

A direct inversion scheme for deep resistivity sounding data using artificial neural networks

JIMMY STEPHEN*, C MANOJ and S B SINGH

National Geophysical Research Institute, Hyderabad 500 007, India

**e-mail: jimmy@ngri.res.in*

Initialization of model parameters is crucial in the conventional 1D inversion of DC electrical data, since a poor guess may result in undesired parameter estimations. In the present work, we investigate the performance of neural networks in the direct inversion of DC sounding data, without the need of *a priori* information. We introduce a two-step network approach where the first network identifies the curve type, followed by the model parameter estimation using the second network. This approach provides the flexibility to accommodate all the characteristic sounding curve types with a wide range of resistivity and thickness. Here we realize a three layer feed-forward neural network with fast back propagation learning algorithms performing well. The basic data sets for training and testing were simulated on the basis of available deep resistivity sounding (DRS) data from the crystalline terrains of south India. The optimum network parameters and performance were decided as a function of the testing error convergence with respect to the network training error. On adequate training, the final weights simulate faithfully to recover resistivity and thickness on new data. The small discrepancies noticed, however, are well within the resolvability of resistivity sounding curve interpretations.

1. Introduction

Solving a geoelectrical inverse problem consists of using a set of measurements to evaluate subsurface parameters like resistivity and thickness of different layers. A proper mathematical relation should be developed between the observations and subsurface parameters. The 'good fit' between the measurements and unknown model parameters (resistivity and thickness) is related to an error function, normally a non-linear function, of the subsurface parameters. An optimal solution is derived by the iterative minimization based on local derivatives of the error function. Several attempts were made in the last three decades for 1D resistivity inversion (Kunetz and Rocroi 1970; Ghosh 1971; Zohdy 1989; Meheni *et al* 1996). None of these attempts guarantee a geologically plausible model since most of these algorithms critically depend on the initial parameters given to it. A Monte-Carlo method,

which uses randomly drawn models, is useful when a good starting point is not available (Rubinstein 1981). Simulated annealing (Kirkpatrick 1983) and genetic algorithms (Horne and MacBeth 1994) uses previous model evaluations to drive securely in the model parameter space to the most promising part of error surface. These methods are computationally expensive than the linearized techniques and therefore are more useful only when a good starting model is not available.

In the present study we try to investigate the application of artificial neural networks (ANN) to interpret one-dimensional (1D) electrical deep resistivity sounding (DRS) data over a wide range of model parameters. The use of ANN is motivated by its resident intelligence that resembles the biological neural network having the capability of perceptual interpretation, abstraction and learning (Lippman 1987). Recent application areas of ANN involves the pattern recognition,

Keywords. Artificial neural networks; back propagation; deep resistivity sounding; 1D inversion.

clustering/classification, image processing, content addressability, optimization and control systems. Unlike the conventional methods that incorporate a fixed algorithm to solve a particular problem in geophysics, ANN performs an intelligent non-linear mapping between input and output data allowing the network to acquire important information on the problem being solved. It is featured with its adaptive discrimination or ‘learning’ through repeated exposure to examples and the versatile generalization capability. The use of ANN in locating subsurface targets from geophysical data has been studied by Paulton *et al* (1992). Recent studies show the efficacy of ANN in providing automation for geophysical inversions (Spichak and Popova 2000) and noise discriminations (Manoj and Nagarajan 2003). To the interest of this paper, Macias *et al* (2000) discusses the implementation of ANN for the otherwise complex problem of geophysical parameter estimation with emphasis on vertical electrical sounding and seismic data. The electrical resistivity data inversion is also addressed by Qady and Ushijima (2001) and discusses the performance of different ANN paradigms.

Though the earlier attempts to invert DC resistivity data were successful, the application on real-world data was limited, as they depend mainly on the synthetic data with less parameter variations in more specific curve types. Here we made an attempt to overcome some of these limitations by designing a network that can incorporate a wide spectrum of model parameters. The inversion is done in two steps,

- identifying the curve type and
- the estimation of model parameters.

In both the cases we use feed-forward neural network (FNN) as a mapping function between inputs and outputs. Fast back propagation with momentum (FBPM) is used as the learning rule, which is sufficient to design any kind of non-linear classifier and by far the most popular learning algorithm (Guyon 1991; Haykin 1994). Since an optimum size and efficiency of the training data set is important to render generalization capability to the network (Kung and Hwang 1988; Hush and Horne 1993), the database is generated cautiously. Adequate cross-validation is done for ensuring the performance of network on new data examples. The details of network and learning schemes are discussed later.

2. ANN architecture

Figure 1 shows the architecture of the 3 layer FNN we adopted in the present study. The structure of the processing element (PE), called neuron, which

has a non-linear activation function is also shown. The input layer receives single inputs at each node (1 to I). Second and third layers represent the hidden layer and output layer respectively and have the same activation function in all neurons. Here we use a logistic (unipolar) sigmoid function as the activation function for neurons. Sigmoidal function can produce outputs with reasonable discriminating power (Bishop 1995) and its output functions are differentiable, which is essential for the back-propagation of errors (Yegnanarayana 2001). Its Gaussian shaped derivatives help to stabilize the network and compensate for over-correction of the weights (Caudill 1988). During FNN training, each hidden and output neuron process inputs by multiplying each input by its weights. The products are summed and processed using the activation function. The expression for logistic sigmoid function and its derivative are given below.

$$f(x) = \frac{1}{(1 - e^{-x})}, \quad (1)$$

$$f'(x) = f(x)[1 - f(x)]. \quad (2)$$

The number of neurons in the hidden layer varies as per the requirement of optimum performance, which could be decided on trial and error basis. Initial weights are assigned at random in the range suitable for the activation function at neurons. The signal is fed forward through the network as shown in figure 1 and hence the name feed-forward. The learning cycle begins with the updating of the weights of output layer ($W2_{kj}$) and utilized to adjust the input weights ($W1_{ji}$) to obtain the desired output, known as a back propagation, or otherwise generalized delta rule (Werbos 1974; Werbos 1990; Rumelhart *et al* 1986). Back propagation is a gradient descent algorithm, as the weight updating is in the direction of negative gradient of the defined error function. The ability to obtain strict convergence for a lengthy set of data speaks strongly of the pattern recognition capability of the neural network technique, but the true test of a model is its ability to reproduce features that were not included in its training set. In other words, the performance of a network depends greatly on its generalization capabilities, rather than being a look-up table.

Using the back propagation algorithm, the weighting coefficients and biases are converged. In a three layer FNN, the outer two layers represent the input and output, while the inner one represents the hidden layer (figure 1). The number of neurons in all these layers is decided on the interest of the user. The hidden layer maintains the connectivity between the input and the output through the activation functions assigned to its neurons. Here $X_1, X_2, \dots, X_i, \dots, X_I$, form the

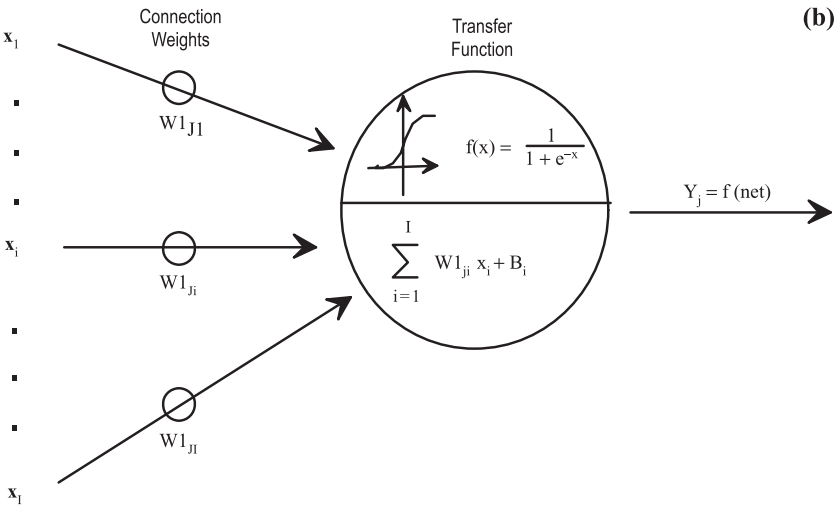
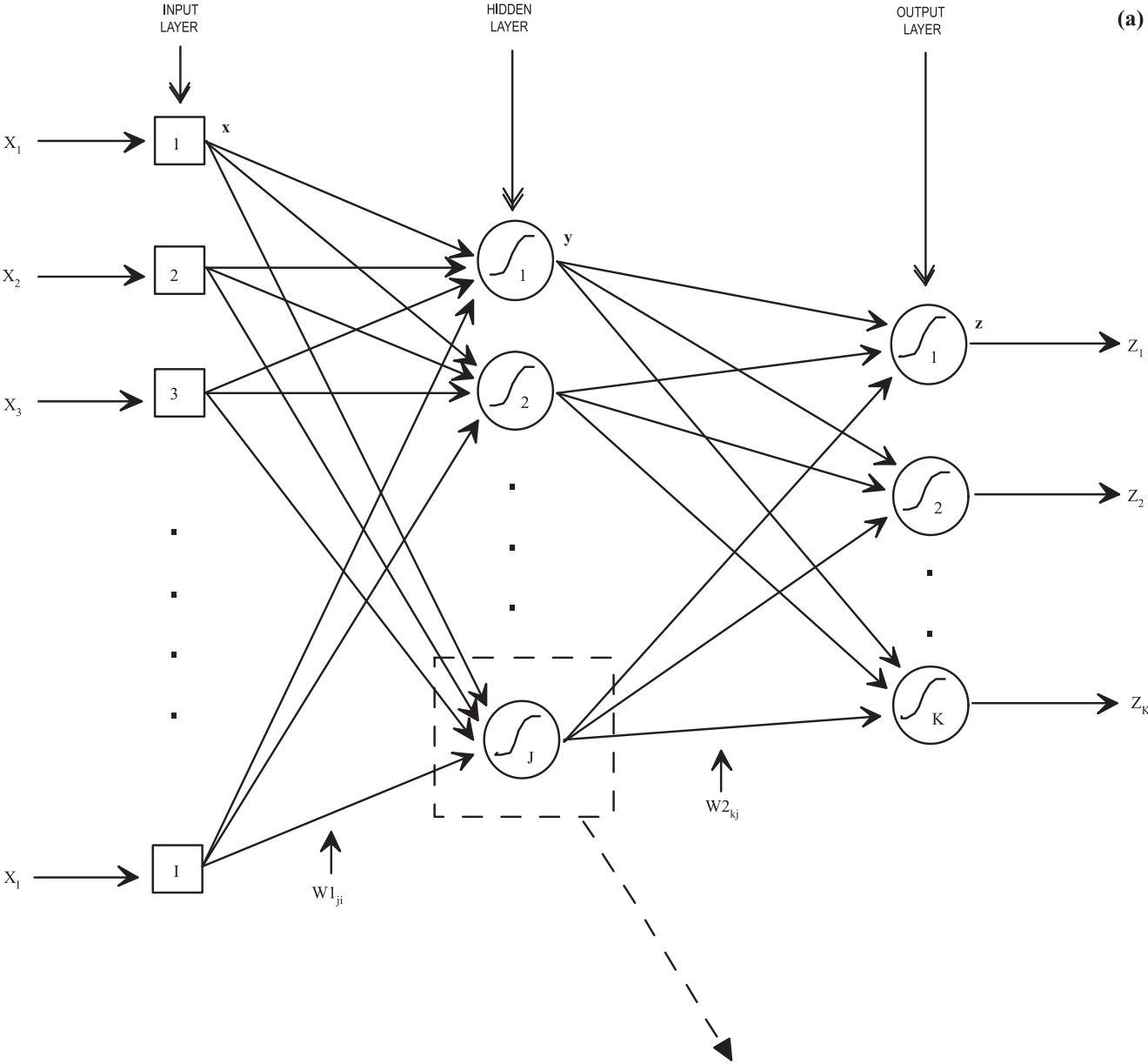


