Ray tracing in 3-D media by parameterized shooting

Yonghe Sun

Department of Geology and Geophysics, University of Utah, Salt Lake City, UT 84112, USA

Accepted 1992 December 8. Received 1992 December 8; in original form 1992 May 23

SUMMARY

A new ray-bending method is presented for tracing rays in a 3-D isotropic medium. Ray paths are iteratively updated by integrating the ray equation along the approximate ray path. The improved path is expressed in terms of the path integrals along the approximate path. The new updating procedure achieves similar or superior performance with often significantly less operation counts than other bending methods. The method further speeds up convergence by tracing the low-frequency smooth components of the ray path with coarse grids and the high frequency components with fine grids.

Key words: multi-level, parameterized shooting, ray tracing.

1 INTRODUCTION

Despite its limitations, ray tracing remains a powerful method for seismological studies. The literature on seismic ray tracing is extensive. For a review of recent ray tracing methods, see Červený (1987). There are three approaches for two-point ray tracing: shooting methods, methods of characteristics and bending methods.

In shooting methods, ray tracing is an iterative-initial-value problem (e.g. Langan, Lerche & Cutler 1985). The shooting is guided by Snell’s law or the ray equation from the source at \( x_s \) to the receiver at \( x_r \). The shooting angle at the source point is iteratively adjusted until a ray intersects the targeted receiver. Shooting methods can be very efficient if a large set of rays are traced for closely spaced receivers. The drawback with shooting methods is that the target points are frequently not well behaved functions of the shooting angles, making some rays impossible to trace.

In the method of characteristics, the traveltime fields or wavefronts are first constructed by (say) solving the eikonal equations (e.g. Vidale 1988, or Qin et al. 1992). The stationary paths with minimum values of \( t(\mathbf{x}, \mathbf{x}_s) + t(\mathbf{x}, \mathbf{x}_r) \) give the ray paths, where \( t(\mathbf{x}, \mathbf{x}_s) \) is the traveltime at a point \( \mathbf{x} \) for the source at \( \mathbf{x}_s \). Rays can also be found by following the wavefront normals from the receiver to the source. Moser (1991) proposed a shortest path method to find the first arrival traveltimes and ray paths using network theory. The method of characteristics can be efficient if rays for a large set of receivers and sources are required.

In bending methods, ray tracing is an iterative-boundary-value problem. To trace a ray, a proposed ray path connecting a source and receiver is iteratively refined. In many bending algorithms, the path is refined by discretizing some form of the ray equation or by minimizing the traveltime with respect to the node positions \( (\mathbf{x}_i, i = 1, 2, \ldots, N) \) on the path. Such algorithms (see, for example, Wesson 1971; Julian & Gubbins 1977; Pereya, Lee & Keller 1980) often lead to a set of equations with the node positions as unknowns. These methods are advantageous over shooting methods when the receiver positions are ill-behaved functions of the shooting angles. The disadvantage is that it is computationally expensive to prepare and simultaneously solve this linear system of equations.

There are also bending methods in which a path is refined locally to decrease the traveltime on a segment-by-segment basis (e.g. Chander 1977; Um & Thurber 1987). Such local-updating schemes are typically expressible by,

\[ \mathbf{x}^{(k)}_i = \mathbf{F}_i \mathbf{x}^{(k-1)}_{i-1} + \mathbf{G}_i \mathbf{x}^{(k-1)}_{i+1}, \]

where \( \mathbf{x}^{(k)}_i \) is the \( i \)th node position on the path after the \( k \)th iteration, \( \mathbf{F}_i \) and \( \mathbf{G}_i \) are some \( D_x \times D_x \) matrices. Local-updating schemes usually require less computation per iteration than those solving a linear system of equations. They have, however, a serious convergence stalling problem. If a portion of the path (such as a straight segment in a uniform slowness region) nearly satisfies the ray equation (which does not mean that the portion is close to the true ray path), the local-updating schemes will be ineffective for that portion because, locally, there is little need for improvement. This problem is illustrated by an example in the Appendix.

I present a 3-D ray-bending algorithm using a parameterized shooting procedure that achieves the performance virtues of both the shooting and bending approaches while avoiding the aforementioned deficiencies. Throughout this paper, \( D_x \) denotes the dimension of the space occupied by the rays, \( 2 \leq D_x \leq 3 \). If the velocity of the medium varies for \( D_m \) of the \( D_x \) Cartesian coordinates in the ray space,
2 THE PARAMETERIZED SHOOTING PROCEDURE

A ray path \( x(s) \) parameterized by the path length \( s \) satisfies the ray equation,

\[
\frac{d}{ds} \left[ n \frac{dx(s)}{ds} \right] = \nabla n,
\]

where \( n(x) = \frac{v_0(x)}{v(x)} \) is the refractive index field (or slowness field if \( v_0 = 1 \)) of the velocity profile \( v(x) \), and \( v_0 \) is a reference velocity. To trace a ray, we begin with a path \( x(s) \) that approximates the ray path. The approximate ray path originates at the source point \( x(0) = x_s \), and terminates at or near the receiver point \( x_r \). Thus \( x(S) = x_r \), where \( S \) is the total length of the path. For convenience of reference, this approximate path will be referred to as the old path to distinguish it from the new path (to be sought) that better approximates the ray. The integration of the ray equation with respect to \( s \) along the old path gives,

\[
n(s) \frac{dx(s)}{ds} = a_o + p(s),
\]

where

\[
p(s) = \int_0^s \nabla n(s) \, ds,
\]

and \( a_o = n(0)(dx/ds)_{s=0} \), with \( n(s) \) understood as \( n(x(s)) \). The constant vector \( a_o \) of integration contains the ‘shooting angle’ vector \( (dx/ds)_{s=0} \) of the old path. Integrating once more gives a new path

\[
x^\text{new}(s, a_o) = x_s + a_o f(s) + q(s), \quad 0 \leq s \leq S,
\]

where

\[
f(s) = \int_0^s 1/n(s) \, ds, \quad q(s) = \int_0^s p/n(s) \, ds.
\]

