RSTTI package:
Ray based Seismic Travel Time Inversion

Stéphane Operto

University of Texas Institute for Geophysics
Technical Report No. 148

RSTTI package:
Ray Based Seismic Travel Time Inversion

LIST OF CONTENTS

1. Introduction

This report describes improvements of an existing package (***RAYINVR***) [*Zelt and Smith*, 1992]. The ***RAYINVR*** package performs 2-D ray tracing and iterative damped least-squared travel time inversion. The improvements of ***RSTTI*** package are:

- a new inversion scheme which uses: (1) a conjugate gradient algorithm (2) smoothing and variable damping regularizations.
- ray tracing and travel times inversion of normal incidence reflected waves.
- simultaneous travel time inversion from 2-D intersecting profiles using common parameters [*Zelt*, 1994].

Objectives of these tools are:

- to be able to process large number of travel times and model parameters.
- to have a robust inversion scheme in the case of non uniform ray coverage and to have a good control on the smoothness of the model.
- to allow simultaneous travel time inversion of coincident normal incidence reflection data and wide angle reflection/refraction data. Normal incidence reflection data provide good control on the geometry of seismic interfaces and refraction data provide good control on the velocities within the layers delineated by these interfaces.
- to estimate a preliminary 3-D velocity model from a grid of 2-D in-line profiles. This preliminary model can be used as a starting model for 3-D seismic tomography using a linear inversion scheme.

The first part of this report briefly reviews the ***RAYINVR*** package.

In the second part, the ***RSTTI*** package is presented. First, the 3 improvements mentioned above are described (inversion scheme, normal incidence ray tracing, inversion for a grid of 2-D lines). Second, the programs are presented including description of the related IO (input/output files and parameters).

In the third part, each tool of the ***RSTTI*** package is illustrated with synthetic or real data examples. The first and second examples illustrate how to use the inversion scheme (particularly how to judiciously use damping and smoothing regularizations). For this, ray tracing and travel time inversion are performed for two wide angle seismic profiles. In the first example, a coarse parametrization is used which is characteristic of the one used with ***xrayinvr***. In the second example, a finer parametrization (called hybrid parametrization as it combines layered and gridded parametrizations) is proposed to take advantage of the new inversion scheme. The third example illustrates how to run the normal incidence reflection ray tracing and inversion. The fourth example describes how to perform a simultaneous inversion for intersecting 2-D profiles.

In appendix 1 the content of the ***RSTTI*** package is listed and compilation instructions are given.

In appendix 2 descriptions of several programs for pre-processing are provided. These programs allow to set bathymetry in the starting velocity model, to format the file containing observed travel times, to write starting model automatically and to prepare inputs for the simultaneous inversion of a grid of 2-D lines.

In appendix 3 programs for automatic picking of travel times and estimation of their relative uncertainties are presented.

Most of the programs described in this report are modification of the programs of the ***RAYINVR*** package, and it is assumed that the reader is familiar with this package. I shall refer frequently to the documentation of ***RAYINVR*** and the paper of *Zelt and Smith* (1992). Part of this work was also inspired by *Wang* (1993).

2. The existing software: RAYINVR [*Zelt and Smith*, 1992]

2.1. Brief description

Ray tracing

The ***xrayinvr*** program allows the user to perform a shooting or a two-point ray tracing for a 2-D velocity model. Rays are computed numerically. Ray tracing is stabilized in the case of a complex models using smoothing of the interfaces. The velocity model is parametrized in term of layers delineated by first- or second- order discontinuities. The interfaces delineating the layers and the velocities at the top and at the bottom of the layers are independently parametrized with an irregular grid of nodes. Layer pinch-out can also be modeled. Wide angle reflected rays, refracted (turning) rays and head-waves can be traced in the model. Multiples and P- to S- mode conversion can also be computed.

Travel time inversion

During ray tracing, partial derivatives of travel times with respect to model parameters (depths and top and bottom velocities) can be computed analytically. The vertical gradients in a layer or the thickness of a layer can be kept fixed during an inversion. The output returned by ***xrayinvr*** can be used by the program ***dmplstsqr*** as input to perform a linearized damped least-squared travel time inversion. The ***dmplstsqr*** program returns an updated velocity model. This velocity model can be used as the starting model for the next iteration. Thus, the model is updated iteratively by successive ray tracing and travel time inversion. There is no fixed rule for stopping criteria of the iterations. Commonly, changes in the RMS (root-mean-square) erro, $\chi 2$ (chi-square) error and the ray coverage in the model are used by the user to select a stopping criteria. During inversion a layer stripping approach is generally used processing from top to bottom of the structure. The inversion scheme generally requires a coarse parametrization (which will depend among other of the density of sources and receivers) of the model to be efficient. Parameters which are not 'covered' by the rays will not be updated after inversion (resolution for these parameters is zero). This is often the case for the parameters located on the sides of the model. Thus, editing of the model is often required after an inversion to remove artifact due to non uniform ray coverage.

2.2. How to improve it?

A new inversion scheme

A new inversion scheme was implemented to be able to process a large number of travel times and model parameters in the case of a non uniform ray coverage.

A conjugate gradient algorithm by opposition to a least-squared algorithm is suitable for sparse matrix inversion. It is faster saves disk space in the case of large and sparse matrix (the zero elements of the partial derivatives matrix are not stored). Regularizations were implemented to stabilize the inversion in the case of non uniform data coverage and to control the smoothness of the velocity model.

Inversion of normal-incidence reflection data

Normal incidence reflection data and wide angle reflection/refraction data are often recorded simultaneously. Normal incidence reflection data can be incorporated in the travel

time analysis. The two-way travel times of the reflectors are picked on the reflection profiles and are inverted to update the depth of the seismic interfaces. This complements the more classic travel time inversion of refraction data which provides a better control on the seismic velocities. In the programs of the ***RSTTI*** package, ray tracing and inversion of normal incidence reflection data and wide angle reflection/refraction data are performed simultaneously.

Inversion from a grid of 2-D profiles

A 3-D velocity model (although the method is not 3-D) can be inferred from simultaneous travel time inversion from a grid of 2-D in-line profiles using common parameters between several intersecting profiles [*Zelt*, 1994]. The 3-D velocity model is built by interpolation between adjacent 2-D velocity models. The conjugate gradient algorithm is particularly suitable for this approach because the matrix tends to be very sparse (as the intersecting profiles share a small number of parameters along one vertical axis). Moreover, if the number of profiles is large, a fast algorithm (like the conjugate gradient algorithm) is required as the size of the matrix to be inverted tends to increase.

3. The RSTTI package

3.1. The new inversion scheme

Conjugate gradient algorithm

The subroutine *sparse* [*Press et al.*, 1986] is used to perform matrix inversion. The matrix is stored using 3 vectors: the first one stores the number of non zero elements per row of the matrix; the second stores the column index of each non zero elements of each row; the third one stores the value of each non zero element of each row.

Regularizations

The system of linear equations which relate the travel time residuals to the perturbational model is augmented with smoothing and variable damping regularizations

$$\begin{pmatrix} C_d.\partial t \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} C_d.A \\ \lambda.C_s \\ \varepsilon.C_p \end{pmatrix} . \begin{pmatrix} \partial m \\ m \end{pmatrix} \qquad (1)$$

$$C_d = \begin{pmatrix} \frac{1}{\sigma_i} \end{pmatrix} \qquad (2)$$

is the diagonal matrix with standard deviation of observed travel times. ∂t are the travel time residuals. C_s is the smoothing matrix. C_p is the variable damping matrix. λ and ε are the two factors which together balance constraints provided by the data and the *a priori* information provided by regularization. The relative value between λ and ε parameters control weight between smoothing and damping during inversion. Practically, damping controls the magnitude of the perturbations (high damping will lead to small perturbations) while smoothing controls the relative perturbation between adjacent nodes (large smoothing will lead to a flat perturbational model, i.e., perturbations applied to adjacent nodes will be closed). Typically, first iterations must be performed with large smoothing and low damping. When the fit is in the range of the travel time uncertainty, damping must be increased while smoothing is decreased. Details about this strategy are provided in *Wang* (1993).

Smoothing matrix

Smoothing regularizations with distance are formulated with a system of linear equations. These equations relate linearly each normalized perturbational parameter to the other perturbational parameters. For the i^{th} perturbational model parameter, a smoothing constraint takes the following form [*Toomey, et al.*, 1994]

$$\frac{\partial m_i}{m_i} = \frac{\sum_{j=1}^{m} \beta_j \frac{\partial m_j}{m_j}}{\sum_{j=1}^{m} \beta_j} \qquad (3)$$

m_j is the j^{th} model parameter and ∂m_j is the j^{th} perturbational model parameter.

The weights are defined with a Gaussian of the form [*Toomey, et al.*, 1994]

$$\beta_j = \exp\left\{-\frac{(x_j - x_i)^2}{\tau_x^2} - \frac{(z_j - z_i)^2}{\tau_z^2}\right\} \quad if (x_j - x_i)^2 \le \tau_x^2 \text{ and } (z_j - z_i)^2 \le \tau_z^2 \qquad (3)$$

$$\beta_j = 0. \quad if \quad (x_j - x_i)^2 \ge \tau_x^2 \text{ or } (z_j - z_i)^2 \ge \tau_z^2$$

The Cartesian coordinates (x,z) are chosen relative to the upper left corner of the velocity model. Decay parameters control the correlation length in the horizontal and vertical direction.

Smoothing is applied simultaneously in the horizontal and vertical direction for simplicity. This approach differs from that of *Toomey, et al.* (1994) who applied horizontal and vertical smoothing separately.

The model is parametrized in term of layers delineated either by first- (velocity discontinuity) or second- (gradient discontinuity without velocity discontinuity) order discontinuities. To be consistent with the terminology used in *Zelt and Smith* (1992), I shall define 'unit' as a part of the model delineated by two first-order discontinuities. For example, the user can set several layers delineated by second-order discontinuities as a unit to mimic a grid of nodes (see example 2). Smoothing for depth and velocity nodes is applied separately to several sets of nodes located in the same unit both in horizontal and vertical directions (on the contrary, two nodes located in two different units will never be related by a smoothing linear equation).

*Variable damping matrix [*Wang, *1993]*

Variable damping matrix is defined by a diagonal matrix C_p whose elements are

$$d_j = \frac{DWS(m_j).(Max_k(DWS(m_k))^{-1}}{\sigma_{m_j}^2} \qquad (5)$$

$$\text{where } DWS(m_j) = \sum_{i=1}^{n} \frac{\partial t_i}{\partial m_j} \qquad (6)$$

is the density of ray path for perturbational model parameter j [*Thurber*, 1983].

σ_{m_j} is the a priori uncertainty (standard deviation) assigned to the parameter mj. Variable damping in contrast to constant damping is used to avoid exaggerated control on parameters for which the ray path density is high.

Normalization

Each perturbation can be normalized with respect to the original value of the model parameter.

3.2. Normal Incidence Ray Tracing and Inversion

Normal incidence rays are traced using the "*exploding reflector*" concept [*Wang*, 1993]. The ray initiates at the reflected point and is propagated upward to the surface with a take off angle of 90 degrees. The ficticious source (reflected point) is moved with a constant increment along the interface. Travel times are multiplied by two to account for the two-way propagation (down- and up- going). Practically, the existing subroutines (***subroutine trc*** among other) to trace rays in ***RAYINVR*** were used using a ficticious source along the interfaces and fixing the take off angle to 90 degrees. Partial derivatives are computed analytically using the existing formulas implemented in ***RAYINVR***.

