USC-SIPI REPORT #168

Adaptive Minimum Prediction-Error Deconvolution and Source Wavelet Estimation Using Hopfield Neural Networks

by

Li-Xin Wang and Jerry M. Mendel

Signal and Image Processing Institute
UNIVERSITY OF SOUTHERN CALIFORNIA
Department of Electrical Engineering-Systems
3740 McClintock Avenue, Room 400
Los Angeles, CA 90089-2564 U.S.A.

ADAPTIVE MINIMUM PREDICTION-ERROR DECONVOLUTION AND SOURCE WAVELET ESTIMATION USING HOPFIELD NEURAL NETWORKS

Li-Xin Wang and Jerry M. Mendel

Abstract

In this report, three Hopfield neural networks are developed to realize a new Adaptive Minimum Prediction-Error Deconvolution procedure. The first neural network is developed to detect the reflectivity sequence. The second neural network is developed to determine the magnitudes of the detected reflections. The third neural network is developed to estimate the seismic wavelet. A Block-Component Method is proposed for simultaneous reflectivity estimation and wavelet extraction based on these three neural networks. These three neural networks and the Block-Component Method are simulated for broad-band and narrow-band wavelets. Real seismic data are processed using the Block-Component Method, and the results are compared with those using the MVD Filter and the maximum-likelihood based SMLR Detector [8,9].

Compared with existing deconvolution methods, the Neural Network Adaptive Minimum Prediction-Error Deconvolution method of this report has the following advantages: (1) it is totally realized by standard Hopfield neural networks which are suitable for hardware implementation; (2) the new Block-Component Method gives better results for real seismic data processing than the MLD-based Block Component Method; (3) the Neural Reflectivity Estimator gives much better results than the SMLR Detector in the case of a narrow-band wavelet and low signal-to-noise ratio; and, (4) it needs very weak assumptions about the wavelet and the reflectivity sequence, i.e., it is suitable for a nonminimum-phase wavelet, non-Gaussian or colored measurement noise, and, the reflectivity sequence can be random or deterministic.

1 INTRODUCTION

Seismic signal processing is very time-consuming and expensive. Some of the fastest computers all around the world run every day to process large volumes of seismic data, in order to determine where there is oil or natural gas. Developing some fast algorithms to speed up seismic data processing, especially some hardware-realizable methods, is therefore of great importance.

Recent research has shown that the massive parallel processing architectures and algorithms of artificial neural networks have many computational advantages over traditional computers. For example, the Hopfield neural network [5,6,12], which is suitable for analog VLSI and optical realizations [1,3,11], has proven to be a powerful tool to solve a wide variety of optimization problems [6,12]. In this report, we develop three Hopfield neural networks which are collectively used for seismic deconvolution or seismic wavelet extraction; we also develop a Neural Block-Component Method for simultaneous deconvolution and wavelet extraction. The criteria functions of these neural networks are adaptive versions of the prediction-error of the seismic trace; hence, we refer to our new method as "Adaptive Minimum Prediction-Error Deconvolution (AMPED)".

In Section 2, the AMPED method is formulated and compared with the Maximum-Likelihood Deconvolution (MLD) method [8,9]. In Section 3, the dynamics of the Hopfield neural network and its application to optimization problems are briefly reviewed. In Section 4, a Neural Reflectivity Estimator, which consists of two Hopfield neural networks, is developed for reflectivity detection and amplitude estimation. In Section 5, a Neural Wavelet Extractor, which consists of one Hopfield neural network, is developed for wavelet extraction. In Section 6, the Neural Reflectivity Estimator and the Neural Wavelet Extractor are combined to form a new Block-Component Method (BCM) for simultaneous reflectivity estimation and wavelet extraction. Simulations are provided throughout Sections 4-6 to demonstrate the performance of the corresponding neural networks and to compare the new approaches with the well-known MVD Filter and the maximum-likelihood based SMLR Detector [8,9]. In Section 7, the new BCM is applied to a 30-trace real seismic section, and, results are compared with those using the MVD Filter and SMLR Detector. Conclusions are given in Section 8.

2 ADAPTIVE MINIMUM PREDICTION-ERROR DECONVOLUTION

Consider the following convolutional model for seismic trace z_k,

$$z_k = \sum_{i=1}^{N} V_{k-i}\mu_i + n_k, \qquad k = 1, 2, ..., N \tag{1}$$

where V_k is the seismic source wavelet, with $V_k = 0$ for $k < 0$; μ_i is the earth's reflectivity sequence; n_k is the measurement noise; and, there are N data samples.

The cost function for Minimum Prediction-Error Deconvolution (MPED) is the following prediction-error of the seismic trace

$$E = \frac{1}{2}\sum_{k=1}^{N}[z_k - \sum_{i=1}^{N} V_{k-i}\mu_i]^2. \tag{2}$$

The *objective* of MPED is to obtain estimates of the reflectivity sequence and wavelet which minimize cost function E. The neural network approaches developed in later sections actually realize an *Adaptive* Minimum Prediction-Error Deconvolution (AMPED) procedure in which cost function (2) changes during the optimization procedure according to the magnitudes of the reflections which are to be detected.

MPED makes no assumptions about the wavelet, the reflectivity sequence, or the measurement noise, i.e., the wavelet can be non-minimum phase, the measurement noise can be any random process, Gaussian or non-Gaussian, white or colored, and the reflectivity sequence can be random or deterministic (hence, one is able to back away from the somewhat controversial random reflection model, if one so desires).

How are MPED and Maximum-Likelihood Deconvolution (MLD) [8,9] related ? In MLD, the reflectivity sequence μ_i is modeled as a Bernoulli-Gaussian sequence [8], i.e.,

$$\mu_i = q_i r_i, \tag{3}$$

where μ_i is a Bernoulli sequence with $Pr[q_i = 1] = \lambda$ and $Pr[q_i = 0] = 1 - \lambda$, and r_i is a zero-mean white Gaussian sequence with variance v_r. The log-likelihood function used in MLD is [9] (

concatenate Eq.(1) using Eq.(3), for $k = 1, 2, ..., N$, to obtain $\underline{z} = WQ\underline{r} + \underline{n}$)

$$\begin{aligned} L[\underline{\theta}, \underline{q}, \underline{r}, \underline{s}|\underline{z}] &= -\frac{N}{2} ln(v_r v_n) - \underline{r}^T \underline{r}/2v_r - (\underline{z} - WQ\underline{r})^T(\underline{z} - WQ\underline{r})/2v_n \\ &+ m(\underline{q}) ln(\lambda) + [N - m(\underline{q})] ln(1-\lambda) \end{aligned} \tag{4}$$

where $\underline{\theta}$ is the parameter vector for the wavelet (matrix W depends on $\underline{\theta}$); $\underline{q} = (q_1, q_2, ..., q_N)^T$ is the Bernoulli component of the reflectivity sequence (matrix Q depends on $\underline{q}$); $\underline{r} = (r_1, r_2, ..., r_N)^T$ is the Gaussian component of the reflectivity sequence; $\underline{s} = (v_r, v_n, \lambda)$ where v_n is the variance of the measurement noise; $\underline{z}$ is the vector of measurement data of length N; and, $m(\underline{q})$ is the number of unity components of $\underline{q}$.

If we don't model the reflectivity sequence as a Bernoulli-Gaussian process, but instead model it as a deterministic product sequence, (i.e., $\underline{q}$ and $\underline{r}$ are deterministic sequences where $\underline{q}$ indicates the locations of reflections and $\underline{r}$ represents the magnitudes of the reflections), then the log-likelihood function simplifies to

$$L[\underline{\theta}, \underline{q}, \underline{r}, v_n|\underline{z}] = -\frac{N}{2} ln(v_n) - (\underline{z} - WQ\underline{r})^T(\underline{z} - WQ\underline{r})/2v_n. \tag{5}$$

In this case $L[*]$ is a scaled version of the prediction-error. In fact, if v_n need not be estimated, then

$$\begin{aligned} E(\underline{\theta}, \underline{q}, \underline{r}|\underline{z}) &= -(L + \frac{N}{2} ln(v_n)) v_n \\ &= \frac{1}{2}(\underline{z} - WQ\underline{r})^T(\underline{z} - WQ\underline{r}) \\ &= Prediction - Error; \end{aligned} \tag{6}$$

hence, in the case of the non-random reflectivity model, the criterion functions of MPED and MLD are the same; however, due to the adaptive nature of AMPED, AMPED is different from MLD even for the deterministic reflectivity model.

In the case of a random reflectivity model, Eq.(4) is the correct likelihood function, and in this case the log-likelihood function is different from the prediction-error. Performance comparisons of the MLD-based methods and the AMPED-based methods are given in Sections 4, 6 and 7.

3 HOPFIELD NEURAL NETWORK AND ITS APPLICATION TO OPTIMIZATION PROBLEMS

A Hopfield neural network is a single-layer feedback neural network [5,6,12]. Let T_{ij} denote the connection weight from neuron j to neuron i, I_i denote the externally supplied input to neuron i, and q_i denote the output of neuron i, then the dynamics of a Hopfield neural network are

$$q_i(t+1) = \begin{array}{ll} 1 & if \quad D_i(t) > 0 \\ 0 & if \quad D_i(t) \leq 0 \end{array} \tag{7}$$

where $t = 0, 1, 2, ...$, $i = 1, 2, ..., N$, N is the total number of neurons, and

$$D_i(t) = \sum_{i \neq j = 1}^{N} T_{ij} q_j(t) + I_i. \tag{8}$$

This is the discrete-time version of the Hopfield neural network [6,12], i.e., it is the Hopfield neural network that has reached its stable state, and whose stable neuron outputs are either zero or one. Figure 1 shows a standard hardware diagram of a Hopfield neural network [5,6,12], in which the small solid squares denote the connection weights T_{ij}, and the amplifiers approximate the non-linear function in Eq.(7).

The Hopfield neural network has the following energy function

$$E = -\frac{1}{2}\sum_{i=1}^{N}\sum_{i\neq j=1}^{N} T_{ij}q_iq_j - \sum_{i=1}^{N} I_iq_i. \tag{9}$$

If we choose T_{ij} to be symmetric, then it can be shown that this energy function will never increase as the states of the neurons ($q_i's$) change. Specifically, let $\Delta E_i(t)$ denote the change of the energy function due to the change of q_i from $q_i(t-1)$ to $q_i(t)$ (we define $\Delta q_i(t) = q_i(t) - q_i(t-1)$); then, from Eqs.(9) and (8), we have

$$\begin{aligned}\Delta E_i(t) &= -(\sum_{i\neq j=1}^{N} T_{ij}q_j(t) + I_i)\Delta q_i(t) \\ &= -D_i(t)\Delta q_i(t).\end{aligned} \tag{10}$$

If $\Delta q_i(t) = q_i(t) - q_i(t-1) < 0$ which means $q_i(t-1) = 1$ and $q_i(t) = 0$, then from Eqs.(7) and (8) we have $D_i(t) \leq 0$, hence $\Delta E_i(t) = -D_i(t)\Delta q_i(t) \leq 0$; if $\Delta q_i(t) = q_i(t) - q_i(t-1) > 0$ which means $q_i(t-1) = 0$ and $q_i(t) = 1$, then from Eqs.(7) and (8) we have $D_i(t) > 0$, hence $\Delta E_i(t) = -D_i(t)\Delta q_i(t) < 0$. In conclusion, as the Hopfield neural network operates from iteration to iteration, we always have $\Delta E_i(t) \leq 0$ which means that the network will converge to a state at which the energy function E is locally minimized.

In Eqs.(8) and (9) we assume that $i \neq j$. This is equivalent to replacing $i \neq j = 1$ in Eqs.(8) and (9) by $j = 1$ but $T_{ii} = 0$, for $i = 1, 2, ..., N$. In fact, it was shown in [5,6,12] that if i equals j but $T_{ii} \neq 0$, we cannot always have $\Delta E_i(t) \leq 0$ as q_i changes; hence, in the following development of the neural networks for MPED, we will choose $T_{ii} = 0$.

One of the most important applications of a Hopfield neural network is to optimization problems [6,12]. The key step in these applications is to relate the cost function of an optimization problem to the energy function, Eq.(9), of the network. Since the energy function can be used to define the connection weights of, and the inputs to the network, relating a specific problem to a specific energy function provides the information for a detailed circuit diagram of the network. When the network reaches its stable state, for which the energy function is locally minimized, the output of the network gives the solution to the optimization problem.

The basic idea in this paper is to relate different versions of cost function (2) with energy function (9), to obtain estimates of the reflectivity sequence and/or the wavelet, depending on what we assume is known or unknown.

4 NEURAL REFLECTIVITY ESTIMATOR

A . Construction of the Network

In this report, we model the reflectivity sequence μ_i by a product model, i.e., Eq.(3), where q_i is a 0-1 sequence (which can be deterministic or random) that indicates the locations of the reflectivity

sequence; and, r_i is any sequence (deterministic or random) that represents the magnitude of the reflectivity sequence. We determine q_i and r_i separately, each by its own Hopfield neural network. First, we set r_i in Eq.(3) equal to a constant α, substitute $\mu_i = q_i\alpha$ into Eq.(2), and construct a Hopfield neural network for the resulting prediction-error function. This neural network gives us estimates of q_i which indicate the locations of the reflectivities that have a magnitude corresponding to α. Then, we set q_i in Eq.(3) equal to the just estimated q_i, substitute $\mu_i = q_i r_i$ into Eq.(2), and construct a Hopfield neural network for the corresponding prediction-error function. This neural network gives us estimates of r_i. After the reflections corresponding to α are detected and their magnitudes estimated, they are removed from the trace; then, this updated trace is used to form a new prediction-error which is used to detect the reflections corresponding to the next α. By varying the constant α, we can detect the locations and determine the magnitudes of all the reflections. The details are given below.

Setting r_i in Eq.(3) equal to α and substituting the result into Eq.(2), we have

$$\begin{aligned} E &= \frac{1}{2}\sum_{k=1}^{N}[z_k - \sum_{i=1}^{N} V_{k-i}q_i\alpha]^2 \\ &= \frac{\alpha^2}{2}\sum_{k=1}^{N}[z_k/\alpha - \sum_{i=1}^{N} V_{k-i}q_i]^2. \end{aligned} \tag{11}$$

Since α is a constant, minimizing E is equivalent to minimizing

$$\begin{aligned} E &= \frac{1}{2}\sum_{k=1}^{N}[z_k/\alpha - \sum_{i=1}^{N} V_{k-i}q_i]^2 \\ &= \frac{1}{2}\sum_{k=1}^{N}(\frac{z_k}{\alpha})^2 - \sum_{k=1}^{N}\sum_{i=1}^{N} V_{k-i}q_i z_k/\alpha + \frac{1}{2}\sum_{k=1}^{N}\sum_{i=1}^{N}\sum_{j=1}^{N} V_{k-j}V_{k-i}q_i q_j. \end{aligned} \tag{12}$$

Comparing Eqs.(12) and (9), we observe that the first term in Eq.(9) does not have T_{ii} terms, whereas the comparable term in Eq.(12) does have $i = j$ terms. Because the energy function in Eq.(9) is equivalent to

$$E = -\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} T_{ij}q_i q_j - \sum_{i=1}^{N} I_i q_i \tag{13}$$

where $T_{ii} \equiv 0$ for $i = 1, 2, ..., N$, the following term must be added to Eq.(12) in order to represent it in the form of Eq.(13):

$$-\frac{1}{2}\sum_{i=1}^{N}\sum_{k=1}^{N} V_{k-i}^2 q_i(q_i - 1). \tag{14}$$

Since $q_i = 0$ or 1, this additional term always equals zero; hence it has no influence on the value of E. Adding this term to Eq.(12), we obtain the energy function which is used to construct a Hopfield neural network for detecting the reflectivity sequence:

$$\begin{aligned} E &= \frac{1}{2}\sum_{k=1}^{N}[z_k/\alpha - \sum_{i=1}^{N} V_{k-i}q_i]^2 - \frac{1}{2}\sum_{i=1}^{N}\sum_{k=1}^{N} V_{k-i}^2 q_i(q_i - 1) \\ &= -\frac{1}{2}\sum_{i=1}^{N}\sum_{i\neq j=1}^{N}[-\sum_{k=1}^{N} V_{k-i}V_{k-j}]q_i q_j \\ &\quad -\sum_{i=1}^{N}[-\sum_{k=1}^{N} V_{k-i}^2/2 + \sum_{k=1}^{N} V_{k-i}z_k/\alpha]q_i + \frac{1}{2}\sum_{k=1}^{N} z_k^2/\alpha^2. \end{aligned} \tag{15}$$

Comparing Eq.(15) with Eq.(9) and ignoring the constant term $\frac{1}{2}\sum_{k=1}^{N} z_k^2/\alpha^2$, we obtain

$$T_{ij} = -\sum_{k=1}^{N} V_{k-i}V_{k-j} \tag{16}$$

$$I_i = \sum_{k=1}^{N} V_{k-i}z_k/\alpha - \frac{1}{2}\sum_{k=1}^{N} V_{k-i}^2 \tag{17}$$

for $i \neq j$, where $1 \leq i,j \leq N$; and, $T_{ii} = 0$ for $i = 1,2,...,N$.

