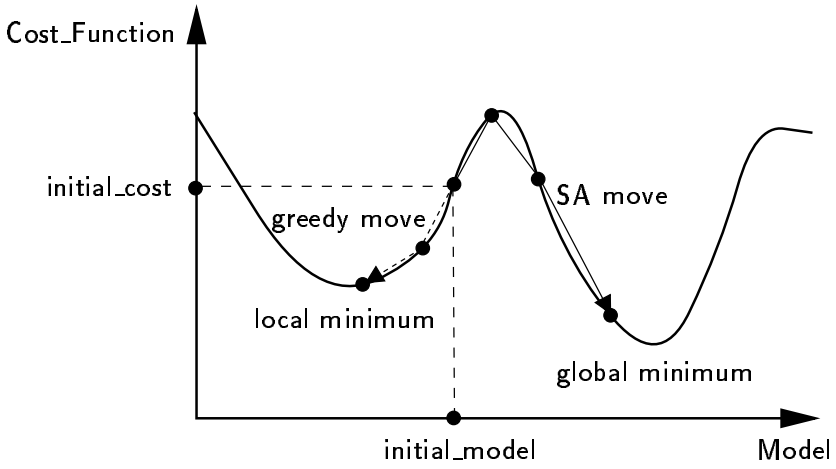

Figure 1.1: Comparison of "greedy" algorithms with SA. (a) "Greedy" algorithms (e.g. steepest descent) always perform downhill moves (dotted line), guiding the solution to the minimum which is closest to the initial model. (b) SA occasionally accepts uphill moves (solid line) so that it does not get stuck in local minima.

location [Bil94], earthquake source parameter estimation [HL95], residual statics corrections [LHS95], combined first-arrival traveltime and reflection coherency optimization [PL97], etc. Though the SA method is a very time-consuming algorithm, the results shown so far are encouraging as the numerous applications illustrate.

1.3 Simulated annealing, linearizing methods, and hybrid strategies

Both SA and linearizing optimization methods have commonly been used for solving geophysical parameter estimation problems. The drawbacks of the local (greedy) optimization algorithms have already been described. Not only is the issue of local convergence a problem, but also the fact that many geophysical inverse problems deal with very complex cost functions which are not easily linearized. Despite the limitations of local algorithms, their rapid convergence as compared to SA make them extremely useful for most geophysical applications, since the amount of data usually involved is enormous and

computational speed becomes a necessity. SA, on the other hand, is a time-consuming algorithm. This is the price to be paid in exchange for obtaining the global minimum of the objective function. Usually, hundred or thousand iterations should be performed until convergence, even using fast approaches like very fast simulated annealing (VFSA) [Ing89]. Simulated quenching (a variant of SA) is not always the best remedy for this problem, for a trade-off between convergence speed and global convergence arises.

One way of exploiting the advantages of both methods and work out their respective drawbacks, is to combine them into some hybrid strategy. Chunduru et al [CSS96] combine the conjugate gradient method with VFSA for the inversion of 2-D resistivity data and for seismic velocity analysis. Liu et al. [LHS95] combine the simplex method with a SA algorithm based on Lane's approach [Lan92] for 1-D acoustic waveform inversion and residual statics. Hartzell and Liu [HL95] use the hybrid method of Liu et al. for earthquake source parameters estimation. Both hybrid strategies have shown to be more efficient than the approach based on SA alone, although their *global convergence* properties are not clear. In the next chapters, I will develop hybrid strategies to improve the performance of VFSA for wavelet estimation and two-point seismic ray tracing. In particular I combine VFSA with the steepest descent method and a quasi-Newton method, respectively.

1.4 Alternatives to simulated annealing

Simulated annealing optimization encompasses a wide family of algorithms which use different strategies regarding the perturbation stage, temperature schedule, and acceptance criterion, as the variety of application problems illustrates. Other global optimization algorithms exist to solve the same problems. Branch-and-bound and related exhaustive

methods are guaranteed to find the global minimum, though not necessarily in an acceptable time [RR88]. Pure random search and the multistart methods [And72] are two closely related stochastic methods which have been traditionally used for global optimization. These methods are rarely used in nowadays applications due to their low efficiency in terms of computational cost to obtain the global minimum. Simulated evolutionary optimization frequently offers more attractive alternatives [Fog94]. These methods encompass genetic algorithms [Dav87, Gol89], evolution strategies and evolutionary programming [Fog93]. Among other rather unlikely methods for global optimization, I can mention taboo search [CK95], neural networks [RS88], etc.

Despite the fact that these methods do not lend themselves to theoretical understanding (particularly regarding convergence), researchers have found them very effective in some geophysical problems. Such is the case of genetic algorithms (see for example [SS95] for detailed accounts of genetic algorithms in geophysical applications), taboo search in connection to seismic inversion [VM96], and neural networks for automate velocity picking [FK94]. The methods mentioned, together with "greedy methods", and countless hybrid combinations have been used successfully in many applications. However, neither of the mentioned algorithms seems to be universally preferred for all problems. Researchers find, usually through experimentation, which tool best fits the problem at hand.

1.5 Dissertation outline

This dissertation addresses some fundamental inverse problems arising in geophysics which have not been solved satisfactorily using standard "greedy" algorithms nor ever addressed using SA. In all cases I use very fast simulated annealing (VFSA) to globally minimize the multimodal, nonlinear cost functions that are involved. In some cases I

combine both VFSA with linearizing algorithms to expedite convergence and/or to refine the solution. The first portion of the thesis, Chapter 2, provides enough introductory material on SA that the dissertation stands on its own. It also points out the differences between SA and *simulated quenching*, and explores strategies for dealing with constraints other than bounding ranges.

Chapter 3 deals with the problem of wavelet estimation using the fourth-order cumulant matching method [Tug87, Laz93, VU95a, VU96b]. The fourth-order cumulant matching (CM) method has been recently developed for estimating a mixed-phase wavelet from a convolutional process [Laz93]. Matching between the trace fourth-order cumulant and the wavelet fourth-order moment is done in a minimum mean-squared error sense under the assumption of a non-Gaussian, stationary, and statistically independent reflectivity series. This leads to a highly nonlinear optimization problem, usually solved by techniques that require a certain degree of linearization, and that invariably converge to the minimum closest to the initial model. Alternatively, the SA approach developed in Chapter 3 provides reliability of the numerical solutions, reducing the risk of being trapped in local minima.

Beyond the numerical aspect, the reliability of the derived wavelets depends strongly on the amount of data available. However, using a multidimensional taper to smooth the trace cumulant, I show that the method can be used even in a trace-by-trace implementation, a point that is, I believe, very important from the point of view of stationarity and consistency. Here I combine both optimization techniques into a hybrid strategy to exploit the efficiency of the linearizing algorithm and the reliability of the SA solutions. I also demonstrate the viability of the method under several reflectivity models that may arise in real situations. The application of the hybrid strategy is illustrated by means of marine and field data examples. The consistency of the results is very encouraging because the improved cumulant matching strategy I developed can be effectively used

in a trace-by-trace implementation. This chapter appeared previously as a paper in *Geophysics* [VU96b].

The purpose of Chapter 4 is to develop a new method for solving the two-point seismic ray-tracing problem based on Fermat's principle of stationary time, which I call SART (simulated annealing ray tracing). In models where more than one raypath satisfies Fermat's principle (multipathing), SART focuses on finding the direct ray trajectory whose traveltime is minimum. The nonlinear algorithm overcomes some well known difficulties that arise in standard ray shooting and bending methods. Problems related to: (1) the selection of new take-off angles, and (2) local minima in multipathing cases, are overcome by using an efficient SA algorithm to find the ray parameters that generate the absolute minimum path connecting source and receiver. At each iteration, the ray is propagated from the source by solving a standard initial-value problem. The last portion of the raypath is then forced to pass through the receiver. Using SA, the total traveltime is then globally minimized and the appropriate initial conditions that produce the absolute minimum path are so obtained.

As it is described in Chapter 4, SART is suitable for accurately tracing rays through 2-D models only (an extension to 3-D is well developed in Chapter 5). The model is parameterized by rectangular cells of constant velocity (or slowness) and at least one additional discontinuity (e.g. a reflector) can be included. This allows one to trace not only direct waves, but also reflections and headwaves of a predefined signature. In fact, SART can be used in conjunction with any available or user-preferred initial-value solver system (in Chapter 5 I use a numerical ray-tracing procedure). A number of examples of multipathing in moderately complex 2-D media are examined. The results using the nonlinear two-point ray-tracing system demonstrate that the absolute minimum path connecting source and receiver can be obtained regardless the initial guess. Parts of this chapter appeared as a paper in *Geophysical Research Letters* [VU96a].

Chapter 5 has two purposes: (1) to extend SART to encompass 3-D laterally varying media of arbitrary complexity, and (2) to present a very flexible model parameterization that can be used in conjunction with SART to trace complex raypaths through general 3-D media. Though the first purpose is the main goal, the second one is very important too, since solving the forward problem is not always a well understood problem, especially in complex velocity structures. In the present formulation, the model is characterized by any number of regions separated by interfaces. The velocity field within each individual region is specified separately. It may be any function of the space coordinates with the constraint of being twice differentiable. Interfaces can have arbitrary form with the only condition of being univalued. The problems of local minima and take-off angles selection are more severe in the 3-D than in the 2-D case. These problems are overcome easily by SART, for the basic principles regarding global convergence defined for 2-D, apply to 3-D as well.

In SART (3-D case) the differential equations that govern the wave propagation are integrated using standard numerical ODE solvers until the ray emerges at the desired endpoint following the absolute minimum traveltime path. This boundary-value problem (BVP) is cast as a nonlinear optimization problem which is in turn solved by means of VFSA. This guarantees convergence to the global minimum of the cost function representing the total traveltime from source to receiver. In 3-D, in contrast to 2-D, the cost function is at best a unimodal two-dimensional continuous smooth surface. But in real life, the cost function is often a multimodal, discontinuous, nondifferentiable, rough surface of two (or more than two) variables. To obtain the absolute minimum of such an ill-behaved function, a stochastic technique like SA becomes an invaluable tool. A number of synthetic examples are shown to demonstrate the versatility of the model parameterization and the viability of SART to solve the BVP effectively. Although the main goal of SART is to trace first-arrival direct waves, several strategies for dealing

with complex raypaths (reflections, headwaves, multiples, etc) are well described. A discussion about SART performance in terms of computational cost and accuracy is also presented and illustrated with numerous synthetic examples. This includes a quantitative comparison between SART and some standard shooting techniques. Finally, several alternative formulations regarding the optimization problem are explored and discussed in detail. Some parts of this chapter were presented at *V Congreso Argentino de Mecánica Computacional, Tucumán, Argentina*, and also published in the proceedings [Vel96].

The last portion of this thesis, Chapter 6, deals with the problem of traveltime tomography using SA. The subsurface geology is parameterized either using (1) cubic B-splines defined over a 2-D grid with variable spacing in both horizontal and vertical coordinates, or (2) parametric 2-D functions aimed to represent anomaly-like bodies submerged in a smooth background velocity. The flexible grid used in the spline approach can accommodate moderately complex smooth structures with only a few nodes. The parametric approach, allows one to image velocity anomalies of various shapes, including abrupt changes of velocity, using a small set of parameters. The approaches attempt to minimize the *rms* error between observed and predicted arrival times, which are computed using a finite-difference algorithm. VFSA was used to estimate the velocity at the spline node locations, the grid spacing in each dimension, or the parameters defining the anomaly bodies. The solution is independent of the initial model because local minima are avoided during the process by a convenient uphill-downhill exploration of the model space. Results using synthetic examples show how moderately complex structures can be obtained using a few parameters without introducing undesirable artifacts [VU95b]. The spline approach is especially suited for smooth models, and the parametric approach, for imaging buried structures such as those arising in archaeology and other near-surface studies [Vel98].

Chapter 2

Simulated Annealing

The development of high-speed computers has made it possible to incorporate a 'randomizing unit' into some machines which will select random samples ad hoc. The use of such mechanisms requires investigation.

Maurice G. Kendall – *The Advanced Theory of Statistics, 1969*

2.1 Introduction

The SA method is a stochastic computational algorithm for finding near-optimal solutions to hard optimization problems. Van Laarhoven and Aarts provide a complete introductory treatment of SA and describe traditional applications [vLA88]. The SA method is derived from statistical mechanics where the behavior of a large number of particles at a given temperature is analyzed. In 1953, Metropolis et al. [MRR$^+$53] presented a Monte Carlo sampling technique for modeling the evolution of a solid submerged in a heat bath at a given temperature. Three decades later, the technique was generalized and applied to nonlinear optimization problems [KGV83], specifically to the problem of partitioning, component placement, and wire routing in VLSI (very large scale integration) design. Here, the unknown parameters play the role of the particles in the solid, and the cost (objective) function represents the energy of the system. In the SA approach, the model space is randomly perturbed and new configurations are accepted (or rejected) so that the cost function or energy decreases iteration after iteration. Occasionally, some increases of the cost function are allowed and it is this *uphill* movement that allows the system to escape from local minima. Initially, the temperature (a control parameter) is

high enough so that almost all proposed configurations are accepted. On the contrary, at low temperatures the probability of accepting a new configuration corresponding to an increase of the cost function is small. The fluctuations of the random perturbations are gradually decreased along with the temperature following a predefined cooling schedule, and the process is stopped after a maximum number of iterations or when no noticeable change in the cost function is observed during a certain number of consecutive steps. At this point it is said that the system has "crystallized" in the sense that the minimum energy state (global minimum) has been obtained.

2.2 Statistical mechanics, the Metropolis algorithm, and the annealing process

The basics of statistical mechanics should be understood to appreciate the connection between condensed matter annealing and the solution of hard optimization problems, such as those occurring in the solution of some geophysical inverse problems. A system consisting of a large number of particles interacting at a finite temperature can be described by the position of the particles. The configuration of the system is referred to as the "state" x. Such is the case of a fluid or melted solid submerged in a heat bath. If T is the temperature of thermal equilibrium of the system with energy $E(x)$, then the probability $\pi_T(x)$ that the system is in state x can be described by the Boltzmann or Gibbs distribution

$$\pi_T(x) = \frac{1}{Z(T)} e^{\frac{-E(x)}{\kappa T}} \tag{2.1}$$

where κ is the Boltzmann's constant,

$$Z(T) = \sum_{w \in X} e^{\frac{-E(w)}{\kappa T}} \tag{2.2}$$

is the partition function, and X is the set of all possible states.

The Metropolis algorithm (MA) [MRR+53] is a simple stochastic algorithm for simulating these kind of processes developed in the earliest days of scientific computing. In each step of this procedure, a small random displacement (perturbation) Δx to the current state x with energy $E(x)$ is performed, and the energy change $\Delta E = E(x + \Delta x) - E(x)$ is then computed. Now, the ratio h between the probabilities of both states is evaluated:

$$h = \frac{\pi_T(x + \Delta x)}{\pi_T(x)} = e^{\frac{-\Delta E}{\kappa T}}. \tag{2.3}$$

If the energy of the new state is less than the energy of the previous state, that is if $h > 1$, then the new configuration is accepted as the new state. If $h \leq 1$, that is if the energy of the new state is greater than or equal to the energy of the previous state, then the new configuration is accepted with probability h. This criterion for accepting or rejecting a new state is known as the "Metropolis criterion" [MRR+53]. It can be shown that as the number of iterations tends to infinity, the probability that the system is in a given state x equals $\pi_T(x)$, and hence the distribution of the generated configurations converges to the Boltzmann distribution [GG84].

It interesting to study the behavior of the system at low energy states, in order to appreciate the crystalline structure of the melted solid as temperature is gradually decreased (chemical annealing). For this purpose, the temperature of the heat bath must be decreased slowly enough so that the probability of obtaining lower energy states according to the Boltzmann distribution (2.1) predominates. These states can be obtained by

keeping the system in thermal equilibrium. As a counterpart, "quenching" is the process by which the temperature of the heat bath is lowered very quickly giving not enough time for the system to reach thermal equilibrium. Consequently, the probability of obtaining low energy states is greatly reduced, and the resulting structure of the material may be far from constituting a crystalline lattice.

2.3 Simulated annealing optimization

In 1983, Kirkpatrick et al. [KGV83] made the analogy of a computer design problem with the physical system described in the previous section. They devised a strategy for obtaining a near-optimal solution to the problem of partitioning, component placement and wire routing in VLSI design. Basically, a Metropolis algorithm was applied at each of a series of iterations with gradually decreasing temperatures. For applying SA to solve an optimization problem of the form of equation (1.1), the following analogies must be made:

- Identify the configuration of parameters (model) with the configuration of particles.
- Identify the objective or cost function, say Φ, with the energy of the system.
- Identify the process of finding a near-optimal solution with the process of finding the lowest energy states.

The Boltzmann constant κ is set to 1 and the temperature T becomes a control parameter. In addition, three very important functionals must be defined:

1. $g(\Delta x)$: the probability density function (pdf) for generating a new configuration in the model space;

2. $h(\Delta\Phi)$: the probability density function for accepting a new configuration; and

3. T_k: the cooling schedule for the annealing process (temperature T at iteration k).

Here x represents the model space of M parameters. Note that both the generating and the accepting pdf's are a function of the temperature as well. The main differences concerning performance among various SA techniques reside in the choice of these three functions. Figure 2.1 shows a flow chart describing a typical SA algorithm. At each iteration, a new state is generated by drawing from the pdf $g(\Delta x)$. The Metropolis criterion is then evaluated and, after lowering temperature according to the cooling schedule T_k, a new iteration is started. The process is stopped when a maximum number of iterations, k_{max}, is reached (note that other stopping conditions may be used). When the cost function increases, the probability of accepting the uphill move is given by h. In practice, the new state is accepted if h is greater than a random number r, which is drawn from a uniform pdf, $U[0,1]$.

2.3.1 Generating function

Initially, the Metropolis algorithm searched the model space using a uniform pdf for g. Faster convergence was later achieved by using a Gaussian pdf instead, which concentrates on the states with lowest energy by narrowing the search as temperature decreased ("Boltzmann annealing" [SH87]). In this case, the M-dimensional Gaussian pdf may be expressed by

$$g(\Delta x) = (2\pi T)^{-M/2} e^{\frac{-\Delta x^2}{2T}}, \tag{2.4}$$

where $\Delta x = x - x_0$ is the perturbation applied to the previous configuration, x_0. Here temperature T can be viewed as a measure of the fluctuations of g around x_0. At high

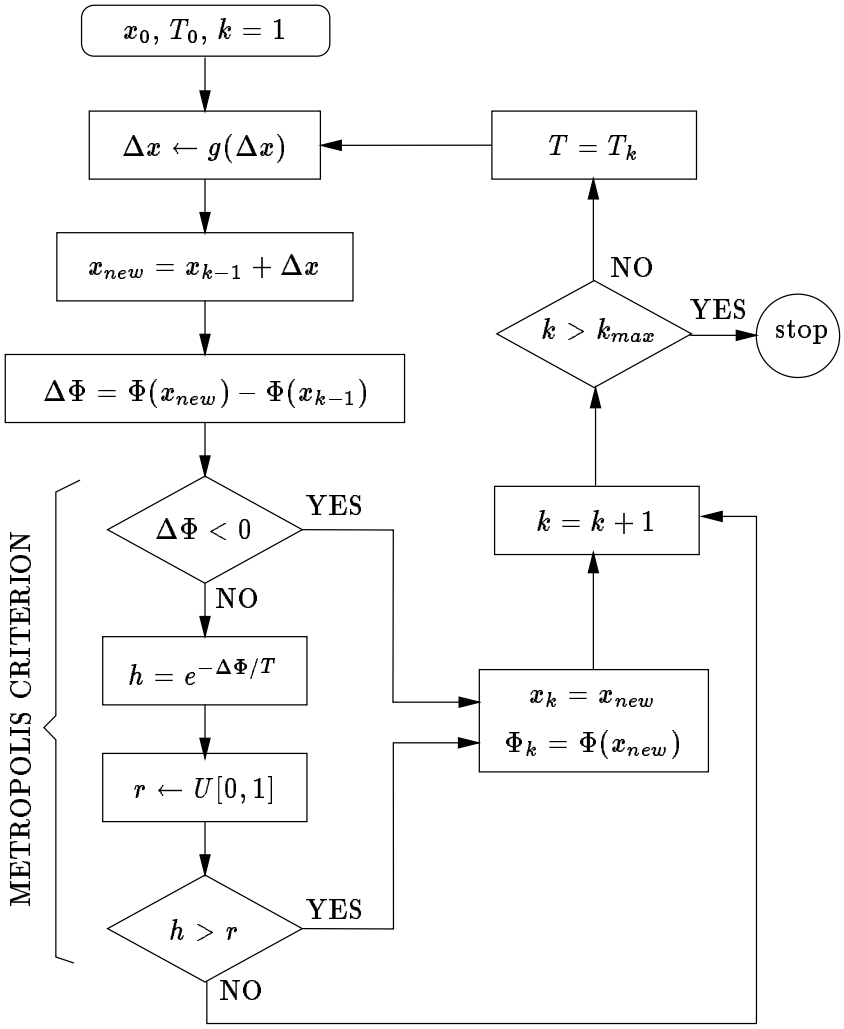

Figure 2.1: Flow chart of a basic SA algorithm.

temperatures, the model space is sampled more or less uniformly. Conversely, when temperatures are low, perturbations tend to be smaller.

2.3.2 Acceptance function

Generally, the Metropolis criterion is used for accepting/rejecting new configurations:

$$h(\Delta\Phi) = \frac{e^{\frac{-\Phi_{k+1}}{T}}}{e^{\frac{-\Phi_{k+1}}{T}} + e^{\frac{-\Phi_k}{T}}} = \frac{1}{1 + e^{\frac{\Delta\Phi}{T}}} \simeq e^{\frac{-\Delta\Phi}{T}}. \tag{2.5}$$

This expression is based on the probability of obtaining a new state with energy Φ_{k+1} relative to the previous state with energy Φ_k, where $\Delta\Phi = \Phi_{k+1} - \Phi_k$. This is essentially the Boltzmann distribution. Following Metropolis, a new configuration is accepted unconditionally if $\Delta\Phi < 0$, and accepted with probability $h(\Delta\Phi)$ if $\Delta\Phi \geq 0$. Clearly, as the temperature approaches zero, the acceptance probability decreases exponentially and only the lowest energy states are accepted. This is the key point of SA optimization. It states the most important difference with the so-called "greedy" algorithms. While other optimization methods perform only downhill moves (i.e. accept only those states for which the cost function decreases), SA allows uphill moves too. This behavior facilitates exploring the whole model space without being trapped in local minima. Linearizing techniques (e.g. steepest descent) belong to the class of greedy algorithms, and their success for obtaining the global minimum of multimodal cost functions is very much influenced, as is well known, by the starting model.

2.3.3 Cooling schedule

The schedule used to lower the temperature of the annealing process plays a key role in SA optimization. Ideally, one would like to lower the temperature as fast as possible in order to reach the lowest energy states in a few iterations. However, theory indicates that the

cooling rate cannot be faster than a prescribed function if annealing, and not quenching, is desired. Back to the thermal process analogy arising in statistical mechanics, if the metal is cooled slowly enough, the crystal lattice structure results in a configuration with minimal energy (global minimum). On the contrary, if temperature is lowered too fast, distortions in the crystal structure which represent higher energy states (local minima) will be preserved at the end of the process. These two different processes, that is annealing and quenching, are well known in chemical annealing.

Thus, for true annealing, thermal equilibrium must be obtained before temperature is lowered. Originally, this state was reached by keeping temperature at a constant level [MRR+53]. Later, it was found that the equilibrium could be obtained provided that the temperature was lowered slowly enough. It was Geman and Geman [GG84] who showed that the global minimum of Φ can be (statistically) obtained if temperature is lowered no faster than

$$T_k = \frac{T_0}{\ln(1+k)}, \tag{2.6}$$

where T_0 is a constant and k is iteration. If the temperature is lowered faster than what expressed in equation (2.6), a premature crystallization may occur and the system may get trapped in a local minimum. A heuristic demonstration can be given to show that expression (2.6) is a sufficient condition for the convergence to a global minimum [SH87] (see Appendix A).

2.4 Fast annealing (FA)

A faster cooling schedule can indeed be used without affecting the convergence proof by modifying the generating function. "Fast annealing" (FA) uses a Cauchy pdf of the form

$$g(\Delta x) = \frac{1}{\pi} \frac{T}{(\Delta x^2 + T^2)^{(M+1)/2}}. \tag{2.7}$$

The cooling schedule that guarantees (statistically) that a global minimum can be found for this particular choice of g is

$$T_k = \frac{T_0}{k}. \tag{2.8}$$

The corresponding heuristic demonstration is given in Appendix A. FA has proven to be superior to standard annealing [SH87] due to the fact that its cooling rate is exponentially faster than the logarithmic schedule used by BA. The fatter tail of the Cauchy distribution (see Figure 2.2) allows the system to explore the model space more conveniently and local minima are more easily avoided.

2.5 Very fast simulated annealing (VFSA)

Ingber [Ing89] proposed a SA technique that allows a still faster cooling schedule. This scheme is superior to the aforementioned SA approaches [Ros92] and in some cases superior to evolutionary methods such as genetic algorithms [IR92]. VFSA uses a generating function with fatter tails than the Cauchy's. Contrarily to BA and FA which sample infinite ranges, VFSA samples finite ranges. Besides, VFSA takes into account the differences in each parameter-dimension by allowing g to expand or contract adaptively according to the sensitivity of the cost function of each dimension (this procedure is called "re-annealing").

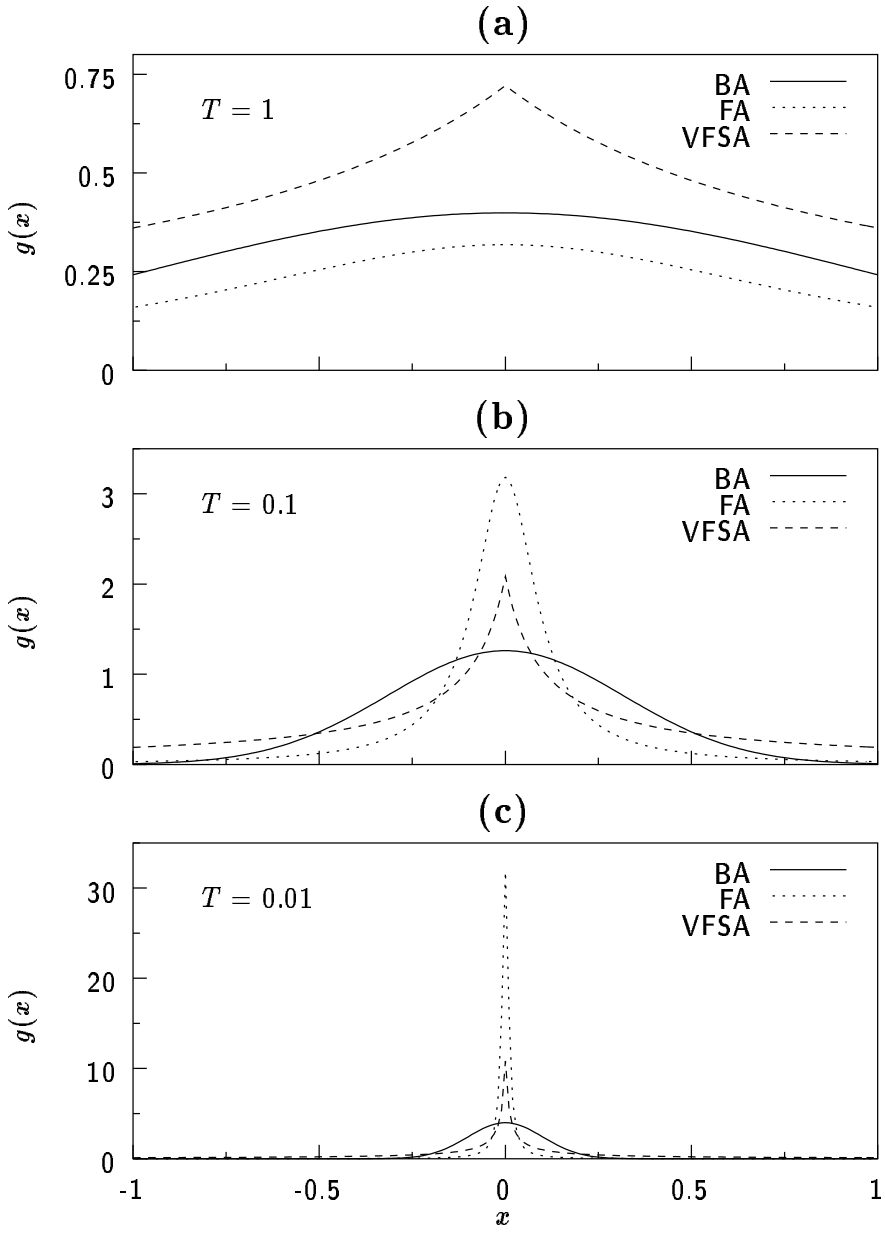

Figure 2.2: In SA, a new model state is generated by drawing a perturbation from the probability density function $g(x)$. The figures show the pdf corresponding to Boltzmann annealing (BA), fast annealing (FA) and very fast simulated annealing (VFSA), for $T = 1$, $T = 0.1$ and $T = 0.01$. Note the fatter tail of the VFSA generating function, even at low temperatures.

2.5.1 Description of the algorithm

In VFSA, new parameters $x_i^{(k)}$ in dimension i are generated via the random variable y_i using

$$x_i^{(k+1)} = x_i^{(k)} + y_i(B_i - A_i), \qquad y_i \in [-1, 1]. \tag{2.9}$$

The new parameters $x_i^{(k+1)}$ are constrained to the range $[A_i, B_i]$, and formula (2.9) is repeated for all i until a set of M new parameters are generated. The generating function in the VFSA algorithm is defined as

$$g(y) = \prod_{i=1}^{M} \frac{1}{2(|y_i| + T_i)\ln(1 + 1/T_i)}, \tag{2.10}$$

where I have dropped the superscript k for simplicity. The cumulative probability distribution may be obtained by the integral

$$G(y) = \int_{-1}^{y_1} \cdots \int_{-1}^{y_M} g(y)dy_1 \cdots dy_M = \prod_{i=1}^{M} G_i(y_i), \tag{2.11}$$

$$G_i(y_i) = \frac{1}{2} + \frac{\mathrm{sgn}(y_i)}{2} \frac{\ln(1 + |y_i|/T_i)}{\ln(1 + 1/T_i)}. \tag{2.12}$$

As a result, new points ruled by this distribution may be generated from a uniform distribution $u_i \in [0, 1]$ considering

$$y_i = G_i^{-1}(u_i), \tag{2.13}$$

hence

$$y_i = \mathrm{sgn}(u_i - \frac{1}{2})\, T_i \left[(1 + 1/T_i)^{|2u_i - 1|} - 1\right]. \tag{2.14}$$

Having defined $g(y)$, the cooling schedule that statistically guarantees the convergence to a global minimum can be given by

$$T_i = T_{0i}\, e^{-c_i k^{1/M}}, \tag{2.15}$$

where c_i is some *user-defined* constant that does not affect the convergence proof given in Appendix A. In practical contexts, this constant may be used to tune the algorithm for particular applications. For example, to reach a final temperature T_{fi} in k_{fi} iterations, set

$$c_i = k_{fi}^{-1/M} \log \frac{T_{0i}}{T_{fi}}. \tag{2.16}$$

2.5.2 Re-annealing

It seems reasonable to use a wider generating function for the insensitive dimensions relatively to the more sensitive ones. To account for different sensitivities of the cost function of each parameter at different locations of the model space, temperatures are periodically rescaled (every a hundred iterations, or so) so that $g(y)$ expands or contracts accordingly. The sensitivities are computed at the current lowest minimum, and the parameter temperatures rescaled relatively to the maximum sensitivity by means of

$$T_i' = T_i \frac{\max_i\{s_i\}}{s_i}, \tag{2.17}$$

where

$$s_i = \frac{\partial \Phi}{\partial x_i} \simeq \frac{\Phi(x_i + \delta) - \Phi(x_i - \delta)}{2\delta}, \tag{2.18}$$

with δ small.

2.5.3 Quenching

Whenever the number of parameters is large, the cooling rate (2.15) may be still too slow for some applications (consider the exponent $1/M$). In these cases, the cooling rate can be accelerated using

$$T_i = T_{0i}\, e^{-c_i k^{Q/M}}, \tag{2.19}$$

where Q is a *quench* factor set between 1 and M. Note that only for $Q = 1$ the statistical convergence is ensured (Appendix A).

2.6 Annealing or quenching?

As seen in previous sections, the cooling rate is somehow governed by the selected generating function. Figure 2.3 shows various cooling schedules for different SA algorithms. For simplicity, T_0 is such that T_1 is about the same in all cases. Note that in VFSA, temperature decreases much faster than in BA and FA, especially for M small.

For many practical applications, the cooling schedule which is consistent with the heuristic proof of global convergence is too slow because too many iterations are required to reach the lower energy states, particularly when the number of unknowns is relatively large. With expediency the only reason given, many researchers use faster cooling schedules that do not guarantee convergence, claiming to use SA to solve their problems.

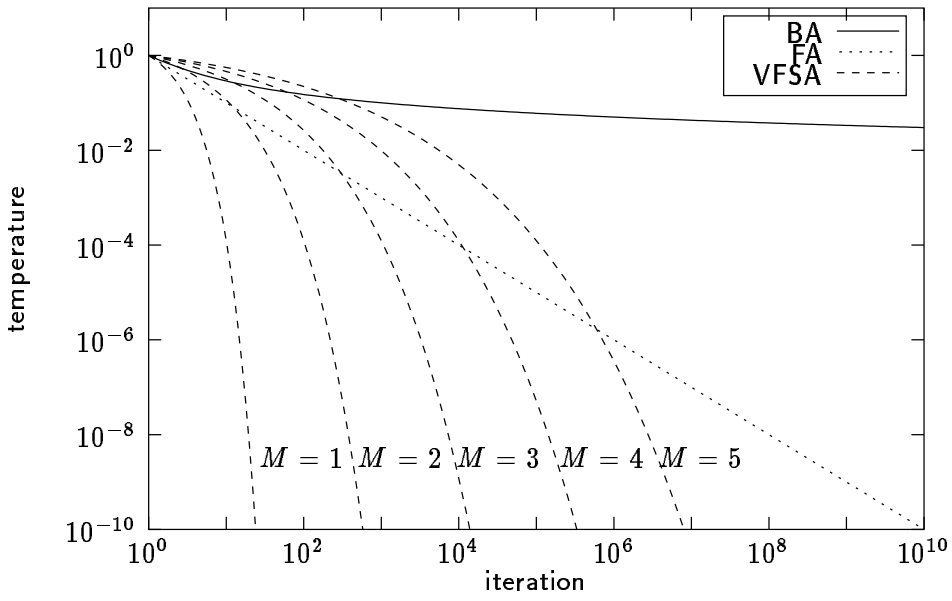

Figure 2.3: Comparison of various SA cooling schedules.

For example, a commonly used cooling rate in BA is the exponential schedule of the form

$$T_k = T_0\, e^{-\alpha k}, \qquad \alpha \text{ small}, \tag{2.20}$$

which is equivalent to another frequently used cooling rate

$$T_k = \beta T_{k-1} = \beta^k T_0, \qquad \beta = e^{-\alpha}. \tag{2.21}$$

These two schedules are particular cases of equation (2.19) for $Q = M$ and $c_i = \alpha$. It is clear that the convergence proof given for the BA algorithm is violated if one of the above two cooling schedules is used. These algorithms should be called *simulated quenching* (SQ) rather than simulated annealing, for the system is not guaranteed to reach the lowest energy states in annealing time. Nevertheless, any attempt to reduce

the number of iterations it is always worthwhile, and quenching has been found very useful in a number of circumstances [Ing93]. For example, Mirkin et al. compared various (quenching) schedules for the residual statics problem and obtained satisfactory results [MVC93]. An interesting discussion on annealing vs quenching can be found in [Ing96].

2.6.1 Initial temperature

The selection of the initial temperature is sometimes a critical issue in standard SA algorithms. If T_0 is too high, the system will take longer to reach the lowest energy states. On the other hand, if T_0 is too low, the system might freeze too soon. The convergence proof for BA requires that the initial temperature in equation (2.6) is sufficiently high. This is not the case for FA and VFSA, where T_0 is arbitrary, as shown in Appendix A. Besides, since the cooling rate of both FA and VFSA is much faster that BA (see Figure 2.3), the selection of a high value for T_0 is not critical at all. In practical contexts, the initial temperature should be such that the expected cost at this temperature is within one standard deviation of the mean cost [Whi84]. This is to ensure that almost all proposed configurations are accepted at this stage (the system is completed "melted"). For this purpose it is enough to set T_0 numerically equal to the mean cost function for a set of arbitrary points selected at random from the model space.

In Figure 2.3 I illustrated various cooling rates for different SA algorithms. For the sake of simplicity, I set the initial temperatures so that $T_1 = 1$ in all cases. Since T_0 is arbitrary for FA and VFSA, smaller values could have been selected for higher cooling rates without affecting the convergence proof. In terms of rate of convergence, it is worthwhile mentioning that too high values for T_0 in BA might diminish its efficiency tremendously. In FA and VFSA one can be much less cautious in selecting the initial temperature.

At this point it is necessary to stress the difference, in the VFSA algorithm, between *acceptance temperature*, which is associated with the Metropolis criterion, and *parameter temperature*, which is associated with the generating function. Even though the cooling rate for both temperatures is generally the same, the initial temperatures might be different. In the first case, T_0 is chosen as described above (mean cost at arbitrary points). In the second case, T_{0i} is generally set equal to 1 for all i's, so that the model space is sampled widely independently of the initial configuration. These are the methods I will use for selecting the initial temperatures throughout this thesis.

2.7 Constrained simulated annealing

There are two types of constraints that may arise in nonlinear optimization problems such as equation (1.1):

- bounding constraints, and
- model constraints.

Any model configuration that satisfies the given constraints, is said to belong to the *feasible region*. While configurations not satisfying the constraints are in the *infeasible region*. The union of both regions form the *model space*. Bounding constraints usually consist of minimum and maximum values for each model parameter (search space). These constraints are naturally incorporated in VFSA by specifying the ranges for the "random" perturbation stage in equation (2.9). Whenever a new configuration falls outside the given ranges, equation (2.9) is repeated until all model parameters fall inside the feasible region. The incorporation of model constraints is not so direct. Model constraints are of four types:

1. exact linear constraints,

2. exact nonlinear constraints,

3. inequality linear constraints, and

4. inequality nonlinear constraints.

The model constraints can be any relationship, explicit or implicit, between the model parameters. Often they represent a difficult optimization problem on their own. Handling these four types of model constraints using any optimization algorithm requires sometimes very different approaches and strategies. But in general, there are two ways of addressing the resulting constrained optimization problem. One way is to devise an algorithm able to generate always feasible points in an efficient way. These points are then checked for acceptance/rejection using the Metropolis algorithm and SA proceeds as usual. The second approach is to transform the constrained optimization problem into an unconstrained one by means of penalty terms. During the first stages of the optimization, model parameters usually are infeasible points, but as iteration proceeds, they tend to get into the feasible region.

It is not the purpose of this chapter to develop general methods for handling model constraints with SA, although the literature in these matters is not complete and further investigation is required. In general, the first of the above two approaches (sampling from the feasible region) is desirable for nonlinear optimization, for SA concentrates on decreasing the value of the cost function within the feasible region only and does not waste the time exploring the infeasible model space. The second approach (penalizing infeasible points) is perhaps more widely used for its apparent simplicity, though the sampling may become biased towards undesirable model regions during too many SA iterations. Moreover, if penalty terms are not chosen properly, the algorithm may get easily trapped in local minima. Sampling from the feasible region for linear model constraints is in

general easier than for nonlinear ones, for linear constraints constitute a *convex set*[1]. This convex set can be sampled with no difficulties using a number of techniques (see for example [BBK+87]). These sampling techniques can be used in conjunction with SA for generating feasible (random) points. The same algorithms can be extended for nonlinear model constraints [CH82], but only if they define a convex set. A general technique for sampling feasible points with arbitrary nonlinear model constraints remains to be found.

Fortunately, particular methods can be devised for the optimization problem at hand, and a general technique is not always required. In Chapters 4 and 5 I devised some schemes for dealing with nonlinear model constraints in connection to the two-point ray-tracing problem via simulated annealing (SART). The model constraint here is that the raypath must arrive to a given receiver point. The cost function is represented by the traveltime. It is not obvious which is the feasible region in this problem. Actually, it corresponds to those take-off angles which generate a ray propagating from the source to the receiver (in the two-dimensional space defined by the two take-off angles, declination and azimuth, the feasible region consists of isolated points). Clearly, not only the constraint is nonlinear, but also the feasible region is nonconvex. If one were able to sample from the feasible region, the problem would be solved almost immediately. But this is not possible in general. In SART the feasible region is extended to encompass all possible take-off angles (within a given range), though not necessarily corresponding to raypaths with physical sense. However, when traveltime is minimum, the final raypath satisfies the ray equations from source to receiver.

[1] A set is said to be *convex* if it contains straight line segments between any two of its points (see for example [Rud87]).

2.8 Conclusions

SA is a very powerful and important tool for optimizing difficult cost functions in a variety of disciplines. VFSA, a variant of SA, outperforms other SA algorithms in terms of computational cost. This is achieved by exploring the model space more efficiently and allowing for fast cooling rates. At the same time, one can be much less cautious for selecting the initial temperature. Contrarily to *simulated quenching* (SQ), VFSA strictly adheres to the sufficient conditions for global convergence. Nevertheless, SQ is a valuable and valid alternative in terms of its practicality.

Constraints come in two flavors: bounding constraints and model constraints. The first type is naturally implemented in VFSA, since a search range is specified for each model parameter. The second type requires a special consideration which may be problem dependent. Linear model constraints are by far more easily implemented than nonlinear ones.

It is worth mentioning that there exist in the literature other SA algorithms not described in this chapter. To name a few I can mention the *heat bath algorithm* [Rot86], *SA without rejected moves* [GS84], *mean field annealing* [BMT+89], etc. The first two are intended to reduce the low acceptance to rejection ratio of the standard SA algorithm. The third one is a SQ algorithm that performs very well when a mean-field theory is a good approximation to a stochastic cost function[2].

[2]In statistical mechanics, the mean-field theory is based on approximating the stochastic behavior of large systems of electronic spin elements with complex interactions, by a deterministic set of equations. In the SA approach, these equations are solved iteratively.

Chapter 3

Wavelet Estimation Using Fourth-Order Cumulant Matching

Models are to be used but not to be believed.

Henry Theil

3.1 Introduction

In this chapter I am concerned primarily with the problem of wavelet estimation via the 4th-order cumulant matching (CM) approach. In essence, the problem consists of estimating, at best, two unknowns from one equation. In the seismic case, as is well known, the model of the recorded seismic trace, x_t, is generally formulated as a convolution plus additive noise

$$x(t) = a(t) * r(t) + v(t), \tag{3.1}$$

where $a(t)$ is the seismic wavelet and $r(t)$ is the reflectivity series.

The CM approach makes use of higher-order cumulants, higher-order covariance functions with special properties. It turns out that, as a result of the convolutional model which I have described, including additive Gaussian noise, the seismic wavelet can be estimated from the noisy seismogram under the constraint that the reflectivity is a stationary, non-Gaussian, and statistically independent random process [BR67, LR82, NR87]. Since the 2nd-order cumulant of a zero-mean process (autocorrelation) is a phaseless function,

and the 3rd-order cumulant is identically zero when the process is symmetrically distributed (like most seismic reflectivity sequences), higher-order cumulants are needed for mixed-phase estimation. The information contained in the 4th-order cumulant (a 3-D function) is enough to determine the wavelet within a polarity reversal and a time shift, and no assumptions are required concerning the wavelet phase.

Tugnait [Tug87] presented a 4th-order CM method in which a mixed-phase moving-average (MA) wavelet is estimated from the data 4th-order cumulant, which is generally an estimate of the wavelet 4th-order moment. Following that work, Lazear [Laz93] presented a parametric CM method for estimating a mixed-phase wavelet using real seismic data. The CM problem turns into an optimization problem where a highly nonlinear cost function is to be minimized. The main purpose of this chapter is to offer an improved strategy for solving the optimization problem that makes use of a very fast simulated annealing (VFSA) algorithm [Ing93], reducing the risk of the solution being trapped in local minima. I also propose a hybrid strategy aimed at a trace-by-trace implementation. This approach combines the fast performance of a standard linearizing technique with the reliability of the results provided by VFSA.

I propose the application of a multidimensional taper to smooth the trace cumulant as a method to provide better estimates even when small amounts of data are used. This is particularly important because the accuracy of the trace cumulant estimate depends strongly on the amount of (stationary) data available. The importance of tapering of the trace cumulant is illustrated by means of both synthetic and field data examples.

Very recently, Hargreaves [Har94] has pointed out that the 4th-order cumulant, like the kurtosis, may be phase insensitive below a limiting bandwidth. I explore this crucial aspect of the CM approach and show that, in general, any lag of the 4th-order moment exhibits the same degree of sensitivity to wavelet phase.

I draw attention to the behavior of the CM method under various non-Gaussian

reflectivity models that may arise in real situations [WH86]. Here I show various synthetic examples using as few as 250 data samples and as many as 10,000 data samples. The results are very encouraging because very good wavelet estimates are obtained for most models.

Finally I illustrate the improved CM strategy in a trace-by-trace implementation using both on-shore and marine data. With these examples I demonstrate the viability of the CM approach even when a small amount of data is used.

3.2 Background theory

3.2.1 Cumulants and moments

Cumulants and moments are higher-order covariance functions with properties that make them very useful for describing both stochastic and deterministic signals (see for example [Men91]). Given a real stationary discrete-time random process $x(t)$, $t = 0, \pm 1, \pm 2 \cdots$, its nth-order moment function is expressed by

$$m_n^x(\tau_1, \tau_2, \cdots, \tau_{n-1}) = E\{x(t), x(t+\tau_1) \cdots x(t+\tau_{n-1})\}, \tag{3.2}$$

where $E\{\cdot\}$ denotes statistical expectation. In the case of real discrete-time deterministic finite-length signals, $a(t)$, $t = 0, 1, \cdots, M-1$, expectations are replaced by sums[1]:

$$m_n^a(\tau_1, \tau_2, \cdots, \tau_{n-1}) = \sum_{t=0}^{M-1} a(t)a(t+\tau_1) \cdots a(t+\tau_{n-1}). \tag{3.3}$$

These moments measure the degree of similarity between a signal and a product of delayed or advanced versions of itself. For example, m_1^a is known as the "mean" value, $m_2^a(\tau_1)$ is the autocorrelation function, etc.

[1] In practice, higher-order moments are estimated using this formula for both stationary random processes and finite-length deterministic signals [NP93].

For $n = 3, 4$, the nth-order cumulant function of a non-Gaussian stationary random process, $x(t)$, can be written as

$$c_n^x(\tau_1, \tau_2, \cdots, \tau_{n-1}) = m_n^x(\tau_1, \tau_2, \cdots, \tau_{n-1}) - m_n^G(\tau_1, \tau_2, \cdots, \tau_{n-1}), \tag{3.4}$$

where $m_n^G(\tau_1, \tau_2, \cdots, \tau_{n-1})$ is the nth-order moment function of an equivalent Gaussian random process that has the same mean value and autocorrelation function as $x(t)$. Clearly, if $x(t)$ is Gaussian, $c_n^x(\tau_1, \tau_2, \cdots, \tau_{n-1}) = 0$. Since there is no clear advantage to using cumulants for the analysis of deterministic signals, the previous definition applies only to stochastic processes.

The 1st-order cumulant is equal to the 1st-order moment, which is the mean value. The 2nd-order cumulant is equal to the covariance, and in the case of a zero-mean process, it is equal to the autocorrelation. This function is symmetric about $\tau = 0$ and hence all phase information is lost. As a result and, as is well known, methods based on the use of the autocorrelation are only capable of identifying a minimum-phase wavelet from a convolutional process. The 3rd-order cumulant equals the 3rd-order moment in the case of a zero-mean process (i.e. $m_3^G(\tau_1, \tau_2) = 0$ for all τ_1, τ_2). It is a 2-D function that preserves phase information in its structure that may be exploited for system identification. However, when its argument is an independent identically distributed (IID) random process (such as most reflectivities), it is identically zero. On the other hand, the 4th-order cumulant of a non-Gaussian stationary signal not only preserves phase information, but also is nonzero for an IID process. In the case of zero-mean signal, $x(t)$,

$$m_4^G(\tau_1, \tau_2, \tau_3) = m_2^x(\tau_1)m_2^x(\tau_3 - \tau_2) + m_2^x(\tau_2)m_2^x(\tau_3 - \tau_1) + m_2^x(\tau_3)m_2^x(\tau_2 - \tau_1). \tag{3.5}$$

By putting $\tau_1 = \tau_2 = \tau_3 = 0$ in the previous definitions, we have the familiar quantities

$$\begin{cases} \gamma_2^x = & c_2^x(0) = & E\{x^2(t)\}, & (variance) \\ \gamma_3^x = & c_3^x(0,0) = & E\{x^3(t)\}, & (skewness) \\ \gamma_4^x = & c_4^x(0,0,0) = & E\{x^4(t)\} - 3E^2\{x^2(t)\}. & (kurtosis) \end{cases} \tag{3.6}$$

The normalized kurtosis is defined as $\gamma_4^x/(\gamma_2^x)^2$, and equals zero for a Gaussian process.

3.2.2 Cumulant matching (CM) for a convolutional process

Combining equations (3.1) and (3.4) and assuming that the reflectivity is a non-Gaussian, independent and IID process, the following important result is obtained [BR67]:

$$c_n^x(\tau_1, \tau_2, \cdots, \tau_{n-1}) = \gamma_n^r m_n^a(\tau_1, \tau_2, \cdots, \tau_{n-1}) + c_n^v(\tau_1, \tau_2, \cdots, \tau_{n-1}). \tag{3.7}$$

Since $v(t)$ is assumed Gaussian, its nth-order cumulant is equal to zero. This means that the approach is blind (in theory) to additive Gaussian noise. Thus, I can rewrite formula (3.7) for $n = 4$, as

$$c_4^x(\tau_1, \tau_2, \tau_3) = \gamma_4^r m_4^a(\tau_1, \tau_2, \tau_3). \tag{3.8}$$

Equation (3.8) expresses the fact that the 4th-order moment of the wavelet, $a(t)$, equals, within a scale factor, the 4th-order cumulant of the trace, $x(t)$. In practice, neither the 4th-order cumulant of the noise is zero nor the 4th-order cumulant of the reflectivity is equal to a spike of amplitude γ_4^r at lag $(0,0,0)$, as expressed by formula (3.8). This is due to the fact that, despite the assumed stochastic properties of both noise and reflectivity, usually only a finite amount of data is available. Equation (3.8) is strictly true only for $N \to \infty$. For N finite, one has an estimate of the trace 4th-order cumulant, and the CM

is computed in a mean-squared error sense. So I define the following cost function that relates the wavelet parameters with the estimated trace 4th-order cumulant, $\hat{c}_4^x(\tau_1, \tau_2, \tau_3)$:

$$\Phi = \sum_{\tau_1} \sum_{\tau_2} \sum_{\tau_3} [\hat{c}_4^x(\tau_1, \tau_2, \tau_3) - \gamma_4^r m_4^a(\tau_1, \tau_2, \tau_3)]^2. \tag{3.9}$$

In the optimization algorithm, Φ is minimized with respect to the wavelet parameters (the scale factor γ_4^r can be absorbed by m_4^a). This method for estimating a moving-average (MA) system from a convolutional model was first proposed by Lii and Rosenblatt [LR82] and extended by Tugnait [Tug87] in connection with the identification of an ARMA (autoregressive-MA) system. Due to various symmetry planes of the 4th-order moment function and the finite length of the wavelet, the summations in (3.9) are considerably reduced. It can be proved that all the necessary information is contained in one of 24 regions that compound the total domain of the moment function [PIIF92]. This means that if we know the moment at any of these 24 regions, we can map across the symmetry planes and produce the entire 4th-order moment. As a result, it is enough to evaluate the summations in (3.9) for $0 \leq \tau_1 \leq M$, $0 \leq \tau_2 \leq \tau_1$, and $0 \leq \tau_3 \leq \tau_2$ only, M being the length of the MA system. Usually, M can be estimated from the autocorrelation function.

3.3 Optimization problem

Basically, two approaches can be used to minimize Φ: (1) a linearizing technique based on gradient directions, and (2) a stochastic global optimization algorithm. It should be noted that Φ, a multidimensional cost function, is highly nonlinear because it involves higher-order covariances. It seems reasonable to expect it to be a multimodal function. In previous works Φ has been minimized with respect to the MA parameters using a steepest descent algorithm. I am not going to consider the linearizing method here, and

the reader is referred to the works of Tugnait [Tug87] and Lazear [Laz93]. However, it is well-known that this kind of optimization schemes converge to the local minimum which is closest to the initial model. I have found some numerical examples (I show this below) in which the linearizing scheme converges to a local minimum which is far from the required solution. To avoid local minima that may lead to poor wavelet estimates, I propose the use of a stochastic global optimization algorithm. I use a very fast simulated annealing (VFSA) technique to solve for the wavelet parameters which are obtained independently of the initial estimate. Nevertheless, the linearizing technique should not be discarded because in general I have found that it is faster than VFSA.

3.4 Discussion and explanatory examples

3.4.1 Tapering the cumulant estimate

Since in practice, neither the cumulant of the noise is zero nor is the cumulant of the reflectivity series a multidimensional spike at zero lag, the estimated wavelet moment obtained from equation (3.8) is a distorted version of the true wavelet moment. Figures 3.1a and 3.1b show a view of the 4th-order moment of a zero-phase Ricker wavelet and the 4th-order cumulant of a trace (250 samples) that was generated by convolving a sparse reflectivity with the wavelet, plus 10% by amplitude Gaussian noise[2]. Note that for the lags shown, the trace cumulant is a poor estimate of the wavelet moment. Any attempt to recover the wavelet using this cumulant estimate will likely fail.

To improve the estimate, I have found it very useful to apply a 3-D smoothing-taper window to the trace cumulant, especially for those cases in which the cumulant estimate is very poor due to low amount of data available. In these cases, I can re-define the cost function as

[2]The standard deviation of noise was made equal to 10% of the maximum amplitude present in the data.

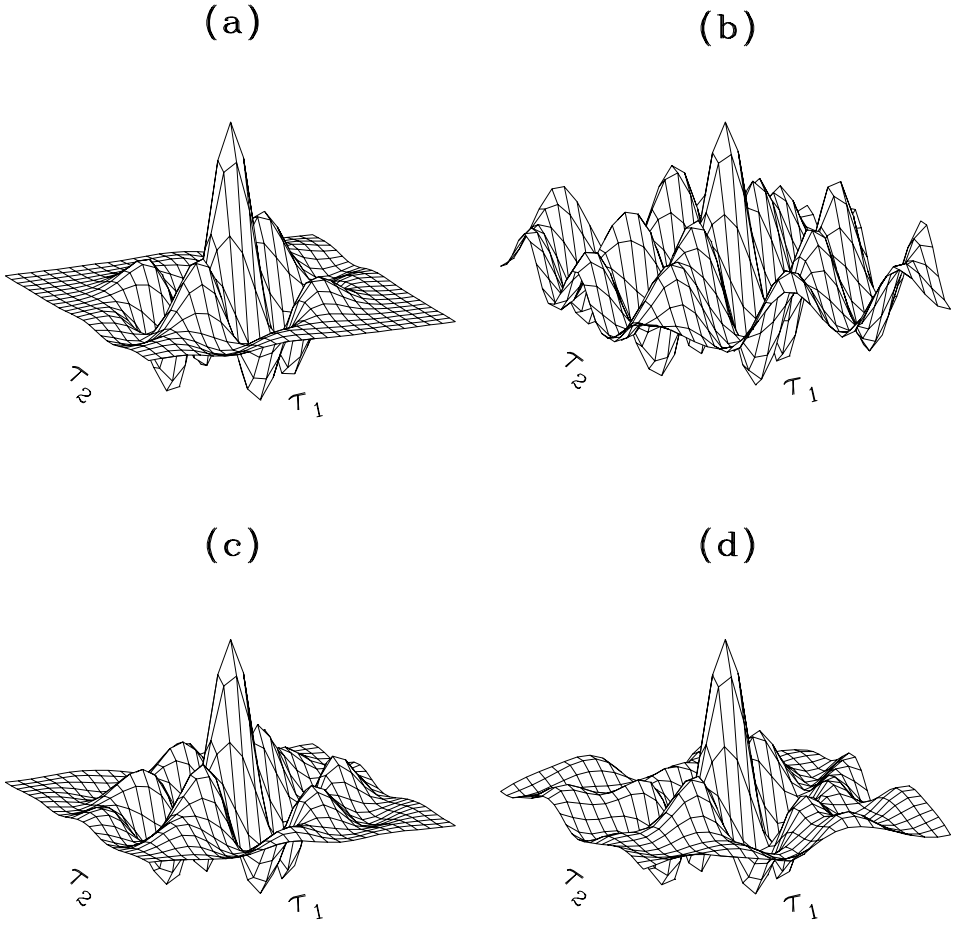

Figure 3.1: Tapering the trace cumulant. (a) Fourth-order moment function of a 25-points Ricker wavelet. (b) Fourth-order cumulant function of a 250-points trace generated by convolving the Ricker wavelet with a sparse reflectivity series (plus 10% by amplitude Gaussian noise). (c) The same cumulant function in (b) after the application of a 3-D Parzen window. (d) Fourth-order cumulant of the trace using 5,000 points. All diagrams correspond to $-12 \leq \tau_1, \tau_2 \leq 12$, and $\tau_3 = 0$.

$$\Phi = \sum_{\tau_1}\sum_{\tau_2}\sum_{\tau_3}[h(\tau_1,\tau_2,\tau_3)\hat{c}_4^x(\tau_1,\tau_2,\tau_3) - \gamma_4^r m_4^a(\tau_1,\tau_2,\tau_3)]^2. \tag{3.10}$$

where $h(\tau_1,\tau_2,\tau_3)$ is a 3-D window function. Usually, suitable windows are applied in the case of power spectral estimation and higher-order spectral estimation to produce better estimates. The window function should have the following properties:

a) $h(\tau_1,\tau_2,\tau_3) = h(-\tau_1,\tau_2-\tau_1,\tau_3-\tau_1) = h(\tau_1-\tau_2,-\tau_2,\tau_3-\tau_2) = h(\tau_1-\tau_3,\tau_2-\tau_3,-\tau_3)$ and any possible exchange of any pair of the three arguments (symmetry properties of the 4th-order cumulant);

b) $h(\tau_1,\tau_2,\tau_3) = 0$ for $|\tau_i| > L$ [L defines the region of support of $\hat{c}_4^x(\tau_1,\tau_2,\tau_3)$];

c) $h(0,0,0) = 1$ (normalizing condition);

d) $H(\omega_1,\omega_2,\omega_3) \geq 0$ for all frequencies $(\omega_1,\omega_2,\omega_3)$.