3.3. Travel time inversion from intersecting in-line 2-D profiles

The method for simultaneous travel tinversion from intersecting in-line 2-D profiles is described in *Zelt* (1994). Ray tracing is performed separately for each 2-D profile. During each ray tracing, the partial derivatives of travel times with respect to model parameters are computed for the current profile. When all the partial derivatives matrix are computed they are merged together to form one unique matrix. The model parameters of two different profiles located at the intersection, if at the same depth, are processed as one parameter (i.e., the partial derivatives of travel times with respect to these parameters are in the same column of the matrix). This implies that after inversion the intersecting 2-D profiles have the same properties at their intersection (if the starting models also have the same properties at their intersection). A 3-D velocity model can be built by interpolation between the grid of 2-D profiles. This method is suitable to provide a preliminary 3-D model in the case of 3-D slowly varying structure. The distance between two parallel profiles must be larger that the horizontal resolution provided by the ray coverage. Otherwise, artifacts will be introduced due to decoupling between the 2-D profiles.

3.4. Description of the programs

- ***XRAYINV.CGN***

Ray tracing for normal incidence reflected waves and wide angle reflected/refracted waves.

IO:

Input files:

- **v.in:** velocity model.
- **r.in:** input parameters for ray tracing and further inversion.
- **tx.in:** observed travel times for wide angle reflection/refraction data.
- **txn.in:** observed travel times for normal incidence reflection data.

Output files:

Output files used as input by ttinver.cg

- **idt.spars.out:** number of non zero elements for each row of the partial derivative matrix.

- **idt. indx1.out:** column index of each non zero element of the partial derivative matrix.
- **idt.indx2.out:** value of each non zero element of the partial derivative matrix.
- **ip.out:** information concerning the model parameters.
- **ires.out:** travel time residuals
- **iunc.out:** a priori travel times uncertainties

Other output files:

- **ip.out.ascii:** formatted version of ip.out
- **frays (optional):** coordinates of the ray traced
- **tx.out (optional):** computed travel times for wide angle reflection/refraction data
- **txn.out:** computed travel times for normal incidence reflection data

v.in, tx.in and **tx.out** have the same format that of ***RAYINVR***.

r.in has the same basic format (namelists *PLTPAR, AXEPAR, TRAPAR, INVPAR*) but additional parameters were added to the namelists *TRAPAR* and *INVPAR* for the normal incidence ray tracing (see description below).

txn.in and **txn.out** has the following format:

x, t, unc, code (3f10.3, I10) where x is the distance from the origin of the model to the distance at which the travel time is recorded, t is the two-way travel time, unc is the uncertainty assigned to the travel time and code is an integer used to identify a set of travel times reflecting from a given interface (fixed by the user). The parameter ***ivrayn*** in the file **r.in** correlates the computed and observed travel times (similar implementation to the one used for wide angle reflection/refraction travel times inversion in ***RAYINVR***).

txn.out has the same format as that as *txn.in* and stores the computed travel times for normal incidence ray tracing.

Input parameters:

The input parameters which differ from that of program ***RAYINVR*** are described here. They are all related to the normal incidence ray tracing except ***ipray*** which allows the user to store the ray coordinates.

Input parameters related to namelist PLTPAR:

- ***ipray:*** store (ipray=1) the ray coordinates in the file ***rayfile*** (default: 0); each ray is separated by '>' (format suitable for GMT plot).

Input parameters related to namelist TRAPAR:

- **irayn:** plot normal incidence rays (1) or not (0) (default: 0)
- **ilayern:** array specifying index of layers from top of which normal incidence rays reflect.
- **dxn:** array specifying horizontal increment between rays defined by array ilayern.

Input parameters related to namelist INVPAR:

- **invrn:** compute partial derivatives of the normal incidence travel times with respect to the selected model parameters (default: 0).
- **ivvn:** compute partial derivatives of the normal incidence travel times with respect to depth and velocity parameters (0) or depth parameters only (1) (default: 0).
- **xtoln:** the maximum distance allowable between the nearest normal incidence reflected ray end point and an observed seismogram location for interpolation of the partial derivatives and travel time at that observed seismogram location (default: 0.5).

- **ivrayn**: an array of non-zero integers corresponding to the normal incidence ray groups listed in the array ilayern used to allow for the appropriate comparison with the observed travel times in the file ***txn.in*** if **irayn**=1.

To run the program:

When the inputs are correctly set (v.in, r.in, tx.in and optionally txn.in), just type ***xrayinv.cgn*** and return. Once the ray tracing is finished the number of observed travel times for which partial derivatives were computed, the RMS error and the χ^2 error are printed on the screen.

• ***TTINVER.CG***

Linear travel time inversion using a conjugate gradient algorithm. Uses the output returned by ***XRAYINV.CGN***.

IO:

Input files:

- **v.in:** velocity model used for previous ray tracing.
- **idt.spars.out:** number of non zero elements for each row of the partial derivatives matrix.
- **idt. indx1.out:** column index of each non zero elements of the partial derivatives matrix.
- **idt.indx2.out:** value of each non zero elements of the partial derivatives matrix.
- **ip.out:** informations concerning the model parameters.
- **ires.out:** travel time residuals.
- **iunc.out:** *a priori* travel times uncertainties.

Output files:

- updated velocity model.

Input parameters:

- **iopd:** apply damping (y/n).
- **epsilone:** factor applied to damping matrix.
- **bndunc, velunc:** *a priori* uncertainties assigned to depth and velocity parameters respectively.
- **iops:** apply smoothing (y/n)
- **lambdav, lambdad:** factors applied to smoothing matrix for velocities and depth parameters.
- **iopsl:** apply a specific lambda factor for one layer (y/n). A specific lambda factor can be applied to the velocity nodes of one of the layer of the model. This option was implemented because velocities in the shallower layer of the velocity model (for example, the sedimentary layer) are less constrained that the deeper ones and thus requires larger smoothing.
- **ilays**: Layer index for which specific smoothing must be applied
- **lambdas**: specific lambda for the layer ilays.
- **tauxv, tauzv, tauxd, tauzd:** decay parameters in the x direction for velocities, in the z direction for velocities, in the x direction for depths, in the z direction for depths.

To run the program:

Input parameters can be set in a file called parfile for example.
Type ttinver.cg < parfile.

Check the format of input files for program ***ttinver.cg*** (called par_it(i).in) where (i) is the iteration number in the directory **EXAMPLES**.

• ***RAYDENS***

Compute the ray path distribution in the velocity model. The display of the ray path distribution is useful to define an optimized parametrization of the model (variable grid spacing) which takes into account the ray coverage [*Wang*, 1993]. To display the ray path distribution, the practical procedure is:

- generate a velocity model with a constant grid spacing.
- trace rays in this model (setting a 1 below each parameter if those parameters should be involved in further inversion)
- run ***raydens*** program
- grid the ray path distribution model returned by ***raydens*** (setting parameters *igrid*=1, *xgrid*=dx and *zgrid*=dz in namelist *PLTPAR* of file **r.in** and running program ***xtramp*** of ***RAYINVR*** package; this program will return a file **m.out** which stores the gridded velocity model).
- display the gridded model (with ***GMT*** software [*Wessel and Smith*, 1991], for example).

IO:

Input files:

- **v.in:** velocity model used for previous ray tracing
- **idt.spars.out:** number of non zero elements for each row of the partial derivative matrix.
- **idt. indx1.out:** column index of each non zero element of the partial derivative matrix.
- **idt.indx2.out:** value of each non zero elements of the partial derivative matrix.
- **ip.out:** informations concerning the model parameters

Output files:

- **densray.out:** file with the same format as *v.in*. The values of the ray path distribution at each velocity node are stored instead of the values of the velocity.

Input parameters:

• **xmax**

• **ilmin, ilmax:** range of layer indexes for which ray path distribution must be computed. The nodes which belong to layers outside this range will have a value of 0.01. These values are not set to 0 to allow gridding of the 'ray density' model by interpolation.

• ***XRAYINV.3DCGN***

Ray tracing for intersecting 2-D profiles (ray tracing, followed by update of output files, is performed separately for each profile. The output files are used as input by the program ***ttinver.3dcg*** for the simultaneous travel time inversion; see example 4 for more details).

IO:

Input files:

- ***parindx0.out*** *(returned by program column_indx1; see appendix 2)*
- ***parindx1.out*** *(returned by program column_indx1; see appendix 2)*
- velocity model (v.in) for each 2-D profile

- observed travel times file (*tx.in*) for each 2-D profile
- parameter file (*r.in*) for each 2-D profiles
- optional normal incidence travel times files *(txn.in)*.

Input parameters:

- same as ***xrayinv.cgn***

Output files:

- same as ***xrayinv.cgn*** but files **idt.spars.out, idt.indx1.out, idt.indx2.out, ires.out** and **iunc.out** store the partial derivatives matrix, the residual vectors and the standard deviation of travel times of each of the 2-D profiles sequentially; thus, these files are updated after each ray tracing to store the results of ray tracing related to the currently processed profile. The partial derivatives matrix for simultaneous inversion is generated at the beginning of the program ***ttinver.3dcg*** only.

• ***TTINVER.3DCG***

Simultaneous travel time inversion from intersecting 2-D profiles.

IO:

Input files:

- **parindx0.out**
- **parindx1.out**
- velocity model (v.in) for each 2-D profile
- observed travel times file (tx.in) for each 2-D profile

Input parameters:

- same as ***ttinver.cg*** but factors ***lambdad, lambdav*** and ***epsilone*** (for regularization) must be provided for each profile.

Output files:

- Updated velocity models for all 2-D profiles.

4. How to use the programs?

4.1. Introduction

In this section, three kind of applications are presented to illustrate the main features of the ***RSTTI*** package. The first example consists of ray tracing for wide-angle reflection/refraction data and travel time inversion. This example illustrates how to use the input parameters of the program ***TTINVER.CG*** for travel time inversion. The second example consists of simultaneous ray tracing for normal incidence reflection data and wide angle reflection/refraction data and inversion. This example illustrates how to use the input parameters for normal incidence ray tracing which is an option of the program ***XRAYINV.CGN***. The third example consists of simultaneous travel time inversion from four intersecting 2-D profiles. This example is to illustrates how to use the programs ***XRAYINV.3DCGN*** and ***TTINVER.3DCG***.

4.2. Wide-angle reflected/refracted travel time inversion (programs *XRAYINV.CGN* and *TTINVER.CG*)

General procedure

The general procedure is similar to the one used in ***xrayinvr*** and ***dmplstsqr***: it consists of successive ray tracing and inversion. Each inversion step returns an updated velocity model used as input by the ray tracing program. Like ***xrayinvr***, there is no fixed criterion for stopping iterations. Generally, the user checks at the end of the ray tracing the *RMS* (root-mean-square) residual and the ray coverage (i.e., number of observed travel times used by inversion). Iterations are performed until the *RMS* decreases while the ray coverage increases or remains constant and the velocity model is geologically plausible.

The inversion scheme implemented in ***ttinver.cg*** must be understood as an 'adaptive inverse modeling tool rather than a black-box inversion method' [*Toomey, et al.*, 1994]. This means that, through the input parameters of the program ***ttinver.cg***, the user has to define a prior state of (subjective) information about the structure (i.e., the relative heterogeneity of the structure in the horizontal and vertical directions). The regularizations allow the user to test different kind of parametrizations (i.e., coarse, fine, layered, uniformly and non uniformly gridded parametrizations). The judicious use of smoothing and damping allows the user to control the smoothness of the velocity model.

Example 1: Line 229 - TICOSECT (Leg 2). Coarse parametrization.

a. The data:

The data set consists of wide angle reflected/refracted data recorded by four OBSs along a 35km-long 2-D profile (Figure 1.a). Data show mainly the direct water wave, the reflection from the sediment-basement interface, the refraction in the sediments and the refraction in the basement recorded as first arrivals. These travel times are inverted to obtain a velocity model.

b. Parametrization of the velocity model:

The travel time curves exhibit smooth relief. Thus, coarse parametrization should allow a good fit between observed and computed travel times.