Figure 1. Neural network architecture; **(a)** structure of the 3 layer feed-forward ANN used for present study, **(b)** model of the artificial neuron/processing element (PE).

input vector of examples submitting to the network. At the input layer, these inputs are activated by the activation function of the layer and outputs the activation vector of input layer given by $x = x_1, x_2, \dots, x_i, \dots, x_I$. These vector elements are generally known as the input neurons. Each of these input neurons make connections with all the J hidden neurons by a connection weight $W1_{ji}$ and each hidden neuron holds a bias to the input vector. These vector units get activated at the hidden layer and passes the activation vector of hidden layer $y = y_1, y_2, \dots, y_j, \dots, y_J$, to the right hand side. Similar to the input layer, all the components of this activation vector also make connections with all the K nodes of output layer by a connection weight $W2_{kj}$. These output layer nodes also establish a bias with the hidden layer. The activation function of the output layer transforms the input to this layer to the activation vector of output layer, viz., $z = z_1, z_2, \dots, z_k, \dots, z_K$. The number of nodes or neurons in the outer layers are decided on the basis of our problem definition, whereas the hidden neurons are fixed on trial and error methods to achieve the optimum network performance. The increase and decrease in the learning rate enables the network to learn and converge fast. As the training progresses the learning rate is adjusted with respect to the weight adjustments. The details of network components are discussed later in this paper, separately for both the networks.

3. FNN training

At any stage of the training, say at a step m , let (x, d) be the current sample of the function mapping the input space to the output space $R^I \rightarrow R^K$. Then the mean squared error function at ' m ' is given as:

$$E^{(m)} = \frac{1}{K} \sum_{k=1}^K (d_k^{(m)} - z_k^{(m)})^2 \quad (3)$$

where, d_k is the desired output and z_k is the network output given by

$$z_k = f \left(\sum_{j=1}^J W2_{kj} \cdot Y_j \right) \quad (4)$$

where, ' f ' is the sigmoidal function defined by equation (1). The subscripts i, j and k used in all these equations represent the inputs, hidden neurons and outputs respectively and can be easily realized from figure 1.

Here the error function conceived in the n -dimensional hyperspace can be deemed as a bowl, whose bottom-most point indicates the optimum

set of weights (Lippmann 1987). Weights are to be changed in the direction of negative gradient of this error function. Since the input vector ' x ' is given at the input layer and the desired output d is available only at the output layer, the error between the desired output vector and the actual output vector ' z ' is available only at the output layer. Using this error, it is necessary to adjust the weights ($W1_{ji}$) from the input units to the hidden units and the weights ($W2_{kj}$) from the hidden units to the output units. It is done in two steps, initially the output layer weights are updated and subsequently the previous (input) layer weights are updated. The mathematical treatment for these two-step weight adjustments is given below. A detailed derivation and weight adjustment procedure can be obtained from Lippmann (1987); Yegnanarayana (2001) and Haykin (1994).

3.1 Updating of layer weights

In a multi-layer FNN, the output layer is first adjusted for the weight variations. The gradient descent along the error surface to determine the increment in the weight connecting the units k and j at current iteration step ' m ' is given by,

$$\Delta W2_{kj}^{(m)} = -\eta \frac{\partial E^{(m)}}{\partial W2_{kj}^{(m)}}, \quad (5)$$

where $\eta > 0$ is the *learning rate*, which may also vary for each presentation of the training pair. In the above equation, substituting for ' E ' from equation (3) in the above equation, we have,

$$\Delta W2_{kj}^{(m)} = \eta \delta_k^m y_j^m, \quad (6)$$

where, y_j is the hidden layer output and δ_k is the derivative of output z_k as per the equation (2), given by,

$$\delta_k^m = z_k^m (1 - z_k^m). \quad (7)$$

The weight update is now given by,

$$\begin{aligned} W2_{kj}^{(m+1)} &= W2_{kj}^{(m)} + \Delta W2_{kj}^{(m)} \\ &= W2_{kj}^{(m)} + \eta \delta_k^{(m)} y_j^{(m)} \\ &\text{for } j = 1, 2, 3, \dots, J. \end{aligned} \quad (8)$$

The above equation is repeated for $k = 1, 2, 3, \dots, K$, thus completing update of all $(k \times j)$ weights of the output layer. Similarly for the input layer updating,

$$\Delta W1_{ji}^{(m)} = \eta \delta_j^{(m)} x_i^{(m)}. \quad (9)$$

The input weight update is given by,

$$\begin{aligned} W1_{ji}^{(m+1)} &= W1_{ji}^{(m)} + \Delta W1_{ji}^{(m)} \\ &= W1_{ji}^{(m)} + \eta \delta_j^{(m)} x_i^{(m)} \\ &\text{for } i = 1, 2, 3, \dots I. \end{aligned} \quad (10)$$

The above equation is repeated for $j = 1, 2, 3, \dots J$, thus completing update of all $(j \times i)$ weights of the input layer. To stabilize the convergence, a momentum gain (α) is added to the weight change given by,

$$\Delta W1_{ji}^{(m)} = \alpha \Delta W1_{ji}^{(m-1)} + \eta \delta_j^{(m)} x_i^{(m)}, \quad (11)$$

where $\alpha (0 \leq \alpha < 1)$ is the momentum constant and m represents the current iteration index. The momentum constant helps to stabilize the oscillations during the learning process, accelerates the descent to the minimum of the error surface and reduces the effects of local minima of the error surface (Plaut *et al* 1986; Rumelhart *et al* 1986; Fahlman 1989). Momentum gain can also speed up training in very flat regions of the error surface and suppresses the weight oscillations in steep valleys or ravines (Schiffman *et al* 1992). All the weight adjustments illustrated above represents one iteration for a given example. Likewise, adjustments of weights are done for other examples in the training database. Presentation of the entire database (all training examples) and subsequent adjustment of weight constitute one epoch. The training can be stopped on the convergence of weighting coefficients at which the mean square error at the output falls below a desired error goal or at the completion of a certain number of iterations. More details on the convergence of weighting coefficients are discussed by Luo and Unbehauen (1997). In deciding the optimum training, the testing performance on new data sets is also important. Because a monotonically observed convergence in training may not hold true with the case of testing samples where the generalization capability of network is an important factor. The details are discussed in the later portions.

Here we discuss the approach introduced in this paper to use FNN based technique for direct parameter estimation. We propose two networks in the same architecture discussed above for inverting the measured DRS apparent resistivity values. This two-stage network scheme is shown in figure 2. The apparent resistivity data sampled at 30 predetermined AB/2 values are fed into the first network where it identifies the curve type. Then it is passed into the second network, which outputs the model parameters. A total of 20 curve types comprising most of the commonly encountered 3, 4 and 5 layer resistivity cases were included in the study.

In each case the ‘loading’ problem, the determination of weights from the training examples (Judd 1990; Blum and Rivest 1992), is better solved and weights are saved for further simulation with new examples.

4. The database

The efficacy of any network application depends mainly on the database used for training and testing the network. The training data set should contain all the possible curve types with an optimum number of data points depending on the problem to be addressed. In this work we use both field data and synthetically generated data to train the neural network. A wide range of resistivity and thickness were included in the data to represent ranges observed in crystalline terrains as such over the Southern Granulite Terrain (SGT) of India. It exposes the exhumed lower crust characterized by high resistive granulite and gneissic complexes with many deep-seated faults (Grady 1971). The region also provides the window for the oldest forming crusts of the world with wide spectrum of metamorphic events comprising a number of shear zones separating diversified crustal blocks (Drury *et al* 1984). Deep Resistivity Sounding (DRS) measurements were carried out along the corridors of a major N-S geo-transect extending from Kuppam to Palani as part of an integrated geophysical program (Singh *et al* 2000; Reddy *et al* 2001; Singh *et al* 2003). A simplified geological map of this region is shown with the DRS locations in figure 3. Schlumberger electrode configuration with an electrode spread (AB) of 10 km was used to probe the deeper resistive structures. The model parameter ranges for generating the synthetic data in the present ANN scheme were selected on the basis of these DRS measurements. Figure 4 shows the layer-wise parameter ranges for three orders of curve types used in the present study.