The traveltime along the old path is defined by the path integral,

\[
\tau = \left( \frac{1}{v_0} \right) \int_0^S n(s) \, ds.
\]

It can be verified that \( x^\text{new}(0, a_o) = x_s \). The new path in general does not terminate at the receiver point because \( (dx/ds)_{s=0} \) is only an approximate shooting angle vector. To find a path that does terminate at the receiver, one modifies the parameter \( a_o \) in eq. (2) to vary as a free vector parameter \( a \):

\[
x^\text{new}(s, a) = x_s + a f(s) + q(s), \quad 0 \leq s \leq S.
\]

The \( x(s, a) \) defines a parameter family of paths emanating from the source point (Fig. 1). In this family of paths (for the given old path), one path will terminate at the receiver point \( x_r \). The corresponding value of the free parameter \( a = a^\text{new} \) of the new path can be determined by setting

\[
x_r = x^\text{new}(S, a^\text{new}) = x_s + a^\text{new} f(S) + q(S),
\]

and solving for \( a^\text{new} \),

\[
a^\text{new} = \frac{x_r - x_s - q(S)}{f(S)}.
\]

All the functions \( f(s), p(s), \) and \( q(s) \) that appear on the right-hand sides of eqs (3) and (4) are integrals along the old path. The new path \( x^\text{new}(s, a^\text{new}) \) connects the source to the receiver.

The path integrals can be numerically evaluated by dividing the old path into \( N \) short segments separated by \( N + 1 \) nodes, with the source at the zeroth node \( x_0 = x_s \) and the receiver at the last node \( x_N = x_r \). The path integrals along the \( i \)th ray segment between the two nodes at \( x_i \) and \( x_{i-1} \) are approximated by,

\[
f_i - f_{i-1} = d_i/n_i + O(d_i^3),
\]

\[
p_i - p_{i-1} = \nabla n_i d_i + O(d_i^3),
\]

\[
q_i - q_{i-1} = (d_i/2n_i)(p_i + p_{i-1}) + O(d_i^3),
\]

\[
x^\text{new}_i = x_i + a^\text{new}_i f_i + q_i,
\]

\[
v_i(\tau - \tau_{i-1}) = d_i n_i + O(d_i^3),
\]

where the path length of the segment is \( \delta s_i = d_i + O(d_i^3) \), with \( d_i = x_i - x_{i-1} \), and \( n_i \) and \( \nabla n_i \) are evaluated at the midpoint \( x_i = (x_{i-1} + x_i)/2 \) between the two nodes, and

\[
a^\text{new}_i = \frac{x_r - x_{i-1} - q_i} {f N}.
\]

The above procedure parameterizes the ray path by the path length and the new approximated ray path is determined analytically from a family of paths parameterized by the ‘shooting vector’ \( a \). Hence this procedure is designated as parameterized shooting although no shooting is actually required to update the new path. The procedure can be repeated for many iterations until convergence. The initial paths are not required to update the new path. The procedure can be an advantage when tracing rays with closely spaced receivers for which neighbouring rays may be used as initial guesses.

The parameterized shooting procedure does not have the convergence-stalling problem that plagues the local-updating schemes. For a region with uniform slowness, \( p(s) \) is constant, \( q(s) \) and \( f(s) \) are linear functions of \( s \), so the new
updated portion of the path will be a straight segment even if the corresponding portion along the old path is not straight. The straight path segment connecting \( x_{s}^{(1)} \) and \( x_{s}^{(0)} \) in Fig. Al(a), which is only approached with many iterations using a local-updating scheme, is obtained by just one iteration using the parameterized shooting procedure.

### 3 THE MULTILEVEL STRATEGY

The smoother, larger-scale bendings of the ray path can be traced with proportionally less amount of computation. This is achieved by coarse-path discretization, coarse-grid sampling of the slowness medium, and less constraints on the ray path. As the accuracy of the ray path increases, the discretization grid is refined to trace the higher frequency bending and to correct for the small low-frequency residuals along the path. Fig. 2 illustrates this idea.

#### 3.1 Multilevel path discretization

Variable or multilevel path discretization has been used for bending methods (e.g. Julian & Gubbins 1977; Um & Thurber 1987). The multilevel discretization can also be used with the parameterized shooting procedure. Starting with a small number of ray segments of roughly the same length for the path discretization, the iterations continue until the total traveltime along the new path no longer decreases. This forms a cycle of iterations (Fig. 3). The number of ray segments is then doubled to begin the next cycle of iterations. New nodes may be added by interpolating the existing node positions as a function of the path length. Each cycle is completed when the traveltime \( \tau \) no longer decreases, or when the decrease \( \Delta \tau / \tau \) is less than a preset tolerance limit. The number of ray segments in the \( c \)th cycle will be denoted by \( N_c \), where \( c = 1, 2, \ldots, f \), with \( f \) being the total number of cycles. Note that \( N_c = 2^{c-1} N_1 \) and the number of iterations required for the \( c \)th cycle will be denoted by \( I_c \).

