Computational geometry applied to kinematic ray tracing

Lucas B. Freitas*, Rodrigo S. Portugal and Martin Tygel, State University of Campinas

This paper was prepared for presentation at the 8th International Congress of The Brazilian Geophysical Society held in Rio de Janeiro, Brazil, 14-18 September 2003.

Abstract

In the wavefront construction (WFC) method there are geometrical problems which arise from the internal ray tracing procedures. For instance there is the problem of interception of ray on interfaces and the determination of the closest distance between a wavefront and a receiver. In this work, we address some computational geometry algorithms that solve those problems, as well as some speed up heuristics. Those geometry problems can be decomposed in simpler and well known computational geopmetry problems, such as: (a) finding the intersection between two segments and (b) determining whether a point is inside a polygon and (c) computating the distance between a point and a segment. As a demonstrative application of these algorithms, we present a Matlab package that constructs kinematic synthetic seismograms of primary reflection events that comes from a given homogeneous layered medium.

Introduction

On the seismic studies, ray theory is largely used, mostly because it provides a very good understanding about the wave phenomena. Its main paradigm is describing the wave phenomena as a superposition of some isolated events, studying their trajectories, termed rays, and the energy flowing through these paths as well.

As the name says, the fundamental object of the ray theory is the ray itself which is a path which satisfies the Fermat's principle: *The ray paths between two points are those for which the traveltime is an extremum, either a minimum, maximum or saddle, with respect to the all nearby possible paths.*

The Snell's law, which is derived from the above principle, describes the direction of the ray's propagation when it strikes an interface. According to this law, the ray, when being transmitted through an interface, suffers a rotation of an angle that depends on the ratio between the velocities before and after the interface.

The above two statements provide the basis to determine the trajectory of a ray, given source point and initial direction, i. e., it is possible to determine when and where the ray intercepts an interface. As a consequence, by positioning a source point on the intercepting point and computing the initial direction by the Snell's law, two new ray tracing problems are created, one for the transmitted and other for reflected rays. These procedure can be repeated as many times as necessary to trace all ray paths in a medium.

The paradigm of wavefront construction is based on Huygens' principle which states that every point on the wavefront is a secondary point source of a wavefront. In this way, considering a wavefront for a a given time, the wavefront for the next time step is the envelope of all secondary wavefronts generated by the point sources located on the previous wavefront.

Therefore, the wavefront construction can be seen a stepwise Huygens' principle. However, in this case, the wavefronts are a set of sampled spatial points and at each time step a new wavefront is constructed. Thus, each sampled point is considered as a point source for a ray equation, which is numerically integrated for one time step. The end points of each of those short duration rays form the newly sampled wavefront. Figure 1 illustrates the WFC method.

In our case, we rescricted our algorithms to the kinematic problem. This means that traces amplitudes are not considered, only the ray paths, wavefronts and traveltimes. Also, we have not included the ray density control commonly used in WFC algorithms.

Due to the WFC characteristic, the traveltime along the rays is automatically computed. However, only a finite number of rays is numerically traced, therefore the estimative of traveltime on each receiver (to buid a seismogram, for example) is a critical procedure.

In this work, we describe how to solve specifically two major geometrical problems which arise from the ray tracing technique, showing the computational geometry algorithms as well as some improving heuristics.

Wavefront Construction and Geometric Problems

In order to use the WFC method for seismic modeling purposes in a layered medium, it is necessary to take into account two major problems: (a) monitoring ray-interface interceptions and (b) estimating traveltimes on receivers;

More specifically, the WFC method for layered media can be described as the following algorithm:

Algorithm 1-1: Basic wavefront construction

1) Given a layered homogeneous medium, set the point source location and the shooting directions; Set the receivers locations;

2) Propagate a fan of rays, parameterized by traveltime, forming sampled wavefronts; Along the ray integration, monitor ray-interface interceptions, applying Snell's law accordingly;

3) Interpolate the traveltime at the receivers, using the sampled rays and wavefronts.

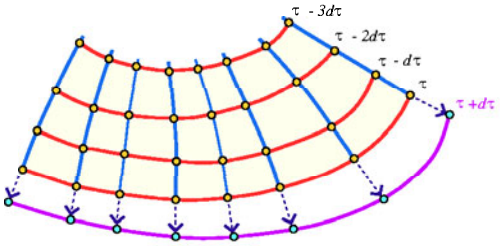

Figure 1: Illustration of of basic 2-D wavefront construction algorithm. The wavefronts are the red lines and the rays are the blue lines. The purple line is the newly constructed wavefront.