A total of 20 DRS curve types belonging to 3, 4 and 5 layered resistivity structures (shown in figure 4), frequently encountered along the crystalline terrain in the scope of the present study, were considered. To represent each type, 150 examples were used among which two-thirds were used for training and one-third was used for testing the network. The examples are selected in such a way that, the range of layer parameters falls within the specified range and covers all the possible model parameter combinations in that particular curve type, which is essential to obtain the generalization capability for the trained network. Apparent resistivities at 30 successive AB/2 positions, 1.5 m to 5000 m, were used as input. The large range of apparent resistivity necessitated a logarithmic scale to represent

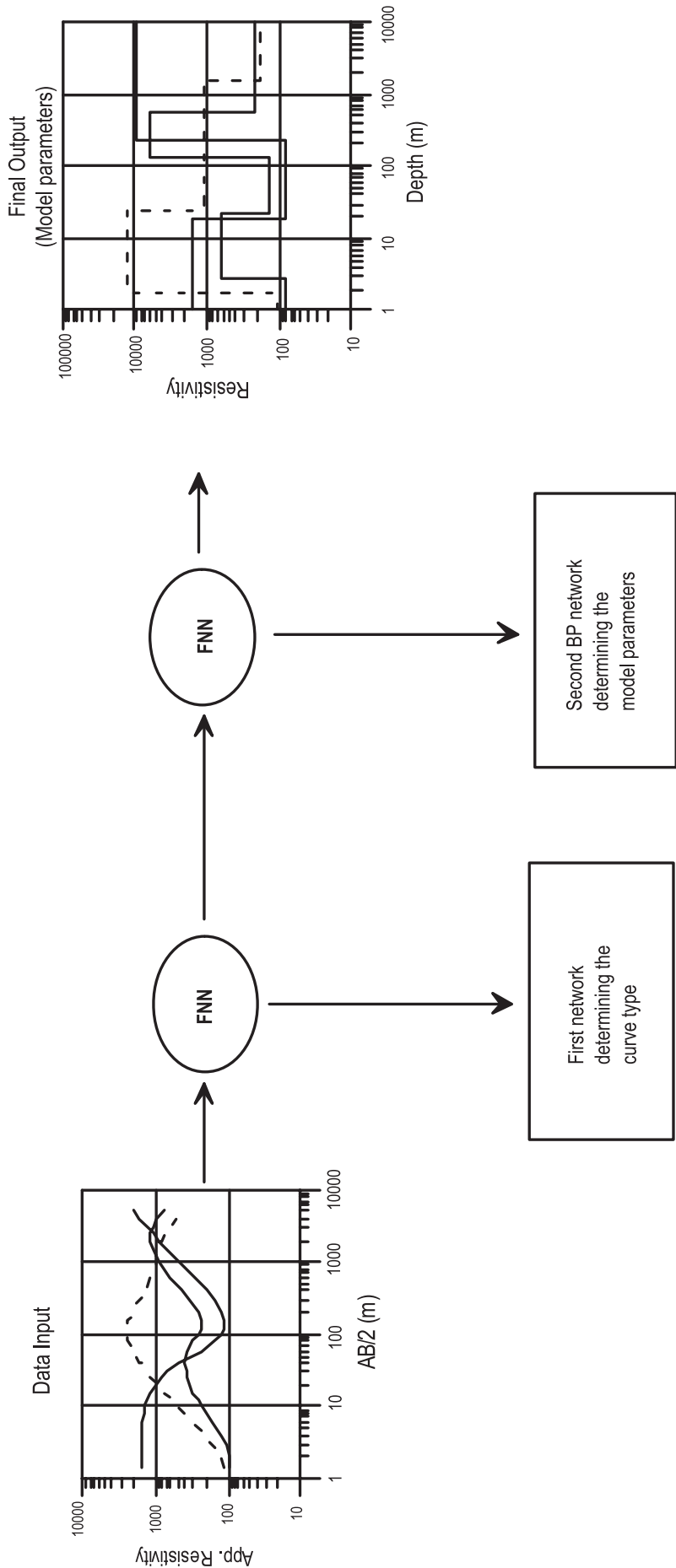


Figure 2. The 1D model parameter estimation scheme for VES data using artificial neural networks. First network identifies the apparent resistivity curve type and the second one estimates the model parameters.

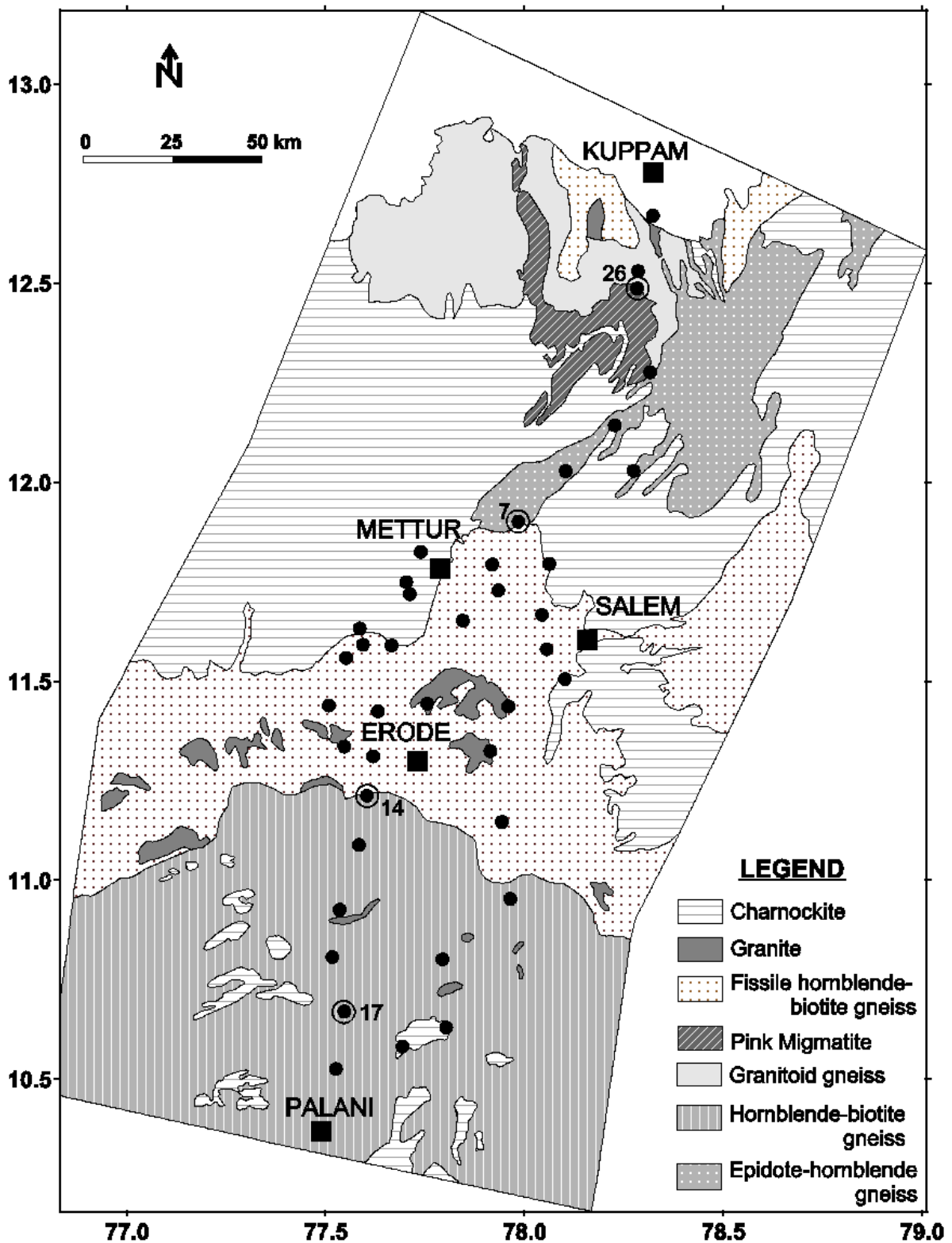


Figure 3. The distribution of DRS data points (solid circles) along with the geological setting of the granulite terrain of South India. The encircled symbols show the four test data used to analyze the network performance.

the apparent resistivities. The entire database is then remapped between zero and one. This normalization of inputs is essential as 0 and 1 are the limiting values of the sigmoidal transfer function of

neurons. There are two sets of output for a given input data, viz., curve identity and model parameters. The curve identity output vector is all zeroes but only one non-zero value, location of which tells

