Lecture Notes on

Nonlinear Inversion and Tomography:

I. Borehole Seismic Tomography

Developed from a Series of Lectures by

James G. Berryman
University of California
Lawrence Livermore National Laboratory
Livermore, CA 94550

Originally Presented at

Earth Resources Laboratory
Massachusetts Institute of Technology

July 9–30, 1990

Revised and Expanded

October, 1991

Chapter 7

Nonlinear Seismic Inversion

The introduction of feasibility constraints into the traveltime inversion problem offers a unique opportunity to develop a variety of new reconstruction algorithms. A few of the ones that have been explored so far will be discussed in this Chapter.

7.1 Linear and Nonlinear Programming

We will see that linear tomography maps easily onto linear programming, and nonlinear tomography onto nonlinear programming [Strang, 1986; Fiacco and McCormick, 1990].

Recall that, if $\mathbf{u}^T = (1, \ldots, 1)$ is an m-vector of ones and $\mathbf{v}^T = (1, \ldots, 1)$ is an n-vector of ones, then

$$\mathbf{u}^T \mathbf{M} = \mathbf{v}^T \mathbf{C}, \tag{7.1}$$

where $\mathbf{C}$ is the coverage *matrix*, *i.e.*, the diagonal matrix whose diagonal elements are the column sums of the ray-path matrix. We will now define the coverage *vector* as

$$\mathbf{c} = \mathbf{C}\mathbf{v}. \tag{7.2}$$

7.1.1 Duality

The concept of duality in linear programming leads to some useful ideas both for linear and nonlinear traveltime inversion. (Actually it is even more useful for electrical impedance tomography as we will see in Part II.) We will first define the following:

Definition 7.1.1 *The primal problem for traveltime inversion is to find the minimum of* $\mathbf{c}^T\mathbf{s}$ *subject to* $\mathbf{M}\mathbf{s} \geq \mathbf{t}$ *and* $\mathbf{s} \geq 0$.

Definition 7.1.2 *The dual problem associated with the primal is to maximize* $\mathbf{w}^T\mathbf{t}$ *subject to* $\mathbf{w}^T\mathbf{M} \leq \mathbf{c}^T$ *and* $\mathbf{w} \geq 0$.

The m-vector $\mathbf{w}$ has no physical significance, but plays the role of a nonnegative weight vector. One of the first consequences of this formulation is that, if we multiply the primal

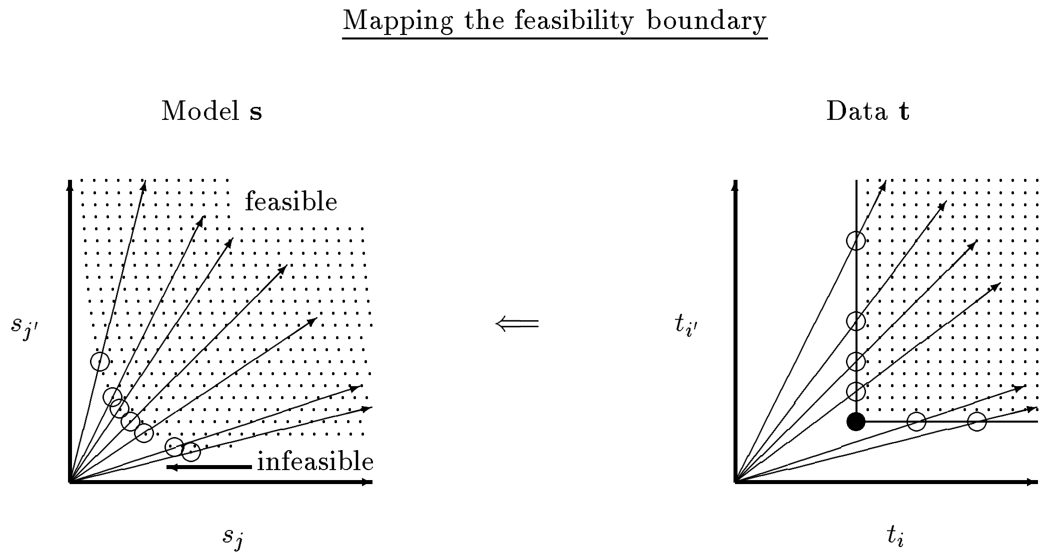


Figure 7.1: Mapping the feasibility boundary.

inequality on the right by $\mathbf{w}^T$ and the dual inequality on the left by $\mathbf{s}$ for feasible $\mathbf{s}$ and $\mathbf{w}$, then

$$\mathbf{c}^T\mathbf{s} \geq \mathbf{w}^T\mathbf{M}\mathbf{s} \geq \mathbf{w}^T\mathbf{t}. \tag{7.3}$$

We introduce a Lagrangian functional

$$\mathcal{L}(\mathbf{s},\mathbf{w}) = \mathbf{c}^T\mathbf{s} + \mathbf{w}^T(\mathbf{t} - \mathbf{M}\mathbf{s}) \tag{7.4}$$

$$= (\mathbf{c}^T - \mathbf{w}^T\mathbf{M})\mathbf{s} + \mathbf{w}^T\mathbf{t}. \tag{7.5}$$

An admissible (feasible) weight vector is $\mathbf{w} = \mathbf{u}$. In fact, this is the only weight vector we need to consider because it saturates the dual inequality, producing equality in all components following (7.1) and (7.2). Thus, the dual problem in traveltime inversion is completely trivial. We introduced it here because, despite its apparent triviality, there is one interesting feature.

In problems with nontrivial duality structure, it is possible to obtain useful bounds with inequalities equivalent to (7.3). Here we are left with just the condition

$$\boxed{\mathbf{c}^T\mathbf{s} \quad \geq \quad \mathbf{u}^T\mathbf{t} \quad = \quad T,} \tag{7.6}$$

defining the hyperplane of constant total traveltime. Equation (7.6) could have been derived directly from the feasibility conditions $\mathbf{M}\mathbf{s} \geq \mathbf{t}$ for $\mathbf{s}$. The ease of its derivation should not, however, lead us to think that this equation is trivial. This hyperplane can play an important role in linear and nonlinear programming algorithms for traveltime inversion.

7.1.2 Relaxed feasibility constraints

Given the set of observed traveltimes, t_i for $i = 1, \ldots, m$, we define two more types of feasibility sets.