Figure 2 depicts a hardware diagram for the neural network with connective weights given by Eq.(16) and inputs determined by Eq.(17). If $T_{ij} > 0$, positive amplifier j (denoted by small triangles) is connected to the resistor; otherwise, negative amplifier j (denoted by small triangles with a small circle) is connected. Since $V_k = 0$ for $k < 0$, input I_i in Eq.(17) can be rewritten as $I_i = V_0(z_i/\alpha) + V_1(z_{i+1}/\alpha) + \cdots + V_{N-i}(z_N/\alpha) - (1/2)\sum_{k=1}^{N} V_{k-i}^2$; Hence, if, as shown in Fig.2, we let z_k/α $(k = 1,2,...,N)$ be the external inputs to the circuit, then the solid squares in the figure will all be determined by wavelet V_k. For example, the top square in the column corresponding to q_2 has the value $(1/2)\sum_{k=1}^{N} V_{k-2}^2$ if we assume that external voltage V in Fig.2 is unity; the next (in top-down direction) $N-1$ squares have values $V_0, V_1, ..., V_{N-2}$ respectively; and, the remaining $N-1$ squares are determined by T_{2j} of Eq.(16).

The neural network with connective weights determined by Eq.(16) and inputs by Eq.(17) can be viewed as a detector of reflections which have a magnitude corresponding to α. The true magnitudes of these reflections do not equal α. We shall now develop another Hopfield neural network to determine the true magnitudes of the detected reflections.

Suppose a set of K_α reflections, $[q_i, i \in I_\alpha]$, has been detected for value α, where I_α is an index set with K_α elements (K_α can equal zero). Substituting Eq.(3) into Eq.(2) and noticing that the q_i's belonging to $[q_i, i \in I_\alpha]$ equal unity and all other q_i's equal zero, we have

$$\begin{aligned} E &= \frac{1}{2}\sum_{k=1}^{N}[z_k - \sum_{i=1}^{N} V_{k-i}q_i r_i]^2 \\ &= \frac{1}{2}\sum_{k=1}^{N}[z_k - \sum_{i\in I_\alpha} V_{k-i}r_i]^2 \end{aligned} \tag{18}$$

From the physical meaning of the reflectivity sequence, we have $|r_i| \leq 1$; hence, we can represent r_i, approximately, as

$$r_i = (\sum_{j=1}^{M} \frac{1}{2^{j-1}} p_{ij}) - 1 \quad i \in I_\alpha \tag{19}$$

where $p_{ij} = 0$ or 1. Viewing the p_{ij}'s $(i \in I_\alpha, j = 1,2,...,M)$ as outputs of neurons, the task here is to construct a Hopfield neural network with energy function Eq.(18) in which r_i is represented by Eq.(19). Substituting Eq.(19) into Eq.(18), we have

$$\begin{aligned} E &= \frac{1}{2}\sum_{k=1}^{N}[z_k + \sum_{i\in I_\alpha} V_{k-i} - \sum_{i\in I_\alpha}\sum_{j=1}^{M} \frac{1}{2^{j-1}} V_{k-i}p_{ij}]^2 \\ &= \frac{1}{2}\sum_{k=1}^{N}\sum_{i_1\in I_\alpha}\sum_{j_1=1}^{M}\sum_{i_2\in I_\alpha}\sum_{j_2=1}^{M} \frac{1}{2^{j_1-1}}\frac{1}{2^{j_2-1}} V_{k-i_1}V_{k-i_2}p_{i_1 j_1}p_{i_2 j_2} \end{aligned}$$

$$-\sum_{k=1}^{N}\sum_{i\in I_\alpha}\sum_{j=1}^{M}\frac{1}{2^{j-1}}V_{k-i}(z_k+\sum_{i\in I_\alpha}V_{k-i})p_{ij}+\frac{1}{2}\sum_{k=1}^{N}[z_k+\sum_{i\in I_\alpha}V_{k-i}]^2. \tag{20}$$

In order for the diagonal elements of the connection matrix to be zero, we add the following zero term to Eq.(20),

$$-\frac{1}{2}\sum_{k=1}^{N}\sum_{i\in I_\alpha}\sum_{j=1}^{M}\frac{1}{4^{j-1}}V_{k-i}^2p_{ij}(p_{ij}-1), \tag{21}$$

and obtain

$$\begin{aligned}E &= \frac{1}{2}\sum_{i_1\in I_\alpha}\sum_{j_1=1}^{M}\sum_{i_1\neq i_2\in I_\alpha}\sum_{j_1\neq j_2=1}^{M}\sum_{k=1}^{N}\frac{1}{2^{j_1-1}}\frac{1}{2^{j_2-1}}V_{k-i_1}V_{k-i_2}p_{i_1j_1}p_{i_2j_2}\\ &\quad-\sum_{i\in I_\alpha}\sum_{j=1}^{M}\sum_{k=1}^{N}[\frac{1}{2^{j-1}}V_{k-i}(z_k+\sum_{i\in I_\alpha}V_{k-i})-\frac{1}{2}\frac{1}{4^{j-1}}V_{k-i}^2]p_{ij}\\ &\quad+\ constant.\end{aligned} \tag{22}$$

Comparing Eq.(22) with Eq.(9) and ignoring the constant term, we obtain

$$T_{i_1j_1,i_2j_2}=-\sum_{k=1}^{N}\frac{1}{2^{j_1-1}}\frac{1}{2^{j_2-1}}V_{k-i_1}V_{k-i_2} \tag{23}$$

$$I_{ij}=\sum_{k=1}^{N}[\frac{1}{2^{j-1}}V_{k-i}(z_k+\sum_{i\in I_\alpha}V_{k-i})-\frac{1}{2}\frac{1}{4^{j-1}}V_{k-i}^2] \tag{24}$$

for $i_1\neq i_2$ and $j_1\neq j_2$, where $i_1,i_2\in I_\alpha, 1\leq j_1,j_2\leq M$; and, $T_{ij,ij}=0$ for $i\in I_\alpha, j=1,2,...,M$.

The number of neurons in this neural network is $K_\alpha M$. If the reflectivity sequence is sparse, very few reflections will be detected for a specific value of α so that K_α will be very small (however, our method is not limited to sparse reflectivity sequences). From Eq.(19), we see that in order for the r_i to have a representative error less than ϵ, M should satisfy $\frac{1}{2^{M-1}}<\epsilon$. For $\epsilon=0.01$, $M=8$ is sufficient; hence, if K_α is small, we need relatively few neurons to determine the amplitudes of the detected reflections.

Although we have used two indexes for neuron p_{ij}, p_{ij} can also be represented using only one index $s=(i-1)M+j$, where $i=1,2,...,K_\alpha$, $j=1,2,...,M$, so that $s=1,2,...,K_\alpha M$. Since the $K_\alpha M$ neurons are fully connected with each other, the network with connections (23) and inputs (24) is no different than the standard Hopfield neural network. In fact, we can represent $T_{i_1j_1,i_2j_2}$ in (23) by only two indexes $s_1=(i_1-1)M+j_1$ and $s_2=(i_2-1)M+j_2$, and I_{ij} in (24) by only one index $s=(i-1)M+j$, where $i_1,i_2,i\in I_\alpha$, $1\leq j_1,j_2,j\leq M$, and $1\leq s_1,s_2,s\leq K_\alpha M$.

The detailed steps of our Adaptive Minimum Prediction-Error (AMPE) Neural Reflectivity Estimator are (for a given wavelet):

- Step 1: Choose α to be a large positive value, and run the neural network with connective weights given by Eq.(16) and inputs given by Eq.(17) to obtain a set of K_α detected reflections $[q_i, i\in I_\alpha]$;

- Step 2: Run the neural network with connective weights determined by Eq.(23) and inputs determined by Eq.(24), and substitute the converged p_{ij} into Eq.(19) to obtain the magnitudes of the detected reflections;

- Step 3: Convolve the wavelet with the estimated reflectivity sequence corresponding to α (locations are detected in Step 1 and amplitudes are determined in Step 2), and *subtract this convolved sequence from the seismic trace to obtain an updated trace*;

- Step 4: Update α using $\alpha = \alpha - \epsilon_0$, where ϵ_0 is a small positive number, and repeat by going to Step 1 for the new α and updated seismic trace, until α equals a preset minimum α.

- Step 5: Choose α to be a large negative value (in the sense of absolute value) and choose ϵ_0 to be a small negative value (also in the sense of absolute value), repeat Steps 1 to 4 to detect and determine the magnitudes of negative reflections.

Because of Step 3, the cost functions for the neural networks change during the deconvolution procedure. When a reflection is detected and its magnitude estimated, it is removed from the trace, where "removed" (which is used through out this report) is in the sense of the operation of Step 3. In this way, the prediction-error is changing adaptively according to the magnitudes of the reflections. This is why we call this estimator an "AMPE Neural Reflectivity Estimator".

There is an alternative procedure to implement this AMPE Neural Reflectivity Estimator: we start from the maximum positive α, then switch to the maximum negative α, then switch back to the second largest positive α, then switch to the second largest negative α, etc., until we get to the minimum negative α. In this way, we first detect and estimate both large positive and large negative reflections. In the simulations and real data processing that are described below, we use this alternative procedure.

B . Simulations

We simulated the AMPE Neural Reflectivity Estimator for known broad-band [8] and narrow-band wavelet cases [2]. The true reflectivity sequence, which is shown in Fig.3, was chosen to be a Bernoulli-Gaussian sequence (Eq.(3)) with $\lambda = 0.08$ and $v_r = 0.08$. (If we just want to simulate the AMPE Neural Reflectivity Estimator, the reflectivity sequence need not be Bernoulli-Gaussian. Because we want to compare our new method with the MVD Filter and the maximum-likelihood based SMLR Detector, where a Bernoulli-Gaussian model of the reflectivity sequence is a prerequisite, we choose our simulated reflectivity sequence to be Bernoulli-Gaussian.) The broad-band and narrow-band wavelets are depicted in Figs.4 and 5, respectively. Convolving the reflectivity sequence in Fig.3 with the wavelets in Figs. 4 and 5, and adding white Gaussian noises to the results, we obtained the seismic traces shown in Figs. 6 and 7. The signal-to-noise ratio (SNR) was 4 in both cases, where

$$SNR = \sqrt{\frac{Pv_r}{v_n}} \tag{25}$$

in which

$$P = \sum_{k=1}^{N} V_k^2 \tag{26}$$

represents the energy of the wavelet, and v_n is the variance of the measurement noise.

Applying the AMPE Neural Reflectivity Estimator to the two traces of Figs.6 and 7, we obtain the estimated reflectivity sequences shown in Figs.8 and 11, respectively. The initial q_i's for Step 1 of the AMPE Neural Reflectivity Estimator were chosen to be zero. The initial p_{ij}'s in Step 2 of the AMPE Neural Reflectivity Estimator were chosen in such a way that the corresponding initial r_i's (Eq.(19)) were zero (i.e., $p_{i1} = 1, p_{ij} = 0$ for $j = 2, 3, ..., M$). We chose $M = 8$. The initial α was chosen to be 0.42, and the initial negative α was chosen to be -0.42; the $|\epsilon_0|$ was chosen to be 0.02; and, the minimum positive (negative) α was chosen to be 0.06 (-0.06). We chose the initial positive and negative α's to be about twice the magnitudes of the maximum true positive and negative reflections, respectively. The two neural networks for location detection and amplitude estimation each converged in 1 to 4 iterations, where "one iteration" indicates that the outputs of neurons change from $q_i(t)$ to $q_i(t+1)$ according to Eqs.(7) and (8) for all $i = 1, 2, ..., N$; and, "convergence" means that the outputs of neurons did not change from one iteration to the next.

In order to compare the new AMPE Neural Reflectivity Estimator with some existing methods, we deconvolved the traces of Figs.6 and 7 using a pure MVD Filter and an SMLR Detector [8,9]. The purpose for comparing the new detector with pure MVD is to see what improvements are obtained via detection, whereas the purpose for comparing the new detector with the SMLR detector is to compare the new detector with a well-established statistical detector. Estimates of the reflectivity sequence using a pure MVD Filter are shown in Figs.9 and 12, for the traces of Figs.6 and 7, respectively, whereas stimates of the reflectivity sequence using the SMLR Detector are shown in Figs.10 and 13, for the traces of Figs.6 and 7, respectively.

Comparing Fig.8 with Figs.9 and 10 we see that for the broad-band case the new estimator gives comparable results to both the MVD Filter and SMLR Detector.

Comparing Fig.11 with Figs.12 and 13 we see that our new AMPE Neural Reflectivity Estimator gives much better results than the MVD Filter and SMLR Detector for the narrow-band wavelet and low SNR case. We also simulated the AMPE Neural Reflectivity Estimator and the SMLR Detector for four other SNR cases using the same narrow-band wavelet. Figure 14 shows the results. Except for very high SNR's, the AMPE Neural Reflectivity Estimator outperformed the SMLR Detector. We conjecture that the relatively invariant performance of the AMPE Neural Reflectivity Estimator as SNR varies is due to its adaptive objective function.

Since the cost functions for the two neural networks in the AMPE Neural Reflectivity Estimator change adaptively according to α, it is interesting to show the estimated reflections for some specific α's. It is even more interesting to show the entire procedure, i.e., starting with the original trace, show: the first estimated largest reflections; then the updated trace obtained by removing these first-estimated large reflections; then the estimated reflections obtained for this updated trace; then the next updated trace; etc., until the estimated reflections for the minimum α are removed from the trace. This last trace is considered to be noise. This entire procedure is shown in Fig.15 for the broad-band trace of Fig.6. We show only the results for those α's for which reflections were detected (for many α's, no reflections were detected).

It is interesting to observe the reflection at the time point 119. When it was first detected, its amplitude estimate was too large (see the right-hand figure of DATA 1). As a result, we removed too much from the trace corresponding to this positive reflection; however, as the adaptive optimization proceeded, we later obtained a negative reflection at the same time point (see the right-hand figure of DATA 7) which offset the overly large positive estimate obtained earlier. This

result suggests that even if we make some mistakes in the early stages, our method can still give correct final estimates by providing corrective estimates in later stages.

C . Discussions

From our simulation results we draw the following conclusions: (1) the AMPE Neural Reflectivity Estimator gives much better results than the MVD Filter and the SMLR Detector in the case of a narrow-band wavelet and low SNR (Figs.11-14); (2) the AMPE Neural Reflectivity Estimator is relatively insensitive to SNR, i.e., it gives almost the same results for a wide range of SNR cases for which the SMLR Detector shows very different performance (Fig.14); and, (3) for the broad-band wavelet case, the performances of the AMPE Neural Reflectivity Estimator and the SMLR Detector are quite similar (Figs.8 and 10).

A significant advantage of the new AMPE Neural Reflectivity Estimator is that it gives good results in the case of a narrow-band wavelet and low SNR. The real seismic wavelet is usually narrow-band, and real seismic data is usually corrupted by a high level of noise, especially for data coming from great depths; hence, this advantage makes the AMPE Neural Reflectivity Estimator a very attractive method for real data processing. See Section 7 for real data processing results.

Why does this new method have such an advantage ? A theoretical analysis is under investigation. A qualitative explanation is given next.