The computational geometry algorithms are mainly used in second and third steps of the WFC Algorithm 1-1. More specifically at each of these steps there is a main geometric problem. In step 2, there is the necessity of monitoring the ray-interface interceptions, in order to apply Snell's law. In step 4 it is necessary to identify which ray points will contribute to the traveltime interpolation, using the receiver position.

Ray-interface interception

The way WFC method works implies that a verification of intersection between ray segments an interfaces has to be done at each time step. As soon as a ray intercepts an interface it is necessary to apply the Snell's law, therefore, creating a reflected and a transmitted rays. In this case it is needed the normal vector to the interface computed at the incidence point. This vector can be computed by means of finite differences of sampled points along the interface.

Theoretically both rays and interfaces are plane curves (in 2D situation); However, due to the finite precision and WFC characteristics, they are sampled: the rays are composed by points computed by numerical integration of ray equations and the interfaces are sampled or in spline format.

This means that rays and wavefronts are, in fact, polygonal lines. A fair first approximation is to consider that interfaces are also sampled, meaning that they are polygonal lines as well. In this case, there is a very simple algorithm that computes the interception points between one ray and one interface:

Algorithm 2-1: Ray-interface interception

1) For each segment of the polygonal line which represents the interface, verify whether it intercepts each ray segment using the computational geometry result 1 presented at the Appendix.

It is not hard to see that the computational time of above algorithm depends linearly on the number of segments of the polygonal line.

In order to reduce the computing time, some speed-up heuristics can be included in the previous Algorithm 2-1:

intersection: If it is possible to find two disjoint polygons, P and Q, which contain respectively the ray segment and the interface segment there is no intersection between them;

x-coordinate comparison: If the horizontal projections of both segments are disjoints, they do not intercept.

Applying these heuristics, Algorithm 2-1 becomes:

Algorithm 2-2: Improved ray-interface interception

1) Construct the rectangle P which diagonal is the ray segment and which sides are parallel to the axes (dashed rectangle in Figure 1);

2) Construct the rectangle Q which sides are parallel to the axes and are the maximum and minimum coordinates of the sampled interface (green dashed rectangle in Figure 1);

3) if P and Q are disjoints then stop: The ray segment does not intercept the interface; else continue;

4) Find the segment s such that the x-coordinate of the ray segment's left extremity (LE) is between the extremities x-coordinates of s;

5) Find the segment e such that the x-coordinate of the ray segment's right extremity (RE) is between the extremities x-coordinates of e;

6) For each segment of the polygonal line from S until E (solid in Figure 2), verify if it intercepts the ray segment.

A complexity analysis of this algorithm, using a well-known binary search in steps 4 and 5, shows that its computational time is $O(2\log(n) + m)$, where n is the number of segments in the polygonal line and

$$m = \max_r\{\ell_r/\ell_i\},$$

where ℓ_r is the length of the horizontal projection of ray segment r and ℓ_i is the minimum length of the horizontal projection of interface segments.

This means that the algorithm is linear in the worst case, which is an improvement in comparison to the previous one, which is always linear. Otherwise, if the ratio m is reasonable, which is acceptable in ray tracing, the computational time is logarithmic on the number of segments in the polygonal line.

Point Location

With the purpose of estimating the traveltime at a receiver, a non-catesian grid is formed by subsequent wavefronts and adjacent ray paths. After that, the problem of locating a receiver on this grid becomes the problem of verifying whether a point is inside a four-sided polygon called cell.