DEFINITION 7.1.3 (RELAXED LOCAL FEASIBILITY SET) *The relaxed local feasibility set with respect to a set of trial ray paths $\mathcal{P} = \{P_1, \ldots, P_m\}$ and observed traveltimes $t_1, \ldots, t_m$ is*

$$\mathcal{R}^{\mathcal{P}} = \{\mathbf{s} \mid \sum_{i=1}^{m} \tau_i^P(\mathbf{s}) \geq \sum_{i=1}^{m} t_i\}. \tag{7.7}$$

DEFINITION 7.1.4 (RELAXED GLOBAL FEASIBILITY SET) *The relaxed global feasibility set with respect to a set of observed traveltimes $t_1, \ldots, t_m$ is*

$$\mathcal{R}^* = \{\mathbf{s} \mid \sum_{i=1}^{m} \tau_i^*(\mathbf{s}) \geq \sum_{i=1}^{m} t_i\}. \tag{7.8}$$

PROPOSITION 7.1.1 (SUM OF CONCAVE FUNCTIONS) *A (nonnegatively) weighted sum of concave functions is concave.*

Proof: Let $\tau_i(\mathbf{s})$ for $i = 1, \ldots, m$ be a set of concave functions and let w_i be a set of nonnegative weights. Then,

$$\sum_{i=1}^{m} w_i \tau_i(\lambda \mathbf{s}_1 + (1-\lambda)\mathbf{s}_2) \geq \sum_{i=1}^{m} w_i[\lambda \tau_i(\mathbf{s}_1) + (1-\lambda)\tau_i(\mathbf{s}_2)] \tag{7.9}$$

$$= \lambda \sum_{i=1}^{m} w_i \tau_i(\mathbf{s}_1) + (1-\lambda) \sum_{i=1}^{m} w_i \tau_i(\mathbf{s}_2), \tag{7.10}$$

so the weighted sum is concave. ∎

THEOREM 7.1.1 *$\mathcal{R}^{\mathcal{P}}$ is a convex set.*

THEOREM 7.1.2 *$\mathcal{R}^*$ is a convex set.*

Proof: Both theorems follow from the proposition and the fact that the unit-weighted sums in the definitions of the sets $\mathcal{R}^{\mathcal{P}}$ and $\mathcal{R}^*$ are respectively sums of the concave functions $\tau_i^P(\mathbf{s})$ and $\tau_i^*(\mathbf{s})$. ∎

THEOREM 7.1.3 *Any point $\mathbf{s}^*$ that lies simultaneously on the boundary of both $\mathcal{F}^{\mathcal{P}}$ and $\mathcal{R}^{\mathcal{P}}$ solves the inversion problem.*

THEOREM 7.1.4 *Any point $\mathbf{s}^*$ that lies simultaneously on the boundary of both $\mathcal{F}^*$ and $\mathcal{R}^*$ solves the inversion problem.*

Proof: The boundary of $\mathcal{R}^{\mathcal{P}}$ is determined by the single equality constraint

$$\sum_{i=1}^{m} \tau_i^P(\mathbf{s}) = \sum_{i=1}^{m} t_i = T. \tag{7.11}$$

The boundary of $\mathcal{F}^{\mathcal{P}}$ is determined by the set of inequality constraints

$$\tau_i^P(\mathbf{s}) \geq t_i, \quad \text{for all} \quad i = 1, \ldots, m, \tag{7.12}$$

with equality holding for at least one of the constraints. Summing (7.12) gives

$$\sum_{i=1}^{m} \tau_i^P(\mathbf{s}) \geq T, \tag{7.13}$$

where the equality applies if and only if $\tau_i^P(\mathbf{s}) = t_i$ for all i. Therefore, any model $\mathbf{s}^*$ that satisfies both (7.11) and (7.12) must solve the inversion problem.

The proof of the second theorem follows the proof of the first, with $\tau^*(\mathbf{s})$ replacing $\tau^P(\mathbf{s})$ everywhere. ▮

The boundaries of relaxed feasibility sets (either local or global) are easier to compute than those for unrelaxed feasibility sets. The difference is, for example, a single hyperplane boundary for a relaxed local feasibility set versus up to m (the number of traveltime measurements) hyperplanes composing the boundary of an unrelaxed local feasibility set. Yet, the characteristics of the relaxed feasibility sets are very similar to the unrelaxed ones in other ways.

If the correct ray-path matrix for the inversion problem has been found and the data are noise free, then we expect that the hyperplane defined by $\mathbf{c}^T\mathbf{s} = T$ will intersect the feasibility boundary exactly at the point or points that solve the inversion problem. If the correct ray-path matrix has not been found or there is uncorrelated noise in our data $\mathbf{t}$, then there will be a *splitting* between the hyperplane of constant total traveltime and the feasible region. The point (or points) of closest approach between the convex feasible set and the hyperplane may then be defined as the set of points *solving* the linear programming problem for fixed $\mathbf{M}$. An iterative nonlinear programming algorithm may then be constructed wherein the updated $\mathbf{M}$ is determined based on the solution of the last linear programming problem. This procedure converges if the degree of splitting (Euclidean distance) between the feasible set and the hyperplane of constant traveltime tends to zero from one iteration to the next.

Problem

PROBLEM 7.1.1 *Consider the following backprojection formulas (see* PROBLEMS *1.5.1 and 1.5.3):*

1. $\mathbf{s} = \mathbf{N}^{-1}\mathbf{H}^T\mathbf{L}^{-1}\mathbf{t}$;
2. $\mathbf{s} = \mathbf{C}^{-1}\mathbf{M}^T\mathbf{L}^{-1}\mathbf{t}$.

Does either formula always satisfy the constraint $\mathbf{c}^T\mathbf{s} = T$ *? Find another backprojection formula that does satisfy the constraint.*

7.2 More about Weighted Least-Squares

We learned in Sections 3.5 and 4.3 that a good set of weights for use with weighted least-squares was $\mathbf{L}^{-1}$ for the traveltime errors and $\mathbf{C}$ for the smoothing or regularization term in a damped least-squares method. The arguments were based on assumptions of small deviations from a constant background or on the desire to precondition the ray-path matrix so its eigenvalues λ were normalized to the range $-1 \le \lambda \le 1$.