Our new method does not optimize a single criterion function, i.e., for different magnitude reflections we have different criterion functions. Perhaps these different criterion functions are individually suited for detecting the reflections of different magnitudes; thus, better performance is obtained. In the MVD Filter, SMLR Detector, and most commonly used deconvolution methods, the associated criterion function is the same for processing the entire trace, i.e., large reflections and small reflections are treated the same. This single criterion function may not be suitable for detecting some reflections. In other words, a single criterion function provides a compromised measure for all the reflections (large and small), whereas our adaptive criterion function method does not.

5 NEURAL WAVELET EXTRACTOR

A . Construction of the Network

Suppose $V_0 = 0$ and $\mu_i = 0$ for $i \leq 0$, then Eq.(2) can be rewritten as

$$E = \frac{1}{2}\sum_{k=1}^{N}[z_k - \sum_{i=1}^{N}\mu_{k-i}V_i]^2 \tag{27}$$

Normalizing V_i to satisfy $|V_i| \leq 1$, then V_i can be expressed, approximately, as (similar to Eq.(19))

$$V_i = (\sum_{j=1}^{M}\frac{1}{2^{j-1}}x_{ij}) - 1 \tag{28}$$

where x_{ij}=0 or 1. Substituting Eq.(28) into Eq.(27), we have

$$E = \frac{1}{2}\sum_{k=1}^{N}[z_k - \sum_{i=1}^{N}\mu_{k-i}\sum_{j=1}^{M}\frac{1}{2^{j-1}}x_{ij} + \sum_{i=1}^{N}\mu_{k-i}]^2$$

$$= \frac{1}{2}\sum_{k=1}^{N}\sum_{i=1}^{N}\sum_{j=1}^{M}\sum_{p=1}^{N}\sum_{q=1}^{M}\frac{1}{2^{j-1}}\frac{1}{2^{q-1}}\mu_{k-i}\mu_{k-p}x_{ij}x_{pq}$$
$$-\sum_{k=1}^{N}\sum_{i=1}^{N}\sum_{j=1}^{M}\frac{1}{2^{j-1}}\mu_{k-i}y_k x_{ij} + \frac{1}{2}\sum_{k=1}^{N}y_k^2 \tag{29}$$

where

$$y_k = z_k + \sum_{i=1}^{N}\mu_{k-i}. \tag{30}$$

In the development of this section's Neural Wavelet Extractor, we assume that μ_i is given. Now view x_{ij} $(i = 1, 2, ..., N; j = 1, 2, ..., M)$ as outputs of neurons. The purpose here is to construct a Hopfield neural network, where energy function, E, is given by Eq.(29). Similar to the neural networks developed in the last section, we add the following term to Eq.(29) for the purpose of conceling the diagonal elements of the connection matrix:

$$-\frac{1}{2}\sum_{k=1}^{N}\sum_{i=1}^{N}\sum_{j=1}^{M}\frac{1}{4^{j-1}}\mu_{k-i}^2 x_{ij}(x_{ij}-1). \tag{31}$$

This term always equals zero because $x_{ij} = 0$ or 1. Adding this term to Eq.(29) and dropping the constant term $\frac{1}{2}\sum_{k=1}^{N}y_k^2$, we have the energy function for constructing the Neural Wavelet Extractor:

$$\begin{aligned} E &= \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{M}\sum_{i\neq p=1}^{N}\sum_{j\neq q=1}^{M}\sum_{k=1}^{N}\frac{1}{2^{j-1}}\frac{1}{2^{q-1}}\mu_{k-i}\mu_{k-p}x_{ij}x_{pq} \\ &\quad -\sum_{i=1}^{N}\sum_{j=1}^{M}[\sum_{k=1}^{N}\frac{1}{2^{j-1}}\mu_{k-i}y_k - \frac{1}{2}\sum_{k=1}^{N}\frac{1}{4^{j-1}}\mu_{k-i}^2]x_{ij} \\ &= -\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{M}\sum_{p=1}^{N}\sum_{q=1}^{M}T_{ij,pq}x_{ij}x_{pq} - \sum_{i=1}^{N}\sum_{j=1}^{M}I_{ij}x_{ij} \end{aligned} \tag{32}$$

where

$$T_{ij,pq} = -\sum_{k=1}^{N}\frac{1}{2^{j+q-2}}\mu_{k-i}\mu_{k-p} \tag{33}$$

$$I_{ij} = \sum_{k=1}^{N}[\frac{1}{2^{j-1}}\mu_{k-i}y_k - \frac{1}{2}\frac{1}{4^{j-1}}\mu_{k-i}^2] \tag{34}$$

for $i \neq p$ and $j \neq q$, in which $1 \leq i, p \leq N, 1 \leq j, q \leq M$. Additionally, $T_{ij,ij} = 0$ for $i = 1, 2, ..., N, j = 1, 2, ..., M$.

This Neural Wavelet Extractor has NM neurons, where M neurons determine the value of wavelet V_k at one time point, and there are N time points. Usually $V_k \approx 0$ for $k > L$, where $L \ll N$; hence, this network can be simplified to have LM neurons, as described next.

Simplified Neural Wavelet Extractor

If $V_k = 0$ for $k > L$, then the Neural Wavelet Extractor (with connective weights and inputs given by Eqs.(33) and (34), respectively) can be simplified to have only LM neurons: the connective

weights are still given by Eq.(33), but the inputs are changed to

$$I_{ij} = \sum_{k=1}^{N} [\frac{1}{2^{j-1}} \mu_{k-i} y_k - \frac{1}{2} \frac{1}{4^{j-1}} \mu_{k-i}^2] + \sum_{p=L+1}^{N} T_{ij,p1} \tag{35}$$

where $1 \leq i, p \leq L, 1 \leq j, q \leq M$.

The derivation of the Simplified Neural Wavelet Extractor is given in Appendix A.

One fundamental difference between our Neural Wavelet Extractor (including the Simplified Neural Wavelet Extractor) and the neural networks developed in the last section is the choice of initial states of the neurons. The initial states of the neurons of the AMPE Neural Reflectivity Estimator are chosen in such a way that the corresponding q_i and r_i are zero; however, the initial states of the MPE Neural Wavelet Extractor cannot be so easily chosen. In fact, unless the initial states of the Neural Wavelet Extractor are chosen in such a way that the corresponding initial V_i (Eq.(28)) is close to the true wavelet value, the Neural Wavelet Extractor may converge to incorrect wavelets. In this paper, we chose these initial states by using an initial guess wavelet, provided by the Iterative Least-Squares Wavelet Extraction Algorithm given in Appendix B. This Iterative Least-Squares Wavelet Extraction Algorithm gives more accurate initial wavelet estimates than the "minimum-eigenvalue estimator" used in MLD [8,9].

B . Simulations

We simulated the Simplified Neural Wavelet Extractor for the two synthetic seismic traces of Figs. 6 and 7. In the simulations, we chose $M = 8$ and $L = 50$. The results are shown in Figs. 16 and 17 for the traces of Figs. 6 and 7, respectively. Convergence occurred after three iterations for both cases, where "convergence" means that the neuron states did not change from iteration to iteration.

C . Discussions

The Simplified Neural Wavelet Extractor is very effective, as seen from its fast convergence and relatively accurate estimates. Usually, it is sufficient to choose M=8 (so that the approximation error is less than 2^{-7} which is less than one percent of $maxV_k$) and $L = 50$ (if the sample interval is 4 ms, L=50 corresponds to 200 ms, after which the wavelet usually decays to zero); hence, this Simplified Neural Wavelet Extractor needs only about 400 neurons.

A disadvantage of this Simplified Neural Wavelet Extractor is that it needs a relatively accurate initial guess wavelet. We simulated it with some less accurate initial guess wavelets, and the final estimated wavelets were incorrect. For example, we simulated it with the initial guess wavelet provided by the "minimum-eigenvalue estimator" used in MLD [8,9], and got an incorrect final wavelet estimate. From these simulations, it seems that this new AMPE Simplified Neural Wavelet Extractor is not as robust as MLD is to the choice of the initial guess wavelet. This is not a serious problem in practice, because for real seismic data we often have some knowledge about the wavelet, based either on our experiences from previous processings or from using other methods applied to similar data. This knowledge can provide us with a relatively accurate initial guess wavelet. Of course, we now have a new advanced wavelet estimator (the Iterative Least-Squares Wavelet Extraction Algorithm) which can be used to provide an accurate initial guess wavelet.

6 BLOCK-COMPONENT METHOD FOR SIMULTANEOUS WAVELET EXTRACTION AND REFLECTIVITY ESTIMATION

We now have an AMPE Neural Reflectivity Estimator, which estimates the reflectivity sequence under the condition that the wavelet is known, and, a MPE Simplified Neural Wavelet Extractor, which estimates the wavelet under the condition that the reflectivity sequence is known. It is natural to combine these neural networks (i.e., to view them as computing blocks) to develop a Block-Component Method (BCM) for simultaneous wavelet extraction and reflectivity estimation. This BCM is shown in Fig.18. In this BCM, convergence is accepted when the outputs of the Neural Reflectivity Estimator and the Neural Wavelet Extractor do not change from iteration to iteration.

We simulated the BCM for the traces of Figs.6 and 7. The final estimates of the reflectivity sequence are shown in Figs.19 and 20 for the traces of Figs.6 and 7, respectively. The final estimates of the wavelets are shown in Figs.21 and 22 for the traces of Figs.6 and 7, respectively. In these simulations, the: initial q_i's for Step 1 of the Neural Reflectivity Estimator were chosen to be zero; initial p_{ij}'s in Step 2 of the Neural Reflectivity Estimator were chosen in such a way that the corresponding initial r_i's (Eq.(19)) were zero; initial α was chosen to be 0.42, and initial negative α was chosen to be -0.42; $|\epsilon_0|$ was chosen to be 0.02; minimum positive (negative) α was chosen to be 0.06 (-0.06); and, initial states of the neurons of the MPE Neural Wavelet Extractor were chosen to match the initial wavelet estimate obtained from the Iterative Least-Squares Algorithm in Appendix B.

From these simulation results we see that the BCM gave slightly worse results than the pure AMPE Neural Reflectivity Estimator and the pure Simplified Neural Wavelet Extractor (compare Figs.19 and 8, 20 and 11, 21 and 16, and 22 and 17), but that was to be expected, because in the present case *both* the wavelet and reflectivity sequence are unknown.

7 REAL SEISMIC DATA PROCESSING

In this section we apply the new BCM to a section of real seismic data. Figure 23 shows the input section (prestacked data), the estimated reflectivity, and, the estimated wavelets. In this processing, we chose the: initial positive α and initial negative α to be large enough (in the sense of absolute value) such that no reflections would be detected for such initial α; $\epsilon_0 = 0.02$; and, $\alpha_{min} = 0.03$ for positive reflections, and $\alpha_{min} = -0.03$ for negative reflections. For the 30 traces in the input section, we used the Iterative Least-Squares Algorithm only for one trace (the fifth trace down from the top trace) to get the initial wavelet estimate; the initial wavelets for all other traces were then chosen to be the final converged wavelet of this trace.

In real data processing, it is important to know how to handle backscatter [9]. Our AMPE Neural Reflectivity Estimator can deal with backscatter in a very effective way, by choosing different α_{min}. If we choose α_{min} to be large, only larger reflections will be detected; hence, the backscatter and small reflections are filtered out. If we choose α_{min} to be small, smaller reflections will be detected in which some are considered to be backscatter. Consequently, if we only want to see larger reflections, we choose α_{min} to be larger; and, if we want to see smaller reflections as well as larger reflections, we choose α_{min} to be smaller. Figures 24 and 25 show the results for the same input section but for different α_{min}'s. Also shown are results obtained from the MVD Filter and

SMLR Detector using the extracted wavelets determined from our BCM for the $\alpha_{min} = 0.03$ case (the results using the extracted wavelets of other α_{min} cases were almost the same). Both the SMLR and neural BCM results are of higher resolution than the MVD results. Observe also that as α_{min} is chosen to be smaller and smaller, more and more reflections are indeed detected. All the neural BCM results depicted in Figs.24 and 25 were obtained in one processing of the data, i.e., lowering α_{min} does not alter the results for larger α_{min}'s; it just adds more reflections.

8 CONCLUSIONS

In this report, three Hopfield neural networks were developed for seismic reflectivity detection, reflectivity magnitude estimation, and wavelet extraction, respectively. These neural networks were combined into a Block-Component Method to realize an Adaptive Minimum Prediction-Error Deconvolution procedure which performs reflectivity estimation and wavelet extraction simultaneously. Simulations were performed for broad-band and narrow-band wavelets and different SNR cases; and, real seismic data was also processed. All results were quite good.

The neural network approaches of this report: (1) are directly suitable for VLSI or optical hardware realizations, since the basic computing element – the Hopfield Neural Network, is suitable for VLSI or optical realization [1,3,11]; (2) show that the Neural Reflectivity Estimator gives much better results than the MVD Filter and even the high resolution SMLR Detector in the case of a narrow-band wavelet and low SNR, which is often the case in real seismic data; (3) require very weak modeling assumptions, i.e., the wavelet can be non-minimum phase and we do not need a parametric model, the measurement noise can be non-Gaussian and colored, and, the reflectivity sequence can be random or deterministic; and, (4) show that backscatter can be treated in a very effective way.

9 ACKNOWLEDGMENT

The work reported herein was performed at the University of Southern California, and was supported by USC Faculty Research and Innovation Grant 22-1503-9336, 1989-1990. The real seismic data were provided to us by UNOCAL Research Division, Brea, CA.

10 REFERENCES

[1] Alspector,J., "Neural-Style Microsystem that Learn," *IEEE Communications Magazine*, Vol.27, No.11, pp.29-36, 1989.

[2] Chi,C.Y. and J.M.Mendel, "Performance of Minimum-Variance Deconvolution Filter," *IEEE Trans. on ASSP*, Vol.32, No.6, 1984.

[3] Farhat,N.H., D.Psaltis, A.Prata and E.Paek, "Optical Implementation of the Hopfield Model," *Applied Optics*, 24, pp.1469-1475, 1985.

[4] Goodwin,G.C. and R.L.Payne, *Dynamic System Identification*, Academic Press, 1977.

[5] Hopfield,J.J, "Neurons with Graded Response Have Collective Computation Properties Like Those of Two-State Neurons," *Proc. National Academy of Sciences*, pp.3088-3092, 1984.

[6] Hopfield,J.J. and D.W.Tank, "Neural Computation of Decisions in Optimization Problems," *Biological Cybernetics*, Vol.52, pp.141-152, 1985.

[7] Giannakis,G.B., J.M.Mendel and X.F.Zhao, "A Fast Prediction-Error Detector for Estimating Sparse-Spike Sequences," *IEEE Trans. on Geoscience and Remote Sensing*, Vol.27, No.3, pp.344-351, 1989.

[8] Mendel,J.M., *Optimal Seismic Deconvolution: An Estimation-Based Approach*, Academic Press, 1983.

[9] Mendel,J.M., *Maximum-Likelihood Deconvolution*, Springer-Verlag New York Inc., 1990.

[10] Marple,S.L., *Digital Spectral Analysis*, Prentice-Hall, Inc., 1987.

[11] Sivilotti,M.A., M.A.Mahowald and C.A.Mead, "Real-time Visual Computations Using Analog CMOS Processing Arrays," *Advanced Research in VLSI: Proc. of the 1987 Stanford Conference*, P.Losleben (Ed.), Combridge, MA: MIT Press, pp.295-312, 1987.

[12] Tank,D.W. and J.J.Hopfield, "Simple 'Neural' Optimization Networks: An A/D Converter, Signal Decision Circuit, and a Linear Programming Circuit," *IEEE Trans. on Circuits and Systems*, 33(5), pp.533-541, 1986.

