An Object-Oriented System for Full Waveform Data Processing

by

Marc Larrère

Earth Resources Laboratory
Department of Earth, Atmospheric, and Planetary Sciences
Massachusetts Institute of Technology
Cambridge, MA 02139

ABSTRACT

A new approach to the processing of sequences of full waveform acoustic logs is investigated. The rationale for this approach is primarily based on the observation that processing and interpretation tasks strongly depend on each other. Hence, a system that incorporates geologic knowledge in data processing naturally and uses processing results for petrophysical evaluation can improve the overall geological interpretation. The implementation of such ideas requires the use of a versatile computer environment, allowing numeric and symbolic processing. The new generation of *Lisp machines* satisfies these characteristics.

An interactive environment for the processing of sequences of acoustic signals was designed using *object-oriented* programming. The package includes a novel method for acoustic full waveform signal matching that uses *dynamic time warping.* The system is tested on synthetic data and field data are processed.

INTRODUCTION

The motivation for the AMIS[1] system is twofold: first,to provide an interactive processing environment for *sequences* of full waveforms (as well as single waveforms), and second,

[1] Acronym for A Modern Interactive System

to enable the operator to combine numeric and symbolic operations on signals. These two goals are complementary, since achieving these tasks requires a flexible structure oriented toward the easy manipulation of arrays of signals, elementary signals, and segments of signals. The general philosophy of the system is to make few assumptions about the specific methodologies of processing. The implementation on the *Lisp machine* uses object-oriented programming and takes advantage of the powerful programming environment — especially for graphical applications.

A determinant design choice was to define the concept of *sequence* of waveforms as the elementary object, as opposed to more general-purpose data processing systems that represent isolated signals (see for instance Kopec, 1984; Dove et al., 1984). This choice is essential in full wave acoustic data processing where the principal processing operations concern arrays of waveforms. It would be awkward to implement a velocity analysis or a controlled threshold detection technique if the elementary concept were a single signal. User interaction is an important facet of the system; Appendix A in Larrère (1987) illustrates the "style" of interaction and demonstrates the use of some operators and the geophysical applications of the AMIS system.

Since the system's philosophy and performance are strongly influenced by LISP programming and, more specifically, object-oriented programming, the main characteristics of these programming techniques are briefly discussed before describing the system's structure, the processing operators, and presenting applications to full waveform acoustic data.

LISP AND OBJECT-ORIENTED PROGRAMMING

LISP[2] is a language primarily devoted to symbol manipulation that originated at the same period as FORTRAN — the late fifties. It is being widely used now that suitable hardware has become available. LISP is a functional language, i.e. most programming is done by combining existing functions at different levels of specialization rather than by describing a sequence of operations. This process, called *procedural abstraction*, favors the partition of the task to more manageable subtasks and therefore makes incremental programming easy. LISP structure encourages a type of programming characterized by an "applicative" style, close to the composition of functions in mathematics. Furthermore, recursive applications of functions are possible and commonly used to describe procedures. In fact, the representation of programs is done with the same data structure (lists) as any other data. This enables the system to handle complex programs with the same ease as elementary data. Despite its orientation toward symbolic operations, compiled versions of LISP are also suitable for arithmetic computations (Winston and

[2] Acronym for List Processing.

Horn, 1984).

Another strategy to augment abstraction in programming is to encourage the organization of data. Suppose we are to write processing operators for single seismic waveforms. A simple and useful data structure that describes a general concept `WAVEFORM` will be composed of a time-series, a sampling rate and some identification. These components are also called the *slots* (or *attributes*) of the abstract data type (or *object*) `WAVEFORM`. The strength of *data-abstraction* is that pieces of related data can be treated as a unique entity. In addition, specific procedures are defined to construct, access and modify the elementary slots. This suppresses the burden of retrieving and organizing the diverse components for every specific task and enables better programming since, according to Winston and Horn (1984), "keeping track of such details can cause brain damage". Data abstraction allows concentration on high-level concepts and makes programs easier to modify since information is organized in well-defined compound structures.

A systematic recourse to data abstraction where objects are also responsible for the management of functions is called *object-oriented* (or object-centered) programming. In object-oriented programming, procedures are attached to objects in much the same way as any other attribute. LISP makes this easy to handle since data and programs are represented with the same basic structure: a list. *Message-centered* languages are a subspecies of object-oriented languages characterized by an original syntactic feature: a given procedure attached to an object is executed in response to a message sent by another object. Suppose we have two data types `WAVEFORM` and `SEQUENCE` (representing sequences of waveforms). We can associate a procedure "`draw-self`" to both objects that operates differently for a single waveform and for a sequence of waveforms. The adequate response is given when an instance of `SEQUENCE` or `WAVEFORM` receives the message "`draw-self`". The specific details for the actual execution of the task, however, are transparent for the higher level operations.

ZetaLisp is a dialect of LISP that includes a *message-centered* language called the *flavor system*. Flavors are non-hierarchically structured objects that can be mixed together to form a new concept. The "mixed" flavor inherits the attributes of each parent flavor as well as attached messages (also called *methods*). Invoking the application of a method is called message passing. A thorough description of the principles of message-centered programming and flavors can be found in Winston and Horn (1984).

In pure message-centered programming, an operation can occur only when an object sends a message to another object. In practice, every operation does not need to be initiated by message passing and low level procedures are performed via basic LISP functions. Operators on arrays are also written in LISP since compiled *ZetaLisp* is as fast as FORTRAN for arithmetic operations and includes powerful built-in functions for the description and manipulation of arrays[3].

THE AMIS SYSTEM

The AMIS system owes much to object-oriented programming concepts as described in the section below. The description of a few basic data types forms the core of the processing environment. This confers the ability of a fast and simple access to data, operators, and results. Another advantage of an object-oriented design for signal processing is that the development of processing algorithms and the practical utilization on real data are done with a unique language (Kopec, 1984). Thus, the tasks of development and utilization can be tackled within the same environment, which allows incremental improvement of the processing operators. Also, this type of structure is very well suited for encapsulating the processing operators into a knowledge-based system, both from the standpoint of data and result description and of the planning of processing operations.

Since data and results are represented as abstract objects, they can be easily manipulated and accessed. The only drawback of the actual implementation may be that instances of objects and attributes have no memory of their past values, i.e., application of an operator twice leads to loss of the first result. This type of bookkeeping can be handled by a higher level object structure. Since processing operators are defined as messages attached to data structures, they are manipulated with the same ease as data. Their applications can be easily controlled by high level constructor procedures.

The basic data-types for signals in the AMIS system are TRACE and SEQUENCE, representing respectively waveforms and arrays of waveforms. Processing operators are attached to sequences and/or traces depending on the nature of the task they perform. Some operators applied on certain classes of data can produce side effects, i.e., related processing results are assigned to the adequate slots of traces. Initial arrays of waveforms can be transformed with operators or they can be segmented: this leads to the creation of new instances of specialized sequences, respectively TRANSFORMED-SEQUENCE and SUB-SEQUENCE. Thanks to the object-centered structure, any creation of more specific instances confers also the ability of invoking the same collection of operators and of accessing all relevant information. In particular, "images" (i.e., graphical representation of objects in windows), stay physically present and are readily available in the environment. Again, an illustration of the possibilities of the system, with the help of practical sessions, is given in Appendix A of Larrère (1987).

[3]Basic mathematical operators on arrays could also be implemented with FORTRAN subroutines, using either network links or the *Symbolics* FORTRAN.

General Structure

The objects in AMIS are structured hierarchically. Two main concepts are defined at the root of the tree, representing in one part waveforms and arrays of waveforms (DATA-TYPE), and in the other part abstract data types useful for results and for the practical implementation (ABSTRACT-TYPE). A concept subsumed by other objects inherits their slots. Figure 1 shows a portion of the hierarchical relations between AMIS objects subsumed by DATA-TYPE. The data type SUB-SEQUENCE has an *instance* sub-sequence-03 that is an actual piece of data. It is subsumed by the object SEQUENCE which is a specific DATA-TYPE. The list of the definitions of objects is presented in Appendix B of Larrère (1987). The central concept is the data type SEQUENCE that embodies two important slots, **image** and **list-of-traces**.

Image represents the abstracted part of the object SEQUENCE, related to graphical representations. Processing operations are primarily attached to images of sequences rather than to sequences themselves.

list-of-traces relates a sequence to its primary components, i.e., the individual waveforms, represented by the data type TRACE.

Since every data type slot is assumed to be an instance of some defined object in the environment, the overall structure forms a description of the *semantic* of the domain, i.e., of the *meaning* of links between the various concepts. A piece of this network is shown in Figure 2. This network shows, in particular, that a SEQUENCE has an image, which is an instance of the particular object SEK-IMAGE, that is itself an ABSTRACT-TYPE containing other members of ABSTRACT-TYPE called REGIONS that contain DATA-TYPE objects.

Processing Operators

Processing operators in AMIS are messages that can be sent to instances of SEQUENCE or TRACES. The listing of these messages is given in Appendix C of Larrère (1987). Most operators are built on lower level array processing functions. The most important operators on SEQUENCE are:

1. Operators for single signals, generalized for sequences of traces, including normalization, interpolation with cubic splines, estimation of maxima in time-windows and computation of envelopes via moving average.
2. Two methods for picking:

- Automatic threshold detection for all kinds of waves. The appropriate time-band for picking is restricted by taking into account the nature of the wave.
- Manual picking with the mouse[4]. A minimum of two picked points is required. The values for other waveforms in the sequence are linearly extrapolated or interpolated.