It is easy to build multidimensional windows satisfying these constraints by using standard 1-D windows (see for example [NP93]). So I can write

$$h(\tau_1,\tau_2,\tau_3) = d(\tau_1)d(\tau_2)d(\tau_3)d(\tau_2-\tau_1)d(\tau_3-\tau_2)d(\tau_3-\tau_1) \tag{3.11}$$

where $d(\tau) = d(-\tau)$; $d(\tau) = 0$, $\tau > L$; $d(0) = 1$ and $D(\omega) \geq 0$ for all ω. Among many possible 1-D windows (Hamming, triangular, etc.) I have obtained very good results using a Parzen window, which is defined by

$$d(\tau) = \begin{cases} 1 - 6(|\tau|/L)^2 + 6(|\tau|/L)^3, & |\tau| \leq L/2; \\ 2(1-|\tau|/L)^3, & L/2 \leq |\tau| \leq L; \\ 0, & |\tau| > L. \end{cases} \tag{3.12}$$

Figures 3.1c and 3.1d show the trace cumulant after applying a 3-D Parzen window and the trace cumulant when using 5,000 data samples. Observe that the improvement due to the tapering is quite important. This obviates the use of large amounts of data where the validity of convolutional stationarity is very much in question. I have found that the CM method can be applied even for a trace-by-trace implementation, and in many cases only a few traces are required to obtain cumulant estimates good enough to provide reliable wavelet estimates. In the last section I illustrate this point using a small set of field data.

3.4.2 VFSA vs linearizing solutions: a hybrid strategy

The reliability of the derived wavelets depends not only on the degree of validity of equation (3.8), but also on the success in the minimization of the multidimensional, nonlinear, and possibly multimodal cost function Φ. Hargreaves [Har94] proposed a procedure to assess the reliability of the derived wavelets that consists of applying a 90-degree phase shift to the data and verifying whether or not the new wavelet estimate exhibits the same phase shift. However, a procedure to assess the reliability of the optimization solution is still required. In his steepest descent algorithm, Lazear [Laz93] starts the iterations with a centered spike as the initializing wavelet. The results obtained by Lazear show that the solution is rather consistent and good estimates are obtained in all the examples shown.

I have found that the VFSA approach generally provides an increased confidence in the solutions. As an example, Figure 3.2a shows a synthetic trace consisting of 250 points ($\Delta t = 4$ ms), generated by convolving a mixed-phase Berlage wavelet [Ald90] with a sparse reflectivity series. The trace has been contaminated with 10% by amplitude of Gaussian noise. In Figures 3.2b to 3.2d the true wavelet as well as the solutions obtained after minimizing Φ are plotted (20 realizations using different noisy traces). In

this particular example the linearizing solution has been trapped in a local minimum of the cost function[3], while the VFSA algorithm was able to recover the actual wavelet more accurately. Finally, Figure 3.3 shows the deconvolved traces using the wavelets in Figures 3.2b to 3.2d, as well as the corresponding error curves. The deconvolution was simply done by spectral division (see the book of Robinson and Treitel [RT80] for an excellent treatment of the deconvolution problem). Though data are very band-limited, a quick view at the error curves shows that the SA solution is a better estimate to the true wavelet.

It should be pointed out here that for *annealing* (and not *quenching*) the CPU time required to find the VFSA solution is substantially greater than that required to find the linearizing one. In the example of Figure 3.2, about 20 iterations were required using the steepest descent approach. Each iteration involved the computation of the current wavelet moment function, its gradient and the step size, as well as updating the wavelet estimate. On the other hand, the VFSA solution required about 1,000 iterations, where each iteration involved basically computing the cost function and perturbing the wavelet estimate. In terms of CPU time, the linearizing approach took about 1.1 seconds for 20 iterations on a Sun Ultra 1 workstation, and VFSA about 4.0 seconds for 1,000 iterations. This implies that each linearizing iteration was about one order of magnitude more expensive than each VFSA iteration. When quenching is performed, the number of iterations may be reduced substantially, and the CPU time becomes comparable to the linearizing optimization.

In a trace-by-trace processing, CPU time is a very important aspect that has to be taken into account. A strategy for an efficient implementation in real situations may be to apply the VFSA algorithm to a single trace and use this estimate as the initial model

[3] As suggested in the previous work, I set the initial guesses for the linearizing approach equal to a centered spike. The fact that the true wavelet contains two main lobes made the method to converge to a local minima. The global minimum might have been obtained using a different starting model.

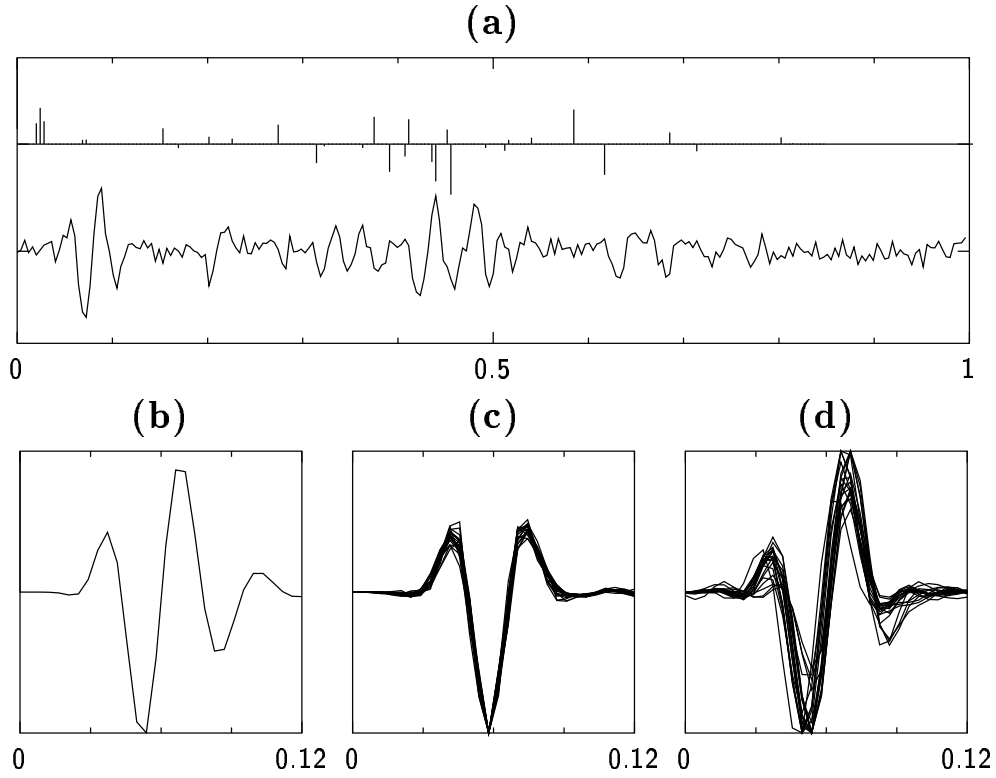

Figure 3.2: Synthetical example: linearizing vs VFSA solutions. (a) Reflectivity series and trace. (b) Actual wavelet. (c) Linearizing wavelet estimates (20 realizations). (d) SA wavelet estimates (20 realizations).

for the remaining traces where the linearizing algorithm can be used. This procedure combines the fast performance of the linearizing technique with the more reliable solutions provided by the VFSA algorithm. At the same time, the risk of being trapped in a local minimum when using the linearizing technique is reduced (if not eliminated) because a reasonable initial model is now used.

3.4.3 Sensitivity to wavelet phase

The sensitivity of the zero-lag 4th-order cumulant (kurtosis) to wavelet phase has been studied by various authors in connection to the estimation of a residual phase-shift present

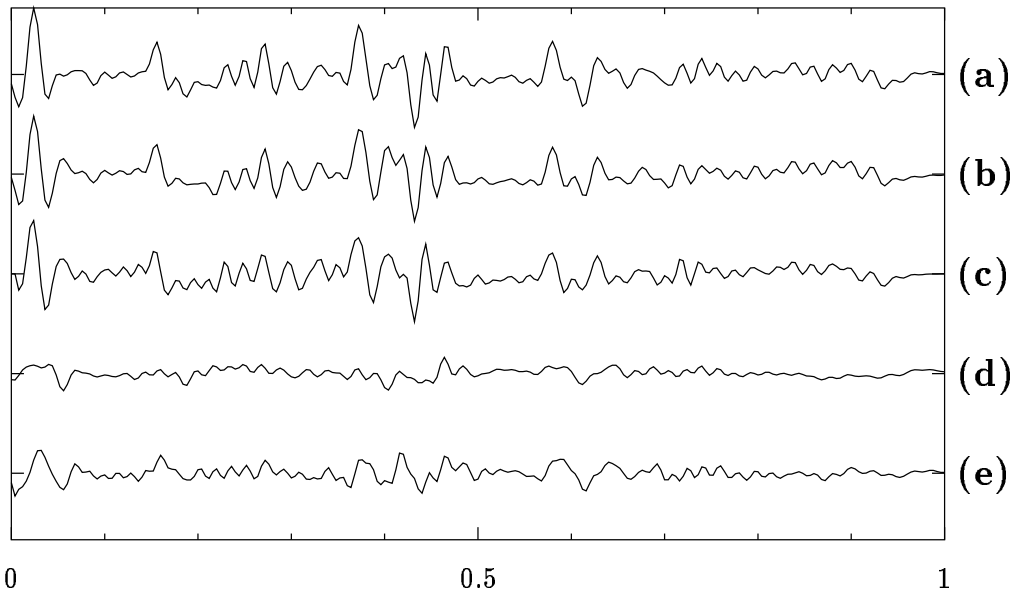

Figure 3.3: Synthetical example: deconvolved traces and error. Deconvolved traces using (a) the actual wavelet, (b) the average SA wavelet, and (c) the average linearizing wavelet. (d) Difference between series (a) and (b). (e) Difference between series (a) and (c).

in the data by maximizing the kurtosis or the varimax norm [Whi86, LO87]. It turns out that if the data effective bandwidth, B, is small compared to the central frequency, f_0, the kurtosis (and hence the varimax norm) is rather insensitive to phase changes. Fortunately, most seismic data satisfy $B > f_0$ and the maximum kurtosis criterion is widely used in seismic processing.

In order to perform a more detailed bandwidth study, I have extended the analysis to other lags of $c_4^a(\tau_1, \tau_2, \tau_3)$. For this purpose, I generated a passband-type wavelet by subtracting two low-pass filters with cut-off frequencies f_l and f_h:

$$a(t) = 2f_h \frac{\sin(2\pi f_h t)}{2\pi f_h t} - 2f_l \frac{\sin(2\pi f_l t)}{2\pi f_l t} = 2B\,\mathrm{sinc}(\pi B t)\cos(2\pi f_0 t), \qquad (3.13)$$

where $B = f_h - f_l$ and $f_0 = (f_l + f_h)/2$. I then computed phase-shifted versions of the

wavelet, $a_\epsilon(t)$, which are related to $a(t)$ by the formula

$$a_\epsilon(t) = \cos(\epsilon)a(t) + \sin(\epsilon)\mathcal{H}\{a(t)\}, \tag{3.14}$$

where ϵ is the phase-shift and $\mathcal{H}\{\cdot\}$ is the Hilbert transform [AR80]. To determine the sensitivity of any lag of $m_4^{a_\epsilon}(\tau_1, \tau_2, \tau_3)$ to a phase-shift ϵ in the range $[0, \pi]$, I define the quantity

$$S_\epsilon(\tau_1, \tau_2, \tau_3) = \frac{|\max_\epsilon\{m_4^{a_\epsilon}(\tau_1, \tau_2, \tau_3)\} - \min_\epsilon\{m_4^{a_\epsilon}(\tau_1, \tau_2, \tau_3)\}|}{\max_\epsilon\{m_4^{a_\epsilon}(0, 0, 0)\}} \times 100\%, \tag{3.15}$$

This quantity expresses the variation of $m_4^{a_\epsilon}$ relative to the maximum kurtosis between the maximum and minimum values when ϵ varies in the range $(0, \pi)$. Figures 3.4a and 3.4b show a plot of S_ϵ as a function of B and f_0 for lags $(0, 0, 0)$ and $(3, 3, 3)$. Note that greatest sensitivities (around 50%) of both moment lags are attained when $B \simeq 2f_0$ (which implies that the bandwidth extends to zero frequency). On the other hand, for $B < f_0$ both moment lags are rather insensitive to a phase-shift. That is, the greater the bandwidth relative to the central frequency, the greater the sensitivity. Moment lags different from that corresponding to the kurtosis, show a similar sensitivity to phase-shifts. In general, the same behavior can be expected for other wavelets.

The above analysis demonstrates the data requirements for a meaningful maximum kurtosis deconvolution or 4th-order CM. In the presence of noise, the effective bandwidth of the signal may be reduced and hence the sensitivity of m_4 to wavelet phase will be also reduced. Levy and Oldenburg [LO87] recognized the importance of the low frequencies for the varimax norm to be sensitive to wavelet phase. To increase the effective bandwidth, whitening the data is recommended. This can be done by applying a zero-phase deconvolution. On the other hand, Hargreaves [Har94] proposed the application

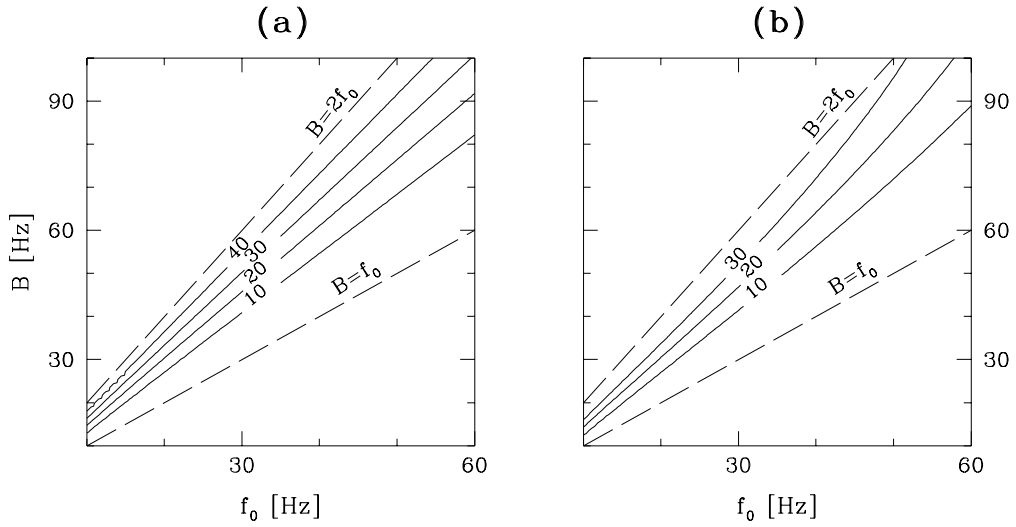

Figure 3.4: Fourth-order moment sensitivity to wavelet phase. (a) Sensitivity for lag (0,0,0) (i.e. kurtosis). (b) Sensitivity for lag (3,3,3). Below a critical bandwidth relative to the central frequency ($B \leq f_0$), the 4th-order moment is insensitive to wavelet phase. Maximum sensitivity is attained when the frequency content extends to zero frequency ($B \simeq 2f_0$).

of AGC (automatic gain control) to the signal, before wavelet estimation, to improve the SNR (signal-to-noise ratio) and hence the effective bandwidth. He showed that the reliability of the results of the CM method are considerably improved after the application of this simple preprocessing.

In summary, the sensitivity of the moment function to wavelet phase has been demonstrated for wavelets with different frequency contents. I showed that, on average, m_4 is as sensitive as the kurtosis to lags other than that at the origin. If $B < f_0$ both maximum kurtosis deconvolution and the 4th-order CM method will likely yield unreliable results, particularly for low SNR. In these critical cases, the results can be improved by applying either AGC and/or zero-phase deconvolution prior to wavelet estimation. It is also recommended that errors in the low frequency portion of the spectrum arising due

to spectral windowing (see [Tho82, Wal90]) are reduced as much as possible during the processing sequence prior to wavelet estimation.

3.4.4 Non-Gaussianity assumption

Various authors have established that the primary reflectivity series can indeed be modeled as a non-Gaussian process [WH86]. The distribution appears to be essentially symmetric with a sharper central peak and larger tails than that of a Gaussian distribution. These kind of distributions are called *leptokurtic* because the normalized kurtosis is greater than the kurtosis of a Gaussian distribution which is zero. This is the basis of the so-called minimum entropy deconvolution (MED) approaches where a spiky reflectivity is expected (see [Wal85] for a comprehensive discussion of several MED-type algorithms in connection with the non-Gaussianity assumption). The 4th-order CM approach is particularly effective when the reflectivity probability density function (pdf) is far from Gaussian.

The sensitivity of the CM method to certain types of reflectivity models has been tested in [Laz93]. In this work, the models consist of a Gaussian reflectivity raised to a power, γ, greater than one, preserving the sign (Gaussian-γ model). If y_k is the k-th sample of a zero-mean Gaussian process, then the actual reflectivity coefficient, r_k, is obtained by

$$r_k = |y_k|^{\gamma+1}/y_k, \qquad \gamma > 1. \tag{3.16}$$

These models have leptokurtic pdf's and it is this non-Gaussian nature that is observed in well-log data. Here, I consider also other kinds of reflectivity models: one that is essentially sparse, and one that has been proven to fit very well the empirical amplitude distribution of block-averaged well logs. The first one can be described by a Bernoulli-Gaussian process [KM78, GR81] with pdf

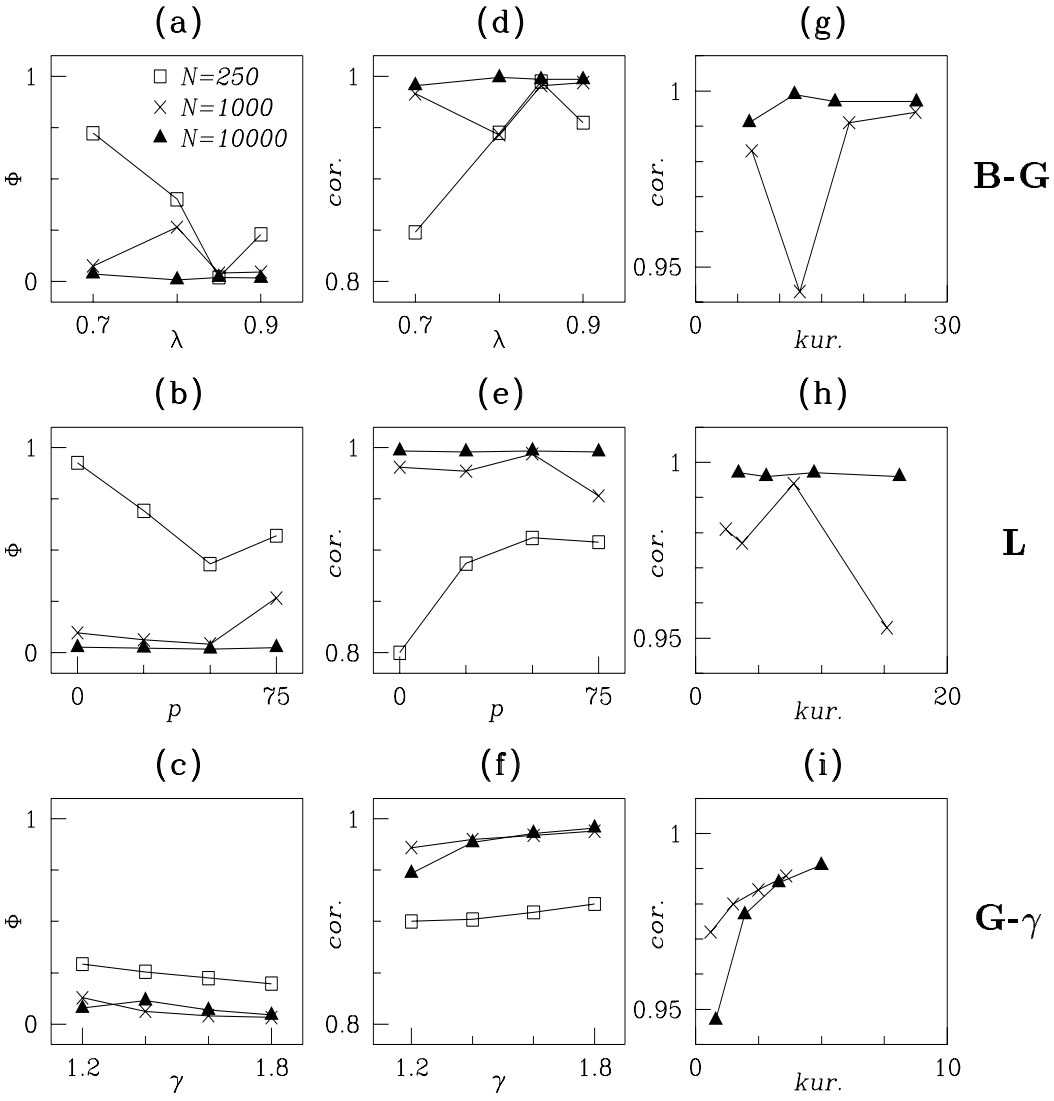

Figure 3.5: Non-Gaussianity assumption. From top to bottom, subsequent rows correspond to Bernoulli-Gaussian, Laplace mixture, and Gaussian-γ models. (a,b,c) Normalized cost function for the different models. This quantity expresses the matching between the true wavelet moment and the corresponding trace cumulant. (d,e,f) Correlation between each wavelet estimate and the actual wavelet for the different models. (g,h,i) Correlation versus kurtosis for the different models.

$$f(r) = \begin{cases} 0 & \text{with probability } \lambda \\ N(0, \sigma_r^2) & \text{with probability } 1 - \lambda, \end{cases} \tag{3.17}$$

where $N(0, \sigma_r^2)$ denotes a zero mean Gaussian distribution with variance σ_r^2, and $0 \leq \lambda \leq 1$. The normalized kurtosis can be easily computed and is equal to $3/(1-\lambda) - 3$. For obtaining realistic reflectivity models, λ ranges typically from 0.7 to 0.9, corresponding to models consisting of reflections of 1 in 3 samples (kurtosis=7), to 1 in 10 samples (kurtosis=27), approximately.

The second model (perhaps the one that is closest to a real reflectivity sequence from the point of view of stratigraphic considerations) is that in [WH86]. In this paper it is shown that real primary reflectivity sequences can be modeled as a mixture of two Laplace (or double-sided exponential) distributions:

$$f(r) = \begin{cases} \frac{1}{2\lambda_1} e^{-|r|/\lambda_1} & \text{with probability } p \\ \frac{1}{2\lambda_2} e^{-|r|/\lambda_2} & \text{with probability } 1 - p, \end{cases} \tag{3.18}$$

where λ_1 and λ_2 denote the scale parameters of each population and p the proportion of the mixture, $0 \leq p \leq 1$. This alternative model makes very clear the distinction between sedimentary beds and lithologic units by simply changing the proportion of the mixture. Such a model has normalized kurtosis $6(1+c) - 3$, $c \geq 0$, which is always greater than or equal to 3 (here c is a function of p, λ_1 and λ_2 only [WH86]). Several well-log data sets were fitted using this flexible model, and in all the cases shown in the aforementioned work, the results indicated a kurtosis exceeding 3.

I generated several reflectivity sequences according to the three models (i.e. Gaussian-γ, Bernoulli-Gaussian, Laplace mixture) using various parameters. The models were intended to reproduce real reflectivity sequences and the purpose was to test the effectiveness of the CM method to recover the true wavelet for the various models using

different amounts of data. A zero-phase Ricker wavelet was used to generate each seismogram, and 10% by amplitude Gaussian noise was added. Figures 3.5a to 3.5c illustrate the results of computing the normalized mean-squared error between the trace cumulant estimate and the true wavelet moment:

$$MSE = \frac{\sum_{\tau_1,\tau_2,\tau_3}[c_4^x(\tau_1,\tau_2,\tau_3) - \alpha m_4^a(\tau_1,\tau_2,\tau_3)]^2}{\sum_{\tau_1,\tau_2,\tau_3}[c_4^x(\tau_1,\tau_2,\tau_3)]^2}, \tag{3.19}$$

where α is a constant that minimizes MSE. Also, I have computed the correlation of the estimated wavelets with the true wavelet, which is shown in Figures 3.5d to 3.5f. As expected, the matching increases with the amount of data. In general, good estimates were obtained even for models with small kurtosis and a relatively small amount of data (correlations exceeding 0.95 were obtained for most of the cases). For sparse models, I have found excellent results using as few as 250 samples. In Figures 3.5g to 3.5i I have plotted the effective kurtosis (as computed from the reflectivity series) versus wavelet correlation for models using 1,000 and 10,000 data samples. It is interesting to note that there is not a rigorous correlation between kurtosis and accuracy of results. This may be true only for some specific models (e.g. the Gaussian-γ) since the results appear to depend on the manner in which the wavelet interferes with the structure of the hidden reflectivity coefficients. For example, consider Figure 3.5i corresponding to the Gaussian-γ model. The higher the kurtosis, the higher the correlation between true and estimated wavelets. This is not the case for Bernoulli-Gaussian and Laplace models (Figures 3.5g to 3.5h).

3.5 Real data examples

3.5.1 Field data

In this example, the field data I used are shown in Figure 3.6a. The window consists of 21 traces, from 1.0 to 2.0 s. To illustrate the trace-by-trace process, I obtained an individual wavelet estimate for each trace and then averaged the optimally shifted estimates to obtain an average estimate. Only 250 samples were used to obtain each individual estimate. The consistency obtained in this example is clear from inspection of the individual estimates plotted in Figure 3.7a. The average estimate is shown in Figure 3.7b. If all data are used simultaneously to estimate the trace cumulant, the wavelet in Figure 3.7c is obtained. This is very similar to the wavelet obtained after synchronizing the individual wavelets.

To assess the reliability of the wavelet estimate I deconvolved the original data by spectral division using the average estimate. The "zero-phase" section is shown in Figure 3.6b. These data were in turn used to obtain the residual wavelets which are shown in Figures 3.7d to 3.7f. As expected, the residual wavelets are very close to zero-phase, a fact that demonstrates the reliability of the wavelet estimate.

3.5.2 Marine data

The real data in this example come from a marine section which had been processed carefully for impedance recovery. Figure 3.8a shows the 24 traces (325 samples per trace) used for wavelet estimation. The 24 individual wavelet estimates are shown in Figure 3.8b. The average wavelet estimate, after the hybrid trace-by-trace strategy and stacking the wavelets in panel (a) of the same figure, is illustrated in Figure 3.8c. As expected, due to the pre-processing steps employed, the recovered wavelet is very nearly zero-phase. Finally, the wavelet shown in Figure 3.8d corresponds to the estimate using

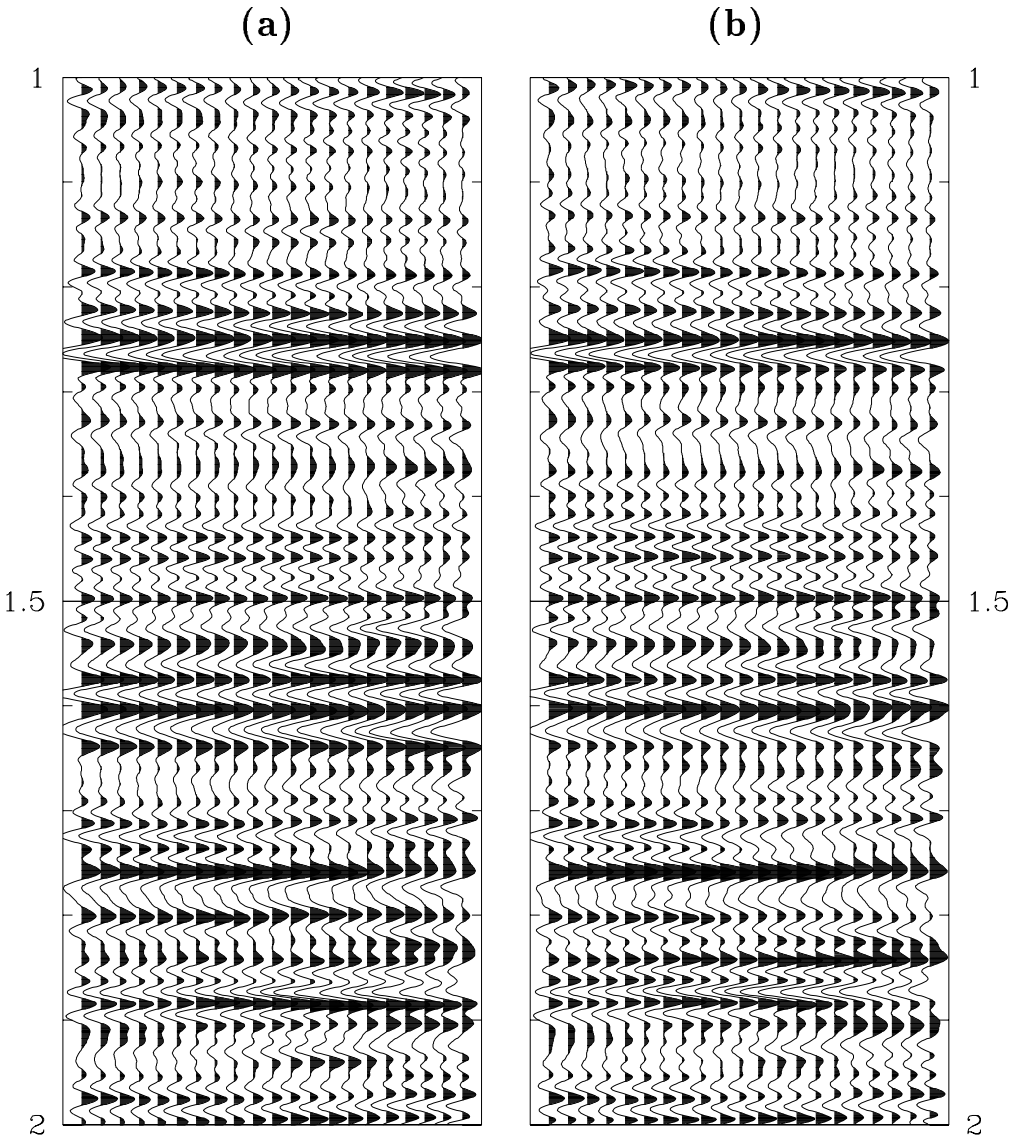

Figure 3.6: Field data example: input data. (a) Original data section used for a trace-by-trace CM wavelet estimation (b) Same section in (a) after removing the wavelet phase using the average estimate.

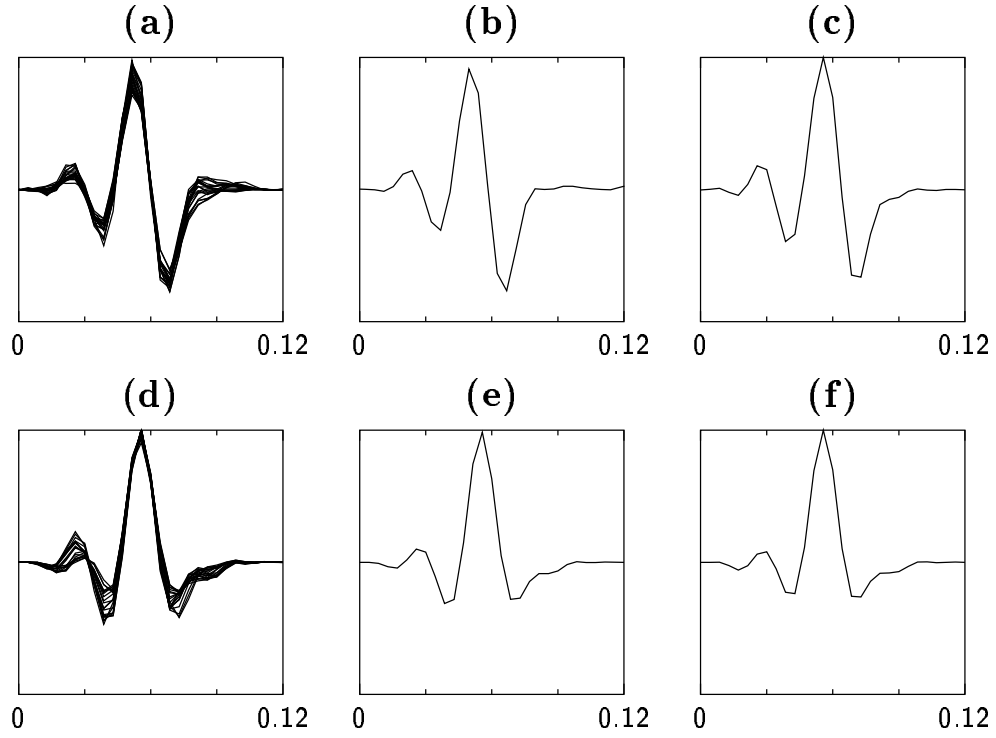

Figure 3.7: Field data example: wavelet estimates. (a) 21 individual wavelet estimates of the section shown in Figure 3.6a. (b) Average wavelet estimate after synchronizing and stacking all the individual estimates. (c) Wavelet estimate using all data simultaneously. (d), (e), and (f) residual wavelets corresponding to the zero-phased section in Figure 3.6b.

all data simultaneously.

3.6 CM in the frequency domain

In as much as the information contained in the autocorrelation sequence is essentially present in the power spectrum, so the information contained in higher-order moments and cumulants is present in *polyspectra* (higher-order spectra). This duality is established upon the polyspectra definition. Polyspectra are defined, naturally, as the Fourier transform of the corresponding time domain sequence. For instance, the *bispectrum*

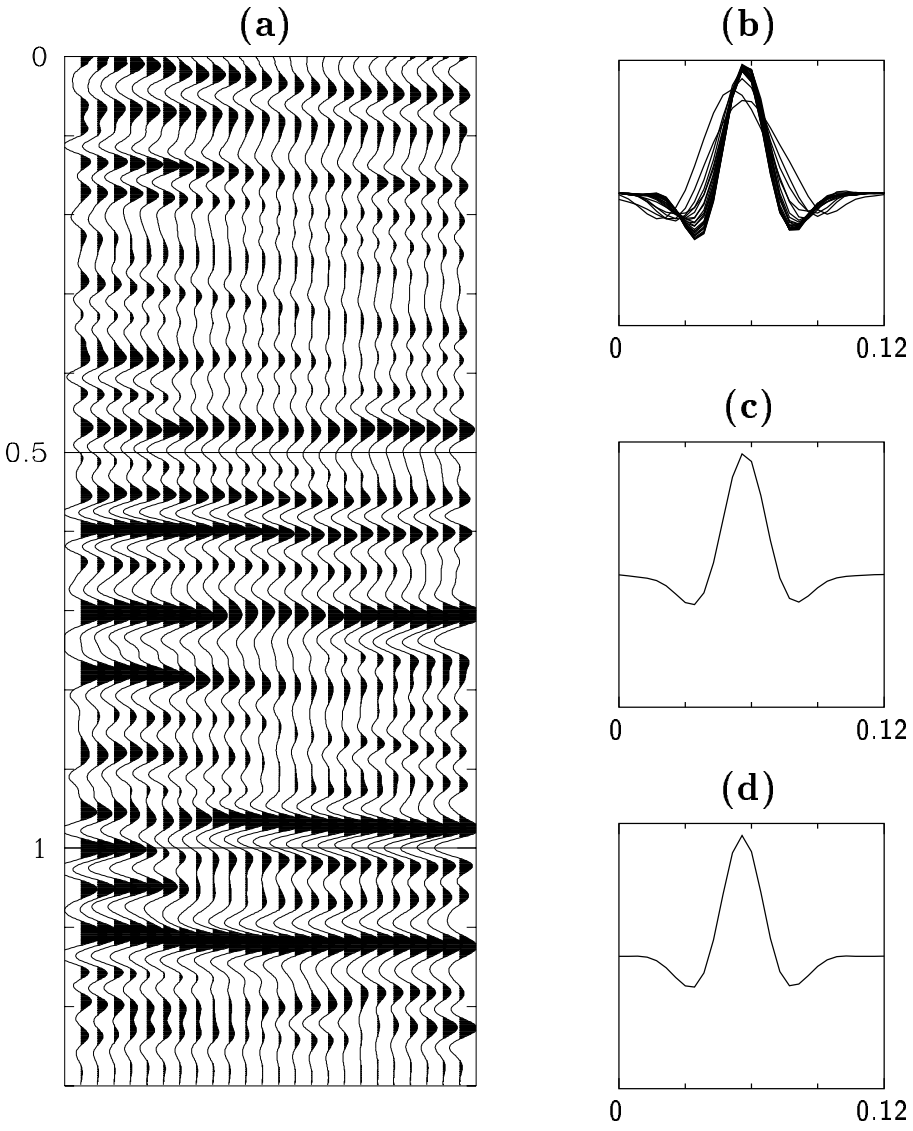

Figure 3.8: Marine data example. (a) Section of data used in the trace-by-trace CM wavelet estimation. (b) 24 individual wavelet estimates. (c) Average wavelet estimate. (d) Wavelet estimate using all data simultaneously.

(third-order spectrum) is defined to be the Fourier transform of the third-order cumulant sequence, and the *trispectrum*, which is the fourth-order spectrum, is defined to be the Fourier transform of the fourth-order cumulant sequence. It is worthwhile to make a distinction between two types of polyspectra: (1) *cumulant spectra*, which are particularly useful in the analysis of stochastic processes, and (2) *moment spectra*, which are of great importance in the analysis of deterministic signals.

Since we are interested in fourth-order statistics, let $x(t)$, $t = 0, \pm 1, \pm 2 \cdots$, be a real stationary discrete-time random process with fourth-order cumulant sequence $c_4^x(\tau_1, \tau_2, \tau_3)$. Then, the 4th-order cumulant spectrum (trispectrum) of the process $x(t)$ is defined as the 3-D Fourier transform

$$C_4^x(\omega_1, \omega_2, \omega_3) = \sum_{\tau_1=-\infty}^{+\infty} \sum_{\tau_2=-\infty}^{+\infty} \sum_{\tau_2=-\infty}^{+\infty} c_4^x(\tau_1, \tau_2, \tau_3) \exp[-j(\omega_1\tau_1 + \omega_2\tau_2 + \omega_3\tau_3)], \quad (3.20)$$

where $|\omega_i| \leq \pi$ for $i = 1, 2, 3$, and $|\omega_1 + \omega_2 + \omega_3| \leq \pi$ [Bri65]. In general, $C_4^x(\omega_1, \omega_2, \omega_3)$ is complex, and it is periodic with period 2π in each dimension. Many symmetry properties can be derived due to the fourth-order cumulant definition. As a result, there is a lot of redundant information in the cube $|\omega_1 + \omega_2 + \omega_3| \leq \pi$ (see for example [PIIF92]).

In the case of real discrete-time deterministic signals, $a(t)$, the fourth-order moment spectrum is defined as the Fourier transform of the fourth-order moment sequence. A more compact and interesting expression can be obtained in terms of the Fourier transform of the signal, $A(\omega)$, after manipulating algebraically the corresponding definitions:

$$M_4^a(\omega_1, \omega_2, \omega_3) = A(\omega_1)A(\omega_2)A(\omega_3)A^*(\omega_1 + \omega_2 + \omega_3), \quad (3.21)$$

where $|\omega_i| \leq \pi$ for $i = 1, 2, 3$; $|\omega_1 + \omega_2 + \omega_3| \leq \pi$, and "$*$" denotes conjugate. In terms of magnitude and phase, equation (3.21) is equivalent to

$$\begin{cases} |M_4^a(\omega_1,\omega_2,\omega_3)| = |A(\omega_1)||A(\omega_2)||A(\omega_2)||A(\omega_1+\omega_2+\omega_3)| \\ \Psi_4^a(\omega_1,\omega_2,\omega_3) = \phi_a(\omega_1)+\phi_a(\omega_2)+\phi_a(\omega_2)-\phi(\omega_1+\omega_2+\omega_3), \end{cases} \tag{3.22}$$

where $|A(\omega)|$ and $\phi_a(\omega)$ are the magnitude and phase spectrum of the signal $a(t)$, respectively.

Back to the wavelet estimation problem, taking Fourier transform to equation (3.8) for N finite yields

$$\hat{C}_4^x(\omega_1,\omega_2,\omega_3) \simeq \gamma_4^r A(\omega_1)A(\omega_2)A(\omega_3)A^*(\omega_1+\omega_2+\omega_3), \tag{3.23}$$

where $A(\omega)$ is the Fourier transform of the seismic wavelet. This expression states that then trispectrum of the trace approximates, within a scale factor (and a linear phase shift), the fourth-order moment spectrum of the wavelet. In this context, there are several methods in the literature for recovering the signal from high-order spectra. These include phase recovery algorithms [LR82, MU84], phase and magnitude recovery algorithms, etc. (see for example [NP93], Chapters 6&7, for a concise description of various methods). One of such methods is based on the least squares optimization of the following cost functions:

$$\begin{cases} \Phi_{phase} = \sum_{\omega_1}\sum_{\omega_2}\sum_{\omega_3}[\hat{\Psi}_4^x(\omega_1,\omega_2,\omega_3)-\Psi_4^a(\omega_1,\omega_2,\omega_3)]^2 \\ \Phi_{magnitude} = \sum_{\omega_1}\sum_{\omega_2}\sum_{\omega_3}[|\hat{C}_4^x(\omega_1,\omega_2,\omega_3)|-\gamma_4^r|M_4^a(\omega_1,\omega_2,\omega_3)|]^2 \end{cases} \tag{3.24}$$

where $\hat{\Psi}_4^x(\omega_1,\omega_2,\omega_3)$ and $|\hat{C}_4^x(\omega_1,\omega_2,\omega_3)|$ are estimated from the data. These cost functions establish the duality between the CM in the time domain and the CM in the frequency domain. Once $\hat{\Psi}_4^x(\omega_1,\omega_2,\omega_3)$ and $|\hat{C}_4^x(\omega_1,\omega_2,\omega_3)|$ are estimated, VFSA can be applied to minimize the sums with respect to the wavelet spectral coefficients. Alternatively, the cost functions may be written as the squared error between the real and

imaginary parts of data and wavelet polyspectral coefficients. Though not tested here, the results in the frequency domain are expected to be (apart from multidimensional unwrapping and other computational issues that may arise) similar to the results in the time domain, for the information present in the polyspectra is essentially contained in the corresponding higher-order cumulant or moment sequence, as mentioned in the first paragraph of this section.

Finally, it is worth mentioning that an alternative and elegant approach is to use the cepstrum of higher-order cumulants (*polycepstra*), and in particular the *tricepstrum* [SVU98]. This method allows one to obtain the wavelet coefficients analytically from the estimated tricepstrum of the data. The analysis of these techniques is beyond the scope of this work.

3.7 Conclusions

1. The VFSA optimization should be seen as a method for assessing the reliability of the numerical solution, and a way to provide consistent results regardless of the initial model used in the optimization process. This leads to a hybrid strategy that combines the fast performance of the linearizing technique with the reliability of the VFSA solution.

2. Limited effective bandwidth and missing low frequencies in the recorded data reduce the 4th-order cumulant sensitivity to wavelet phase. This effect may be partially compensated by either whitening the data or performing AGC before carrying out the wavelet estimation.

3. It is assumed that the reflectivity is a non-Gaussian process. This assumption appears, in fact, to represent a property of the earth's reflectivity. Further, if the reflectivity is also somewhat sparse, which is not an unreasonable expectation,

the CM method may be used with confidence with relatively short data segments. In these cases, the application of a multidimensional taper to smooth the trace cumulant estimate and improve the matching, is strongly recommended.

4. The combination of both the hybrid strategy and a convenient multidimensional tapering, permitted us to develop an efficient and consistent trace-by-trace implementation. Numerical consistency is achieved by the VFSA algorithm and tapering, CPU efficiency is attained by the fast performance of a linearizing technique.

5. Despite the inherent limitations of the CM method due to bandwidth content and amount of data available, the results obtained both with synthetic and real data demonstrate not only the viability of the method, but also the reliability of the convolutional model which is at the heart of wavelet estimation philosophy.

Chapter 4

Two-Dimensional Boundary Value Ray Tracing

Begin at the beginning...and go on till you come to the end, then stop.

Lewis Carroll – *Alice's Adventures in Wonderland*

4.1 Introduction

Ray tracing plays a key role in seismological studies. Large attention has been devoted to the *initial-value problem* (IVP), in which the ray is specified by the initial conditions: initial point and initial direction of propagation (*take-off angle*). The IVP is in general a well resolved problem (see for example [Čer87]). However, geotomographic methods and earthquake location usually require precise traveltime and trajectory computations of seismic waves propagating between two fixed points in an laterally heterogeneous medium. This represents a *boundary-value problem* (BVP) because the ray is not only specified by the initial conditions. In a more general case a ray may be specified by more complicated boundary conditions at different points of its trajectory. Such is the case of ray tracing reflected or headwaves connecting two fixed points.

Traditionally, there are two techniques to find the raypath of a seismic wave propagating between two points in a heterogeneous medium: shooting and bending [JG77, AR80]. The method of finite-differences [Vid88] has become widely used especially for dealing with smooth velocity fields. The first arrival traveltimes field is first calculated at all nodes of a gridded model, and then raypaths are traced by following the wavefront normals. Most of these techniques perform very well for the particular type of velocity model

they were devised for. But in general they are applicable to a limited class of models, and the more complex the media, the more obvious become their limitations. Other techniques include graph theory methods, recently developed for obtaining the shortest paths in a gridded model [Mos89, FL93]. Though these methods work well for moderately complex 2-D media, their applicability for dealing with complex 3-D structures has not been studied thoughtfully, especially because it becomes computationally very much involved when a certain accuracy is required.

In this chapter I develop a new ray-tracing method for solving the BVP through a heterogeneous 2-D medium. Although a more general technique for solving the BVP through an arbitrary 3-D medium will be developed in next chapter, the ray-tracing method will be called "simulated annealing ray tracing" (SART) throughout. The purpose of SART is to overcome the usual deficiencies of shooting and bending for solving the BVP. The philosophy of the method is to put the BVP into a convenient optimization framework which is in turn solved by means of an efficient simulated annealing algorithm like VFSA. In the two-point ray-tracing case, SART is an iterative procedure that attempts to find the optimum take-off angle corresponding to the raypath with minimum traveltime connecting any given source-receiver pair [Vel96]. The main advantage of SART lies in the fact that it overcomes the problem of multipathing inasmuch as convergence to the absolute minimum traveltime is statistically guaranteed. This is demonstrated by a number of synthetic examples where not only direct waves are traced, but also reflected (including normal rays) and headwaves.

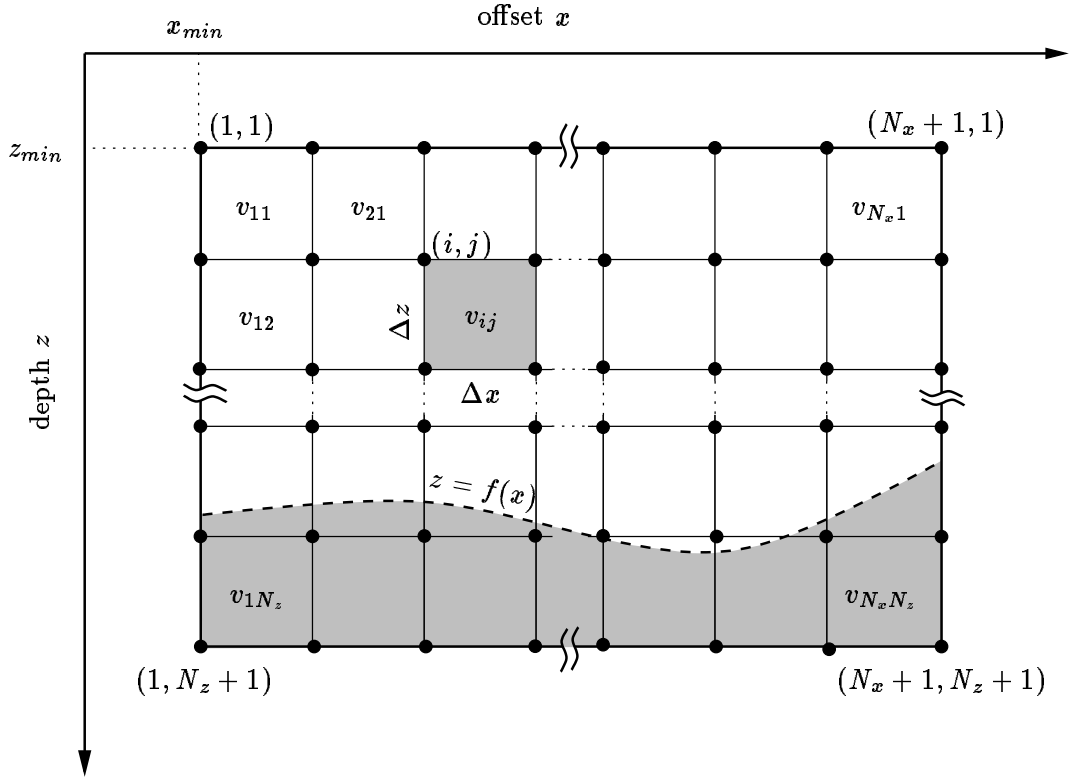

Figure 4.1: Two-dimensional cell parameterization. The velocity field is piece-wise constant. The function $z = f(x)$ represents a reflector/refractor.

4.2 Earth model

The modeling of a complex geology in a way that it can be input and easily manipulated by a computer program is not a trivial task. Usually, it is more difficult and time consuming to devise a mathematical method for representing real velocity fields than devising a method for tracing rays through it. Fortunately, some simple forms of parameterization can approximate very well real subsurface velocity structures for the purpose of ray tracing and understanding of the method.

In the 2-D case I use a cell ray-tracing method which is fast and accurate for the purposes of this work. This kind of parameterization requires the velocity field to be

represented by rectangular cells of constant velocity and fixed size. The model consists of N_x by N_z rectangular cells of size Δx by Δ_z which define $N_x + 1$ by $N_z + 1$ cell nodes (Figure 4.1). Node $(1, 1)$ has coordinates (x_{min}, z_{min}) as shown in the Figure. The two-dimensional velocity field is represented by $v = v(x, z)$, where x stands for offset distance and z for depth (positive z-axis points downwards). Velocity v may correspond to either P- or S-wave velocity of propagation. Each rectangular cell is assigned the velocity corresponding to the upper-left node, thus

$$v_{ij} = v\left[x_{min} + (i-1)\Delta x, z_{min} + (j-1)\Delta z\right], \qquad i = 1, \cdots, N_x; \quad j = 1, \cdots, N_z. \quad (4.1)$$

Additionally, a predefined interface $z = f(x)$ can be superposed for tracing reflections or headwaves as will be described later. Although this way of representing the velocity field presents some limitations for tracing particular raypaths (e.g. turning rays), the simplicity associated with the geometry of the ray trajectory makes it attractive. The raypath will be composed of a series of segments honoring Snell's Law at each cell boundary. In the next chapter, a more flexible and general model parameterization will be proposed.

4.3 Solving the initial-value problem (IVP)

4.3.1 Two-dimensional cell ray tracing

Before describing the BVP it is essential to describe the IVP for the model parameterization at hand. To solve the IVP I use a cell ray-tracing scheme where the velocity within each rectangular cell is assumed to be constant. Both traveltime and ray trajectory are to be found. The ray is propagated using Snell's Law at each cell boundary, and thus it

is composed of a number of segments. Intersection points and angles are obtained using simple geometrical relations as will be described in the following paragraphs.

Four cases can be distinguished according to which edge the ray enters a given cell: *left, top, right* or *bottom*. I will describe the first case only, since the others follow a similar reasoning. Let us suppose the ray enters cell (i, j) from left edge at point $\mathbf{x}_k = (x_k, z_k)$, as shown in Figure 4.2a, where k denotes the k-th point in the ray trajectory. The next point will lie either on the top, right or bottom edges of the current cell, depending on the angle θ_k and the cell geometry:

$$\begin{cases} 0 < \theta_k \leq \varphi_a & \text{bottom}, \\ \varphi_a < \theta_k \leq \varphi_b & \text{right}, \\ \varphi_b < \theta_k < \pi & \text{top}. \end{cases} \tag{4.2}$$

where angles φ_a and φ_b are computed using

$$\begin{cases} \varphi_a = \frac{\pi}{2} - \arctan\left[\frac{z_{min} + j\Delta z - z_k}{\Delta x}\right], \\ \varphi_b = \frac{\pi}{2} - \arctan\left[\frac{z_k - z_{min} - (j-1)\Delta z}{\Delta x}\right]. \end{cases} \tag{4.3}$$

The three subcases are depicted in Figures 4.2b to 4.2d, and the formulas required to compute coordinates (x_{k+1}, z_{k+1}), angle θ_{k+1}, and traveltime within the current cell are summarized in Table B.1.

Snell's Law is applied at point $\mathbf{x}_{k+1} = (x_{k+1}, z_{k+1})$ to determine the next direction:

$$\frac{\sin \alpha_k}{v_k} = \frac{\sin \alpha_{k+1}}{v_{k+1}}, \tag{4.4}$$

where α_k and α_{k+1} are the angles between the ray trajectory and the normal to the corresponding cell edge, and v_k and v_{k+1} are the velocities at opposite sides. For example, in the case of Figure 4.2b,

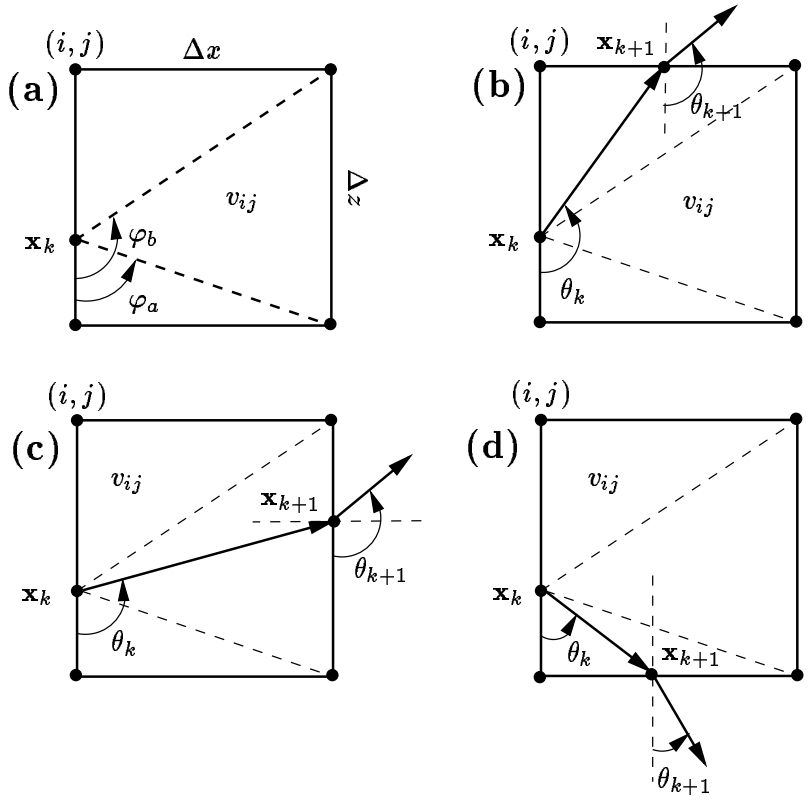

Figure 4.2: Detailed geometry of a ray entering cell (i, i) from the left edge. (a) The ray enters at point $\mathbf{x}_k$ which determines angles ϕ_a and ϕ_b. (b), (c) and (d) Depending on θ_k resulting from Snell's Law, point $\mathbf{x}_{k+1}$ is then computed. See Table B.1 and text for details.

$$\begin{cases} \alpha_k = \pi - \theta_k, \\ \alpha_{k+1} = \pi - \theta_{k+1}, \\ v_k = v_{ij}, \\ v_{k+1} = v_{i,j-1}. \end{cases} \tag{4.5}$$

This process is repeated until the ray exits the model boundaries or crosses the predefined discontinuity, where a special attention must be paid. Finally, total traveltime is computed as the sum of all the individual contributions corresponding to each ray

segment:

$$T = \sum_k t_k. \tag{4.6}$$

Figure 4.2 illustrates one of the four possible cases: $\mathbf{x}_k$ lying on the left edge of the cell. As a result there is a total of twelve subcases depending on the location of $\mathbf{x}_k$ and the angle θ_k. I will not describe them all here. The reader is referred to Appendix B where all the required formulas are summarized.

