CHAPTER-5

IMPLEMENTATION OF GENETIC ALGORITHM

5.1 Geoelectrical Sounding

The main objective of resistivity sounding is to obtain the variation of electrical resistivity of the subsurface with depth by measuring the variation of surface potential. This is done by measuring the surface potential at different current electrode spacing. There are different types of electrode configurations to measure the surface potential. The most commonly used configuration is the Schlumberger configuration. This consists of four electrodes in a straight line on the surface, which is symmetric. Two potential electrodes (M and N) are inside and two current electrodes (A and B) are outside as shown in Figure 5.1.

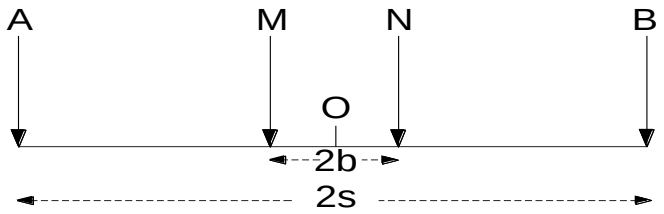

Figure 5.1 Schlumberger electrode configurations (Telford et al. 1976).

The potential, at the surface of a horizontally layered-Earth model, due to the point current source of intensity (I) is derived by Stefanesco (1930). It is given as (Sri Niwas and De Lima, 2006);

$$V = \frac{\rho_1 I}{2\pi} \int_0^\infty K(\lambda) J_0(\lambda r) d\lambda \qquad (5.1)$$

Where $K(\lambda)$ is known as the Stefanesco kernel function, $J_0(\lambda r)$ is the Bessel function of the first kind of zero order, ρ_1 is the resistivity of the first layer, and r is a distance of measuring point with respect to current electrode.

The apparent resistivity (ρ_{as}), for Schlumberger configuration, is derived by using equation (5.1), taken from the book 'Geosounding Principle 1' by Koefoed (1979), and is given as:

$$\rho_{as} = s^2 \int_0^{\infty} T(\lambda) J_1(\lambda s) \lambda d\lambda \tag{5.2}$$

Where T (λ) =ρ_1K (λ) is the resistivity transform function, $J_1(\lambda r)$ is the Bessel function of the first kind of order one, and s is half of the current electrode spacing.

The apparent resistivity, given in equation (5.2), is computed numerically using Ghosh's inverse filter method described in the book 'Geosounding Principle 1' by Koefoed (1979). This method consists of two steps.

Step 1- In the first step, the sample values of T(λ) are calculated for the given model using a suitable recurrence relation. Since the Pekeris recurrence relation is more straightforward than the Flathe recurrence relation, so Pekeris recurrence relation for N-layer is used and given as:

$$T_N = \frac{T_{i+1} + \rho_i \tanh(\lambda h_i)}{1 + T_{i+1} \tanh(\lambda h_i)/\rho_i} \tag{5.3}$$

Where T_N is the same as ρ_N, the resistivity of the bottom-most layer (half space), T_1 is the value of T(λ) at the earth's surface, and ρ_i is the resistivity of the ith layer, and h_i is the thickness of the ith layer.

Step 2- In this step, sampled values of apparent resistivity are calculated by convolving the sampled values of T(λ), calculated in the first step, with Ghosh's inverse filter coefficients as:

$$\rho_{ak} = \sum_j f_j T_{k-j} \tag{5.4}$$

Where f_j is Ghosh's inverse filter coefficients, and k stands for the sample

point of apparent resistivity (Koefoed, 1979). We have used six points per decayed Ghosh's inverse filter coefficients, given in the book 'Geosounding Principle 1' by Koefoed (1979), with equations (5.3) and (5.4), to develop a program for computing the apparent resistivity.

5.2 Use of Genetic Algorithm (GA)

The discussion given in this section is taken from the books by Goldberg (1989); Sen and Stoffa (2013); and the research paper by Parker (1999). The Genetic Algorithm (GA) was first proposed by John Holland (1975); and the first time it was used by Sen and Stoffa in Geophysics. Genetic Algorithms (GAs) are nonlinear global optimization methods. Its computational technique is based on a strong analogy with the process of biological evolution proposed by Darwin. So, GAs may be called simulated evolution, like simulated annealing.

To understand biological evolution, let us suppose a population of different creatures (or individuals) at a time. As time passes, these creatures struggle with each other and their environment for survival. Those who survive with ecology remain alive. This process is called natural selection, and they go through reproduction, natural mutation, and so on. After a long time in geological time scale, a new population is generated by repeating these processes. This new population has better survival than the initial population. These processes of biological evolution are implemented in Genetic Algorithm (GA) by using some analogies, given in **Table 5.1**, Discussed as follows:

Table 5.1 Some analogies between biological evolution and Genetic Algorithms

Biological evolution	**Genetic Algorithm (GA)**
Population	Set of models (or set of solutions)
Individual	The solution to a problem
Fitness	Quality of solution
Chromosome	Encoded solution (or binary string)
Gene	Part of the encoded solution (or binary string)
Natural selection	selection
Reproduction	Crossover
Natural mutation	Mutation

In Genetic Algorithms (GAs), binary strings (called chromosomes) are generated randomly. A single binary string represents a single model. The set of binary strings is analogous to the population of different creatures in biological evolution, and these binary strings are analogous to the individuals. Each binary string of population is decoded into real values of model parameters, and fitness values are evaluated by using the fitness function for each model in the population. This fitness value is analogous to the survival level of the individual. Now models are selected from the population in proportion to their fitness values. This process is analogous to natural selection in biological evolution. These selected models go through crossover and mutation, and a new population with more average fitness than the initial population is generated. Crossover and mutation are analogous to reproduction and natural mutation in biological evolution. These processes are repeated until the termination criteria are not achieved. Finally, a population with the best fitness value is achieved (details of the flow chart of the algorithm are shown in Figure 5.2).

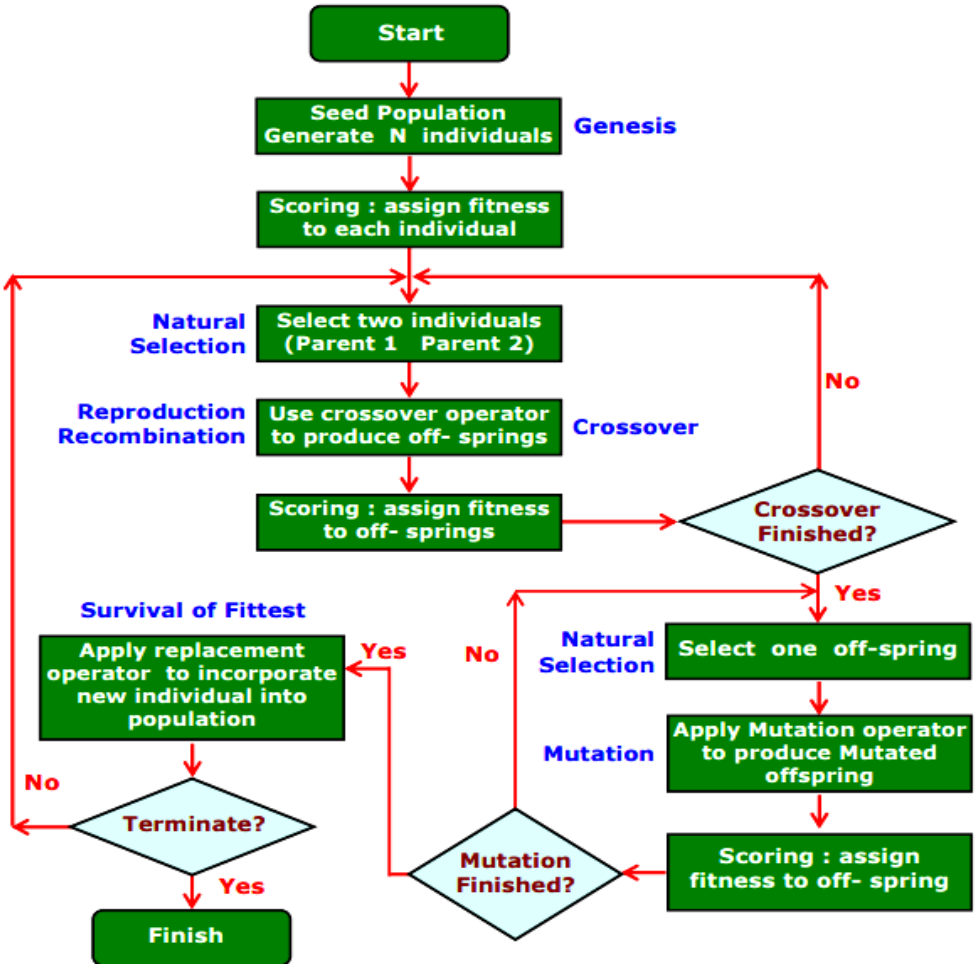

Figure 5.2 General flow chart of Genetic Algorithm (GA).

(Source:http://www.myreaders.info/08_Fundamentals_of_Genetic_Algorithms.pdf)

5.2.1 Main features of the developed program

A simple Genetic Algorithm (SGA) is implemented as a program in MATLAB. For solving any global optimization problem, the Simple Genetic Algorithm (SGA) involves somestandard basic steps. These steps are given as follows:

1. Defining search space.
2. Coding the models and constructing the objective and fitness functions.
3. Generating an initial population of random models.
4. Decoding and evaluating the fitness of each model.

5. Selection
6. Crossover
7. Mutation
8. Overwriting the old generation.
9. Stretching

This developed program for Simple Genetic Algorithm (SGA) can be used for any global optimization problem. In our work, this program is used for the inversion of 1-D Geoelectrical data. This program has the main file "ga" and several function files. Now we will explain the above steps used in the program, with a discussion of the main features of the program.

1. Defining search space and generating a random population

Defining a suitable search space for each model parameter is important to minimize computational time and maximize program efficiency. A suitable search space must contain the optimal global solution to the problem. Some crude guesses solution to the problem defines a suitable search range for models. Since the physical property of the formation changes from place to place, a preliminary idea about the formation helps to set the search space. Previous survey output/ observations and borehole data might help define the search space more closely to actual values. Initially, a crude range is used, and output is used as input for the next iteration. After a few iterations, the search space becomes precise.

In this coding scheme, Search space is limited by combining the minimum value of each model parameter, desired resolution, and the number of bits for each model parameter. All these entries are made while running the *'GA'* code. Initially, a sufficient number of the model (Population = pop), say 100, are randomly generated

in the binary string. The length of a binary string solely depends on the sum of bit numbers per model parameter.

---

2. Decoding

Once the random population is generated, a code is needed to decode the binary stings into respective model parameters i.e., resistivity and thickness of layers. Combining minimum values, resolution, and decimal value of the string, the binary string of a model is decoded into decimal values. The MATLAB code for decoding is given below:

```
function [ M ] = decode( S,npr,nbp,mpv,res )
for j = 1:nbp(1)
    w(j,:) = 2^(nbp(1)-j);
end
M1 = (res(1)*S(1:nbp(1))*w)+mpv(1);
M1;
for i = 2:npr
    clear w
    for j = 1:nbp(i)
        w(j,:) = 2^(nbp(i)-j);
    end
     n1 = sum(nbp(1:(i-1)))+1;
     n2 = sum(nbp(1:i));
     M2(i-1) = (res(i)*(S(n1:n2)*w))+mpv(i);
end
M2;
M=[M1 M2];
```

---

3. Response and Resistivity transform

Once decoding is done, response or synthetic data for each model is generated to compare it with observed data. The coding for response includes the calculation of resistivity transform and apparent resistivity.

The code for resistivity transform is given as:

```
function [ T1] = resistivitytransform( M ,L )
N=(length(M)+1)/2;
r=M(1:N);
t=M(N+1:(2*N)-1);
Told=r(N);
for i=N-1:-1:1
    Tnew=(Told+(r(i)*tanh(L*t(i))))/(1+(Told*tanh(L*t(i)))/r(i));
    Told=Tnew;
end
T1=Told;
end
```

The code for response is given as:

```
function [ RHOA ] = responsen( M,s )
x0=log(s);
dely=log(10)/6;
fc=[0.003042 -0.001198 0.01284 0.02350 0.08688 0.2374 0.6194 1.1817
0.4248 -3.4507 2.7044 -1.1324 0.3930 -0.1436 0.05812 -0.02521 0.01125
-0.004978 0.002072 -0.000318];
abs=-1.7881;
m=length(s);
n=length(fc);
for i=1:m
    y0=x0(i)+dely-abs;
```

```
    for j=1:n
        y=y0-(j*dely);
        L=exp(-y);
        [ T1 ] = resistivitytransform( M ,L );
        T(j,:)=T1;
        u(i,j) = x0(i)-y;
    end
    RHOA(:,i)=fc*T;
end
end
```

---

4. Constructing the objective and fitness functions

A binary string is like a biological chromosome in which a binary number represents each gene. Similar to a gene, the binary value in the string determines its closeness to the optimum solution. A random population of models is generated by combining the total string length using the round & rand function of MATLAB. Each string may represent an optimum solution to the problem. The effectiveness of each string is tested through the objective and fitness function of the problem. Since each string of the population is the competitor of others, the selection of a string is based on its fitness value. The fitness value of the model is determined through Relative Root Mean Square Error.

The objective function as Relative Root Mean Square Error (RRMSE) is given as:

$$RRMSE(m) = \sqrt{\frac{\sum_{i=1}^{n}\left(\frac{\rho_{aoi}-\rho_{aci}}{\rho_{aoi}}\right)^2}{n}} \qquad (5.5)$$

and the fitness function F(m) is defined as:

$$F(m) = \left(\frac{1}{1 + RRMSE}\right)^2 \quad (5.6)$$

Where ρ_{aoi} & ρ_{aci} represent the observed apparent resistivity & calculated apparent resistivity for model (m), respectively, at different electrode spacing, and n represents the number of data points.

The code for the fitness function is given below:

```
function [ f ] = fitness( pop,data,npr,nbp,mpv,res,s )
[m,n]=size(pop);
for i=1:m
    [M]=decode(pop(i,:),npr,nbp,mpv,res);
    [RHOA]=responsen(M,s);
    rrmse=sqrt(sum(((data-RHOA)./data).^2)/length(data));
    f(i,:)=1/(1+rrmse);
end
end
```

---

5. Selection

This is the first genetic operator. This process is applied after evaluating the fitness values of each model in the initial population. This is analogous to the natural selection in biological evolution. As the populations of individuals with higher surviving ability increase in the total population of individuals after natural selection, similarly, the populations of individual strings with higher fitness values are increased by copying the individual strings in proportional to their fitness values. So, it is clear that the individual strings with higher fitness values have a higher probability of selection in the next generation and can be copied more than times as offspring. There

are many types of selection processes. We have implemented the *roulette wheel selection* method in the program as a function file 'selection'.

In roulette wheel selection, the wheel is divided into some slices with different areas. The areas of slices are proportional to the probability of the selection of models with different fitness values. The probability of selection is proportional to the fitness of corresponding models. We have used the following probability (*Ps* (*mi*)) of selection.

$$P_s(m_i) = \frac{F(m_i)}{\sum_{j=1}^{n} F(m_j)} \tag{5.7}$$

A point indicator is fixed near the circumference of the wheel. This wheel is rotated and the point indicator falls into a slice when the wheel stop then the model corresponding slice is selected. This process is done as many times as the number of offspring is required in the new population.

The roulette wheel selection method is implemented in the program by choosing an individual randomly from the population and the probability (*ps*(***mi***)) of selection to the corresponding model is evaluated. A random number (***r***), between zero and one is generated. If (*ps*(***mi***)) is greater than that of (***r***) then the model is selected for a new population otherwise discarded. This process is done up to which a predefine population size is not achieved.