3. An accurate computation of move-out between traces using *dynamic time warping*[5] with the possibility of interactively adapting and optimizing the parameters — i.e., the position of windows and the number of points.
4. The study of wave dispersion in the time domain — computation of phase velocity variations as a function of the length of the path of propagation.

Some operators are very general and can be invoked for any type of sequence and trace, others are restricted to certain data types. The two following examples illustrate why the field of application of operators is sometimes restricted:

- The message "`envelope`" can be sent to any type of trace and sequence, including fragments of sequences and already transformed sequences. The operator is not task-dependent, hence messages for traces and for sequences are built on the same very general LISP function.
- The message "handpick-arrival-time" is only defined for instances of `RAW-SEQUENCE` of at least two traces, and does not make sense for an instance of `SUB-SEQUENCE` except when the value of the type slot of `SUB-SEQUENCE` is P-, S-, or Stoneley waves.

Processing results are described by two objects linked with the instances of `TRACE`. These objects are `INITIAL-PROCESSING-VALUES` and `FINAL-PROCESSING-VALUES`. Picking methods (i.e., automatic threshold detection and manual picking) fill the "initial-values" slots of waveforms. The initial arrival time values are then used to compute velocities with signal matching and the results are transferred to the "final-values" slots of waveforms. All results, as well as the history of operations applied to a given instance of `SEQUENCE` can be retrieved with the help of specific messages (see the list of operators in Appendix C).

[4]This technique is not intended to give precise arrival time estimates since there is the limitation of the initial sampling rate. Nevertheless, it can provide high manual precision picking if done after spline interpolation.

[5]The technique is described in the next section.

SIGNAL MATCHING WITH DYNAMIC TIME WARPING

Dynamic Time Warping

Dynamic time warping can be regarded as a generalization of cross-correlation that allows not only shifting but also stretching and squeezing of one signal with respect to the other. A general signal matching problem consists of estimating a mapping function between two time series at any point in time. This mapping function must be such that it minimizes a given measure of dissimilarity or *distance* between the two signals. Therefore, the problem can be formulated as an optimization problem, and was tackled with two different approaches:

- A non-linear least-square inversion for estimating the mapping function as a sum of simple analytical functions. Martinson et al. (1982) used truncated Fourier series and successfully applied this technique to geophysical data.
- The problem can also be formulated as a search in the two-dimensional discrete space of accumulated distances between the two signals. Sakoe and Chiba (1971) proposed an algorithm using dynamic programming. The technique, called *dynamic time warping*, was widely used and developed for speech recognition problems (see Rabiner et al., 1978). Anderson and Gaby (1983) review some possible applications in the geophysical domain, among which waveform classification and well-to-well log correlation (Lineman, 1986) were developed. He suggested the use of dynamic time warping for the processing of entire sonic waveforms.

Figure 3 shows the dynamic time warping problem for two discrete signals a_i and b_j of respective lengths N and M. We need to determine a discrete mapping function $c_k = [i(k), j(k)]$ that corresponds to a minimum distance between each couple of samples. Choosing a local cost function $d(c(k))$, we are to minimize the overall cost function $D(c) = \sum d(c(k))$. This problem is equivalent to a path finding problem in the $N \times M$ discrete domain of accumulated costs (see Figure 3). Given constraints on endpoints and with the definition of the legal local moves (the set of allowed moves from any point $[i, j]$ to its neighbors), it can be shown that the minimum cost path from the origin $[0, 0]$ to any point $[i, j]$ is independent of what happens beyond this point. The minimum cost path is determined recursively by minimizing more and more local costs. An optimal path finding algorithm using dynamic programming can be applied to evaluate the mapping function c_k (Sakoe and Chiba, 1971). A complete description of the algorithm can be found in Parson (1986).

Application to Full Waveform Processing

The mapping function $c_k = [i(k), j(k)]$ is a representation of time shifts between the two input signals for all samples. The time-shifts are $\delta t_k =\mid i(k) - j(k) \mid$. Figure 4a illustrates the case where the two signals are identical: we have $i(k) = j(k)$ for all k, hence c_k is a straight line between the initial and final tie-points, and $\delta t_k = 0$, for all k. Figure 4b shows that if two identical signals are shifted by a constant number of samples s (corresponding to a time move-out t_s), the theoretical mapping function is a straight line beginning at $c_1 = [0, s]$. If the time shift between the two signals increases with time, as shown in Figure 5, the mapping function departs from the constant slope. The dynamic time warping technique presents applications two for full waveform matching in the context of the AMIS system.

- The method is potentially very accurate for recovering the variation of move-out with time due to wave dispersion between a *couple* of waveforms.
- The nature of the algorithm enables the operator to control the mapping and to set constraints in order to limit the space of possible matches. These constraints can drastically prune the search tree, hence making the matching computationally effective.

This signal matching technique was applied to two slightly different problems: first, to perform fast correlations on short windows to estimate travel times (and therefore wave velocities); second, to make detailed analyses of the dispersion of arrivals in the time domain. The first task could be addressed with traditional cross-correlation techniques since the very beginning of arrivals is in general not dispersed. However, for dispersed waves (i.e., PL modes, pseudo-Rayleigh and Stoneley) determining the phase velocity as a function of time can be done by non-linear matching techniques.

Velocity determination The determination of a wave velocity with dynamic signal matching involves five steps:

1. Determination of the arrival-time t_0 of the wave for each waveform in the sequence using a fast picking method.
2. Windowing the arrival around t_0; the window length depends on the dominant frequency — for instance, the window is longer for the S wavetrain than for the P-wave.
3. Interpolation of the windowed waveform with a cubic spline. The interpolation factor depends on the precision required.

4. Correlation by dynamic time warping with a maximum time shift constraint. The maximum time shift is the result of a trade-off between confidence in the first estimate and computational cost.

5. Computation of the wave velocity from the initial move-out and the dynamic time warping time shift.

The process is repeated for every couple of waveforms in the sequence. The first step is essential for the reliability of results, especially in the case of S-wave detection. For an interactive process, P- and S-wave first arrival estimates are obtained with automatic or manual picking and errors can be easily and quickly corrected. For an automated process, a priori assumptions must be made (choice of a model) and control procedures must be set in order to check the validity of the picks.

Dispersion studies Signal matching with dynamic time warping allows the study of the dispersion of arrivals in the time domain for two waveforms. This information could also be obtained in the frequency domain or via τ-p transform but these methods require arrays of waveforms to work. The study of dispersion is restricted to a part of the waveform for two reasons:

- The measure of similarity between signals emphasizes the resemblance of the prominent waves. Because high amplitude arrivals have a prevailing contribution on the cost function, weak arrival are matched less accurately. For instance, in a hard formation and for short offsets, the method cannot resolve the P-wave time delays because of the dominant energy in the pseudo-Rayleigh wave.
- The computational cost is too high when two entire waveforms are matched[6] with the high sampling-rate required for sufficient precision.

RESULTS

Synthetic Microseismograms

Results using synthetic waveforms are presented first in order to test the accuracy of the method. Synthetic microseismograms were generated with the discrete wavenumber

[6]The computational cost of dynamic time warping is theoretically proportional to the product of the two signal lengths. In fact, the internal building of the recursion slows down the computation when signal lengths pass a given threshold.

method in the case of an open borehole surrounded by an homogeneous formation. The mechanical properties are: $V_P = 4000$ m/s, $V_S = 2310$ m/s, $Q_P = 50$, $Q_S = 25$, $\rho = 2.4\times 10^3$ kg/m^3. The time sampling rate is 11.84 microseconds. The borehole radius is 10 cm and the center frequency of the source is 5 kHz. Figure 6 shows the complete sequence of synthetic waveforms. The source to receiver distances range from 2.50 m to 8.00 m in increments of 0.50 m. The energy in the Stoneley wave is prodominant for all of the waveforms.

Velocity determination A P-wave velocity analysis was performed on the synthetic data shown on Figure 6. For the sake of illustration, the initial automatic threshold detection step was ill-done. There is a cycle-skip for the 7.00 m offset. This cycle-skipping was the result of a relatively poor signal-to-noise ratio for large offsets due to the attenuation. Figure 7 and 8 show the time-windowed P-waves after spline interpolation. The window length is about two and a half cycles and the time sampling is less than a microsecond.

As shown on Figures 9 and 10, the mapping functions are nearly perfect straight lines for offsets less than 5.00 m. The quality of match decreases for larger offsets, as the level of numerical noise increases. Note also that the mapping function between the 6.50 m and 7.00 m offsets — involving a cycle-skip — shows a linear part that corresponds to the maximum shift constraint on the match. This indicates that paths with lower cost could be found if larger move-outs were tolerated. The move-out value is estimated by taking the average of the three most common time-shift values, excluding the extremities of the mapping function. This estimate was found to be more robust than the straightforward average value.

Since the mapping function is constrained to stay in a diagonal band defined by a maximum time shift, matching the 7.00 m offset (with initial cycle-skipping) corresponds to a wrong estimate. In order to make the proper correlation the initial window length must be larger, so that it includes the first skipped cycle, and the constraint on the maximum shift must be relaxed. These increase the computational cost significantly.