In summary, starting from the source, $\mathbf{x_0} = \mathbf{x}_s$, a sequence of segments is generated according to simple geometrical relations and Snell's Law at cell boundaries or predefined discontinuities. At each step, it must be determined which edge of the cell the ray comes from, so that one of the four mentioned cases is applied. Also, each segment of the ray which is being computed must be checked to see whether it crossed the predefined reflector (or refractor) $z = f(x)$.

Special cases

A distinction must be made whenever the current point lies within the cell boundaries (e.g. when the source does not lie exactly on some cell edge), and when the ray trajectory arrives at some predefined discontinuity (e.g. a reflector). These situations are depicted in Figure 4.3a and 4.3b respectively. In the first case, it is possible to use the same formulas as in the normal case (Table B.1) by simply exchanging Δx by $\Delta x' = i\Delta x - x_k$ as shown in Figure 4.3a. In the second case (Figure 4.3b), a more elaborate situation must be considered. After shooting from $\mathbf{x}_k$ with angle θ_k as usual, the auxiliary point $\mathbf{x}'_{k+1}$ is determined. If a reflector crossing is detected, the intersection point $\mathbf{x}_{k+1}$ should be obtained becoming the next point in the ray trajectory. At this point, Snell's Law of reflection (or refraction if required) is applied and angle θ_{k+1} so determined. Now, the

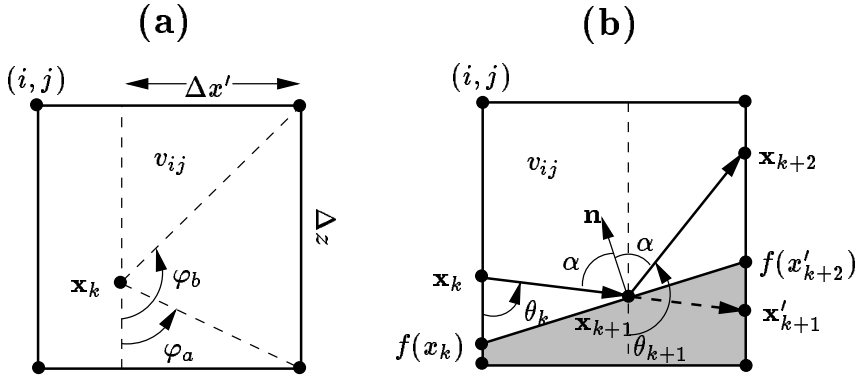

Figure 4.3: Special cases of the ray geometry. (a) Shooting from inside-left a given cell (i,j). (b) The ray meets a reflector at point $\mathbf{x}_{k+1}$ and travels towards point $\mathbf{x}_{k+2}$ after undergoing a reflection.

propagation continues as in the case of Figure 4.3a. When a refraction is requested at this intersection point, the velocity on either side of $z = f(x)$ is generally different. This means that some cells have two velocity values up and below the predefined refractor.

The reflector (or refractor) $z = f(x)$ is approximated by a straight line within each cell so that the intersection with the raypath is found in one step. After some simple geometrical relations, the coordinates of point $\mathbf{x}_{k+1}$ are determined with

$$\begin{cases} x_{k+1} = x_k + (x'_{k+1} - x_k)\frac{z_k - f(x_k)}{f(x'_{k+1}) - f(x_k) - z'_{k+1} + z_k}, \\ z_{k+1} = f(x_{k+1}), \end{cases} \tag{4.7}$$

which are the coordinates of the intersection point between the raypath segment connecting $\mathbf{x}_k$ with $\mathbf{x}'_{k+1}$, and the reflector (or refractor) segment connecting $f(\mathbf{x}_k)$ with $f(\mathbf{x}'_{k+1})$ respectively. An iterative procedure (not assuming the reflector/refractor is a straight line within each cell) to find the intersection point could also be tried. But the approximation made is accurate enough for the purposes of this work, provided the reflector (refractor) is a smooth function and the number of cells is relatively large.

Ray signature and stopping conditions

It is also required to define a sort of ray signature at the beginning of the process to make a decision after function $z = f(x)$ has been crossed. Having defined the ray signature, it is possible to trace ray conversions by using different velocities after a reflection (or refraction) has occurred. For example, a ray signature may be simply assigning a series of binary numbers $IOP(I)$, $I = 1, 2, \cdots$, associated with the discontinuity. These numbers are expressed as decimal integer values. Index I indicates the number of times the discontinuity is met, thus allowing for more than one reflection, refraction and/or conversion at different points of the same discontinuity. For example, the first time the discontinuity is met, the number $IOP(1)$ is used to make the decision, the second time, $IOP(2)$, etc. Each bit of $IOP(I)$ may be either 0 or 1. The first bit is reserved for existence of the discontinuity, the second for refraction, the third for reflection, and the fourth for phase conversion. A "0" means to ignore the corresponding event, and a "1" means to follow the directives. Table 4.1 shows the decisions to make according to each bit value.

As an example let $IOP(1) = 13$ (binary 1101) and $IOP(2) = 1$ (binary 0001). The first time the discontinuity is met, the ray is requested to undergo a reflection and a change of phase. If by chance the discontinuity is met again, the propagation is requested to stop. If refraction is required setting bit 2 to 1 instead, but a total reflection occurs, the propagation is required to stop by default. In the case a discontinuity is not defined or IOP(I) is set to 0, the propagation ends when point $\mathbf{x}_{k+1}$ lies on some model boundary (i.e. $x_{k+1} = x_{max}$ and/or $z_{k+1} = z_{max}$), when $k > k_{max}$, or when $T > T_{max}$, where k_{max} and T_{max} are user-predefined values.

bit	0	1
1	ignore $z = f(x)$	do not ignore $z = f(x)$
2	no refraction	refraction
3	no reflection	reflection
4	no phase conversion	phase conversion

Table 4.1: Decision table indicating what to do when discontinuity $z = f(x)$ is met at some point of the ray trajectory. See text for explanation.

4.3.2 Pros & cons of cell ray tracing: a discussion

As already mentioned, cell ray tracing with piecewise constant velocity presents some limitations, especially for tracing turning rays. The problem arises when at a certain cell edge a total reflection occurs due to large incident angles. As a counterpart, the method is computationally inexpensive, simple and accurate enough for the purposes of illustrating the two-point ray-tracing method that will be described in next sections. The total reflection is artificially introduced by the cell parameterization, because of the first-order discontinuities in the velocity that are generated (Figure 4.4a). The question is how to deal with large incident angles that may certainly occur. The answer would be to try to eliminate first-order discontinuities. First-order discontinuities may be eliminated by using nonconstant velocities within each cell. For rectangular cells, the simplest form which admits analytical solution and ensures continuity is the quadratic slowness:

$$v^{-2}(x, z) = a + bx + cz + dxz, \tag{4.8}$$

where a, b, c and d are constants [Čer87]. Clearly, the ray trajectory would be no longer composed of a series of straight segments. Although the shape of the ray for a velocity function as given in equation (4.8) is known, the computational cost and coding complexity would be much greater than using piecewise constant velocities because

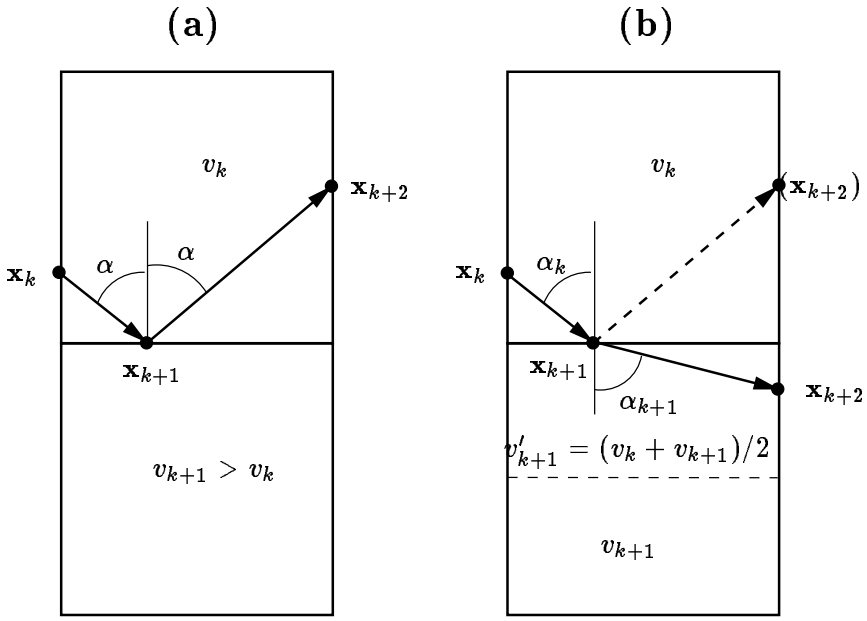

Figure 4.4: (a) Total reflection occurs when the angle $\alpha_k > \alpha^c$, where α^c is the critical angle derived from Snell's Law: $\sin\alpha^c = v_k/v_{k+1}$, $v_k < v_{k+1}$. This is a fictitious reflection introduced by the cell parameterization. (b) A method to eliminate the fictitious reflection in (a). The cell with velocity v_{k+1} is subdivided into two smaller cells, one of them with velocity $v'_{k+1} = (v_k + v_{k+1})/2 < v_{k+1}$, which makes the new critical angle, $\alpha^{c'}$, smaller. As a result $\alpha_k < \alpha^{c'}$ and a refraction occurs.

trigonometric and hyperbolic functions are involved. Alternatively, triangular cells could be used. Now

$$v(x,z) = a + bx + cz, \tag{4.9}$$

is the simplest form of velocity which ensures continuity. The raypath is an arc of a circle, which also leads to more expensive computations than using straight segments.

However, in the case of piecewise constant velocity cells, if the velocity field is smooth and the size of the cells is relatively small, total reflections do not represent a serious problem. In this case total reflection would occur only for incident angles close to 90

degrees. This issue does not mean that the velocity field must be smooth for the cell ray-tracing scheme to be an accurate procedure. Jumps in the velocity model can be introduced without loosing accuracy provided that these velocity jumps coincide with cell boundaries. Besides, additional discontinuities of arbitrary shape can be introduced through function $z = f(x)$ as a refractor, as shown in this section. So far the method I have developed allows only one function $z = f(x)$ to be defined, but more than one could be similarly introduced for greater flexibility in modeling complex structures.

The use of smaller cells would obviously increase computational cost, but not in the same measure as in the case of nonconstant velocity cells. An alternative not tested in this work would be to use some kind of adaptive gridding so that cells are larger where velocity is smooth, and smaller where velocity varies rapidly. This can be done only at those cells where a total reflection has been detected. At this point the cell is subdivided into smaller cells until a refraction occurs, as illustrated in Figure 4.4b. However, this scheme does not eliminate the problem at all. Suppose the velocity increases linearly with depth and a ray is shot from the source with $\theta_s = 45^o$. In theory, the ray follows an arc of a circle and travels downwards until a maximum depth where $\theta = 90^o$ (turning point). Then it starts going upwards. It is clear that this path cannot be reproduced by the described cell ray-tracing scheme, because once $\theta = 90^o$ has been reached, there is no chance for θ to take on greater values. A better alternative is to use a constant gradient velocity within the corresponding cell. Locally, the raypath is an arc of a circle of known equation, thus allowing for turning rays. Although cell ray tracing can be improved in several ways, these kind of refinements will not be considered here. Instead, in next chapter I will present a numerical ray-tracing method that is more flexible and accurate for dealing with complex 2-D or 3-D media.

4.4 Solving the boundary-value problem (BVP)

4.4.1 Problem definition

In general, it is the goal of a ray-tracing BVP solver to find the raypath connecting any two fixed points within the model boundaries (two-point BVP). The raypaths must be consistent with the ray theory which is used to propagate the wave through the given velocity field (e.g. Snell's Law at each cell boundary in a cell ray-tracing scheme). Usually, this is carried out by finding the unknown take-off angle that gives rise to a ray propagating from the source to the receiver. In general, the BVP must be solved iteratively, except in special cases. Although the solution sometimes is nonunique (several raypaths may connect source and receiver satisfying the ray equations), the purpose of SART is to find the raypath that exhibits the absolute minimum traveltime.

At this point it is worthwhile to make a distinction among various BVP alternatives. In the standard two-point ray-tracing problem, the wave travels from source to receiver without any additional constraint along its trajectory. I refer to this BVP as tracing *direct* waves. Other BVP alternatives introduce additional constraints along the ray trajectory. Such is the case of tracing reflections (including normal rays), headwaves, or point diffractions.

Direct waves

Consider Figure 4.5. Let $\mathbf{x}_s$ represent the location of the source and $\mathbf{x}_r$ the location of the receiver. These two points are fixed in space and must lie within the model boundaries. For the sake of simplicity, I put them coincident with some of the model boundaries, but they could be placed in the interior of the rectangular area shown in the figure as well. In a tomographic experiment, for example, the source is detonated at $\mathbf{x}_s$ at time $t = 0$ and the energy propagates in all directions. In particular, the ray that leaves the source with

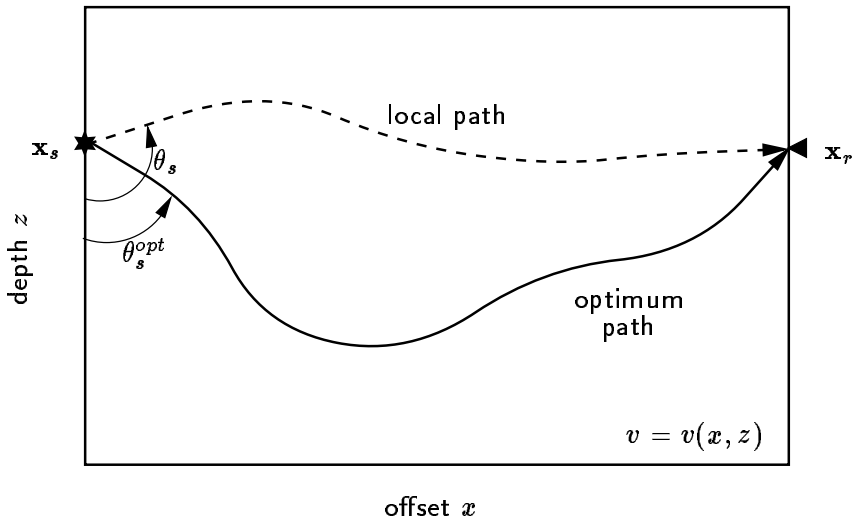

Figure 4.5: Source, receiver and raypath geometry in a two-dimensional two-point experiment for tracing direct waves.

angle θ_s^{opt} arrives to the receiver with the minimum traveltime among all other possible trajectories. Here θ_s^{opt} determines uniquely the raypath shown in the figure. It is the purpose of SART to find θ_s^{opt} as well as the associated traveltime.

Reflections

In a reflection experiment (see Figure 4.6), not only must the ray connect both source and receiver, but it must also undergo a reflection at a prescribed reflector. The reflector can be described by a single-valued function of the form $z = f(x)$. The absolute minimum raypath corresponds to the one that leaves the source with take-off angle θ_s^{opt}, undergoes a reflection at point $\mathbf{x}_u$ and arrives to the receiver. The only unknown is θ_s^{opt}, for $\mathbf{x}_u$ is uniquely determined by the take-off angle.

Normal rays can be viewed as a particular case of reflected waves. Now both source and receiver are coincident. Naturally, the angle between the ray trajectory and the

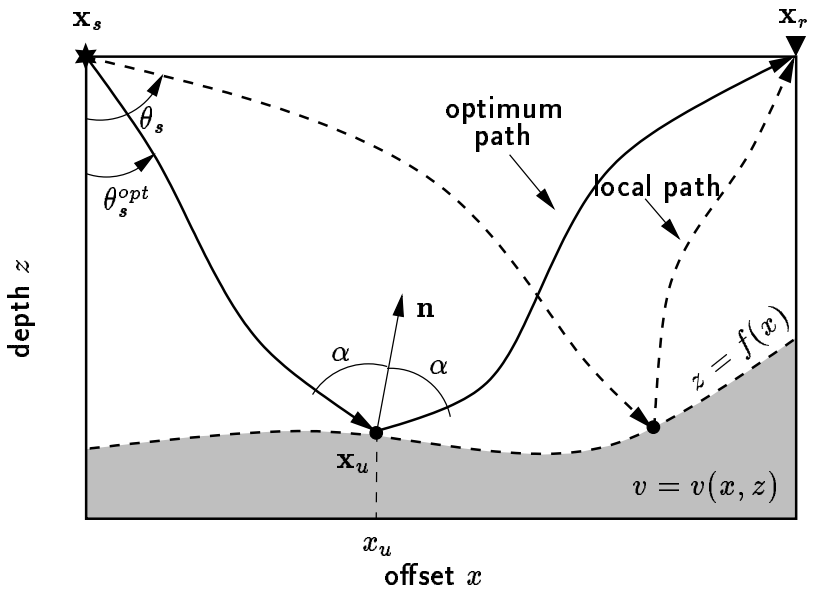

Figure 4.6: Source, receiver, reflector and raypath geometry in a two-dimensional two-point experiment for tracing reflections.

normal to the reflector at point $\mathbf{x}_u$ is zero. Rather than finding the take-off angle corresponding to the minimum reflecting raypaths, is is more efficient to find the coordinate x_u because half of the raypath needs to be traced.

Headwaves

Figure 4.7 depicts the case of tracing headwaves. The ray is now constrained to travel along the refractor $z = f(x)$. The ray leaves the source with take-off angle θ_s^{opt} and arrives to the refractor at point $\mathbf{x}_u$. After traveling from $\mathbf{x}_u$ to $\mathbf{x}_v$ along the refractor, the ray ends up in the receiver. Note that the angles between the ray trajectory and the normals to the refractor at points $\mathbf{x}_u$ and $\mathbf{x}_v$ are equal to the critical angles as defined by Snell's Law. In this problem the unknowns are two: either θ_s^{opt} and x_v, or x_u and x_v.

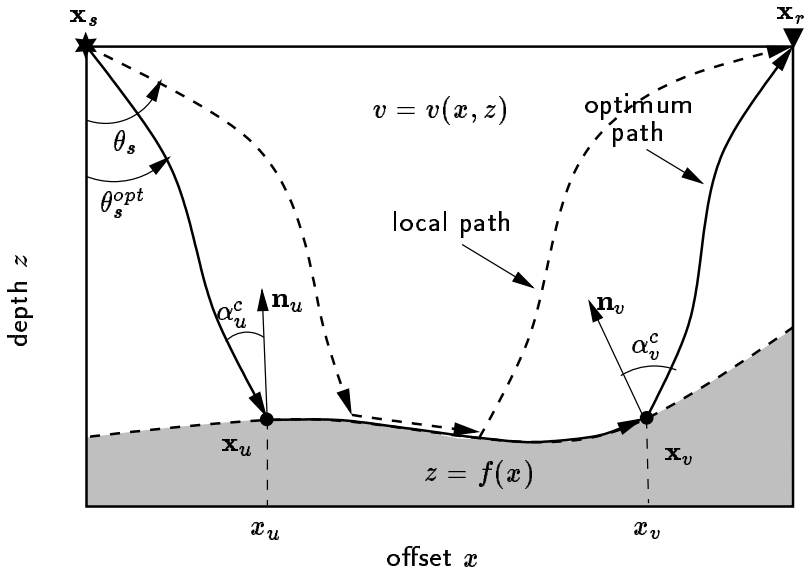

Figure 4.7: Source, receiver, refractor and raypath geometry in a two-dimensional two-point experiment for tracing headwaves.

Diffractions

Diffraction ray tracing is another typical case of BVP. In a two-dimensional model, a diffracting point $\mathbf{x}_u$ is also fixed within the model boundaries, and the minimum raypath connecting $\mathbf{x}_s$ with $\mathbf{x}_u$, and then $\mathbf{x}_u$ with $\mathbf{x}_r$, is to be found (see Figure 4.8). The problem can be split into two standard two-point ray-tracing problems, each one characterized by independent take-off angles.

Different BVPs can also be defined, as is the case of multiples and complicated headwaves. In any case, it is not the purpose of this work to deal with complicated wave phases nor later arrivals that may obscure the description of the two-point ray-tracing method.

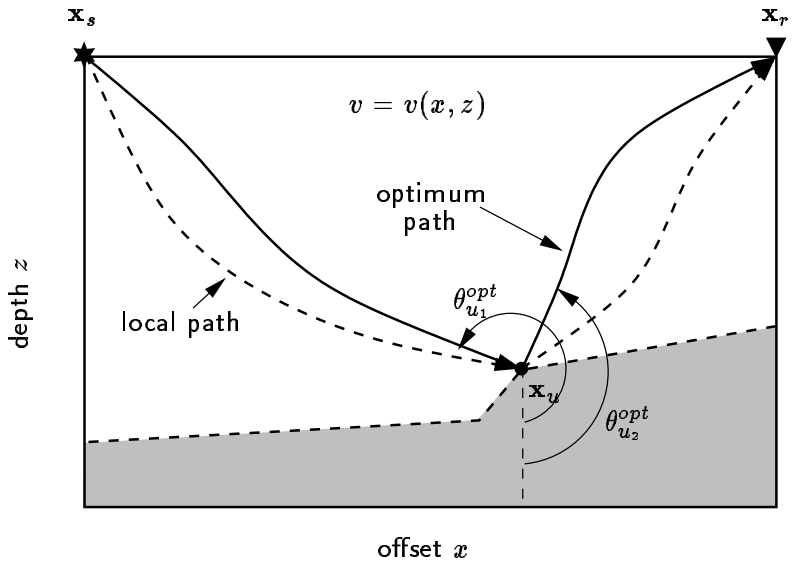

Figure 4.8: Source, receiver, diffracting point and raypath geometry in a two-dimensional two-point experiment for tracing diffractions.

4.4.2 Conventional methods for solving the BVP

A brief description of some conventional methods for solving the BVP arising in ray tracing follows. In [Čer87] a more detailed review can be found. The next section presents a novel approach called SART, a BVP solver that is intended to overcome the usual deficiencies of the conventional methods.

Shooting

The shooting method represents an iterative IVP. First, an initial point (*source*) is fixed and the ray is propagated by specifying the take-off angle. Then, a search strategy is used to update these angles until the ray emerges through the desired endpoint (*receiver*) within a give tolerance.

Since frequently the receiver location is an ill-behaved function of the take-off angle,

the strategy for updating the take-off angle may become a difficult task, and divergence is a common issue unless the model, or the results of the computations, are conveniently smoothed [LLC85]. As a consequence, some raypaths can be missed. The problem is even more severe in the 3-D case, where two take-off angles are to be found. Headwaves are obtained by using a similar trial-and-error search routine [Cas82], which is more difficult to implement and more likely to diverge, because more variables are involved.

To find the optimum take-off angle corresponding to the ray connecting both source and receiver, the distance

$$d = d(\theta_s) = \|\mathbf{x}_e - \mathbf{x}_r\| \tag{4.10}$$

is to be minimized, where $\mathbf{x}_e$ is the emerging point (the point where the ray leaves the model boundaries), and $\mathbf{x}_r$ is the desired endpoint (see Figure 4.10). This is a rather difficult optimization problem. Equation (4.10) is zeroed using some kind of linearizing method that requires an estimate of the partial derivatives of d with respect to θ_s.

In the simplest form, shooting produces a fan of rays with equally spaced take-off angles covering a given range. As a result, it is possible to generate function $d = d(\theta_s)$ empirically, and then use a standard zero-finder to solve the equation $d(\theta_s^{opt}) = 0$. In the method of False position described by Julian and Gubbins [JG77] the partial derivatives of the receiver coordinates with respect to the take-off angle are estimated numerically by tracing three trial rays with slightly varying take-off angles. This method usually has slow convergence caused by the inaccuracies in the estimation of the partial derivatives. In [SK90] a more accurate set of derivatives is obtained by exploiting the fact that the information regarding the position of the ray at any given point and the direction is contained in the curvature of the local wavefront. So, the partial derivatives of the receiver coordinates with respect to the take-off angle can be determined by solving

the equations describing the geometrical spreading of the wavefront along with the ray equations [SK90]. The procedure requires the solution of a system of 15 linear equations per iteration for the 3-D case. To deal with discontinuities, extra equations to account for the boundary conditions must be added. Despite the efforts made to avoid slow convergence and to increase the accuracy, multipathing, discontinuities, and in general the ill-behavior of the objective function represent serious problems for the shooting method [SK90], and divergence can be expected in complex 2-D structures. In the same work, a sort of climb-out procedure to avoid local minima is devised, but I do not believe this is an efficient strategy when dealing with complicated 3-D structures.

Bending

In the bending method both points are linked by an initial guess path, which is then perturbed iteratively so as to satisfy the ray equations or Fermat's principle of stationary time. Several bending techniques are reported in the literature [JG77, UT87, PTE88, MNS92, Yon93] which, unlike shooting, always produce a ray connecting any source-receiver pair. In general, bending involves the solution of a highly nonlinear optimization problem, which requires some kind of gradient directions to update the raypath. In complicated velocity structures, bending tends to overlook multipath propagation because the solution depends on the first guess. As a result, the absolute minimum raypath can be missed. Although a method based on advanced graph theory for choosing an initial guess close to the global minimum exists [Mos89, Mos91, FL93], it is not clear whether the final raypath is a global or a local minimum [MNS92]. Besides, it is worth mentioning that the accuracy obtained with bending methods depends on the selected parameterization (e.g. number of nodes for representing the ray trajectory), and that the phase of the resulting ray may remain unknown throughout the process. Bending methods are usually faster than shooting methods for relatively smooth models, except

for tracing a large set of closely spaced rays where shooting may be preferred. In complex models where shooting has its main drawbacks, bending is advantageous but at a higher computational cost.

Figure 4.9 illustrates a standard bending method. The algorithm starts with an *unrealistic* straight raypath and perturbs it iteratively until the traveltime is minimum (in this example the raypath is parameterized with a set of segments and the algorithm perturbs the node coordinates). Clearly, whenever multipathing exists, the final solution depends on the starting model. Further, if the velocity model contains discontinuities, bending would present some difficulties in dealing with the node perturbation scheme, which is usually based on gradient directions. Filho and Thedy [FT95] use genetic algorithms (GA) to overcome these drawbacks. However, the optimization problem may become too expensive when the number of unknowns (nodes) is large. At the same time, the number of nodes can not always be kept small when a certain accuracy is required or the velocity field is complicated.

Finite differences

Finite-difference (FD) methods are based on solving the eikonal equation[1] by means of finite differences [Vid88, PL91]. The scheme allows one to obtain simultaneously the traveltime field at the nodes of the finite-differences grid. By interpolation, the traveltime corresponding to the ray connecting both source and receiver can be obtained when the endpoint does not coincide with some grid node. The corresponding raypaths are found by following the normals to the wavefronts from source to receiver. FD methods are very suitable for computing traveltimes in relatively smooth velocity models, and the efficiency reached by these schemes is quite high, especially when a large set of rays needs to be

[1] A form of the wave equation for harmonic waves valid only where the variation of properties is small within a wavelength (high-frequency approximation).

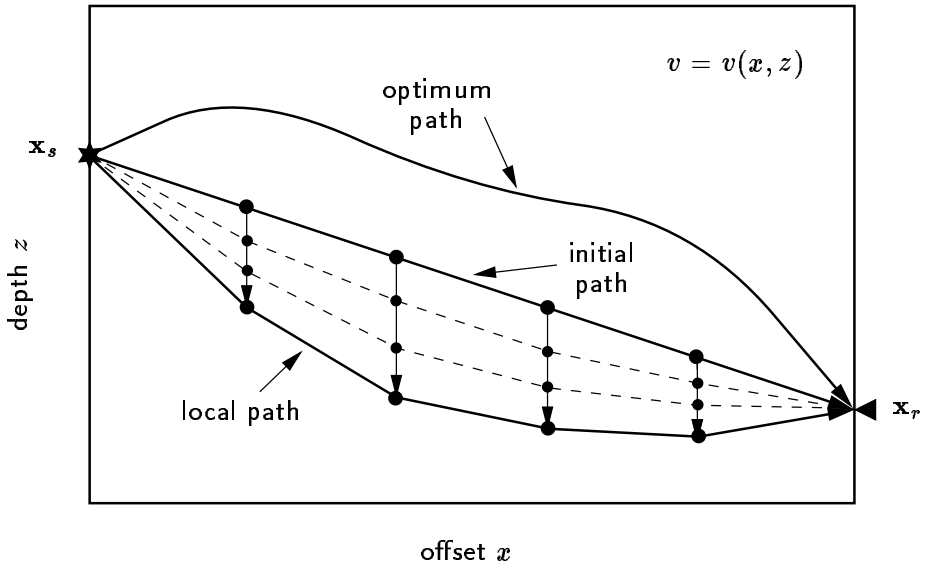

Figure 4.9: In the bending method, an initial guess path is iteratively perturbed until the traveltime is minimum or the ray equations are satisfied. Usually bending gets trapped in local minima and misses the optimum (global minimum) raypath.

traced. However, since their accuracy depends upon the validity of the finite-difference approximation, when the model contains discontinuities or the velocity varies rapidly, the limitations become apparent. The use of a denser grid usually does not help much. The main drawback of the FD method is, however, the fact that it only produces first arrivals, regardless the phase of the obtained raypath. That is, given a gridded velocity field and a source-receiver geometry, in general it is not possible to know which type of wave phase the traveltime obtained by the FD method corresponds to. It may correspond to a diffraction, or a headwave, or a direct wave, etc., whichever arrives first. This issue is very important from the point of view of phase identification and the energy of the arrivals [GB93].

Linear traveltime interpolation

Linear traveltime interpolation (LTI) was first introduced in [IRS88] and later modified in [AK90]. As in the FD method, a gridded 2-D cell model is used. Raypaths are assumed straight within each cell, and traveltimes are computed on the cell boundaries based on the linear interpolation of traveltimes. Cell boundaries are subdivided into several segments and the minimum traveltime is calculated at every resulting grid point. After computing traveltimes, a backward process is utilized to trace the raypaths based on the reciprocity principle. The LTI method is reported to be more accurate and more suited for gridded 2-D cell models than the FD method [AK93]. LTI can even handle abrupt changes of velocity without the need to smooth the velocity field or to increase the number of cells. Like FD, LTI produces only first arrivals.

Huygens' principle method

Methods based on the Huygens' principle have been developed in order to avoid the difficulties associated with the branching points of a ray (see for example [Sai89]). Methods based on graph theory may also be included in this category [Mos89, Mos91, FL93]. Like FD and LTI, these methods are applicable to gridded cell models. According to Huygens' principle each grid point becomes a secondary source, from which a new ray propagates to all directions. In practice, the number of directions is fixed and therefore the number of paths connecting two points becomes finite. Traveltimes corresponding to each possible path are compared and the one with minimum traveltime is selected. The accuracy is limited to the model discretization, and the computational requirements increase dramatically with the number of cells, nodes and directions, in particular for 3-D models. Recently, graph-theory methods have been applied to 3-D media [CH96], but a thoughtful study of its applicability for complex 3-D structures remains to be done.

4.4.3 Simulated annealing ray tracing (SART)

As a counterpart, I present here a novel method for solving the BVP that combines SA with ray-tracing techniques [VU96a, Vel96]. SART has both shooting and bending features. On the one hand, a standard IVP is solved at each iteration. On the other hand, the raypath is arbitrarily modified so as to satisfy the boundary conditions. At convergence, the solution to the IVP yields the optimum ray trajectory connecting source and receiver.

It is the goal of SART to find the raypath that exhibits the absolute minimum traveltime. It is not the objective of the method to find all possible rays connecting the two fixed points, but only the one that exhibits the absolute minimum traveltime for a given wave phase. SART focuses on *direct* first arrivals, for it is based on solving the IVP. In many ray-tracing applications these are of greatest importance because they usually represent the most energetic events [GB93]. Arbitrary headwaves, diffractions and waves propagating through shadow zones cannot, in general, be found by SART. In these cases, bending, finite-difference, or graph-theory methods should be used. However, SART can be easily modified so as to handle these kind of waves if the ray signature is defined beforehand, as shown in Figures 4.6, 4.7, and 4.8. It is important to note that the method is not suitable for obtaining raypaths corresponding to local minima (such as caustics and triplications) nor minimum paths that exhibit exactly the same traveltime in some pathological cases, which are beyond the scope of this work.

SART, unlike bending, avoids local minima and converges to the absolute minimum traveltime, provided the proper cooling schedule is utilized. As mentioned before, the number of unknowns in the bending method is usually large if a certain accuracy is required. On the contrary, the number of unknowns in SART reduces to a few and it is not affected by local minima nor discontinuities. Any ill-behavior of the objective

function does not represent a serious problem, in general, for VFSA, which is a natural tool for dealing with highly nonlinear optimization problems.

4.5 Simulated annealing ray tracing (SART)

SART essentially proceeds as follows (see Figure 4.10). Given a fixed source point, $\mathbf{x}_s$, and an initial take-off angle, θ_s, the ray is propagated until it leaves the model boundaries or arrives at the boundary of a prescribed near-receiver region (e.g. the receiver cell), determining the emerging point $\mathbf{x}_e$. This point is then connected with a straight line directly to the receiver point $\mathbf{x}_r$. The total traveltime, which is computed as the path integral of the slowness between $\mathbf{x}_s$ and $\mathbf{x}_r$, is minimized with respect to θ_s using VFSA. Notice that the ray satisfies Snell's Law at all crossed cell boundaries, except for the segment connecting $\mathbf{x}_e$ to $\mathbf{x}_r$. Nevertheless, when the minimum is obtained, Snell's Law is also satisfied for the last portion of the raypath, too.

4.5.1 Description of the algorithm

Basically, at each iteration an IVP is solved starting from the source, $\mathbf{x}_s$, with take-off angle θ_s. The propagation is terminated provided either

1. the ray arrives at the model boundary; or

2. the ray arrives at a prescribed near-receiver region (e.g. receiver cell).

The point where the ray meets one of the above two conditions is called $\mathbf{x}_e$. The raypath is completed by connecting $\mathbf{x}_e$ with the receiver, $\mathbf{x}_r$, with a straight line. Figure 4.10 illustrates this situation when condition (1) is met for a 2-D model. Note that the segment connecting $\mathbf{x}_e$ with $\mathbf{x}_r$ is quite arbitrary, and the resulting raypath may be completely unrealistic. The purpose of the straight-ray construction is to force the raypath to satisfy

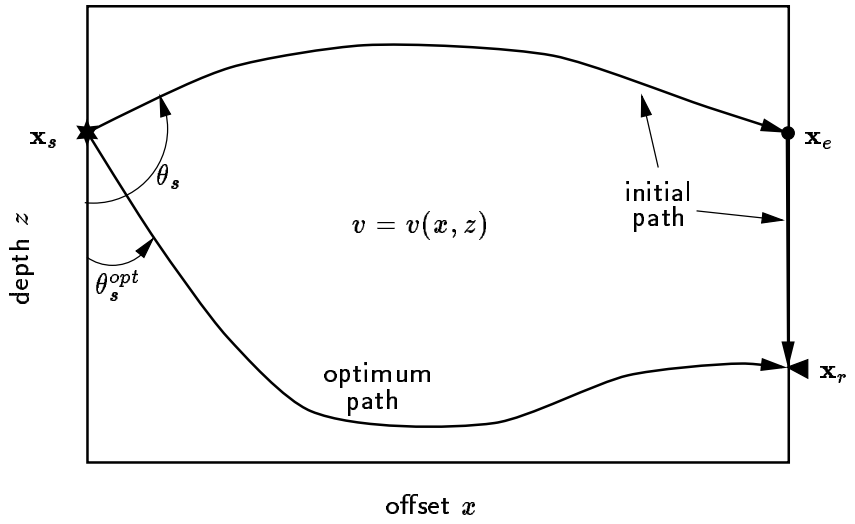

Figure 4.10: Straight-ray construction used by SART. When traveltime is minimum, $\mathbf{x}_e$ coincides with $\mathbf{x}_r$ and the minimum path with optimum take-off angle θ_s^{opt} has been found.

the constraint imposed by the BVP. This is an intermediate raypath that, like in bending, is updated iteratively.

The total traveltime is computed by integrating the slowness field $s(x, z) = 1/v(x, z)$ along the path l between both endpoints:

$$T = \int_{\mathbf{x}_s}^{\mathbf{x}_r} s \, dl = T_{se} + T_{er}, \tag{4.11}$$

where

$$T_{se} = \int_{\mathbf{x}_s}^{\mathbf{x}_e} s \, dl \tag{4.12}$$

is the traveltime associated with the first portion of raypath that starts at $\mathbf{x}_s$, and

$$T_{er} = \int_{\mathbf{x}_e}^{\mathbf{x}_r} s \, dl \tag{4.13}$$

is the traveltime associated with the straight-ray construction starting at $\mathbf{x}_e$. The integration here is done along the straight line. In the cell ray-tracing scheme, the integrals are approximated by discrete sums over all ray segments that make the raypath.

Since source and receiver are fixed and $\mathbf{x}_e$ is uniquely determined by the solution of the IVP (I assume the IVP can be solved for any given take-off angle θ_s), T becomes a function of the take-off angle only, so

$$T = T(\theta_s). \tag{4.14}$$

The final raypath is found by recalling Fermat's principle. When traveltime T is minimum, $\mathbf{x}_e$ coincides with $\mathbf{x}_r$ and the whole raypath satisfies the ray equations. As I will show later, this is not always strictly true, and some ray for which $\mathbf{x}_e \not\equiv \mathbf{x}_r$ may arrive earlier than any other *realistic* path. However, I will reformulate the problem so as to overcome this eventual difficulty. The straight-ray construction could be replaced by any other arbitrary ray construction inasmuch as traveltime T_{er} tends to zero as $\mathbf{x}_e$ tends to $\mathbf{x}_r$. The straight-ray construction has been chosen for expediency only, because it adds little extra effort to computing T_{er}.

The basic problem now becomes an optimization one in which one parameter (take-off angle) is to be found so that T is a global minimum. Since expression (4.14) is a highly nonlinear, multimodal, and nondifferentiable function, it cannot be properly minimized using classical linearizing methods. I use instead VFSA, which is a very powerful tool for minimizing cost functions independently of its nonlinearities, discontinuities, and stochasticity.

4.5.2 SART for more complex BVPs

So far I have described the strategy for solving the two-point boundary-value ray-tracing problem that takes into account direct waves only. It is capable of finding the raypath connecting the two endpoints $\mathbf{x}_s$ and $\mathbf{x}_r$ with the absolute minimum traveltime, independently of any possible reflection along its trajectory. However, many times one is interested in finding the raypath corresponding to complex raypaths, such as reflected waves or headwaves. This can be accomplished by incorporating minor modifications to the already described two-point ray-tracing scheme.

Handling reflected waves

In the case of reflected waves, the ray must undergo a reflection at a predefined boundary $z = f(x)$. The reflection occurs at point $\mathbf{x}_u = [x_u, f(x_u)]$, which is unknown (see Figure 4.11a). Here the ray is forced to arrive to the prescribed boundary where a reflection is desired. This is done by penalizing those rays that do not reach the prescribed boundary. Once the ray reaches the boundary, it undergoes a reflection and propagates towards the emerging point $\mathbf{x}_e$. Then $\mathbf{x}_e$ is connected with $\mathbf{x}_r$ using a straight line as described above. Since the ray trajectory is comprised of three portions, I define the following cost function to be globally minimized:

$$\Phi_1 = \Phi_1(\theta_s) = \begin{cases} T_{su} + T_{ue} + T_{er} = \int_{\mathbf{x}_s}^{\mathbf{x}_u} s\, dl + \int_{\mathbf{x}_u}^{\mathbf{x}_e} s\, dl + \int_{\mathbf{x}_e}^{\mathbf{x}_r} s\, dl, & z_k = f(x_k) \text{ for some } k \\ T_{max}, & \text{otherwise} \end{cases} \tag{4.15}$$

where

$$T_{max} \geq \int_{\mathbf{x}_s}^{\mathbf{x}_r} s\, dl, \quad \forall \theta_s, \tag{4.16}$$

can be guessed easily from a set of trial raypaths or set simply as $T_{max} = l_{max}/v_{min}$, where l_{max} is the maximum expected path length, and v_{min} is the minimum velocity of the model. In equation (4.15) T_{ue} is the traveltime associated with the second portion of the raypath starting at point $\mathbf{x}_u$. Cost function $\Phi_1(\theta_s)$ as defined above ensures that the resulting traveltime for those rays that do not reach the reflector is always greater than that corresponding to any reflected wave. As a consequence, the global minimum of equation (4.15) corresponds to a reflection, and $\mathbf{x}_e \equiv \mathbf{x}_r$. Notice that equation (4.15) introduces first-order discontinuities at those angles where the ray stops reaching the discontinuity. This is not an issue for VFSA to be worried about, but may lead to extra iterations during the optimization process, because those rays that do not reach the discontinuity are generated unnecessarily.

An alternative scheme that avoids using a penalty cost function and that ensures that all generated rays correspond to a reflection, is to start the ray propagation from point $\mathbf{x}_u$ towards the emerging points $\mathbf{x}_{e_1}$ and $\mathbf{x}_{e_2}$ as shown in Figure 4.11b. In this scheme the number of unknowns has increased by one. Not only coordinate x_u must be obtained, but also take-off angle θ_{u_1}. The take-off angle corresponding to the ray traveling from $\mathbf{x}_u$ to $\mathbf{x}_{e_2}$ is related to θ_{u_1} by

$$\theta_{u_2} = 2\pi - \theta_{u_1} - 2\arctan\left[f'(x_u)\right], \tag{4.17}$$

easily derived from the figure. Then, it is convenient to define the cost function

$$\Phi_2 = \Phi_2(\theta_{u_1}; x_u) = T_{ue_1} + T_{e_1 s} + T_{ue_2} + T_{e_2 r} = \int_{\mathbf{x}_u}^{\mathbf{x}_{e_1}} s\, dl + \int_{\mathbf{x}_{e_1}}^{\mathbf{x}_s} s\, dl + \int_{\mathbf{x}_u}^{\mathbf{x}_{e_2}} s\, dl + \int_{\mathbf{x}_{e_2}}^{\mathbf{x}_r} s\, dl, \tag{4.18}$$

that is to be globally minimized using VFSA.

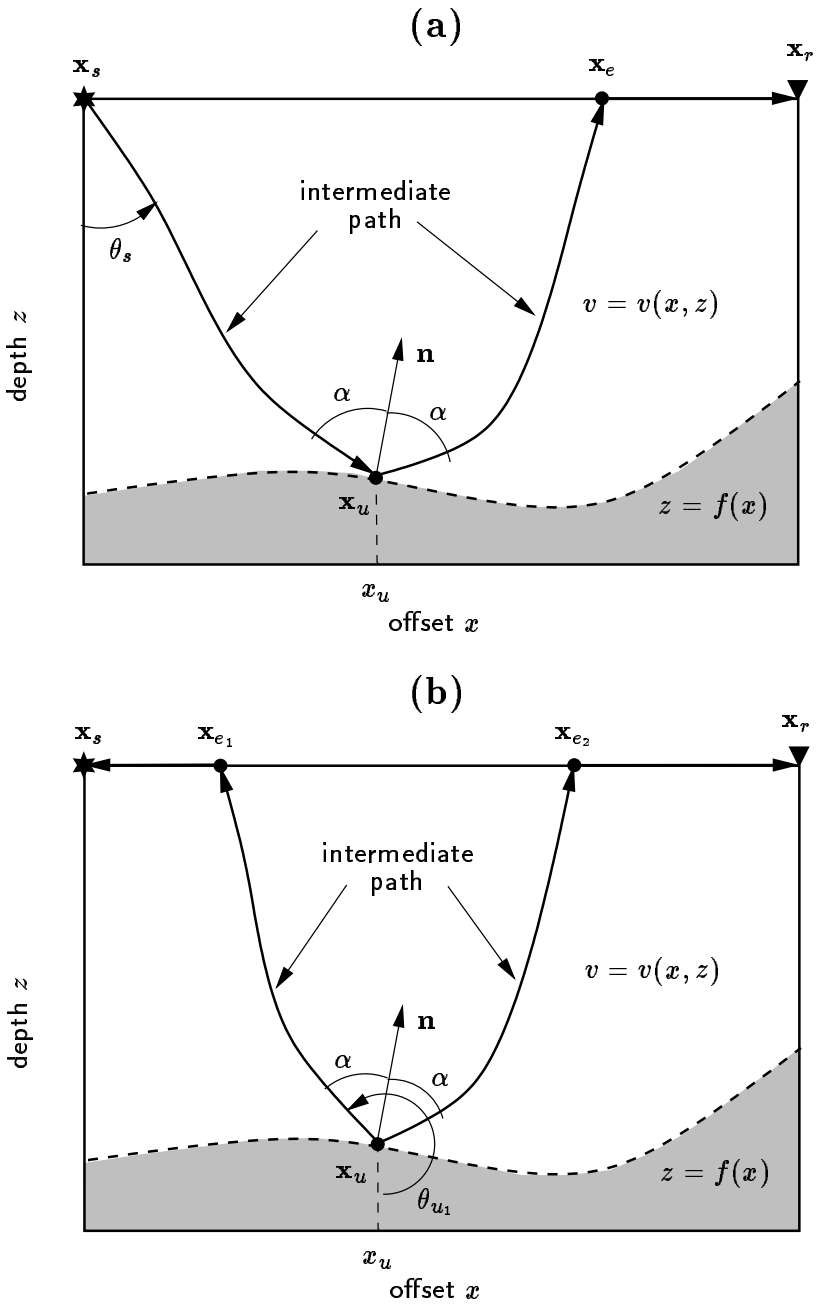

Figure 4.11: Alternative strategies used by SART for tracing reflections connecting source and receiver (see text for details). (a) Here SART searches for the optimum take-off angle θ_s until $\mathbf{x}_e \equiv \mathbf{x}_r$ and traveltime is minimum. (b) Here the search involves x_u and θ_{u_1}. At convergence, $\mathbf{x}_{e_1} \equiv \mathbf{x}_s$ and $\mathbf{x}_{e_2} \equiv \mathbf{x}_r$.

Yet another alternative scheme could be devised that depends on a single unknown, as in the first case, and that always generates a reflection at the discontinuity, as in the second one. This scheme is illustrated in Figure 4.12a. Here the ray leaves the source with take-off angle θ_s and arrives to either point $\mathbf{x}_u$ on the discontinuity, or to point $\mathbf{x}_e$ on some model boundary. If $\mathbf{x}_u$ is met first, the ray undergoes a reflection and travels towards $\mathbf{x}_e$ as in Figure 4.11a. But if $\mathbf{x}_e$ is met first, a vertical straight ray segment is used to join it with $\mathbf{x}_{\hat{u}} = [x_e, f(x_e)]$ and back again $\mathbf{x}_e$. From $\mathbf{x}_e$ to $\mathbf{x}_r$ the straight-ray construction is used as usual. Then I write

$$\Phi_3 = \Phi_3(\theta_s) = \begin{cases} T_{su} + T_{ue} + T_{er}, & z_k = f(x_k) \text{ for some } k \\ T_{se} + 2T_{e\hat{u}} + T_{er}, & \text{otherwise} \end{cases} \tag{4.19}$$

where

$$T_{e\hat{u}} = \int_{z_e}^{f(x_e)} s(x_e, z)\, dz. \tag{4.20}$$

In essence, equation (4.19) is equivalent to equation (4.15), where T_{max} have been replaced by $T_{se} + 2T_{e\hat{u}} + T_{er}$. It is clear that this construction will generate rather arbitrary ray trajectories with no physical sense except at convergence.

Which of the three alternative schemes would be the most effective and efficient SART scheme for obtaining the optimum reflection connecting source and receiver, cannot be said a priori. Clearly the first one involves the fewest operations per iteration, but no final conclusion can be drawn regarding which one converges faster. At first sight the second approach would require more iterations since the model space to be explored by VFSA is two-dimensional. The underlying velocity model and source-receiver geometry play an important role, too. It can be said, however, that once the optimum take-off angle has been found

$$\Phi_1(\theta_s^{opt}) = \Phi_2(\theta_u^{opt}; x_u^{opt}) = \Phi_3(\theta_s^{opt}). \tag{4.21}$$

Normal rays In the normal rays case $\mathbf{x}_s \equiv \mathbf{x}_r$, and the principle of reciprocity is applied. The ray leaves the reflector at point $\mathbf{x}_u$ with take-off angle θ_u and arrives to the emerging point $\mathbf{x}_e$ (see Figure 4.12b). Note that the tangent to the raypath is parallel to $\mathbf{n}$ at point $\mathbf{x}_u$, the normal to the reflector. The only unknown is x_u, since θ_u can be computed with

$$\theta_u = \pi - \arctan\left[f'(x_u)\right]. \tag{4.22}$$

Thus, for this particular case, the cost function to be globally minimized is written as

$$\Phi = \Phi(x_u) = 2(T_{ue} + T_{es}), \tag{4.23}$$

where

$$T_{ue} = \int_{\mathbf{x}_u}^{\mathbf{x}_e} s\, dl, \quad T_{es} = \int_{\mathbf{x}_e}^{\mathbf{x}_s} s\, dl. \tag{4.24}$$

In SART, equation (4.23) is globally minimized using VFSA.

Handling headwaves

To trace headwaves along a prescribed boundary, $z = f(x)$, the ray trajectory is broken into various parts. As shown in Figure 4.7 and described in section 4.4.1, the final raypath has three parts, but the intermediate raypaths which are obtained at each iteration are normally composed of five, as shown is Figure 4.13 and described below:

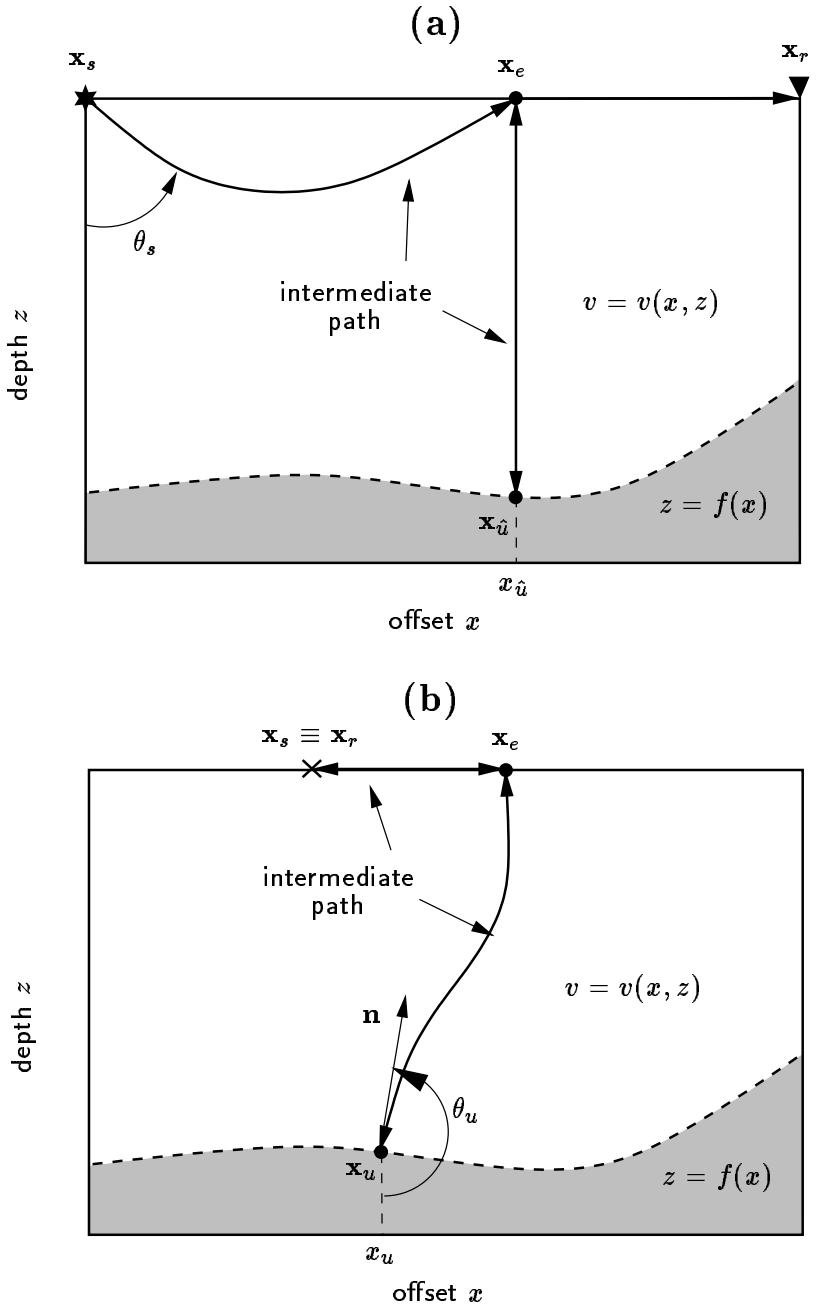

Figure 4.12: (a) SART for tracing reflected waves. Yet another alternative strategy. When the ray leaves the model at point $\mathbf{x}_e$, a double straight vertical line is used to force an (unrealistic) reflection at point $\mathbf{x}_{\hat{u}} = [x_e, f(x_e)]$. (b) Normal rays case. In the optimization problem, coordinate x_u represents the only unknown.

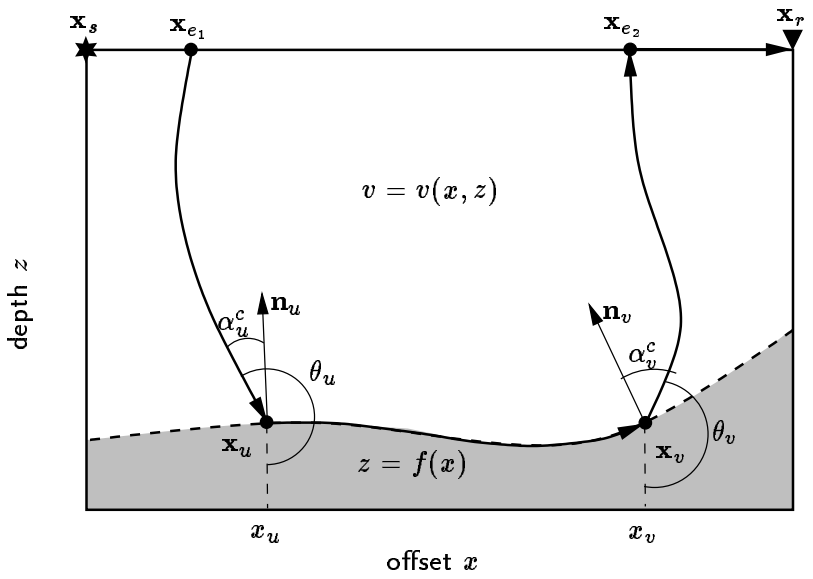

Figure 4.13: SART for tracing headwaves. Both coordinates x_u and x_v are globally optimized until the total traveltime is minimum.

1. the ray that leaves a point $\mathbf{x}_u = [x_u, f(x_u)]$ with take-off angle θ_u, and arrives to the emerging point $\mathbf{x}_{e_1}$,

2. the ray that leaves a point $\mathbf{x}_v = [x_v, f(x_v)]$, $x_v \geq x_u$, with take-off angle θ_v, and arrives to the emerging point $\mathbf{x}_{e_2}$,

3. the ray that propagates along the boundary from $\mathbf{x}_u$ to $\mathbf{x}_v$,

4. the straight-ray segment that connects $\mathbf{x}_{e_1}$ with $\mathbf{x}_s$, and

5. the straight-ray segment that connects $\mathbf{x}_{e_2}$ with $\mathbf{x}_r$.