A APPENDIX A

Assume $V_k = 0$ for $k > L$; then, from Eq.(28), we have $x_{i1} = 1$ and $x_{ij} = 0$ for $i > L, j = 1, 2, ..., M$. Using this fact, Eq.(32) can be rewritten as (our purpose is to simplify the argument p from $1 \leq p \leq N$ to $1 \leq p \leq L$)

$$\begin{aligned} E &= -\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{M}[\sum_{p=1}^{L}\sum_{q=1}^{M} T_{ij,pq}x_{pq} + \sum_{p=L+1}^{N}\sum_{q=1}^{M} T_{ij,pq}x_{pq} + 2I_{ij}]x_{ij} \\ &= -\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{M}[\sum_{p=1}^{L}\sum_{q=1}^{M} T_{ij,pq}x_{pq} + \sum_{p=L+1}^{N} T_{ij,p1} + 2I_{ij}]x_{ij} \\ &= -\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{M}[\sum_{p=1}^{L}\sum_{q=1}^{M} T_{ij,pq}x_{pq} + 2I'_{ij}]x_{ij} \end{aligned} \tag{A.1}$$

where

$$I'_{ij} = I_{ij} + \frac{1}{2}\sum_{p=L+1}^{N} T_{ij,p1}. \tag{A.2}$$

Using the fact in the first sentence of this paragraph, Eq.(A.1) can be further rewritten as (now our purpose is to simplify the argument i from $1 \leq i \leq N$ to $1 \leq i \leq L$)

$$\begin{aligned} E &= -\frac{1}{2}\sum_{i=1}^{L}\sum_{j=1}^{M}[\sum_{p=1}^{L}\sum_{q=1}^{M} T_{ij,pq}x_{pq} + 2I'_{ij}]x_{ij} \\ &\quad -\frac{1}{2}\sum_{i=L+1}^{N}\sum_{j=1}^{M}[\sum_{p=1}^{L}\sum_{q=1}^{M} T_{ij,pq}x_{pq} + 2I'_{ij}]x_{ij} \\ &= -\frac{1}{2}\sum_{i=1}^{L}\sum_{j=1}^{M}[\sum_{p=1}^{L}\sum_{q=1}^{M} T_{ij,pq}x_{pq} + 2I'_{ij}]x_{ij} \\ &\quad -\frac{1}{2}\sum_{p=1}^{L}\sum_{q=1}^{M}[\sum_{i=L+1}^{N} T_{i1,pq}]x_{pq} - \sum_{i=L+1}^{N} I'_{i1}. \end{aligned} \tag{A.3}$$

Since $\sum_{i=L+1}^{N} I'_{i1}$ is a constant, it can be dropped from Eq.(A.3). Define

$$I''_{ij} = I'_{ij} + \frac{1}{2}\sum_{m=L+1}^{N} T_{m1,ij} \tag{A.4}$$

then we have, from Eq.(A.3), that

$$E = -\frac{1}{2}\sum_{i=1}^{L}\sum_{j=1}^{M}\sum_{p=1}^{L}\sum_{q=1}^{M} T_{ij,pq}x_{ij}x_{pq} - \sum_{i=1}^{L}\sum_{j=1}^{M} I''_{ij}x_{ij}. \tag{A.5}$$

From Eqs.(A.4), (A.2), and (34), and the fact that $T_{ij,p1} = T_{p1,ij}$, we have

$$\begin{aligned} I''_{ij} &= I_{ij} + \sum_{p=L+1}^{N} T_{ij,p1} \\ &= \sum_{k=1}^{N}[\frac{1}{2^{j-1}}\mu_{k-i}y_k - \frac{1}{2}\frac{1}{4^{j-1}}\mu_{k-i}^2] + \sum_{p=L+1}^{N} T_{ij,p1}. \end{aligned} \tag{A.6}$$

Equation (A.5) is the energy function of the Simplified Neural Wavelet Extractor, with $T_{ij,pq}$ given by Eq.(33) and I''_{ij} given by Eq.(A.6).

B APPENDIX B : An Iterative Least-Squares Algorithm for Initial Wavelet Estimation

We begin by modeling the wavelet by an ARMA model, whose transfer function is

$$V(z) = \frac{1 - \sum_{i=1}^{n}\beta_i z^{-i}}{1 - \sum_{i=1}^{n}\alpha_i z^{-i}} \tag{B.1}$$

where we normalize the wavelet so that $V_0 = 1$. From Eqs.(1) and (B.1), ignoring the measurement noise n_k, we have

$$z_k = \sum_{i=1}^{n}\alpha_i z_{k-i} - \sum_{i=1}^{n}\beta_i\mu_{k-i} + \mu_k. \tag{B.2}$$

Assume

$$f_k(\underline{\theta}) = \sum_{i=1}^{n}\alpha_i z_{k-i} - \sum_{i=1}^{n}\beta_i\mu_{k-i} \tag{B.3}$$

where

$$\underline{\theta} = [\alpha_1, \cdots, \alpha_n, \beta_1, \cdots, \beta_n]^T \tag{B.4}$$

denotes the ARMA parameters we want to estimate. Let

$$\tilde{z_k} = z_k - f_k(\underline{\theta}), \tag{B.5}$$

then we obtain the ARMA parameter estimates by minimizing the sum of all the squared error terms, i.e., we minimize

$$S(\underline{\theta}) = \sum_{k=1}^{N}\tilde{z_k}^2 \tag{B.6}$$

with respect to $\underline{\theta}$.

We use the following Levenberg-Marquardt [4] iterative algorithm to solve this non-linear optimization problem

$$\hat{\underline{\theta}}^{i+1} = \hat{\underline{\theta}}^{i} - (X_i^T X_i + p_i I)^{-1} X_i^T \tilde{\underline{Z}}_i \tag{B.7}$$

where $\hat{\underline{\theta}}^{i}$ denotes the i'th estimate of $\underline{\theta}$; p_i is an acceleration constant;

$$\tilde{\underline{Z}}_i = [z_1 - f_1(\hat{\underline{\theta}}^{i}), \cdots, z_N - f_N(\hat{\underline{\theta}}^{i})]^T; \tag{B.8}$$

and

$$X_i = \left[\begin{array}{ccc} \frac{\partial f_1}{\partial \theta_1} & \cdots & \frac{\partial f_1}{\partial \theta_{2n}} \\ \cdots & \cdots & \cdots \\ \frac{\partial f_N}{\partial \theta_1} & \cdots & \frac{\partial f_N}{\partial \theta_{2n}} \end{array} \right]_{\underline{\theta}=\hat{\underline{\theta}}^{i}} \tag{B.9}$$

in which θ_j denotes the j'th component of $\underline{\theta}$. Next, we see how to calculate $\tilde{\underline{Z}}_i$ and X_i, and how to choose $\hat{\underline{\theta}}^{0}$.

First, we see how to obtain $\tilde{\underline{Z}}_i$. From Eqs.(B.2) and (B.3), we have

$$\mu_k = z_k - f_k(\underline{\theta}). \tag{B.10}$$

Substitute Eq.(B.10) into (B.3) to see that

$$\begin{aligned} f_k(\underline{\theta}) &= \sum_{i=1}^{n} \alpha_i z_{k-i} - \sum_{i=1}^{n} \beta_i [z_{k-i} - f_{k-i}(\underline{\theta})] \\ &= \sum_{i=1}^{n} (\alpha_i - \beta_i) z_{k-i} + \sum_{i=1}^{n} \beta_i f_{k-i}(\underline{\theta}). \end{aligned} \tag{B.11}$$

This is a recursive formula for $f_k(\underline{\theta})$. In order to use this recursive formula to calculate $f_k(\underline{\theta})$ $(k = 1, 2, ..., N)$, we need to know the initial values of $f_k(\underline{\theta})$: $f_0(\underline{\theta}), f_{-1}(\underline{\theta}), \cdots, f_{-n+1}(\underline{\theta})$. Since $z_k = 0$ and $\mu_k = 0$ for $k < 0$, we have, from Eq.(B.3), that

$$f_0(\underline{\theta}) = f_{-1}(\underline{\theta}) = \cdots = f_{-n+1}(\underline{\theta}) = 0; \tag{B.12}$$

hence, $\tilde{\underline{Z}}_i$ (Eq.(B.8)) can be obtained from Eq.(B.11) with initial conditions in Eq.(B.12).

Next, we see how to obtain X_i. The elements of X_i are $\frac{\partial f_k}{\partial \alpha_i}$ and $\frac{\partial f_k}{\partial \beta_i}$ where $k = 1, 2, ..., N$ and $i = 1, 2, ..., n$. From Eq.(B.3), we have

$$\frac{\partial f_k}{\partial \alpha_i} = z_{k-i} - \sum_{j=1}^{n} \beta_j \frac{\partial \mu_{k-j}}{\partial \alpha_i}, \tag{B.13}$$

$$\frac{\partial f_k}{\partial \beta_i} = -\mu_{k-i} - \sum_{j=1}^{n} \beta_j \frac{\partial \mu_{k-j}}{\partial \beta_i}. \tag{B.14}$$

From Eq.(B.10), we have

$$\frac{\partial \mu_k}{\partial \theta_i} = -\frac{\partial f_k(\underline{\theta})}{\partial \theta_i}. \tag{B.15}$$

Substituting Eq.(B.15) into Eqs.(B.13) and (B.14), we have

$$\frac{\partial f_k}{\partial \alpha_i} = z_{k-i} + \sum_{j=1}^{n} \beta_j \frac{\partial f_{k-j}}{\partial \alpha_i} \tag{B.16}$$

$$\frac{\partial f_k}{\partial \beta_i} = -\mu_{k-i} + \sum_{j=1}^{n} \beta_j \frac{\partial f_{k-j}}{\partial \beta_i}. \tag{B.17}$$

Eqs.(B.16) and (B.17) are recursive equations to calculate $\frac{\partial f_k}{\partial \alpha_i}$ and $\frac{\partial f_k}{\partial \beta_i}$ where $k = 1, 2, ..., N$ and $i = 1, 2, ..., n$. From Eq.(B.12), the initial conditions for Eqs.(B.16) and (B.17) are

$$\frac{\partial f_0}{\partial \alpha_i} = \frac{\partial f_{-1}}{\partial \alpha_i} = \cdots = \frac{\partial f_{-n+1}}{\partial \alpha_i} = 0 \tag{B.18}$$

$$\frac{\partial f_0}{\partial \beta_i} = \frac{\partial f_{-1}}{\partial \beta_i} = \cdots = \frac{\partial f_{-n+1}}{\partial \beta_i} = 0 \tag{B.19}$$

where $i = 1, 2, ..., n$. Signal μ_{k-i} in Eq.(B.17) is obtained from Eq.(B.10). In summary, X_i (Eq.(B.9)) can be obtained from Eqs.(B.16) and (B.17) with initial conditions (B.18) and (B.19).

Finally, we need a method to choose $\hat{\underline{\theta}}^0$. Our approach is to approximate the nth-order ARMA model in Eq.(B.1) by a 2nth-order AR model, i.e.,

$$V(z) = \frac{1}{1 - \sum_{i=1}^{2n} \gamma_i z^{-i}}. \tag{B.20}$$

Comparing Eqs.(B.1) and (B.20), we have

$$\frac{1 - \sum_{i=1}^{n} \beta_i z^{-i}}{1 - \sum_{i=1}^{n} \alpha_i z^{-i}} = \frac{1}{1 - \sum_{i=1}^{2n} \gamma_i z^{-i}} \tag{B.21}$$

or,

$$(1 - \sum_{i=1}^{n} \beta_i z^{-i})(1 - \sum_{i=1}^{2n} \gamma_i z^{-i}) = 1 - \sum_{i=1}^{n} \alpha_i z^{-i}. \tag{B.22}$$

Equating the cofficients of z^{-i} on both sides of Eq.(B.22), we have

$$\alpha_1 = \beta_1 + \gamma_1 \tag{B.23}$$

$$\alpha_2 = \beta_2 - \beta_1 \gamma_1 + \gamma_2 \tag{B.24}$$

$$\alpha_n = \beta_n - \beta_{n-1}\gamma_1 - \cdots - \beta_1 \gamma_{n-1} + \gamma_n; \tag{B.25}$$

and

$$0 = -\beta_n \gamma_{j-n} - \beta_{n-1}\gamma_{j-n+1} - \cdots - \beta_1 \gamma_{j-1} + \gamma_j \quad j = n+1, \cdots, 2n. \tag{B.26}$$

Hence, if we can obtain γ_l for $l = 1, 2, ..., 2n$, then β_i ($i = 1, 2, ..., n$) can be obtained from Eq.(B.26), because it contains exactly n equations; and, α_i ($i = 1, 2, ..., n$) can then be obtained from Eqs.(B.23)-(B.25).

Because V_k is now modeled as an AR(2n) model with AR parameters γ_l ($l = 1, 2, ..., 2n$), we have, from Eqs.(1) and (B.20) (ignore the noise n_k), that

$$z_k = \sum_{i=1}^{2n} \gamma_i z_{k-i} + \mu_k. \tag{B.27}$$

Using the Yule-Walker equations [10] for Eq.(B.27), we obtain the estimates of γ_i from

$$\begin{bmatrix} \hat{\gamma}_1 \\ \hat{\gamma}_2 \\ \cdots \\ \hat{\gamma}_{2n} \end{bmatrix} = \begin{bmatrix} 1 & \hat{\rho}_1 & \cdots & \hat{\rho}_{2n-1} \\ \hat{\rho}_1 & 1 & \cdots & \hat{\rho}_{2n-2} \\ \cdots & \cdots & \cdots & \cdots \\ \hat{\rho}_{2n-1} & \hat{\rho}_{2n-2} & \cdots & 1 \end{bmatrix}^{-1} \begin{bmatrix} \hat{\rho}_1 \\ \hat{\rho}_2 \\ \cdots \\ \hat{\rho}_{2n} \end{bmatrix} \tag{B.28}$$

where

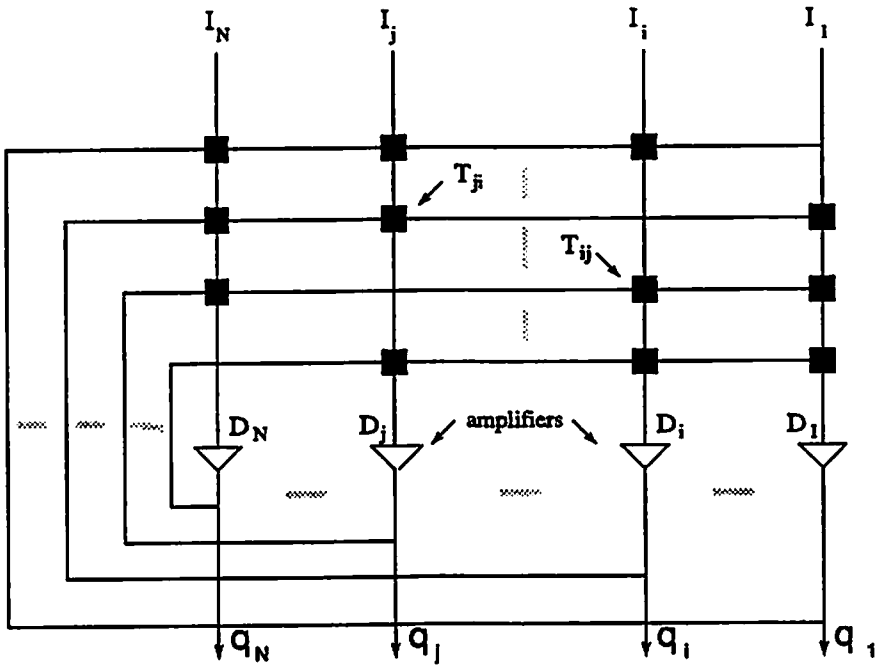
$$\hat{\rho}_j = \hat{r}_j / \hat{r}_0, \tag{B.29}$$

and

$$\hat{r}_j = \frac{1}{N-j} \sum_{k=1}^{N-j} z_k z_{k+j}, \tag{B.30}$$

$j = 0, 1, ..., 2n-1$.

Our final Iterative Least-Squares Algorithm for initial wavelet estimation is summarized in Fig.B1.



$$q_i = \begin{cases} 0, & \text{if } D_i < 0 \\ 1, & \text{if } D_i > 0 \end{cases} \qquad D_i = \sum_{j=1}^{N} T_{ij} q_j + I_i$$

Figure 1: The Hopfield neural network.

