Traveltime tomography using irregular parameterised grids

IMMO TRINKS

trinks@esc.cam.ac.uk

Motivation

Modern wide angle marine seismic surveys with simulated streamer lengths of up to 30 km contain long offset reflections and refractions that yield valuable additional information to conventional seismic data. The huge size of those densely sampled long offset data sets does not allow for an inversion with present tomography approaches. Typically it would be necessary to decimate the data set by discarding entire shot records in order to use one of the available methods. The current LITHOS tomography codes *Jive3D* and *CRAYFISH* are optimised for relatively sparsely-sampled data sets (OBS, borehole, etc.), and do not scale well to dense multi-channel surveys. Therefore an efficient inversion technique is needed, that is capable of handling such densely sampled data sets without the need of data decimation. Our aim is to develop an algorithm that is able to invert marine long offset data sets in their entirety.

In the following we present our approach to model parameterisation, ray tracing and inversion. It is based on the approach of McCaughey & Singh (1997) but uses an irregular parameterised gird of Delaunay triangles instead of a regular mesh, and incorporating not only traveltimes but also its derivatives, thus inverting the traveltime curve instead of traveltime points, as suggested by Zangh & Toksöz (1998). The geometry of marine long offset surveys can generally be regarded as two dimensional. Therefore our algorithm will be developed for 2-D applications. The chosen programming language is C++.

Model parameterisation

Regular parameterisations of velocity models are appealing because of their simplicity, but they can cause the over-parameterisation of large regions of the model in the case when high resolution of some structures is required. Therefore we use an irregular parameterisation to construct an optimal grid by adapting the local resolution to the available ray density. In two dimensions we can group irregular distributed velocity nodes together using Delaunay triangulation. A mesh of Delaunay triangles is exactly defined for each node distribution. A comparable parameterisation was described by Böhm *et al.* (2000). While their model is constructed of Voroni cells, that are the geometrical complement to Delaunay triangles, with constant velocities within each cell, our model uses triangular shaped cells with a linear interpolation of the squared slowness over each cell. Thus we avoid velocity discontinuities between cells and achieve a smooth ray path.

Our velocity model is composed of a sequence of layers separated by interfaces. It is necessary that those interfaces cross the model from one side to the other, but the layer thickness may be reduced to zero, allowing pinchouts and isolated bodies. Each interface is described by at least two interface nodes with horizontal position, depth and depth gradient values assigned. Currently only one depth gradient is assigned to each interface node but it is planned to define separate gradients on either side of the node, allowing for kinks in the interface. Between fixed interface nodes the interface is obtained by cubic polynomial interpolation.

The velocity within each layer is described by a set of velocity nodes which may be distributed randomly. These velocity nodes are gridded using Delaunay triangulation, currently with the 'Sweepline' triangulation algorithm developed by Steve Fortune (1987) as an external program. This algorithm is described to be more efficient than the 'Flipping', 'Divide-and-conquer' or 'Random-incremental' algorithms (Fortune, 1992). It is planned to incorporate the triangulation in our code to avoid extra file handling and to decrease run time.

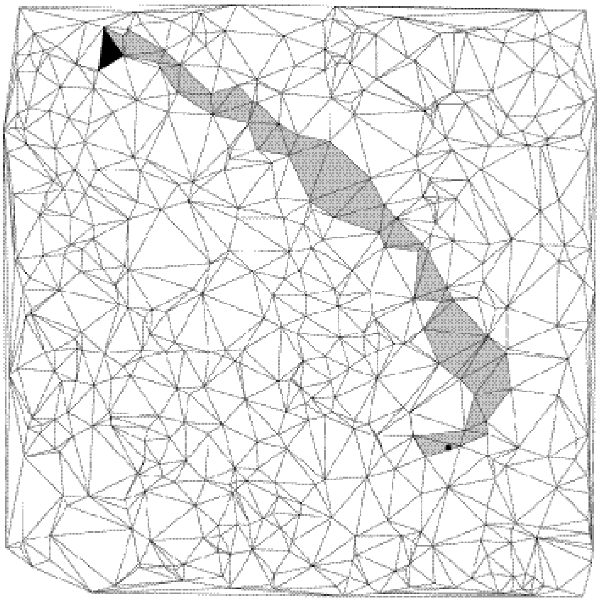

Figure 1. The 'Walking triangle algorithm' (see Sambridge & Guđmundsson, 1998) is used to locate the triangle that contains a given point **x** (black point) in this grid of 985 triangles. Starting from a randomly chosen triangle (black) we test only those triangles that lie on the way (grey) to **x**, in this case 48.