Excluding results corresponding to the arrival detected with a cycle-skip, all final velocity values are within 2.0 % of the theoretical value. The average value for the entire array is 3960 m/s (the theoretical value being 4000 m/s). The determination of the S-wave velocity (not shown here) was done with the same relative error. This example is representative of the order of precision of the method as applied to a few sequences of synthetic seismograms. Tests for waveforms without attenuation showed less deviation. All results, including the P-wave velocity determination, have a systematic negative bias, i.e., an underestimation of velocities, on the order of 0.5 % to 1 %. This is consistent with the fact that body wave velocities are in all cases upper bounds to the phase velocities of guided arrivals and leaky waves.

Dispersion study A study of the dispersion of the first cycles of the pseudo-Rayleigh arrival in the time domain was done on the same set of microseismograms. Figure 11 shows the initial sequence, from which the beginning of the pseudo-Rayleigh wavetrains corresponding to offsets of 5.00 m and 5.50 m were correlated with dynamic time warping. The two signals are normalized and interpolated before signal matching.

The result is shown on Figure 12. As expected, the general trend is a decrease of the phase velocity with time. The very beginning of the arrivals is weak in amplitude and contains a small component of P-wave arrival which explains the scatter in the results. For the first 500 microseconds, the average velocity is about 2300 m/s (the theoretical S-wave velocity being 2310 m/s). The last 300 microseconds correspond to a decreasing phase velocity, from 2300 m/s to about 2100 m/s. Thus, the underestimation of the S-wave velocity depends on the length of the window estimate. Nevertheless, taking the average over a 1000 microsecond window still provides a good estimate of the S-wave phase velocity (2250 m/s).

Field Data

The sequence displayed on Figure 13 is twelve traces of field data. The first receiver is ten feet from the source and the distance between successive traces is a half foot. Each trace contains a P wave, a pseudo-Rayleigh wavetrain and a Stoneley arrival. The relative amplitude of the pseudo-Rayleigh arrival is low.

A velocity analysis with signal matching was performed for each couple of traces for the P, S and Stoneley waves. The respective average values for the velocities are 4100 m/s, 2440 m/s and 1460 m/s. These values agree very well with results obtained with the semblance method and the maximum likelihood method (Ellefsen et al., this volume). The velocities between successive slices of formation, however, show important variations. For the P wave, velocities vary between 3400 m/s and 4500 m/s, for the S wave between 2140 m/s and 2900 m/s. Accurate signal matching gave no significant trend for the variation of P- and pseudo-Rayleigh wave phase velocities. As shown on Figure 14, the pseudo-Rayleigh arrival seems to be fairly non dispersive.

CONCLUSIONS

The AMIS system proved to be well-suited for interactive processing of sequences of waveforms. The primary advantage over other types of structure is that the core concept of *sequence* makes possible the easy manipulation of complex two-dimensional objects.

The system is adequate for a moderate amount of data, i.e., for sets of a few dozens of traces. The operators are flexible and accurate, as demonstrated by tests on synthetic data. These tests also confirm that the S-wave velocity is in general well-estimated from the characteristics of the pseudo-Rayleigh arrival. Nevertheless, the general trend is to underestimate velocities, by an amount that depends on the mechanical properties of the formation.

Working with graphical representations of signals provides an instantaneous understanding of the effects of operators. Thus the user has the ability to redo operations easily, until the processing results are satisfactory. This type of approach is very useful for development and for testing tasks.

The *message-oriented* style of programming allows modularity. Operators can be easily encapsulated in more complex and general structures. This latter characteristic is essential for further development and provides a wide range of applicability. Basic operators form a very top-level language that can serve as a basis to construct more specific tools. The ability to treat the operators as abstract structures is also essential for integration in a knowledge-based system for full waveform interpretation.

ACKNOWLEDGEMENTS

This work was supported by the Full Waveform Acoustic Logging Consortium at M.I.T. and by an Elf Aquitaine Fellowship.

REFERENCES

Anderson, K.R. and Gaby, J.E., 1983, Dynamic waveform matching; Information Sciences, 31, 221–242.

Dove, W.P., Myers C., and Milios, E.E., 1984, An object-oriented signal processing environment; The Knowledge-Based Signal Processing Package: M.I.T., R.L.E., Technical Report No. 502.

Kopec, G. E., 1984, The integrated signal processing system ISP; IEEE Trans. Acoust., Speech, and Sig. Proc., ASSP-32, No. 4.

Lineman, D.J., Mendelson, J.D., and Toksöz, M.N., 1987, An expert system for well-to-well correlation; Submitted to Bull. AAPG.

Martinson, D.G., Menke, W., and Stoffa, P., 1982, An inverse approach to signal correlation; J. Geophys. Res., 78, 4807–4818.

Myers, C.S., 1980, A comparative study of several dynamic time warping algorithms for speech recognition; M.S. Thesis, Massachusetts Institute of Technology, Cambridge, MA.

Parson, T.W., 1986, Voice and Speech Processing; McGraw-Hill, 297–303.

Rabiner, L.R., Rosenberg, A.E., and Levinson, S.E., 1978, Considerations in dynamic time warping algorithms for discrete word recognition; IEEE Trans. Acoust., Speech, and Sig. Pro., ASSP-26, No. 6.

Sakoe, H., and Chiba, S., 1971, A dynamic programming approach to continuous speech recognition; Proc. Int. Cong. Acoust., Budapest, Hungary, Paper 20C-13.

Winston, P.H., and Horn, K.P., 1984, Lisp, Addison-Wesley, Reading, MA.

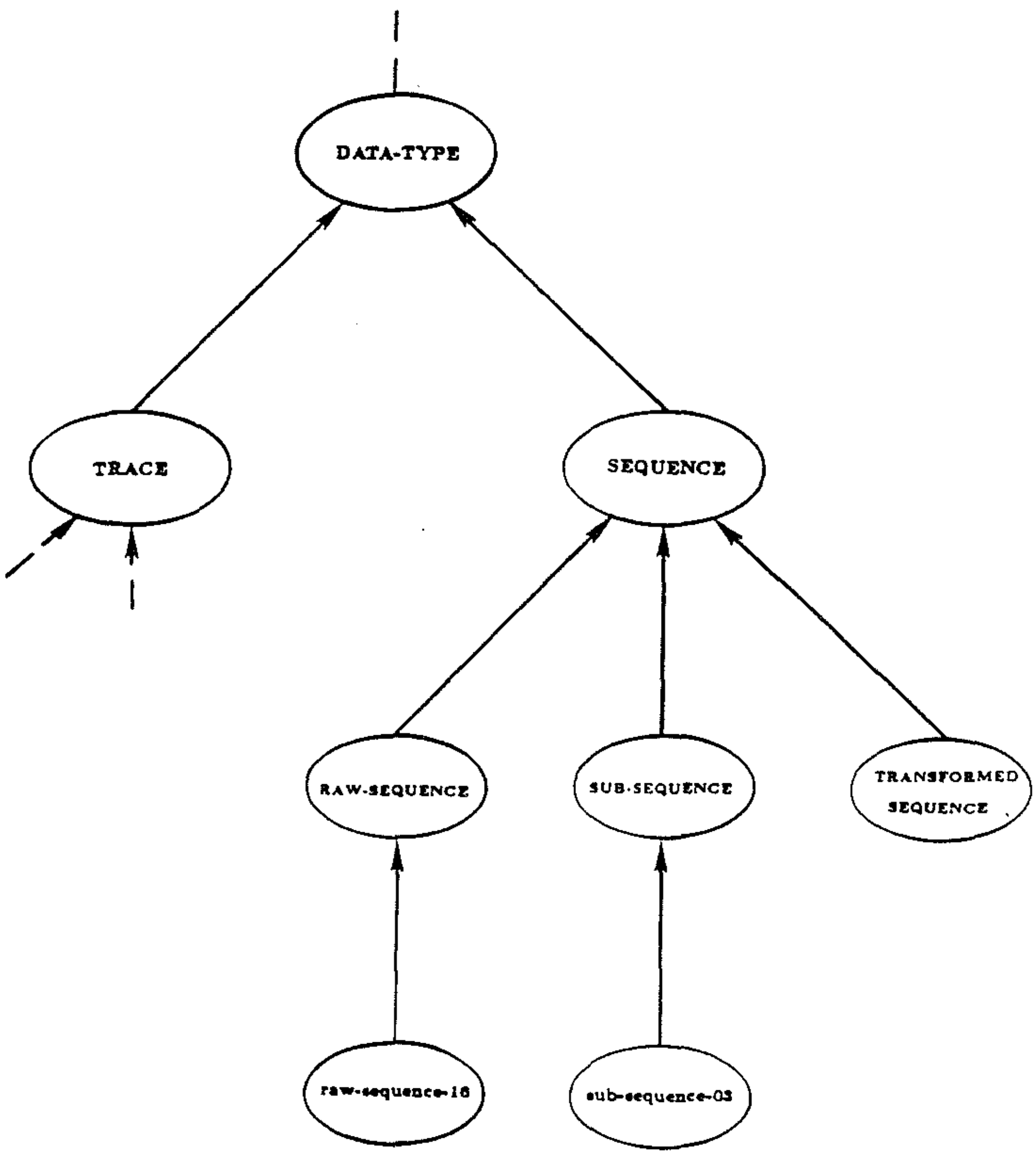


Figure 1: Hierarchical relations between AMIS objects: raw-sequence-16 and sub-sequence-03 are instances of the abstract data types RAW-SEQUENCE and SUB-SEQUENCE.