The code for selection is given below:

```
function [ pops ] = selection( pop,f,T )
 [m, n] = size(pop);
ef = exp(f/T);
```

```
num = 1;
while num <= m;
    r = randi(m,1);
    prob = ef(r)/sum(ef);
    s = rand(1,1);
    if prob >= s
        pops(num,:) = pop(r,:);
        num = num + 1;
    end
end
end
```

---

6. Crossover

This second genetic operator is responsible for the performance of the algorithm for optimization problems by generating a new model with better fitness. Crossover is analogous to reproduction in biological evolution. As the genetic material (gene) of two chromosomes of the parents are combined to form new offspring. Similarly, crossover combines the genetic material (bits) of two randomly chosen individual strings are combined to form two new offspring. Although crossover is a necessary step but too much crossover loses important information. So only some percentages of population passes through crossover to generate new offspring and the rest are directly copied to the next generation as offspring. How many individuals will go through crossover depends on crossover probability $\boldsymbol{P_c}$.

There are many crossover methods. Here we have implemented *single point crossover* in the program as a function file 'crossover'. In the single-point crossover, a pair of individuals, called parents are chosen randomly and a crossover point in between two bits is decided by generating an integer between one and the length

of the string minus onefrom a uniform distribution. Then all the bit values after this crossover point interchange between the parent strings and produce two new offspring. This is shown as follows.

Parent's strings before crossover	**Offspring strings after crossover**
1101000**110111001**	1101000**011100101**
1001101**011100101**	1001101**110111001**

It should be high to get optimal performance of algorithms and mostly kept around 0.9.

The code for crossover is given below:

```
function [ popc ] = crossover( pop, pc)
 [m, n] = size(pop);
     m1 = round((m/2)*pc);
for i = 1:m1
    p1 = pop(randi(m,1),:);
    p2 = pop(randi(m,1),:);
    r = randi(n-1,1);
    p3(i,:) = [p1(1:(r-1)) p2(r:n)];
    p4(i,:) = [p2(1:(r-1)) p1(r:n)];
end
p = [p3; p4];
for i = 1:(m-(2*m1))
    r1 = randi(m,1);
    q(i,:) = pop(r1,:);
end
popc = [p; q];
end
```

---

7. Mutation

This last genetic operator is analogous to the natural mutation. As natural mutations happen due to changes in genes randomly. In the same way, the mutation operator is performed by altering the bit values randomly with some mutation probability P_m. This process is necessary to avoid the trapping of the algorithm at any local optima. This process is implemented in the program as a function file 'mutation' in which, a random number between zero to one is generated for each bit in string. If this number is greater than that of mutation probability then the corresponding bit value is altered, either from zero to one or one to zero, otherwise remains the same. This is shown as follows.

Before mutation	11010**0**0011001101
After mutation	11010**1**0011000101

Single point mutation has been used in the code, and its probability is kept relatively low, i.e., 0.05, to preserve the model's originality.

The code for mutation is given below:

```
function [ popm ] = mutation( pop, pm )
 [m, n] = size(pop);
for i = 1:m
    M = pop(i,:);
    for j = 1:n
        r = rand(1,1);
        if r < pm
            if M(j) ==1
                M(j) = 0;
            else
```

```
                M(j) = 1;
            end
        else
            M(j) = M(j);
        end
            Mm(:,j) = M(j);
        end
        Mm;
        popm(i,:) = Mm;
    end
end
```

---

8. Overwriting the old generation

In this step, the initial population is replaced by a new population. This new population is generated by copying some individuals from the initial population and some from that population which is obtained after the three genetic operators. This copying process is done by following some criteria which depend on the fitness values of models. We have implemented in the main file 'ga' of the program by using the Metropolis criterion.

Up to this step a single generation is completed and the new population has more average fitness than that of the initial population. After this, all the above steps are repeated for few times.

9. Stretching

After a few numbers of generations, the average fitness does not improve significantly. This is due to small differences within the fitness values of models in generations. So for further improvement, we have to introduce exaggeration in

selection probability by scaling the fitness values. So equation (5.7) is replaced by the following selection probability.

$$p_s(m_i) = \frac{\exp\left(\frac{F(m_i)}{T}\right)}{\sum_{j=1}^{n} exp\left(\frac{F(m_j)}{T}\right)} \tag{5.8}$$

Where all notations have the usual meaning and T is called the scaling parameter. This is called stretching. This is applied after an interval of a few numbers of generations. This process is also implemented in the main file 'ga' of the program by using linear scaling. All these steps are repeated up to a suitable stopping criterion that is not achieved. The model having the highest fitness value in the final population is the solution to the problem.

10. Main Genetic Algorithm Code (GA):

```
clear all
close all
npr=input('Enter number of parameter used in ff = ');
ps = input('Enter number of models (or population size) = ');
for i = 1:npr
    mpv(:,i) = input('Enter minimum value of each parameter = ');
end
for i = 1:npr
    res(:,i) = input('Enter resolution for each parameter = ');
end
for i = 1:npr
    nbp(:,i) = input('Enter number of bit per parameter = ');
end
mniT = input('Enter maximum number of iteration over temperatur
```

```
parameter = ');
NG = input('Enter maximum number of generation per temperatur
parameter = ');
pc = input('Enter crossover probability = ');
pm = input('Enter mutation probability = ');
T0 = input('Enter initial temperature = ');
lfv = input('Enter limited value of fitness = ');
n = sum(nbp);
m = rand(ps,n);
pop = round(m);
DATA=load('Input Data Path');
s=(DATA(:,1))';
data=(DATA(:,2))';
 [f] = fitness( pop,data,npr,nbp,mpv,res,s );
for k = 1:mniT
    T = T0/k;
    for g = 1:NG
        if g>1
            fnew = fitness(popnew,data,npr,nbp,mpv,res,s);
            F=max(fnew);
            if F>lfv
                break
            end
            for i=1:m
                if fnew(i)>=f(i);
                    f(i)=fnew(i);
                else pup=exp((fnew(i)-f(i))/T);
                    if pup>rand(1,1);
                        pop(i,:)=popnew(i,:);
                        f(i)=fnew(i);
```

```
                    else pop(i,:)=pop(i,:);
                        f(i)=f(i);
                    end
                end
                pop1(i,:)=pop;
                f1(i,:)=f(i);
                pop=pop1;
                f=f1;
            end
        else pop=pop;
            f=f;
        end
        [pops]=selection(pop,f,T);
        [popc]=crossover(pops,pc);
        [popnew]=mutation(popc,pm);
    end
end
fnew=fitness(popnew,data,npr,nbp,mpv,res,s);
F=max(fnew);
I=find(fnew==F);
I1=I(1);
solution =decode(popnew(I1,:),npr,nbp,mpv,res);
loglog(s,responsen(solution,s),'r*-')
hold on
loglog(s,data,'g*-')
```

Abbreviation:

f	:	Fitness
fc	:	Ghosh filter coefficient
L	:	Lamda
M	:	Model parameter vector
mniT	:	Maximum number of iterations per temperature
mpv	:	Minimum parameter value
N	:	No. of layers
NG	:	Maximum number of generations per temperature
nbp	:	Bit per parameter
npr	:	Number of parameters
pc	:	Cross-over probability
pm	:	Mutation probability
pop	:	Population
ps	:	Population size/ No. of models
r	:	Resistivity of the layer
RHOA	:	Apparent resistivity
res	:	The resolution of model parameter
S	:	Binary string
s	:	AB/2
T_0	:	Initial temperature
T	:	The thickness of the layer

5.2.2 Parameters of SGAs

These are the following important parameter of Simple Genetic Algorithms (SGAs). These parameters have some values for the optimal performance of algorithms.

1 . Population size

This is the number of individual strings in the population. This parameter depends on the number of model parameters needed to be optimized. Based on the performance of algorithms on different models with different parameters, we have used a population size of 100 to 200.

2 . Crossover probability

This parameter defines how many individuals in the population will combine their genetic materials with each other. This should be high to get the performance of the optimal algorithms but not too high. Based on synthetic data results, we have used a crossover probability of 0.9 in our work.

3 . Mutation probability

This parameter tells whether a particular bit should be altered or not. This should be small; otherwise, valuable information is lost. Based on synthetic data results, we have used a mutation probability of 0.05 in our work.

5.3 Result and Discussion

5.3.1 Validation of developed Simple Genetic Algorithm program

The developed simple Genetic Algorithm program for the inversion of 1-D geoelectrical data is applied to synthetic data. To check further validity of the

program, it is applied to published data, taken from research work of Mandal et al. (2021).

5.3.1.1 Inversion of synthetic data

Some results are discussed with different layers parameters in synthetic data. Here, ρ represents the resistivity of layers, h represents the thickness of layers, and rrmse stands for relative root mean square error between synthetic data and the response of the inverted model (i.e., response misfit).

1. Three-layer model

The results of the inversion of synthetic VES data of the three-layer model are given in Table 5.2 and Figure 5.3. The GA parameters, the population size of 100, crossover probability of 0.9, mutation probability of 0.05 and number of iterations 4 are used in this case.

Table 5.2 Result of the Three-layer model

Parameter	**True Value**	**Search Range**	**Inverted Model**
ρ1(Ω-m)	5	1-18	4.5
ρ2(Ω-m)	50	15-100	50
ρ3(Ω-m)	15	2-42	15.2
h1(m)	2	0.1-8	1.7
h2(m)	18	4-54	17.2
Response Misfit (RRMSE)		0.018	

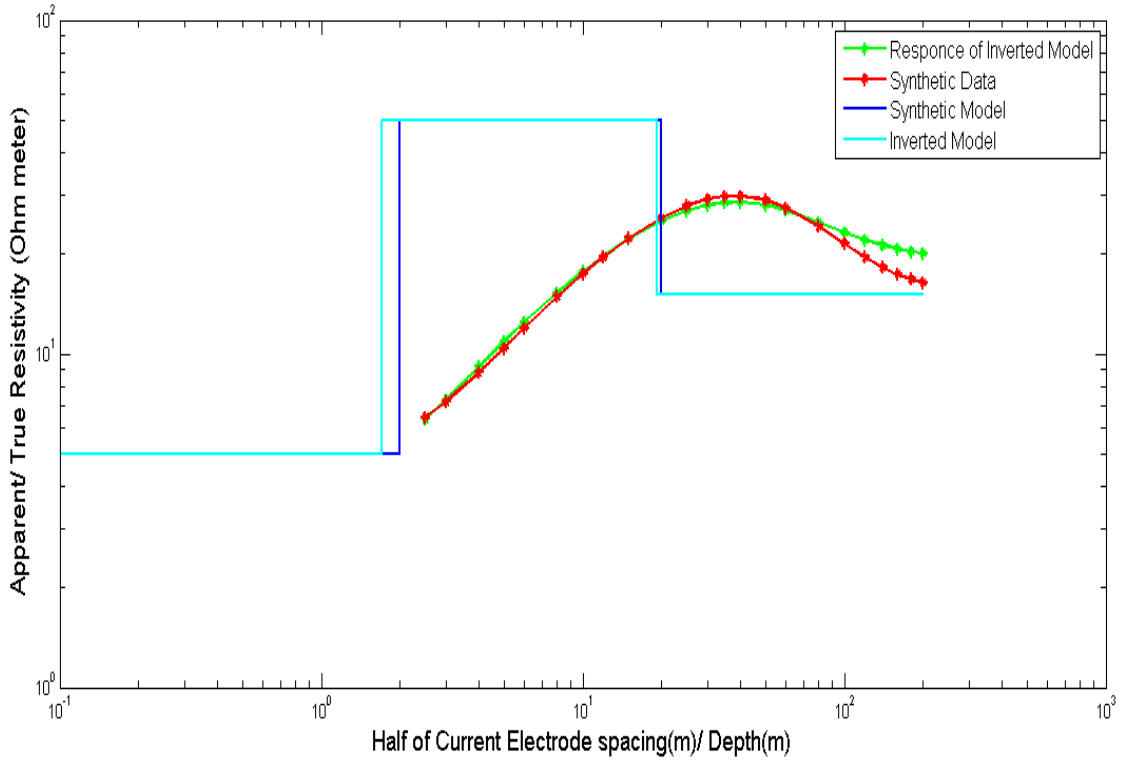

Figure 5.3 Inversion of the three-layer model.

2. Four-layer model

The results of the inversion of synthetic VES data of the four-layer model are given in Table 5.3 and Figure 5.4. The GA parameters, the population size of 100, crossover probability of 0.9, mutation probability of 0.05 and numbers of iterations 5 are used in this case.

Table 5.3 Result of four layers model

Parameter	**True Value**	**Search Range**	**Inverted Model**
ρ1(Ω-m)	150	85 - 212	170
ρ2(Ω-m)	35	10 - 60	28
ρ3(Ω-m)	450	140 - 800	357
ρ4(Ω-m)	50	2 - 128	35
h1(m)	1	0.1 - 8	1.35
h2(m)	6	1 - 8	4.2
h3(m)	25	10 - 75	37
Response Misfit (RRMSE)		0.027	

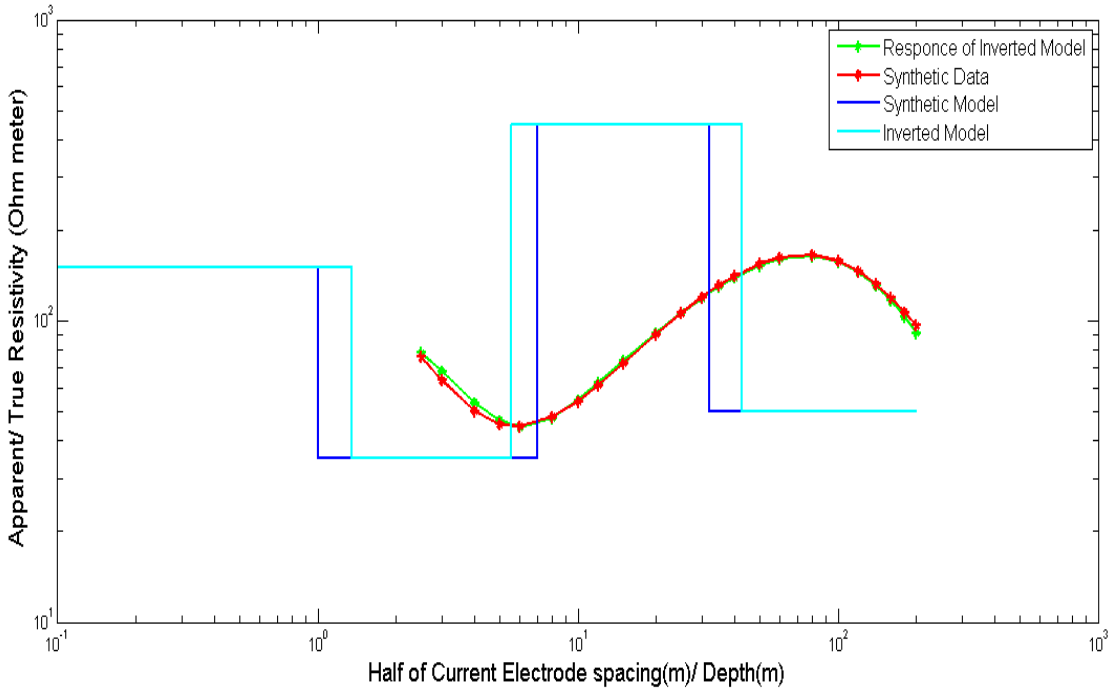

Figure 5.4 Inversion of Four-layer model.

3. Five-layer model

Table 5.4 and Figure 5.5 show the results of the inversion of synthetic VES data for a five-layer model. In this case, the GA parameters of 150 population size, 0.9 crossover probability, 0.05 mutation probability and number of iterations 7 are applied.