In a sense the methods used to choose the weights previously were based on ideas of *linear inversion.* We should now try to see if these ideas need to be modified for *nonlinear inversion.* Let $\mathbf{s}$ be the latest estimate of the slowness model vector in an iterative inversion scheme. Then, if $\mathbf{u}^T = (1, \ldots, 1)$ is an m-vector of ones and $\mathbf{v}^T = (1, \ldots, 1)$ is an n-vector of ones,

$$\mathbf{Ms} = \mathbf{Tu}, \tag{7.14}$$

$$\mathbf{M}^T\mathbf{u} = \mathbf{Cv} \equiv \mathbf{Ds}, \tag{7.15}$$

where $\mathbf{C}$ is the coverage matrix (diagonal matrix containing the column sums of $\mathbf{M}$) defined previously and the two new matrices ($\mathbf{T}$ and $\mathbf{D}$) are diagonal matrices whose diagonal elements are T_{ii}, the estimated traveltime for the ith ray path through the model $\mathbf{s}$,

$$T_{ii} = \sum_{j=1}^{n} l_{ij} s_j, \tag{7.16}$$

and D_{jj} where

$$D_{jj} \equiv C_{jj}/s_j = \sum_{i=1}^{m} l_{ij}/s_j. \tag{7.17}$$

For the sake of argument, let the inverse of the diagonal traveltime matrix $\mathbf{T}^{-1}$ be the weight matrix, and compute the scaled least-squares point. The least-squares functional takes the form

$$\psi(\gamma) = (\mathbf{t} - \mathbf{M}\gamma\mathbf{s})^T\mathbf{T}^{-1}(\mathbf{t} - \mathbf{M}\gamma\mathbf{s}), \tag{7.18}$$

which has its minimum at

$$\gamma = \frac{\mathbf{s}^T\mathbf{M}^T\mathbf{T}^{-1}\mathbf{t}}{\mathbf{s}^T\mathbf{M}^T\mathbf{T}^{-1}\mathbf{Ms}}. \tag{7.19}$$

Equation (7.19) can be rewritten using (7.14) as

$$\gamma = \frac{\mathbf{u}^T\mathbf{t}}{\mathbf{u}^T\mathbf{Tu}}. \tag{7.20}$$

The factor γ that minimizes the least-squares error is therefore the one that either increases or decreases the total traveltime of the model $\mathbf{s}$ so it equals that of the data. If we assume that the measurement errors in the traveltime data $\mathbf{t}$ are unbiased, then it is very reasonable to choose models that have this property, because the total traveltime $\mathbf{u}^T\mathbf{t} = T$ will tend to have smaller error (by a factor of $m^{-1/2}$) than the individual measurements.

We see that requiring the models $\mathbf{s}$ to have the same total traveltime as the data is equivalent to requiring that the models all lie in the hyperplane defined by

$$\mathbf{u}^T\mathbf{M}\mathbf{s} = \mathbf{v}^T\mathbf{C}\mathbf{s} = \mathbf{c}^T\mathbf{s} = T. \tag{7.21}$$

But this is precisely the same hyperplane (7.6) that arose naturally in the earlier discussion of linear and nonlinear programming.

To carry this analysis one step further, consider the weighted least-squares problem

$$\phi_\mu(\mathbf{s}) = (\mathbf{t} - \mathbf{M}\mathbf{s})^T\mathbf{T}^{-1}(\mathbf{t} - \mathbf{M}\mathbf{s}) + \mu(\mathbf{s} - \mathbf{s}_0)^T\mathbf{D}(\mathbf{s} - \mathbf{s}_0), \tag{7.22}$$

where we assume that the starting model $\mathbf{s}_0$ satisfies $\mathbf{c}^T\mathbf{s}_0 = T$. Then, the minimum of (7.22) occurs for $\mathbf{s}_\mu$ satisfying

$$(\mathbf{M}^T\mathbf{T}^{-1}\mathbf{M} + \mu\mathbf{D})(\mathbf{s}_\mu - \mathbf{s}_0) = \mathbf{M}^T\mathbf{T}^{-1}(\mathbf{t} - \mathbf{M}\mathbf{s}_0). \tag{7.23}$$

Multiplying (7.23) on the left by $\mathbf{s}_0^T$, we find that

$$(1 + \mu)\mathbf{c}^T(\mathbf{s}_\mu - \mathbf{s}_0) = \mathbf{u}^T(\mathbf{t} - \mathbf{M}\mathbf{s}_0) = 0, \tag{7.24}$$

so the solution of the weighted least-squares problem (7.23) also has the property that its estimated total traveltime for all rays is equal to that of the data

$$\mathbf{c}^T\mathbf{s}_\mu = \mathbf{c}^T\mathbf{s}_0 = T. \tag{7.25}$$

Our conclusion is that the particular choice of weighted least-squares problem (7.23) has the *unique* property of holding the total estimated traveltime equal to the total of the measured traveltimes, *i.e.*, it constrains the least-squares solution to lie in the hyperplane $\mathbf{c}^T\mathbf{s} = T$. Assuming that the traveltime data are themselves unbiased (*i.e.*, $\mathbf{u}^T\Delta\mathbf{t} = 0$ where $\Delta\mathbf{t}$ is the measurement error vector), the result $\mathbf{s}$ is an unbiased estimator of the slowness. Moreover, this property is maintained for *any* value of the damping parameter μ. This result provides a connection between the linear programming approach and weighted linear least-squares. We can now use weighted least-squares and the formula (7.23) in a linear program as a means of moving around in the hyperplane $\mathbf{c}^T\mathbf{s} = T$.

Now, from our general analysis of the eigenvalue structure of weighted least-squares, recall that (4.108) shows, for $\mathbf{F} = \mathbf{T}$ and $\mathbf{G} = \mathbf{D}$, that we have

$$\frac{L_{ii}C_{jj}}{T_{ii}D_{jj}} \geq \lambda^2, \tag{7.26}$$

which must hold true for all values of i, j. From (7.17), we have $C_{jj}/D_{jj} = s_j$ so

$$\frac{L_{ii}s_j}{T_{ii}} \geq \frac{L_{ii}s_{\min}}{T_{ii}} \geq \lambda^2, \tag{7.27}$$

and from the definition of T_{ii} we have

$$T_{ii} = \sum_{j=1}^{n} l_{ij}s_j \geq L_{ii}s_{\min}. \tag{7.28}$$

We conclude that this choice of weight matrices also constrains the eigenvalues to be bounded above by unity, *i.e.*, $1 \geq \lambda^2$.