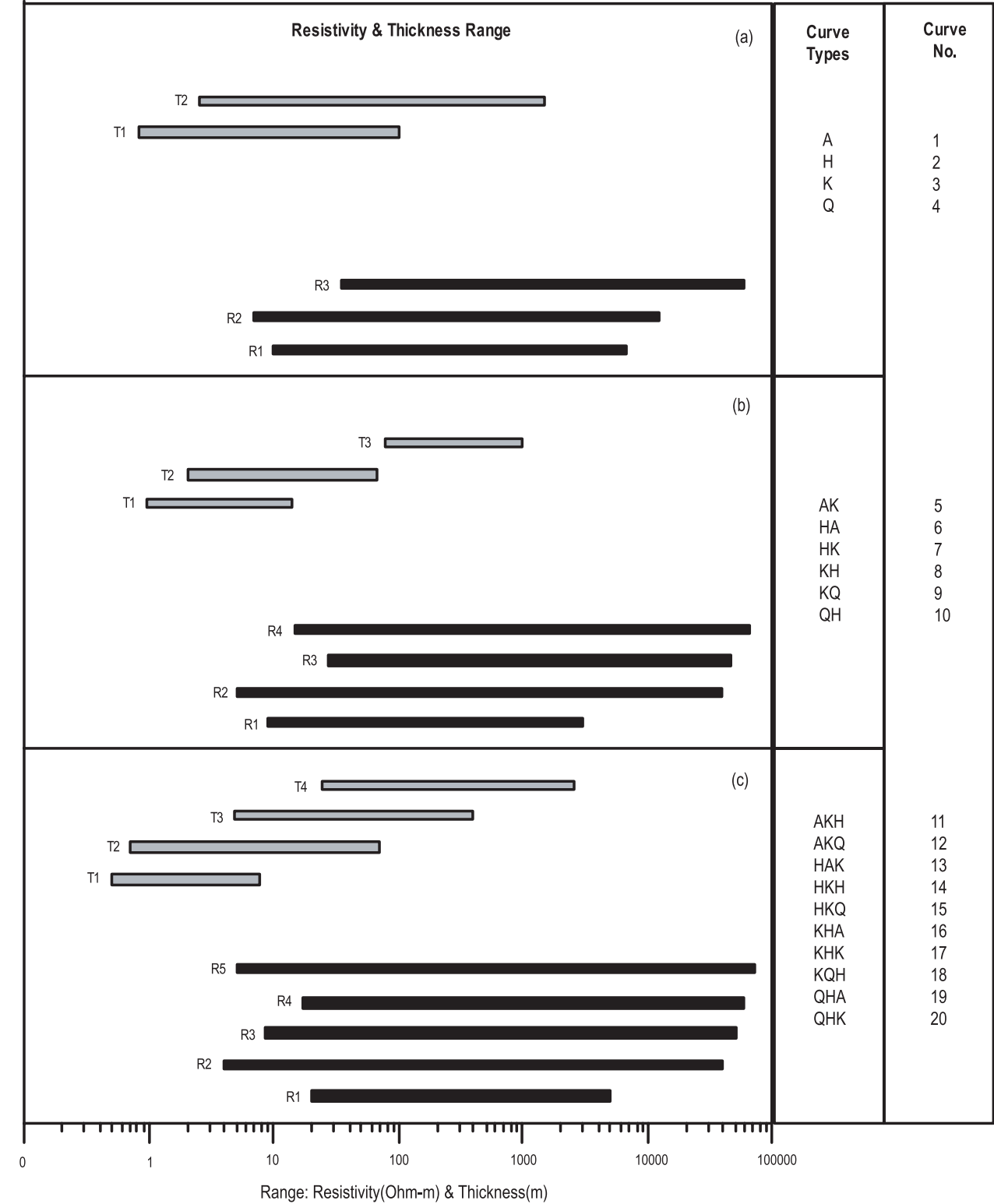


Figure 4. Range of model parameters, resistivity and thickness, used to make the database for network application are shown on the layered basis. (a) 3 layer models, (b) 4 layer models and (c) 5 layer models. Here, *R* and *T* represent the resistivity and the thickness of the individual layers.

the curve type. The model parameters are the resistivity and thickness itself. These model parameters were derived by careful modeling of the observed resistivity curve. Figure 5 shows the input and output patterns used in the present network, where

the symbols represent the neurons. The number of input neurons (i.e., the $AB/2$ spaces) was chosen to be 30 in all cases. But number of output neurons varies, with 20 for curve identification (i.e., the number of curve types used in the present study)

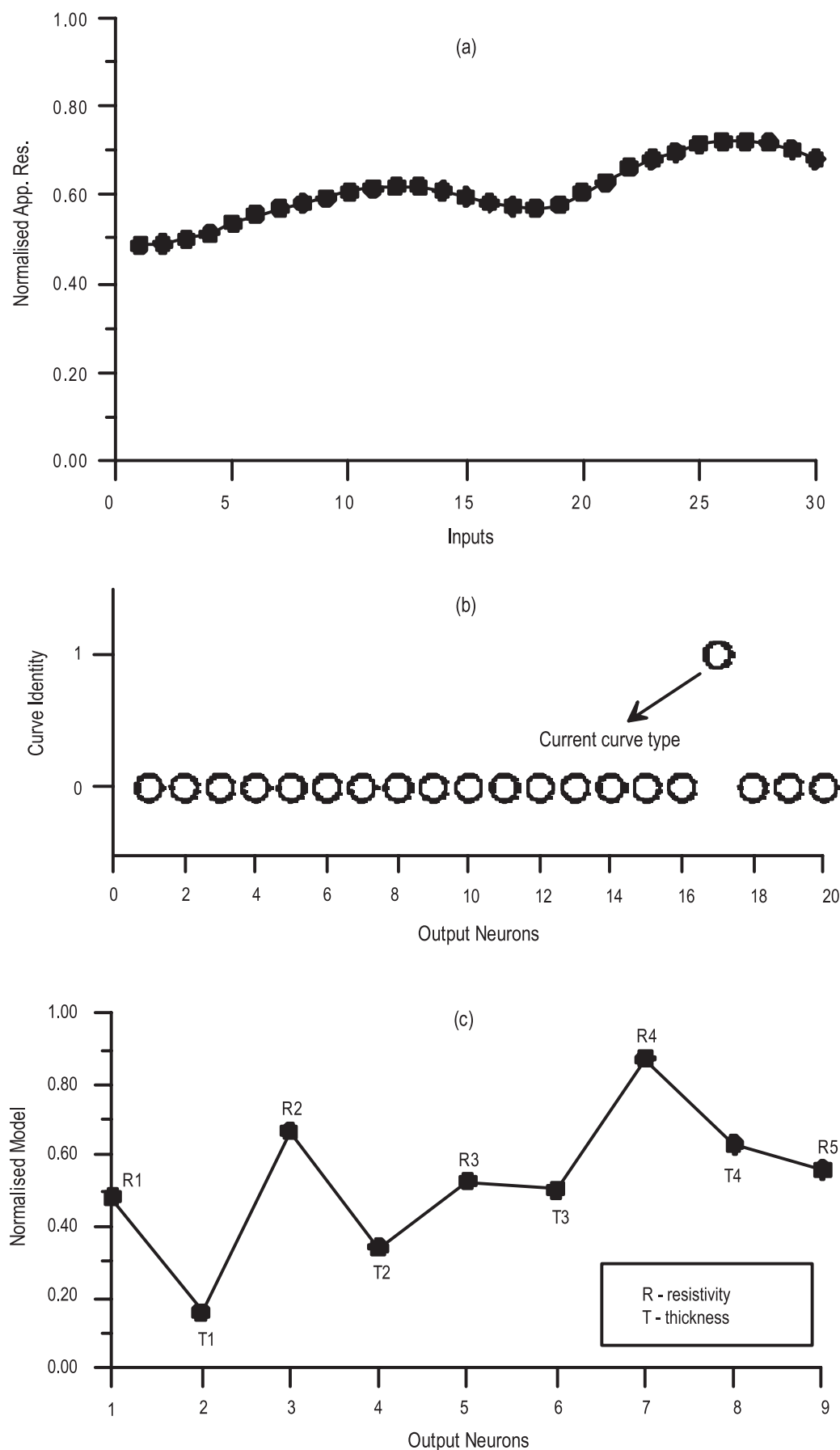


Figure 5. The normalized input (a) and output patterns (b & c) used for training the networks. (b) shows the binary output pattern used in the curve identification network and (c) shows the output pattern used in the inversion network.

Table 1. *Curve identification network parameters.*

No. of hidden neurons	=	30
Learning rate (LR)	=	0.001
LR increase	=	1.01
LR decrease	=	0.7
Sum-squared error goal	=	.01
Maximum error ratio	=	1.04
Momentum constant	=	0.9
Maximum epochs	=	20,000
Neuron's activation function	=	logistic sigmoid

and 5, 7 and 9 for 3 layer, 4 layer and 5 layer cases respectively (i.e., the total number of model parameters).

5. Inversion scheme procedures

5.1 First step: curve identification network

Curve identification is the first step for the ANN based inversion scheme. Training data example consists of apparent resistivities at 30 AB/2 and its

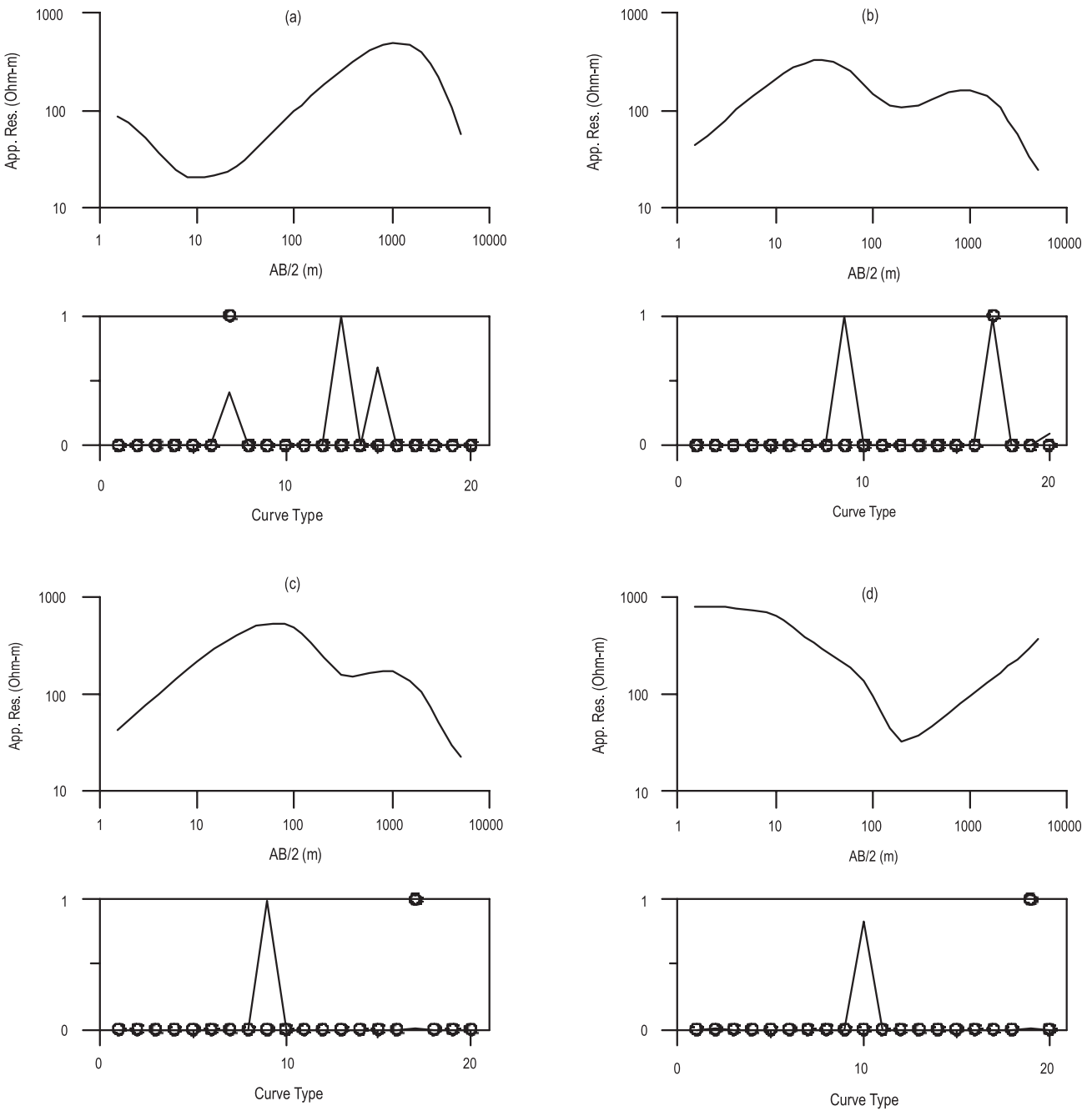


Figure 6. Problems due to the lack of training (a) and (b) and excess training (c) and (d).