Table 5.4 Result of the five-layer model

Parameter	**True Value**	**Search Range**	**Inverted Model**
ρ1(Ω-m)	25	5 - 75	20
ρ2(Ω-m)	4	1 - 8	3.5
ρ3(Ω-m)	50	20 - 100	55
ρ4(Ω-m)	3	0.5 - 6	2.1
ρ5(Ω-m)	30	5 - 75	26
h1(m)	1	0.1 - 6	1.1
h2(m)	2.5	0.5 - 8	2.4
h3(m)	15	1 - 32	16
h4(m)	50	10 - 120	37
Response Misfit (RRMSE)		0.049	

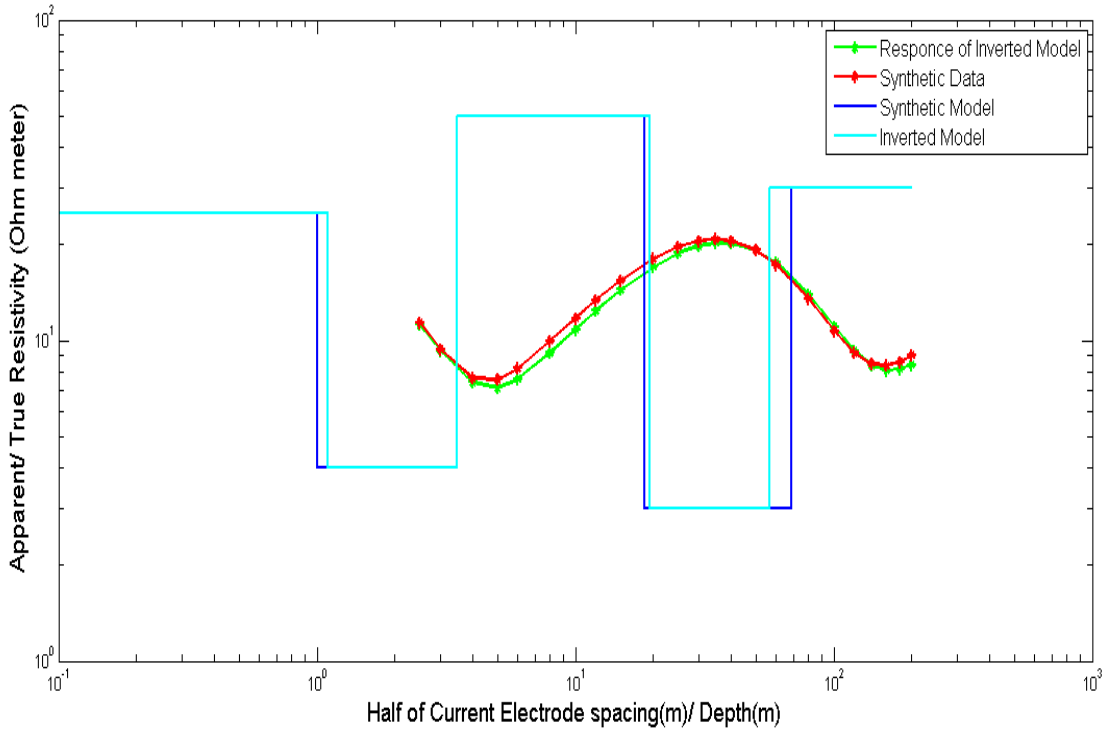

Figure 5.5 Inversion of Five-layer model.

The above results demonstrate that a program based on Simple Genetic Algorithms performs effectively for the inversion of 1-D geoelectrical synthetic data. The response misfit (RRMSE) ranges from 0.018 to 0.049, within the acceptable range. The above findings show that as the number of layers (i.e., the number of parameters) rises, we must maintain the search range more precise.

5.3.1.2 Inversion of published VES data

To validate the further performance of the algorithm, it is applied to the five VES field data taken from the research work of Mandal et al. (2021). This data reflects four- layers. So this data is inverted in the same way as the above synthetic data for the four-layer model is inverted.

The results of the inversion of published five VES data of the four-layer model are given in the following Table and Figure. In this case, the GA parameters

population size of 100-200, crossover probability of 0.9, mutation probability of 0.05 and number of iterations 5 to 10 are employed.

Table 5.5 Result of VES-1

Parameter	**Published Model**	**Search Range**	**Inverted Model**
ρ1(Ω-m)	5	1 - 8	4.2
ρ2(Ω-m)	2.3	0.1 - 6	3.6
ρ3(Ω-m)	9.6	2 - 30	12.9
ρ4(Ω-m)	230	80 - 500	257
h1(m)	0.9	0.1 - 6	1.4
h2(m)	2.8	0.1 - 6	2.55
h3(m)	11.3	2 - 28	9.5
Response Misfit (RRMSE)	0.047		

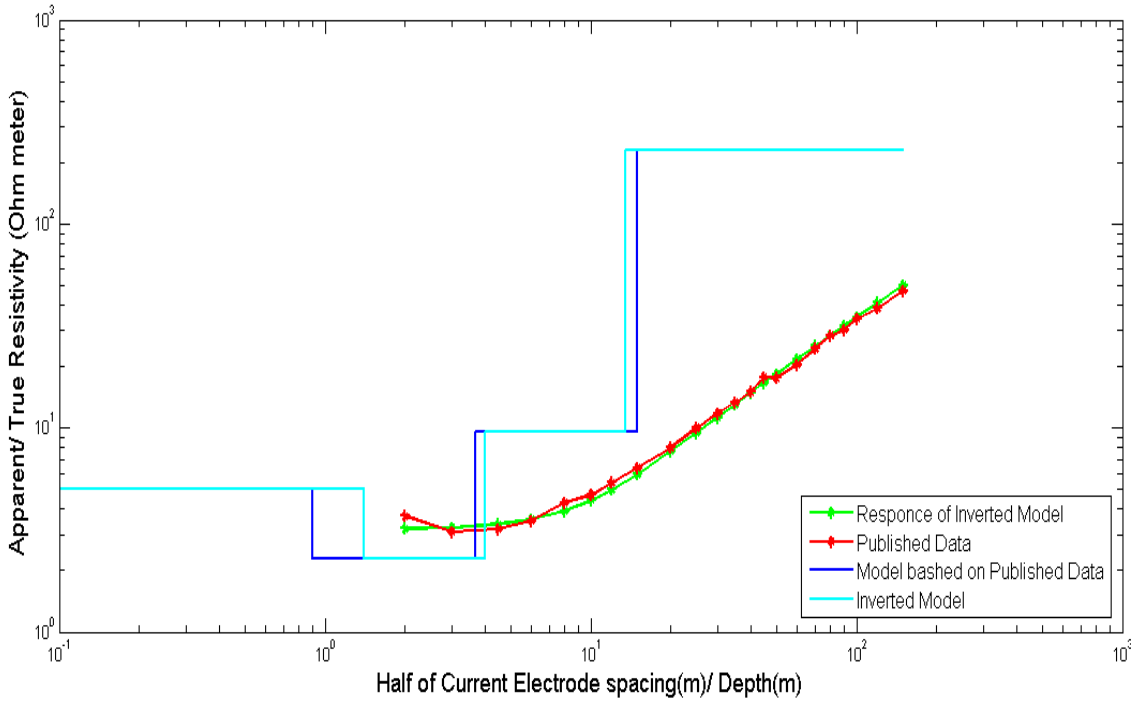

Figure 5.6 Inversion of Published data (VES-1).

Table 5.6 Result of VES-2

Parameter	Published Model	Search Range	Inverted Model
ρ1(Ω-m)	60	20 - 150	88
ρ2(Ω-m)	27.5	5 - 55	32
ρ3(Ω-m)	11	2 - 28	13.6
ρ4(Ω-m)	69	25 - 125	79
h1(m)	1.2	0.1 - 6	0.86
h2(m)	2.2	0.1 - 8	3.1
h3(m)	12.7	1 - 32	9.1
Response Misfit (RRMSE)	0.078		

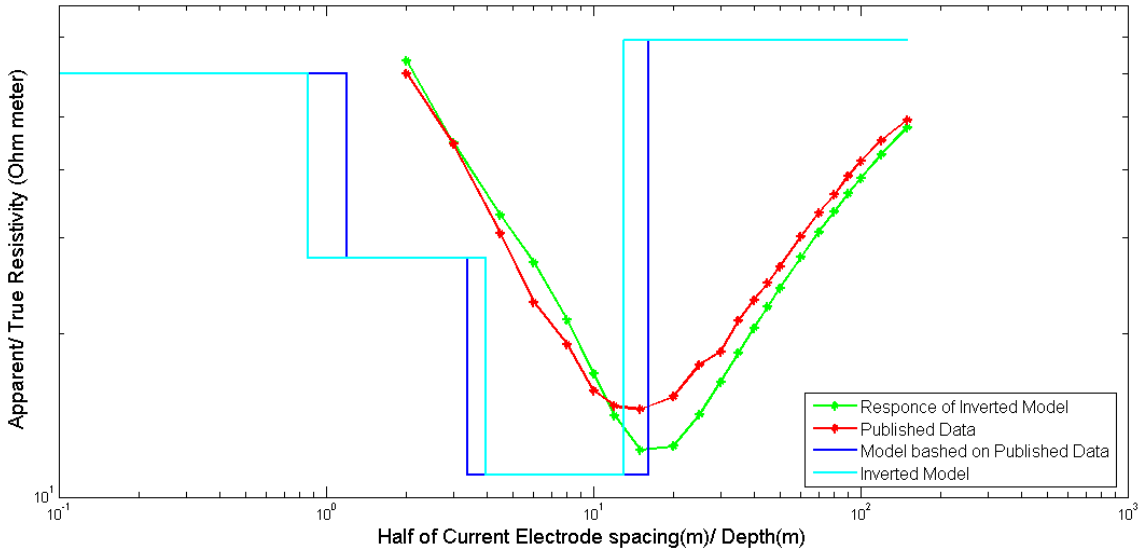

Figure 5.7 Inversion of Published data (VES-2).

Table 5.7 Result of VES-3

Parameter	Published Model	Search Range	Inverted Model
ρ1(Ω-m)	13	2 - 32	9
ρ2(Ω-m)	5.5	0.1 - 12	6.4
ρ3(Ω-m)	27	5 - 55	26
ρ4(Ω-m)	342	150 - 600	346
h1(m)	1	0.1 - 6	1.9
h2(m)	1.5	0.1 - 6	1.1
h3(m)	14	2 - 32	12
Response Misfit (RRMSE)	0.051		

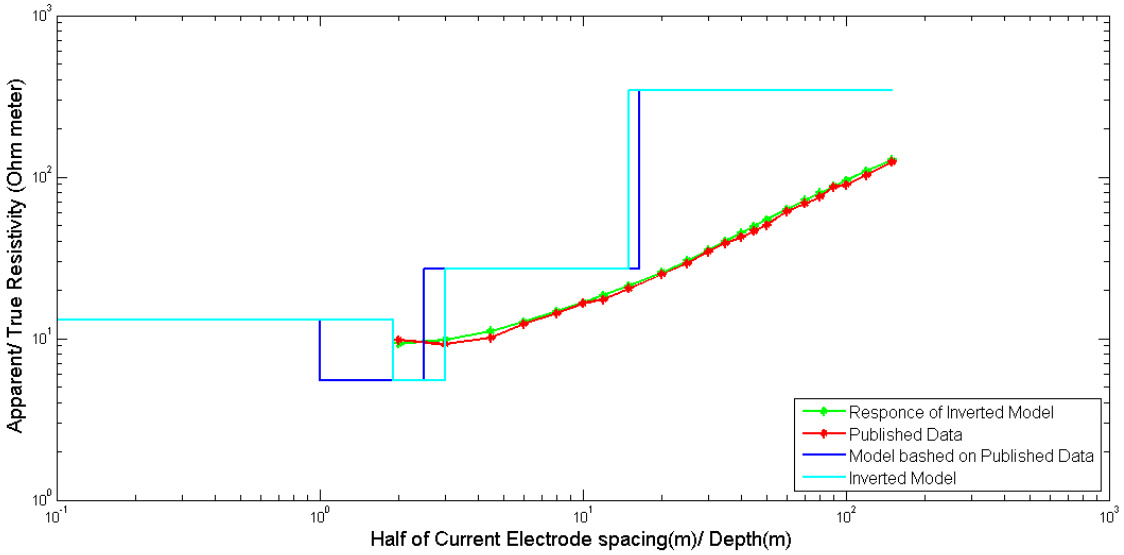

Figure 5.8 Inversion of Published data (VES-3).

Table 5.8 Result of VES-4

Parameter	**Published Model**	**Search Range**	**Inverted Model**
ρ1(Ω-m)	20	1 - 33	21
ρ2(Ω-m)	10	1 - 17	13
ρ3(Ω-m)	155	50 - 400	269
ρ4(Ω-m)	450	200 - 750	455
h1(m)	1.4	0.1 - 6	1.15
h2(m)	3.2	0.5 - 8	5
h3(m)	9.4	2 - 32	10
Response Misfit (RRMSE)	0.047		

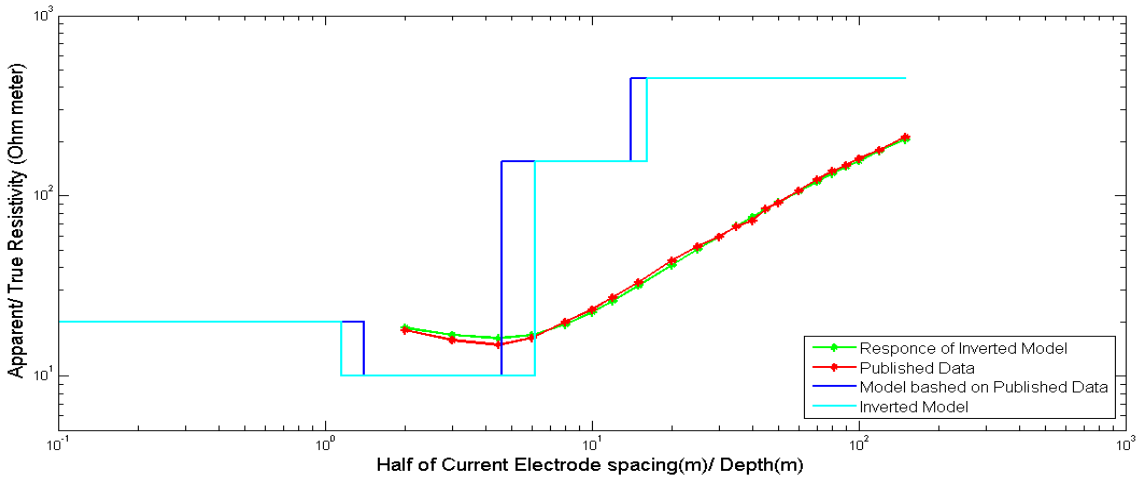

Figure 5.9 Inversion of Published data (VES-4).