#### 3.2 Multilevel velocity sampling

The velocity medium is discretized into a uniform mesh of grids or cells with grid spacings \( h_j, j = 1, 2, \ldots, D_m \). A value of the refractive index \( n \) is specified at each grid point or at the centre of each cell. For the \( i \)th path segment with end nodes at \( x_i^1 \) and \( x_i \), the values of \( n \) and \( \nabla n \) at the midpoint \( \bar{x}_i = (x_i^1 + x_i)/2 \) should be representative of the 'averaged' variation in the neighbourhood of the ray segment. This can be performed by the following two alternative sampling procedures.

#### 3.2.1 Method 1: size adaptive sampling

In the first method, the values of \( n \) and \( \nabla n \) are obtained by interpolating the values of \( n \) on a sampling grid (Fig. 4). The sampling grid consists of three grid points for 1-D, \( 3 \times 3 \) grid points for 2-D and \( 3 \times 3 \times 3 \) grid points for 3-D media. The centre of the sampling grid is at the grid point nearest to \( \bar{x}_i \).

![Figure 2](http://gji.oxfordjournals.org/)

**Figure 2.** A multiclevel strategy uses: (a) coarse grids to trace long wavelength ray bending and (b) fine grids to trace short wavelength ray bending.
The remaining sampling grid points are picked so that each side of the sampling grid is roughly \( d_i \) (or \( \tilde{d} \)) long (Fig. 5).

The values \( n \) and \( \nabla n \) at \( \bar{x}_i \) may be obtained by a truncated Taylor expansion,

\[
n = n + \frac{\partial n}{\partial x_i} \Delta x_i + \frac{1}{2} \sum_{i,j=1}^{m} \frac{\partial^2 n}{\partial x_i \partial x_j} \Delta x_i \Delta x_j + O(|\Delta x|^3),
\]

(11)

where \( \Delta x \) is the displacement from the centre grid to \( \bar{x}_i \), and the expansion coefficients are the partial derivatives of \( n \) at the centre grid point (computed by finite differencing the slowness values on the sampling grid). Another approach is to apply interpolation to the sampling grid. For a 1-D medium, the three-point Lagrange interpolation formula (e.g. Abramowitz & Stegun 1970) may be used,

\[
n(x + ph) = \frac{p(p-1)}{2} n(x-h) + (1-p^2)n(x) - \frac{p(p+1)}{2} n(x+h).
\]

(12)

The multidimensional interpolation for a 2-D or 3-D sampling grid can be accomplished by a sequence of 1-D interpolations.

### 3.2.2 Method 2: level adaptive sampling

In the second method, the medium is discretized with several levels of coarseness (Fig. 2). The finest level of the discretization has grid spacings \( h_i \), with \( i = 1, 2, \ldots, D_m \). The successively coarser levels have grid spacings of \( 2h_i, 4h_i, 8h_i, \ldots \), and so on. On the coarsest level, the whole region may contain just one or a few cells.

At each level of discretization, the values of the spatial derivatives \( \partial n/\partial x_i \) and \( \partial^2 n/\partial x_i \partial x_j \) (\( i, j = 1, 2, \ldots, D_m \)) for each grid point are computed and stored prior to the ray-tracing session. The values of \( n \) and \( \nabla n \) at a point \( \Delta x \) away from the closest grid point in the selected level of discretization are computed by the Taylor expansion (eq. 11). An appropriate level of medium discretization will be selected for each cycle of iterations. The criterion for the level selection is that the typical node spacing \( d \) along the path roughly matches the grid spacings for the discretization.

### 3.3 Multilevel path constraint

Because the velocity distribution is sampled only in the vicinity of the old ray path, the improved ray path must not be too far from the old ray path to ensure convergence. Let \( \Delta x_{\text{pred}} \) be the predicted change in the position of the \( i \)th node (see eq. 8) for an iteration in the \( c \)th cycle.

\[
\Delta x_{\text{pred}} = \Delta x_{\text{new}} - \Delta x_{\text{old}}, \quad i = 1, 2, \ldots, N_c - 1.
\]

(13)

If the maximum displacement

\[
|\Delta x|_{\text{max}} = \max \{ (\Delta x_{\text{pred}})^2, i = 1, 2, \ldots, N_c - 1 \}^{1/2}.
\]

(14)

is larger than a half of the typical node spacing \( \bar{d} \) along the path, then all of the predicted node displacements will be scaled down by a common factor, \( r = \min \{ 1, (d/2)/|\Delta x|_{\text{max}} \} \), so that \( \Delta x = r \Delta x_{\text{pred}} \). In general, the displacement vector \( \Delta x_{\text{pred}} \) has a tangential component to the old path. It is desirable to eliminate this component to regulate node spacing by preventing the nodes from 'sliding' along the path. Thus, the new path will be computed by,

\[
\Delta x_{\text{new}} = \Delta x_{\text{old}} + \Delta x_{\text{pred}}.
\]

(15)

It is adequate to approximate \( \Delta x_{\text{pred}} \) by \( \Delta x_{\text{new}} - \Delta x_{\text{old}} / d \). If the traveltime \( t_{\text{new}} \) along the new path is not less than the traveltime \( t_{\text{old}} \) along the old path, the value of \( r \) can be halved several times. Rediscretizing the path is recommended if some \( d_i \) gets too large (say \( d_i > 1.2 \bar{d} \)) due to node 'sliding'.