If the matrix $\mathbf{M}$ is very large, it may be impractical to solve (7.23) by inverting the matrix $(\mathbf{M}^T\mathbf{T}^{-1}\mathbf{M} + \mu\mathbf{D})$. Instead, we may choose to use some version of the method we called "simple iteration" in Section 4.4.3. For example, suppose that the kth iteration yields the model vector $\mathbf{s}_\mu^{(k)}$. Then, one choice of iteration scheme for finding the next iterate is

$$\mathbf{D}\mathbf{s}_\mu^{(k+1)} = \mathbf{D}\mathbf{s}_\mu^{(k)} + \mathbf{M}^T\mathbf{T}^{-1}(\mathbf{t} - \mathbf{M}\mathbf{s}_0) - (\mathbf{M}^T\mathbf{T}^{-1}\mathbf{M} + \mu\mathbf{D})(\mathbf{s}_\mu^{(k)} - \mathbf{s}_0). \tag{7.29}$$

It is not hard to show that this iteration scheme converges as long as the damping parameter is chosen so that $0 < \mu < 1$. [1] Furthermore, if we multiply (7.29) on the left by $\mathbf{s}_0^T$, we find that

$$\mathbf{c}^T(\mathbf{s}_\mu^{(k+1)} - \mathbf{s}_\mu^{(k)}) = (1+\mu)\mathbf{c}^T(\mathbf{s}_0 - \mathbf{s}_\mu^{(k)}). \tag{7.30}$$

It follows from (7.30) that, if $\mathbf{c}^T\mathbf{s}_0 = T$ and if $\mathbf{s}_\mu^{(0)} = \mathbf{s}_0$, then

$$\mathbf{c}^T\mathbf{s}_\mu^{(k)} = T, \tag{7.31}$$

for all k. Thus, all the iterates stay in the hyperplane of constant total traveltime. If we choose not to iterate to convergence, then this desirable feature of the exact solution $\mathbf{s}_\mu$ proven in (7.25) is still shared by every iterate $\mathbf{s}_\mu^{(k)}$ obtained using this scheme.

Problem

PROBLEM 7.2.1 *Use the definition of the pseudoinverse in* PROBLEM 4.1.16 *to show that, if*

$$\mathbf{M}' = \mathbf{T}^{-\frac{1}{2}}\mathbf{M}\mathbf{D}^{-\frac{1}{2}},$$

then

$$\mathbf{X} = \mathbf{D}^{-\frac{1}{2}}(\mathbf{M}')^\dagger\mathbf{T}^{-\frac{1}{2}}, \tag{7.32}$$

where $\mathbf{X}$ *is an approximate generalized inverse satisfying the first two conditions (*$\mathbf{MXM} = \mathbf{M}$ *and* $\mathbf{XMX} = \mathbf{X}$*). Use (7.32) to show that the SVD of* $\mathbf{X}$ *has the form*

$$\mathbf{X} = \frac{\mathbf{s}_0\mathbf{u}^T}{\mathbf{u}^T\mathbf{T}\mathbf{u}} + \ldots,$$

where the terms not shown are for eigenvectors of $\mathbf{M}'$ *with eigenvalues* $\lambda < 1$*. Apply this result to the inversion problem to show that*

$$\mathbf{s} \simeq \mathbf{X}\mathbf{t} = \mathbf{s}_0 + \ldots,$$

where in this case the terms not shown are contributions orthogonal to $\mathbf{s}_0$*.*

[1] The reader may want to check this result using the methods of this section and the ones developed in Section 4.4.3 on simple iteration.

7.3 Stable Algorithm for Nonlinear Crosswell Tomography

Here we combine several ideas from the previous sections into an algorithm for nonlinear traveltime tomography. We recall that such algorithms are inherently iterative. In the general iterative algorithm posed earlier, the questionable step was how to update the current model $\hat{\mathbf{s}}$ to obtain an improved model. Here we propose a method for choosing this step [Berryman, 1989b; 1990].

Let $\mathbf{s}^{(k)}$ be the current model. An algorithm (see Figure 7.2) for generating the updated model $\mathbf{s}^{(k+1)}$ is as follows:

1. Set $\mathbf{s}_1$ to the scaled least-squares model:
$$\mathbf{s}_1 = \hat{\mathbf{s}}_{\mathrm{LS}[\mathbf{s}^{(k)}]}.$$
2. Set $\mathbf{s}_2$ to the damped least-squares model with respect to $\mathbf{s}_1$:
$$\mathbf{s}_2 = \hat{\mathbf{s}}_{\mathrm{LS}[\mathbf{s}_1,\mu]}.$$
3. Define the family of models
$$\mathbf{s}(\lambda) = (1-\lambda)\mathbf{s}_1 + \lambda\mathbf{s}_2,$$
where $\lambda \in [0,1]$.
4. Solve for λ^*, defined so that $\mathbf{s}(\lambda^*)$ yields the fewest number of feasibility violations. The number of feasibility violations is defined as the number of ray paths for which $t_i > \tau^*(\mathbf{s}(\lambda))$.
5. If λ^* is less than some preset threshold (say 0.05 or 0.1), reset it to the threshold value.
6. Set $\mathbf{s}^{(k+1)} = \mathbf{s}(\lambda^*)$.

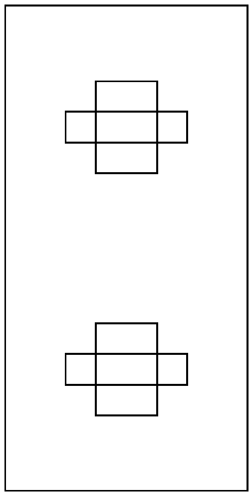
The feasibility structure of the algorithm is illustrated in Fig. 7.3. The model labeled $\mathbf{s}_3$ is a scaled version of $\mathbf{s}(\lambda^*)$, scaled so that $\mathbf{s}_3$ is on the boundary of the feasible region ($\mathcal{F}^*$). The iteration sequence stops when the perimeter of the triangle formed by $\mathbf{s}_1$, $\mathbf{s}_2$ and $\mathbf{s}_3$ drops below a prescribed threshold.

This algorithm has been tested on several problems both with real and with synthetic data and compared with a traditional damped least-squares algorithm (i.e., setting $\lambda^* = 1$ on each iteration). The new algorithm was found to be very stable and avoids the large oscillations in slowness often found in traditional least-squares methods.

Example 7.3.1 *Reconstructions were performed on models having* 16×8 *cells using 320 rays —including 256 rays (16 sources* $\times$ *16 receivers) from left to right and 64 rays (8 sources* $\times$ *8 receivers) from top to bottom. This measurement configuration was chosen to minimize the effects of ghosts, since the main focus of the exercise is to evaluate the usefulness of the feasibility constraints in stabilizing the algorithm.*

The traveltime data were generated with a bending method using the simplex search routine [Prothero et al., 1988; Nelder and Mead, 1965].