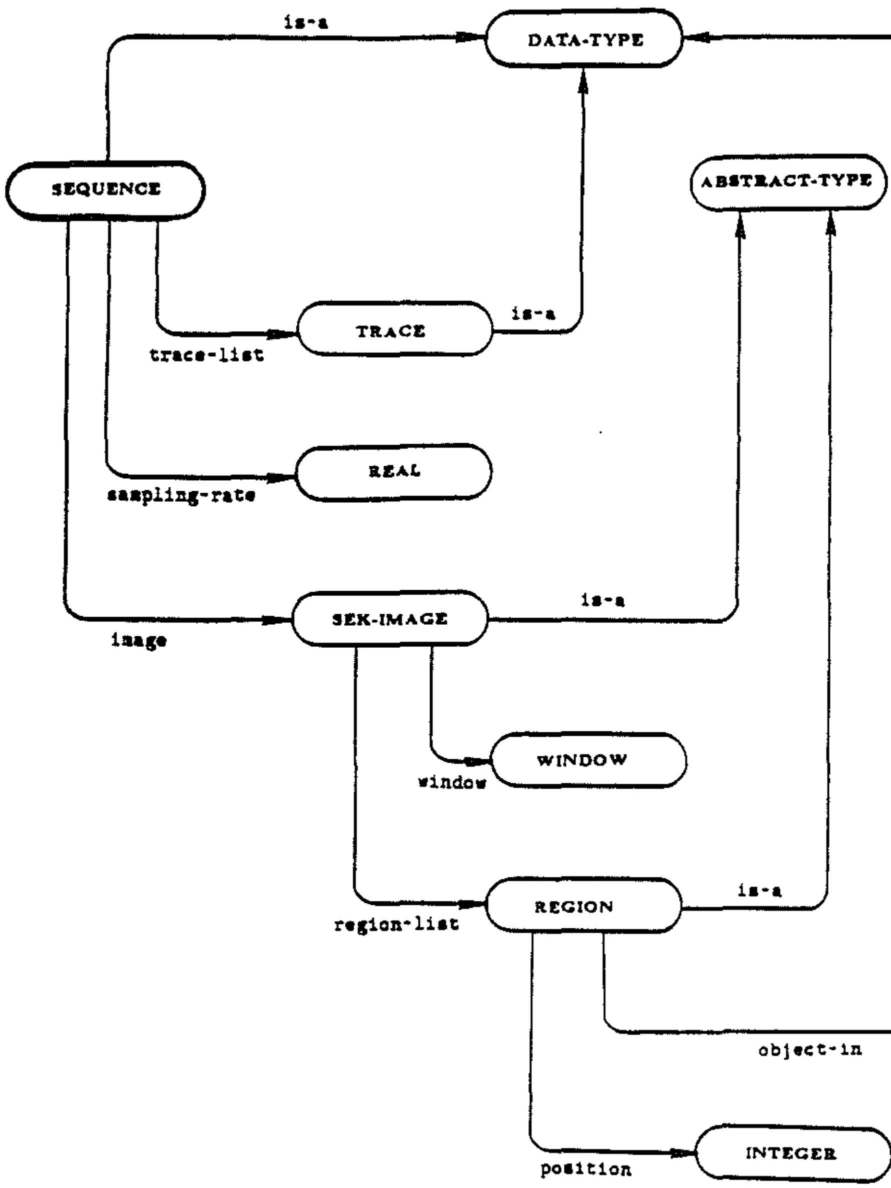


Figure 2: A partial representation of the network of relations between AMIS objects and attributes. IS-A links represent subset-set relations between concepts.

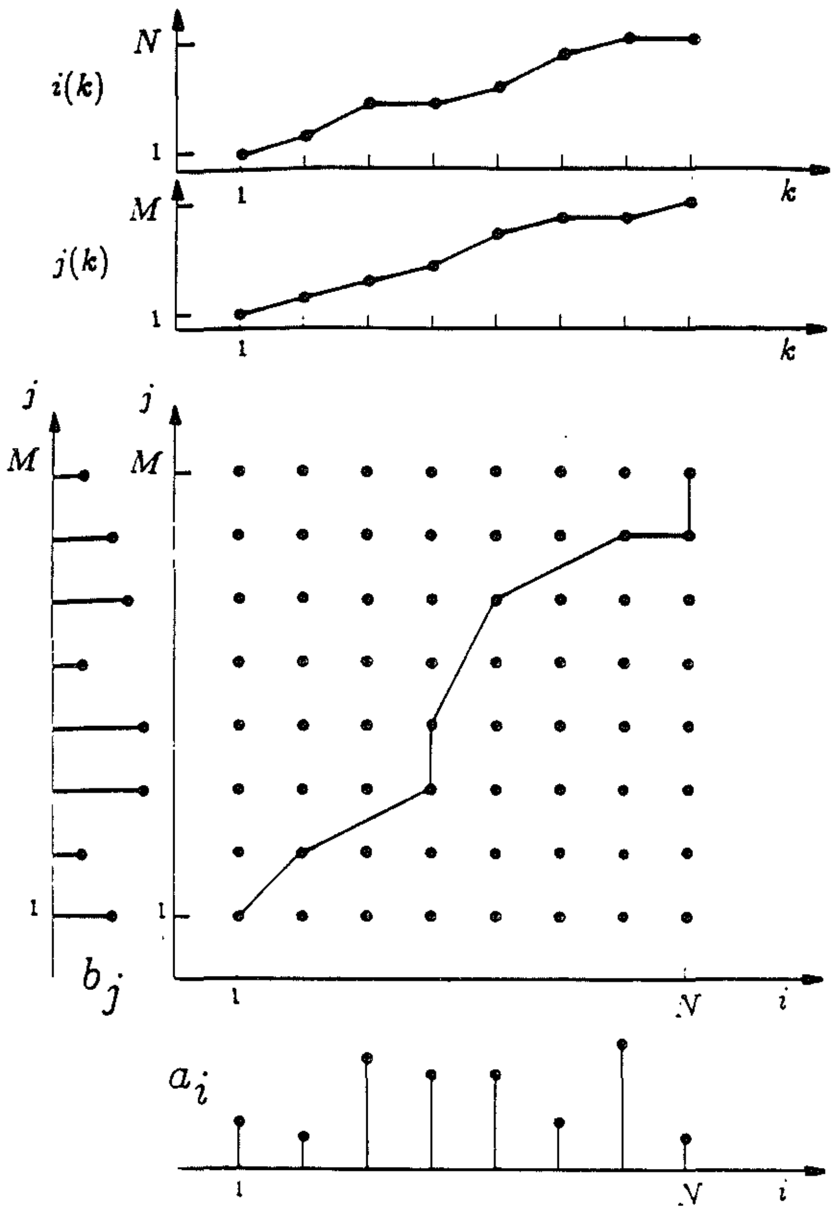
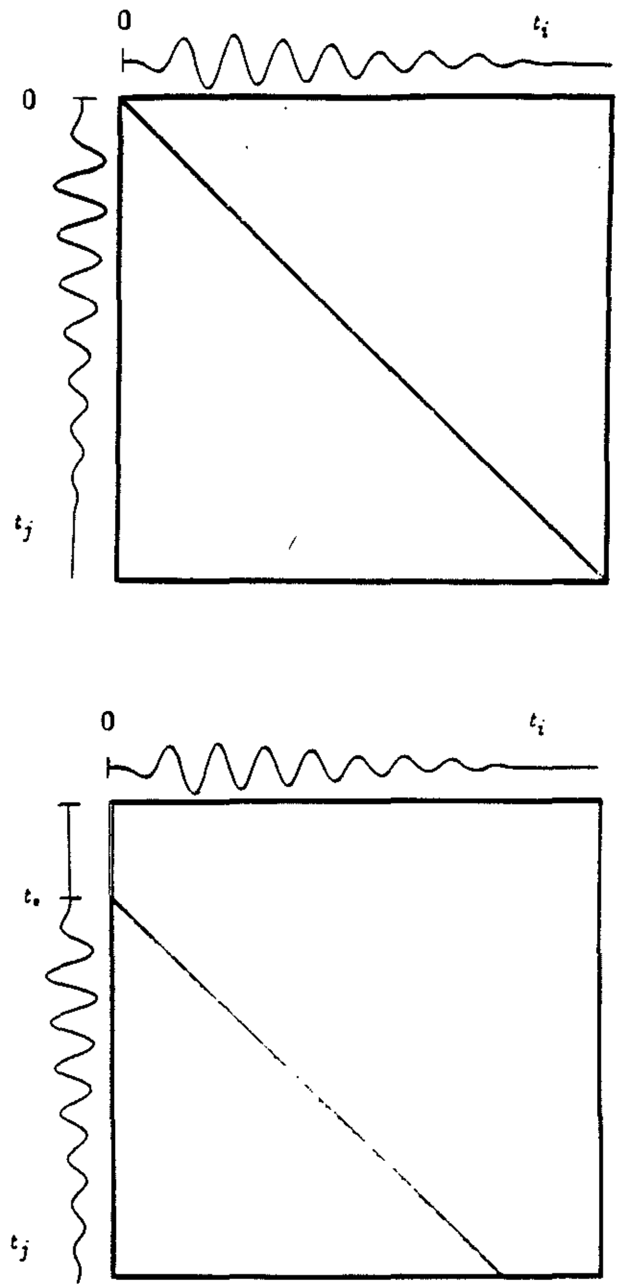


Figure 3: Matching of two discrete signals with dynamic time warping. The mapping function is $c_k = [i(k), j(k)]$. (After Myers, 1980.)