The geological model consists of a water layer, a sedimentary layer and the basement. The sedimentary layer is parametrized as a homogeneous layer. Basement is parametrized as one layer with horizontal and vertical gradients. The horizontal distance between two adjacent velocity nodes on top and on bottom of the layer is 2 km.

The starting model is a flat layer model. The number of perturbational parameters (i.e., parameters which will be updated after inversion) is 58.

c. Ray tracing:

The phases used are the head wave propagating along the water-sediment interface, the wide-angle reflection from the top of the sediment-basement interface and the refraction within the basement (Figure 2). To insure good ray coverage at short offsets for the refraction into the basement, the head wave at the sediment-basement interface and turning rays within the basement are used simultaneously. If both rays reaches a given offset the turning ray provides the travel time for inversion. If the tuning ray fails (i.e., shadow zone), the head wave provides the computed travel time for inversion. Head waves are traced from the critical distance to a maximum offset (using parameters *hws* and *nhray* in namelist *TRAPAR* of **r.in**). This maximum offset is an estimation of the offset until which travel times of a head wave are a good approximation to those of turning rays.

d. Inversion:

Three iterations were used to converge towards the final model (Figure 2). Over the iterations, smoothing was progressively decreased while damping was increasing (Table 2).

Input parameters for inversion over the iterations:

Tunc (s)	*bndunc (km)*	*velunc (km/s)*	*Tauxv (km)*	*Tauxd (km)*	*Tauzv (km)*	*Tauzd (km)*
0.04	0.2	0.1	10	5	6	3

Table 1: Input parameters for inversion. These parameters were kept constant over iterations. *Tunc*: uncertainty assigned to the observed travel times; *bndunc*: a priori uncertainty assigned to the depth parameters; *velunc*: a priori uncertainty assigned to the velocity parameters; *Tauxv*: horizontal decay parameter for the velocity nodes; *Tauxd*: horizontal decay parameter for the depth nodes; *Tauzv*: vertical decay parameter for velocity nodes; *Tauzd*: vertical decay parameter for depth nodes.

Iteration	λ_v	λ_d	ε
1	400	200	0.03
2	200	100	0.05
3	100	50	0.1

Table 2: Regularization parameters used over iterations. Note increasing of damping and decreasing of smoothing with iterations.

Note that the correlation length is higher for velocities than for depth (Table 1). This is a geological motivated assumption which postulates that the lateral variation of velocities in the basement is smoother that the lateral depth variations of the sediment-basement interface. The same remark can be applied to the relative weight applied to depth and velocity nodes for damping *(bndunc, velunc)*. The higher value assigned to the depth nodes in comparison to the velocity nodes allows a higher magnitude of perturbation of the sediment-basement interface compared to basement velocities.

Only 2 iterations were required to reach a value of $\chi 2$ lower than 1.0 (Figure 2, Table 2). No editing of the velocity model is required after each iteration.

Iteration	NTobs	RMS	χ^2
0	2215	0.950	564.381
1	2192	0.103	6.893
2	2184	0.030	0.678
3	2162	0.024	0.459

Table 3: Results of inversion. NTobs is the number of observed travel times used during inversion. RMS is the root-mean square traveltime residual and $\chi 2$ is the chi-square value.

e. To run this example:

setup1 *(set the input files)*
xrayinv.cgn
ttinver.cg < par_it1.in (iteration n1)
cp vcoar.it1.in v.in *(copy the updated velocity model returned by ttinver.cg in v.in)*
xrayinv.cgn
ttinver.cg < par_it2.in (iteration n2)
cp vcoar.it2.in v.in *(copy the updated velocity model returned by ttinver.cg in v.in)*

xrayinv.cgn
ttinver.cg < par_it3.in (iteration n3)
cp vcoar.it3.in v.in *(copy the updated velocity model returned by ttinver.cg in v.in)*
xrayinv.cgn

Example 2: Line 200 - TICOSECT (Leg 2). Fine parametrization.

a. The data:

The data set consists of wide angle reflected/refracted data recorded by 5 OBSs along a 70km-long 2-D profile (Figure 1.b). Data show mainly the direct water wave, the refraction in the sediments and the refraction in the basement recorded as first arrivals to 65 km of offset.

b. Parametrization of the velocity model:

Another kind of parametrization is used compared to that usually used with the *xrayinvr* program. The geological model consists of a water layer, a sedimentary layer and a basement, the top of which is delineated by a velocity discontinuity. The sedimentary layer is parametrized as a vertically homogeneous layer. The internal structure of the basement is parametrized as a series of thin layers (i.e., 1 km-thick) delineated by second-order discontinuities. This mimics a grid of velocity nodes inside the basement. Thicknesses of the second-order discontinuity (s.o.d) layers are kept fixed during an inversion and thus, only velocities are involved during an inversion. The depth nodes which delineate the sediment-basement interface are involved during inversion. As the thickness of the s.o.d layers is kept fixed into the grid, the rectangular grid is going to deform after inversion to follow the topography of the sediment-basement interface (Figure 3). This is important because it avoids artificial change of gradients in the shallow structure (which would be the case if the position of the grid was fixed but not the sediment-basement interface) and thus favors good ray coverage just below the sediment-basement interface. The number of perturbational parameters is 1104.

c. Ray tracing:

The phases used are the head wave along the water-sediment interface, the head-wave along the sediment-basement interface over 15 km of distance, and the turning waves in the basement. Note that turning rays in each layer of the grid must be set in the array *ray* of input file r.in as the basement is parametrized with a series of thin layers (without velocity discontinuity). The ray path distribution is shown on figure 3.b. In this example, a uniform grid spacing in the basement is used for simplicity and to test the inversion scheme in the case of non uniform ray coverage and a high number of perturbational parameters (compared to that used with ***xrayinvr***). Inversion can be optimized using a variable grid spacing [*Wang*, 1993]. The ray path distribution diagram is used to illuminate the parts of the model which are well constrained by ray tracing. In these parts, a finer parametrization can be used. On the contrary, a coarser parametrization should be used in the parts of the model where ray path distribution is poor.

d. Inversion:

Four iterations were used to converge towards the final model. After the third iteration, a value of χ^2 less than 1.0 is reached. The iterations were stopped when the ray tracing was not able to trace ray to all the observed travel times. Like example 1, smoothing was progressively decreased while damping was increased for successive iterations (Table 4)

and smoothing and damping were higher for velocity nodes that for depth nodes. Correlation length is also higher for velocity than for depth (Table 3).

Tunc (s)	*bndunc (km)*	*velunc (km/s)*	*Tauxv (km)*	*Tauxd (km)*	*Tauzv (km)*	*Tauzd (km)*
0.04	0.2	0.1	10	5	6	3

Table 4: Input parameters for inversion. These parameters were kept constant over iterations. *Tunc:* uncertainty assigned to the observed travel times; *bndunc*: a priori uncertainty assigned to the depth parameters; *velunc:* a priori uncertainty assigned to the velocity parameters; *Tauxv:* horizontal decay parameter for the velocity nodes; *Tauxd:* horizontal decay parameter for the depth nodes; *Tauzv*: vertical decay parameter for velocity nodes; *Tauzd:* vertical decay parameter for depth nodes.

Iteration	λ_v	λ_d	ε
1	400	300	0.03
2	200	150	0.06
3	200	100	0.10
4	200	100	0.15

Table 5: Input parameters used for regularizations with respect to the iteration number.

Iteration	NTobs	RMS	χ^2
0	4403	0.893	491.849
1	4450	0.078	3.664
2	4416	0.047	1.356
3	4430	0.036	0.816
4	4420	0.032	0.668

Table 6: Results of inversion

To run this example:
setup2 *(set the input files)*
xrayinv.cgn
ttinver.cg < par_it1.in (iteration n1)
cp vit1.in v.in *(copy the updated velocity model returned by ttinver.cg in v.in)*
xrayinv.cgn
ttinver.cg < par_it2.in (iteration n2)
cp vit2.in v.in *(copy the updated velocity model returned by ttinver.cg in v.in)*
xrayinv.cgn
ttinver.cg < par_it3.in (iteration n3)
cp vit3.in v.in *(copy the updated velocity model returned by ttinver.cg in v.in)*
xrayinv.cgn
ttinver.cg < par_it4.in (iteration n4)
cp vit4.in v.in *(copy the final updated velocity model returned by ttinver.cg in v.in)*
xrayinv.cgn

4.2. Simultaneous wide-angle reflected/refracted and normal incidence reflected ray tracing and travel time inversion

General procedure

The normal incidence rays are traced while wide angle ray tracing related to each OBS is performed. Input parameters of the program ***ttinver.cg*** are the same as when wide angle ray tracing is used only. Partial derivatives of normal incidence travel times are computed with respect to depth and velocity parameters (for velocity parameters located above the reflector from top of which the waves reflect at normal incidence) or with respect to depth parameters only (see parameter *ivvn* in namelist *INVPAR*). It is generally advised to involve depth parameters only for normal incidence travel time inversion since reflection data does not allow much control on velocity [*Wang*, 1993]. The simultaneous inversion of normal incidence reflection and wide angle reflection/refraction and travel times allows an optimized control for depth and velocity.

Example 3: Synthetic example: pinch-out and fault model

To illustrate the use of simultaneous ray tracing and inversion for normal incidence reflection and refraction data, a synthetic example is shown.

a. Velocity model

The model is a 4 layer model characterized by a pinch-out and a fault (Figure 4). The initial model is a flat layer model except for the pinch-out layer. It is assumed that the x coordinate of the pinch-out in the starting model is correct (reflection data would allow the user to locate with a reasonable accuracy the position of the pinch-out but not the shape and the velocities in the pinch-out layer) [*Wang*, 1993].

b. Ray tracing

The turning rays in each layer and the normal incidence rays from the top of each layer are computed (Figure 5).

c. Inversion

A first set of inversions were performed using simultaneously refraction and normal incidence reflection travel times (Table 7). The maximum error for depth nodes is 30 meters. Error in velocity estimation varies between 0.00 and 0.51 km/s. This latter high value is related to the estimation of vertical gradients which are poorly constrained by travel time analysis alone. A second set of inversions were performed using refraction travel times only, geometry of interfaces were not recovered (i.e., the fault). Error in velocity estimation is higher to balance the error in depth estimation and varies between 0.00 and 0.90km/s.

Refraction travel times				Normal incidence travel times		
Iteration	NTobs	RMS	χ^2	NTobs	RMS	χ^2
0	1271	0.230	531.332	565	0.171	1196.438
1	1230	0.028	7.874	567	0.112	17.038
2	1437	0.034	11.844	565	0.027	30.157
3	1460	0.004	0.178	565	0.025	0.461
4	1567	0.005	0.223	565	0.007	0.619

Table 7: Results of inversion using normal incidence reflection travel times and refracted travel times.

Iteration	NTobs	RMS	χ^2
0	1271	0.230	531.332
1	1614	0.036	13.169
2	1615	0.008	0.683
3	1610	0.003	0.067
4	1585	0.003	0.084

Table 8: Results of inversion using refraction travel times only. Same regularizations were used that of table 7. Good fit between observed and computed travel times is reached but the errors in the final velocity model are high (the fault was not recovered).

e. To run the example

In the case of simultaneous normal and wide angle incidence ray tracing:

```
setup3_n
xayinv.cgn
ttinver.cg < par_it1.in
cp vit1.in v.in
xrayinv.cgn
....
```

In the case of wide angle incidence ray tracing only:

```
setup3_w
xrayinv.cgnttinve.cg < par_it1.in
cp vit1_w.in v.in
xrayinv.cgn
....
```

4.3. Simultaneous wide-angle reflected/refracted and normal incidence reflected travel time inversion from 2-D intersecting profiles

General procedure

After having generated the starting intersecting 2-D profiles, the user has to indicate which are the common parameters (i.e., the parameters which are at the intersection of two profiles and whose perturbations will be identical after inversion). Common parameters belonging to two intersecting profiles may or may not be identical. If the structure is isotropic, the common parameters should be identical in the starting models. If the structure is azimuthally anisotropic and if the user has an estimation of the anisotropy, common parameters should differ according to the anisotropy. After inversion, difference between common parameters will be the same as that before inversion.