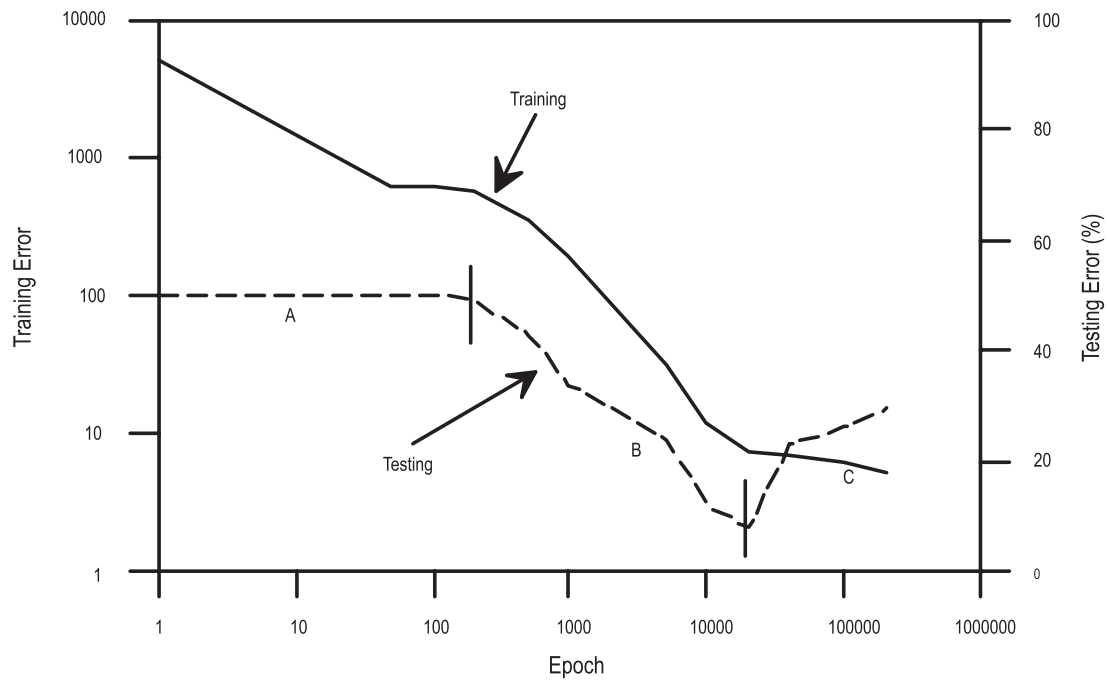


Figure 7. The performance of curve identifying network, showing the training error convergence and the error per cent on test samples.

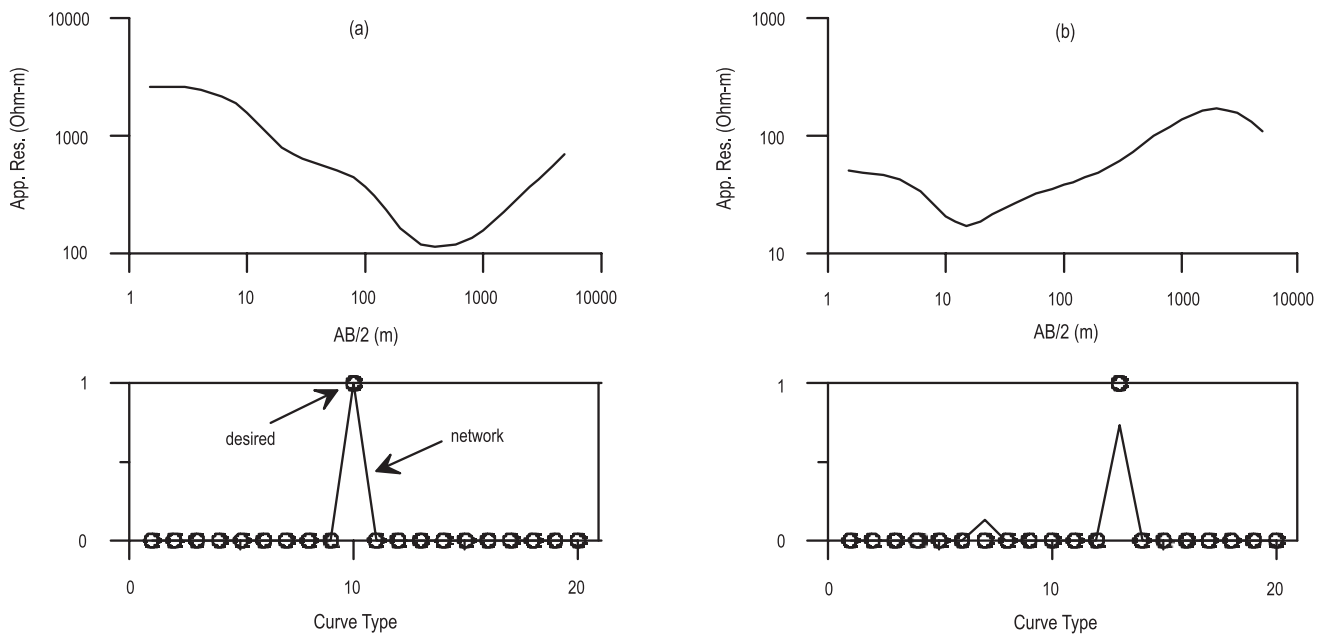


Figure 8. Examples of correct curve identification on training the network for the optimum number of epochs. The network performance is good, even if the network output is not the desired binary values of zeroes and one as seen in (b).

curve type. The database for training and testing consists of 900 examples selected from the original database, discussed in the earlier section, with 45 samples from each type giving true representation. The different network parameters like number of hidden layers and neurons, learning rate and momentum ratio are optimized in a trial and error basis. No appreciable change in network perfor-

mance (a function of error on training, testing and number of epochs taken for convergence) was found by using more than one hidden layer. 30 hidden neurons were found to be sufficient for the curve identification. Table 1 shows the network parameters used in the present study. Maximum epoch is set on the basis of training convergence as well as the testing performance.

Table 2. *Inversion network parameters.*

No. of hidden neurons:		
(a) 3 layer	=	30
(b) 4 layer	=	40
(c) 5 layer	=	45
Learning rate (LR)	=	0.01
LR increase	=	1.05
LR decrease	=	0.7
Sum-squared error goal	=	variable
Maximum error ratio	=	1.04
Momentum constant	=	0.9
Maximum epochs	=	variable
Neuron's activation function	=	logistic sigmoid

Experiments show that both under-training and over-training will result in the poor performance of network. Choosing appropriate number of epochs is a tricky business. Though training error is an indicator, it seldom helps as the network performance on novel dataset (which was not used in training) depends on another property of neural network, the ‘generalization ability’. A satisfactory convergence of the network on training data need not imply a good performance with the testing data. Some of the illustrations of the problems generally risen due to under-training and over-training is shown in figure 6. An under-trained network

may fail completely on novel data. As illustrated in figure 6(a), the desired output is HK (7), but the network output classifies it more strongly as the HAK (13) type and also gives some values for HKQ (15) and HK types. In another case shown in figure 6(b), the same input for a KHK (17) type is also classified into KQ (9), both giving the binary value. Both the above problems can be well solved by allowing the iteration to go few more times. But it is very important to understand the level up to which one can train a particular network with the given database. Because the network after considerable epochs of training could produce good results on training database, but fail on novel data set. Once we cross this optimum level, the network performance seems to be distorted as seen in figures 6(c) and (d). As the network is over-trained, the weights will try to adjust to the minor details of the training data set itself. It can be compared to the ‘over-fitting’ of a polynomial to a given set of noisy data. This unnecessary fitting will reduce networks performance on new data sets. In both the above cases it is noticeable that the network outputs more or less binary values but with wrong results. Here the KHK type is clearly outputted as a KQ type (c), where as earlier the network outputted the value 1 for both the types. Experiments show that at some optimum level of training, the same network performs more faithfully with a comparatively less error