For a medium with high-contrast discontinuities or rapid-slowness variations, a slight change in the path position may lead to a large velocity change along the path. A fine discretization of the path is required to begin the first cycle of iterations. Inadequate path constraints due to coarse path discretizations may result in instability, poor convergence, or premature cycle termination.

### 4 EFFICIENCY ANALYSIS

What distinguishes one bending method from another is its updating procedure. The updating procedure consists of all the computations required to find the traveltimes and the \( N \) node positions on the new path; this excludes the calculation of the slowness and its gradient. Many auxiliary procedures are also needed for a bending method. Examples are variable or multilevel path discretizations, optimal-slowness
Figure 4. The sampling grid for (a) 1-D, (b) 2-D, and (c) 3-D media.
interpolation procedures, and so on. But these auxiliary procedures are transferable among bending methods. Therefore the operation counts per node for the updating procedures provides a benchmark for comparing different bending methods.

4.1 Operation counts for the parameterized shooting

The updating procedure for the parameterized shooting method specifically consists of the computations for implementing eqs 5, 6, 7, 8, 9, 13, 14, and 15. Computations for constraining the path (eqs 13 and 14) are included as a necessary ingredient of the updating procedure to insure convergence.

The operation counts per node for the parameterized shooting procedure can be found by totalling operations required for implementing these equations. Table 1 lists the itemized operation counts per node for the individual evaluations involved. Optimizations such as computing \( a \cdot b + a \cdot f/2^3 \) as \( a \cdot (b + 0.125) \) have been taken into account. The terms enclosed in square brackets are not considered. Operations for taking minima or maxima (e.g. in eq. 14) are ignored. Operations for the expression of \( a_{\text{new}} \) and \( r \) are performed once per iteration and are therefore insignificant per node. As can be seen from Table 1, for each node, the updating procedure for the parameterized shooting method requires 15\( D_i + 6D_m + 2 \) arithmetic operations, one square-root operation, and no other functional evaluations. Table 2 compares the CPU time for performing one arithmetic or functional operation on a SUN SPARC station 1 using single precision Fortran. It can be seen that one functional evaluation is equivalent to roughly 13 arithmetic operations and should be avoided as much as possible.

4.2 Comparison with other ray-tracing methods

Table 3 compares the operation count per node for a few typical bending methods. Since expressions involved in the updating procedures are not always explicitly presented in the relevant literature, details are filled in wherever required to make the counting possible. The counts may vary depending on how the explicit expressions are written out and on how optimizations are made. Optimization efforts are made to reduce the number of counts as much as possible for each updating procedure. The updating procedures listed in Table 3 are the following:

- **[PAR SHOOT]**: is the updating procedure for the parameterized shooting method. The operation counts per node are the total from Table 1. The smaller operation counts per node for the parameterized shooting updating procedure (Table 3) relative to other bending methods (except for **PARABOLA**) means that the parameterized shooting method requires less computation per iteration. After the slowness computations are included, the operation counts per iteration using the **J&G** method will still be several times more than that of the parameterized shooting method. Moreover, the parameterized shooting method does not have the convergence stalling problem associated with the **PARABOLA** and **U&T** methods.

- **[J&G]**: refers to the updating procedure (for \( D_i = 3 \) and \( D_m = 3 \)) presented by Julian & Gubbins (1977). The updating procedure includes preparing and solving a linear system of equations. The \( 8D_i - 0 = 22 \) additional arithmetic operations per node for path constraint has been added to the procedure because they are generally required for convergence. As can be seen from Table 3, the operation counts per node for the **J&G** method is much larger than counts for other bending methods listed in the table. This is typical of bending methods that solve a linear system of equations with the new node positions as unknowns.

---

Table 1. Operation counts per node for evaluating the expressions involved in the parameterized shooting updating procedure.

<table>
<thead>
<tr>
<th>Expressions Evaluated</th>
<th>+</th>
<th>-</th>
<th>*</th>
<th>/</th>
<th>sqrt</th>
</tr>
</thead>
<tbody>
<tr>
<td>( x_i - x_{i-1} )</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>( d_i/n_i )</td>
<td>2</td>
<td>D_i</td>
<td>2</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>( \sqrt{x_i} )</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>( \Delta x_{\text{new}} )</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>( t_i )</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>( \Delta x_{\text{new}}^2 )</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>( x_i^{\text{new}} )</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

Table 2. The CPU time for performing one arithmetic or functional operation on a SUN SPARC station 1 using single precision.

<table>
<thead>
<tr>
<th>Arithmetic Operation</th>
<th>CPU time (( \mu ) sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>+</td>
<td>0.8</td>
</tr>
<tr>
<td>-</td>
<td>0.8</td>
</tr>
<tr>
<td>*</td>
<td>0.8</td>
</tr>
<tr>
<td>/</td>
<td>2.4</td>
</tr>
<tr>
<td>( \sqrt{x} )</td>
<td>10.1</td>
</tr>
<tr>
<td>( \sin )</td>
<td>10.4</td>
</tr>
<tr>
<td>( \cos )</td>
<td>10.4</td>
</tr>
<tr>
<td>( \exp )</td>
<td>11.4</td>
</tr>
</tbody>
</table>

Table 3. Operation counts per node for updating procedures from several ray tracing methods. Listed are the counts for general \( D_i \) and \( D_m \) values and for \( (D_i, D_m) = (2, 2) \) and \( (3, 3) \).