(a)

(b)

Figure 4: Mapping functions corresponding to: a) Two identical signals; b) Two identical signals with a time shift t_s.

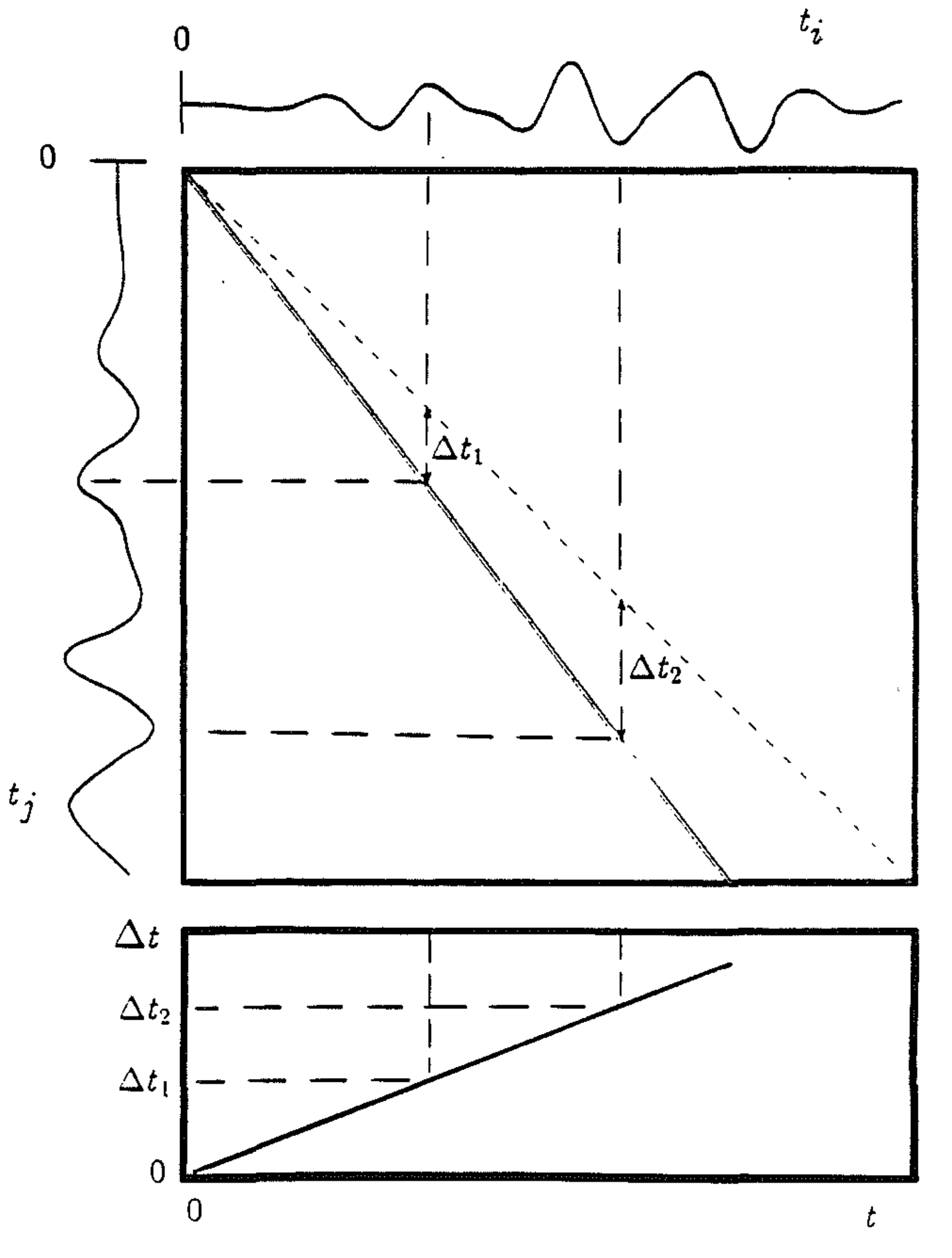


Figure 5: The mapping function between two signals (nearly identical but one — t_j — is stretched relative to the other — t_i) and the corresponding time shifts $\Delta t(t)$.

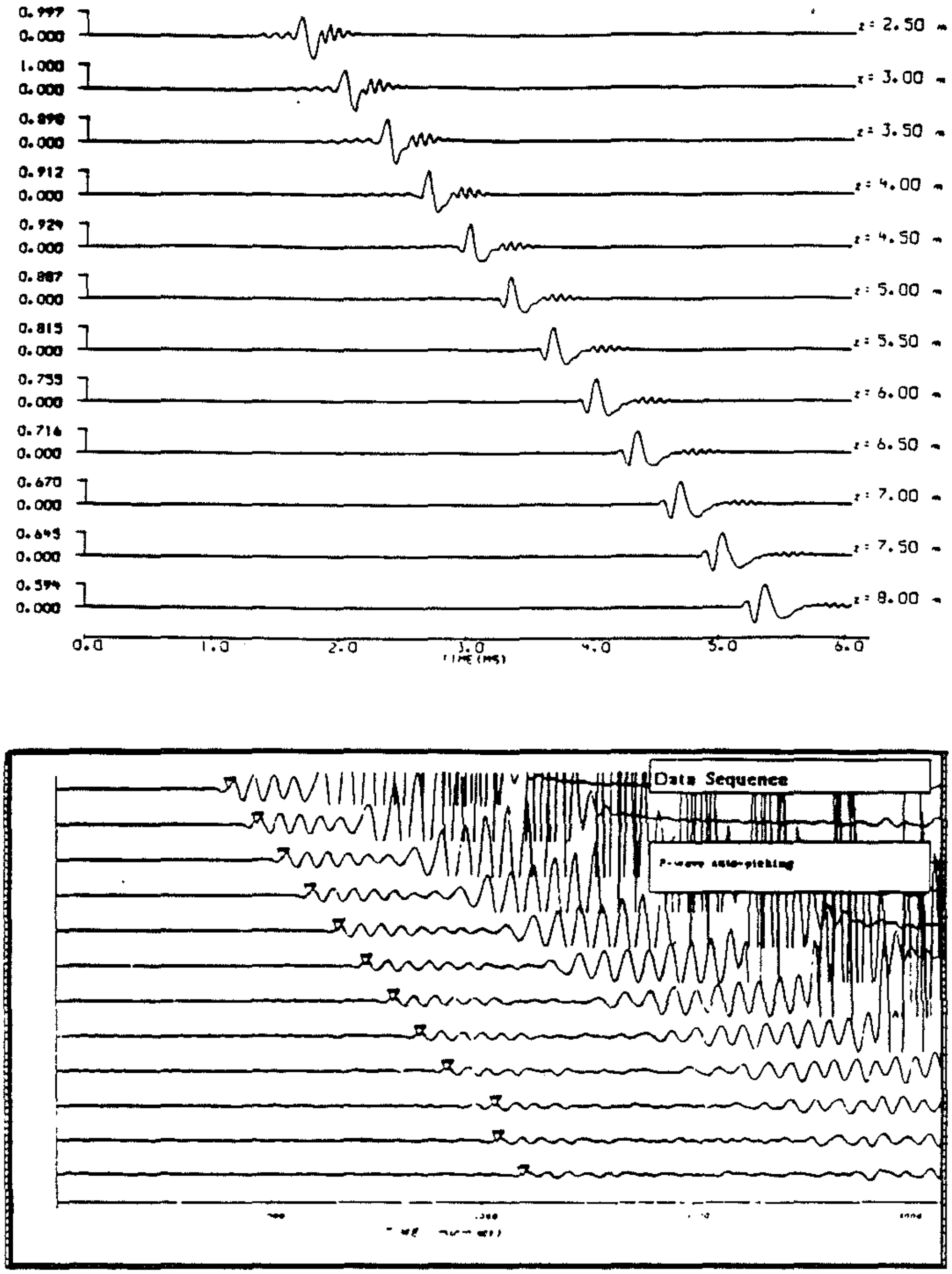


Figure 6: Synthetic microseismograms and automatic threshold detection of the P-wave arrival. The amplitudes are magnified in the lower diagram to show the P-waves. Note the cycle-skip at 7.00 m.