Table 5.9 Result of VES-5

Parameter	**Published Model**	**Search Range**	**Inverted Model**
ρ1(Ω-m)	42	10 - 80	38.6
ρ2(Ω-m)	30	10 - 60	22
ρ3(Ω-m)	55	20 - 120	78
ρ4(Ω-m)	990	500 - 1600	1350
h1(m)	1.4	0.1 - 6	1.74
h2(m)	3.5	0.5 - 8	3.86
h3(m)	9.1	3 - 22	11.2
Response Misfit (RRMSE)	0.031		

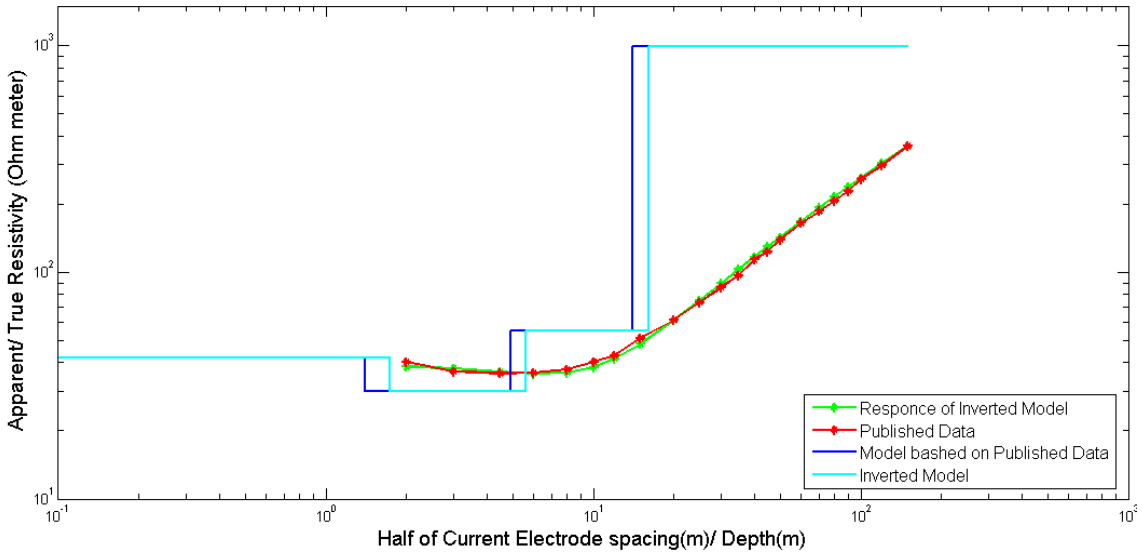

Figure 5.10 Inversion of Published data (VES-5).

The above results show that the inverted model closely matches the research paper model and the response misfit (RRMSE) is within an acceptable range, ranging from 0.031 to 0.078. Hence we find that program effectively inverts 1-D Geoelectrical data.

5.3.2 Inversion of study area VES data

The resistivity survey was carried out in the Singrauli Coalfield regions. Fifty-

five VES data were acquired using Schlumberger electrode configuration with a maximum current electrode separation (AB) of 400 m and potential electrode spacing (MN) of 20 m. The locations of VES stations are shown in **Figure 5.11**. The developed program was used to interpret the acquired VES data to obtain layer parameters (true resistivity, thickness). The GA-based computer program was executed multiple times (approximately 4 to 10 times) for each set of VES data to get reliable layer parameters. The true layer resistivity and thickness values of various layers obtained from the inversion of VES data were correlated with the available borehole lithology. Based on this correlation and the experiences of local field hydrogeologists, the types of sub-surface layers present at different depths, including water-bearing formations, were identified and described in detail in Chapter 6.

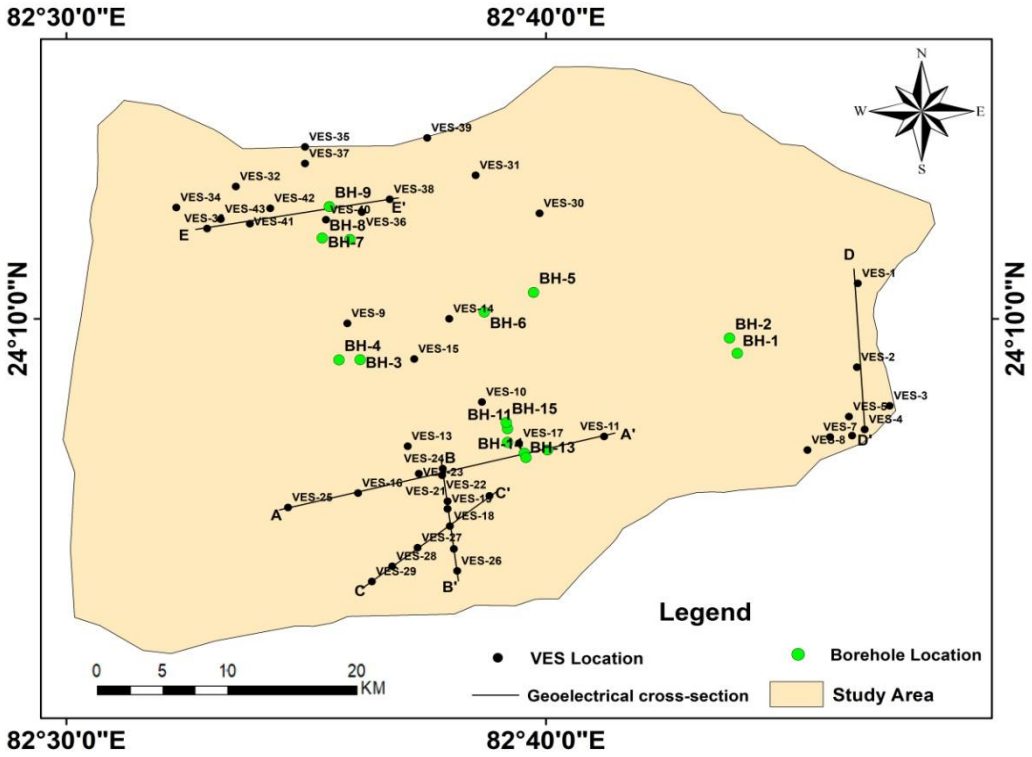

Figure 5.11 Location of VES station.

The results of the inversion of fifty-five VES data from the study area are

shown in the following table and figure below. The GA parameters population size of 100-200, crossover probability of 0.9, mutation probability of 0.05 and number of iterations 5 to 10 are employed.

Table 5.10 Result of VES-1

Parameter	**Search Range**	**Inverted Model**
ρ1(Ω-m)	3 - 45	9.7
ρ2(Ω-m)	100 - 500	22.9
ρ3(Ω-m)	60 - 150	74
ρ4(Ω-m)	160 - 450	225
ρ5(Ω-m)	3 - 60	19.4
h1(m)	0.1 - 7	3
h2(m)	2 - 32	4.9
h3(m)	2 - 50	8.8
h4(m)	5 - 75	27.7
Response Misfit (RRMSE)	0.076	

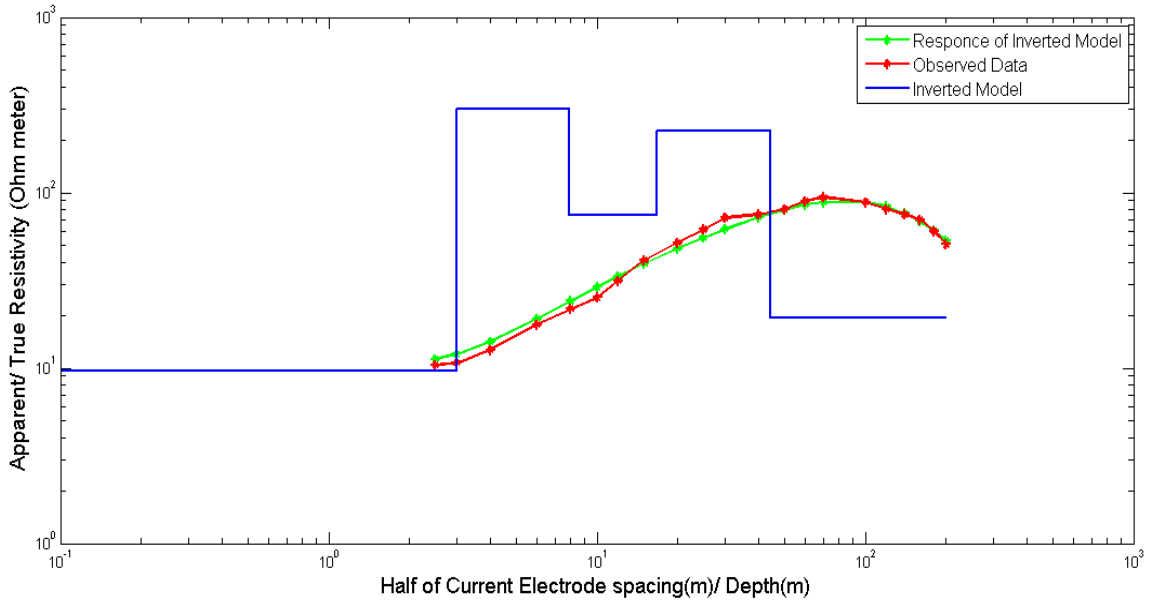

Figure 5.12 Inversion of VES-1.

Table 5.11 Result of VES-2

Parameter	**Search Range**	**Inverted Model**
ρ1(Ω-m)	30 - 180	92.8
ρ2(Ω-m)	5 - 75	25
ρ3(Ω-m)	5 - 120	76.3
ρ4(Ω-m)	50 - 200	86
ρ5(Ω-m)	8 - 32	15.2
h1(m)	0.1 - 5	1.4
h2(m)	0.2 - 8	5.1
h3(m)	2 - 60	32.9
h4(m)	5 - 45	13.8
Response Misfit (RRMSE)	0.059	

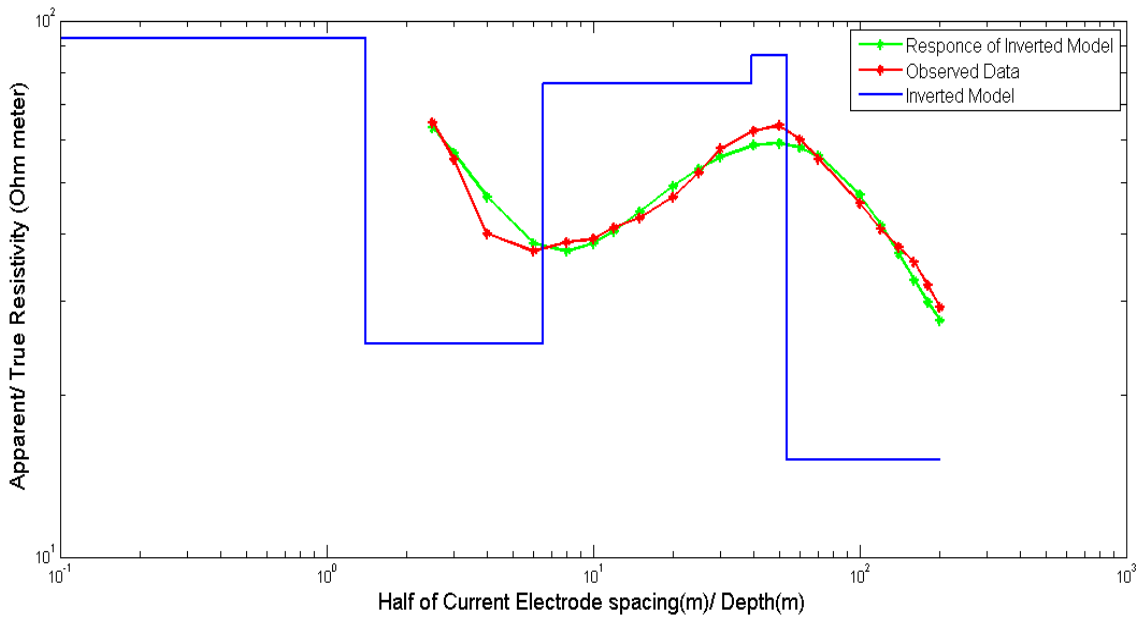

Figure 5.13 Inversion of VES-2.

Table 5.12 Result of VES-3

Parameter	**Search Range**	**Inverted Model**
ρ1(Ω-m)	2 - 40	13.1
ρ2(Ω-m)	10 - 50	24.6
ρ3(Ω-m)	100 - 200	144
h1(m)	0.2 - 10	11.3
h2(m)	25 - 100	64.8
Response Misfit (RRMSE)	0.038	

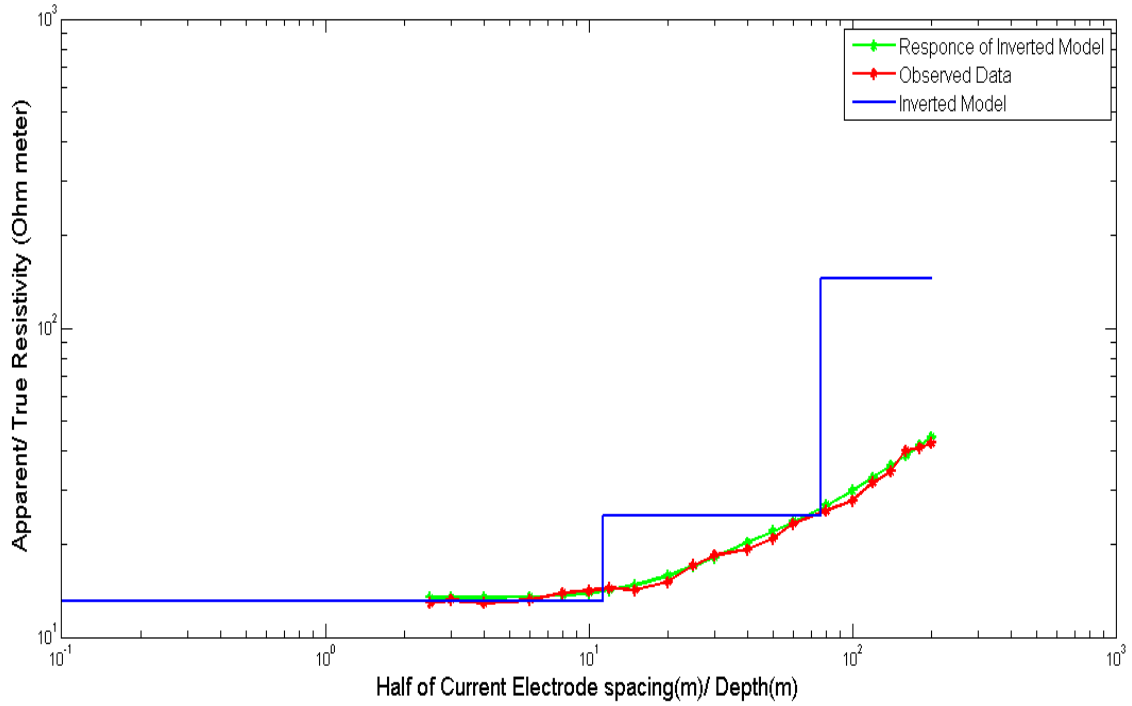

Figure 5.14 Inversion of VES-3.

Table 5.13 Result of VES-4

Parameter	**Search Range**	**Inverted Model**
ρ1(Ω-m)	5 - 45	32.8
ρ2(Ω-m)	2 - 32	9.1
ρ3(Ω-m)	5 - 40	15.8
ρ4(Ω-m)	70 - 200	102
h1(m)	0.2 - 4	0.6
h2(m)	0.2 - 10	7.4
h3(m)	3 - 60	32.6
Response Misfit (RRMSE)	0.062	

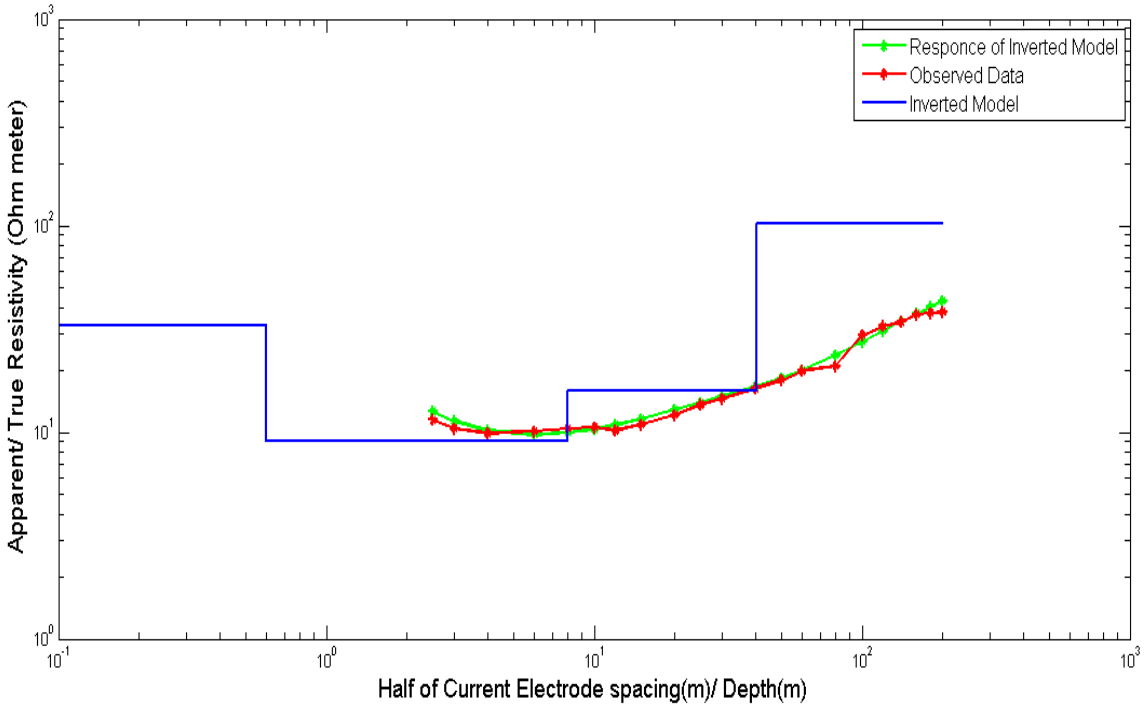

Figure 5.15 Inversion of VES-4.