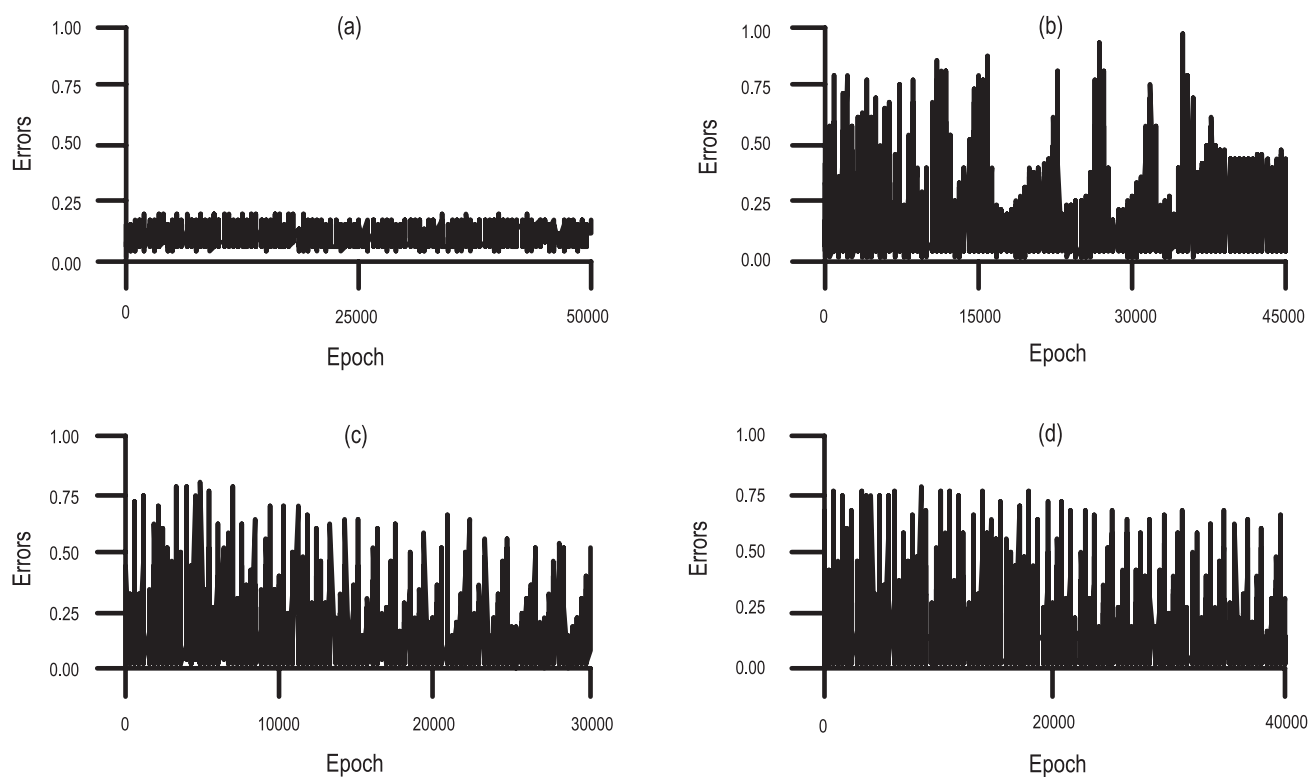


Figure 9. Nature of variations in the learning rate while training the inversion networks using fast back propagation scheme. Examples are shown for A (a), KQ (b), HAK (c) and QHA (d) curve type training networks.

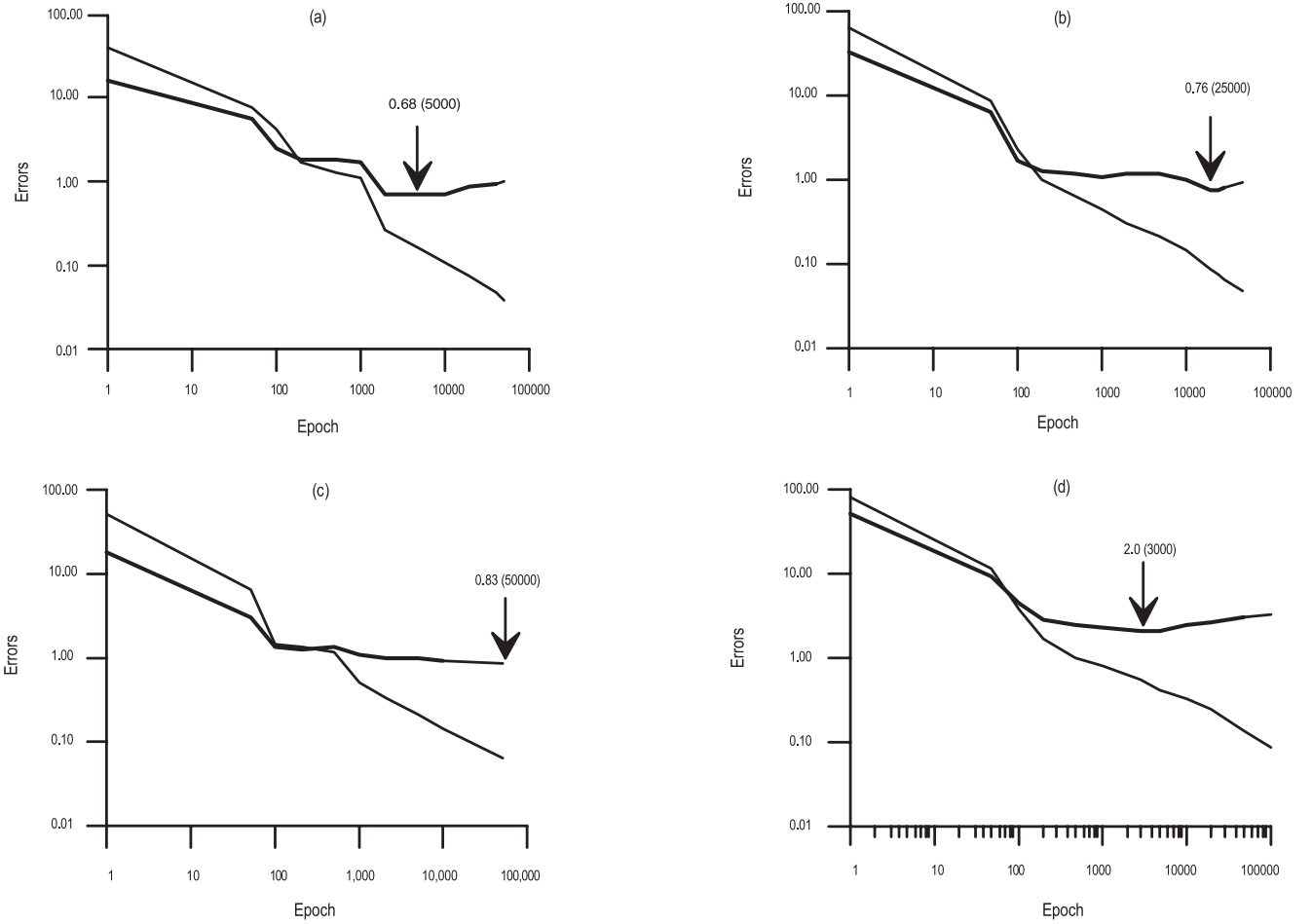


Figure 10. Diagram showing the network training error convergence (thin line) and testing performance (thick line) for different curve types. Figures (a & b) show the difference in convergence for the same network (KH type) due to different weight initializations. (c) shows the case of a progressive convergence (HK type) and (d) shows error on training with KHK curve type.

percentage. In figure 7, the sum squared error (SSE) on training and testing database is plotted against the training epoch itself. It is noticeable that the training error is continuously decreasing, but the testing error initially decreases; reaches a minimum at 20,000 epochs and thereafter increases again. At the optimum chosen level of 20,000 epochs, the network failure is only 2% (i.e., 6 out of 300). Figure 8 shows the network performance on optimal training.

5.2 Second step: inversion network

The inversion of DRS data or the parameter estimation is worked out in the second FNN working on the same architecture discussed earlier. This network differs only in the design of network parameters as well as the database. Apparent resistivity values from the existing database were used as input, whereas the output consists of its derived model. Here the entire 20 curve types discussed earlier were used. A total of 3000 examples were

generated with each type representing 100 and 50 examples for training and testing respectively, in the discussed range (figure 4). The network parameters such as learning rate, number of hidden neurons, momentum constant, etc. are optimized as we discussed earlier. A summary of the network parameters used for all the curve types are given in table 2.

A fast network convergence was obtained by varying the learning rate than keeping it constant throughout the network convergence. For example, in the case of a flat error plane, we can make fast learning by decreasing the learning rate. The initial learning rate and the increase and decrease in learning rate optimized for the fast network convergence are given in tables 1 and 2. The learning rate adjustments with respect to the weight adjustments during training are shown in figure 9, for selective network types (here the network type indicates the network for a particular curve type). It is evident that in case of less complex problems with fewer output units, the learning rate