In order to organize the data from the ray paths and wavefronts into cells is necessary to apply the following algorithm:

Algorithm 3-1: Organizing the isochron grid

1) Sort the rays points according to their x-coordinates resulting in a sequence of points;

2) For each pair of adjacent rays: group the points of

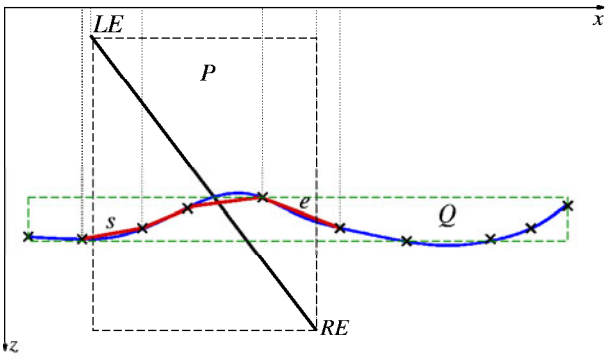

Figure 2: Illustration of improved ray-interface interception Algorithm 2-2. The blue line is the interface and the crosses are their sampled points. The ray segment is the black line segment $LE - RE$.

isochron of time t with the ones of isochron of time $t + dt$ (see Figure 3).

The time complexity of the above algorithm is $O(nlog(n) + n\,t)$, where n is the number of rays and t the isochron scale cardinality (the meaning of the "big oh" O notation can be found in Appendix).

Once Algorithm 3-1 is performed, takes place a simple algorithm which locates the receiver on the isochron grid is:

Algorithm 3-2: Basic point location

1) For each cell on the grid: if the receiver is inside, stop and return the cell index;

As it can be seen, the above algorithm has a computational time depending linearly on the number of cells. We can improve the above algorithm using some additional information, assuming that the ultimate goal is to perform a seismic modeling.

Firstly the isochron grid is kept fixed during all the location process, and, secondly, the points which are going to be located are seldom scattered, they have a spatial structure such as an array of receivers. These information can be used heuristically to reduce the set of eligible cells which contain the points. This is because, usually, near to a cell which contains a receiver there are neighbour cells which can contain another receivers, or even the same cell can contain more than one receiver.

The heuristics is to consider the eligible subset of cells only the ones that have some part intersecting the interface. This subset can be seen as the green cells in Figure 6. Therefore, this subset covers all the interface and, consequently, contains every possible receiver. The construction of this subset can be made with a procedure similar to Algorithm 3-1, having the same time complexity:

Algorithm 3-3: Grouping eligible cells

1) Sort the ray points accordingly to their x-coordinates;
2) For each pair of adjacent rays: store the cells which intersect the interface (see Figure 6).

Therefore, the optimized algorithm becomes:

Algorithm 3-4: Improved point location

1) For each eligible cell (cells in solid green line in Figure 6): if the receiver is inside the cell, then stop and return the cell index;

The computational time cost analysis of Algorithm 3-4 does not reveals much about its complexity. It varies for different models. Basically, it depends linearly on the number of sectors in the subset and does not vary for different receivers.

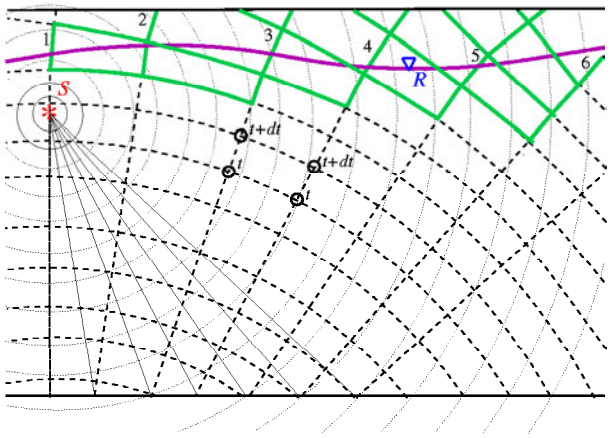

Figure 3: Illustration of optimized point location Algorithm 3-4. The source and receiver are indicated by S and R, respectively. The ray-isochron grid is dashed line and the subset of sectors is in solid purple line. The set of eligible cells are depicted in solid green line.

Example

In order to test all geometrical algorithms, we have developed a small MATLAB package which constructs and displays rays and wavefronts, for a homogeneous layered medium. This module also computes kinematic common shot sections with the primary reflections.

In Figure 7 we see an example of a medium composed by three layers separated by smooth interfaces. Also it is shown the reflected rays in second and third interfaces, respectively. For each ray the incidence points were computed using Algorithm 2-2 and the traveltimes were interpolated linearly inside the cells which contain the receivers, as a result of Algorithm 3-4. Figure 8 shows a kinematic common-shot section of the primary reflection events depicted in Figure 7.