If part of the model is anisotropic and if the the value of anisotropy is unknown, the model parameters located at the intersection of two profiles, if belonging the anisotropic layers, can be processed as standard parameters (in contrast to common parameters) to estimate the azimuthal anisotropy. On the contrary, the model parameters belonging the isotropic layers can be processed as common parameters, if located at the intersection of several profiles.

To set the common parameters, a file named *d3d.in* must be created. Each line of this file has the following format:

ncross (index_par(j), j=1,n_intersection) (index_profile(j), j=1,n_intersection)

ncross is the number of profiles sharing one common parameter. This number is generally equal to 2 but may be greater than 2 (if several profiles intersect at the same location).

One index (index_profile) is associated (subjectively) to each 2-D profile by the user (index are from 1 to n_profile). For each profile, each model parameter of the current profile has an index (index_par) (from 1 to n perturbational parameter). This index can be read using program *numpar (see appendix 2)*. Given the velocity model, this program returns the same velocity model but with the index associated to each perturbational parameter (index is indicated below the value of the parameter instead of 1 in the original file). The user indicates in file d3d.in the indexes of the common parameters and the associated profile indexes. A line of file d3d.in is used per intersection, per type of parameter (depth or velocity) and per depth.

Example for two intersecting profiles.

Model n1 (model whose index is 1):

```
1  20.00
0  00.00
        0
1  20.00
0  01.50
        0
1  20.00
0  00.00
        0
2  00.00  05.00  10.00  15.00  20.00
0  02.00  02.00  02.00  02.00  02.00
        1     2     3     4     5
2  00.00  05.00  10.00  15.00  20.00
0  03.00  03.00  03.00  03.00  03.00
        6     7     8     9    10
2  00.00  05.00  10.00  15.00  20.00
0  04.00  04.00  04.00  04.00  04.00
       11    12    13    14    15
3  20.00
    05.00
```

Model n2 (model whose index is 2):

```
1  20.00
0  00.00
        0
1  20.00
0  01.50
        0
1  20.00
0  00.00
        0
2  00.00  05.00  10.00
0  01.70  02.00  02.20
        1     2     3
```

```
2  00.00  05.00  10.00
0  03.30  03.00  03.40
       4     5     6
2  00.00  05.00  10.00
0  04.20  04.00  04.60
       7     8     9
3  20.00
   05.00
```

The two profiles intersect at x=10 km and 5 km for profiles 1 and 2 respectively. The parameters of layer 2 are involved during inversion.

The two models shown above are returned by program **numpar**. The indexes of the model parameters are shown on lines 12, 15 and 18 of the models. There are 18 and 9 model parameters for profiles 1 and 2 respectively.

The file d3d.in takes the form:

```
2  3     2     1     2
2  8     5     1     2
2  13    8     1     2
```

The first column indicates the number of profiles sharing common parameters at a given depth and for a type of model parameters (depth, top and bottom velocities). There are two profiles so two intersections. The second and third columns are the indexes of common parameters for the two intersecting profiles whose indexes are indicated on columns 4 and 5 respectively (these parameters are located at x=10 km and 5 km for profiles 1 and 2 respectively). The model parameter indexes in column 2 are associated with the model index of column 4 (profile n1) and the model parameter indexes in column 3 are associated with the model index of column 5 (profile n2).

When file *d3d.in* is generated, program ***column_indx1*** *(see appendix 2)* must be run. This program returns output files *parindx0.out* and *parindx1.out* used as input by ***xrayinv3dcgn*** and ***ttinver.3dcg***.

Next, 2-D ray tracing must be performed for each 2-D profiles separately using program ***xrayinv3dcgn*** starting with profile with index 1 and finishing with profile with index n_profile. After, all 2-D ray tracing are performed, inversion of travel times picked along each 2-D profile can be performed simultaneously by running program ***ttinver.3dcg***. This program returns the *n_profile* updated velocity models. Before computing matrix inversion, this program merges the partial derivative matrix associated with each 2-D profiles into one unique matrix taking into account the common parameters (travel time derivatives with respect to these parameters are located in the same column of the matrix). Input parameters (for regularizations) of program ***ttinver.3dcg*** are identical that of ***ttinver.cg*** except that the regularization parameters (λ and ε) must be provided for each profile.

After, the inversion is completed, the process (ray tracing and inversion) can be iterated several times. Note that before running a series of ray tracing, the output files returned by ***xrayinv3dcgn*** from the previous iterations must be removed (all the *.out files except *parindx.out* and *parindx1.out*).

In summary the different steps can be summarized as follow:

1. Generate the starting models
2. Run program ***numpar*** for each 2-D model
3. Generate file *d3d.in*
4. Run program ***column_indx1***
5. Run ***xrayinv3dcgn*** for each 2-D profiles (ray tracing)
6. Run ***ttinver3dcg*** (inversion). This programs returns each updated velocity model.

7. Remove all the *.out files except *parindx.out* and *parindx1.out*.
8. Go back to step 5 and iterate the process until final models are reached.

Note that the final models may not be reached at the same iterations for different 2-D models. If this occurs, the model for which the final iteration was reached can be removed from the process but the common parameters associated with this profile must be kept constant during the further iterations in the intersecting profiles.

Example 4: Synthetic example: convergent margin model.

To illustrate the use of simultaneous ray tracing and inversion for normal incidence reflection and refraction data, a synthetic example is shown. The purpose of this example is not to prove the validity of the method but to illustrate the practical steps required to apply it.

The structure of a convergent margin is mimicked. One profile is perpendicular to the margin and three intersecting profiles are parallel to the margin. One intersecting profile is on the oceanic domain, the second one is close to the trench and the third one is along the wedge landward. The assumption of one-dimensionality parallel to the margin is made. For simplicity, the starting models are relatively closed to the final models.

a. Parametrization of the velocity model:

The profile perpendicular to the margin is 45 km long (Profile 1). Distance between nodes is 5 km. It consists of 6 layers: the water layer (layer 1), the wedge sedimentary layer (layer 2), the wedge layer (layer 3), the oceanic sedimentary layer (layer 4), the oceanic crust layer (layer 5) and the upper mantle layer (layer 6). The wedge sedimentary and the wedge layers pinch out at the trench. Profiles 2, 3 and 4 are one-dimensional, and crosses profile 1 at x=5 km, 25 km and 40km respectively. Number of perturbational parameters is 66 for profile 1, 5 for profile 2, 9 for profiles 3 and 4. Note that the subducting sedimentary layer is modeled and thus constitutes a low velocity layer intercalated between the base of the wedge and the subducting oceanic crust.

b. Ray tracing (Figure 6):

The phases used are: (1) the refractions within the sedimentary layers on top of the oceanic crust and on top of the wedge (2) refraction within the oceanic crust (3) refraction within the wedge (4) wide angle reflection from Moho (5) head wave along Moho (6) normal incidence reflection at the sediment-oceanic crust interface (7) normal incidence reflection at the sediment-wedge interface (8) normal incidence reflection at the bottom wedge-oceanic sediment interface.

c. Inversion:

Nine iterations were conducted to derive the final model. Strategy used for selecting input parameters for inversion is similar to the one described in previous examples (Figure 7). No editing of the model was required during iterations (no crossing between interfaces occurs despite the potential ambiguity related to an existing low velocity zone in the deep structure). The maximum error for the depth nodes is 50 meters except for the Moho for which an errors of 130 meters and 950 meters are recorded on the two extreme sides of the model (rays do not propagate in this part of the model). The maximum error on velocity in the wedge is 0.05 km/s (without taking into account the velocity on the side). Error is larger in the oceanic crust where the error reaches 0.17 km/s for the velocities on top and 0.32 km/s for the velocity on bottom. This latter error could be avoided by fixing the vertical gradient with a reasonable value during inversion (vertical gradient are known to be poorly constrained by travel time analysis).

The number of observed travel times used by each inversion and the *RMS* residual are plotted with respect to the iteration number for each profile on figure 8.

Tunc (s)	*bndunc (km)*	*velunc (km/s)*	*Tauxv (km)*	*Tauxd (km)*	*Tauzv (km)*	*Tauzd (km)*
0.02	0.2	0.2	20	10	10	5

Table 4: Input parameters for inversion. These parameters were kept constant over iterations. *Tunc*: uncertainty assigned to the observed travel times; *bndunc:* a priori uncertainty assigned to the depth parameters; *velunc:* a priori uncertainty assigned to the velocity parameters; *Tauxv:* horizontal decay parameter for the velocity nodes; *Tauxd:* horizontal decay parameter for the depth nodes; *Tauzv:* vertical decay parameter for velocity nodes; *Tauzd:* vertical decay parameter for depth nodes.

To run this example:

Iteration n1

script.rm *(remove output files)*
setup4.0 *(set inputs for line 1)*
xrayinv3dcgn *(ray tracing)*
setup4.1 *(set inuts for line 2)*
xrayinv3dcgn *(ray tracing)*
setup4.2 *(set inputs for line 3)*
xrayinv3dcgn *(ray tracing)*
setup4.3 *(set inputs for line 4)*
xrayinv3dcgn *(ray tracing)*
ttinver.3dcg < par_it1.in *(it 1)*

Iteration n2

script.rm
setup4.0
cp vit1.0.in v.in
xrayinv3dcgn
setup4.1
cp vit1.1.in v.in
xrayinv3dcgn
setup4.2
cp vit1.2.in v.in
xrayinv3dcgn
setup4.3
cp vit1.3.in v.in
xrayinv3dcgn
ttinver.3dcg < par_it2.in

Iteration n3

script.rm
setup4.0
cp vit2.0.in v.in
xrayinv3dcgn
setup4.1
cp vit2.1.in v.in
xrayinv3dcgn
setup4.2
cp vit2.2.in v.in
xrayinv3dcgn
setup4.3
cp vit2.3.in v.in
xrayinv3dcgn
ttinver.3dcg < par_it3.in

and so on until iteration 9.

REFERENCES

Christeson, G., OBSTOOL: Software for Processing UTIG OBS Data, UTIG, Austin, 1995.

Press, W., H., S. Teukolsky, A., W. Vetterling, T. and B. Flannery, P., *Numerical Recipes in FORTRAN,* Cambridge University Press, Cambridge, 1986.

Takanami, T. and G. Kitagawa, A new efficient procedure for the estimation of onset times of seismic waves, *J. Phys. Earth, 36, 267-290*, 1988.

Thurber, C. H., Earthquake locations and three-dimensional crustal structure in the Coyote Lake area, central California, *Journal of Geophysical Research, 88, 8226-8236*, 1983.

Toomey, D., R., S. C. Solomon and G. M. Purdy, Tomographic imaging of the shallow crustal structure of the East Pacific Rise at 9deg30'N, *Journal of Geophysical Research, 99, 24135-24157*, 1994.

Wang, B., Improvement of seismic travel-time inversion methods and application to observed data, West Lafayette, Indiana, Purdue University, 1993.

Wessel Paul and Walter H. F. Smith, Free software helps map and display data, *EOS Trans. AGU, 72, 441,445-446*, 1991.

Zelt, C. A., 3-D velocity structure from simultaneous traveltime inversion of in-line seismic data along intersecting profiles, *Geophysical Journal International, 118, 795-801*, 1994.

Zelt, C. A. and R. B. Smith, Seismic traveltime inversion for 2-D crustal velocity structure, *Geophysical Journal International, 108, 16-34*, 1992.

Appendix 1: List of content of RSTTI package

Programs are located in directory ***RSTTI*** on ***/utig/usr4/operto***

RSTTI contains 10 subdirectories:

- ***RAYINV.CGN***

Contains the main program **xrayinv.cgn.f** and associated subroutines, two include files (raycg.par and raycg.com) and a makefile.