<table>
<thead>
<tr>
<th>Procedures</th>
<th>Type</th>
<th>Local Arithmetic Operations</th>
<th>CPU time (( \mu ) sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>[PAR SHOOT]</td>
<td>Bending</td>
<td>( 15D_i + 6D_m + 2 )</td>
<td>44 65</td>
</tr>
<tr>
<td><strong>J&amp;G</strong></td>
<td>Bending</td>
<td>( 10D_i + 5D_m + 2 )</td>
<td>58 82</td>
</tr>
<tr>
<td><strong>U&amp;T</strong></td>
<td>Bending</td>
<td>( 12D_i + 5D_m + 2 )</td>
<td>56 81</td>
</tr>
<tr>
<td><strong>PARABOLA</strong></td>
<td>Bending</td>
<td>( 15D_i + 6D_m + 2 )</td>
<td>44 65</td>
</tr>
<tr>
<td><strong>LLC SHOOT</strong></td>
<td>Shooting</td>
<td>( 10D_i + 3D_m + 15 )</td>
<td>41 54</td>
</tr>
</tbody>
</table>
such methods, however, the operations counts may vary significantly.

[U&T]: refers to the updating procedure presented by Um & Thurber (1987). Computations for using the enhancement factors in that paper are included. As can be seen, its operation counts per node is more than that for the parameterized shooting procedure but much less than the counts for the J&G procedure.

[PARABOLA]: refers to an updating procedure like that of U&T except that parabolas are used for updating the nodes. To update the $i$th node in the $k$th iteration, the parabola that best approximates the ray segment between the neighbouring nodes $x_{i}^{(k-1)}$ and $x_{i+1}^{(k-1)}$ is used. Near the midpoint $x = \frac{x_{i}^{(k-1)} + x_{i+1}^{(k-1)}}{2}$ the parabola deviates the most from the straight line connecting the neighbours. The $i$th node is moved to the maximum deviation point,

$$x_{i}^{(k)} = x - [\nabla n |\Delta x| - \Delta x(\nabla n \cdot \Delta x)]/8n,$$

where $\Delta x = x_{i}^{(k-1)} - x_{i}^{(k-1)}$, and $n$ and $\nabla n$ are evaluated at $x$. The second term in the square brackets prevents the node from ‘sliding’ along the path toward high-velocity regions. Computation of the traveltime and use of the enhancement factor are included as part of the procedure. This procedure has the smallest counts among the bending methods listed in Table 3. Unfortunately it suffers the same convergence problem as the U&T procedure because it is a local-updating method.

[LLC SHOOT]: refers to the operations required for shooting one segment (across a slowness cell) by the method of Langan et al. (1985). The LLC SHOOT procedure is included here to put the operation counts for bending and shooting methods in perspective. The operation count per node for a shooting method is generally smaller than that for the bending methods. The LLC SHOOT method is an efficient shooting method because the slownesses and its gradient are precomputed and no interpolation of slownesses is required during the shooting.

If the same number of nodes are used to discretize the ray path, a procedure with smaller operation count per node requires less computation per iteration. The number of iterations may increase significantly with the complexity (such as the presence of the discontinuities) of the slowness medium, but it is not particularly sensitive to $D_{x}$ or $D_{n}$ for a bending method. If a conventional shooting method is used for 3-D ray tracing, the number of iterations required will in general be much larger (Julian & Gubbins 1977) than for 2-D ray tracing because there will be two shooting angle parameters to adjust and two degrees of freedom to miss the target.

4.3 Using different auxiliary procedures

Combining the parameterized-shooting updating procedure with different auxiliary procedures may lead to bending schemes of different efficiency. Table 4 lists the total equivalent arithmetic operations counts per node for three schemes, denoted by PAR SHOOT1, PAR SHOOT2, and PAR SHOOT3. The total equivalent arithmetic operations includes the operations for both the updating procedure and the auxiliary procedure of computing the slowness and its gradient, with one square-root operation counted as 13 arithmetic operations. Size-adaptive sampling is used for PAR SHOOT1 and PAR SHOOT2, and level-adaptive sampling is used for PAR SHOOT3. To compute the slownesses, Lagrange three-point interpolation is used for PAR SHOOT1 and truncated Taylor series is used for PAR SHOOT2 and PAR SHOOT3. PAR SHOOT3 requires the computation and storage of the Taylor coefficients prior to a ray tracing session. PAR SHOOT2 is a compromise between PAR SHOOT1 (the least efficient) and PAR SHOOT3 (the most memory demanding). It can be seen from Table 4 that PAR SHOOT3 is the most efficient scheme among the three.

The computation time for tracing a ray is proportional to the number $N_{t}$ of nodes on the ray and to the number of iterations required for convergence. The required number of iterations varies from ray to ray. For media with similar rapidity of slowness variations, the number of iterations required per ray will be typically the same regardless of the medium and spatial dimensions $D_{m}$ and $D_{r}$, provided the same $N_{t}$ number of nodes are used for the rays of similar path lengths. In this context, Table 4 suggests that, if PAR SHOOT2 is used, tracing a ray in a 3-D medium $(D_{x} = 3, D_{n} = 3)$ is 156/102 = 1.5 times as costly as in a 2-D medium $(D_{x} = 2, D_{n} = 2)$ which in turn is 1.5 times as costly as in a 1-D medium $(D_{x} = 2, D_{n} = 1)$.

5 NUMERICAL TESTING

The performance of the parameterized shooting method has been tested with a single-precision Fortran code on a SUN SPARC station 1.