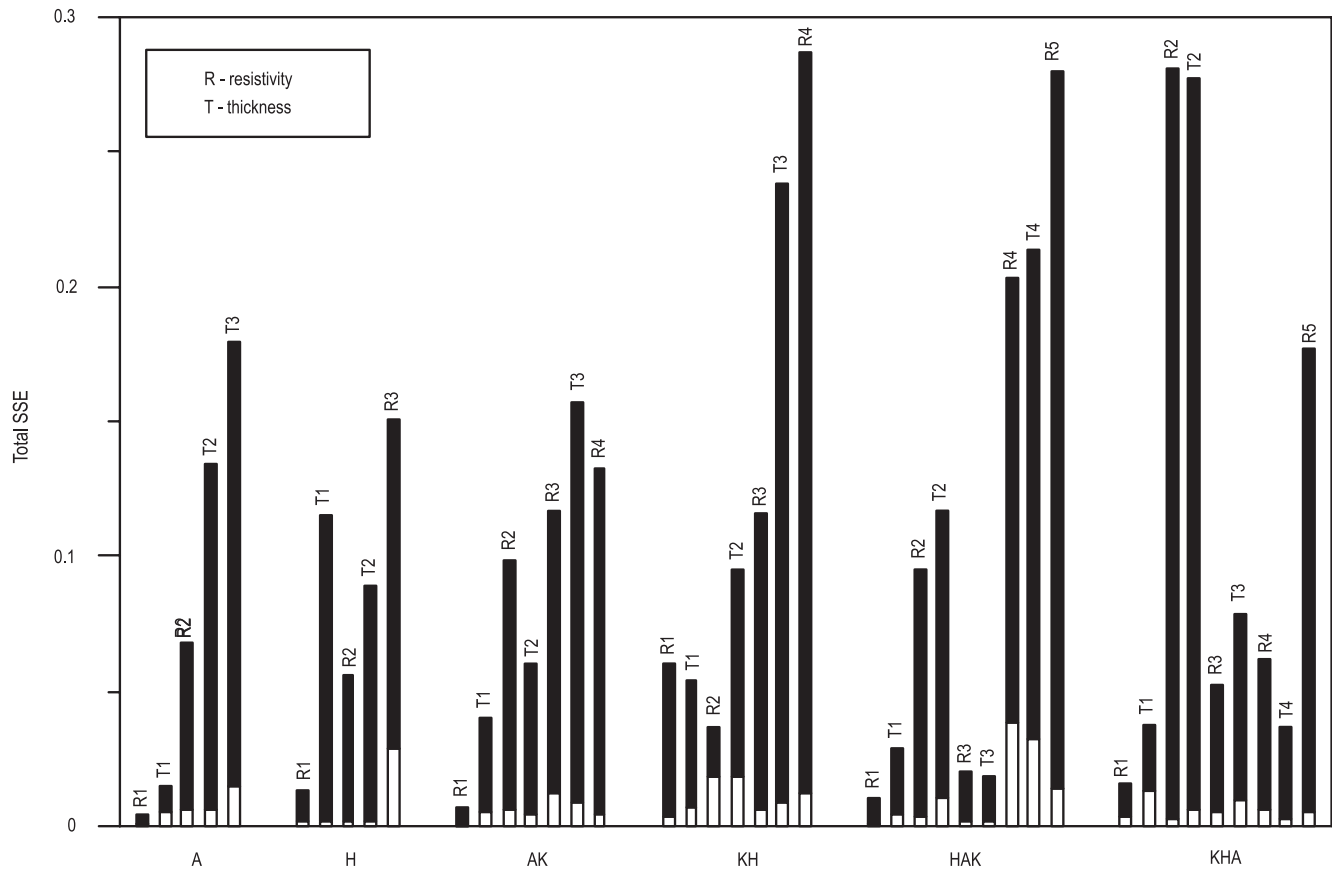


Figure 11. Bar chart showing the training (white bars) and testing (black bars) performance of networks for selective curve types. The training and testing errors (SSE) for the entire samples are shown for each model parameter.

variation is within a small range (figure 9a). The training characteristics for some of the selected networks are shown in figure 10, where both the training and testing errors are plotted against the epochs. Figure 10 (a) and (b) demonstrates the importance of a proper weight initialization to start with the training procedure, though there doesn't exist any mechanism for a correct initialization other than a trial and error. It illustrates the performance of a network for two different sets of initial weights and biases. Figure 10(c) shows the case of a progressive error convergence in both training and testing, observed for a typical 3 layer resistive structure. Error convergence on these networks is fast because training and testing error falls with larger epochs. The last case shown (figure 10d) is more complex since the curve type is two orders high. Figure 10(d) depicts the most difficult case (KHK type) out of the 10 networks designed for 5 layer resistivity structures. The solid arrow in these figures show the optimum level of training we selected in each network. The errors shown are the total sum-squared errors for the entire training (100 samples) and testing (50 samples) data. Though we prefer the optimum network training with reference to the test per-

formance, care has also been given to attain a minimum training SSE of 0.5 (i.e., 0.005 SSE per sample for entire model parameters) on training data.

The network performance for new data set is analyzed. We try to see the error statistics for the resistivity and thickness for all layers separately, to look into the performance on each parameter of all the curve types. Thereby the adaptability of the network is tested. Figure 11 shows the training (white bars) and testing (black bars) errors for few selected curve types. Here, the sum-squared error of each model parameter calculated over the entire samples used for training and testing are shown. The detailed list of network errors is given in table 3. Good results are observed with the 3 layer type curves such as *K* and *Q* with very small network errors. At the same time, the network errors rise with higher order of layers as obtained in the case of KHK curve type (see table 3). In all the cases it is observed that the testing error on new examples will be much greater than the training errors as visible in the bar chart (figure 11). In fact, these errors for model parameters are well within an allowable limit of a conventional inversion scheme.

Table 3. Error statistics for training (TR) and testing (TS).

Curve type	Layer 1		Layer 2		Layer 3		Layer 4		Layer 5
	R1	T1	R2	T2	R3	T3	R4	T4	
A	TR 0.0010	0.0051	0.0059	0.0065	0.0145				
	TS 0.0033	0.0101	0.0620	0.1277	0.1653				
H	TR 0.0016	0.0018	0.0018	0.0020	0.0286				
	TS 0.0118	0.1137	0.0538	0.0866	0.1221				
K	TR 0.0006	0.0056	0.0149	0.0121	0.0010				
	TS 0.0056	0.0092	0.0223	0.0241	0.0186				
Q	TR 0.0042	0.0058	0.0014	0.0028	0.0008				
	TS 0.0264	0.0511	0.0241	0.0298	0.0075				
AK	TR 0.0004	0.0048	0.0062	0.004	0.0121	0.0084	0.0041		
	TS 0.0066	0.0352	0.0922	0.0560	0.1046	0.1482	0.1284		
HA	TR 0.0014	0.0050	0.0035	0.0059	0.0023	0.0022	0.0318		
	TS 0.0258	0.0194	0.0974	0.1054	0.0198	0.0230	0.6299		
HK	TR 0.0010	0.0033	0.0012	0.0021	0.0216	0.0094	0.0236		
	TS 0.0151	0.0142	0.0540	0.0630	0.2299	0.2944	0.2035		
KH	TR 0.0031	0.0068	0.0184	0.0183	0.0063	0.0086	0.0118		
	TS 0.0569	0.0471	0.0186	0.0766	0.1100	0.2294	0.2750		
KQ	TR 0.0015	0.0061	0.0650	0.0622	0.0015	0.0003	0.0001		
	TS 0.0082	0.0171	0.5306	0.1867	0.0088	0.0559	0.0830		
QH	TR 0.0005	0.0016	0.0006	0.0012	0.0029	0.0031	0.0201		
	TS 0.0126	0.0015	0.0133	0.0194	0.0194	0.0491	0.7253		
AKH	TR 0.0009	0.0058	0.0028	0.0057	0.0031	0.0066	0.0034	0.0027	0.0087
	TS 0.1364	0.0946	0.1729	0.4131	0.1656	0.3314	0.0880	0.0522	0.5701
AKQ	TR 0.0016	0.0136	0.0022	0.0039	0.0252	0.0065	0.0012	0.0012	0.0014
	TS 0.0098	0.1366	0.0214	0.0458	0.1673	0.1842	0.0058	0.0188	0.0305
HAK	TR 0.0006	0.0045	0.0033	0.0102	0.0021	0.0022	0.0384	0.0322	0.0142
	TS 0.0102	0.0242	0.0920	0.1069	0.0176	0.0157	0.1650	0.1815	0.2661
HKH	TR 0.0006	0.0076	0.0064	0.0155	0.0051	0.0044	0.0012	0.0027	0.0007
	TS 0.0192	0.0360	0.0660	0.0520	0.1520	0.1295	0.0954	0.0638	0.2458
HKQ	TR 0.0006	0.0071	0.0020	0.0045	0.0067	0.0057	0.0010	0.0043	0.0131
	TS 0.0064	0.0456	0.0978	0.1576	0.1799	0.1273	0.0309	0.0511	0.0097
KHA	TR 0.0032	0.0128	0.0030	0.0061	0.0053	0.0098	0.0057	0.0025	0.0049
	TS 0.0122	0.0249	0.2782	0.2714	0.0467	0.0688	0.0563	0.0344	0.1718
KHK	TR 0.0008	0.0041	0.0464	0.0365	0.0132	0.0154	0.0282	0.0278	0.0238
	TS 0.0597	0.0932	0.0984	0.1255	0.2787	0.2355	0.4874	0.6492	0.5029
KQH	TR 0.0012	0.0069	0.0025	0.0031	0.0008	0.0015	0.0025	0.0032	0.0183
	TS 0.0315	0.0339	0.0399	0.0509	0.0199	0.0096	0.0232	0.0764	0.3212
QHA	TR 0.0006	0.0135	0.0006	0.0014	0.0060	0.0136	0.0025	0.0020	0.0214
	TS 0.0044	0.0359	0.0045	0.0092	0.0138	0.0600	0.0098	0.0452	0.4385
QHK	TR 0.0004	0.0016	0.0005	0.0007	0.0008	0.0009	0.0042	0.0030	0.0182
	TS 0.0119	0.1101	0.1359	0.1764	0.0701	0.0250	0.3682	0.4997	0.9863

6. Application of ANN

The performance of trained network is tested on four new DRS data over the SGT. The locations of these data are also shown in figure 3. These four curves are selected from different geologic provinces

to represent the entire terrain. The complex resistive structures in the crystalline rock terrains like SGT are very well reflected in the observed apparent resistivity curves over SGT. The DRS curves and model parameters derived using the present ANN scheme as well as a conventional inversion