To compile, set in the makefile the directory where the executable should be copied and run the makefile.

Parameters which define the physical dimension of the arrays used by the program are set in the *raycg.par* file. Check that the dimension assigned to each array is sufficient for the model and ray path you are using. The meaning of each parameter defining the dimension of the arrays is described in the include file *(raycg.par).*

The include file *raycg.com* stores the common block.

- ***TTINVER.CG***

Contains the program ttinver.cg.f and raydens.f. To compile it, just use:

f77 -O4 -o ttinver.cg ttinver.cg.f and *f77 -O4 -o raydens raydens.f.*

Include files: tticg.par and sparcg.com. Check the physical dimension assigned to the arrays before compiling.

- ***RAYINVR.3DCGN***

Contains main program and associated subroutines for ray tracing in the case of a grid of 2-D lines. Program is compiled using a similar Makefile to that of **xrayinv.cgn**.

Include files: raycg3d.par and *raycg3d.com.*

- ***TTINVER.3DCG***

This directory contains the program **ttinver.3dcg.f**. To compile it, just use:

f77 -O4 -o ttinver.3dcg ttinver.3dcg.f.

Include files: tticg.par and *sparcg.com.*

- ***MISC***

This directory contains the programs for pre-processing. See Appendix 2 for description of these programs. To compile use the f77 command.

- ***EXAMPLES***

This directory contains the files to run the examples described in this report.

- ***DOCUMENTATION***

This directory contains the present documentation.

- ***AUTOPICK***

This directory contains programs for automatic picking of travel times and estimation of their relative uncertainties.

- **BIN**

This directory contains the compiled programs.

- **PLTLIB**

The directory contains the graphic library (same as that of ***XRAYINVR***).

Appendix 2: List of programs for pre-processing

• ***XDEPTH_OBS***

Returns a velocity model with the bathymetry and the x and z coordinates of the OBSs along the model.

IO:

Input files:

- shotfile (format is that used by ***OBSTOOL*** software, [*Christeson*, 1995])
- *obsloc.dat* file which stores the location and depth of the OBSs with the following format[*Christeson*, 1995]:

OBS drop number, Latitude (decimal degrees), Longitude (decimal degress), Water depth (m), Water velocity (m/s), Azimuth of the horizontal components (decimal degrees), secondary clock correction (s).

Input parameters:

- Output fine name for the velocity model
- Output file name for the file storing the (x,z) coordinates of the OBSs (Cartesian coordinates are chosen relative to upper left corner of the velocity model)
- Distance (m) between depth nodes for bathymetry in the velocity model.
- Water wave velocity (m/s) (used to set the water layer in the velocity model)
- Set if the first shot of the shot file is located at the origin of the velocity model (1) or at the end (2)
- Deviation (m) of the depth of the OBS with respect to the depth of the interface (source must not be set exactly on the interface delineating the sea bottom. A negative deviation move the OBS upward. -5 meters is a good pragmatical value.

Output files:

- Starting velocity model. Only the water layer and the sea bottom interface is set in the model.
- File with the (x,z) coordinates of each OBS with respect to the upper left corner of the velocity model.

NB: this program assumes that the shots recorded in the shotfile define the size and location of the velocity model to be set. If fewer shots are needed to define the velocity model, a smaller shot file must be created first.

Compilation: *f77 -o xdepth_obs xdepth_obs.f*

• ***OBST2RAY***

Produce a tx.in type file based on a picks file with a similat format as that of ***OBSTOOL***.

IO:

Input file:

- picks file with format similar as that of ***OBSTOOL***.

Input parameters:

- x coordinate of the OBS with respect to the velocity model
- set if negative offsets stored in the SEG-Y headers of the OBS data file correspond to the shots located on the left (1) or on the right (-1) side of the OBS with respect to the velocity model.

Output file:

- Observed travel time file (with format of tx.in) for the current OBS.

NB: the observed travel time files for each of the OBS along the profile will have to be merged together to form the tx.in file.

Compilation: *f77 -o obst2ray obst2ray.f*

• ***RAY2OBST***

Given a travel time file used by ***RAYINVR*** (*tx.in* or *tx.out*), return a pick file for each OBS whose travel times are stored in the input file with a format compatible with that of ***OBSTOOL***.

IO:

Input file: Travel time file used by ***RAYINVR*** (tx.in or tx.out)

Output files: One picks file associated with each OBS named ttfilexx where xx=1,...,nobs (nobs is the number of OBSs along the profile).

Compilation: *f77 -o ray2obst ray2obst.f*

• ***MERGE_TX.IN***

Merge together several observed travel time files (i.e., those returned by obst2ray.f).

IO:

Input files: observed travel time files related to each OBS.

Input parameters:

- Number of files to be merged
- Name of each travel time files
- Output file name

Output file: Merged travel time file (with format of tx.in).

Compilation: *f77 -o merge_tx.in merge_tx.in.f*

• ***RAYN2OBST***

Given a normal incidence travel time file (i.e., *txn.in* or *txn.out*), returns a pick file with a format compatible with that of ***OBSTOOL***.

IO:

Input file: Normal incidence travel time file with format of *txn.in*.

Output files: Normal incidence travel time file with format of ***OBSTOOL***.

Compilation: *f77 -o rayn2obst rayn2obst.f*

• ***FLATLAYER***

Generates a starting velocity model made of laterally homogeneous layers.

IO:

Input parameters:

- **name_out:** name of the output file storing the velocity model
- **xmax (km):** length of the model
- **dx (km):** horizontal distance between adjacent nodes in the layers (km)
- **ilayer0:** layer index of the first layer to be set (may be greater than one if only part of the model must be written).

- **nlayer:** number of layers to be written

Output file:

- Velocity model with format of v.in.

Compilation: f77 -o flatlayer flatlayer.f

• ***CTLAYER***

Generates a starting velocity model which merges an input velocity model (generally a velocity model for which only the water column and the sea floor bathymetry were set) with a model constituted by a first layer (called layer1) with below a series of layers delineated by second-order discontinuities (i.e., grid of velocity nodes). A first-order discontinuity delineates the layer1 from the top of the grid. This type of file was used in Example 2.

IO:

Input file:

Velocity model to be merged with the new velocity file. The input file must end with a series of depth nodes (generally the sea floor bathymetry).

Input parameters:

- Name of the input velocity file.
- **xmax, zmax (km):** Length and maximum depth of the model.
- **ilayer0:** Index of layer 1 (i.e., of the last layer whose topography is set in the input file)
- **dxv, vtop, vbot:** distance between nodes, velocity on top and velocity on bottom in the layer 1.
- **xthick1 (km):** thickness of the layer1.
- **dxz(km):** distance between depth nodes for the interface between layer 1 and the grid.
- **dx(km):** distance between velocity nodes in the grid.
- **nlayer:** number of layers in the grid
- **xthick (km):** thickness of the layers in the grid
- **v1 (km/s):** velocity on top of the grid
- **grad (s^{-1}):** vertical gradient in each layer of the grid.

Compilation: f77 -o ctlayer ctlayer.f

• ***COLUMN INDX1***

Generates the input files required to run xrayinv3dcgn and ttinver.3dcg.

IO:

Input file:

- **d3d.in:** this files stores the indexes of the common parameters between intersecting profiles. The format of this file is as follow:

ncross, indexes for each intersecting profile of the common parameter, indexes of the intersecting profiles (*ncross* indexes must be indicated both for parameters and profiles).

ncross is the number of profiles which share the current common parameter. The index of the common parameter for each of the intersecting profile can be read in the velocity models returned by the program ***numpar***.

Output files:

- **parindx0.out:** This file is used as input by ttinver.3dcg.
- **parindx1.out:** This file is used as input by ttinver.3dcg.

Compilation: f77-o column_indx1 column_indx1.f

• ***NUMPAR***

Given a velocity model, return the same velocity model but provides the index of each perturbational parameter (The indexes vary from 1 to nmax from top to bottom of the structure and from the distance 0 to xmax). This is useful for generating the **d3d.in** input file required to run program column_indx1.

IO:

Input file: velocity model

Output file: velocity model with the index of each perturbational parameter indicated below the value of the parameter.

Compilation: f77 -o numpar numpar.f

Appendix 3: Automatic picking of travel times and estimation of their relative uncertainties.

A3.1. Method

The method for automatic picking is described in *Takanami and Kitagawa* (1988) and was applied in the case of a 3-D tomographic study [*Toomey, et al.*, 1994]. This method is based on autoregressive modeling of the observed seismic signals. The travel time is searched by optimizing the fit of two autoregressive models to the seismic data. Minimization of the AIC criterion (Akaike Information Criterion) is performed using a Householder transformation. It is assumed that the seismic trace constitutes a time series {x1, ..., xn} and that the arrival time is within the interval [n0,n1]. Practically, the user has to define the first and last samples of the trace which delineate the time series (x1 and xn) and the two boundaries n0 and n1 which delineates the data window used to evaluate the autoregressive models.

A3.2. List of programs

• **Autopick:** automatic picking of a single trace

IO:

Input files:

- Trace file. The samples of the trace is stored in a direct access file. The first sample of the trace represents x1 and the last sample xn.

Input parameters:

- Name of the trace file.
- **n0, n1:** sample indexes which delineate the data window used to evaluate the autoregressive models (i.e., data window within which arrival should be located).
- **k:** order of the autoregressive model
- **p:** resolution in number of sampling intervals
- **tori, dt:** time origin of the trace (s), sampling interval (s).

Output file:

- **faic:** stores the AIC function in direct access format.
- **ftime:** trace with samples equal to 1000 except at the location of the pick (value equal to zero). Formal of this file is direct access.

The value of the estimated travel time is printed on the screen.

Compilation: *f77 -o autopick autopick.f*

•**Formautop:**

Using pick file returned by ***OBSTOOL,*** generates file *fbound* used as input by ***window-segy.*** Before running this program, the user has to pick with OBSTOOL several limits corresponding to x1, n0, n1, xn. These limits must be defined for a chosen number of traces. The boundaries x1, n0, n1, xn are interpolated linearly for the traces located between two adjacent traces for which these boundaries were picked. The user can start picking all the boundaries x_1 along the profile, then move to the boundary n_0, then n_1 and x_n or can pick the four boundaries sequentially trace per trace from boundary x_1 to boundary x_n via n_0 and n_1. While picking, integer code 1 must be assigned to the picks associated with x_1, 2 to the picks associated with n_0, 3 to the picks associated with n_1 and 4 to the picks associated to x_n.

IO:

Input file: picks file returned by OBSTOOL

Input parameters:

- **iop:** format of the input file. User can pick first each 'node' of the first boundary then move to the second boundary and so on (option 2) or pick the 4 nodes of one trace then move to another trace and so on (option 1).

Output file: *fbound*

Compilation: f77 -o formautop formautop.f

• **Window-segy:**

Given a segy file and the file *fbound* returned from program ***formautop***, returns input files (*files ftrace and ftrace.par*) required to run program ***autopick_2D***. A new segy file is also generated with the windowed traces for visual check of the location of the fourth boundaries plotting the windowed data with ***OBSTOOL***.

IO:

Input file:

- SEG-Y file
- *fbound*

Input parameters:

-Name of the input Segy file
- Name of the windowed output segy file
- **vr:** Reduction velocity used to pick the four boundaries
- **idtype:** samples have IBM (1) or IEEE (6) format
- **nchan0:** channel to be stored

Output files:

- *ftrace*: windowed traces file used for picking (the trace is windowed between the two extreme boundaries x1 and xn); used as input by ***autopick_2D***.
- *ftrace.par*: parameter file (used as input by ***autopick_2D***)
- windowed segy file (used for visual check with ***OBSTOOL***)

Compilation: *f77 -o window-segy window-segy.f -lsegy.*
segy is from the library segylib used by ***OBSTOOL.***

•**Autopick_2D:** automatic picking from a series of traces and estimation of relative uncertainties between picks. Input files used by this program are output files returned by ***window-segy.***

IO:

Input file:

- *ftrace*
- *ftrace.par*

Input parameters:

- **k:** order of the autoregressive models *(10 is a good pragmatical value).*
- **p:** resolution interval in number of sampling interval *(2 is a good pragmatical value).*
- **vr (km/s):** reduction velocity used to write out travel times (0 if not required).
- **lseuil:** number of samples of the AIC curves which are stored in output file *faic.*

- **thres:** threshold used for uncertainty estimation. Travel times whose AIC value is between AICmin and AICmin+thres constitutes a potential traveltime.