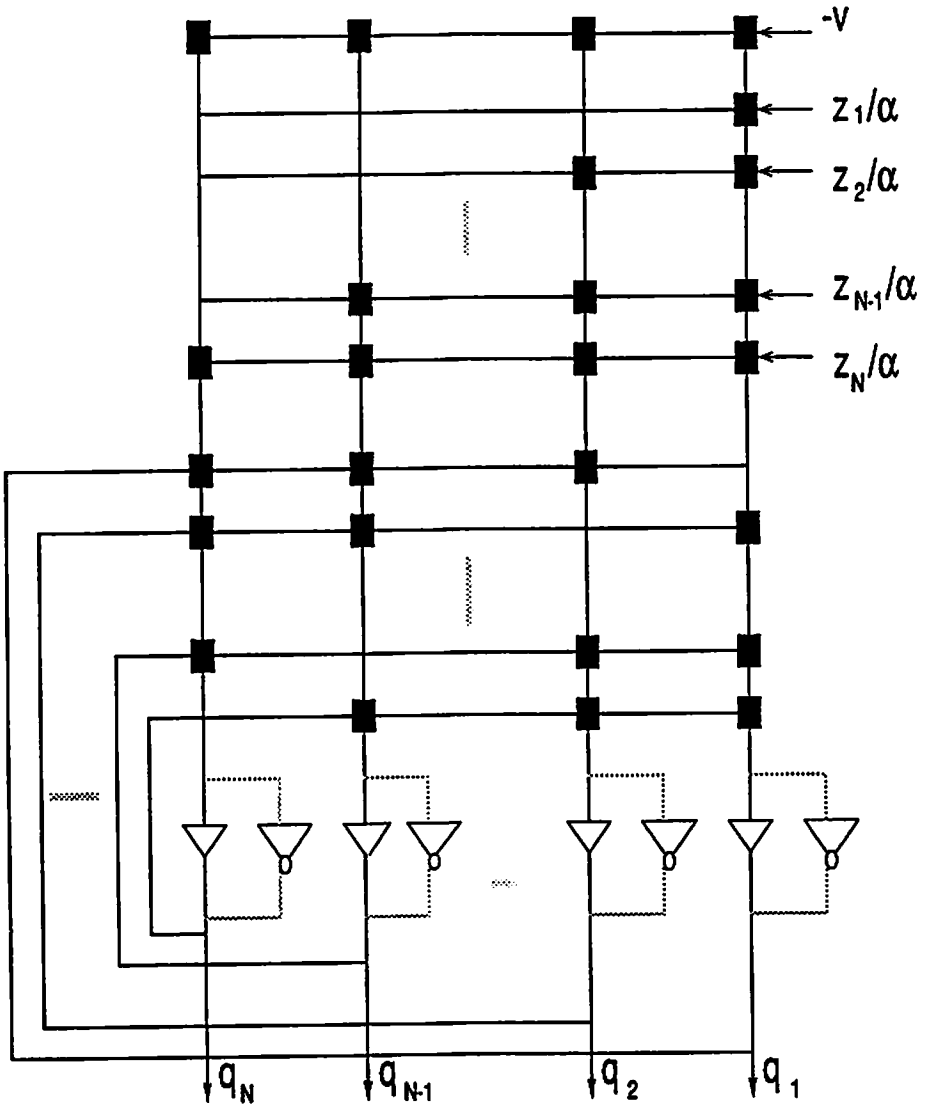


Figure 2: Hardware diagram of the detection Hopfield neural network of the Neural Reflectivity Estimator.

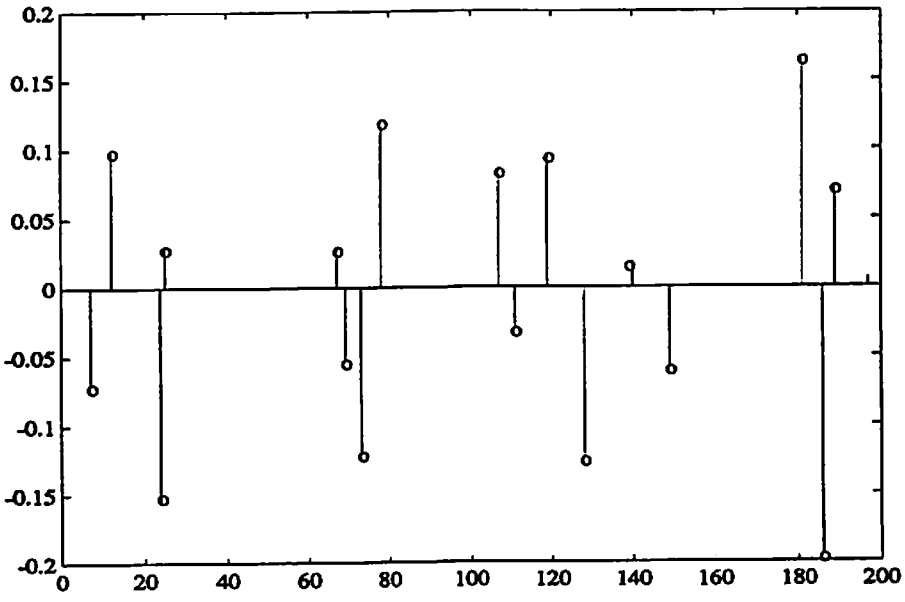


Figure 3: The true reflectivity sequence.

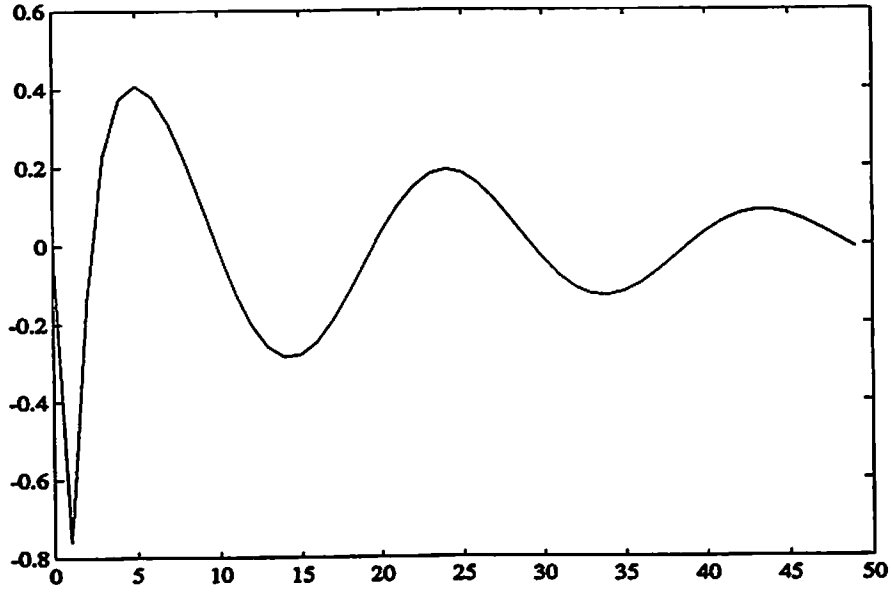


Figure 4: The true broad-band wavelet.

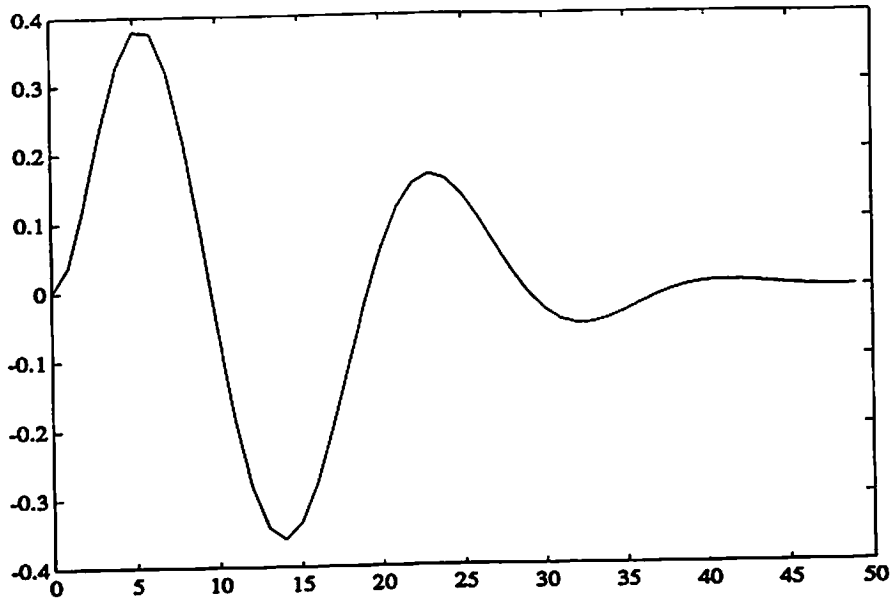


Figure 5: The true narrow-band wavelet.

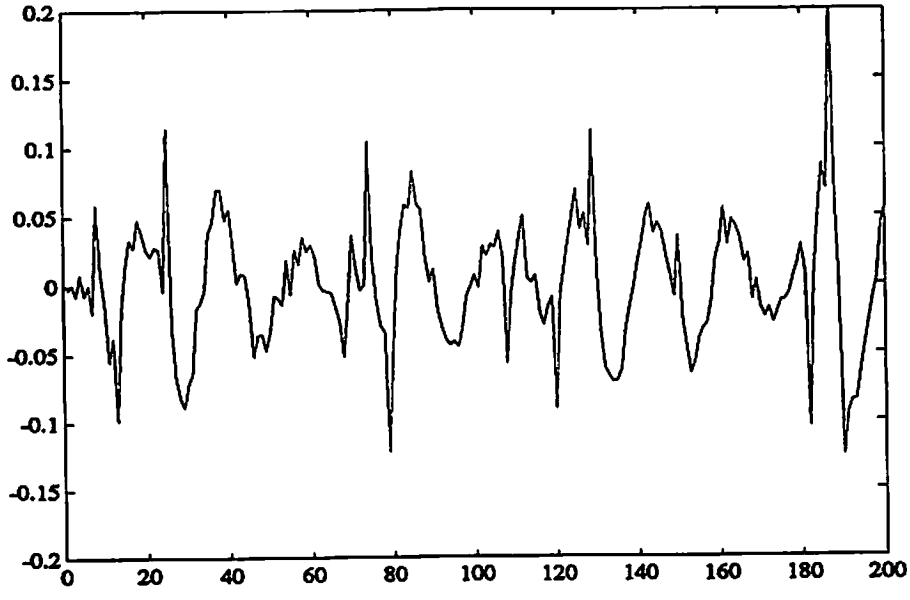


Figure 6: The seismic trace obtained by convolving the true reflectivity sequence in Fig.3 with the broad-band wavelet of Fig.4 and adding a white Gaussian noise with SNR=4.

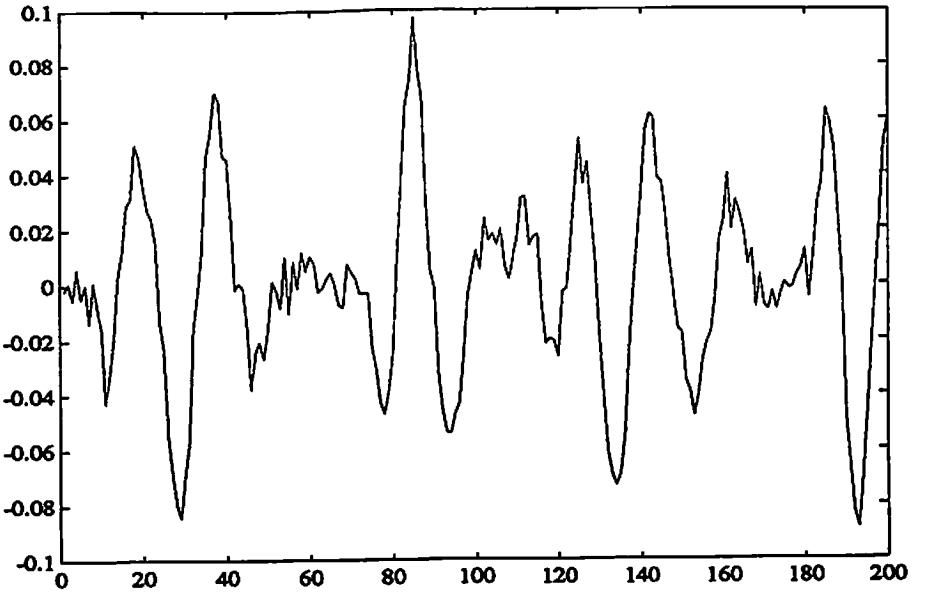


Figure 7: The seismic trace obtained by convolving the true reflectivity sequence in Fig.3 with the narrow-band wavelet of Fig.5 and adding a white Gaussian noise with SNR=4.

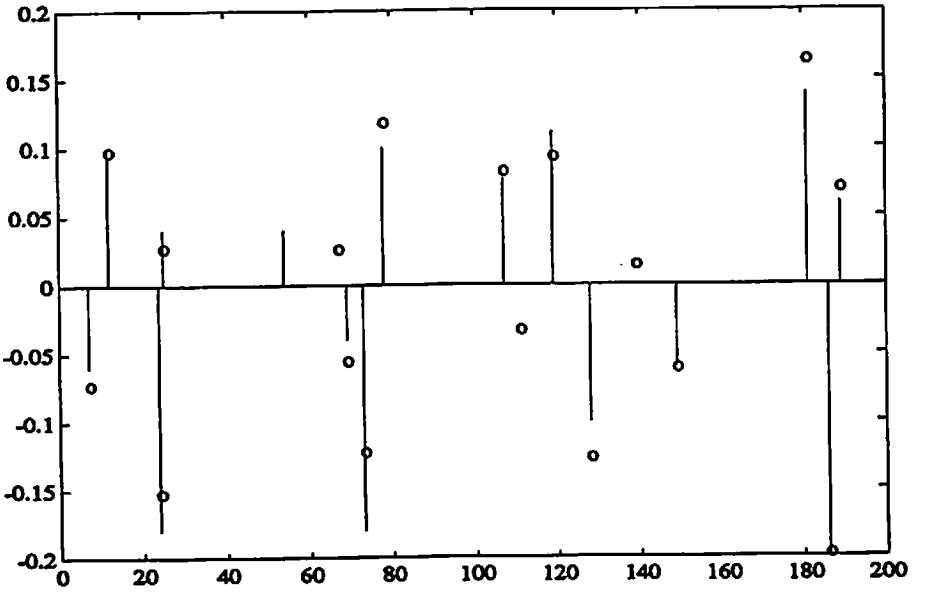


Figure 8: Estimate of the reflectivity sequence using the MPE Neural Reflectivity Estimator for the broad-band trace of Fig.6.

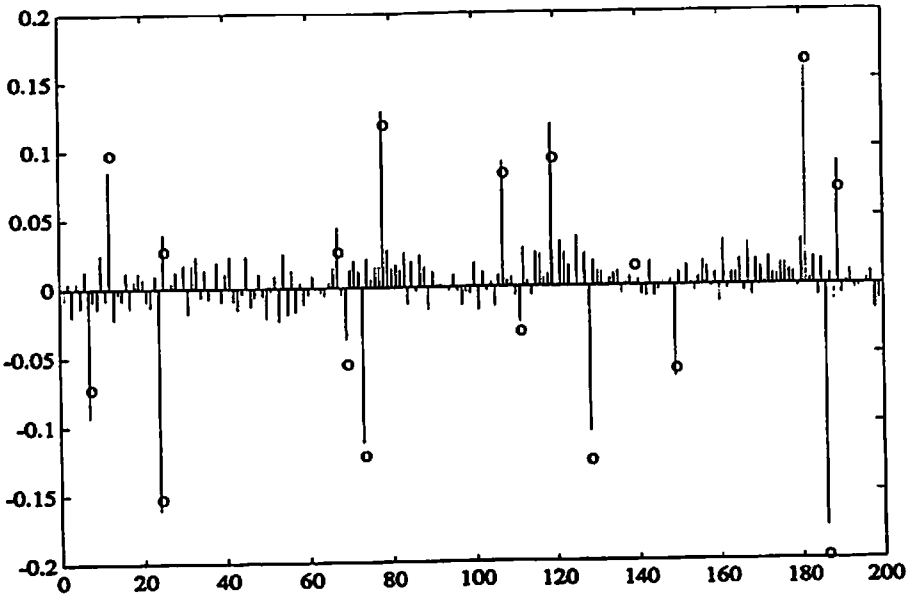


Figure 9: Estimate of the reflectivity sequence using the MVD Filter for the broad-band trace of Fig.6.