The area covered by the triangulated grid is allowed to exceed the definition of the layer, and grids may overlap to allow the interfaces to move during the inversion. Identical grids can be used for different layers. Consequently, velocity discontinuities across layer boundaries are allowed but not required. Velocity nodes may either be provided in the form of an ascii file or they can be created using an internal random node generator. Once the irregular node distribution is triangulated, the squared slowness at each velocity node is computed and interpolated linearly over the area of the Delaunay triangles.

One of the main book-keeping problems encountered with an irregular mesh of triangles is to find the one containing a given point **x**. A brute force search through all of the triangles becomes highly inefficient when a large number of triangles is involved and when many points need to be located. Therefore, the 'walking triangle algorithm' as described by Sambridge & Guđumdsson (1998) was implemented. Starting from a randomly chosen triangle it checks on which side of the current triangle the point **x** is located. Then it progresses towards the neighbouring triangle in direction of **x** until **x** is contained in the current triangle.

Figure 1 illustrates this search algorithm for a grid of 985 triangles that was formed through the triangulation of 500 velocity nodes. The starting triangle is the randomly chosen black triangle. The algorithm aborts when the black point **x** is located inside the current triangle. Thus, in this case, only 48 triangles of 985 had to be tested. The efficiency of this search can be further optimised by selecting a few randomly chosen triangles and starting with the one that is closest to **x**, as well as by subdividing the mesh into smaller search-sections. We keep the last triangle of each search in memory and use it as the starting triangle for the next search procedure.

The 'walking triangle algorithm' is used to locate a specific triangle every time the source position of a ray has to be determined. Furthermore this algorithm is used intensively whenever the ray intersects an interface, as discussed later. Whenever a ray intersects an interface, the intersection location is assigned as the new source position for each refracted or reflected branch.

Adaptive gridding

Böhm *et al.* (2000) described an algorithm for automatic regridding that is able to fit the local resolution to the available ray paths. The algorithm inserts additional velocity nodes in regions of large velocity gradients and reduces them elsewhere. Furthermore, it takes the null space energy into account as an additional criterion for the choice of optimal node distribution. In the present implementation of the code it is so far possible to reduce the number of velocity node points in regions of low velocity contrast, but future versions will provide as well the possibility to increase the number of nodes in regions of strong contrast. Thus, we can reduce the number of triangles that do not contain any rays and thereby minimise the number of parameters in the inversion. New developments to be incorporated include the possibility of increasing the node density to adapt the tomographic grid to the velocity gradient, and the possibility to control the shape of triangles to avoid very thin triangles (which occur where the size of the triangles varies largely, see **Figure 2**).

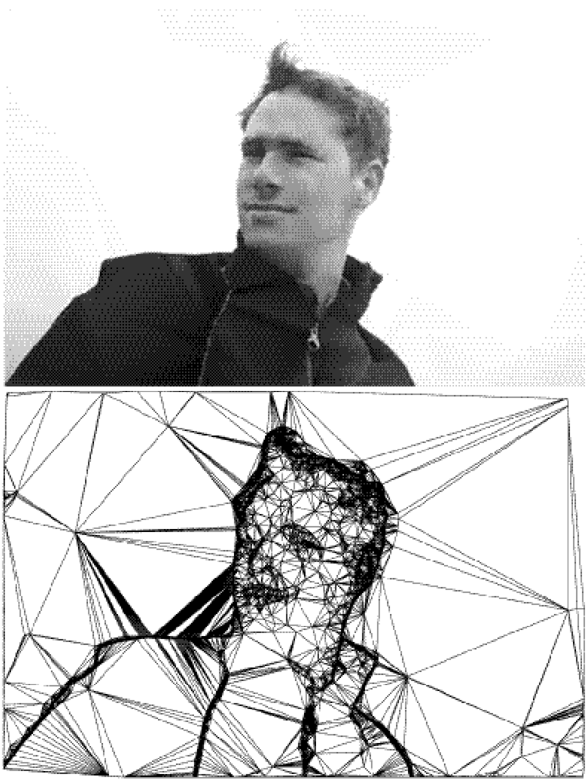

Figure 2. Adaptive gridding by reducing the node points in regions of low 'velocity' gradient.