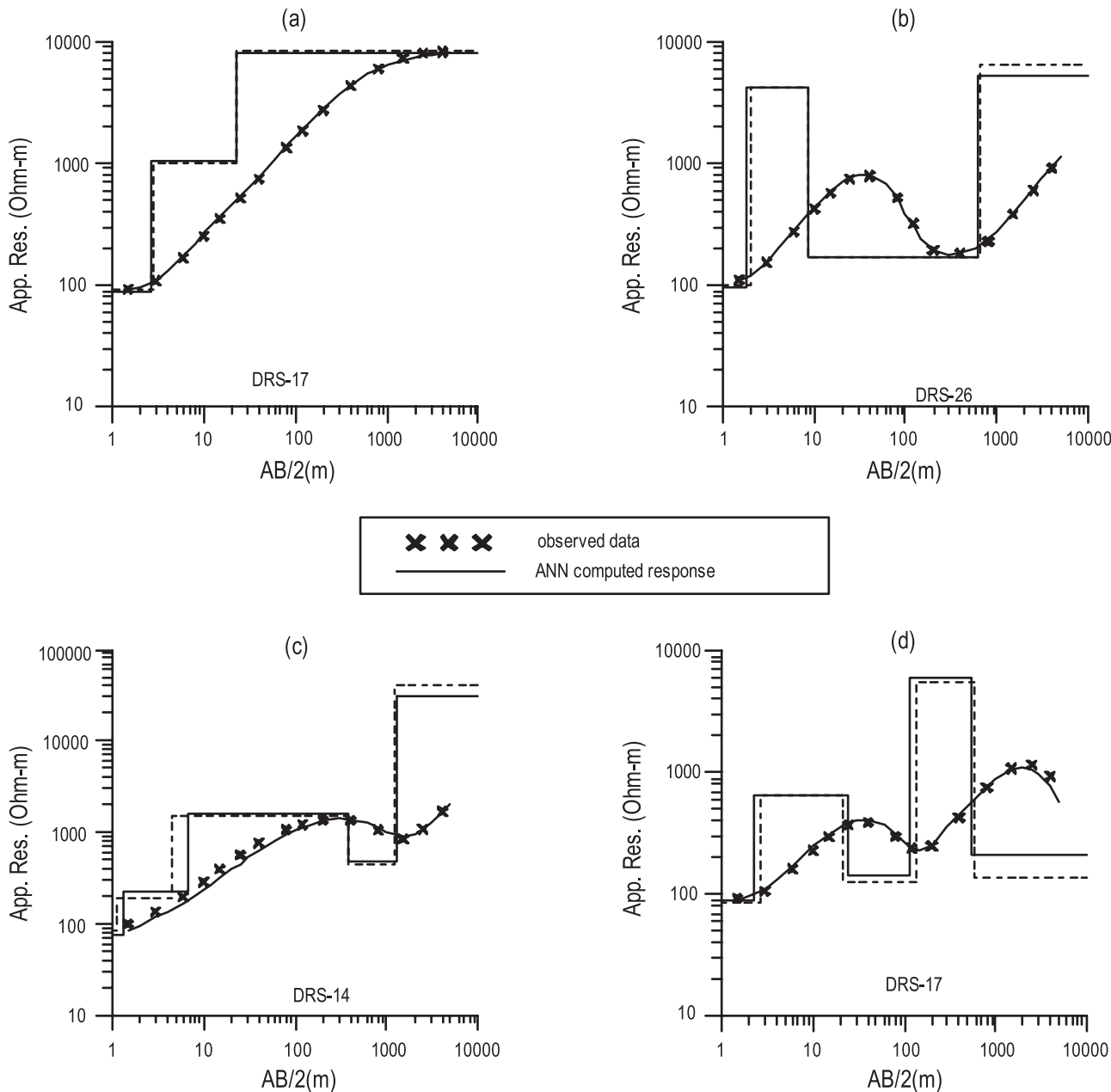


Figure 12. The ANN results for different field DRS data collected over the crystalline terrain of south India. The model obtained by conventional inversion scheme (dashed line) and the ANN application (solid line) are shown for A, KH, AKH and KHK in (a), (b), (c) and (d) respectively.

scheme are shown in figure 12. The inverted models show resistive upper crust with resistivities of the order of few thousands of Ohm-m as seen in the deeper parts with intermediate low resistive zones. Wherein, the DRS-07 near Mettur, marked by geologic boundaries, shows low resistivities even to the higher depths (figure 12d). The model parameters are well within the resolvability limits of resistivity technique and show a comparable match with the results of a conventional method as seen in the figure. The currently proposed ANN based inversion (shown in

solid line), resulted in similar models well within about 10% error limit.

7. Conclusions

The conventional algorithm was constrained by the requirement of an initial model, where a good initialization results with good results. Whereas, the ANN based inversion resulted in a geologically plausible resistivity model, without any *a priori* information, making use of the internal constraints

stored as weights. These weights themselves are inherited from the training.

In this paper we have shown the adaptability of ANN in direct parameter estimation from DRS data on a crystalline terrain involving a variety of curve types with a wide range of model parameter values. Though the problem gets complex with increase in layers, it is found that a proper network design can solve it, provided a good representative database for the training. The inherent problems of ambiguities, especially the equivalence problem in the case of 1D electrical sounding methods seem to be affecting the ANN inversion to some extent. We restrict to carefully selected 100 examples for training each network. The present approach of two-step network parameter estimation performs effectively and once trained, the model parameters are calculated fast on simulating the network with saved weights and biases on new examples. The network also achieved the generalization capability, but within the limits of the presented model parameter ranges representing a typical crystalline terrain. With due representation in a wider range or with other selective terrain ranges, the scope of the technique is commendable. The technique can also be developed for assisting with first hand information for the model initialization in other conventional inversion schemes. This will help in a greater extent for on-site interpretation of DRS data.

Acknowledgements

The authors are thankful to Dr. V P Dimri, Director, NGRI, for granting permission to publish this work. They also thank the reviewers for their constructive suggestions, which helped to further improve the paper. Mr. Jimmy Stephen, greatly acknowledges the Council of Scientific and Industrial Research (CSIR) for its Senior Research Fellowship. Thanks are also due to all field participants of the DRS Group.

References

- Bishop C M 1995 *Neural networks for pattern recognition* (Clarendon Press)
- Blum A L and Rivest R 1992 Training a 3-node neural network is NP-complete; *Neural Networks* **5** 117–128
- Caudill M 1988 Neural networks primer, Part 4; *AI Expert* **8** 61–67
- Drury S A, Harris N B W, Holt R W, Reeves-Smith G J and Wightman R T 1984 Precambrian tectonics and crustal evolution in South India; *J. Geophys.* **92** 3–20
- Fahlman S E 1989 Fast learning variations in backpropagation: An empirical study; In: Proc. 1988 connectionist models summer school (eds) D S Touretzky, G E Hinton and T J Sejnowski; *San Mateo, CA: Morgan Kaufmann* 38–51
- Ghosh D 1971 The application of linear filter theory on the direct interpretation of geoelectrical resistivity sounding measurements; *Geophys. Prosp.* **19** 192–217
- Grady J C 1971 Deep main faults in South India; *J. Geol. Soc. India* **17** 56–62
- Guyon I 1991 Applications of neural networks to character recognition; *Int. J. Pattern Recognition and Artificial Intelligence* **5** 353–382
- Haykin S 1994 *Neural Networks: A comprehensive foundation* (New York: Macmillan College Publishing Company Inc.)
- Horne S and MacBeth C 1994 Inversion of seismic anisotropy using genetic algorithms; *Geophys. Prosp.* **42** 953–974
- Hush D R and Horne B G 1993 Progress in supervised neural networks: Whats new since Lippmann; *IEEE Signal Processing Magazine* **10** 8–39
- Judd J S 1990 *Neural network design and the complexity of learning* (Cambridge, MA: MIT Press)
- Kirkpatrick S, Gelatt Jr C D and Vecchi M P 1983 Optimization by simulated annealing; *Science* **220** 671–680
- Kunetz G and Rocroi J 1970 Automatic processing of electrical soundings; *Geophys. Prosp.* **18** 157–198
- Kung S and Hwang J 1988 An algebraic projection analysis for optimal hidden units size and learning rates in back-propagation learning; In: *1st Int. Conf. on Neural Networks, IEEE Proc.* (eds) M Caudill and C Batler, SoS Printing, San Diego, I-363–370
- Lippmann R P 1987 An introduction to computing with neural nets; *IEEE ASSP Magazine* **4** 4–22
- Luo F and Unbehauen R 1997 *Applied neural networks for signal processing* (Cambridge, UK: Cambridge University Press) 123 p
- Macias C C, Sen M K and Stoffa P L 2000 Artificial neural networks for parameter estimation in geophysics; *Geophys. Prosp.* **48** 21–47
- Manoj C and Nagarajan N 2003 The application of artificial neural networks to magnetotelluric time-series analysis; *Geophys. J. Int.* **153** 409–423
- Meheni Y, Guerin R, Benderitter Y and Tabbagh A 1996 Surface DC resistivity mapping: approximate 1D interpretation; *J. Appl. Geophys.* **34** 255–270
- Paulton M, Stenberg B and Glass C 1992 Location of subsurface targets in geophysical data using neural networks; *Geophysics* **57** 1534–1544
- Plaut D S, Nowlan S J and Hinton G E 1986 Experiments on learning by back propagation; *Tech. Report CMU-CS-86-126*, Dept. of Computer Science, Carnegie-Mellon University, Pittsburgh, PA.
- Qady G E and Ushijima K 2001 Inversion of DC resistivity data using artificial networks; *Geophys. Prosp.* **49** 417–430
- Reddy P R, Rao V D, Rao Y J B, Mishra D C, Harnarayana T, Singh S B and the DCS Research Group of NGRI 2001 Modeling the tectonic evolution of southern granulite belt of the Indian shield using coincident seismic reflection/refraction, geological/geochemical, geochronological, gravity/magnetic, magnetotelluric and deep resistivity studies along the southern geotranssect; *Tech. Report NGRI-2001-EXP-317*, National Geophysical Research Institute, Hyderabad, India
- Rubinstein R Y 1981 *Simulation and the Monte Carlo method* (New York: John Wiley and Sons) 278 p
- Rumelhart D E, Hinton G E and Williams R J 1986 Learning internal representations by error propagation in parallel distributed processing: explorations in the microstructure of cognition (eds) D E Rumelhart,

- J L McClelland & the PDP Research Group (Cambridge, MA: MIT Press) **1** 318–362
- Schiffman W, Joost M and Werner R 1992 Optimization of the backpropagation algorithm for training multiplayer perceptrons; Technical report, Institute of Physics, University of Koblenz
- Singh S B, Ashok Babu G, Singh K P, Srinivas Y, Stephen J, Singh U K and Reddy J 2000 Deep resistivity sounding studies in the South Indian granulite terrain along Kuppam-Palani geo-transect; *Indian Mineralogist* **34** 48–51
- Singh S B, Stephen J, Singh U K, Srinivas Y, Ashok Babu G, Singh K P and Reddy J 2003 Electrical signatures in the high-grade metamorphic terrain of South India using deep resistivity sounding studies; *Mem. Geol. Soc. India* **50** 125–138
- Spichak V and Popova I 2000 Artificial neural network inversion for magnetotelluric data in terms of three-dimensional earth macro parameters; *Geophys. J. Int.* **142** 15–26
- Werbos P J 1974 Beyond regression: New tools for prediction and analysis in the behavioral sciences; PhD thesis, Harvard University, Cambridge, MA
- Werbos P J 1990 Backpropagation through time, what it does and how to do it; *Proceedings of the IEEE* **78** 1550–1560
- Yegnanarayana B 2001 *Artificial neural networks*; (New Delhi: Prentice Hall of India Pvt. Ltd.) 461 p
- Zohdy A R 1989 A new method for automatic interpretation of Schlumberger and Wenner sounding curves; *Geophysics* **54** 245–253

MS received 9 June 2003; revised 22 July 2003