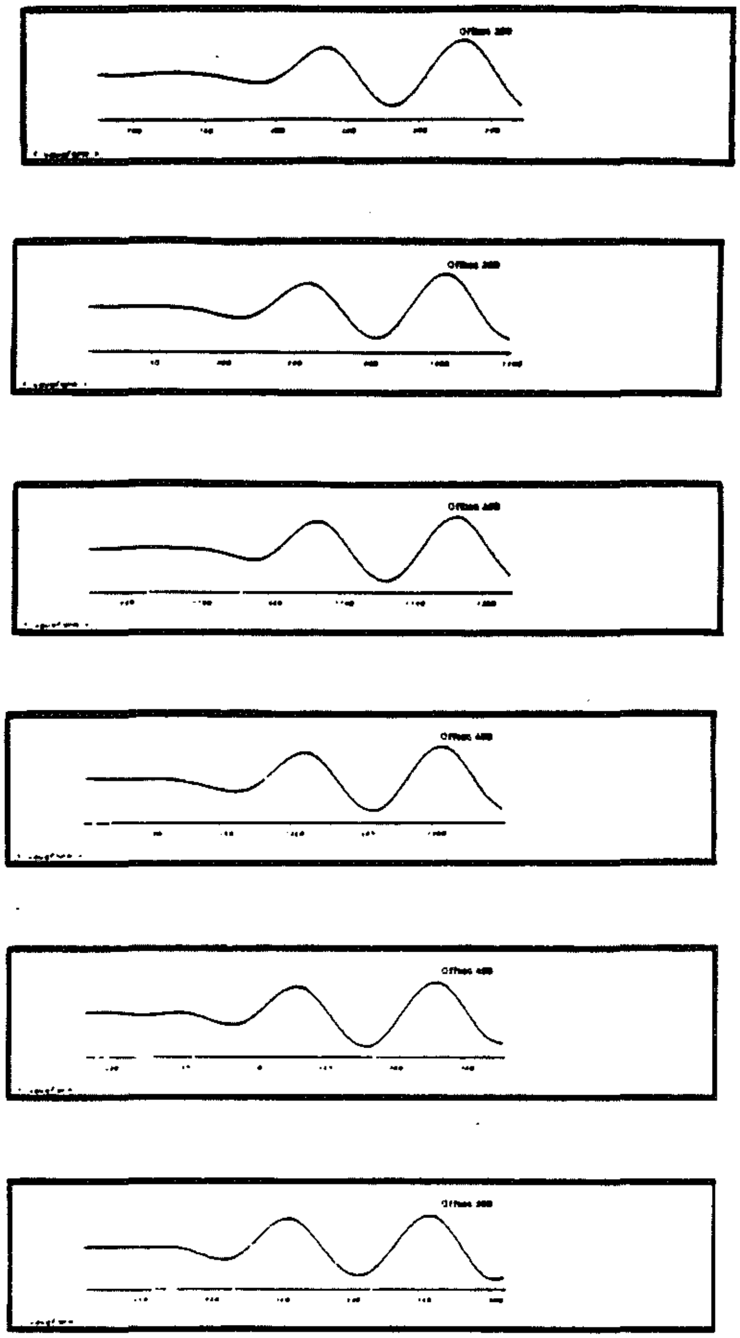


Figure 7: Windowed P-waves for signal matching. Offsets are from 2.50 m to 5.00 m.

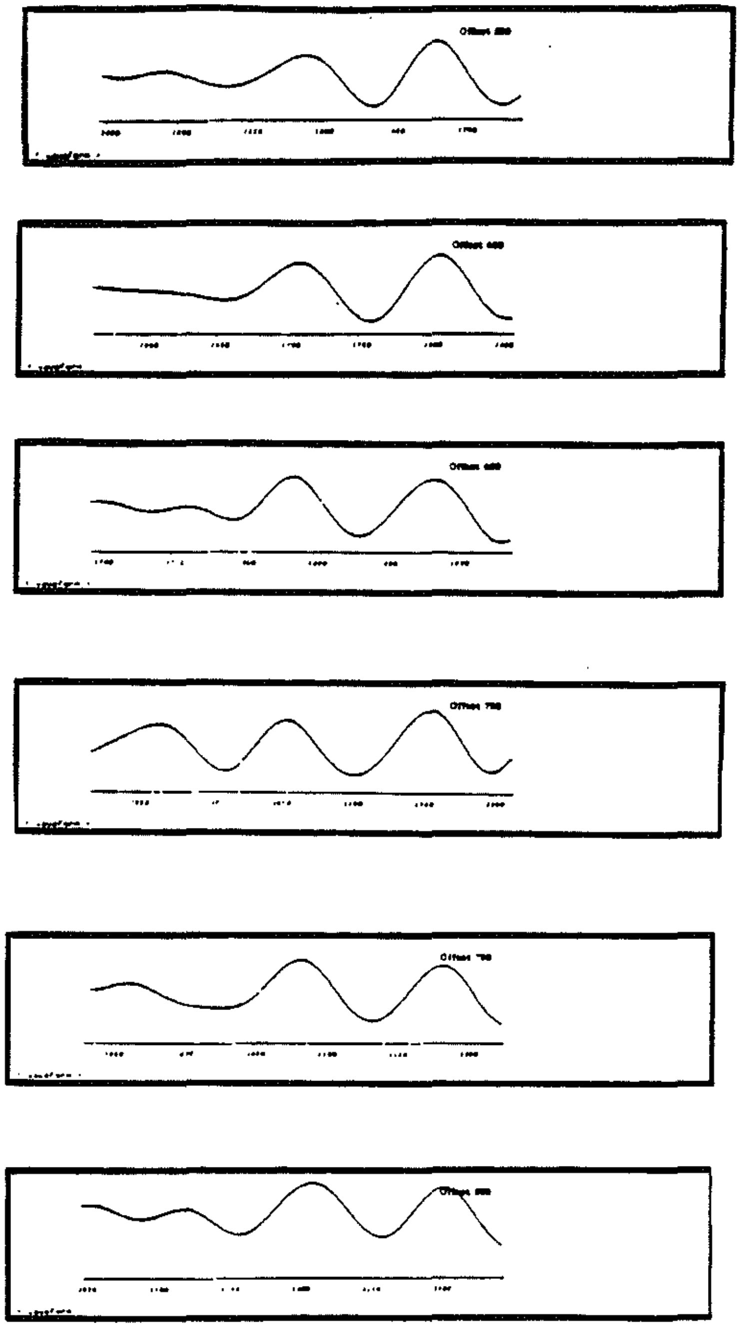


Figure 8: Windowed P-waves for signal matching. Offsets are from 5.50 m to 8.00 m.

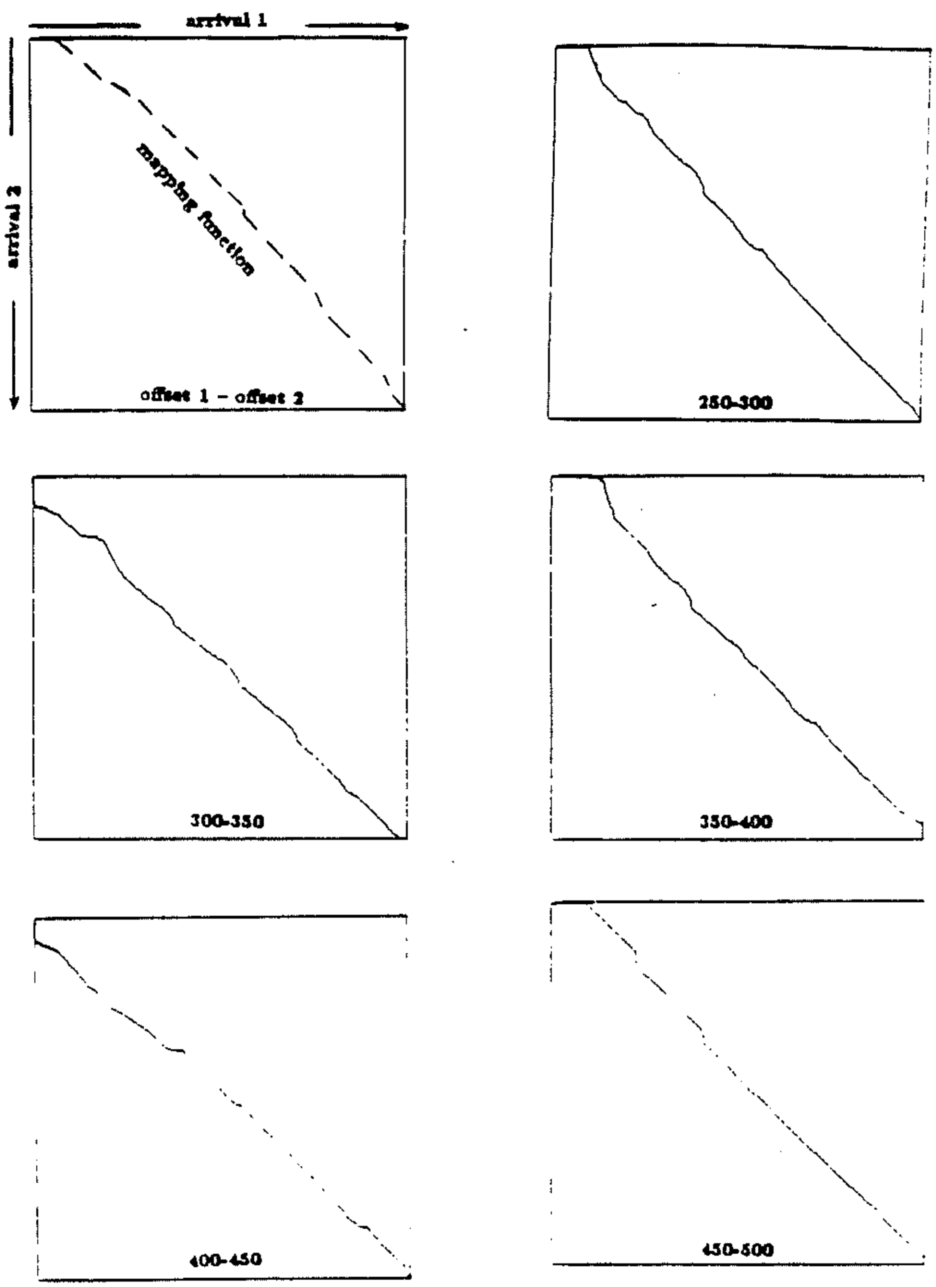


Figure 9: The mapping functions for adjacent waveform pairs. Offsets are from 2.50 m to 5.00 m.