Final remarks and conclusions

We have presented a set of geometrical algorithms designed to solve the problems which arise from the wavefront construction method. Basically two major problems are adressed: (1) the ray-interface interception and (2) the point location. Also it was shown that some heuristics can speed-up the algorithms. This is mainly because of the spatial structure provided by WFC method which comprises ubsequent wavefronts and adjacent rays, forming a non-Cartesian grid. Also, in the case of modeling, the re-

ceivers position have another spatial structure. These combined features allow the use of speed-up heuristics to improve the velocity and reliability of computational geometry methods.

Acknowledgments

The authors gratefully acknowledge the National Agency of Petroleum (ANP-PV-PRH15), National Council of Research (CNPq) and Wave Inversion Technology (WIT) consortium for financial support.

References

[1] VINJE, V., IVERSEN, E., & GJOYSTDAL, H., 1993, Traveltime and amplitude estimation using wavefront construction. Geophysics, **58**, 1157-1166.

Appendix

Basic Results of Computational Geometry

Here we show some basic results from computational geometry

1. Segment-segment intersection:

   Let $\overline{AB}$ and $\overline{CD}$ be the segments which intersection is sought. The first step to locate the intersection is to verify whether the segments really do intercept. This is true provided two (sufficient and necessary) conditions are satisfied:

   (a) The points C and D are in different semi-planes (Π_1 and Π_2) defined by the plane that contains the two segments and the straight line that contains A and B;

   (b) The points A and B are in different semi-planes (Π_3 and Π_4) defined by the plane that contains the two segments and the straight line that contains C and D.

   The first one is satisfied if the cross product between $\vec{AB}$ and $\vec{AC}$ has a opposite orientation of the cross product between $\vec{AB}$ and $\vec{AD}$ as illustrated by Figure 4. The other condition is completely similar.

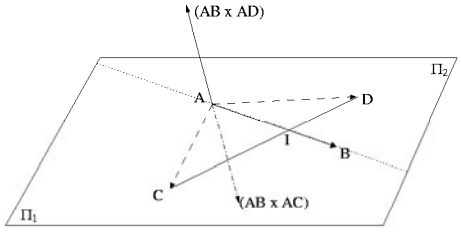

Figure 4: Illustration of a methodology to find the intersection point I between segments $\vec{AB}$ and $\vec{CD}$.

   Once assured that there exists an intersection, a simple algebraic formula can be used to compute exactly the intersection point I

$$I = A + \lambda \vec{AB},$$

   where

$$\lambda = \frac{\|\vec{AC} \times \vec{CD}\|}{\|\vec{AB} \times \vec{CD}\|}$$

   and $(\times)$ is the cross product operator.

2. Location of a point inside a four-sided polygon:

   A point P is inside a polygon $ABCD$ if, and only if, it is either inside the triangle ABC or inside the triangle CDA (as it can be seen in Figure 5).

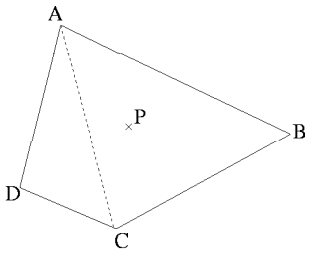

Figure 5: A point P inside a four-sided polygon $ABCD$

   The above statement means that the problem of locating a point in a four-sided polygon is reduced to locating the point in two adjacent triangles.

   In order to verify if a point is inside a triangle, all that is needed is to check if it belongs to the convex hull its vertexes. A point P belongs to the convex hull of $\{A, B, C\}$ if and only if there are α, β and γ, non-negatives, such that:

$$\begin{cases} \alpha\, A_1 + \beta\, B_1 + \gamma\, C_1 = P_1, \\ \alpha\, A_2 + \beta\, B_2 + \gamma\, C_2 = P_2, \\ \alpha + \beta + \gamma = 1 \end{cases}$$

   where A_k, B_k, C_k and P_k are the k-coordinates of points A, B, C and P, respectively.

3. The "Big Oh" notation (O)

   We say that a function $f : \mathbb{N} \to \mathbb{R}$ belongs to the set $O(g)$ if exists $c \in \mathbb{R}^+$ and $n_0 \in \mathbb{N}$ such that:

$$f(n) \leq c\, g(n), \quad \forall n > n_0.$$

   This means that, asymptotically, $c\,g(n)$ is an *upper bound* of $f(n)$. Figure 8 illustrates this concept.