Figure 2 shows the result of an adaptive reduction of a regular grid of 194,937 node points (543×359 pixels) to an irregular grid of 11,368 nodes. The upper picture displays the linear interpolation of the squared slowness over the triangulated grid after reduction, which is displayed below. Regions of fine structure are sufficiently represented and well perceivable while over-sampling of areas with low contrast is avoided.

Ray tracing

To ensure the efficiency of the code we use analytic initial-value ray tracing in an isotropic medium, as formulated by Farra (1990). The two-point problem is avoided by shooting a equidistant fan of rays and interpolating the traveltimes at the receiver locations. The quadratic slowness ($1/V^2$) model was chosen instead of the propagation velocity V, since it offers the simplest analytical solution for an inhomogeneous medium with constant gradients (Červený, 1987).

The source may be positioned anywhere in the model except in layers with zero thickness. Rays may be shot in any direction, given by the angle with the vertical. Currently, rays can be modelled that either reflect from a certain interface or turn in a specified layer, and the user provides a list of interfaces from which the ray shall reflect, or alternatively a layer in which the ray shall turn.

Each ray is constructed of ray-segments, which describe the ray path within single triangles. All ray-segments between the ray source and an interface intersection or between two ray interface intersections are combined into ray-macro-segments. The sum of those ray-macro-segments makes up one single ray.

For efficiency reasons only those triangles along the ray path that are crossed by an interface are checked for a ray-interface intersection. When the ray enters a triangle that contains an interface a possible ray-interface intersection is computed numerically in two steps. First we step along the ray in increments and check whether the test point is still on the same side of the interface as the previous point. Once the test point is located in a different layer (this might be the next layer, but can as well be some other layer, since layers of zero thickness are allowed) the stepping procedure aborts and the intersection is approximated using bisection until the distance between two successive test points, each one on either side of the interface, has fallen below a predefined tolerance. The bisection method converges linearly. We are in the process of replacing it with the more efficient Newton-Raphson method with a quadratic convergence at best (Press *et al.*, 1992).

At the ray-interface-intersection Snell's law is applied and the new direction of the reflected and refracted branches computed. Since it is possible to choose which route the rays take at each interface within the model, it is also possible to model arrival times for phases other than the first to arrive, e.g. to study multiple reflections.

At the receiver end traveltimes will be interpolated between the bracketing rays. Additional rays will be shot where the number of rays is too sparse. In case of a triplication in the traveltime curve a simple bisection technique will be employed to calculate the desired traveltime and slowness values. The computation of amplitudes might be incorporated later on.

Figure 3 shows a simple example for ray tracing within a velocity model with constant velocity.

Inversion

Our aim is to determine the velocity distribution and the shape of the reflectors by joint reflection and refraction traveltime tomography. Therefore, we plan to implement a linearized simultaneous inversion of reflector depths and velocities as described by McCaughey & Singh (1998).

Instead of inverting in an iterative manner for only one parameter at a time, or inverting simultaneously velocities and depths using a fine discretization, we plan to reduce the number of parameters that describe the model to the necessary minimum whilst incorporating *a priori* information by positioning the nodes and inverting then simultaneously for velocities and interface depths. The optimised node distribution allows us to avoid over-parameterisation whilst at the same time we are still able to sample sufficiently finely in regions of high ray density and structural complexity.

To achieve a higher efficiency of the inversion it is planned not to invert each single traveltime value but to incorporate instead the local slowness as additional information about the traveltime curve. Thus, we will invert traveltime curves rather than traveltime points alone.