Table 5.14 Result of VES-5

Parameter	**Search Range**	**Inverted Model**
ρ1(Ω-m)	1 - 32	11.1
ρ2(Ω-m)	1 - 32	15.2
ρ3(Ω-m)	25 - 150	81
h1(m)	0.1 - 8	4.5
h2(m)	2 - 75	37.8
Response Misfit (RRMSE)	0.041	

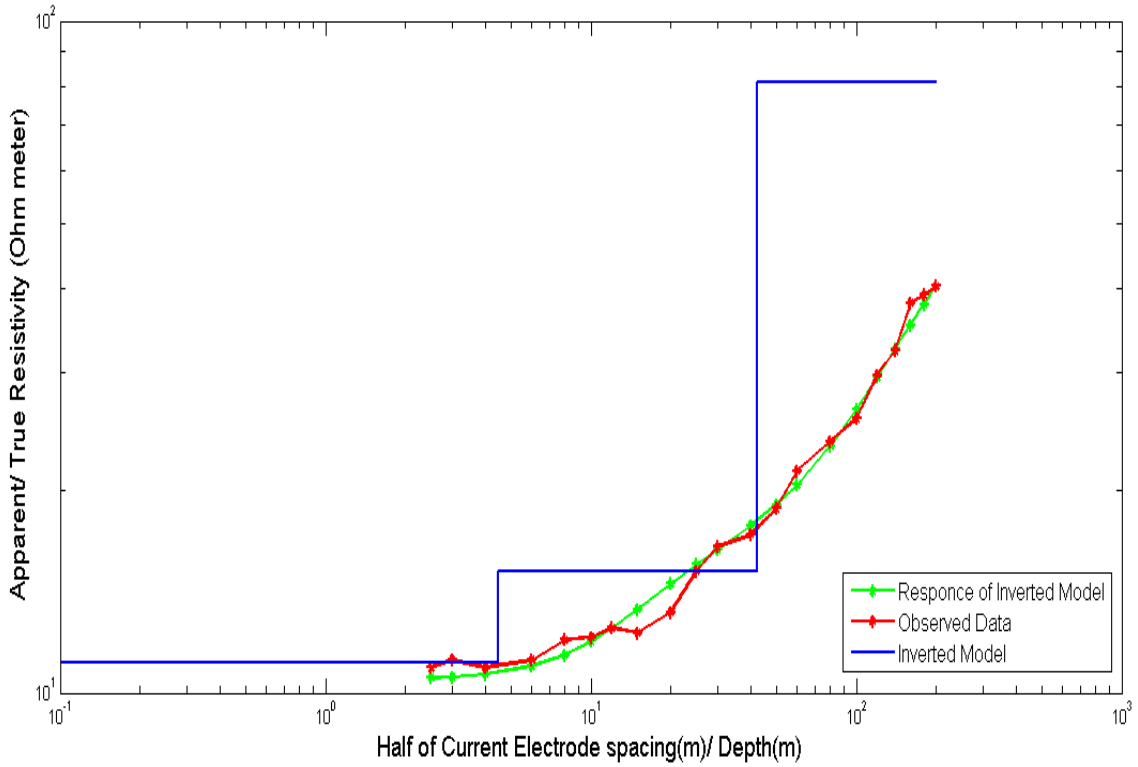

Figure 5.16 Inversion of VES-5.

Table 5.15 Result of VES-6

Parameter	**Search Range**	**Inverted Model**
ρ1(Ω-m)	0.5 - 8	7.9
ρ2(Ω-m)	10 - 120	51.5
ρ3(Ω-m)	5 - 75	22.7
ρ4(Ω-m)	4 - 70	31.8
h1(m)	0.1 - 8	6
h2(m)	0.1 - 8	2.4
h3(m)	10 - 75	32.6
Response Misfit (RRMSE)	0.082	

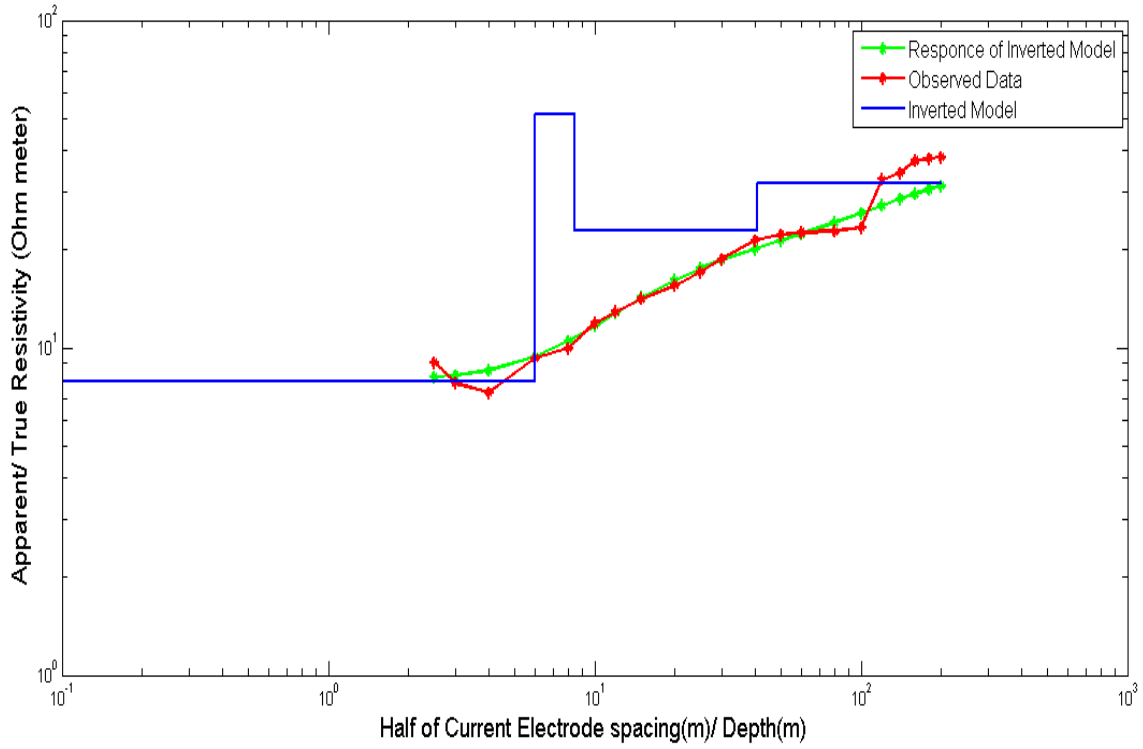

Figure 5.17 Inversion of VES-6.

Table 5.16 Result of VES-7

Parameter	**Search Range**	**Inverted Model**
ρ1(Ω-m)	0.2 - 15	9.9
ρ2(Ω-m)	0.2 - 9	5.7
ρ3(Ω-m)	5 - 35	14.4
ρ4(Ω-m)	20 - 80	44.3
ρ5(Ω-m)	40 - 200	133
h1(m)	0.1 - 8	1.5
h2(m)	0.2 - 10	6.3
h3(m)	5 - 45	19.7
h4(m)	30 - 120	56.4
Response Misfit (RRMSE)	0.057	

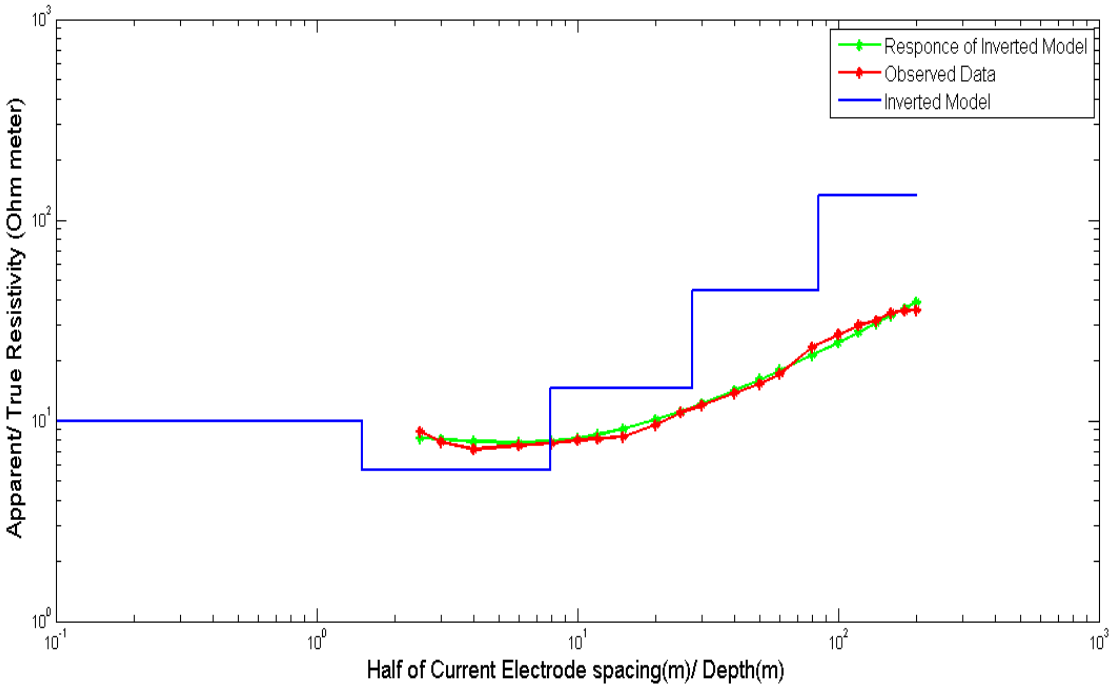

Figure 5.18 Inversion of VES-7.

Table 5.17 Result of VES-8

Parameter	**Search Range**	**Inverted Model**
ρ1(Ω-m)	1 - 20	10.7
ρ2(Ω-m)	5 - 75	22.9
ρ3(Ω-m)	2 - 60	31.6
ρ4(Ω-m)	50 - 150	98.9
h1(m)	1 - 18	11.2
h2(m)	5 - 75	24.7
h3(m)	10 - 110	47.5
Response Misfit (RRMSE)	0.043	

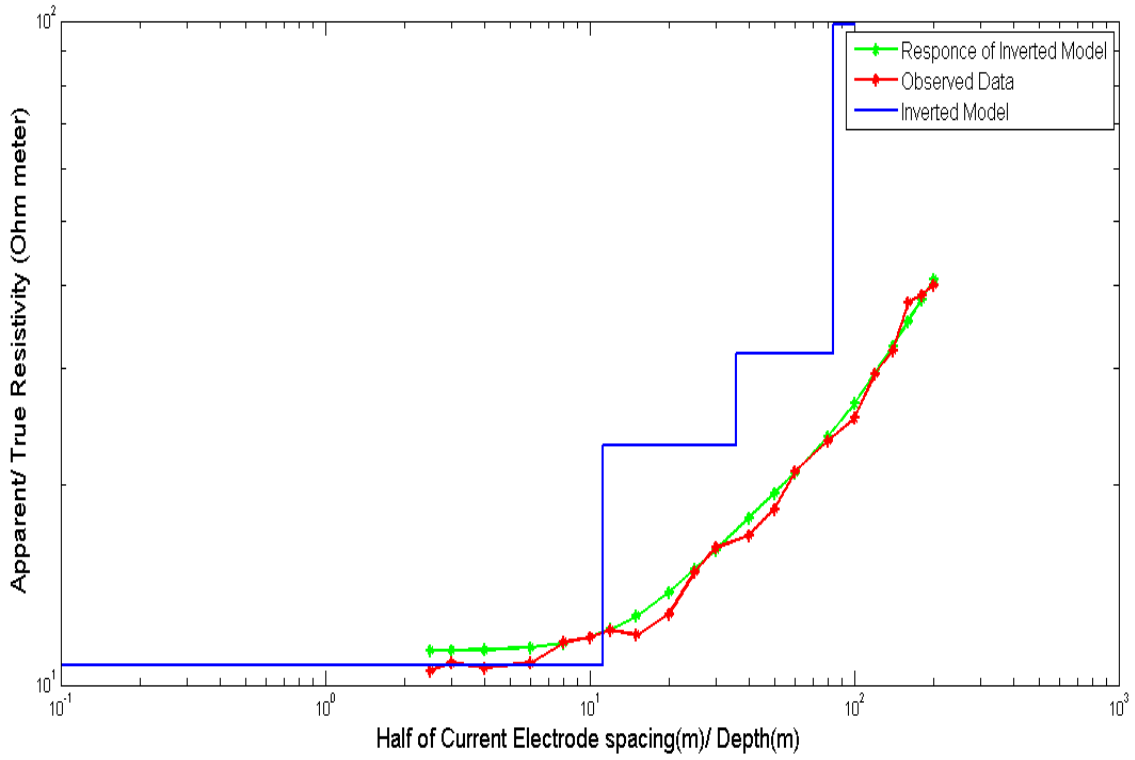

Figure 5.19 Inversion of VES-8.

Table 5.18 Result of VES-9

Parameter	**Search Range**	**Inverted Model**
ρ1(Ω-m)	150 - 550	319
ρ2(Ω-m)	100 - 400	175
ρ3(Ω-m)	150 - 750	534
ρ4(Ω-m)	300 - 1000	655
h1(m)	0.1 - 5	1.5
h2(m)	0.2 - 8	2.5
h3(m)	15 - 55	31.7
Response Misfit (RRMSE)	0.038	

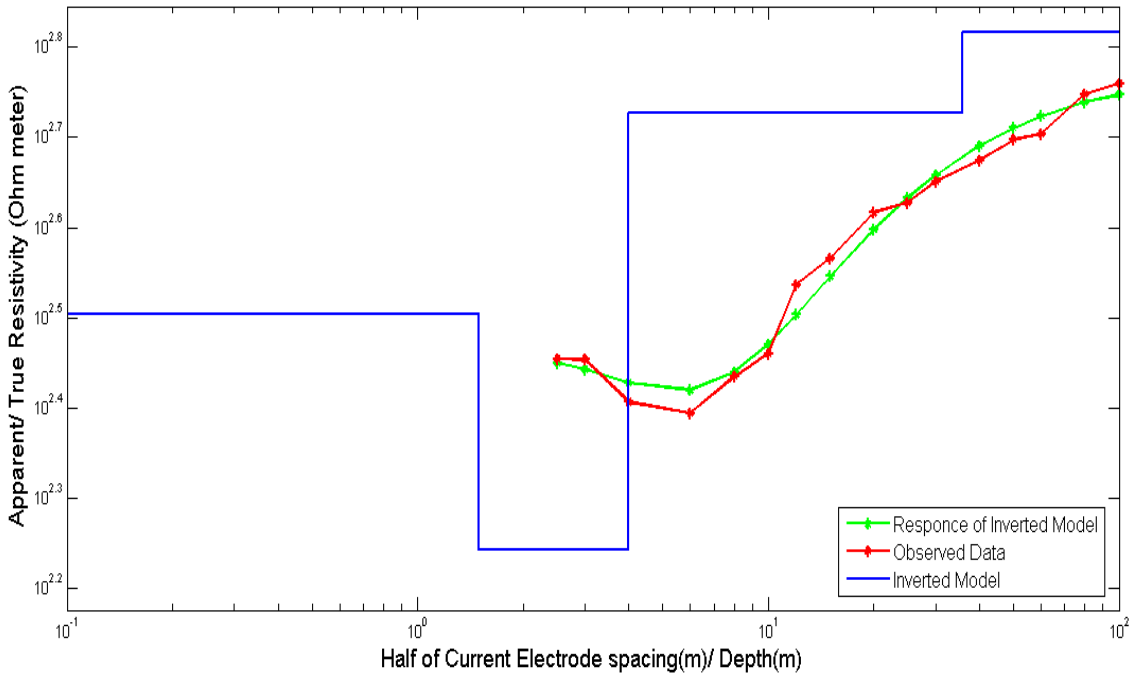

Figure 5.20 Inversion of VES-9.