The problem now consists on finding the unknown points, $\mathbf{x}_u$ and $\mathbf{x}_v$, so that the first part of the ray arrives to $\mathbf{x}_s$, the third one arrives to $\mathbf{x}_r$, and the total traveltime is minimum. That is, the problem has been split into two IVPs (now the take-off angles are known and the initial point is not), plus the ray propagation along the refractor. Each of

both IVPs, are not, however, independent. The take-off angles to trace ray parts (1) and (2) above are closely related to the respective critical angles and normals to the refractor at points $\mathbf{x}_u$ and $\mathbf{x}_v$. That is

$$\begin{cases} \theta_u = \pi + \arcsin(v_u/v_r) - \arctan f'(x_u) \\ \theta_v = \pi - \arcsin(v_v/v_r) - \arctan f'(x_v) \end{cases} \tag{4.25}$$

where v_u and v_v are the velocities right above the refractor at points $\mathbf{x}_u$ and $\mathbf{x}_v$ respectively, and v_r is the refractor velocity ($v_r > v_u$, $v_r > v_v$).

For this problem the total traveltime to be globally minimized can be written as

$$\Phi_1 = \Phi_1(x_u, x_v) = T_{ue_1} + T_{ve_2} + T_{uv} + T_{e_1 s} + T_{e_2 r}, \tag{4.26}$$

where

$$T_{uv} = \int_{x_u}^{x_v} s[x, f(x)]\, dx, \tag{4.27}$$

is the traveltime along the refractor from $\mathbf{x}_u$ to $\mathbf{x}_v$. I minimize equation (4.26) with respect to the two unknowns x_u and x_v using VFSA. Again it must be stressed that equation (4.26) is highly nonlinear and multimodal, and that although a trial-and-error procedure for finding x_u and x_v is feasible, the same kind of difficulties that arise in shooting to choose the take-off angle is expected. In a 3-D velocity model, if the prescribed boundary is defined by the surface $z = f(x, y)$, the total traveltime is a function of four variables, say (x_u, y_u) and (x_v, y_v). A trial-and-error procedure to find the four unknowns may be both inefficient and ineffective for obvious reasons.

Alternatively, the total traveltime can be defined in terms of θ_s and x_v (or θ_r and x_u):

$$\Phi_2 = \Phi_2(\theta_s, x_v) = T_{su} + T_{uv} + T_{ve} + T_{er}, \tag{4.28}$$

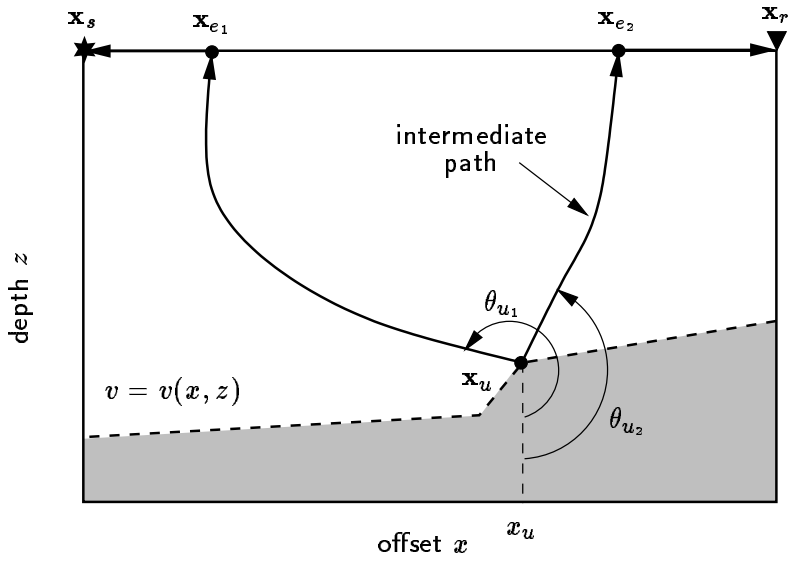

Figure 4.14: SART for tracing diffractions. See text for explanation.

Now the ray trajectory has been split into four portions, since $\mathbf{x}_u$ is uniquely determined by θ_s. In this formulation some penalization, as in the case of tracing reflections, must be introduced to force the ray arriving to the refractor. In practice I have found that the previous formulation, equation (4.26), is more easily optimized than equation (4.28).

Handling diffracted waves

SART for diffractions of the type shown in Figure 4.8 (point diffraction) follows a similar strategy as in the cases of reflections and headwaves. The problem is split into two BVPs that can be solved separately or simultaneously. Again the principle of reciprocity is applied to trace the rays from the diffraction point $\mathbf{x}_u$ towards the emerging points $\mathbf{x}_{e_1}$ and $\mathbf{x}_{e_2}$, as shown in Figure 4.14. Then a straight line is used to connect these points with source and receiver respectively. Thus the ray trajectory is composed of four portions. Now I define the cost function

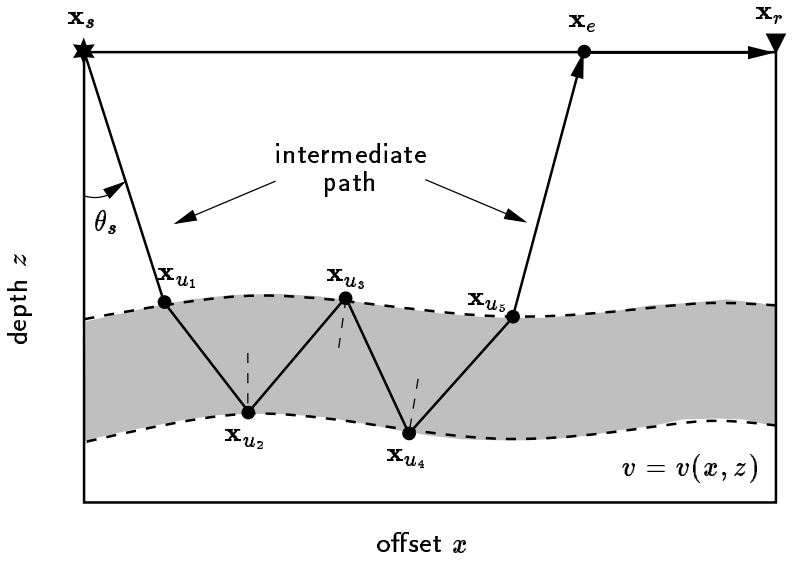

Figure 4.15: SART for tracing multiples. See text for details.

$$\Phi = \Phi(\theta_{u_1}, \theta_{u_2}) = T_{ue_1} + T_{e_1 s} + T_{ue_2} + T_{e_2 r}, \tag{4.29}$$

that is to be globally minimized with respect to the take-off angles θ_{u_1} and θ_{u_2}.

4.5.3 Shadow zones, multiples, and more

Despite the fact that SART can be modified to take care of other kind of waves, it is not suitable, in general, for obtaining rays propagating through shadow zones, for it is based on solving the IVP. The best it can do in these cases is to detect a candidate shadow zone whenever $\mathbf{x}_e$ fails to converge to $\mathbf{x}_r$ within a given tolerance and after a maximum number of iterations. But as already described, simple diffractions and headwaves that may propagate through shadow zones are easily incorporated provided the ray signature is specified a priori.

It is also possible to trace more complex raypaths such as multiples and complex

headwaves. In all the cases the ray signature must be specified a priori. Take Figure 4.15 as an example. The multiple is generated in the low velocity layer, which is delimited by two predefined boundaries, $z = f_1(x)$ and $z = f_2(x)$. The ray signature for this particular case can be summarized by setting $IOP_1(I) = \{3, 5, 3, 0\}$ and $IOP_2(I) = \{5, 5, 0\}$. If the propagation is started from $\mathbf{x}_s$ it is necessary to devise some strategy so that the resulting ray undergoes two reflections on $z = f_2(x)$, and one reflection and two refractions on $z = f_1(x)$. As in the case of a single reflection, various alternative schemes are possible. The simplest one would be to define $\Phi = \Phi(\theta_s)$ as in equation (4.15), taking into account the various portions of the raypath and a penalty term:

$$\Phi = \Phi(\theta_s) = \begin{cases} T_{su_1} + T_{u_1u_2} + T_{u_2u_3} + T_{u_3u_4} + T_{u_4u_5} + T_{u_5r}, & \text{signature is satisfied} \\ T_{max} & \text{otherwise} \end{cases} \tag{4.30}$$

but an approach as in equation (4.19) is also possible.

At this point, the reader has probably noticed that the philosophy underlying the SART method is to put the BVP into a convenient optimization framework which is in turn solved by means of VFSA.

4.6 Numerical examples and discussion

In all the examples I am going to show, the 2-D velocity field is defined over a grid with rectangular cells. The velocity is piecewise constant within each cell. Distances are given in *meters*, traveltimes in *milliseconds*, velocities in *kilometers per second*, and angles in *degrees*. All take-off angles are measured counterclockwise from the positive z-axis that points downwards. The coordinates of the first grid node, i.e. node $(1, 1)$, are $(x_{min}, z_{min}) = (0, 0)$, and the size of each cell is 1×1. For the sake of simplicity, source

and receiver are always located on some model boundary. Velocity fields are plotted in a grey scale where dark grays correspond to higher velocities.

4.6.1 Model 1: smooth model

Take Figure 4.16 as an example to illustrate the two-point ray-tracing algorithm. The velocity field varies smoothly between 1.3 and 2.5 km/s over a grid of 100×100 square cells. Note that various low and high velocity anomalies have been introduced. I put a source at $(0, 0)$ and produced a fan of rays with equally spaced take-off angles in the range $[20^o, 80^o]$ (see left panel of the Figure). Despite the fact that the velocity model is smooth, it is evident from the graph that the raypath distribution is rather complex, particularly in the lower-right quarter of the model where raypaths start to cross.

The behavior of the raypaths make it difficult to obtain the optimum arrival. This requires to globally minimize cost function $\Phi(\theta_s)$, which is not a trivial task for conventional ray-shooting and bending methods. The main difficulty arises from the fact that there are several rays that connect source and receiver (multipathing). In effect, the five rays shown in the right panel of Figure 4.16 arrive at the receiver following different paths. Their traveltimes correspond to local minima of function $\Phi(\theta_s)$. Table 4.2 summarizes their take-off angles and traveltimes. Notice that some of the rays exhibit quite similar traveltimes (rays 1 and 2; rays 4 and 5), but their trajectories are rather different. SART found the absolute minimum traveltime among all possible raypaths after a maximum of 200 iterations:

$$\theta_s = 70.4^o, \qquad \Phi = T_{opt} = 71.85 \text{ ms.}$$

This example shows the importance of using a global optimization algorithm for finding the optimum take-off angle. Clearly, a bending algorithm would converge to the path which is closest to the initial guess. Similarly, it is not possible to anticipate which

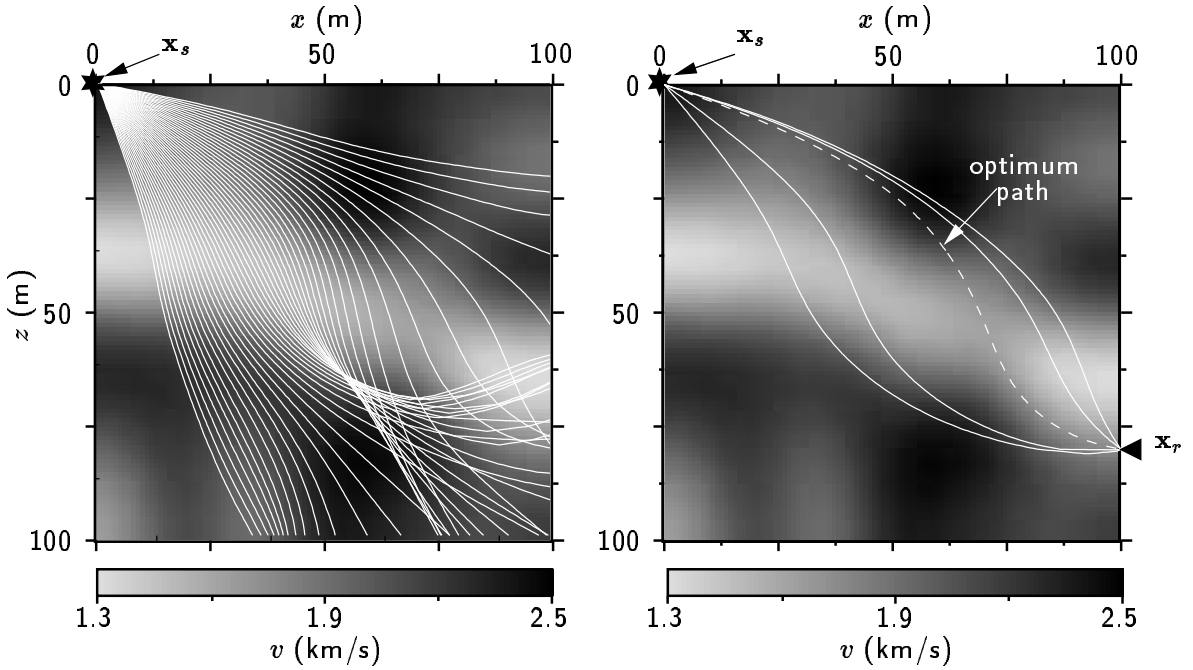

Figure 4.16: Model 1: tracing direct waves. Left panel: two-dimensional smooth velocity model and raypath distribution. The figure shows a fan of rays with equally spaced take-off angles starting at the source point $\mathbf{x}_s = (0, 0)$. Right panel: multipathing for the given source-receiver geometry. The raypath plotted with dotted line exhibits the minimum traveltime. All raypaths honor Snell's Law at cell boundaries.

solution the shooting method will yield but obtaining the five raypaths.

4.6.2 Model 2: reflector model

This example illustrates SART for tracing reflected waves. I constructed the velocity model depicted in Figure 4.17. The velocity has a constant gradient increasing with depth as $v(x, z) = 2 + 0.02z$, and an irregular reflector is located at a depth of about 45 m, with equation

$$f(x) = 48 - (x - 40)^2/160 + 1.75 \sin(x/4). \tag{4.31}$$

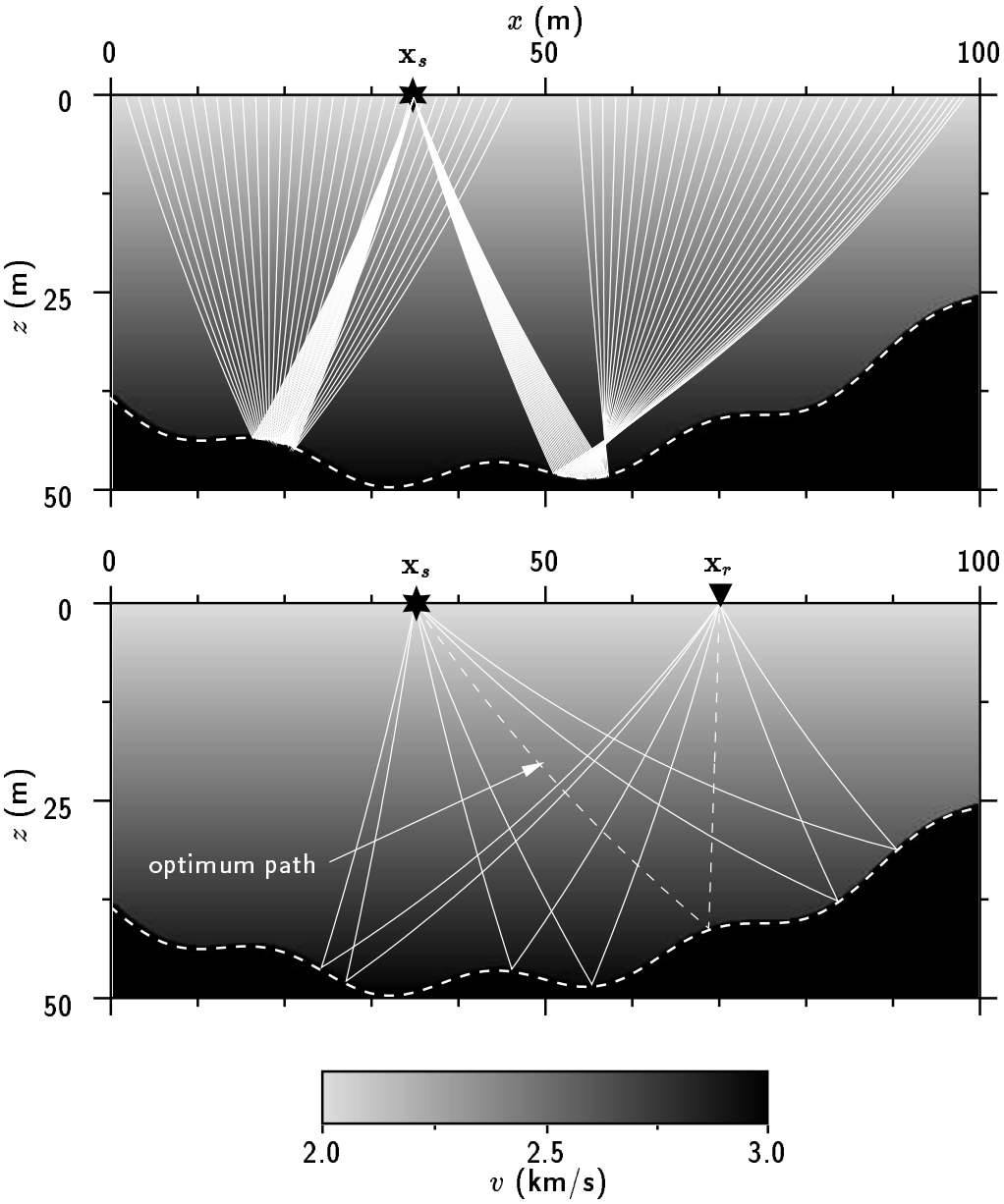

Figure 4.17: Model 2: tracing reflected waves. Top panel: two-dimensional velocity model and raypath distribution. The dotted line represents the reflector. The figure shows two fans of rays with equally spaced take-off angles in the ranges $[-19^o, -14^o]$ and $[15^o, 20^o]$ respectively. Bottom panel: multipathing for a given source-receiver geometry. The raypath plotted with dotted line exhibits the minimum traveltime.

	Model 1		Model 2		Model 3		
Ray #	θ_s (deg)	T (ms)	θ_s (deg)	T (ms)	x_u (m)	x_v (m)	T (ms)
1	39.84	74.6587	-10.71	46.1864	33.53	67.94	44.4353
2	54.01	74.9838	-7.63	46.0965	**33.53**	**84.89**	**42.7057**
3	**70.45**	**71.8520**	10.98	41.0759	51.05	67.94	45.3943
4	72.34	72.3441	18.62	42.0453	51.05	84.89	43.7022
5	73.10	72.1679	**31.58**	**39.5869**	-	-	-
6	-	-	40.89	43.0836	-	-	-
7	-	-	47.45	43.7701	-	-	-

Table 4.2: Multipathing in the three models. SART solutions are shown in boldface.

Below $z = f(x)$, $v(x,z) = 3$ km/s. To visualize the multipath nature of the model, I located a source at $(35,0)$ and produced a fan of rays with equally spaced take-off angles in the range $[-40^o, 60^o]$. The resulting receiver-distance $\tilde{d}$ and traveltime curves are shown in Figure 4.18. Here $\tilde{d}$ is the distance between the raypath endpoint and the origin, measured along the model boundaries starting at (0,0). For a receiver located at $\mathbf{x}_r = (70,0)$, $\tilde{d}(\mathbf{x}_r) = x_r - x_{min} = 70$ m. For clarity, I have plotted only two subsets of raypaths in Figure 4.17, for take-off angles in the ranges $[-19^o, -14^o]$ and $[15^o, 20^o]$. Their corresponding traveltimes and receiver distances map in Figure 4.18 (thick solid lines). By inspecting the receiver distance and traveltime curves, it can be appreciated that any receiver located on the surface ($\tilde{d} \leq 100$) is reached by a number of rays (note the multivalued nature of the traveltime curve shown in Figure 4.18a). In particular, a receiver located at $(70,0)$ receives the arrival of seven quite different rays, which are plotted in Figure 4.17 (bottom panel), and marked with filled circles in Figure 4.18. Table 4.2 summarizes their take-off angles and traveltimes. The raypath obtained by SART corresponds to

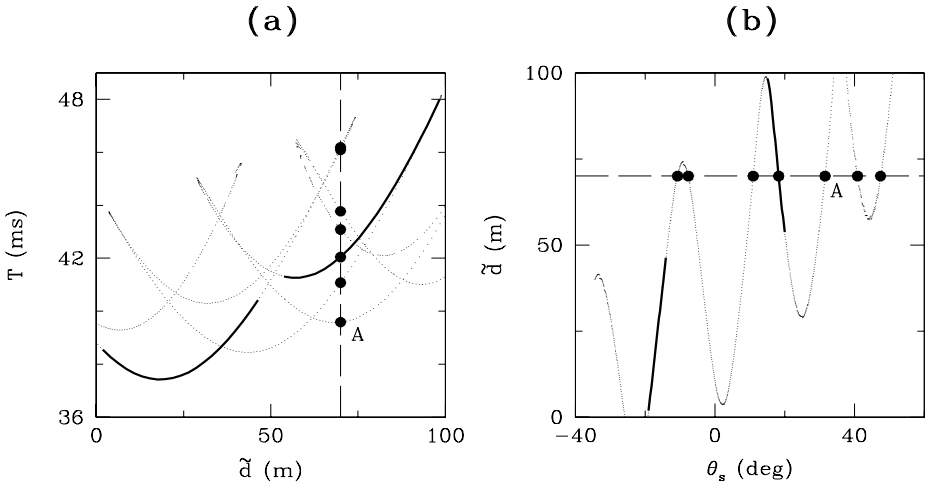

Figure 4.18: Model 2. (a) Traveltime vs receiver distance. (b) Receiver distance vs take-off angle (see text for explanation).

$$\theta_s = 31.6^o, \qquad \Phi = T_{opt} = 39.59 \text{ ms}$$

(point A in Figure 4.18 and dotted line in Figure 4.17). This BVP solution is the one which globally minimizes cost function (4.15), the others being local minima (later arrivals) of the same function. SART globally minimized the cost function successfully after a maximum number of iterations equal to 300. Figure 4.19 shows cost function as a function of θ_s, where the global minimum (point A in the left panel) is clearly seen, and the convergence curve after 300 iterations. For such a multimodal cost function, it is necessary to use a nonlinear optimization algorithm to avoid getting trapped in local minima.

4.6.3 Model 3: refractor model

The model shown in Figure 4.20 is intended to illustrate SART for tracing headwaves. The velocity increases with depth as $v(x, z) = 2+0.01z$. Below $f(x) = 45-x/5$ (refractor)

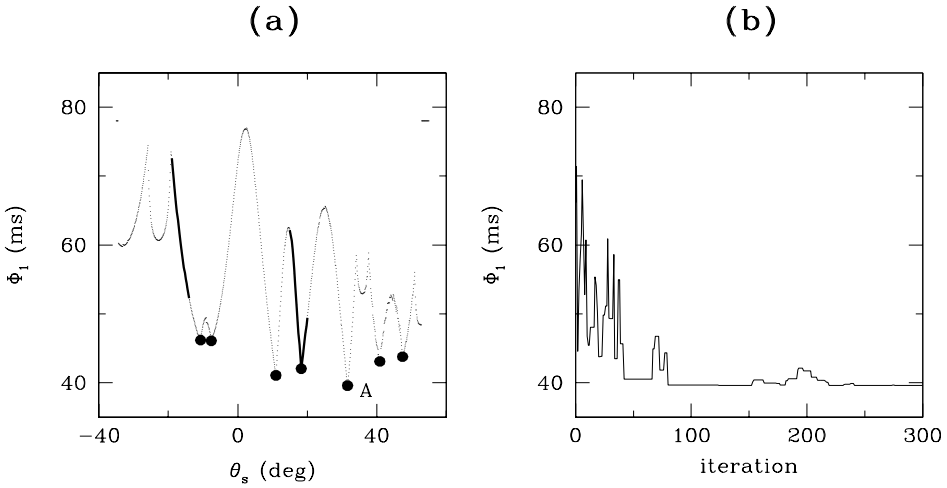

Figure 4.19: Model 2. (a) Cost function vs take-off angle. This function exhibits several local minima. (b) SART convergence after 300 iterations.

the velocity is 6 km/s. A rectangular high velocity anomaly ($v = 3$ km/s) that generates a strong discontinuity has been included as shown in the figure. By locating the source at $(10, 0)$ and the receiver at $(90, 0)$, I have found a total of four possible trajectories that connect both endpoints. These four raypaths, which are also shown in Figure 4.20, top panel, undergo refractions at critical angles at the prescribed horizon. The take-off angles at points $\mathbf{x}_u$ and $\mathbf{x}_v$ are computed using formula (4.25). Table 4.2 shows traveltimes and coordinates x_u and x_v for the four raypaths. It is important to point out that although any two raypaths share one of the two coordinates, their total trajectory are independent. This means that it is not possible to optimize one of the two unknown coordinates (x_u or x_v) separately. To find the global minimum traveltime, both coordinates must be optimized simultaneously.

Cost function (4.26) is a two-parameter function that requires special care in order to be minimized. The anomaly in the velocity field not only introduces first-order discontinuities, but also makes $\Phi(\mathbf{x}_u, \mathbf{x}_v)$ multimodal. Figure 4.21 shows Φ as a function of both

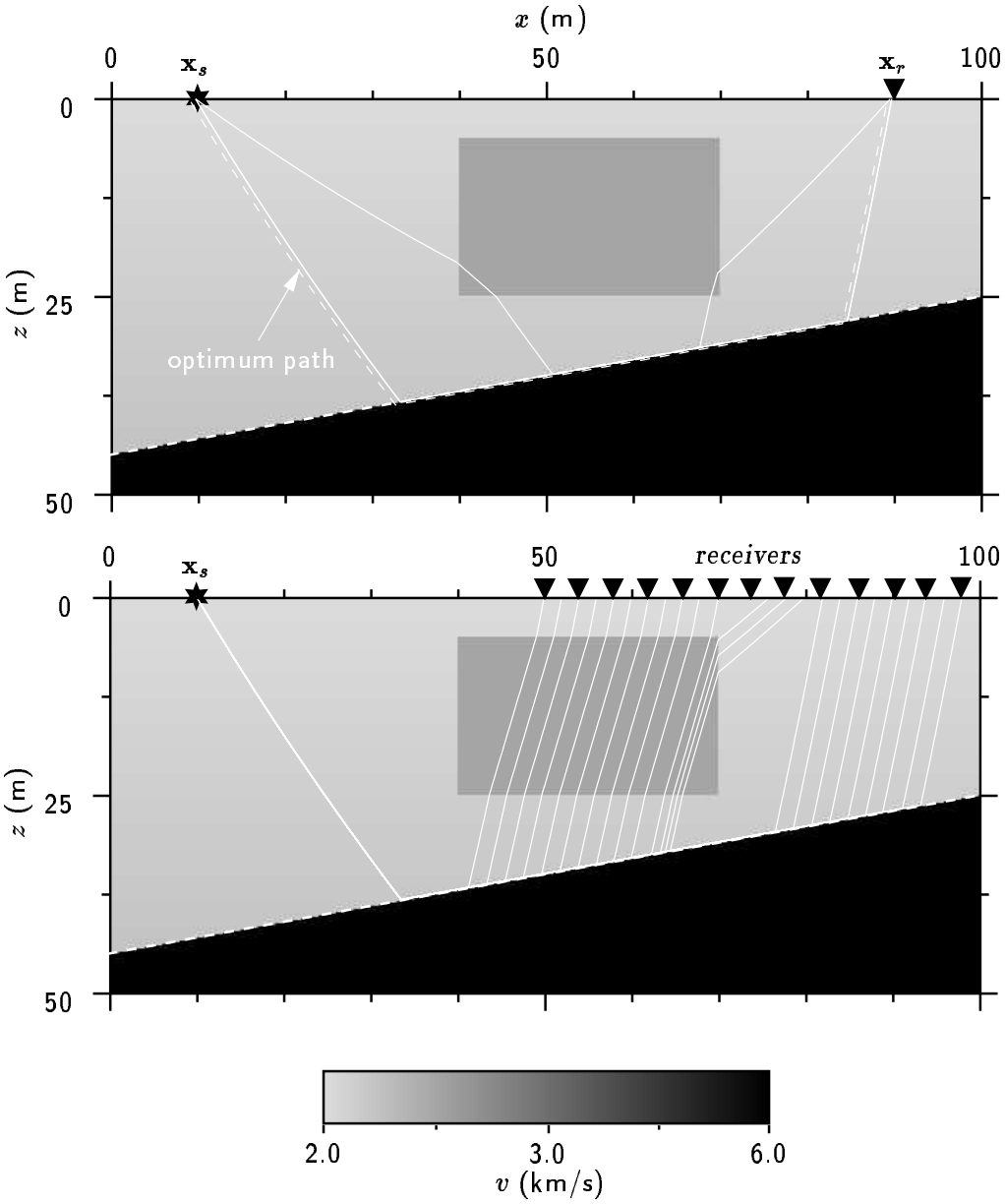

Figure 4.20: Model 3: tracing headwaves. Top panel: two-dimensional velocity model and multiple ray trajectories connecting the given source-receiver pair. The dotted line at the bottom represents the refractor. The raypath plotted with dotted line exhibits the minimum traveltime, and corresponds to the SART solution. Bottom panel: SART raypaths for a set of 25 source-receiver pairs. Note the shadow zone around $x = 72.5$ m.

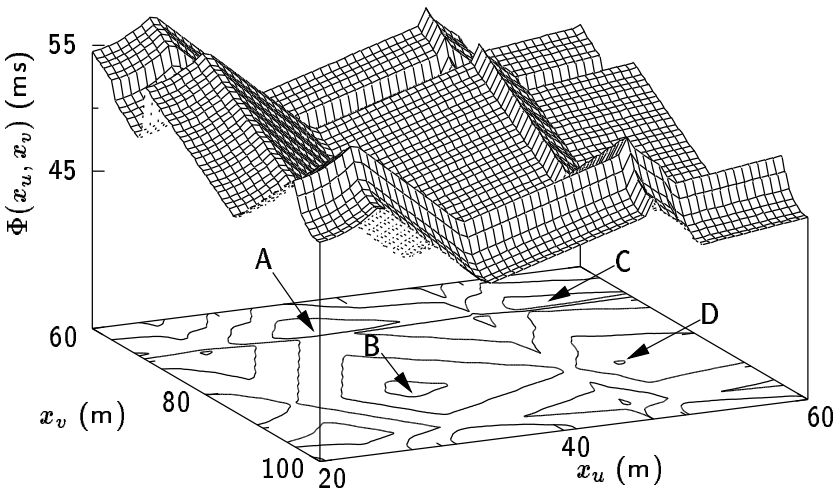

Figure 4.21: Model 3: cost function vs coordinates x_u and x_v. Contour lines correspond to $\Phi = 44$, 47, 50, 53 and 57 ms. Points A, B, C and D denote local minima. Point B is the global minimum obtained by SART.

unknown coordinates x_u and x_v. Note the complexity of the cost function, despite the fact that the model is quite simple[2]. The solution obtained by SART after a maximum of 300 iterations yields

$$x_u = 33.5, \qquad x_v = 84.9, \qquad \Phi = T_{opt} = 42.71 \text{ ms}.$$

The convergence curve is depicted in Figure 4.22.

Finally, Figure 4.20 (bottom panel) shows the resulting optimum raypaths after locating a single source at $(10, 0)$ and 25 receivers uniformly distributed along the surface between $x = 50$ m and $x = 100$ m. These raypaths were obtained using SART with a maximum of 300 iterations. Note that a few arrivals undergo a refraction at the rectangular anomaly after reaching the prescribed boundary. Also, there is a shadow zone at about $x = 72.5$ m caused by the discontinuity. In SART, shadow zones are detected

[2]Even though the cost function presents some discontinuities of first order, it is shown as a single connected mesh for plotting purposes only.

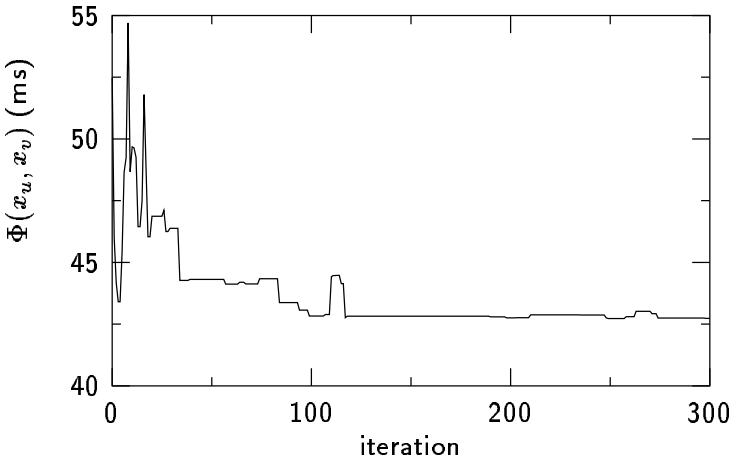

Figure 4.22: Model 3: SART convergence after 300 iterations.

when the last point in the raypath does not coincide, within a certain tolerance, with the receiver after a maximum number of iterations.

4.7 Conclusions

I presented a new method for solving the two-point seismic ray-tracing problem. It uses VFSA to find to global minimum traveltime among all possible raypaths connecting both endpoints. The strategy overcomes effectively many problems that arise in conventional two-point ray-tracing systems, namely bending and shooting methods. The problem of local minimum path in the bending methods is completely overcome by using a global optimizer. The difficulties regarding the strategy for choosing the appropriate take-off angles in the shooting method, are also obviated. This is very important particularly in 3-D models, where two take-off angles are to be examined. Also for tracing headwaves, where additional variables should be included in the search strategy. On the other hand, the addition of a few variables to the nonlinear optimization problem implies almost no extra effort for the VFSA algorithm, but certainly it does for a trial-and-error search

routine.

Later arrival times, like reflections and simple headwaves, can be easily obtained by constraining the ray to arrive at a prescribed boundary. When headwaves are involved, the problem consists of finding the coordinates of the points where the ray enters and exits the prescribed boundary at critical angles. The methodology can be extended to deal with more complex raypaths (multiples, higher order headwaves, diffractions, etc), provided the total traveltime equation can be computed, no matter how nonlinear it is with respect to the variables that define the raypath.

The method is very flexible and general in that any available one-point ray-tracing system can be used to solve the IVP at each iteration. The accuracy of the resulting traveltimes is not dependent on the parameterization of the raypath, but on the selected initial value ray-tracing system. This means that any kind of velocity models can be involved and any desired accuracy can be obtained, provided a suitable initial value ray tracer is chosen.

Note that the resulting raypath signature is known a priori and a posteriori, an issue that is very important from the point of view of phase identification.

One possible disadvantage is that raypaths are obtained one at a time and the fact that the price for obtaining global convergence is paid by requiring a high number of iterations. However in all the tests I have performed so far, a maximum of 300 iterations were enough. For simple velocity models, or single pathing, the number of iterations required for convergence may be reduced to a few tens. The addition of extra variables to define certain type of waves (e.g. headwaves), did not require more iterations for convergence.

I have tested SART in a number of clarifying synthetic examples and demonstrated the importance of obtaining the global minimum path in multipathing cases. At this point I would like to point out that the potential of the method will be more noticeable

in 3-D than in 2-D models, and particularly in complex media, where standard shooting and bending methods may fail to provide global convergence efficiently. This will be demonstrated in next chapter, where an extension of SART for 3-D models is developed.

Chapter 5

Boundary Value Ray Tracing In Complex 3-D Media

Zeus no podrá desatar las redes
de piedra que me cercan. He olvidado
los hombres que antes fui; sigo al odiado
camino de monótonas paredes
que es mi destino. Rectas galerías
que se curvan en círculos secretos
al cabo de los años. Parapetos
que ha agrietado la usura de los días.
En el pálido polvo he descifrado
rastros que temo. El aire me ha traído
en las cóncavas tardes un bramido
o el eco de un bramido desolado.
Sé que en la sombra hay Otro, cuya suerte
es fatigar las largas soledades
que tejen y destejen este Hades
y ansiar mi sangre y devorar mi muerte.
Nos buscamos los dos. Ojalá fuera
éste el último día de la espera.

Jorge Luis Borges – *El Laberinto*

5.1 Introduction

With the advent of higher computing performance, 3-D seismic processing started to play an important role in today's seismological studies. Usually, the approximation of the raypath being limited to a plane is not accurate enough for many applications. It is clear that the physical phenomena of seismic wave propagation can be better approximated by considering a realistic 3-D trajectory, especially for arbitrary laterally varying media.

Several methods for solving the two-point ray-tracing problem in 3-D have been developed in the literature [JG77, Čer87, UT87, PTE88, SK90, VF91, Per92, Sun93, MS97].

However, none of them gives a satisfactory solution to the multipathing problem. They are all either shooting or bending algorithms that proceed iteratively from an initial guess until the ray arrives to the receiver or until traveltime is stationary. When more than one raypath exists between source and receiver, it is not clear which solution the algorithm will converge to. In most cases, the algorithms converge to the raypath that is closest to the initial guess. Attempts to obtain the global minimum traveltime raypath have been made in [SK90], but convergence to the global minimum is not guaranteed. Their strategy is based on a hit-or-miss search that becomes very inefficient, and its results unpredictable, when dealing with complicated 3-D structures. In the 2-D case, as mentioned in previous chapter, methods based on graph theory for calculating the shortest path have been applied successfully to the two-point seismic ray tracing problem in 2-D media [Mos91, FL93], and recently in simple 3-D media [CH96]. When the velocity model at hand is complicated, unless a very dense network is used to allow for all possible connections, graph-theory methods do not guarantee convergence to the global minimum. In practice there is a trade-off between accuracy and computational requirements, both in terms of speed and memory. Nevertheless, they are attractive methods because several raypaths are obtained simultaneously, and provide good starting paths for more accurate iterative methods like bending. But the implementation of these techniques for complex 3-D media has not been tested yet nor their global convergence in these cases assessed.

The purpose of this chapter is twofold:

- to present an extension of SART to general 3-D media, and
- to develop a very flexible model parameterization that can be used in conjunction with SART to trace complex raypaths through arbitrary laterally varying 3-D media.

I will not describe here the advantages and disadvantages of SART with regards to the

mentioned two-point ray-tracing systems, for they were already discussed in previous chapter for the 2-D case. Issues concerning accuracy and phase identification discussed in Chapter 4 apply here, too. For example, the accuracy obtained by bending methods, including graph-theory-based methods, rely in the number of nodes used to approximate the ray trajectory. This leads also to a strong trade-off between efficiency and accuracy less serious in SART. It is worth mentioning, however, that global convergence is the main goal and advantage of SART over the other methods for very complex 3-D structures, with a relatively small computational effort.

Though the main goal of this chapter is the 3-D extension of SART, the second objective, which involves solving the forward problem, is no less important. Often, the forward problem in ray tracing is a very difficult task, especially for complex 3-D structures with arbitrary interfaces. Many codes for ray tracing in laterally varying media are available in the literature (see for instance [Čer87] for a concise review) and even from Internet, but they are usually applicable to a limited class of models (e.g. layered structures). On the contrary, the method implemented here may be applied to large class of velocity models. In the 3-D case, SART is based on the numerical solution of the ray equations. The differential equations that govern the wave propagation are integrated using standard numerical ODE solvers until the ray emerges through the desired endpoint following the absolute minimum traveltime path. The model is characterized by any number of regions separated by arbitrary curved interfaces. The velocity field within each individual region is specified separately. It may be any function of the space coordinates with the constraint of being twice differentiable. This provides greater flexibility and accuracy for dealing with general 3-D media.

SART extension to 3-D follows the same philosophy as that given in previous chapter. The BVP is put into a nonlinear optimization framework which is in turn solved by means of VFSA. This guarantees convergence to the global minimum of the cost function that

represents the total traveltime from source to receiver. Unlike the 2-D case, the cost function is at best a unimodal two-dimensional continuous smooth surface. But in real life, the cost function is a multimodal, discontinuous, nondifferentiable rough surface of two (or more) variables. To obtain the absolute minimum of such an ill-behaved function, a stochastic technique like SA becomes an invaluable tool.

After a discussion of the mathematical basis of the ray tracing system, the algorithm is tested on various synthetic examples. Although speed is not an attribute of SART, the results obtained demonstrate the ability of the method for solving the BVP in complex 3-D media where conventional methods may fail to give the desired solution in a reasonable time.

5.2 Earth model

Instead of using constant-velocity cells to represent the velocity field, this section describes a model representation that allows greater flexibility and accuracy. The system can accommodate any number of regions with different velocities separated by arbitrary interfaces. The velocity model is composed thus of any number of regions separated by curved interfaces representing geologic horizons, fault planes, etc. I assume all interfaces are arbitrary one-to-one functions given by

$$z = f(x, y). \tag{5.1}$$

The velocity within each region may be specified by any function $v = v(\mathbf{x}), \quad \mathbf{x} = (x, y, z)$, and must be twice differentiable. This model representation is quite flexible to represent a wide variety of velocity structures. More elaborate interface descriptions in terms of triangular GOCAD faces [MJC89], parametric surface patches [Per92], implicit B-splines [VF91], or natural neighbor interpolation [SBM95] can be devised, but finding the

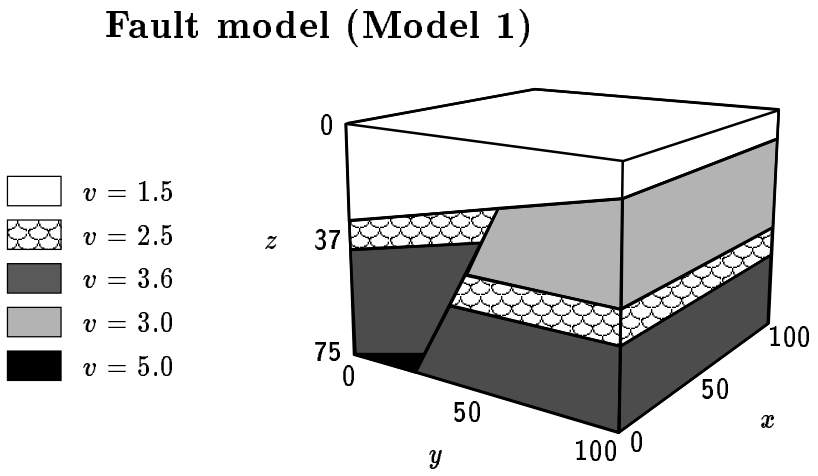

Figure 5.1: Three-dimensional model with six planar interfaces and seven constant velocity regions. Note the fault plane.

intersection points between the raypath and the surface becomes a more costly task than using explicit surfaces as in equation (5.1). As it is, SART can accommodate, however, any set of cubic B-splines with regular and/or irregular spacing to define $f(x, y)$ and obtain greater flexibility, as done in [MS97]. In SART it is also possible to have a multiple definition for $f(x, y)$ according to the values of coordinates x and y (i.e. $f(x, y)$ need not be a single analytical expression). This may imply that interfaces are nondifferentiable, which does not represent a problem for the following ray-tracing system. Take Figure 5.1 as an illustrative example. The figure shows a 3-D velocity model where several regions are determined by planar interfaces. Some interfaces do not extend for all values of x and y. For instance, the fault plane extends from $z = 75$ until it meets the top interface. More elaborate models can be easily built by considering variable velocities and curved interfaces, as will be shown later.

5.3 Solving the initial-value problem (IVP)

In Chapter 4 and in [VU96a] a cell ray-tracing scheme was used where the velocity within each rectangular cell was assumed to be constant. The ray was propagated using Snell's Law at each cell boundary, and thus it was composed of a number of segments. Cell ray tracing, although fast and useful in many applications, presents some difficulties and limitations in others. Here I develop a 3-D system based on the numerical integration of the ray equations, which is more flexible for dealing with general complex structures [Vel96].

5.3.1 The ray equations

The ray equations are well developed in the literature [Eli65, Čer87]. In their final form, they may be written

$$\begin{cases} \partial_t x = v \sin\theta \cos\xi \\ \partial_t y = v \sin\theta \sin\xi \\ \partial_t z = v \cos\theta \\ \partial_t \theta = -\cos\theta \left(\frac{\partial v}{\partial x} \cos\xi + \frac{\partial v}{\partial y} \sin\xi \right) + \frac{\partial v}{\partial z} \sin\theta \\ \partial_t \xi = \frac{1}{\sin\theta} \left(\frac{\partial v}{\partial x} \sin\xi - \frac{\partial v}{\partial y} \cos\xi \right), \end{cases} \tag{5.2}$$

where θ and ξ stand for declination and azimuth angles which describe the direction of the ray at every point of its trajectory (Figure 5.2), and $v = v(\mathbf{x})$ is the wavespeed. By solving the system of differential equations (5.2), it is possible to describe the ray trajectory at every time t, the independent variable of integration. Given the appropriate initial conditions, I solve equations (5.2) using standard ODE solvers such as Euler and Runge-Kutta methods (see for example [PTVF92]). The numerical integration requires the right-hand side of (5.2) to be continuous and $v(\mathbf{x})$ to be twice differentiable. To handle

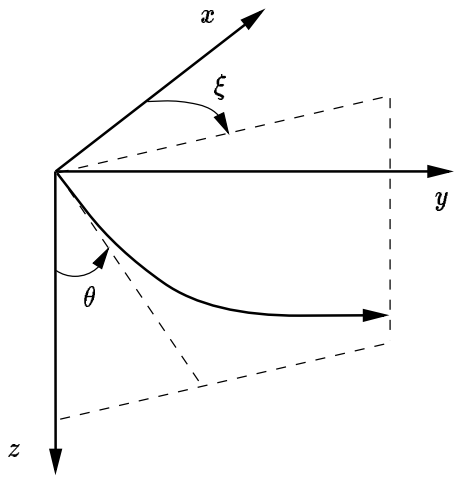

Figure 5.2: The ray direction is described by declination θ and azimuth ξ. For $t = 0$, these are the take-off angles.

discontinuities, I describe the model by a number of regions separated by arbitrary curved interfaces. Rays propagate until they find a discontinuity where Snell's Law applies.

For a ray passing from medium 1 to medium 2 at point $\mathbf{x}$, one can write

$$\mathbf{s}_2 = \mathbf{s}_1 + \left\{ \sigma \left[\frac{1}{v_2^2} - \frac{1}{v_1^2} + (\mathbf{s}_1 \cdot \mathbf{n})^2 \right]^{1/2} - (\mathbf{s}_1 \cdot \mathbf{n}) \right\} \mathbf{n} \tag{5.3}$$

where $\mathbf{s}_1$ and $\mathbf{s}_2$ are the slowness vectors on either side of the discontinuity, $\mathbf{n}$ is the unit normal to the interface, and $\sigma = \text{sign}(\mathbf{s}_1 \cdot \mathbf{n})$ is called the orientation index. If the ray is reflected, then $v_2 = v_1$ and $\sigma = -1$, resulting

$$\mathbf{s}_2 = \mathbf{s}_1 - 2(\mathbf{s}_1 \cdot \mathbf{n})\mathbf{n}. \tag{5.4}$$

Slowness vectors are computed using

$$\mathbf{s} = \frac{1}{v}(\cos\alpha_x \, \mathbf{i} + \cos\alpha_y \, \mathbf{j} + \cos\alpha_z \, \mathbf{k}), \tag{5.5}$$

where α_x, α_y, and α_z are the corresponding direction angles in rectilinear coordinates, which are related with θ and ξ by

$$\begin{cases} \cos\alpha_x = \sin\theta\cos\xi \\ \cos\alpha_y = \sin\theta\sin\xi \\ \alpha_z = \theta. \end{cases} \tag{5.6}$$

The initial conditions ($t = 0$) for solving system (5.2) are given by

$$\begin{cases} \mathbf{x}(0) = \mathbf{x}_s & \textit{initial point} \\ \theta(0) = \theta_s & \textit{initial declination} \\ \xi(0) = \xi_s, & \textit{initial azimuth} \end{cases} \tag{5.7}$$

where subscript s stands for *source*.

5.3.2 Intersection points

To take care of the interfaces that the ray propagation may encounter in its way to the receiver, the intersection point between the raypath and the corresponding interface must be found. This point, say $\mathbf{x}^i$, is found by iteratively adjusting the timestep until the ray coordinates lie right on the interface. Let $\mathbf{x}_k$ and $\mathbf{x}_{k+1}$ be the raypath coordinates at times t_k and $t_{k+1} = t_k + \Delta t$ respectively. The ray has crossed the interface of equation $z = f(x, y)$ if one of the two following conditions is satisfied:

$$\begin{cases} z_k \le f(x_k, y_k) \quad \text{and} \quad z_{k+1} > f(x_{k+1}, y_{k+1}), \\ z_k > f(x_k, y_k) \quad \text{and} \quad z_{k+1} \le f(x_{k+1}, y_{k+1}). \end{cases} \tag{5.8}$$

Once an interface crossing has been detected, a root-finder is used to find the new Δt^i which yields $z_{k+1} = f(x_{k+1}, y_{k+1})$. For this purpose, I define the function

$$g(\Delta t) = z_{k+1} - f(x_{k+1}, y_{k+1}), \tag{5.9}$$

which must be equal to zero (within a given tolerance) at the intersection point, $\mathbf{x}^i$. That is

$$g(\Delta t^i) = z^i - f(x^i, y^i) \simeq 0. \tag{5.10}$$

Since $g(0) = z_k - f(x_k, y_k)$, then the root of equation (5.9) is bracketed in the interval $[0, \Delta t]$.

The selection of an appropriate root-finder is very important. Robustness under difficult situations is required because function $g(\Delta t)$ may take any form. Also, the convergence rate must be as fast as possible for best efficiency (note that every time $g(\Delta t)$ is evaluated, the ray equations must be advanced one stepsize Δt). The bisection method never fails, but its convergence rate is very poor. A higher order approximation is achieved with Brent's method [PTVF92]. I chose this method because of its convergence rate, robustness, and because the derivatives are not required (note besides that interfaces may be discontinuous). Brent's algorithm has quadratic termination in most cases and is as robust as the bisection method [PTVF92]. In the worst cases, such as pathological functions, Brent's method takes a bisection step. In practice I have found that one to three adjustments are enough to find the intersection point between the raypath and the interface within a small tolerance. Once the intersection point has been found, the current raypath coordinates and traveltime are updated accordingly:

$$\begin{cases} \mathbf{x}_{k+1} = \mathbf{x}^i, \\ t_{k+1} = t_k + \Delta t^i. \end{cases} \tag{5.11}$$

Here the slowness vector is updated using the reflection/transmission laws given in equations (5.3) and (5.4). Then the timestep is restored to its initial value and the ray propagation continues with the new initial conditions. If more than one interface is crossed at

a given timestep, the one whose corresponding adjusted timestep is smallest is selected as the candidate interface for applying the reflection/transmission laws. The same kind of iterative adjustment is done when the ray crosses any of the model boundaries. In this particular case, the propagation is terminated.

The unit normal to the surface $z = f(x, y)$ at a generic point $\mathbf{x}$, is calculated by

$$\mathbf{n} = \frac{1}{p}\left(\frac{\partial f}{\partial x}\mathbf{i} + \frac{\partial f}{\partial y}\mathbf{j} - \mathbf{k}\right), \qquad p = \left[\left(\frac{\partial f}{\partial x}\right)^2 + \left(\frac{\partial f}{\partial y}\right)^2 + 1\right]^{1/2}, \tag{5.12}$$

which is needed when applying Snell's Law at the intersection point. This requires the additional condition of differentiability, at least locally, of the interfaces.

5.3.3 Ray signature and stopping conditions

During the propagation, an index indicating the current region is saved and updated after each interface crossing. This index is used to determine which velocity function must be used in the integration of the equations (5.2). A flag for each interface is also provided which indicates the decision to make in case the ray arrives at the interface (see "Ray signature and stopping conditions" in Chapter 4). This allows us to model *P*- and *S*-waves, or even a conversion between the two, or to force a reflection at any given interface to simulate a reflector.

Note that there are various stopping conditions that may be implemented. The ray propagation stops whichever of the following conditions occurs first:

1. the ray arrives at some model boundary,

2. the ray arrives at some prescribed interface (e.g. some target plane),

3. a refraction has been requested but total reflection occurred,

4. the number of points in the raypath or the traveltime is greater than a predefined value, etc.

In my experiments, conditions (1) and (2) are set by default. For simplicity, I locate all sources and receivers on the model boundaries. In general, a refraction is requested at all interfaces, except at some individual interface for modeling reflections or headwaves. In case a refraction is not possible because the incident angle is beyond the critical angle [negative square-root argument in formula (5.4)], a reflection is generated and this is notified on output.

In summary, given an initial point and an initial direction the system described above is able to propagate any ray provided the velocity and interfaces can be evaluated at any point within the model boundaries, and provided a ray signature is specified a priori to make a decision at each interface intersection. It is also required to know which region is being traversed at any point. Raypaths honor bending in inhomogeneous regions and Snell's Law at discontinuities.

5.4 Solving the boundary-value problem (BVP) by means of SART

As already described in previous chapter for the 2-D case, the BVP is solved by means of SART. In 3-D the procedure is analogous, being the number of degrees of freedom the only fundamental difference. Of course, in this scheme I use a numerical ray-tracing algorithm instead of a cell ray-tracing one. But this does not concern to SART, a system which is independent of the selected IVP solver. The straight-ray construction applies here too, in such a way that the raypath is forced to reach the desired endpoint. Direct waves, reflections, headwaves, diffractions, etc., are treated similarly as in the 2-D case.

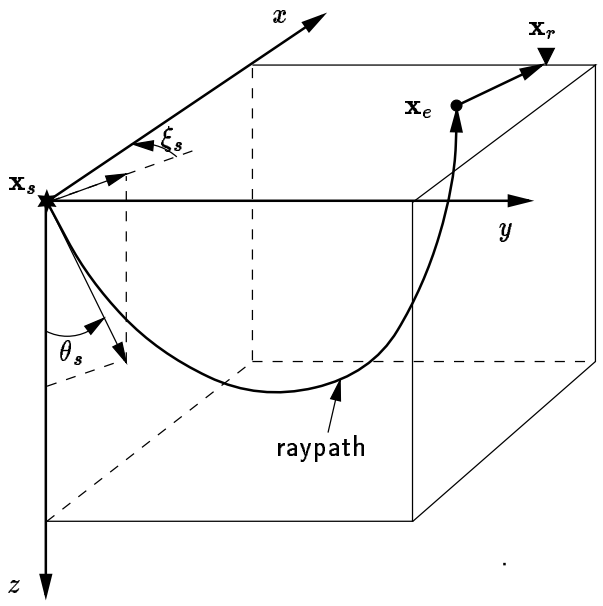

Figure 5.3: Ray tracing direct waves through a 3-D media using SART. At convergence $\mathbf{x}_e \equiv \mathbf{x}_r$.

5.4.1 Problem definition

Direct waves

Tracing direct waves is the simplest case and the basis for tracing the other types of waves. Both source and receiver are fixed and the optimum take-off angles θ_s^{opt} and ξ_s^{opt} are to be found so that the total traveltime is a global minimum. The total traveltime is written as

$$T = \int_{\mathbf{x}_s}^{\mathbf{x}_r} s \, dl = T_{se} + T_{er}. \tag{5.13}$$

In equation (5.13), T_{se} is the traveltime which is obtained after solving the IVP for the given initial conditions. The second term in equation (5.13), T_{er}, is the traveltime associated with the straight-ray construction (see Figure 5.3). Since there are two angles

needed to determine uniquely the whole ray trajectory, traveltime T is a two-dimensional function, thus

$$T = T(\theta_s, \xi_s). \tag{5.14}$$

which is in general multimodal and nondifferentiable. When $T = minimum$, Fermat's principle is satisfied and point $\mathbf{x}_e$ coincides with the receiver. Thus, it can be seen that the only difference between the 2-D and the 3-D case is that the whole final raypath is now defined in terms of two unknowns instead of just one. If in the 2-D case a simple exhaustive search could have been used to find the global minimum of the traveltime equation (4.11), in the 3-D case the same procedure is not recommended for obvious reasons, unless T is a very simple function. Unfortunately, T is in general multimodal and nondifferentiable, which makes the optimization problem a difficult numerical task.

Reflected waves

The case of reflected waves again involves the minimization of a two-dimensional cost function in terms of the two take-off angles:

$$\Phi = \Phi(\theta_s, \xi_s) = \begin{cases} T_{su} + T_{ue} + T_{er}, & \text{if the ray arrives to } z = f(x,y) \\ T_{max}, & \text{otherwise} \end{cases} \tag{5.15}$$

where point $\mathbf{x}_u$ is the point where the ray intersects the reflector, $z = f(x, y)$, and T_{max} is the maximum guessed value the total traveltime may take for all possible take-off angles. As usual, the ray is propagated from the source with take-off angles θ_s and ξ_s until it arrives to the reflector at point $\mathbf{x}_u$ (see Figure 5.4). Here Snell's Law of reflection is applied and the propagation continues until the ray leaves the model boundaries at the

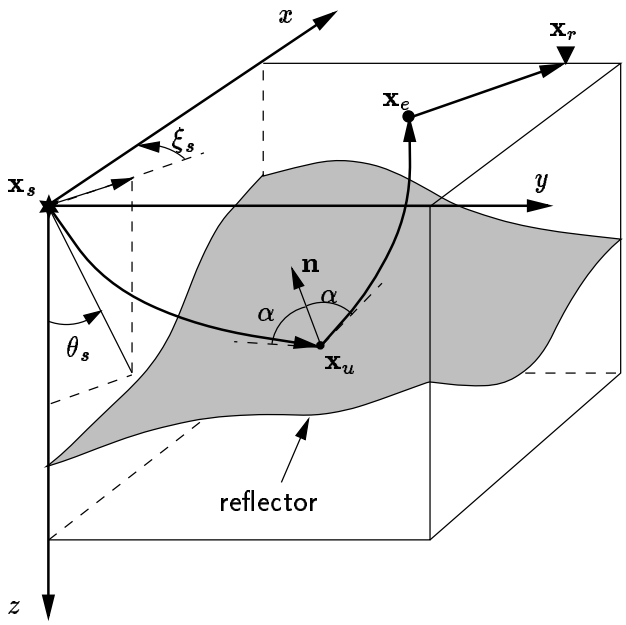

Figure 5.4: Ray tracing reflections through a 3-D media using SART. At convergence $\mathbf{x}_e \equiv \mathbf{x}_r$. An alternative strategy is described in the text.

emerging point $\mathbf{x}_e$. Finally, this point is connected with the receiver using a straight line. When total traveltime Φ is minimum, $\mathbf{x}_e$ coincides with $\mathbf{x}_r$.