Three examples of typical results from the nonlinear algorithm BST (Borehole Seismic Tomography) are displayed on the following pages. The basic structure of the test problems has this form:



These slowness models have a low speed anomaly on top and a high speed anomaly on the bottom in each case. The first example has 20% *anomalies; the second has* 50% *anomalies; the third* 100% *anomalies.* EXAMPLE 7.3.1.1A *shows the target model, i.e., the model that used to generate the traveltime data;* EXAMPLE 7.3.1.1B *shows the reconstructed values using bent rays and feasibility constraints. The other two examples are presented similarly, progressing from smaller anomalies to larger ones.*

The reconstructions were found to converge after 15 or 20 iterations, and did not vary significantly if the iteration sequence was continued. In general, we expect slow anomalies to be harder to reconstruct than fast anomalies, because rays tend to avoid the slow regions in favor of fast regions. Thus, the coverage of slow anomalies tends to be much less than for fast anomalies, and therefore the resolution of such regions tends to poorer than for the fast regions. This effect is observed in all the examples. The reconstructions for 20% *and* 50% *contrasts were quite good, while that for* 100% *was noticeably worse than the other two. The main reason for this difference is that in the presence of high contrasts the ray paths tend to seek out the fastest regions; thus, even the background in the vicinity of a very fast anomaly can become poorly resolved, since all the rays close to the fast anomaly go through it and therefore do not sample the surrounding region well. To improve the resolution of the reconstruction in the presence of high contrasts requires a substantial increase in the density of ray-path sampling.*

1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
1.00	1.00	1.00	1.20	1.20	1.00	1.00	1.00
1.00	1.00	1.20	1.20	1.20	1.20	1.00	1.00
1.00	1.00	1.00	1.20	1.20	1.00	1.00	1.00
1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
1.00	1.00	1.00	0.83	0.83	1.00	1.00	1.00
1.00	1.00	0.83	0.83	0.83	0.83	1.00	1.00
1.00	1.00	1.00	0.83	0.83	1.00	1.00	1.00
1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00

EXAMPLE 7.3.1.1A. The double-cross slowness model with 20% contrast.

0.98	0.97	1.02	1.06	1.04	1.00	0.98	0.96
1.02	1.00	0.98	1.01	1.02	1.00	1.01	1.02
1.00	1.01	0.98	0.98	0.98	0.98	1.02	1.05
1.01	1.00	1.03	1.15	1.14	1.04	1.02	1.03
0.99	0.99	1.12	1.20	1.18	1.14	1.02	1.02
1.00	1.00	1.00	1.15	1.18	1.06	1.00	1.02
0.98	1.01	1.00	1.02	1.03	0.96	1.02	1.01
1.00	1.01	1.00	1.01	1.00	0.99	1.01	1.01
1.00	1.00	1.00	1.01	1.00	1.01	1.00	0.99
0.99	1.00	0.99	0.98	0.99	1.01	1.00	0.99
0.99	0.99	0.99	0.89	0.87	0.95	1.04	0.97
1.00	0.98	0.89	0.85	0.84	0.87	0.94	0.97
1.03	0.96	0.95	0.87	0.85	0.95	0.94	0.99
1.04	1.00	0.99	0.97	0.98	0.95	1.02	1.00
1.01	1.01	1.03	0.98	0.97	1.00	1.01	0.98
1.00	1.04	0.97	0.98	0.98	0.99	1.01	1.02

EXAMPLE 7.3.1.1B. Reconstruction of the double-cross slowness model with 20% contrast using the BST code with feasibility constraints and noisy data (after 41 iterations).

1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
1.00	1.00	1.00	1.50	1.50	1.00	1.00	1.00
1.00	1.00	1.50	1.50	1.50	1.50	1.00	1.00
1.00	1.00	1.00	1.50	1.50	1.00	1.00	1.00
1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
1.00	1.00	1.00	0.67	0.67	1.00	1.00	1.00
1.00	1.00	0.67	0.67	0.67	0.67	1.00	1.00
1.00	1.00	1.00	0.67	0.67	1.00	1.00	1.00
1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00

EXAMPLE 7.3.1.2A. The double-cross slowness model with 50% contrast.

1.04	0.94	0.97	1.20	1.08	1.02	0.99	0.94
1.05	1.05	0.92	1.06	1.02	0.98	1.00	1.06
1.04	0.93	1.07	0.99	1.04	0.90	0.96	1.14
1.02	1.01	0.99	1.10	1.23	1.33	1.04	1.02
0.95	1.00	1.42	2.20	1.28	1.23	1.05	1.04
0.97	1.03	0.96	1.21	1.21	1.10	1.07	1.05
0.99	1.06	0.98	1.08	0.98	0.92	1.11	1.03
0.97	1.08	0.98	1.01	0.95	1.03	1.03	1.04
0.99	1.00	1.01	1.04	0.95	1.05	1.03	1.01
0.98	1.00	0.94	0.96	0.91	1.04	1.00	0.94
0.97	1.03	0.93	0.81	0.73	0.90	1.06	0.95
1.00	0.98	0.75	0.68	0.68	0.74	0.88	0.96
1.09	0.91	0.88	0.78	0.71	0.92	0.93	0.94
1.06	1.02	0.92	1.05	0.93	0.93	1.02	0.96
1.01	1.03	1.00	0.98	0.97	0.98	1.04	0.95
0.98	1.14	0.92	0.94	0.96	0.96	1.06	1.01

EXAMPLE 7.3.1.2B. Reconstruction of the double-cross slowness model with 50% contrast using the BST code with feasibility constraints and noisy data (after 41 iterations).

1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
1.00	1.00	1.00	2.00	2.00	1.00	1.00	1.00
1.00	1.00	2.00	2.00	2.00	2.00	1.00	1.00
1.00	1.00	1.00	2.00	2.00	1.00	1.00	1.00
1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
1.00	1.00	1.00	0.50	0.50	1.00	1.00	1.00
1.00	1.00	0.50	0.50	0.50	0.50	1.00	1.00
1.00	1.00	1.00	0.50	0.50	1.00	1.00	1.00
1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00

EXAMPLE 7.3.1.3A. The double-cross slowness model with 100% contrast.

1.04	0.98	1.03	1.04	1.18	1.07	1.00	1.02
1.12	1.05	0.84	1.00	1.10	1.01	1.04	1.16
1.08	0.94	1.01	1.00	1.02	0.81	0.93	1.39
1.05	0.99	1.00	1.09	1.38	1.53	1.08	1.19
0.90	1.00	1.70	2.38	1.46	1.23	1.04	1.12
0.91	1.09	0.97	1.28	1.40	1.10	1.02	1.11
1.08	1.16	0.99	1.17	0.93	0.86	1.15	1.09
1.07	1.13	0.99	0.98	0.91	0.97	1.05	1.09
1.09	0.97	1.00	1.02	0.94	1.03	0.98	1.04
0.99	0.95	0.94	0.86	0.88	1.02	1.00	0.94
0.99	1.03	0.89	0.66	0.62	0.82	1.08	0.90
1.08	0.86	0.66	0.56	0.57	0.62	0.80	0.94
1.05	0.83	0.81	0.69	0.62	0.77	0.84	0.84
1.12	1.00	1.01	0.91	0.81	0.90	0.93	1.02
1.01	0.96	1.04	0.97	0.86	0.99	0.97	0.96
0.97	1.12	0.99	0.95	0.92	0.90	1.08	1.01

EXAMPLE 7.3.1.3B. Reconstruction of the double-cross slowness model with 100% contrast using the BST code with feasibility constraints and noisy data (after 41 iterations).