Table 5.19 Result of VES-10

Parameter	**Search Range**	**Inverted Model**
ρ1(Ω-m)	10 - 100	64.3
ρ2(Ω-m)	5 - 55	32.5
ρ3(Ω-m)	2 - 32	11.7
ρ4(Ω-m)	2 - 64	33.2
h1(m)	0.2 - 15	5.3
h2(m)	2 - 50	22.9
h3(m)	5 - 55	24.9
Response Misfit (RRMSE)	0.045	

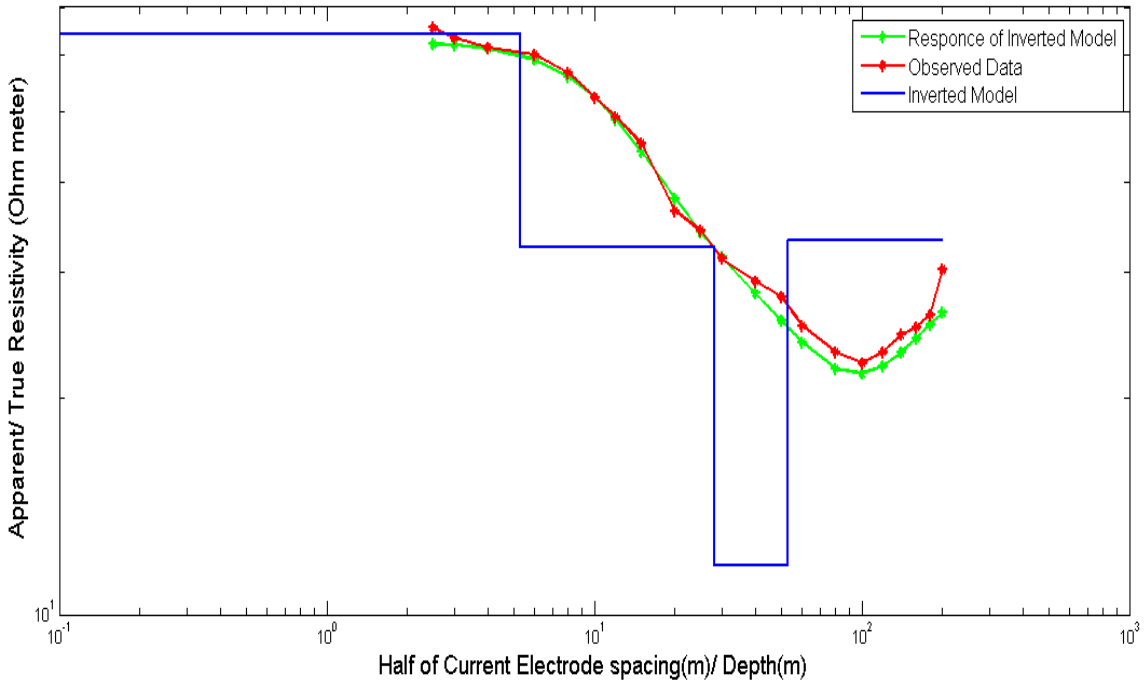

Figure 5.21 Inversion of VES-10.

Table 5.20 Result of VES-11

Parameter	**Search Range**	**Inverted Model**
ρ1(Ω-m)	20 - 80	55.4
ρ2(Ω-m)	0.1 - 6	2.8
ρ3(Ω-m)	25 - 120	69.6
ρ4(Ω-m)	0.1 - 6	2.1
ρ5(Ω-m)	2 - 50	23.4
h1(m)	0.1 - 6	1.3
h2(m)	0.1 - 8	1.9
h3(m)	0.2 - 10	3.6
h4(m)	5 - 75	45.4
Response Misfit (RRMSE)	0.10	

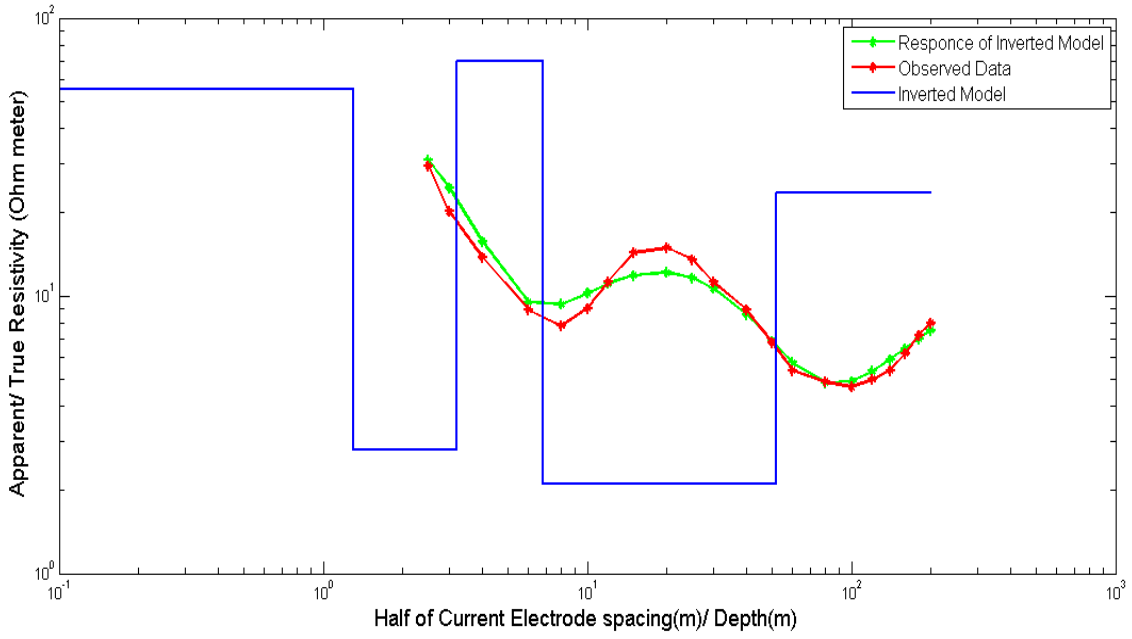

Figure 5.22 Inversion of VES-11.

Table 5.21 Result of VES-12

Parameter	**Search Range**	**Inverted Model**
ρ1(Ω-m)	5 - 55	36.7
ρ2(Ω-m)	0.2 - 10	4.8
ρ3(Ω-m)	2 - 32	12.8
ρ4(Ω-m)	4 - 48	24.9
h1(m)	0.1 - 8	2.6
h2(m)	1 - 32	11.1
h3(m)	2 - 75	38.5
Response Misfit (RRMSE)	0.094	

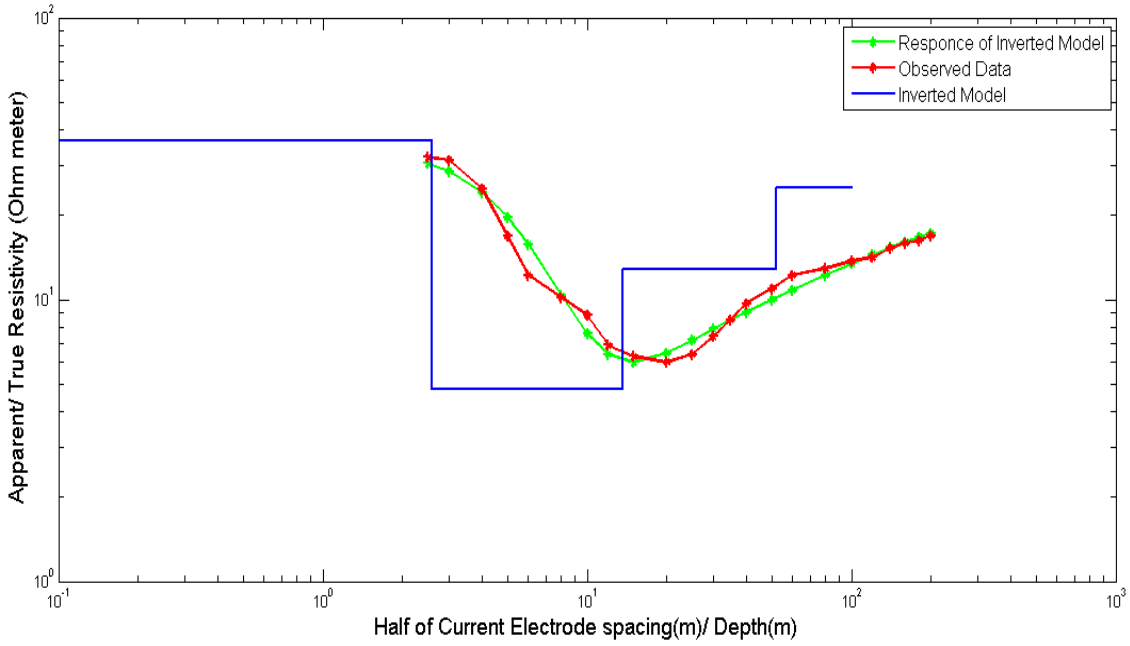

Figure 5.23 Inversion of VES-12.

Table 5.22 Result of VES-13

Parameter	**Search Range**	**Inverted Model**
ρ1(Ω-m)	15 - 120	57.5
ρ2(Ω-m)	1 - 30	15.5
ρ3(Ω-m)	15 - 75	62.6
h1(m)	0.1 - 6	1.3
h2(m)	0.2 - 10	5
Response Misfit (RRMSE)	0.034	

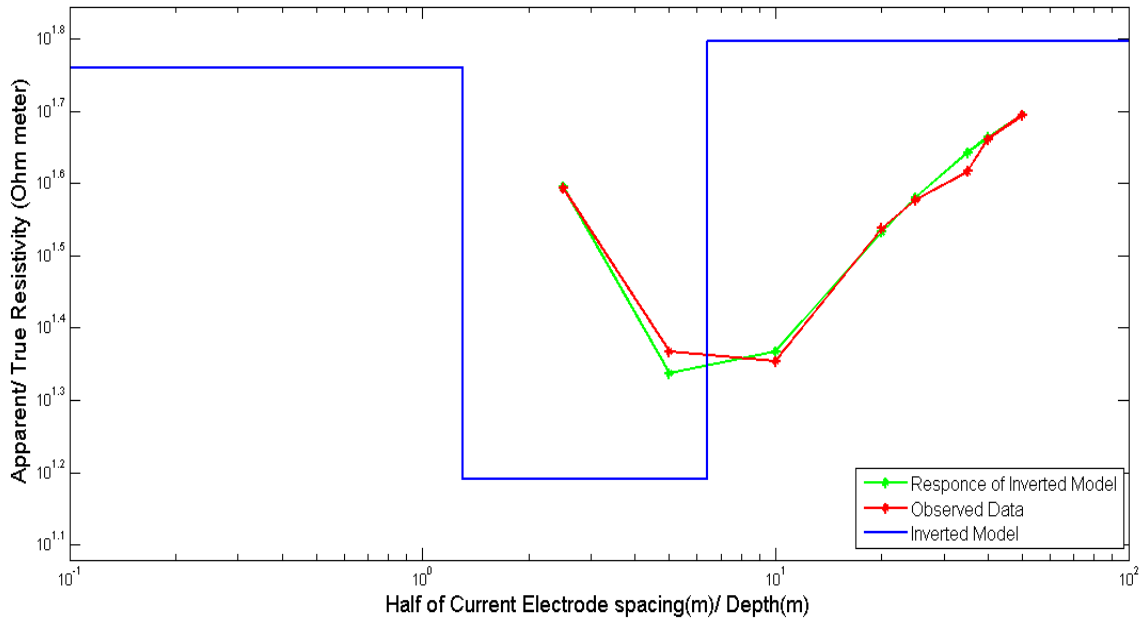

Figure 5.24 Inversion of VES-13.

Table 5.23 Result of VES-14

Parameter	**Search Range**	**Inverted Model**
ρ1(Ω-m)	350 - 1200	788
ρ2(Ω-m)	20 - 140	77.5
ρ3(Ω-m)	100 - 500	275
ρ4(Ω-m)	0.2 - 10	5.9
h1(m)	0.1 - 6	1
h2(m)	0.1 - 6	1.2
h3(m)	2 - 32	19.9
Response Misfit (RRMSE)	0.087	

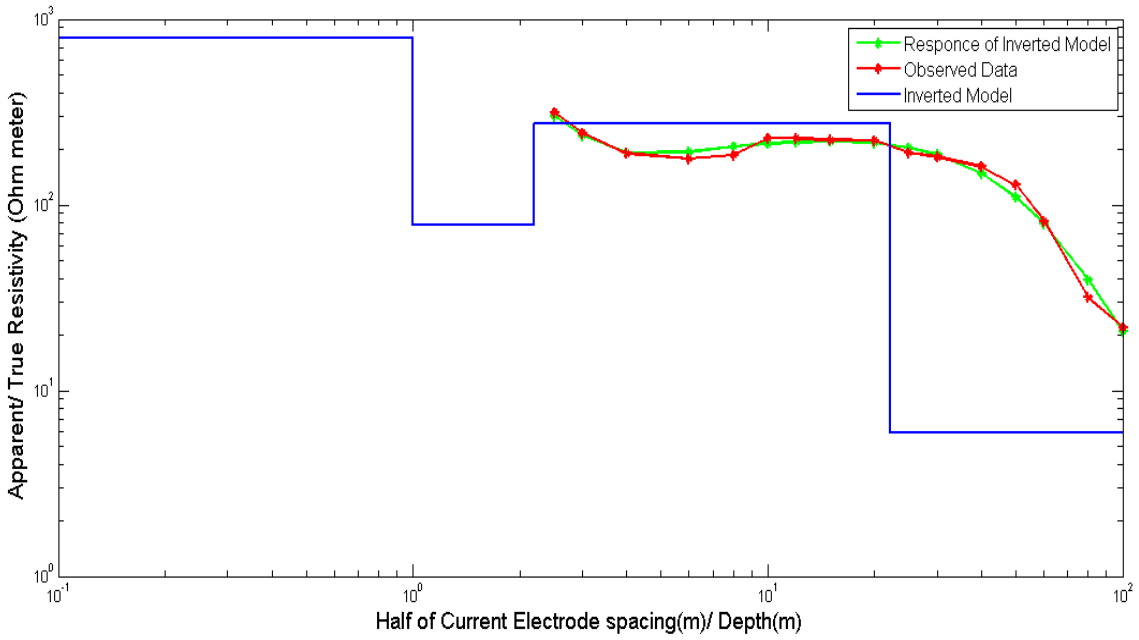

Figure 5.25 Inversion of VES-14.