As in the 2-D case, alternative strategies for dealing with reflected waves can be devised. One of such alternative strategies starts the propagation from the unknown point $\mathbf{x}_u$ on the reflector surface towards the source with take-off angles θ_u and ξ_u, also unknowns. Then the raypath is completed by shooting towards the receiver starting at $\mathbf{x}_u$ with take-off angles θ'_u and ξ'_u. These angles can be expressed in terms of the slowness vector $\mathbf{s}_u$ and the unit normal to the surface, $\mathbf{n}$, according to equation (5.4). Then

$$\begin{cases} \cos\theta'_u = \cos\theta_u - 2an_z \\ \sin\theta'_u \cos\xi'_u = \sin\theta_u \cos\xi_u - 2an_x, \end{cases} \tag{5.16}$$

where

$$a = \mathbf{s}_u \cdot \mathbf{n} = n_x \sin\theta_u \cos\xi_u + n_y \sin\theta_u \cos\xi_u + n_z \cos\theta_u \tag{5.17}$$

is the dot product between the slowness vector and the unit normal. As a result, the total traveltime becomes a four dimensional function

$$\Phi' = \Phi'(\theta_u, \xi_u; x_u, y_u) = T_{ue_1} + T_{e_1 s} + T_{ue_2} + T_{e_2 r}. \tag{5.18}$$

All raypaths correspond to reflections now, and there is no need for penalizing those rays not undergoing any reflection as in the previous case. In equation (5.18), $\mathbf{x}_{e_1}$ and $\mathbf{x}_{e_2}$ are the emerging points after shooting from $\mathbf{x}_u$ with take-off angles (θ_u, ξ_u) and (θ'_u, ξ'_u) respectively. The terms $T_{e_1 s}$ and $T_{e_2 r}$ are the traveltime contributions associated to the corresponding straight-ray construction, as done for the direct wave case.

Normal rays The normal rays case is a particular case of reflections. The problem can be cast as finding the optimum coordinates, (x_u, y_u), of the ray that travels towards the source-receiver point with an initial direction which is parallel to the normal of the surface at that point. That is, $\mathbf{s} = \mathbf{n}/v$. Take-off angles are then calculated from

$$\begin{cases} \cos\theta_u = n_z, \\ \sin\theta_u \cos\xi_u = n_x, \quad (\text{or} \quad \sin\theta_u \sin\xi_u = n_y). \end{cases} \tag{5.19}$$

Consequently, the total traveltime to be globally minimized is written as

$$\Phi = \Phi(x_u, y_u) = 2(T_{ue} + T_{es}), \tag{5.20}$$

again a two-dimensional function.

Headwaves

Now the number of unknowns is four: the coordinates of the points the ray enters and leaves the refractor: $\mathbf{x}_u$ and $\mathbf{x}_v$. Actually, the ray is propagated starting at these two locations towards the source and receiver respectively. As a result the final raypath is composed of five portions, as described in Chapter 4 for the 2-D case. Figure 5.5 depicts how SART handles a headwave in a 3-D media. Here I will assume the refractor is a plane interface of equation $z = a_0 + a_1 x + a_2 y$ with velocity $v_r = constant$. So, the raypath connecting $\mathbf{x}_u$ with $\mathbf{x}_v$, which is the shortest path, honors the parametric equation

$$\begin{cases} x = \lambda, & \lambda \in [x_u, x_v] \\ y = y_u + b_0(\lambda - x_u) \\ z = z_u + b_1(\lambda - x_u) \end{cases} \tag{5.21}$$

where b_0 and b_1 are constants given by

$$\begin{cases} b_0 = (y_v - y_u)/(x_v - x_u), \\ b_1 = (z_v - z_u)/(x_v - x_u). \end{cases} \tag{5.22}$$

Slowness vectors $\mathbf{s}_r$ (see Figure 5.5) at both points are expressed in terms of the tangent vector to the parametric curve defined in equation (5.21):

$$\mathbf{s}_r = \frac{1}{v_r p}\left(\frac{\partial x}{\partial \lambda}\mathbf{i} + \frac{\partial y}{\partial \lambda}\mathbf{j} + \frac{\partial z}{\partial \lambda}\mathbf{k}\right), \qquad p = \left[\left(\frac{\partial x}{\partial \lambda}\right)^2 + \left(\frac{\partial y}{\partial \lambda}\right)^2 + \left(\frac{\partial z}{\partial \lambda}\right)^2\right]^{1/2}. \tag{5.23}$$

At point $\mathbf{x}_v$ the slowness vector follows the direction of this ray segment. At point $\mathbf{x}_u$, the range of variation of λ must be reversed, so that slowness vectors have opposite sign. Computing the derivatives, it yields

$$\mathbf{s}_r = \begin{cases} -\frac{1}{v_r p}(\mathbf{i} + b_0\mathbf{j} + b_1\mathbf{k}) & \text{at point } \mathbf{x}_u \\ \frac{1}{v_r p}(\mathbf{i} + b_0\mathbf{j} + b_1\mathbf{k}) & \text{at point } \mathbf{x}_v \end{cases} \tag{5.24}$$

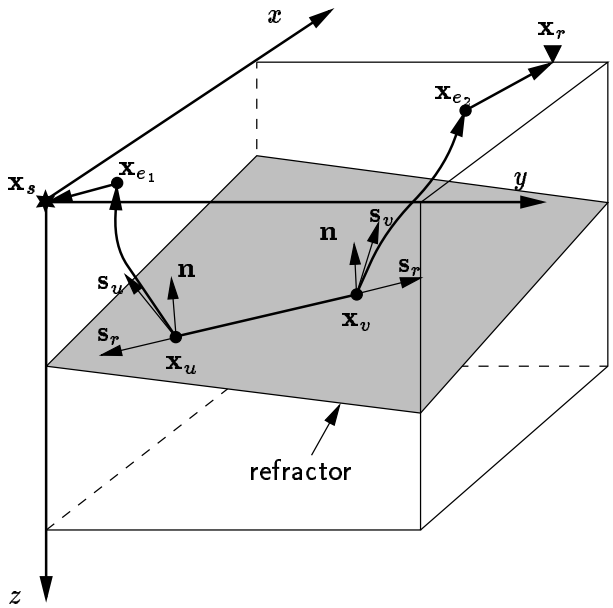

Figure 5.5: Ray tracing headwaves through a 3-D media using SART. The ray travels along the refractor with $v = v_r$, from $\mathbf{x}_u$ to $\mathbf{x}_v$. At convergence $\mathbf{x}_{e_1} \equiv \mathbf{x}_s$, and $\mathbf{x}_{e_2} \equiv \mathbf{x}_r$.

where

$$p = (1 + b_0^2 + b_1^2)^{1/2}. \tag{5.25}$$

The corresponding take-off angles at the unknown points $\mathbf{x}_u$ and $\mathbf{x}_v$ are again computed according to Snell's Law (5.3) with $\sigma = -1$, taking into account that $v_1 = v_r$, and $v_2 = v$, $v_2 < v_r$, is the velocity right above the refractor at $\mathbf{x}_u$ or $\mathbf{x}_v$ (I assume source and receiver are always placed above the refractor).

Since $\mathbf{s}_r$ is perpendicular to $\mathbf{n}$,

$$\mathbf{s}_r \cdot \mathbf{n} = 0. \tag{5.26}$$

Then

$$\mathbf{s} = \mathbf{s}_r - \left(\frac{1}{v^2} - \frac{1}{v_r^2}\right)^{1/2} \mathbf{n} \tag{5.27}$$

From here the corresponding take-off angles (θ_u, ξ_u) and (θ_v, ξ_v) needed to continue the ray propagation towards source and receiver respectively are derived easily from equations (5.5) and (5.6). These take-off angles feed the IVP solver at each SART iteration, until the optimum points $\mathbf{x}_u^{opt}$ and $\mathbf{x}_v^{opt}$ that globally minimize the total traveltime is obtained. Total traveltime is written in terms of the four unknown coordinates:

$$\Phi = \Phi(x_u, y_u; x_v, y_v) = T_{se_1} + T_{e_1u} + T_{uv} + T_{ve_2} + T_{e_2r}. \tag{5.28}$$

In this equation, the first and the last contributions to the sum correspond to the straight-ray constructions, which at convergence go to zero. Term T_{uv} is the traveltime along the refractor from $\mathbf{x}_u$ to $\mathbf{x}_v$. The remaining two terms are obtained after integrating the ray equations as usual.

5.4.2 SART accuracy

As described in previous paragraphs, the global minimum traveltime is obtained after minimizing, by means of VFSA, the appropriate cost function which defines a given wave phase. In this section a brief discussion about the accuracy of the traveltimes obtained using SART is given.

In SART, the ray equations are solved by means of standard ODE solvers, e.g. fourth-order Runge-Kutta (RK). In general, traveltimes obtained by SART are very accurate inasmuch as SA converged to the global minimum. The accuracy can be increased at any desired level (within machine precision) by requiring greater accuracy in the calculation of the intersection points between raypaths and interfaces, and by decreasing the integration

stepsize. This *solution-polishing* process can be done only after SART convergence, so that the computational cost added up as a result of this extra job is negligible. For smooth velocity fields, a fourth-order RK method for integrating the ray equations yields very accurate results (fourth-order RK is exact up to fourth-order). So large stepsizes can be used with confidence for better efficiency. The only issue to be taken into account is that all interface crossings must be detectable at any stage of the ray propagation, and that the intersection points must be found correctly by the iterative root-finder. This means that if the stepsize is too large, some interfaces may be overlooked or the root-finder may not converge to the appropriate solution when looking for the intersection points (consider the case in which the raypath intercepts a given interface at more than one point). These issues are generally of no concern when interfaces are planar (very large stepsizes can be used here). But when interfaces are very irregular and there are many of them, too large stepsizes should not be used blindly. So, there is a trade-off between efficiency/accuracy and stepsize. When velocity fields within some regions vary rapidly, the use of RK with adaptive stepsize is recommended. This feature is also implemented in SART. Otherwise, RK with constant stepsize should be used for faster performance.

The main factor of error is perhaps associated with the distance between the actual ray endpoint and the receiver (see for example [SK90]). However, this problem is virtually eliminated by refining the solution after VFSA convergence, as will be explained below. In summary, the ray-tracing system described so far can be made extremely accurate provided all interface crossing are detected properly.

5.4.3 Refinement of the solution: a hybrid strategy

Due to the nature of its generating function, the convergence achieved by VFSA at lower temperatures is faster than other SA algorithms. Despite this fact, taking T_{er} to zero may take several iterations after reaching the global minimum neighborhood. At

these low temperature stages, when the solution is close to the global minimum, SART switches to a local optimization algorithm to make $d_{er}^2 = \|\mathbf{x}_e - \mathbf{x}_r\|^2 \leq \epsilon_d$, for some small value of ϵ_d. Here the best SA solution obtained so far is used as the initial guess for the linearizing stage. The switch is done after the maximum number of SA iterations, ITMAX, has been reached. This hybrid strategy allows SART to accurately compute the global minimum traveltime in an efficient manner. Since the local optimization algorithm is applied only after SA has converged close enough to the global minimum, problems regarding instability and divergence associated with these methods are of no concern.

Any linearizing method can be used to minimize d_{er}. I selected two alternative algorithms for doing the task: the *method of steepest descent* (SD) and a *quasi-Newton method* (QN). I chose these two methods in part because none of them require the calculation of the Hessian matrix (cost function second-order derivatives), but only the gradient. Since in SART the gradient of the cost function is not available, it is approximated by finite differences, which has been found to be accurate enough. The SD method is much simpler in terms of coding and less computations are required in each iteration. However, as it is well known, the convergence is very poor (the rate of convergence of the method of steepest descent with exact linear search is first order in general). Newton methods have quadratic termination but require the computation of the Hessian matrix. Quasi-Newton methods can have quadratic termination too without second derivatives, and are economical on first derivatives and cost function evaluations when compared to other higher-order algorithms. QN methods are usually more rapidly convergent, robust and economical than conjugate gradient methods [Sca85], but require much more storage. This is not a problem in SART, since the unknown variables are just a few.

The optimization problem at the refinement stage is written

$$\text{minimize} \quad \Phi(\theta_s, \xi_s) = \|\mathbf{x}_e - \mathbf{x}_r\|^2 = \sum_{i=1}^{3} (x_i - x_{r_i})^2, \tag{5.29}$$

where x_i are the coordinates of the emerging point, $\mathbf{x}_e$, and x_{r_i} are the coordinates of the receiver point, $\mathbf{x}_r$. Since equation (5.29) is nonlinear, the minimization is done iteratively. At the beginning of iteration j-th, the current take-off angles estimates are (θ_s^j, ξ_s^j). The j-th iteration then consists of the computation of a *search vector* $(\Delta\theta_s^j, \Delta\xi_s^j)$ from which to obtain the new estimate $(\theta_s^{j+1}, \xi_s^{j+1})$ according to

$$\begin{cases} \theta_s^{j+1} = \theta_s^j + \alpha^j \Delta\theta_s^j, \\ \xi_s^{j+1} = \xi_s^j + \alpha^j \Delta\xi_s^j, \end{cases} \tag{5.30}$$

where α^j is obtained by linear search or other strategy. But the selection of the search vector $(\Delta\theta_s^j, \Delta\xi_s^j)$ is largely what distinguishes one method from another. In the SD approach the greatest reduction in the cost function value is obtained in the direction of the negative gradient, thus

$$(\Delta\theta_s^j, \Delta\xi_s^j) = \nabla\Phi^j = \left[2\sum_{i=1}^{3} (x_i - x_{r_i}) \frac{\partial x_i}{\partial \theta_s}, 2\sum_{i=1}^{3} (x_i - x_{r_i}) \frac{\partial x_i}{\partial \xi_s} \right]^j. \tag{5.31}$$

The derivatives in the above equation are estimated using finite differences

$$\begin{cases} \partial x_i / \partial \theta_s \simeq \left[x_i(\theta_s + \epsilon, \xi_s) - x_i(\theta_s, \xi_s) \right] / \epsilon, \\ \partial x_i / \partial \xi_s \simeq \left[x_i(\theta_s, \xi_s + \epsilon) - x_i(\theta_s, \xi_s) \right] / \epsilon, \end{cases} \tag{5.32}$$

where ϵ is a small positive scalar. Once the search direction has been obtained, the scalar α^j which measures the stepsize must be computed. For this purpose, I write the Taylor expansion to first-order approximation

$$x_i(\theta_s + \Delta\theta_s, \xi_s + \Delta\xi_s) \simeq x_i(\theta_s, \xi_s) + \alpha^j a_i^j, \tag{5.33}$$

where

$$a_i^j = \left(\Delta\theta_s \frac{\partial x_i}{\partial \theta_s} + \Delta\xi_s \frac{\partial x_i}{\partial \xi_s} \right)^j, \qquad i = 1, 2, 3. \tag{5.34}$$

Replacing equation (5.33) into equation (5.29) and differentiating with respect to α, it yields

$$\frac{\partial \Phi}{\partial \alpha} = 2 \sum_{i=1}^{3} (x_i^j + \alpha^j a_i^j - x_{r_i}) a_i^j. \tag{5.35}$$

Finally, setting $\frac{\partial \Phi}{\partial \alpha} = 0$

$$\alpha^j = \frac{\sum_i (x_i^j - x_{r_i}) a_i^j}{\sum_i a_i^2}. \tag{5.36}$$

Quasi-Newton methods are based upon the fact that the Hessian matrix (or its inverse) is approximated with an updating formula which carries information about second derivatives. There are various QN methods that use slightly different updating formulas and linear search approximations. In particular I use the Broyden-Fletcher-Goldfarb-Shanno algorithm [PTVF92], which is one of the most efficient QN methods. The details can be found in [Sca85].

As mentioned above, the local optimization to solve problem (5.29) is only applied when SART is very close to the global minimum (after ITMAX SA iterations). Locally, the cost function $d_{er}^2 = \|\mathbf{x}_e - \mathbf{x}_r\|^2$ is a well-behaved function, and the convergence to the global minimum is guaranteed. In practice a few iterations (two to eight) are enough to take d_{er}^2 virtually to zero within machine precision. SD method takes more iterations that QN method, but in all the total number of cost function evaluations is about the same in both cases. As a result the computational costs using either method are similar.

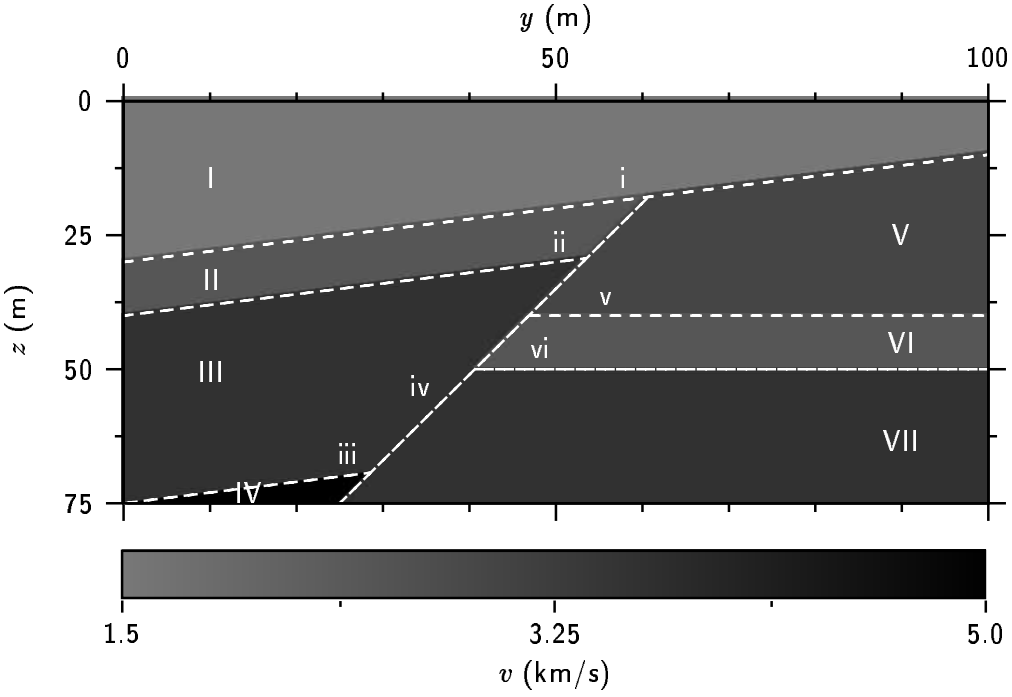

Figure 5.6: Model 1: two-dimensional slice through the fault model in Figure 5.1. Labels refer to the regions and interfaces listed in Table 5.1.

5.5 Numerical examples

In this section I test SART using two models representing 3-D geological structures: Model 1 and Model 2. The first one represents a fractured layered subsurface with planar interfaces and constant velocities within each region, which I call "fault model". The second one is called "salt model", and represents a salt dome that bursts into a layered media. Here interfaces are curved and velocities are not constant within all regions. Though Model 1 is much simpler than Model 2 in terms of raypath and traveltime behavior, both models are intended to illustrate the difficulties that may arise to solve the two-point ray tracing problem in laterally varying 3-D media. For a given source-receiver geometry, these models exhibit multipathing and complicated traveltime curves that make the optimization problem a very difficult one. After showing SART results and

Region #	$v = v(\mathbf{x})$ (km/s)	Interface #	$z = f(x, y)$ (m)
I	1.5	i	$30 - 0.2y$
II	2.5	ii	$40 - 0.2y$
III	3.6	iii	$75 - 0.2y$
IV	5.0	iv	$115 - 1.6y$
V	3.0	v	40
VI	2.5	vi	50
VII	3.6	–	–

Table 5.1: Model 1: velocities and interfaces defining the fault model shown in Figure 5.6.

drawing the attention to the multipathing problem, I present a brief analysis regarding the number of iterations required to obtain the global minimum traveltime.

For simplicity, I located all receivers on some model boundary, so that the target interface coincides with some of the planes defining the model boundaries. Distances and coordinates are expressed in *kilometers*, velocities in *kilometers per second*, traveltimes in *milliseconds* and angles in *degrees*.

5.5.1 Fault model

Model 1 is comprised of seven regions with constant velocities delimited by planar interfaces and a fault plane, as shown in Figure 5.1. For simplicity, I first take into account a 2-D slice of the model: the plane $x = 50$ (Figure 5.6). I placed a source at $(50, 0, 70)$ and produced a fan of rays with take-off angles θ_s varying from 60^o to 140^o (ξ_s is fixed at 90^o so that all ray trajectories lie on the plane $x = 50$). The top panel of Figure 5.7 shows the resulting raypaths. The bottom panel of the same figure shows the wavefronts for $t = 5$ ms, $t = 7.5$ ms, $t = 10$ ms, and so on. Since the velocity within each region is

constant each wavefront is an arc of a circle with radius $\sum_i v_i t_i$. It is clear from the graph the presence of shadow zones and multiple arrivals to any receiver located either on the surface or on the right model boundary. Due to incident angles beyond the critical one, total reflected waves are also generated at some discontinuities.

Using a simple grid-search algorithm I have found all solutions to the BVP (source fixed) corresponding to a set of 76 receivers uniformly distributed according to

$$\mathbf{x}_{ri} = [50, 100, i \times h], \qquad i = 0, 1, \cdots, 75 \tag{5.37}$$

where $h = 1.0$ m is the geophone vertical spacing. In all cases the source was fixed at $(50, 0, 70)$. Figure 5.8a shows the resulting arrival times, and Figure 5.8b the corresponding common-shot gather. For simplicity, amplitudes are not taken into account here. The shot gather was constructed by convolving a zero-phase Ricker wavelet (sampling interval $\Delta t = 0.25$ ms, center frequency $f_0 = 500$ Hz) with a reflectivity series made of unit spikes located at the arrival times. Note that there are various nearly identical arrivals. Later arrivals at depths between 40 and 50 m correspond to multiple reflected waves (total reflection) generated at the low velocity region VI. Geophones at depths below 67.5 m do not receive any arrival (shadow zone). For a better understanding, this figure is to be viewed in conjunction with Figure 5.7, where raypaths and wavefronts are shown.

By placing a receiver at $(50, 100, 16.5)$ (marked in Figure 5.8), I computed distance d and traveltime T, equation (5.14), as a function of θ_s (Figure 5.9a and 5.9b, respectively). The nonlinearity of the objective functions and the complexity of the optimization problem are evident by observing the plots. Local minima, multipathing, and discontinuities are all present in this simple two-dimensional model. Distance d has four zeros (multipathing) and two extra local minima, as well as various discontinuities generated by the

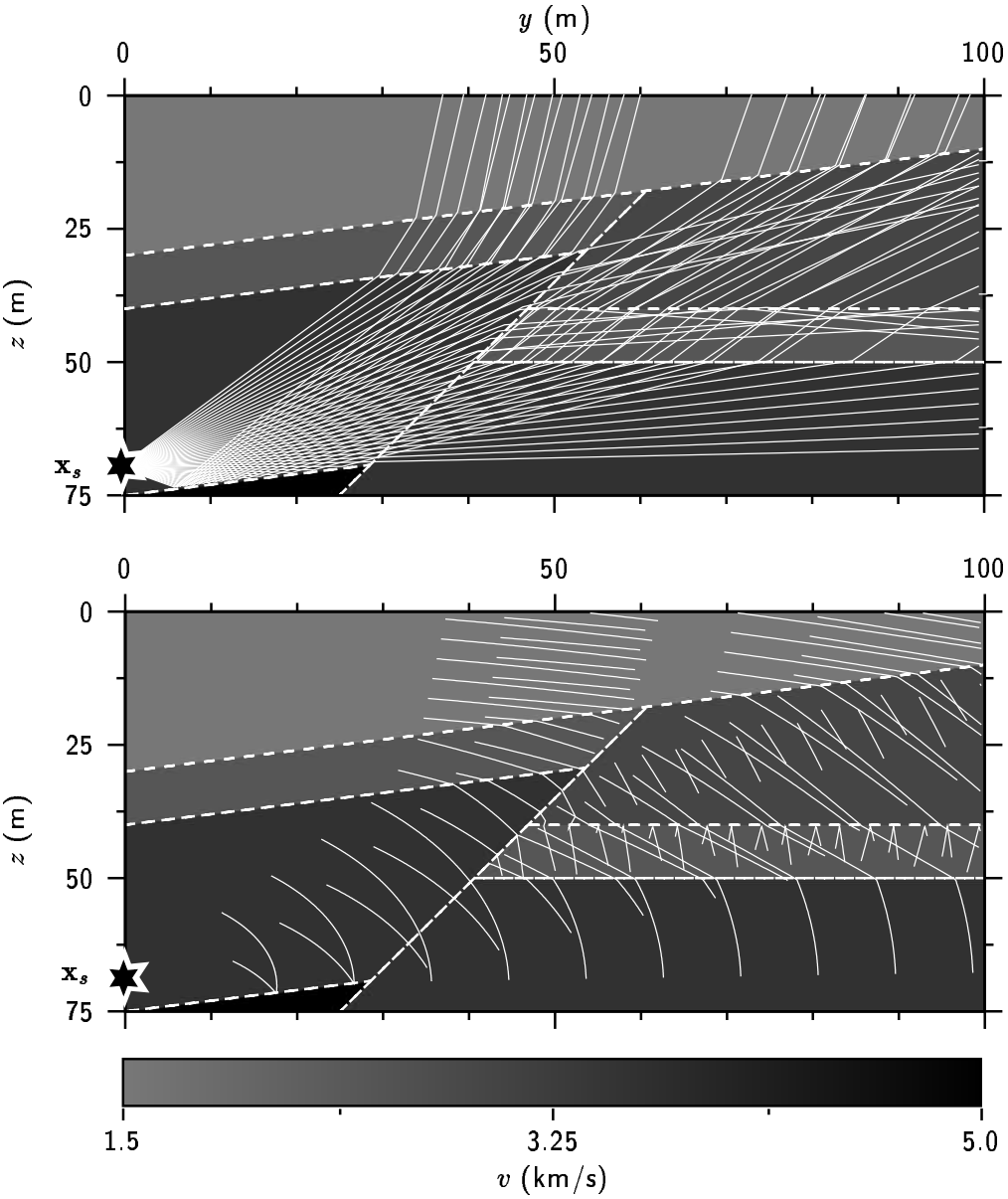

Figure 5.7: Model 1. Top panel: fan of rays with equally spaced take-off angles in the range $(60°, 140°)$. Bottom panel: a series of wavefronts for $t = 5$ ms, 7.5 ms, 10.0 ms, etc. Note the presence of shadow zones and multiple arrivals.

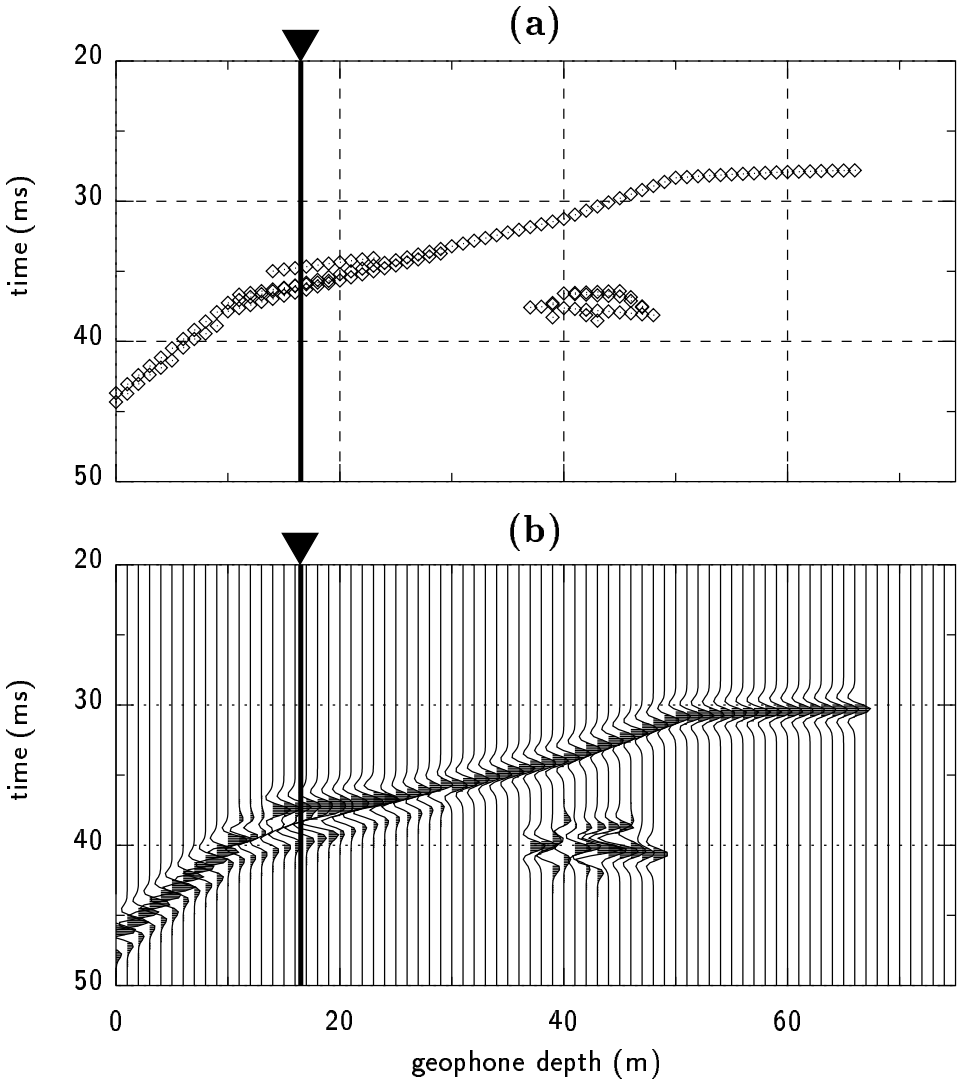

Figure 5.8: Model 2: (a) Traveltime vs geophone depth, and (b) common-shot section, for a set of 76 source-receiver pairs. The source is fixed at $(50, 0, 70)$. Amplitudes in the shot gather are arbitrary.

Ray #	θ_s (deg)	Φ_p (ms)
1	70.16	35.986
2	87.44	36.410
3	108.21	35.915
4	**125.90**	**34.714**

Table 5.2: Model 1: multipathing in the fault model. The raypath number 4 exhibits the absolute minimum traveltime.

velocity structure. Clearly, the multipathing problem cannot be obviated by minimizing d, since there is no unique global minimum. The shape of curve T is similar to the shape of curve d, but the former makes it possible to differentiate among those rays arriving to the receiver. In effect, the curve has a single global minimum:

$$\theta_s = 125.90^o, \qquad T_{opt} = 34.714 \text{ ms.}$$

The other rays arriving to the receiver become local minima now. These raypaths are plotted in Figure 5.10 (top panel), and their corresponding take-off angles and traveltimes are summarized in Table 5.2. Note that two of them correspond to reflected waves due to large incident angles at the deepest model discontinuity. In is important to point out that in a tomographic experiment, for example, a failure to assign the correct trajectories to the picked first-arrival traveltimes, may lead to a wrong interpretation of the underlying velocity model. This is because the ray trajectories are very different and methods like shooting or bending are not capable, in general, to provide the global minimum path. Figure 5.10 (bottom panel) shows all optimum raypaths connecting the source and the receivers distributed along the right vertical line. Note that 8 out of 76 geophones did not receive any arrival (shadow zone).

In the previous two-dimensional example, an grid search is viable in order to find the

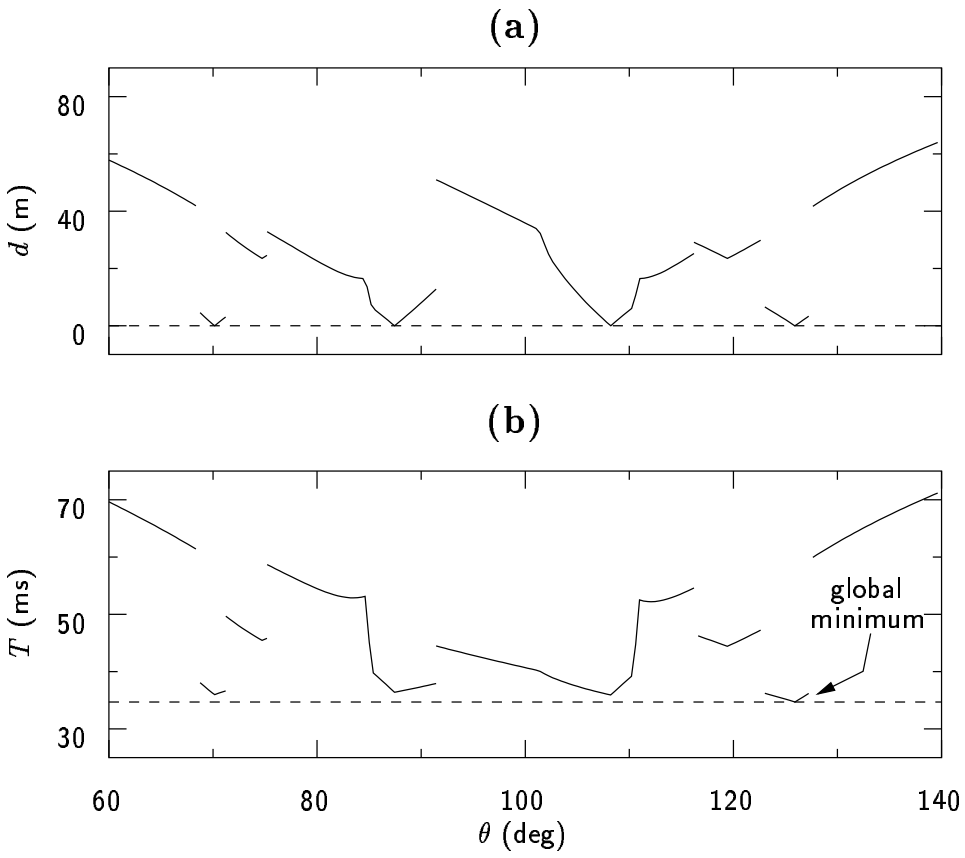

Figure 5.9: Model 1: two cost functions for the fault model. (a) Distance d, and (b) traveltime T, as a function of take-off angle θ_s. Azimuth ξ_s has been fixed to 90°.

absolute minimum raypath in a reasonably number of iterations (as done implicitly for constructing Figure 5.9). But in a three-dimensional model, where raypaths are specified by more than one parameter, a grid search is not recommended for obvious reasons (see section 5.6.3). In addition, the complexity of the cost function increases significantly, particularly in complicated structures. Figure 5.11 shows traveltime T for the three-dimensional fault model shown in Figure 5.1. Observe the complex topography of T. What makes it difficult to globally minimize this function is not only the presence of local minima, but also the great number of discontinuities generated by the model. SA appears

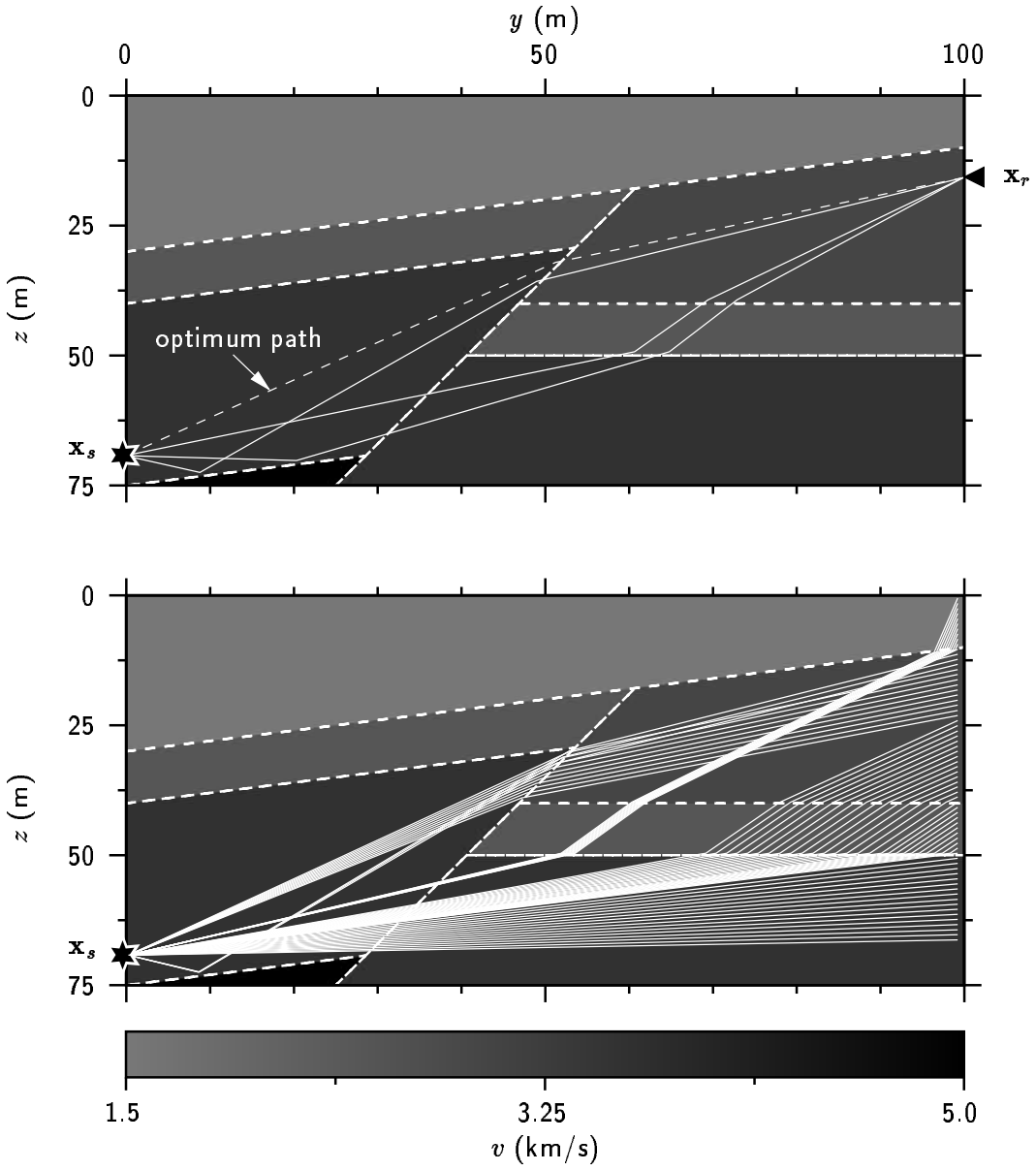

Figure 5.10: Model 1: multipathing in the fault model. Top panel: the four plotted raypaths satisfy the ray equations, but exhibit different traveltimes. The dotted raypath is the SART solution. Bottom panel: a total of 68 receivers are linked with the same source following the path with global minimum traveltime.

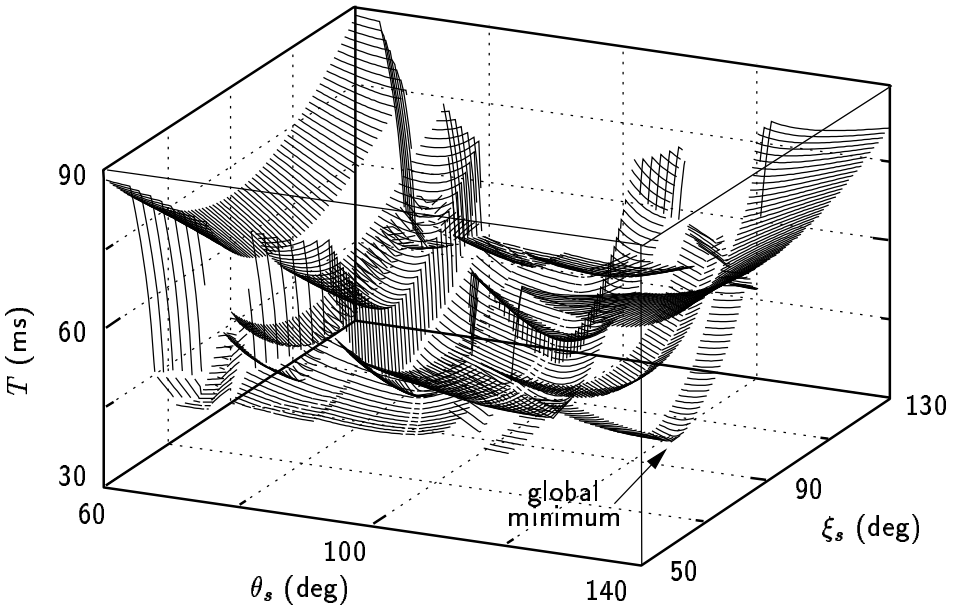

Figure 5.11: Model 1: traveltime as a function of both take-off angles. Note the complex topography. The surface is explored during the SA process to locate the global minimum.

to be a natural tool for solving this kind of nonlinear optimization problems. Convergence was achieved long before a maximum of 500 iterations, as illustrated in Figure 5.12 for a typical realization. Note that both take-off angles, θ_s and ξ_s, were involved in the actual optimization. Figure 5.13 shows a scatter plot for the 500 annealing iterations.

5.5.2 Salt-dome model

Model 2 represents a salt dome with several layers and laterally varying velocities (Figure 5.14). A total of eight regions are delimited by nonplanar interfaces with cylindrical symmetry. The same symmetry, however, is not observed for all the velocities. The model is contained in the cube delimited by planes $x = -50$, $x = 50$, $y = -50$, $y = 50$, $z = 0$ and $z = 40$, though the figure shows only a portion of the full model. For simplicity consider first a vertical slice of the original model (the plane $x = 0$) and $\xi = 90^o$ for all

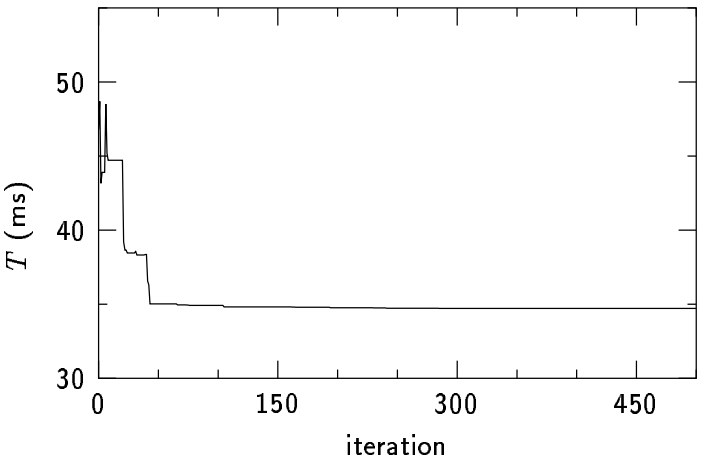

Figure 5.12: Model 1: SART convergence after 500 iterations.

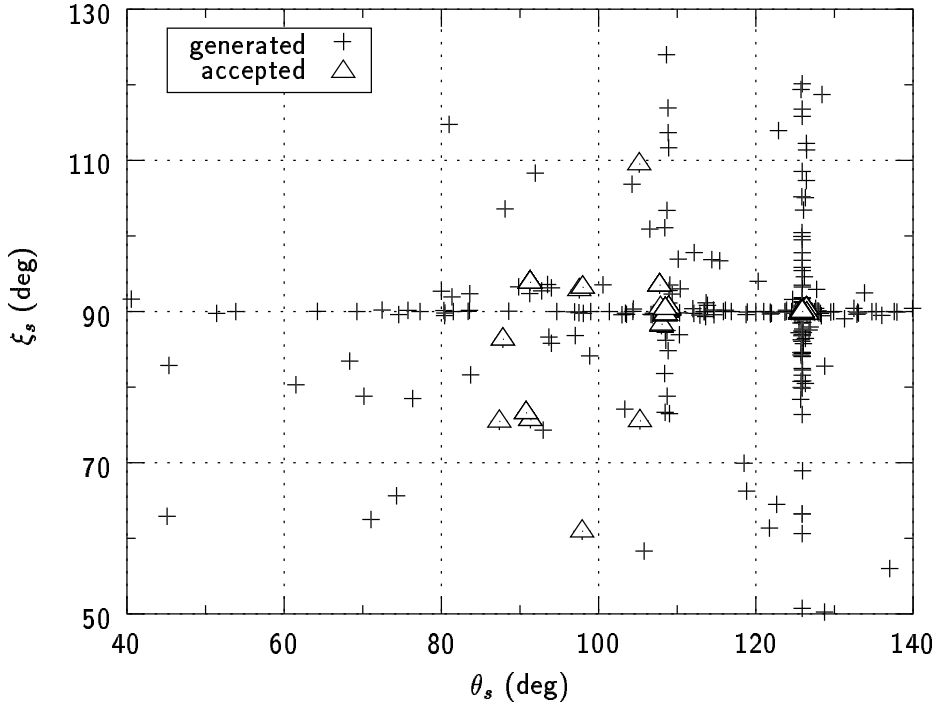

Figure 5.13: Model 1: scatter plot for 500 annealing iterations.

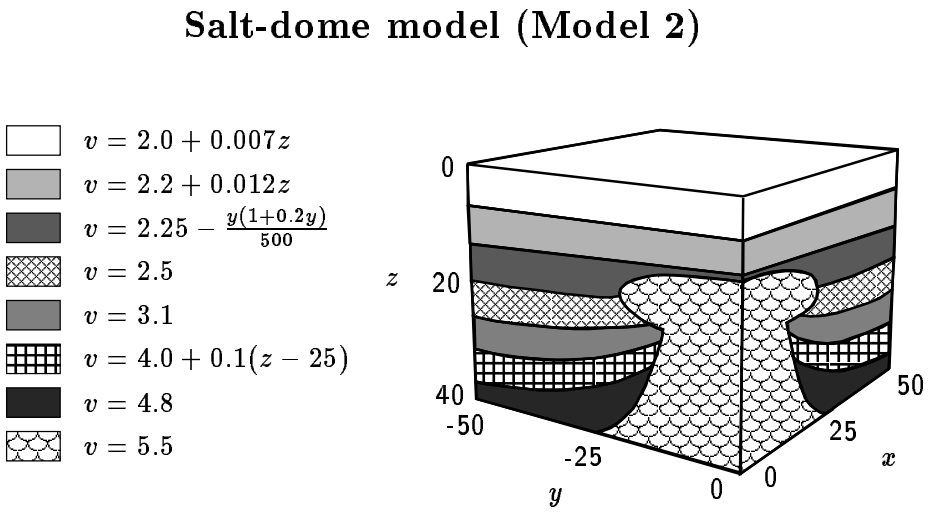

Figure 5.14: Three-dimensional model with several nonplanar interfaces and eight regions with constant and laterally varying velocities. Note the cartoon shows only a quarter of the full model.

raypaths, so that all of them will lie on the same plane (see Figure 5.15 and Table 5.3).

I located a source at $(0, -50, 0)$ and produced a fan of rays with equally spaced take-off angles in the range $(35^o, 85^o)$. Figures 5.16a and 5.16b show d and T, as a function of θ_s, for a receiver located at $(0, 50, 0)$. The complexity of these two functions is enormous. Function d exhibits eight zeros corresponding to eight rays arriving to the receiver satisfying the ray equations. Besides there are more than ten extra local minima and a high number of first- and second-order discontinuities. Figure 5.17 (top panel) shows these eight raypaths, and Table 5.4 summarizes their traveltime and take-off angles.

By inspecting the curves in Figure 5.16, the difficulties that a local minimizer would have to face to find the global minimum are very clear (see section 5.6.3). Even using a global optimization algorithm like VFSA, function d is not the appropriate cost function to choose. This is because it is impossible to differentiate among the various raypaths

Region #	$v = v(\mathbf{x})$ (km/s)	Interface #	$z = f(r^2)$ (m)
I	$2.0 + 0.007z$	i	$20/3 - 0.5/(1 + r^2)$
II	$2.2 + 0.012z$	ii	$40/3 - 2.5/(1 + r^2)$
III	$2.25 - 0.002y\,(1 + 0.2y)$	iii	$60/3 - 5.0/(1 + r^2)$
IV	2.5	iv	$80/3 - 10.0/(1 + r^2)$
V	3.1	v	$100/3 - 15.0/(1 + r^2)$
VI	$4.0 + 0.1\,(z - 25)$	vi	$120/3 - 20.0/(1 + r^2)$
VII	4.8	vii	$12.5 + 5.0r^4$
VIII	5.5	viii	$22.5 - 5.0r^4$
–	–	ix	$140/3 - 10.0/(0.01 + r^2)$

Table 5.3: Model 2: velocities and interfaces defining the salt-dome model shown in Figure 5.15. Interfaces are expressed in terms of $r^2 = (x^2 + y^2)/400$.

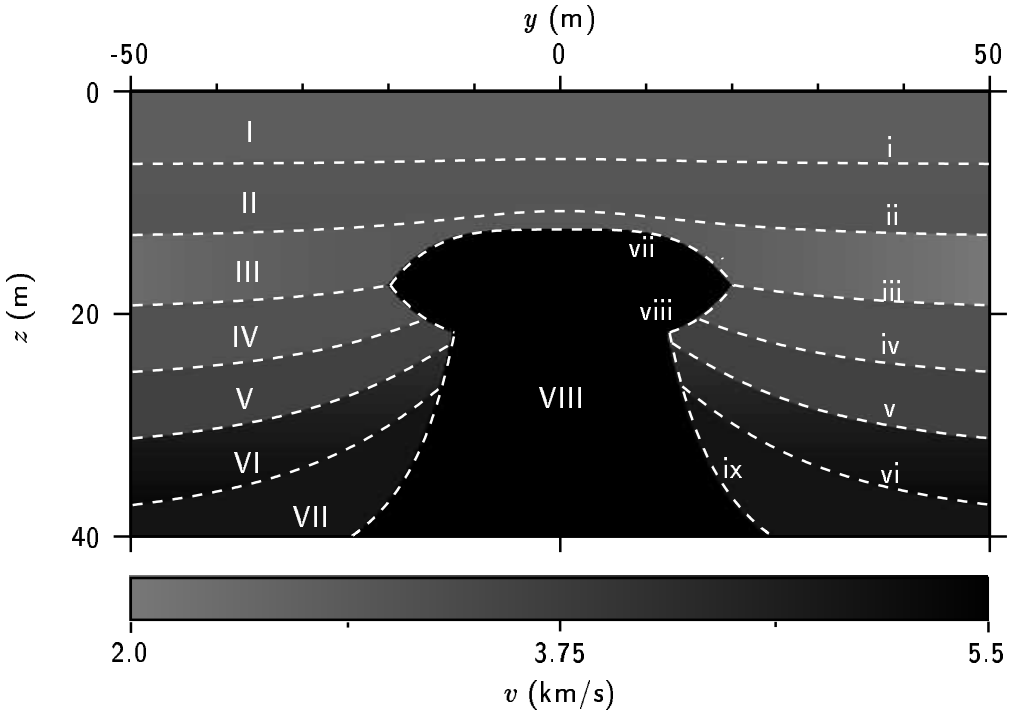

Figure 5.15: Model 2: two-dimensional slice through the salt-dome model in Figure 5.14. Labels refer to the regions and interfaces listed in Table 5.3.

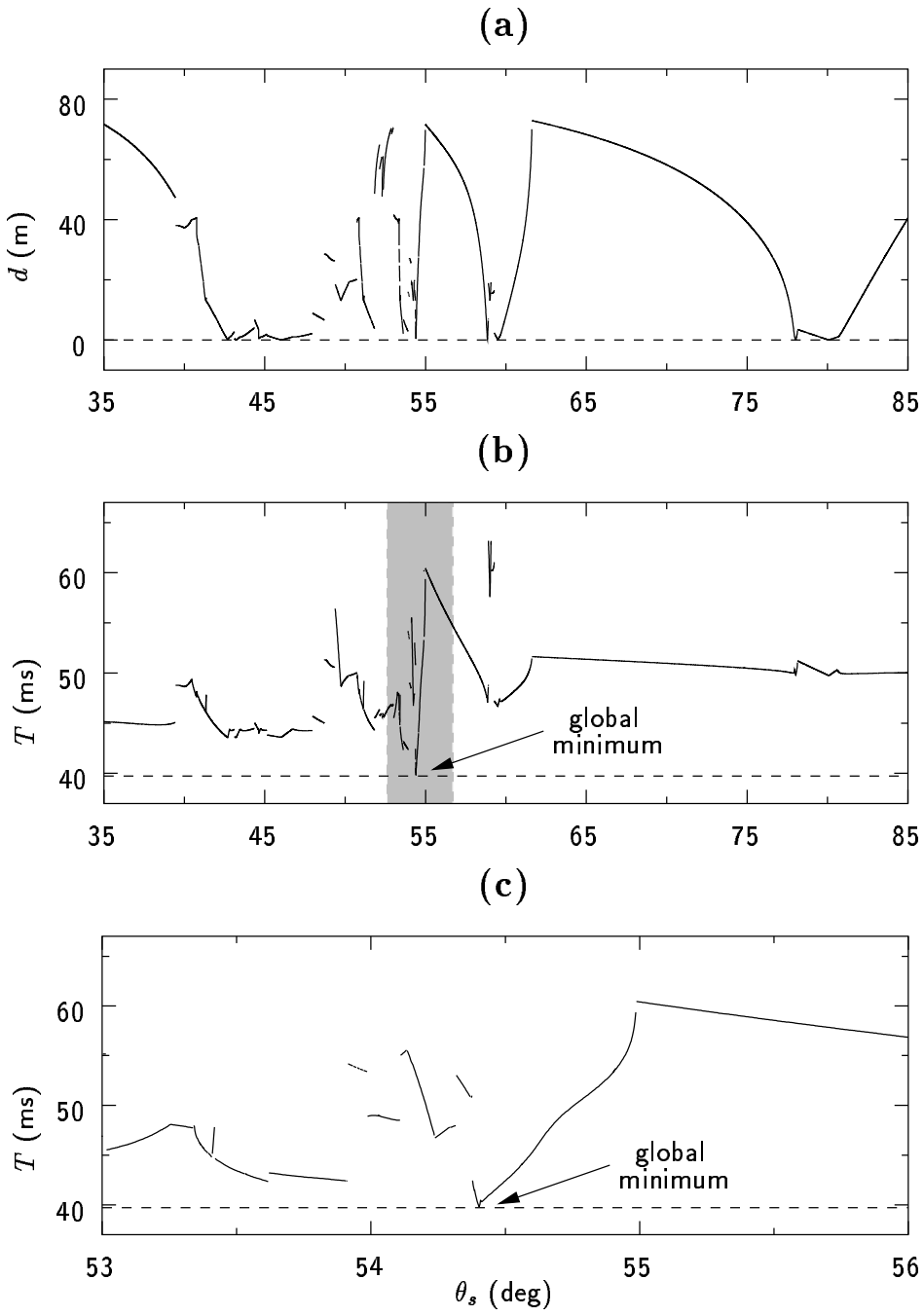

Figure 5.16: Model 2: two cost functions for the salt-dome model. (a) Distance d, and (b) traveltime T, as a function of take-off angle θ_s. The shaded region in (b) is expanded and shown in (c). Azimuth ξ_s has been fixed to 90°.