7.4 Using Relative Traveltimes

When we do not have control over the seismic source location and timing as in the case of earthquakes, the absolute traveltimes are not accurately known and it is important to understand how relative traveltimes may be used in seismic tomography [Aki, Christoffersson, and Husebye, 1977].

Rigorous application of the feasibility constraints $\mathbf{Ms} \geq \mathbf{t}$ requires fairly accurate knowledge of the absolute traveltimes. When such information is sparse or unavailable, we can use the information known about gross geological structure of the region to estimate the mean traveltime. Then we remove the physically meaningless mean of the relative data T/m and add back in the geological mean τ_0.

The *remove-the-mean operator* $\mathbf{R}$ for an m-dimensional vector space is defined as

$$\mathbf{R} = \mathbf{I} - \mathbf{u}\frac{1}{m}\mathbf{u}^T, \tag{7.33}$$

where $\mathbf{u}^T = (1, \ldots, 1)$ is an m-vector of ones. Note that $\mathbf{RR} = \mathbf{R}$ so $\mathbf{R}$ is a projection operator. Then, we see that $\mathbf{R}$ applied to the traveltime vector $\mathbf{t}$ gives

$$\mathbf{Rt} = \mathbf{t} - \frac{T}{m}\mathbf{u}, \tag{7.34}$$

where $T/m = \mathbf{u}^T\mathbf{t}/m$ is the mean traveltime of the data set. Applying $\mathbf{R}$ to the ray-path matrix, we have

$$\mathbf{RM} = \mathbf{M} - \mathbf{u}\frac{T}{m}\mathbf{v}^T\mathbf{C} = \mathbf{M} - \mathbf{u}\frac{T}{m}\mathbf{c}^T. \tag{7.35}$$

The standard procedure for this problem is to solve for $\mathbf{s}$ in the equation

$$\mathbf{M}'\mathbf{s} = \mathbf{t}', \tag{7.36}$$

where $\mathbf{M}' = \mathbf{RM}$ and $\mathbf{t}' = \mathbf{Rt}$. To apply the feasibility constraints, we must modify the problem to

$$\mathbf{Ms} \geq \mathbf{Rt} + \tau_0\mathbf{I}_m. \tag{7.37}$$

Hidden in this analysis is the fact that the earthquake sources are often far from the region to be imaged, so the "effective" source locations may be placed at the boundaries of the region to be imaged.

If we have predetermined the mean for the traveltime data, then it is clearly desirable to use an inversion procedure that preserves this mean, *i.e.*, choosing $\Delta\mathbf{s}$ so that

$$\frac{\mathbf{u}^T\mathbf{M}(\mathbf{s} + \Delta\mathbf{s})}{m} = \tau_0 \tag{7.38}$$

for all $\Delta\mathbf{s}$. Preserving the mean is equivalent to preserving the total traveltime along all ray paths, so

$$\mathbf{c}^T(\mathbf{s} + \Delta\mathbf{s}) = m\tau_0. \tag{7.39}$$

In other words, vary $\mathbf{s}$ so it stays in the hyperplane determined by (7.39). But we have studied exactly this mathematical problem using linear programming in (7.6) and also using weighted least-squares in (7.25). So we do not need to develop any new inversion methods for this special case.

7.5 Parallel Computation

Traveltime inversion algorithms tend to be parallelizable in a variety of ways. The use of the feasibility constraints only increases the degree of parallelism that is achievable by these algorithms.

First, the *forward modeling* may be parallelized. If the forward problem is solved using either shooting or bending methods, then it is straightforward to parallelize the code because each ray may be computed independently of the others, and therefore in parallel. If the forward problem is solved using a finite difference algorithm or a full wave equation method, then whether the algorithm is parallelizable or not depends on the details of the particular algorithm. For example, Vidale's method is not parallelizable, but another related method by van Trier and Symes [1991] is parallelizable.

Second, the use of the feasibility constraints in inversion algorithms suggests that it might be advantageous to map the feasibility boundary and then use the information gained to search for improved agreement between the model and the data. Mapping the feasibility boundary can be done completely in parallel. Each model $\mathbf{s}$ may be treated in isolation, computing the best ray-path matrix for that model, and then finding the scaled model in the direction of $\mathbf{s}$ that intersects the feasibility boundary. The difficulty with this method is that it requires a figure of merit (in real problems) to help us determine whether (and to what degree) one point on the feasibility boundary is better than another. In ideal circumstances (no data error and infinite precision in our computers), the figure of merit would be the number of ray paths that achieve equality while still satisfying all the feasibility constraints

$$\mathbf{Ms} \geq \mathbf{t}. \tag{7.40}$$

When that number equals the number of ray paths, we have found an exact solution and, as the number increases towards this maximum value during an iterative procedure, the trial models $\mathbf{s}$ must be converging towards this solution. But in real problems, a figure of merit based on the number of equalities in (7.40) is not useful.

In a series of numerical experiments [joint work with A. J. DeGroot], we have found that a useful figure of merit for real problems is the nonlinear least-squares functional

$$\Psi(\mathbf{s}) = \sum_{i=1}^{n} w_i[\tau_i^*(\mathbf{s}) - \mathbf{t}_i]^2. \tag{7.41}$$

If we have found an exact solution $\mathbf{s}^*$ to the inversion problem, (7.41) will vanish at that point on the feasibility boundary. As we approach this global minimum, (7.41) is evaluated at an arbitrary point on the feasibility boundary and the values in a cluster of such points are compared, our analysis of convex programming shows that the points with the smallest values of (7.41) form a convex set. The smallest value we find may not actually vanish, in which case there is no exact solution to our inversion problem. This procedure has been implemented on a parallel processing machine, and the results obtained using this algorithm with the figure of merit (7.41) are comparable to those of the stable algorithm discussed earlier.