Table 5.24 Result of VES-15

Parameter	**Search Range**	**Inverted Model**
ρ1(Ω-m)	300 - 1000	759.1
ρ2(Ω-m)	250 - 800	375.2
ρ3(Ω-m)	20 - 150	93.4
ρ4(Ω-m)	100 - 400	189.3
h1(m)	0.1 - 6	1.5
h2(m)	0.2 - 8	6.7
h3(m)	2 - 40	20.6
Response Misfit (RRMSE)	0.047	

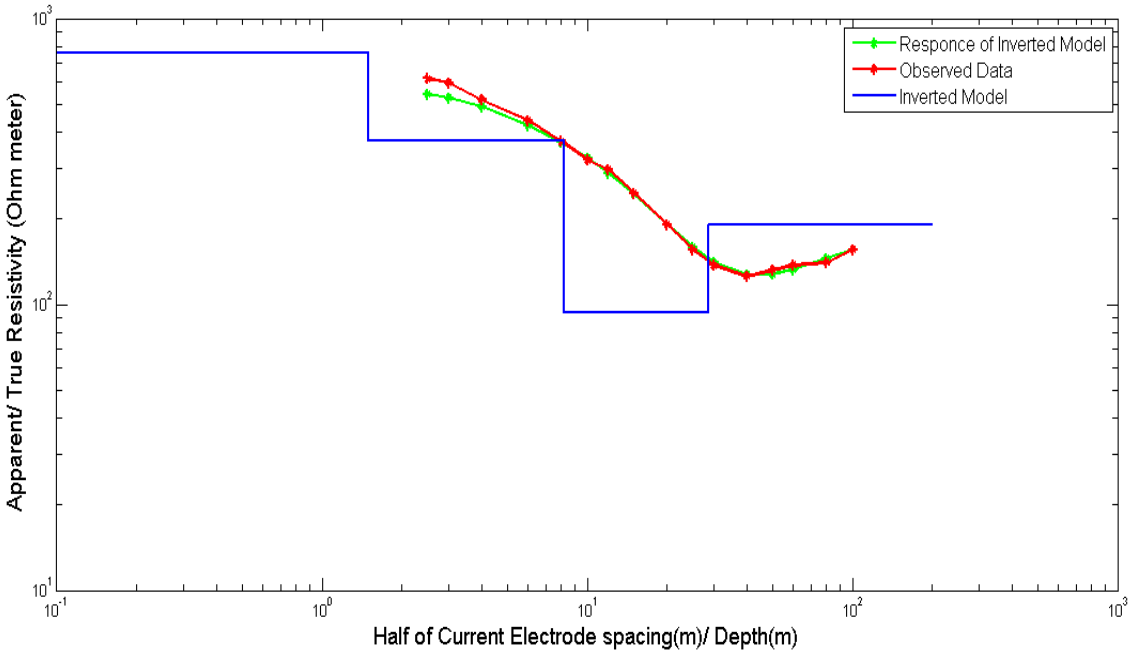

Figure 5.26 Inversion of VES-15.

Table 5.25 Result of VES-16

Parameter	**Search Range**	**Inverted Model**
ρ1(Ω-m)	25 - 150	87.7
ρ2(Ω-m)	2 - 32	7.7
ρ3(Ω-m)	2 - 40	14.2
ρ4(Ω-m)	20 - 120	75.3
h1(m)	0.1 - 6	0.8
h2(m)	1 - 32	18.2
h3(m)	5 - 75	35.8
Response Misfit (RRMSE)	0.061	

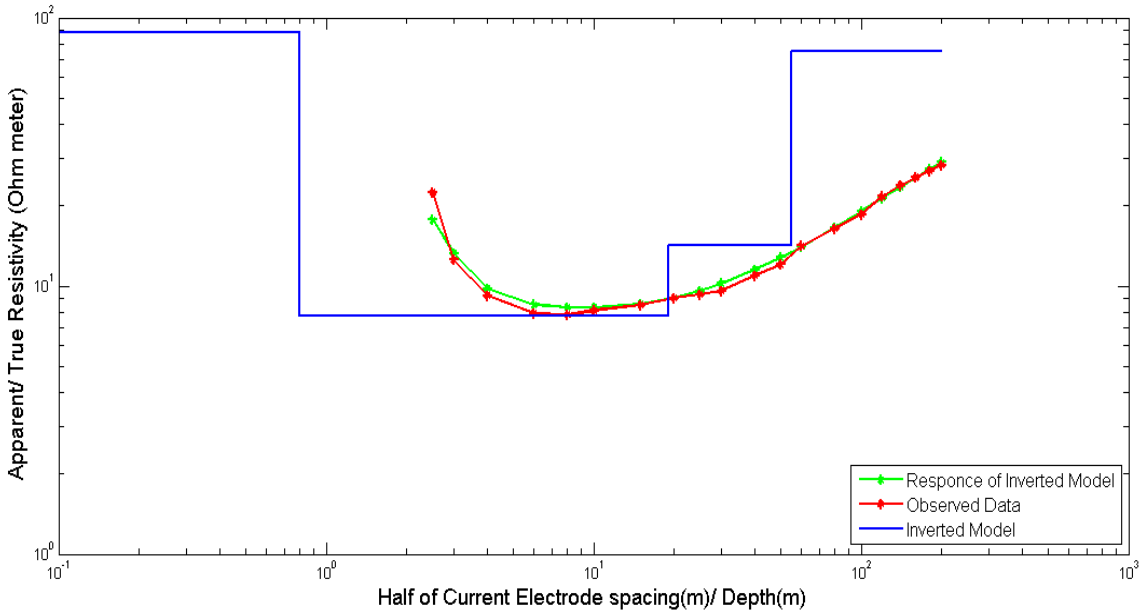

Figure 5.27 Inversion of VES-16.

Table 5.26 Result of VES-17

Parameter	Search Range	Inverted Model
ρ1(Ω-m)	10 - 80	45.3
ρ2(Ω-m)	2 - 40	19.1
ρ3(Ω-m)	20 - 140	70.5
ρ4(Ω-m)	2 - 32	11.9
ρ5(Ω-m)	2 - 32	13.1
h1(m)	0.1 - 6	1.6
h2(m)	0.5 - 8	3.4
h3(m)	2 - 32	13
h4(m)	5 - 60	44.5
Response Misfit (RRMSE)	0.77	

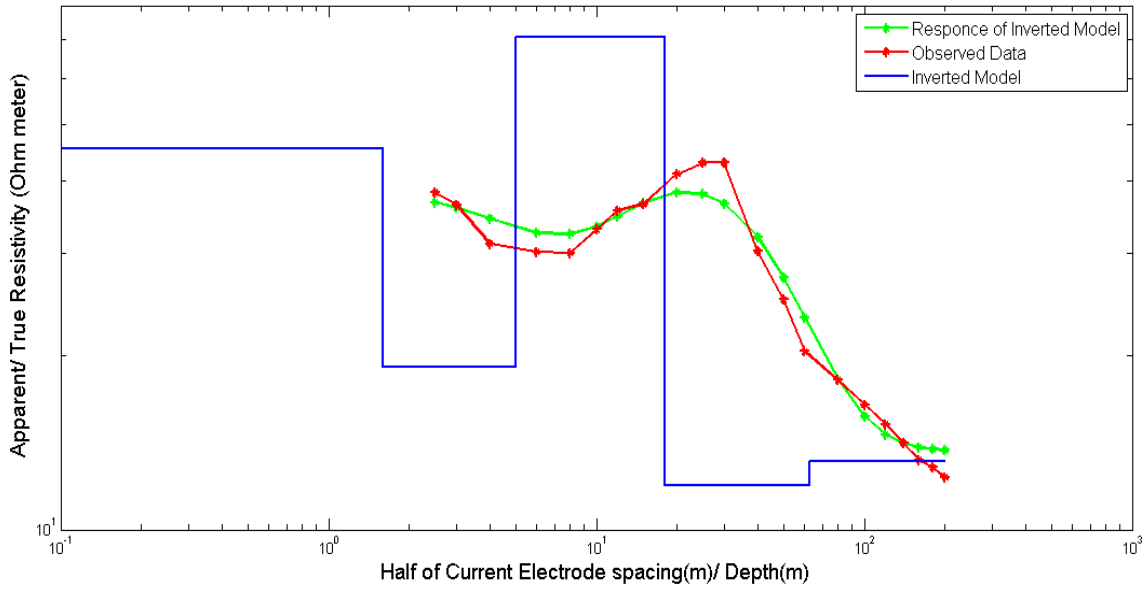

Figure 5.28 Inversion of VES-17.

Table 5.27 Result of VES-18

Parameter	**Search Range**	**Inverted Model**
ρ1(Ω-m)	10 - 80	40.1
ρ2(Ω-m)	0.5 - 15	7.6
ρ3(Ω-m)	2 - 32	17.3
ρ4(Ω-m)	0.1 - 8	3.1
h1(m)	0.1 - 6	1.2
h2(m)	0.5 - 8	2.3
h3(m)	5 - 55	26.5
Response Misfit (RRMSE)	0.088	

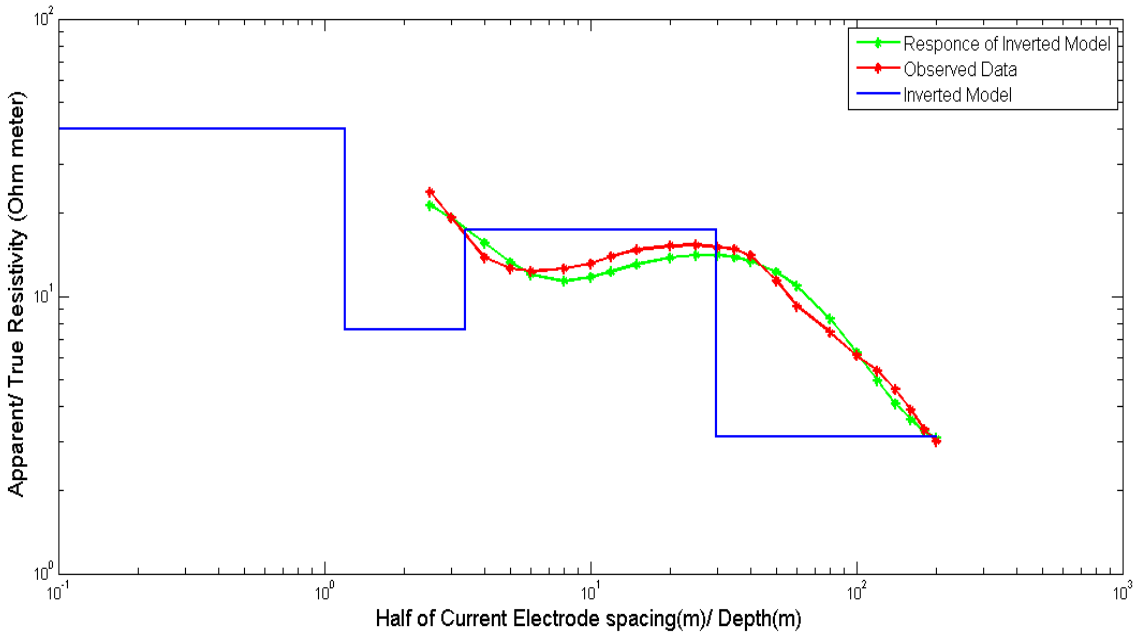

Figure 5.29 Inversion of VES-18.

Table 5.28 Result of VES-19

Parameter	**Search Range**	**Inverted Model**
ρ1(Ω-m)	0.5 - 8	5.25
ρ2(Ω-m)	1 - 32	9.31
ρ3(Ω-m)	2 - 40	16.5
ρ4(Ω-m)	0.5 - 8	7.7
h1(m)	0.1 - 6	1.41
h2(m)	0.2 - 8	2.68
h3(m)	5 - 55	26.3
Response Misfit (RRMSE)	0.067	

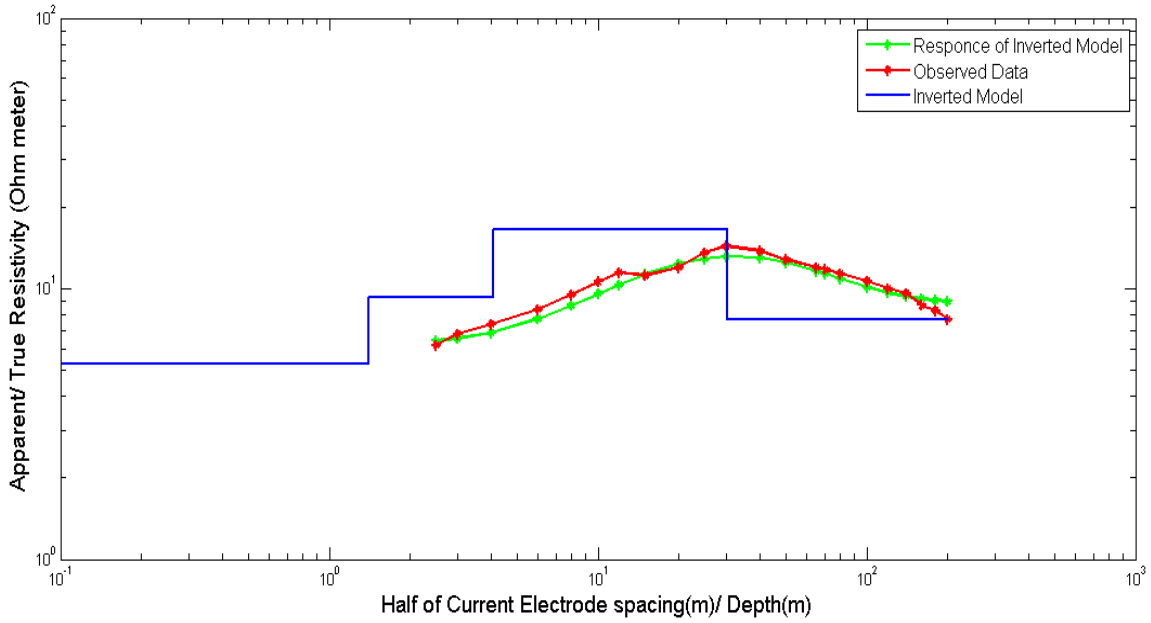

Figure 5.30 Inversion of VES-19.

Table 5.29 Result of VES-20

Parameter	**Search Range**	**Inverted Model**
ρ1(Ω-m)	0.1 - 6	3.83
ρ2(Ω-m)	1 - 22	10.1
ρ3(Ω-m)	1 - 20	18.9
ρ4(Ω-m)	1 - 16	6.37
h1(m)	0.1 - 6	1.35
h2(m)	0.1 - 8	6.99
h3(m)	2 - 32	20.3
Response Misfit (RRMSE)	0.058	

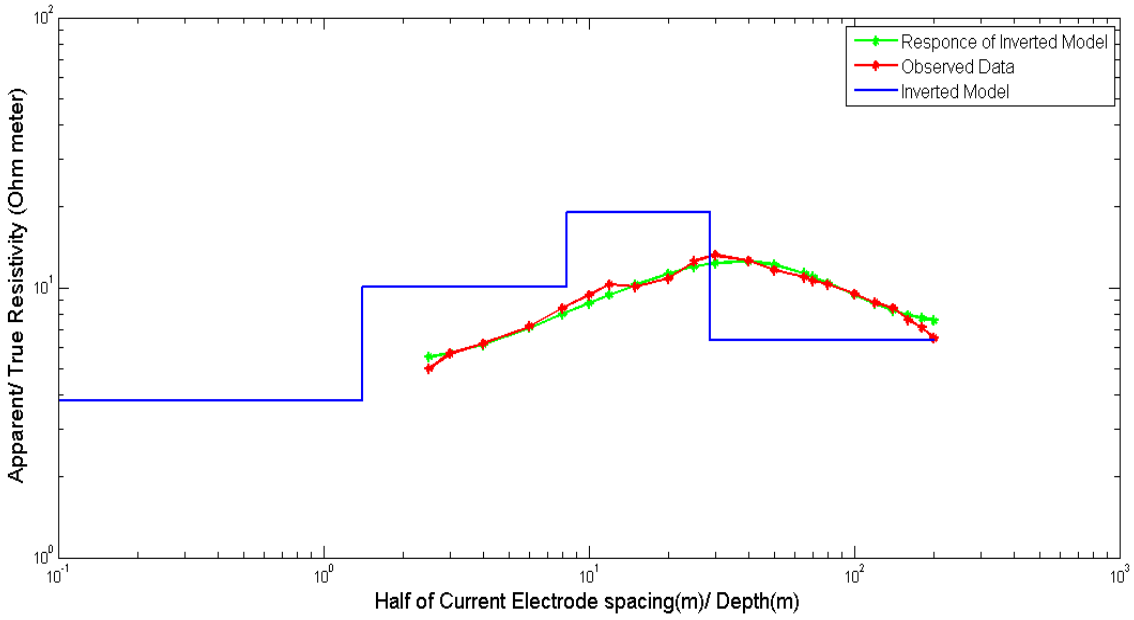

Figure 5.31 Inversion of VES-20.