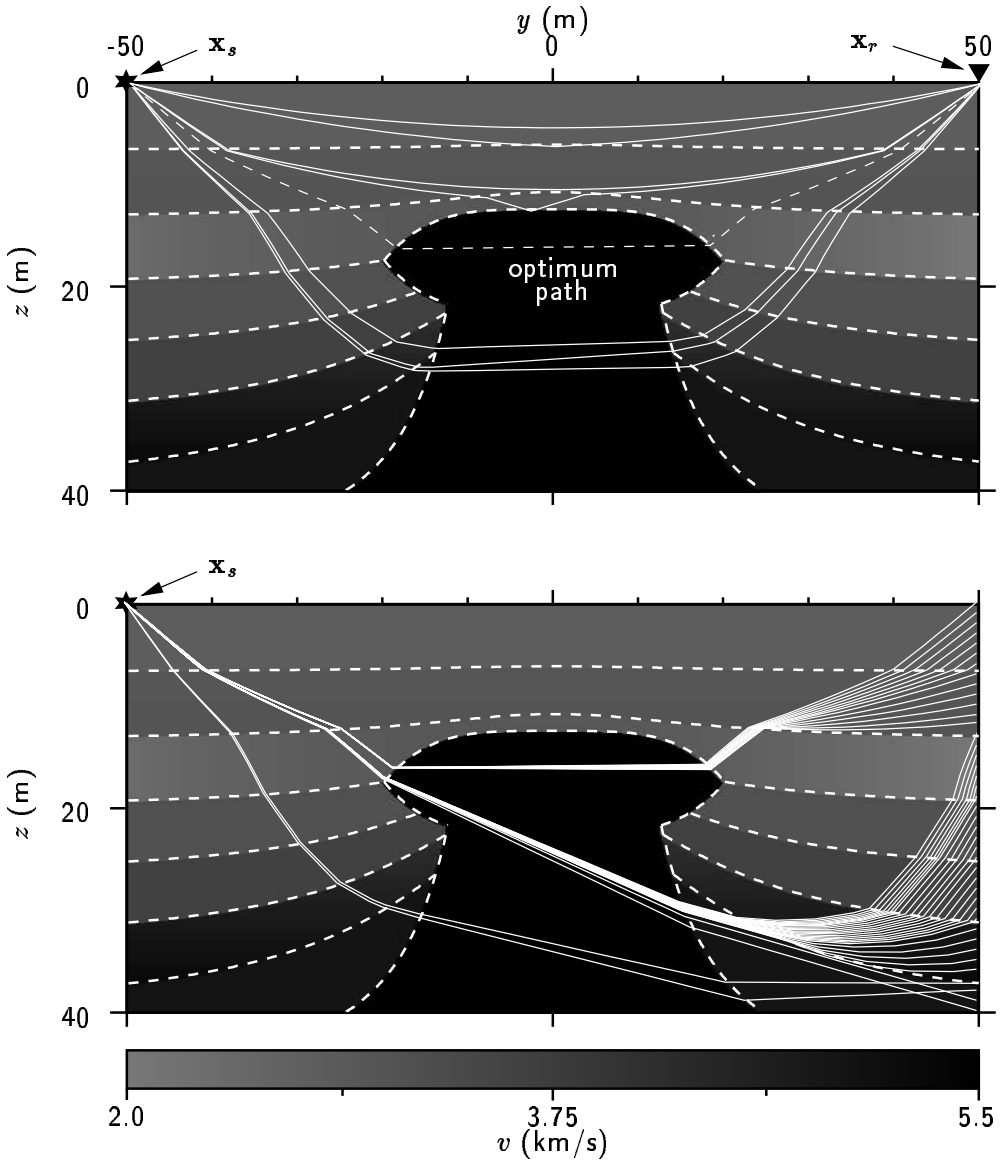

Figure 5.17: Model 2: multipathing in the salt-dome model. Top panel: the eight plotted raypaths satisfy the ray equations, but exhibit different traveltimes. The dotted raypath is the SART solution. Bottom panel: a total of 41 receivers are connected with the same source following the path with global minimum traveltime.

Ray #	θ_s (deg)	Φ_p (ms)
1	42.70	43.551
2	43.22	43.570
3	46.03	43.589
4	**54.40**	**39.735**
5	58.85	47.010
6	59.50	46.652
7	78.02	49.781
8	80.07	49.748

Table 5.4: Model 2: multipathing in the salt-dome model. The raypath number 4 exhibits the absolute minimum traveltime.

that connect source and receiver. On the contrary, the global minimum of T corresponds to the desired raypath. A detailed inspection of curve $T(\theta_s)$, reveals that the global minimum lies in an very narrow valley of about 0.1 degrees wide, what makes the optimization problem even harder (see Figure 5.16c for a detailed plot of T in the global minimum neighborhood). If take-off angle ξ_s is also taken into account, the topography of $T(\theta_s, \xi_s)$ becomes so complex I was not able to plot it at all. Despite the rather difficult optimization problem, SART found the true solution in about 500 iterations, as shown in Figure 5.18 for a typical realization:

$$\theta_s = 54.40^o, \qquad \xi_s = 90.00^o \qquad T_{opt} = 39.735 \text{ ms.}$$

Finally, Figure 5.19 shows a scatter plot for the 500 annealing iterations.

The behavior of the traveltime curves corresponding to other receivers is also very complicated. I used a grid-search algorithm to find all the raypaths connecting the same source and 41 different receivers uniformly distributed along the vertical line $x = 0$, $y = 50$:

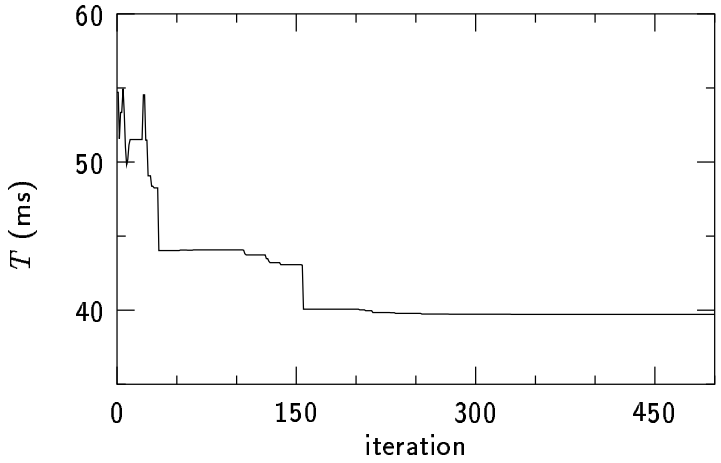

Figure 5.18: Model 2: SART convergence after 500 iterations.

$$\mathbf{x}_{ri} = [0, 50, i \times h], \qquad i = 0, 1, \cdots, 40 \tag{5.38}$$

where $h = 1.0$ m is the geophone vertical spacing. The resulting traveltimes are shown in Figure 5.20a, and the corresponding common-shot gather is shown in Figure 5.20b (the shot gather was constructed as in the fault model case: a zero-phase Ricker wavelet convolved with a reflectivity series containing isolated unit spikes at the arrival times). It can be seen that there are multiple arrivals at all geophone depths. All arrivals follow very different trajectories, but those whose traveltimes are a global minimum, basically share the same first portion until they traverse the salt-dome structure. These raypaths are shown in Figure 5.17 (bottom panel). Note that most of the solutions are within one of two very narrow take-off angle ranges. Raypaths connecting geophones at depths between 0 and 12 m, have take-off angles in the range $(54.376, 54.404)$. Raypaths connecting geophones at depths between 13 and 36 m, have take-off angles in the ranges $(53.499, 53.564)$. These raypaths were very difficult (expensive) to locate using the grid-search algorithm, even though ξ_s was known a priori. In next section a quantitative

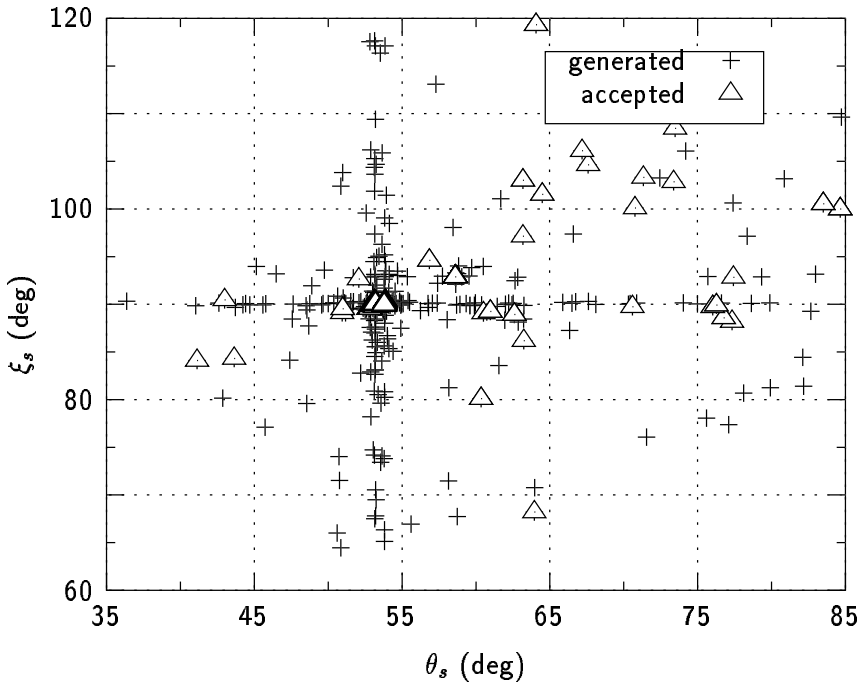

Figure 5.19: Model 2: scatter plot for 500 annealing iterations.

comparison, in terms of computational cost, between SART and the grid-search method is given.

5.5.3 Selection of SART parameters

Contrarily to standard linearizing methods, VFSA is not a black-box tool for solving any nonlinear optimization problem. The user must obtain, usually through experimentation, the parameters that produce the optimum results, both in terms of accuracy and efficiency. In Chapter 2 it has already been discussed how to select the initial temperatures for the annealing process. Parameter temperatures are set equal to 1 in all cases, so that the model space is widely sampled at the initial stages. The acceptance temperature is set equal to the mean cost after evaluating the total traveltime for a set of NT0 arbitrary

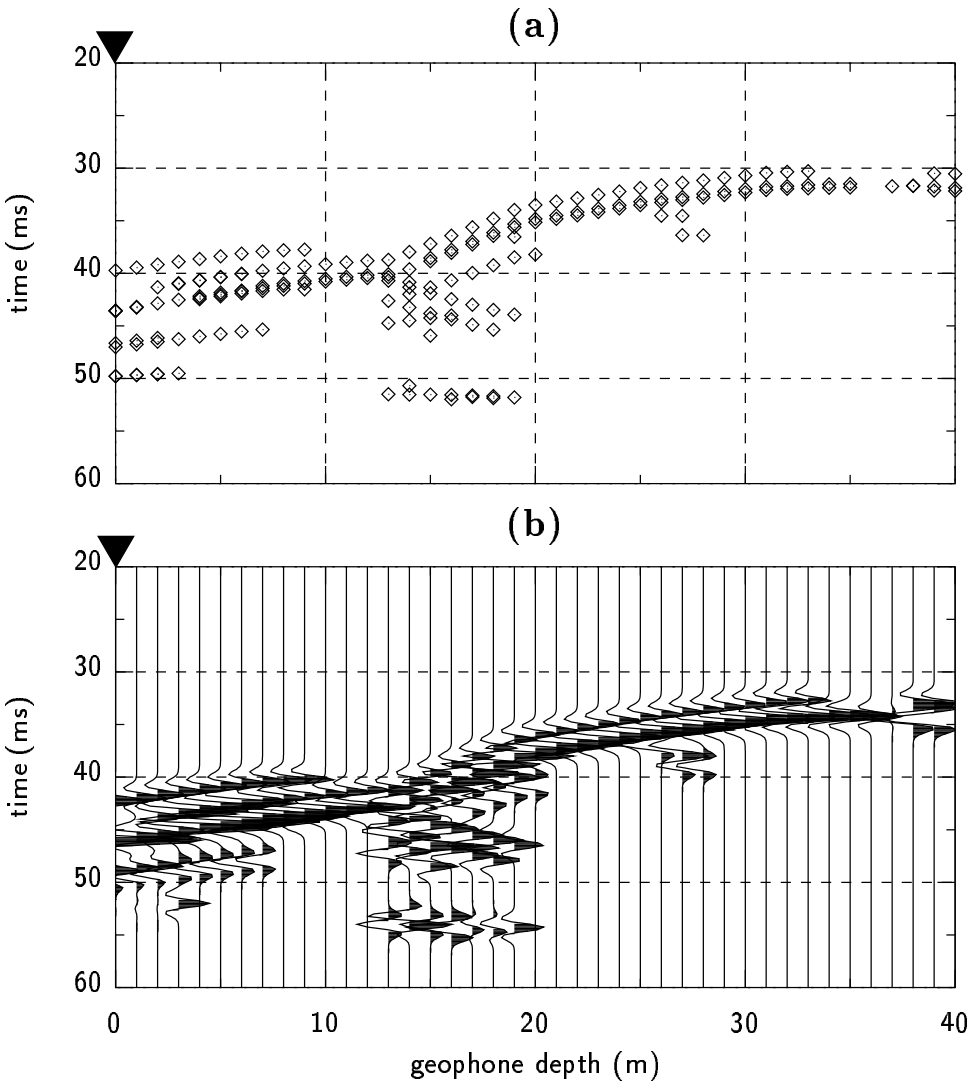

Figure 5.20: Model 2: (a) Traveltime vs geophone depth, and (b) common-shot section, for a set of 41 source-receiver pairs. The source is fixed at $(0, -50, 0)$. Amplitudes in the shot gather are arbitrary.

	Model 1		Model 2	
	A (deg)	B (deg)	A (deg)	B (deg)
θ_s	60.0	140.0	35.0	85.0
ξ_s	50.0	130.0	60.0	120.0

Table 5.5: Parameter search ranges in VFSA.

take-off angles randomly selected. That is

$$\begin{cases} T_0 = \bar{T} = \frac{1}{\mathrm{NT0}} \sum_j T_j, \\ T_{0i} = 1, \qquad i = 1, 2 \end{cases} \tag{5.39}$$

By default I set NT0=10. Other user-defined parameters to be taken into account are the search ranges for each unknown. These are summarized in Table 5.5 for both Model 1 and Model 2. These are the values I selected based on the location of the source and receiver, and on the geometry of the models.

The maximum number of iterations, ITMAX, is one of the most important parameters. It is very difficult to say a priori how many iterations it will take to find the optimum path for a given source-receiver geometry and a given subsurface model. It can be said, however, that the fewer the number of local minima and the smoother the cost function, the fewer the number of iterations would be required for convergence to the global minimum. Take a look at the convergence curves shown in Figures 5.12 and 5.18. In the first case, convergence is achieved very soon since Model 1 is relatively simple as compared to Model 2. The complexity associated with Model 2 is enormous and the global minimum is very difficult to locate, as shown in previous section. Despite this fact, convergence was obtained quite fast. However, the multimodality of traveltime $T(\theta_s, \xi_s)$ sometimes delays convergence, especially when local and global minimum traveltimes are

Model	tolerance	misfit (ms)	mean	std. dev.	in band	$\bar{T}_{er}$ (ms)
1	2.0 %	35.408	168	145	93 %	0.536
	1.0 %	35.061	208	112	92 %	0.287
	0.1 %	34.749	437	136	90 %	0.030
2	2.0 %	40.529	770	552	95 %	0.978
	1.0 %	40.132	887	606	91 %	0.268
	0.1 %	39.774	1236	395	90 %	0.014

Table 5.6: Mean and standard deviation of the number of SA iterations required to achieve a specified misfit after 200 realizations. Global minimum traveltimes for Model 1 and Model 2 are 34.714 ms, and 39.735 ms, respectively.

very similar (this is the case of Model 1), or when the global minimum is in a very small region of the model space (this is the case of Model 2). This is illustrated in Figure 5.21, where I have plotted the convergence curves for 20 independent realizations. Note the various "plateaus" in the convergence curves corresponding to local minima.

To provide a better insight about the number of iterations that are required to reach the global minimum, the following experiment was conducted. SART was re-run 200 times using different seeds, for both Model 1 and Model 2, keeping the same source-receiver geometries. Since the global minimum in each example is known, it was possible to obtain the mean number of iterations required to reach the global minimum, within a certain accuracy. Of course, in this experiment, the linearizing stage was not applied, and SART stopped after achieving the desired misfit. The results of the computations are shown in Table 5.6. The misfit shown in column 3 is defined as

$$\text{misfit} = T_{opt} \times (1 + \text{tolerance}), \tag{5.40}$$

where T_{opt} is the global minimum traveltime, and "tolerance" is tabulated in column

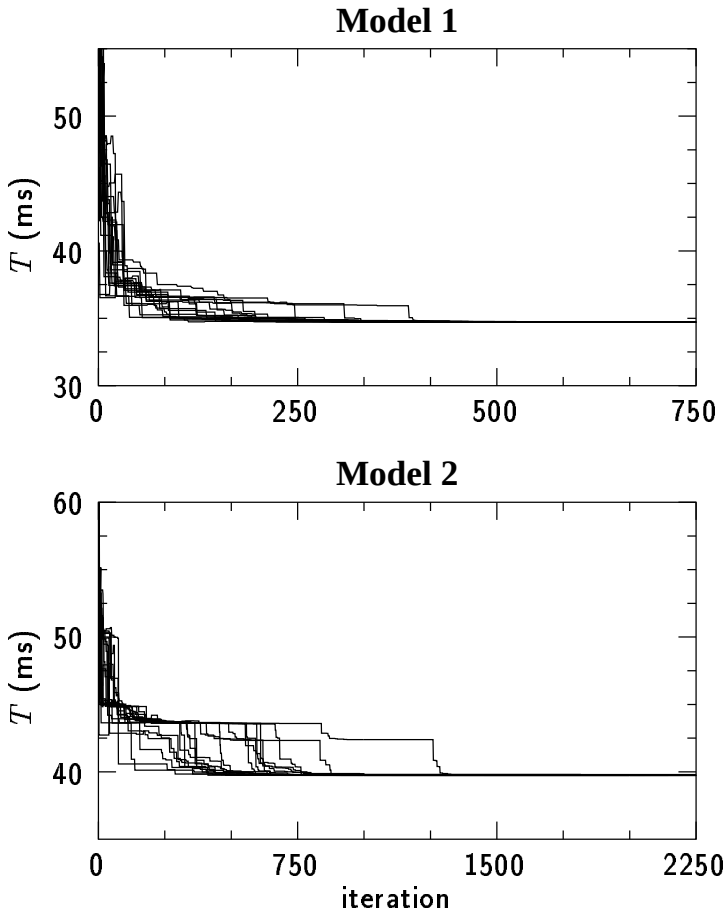

Figure 5.21: SART convergence for 20 independent realizations. Note the plateaus corresponding to local minima.

2. As expected, the mean number of iterations increases with the desired accuracy. In the salt-dome case (Model 2), this number is much larger than in the fault model case (Model 1), for the associated cost function is much more complicated. Figure 5.22 shows the dispersion plots of the number of iterations for each example. It can be seen that most of the realizations (above 90%, see column 6 in Table 5.6) fall within the horizontal bands. These bands are centered at the mean number of iterations, and their widths equal two times the standard deviation.

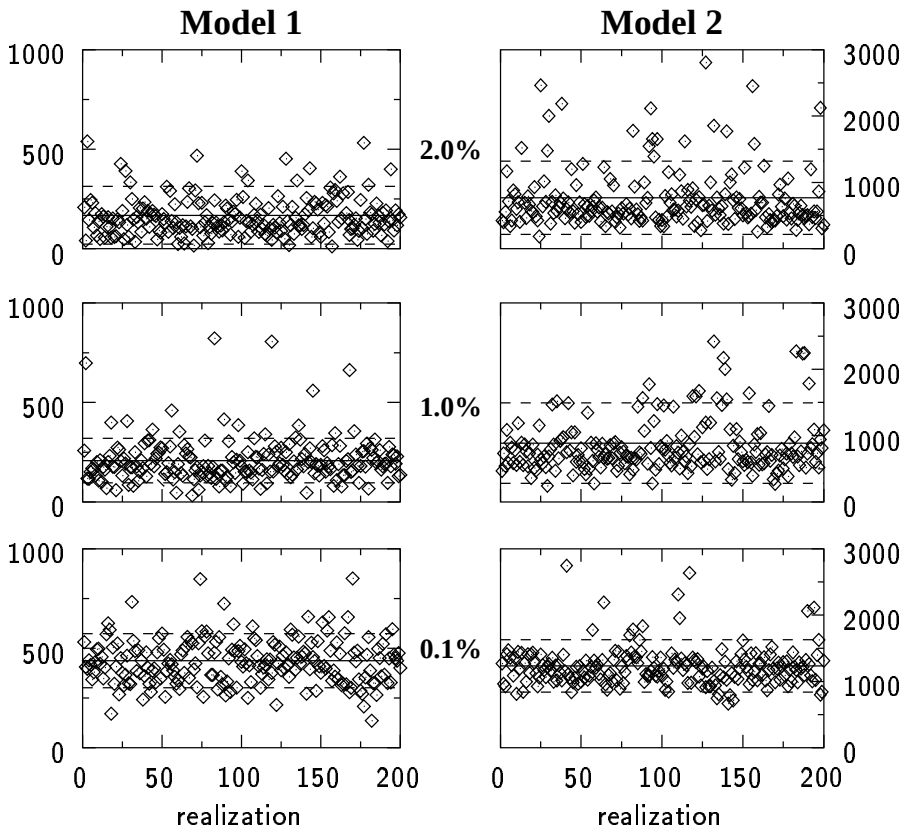

Figure 5.22: Dispersion plots of the number of iterations required to reach the global minimum within a given accuracy. See text for explanation.

As a result of this analysis, it can be said that setting ITMAX=1000 for Model 1, and ITMAX=3000 for Model 2, is enough to obtain the global minimum with a very high confidence. Actually, above 90% of the realizations converged using much smaller values. Most of the outliers that are present in the dispersion plots (number of iterations well above the mean plus one standard deviation), correspond to SA runs where the solution stacked in difficult local minima for many iterations before being able to escape (the plateaus in Figure 5.21). In Model 1, for example, the difficulty arises from the fact that two local minima are within only 4% of the global minimum (see Table 5.2). In Model 2, the global minimum is located in a very narrow valley, while various local minima are

situated in a wide valley with low traveltimes (see Figure 5.16b and c). For the sake of completeness, the last column in Table 5.6 shows the mean traveltime corresponding to the straight-ray construction.

Finally, there are a few other tunable parameters to be set in SART. Such is the case of

1. T_{tol}: tolerance traveltime. If $T_{er} > T_{tol}$ after ITMAX iterations, the SA run is repeated using a different seed. Default value: 1.0 ms.

2. MAXREPEAT: maximum number of times the SA run is repeated before giving up. Default value: 3.

3. ITMAX_LIN: maximum number of iterations for the linearizing stage. Default value: 15.

4. ϵ_d: tolerance distance for the linearizing stage. Default value: 1×10^{-5} m.

The first two parameters, together with ITMAX, are related to the SA optimization stage. The remaining two parameters concern exclusively to the linearizing stage.

5.6 SART performance: analysis and discussion

In this section I shall discuss some important aspects concerning the efficiency of various SART subprocesses, and methods to further improve its overall performance. I will also make a comparison with a standard grid-search algorithm, and with a multistart linearizing procedure. These two methodologies may be viewed as particular cases of shooting techniques.

5.6.1 SART subprocesses efficiency

It is a commonly known fact that SA-based algorithms for the optimization of nonlinear problems are more expensive, in terms of computational cost, than gradient-based methods. SART is not an exception to the rule, even though it uses VFSA. The price to be paid in order to provide a reasonable certainty on finding the global minimum is given by the number of iterations required for convergence, as shown in previous section (see Table 5.6).

To visualize the efficiency of various SART subprocesses under different conditions, the ray-tracing method was applied to both Model 1 (Figure 5.1) and Model 2 (Figure 5.14) using different settings. In the first case, a source was located at $(5, 5, 70)$ and 25 receivers were distributed across the surface, $z = 0$. In the second case, the source was put right below the salt dome at $(-25, 0, 35)$, and 25 receivers distributed across the surface, too. The resulting 3-D raypaths are plotted in Figures 5.23 and 5.24 respectively. SART was repeated using two different constant timesteps in the RK integration. The FORTRAN codes were compiled using the "-pg" option (profile-debugging mode) so that the amount of time spent in each subroutine could be extracted from the resulting executable code. Table 5.7 summarizes some of the profile values obtained in that way[1]. The tabulated values in each row have the following meaning:

1. TT: total computational time (in seconds) required to obtain the 25 optimum raypaths.

2. *rays/s:* number of traced rays per second (i.e. $25/TT$).

3. $\bar{k}$: mean raypath length (in points).

4. $COST$: percentage of TT spent at evaluating the cost function.

[1]The results were obtained on a 90MHz PC-Pentium, running under Linux with f2c.

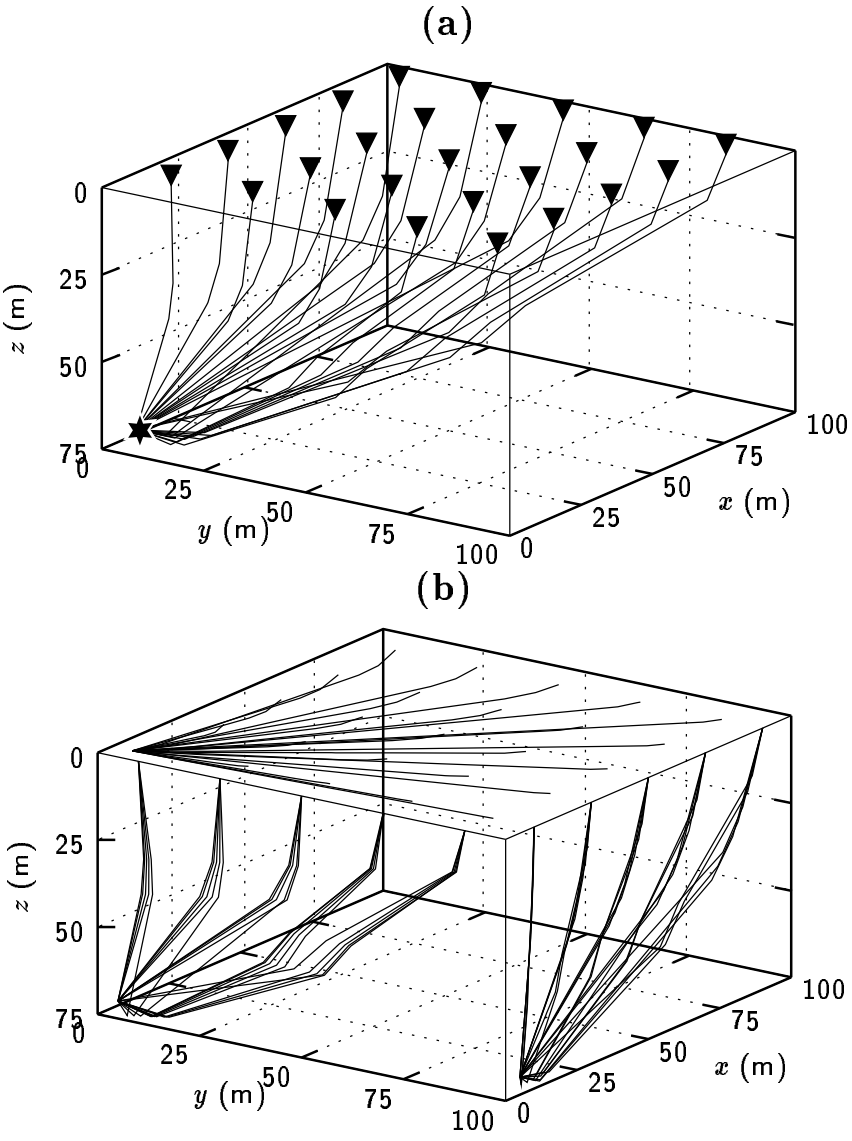

Figure 5.23: Model 1: multiple ray tracing in the fault model. (a) 25 optimum raypaths obtained by SART. The source is located at $(5, 5, 70)$. (b) Raypath projections onto various planes.

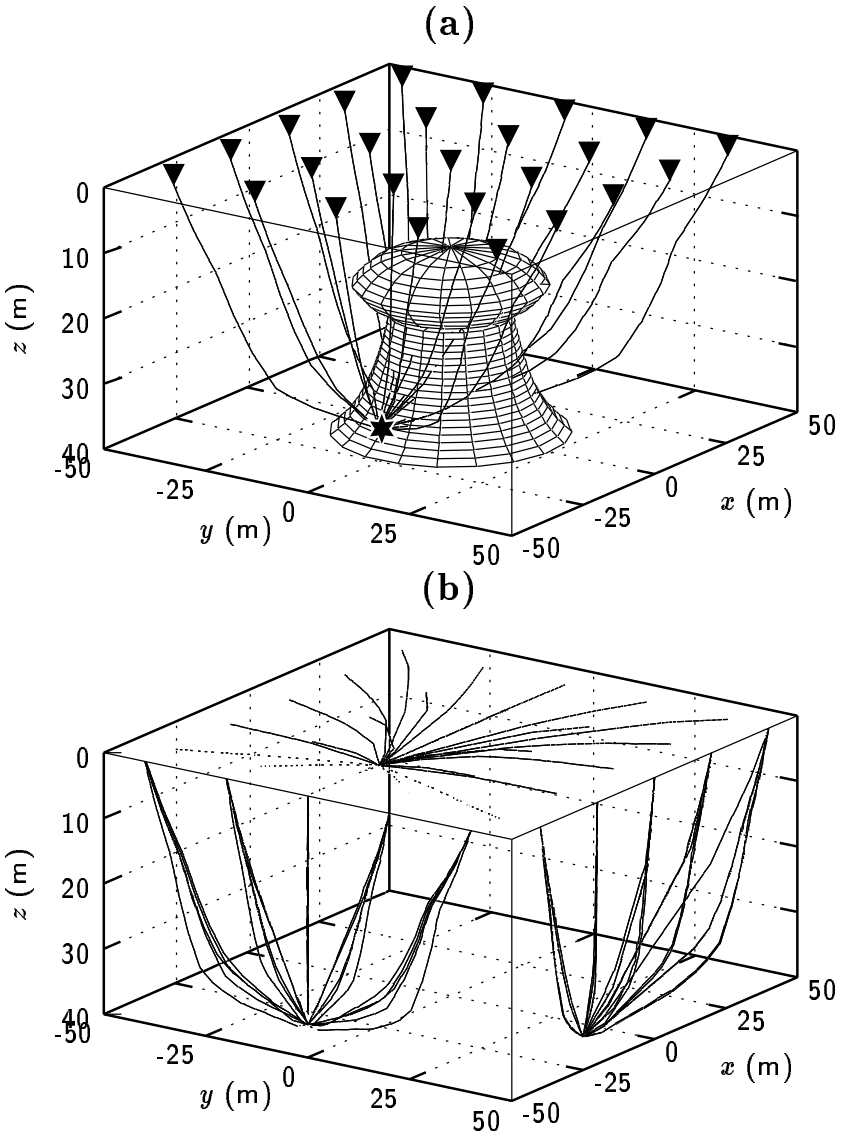

Figure 5.24: Model 2: multiple ray tracing in the salt-dome model. (a) 25 optimum raypaths obtained by SART. The source is located at $(5, 5, 70)$. (b) Raypath projections onto various planes.

5. INT: percentage of TT spent by the numerical integration of the ray equations.

6. $CROSS$: percentage of TT for checking for interface crossings, finding intersections and evaluating Snell Law.

7. $STRGHT$: percentage of TT spent in the straight-ray construction subroutines

8. LIN: percentage of TT spent in the linearizing optimization stage (the method of steepest descent for polishing the SA solution).

9. TT_{Euler}/TT_{RK4}: ratio of TT using Euler and fourth-order RK methods for integrating the ray equations.

10. TT_{RK2}/TT_{RK4}: ratio of TT using second- and fourth-order RK methods for integrating the ray equations.

In SART, each iteration involves the solution of the ray equations for a given set of initial conditions. Traveltime T_{er} along the straight-ray construction needs to be computed, too. At each timestep, the algorithm also checks whether the ray has crossed any model interface. Usually, this procedure, along with the calculation of the corresponding intersection points, is a very time-consuming stage, since all or some interfaces must be evaluated at every point of the ray trajectory. When Δt is small and/or the number of interfaces is large, this process may account for the major part of the total computational time spent in each iteration. All these procedures are involved in the evaluation of the cost function at each iteration. This is by far the most time consuming process of SART, accounting for about 95-99% of the total computational cost, as shown in Table 5.7 (row labeled $COST$).

It is interesting to see the percentage of TT spent by some of the subprocesses involved in the evaluation of the cost function, according to the different models and stepsizes.

	Model 1		Model 2	
Δt	1.0	10.0	0.5	3.0
TT	43.2	18.1	65.6	43.6
rays/s	0.6	1.4	0.4	0.6
$\bar{k}$	43.4	7.4	42.3	11.1
$COST$	98.1	95.3	98.9	98.4
INT	64.7	48.5	50.7	51.1
$CROSS$	18.7	33.7	32.0	52.0
$STRGHT$	10.9	19.7	22.6	22.0
LIN	3.4	2.5	2.0	2.0
TT_{Euler}/TT	0.51	0.64	0.62	0.62
TT_{RK2}/TT	0.68	0.76	0.75	0.74

Table 5.7: SART profile using fourth-order RK integration. The first row shows the total computational time, TT, in seconds. Rows 4 to 8 show the percentage of TT spent in various subprocesses. See text for explanation.

For example, take a look at row INT in the table. This subprocess is the most expensive one in most cases (above 50% of TT), especially for Δt small. However, when the number of interfaces is large (as is the case of Model 2), $CROSS$ starts getting relevant and even exceeds INT in terms of computational effort when Δt is large (52.0% against 51.1% in the last column of the table). This is because the number of iterations performed by $CROSS$ at each interface crossing is almost independent of the integration stepsize.

The profile also yields other interesting results. The straight-ray construction subprocess, $STRGHT$, which involves the calculation of the traveltime between the emerging point and the receiver, accounts for about 10-20% of TT. Finally, subprocess LIN takes only about 2-3% of TT, what is a small value as compared with the other subprocesses. As a conclusion of this analysis, it can be said that the number of interfaces, and their

complexity, plays a key role in the performance of SART. For models with no interfaces, the overall computational time would be significantly reduced. An important reduction in the overall computational time may be obtained by using faster numerical methods at no expense of accuracy when velocities are smooth functions. This point will be discussed in detail below.

5.6.2 Further improvement of SART efficiency

Lower-order integration

A significant overall improvement of SART performance can be obtained if the computational cost is reduced in the most time-consuming subprocess, namely INT. This can be accomplished using a lower-order numerical integration algorithm, especially when velocity fields within each region are very smooth functions. For example, the step's error in the simple Euler method is $O(\Delta t^2)$, which is zero when the velocity is constant. Euler method (see for example [PTVF92]) requires a single evaluation per step of the right-hand side of equation (5.2), whilst fourth-order RK requires four. Second-order RK can also be used. It requires two evaluations per step of the derivatives, and the associated error is $O(\Delta t^3)$. As a result, a significant overall reduction in efficiency can be expected. Appendix C provides a formulae to estimate the ratio between the total CPU time after using two different numerical integrators. For instance, the saving in computational cost is reduced by a factor of two if Euler method is used instead of 4th-order RK in Model 1 with $\Delta t = 1.0$ s (see last two rows in Table 5.7 for estimated ratios).

Static versus *dynamic* integration

In SART, the integration of the ray equations has two options: *static* integration and *dynamic* integration. In the first case, the user must supply which method to use throughout. In the second case, each model region is given a flag number indicating which method must be used for that particular region. In this manner, every time the ray enters a new region, the appropriate method is automatically selected for optimum performance according to the velocity field smoothness (e.g. in the salt-dome model, use Euler for regions IV, V, VII, and VIII; second-order RK for regions I, II and VI; and third-order RK for region III). The same static/dynamic concept applies to the straight-ray construction[2].

Table 5.8 shows the profile values obtained after using Euler method (first four columns) instead of fourth-order RK (Table 5.7), and using dynamic integration (fifth and sixth columns) for the salt-dome model. The results show that the improvement in the overall performance estimated using formula (C.2) is a fairly good approximation to the actual values. In effect, last row of Table 5.8 shows the ratios of the total computational times using Euler and fourth-order RK respectively. These values are to be compared to the estimations shown in Table 5.7 (row labeled TT_{Euler}/TT). Notice also how subprocess INT now surrenders its protagonist role to subprocess $CROSS$ in all the runs. It becomes the most time-consuming subprocess, now, even for Model 1 that has only six planar interfaces. This suggests that SART efficiency will be significantly greater for models with no interfaces or just a few.

Collecting previous cost function evaluations

Whenever more than one BVP is to be solved for a given experiment, it seems reasonable to take advantage of previously computed cost functions to improve next SA runs. This

[2]Since at convergence T_{er} goes to zero, the accuracy at this stage is not very important and Euler method can be used in all cases. Nevertheless, the user can select any method at this stage.

	Model 1 (static)		Model 2 (static)		Model 2 (dynamic)	
Δt	1.0	10.0	0.5	3.0	0.5	3.0
TT	20.6	10.1	42.2	23.2	44.9	23.6
rays/s	1.2	2.5	0.6	1.1	0.6	1.1
$COST$	96.5	92.3	97.4	97.0	98.1	96.5
INT	23.1	15.9	15.3	16.1	14.6	15.2
$CROSS$	41.2	39.3	39.4	52.6	40.8	53.2
$STRGHT$	19.9	20.0	34.5	27.2	33.8	27.4
LIN	3.2	2.8	1.6	2.1	1.6	2.0
TT/TT_{RK4}	0.48	0.56	0.64	0.53	-	-

Table 5.8: SART profile using *static* integration (Euler method in this case) and *dynamic* integration.

is particularly useful for source-receiver pairs sharing the same source point. At each cost function call, not only is the cost function associated with the source-receiver at hand evaluated, but also the cost functions associated with all remaining source-receiver pairs. In turn, the optimum solution to each pair is updated accordingly. When the next pair is considered, the computations already done for all the previous pairs are taken into account. This procedure allows one to require fewer iterations to reach the global minimum than running SA for each pair independently. In the examples below, the maximum number of iterations for the last source-receiver pair is set, by default, equal to half the maximum number of iterations for the first source-receiver pair. For the other receivers, ITMAX varies linearly between these two values. Since the most time consuming process is associated with the first portion of the raypath (from the source, $\mathbf{x}_s$, to the emerging point $\mathbf{x}_e$), for a moderate number of receivers with common shot, in general this strategy leads to a significant overall speedup, though the evaluation of those

extra cost functions contributes to increase the computational cost of each individual run.

5.6.3 Comparison with other ray-tracing methods

In this section I will make a comparison, in terms of computational cost, among SART, a simple grid-search algorithm, and a multistart linearizing procedure. The grid-search method consists of performing an exhaustive search on a regular grid that spans the take-off angles ranges, and evaluating the cost function at each point. The lowest value corresponding to each source-receiver pair is then selected, and the steepest descent (SD) method is finally used to refine the solution. In this way, the global minimum can be located provided the discretization of the take-off angles is dense enough. This method will be called GRID. The multistart procedure, MULTI, simply applies the SD method starting from different initial take-off angles selected at random from the given search ranges. The optimum take-off angles are then chosen as those that produce the lowest cost function values. For the comparison, SART is run in two flavors. SART1 solves the optimization problem for each source-receiver pair independently. SART2, collects information corresponding to previous cost function evaluations and uses this information to improve the current solution, as explained above.

In the experiments, both Model 1 and Model 2 shown in previous sections are used. Several source-receiver pairs are defined in each case, and the optimum take-off angles are searched using SART, GRID, and MULTI, using various ITMAX, grid densities, and number of restarting points, respectively. The comparison is done for 2-D and 3-D experiments. In the first case, only take-off angle θ_s is searched, while ξ_s is fixed and known a priori. Here, the acquisition geometry is that used in Figures 5.10b and 5.17b. That is, all geophones uniformly distributed along the right-hand side borehole. In the 3-D case, both θ_s and ξ_s are searched. Here, I use two acquisition geometries: (1) the one used in the 2-D case (i.e. aligned geophones along right well), and (2) the one depicted

	2-D			3-D, case 1			3-D, case 2		
Method	PAR	ray/s	1.0 %	PAR	ray/s	1.0 %	PAR	ray/s	1.0 %
SART1	100	22.90	94.4	100	16.58	82.4	100	17.86	76.8
	250	11.26	97.3	250	8.72	91.2	250	10.42	82.4
	500	5.35	99.1	500	4.96	91.8	500	5.81	91.2
	1000	2.80	99.7	1000	2.55	96.5	1000	2.94	98.4
SART2	100	27.20	99.1	100	20.61	94.4	100	20.83	87.2
	250	12.83	100.0	250	9.58	96.2	250	10.05	90.4
	500	7.08	100.0	500	5.96	97.6	500	5.21	96.8
	1000	3.76	100.0	1000	3.24	99.4	1000	2.75	99.2
GRID	1.0	343.43	94.4	0.9	20.73	95.6	1.7	20.11	88.0
	0.5	300.88	96.5	0.6	10.97	98.2	1.0	7.14	89.6
	0.2	241.13	99.4	0.4	4.89	98.7	0.4	4.55	94.4
	0.1	162.68	100.0	0.3	2.75	99.2	0.6	2.63	96.0
MULTI	10	23.73	97.9	10	21.94	86.8	15	14.71	87.2
	20	11.14	99.1	25	8.72	92.9	25	9.26	94.4
	40	5.60	100.0	40	4.86	97.9	60	4.10	99.2
	80	2.80	100.0	70	2.98	98.5	100	2.63	100.0

Table 5.9: Model 1: comparison of SART with other methods in the fault model. PAR equals ITMAX, $\Delta\theta_s$ (and $\Delta\xi_s$), and number of starting points for SART, GRID, and MULTI, respectively.

in Figures 5.23a and 5.24a (i.e. geophones distributed across the surface plane).

I run SART1 and SART2 with various ITMAX and measured the total CPU time required to find all solutions[3]. I also measured the CPU time required by GRID for various $\Delta\theta_s$ (and $\Delta\xi_s$), as well as the CPU time required by MULTI with different number of starting points (ITMAX_LIN=5 by default). For statistical purposes, each run was repeated 5 times using different seeds, or slightly shifted grids when running GRID. In

[3]The CPU time was obtained by compiling the codes with the "-pg" option. This compiling option invokes a runtime recording mechanism that makes profiles by procedure.

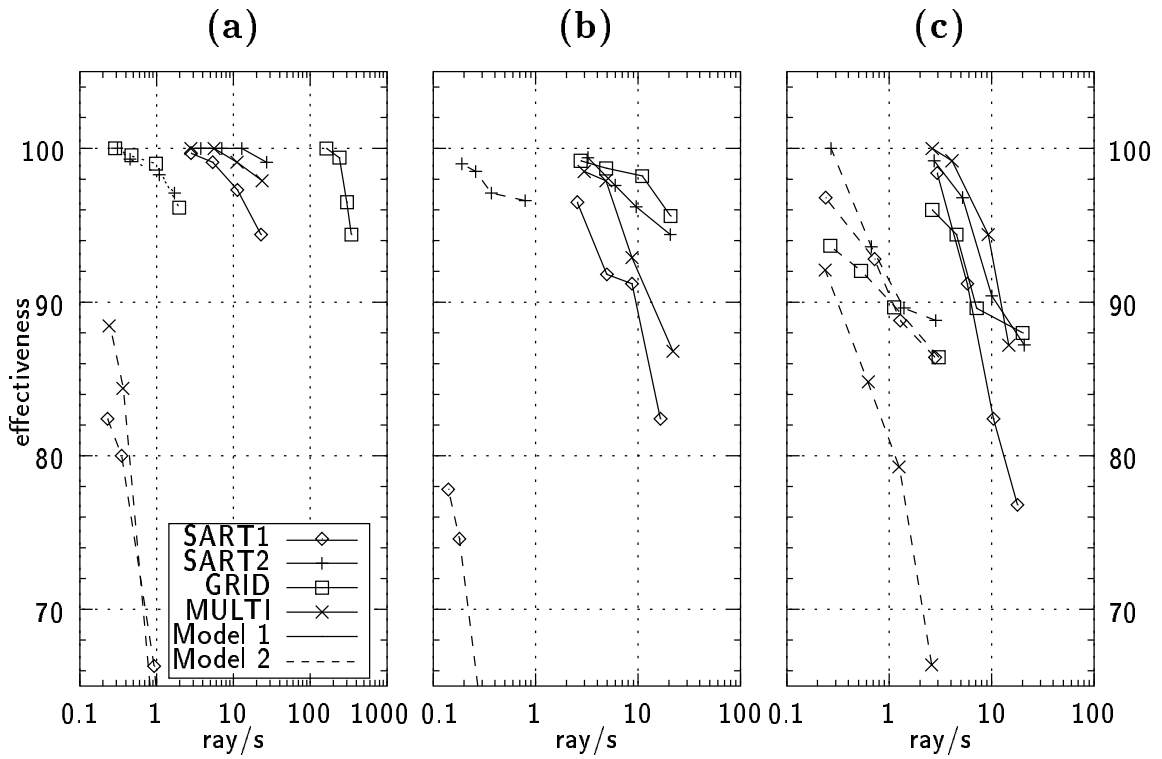

Figure 5.25: Comparison of SART with other methods for both the fault model (solid line) and the salt-dome model (dashed line). (a) 2-D experiment; (b) 3-D experiment, case 1; (c) 3-D experiment, case 2. Data is shown in Tables 5.9 and 5.10. See text for details.

SART2, ITMAX was kept constant in Model 1 (3-D, case 2 only), and in Model 2 (all cases). Otherwise, ITMAX decreased linearly to half the number of iterations for the first source-receiver pair, as explained above. The results are summarized in Table 5.9, for Model 1, and Table 5.10, for Model 2. Also, the tables show the percentage of rays that converged to the actual global minimum within a 1% tolerance. These numbers are a measure of the effectiveness of the algorithms. The actual global minimum raypaths for each source-receiver pair are shown in Figures 5.10b and 5.23a, for Model 1, and in Figures 5.17b and 5.24a, for Model 2. Figure 5.25 plots data in previous tables for

each experiment. Note that in Model 1 (first acquisition geometry), there is a shadow zone for $z > 67.0$ m, approximately. So, the total number of rays in these experiments was $5 \times 68 = 340$. The total number of rays for the second acquisition geometry was $5 \times 25 = 125$. In Model 2, these numbers were $5 \times 41 = 205$, and $5 \times 25 = 125$, respectively.

One obvious conclusion is that the rays traced per second decrease and the effectiveness increase with the number of iterations in SART, the density of the grid in GRID, and the number of starting points in MULTI. In Model 1, 2-D experiment, GRID outperforms both MULTI and SART by about one order of magnitude in terms of CPU time. GRID computes 200-300 rays per second to reach about $98 - 99\%$ effectiveness, while SART2 computes 20-30, only. When the model becomes more complex, such as in the salt-dome case, SART2 attains the same degree of effectiveness and efficiency than GRID. Now the number of traced rays per second decreases to about 1 ray/s for 98-99% effectiveness. To obtain this effectiveness level, SART2 required ITMAX of about 100, for Model 1, and 1000 for Model 2. For GRID, $\Delta\theta_s = 0.5$ was enough in Model 1, but it should be made as small as 0.0015 in Model 2, so that the global minima were not missed. On the other hand, both SART1 and MULTI are slower and take more iterations to reach a certain effectiveness than SART2 and GRID (Figure 5.25a), especially in Model 2.

In the 3-D experiments, first acquisition geometry, things are different (see Figure 5.25b). In Model 1, GRID is still best, but SART2 attains almost the same efficiency and effectiveness. MULTI is not a bad option for the slowest runs (5 rays/s), and produces 98% effectiveness, like GRID and SART2, when it is restarted 40 times. SART requires ITMAX of about 500, and GRID $\Delta\theta_s \simeq 0.4$. In Model 2, SART2 requires 2000-3000 iterations to reach these levels of effectiveness (0.3 rays/s), while SART1 barely reaches 80% after 4000 iterations (0.15 rays/s), GRID as low as 40% with $\Delta\theta_s = 0.1$ (0.1 rays/s), and MULTI misses 9 out of 10 global minima at a low rate of 0.13 rays/s. In the case of MULTI and GRID, these numbers are unacceptable. Note that here the cost functions

	2-D			3-D, case 1			3-D, case 2		
Method	PAR	ray/s	1.0 %	PAR	ray/s	1.0 %	PAR	ray/s	1.0 %
SART1	300	2.12	45.4	1000	0.53	35.1	300	2.81	86.4
	750	0.93	66.3	2000	0.31	62.0	750	1.28	88.8
	1500	0.35	80.0	3000	0.18	74.6	1500	0.72	92.8
	3000	0.23	82.4	4000	0.14	77.8	4000	0.24	96.8
SART2	300	1.72	97.1	1000	0.79	96.6	300	2.84	88.8
	750	1.08	98.3	2000	0.37	97.1	750	1.40	89.6
	1500	0.45	99.3	3000	0.26	98.5	1500	0.67	93.6
	3000	0.31	100.0	4000	0.19	99.0	4000	0.27	100.0
GRID	0.0030	1.98	96.1	0.30	0.72	17.1	2.5	3.05	86.4
	0.0015	0.99	99.0	0.20	0.34	28.3	1.5	1.18	89.6
	0.0007	0.47	99.5	0.15	0.18	31.7	1.0	0.54	92.0
	0.0004	0.29	100.0	0.10	0.09	39.0	0.7	0.27	93.6
MULTI	25	2.28	34.6	100	0.44	4.4	25	2.60	66.4
	50	1.13	57.6	150	0.31	7.3	60	1.26	79.2
	150	0.37	84.4	250	0.20	9.8	120	0.62	84.8
	200	0.25	88.4	350	0.13	12.7	300	0.24	92.0

Table 5.10: Model 2: comparison of SART with other methods in the salt-dome model. PAR equals ITMAX, $\Delta\theta_s$ (and $\Delta\xi_s$), and number of starting points for SART, GRID, and MULTI, respectively.

are very complex, and SART2 outperforms all other methods by far.

In the case of the second acquisition geometry, 3-D experiment, the results of the computations show that almost all methods perform well in both models (see Figure 5.25c). In Model 1, all methods except GRID attain at least 98% effectiveness at rates of about 3 rays/s. In Model 2, only SART2 reaches these levels of effectiveness at a rate of 0.3 rays/s, SART1 reaches 97%, and GRID and MULTI keep below 94%.

One can conclude that for relatively simple 2-D models, GRID is best. In simple 3-D

		2-D			3-D, case 1		
Model	N_r	SART2	GRID	MULTI	SART2	GRID	MULTI
1	1	0.08	0.13	0.13	0.42	5.12	0.25
	10	0.32	0.16	0.78	1.67	4.99	1.91
	25	0.79	0.18	2.01	4.75	5.60	5.40
	50	1.77	0.25	4.72	9.75	6.01	8.48
	100	3.83	0.28	7.86	22.22	7.06	16.71
	150	6.15	0.52	13.25	36.06	7.67	25.84
	200	8.98	0.64	16.67	56.49	9.89	34.02
	250	13.29	0.90	20.58	79.19	10.16	42.73
2	1	0.58	27.77	4.24	3.35	-	39.87
	10	4.29	27.37	53.02	28.77	-	-
	25	10.56	28.16	-	66.62	-	-
	50	21.73	28.34	-	-	-	-
	100	44.04	30.03	-	-	-	-
	150	68.50	29.77	-	-	-	-
	200	92.52	31.01	-	-	-	-
	250	-	36.53	-	-	-	-
1	-	100	0.200	20	750	0.600	40
2	-	400	0.002	320	2300	-	2700

Table 5.11: Comparison of SART with other methods for different number of receivers, N_r. The values indicate CPU time (in seconds) required to find all global minima with 1% accuracy and 98% effectiveness (except Model 1, 2-D case, where the resulting effectiveness was 99%). Values greater than 100 s were not computed. Last two rows show ITMAX, $\Delta\theta_s$ (and $\Delta\xi_s$), and number of restarting points used to achieve the desired effectiveness level in SART2, GRID, and MULTI, respectively.

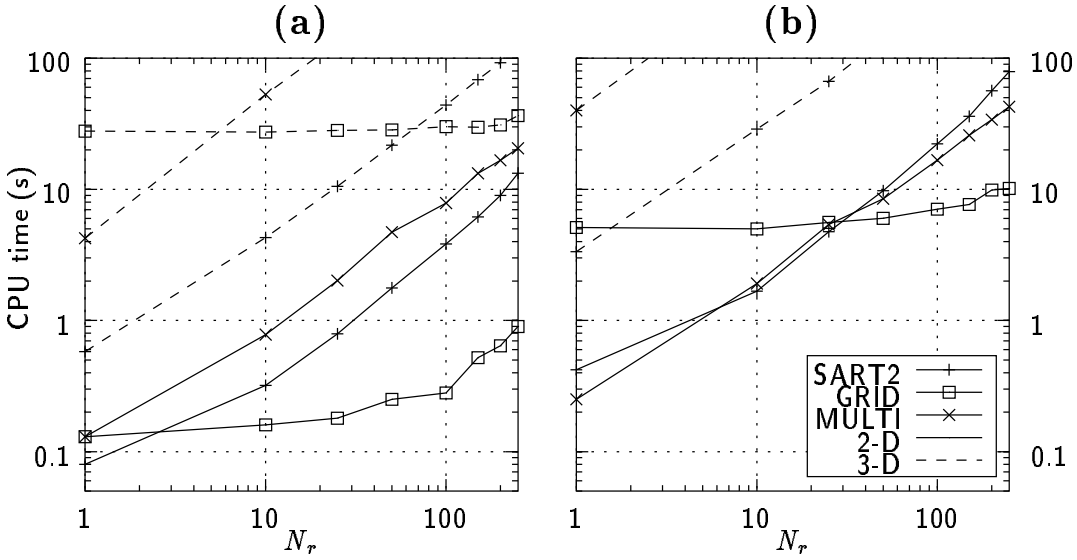

Figure 5.26: Comparison of SART with other methods for different number of receivers. (a) Fault model. (b) Salt-dome model. Data is shown in Tables 5.11.

models, all methods work well, and SART1 is the less effective. In complex situations, SART is the best option, especially in the 3-D case. Here GRID and MULTI are very inefficient, since there are two degrees of freedom. This is not a big obstacle in SART. What makes SART less efficient in 3-D than in 2-D experiments is the complexity of the model (number of discontinuities, smoothness of the velocity fields, etc.), and not the fact that the number of degrees of freedom is doubled. In MULTI and GRID, both issues play a key role in terms of efficiency and effectiveness.

It is important to notice that the number of rays sharing the same source to be traced needs to be considered in this quantitative comparison of methods. If one were to compute a single ray, SART would be much more efficient than GRID and MULTI, for a given effectiveness level. On the other hand, the larger the number of rays with the same source, the more efficient GRID would be relatively to SART and MULTI. Figure 5.26 and Table 5.11 illustrate this point. Here, all computations were done so that the

resulting effectiveness to find the global minima within a 1% accuracy was 98% (except in Model 1, 2-D case, where the effectiveness was 99%). In general, the CPU time increases linearly with N_r (number of receivers) in both SART and MULTI. On the contrary, the CPU time in GRID is rather independent of the number of rays to be traced, as expected. This is because the cost function at all grid nodes needs to be evaluated once, for any value of N_r. In SART, for example, the SA run is repeated for each source-receiver pair independently (though SART2 collects information corresponding to different runs to improve next solutions, as explained above). In simple 2-D models (Figure 5.26a, solid line), GRID is much more efficient than SART and MULTI in all cases, except for tracing a single ray. In complex 2-D models (Figure 5.26b, solid line), GRID is more efficient than SART and MULTI only for $N_r \simeq 25$. In 3-D, things are different. Now, SART is the most efficient method, even for tracing large numbers of rays in complex models (Figure 5.26b, dashed line). In simple 3-D models (Figure 5.26a, dashed line), GRID is superior to SART only for tracing more than 60-70 rays with the same source.

5.7 BVP as a constrained nonlinear optimization problem: a discussion

5.7.1 Straight-ray construction drawback

As mentioned in previous chapter, the straight-ray construction may lead to unrealistic raypaths arriving at earlier times than any other realistic path. Even thought the total traveltime is a global minimum, it is possible in some cases that the resulting raypath corresponds to a path not satisfying the ray equations at all points. One can think of a two-layer model where the velocity in the top layer is much lower than the velocity in the bottom one. Let source and receiver be located, for example, right above the discontinuity in the low velocity layer as shown in Figure 5.27. Clearly, the "unrealistic" critically refracted ray propagating through the high velocity medium arrives earlier than

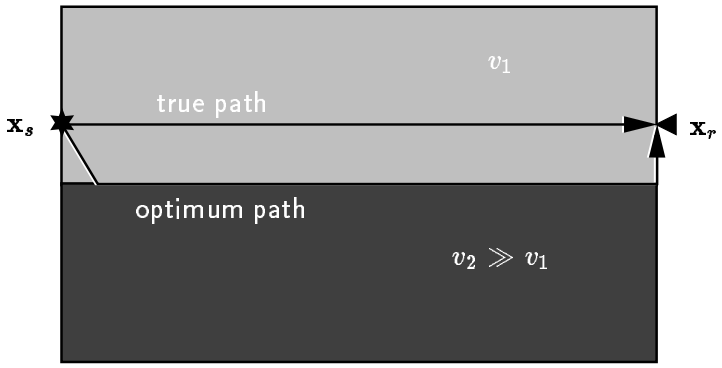

Figure 5.27: Two-layer model where the traveltime of an "unrealistic" raypath is smaller than that of the true (realistic) raypath.

the expected one if $v_2 \gg v_1$. This apparent difficulty will be eliminated in the following reformulation.