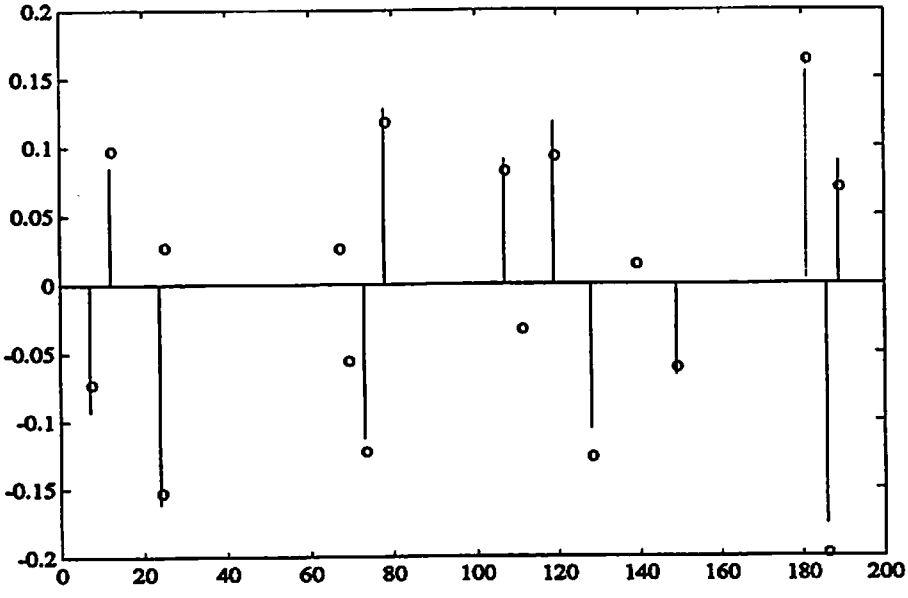


Figure 10: Estimate of the reflectivity sequence using the SMLR Detector for the broad-band trace of Fig.6.

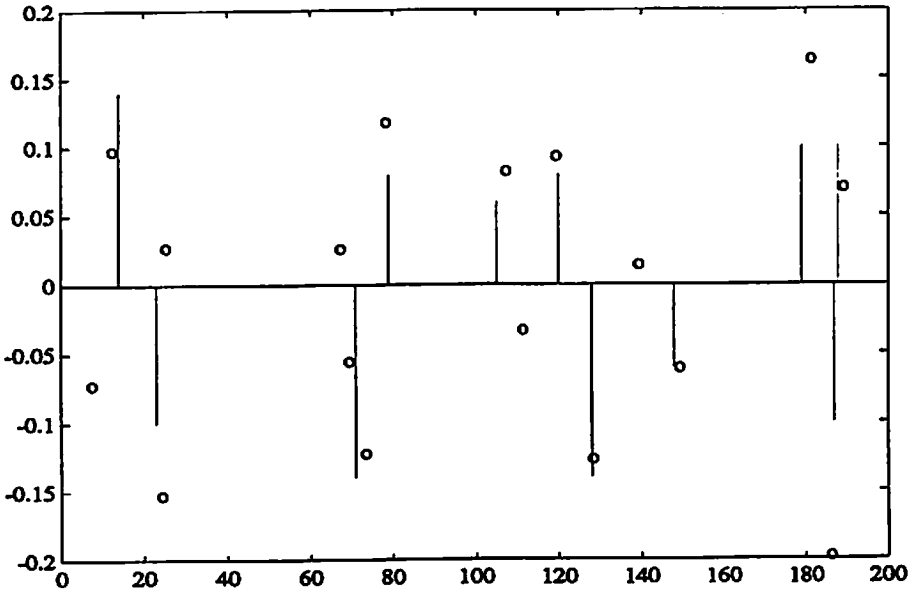


Figure 11: Estimate of the reflectivity sequence using the MPE Neural Reflectivity Estimator for the narrow-band trace of Fig.7.

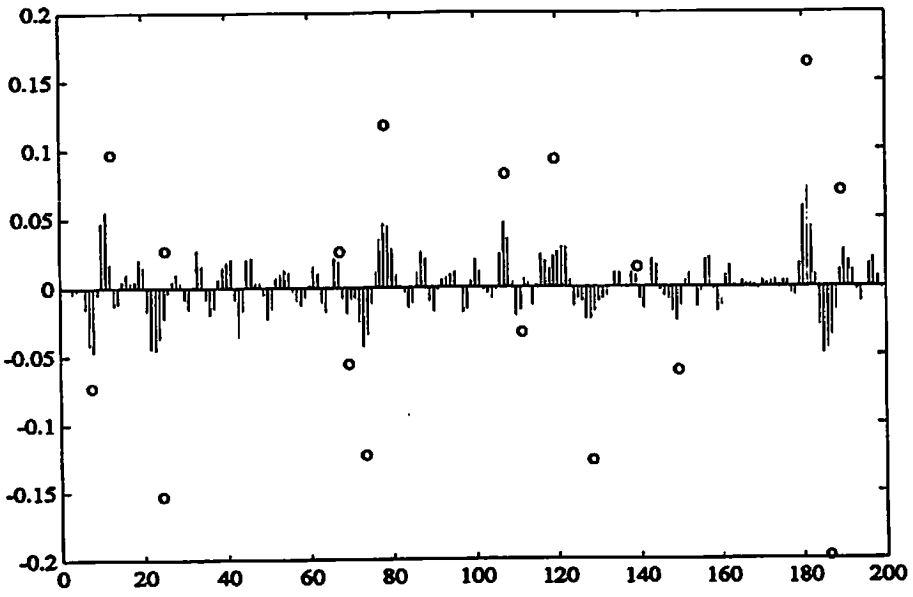


Figure 12: Estimate of the reflectivity sequence using the MVD Filter for the narrow-band trace of Fig.7.

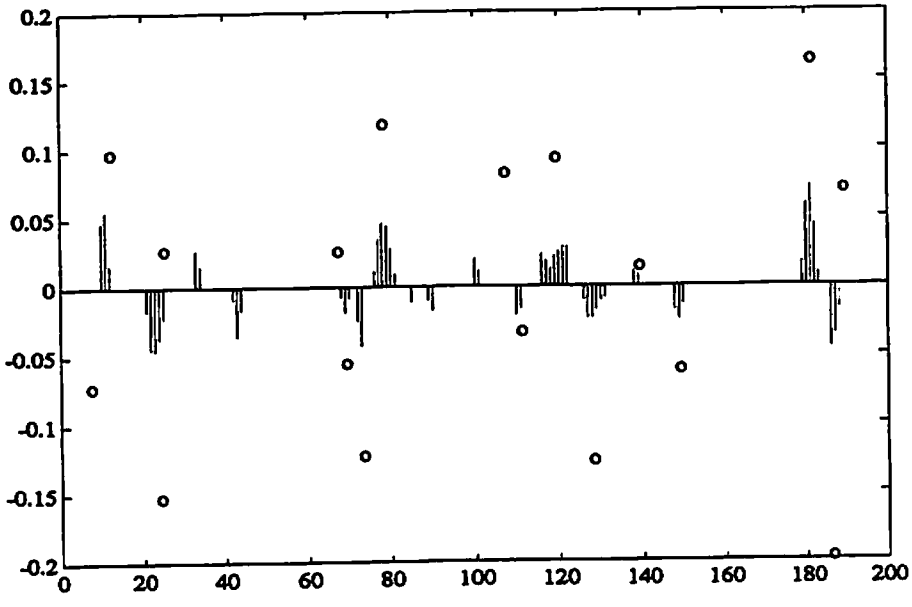


Figure 13: Estimate of the reflectivity sequence using the SMLR Detector for the narrow-band trace of Fig.7.

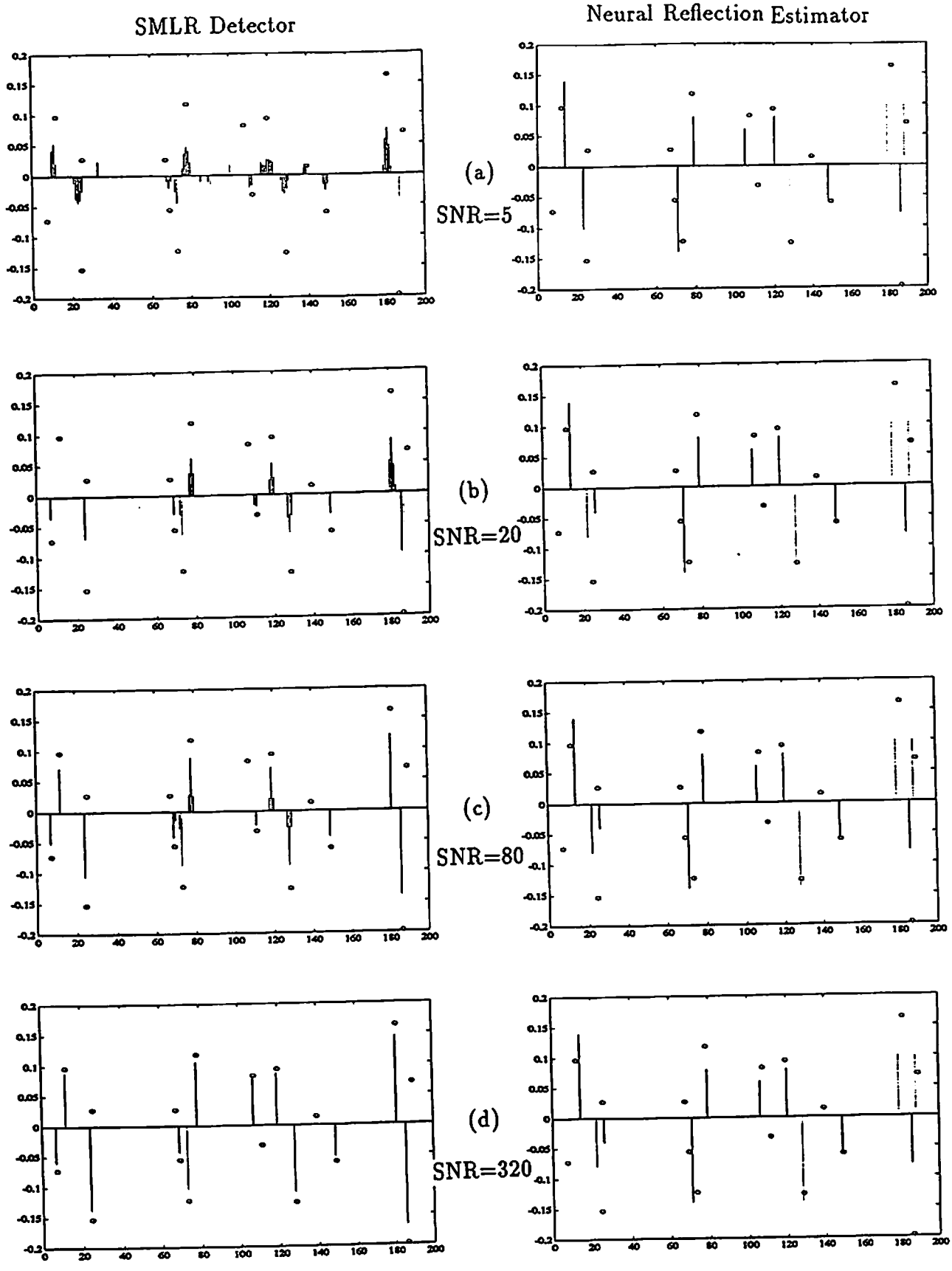


Figure 14 . Comparison of SMLR Detector (left column) and Neural Reflection Estimator (right column) for different SNR cases and the narrow-band wavelet: (a) SNR=5; (b) SNR=20; (c) SNR=80; (d) SNR=320.

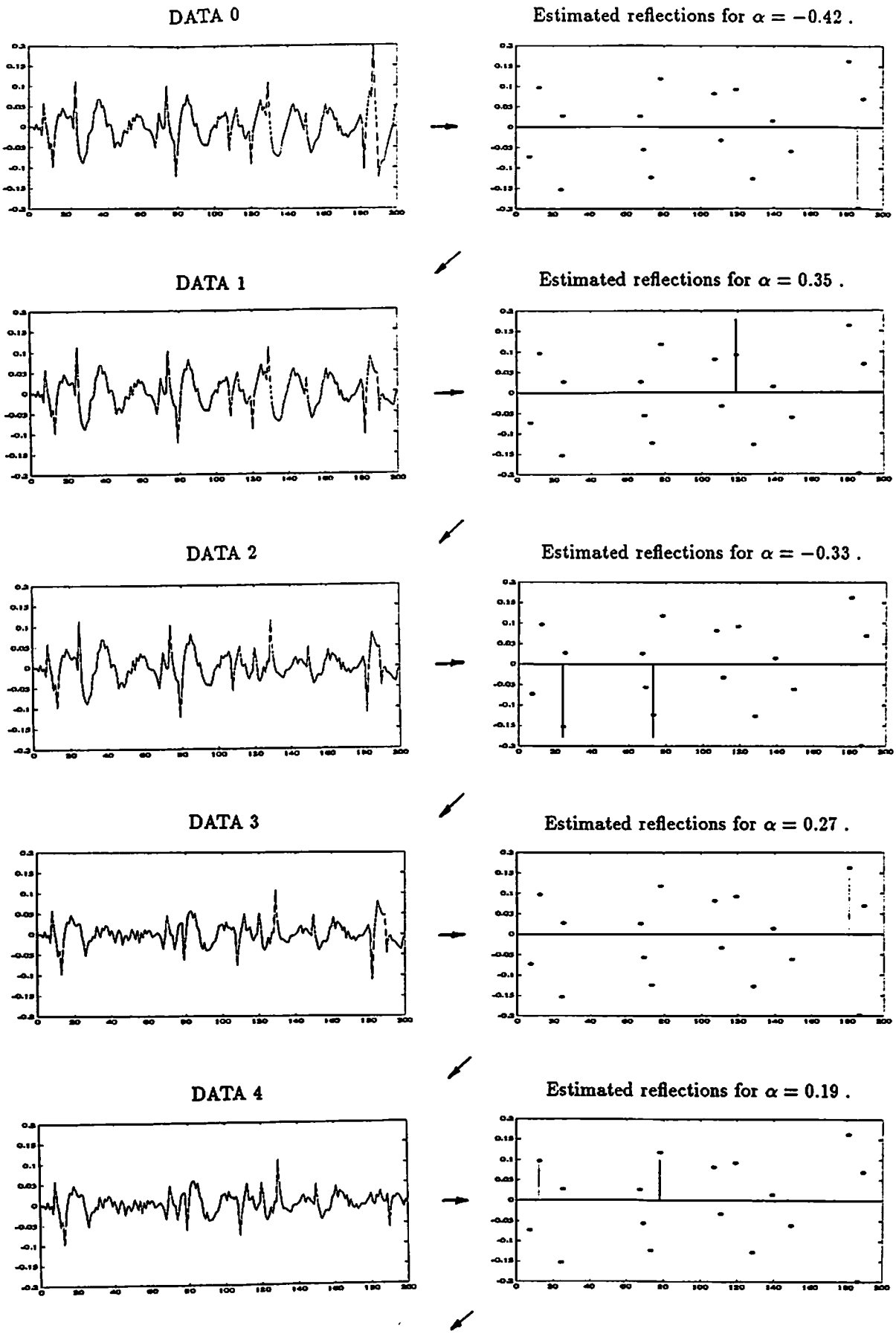


Figure 15: Estimated reflections for different α's and the corresponding updated traces. Data 0 is the original trace which is identical to Fig.6; Data i (i=1,2, ..., 9) is the updated trace after the reflections in the immediate upper-right figure are removed from Data i-1.