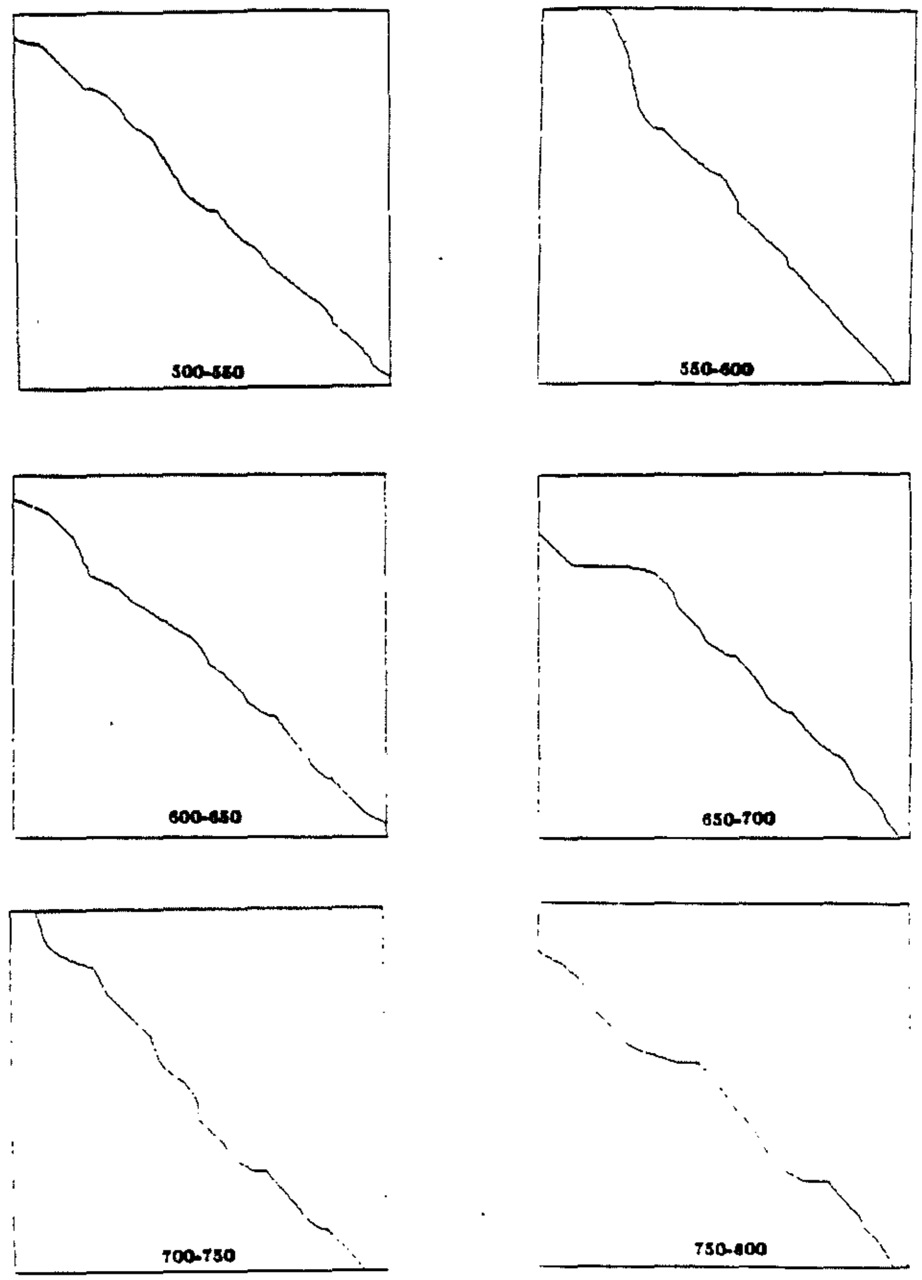


Figure 10: The mapping functions for adjacent waveform pairs. Offsets between 5.00 m and 8.00 m. As the signal-to-noise ratio decreases the mapping function becomes more noisy. Note the effect of the maximum time shift constraint on the beginning of the mapping function at 650-700.

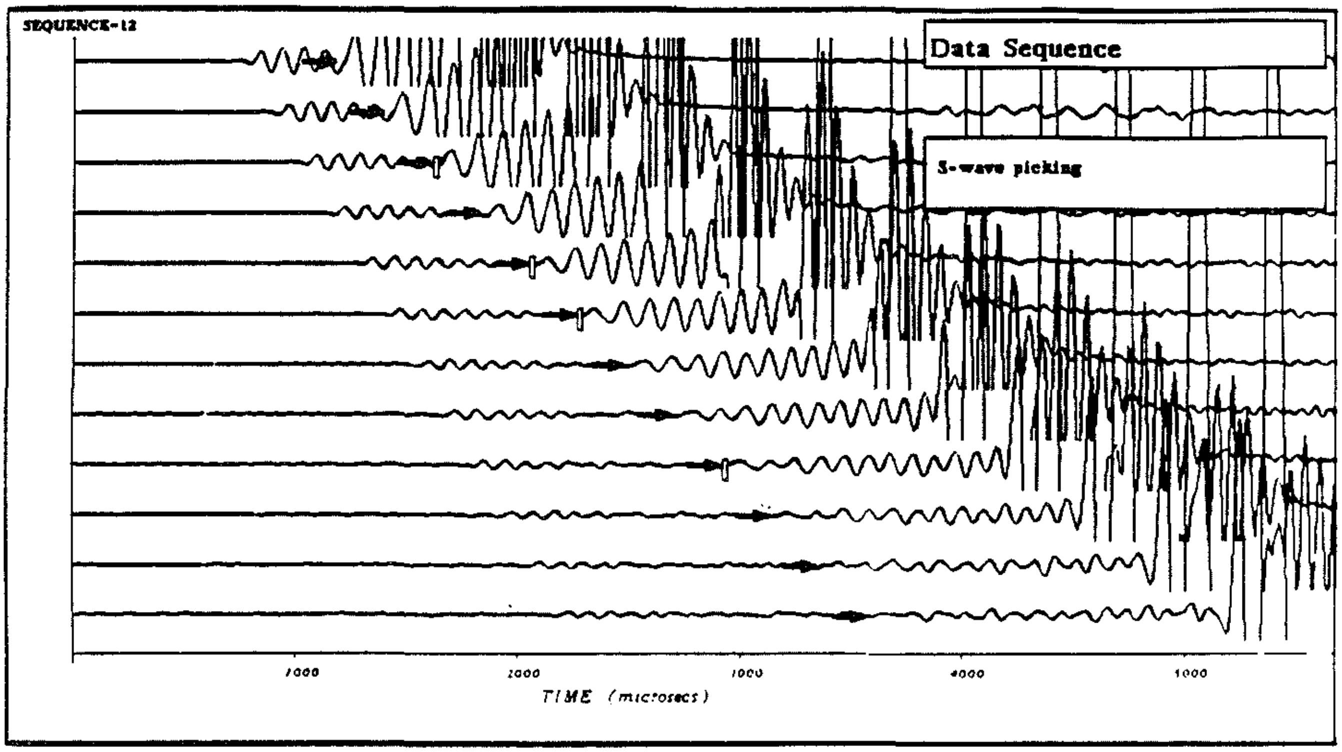


Figure 11: Picking the S-wave onset with the mouse. Small rectangles correspond to the arrival times picked by the user. Arrows point at interpolated time values for all traces.