Output files:

- ***fpick:*** travel time picks. format is the one used by ***OBSTOOL*** (*fttbin* is also generated in direct access format).
- ***faic:*** AIC curves stored in a direct access file. Each AIC curve has lseuil samples.
- ***func.ascii:*** travel time uncertainties (*func* is also generated in direct access format).

Compilation: *f77 -O4 -o autopick_2D autopick_2D.f*

• **Fil_smoothpick.f:** smooth the picks returned by autopick_2D.

IO:

Input file:

- raw picks file

Input parameters:

- ***pts:*** width of the filter in number of samples (see subroutine smooth.f in [*Press, et al.*, 1986]).
- ***xmin, xmax (m):*** range of offsets between which smoothing must be applied.
- ***xint, tol (m):*** theorical distance between two adjacent traces and tolerance. Smoothing is not applied if the distance between two adjacent traces is greater than *xint+tol* or lower than *xint-tol*.
- ***name_in, name_out:*** names of the input and output files

Output files:

- Smooth picks file.

Compilation: *f77 -o fil_smoothpick fil_smoothpick.f sp_smooft.f sp_four1.f sp_realft.f*

A3.2. Practical procedure

a. Case of a simple trace.

The seismic trace is stored in a direct access unformated file.
Fill up a parameter file with the required input parameters to run program ***autopick***.
Type **autopick** < *parfile*.
The program returns two files: *ftime* and *faic*.
ftime is a direct access file whose size is the same that of the trace file and whose samples equal to 1000 except at the location of the pick where the value equals to 0 (this file was generated for plotting purposes; figure A3.2). File *faic* stores the AIC curve in a direct access file (Figure A3.2).

b. Case of a 2-D profile stored in SEG-Y format.

Step 1: pick with ***OBSTOOL*** the four boundaries (x1, n0, n1, xn) required by the algorithm (Figure A3.1.a).

Step 2: Run program ***formautop***. This program returns file *fbound* used as input by program ***window-segy***.

Step 3: Run program ***window-segy***. This program will return files *ftrace* and *ftrace.par* used as input by ***autopick_2D***.

Step 4: Run program ***autopick_2D***. This program returns file *fpick* which stores the picks with a format suitable for ***OBSTOOL***. The picked travel times can be plotted superimposed on the data for visual check (Figure A3.1.b). The program returns also each AIC curve stored in a direct access file *(faic)*. In this file, each AIC curve has the same size for convenience (length of the curve is provided as input by the user) although each trace (between x1 and xn) may not have the same length. The program also returns the file *func* which stores the relative pick uncertainties between picks.

Step 5: Smooth the travel time curve using ***fil_smoothpick*** (Figure A3.1.c).

A3.3. Example

The example illustrated on figures A3.1 and A3.2 is stored in the directory ***/RSTTI/AUTOPICK/Example***.

The pick file returned by ***OBSTOOL*** is called **pickauto36** (provided for information only).

To run the example:

- autopick_2D < autop.in
- autopick < autop_trc1.in
- autopick < autop_trc2.in
- autopick < autop_trc3.in

The SEG-Y file is obs36.l229. The file contains 385 traces. The CPU time on a SPARC 20 required to process this number of traces given the selected value of the input parameters (k=10, p=2) is 11.5 mn.

For picking the raw data are used (no band-pass filtering was applied before picking). Files *trc1*, *trc2* and *trc3* are the direct access files which stores the three traces plotted on figure A3.2. The order of the autoregressive models is 10 and the resolution interval is two sampling intervals. Note that the picking failed at short offsets between 3 and 3.5 km. This is due to the fact that the water wave is recorded in the windowed data as a second arrival and affects the picking of the first arrival. This effect might have been removed by using a narrower time window for this range of distances. Note also that the picking becomes bad in the case of poor signal to noise ratio (offsets between 15 and 22 km).

FIGURES

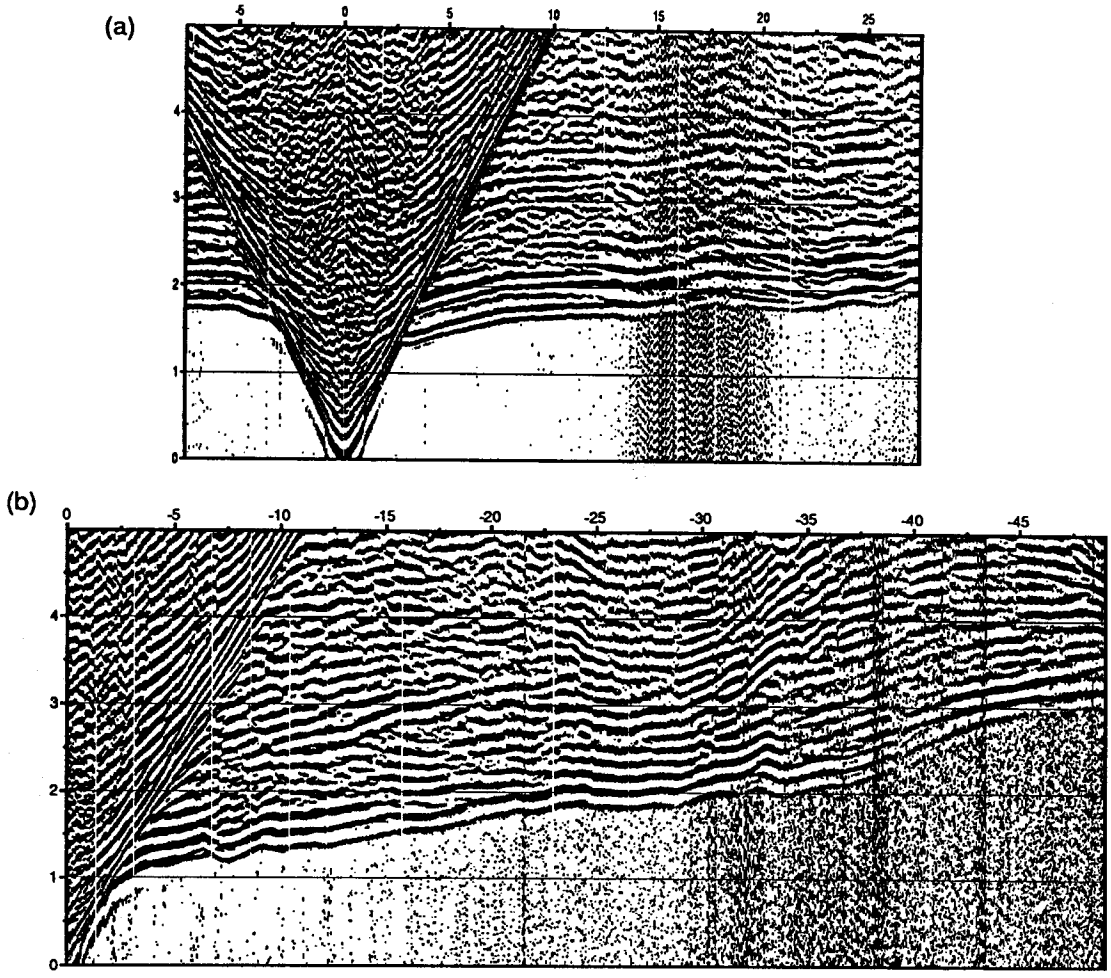

Figure 1: (a) Example of a section recorded along line 229 (OBS 36) (see Example 1 in the text); (b) Example of a section recorded along line 200 (OBS 73) (see Example 2 in the text).

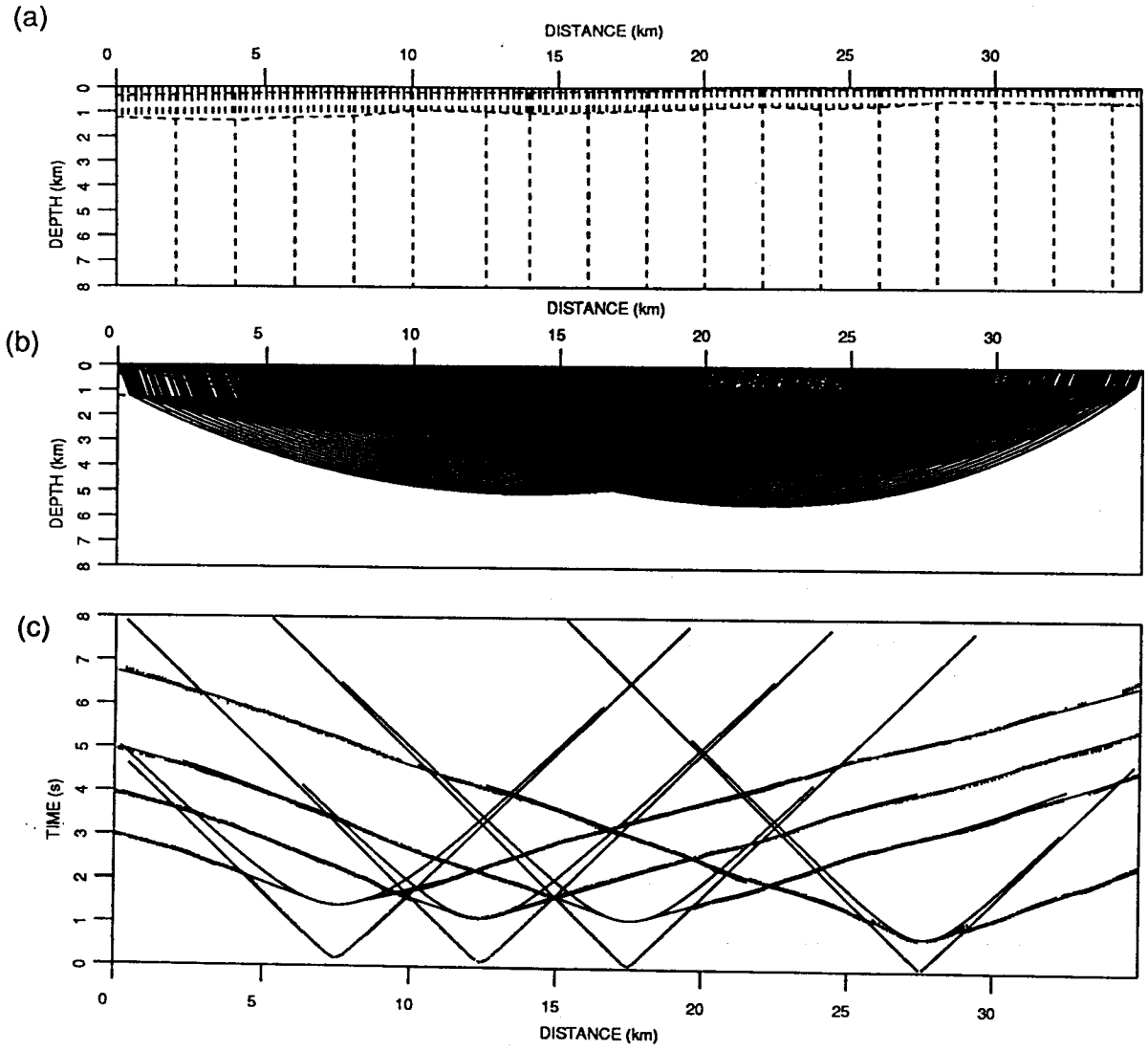

Figure 2: a. parametrization of the velocity model. Dot lines depict the trapezoidal blocks defined by the three grid of nodes (depth, top velocity, bottom velocity); (b) Ray tracing in the final model. (c) comparison between observed (vertical bars) and computed travel times (solid lines) from the final model.