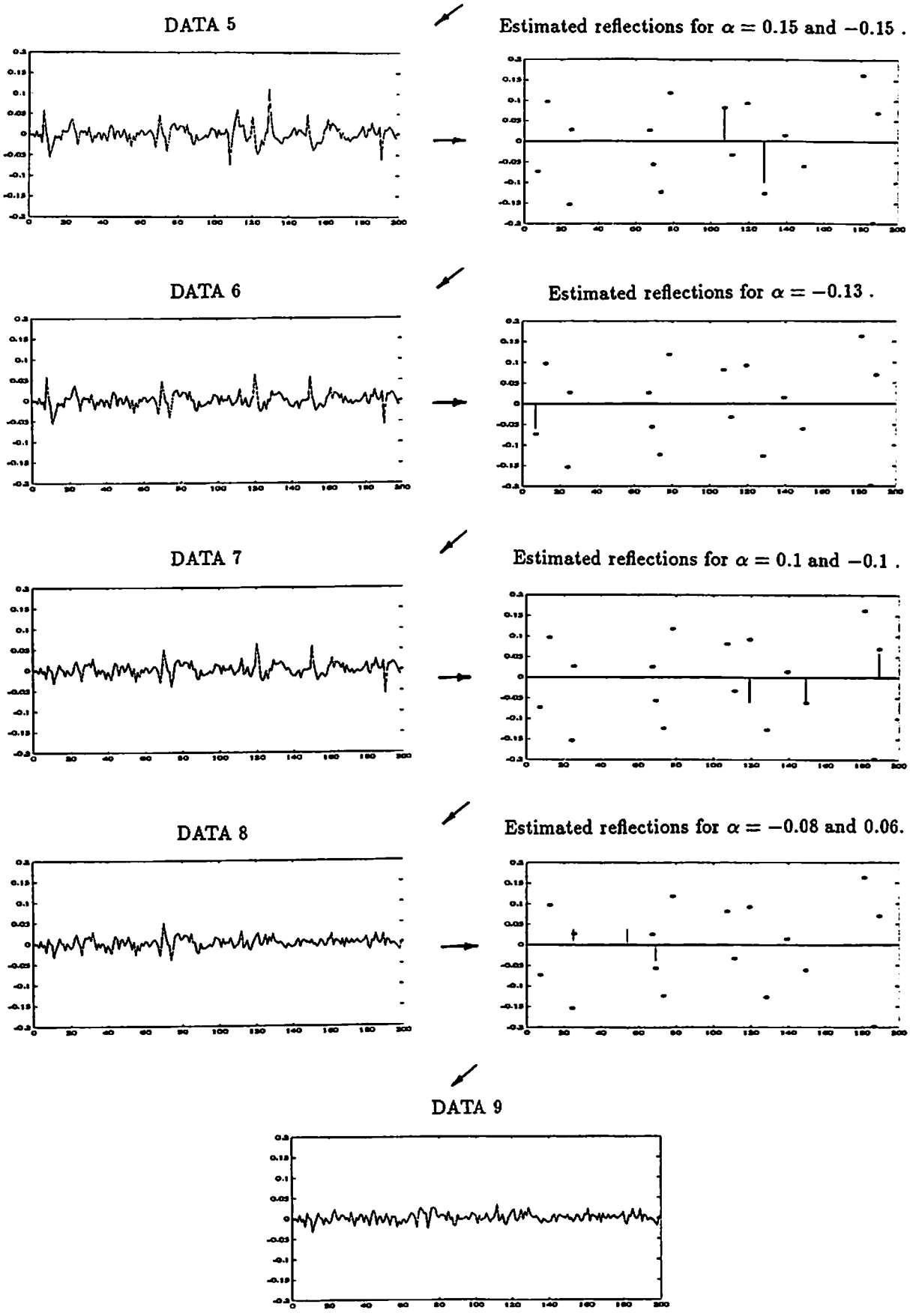


Figure 15: Continued.

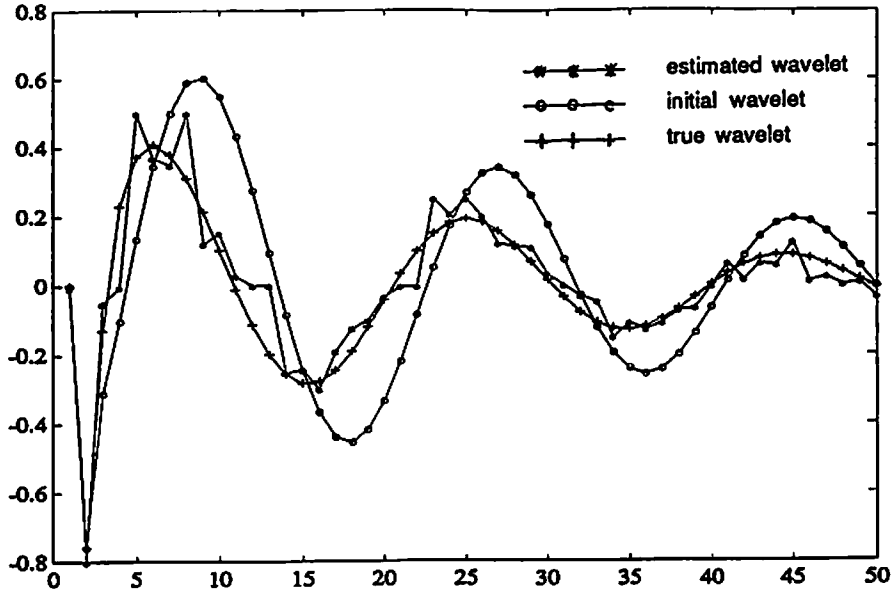


Figure 16: Estimate of the broad-band wavelet using the Simplified Neural Wavelet Extractor.

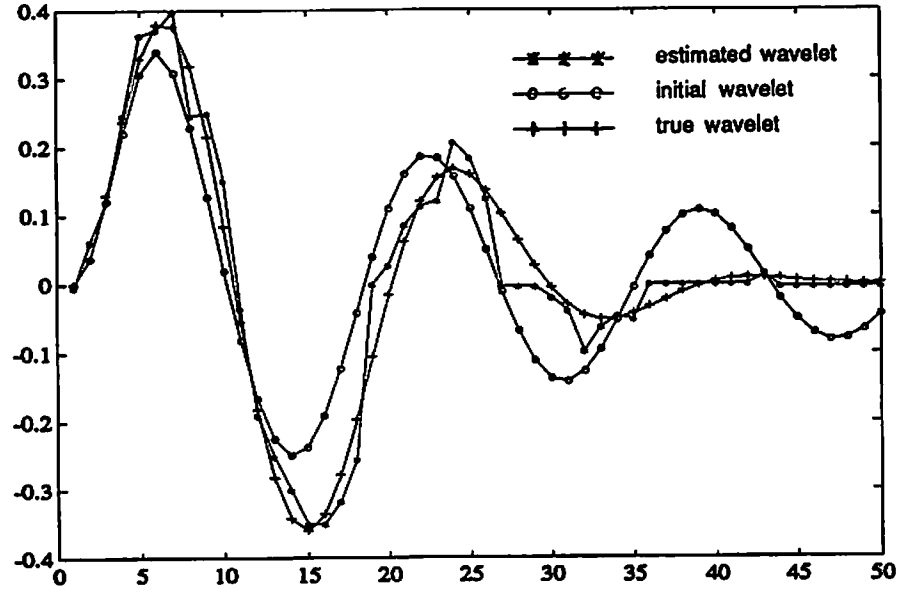


Figure 17: Estimate of the narrow-band wavelet using the Simplified Neural Wavelet Extractor.

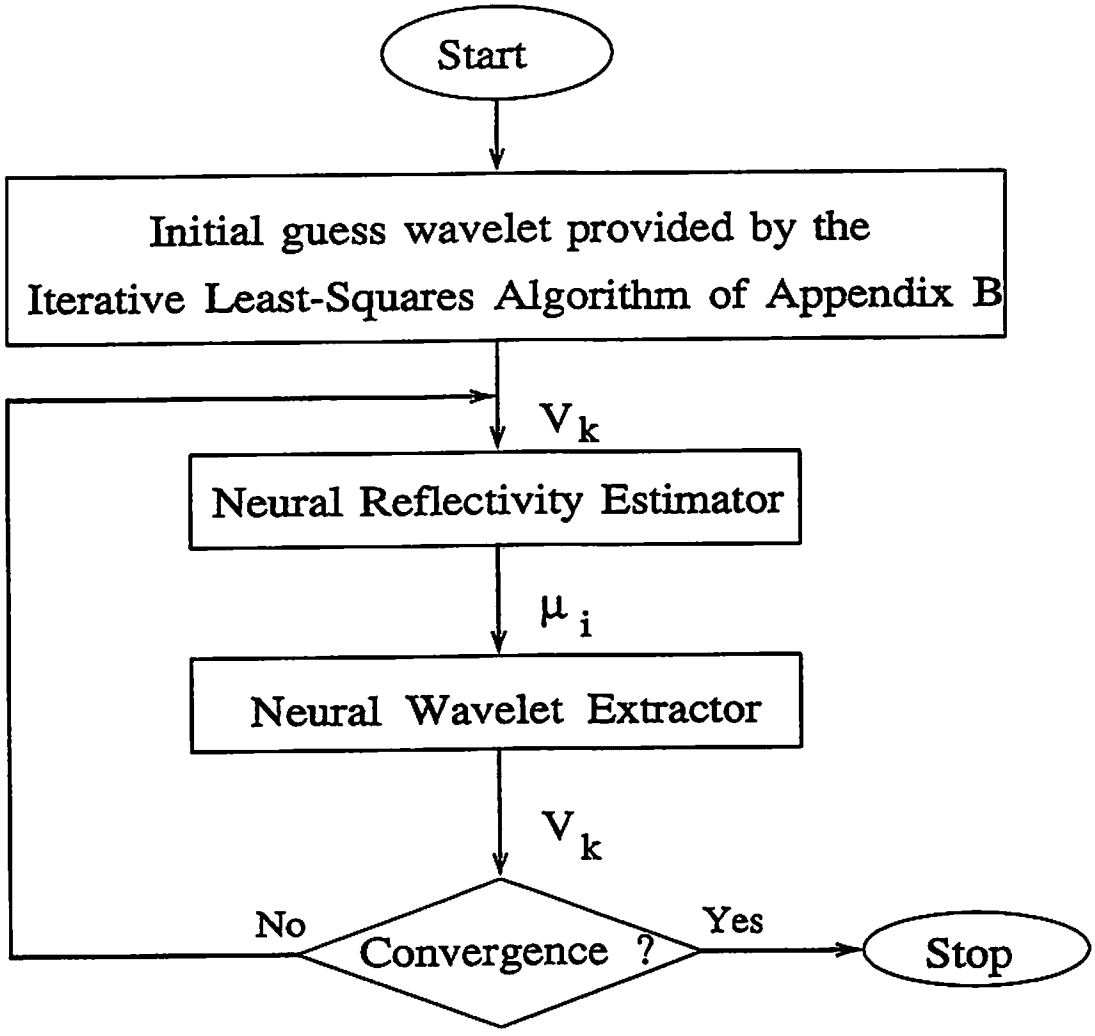


Figure 18: Block-Component Method for simultaneous wavelet extraction and reflectivity estimation.

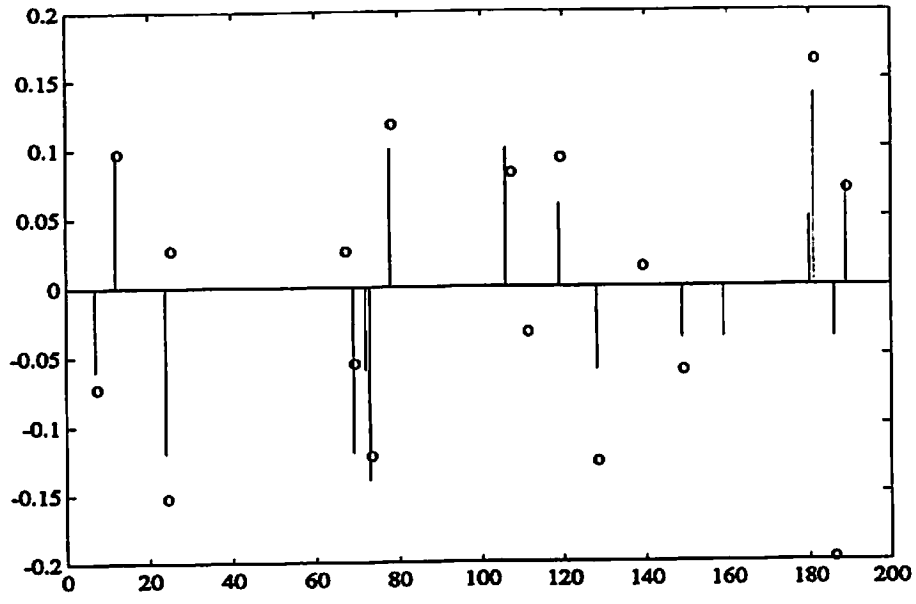


Figure 19: Estimate of the reflectivity sequence using the BCM for the broad-band trace of Fig.6.

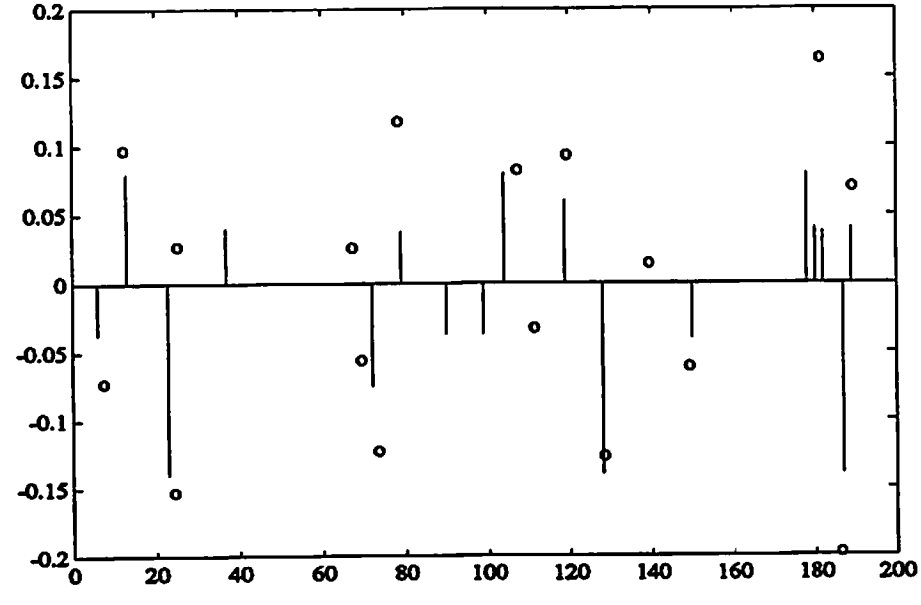


Figure 20: Estimate of the reflectivity sequence using the BCM for the narrow-band trace of Fig.7.

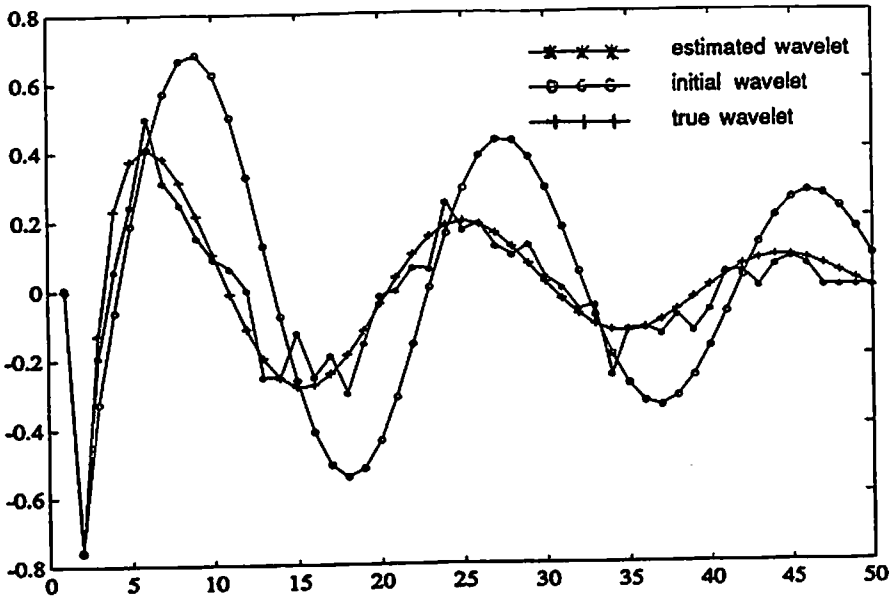


Figure 21: Estimate of the broad-band wavelet using the BCM.