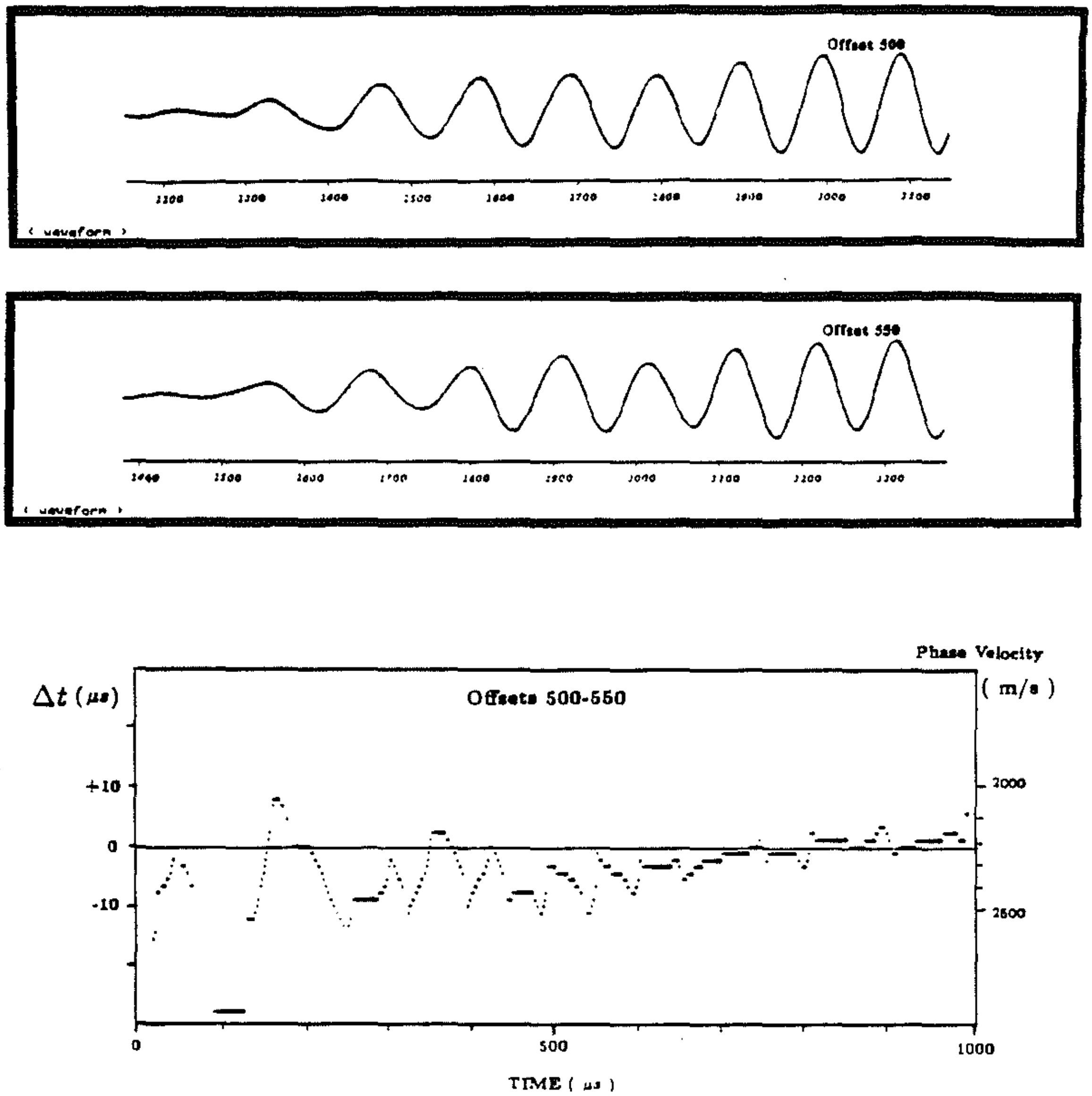


Figure 12: Two pseudo-Rayleigh arrivals for the offsets 5.00 m and 5.50 m and the corresponding variation of time delays.

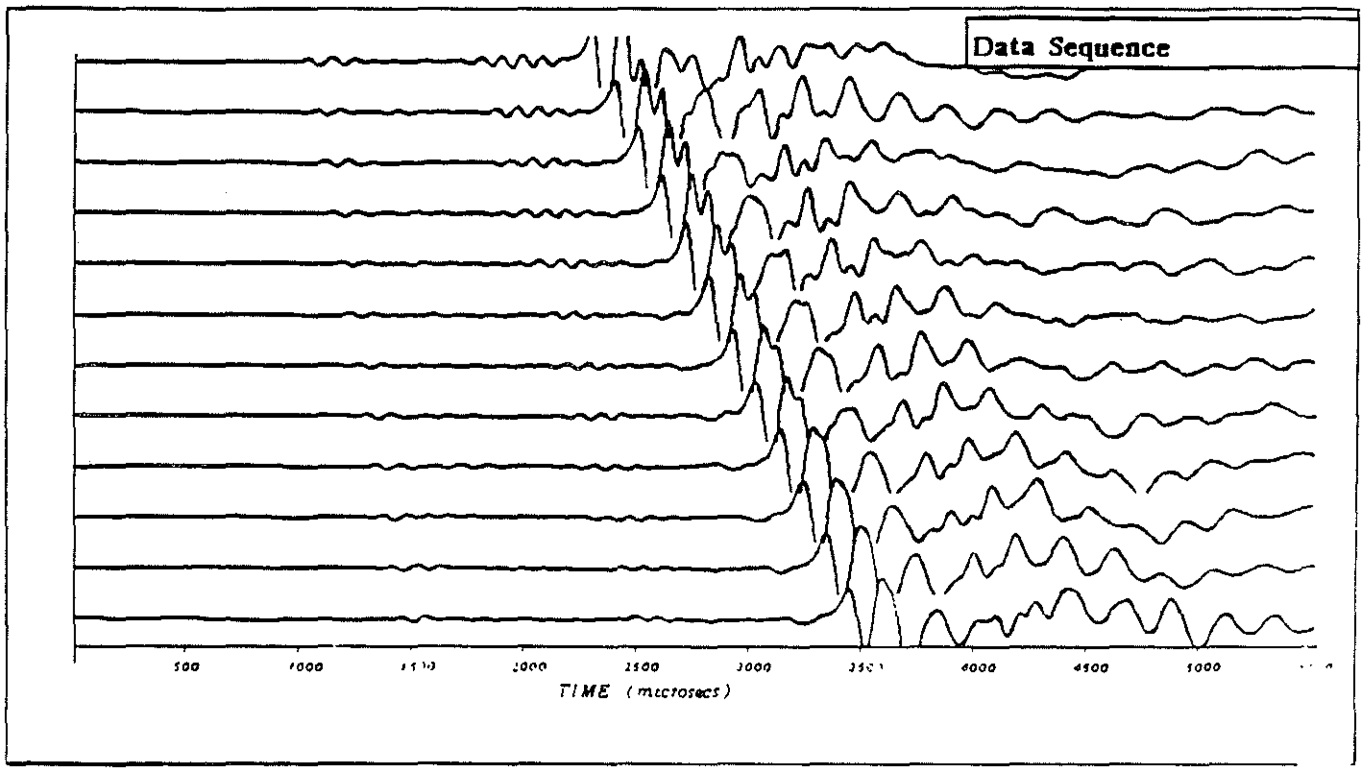


Figure 13: Raw array of full waveforms. The first receiver is 10 feet from the source and the distance between two successive traces is 0.5 foot.

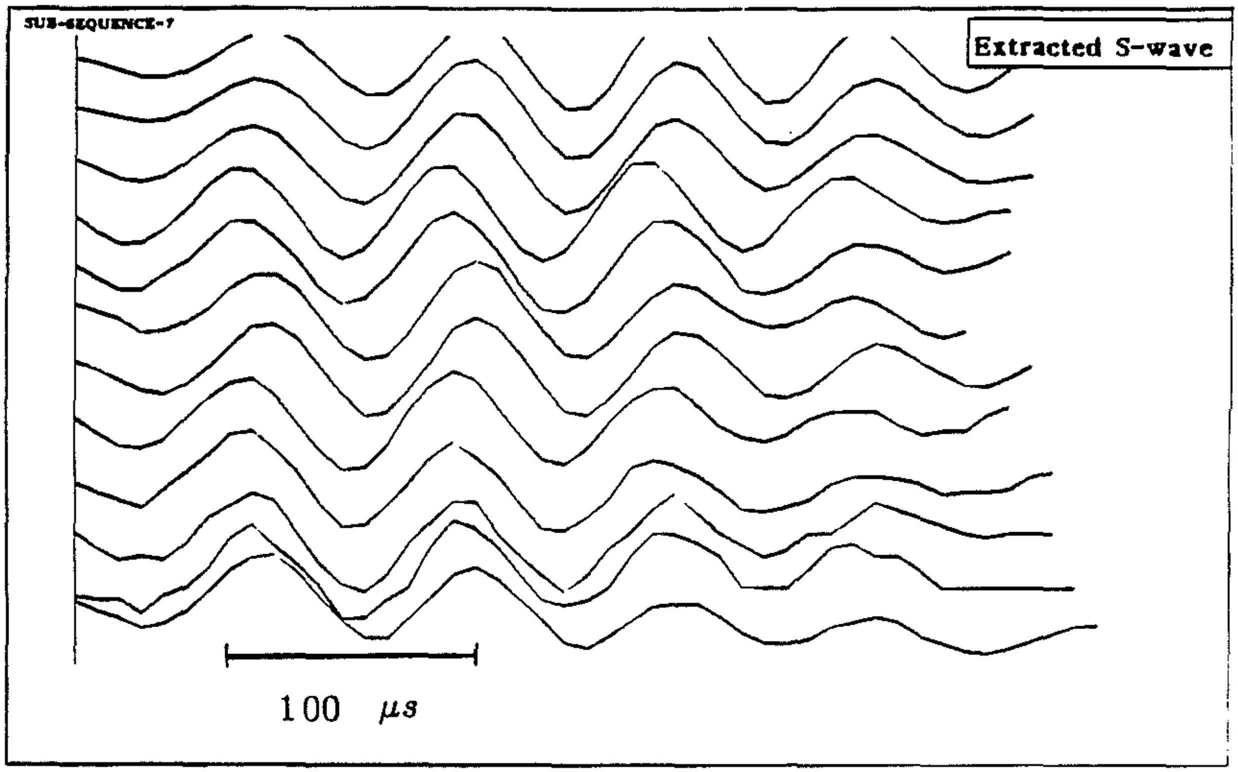


Figure 14: Extracted S (and pseudo-Rayleigh ?) wavetrains from the sequence presented on Figure 13.