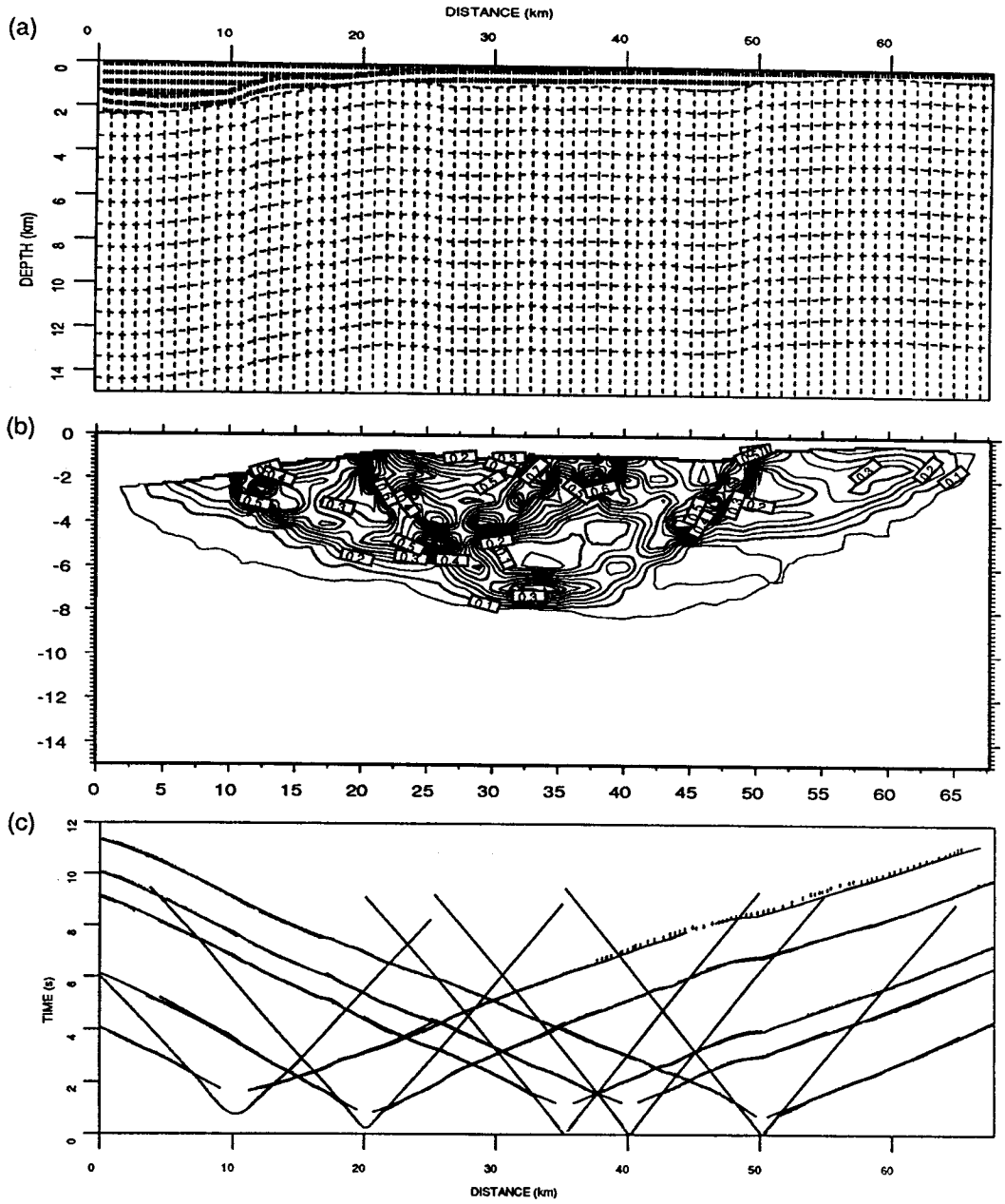

Figure 3: (a) 'Hybrid' parametrization of the model combining layered parametrization and gridded parametrization in the basement. The rectangular grid of velocity nodes in the basement deforms 'following' the topography of the sediment-basement interface. This improves ray coverage provided by turning rays in the shallow structures. (b) Ray path distribution from the final velocity model. This diagram could be used to define a variable grid spacing in the velocity model with respect to the ray path distribution. (c) Comparison between observed travel times (vertical bars) and computed travel times from the final model (solid lines).

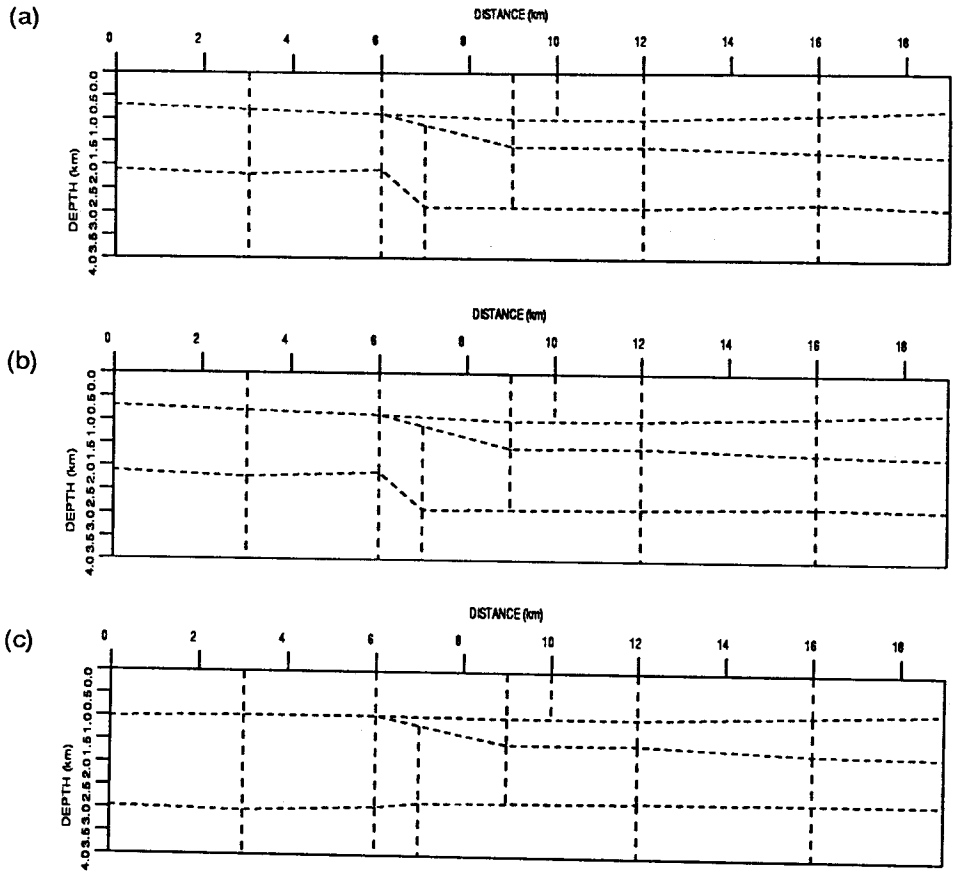

Figure 4: (a) True velocity model; (b) Final velocity model inferred from simultaneous inversion of normal incidence reflection and refraction traveltimes; (c) Final velocity model inferred from the inversion of refraction traveltimes.

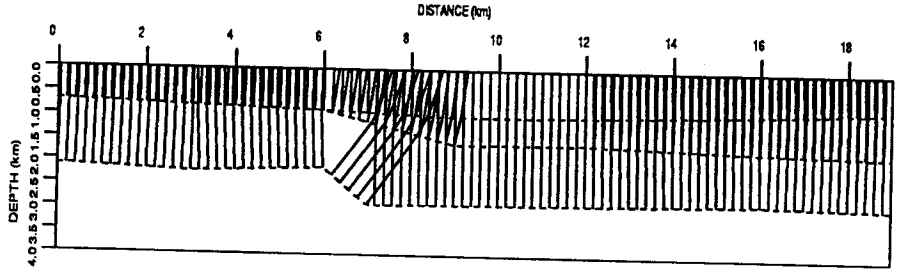

Figure 5: Normal incidence ray tracing from the final velocity model

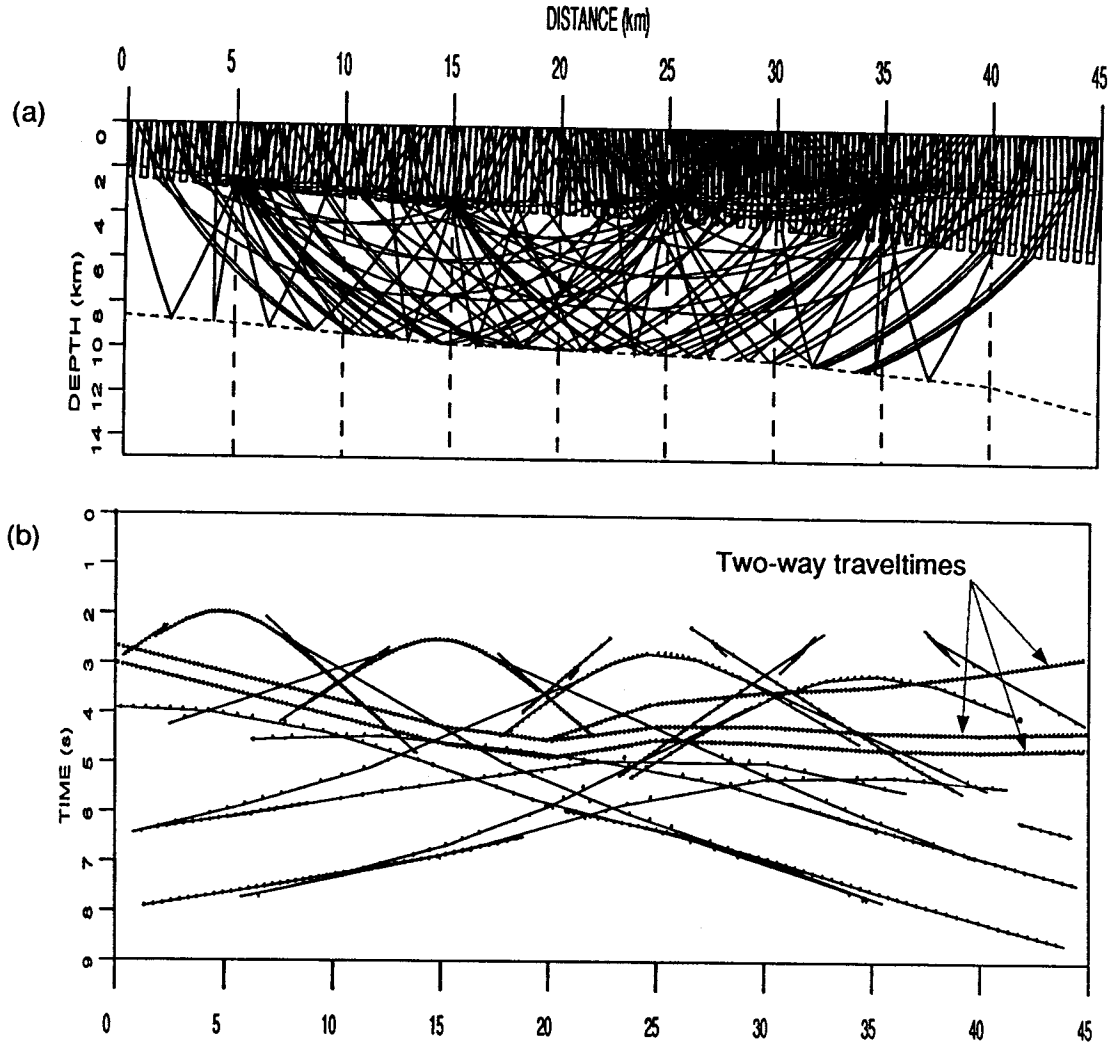

Figure 6: (a) Simultaneous ray tracing in the perpendicular profile. (b) Comparison between observed (symbols) and computed (solid lines) travel times.