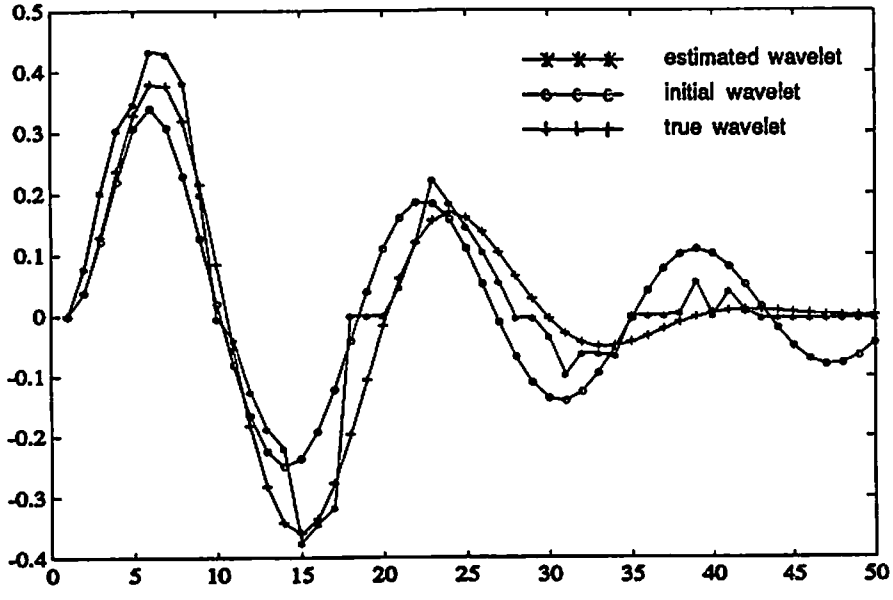


Figure 22: Estimate of the narrow-band wavelet using the BCM.

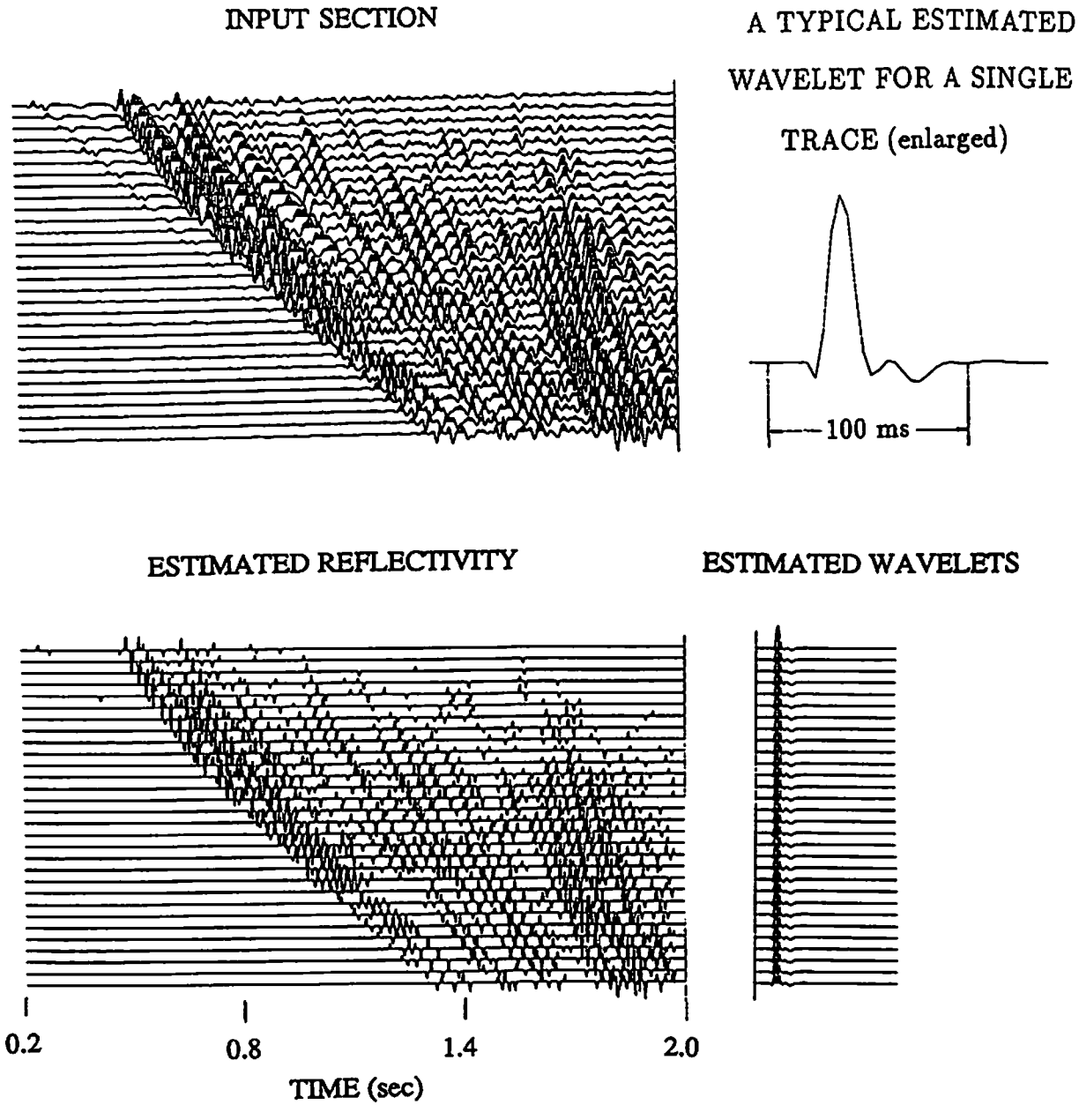


Figure 23: Reflectivity and wavelet estimates using the neural BCM.

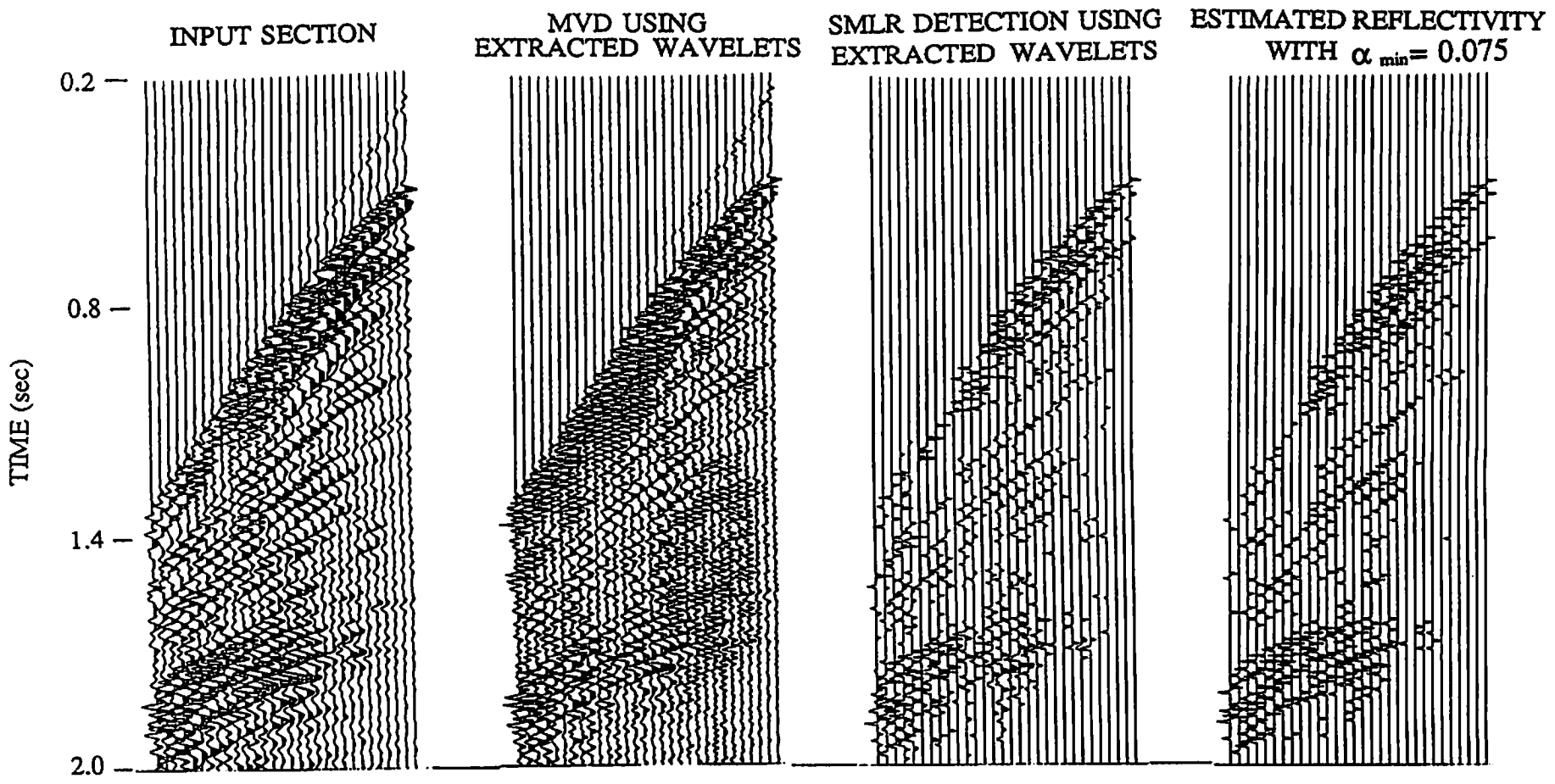


Figure 24: Estimated reflectivity using the neural BCM for α_{min}= 0.075, and Minimum-Variance Deconvolution and SMLR Detection using extracted wavelets from the neural BCM.

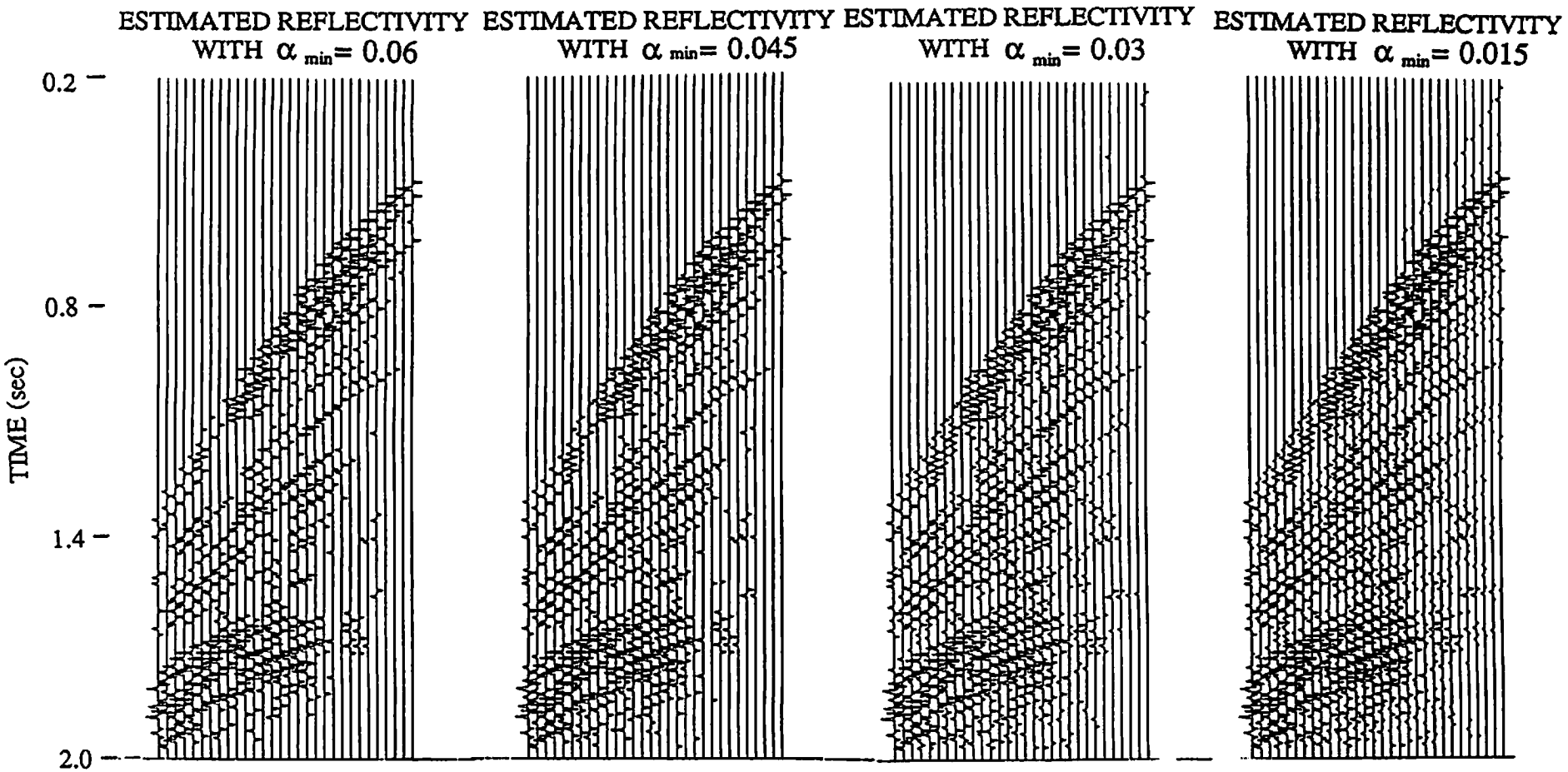


Figure 25: Estimated reflectivity using the neural BCM for α_{min}= 0.06, α_{min} = 0.045, α_{min} = 0.03, and α_{min} = 0.015 .

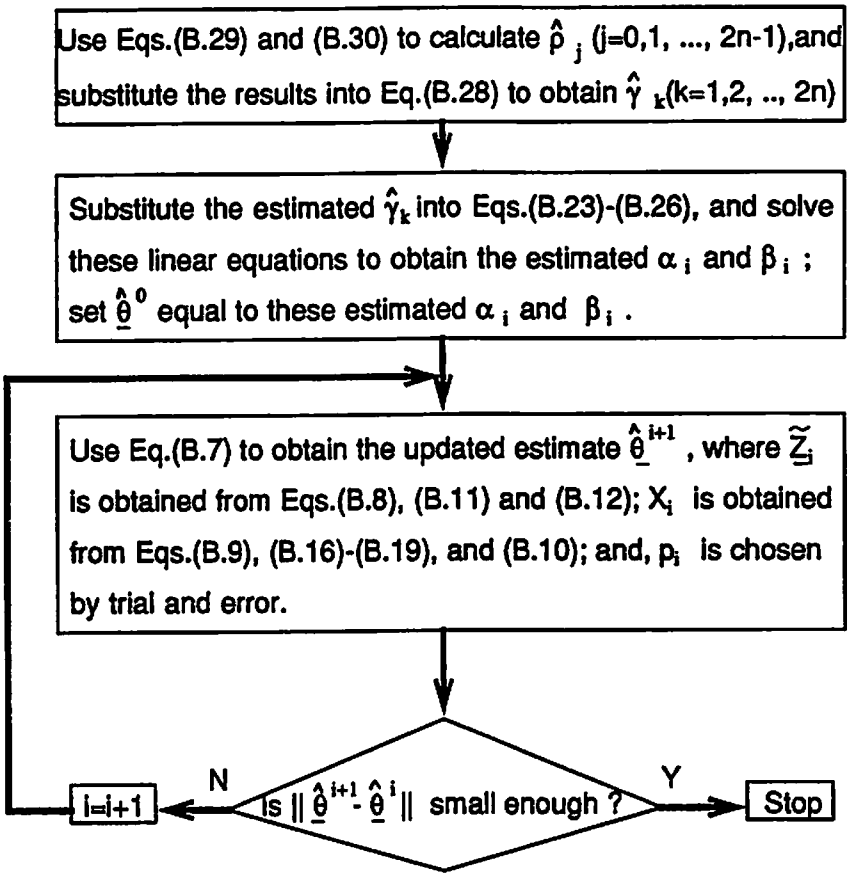


Figure 61: The Iterative Least-Squares Algorithm for initial wavelet estimation.