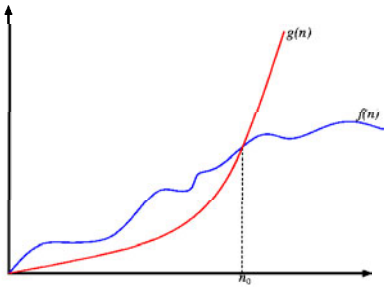

Figure 6: An example of "O" notation. $f(n) \in O(g(n))$

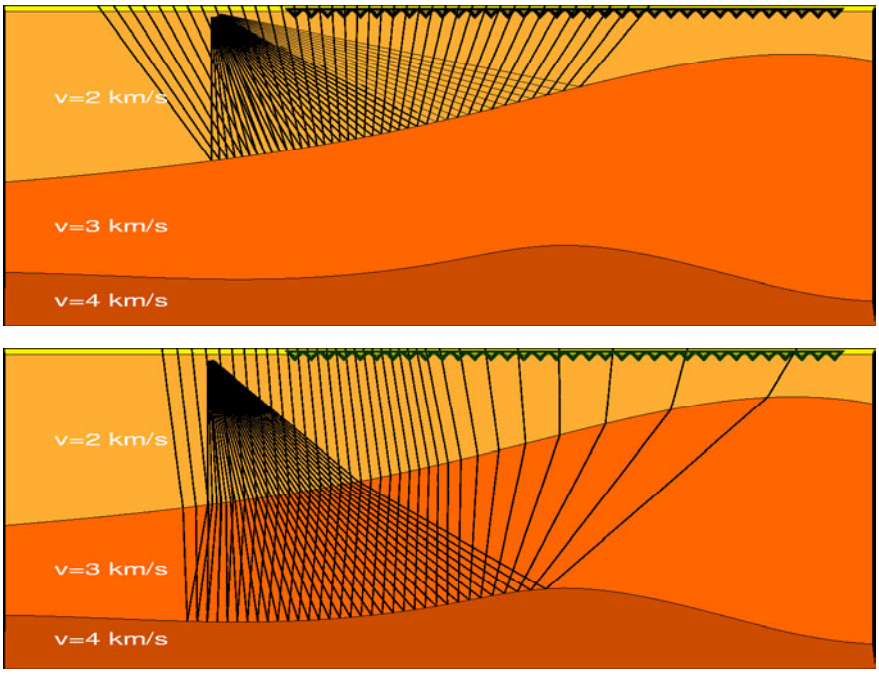

Figure 7: Running an example with the MATLAB program. Only the rays are depicted.

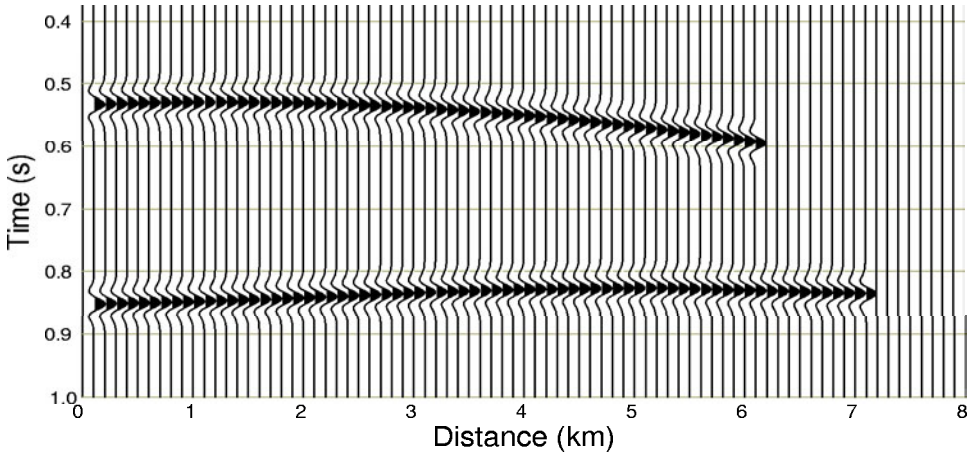

Figure 8: Kinematic common-shot seismogram of primary reflections shown in Figure 7.