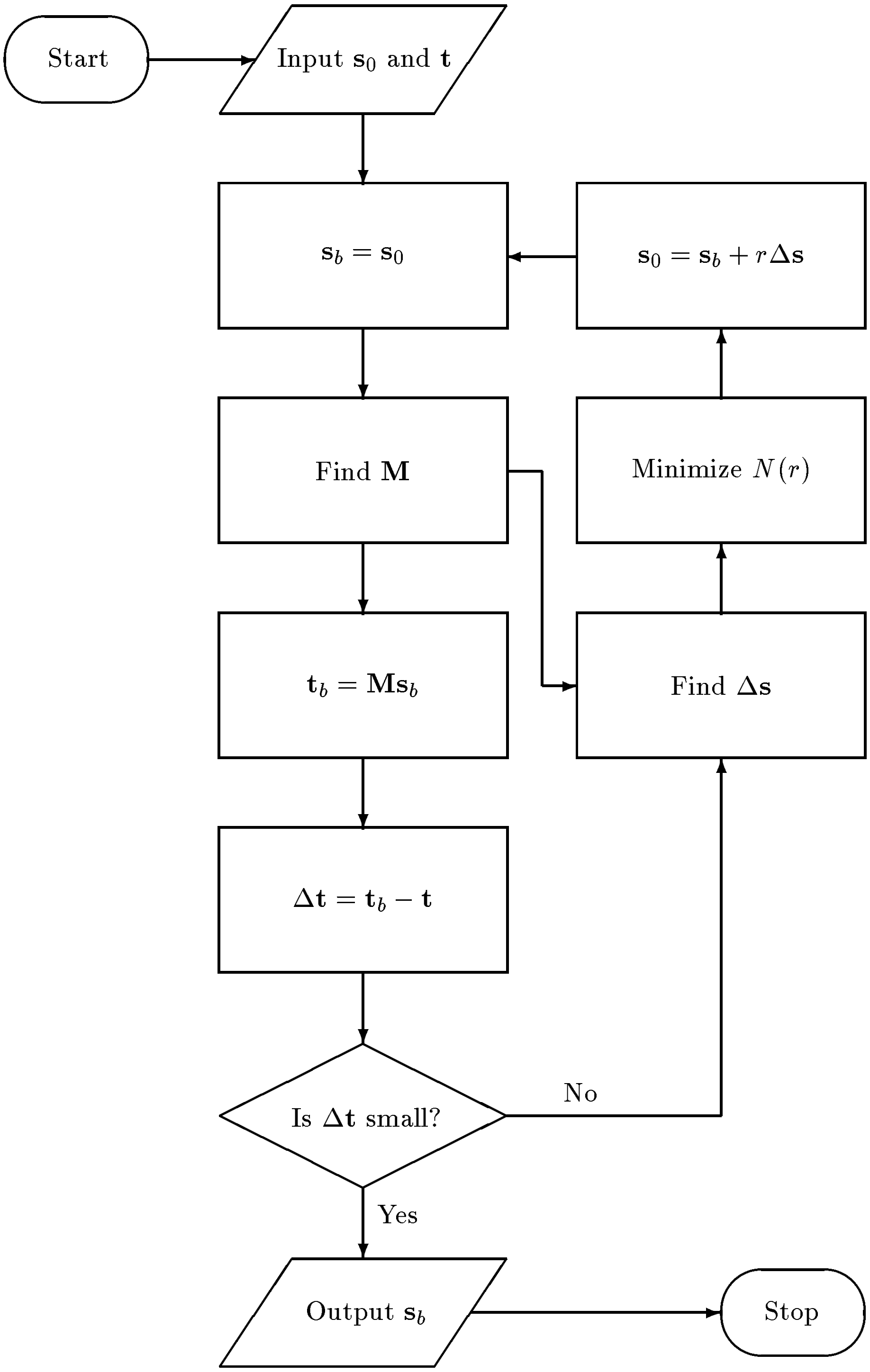


Figure 7.2: Modified iterative algorithm for traveltime inversion using feasibility constraints.

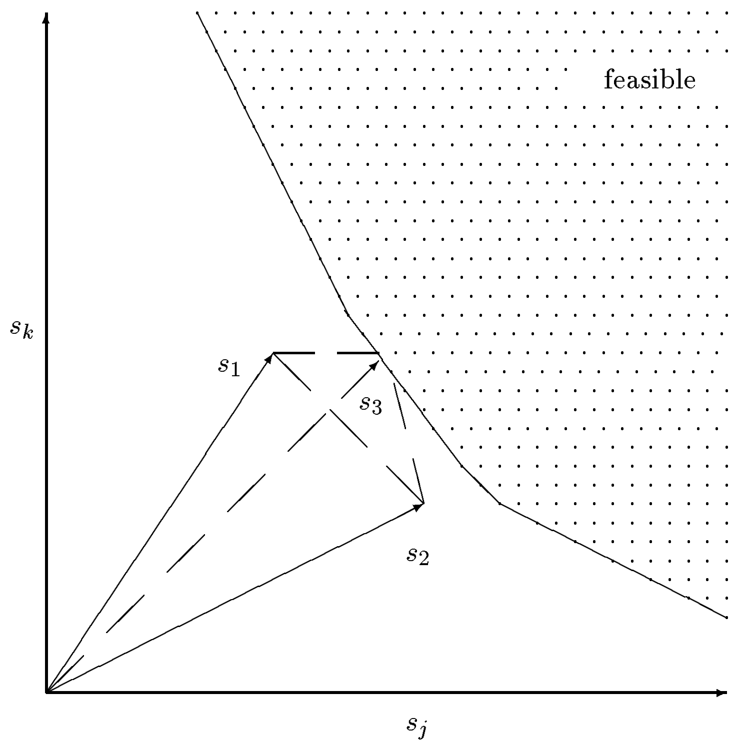


Figure 7.3: Snapshot of one iteration in a nonlinear inversion algorithm based on feasibility constraints.