5.1 Rays in a medium with constant-velocity gradient

The ray tracer was first tested for a medium with a constant velocity gradient for which the exact ray path solutions are known. The following 1-D velocity profile was used,

$$v(x) = 1800 + 0.18x,$$

where lengths are measured in metres and velocities are measured in $m s^{-1}$ and traveltimes in seconds. The 2-D ray path (with $D_{n} = 2$) for a source at $x_{s} = (0, 4500)$ m and a receiver at $x_{r} = (2000, 4000)$ m is a circular arc centred at $x_{c} = (-3000, 2250)$ m with radius $R = 3750$ m and traveltime $t = (2/ln2)/0.6 = 2.310490$ s. The medium is discretized with a grid spacing $h_{x} = 30$ m.

Table 5 lists the computed traveltimes against the number of ray segments $N_{t}$ for tracing the ray. This table applies whether the medium or the ray space is treated as 1-D, 2-D or 3-D (within the tolerance of single-precision computations). The tolerance for cycle termination is $\Delta t/t \leq 1.0 \times 10^{-6}$ and the initial path is the straight line connecting the source and the receiver. It can be seen that
the relative traveltime error reaches the tolerable limit $1.0 \times 10^{-6}$ by the fifth cycle with 64 segments. The computed ray path after the third cycle with 16 segments is indistinguishable from the exact ray path on a plot. The path after the third cycle is accurate enough so that further iterations do not need to improve it. A cycle with $I_1 = 1$ usually makes little or no improvement to the ray path.

5.2 Rays in media with rapid-velocity variations

Figure 6(b) shows a few rays traced by the parameterized shooting method in Langan’s 1-D velocity model [see Fig. 6(a) and Langan et al. 1988] which has rapid-velocity variations. Notice that the two receivers at nearly the same depth level as that of the source are in the ‘shadow zones’ of the source. The traveltime tolerance is set to $\Delta t/t = 1.0 \times 10^{-3}$ and the medium is discretized with grid spacing $h_z = 3$ m. The source well is at $x = 20.0$ m and receiver well is at $x = 233.0$ m. It took 965 CPU seconds to trace 10000 rays by PAR SHOOT2 in this model (with $D_r = 2$ and $D_m = 1$) on the SUN SPARC station 1. The 100 sources and 100 receivers were both spaced with $\Delta z = 3$ m intervals from the depth $z = 6$ m to $z = 303$ m. The initial paths are all straight lines with 16 segments of equal length, and the rays are traced with four cycles of iterations so that $N_f = 2^4 \times 16 = 128$. Thus, the rays were traced at the rate of $(128/N_f) \times 10.4$ rays per CPU second. The rates for other ways of implementing the parameterized shooting procedure may vary by 50 per cent. If the medium is treated as 3-D ($D_m = 3$ and $D_r = 3$) only about half as many or $(128/N_f) \times 5$ rays will be traced per CPU s (see Table 4). The presence of high-contrast discontinuities may further reduce ray-tracing rates by a factor of 2 or more.

As a simple example of applying the parameterized shooting method to media with velocity discontinuities, Fig. 7 shows a ray refracted across an interface separating two uniform half-spaces. Table 6 lists the traveltimes for the cycles. Since the velocity discontinuity at the interface is only approximated by the discretization, the computed traveltine $(=1748.5$ ms) is less than the theoretical traveltine value of 1750 ms. Notice that the maximum deviation between the traced ray and the ideal ray is of the

![Figure 6](http://gji.oxfordjournals.org/)

![Figure 7](http://gji.oxfordjournals.org/)

---

**Table 5.** The accuracy of traveltimes versus the cycles of iterations for the model with a constant velocity gradient.

<table>
<thead>
<tr>
<th>$c$(cycle)</th>
<th>$I_c$</th>
<th>$N_c$</th>
<th>$\tau$ (sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>4</td>
<td>2.500000</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>4</td>
<td>2.352209</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>4</td>
<td>2.317179</td>
</tr>
<tr>
<td>1</td>
<td>3</td>
<td>4</td>
<td>2.317207</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>8</td>
<td>2.312102</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>8</td>
<td>2.312102</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>16</td>
<td>2.310890</td>
</tr>
<tr>
<td>3</td>
<td>2</td>
<td>16</td>
<td>2.310890</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>32</td>
<td>2.310590</td>
</tr>
<tr>
<td>5</td>
<td>1</td>
<td>64</td>
<td>2.310515</td>
</tr>
<tr>
<td>6</td>
<td>1</td>
<td>128</td>
<td>2.310549</td>
</tr>
</tbody>
</table>
Table 6. Traveltimes versus the cycles of iterations for the model of two uniform half spaces.

<table>
<thead>
<tr>
<th>c(cycle)</th>
<th>Ic</th>
<th>Nc</th>
<th>τ (msec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>16</td>
<td>1756.103</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>32</td>
<td>1748.796</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>64</td>
<td>1748.266</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>128</td>
<td>1748.756</td>
</tr>
<tr>
<td>5</td>
<td>1</td>
<td>256</td>
<td>1748.469</td>
</tr>
</tbody>
</table>

The values for the ordinates in Fig. 8(b) are computed by \([t_r(z) - t_e(z)]/t_e(z)\), where \(t_r\) represents the traveltimes by the parameterized shooting method and \(t_e\) represents the traveltimes by the eikonal method. The eikonal traveltimes were also computed by doubling the grid spacings \(h\) and found that the values of \([t_r(z) - t_e(z)]/t_e(z)\) are roughly doubled using same values of \(t_e(z)\). This suggests that the errors in the computed 3-D eikonal traveltimes are linear in the grid spacing \(h\), and that the traveltime computations by the parameterized shooting method can be considered as ‘accurate’.