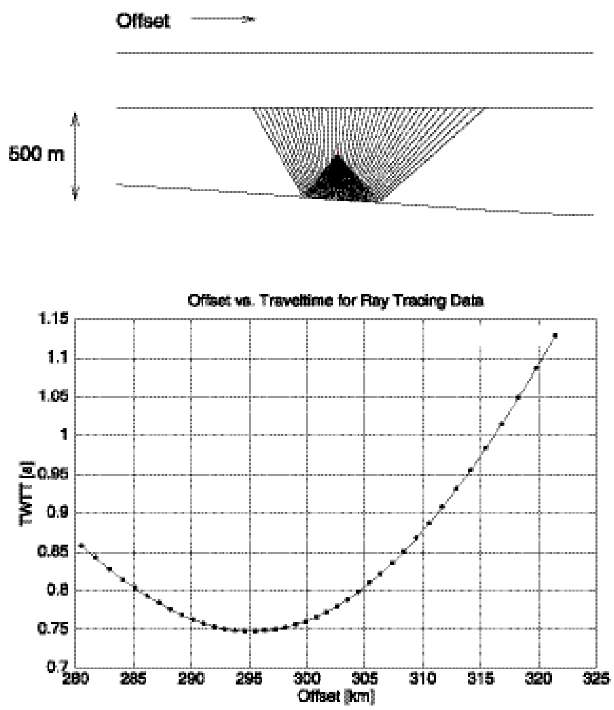

Figure 3. Example for ray tracing and traveltime computation in a velocity model with constant velocity.

Outlook

In 2001 we expect that the ray tracing and inversion code will be completed and optimised. Therefore, as a next step the computation of ray-perturbations will be implemented, which will lead to the formulation of the inversion. Also, the positioning of receivers will be improved to allow arbitrary placement, for instance to allow for the incorporation of OBS and VSP data.

First synthetic tests of the inversion algorithm are planned for the end of 2001 and will be followed by the application of the code to a real marine wide angle data set.

The new ray tracing scheme has potential for application to a model based wide angle multiple suppression scheme. It could be used to predict locations of multiples and thereby define reject zones of a filter in a suitable transform domain, e.g. the τ-p domain, as proposed by Zhou & Greenhalgh (1996).

Acknowledgements

I thank Satish Singh, Chris Chapman, Penny Barton, Hamish Macintyre and especially Adam Cherrett for their support and good ideas. I am grateful to the LITHOS Consortium for funding this studentship.

References

BÖHM, G., GALUPPO, P., & VESNAVER, A.A., 2000, 3D adaptive tomography using Delaunay triangles and Voronoi polygons, *Geophysical Prospecting*, **48**, 723-744.

ČERVENÝ, V., 1987, Ray tracing algorithms in three-dimensional laterally varying layered structures, In: *Seismic Tomography*, Ed.: Nolet, G., 99-133.

MCCAUGHEY, M., & SINGH, C.S., 1997, Simultaneous velocity and interface tomography of normal-incidence and wide-aperture seismic traveltime data, *Geophys. J. Int.*, **131**, 87-99.

FARRA, V., 1990, Amplitude computation in heterogeneous media by ray perturbation theory: a finite element approach, *Geophys. J. Int.*, **103**, 341-354.

FORTUNE, S., 1987, A Sweepline Algorithm For Voronoi Diagrams, *Algorithmica*, **2**, 153-174.

FORTUNE, S., 1992, Voronoi Diagrams and Delaunay Triangulations, *Euclidian Geometry and Computers*, 193-233.

PRESS, W., TEUKOLSKY, S., VETTERLING, W., & FLANNERY, B., 1992, *Numerical Recipes in C: The Art of Scientific Computing*, 2nd ed., Cambridge Univ. Press.

SAMBRIDGE, M., & GUDMUNÐSSON, Ó, 1998, Tomography with irregular cells, *Journal of Geophysical Research*, **103** (B1), 773-781.

ZHANG, J., TEN BRINK, U., & TOKSÖZ, M. N., 1998, Nonlinear refraction and reflection travel time tomography. *J. Geophys. Res.*, **103** (B12), 29743-29757.

ZHANG, J., & TOKSÖZ, M. N., , 1998, Nonlinear refraction traveltime tomography. *Geophysics*, **63** (5), 1726-1737.

ZOUH, B., & GREENHALGH, S. A., 1996, Multiple suppression by 2D filtering in the parabolic τ-p domain: a wave-equation based method, *Geophysical Prospecting*, **44**, 375-401.