5.7.2 Alternative formulations

The boundary-value ray-tracing problem can be viewed as a constrained global optimization problem. In the two-point ray tracing case, the function to be globally minimized is the traveltime from $\mathbf{x}_s$ to $\mathbf{x}_e$, and the constraint is that $\mathbf{x}_e$ must coincide with $\mathbf{x}_r$, within a given tolerance. The set of parameters that satisfy the constraint is called *feasible region.* Put it mathematically,

$$\text{minimize} \quad T_{se} = \int_{\mathbf{x}_s}^{\mathbf{x}_e} \frac{1}{v(\mathbf{x})} ds, \qquad \text{subject to} \quad d \leq d_{tol}, \tag{5.41}$$

where d is the distance between $\mathbf{x}_e$ and $\mathbf{x}_r$, and d_{tol} is a tolerance distance. Both distances are measured, for example, along the boundary plane. Since the traveltime usually represents an observation with a certain error, the optimization problem can be expressed in terms of a tolerance traveltime T_{tol}:

$$\text{minimize} \quad T_{se} = \int_{\mathbf{x}_s}^{\mathbf{x}_e} \frac{1}{v(\mathbf{x})} ds, \qquad \text{subject to} \quad T_{er} = \int_{\mathbf{x}_e}^{\mathbf{x}_r} \frac{1}{v(\mathbf{x})} ds \leq T_{tol}, \tag{5.42}$$

The tolerance traveltime is related to the tolerance distance by

$$d_{tol} \simeq \bar{v} T_{tol}, \tag{5.43}$$

where $\bar{v}$ is some mean velocity around the receiver point. Note that in the previous formulation the straight-ray construction is not explicitly taken into account, except for the fact that the integration involved in the constraint is done along a straight line segment connecting $\mathbf{x}_e$ and $\mathbf{x}_r$. Problem (5.42) is nonlinear in both the cost function and the constraint terms (except for some simple form of the velocity field, for which the path is known) but also multimodal and nondifferentiable in general. This is a rather difficult optimization problem.

In standard shooting methods, the optimization problem reduces to solving

$$d = \|\mathbf{x}_e - \mathbf{x}_r\| \leq d_{tol}. \tag{5.44}$$

Although this formulation seems much simpler, it is necessary to find all minima of equation (5.44) for which $d \leq d_{tol}$ to decide later which one corresponds to the global minimum traveltime.

In SART I solve (5.42) using VFSA, taking special care to properly incorporate the constraint. For this purpose, I transform the constrained optimization problem (5.42) into an unconstrained one. This can be done in several ways. The most direct way of incorporating constraints in a SA optimization problem is to reject those proposed models that do not satisfy the constraint. That is, during the perturbation stage of the

SA process, random perturbations are generated until the proposed configuration satisfies the constraint. Since usually T_{tol} is very small, and so is the feasible region, this strategy may be extremely inefficient. This is because almost all proposed configurations will not satisfy the constraint, especially during the first stages of the process when temperature is relatively high. Alternatively, as in the case of tracing reflected waves, I can define

$$\Phi(\theta_s, \xi_s) = \begin{cases} T_{se}, & T_{er} \leq T_{tol}, \\ T_{max}, & \text{otherwise}, \end{cases} \tag{5.45}$$

where $T_{max} \geq T_{se}$ for all possible take-off angles. In this formulation, all proposed configurations are acceptable. The decision whether to accept or reject a trial configuration is made by the Metropolis criterion instead. Again, since T_{tol} is usually very small, cost function (5.45) is very difficult to optimize. This function is basically a flat surface with a set of "holes", each one corresponding to a beam of rays satisfying the constraint (multipathing).

A better alternative is to define the cost function

$$\Phi(\theta_s, \xi_s) = \begin{cases} T_{se}, & T_{er} \leq T_{tol}, \\ \rho T_{er}, & \text{otherwise}, \end{cases} \tag{5.46}$$

where ρ is a constant, and T_{se} is the traveltime after solving the IVP. I chose ρ so as to guarantee that $\Phi[T_{er} \leq T_{tol}] \leq \Phi[T_{er} > T_{tol}]$ for all (θ_s, ξ_s). This ensures that the global minimizer of Φ belongs to the feasible region. I set

$$\rho \geq T_{max}/T_{tol} \tag{5.47}$$

where T_{max} denotes the maximum traveltime for all possible IVP solutions. Notice that this formulation is not valid if T_{tol} is zero. However, in real-world situations we rarely collect exact data, and a tolerance distance on the target position is allowed.

In formula (5.46) the "holes" are still present, but the surface height decreases in the proximity of the feasible regions. The advantage of this formulation lies in the fact that if the current solution is not in the feasible region all effort is put into guiding the solution towards this region (the "holes"). On the other hand, when the current solution satisfies the constraint, all computational effort is put into minimizing traveltime T_{se}, and traveltime T_{er} is no longer taken into account. Notice that iteration is not stopped as soon as $T_{er} \leq T_{tol}$. Rather, iteration continues within the feasible region so as to obtain the global minimum in case multipathing exists. SART is stopped only after a maximum number of iterations or when the cost function is not improved after a number of consecutive steps. Formulation (5.46) introduces a discontinuity at $d = d_{tol}$ at most of size

$$\Delta\Phi = T_{max} - T_{opt}, \tag{5.48}$$

where T_{opt} is the global minimum and d_{tol} is the tolerance distance defined in equation (5.43). This discontinuity does not represent a major obstacle, in general, for a stochastic optimization algorithm. But if d_{tol} (or T_{tol}) is very small, the "diameter" of the feasible region becomes very small, too (the feasible region reduces to a point and $\Phi \sim T_{er}$ in case $T_{tol} = 0$. This is equivalent to minimizing distance d). The optimization of such a cost function may become very difficult, even for VFSA.

Alternatively the cost function could be defined in terms of a conventional penalty factor:

$$\Phi_p(\theta_s, \xi_s) = T_{se} + \rho_p T_{er}. \tag{5.49}$$

where ρ_p, $\rho_p \geq 1$, is a constant that weights the trade-off between traveltime T_{se} and closeness to the feasible region. Now both terms of the sum are minimized simultaneously.

In the first stages of SART, when $T_{er} > T_{tol}$, it is likely that some local minima of T_{se} may delay convergence. But soon, if ρ_p is selected properly, trial solutions start being feasible points. To ensure that the global minimizer of Φ_p is a feasible point, ρ_p must be selected as in the previous formulation, equation (5.47). These values for ρ and ρ_p ensure that the global minima of both cost functions (5.46) and (5.49) correspond to realistic raypaths, since

$$\Phi[T_{er} > T_{tol}] = \rho T_{er} = \frac{T_{max}}{T_{tol}} T_{er} \geq T_{max}, \tag{5.50}$$

and

$$\Phi_p[T_{er} > T_{tol}] = T_{se} + \rho_p T_{er} = T_{se} + \frac{T_{max}}{T_{tol}} T_{er} \geq T_{se} + T_{max} \geq T_{max}, \tag{5.51}$$

where T_{max} is the maximum traveltime T_{se} may take for all possible take-off angles. Note that the larger the value of ρ_p, the harder the optimization problem, since for $\rho_p \gg 1$ cost function (5.49) is relatively insensitive to a change in T_{se}:

$$\frac{\partial \Phi_p / \partial T_{se}}{\partial \Phi_p / \partial T_{er}} = \frac{1}{\rho_p} \ll 1. \tag{5.52}$$

As a result, all feasible points are equally likely to be obtained, because

$$\Phi_p(\theta_s, \xi_s) \simeq \rho_p T_{er}. \tag{5.53}$$

This is equivalent to lowering temperature too fast during the annealing process (quenching). A premature crystallization is obtained and the solution depends on the initial

model. On the other hand, if ρ_p is too small, the chances of obtaining an unrealistic solution get higher, as discussed in the first paragraphs of this section. There is a trade-off, then, between global convergence and feasible solution.

As an alternative penalty formulation, one can think of a single SA run where $\rho_p = T_{max}/T_{tol}$, for annealing temperature $T = T_0$. Then, as temperature decreases, ρ_p does so following a predefined decreasing ratio. Apparently, this scheme would make the job in a single SA run: at high temperatures the solution is guided towards the feasible region. When temperature is low, the minimization concentrates on finding the minimum traveltime T_{se}. However, the resulting cost function has changed during the process and global convergence cannot be guaranteed anymore. Nothing prevents the system of being trapped in a local minimum during the high-ρ_p stages of the process. When T is low, it may be too late to escape. Besides, the predefined ratio for decreasing ρ_p must be tuned very carefully for each particular problem. This is against the philosophy of SART of being a simple general algorithm for solving the two-point ray-tracing problem.

After testing SART under several numerical experiments I have found that $\rho_p = 1$ is enough to guarantee global convergence to a feasible point in most cases[4]. This is an advantage, since it turns out that although cost function Φ_p with $\rho_p > 1$ tends to guarantee global convergence to a realistic raypath, the optimization of such a cost function takes more iterations, in general, than the $\rho_p = 1$ case, especially when T_{tol} is very small. The same can be said for minimizing cost function Φ, where ρ should not be, in general, smaller than T_{max}/T_{tol}. This issue is demonstrated below.

Take Figure 5.28 as an example. Cost functions Φ and Φ_p, which correspond to the 2-D fault model, were computed using

$$T_{max} = \max\{T_{se}(\theta_s)\} \simeq 35.0 \text{ ms}, \qquad T_{tol} = 1.0 \text{ ms}, \qquad \rho = \rho_p = 35.0.$$

[4]Note that actually equation (5.49) together with $\rho_p = 1$ is the formulation I have used in Chapter 4 in all the examples.

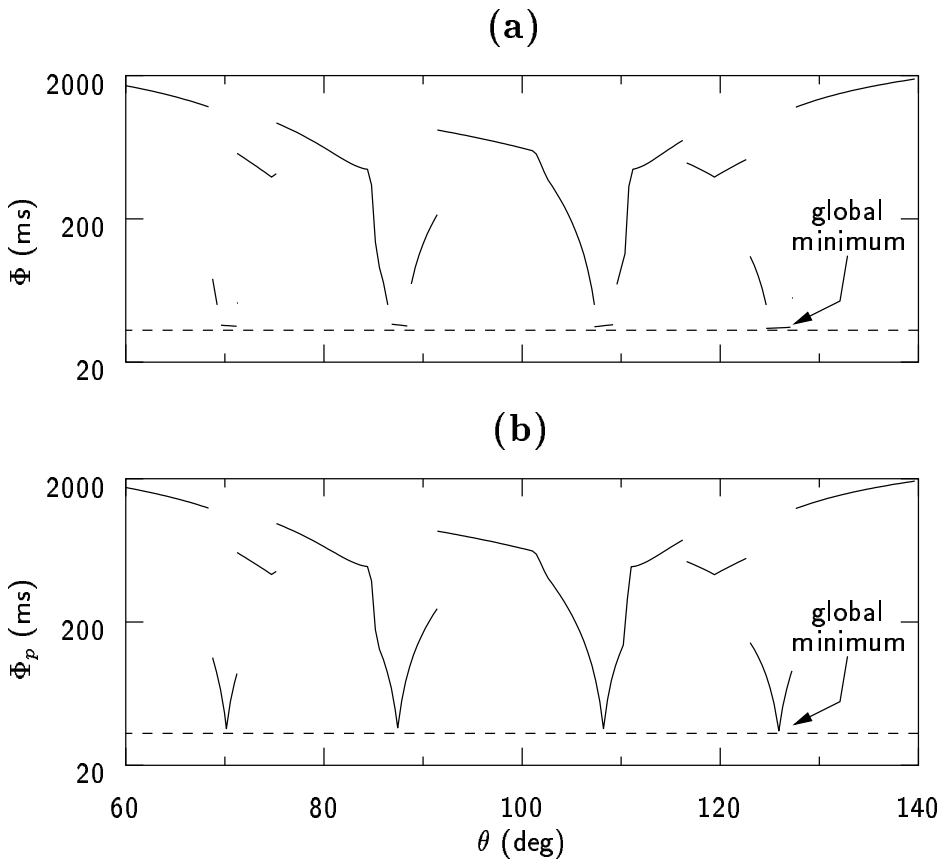

Figure 5.28: Model 1: alternative cost functions for the fault model. (a) Φ, and (b) Φ_p, as a function of take-off angle θ_s, with $\rho = \rho_p = 35.0$.

These curves are to be compared to the cost functions plotted in Figure 5.9, i.e. d and $T = T_{se} + T_{er}$. As a result of the selected values, the absolute minimum of both Φ an Φ_p correspond to a ray satisfying the ray equations from source to receiver. The absolute minimum of Φ_p belongs to the feasible region, in general $d = 0$. The solution coincides with that in Figure 5.9b:

$$\theta_s = 125.90^o, \qquad \Phi_p = 34.714 \text{ ms}.$$

Cost function Φ has also a unique global minimum, but the values are slightly different:

Φ - equation (5.46)					Φ_p - equation (5.49)				
T_{tol}	$\bar{k}$	σ_k	$\overline{T}_{er}$	$\sigma_{T_{er}}$	ρ_p	$\bar{k}$	σ_k	$\overline{T}_{er}$	$\sigma_{T_{er}}$
1.00	339.6	360.6	0.630	0.259	1	239.5	129.0	0.286	0.113
0.75	461.1	575.0	0.464	0.190	2	440.4	415.1	0.119	0.047
0.50	673.5	782.4	0.322	0.124	3	706.4	831.4	0.080	0.029
0.10	1208.2	1034.9	0.067	0.023	5	887.9	1125.2	0.048	0.016
0.03	1663.5	1421.7	0.017	0.006	10	1333.9	1304.0	0.025	0.008
0.01	2252.2	1819.2	0.007	0.002	35	2345.4	1944.6	0.007	0.002

Table 5.12: Mean and standard deviation of the number of iterations, k, required to obtain the global minimum within 1% accuracy for various cost functions, and mean and standard deviation of traveltime T_{er}.

$$\theta_s = 124.61^o, \qquad \Phi = 34.402 \text{ ms.}$$

Now the search for the minimum within the feasible region is not biased towards $d = 0$. Consequently the global minimum raypaths does not necessarily arrive exactly to the receiver, but within a tolerance distance d_{tol} as defined in equation (5.43). In the above example, it is not surprising that the global minimum of cost function Φ corresponds to a distance d of about 3 m, since $d_{tol} = 3$ m for the given receiver location and tolerance traveltime ($v = 3$ km/s in the near receiver region).

In order to demonstrate which is the best option, in terms of number of iterations to find the global minimum, I have run SART using several cost functions, for the 3-D fault model example taking into account both take-off angles. It is assumed that the global minima of all cost functions correspond to a feasible point, and $T_{max} = 35.0$ ms. Table 5.12 summarizes the mean and standard deviation of the number of iterations required to obtain the global minimum within a 1% accuracy, after 200 realizations in each case. Note that the linearizing stage to refine the solution was not applied in this case. In the case of cost function Φ, T_{tol} was varied from 0.01 ms to 1.00 ms, corresponding to ρ in the range

35 to 3500. As T_{tol} decreases, the mean number of iterations increases exponentially, as illustrated in Figure 5.29a (solid line). On the contrary, $\bar{T}_{er}$ decreases linearly with T_{tol}. This implies that for obtaining a very accurate solution (T_{er} small), a large number of iterations should be done. Conversely, if T_{tol} is set too large, the emerging point may be too far away from the receiver point, no matter how many iterations are performed. Cost function Φ_p behaves differently (see Figure 5.29b). The values were obtained for $T_{tol} = 1.0$ ms. Despite the fact that T_{tol} has been fixed to 1.0 ms, the minimization of Φ_p leads to very accurate solutions in fewer iterations, especially for ρ_p close to 1. The intersection point between the curves in Figure 5.29 is the best option for maximum accuracy and greatest efficiency. It is worth mentioning that the refinement stage that was not applied for constructing Table 5.12 and Figure 5.29, would have reduced T_{er} virtually to zero in both examples. However, in the case of cost function Φ, there may be ambiguities when dealing with very similar solutions unless T_{tol} is set very small. If T_{tol} is too large, two or more solutions may lie within a single "hole". Cost function Φ_p does not present this difficulty, but may require a large ρ_p to guarantee global convergence to a feasible point in some complex media.

In SART, $\rho_p = 1$ is used by default. In particular models, as that shown in Figure 5.27, ρ_p can be increased in successive SA runs until an upper bound of $\rho_p = T_{max}/T_{tol}$. Whenever SART converges to an unrealistic raypath, ρ_p may be increased by a given factor (say 2.0) until the global minimum corresponds to a realistic path. This process should only be repeated a few times (two or three) because the failure to converge to a realistic path may indicate the presence of a shadow zone rather than ρ_p being too small.

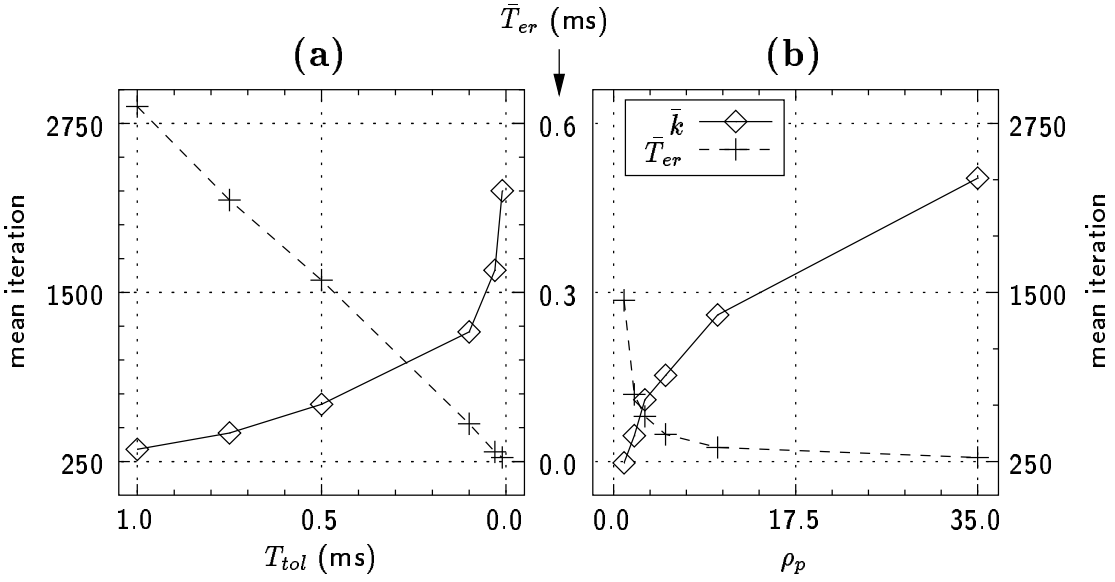

Figure 5.29: Mean number of iterations and traveltime T_{er} for various cost functions. Data is shown in Table 5.12. (a) Cost function Φ. (b) Cost function Φ_p. In both cases, $T_{max} = 35.0$ ms.

5.8 Conclusions

In this chapter I have developed a 3-D extension of the ray-tracing algorithm presented in Chapter 4. SART is a very versatile computational algorithm for solving the boundary-value ray-tracing problem in a general 2-D/3-D model. SART aims to find the raypath connecting any source-receiver pair so that the associated traveltime is a global minimum. The solution is found after solving iteratively a highly nonlinear optimization problem by means of VFSA. The results are independent of the initial guess, since VFSA is a global stochastic optimization algorithm. Contrarily to SART in 2-D, which is based on a cell ray-tracing scheme, SART extension to 3-D solves numerically the equations derived from the high-frequency ray approximation theory. This new formulation provides greater accuracy in the calculations and greater flexibility for dealing with complex models.

For this purpose, I have also developed a versatile model parameterization system that allows to represent a wide variety of geologic 3-D (or 2-D) structures. Any number of regions delimited by arbitrary interfaces can be defined. The velocity (or slowness) within each region is specified separately, which allows to model any type of waves, including converted waves.

As in the 2-D case, SART extension to 3-D exhibits some improvements over existing ray-tracing techniques like bending and shooting. The problem of local minimum paths is eliminated provided a sufficient number of iterations are performed. Despite the fact that SART solves an IVP at each iteration, the selection of the appropriate take-off angles does not represent a serious difficulty even for ill-behaved receiver-distance functions. This issue is particularly important for tracing complex raypaths where more that two unknown parameters are to be found, for example when tracing headwaves.

It is worth to mention that SART goal is to find a single raypath corresponding to the global minimum traveltime. Multiple arrivals can be obtained only if separate search ranges for the unknown parameters are specified. The optimum raypath obtained by SART corresponds to a particular solution of the IVP. Consequently, first-arrival raypaths traveling across shadow zones and arbitrary headwaves cannot, in general, be found. It has been pointed out that these waves are not the most energetic events because they fill space fastest and tend to have maximum geometric spreading, especially in models with strong lateral velocity variations [GB93]. Hence the use of these arrivals (usually obtained by finite-difference methods) for geotomographic applications is limited or presents some difficulties. SART arrivals, on the contrary, correspond to the most energetic events (direct waves) since they are obtained after solving the IVP.

It is emphasized that the physical nature of the resulting ray is known a priori and a posteriori, a point that is very important in phase identification. Later arrivals, like reflections and headwaves, can be obtained by specifying a ray signature in the way of

constraints. This is done by penalizing those rays not following the desired signature. The methodology can be extended to deal with more complex raypaths (multiples, higher order headwaves, diffractions, etc), including ray conversions of any type.

The accuracy of the results, unlike in bending, does not depend on the parameterization of the raypath but on the selected integration method and timestep. Higher-order accuracy is obtained by means of the 4th-order RK integration, for example, at no significant extra computational cost. The source of errors is limited then to the accuracy used in the calculation of the intersections between raypaths and interfaces, and on the distance between the ray ending point and the receiver. The errors associated with the intersection points can be made very small simply by increasing the accuracy of the iterative root-finder. The other source of errors is eliminated by polishing the final raypath after the last annealing iteration. This is carried out by using a standard linearizing optimization method (e.g. steepest descent) to reduce the receiver distance virtually to zero within machine precision.

In terms of computational cost, SART is expensive as compared to conventional boundary-value ray-tracing systems. This is because several iterations are required to ensure global convergence. Contrarily, conventional shooting and bending methods usually converge (if not diverge) in a much smaller number of iterations. But their convergence is only local. For global convergence, it has been demonstrated that SART is much faster than some shooting variants such as grid-search and multistart, when dealing with complex 2-D/3-D media. One disadvantage of SART is that raypaths connecting source-receiver pairs are found one at a time. This makes SART unpractical for traveltime tomography when the number of sources and receivers is large. Nevertheless, when there are a few source-receivers pairs, when the geology is complex, and direct first arrivals are required, SART is a valuable tool. In these cases, conventional shooting, bending and finite-differences methods present serious difficulties that SART overcomes naturally.

Finally, it is worth noting that SART may be readily parallelized. Each source-receiver pair may be treated separately by individual processors. Since the forward problem is by far the most time consuming stage in SART, it may be possible to obtain superlinear speedups by using various processors. This means that the speed per processor increases when processors are added.

Chapter 6

Traveltime tomography

Some circumstantial evidence is very strong, as when you find a trout in the milk.

Henry David Thoreau

6.1 Introduction

Traveltime tomography for imaging subsurface velocity variations represents a very important technique in global seismology and exploration studies. The method is used to obtain the velocity distribution from measured time values (inverse problem). The basic idea can be traced back to as early as 1917, when a method to locate ore bodies by combining traveltimes from a number of sources and receivers in various boreholes was proposed [Fes17]. Though this method requires a fairly simple medium to roughly locate the ore bodies by hand interpretation and elementary geometry, it points forward to the present use of crosswell tomography. In its simplest form, traveltime tomography utilizes straight rays connecting sources and receivers. Then the problem reduces to solve a linear system of algebraic equations. Aside from some issues regarding ill-conditioning and/or sparseness of the derived model matrix, the traveltime tomography problem using straight rays can be solved without major difficulties. But in highly refractive media the straight ray approximation becomes invalid and the resulting inverted model is inaccurate.

Curved ray traveltime tomography, on the other hand, provides a more accurate approximation to the actual physical phenomenon. This technique was originally developed by [BPLT72] for estimating the velocity distribution between two wells. As opposed to

straight ray traveltime tomography, curved ray traveltime tomography is a highly nonlinear problem. This is due to the fact that the arrival-times are nonlinearly related to the unknown slowness field. In other words, not only the velocity distribution is unknown, but also the raypaths. Usually, the problem is solved using linearizing techniques in an iterative fashion. In general, a good starting model is required and some form of regularization or model constraints must be introduced in the objective function to stabilize the solution and to minimize the artifacts generated by the inversion (see for example [Nol87, BBC89, SDG90, Ald92]). In addition, this procedure is also intended to introduce a priori information in connection with the subsurface slowness field, such as flatness, closeness to a prescribed base model, etc.

An alternative is to introduce such constraints in the model itself through a suitable parameterization, an essential stage in any inverse problem. Rather than using box-like cells, the nonlinear tomographic inversion technique described in this chapter makes use of either bicubic B-splines with adaptive node spacing, or 2-D parametric functions for representing velocity anomalies. The motivation for using bicubic B-splines is two-fold: (1) smoothing is obtained beforehand without the need to regularize in the standard fashion, and (2) the number of unknown parameters is reduced substantially making the problem tractable using a computationally intensive algorithm like simulated annealing (SA). Although bicubic B-splines are inherently smooth, relatively sharp velocity changes could be obtained by locating a few nodes at key points [Sha93]. I use VFSA to optimize the velocity values at the spline nodes and the location of the nodes themselves, so the grid is dynamically adapted along with the nodes values. This strategy allows one to model moderately complex structures using only a few nodes (3-5 in each dimension) [VU95b]. In a previous work [Mic95], an adaptive-grid formalism which uses bicubic B-splines for traveltime tomography is also used. Here node coordinates are updated using a local linearizing method rather than a global optimization one.

In other tomographic applications, such as archaeology, mining, and other near surface studies, one is interested in locating several or a single velocity anomaly submerged in a smooth background velocity field. Since it is very difficult to decide a priori whether the underlying structure corresponds to a smooth or a high contrasting velocity distribution, the selection of a proper parameterization that does not favors any of the two forms is desirable. This is the motivation of using parametric 2-D functions as an alternative for reducing the nonuniqueness of the inverse problem at hand. This limits the number of unknowns significantly and virtually eliminates the artifacts and instabilities associated with the tomographic problem. At the same time, it provides sufficient flexibility to model complex structures with a few parameters.

The parameterization adopted in this chapter can be complemented by incorporating a linear trend to the velocity field. The coefficients of the trend represent extra parameters in the traveltime inversion. Velocity constraints can be easily included by imposing range limits to the various parameters which are involved. Prior information about the subsurface structure can also be used to guide the spline node locations and the anomaly size and position within the model boundaries. This a priori information is very useful to reduce the number of forward modeling computations required for convergence, but are not essential for the success of the nonlinear optimization scheme.

The two forms of parameterizations just described, introduce strong nonlinearities into the optimization problem. Besides, the resulting objective function (misfit between observed and calculated arrival times) is multimodal and rather ill-behaved. Unless the initial model is close to the global minimum, the solution is likely to be trapped in a local minimum if a local linearizing method is utilized. This is the reason why I use VFSA, at the expense of a higher computational cost. SA has already been applied to traveltime tomography without raytracing [AV90, AV93, PL93]. But the parameterization they used is based on a standard grid with rectangular cells. Though a spatial smoothing is

introduced at the perturbation stage to reduce the unrealistic artifacts associated to the inversion, the limitations of a box-like parameterization are still present.

This chapter is organized as follows. First I describe the traveltime inversion problem in a general sense. I also describe the two forms of parameterization that will be used to model the velocity field. Second, I present the traveltime inversion as nonlinear optimization problem, which is to be solved by means of VFSA. Finally, the method is illustrated with various synthetic examples. These comprise crosswell, and well-to-surface 2-D experiments. The results of the nonlinear scheme are compared to a linearized inversion based on Vidale's approach [Ald92]. The two forms of parameterization may also be used, combined or alone, in conjunction with other geophysical techniques, such as magnetic, or geoelectric methods, provided the forward problem can be solved at each iteration of the SA algorithm for the given model representation. It is not the purpose of the proposed techniques to replace any preexisting inversion procedure for general seismic exploration studies, but to provide an alternative methodology for dealing with particular models susceptible of being represented by a few parameters. The method is especially suited for imaging smooth (regional) models and/or near surface structures such as archaeological and mining prospects.

6.2 General theory

Under the high-frequency approximation of wave theory, the traveltime of a seismic wave propagating through a slowness field, $s(x,z)$, is given by the path integral (see for example [Nol87]):

$$T(s) = \int_{L(s)} s(x,z)dl, \tag{6.1}$$

where $L(s)$ is the raypath connecting source and receiver, which satisfies Fermat's principle. The problem is to derive $s(x,z)$ from a number of measured traveltimes $T(s)$. This is a very complicated nonlinear inverse problem, since the unknown slowness field is implicit in the path of the integral. The basic procedure is then to linearize using perturbation theory. For this purpose consider a reference model, s_0, and a small perturbation, Δs. Ideally, this yields a small change in traveltime:

$$\Delta T = T - T_0 = \int_L s dl - \int_{L_0} s_0 dl \simeq \int_{L_0} \Delta s dl, \tag{6.2}$$

where $L_0 = L(s_0)$. Note that, after invoking Fermat's principle, we have approximated the integral along the path L by the integral along the path L_0. The above expression is valid to first-order, and allows one to linearly relate a small change in traveltime with a small change in slowness field. In effect, if the slowness field is represented by a set of K cells with constant slowness s_k, $(k = 1, \cdots, K)$, then the raypath within each cell is a straight line segment. For a set of N source-receiver pairs, in matrix notation

$$\mathbf{\Delta T} = \mathbf{A(s)\Delta s}, \tag{6.3}$$

where $\mathbf{A(s)}$ is a $N \times K$ matrix whose elements are raypaths length segments. Equation (6.3) maps a projection of the model space into the data space, where in general $K > N$. The reference model, s_0, is known and

$$\mathbf{\Delta T} = \mathbf{T^o} - \mathbf{T^c}, \tag{6.4}$$

where $\mathbf{T^c} = \mathbf{T_0}$ are the calculated traveltimes, and $\mathbf{T^o}$ are the observed traveltimes. The objective of the inverse problem is to find the perturbation $\mathbf{\Delta s}$ such that the calculated traveltimes obtained using the reference model, approximate the observed traveltimes

within a given tolerance. In the case of straight-ray tomography and if a sufficiently good ray coverage is available, equation (6.3) can be solved directly, despite the fact that matrix $\mathbf{A}$ is generally nonsquare and large. Unfortunately, the rays are usually not straight nor is the coverage of paths optimal. Moreover, the observed traveltimes are contaminated by random noise. As a result, matrix $\mathbf{A}(\mathbf{s})$ is, in addition to large and nonsquare, sparse and rank deficient, and the inverse problem becomes ill-posed. To overcome this difficulty, the inversion procedure must introduce a criterion so that elements in the model space can be ranked with respect to each other (regularization, damping, etc). This is to overcome the nonuniqueness problem, for with the only constraint of ΔT being small, there exist an infinite number of models that would reproduce the data (this is the fundamental difficulty encountered in any underdetermined inverse problem). In general, the problem is solved by linearizing the objective function with respect to the model unknowns, and being extremely careful about which criterion has been selected, based on the information available, to make the inverse problem well-posed. For a complete description of methods to solve this linearized inverse problem, the reader is referred to for example [Nol87].

6.3 Model representation

I adopted two contrasting parameterization schemes to represent the velocity field, $v(x, z)$:

1. bicubic B-splines with adaptive node spacing; and

2. parametric 2-D functions.

These are intended to offer an alternative to box-like parameterization and to reduce the number of parameters to a minimum. Yet, they allow a certain flexibility to accommodate complex structures. The first approach favors smooth velocity fields. The spline coefficients are determined "globally" so that the interpolating function is continuous

everywhere up to the second derivative. The second approach, which was devised to represent velocity anomalies (e.g. buried structures), does not favor neither smooth models nor models with sharp discontinuities, though these characteristics could be forced easily by setting the appropriate bounding ranges for some of its parameters.

6.3.1 Bicubic B-splines

Cubic and bicubic B-splines parameterization in tomography has been effectively applied by other authors [Fir87, MM91, Mic95]. In these cases, solutions are found by linearizing the objective function and solving iteratively. Chunduru et al [CSS94] used bicubic B-spline parameterization for 2-D resistivity inversion using VFSA. Their results are encouraging because they effectively inverted areas of complex geology at a low computational cost, even though a nonlinear optimization method was used.

The bicubic spline parameterization method adopted here is based on that given in [PTVF92]. The two-dimensional rectilinear grid consists of $M_x \times M_z$ nodes. To accommodate complex structures, the spacing in each dimension is not fixed. Rather, node coordinates are given by x_i, $i = 1, \cdots, M_x$, and z_j, $j = 1, \cdots, M_z$, with $x_i < x_j$ and $z_i < z_j$ if $i < j$. For simplicity, the first and last nodes of each row (and column) are located on the model boundaries, i.e.

$$\begin{cases} x_1 = x_{min}, & x_{M_x} = x_{max} \\ z_1 = z_{min}, & z_{M_z} = z_{max}. \end{cases} \tag{6.5}$$

Figure 6.1 illustrates a typical grid for $M_x = M_z = 4$.

To perform a bicubic interpolation, the user must specify the function $v(x, z)$ at each node, the derivatives with respect to the coordinates, $\partial v/\partial x$ and $\partial v/\partial z$, and the corresponding second-order cross derivative, $\partial^2 v/\partial x \partial z$. The goal of bicubic spline interpolation, a special case of bicubic interpolation, is to obtain an interpolating formula

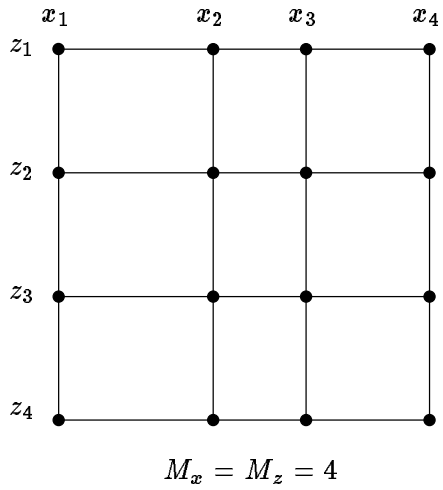

Figure 6.1: Example of a two-dimensional grid used for the bicubic spline parameterization. The node spacing in each dimension is not fixed.

that is smooth in the first derivative and continuous in the second derivative. This is achieved by setting first derivatives equal at either side of each node. As a result, second derivatives can be readily computed provided additional conditions at the boundaries are given. *Natural splines* are obtained if second derivatives are set to zero at the boundaries. Alternatively, these may be set to values so as to make first derivatives equal to zero at the boundaries. This is the approach I adopted here. Having set all these conditions, a unique set of spline coefficients can be calculated and the resulting bicubic interpolating function can be written

$$v_s(x,z) = \sum_{p=1}^{4}\sum_{q=1}^{4} c_{pq} t^{p-1} u^{q-1}, \tag{6.6}$$

where

$$\begin{cases} t = (x - x_i)/(x_{i+1} - x_i), \\ u = (z - z_j)/(z_{j+1} - z_j), \end{cases} \tag{6.7}$$

and c_{pq} are the spline coefficients at each node. More details about bicubic spline parameterization can be found in [PTVF92, LS86].

Additionally, a linear background can also be included:

$$v_b(x,z) = v_0 + g_x(x - x_{min}) + g_z(z - z_{min}), \tag{6.8}$$

where v_0 is the background velocity, and g_x and g_z are the velocity gradients in each dimension. The total number of model parameters, K, is therefore equal to $M_x \times M_z + (M_x - 2) + (M_z - 2) + 3$. That is *number of nodes + number of node coordinates in each dimension + number of linear background coefficients.* For example, if $M_x = M_z = 4$, then the total number of unknown parameters is $K = 23$. So the model space can be expressed as the K-length vector

$$\mathbf{m} = \{\mathbf{v_s}, \mathbf{x_s}, \mathbf{z_s}; v_0, g_x, g_z\}, \tag{6.9}$$

where $\mathbf{v_s}$ is the vector containing the $M_x \times M_z$ spline nodes values, and $\mathbf{x_s}$ and $\mathbf{z_s}$ are the vectors containing the grid coordinates in each dimension. The last three scalars are the velocity background coefficients. Note that the selection of a linear background velocity is arbitrary, and other function could be selected. A constant background would reduce the number of unknowns by two. Yet, this constant may be obviated, too, and the model be represented by the splines alone.

6.3.2 Parametric 2-D functions

The purpose of this parameterization is to represent velocity anomalies using the minimum number of parameters. One possibility is to consider regular geometrical bodies (e.g. circle, rectangle, etc.), which can be defined by its velocity, center coordinates,

radius, width, etc. This approach is very limiting and presupposes a good deal of a priori knowledge about the subsurface structure at hand. As a counterpart, the model representation that I use in this section encompasses a large variety of possible velocity anomalies, being regular geometrical bodies a particular case.

I constructed 2-D anomalies based on 1-D functions. Essentially, the anomaly is represented by two separate parts: (1) a flat region of width $2R$ and amplitude A; and (2) the slopes at each side of the flat region. If the center of the anomaly function is located at x_c, then I define

$$v_a(x) = \begin{cases} A, & |x - x_c| \leq R \\ f(|x - x_c| - R), & \text{otherwise} \end{cases} \tag{6.10}$$

where

$$f(x) = \frac{A}{1 + (\frac{x}{\rho R})^2}, \qquad \rho > 0 \tag{6.11}$$

This function has its maximum, A, at $x = 0$, and tends to zero asymptotically as $x \to \pm\infty$. Factor ρ, a dimensionless constant, is defined in such a way that ρR is the distance from the origin at which the maximum of $f(x)$ decreases by a factor of two. That is,

$$f(|\rho R|) = f(0)/2 = A/2. \tag{6.12}$$

This parameter is a measure of the width of the function $f(x)$. As a result, it controls the *slopes* of function $v_a(x)$. Figure 6.2 shows a series of 1-D anomalies with unit amplitude and width, centered at $x_c = 0$. Notice how the slopes of $v_a(x)$ vary with different values of ρ. For $\rho \to 0$, $f(x) \to 0$, and the anomaly reduces to a box-card. For $\rho \to \infty$, $f(x) \to A$, and the "anomaly" is flat for all x. It is important to remark that equation

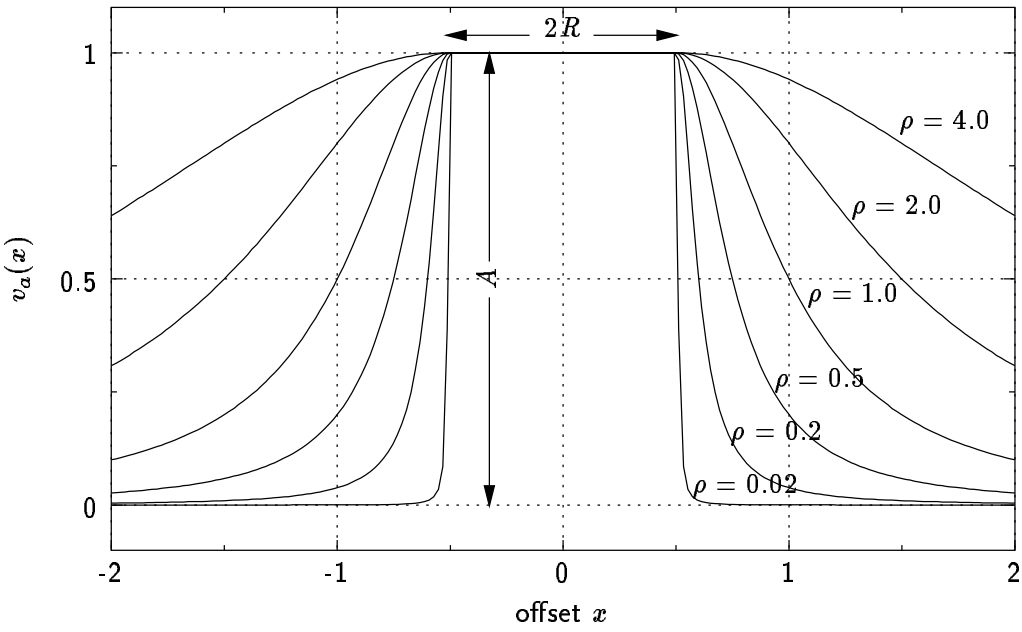

Figure 6.2: One-dimensional velocity anomaly function for various slopes. Both amplitude and width are equal to one.

(6.10) is continuous and differentiable everywhere, even at $x = x_c \pm R$, since $f(0) = A$, and $f'(0) = 0$. Equation (6.10) allows us to model either smooth or high contrasting 1-D anomalies, using only 4 parameters. Namely x_c, A, R, and ρ.

In 2-D, one can make use of equation (6.10) to construct 2-D anomaly functions with similar features. One possibility is to consider a surface of revolution along the axis normal to the xz-plane passing through the center point (x_c, z_c):

$$v_a(r) = \begin{cases} A, & r \leq R \\ f(r - R), & \text{otherwise} \end{cases} \tag{6.13}$$

where

$$r = \sqrt{(x - x_c)^2 + (y - z_c)^2} \tag{6.14}$$

is the distance between the point (x, z) and the center of the anomaly, A is the amplitude, and R is the radius of the flat region. The slopes around this circle are controlled by ρ, as given in formula (6.11). Although this scheme is quite simple and flexible, it is not versatile enough to model complex structures, except those with circular symmetry. We can better approximate complex structures by considering different scales in each dimension to define the flat region: R_x and R_z. Let's redefine the radius of the flat region, R, as the distance between the center of the anomaly, (x_c, z_c), and the following curve:

$$\frac{|x - x_c|^p}{R_x^p} + \frac{|z - z_c|^p}{R_z^p} = 1, \qquad p > 0. \tag{6.15}$$

Here I have introduced a new parameters, p, that helps to control the *shape* of the flat region limits. Notice that for $p = 2$, equation (6.15) becomes an ellipse of center (x_c, z_c) and semiaxes $2R_x$ and $2R_z$. Further, if $R_x = R_z$, it becomes a circle. In practice, R is calculated by finding the intersection point, (x_R, z_R), between the straight segment connecting (x_c, z_c) and (x, z), with the curve (6.15), where (x, z) is a generic point in the xz-plane. After some algebraic manipulation,

$$\begin{cases} x_R = x_c + \frac{x - x_c}{\tilde{r}}, \\ z_R = z_c + \frac{z - z_c}{\tilde{r}}, \end{cases} \tag{6.16}$$

where

$$\tilde{r} = \left(\frac{|x - x_c|^p}{R_x^p} + \frac{|z - z_c|^p}{R_z^p} \right)^{1/p}. \tag{6.17}$$

Since $R(x, y) = r(x_R, z_R)$, combining equations (6.14) and (6.16), it yields

$$R = \frac{r}{\tilde{r}}. \tag{6.18}$$

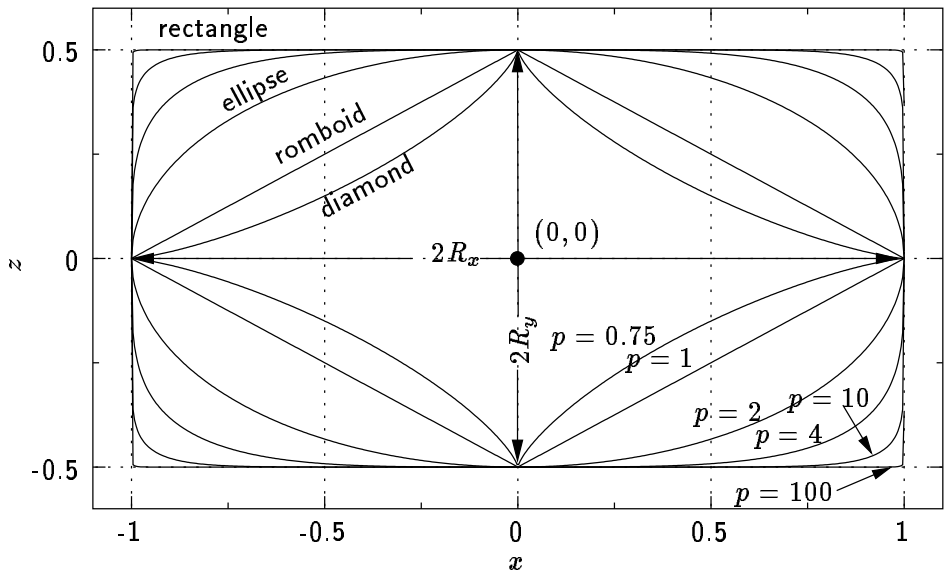

Figure 6.3: Two-dimensional velocity anomaly function (flat region only) for various values of p [equation (6.15)]. In all cases $(x_c, z_c) = (0, 0)$, $R_x = 1.0$, and $R_z = 0.5$. Notice how the anomaly takes on different shapes as p varies.

To define the slopes of the anomaly, it is useful to keep the same shape than used for the 1-D case for values of r greater than R. This result can be obtained by replacing x by r in equation (6.11). It yields

$$f(r) = \frac{A}{1 + (\frac{r}{\rho R})^2}. \tag{6.19}$$

Finally, the 2-D anomaly function is defined like in equation (6.13), with the exception that R is no longer a constant, but computed using equation (6.18), and that $f(r)$ is computed using equation (6.19).

As in the 1-D case, $v_a(r)$ is continuous up to the first derivative, except for $p \leq 1$. The flexibility of this model representation and the meaning of all the parameters required to define the 2-D anomaly, are illustrated in Figure 6.3 for various p. Now the *shape* of the anomaly can be changed easily by setting the appropriate parameters for the flat region,

along with the factor ρ. For example, values of p smaller than or equal to one can also be selected. The resulting flat region takes on a diamond-like shape. If $p \to \infty$, the flat region becomes a rectangle of size $2R_x \times 2R_z$. In effect, from equation (6.15),

$$z = z_c \pm R_z \left(1 - \frac{|x - x_c|^p}{R_x^p}\right)^{1/p}, \qquad |x - x_c| \leq R_x. \tag{6.20}$$

Taking limit,

$$\lim_{p\to\infty} z = \begin{cases} z_c \pm R_z, & |x - x_c| < R_x \\ z_c, & |x - x_c| = R_x. \end{cases} \tag{6.21}$$

In addition, the whole anomaly is rotated an angle ω around the axis normal to the xz-plane passing through its center. This is done using the transformation

$$\begin{cases} x' = x_c + (x - x_c)\cos\omega - (z - z_c)\sin\omega \\ z' = z_c + (x - x_c)\sin\omega + (z - z_c)\cos\omega, \end{cases} \tag{6.22}$$

where x' and z' are the new coordinates in the rotated frame.

In summary, 2-D anomalies are represented using parametric functions defined by 8 parameters, namely A, x_c, z_c, R_x, R_z, p, ρ, and ω. To reduce the number of parameters, it is possible to set $p = constant$ (e.g. $p = 2$), yet having a quite flexible model representation. In this particular case, the number of parameters to be found at the inversion stage reduces to 7. Finally, as in the case of bicubic splines, a linear background velocity can be added, as given in equation (6.8). As a result, the total number of parameters is $K = 11$. That is, *number of anomaly parameters + number of background coefficients.* Then, the model space is given by

$$\mathbf{m} = \{A, x_c, z_c, R_x, R_z, p, \rho, \omega, v_0, g_x, g_z\}, \tag{6.23}$$

a vector of 11 elements.

6.4 Forward modeling

I adopted a finite-difference (FD) method [Vid88, Ald92, AO93] to compute the traveltimes given a velocity model and a source-receiver geometry. Explicit raypaths are not required since the SA inversion is based on the computation of the traveltimes only. The FD method is based on the solution of the eikonal equation using finite-differences on each square cell of a gridded slowness field. The eikonal equation can be easily derived from the high-frequency approximation of the wave equation (see for example [Nol87, Ber91]). The grid is basically the same as that used for ray tracing in Chapter 4 (Figure 4.1), with the exception that $\Delta x = \Delta z = h$, and that velocities are assigned to the nodes. Once a model parameterization has been chosen (bicubic splines or parametric functions), the velocity field is sampled over the equally spaced grid of $N_x \times N_z$ square cells:

$$v_{ij} = v(x_i, z_j), \tag{6.24}$$

where

$$\begin{cases} x_i = x_{min} + (i-1)h, & i = 1, \cdots, N_x \\ z_i = z_{min} + (j-1)h, & j = 1, \cdots, N_z, \end{cases} \tag{6.25}$$

and $v(x,z) = v_b(x,z)$, equation (6.8), plus either $v_s(x,z)$, equation (6.6), or $v_a(x,z)$, equation (6.13). The "sampling" process given by equation (6.24) and (6.25) is repeated at each iteration after the corresponding parameters (spline nodes, background velocity coefficients, etc.) have been updated by the SA algorithm. So, given a source-receiver geometry, and the current gridded slowness field, at each iteration the FD method provides a set of calculated traveltimes

$$T_n^c = T_n^c(\mathbf{m}), \qquad n = 1, \cdots, N, \tag{6.26}$$

which are to be compared to the observed traveltimes, T_n^o (data). Here N is the total number of source-receiver pairs.

6.5 Nonlinear inversion

The traveltime inversion problem is cast as a nonlinear optimization problem. For this purpose, I define the cost function

$$\Phi(\mathbf{m}) = \frac{1}{N} \sum_{n=1}^{N} w_n |T_n^o - T_n^c(\mathbf{m})|^q \tag{6.27}$$

where w_n are weights. This equations expresses the misfit between the observed and calculated traveltimes. In general, $q = 2$, which leads to a standard weighted least-squares optimization. But other values for q can also be used. The objective is to minimize equation (6.27) with respect to $\mathbf{m}$, such that

$$\Phi(\mathbf{m}) \leq \Phi_{tol} = \chi^q, \tag{6.28}$$

where Φ_{tol} is a *tolerance* cost associated with the observational errors, and χ is the expected misfit. Note that χ has the same units as traveltimes. In general, an estimate of the right-hand side of equation (6.28) is available, so χ is a measure of the goodness-of-fit of the model to the observed data.

In addition to minimizing $\Phi(\mathbf{m})$, I specify a set of bounding constraints of the form

$$A_k \leq m_k \leq B_k, \qquad k = 1, \cdots, K \tag{6.29}$$

This is to avoid undesirable models that may lead to erroneous velocity fields (e.g. negative velocity values). Also, they may be used to specify some prior geophysical knowledge about the underlying model, and to "freeze" a certain model parameter by just setting $A_k = B_k$, for some k.

Since the calculated traveltimes T_n^c are nonlinearly related to the unknown parameters (spline nodes, anomaly parameters, etc), the optimization problem represents a constrained nonlinear inverse problem. Besides, cost function $\Phi(\mathbf{m})$ is expected to be multimodal. So, a method based on gradient directions may get trapped in local minima and the resulting model may be inaccurate. To find the global minimum of $\Phi(\mathbf{m})$, I carry out the minimization using VFSA.

6.5.1 Selection of VFSA parameters

The initial temperature for the annealing process has been selected as explained in Chapter 2. That is, all parameter temperatures are set equal to 1 for sampling the model space widely at the first stages. The acceptance temperature is set equal to the mean value of a number of cost function evaluations (ten by default) at randomly selected points within the search ranges. The search ranges, equation (6.29), are specified a priori for the model at hand in next section. The maximum number of iterations, which sets the main stopping condition, is set in all cases equal to 3000.

6.6 Numerical examples

In this section, I will illustrate the nonlinear procedure using both smooth and non-smooth 2-D velocity models. The first model represents a smooth velocity field with high and low velocity zones. I will use bicubic splines here for the inversion. The second model represents a high contrasting anomaly body immersed in a smooth background velocity.

Parametric 2-D function will be used here.

Traveltimes are generated using the FD method, and the data are contaminated with uniform random noise. The noise level is given as a percentage of the maximum traveltime for all source-receiver pairs. For the purpose of traveltime computations and plotting only, the velocity field is defined over a rectilinear grid with square cells. But during the inversion, the velocity field is represented either using bicubic splines or parametric 2-D functions, as explained in previous sections. Distances are given in *meters*, traveltimes in *milliseconds*, velocities in *kilometers per second*, and angles in *degrees*. The coordinates of the first grid node, i.e. node $(1,1)$, are $(x_{min}, z_{min}) = (0,0)$, and the size of each cell is 1×1. For simplicity, I use a 100×100 grid in all examples to compute traveltimes using the FD algorithm. So the size of the model is a square of 100 m in each dimension. The acquisition geometry is depicted in Figure 6.4. Both crosswell and well-to-surface (VSP) data were collected using: (1) 5 sources in each well and 5 receivers on the surface (VSP data), (2) 5 sources in the left well and 5 receivers in the right well (crosswell data). The experiment yielded a total of 75 traveltimes that were input, after being contaminated with random noise, to the inversion algorithm.

Regarding the SA inversion, the iteration stopped when the cost function reached the expected misfit (6.28), or after a maximum number of iterations (ITMAX=3000). In the simulations below, data are contaminated with uniform random noise with zero mean and amplitude $\pm b$, where b is a percentage of the maximum observed traveltime. Assuming all weights are equal to one in equation (6.27), the expected misfit reduces to

$$\chi = \frac{b}{(q+1)^{1/q}}. \tag{6.30}$$

For least-squares optimization, set $q = 2$, then we obtain the familiar quantity

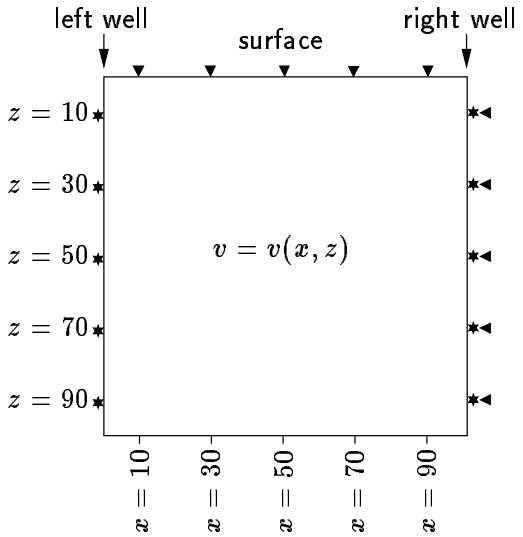

Figure 6.4: Acquisition geometry for the tomographic problem. When the left well is exploited, data are collected on the surface and the right well receivers. When the right well is exploited, data are collected on the surface only. This experiment produces a total of 75 traveltimes.

$$\chi = \frac{b}{\sqrt{3}}. \tag{6.31}$$

The nonlinear inversions are compared with the linearizing inversions. This method basically solves equation (6.3) using the LSQR algorithm [PS82], including suitable model constraints to stabilize the solution [Ald92]. Normally, a few iterations are needed for convergence to the expected misfit. The initial model is in all cases the constant velocity that minimizes the *rms* traveltime residual.

6.6.1 Smooth model

Figure 6.5a shows the smooth velocity field, SPLIN, used to test the nonlinear traveltime inversion method using bicubic splines. Basically, it consists of a low velocity blob zone contrasting the background velocity in about -20%. The background velocity is given by

$v(x,z) = 2.0 - 0.0015x + 0.002z$. The lowest velocity is about 1.8 km/s, and the highest is about 2.2 km/s. The blob was generated using a two-dimensional exponential function with elliptical contours centered at $(30, 40)$ and rotated 10 degrees counterclockwise. Raypaths for this model are shown in Figure 6.5b. The VSP and crosswell traveltimes were contaminated with 0.4% uniform random noise, as described above. Setting $q = 2$ in formula (6.30), along with $b = \pm 0.26$ ms (0.4% of the maximum traveltime), the expected misfit is

$$\chi \simeq 0.15 \quad \text{ms.} \tag{6.32}$$

For the inversion, I chose a spline grid with $M_x = M_z = 3$, and no background velocity. This gives a total of $9 + 2 = 11$ unknown parameters, including the two coordinates for the central nodes. The search ranges for all the parameters, as well as the results of the inversion, are shown in Table 6.1. A total of 20 independent realizations were run using different seeds, and a maximum number of iterations, ITMAX, equal to 3000. Figure 6.5c shows the mean nonlinearizing inversion, indicating a close match with the actual model. In practice, this is the model obtained using the mean values shown in Table 6.1. The small circles show the mean location of the spline nodes. A mean final misfit of 0.19 ms was achieved after 3000 annealing iterations, approximately, as shown in Figure 6.6 (left panel). Similar results are obtained using the linearizing inversion (see Figure 6.5e). In this case the expected misfit, 0.15 ms, was achieved after 4 iterations. Both inverted models are compared with the true model in Figure 6.5a by computing the differential models shown in Figures 6.5d and 6.5f. These images are obtained using

$$e(i,j) = \frac{|v(i,j) - \hat{v}(i,j)|}{v(i,j)} \times 100(\%), \tag{6.33}$$

SPLIN					ANOMA					
m_k	mean	σ_k	A_k	B_k	m_k	true	mean	σ_k	A_k	B_k
v_1	2.02	0.01	1.4	2.6	A	0.50	0.49	0.05	0.0	1.0
v_2	1.94	0.03	1.4	2.6	x_c	40.00	40.07	0.96	10.0	90.0
v_3	1.86	0.01	1.4	2.6	z_c	40.00	39.84	0.54	10.0	90.0
v_4	1.93	0.01	1.4	2.6	R_x	15.00	11.97	2.30	0.0	50.0
v_5	1.85	0.01	1.4	2.6	R_z	25.00	19.73	4.10	0.0	50.0
v_6	1.91	0.02	1.4	2.6	p	4.00	6.02	2.51	1.0	10.0
v_7	2.18	0.02	1.4	2.6	ρ	0.00	0.26	0.21	0.0	1.0
v_8	2.19	0.02	1.4	2.6	ω	60.00	60.48	5.01	0.0	90.0
v_9	2.04	0.02	1.4	2.6	v_0	2.00	1.99	0.01	1.0	3.0
x_2	36.3	8.7	20.0	50.0	g_x	0.00	0.01	0.00	-0.1	0.1
z_2	33.6	6.7	20.0	50.0	g_z	0.00	-0.01	0.00	-0.1	0.1
$\bar{e}$	0.67	0.10	-	-	$\bar{e}$	-	1.31	0.38	-	-
χ	0.19	0.02	-	-	χ	-	0.37	0.03	-	-

Table 6.1: Estimated model parameters after 3000 iterations (20 realizations). The search ranges are specified by (A_k, B_k). Last two rows show the average differential model and final misfit mean and standard deviation values, respectively.

where $v(i,j)$ is the true image, and $\hat{v}(i,j)$ is the result of the inversion. All errors in the nonlinear inversion are below 2%, and in the linearizing inversion they are below 3.5%.