5.3 Comparison with an eikonal method

Figures 8(a) and (b) compare the performance of the parameterized shooting method with the 3-D eikonal method (Qin et al. 1992) for the 3-D medium,

\[ u(x) = 1800 + 0.2x + 0.1y + 0.04z^2, \]

where \(u\) is in ms\(^{-1}\), and lengths are in metres. The grid spacings are \(h_1 = h_2 = h_3 = 2\) m. The source is located at \(x_s = (40, 40, 100)\) m, and 51 receivers are uniformly spaced along a vertical line at \(x = 180\) m and \(y = 180\) m. The receivers are located between the depths \(z = 0\) m and \(z = 200\) m with a spacing of \(\Delta z = 4\) m. The straight initial paths are each discretized by 18 segments and each ray is traced with four cycles. The traveltime tolerance is set at \(\Delta t/t = 1.0 \times 10^{-5}\). The values for the ordinates in Fig. 8(b) are computed by \([t_r(z) - t_e(z)]/t_e(z)\), where \(t_r\) represents the traveltimes by the parameterized shooting method and \(t_e\) represents the traveltimes by the eikonal method. The eikonal traveltimes were also computed by doubling the grid spacings \(h\) and found that the values of \([t_r(z) - t_e(z)]/t_e(z)\) are roughly doubled using same values of \(t_e(z)\). This suggests that the errors in the computed 3-D eikonal traveltimes are linear in the grid spacing \(h\), and that the traveltime computations by the parameterized shooting method can be considered as ‘accurate’.

5.4 A salt-dome model

The algorithm was used to trace rays in a salt dome model. Fig. 9 shows some rays in the \(y = 0\) cross-section of the dome. The top of the dome is an ellipsoid defined by \(x^2/a^2 + y^2/b^2 + z^2/c^2 \leq 1\), with \(a = 3000\) m, \(b = 2500\) m, and \(c = 2000\) m. The bottom part of the dome is a cylinder of radius \(r = 1500\) m, \(y^2 + x^2 \leq r^2\). The model is discretized with \(h_1 = h_2 = h_3 = 100\) m. The source and 10 receivers (evenly spaced at \(\Delta z = 300\) m intervals along the left edge) are located in the \(x - z\) plane. The rays are traced with \(D_r = D_m = 2\), \(\Delta t/t = 1.0 \times 10^{-5}\), \(N_f = 64\), \(f = 3\), and \(N_r = 2^{f-1}N_f = 256\). The nodes are evenly spaced along the initial straight path for each source-receiver pair.

It took 90.5 CPU seconds to trace 100 rays for the 10 sources at the right edge. Tracing a ray in this model typically requires two times as many nodes \(N_f\) and takes three times as many iterations and is therefore six times as costly as tracing a ray in a 2-D model without sharp

Figure 8. Comparison of the 3-D traveltimes computed by the parameterized shooting method with those obtained by the eikonal method for the 3-D continuous velocity model. (a) The traveltimes plotted against the receiver depths. The solid line is for the eikonal method, and the stars are for the parameterized shooting method. (b) The relative traveltime differences plotted against the receiver depths.

Figure 9. The cross-section of the salt-dome model and rays refracted by the dome. Thicker lines delineate the boundaries separating the uniform regions.
discontinuities. Allowing the nodes to be less dense away from the discontinuity regions can reduce the computation time.

Notice how the ray is refracted at each discontinuity boundary. The rays are bent softly at boundaries. This is because the interpolation of the discretized medium will not reproduce truly sharp boundaries of discontinuities. Very slight bendings of ray paths in the uniform regions are also noticeable. There are two reasons for this: (1) after an iteration, the new path in a uniform region will not be exactly straight if the path constraint (eq. 15) is applied and if the old path in the same region is not straight; (2) the errors due to the slowness discretization and interpolation place a limit on the accuracy of traveltine evaluation regardless of the tolerance $\Delta \tau/\tau$ desired. Thus the path slightly bent in the uniform region may, within tolerance or discretization error, have no larger traveltine (and is therefore judged as no ‘worse’) than if the segment in the region were straight. Similarly, the law of refraction is not exactly followed at the elliptically shaped boundary due to the numerical discretization error in computing the traveltine and in modelling the curved boundary.

These imperfections may be reduced given a more precise knowledge about the discontinuities. Without such knowledge, two paths approximating the same ray path may have different numerical imperfections, but as long as their traveltine errors are within tolerance, both paths should be considered equally acceptable because stationarity of traveltine is the only criterion for a ray path.

6 CONCLUSION

I presented a parameterized shooting method for tracing rays in a 3-D acoustic media. In a parameterized shooting iteration, the updated ray path is determined analytically in terms of the path integrals along the old approximate ray path. Extensive numerical tests show that the algorithm is efficient and has a robust convergence. The operation count per node for the updating procedure is only a fraction of that for other bending methods while achieving similar or superior performance. For media with higher-contrast slowness discontinuities, finer path discretization and therefore more computations are required to trace a ray. The multilevel parameterized shooting algorithm is a near-optimal procedure for tracing rays in a 3-D medium with moderate ray coverage. This assumes that rays are traced one ray at a time.

If multiple paths exist between a source-receiver pair, convergence to a desired ray path is not guaranteed, although different choices of the initial path may lead to the convergence to different ray paths. For dense ray coverage or guaranteed first arrival computation, the method of characteristics is probably more suitable.