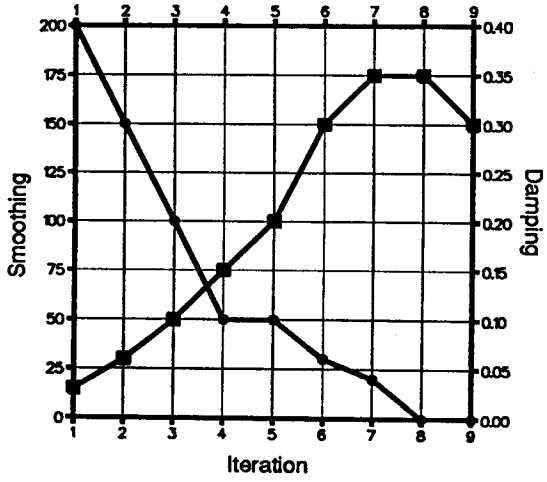

Figure 7: Factors applied to smoothing (circles) and damping (square) matrix over the iterations for Example 4. Note that damping increases with iterations while smoothing decreases.

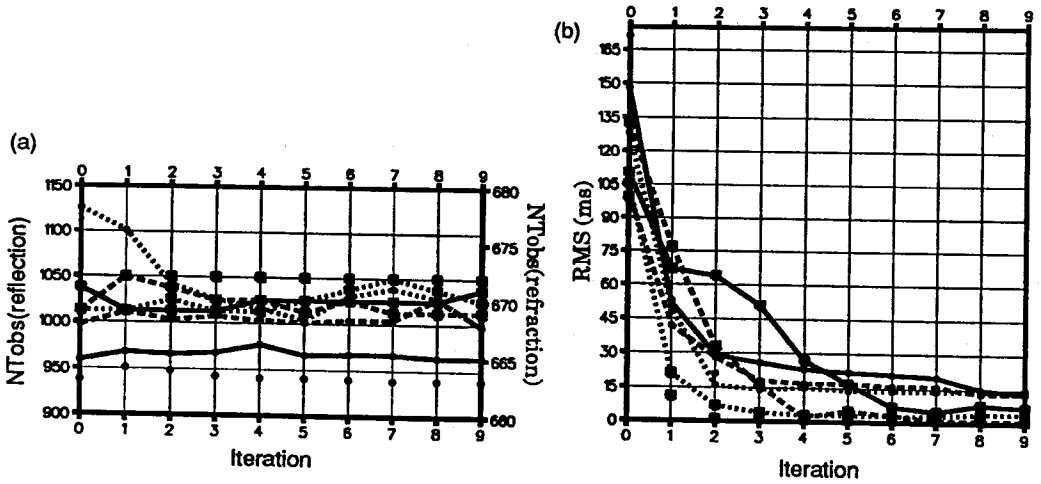

Figure 8 : (a) Number of observed travel times used by inversion with respect to iteration number for the four profiles of example 4. (b) RMS with res[ect to iteration number (circles: refraction travel times; square: normal incidence travel times).

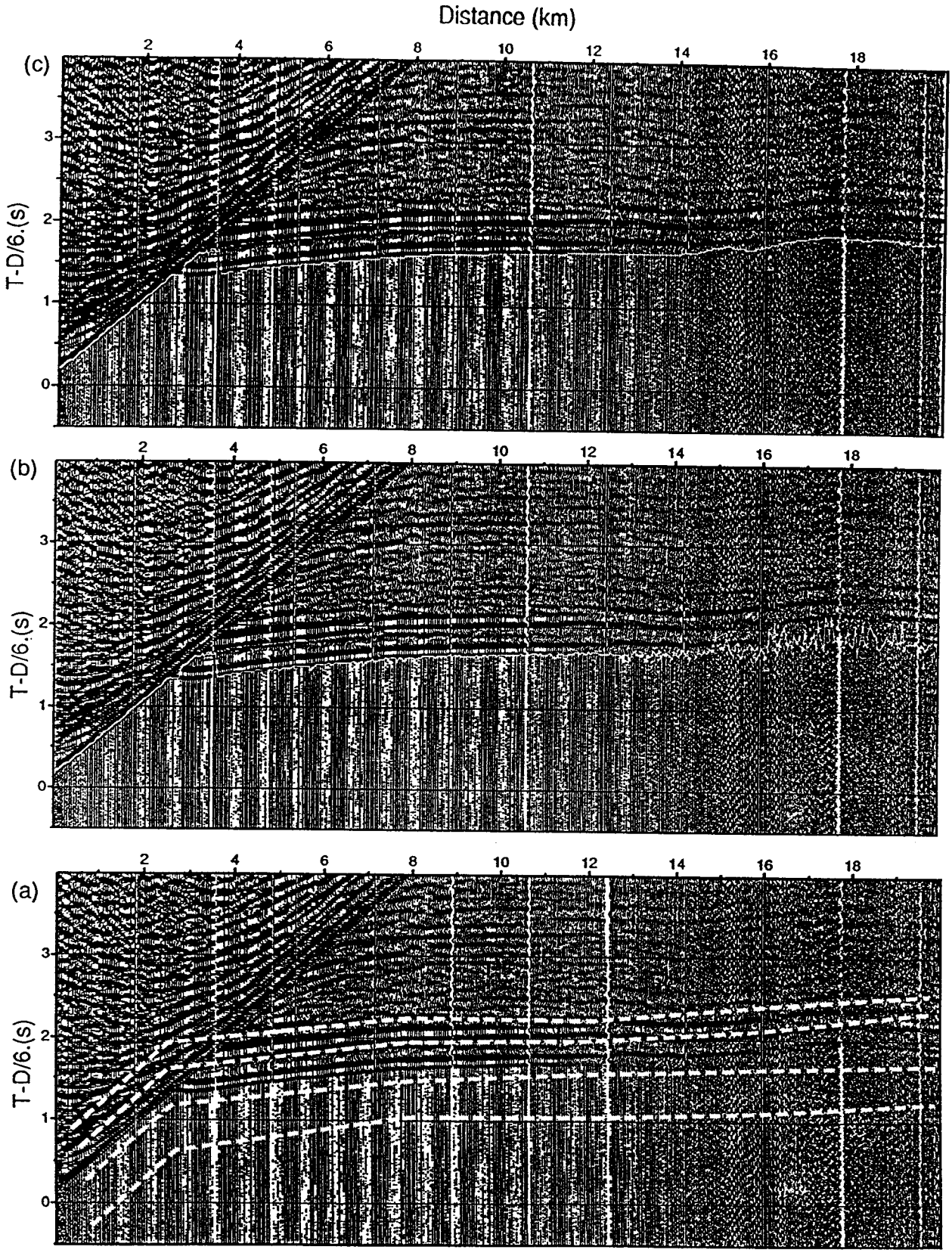

Figure A3.1: (a) Picking of the 4 boundaries required by the algorithm; (b) The picks inferred from the automatic process are superimposed on the date; (c) Same as (b) after smoothing of the picks. A window of 20 traces was used.

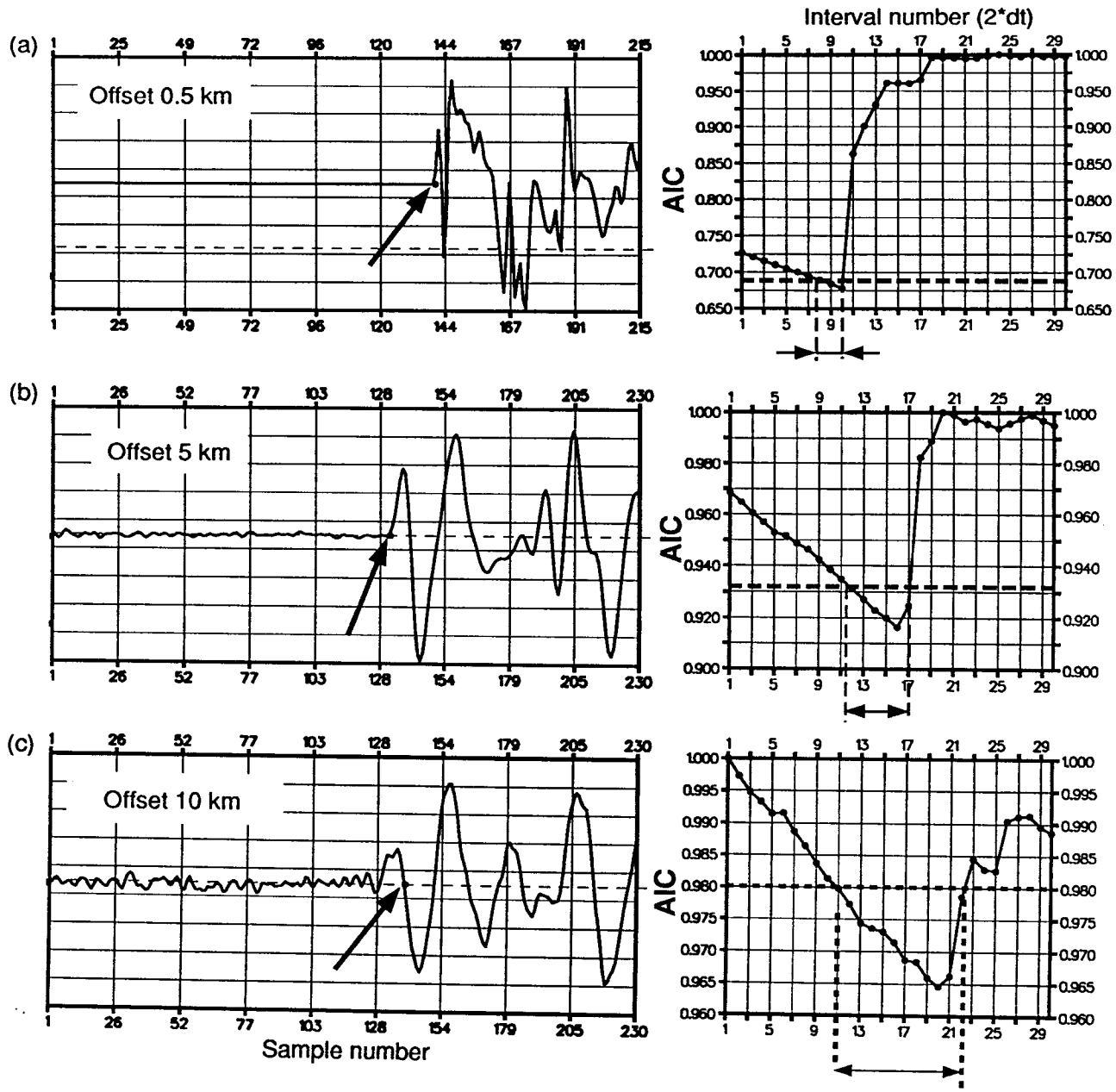

Figure A3.2: Automatic picking and relative uncertainty between picks. Panels on the left display three traces at 0.5km (a), 5km (b) and 10km (c) of distance. Location of the picked travel time is indicated by circle and arrow. On the right panels, the AIC curves are displayed. The travel time is identified by the minimum of the AIC curve (AICmin). To estimate relative uncertainty, a threshold (Thre) is set subjectively. Each travel time whose associated AIC value is between AICmin and AICmin+Thre consitutes a potential travel time. Difference between the minimum and maximum potential travel times provides an estimation of the relative uncertainty of the pick.