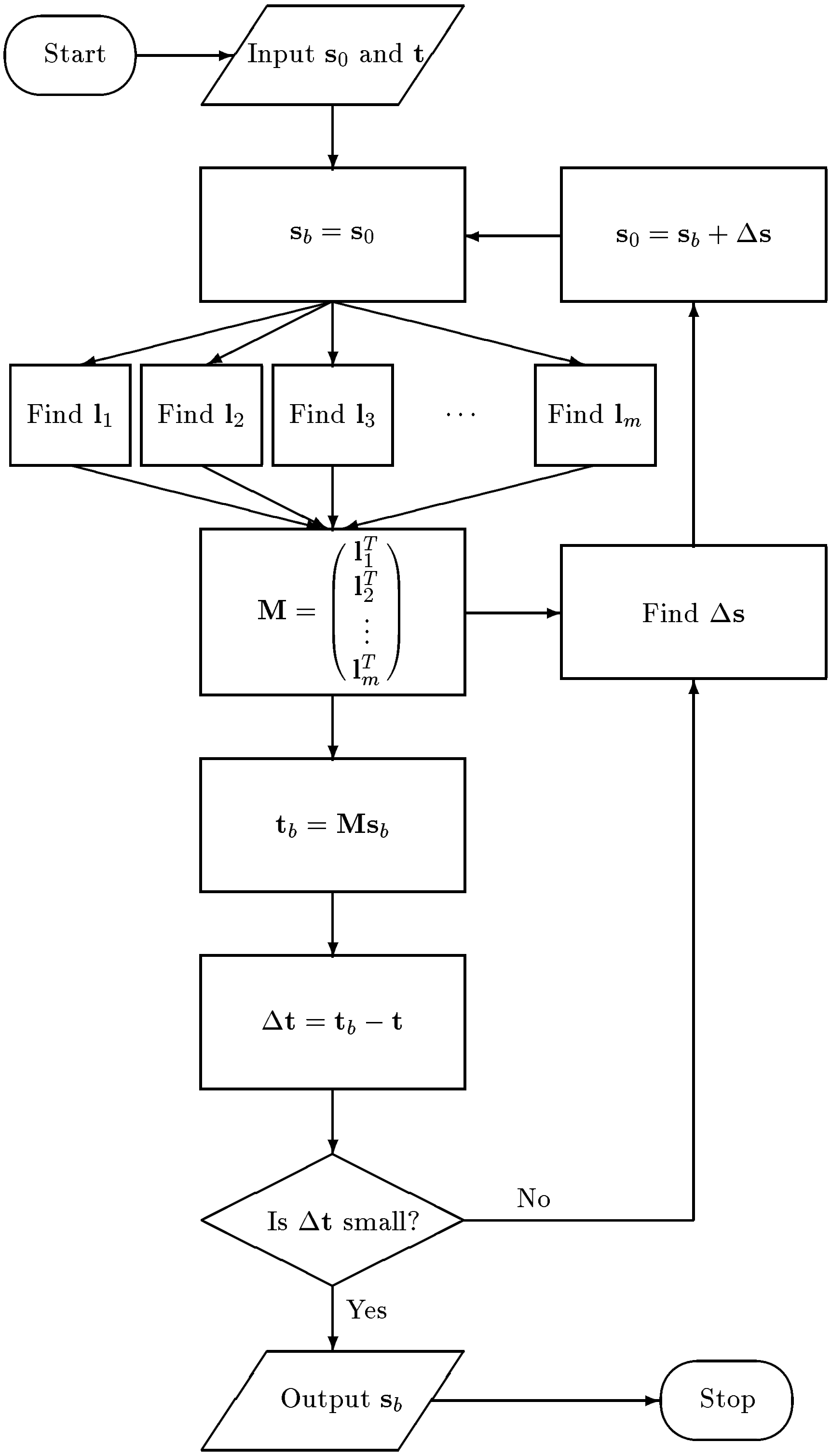


Figure 7.4: Computing ray paths for different source/receiver paairs in parallel.

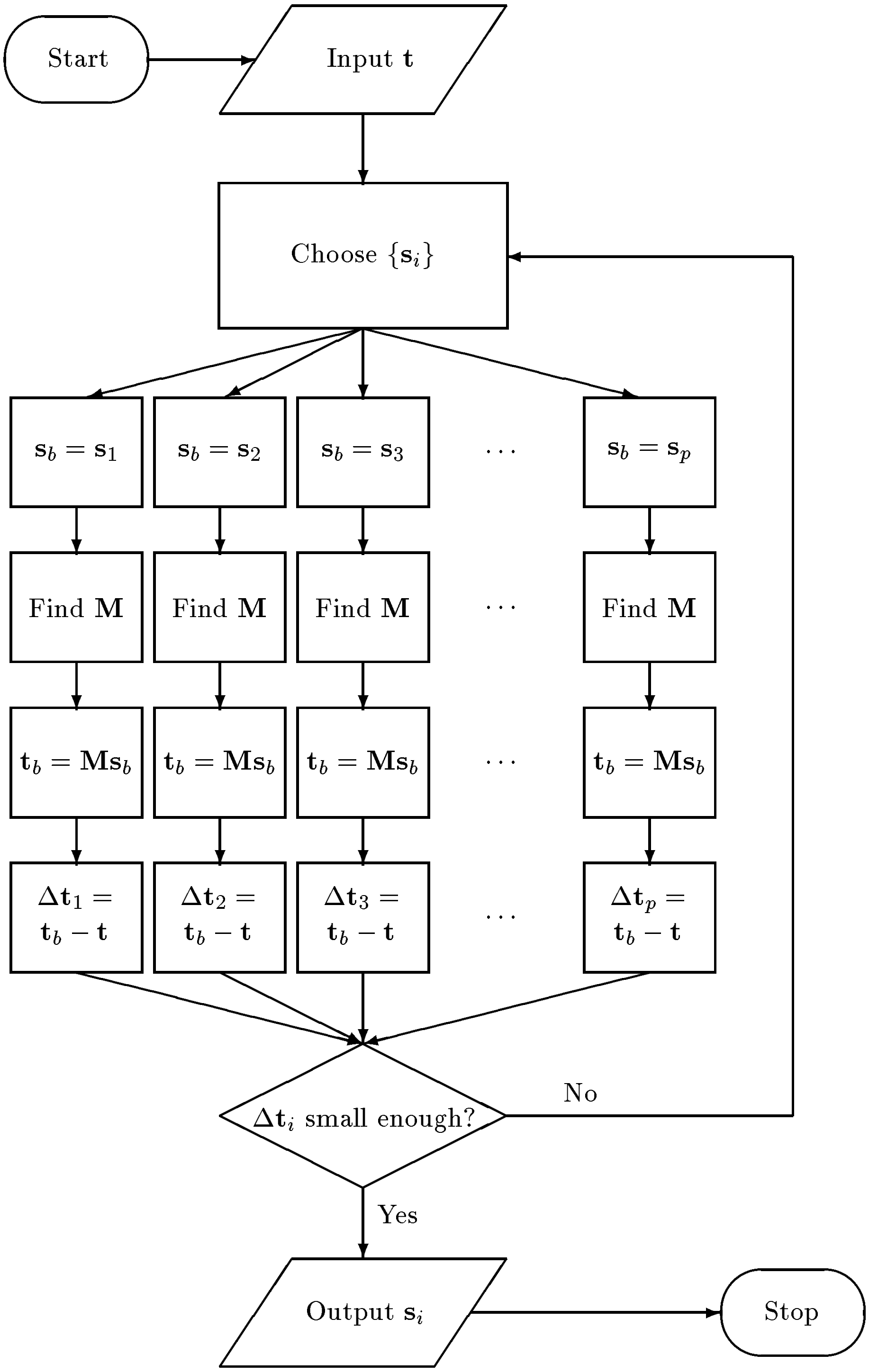


Figure 7.5: Monte Carlo method of mapping feasibility boundary.