ACKNOWLEDGMENTS

I would like to thank Professor Gerard T. Schuster for consultations during the development of the ray tracer. I would also like to thank Nelson Beebe and Dan Tretman for assistance on computing algorithms. Wenyong Cai and Fuhao Qin generated the 3-D eikonal traveltine plots for comparison with the parameterized shooting method. This project was sponsored by the 1990 sponsors of the University of Utah Seismic Tomography consortium. The sponsors were: Amerada Hess, Amoco, Arco, British Petroleum, Chevron, Conoco, Exxon, Gas Research Institute, Institut Francais Du Petrole, Japon, Marathon Oil Co., Mobil, Oryx, Oyo, Philips, Texaco, Unocal and Western Geophysical.

REFERENCES


APPENDIX

This appendix contains an example demonstrating stalled convergence for a local-updating bending method. Consider the situation in Fig. A1(a). A portion of the initial path is a straight segment stretching across a homogeneous region in an otherwise inhomogeneous medium. This portion contains $M = 2m + 1$ equi-spaced nodes from $x^{(0)}_i$ to $x^{(0)}_{i+2m}$. In this homogeneous region, a local-updating scheme during the $k$th iteration will move the $i + j$th ($0 < j < 2m$) node to (say) the mid-point of the straight segment connecting its two neighbours $x^{(k)}_{i+j-1}$ and $x^{(k)}_{i+j+1}$, or,

$$x^{(k)}_{i+j} = [x^{(k)}_{i+j-1} + x^{(k)}_{i+j+1}]/2.$$  

We will define $\delta x^{(k)}_{i+j}$ to be the displacement of the $i + j$th node from $x^{(0)}_{i+j}$ to $x^{(k)}_{i+j}$.

Now suppose that updating other portions of the path requires that the end node $x^{(0)}_i$ be moved by $\delta x$ during the
Ray tracing by parameterized shooting

155

first iteration: \( x^{(1)}_i = x^{(0)}_i + \delta x \). It can be shown that

\[ \delta x^{(1)}_{ij} = \delta x/2. \]

So the path several nodes away from the \( i \)th node is updated by only exponentially small amounts during the first iteration! This is verified by the geometric construction in Fig. A1(a).

To apply the local-updating scheme for more iterations to these \( 2m + 1 \) nodes, let us for simplicity permanently fix the two end nodes at \( x^{(0)}_1 = x^{(0)}_{2m} \). Then it can be shown that all the nodes move in the same direction as \( \delta x \), so that

\[ \delta x^{(k)}_{ij} = a^{(k)}_{ij} \delta x, \]

where \( a^{(k)}_{ij} \) measures, in units of \( \delta x \), the displacement of the \( i+j \)th node from its initial position \( x^{(0)}_{i+j} \) to \( x^{(k)}_{i+j} \). This way, after infinite \( (k = \infty) \) iterations, all the nodes \( x^{(k)}_{i+j} \), \((0 < j < 2m)\) should be on the new straight segment connecting \( x^{(0)}_1 + \delta x \) and \( x^{(0)}_{2m} \), with \( a^{(\infty)}_{ij} = (2m - j)/2m \).

Fig. A1(b) plots the displacement \( a^{(k)}_{i+m} \) of the centre node (with \( j = m \)) of the portion versus \( k \) for three different values of \( M = 2m + 1 = 11, 21 \) and \( 31 \). It can be seen that the number of iterations required for convergence (say to 95 per cent of the desired limit \( a^{(\infty)}_{i+m} = 1/2 \) ) is large, on the order of hundreds instead of a few (say less than 10), and increases rapidly with the number of nodes \( M \). Fig. A1(c) plots the step length \( \Delta a^{(k)}_{i+m} = a^{(k)}_{i+m} - a^{(k-1)}_{i+m} \) versus \( k \). The areas under the three curves are the same, equal to \( a^{(\infty)}_{i+m} = 1/2 \). As \( M \) increases, the curve flattens out rapidly towards large iteration numbers, indicating that convergence stalling is a serious problem if the number of nodes \( M \) is more than a few.

For local-updating schemes, the stalled convergence problem may be relieved by various means, such as multilevel path discretization and application of an ad hoc convergence-enhancement factor as was used by Um & Thurber (1987). The problem can not be entirely eliminated, however, as long as the ray paths traverse regions where fine discretizations are eventually required. A local correction (however small) to the path will still necessitate corrections to the whole path whether the discretization is coarse or fine. Um & Thurber show that their three-point perturbation scheme is 'not appropriate for use in media with first-order velocity discontinuities and/or predominantly constant velocity zones'. To varied degrees, the problem also exists (although less easily demonstrated) when tracing rays through inhomogeneous velocity zones. Notice that, in Fig. A1(c), step length curves all have peaks, with the step lengths (or the convergence rates) being smaller for very small and very large iteration numbers corresponding to straighter paths in the region. In general, the convergence rate stalls when some portions of the path nearly satisfies the ray equation.

**Figure A1.** An example showing stalling for a local-updating scheme. (a) Geometric construction to update a straight portion of an initial path, from the \( i \)th node to the \( i + 2m \)th node, lying in a region with uniform slowness. (b) The displacement \( a^{(k)}_{i+m} \) of the centre (the \( i+m \)th) node versus the iteration number \( k \). (c) The step length \( \Delta a^{(k)}_{i+m} \) of the centre node versus the iteration number \( k \).