The results shown in Table 6.1 reveal that the velocity values at the spline nodes are well determined, in the sense that they show low variances. On the contrary, the relatively large variances of the coordinates of the central nodes, indicate that the traveltimes are not so sensitive to variations in the grid structure, but on the node values.

6.6.2 Anomaly body

The traveltime inversion procedure is also tested using a high contrasting anomaly body, ANOMA, which is illustrated in Figure 6.7a. A buried structure is submerged in a

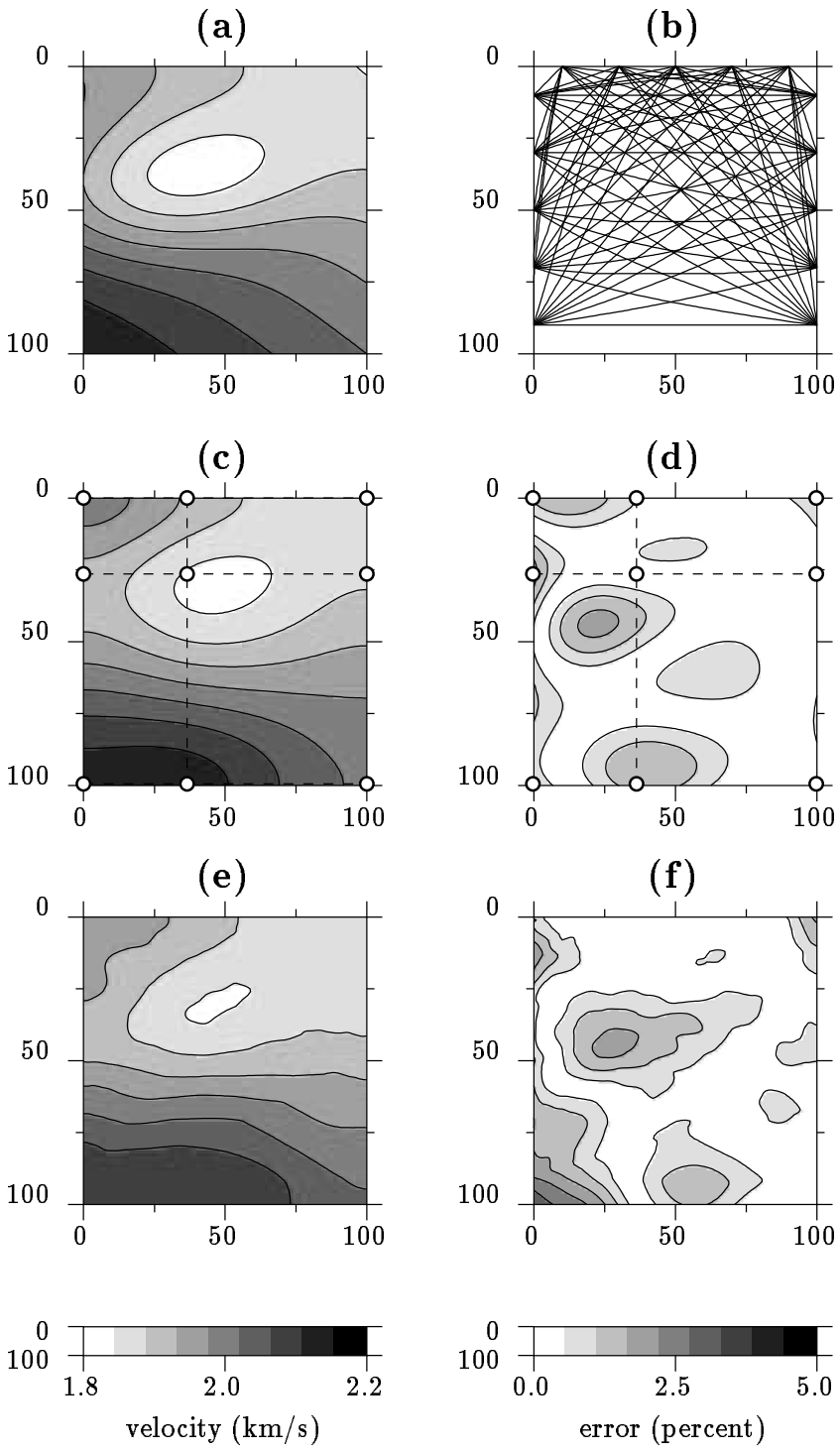

Figure 6.5: Traveltime inversion for the smooth model. (a) True model. (b) Raypath coverage. (c) SA inversion. (d) SA differential model. (e) Linearizing inversion. (f) Linearizing differential model.

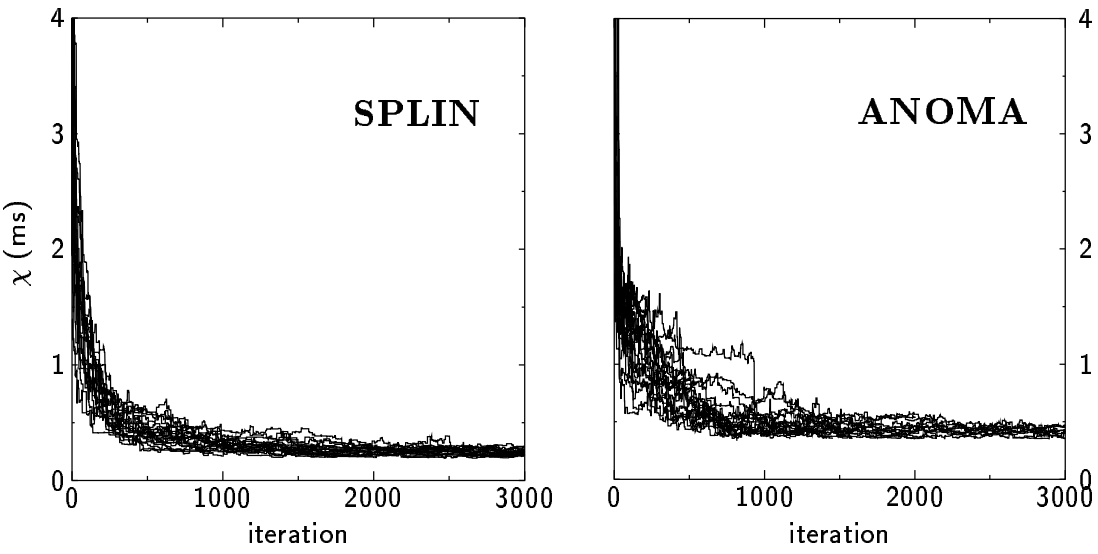

Figure 6.6: SART convergence after 3000 iterations (20 realizations).

constant velocity background field. The velocity of the anomaly is 2.5 km/s and the background velocity is 2.0 km/s. Table 6.1 summarizes the various parameters I used to build this model using the parametric functions described in previous section, along with the search ranges utilized by the SA inversion. Figure 6.7b shows the raypath coverage after exploiting all the sources. Traveltimes were then contaminated with 1% random noise, with $b = \pm 0.61$. According to formula (6.30), with $q = 2$, the expected misfit for this experiment is

$$\chi \simeq 0.35 \quad \text{ms.} \tag{6.34}$$

As with the SPLIN model, a total of 20 independent realizations were run using different seeds, with ITMAX=3000. The expected misfit was achieved well before 3000 iterations in about 50% of the cases. The convergence curves are illustrated in Figure 6.6 (right panel). Figure 6.7c shows the mean inversion using the SA approach (i.e. using the

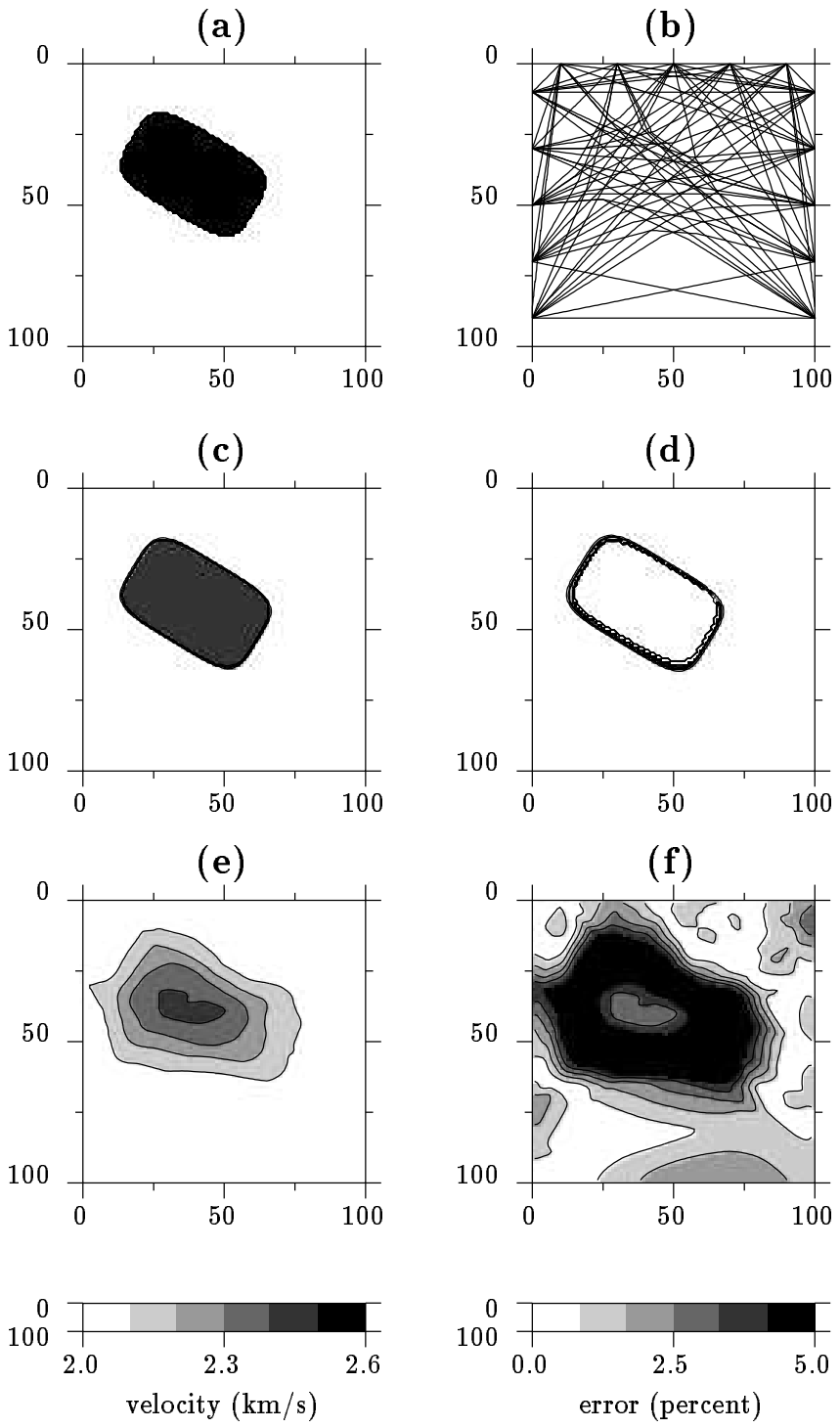

Figure 6.7: Traveltime inversion for the anomaly model. (a) True model. (b) Raypath coverage. (c) SA inversion. (d) SA differential model. (e) Linearizing inversion. (f) Linearizing differential model.

mean parameters values after the 20 realizations). It can be seen that the true model was recovered quite accurately. Figure 6.7e shows the inversion using the linearizing approach. The linearized solution also achieved the expected misfit and converged in about 5 iterations. Figures 6.7d and 6.7f show the differential models between true and reconstructed models. All values of the two differential models fall within 5%. However, the linearized inversion introduced some artifacts not present in the original model. The basic shape of the anomaly was recovered pretty well, but the sharp boundaries of the anomaly body could not be recovered, as expected. On the contrary, the SA inversion recovered the original model very accurately, with most errors below 1%. Though the linearizing approach imposes smoothness and is not appropriate for the inversion of this type of models, its results were shown for illustrative purposes.

6.7 Conclusions

I have demonstrated the ability of the described traveltime tomography procedure for velocity imaging. The results of the inversion using synthetic data are in very good agreement with the model. The method makes use of either bicubic B-splines defined over a grid with nonuniform spacing, or parametric 2-D functions to model the velocity field.

The parameterization is intended to reduce the number of unknowns and to stabilize the inversion. In the case of bicubic splines, smoothing is achieved beforehand. Besides, the grid spacing is dynamically adapted to accommodate moderately complex structures, leaving only a few unknown parameters for the optimization algorithm to work with. As a counterpart, the approach that utilizes parametric functions does not favor neither smooth nor high contrasting velocity distributions. This information is extracted from the data, where this information exists.

The traveltime inverse problem is cast as a constraint nonlinear optimization problem, which is solved by means of VFSA in an attempt to find the global minimum regardless the initial model. This strategy requires no ray tracing and problems related to: (1) initial model, and (2) selection of a proper regularization in the standard inversion procedure, are avoided. The selection of an adequate parameterization is very important, however, and the success of the tomography inversion resides not only in the ability of the parameterization to accommodate the subsurface geology, but also in the amount of data available and ray coverage.

The main drawback of the traveltime inversion problem presented in this chapter relies perhaps in the fact that it is a time consuming process. Even though fast finite-differences are used to compute the traveltimes corresponding to each source-receiver pair, the computational cost is very expensive since a large number of iterations are required for convergence to a near-optimal solution. As compared to linearizing methods, which usually require just a few iterations (3-10), the SA approach requires a much large number (1000-3000). For the size of the models used here, this represented a few minutes in a Sun Ultra 1 workstation. Nevertheless, the parameterization that is used to represent the velocity field cannot be inverted satisfactorily using a linearizing method, for there exists a large number of hidden local minima that inhibit the convergence to a useful solution.

The methodology is especially suited for those situations where a small number of parameters is able to represent the subsurface model. In the case of parametric functions, the method may find application in near surface studies, geoarchaeology, etc., provided the buried structure can be represented adequately with the parametric functions that have been described. The bicubic representation is appropriate, on the contrary, for smooth models only. The extension to 3-D situations may be readily implemented.

Chapter 7

Summary

It is venturesome to think that a coordination of words (philosophies are nothing more than that) can resemble the universe very much. It is also venturesome to think that of all these illustrious coordinations, one of them – at least in an infinitesimal way – does not resemble the universe a bit more than the others.

Jorge Luis Borges – *The Avatars of the Tortoise*

This thesis contributes to knowledge in three fundamental areas of seismic processing and inversion analysis: wavelet estimation, two-point ray tracing, and traveltime tomography. These problems are viewed as nonlinear optimization problems, where some misfit between the observed and calculated data, or some other functional of the model parameters, must be minimized (cost function). The *leitmotiv* of the thesis is centered on solving these hard optimization problems by means of SA, to avoid local minima, and to improve the accuracy and reliability of the results.

The problem of wavelet estimation based on the cumulant matching approach was treated in Chapter 3. Based on the convolutional model, it was assumed that the reflectivity series is non-Gaussian, and that the additive noise is normally distributed. Except for stationarity, there were no assumptions regarding the wavelet phase. Several improvements over previous works were proposed. Special effort was put on those issues concerning amount of data and reliability of the numerical solution. It was shown that the cumulant matching leads to an optimization problem where a good initial model is required for obtaining reliable wavelet estimates, if a linearizing algorithm to solve the optimization problem is utilized. This justified the use of SA. Since the amount of data

available plays a key role, a convenient multidimensional tapering to reduce the trace cumulant estimate variance was recommended. As a result, the algorithm could be used with confidence using relatively short data segments, aiming at a trace-by-trace process. An heuristic analysis of cumulant sensitivity to wavelet bandwidth was also performed. It was verified that, like kurtosis and MED-type deconvolution algorithms, the success of the cumulant matching method relies on the wavelet bandwidth. Finally, a hybrid strategy that combines SA with standard linearizing algorithms was proposed to alleviate the computational overburden associated with the optimization problem. The behavior of the method was explored by using several non-Gaussian reflectivity distributions to build the synthetic traces, and by taking different amounts of data. Real data was also utilized to test the method. The results demonstrated not only the viability of the cumulant matching method, but also the reliability of the convolutional model, which is at the heart of wavelet estimation philosophy.

Chapter 4 introduced a novel method for solving the boundary value ray-tracing problem in laterally varying 2-D media. The method was called SART, that stands for simulated annealing ray tracing. The motivation of SART was to overcome some difficulties arising in standard ray-tracing techniques (e.g. shooting and bending). The ray-tracing problem was put into a nonlinear optimization framework, where the take-off angle which produces the ray connecting two fixed points with absolute minimum traveltime was to be found. The method focused on direct arrivals, since it is based on a shooting procedure, but other phases could also be incorporated (e.g. diffractions) by constraining the raypath at intermediate points of its trajectory. It was not possible, however, to find all the raypaths connecting source and receiver for a given velocity model, nor to trace arbitrary headwaves and phases traveling through shadow zones, unless the raypath signature is totally specified a priori. This apparent disadvantage, which is shared with shooting methods, implied that the phase of the raypath being

traced was always known, a priori and a posteriori. This issue is very important from the point of view of phase identification. All the equations for ray tracing in a gridded cell velocity model were well developed. It was possible to add an arbitrary discontinuity to represent a geologic horizon. This allowed one to trace not only direct waves, but also reflections and simple headwaves. The method was illustrated using various numerical examples. These included multipathing and other nonlinearities which are difficult to handle using shooting or bending. In all the cases, a fast convergence to the global minimum was obtained.

SART philosophy was extended for dealing with complex 3-D models in Chapter 5. Not less important, in this chapter I also developed a complete model representation in terms of a number of blocks or regions that comprise the velocity field. Each region was assigned an arbitrary velocity field in the form of a three dimensional function. Block boundaries were given by arbitrary curve interfaces. SART extension to 3-D is based on the numerical solution of the ray equations, which were solved by means of standard ODE numerical solvers. All necessary equations to solve the initial-value problem, including how to find the intercept points where the ray meets block boundaries, were presented. It was shown how the boundary-value problem can be viewed as an unconstrained non-linear optimization problem, where a few unknown parameters were to be obtained. In general, these were the take-off angles which gave rise to the absolute minimum travel-time trajectory. In the case of headwaves, for example, the unknowns were represented by the coordinates the ray enters and leaves a predefined interface. Various alternative strategies were explored to solve the optimization problem, depending on the type of ray being traced (i.e. direct waves, reflected waves, etc.). It was found convenient to refine the solution at the later stages of the SA optimization (hybridization). Here the optimization problem was redefined in terms of the coordinates of the emerging point, and efficient linearizing methods were utilized. Several numerical examples were devised

to test the method under difficult situations, indicating that the algorithm is reasonably robust. The models included complex structures that lead to extremely ill-behaved cost functions. Finally, an extensive analysis and discussion on SART performance and accuracy was also performed. Several methods for improving SART efficiency were proposed. For example, it was shown that a *dynamic* integration was much efficient than using the same numerical solver for all blocks, without loosing accuracy.

Application of SA to the traveltime tomography problem without rays is treated in Chapter 6. The iterative algorithm is based on two special parameterization schemes. The solution to the inverse problem was stabilized by virtue of the reduction in the number of unknown parameters that define the velocity field, rather than imposing model constraints through regularization. A bicubic spline parameterization with adaptive grid spacing was proposed for dealing with smooth velocity fields. For models that include anomaly-like bodies, such as those of interest in archaeology and other near-surface studies, a versatile parametric scheme based on 2-D functions was presented in full detail. This novel scheme allowed one to model either smooth or high contrasting velocity structures with sharp boundaries, including elliptical, circular, or rectangular bodies, immersed in a smooth velocity background. Numerical examples using both crosswell and VSP data, demonstrated that the method can be used to successfully image complex structures without the need to regularize in the standard fashion, provided the velocity field can be represented using one of the two mentioned parameterization schemes.

7.1 Future research

This work leaves open several areas of research. First, developing the cumulant matching approach in the frequency domain would provide interesting insights in the wavelet estimation problem using higher-order statistics. Different polyspectra may be combined

into a single cost function to optimally extract their information (e.g. second-order statistics may be used for magnitude recovery, and fourth-order statistics for phase recovery). The spectra may be parameterized to reduce the number of unknowns and to increase SA efficiency and accuracy. A comparison with polycepstra-based methods would bring more understanding to this field, as well.

Second, a parallelized approach to the boundary ray tracing problem would give a more quantitative idea of the potential of SART for dealing with several raypaths as it is usual in tomography applications. Methods based on graph theory have not yet been implemented for dealing with complex 3-D models. It would be interesting to test its applicability, from the point of view of efficiency and accuracy, as compared to SART.

Third, better controlled experiments in Chapter 6 would add more understanding to this area. Devising alternative parameterization schemes for dealing with nonregular anomaly structures would provide a more flexible approach for imaging buried complex bodies. Three-dimensional anomalies could also be represented by this technique, as well as three-dimensional smooth velocity fields based on cubic splines, or other interpolation methodology.

Finally, it has been assumed that VFSA is one of the most efficient SA methods available for dealing with nonlinear optimization problems. But other SA variants have emerged in the last few years and have shown to be very efficient in certain applications. Genetic algorithms, taboo search, and other rather unlikely methods are always attractive algorithms for global optimization. Few published reports make a fair comparison between them. Improving existing SA techniques would certainly help to develop more efficient algorithms for solving difficult nonlinear optimization problems arising in seismic exploration and other geophysical areas. Questions like "how many iterations are required to obtain the global minimum?", "how accurate are the results after a certain number of iterations?", "how much faster can a SA algorithm be made without affecting

the reliability of the results?", have no clear and definitive answer, for the results depend tremendously on the nonlinear inverse problem at hand. It is not only the so-called "curse of dimensionality" that plays a key role in SA optimization, but also the complexity of the cost function (e.g. number of local minima, roughness). These are questions that experience will resolve.

References

[AK90] E. Asakawa and T. Kawanaka. A new ray tracing method for seismic tomography. Technical report, 1990. (in Japanese).

[AK93] E. Asakawa and T. Kawanaka. Seismic ray tracing using linear traveltime interpolation. *Geop. Prosp.*, 41:99–111, 1993.

[Ald90] D.F. Aldridge. The Berlage wavelet. *Geophysics*, 55:1508–1511, 1990.

[Ald92] D.F. Aldridge. *Analysis and inversion of seismic refraction traveltimes.* PhD thesis, The University of British Columbia, Vancouver, Canada, April 1992.

[And72] R.S. Anderssen. In R.S. Anderssen, L.S. Jennings, and D.M. Ryan, editors, *Optimisation*, pages 27–34, St. Lucía, Australia, 1972. Univ. of Queensland Press.

[AO93] D.F. Aldridge and D.W. Oldenburg. Two dimensional tomography inversion with finite-difference traveltimes. *J. Seism. Expl.*, 2:257–274, 1993.

[AR80] K. Aki and P.G. Richards. *Quantitative seismology: theory and methods.* W.H. Freeman and Co., 1980.

[AV90] C.J. Ammon and J.E. Vidale. Seismic traveltime tomography using combinatorial optimization techniques. *Seis. Res. Lett.*, 61:39, 1990.

[AV93] C.J. Ammon and J.E. Vidale. Tomography without rays. *Bull. Seism. Soc. Am.*, 83:509–528, 1993.

[BB89] H. Bohr and S. Brunak. A traveling salesman approach to protein conformation. *Complex Sys.*, 3:9–28, 1989.

[BBC89] N.D. Bregman, R.C. Bailey, and C.H. Chapman. Crosshole seismic tomography. *Geophysics*, 54:200–215, 1989.

[BBK+87] H. Berbee, C. Boender, A. Rinnooy Kan, C. Scheffer, R. Smith, and J. Telgen. Hit-and-run algorithms for the identification of nonredundant linear inequalities. *Mathl. Programming*, 37:184–207, 1987.

[Ber91] J.G. Berryman. Nonlinear inversion and tomography (lecture notes). University of California, Lawrence Livermore National Laboratory, Livermore, CA 94550, October 1991.

[Bil94] S.D. Billings. Simulated annealing for earthquake location. *Geophys. J. Int.*, 118:680–692, 1994.

[BJS88] I.O Bohachevsky, M.E. Johnson, and M.L. Stein. Optimal deployment of missile interceptors. *Am. J. Math. Management Sci.*, 8(3 & 4):361–387, 1988.

[BMT+89] G. Bilbro, R. Mann, T.K.Miller, W.E. Snyder, D.E. Van der Bout, and M. White. Optimization by mean field annealing. In D. Touretzky, editor, *Advances in Neural Network Information Processing Systems*, pages 91–98. Morgan-Kaufman, San Mateo, CA, 1989.

[BPLT72] P. Bois, M. La Porte, M. Lavergne, and G. Thomas. Well-to-well seismic measurements. *Geophysics*, 37(3):471–480, 1972.

[BR67] D.R. Brillinger and M. Rosenblatt. Asymptotic theory of estimates of k-th order spectra. In B. Harris, editor, *Spectral Analysis of Time Series*, pages 153–188. John Wiley & Sons, Inc., 1967.

[Bri65] D.R. Brillinger. An introduction to polyspectra. *Ann. Math. Statist.*, 36:1351–1374, 1965.

[Cas82] B.R. Cassel. A method for calculating synthetic seismograms in laterally varying media. *Geophys. J. R. Astr. Soc.*, 69:339–354, 1982.

[Čer87] V. Červený. Ray-tracing algorithms in three-dimensional laterally varying layered structures. In *Seismic Tomography with applications in Global Seismology and Exploration Geophysics*, pages 99–133, Dordretcht, The Netherlands, 1987. E. Reidel.

[CH82] P. Chang and T. Hsieh. Constrained nonlinear optimization approaches to color-signal separation. *IEEE Trans. Image Processing*, 4:81–94, 1982.

[CH96] N. Cheng and L. House. Minimum traveltime calculation in 3-D graph theory. *Geophysics*, 61(6):1895–1898, 1996. Short note.

[CK95] D. Cvijović and J. Klinowski. Taboo search: An approach to the multiple minima problem. *Science*, 267:664–666, 1995.

[CSS94] R.K. Chunduru, M.K. Sen, and P.L. Stoffa. Resistivity inversion for 2-D geologic structures using very fast simulated annealing. In *64th Ann. Internat. Mtg., Soc. Expl. Geophys., Expanded Abstracts*, volume 94, pages 640–643, 1994.

[CSS96] R.K. Chunduru, M.K. Sen, and P.L. Stoffa. Development of efficient hybrid optimization methods for geophysical inversion. In *66th Ann. Internat. Mtg., Soc. Expl. Geophys., Expanded Abstracts*, volume 96, pages 1130–1133, Denver, Colorado, 1996.

[Dav87] L. Davis, editor. *Genetic algorithms and simulated annealing.* Morgan Kaufmann Publishers, Los Altos, CA, 1987.

[Eli65] V.A. Elissevnin. Analysis of rays propagating in an inhomogeneous medium. *Sov. Phys. Acoust.*, 10:242–245, 1965. English translation.

[Fes17] R.A. Fessenden. Method and apparatus for locating ore-bodies. U.S. patent 1.240,328, 1917.

[Fir87] P. Firbas. *Tomography from seismic profiles*, chapter 8, pages 189–202. In Nolet [Nol87], 1987.

[FK94] B.C. Fish and T. Kusuma. A neural network approach to automate velocity picking. In *64th Ann. Internat. Mtg., Soc. Expl. Geophys., Expanded Abstracts*, volume 94, pages 185–188, 1994.

[FL93] R. Fisher and J.M. Lees. Shortest path ray tracing with sparse graphs. *Geophysics*, 58:987–996, 1993.

[Fog93] D.B. Fogel. On the philosophical differences between genetic algorithms and evolutionary algorithms. In D. B. Fogel and W. Atmar, editors, *Proc. of the Sec. Ann. Conf. on Evolutionary Programming*, pages 23–29, La Jolla, CA, 1993. Evolutionary Programming Society.

[Fog94] D.B. Fogel. An introduction to evolutionary optimization. *IEEE trans. on Neural Networks*, 5(1):3–14, 1994.

[FT95] C.A.C. Filho and F.A.C. Thedy. Traçado de raios através de algoritmos genéticos. In *1st Latin American Geophysical Union Conference, Expanded Abstracts*, pages 339–343, 1995. With abstract in English.

[GB93] S. Geoltrain and J. Brac. Can we image complex structures with first-arrival traveltimes? *Geophysics*, 58:564–575, 1993.

[GFR94] W.L. Goffe, G.D. Ferrier, and J. Rogers. Global optimization of statistical functions with simulated annealing. *J. Econometrics*, 60(1/2):65–100, 1994. (http://emlab.berkely.edu/Software).

[GG84] S. Geman and D. Geman. Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. *IEEE Trans. Pattern Analysis and Machine Intelligence*, PAMI-6:721–741, 1984.

[Gol89] D. E. Goldberg, editor. *Genetics algorithms in search, optimization, and machine learning.* Addisson-Wesley Publishing Company, Inc., 1989. The University of Alabama.

[GR81] R. Godfrey and F. Rocca. Zero memory nonlinear deconvolution. *Geophys. Prosp.*, 29:189–228, 1981.

[GS84] M.R. Green and J. Supowit. Simulated annealing without rejected moves. In *Digest International Conference on Computer Design*, pages 658–663, October 1984.

[Har94] N.D. Hargreaves. Wavelet estimation via fourth-order cumulants. In *64th Ann. Internat. Mtg., Soc. Expl. Geophys., Expanded Abstracts*, volume 94, pages 1588–1590, 1994.

[HL95] S.H. Hartzell and P.C. Liu. Determination of earthquake source parameters using a hybrid global search algorithm. *Bull. Seism. Soc. Am.*, 85:516–524, 1995.

[Ing89] L. Ingber. Very fast simulated re-annealing. *Mathl. Comput. Modelling*, 12:967–973, 1989.

[Ing93] L. Ingber. Simulated annealing: practice versus theory. *Mathl. Comput. Modelling*, 18:29–57, 1993.

[Ing96] L. Ingber. Adaptive simulated annealing (ASA): Lessons learned. *Control and Cybernetics*, 25(1):33–54, 1996.

[IR92] L. Ingber and B. Rosen. Genetic algorithms and very fast simulated reannealing: a comparison. *Mathl. Comput. Modelling*, 16:87–100, 1992.

[IRS88] Y. Ishii, S. Rokugawa, and T. Susuki. On the various subsurface models and ART methods. In *Proceedings 1988 Spring Meeting, SEG Japan*, pages 23–26, 1988. (in Japanese).

[JG77] B.R. Julian and D. Gubbins. Three-dimensional seismic ray tracing. *J. Geophys.*, 43:95–113, 1977.

[KGV83] S. Kirkpatrick, C.D. Jr. Gellat, and M.P. Vecchi. Optimization by simulated annealing. *Science*, 220:671–680, 1983.

[KM78] J. Kormylo and J.M. Mendel. On maximum-likelihood detection and estimation of reflection coefficients. In *48th Ann. Internat. Mtg., Soc. Expl. Geophys., Reprints*, volume 78, 1978.

[Lan92] F. Lane. *Estimation of kinematic rupture parameters from historical seismograms: an application of simulated annealing to a nonlinear optimization problem.* PhD thesis, Colorado School of Mines, Golden, Colorado, 1992.

[Laz93] G.L. Lazear. Mixed-phase wavelet estimation using fourth-order cumulants. *Geophysics*, 7:1042–1051, 1993.

[LBT89] E. Landa, W. Beydoun, and A. Tarantola. Reference velocity model estimation from prestack waveforms: Coherency optimization by simulated annealing. *Geophysics*, 54:984–990, 1989.

[LHS95] P. Liu, S. Hartzell, and W. Stephenson. Nonlinear multiparameter inversion using a hybrid global search algorithm: Applications in reflection seismology. *Geophys. J. Int.*, 122:991–1000, 1995.

[LLC85] R.T. Langan, I. Lerche, and R.T. Cutler. Tracing of rays through heterogeneous media: An accurate and efficient procedure. *Geophysics*, 50:1456–1465, 1985.

[LO87] S. Levy and D.W. Oldenburg. Automatic phase correction of common-midpoint stacked data. *Geophysics*, 52:51–59, 1987.

[LR82] K.S. Lii and M. Rosenblatt. Deconvolution and estimation of transfer function phase and coefficients for non-Gaussian linear processes. *Ann. Statist.*, 10:1195–1208, 1982.

[LS86] P. Lancaster and K. Salkauskas. *Curve and surface fitting, an introduction.* Academic Press, New York, 1986.

[Men91] J.M. Mendel. Tutorial on higher-order statistics in signal processing and system theory: Theoretical results and some applications. In *Proc. IEEE*, volume 79, pages 278–305, 1991.

[Mic95] A. Michelini. An adaptive-grid formalism for traveltime tomography. *Geophys. J. Int.*, 121:489–510, April 1995.

[MJC89] J.L. Mallet, P. Jacquemin, and N. Cheimanoff. GOCAD project: Geometric modeling of complex geological surfaces. In *59th Annual Internat. Mtg., Soc. Expl. Geophys., Expanded Abstracts*, volume 89, page 126, 1989. (http://www.ensg.u-nancy.fr/GOCAD/gocad.html).

[MM91] A. Michelini and T.V. McEvilly. Seismological studies at Parkfield I. simultaneous inversion for velocity structure and hypocentres using cubic B-splines parameterization. *Bull. Seism. Soc. Am.*, 81:524–552, 1991.

[MNS92] T.J. Moser, G. Nolet, and R. Snieder. Ray bending revisited. *Bull. Seis. Soc. Am.*, 82:259–288, 1992.

[Mos89] T.J. Moser. Efficient seismic ray tracing using graph theory. In *59th Ann. Internat. Mtg., Soc. Expl. Geophys., Expanded Abstracts*, pages 1106–1108, Dallas, 1989.

[Mos91] T.J. Moser. Shortest path calculation of seismic rays. *Geophysics*, 56:59–67, 1991.

[MRR+53] N. Metropolis, A.W. Rosenbluth, M.N. Rosenbluth, A.H. Teller, and E. Teller. Equation of state calculations by fast computing machines. *J. Chem. Phys.*, 21:1087–1092, 1953.

[MS97] W. Mao and G.W. Stuart. Rapid multi-wave-type ray tracing in complex 2-D and 3-D isotropic media. *Geophysics*, 62:298–308, 1997.

[MU84] T. Matsuoka and T.J. Ulrych. Phase estimation using the bispectrum. In *Proc. of IEEE*, volume 72, pages 1403–1411, October 1984.

[MVC93] G. Mirkin, K. Vasudevan, and F.A. Cook. A comparison of several cooling

schedules for simulated annealing implemented on a residual statics problem. *Geophys. Res. Lett.*, 20(1):77–80, 1993.

[Nol87] G. Nolet, editor. *Seismic Tomography with Applications in Global Seismology and Exploration Geophysics*, Dordretcht, The Netherlands, 1987. D. Reidel.

[NP93] C.L. Nikias and A.P. Petropulu. *Higher-order spectral analysis: A nonlinear signal processing framework.* Prentice-Hall, Inc., 1993.

[NR87] C.L. Nikias and M.R. Raghuveer. Bispectrum estimation: A digital signal processing framework. In *Proc. IEEE*, volume 75, pages 869–891, 1987.

[Per92] V. Pereyra. Two-point ray tracing in general 3-D media. *Geophys. Prosp.*, 40:267–287, 1992.

[PIIF92] L.A. Pflug, G.E. Ioup, J.W. Ioup, and R.L. Field. Properties of higher-order correlations and spectra for bandlimited, deterministic transients. *J. Acoust. Soc. Am.*, 92:975–988, 1992.

[PL91] P. Podvin and I. Lecomte. 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.

[PL93] S.K. Pullammanappallil and J.N. Louie. Inversion of seismic reflection traveltimes using a nonlinear optimization scheme. *Geophysics*, 58:1607–1620, 1993.

[PL97] S.K. Pullammanappallil and J.N. Louie. A combined first-arrival traveltime and reflection coherency optimization approach to velocity estimation. *Geophys. Res. Lett.*, 24(5):511–514, 1997.

[PS82] C.C. Paige and M.A. Saunders. LSQR: an algorithm for sparse linear equations and sparse least squares. *ACM Trans. Math. Software*, 8:43–71, 1982.

[PTE88] W.A. Prothero, W.J. Taylor, and J.A. Eickemeyer. A fast, two-point, three-dimensional ray-tracing algorithm using a simple step search method. *Bull. Seis. Soc. Am.*, 78:1190–1198, 1988.

[PTVF92] W. H. Press, S.A. Teukolsky, W.T. Vetterling, and B.P. Flannery. *Numerical Recipes in FORTRAN: the Art of Scientific Computing.* Cambridge University Press, second edition, 1992.

[Ros92] B. Rosen. Function optimization based on advanced simulated annealing. In *IEEE Workshop on Physics and Computation*, pages 289–293, 1992.

[Rot85] D.H. Rothman. Nonlinear inversion, statistical mechanics, and residual estimation. *Geophysics*, 50:332–346, 1985.

[Rot86] D.H. Rothman. Automatic estimation of large residual statics corrections. *Geophysics*, 51:2784–2796, 1986.

[RR88] H. Ratschek and J. Rokne, editors. *New computer methods for global optimization.* Ellis Horwood Ltd, 1988.

[RS88] J. Ramanujan and P. Sadayappan. Optimization by neural networks. In *Proc. of the Int. Conf. on Neural Networks*, volume II, pages 325–332, 1988.

[RT80] E.A. Robinson and S. Treitel. *Geophysical signal analysis.* Prentice-Hall, Inc., Englewood Cliffs, NJ 07632, 1980.

[Rud87] W. Rudin. *Real and complex analysis.* McGraw-Hill, Inc., 1987.

[Sai89] H. Saito. Traveltimes and raypaths of first arrival seismic waves: computation method based on Huygens' principle. In *59th Ann. Internat. Mtg., Soc. Expl. Geophys., Expanded Abstracts*, pages 244–247, Dallas, 1989.

[SBM95] M. Sambridge, J. Braun, and H. McQueen. Geophysical parameterization and interpolation of irregular data using natural neighbors. *Geophys. J. Int.*, 122:837–857, 1995.

[SBS93] M.K. Sen, B.B. Bhattacharya, and P.L Stoffa. Nonlinear inversion of resistivity sounding data. *Geophysics*, 58:496–507, 1993.

[Sca85] L.E. Scales. *Introduction to nonlinear optimization.* Springer-Verlag New York Inc., 1985.

[SDG90] J.A. Scales, P Docherty, and A. Gersztenkorn. Regularisation of nonlinear inverse problems: imaging the near surface weathering layer. *Inverse Prob.*, 6:115–131, 1990.

[SH87] H. Szu and R. Hartley. Fast simulated annealing. *Phys. Lett. A.*, 122(3,4):157–162, 1987.

[Sha93] E. Shalev. Cubic B-splines: Strategies of translating a simple structure to B-spline parameterization. *Bull. Seism. Soc. Am.*, 83:1617–1627, 1993.

[SK90] M.S. Sambridge and B.L.N. Kennett. Boundary-value ray tracing in a heterogeneous medium: A simple and versatile algorithm. *Geophys. J. Int.*, 191:157–168, 1990.

[SS91] M.K. Sen and P.L. Stoffa. Nonlinear one-dimensional seismic waveform inversion using simulated annealing. *Geophysics*, 56:1624–1638, 1991.

[SS95] M.K. Sen and P.L. Stoffa. *Global optimization methods in geophysical inversion.* Elsevier Publications, 1995.

[Sun93] Y. Sun. Ray tracing in 3-D media by parameterized shooting. *Geophys. J. Int.*, 114:145–155, 1993.

[SVU98] M.D. Sacchi, D.R. Velis, and T.J. Ulrych. Non-minimum phase wavelet estimation using polycepstra. *Journal of Seismic Exploration*, 1998. (in press).

[Tho82] D.J. Thomson. Spectrum estimation and harmonic analysis. In *Proc. IEEE*, volume 70, pages 1055–1096, 1982.

[Tug87] J.K. Tugnait. Identification of linear stochastic systems via second- and fourth-order cumulant matching. *IEEE Trans. Info. Theory*, IT-33:393–407, 1987.

[UT87] J. Um and C. Thurber. A fast algorithm for two-point seismic ray tracing. *Bull. Seis. Soc. Am.*, 77:972–986, 1987.

[Vel96] D.R. Velis. Nonlinear traveltime optimization for ray tracing in complex 3-D media. In *V Congreso Argentino de Mecánica Computacional, MECOM '96*, volume 16, pages 53–61, Tucumán, Argentina, 1996. Asociación Argentina de Mecánica Computacional.

[Vel98] D.R. Velis. Nonlinear traveltime inversion: a parametric approach. In *68th Ann. Internat. Mtg., Soc. Expl. Geophys., Expanded Abstracts*, New Orleans, Louisiana, 1998.

[VF91] J. Virieux and V. Farra. Ray tracing in 3-D complex isotropic media: An analysis of the problem. *Geophysics*, 56(12):2057–2069, 1991.

[Vid88] J. Vidale. Finite-difference calculation of traveltimes. *Bull. Seis. Soc. Am.*, 78:2062–2076, 1988.

[vLA88] P.I.M. van Laarhoven and E.H.L. Aarts. *Simulated annealing: theory and applications*. D. Riedel, Dordrecht, 1988.

[VM96] R. Vinther and K Mosegaard. Seismic inversion through tabu search. *Geophys. Prosp.*, 44:555–570, 1996.

[VU95a] D.R. Velis and T.J. Ulrych. Improved wavelet estimation using fourth-order cumulants based on very fast simulated annealing. In *CSEG National Convention, Expanded Abstracts*, pages 143–144, Calgary, Canada, 1995.

[VU95b] D.R. Velis and T.J. Ulrych. Traveltime tomography using very fast simulated annealing. In *65th Annual Internat. Mtg., Soc. Expl. Geophys., Expanded Abstracts*, volume 95, pages 1055–1057, Houston, USA, 1995.

[VU96a] D.R. Velis and T.J. Ulrych. Simulated annealing two-point ray tracing. *Geophys. Res. Lett.*, 23:201–204, 1996.

[VU96b] D.R. Velis and T.J. Ulrych. Simulated annealing wavelet estimation via fourth-order cumulant matching. *Geophysics*, 61:1939–1948, 1996.

[Wal85] A.T. Walden. Non-Gaussian reflectivity, entropy, and deconvolution. *Geophysics*, 12:2862–2888, 1985.

[Wal90] A.T. Walden. Improved low-frequency decay estimation using the multitaper spectral analysis method. *Geophys. Prosp.*, 38:61–86, 1990.

[WH86] A.T. Walden and J.W.J. Hosken. The nature of the non-Gaussianity of

primary reflection coefficients and its significance for deconvolution. *Geophys. Prosp.*, 34:1038–1066, 1986.

[Whi84] S.R. White. Concepts of scale in simulated annealing. In *Proc. IEEE International Conference on Computer Design*, pages 646–651, Port Chester, 1984.

[Whi86] R.E. White. *Estimation problems on deconvolution: Deconvolution and inversion.* Blackwell Scientific Publications, Inc., 1986.

[WS92] Q. Wu and T.H. Sloane. CMOS leaf-cell design using simulated annealing. Technical report, Buckwell University, Lewisburg, PA, 1992.

[Yon93] S. Yonghe. Ray tracing in 3-D media by parameterized shooting. *Geophys. J. Int.*, 114:145–155, 1993.

Appendix A

Sufficient conditions for the global convergence of various SA algorithms

In this appendix, I give a heuristic demonstration showing that the given temperature schedule is a sufficient condition for the convergence to a global minimum (a more rigorous proof for the convergence of BA can be found in Geman and Geman [GG84]). For this purpose, it is enough to prove that the products of the probabilities of not generating a state x for all annealing times successive to k_0 is zero [SH87, Ing89]:

$$\prod_{k=k_0}^{\infty} (1 - g_k) = 0. \tag{A.1}$$

This means that all model states are given the chance of being sampled and checked for acceptance before the system is too cool to proceed. By taking logarithm of equation (A.1) and Taylor expanding in g_k,

$$\sum_{k=k_0}^{\infty} \ln(1 - g_k) = \sum_{k=k_0}^{\infty} -g_k + g_k^2/2 - g_k^3/3 + \cdots. \tag{A.2}$$

As a result, proving equation (A.1) is equivalent to prove that

$$\sum_{k=k_0}^{\infty} g_k = \infty. \tag{A.3}$$

In BA the generating pdf is a Gaussian distribution, and if T is chosen to be equation (2.6), for a sufficiently high T_0, equation (A.3) yields

$$\sum_{k=k_0}^{\infty} g_k \geq \sum_{k=k_0}^{\infty} e^{-\ln(1+k)} = \sum_{k=k_0}^{\infty} \frac{1}{(1+k)} = \infty. \tag{A.4}$$

In FA, where a Cauchy distribution is used to generate new model states, T can be lowered at a faster rate. In effect, replacing equation (2.8) into the corresponding generating function, for arbitrary T_0, equation (A.3) yields

$$\sum_{k=k_0}^{\infty} g_k = \frac{T_0}{\pi \Delta x^{M+1}} \sum_{k=k_0}^{\infty} \frac{1}{k} \frac{1}{\left[1 + \left(\frac{T_0}{k\Delta x}\right)^2\right]^{\frac{M+1}{2}}} \simeq \frac{T_0}{\pi \Delta x^{M+1}} \sum_{k=k_0}^{\infty} \frac{1}{k} = \infty. \tag{A.5}$$

In VFSA, a still faster cooling rate is allowed due to the nature of the generating pdf. Replacing equation (2.15) into the corresponding generating pdf, for arbitrary T_0, equation (A.3) becomes

$$\sum_{k=k_0}^{\infty} g_k \simeq \sum_{k=k_0}^{\infty} \prod_{i=1}^{M} \frac{1}{2c_i |y_i| k^{1/M}} \sim \sum_{k=k_0}^{\infty} \frac{1}{k} = \infty, \tag{A.6}$$

since

$$2(|y_i| + T_i) \ln(1 + 1/T_i) \simeq 2|y_i|(c_i k^{1/M} - \ln T_{0i}) \simeq 2c_i |y_i| k^{1/M} \tag{A.7}$$

as T_i becomes small for large k. It should be noted that the convergence proof is not affected by the value of c_i. When quenching is desired, equation (2.15) is replaced by equation (2.19) with $1 < Q \leq M$, and convergence is no longer guaranteed:

$$\sum_{k=k_0}^{\infty} g_k \simeq \sum_{k=k_0}^{\infty} \prod_{i=1}^{M} \frac{1}{2c_i |y_i| k^{Q/M}} \sim \sum_{k=k_0}^{\infty} \frac{1}{k^Q} \leq \infty. \tag{A.8}$$

Appendix B

Formulas required for cell ray tracing with piecewise constant velocity

In Chapter 4 I have described a method to solve the IVP through a velocity model that has been parameterized using rectangular cells with constant velocity. In particular I described how to trace each ray segment in the case that the first point lies on the left edge of any given cell. In this appendix I give all the required formulas to trace every possible ray segment. Tables B.1, B.2, B.3 and B.4 summarize the formulas for each particular case. The last row indicates the next set of formulas that must be used. Table B.5 shows the angles φ_a and φ_b that are used to decide which column of formulas must be used. Also, note the following definitions

$$\begin{cases} z_a = z_{min} + (j-1)\Delta z, \\ z_b = z_a + \Delta z, \\ x_a = x_{min} + (i-1)\Delta x, \\ x_b = x_a + \Delta x, \end{cases} \tag{B.1}$$

where x_{min} and z_{min} are the coordinates of node $(1,1)$.

LEFT	$0 < \theta_k \leq \varphi_a$	$\varphi_a < \theta_k \leq \varphi_b$	$\varphi_b < \theta_k < \pi$
x_{k+1}	$x_k + (z_b - z_k)\tan\vartheta$	x_b	$x_k + (z_k - z_a)\tan\vartheta$
z_{k+1}	z_b	$z_k + \Delta x \tan\vartheta$	z_a
θ_{k+1}	$\arcsin\left[\frac{v_b}{v_a}\sin\vartheta\right]$	$\frac{\pi}{2} - \arcsin\left[\frac{v_b}{v_a}\sin\vartheta\right]$	$\pi - \arcsin\left[\frac{v_2}{v_1}\sin\vartheta\right]$
t_k	$\frac{1}{v_a}\frac{x_{k+1}-x_k}{\sin\vartheta}$	$\frac{1}{v_a}\frac{z_{k+1}-z_k}{\sin\vartheta}$	$\frac{1}{v_a}\frac{x_{k+1}-x_k}{\sin\vartheta}$
ϑ	θ_k	$\frac{\pi}{2} - \theta_k$	$\pi - \theta_k$
next cell	$(i, j+1)$	$(i+1, j)$	$(i, j-1)$
next set	**TOP**	**LEFT**	**BOTTOM**

Table B.1: Shooting from the left edge of cell (i, j).

TOP	$0 < \theta_k < \varphi_a$ $\phi_b < \theta_k < 2\pi$	$\varphi_a \leq \theta_k < \frac{\pi}{2}$	$\frac{3}{2}\pi < \theta_k \leq \varphi_b$
x_{k+1}	$x_k + \Delta z \tan\vartheta_k$	x_b	x_a
z_{k+1}	z_b	$z_k + (x_b - x_k)\tan\vartheta$	$z_k + (x_k - x_a)\tan\vartheta$
θ_{k+1}	$\arcsin\left[\frac{v_b}{v_a}\sin\vartheta\right]$	$\frac{\pi}{2} - \arcsin\left[\frac{v_b}{v_a}\sin\vartheta\right]$	$\frac{3}{2}\pi + \arcsin\left[\frac{v_b}{v_a}\sin\vartheta\right]$
t_k	$\frac{1}{v_a}\frac{x_{k+1}-x_k}{\sin\vartheta}$	$\frac{1}{v_a}\frac{z_{k+1}-z_k}{\sin\vartheta}$	$\frac{1}{v_a}\frac{z_{k+1}-z_k}{\sin\vartheta}$
ϑ	θ_k	$\frac{\pi}{2} - \theta_k$	$\theta_k - \frac{3}{2}\pi$
next cell	$(i, j+1)$	$(i+1, j)$	$(i-1, j)$
next set	**TOP**	**LEFT**	**RIGHT**

Table B.2: Shooting from the top edge of cell (i, j).

RIGHT	$\pi < \theta_k \leq \varphi_a$	$\varphi_a < \theta_k \leq \varphi_b$	$\varphi_b < \theta_k < 2\pi$
x_{k+1}	$x_k - (z_k - z_a)\tan\vartheta$	x_a	$x_k - (z_b - z_k)\tan\vartheta$
z_{k+1}	z_a	$z_k + \Delta x \tan\vartheta$	z_b
θ_{k+1}	$\pi + \arcsin\left[\frac{v_b}{v_a}\sin\vartheta\right]$	$\frac{3}{2}\pi + \arcsin\left[\frac{v_b}{v_a}\sin\vartheta\right]$	$2\pi - \arcsin\left[\frac{v_b}{v_a}\sin\vartheta\right]$
t_k	$\frac{1}{v_a}\frac{x_k - x_{k+1}}{\sin\vartheta}$	$\frac{1}{v_a}\frac{z_{k+1} - z_k}{\sin\vartheta}$	$\frac{1}{v_a}\frac{x_k - x_{k+1}}{\sin\vartheta}$
ϑ	$\theta_k - \pi$	$\theta_k - \frac{3}{2}\pi$	$2\pi - \theta_k$
next cell	$(i, j-1)$	$(i-1, j)$	$(i, j+1)$
next set	**BOTTOM**	**RIGHT**	**TOP**

Table B.3: Shooting from the right edge of cell (i, j).

BOTTOM	$\frac{\pi}{2} < \theta_k \leq \varphi_a$	$\varphi_a < \theta_k \leq \varphi_b$	$\varphi_b < \theta_k < \frac{3}{2}\pi$
x_{k+1}	x_b	$x_k + \Delta z \tan\vartheta$	x_a
z_{k+1}	$z_k - (x_b - x_k)\tan\vartheta$	z_a	$z_k - (x_k - x_a)\tan\vartheta$
θ_{k+1}	$\frac{\pi}{2} + \arcsin\left[\frac{v_b}{v_a}\sin\vartheta\right]$	$\pi - \arcsin\left[\frac{v_b}{v_a}\sin\vartheta\right]$	$\frac{3}{2}\pi - \arcsin\left[\frac{v_b}{v_a}\sin\vartheta\right]$
t_k	$\frac{1}{v_a}\frac{z_k - z_{k+1}}{\sin\vartheta}$	$\frac{1}{v_a}\frac{x_{k+1} - x_k}{\sin\vartheta}$	$\frac{1}{v_a}\frac{z_b - z_{k+1}}{\sin\vartheta}$
ϑ	$\theta_k - \frac{\pi}{2}$	$\pi - \theta_k$	$\frac{3}{2}\pi - \theta_k$
next cell	$(i+1, j)$	$(i, j-1)$	$(i-1, j)$
next set	**LEFT**	**BOTTOM**	**RIGHT**

Table B.4: Shooting from the bottom edge of cell (i, j).

cell edge	φ_a	φ_b
LEFT	$\frac{\pi}{2} - \arctan\frac{z_b - z_k}{\Delta x}$	$\frac{\pi}{2} + \arctan\frac{z_k - z_a}{\Delta x}$
TOP	$\arctan\frac{x_b - x_k}{\Delta z}$	$2\pi - \arctan\frac{x_k - x_a}{\Delta z}$
RIGHT	$\frac{3}{2}\pi - \arctan\frac{z_k - z_a}{\Delta x}$	$\frac{3}{2}\pi + \arctan\frac{z_b - z_k}{\Delta x}$
BOTTOM	$\pi - \arctan\frac{x_b - x_k}{\Delta z}$	$\pi + \arctan\frac{x_k - x_a}{\Delta z}$

Table B.5: Angles φ_a and φ_b are used to determine on which cell edge the next point of the ray trajectory will lie on.

Appendix C

CPU time saving by using a different numerical integrator in SART

The reduction in computational cost by using a different, less accurate, numerical integrator in SART can be estimated as follows. Let TT_1 and TT_2 be the total computational time using Method 1 and Method 2 for integrating the differential equations with a fixed stepsize. Also let r_{ev} be the ratio of the number of evaluations per step of the right-hand side of equation (5.2) required by Method 2 and Method 1 respectively (e.g. if Method 2 is Euler, and Method 1 is fourth-order Runge-Kutta, then $r_{ev} = 1/4$). So the computational cost associated with subprocess INT for Method 2 is approximately equal to r_{ev} times INT_1, so I write

$$\begin{cases} TT_1 \simeq INT_1 \cdot TT_1 + TT_0 \\ TT_2 \simeq r_{ev} \cdot INT_1 \cdot TT_1 + TT_0 \end{cases} \tag{C.1}$$

where TT_0 is the computational time spent by all SART subprocesses except INT_1 (i.e. $TT_0 = TT_1 - INT_1$). Subtracting the above two expressions and dividing by TT_1, I get

$$\frac{TT_2}{TT_1} \simeq 1 - (1 - r_{ev}) INT_1. \tag{C.2}$$

In particular, for $r_{ev} = 1/4$, and $INT_1 = 64.7\%$ (first column, fifth row in Table 5.7), it yields

$$\frac{TT_2}{TT_1} = \frac{TT}{TT_{Euler}} \simeq 0.51.$$

Thus SART is expected to double the number of rays per second if one uses Euler instead of fourth-order Runge-Kutta for Model 1, with $\Delta t = 1.0$ ms. The ratios as computed using formula (C.2) when Method 2 is either Euler or second-order Runge-Kutta are shown in the last two rows of Table 5.7. Of course, in the case of the salt-dome model where some velocities are not constant, the results using Euler method are not as accurate as using fourth-order Runge-Kutta, especially for $\Delta t = 3.0$ ms. There is a trade-off, then, between accuracy and efficiency.