Table 5.30 Result of VES-21

Parameter	**Search Range**	**Inverted Model**
ρ1(Ω-m)	0.1 - 8	4.18
ρ2(Ω-m)	1 - 15	6.3
ρ3(Ω-m)	2 - 34	17.81
ρ4(Ω-m)	1 - 34	12.65
h1(m)	0.1 - 6	1.5
h2(m)	0.5 - 8	6.25
h3(m)	5 - 50	32.25
Response Misfit (RRMSE)	0.072	

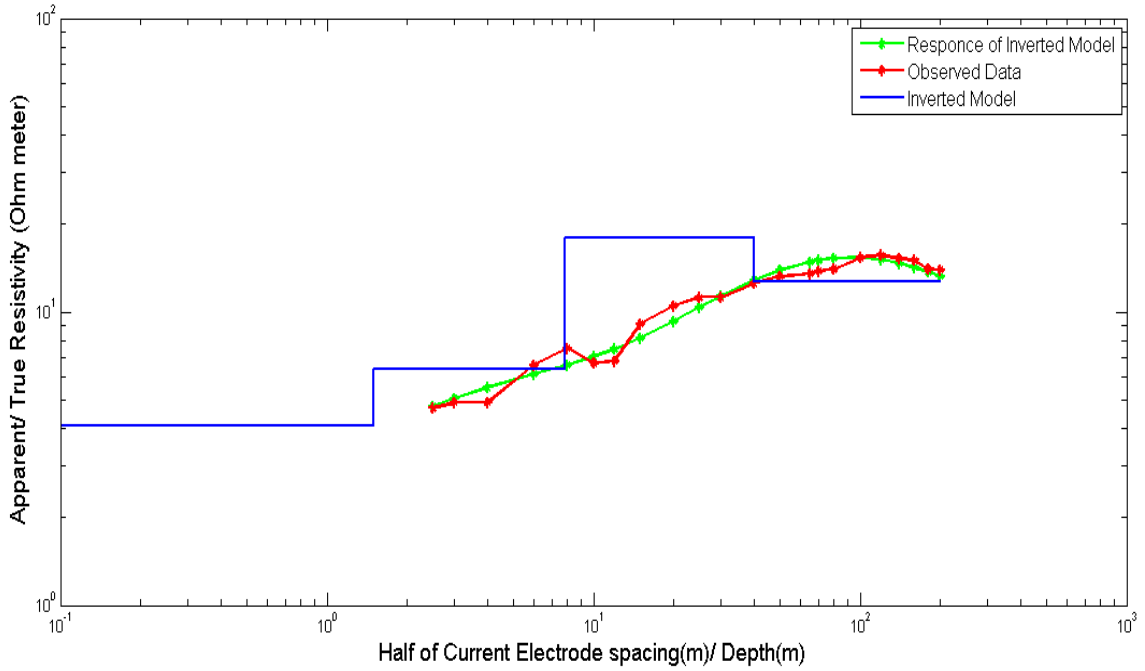

Figure 5.32 Inversion of VES-21.

Table 5.31 Result of VES-22

Parameter	**Search Range**	**Inverted Model**
ρ1(Ω-m)	2 - 32	10.6
ρ2(Ω-m)	5 - 35	20.1
ρ3(Ω-m)	2 - 32	10.9
ρ4(Ω-m)	20 - 120	59.2
h1(m)	0.1 - 8	5.08
h2(m)	0.1 - 8	6.7
h3(m)	5 - 60	33.3
Response Misfit (RRMSE)	0.053	

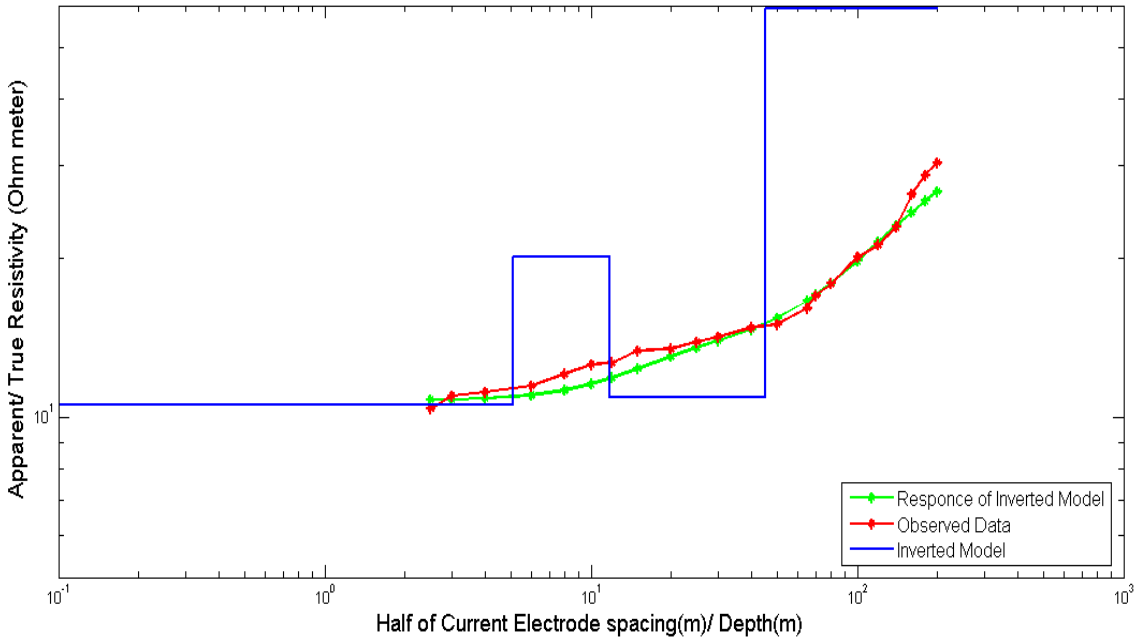

Figure 5.33 Inversion of VES-22.

Table 5.32 Result of VES-23

Parameter	**Search Range**	**Inverted Model**
ρ1(Ω-m)	2 - 32	13.1
ρ2(Ω-m)	2 - 32	16.5
ρ3(Ω-m)	2 - 40	18.6
ρ4(Ω-m)	5 - 75	43.6
h1(m)	0.1 - 6	3.6
h2(m)	10 - 70	39.6
h3(m)	10 - 50	14.7
Response Misfit (RRMSE)	0.041	

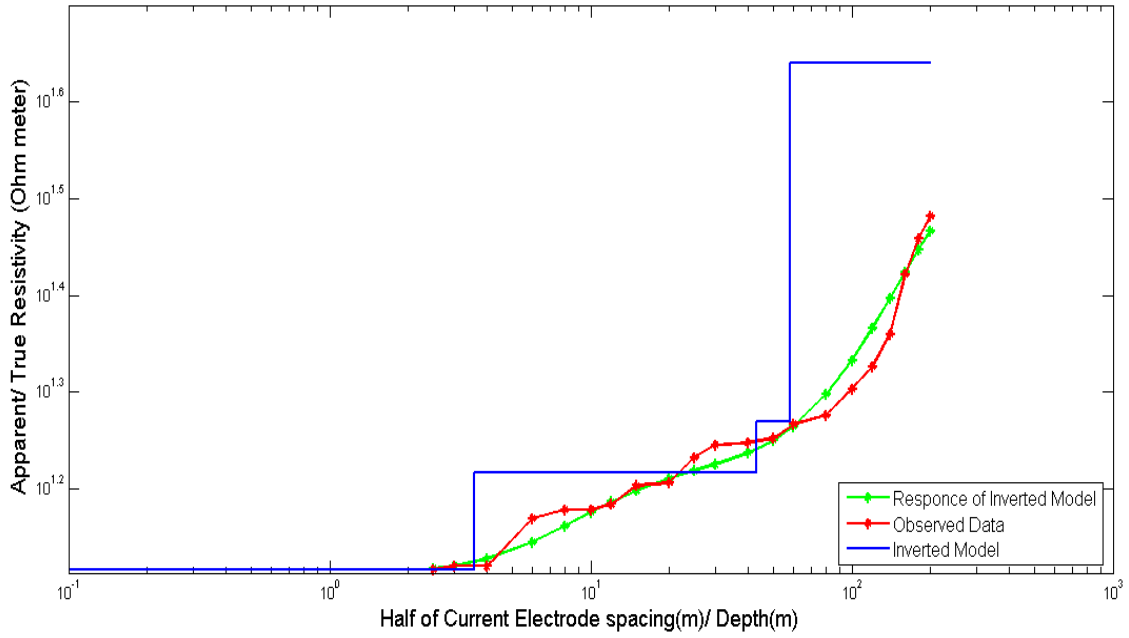

Figure 5.34 Inversion of VES-23.

Table 5.33 Result of VES-24

Parameter	**Search Range**	**Inverted Model**
ρ1(Ω-m)	2 - 25	9.02
ρ2(Ω-m)	2 - 32	12.8
ρ3(Ω-m)	2 - 32	14.7
ρ4(Ω-m)	35 - 100	51.5
h1(m)	0.1 - 8	3.02
h2(m)	2 - 40	18.2
h3(m)	10 - 80	35.8
Response Misfit (RRMSE)	0.046	

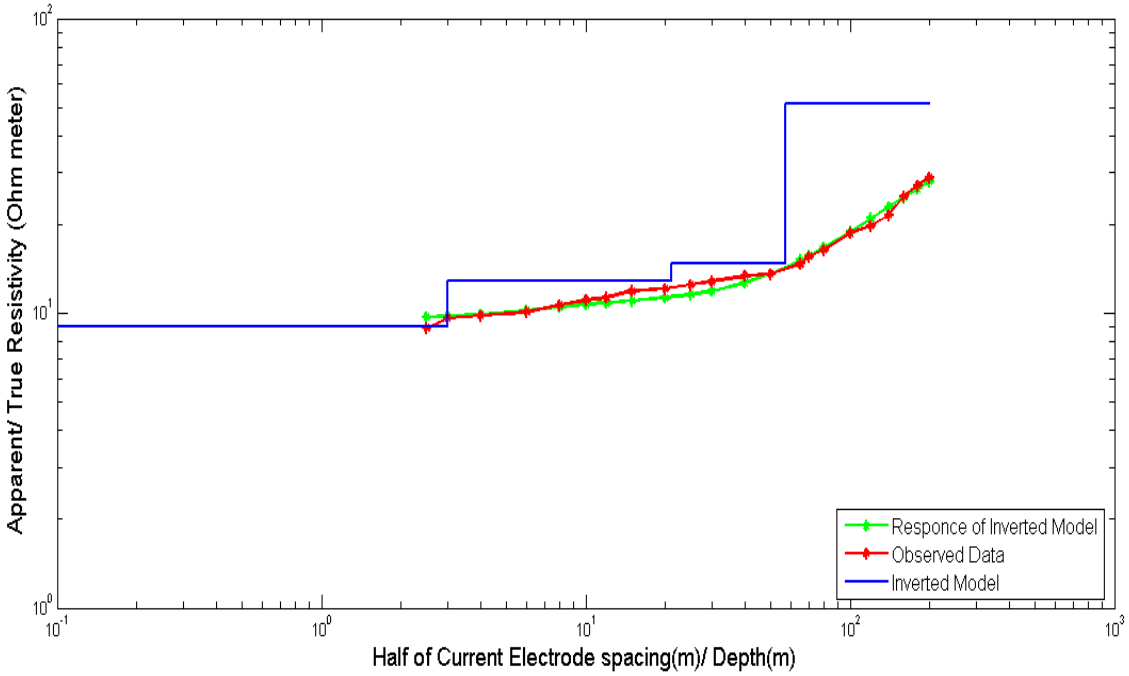

Figure 5.35 Inversion of VES-24.

Table 5.34 Result of VES-25

Parameter	**Search Range**	**Inverted Model**
ρ1(Ω-m)	10 - 60	42.2
ρ2(Ω-m)	5 - 30	9.3
ρ3(Ω-m)	2 - 32	21.5
ρ4(Ω-m)	25 - 100	58.6
h1(m)	0.1 - 6	0.8
h2(m)	2 - 32	18.9
h3(m)	10 - 80	38.3
Response Misfit (RRMSE)	0.053	

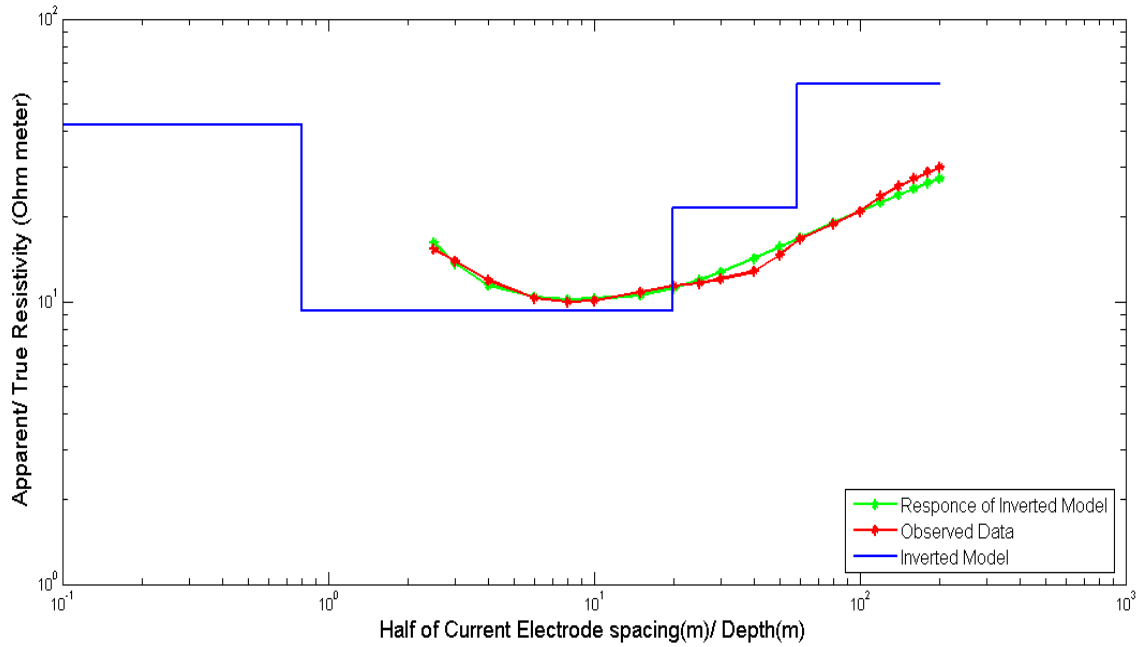

Figure 5.36 Inversion of VES-25.

Table 5.35 Result of VES-26

Parameter	**Search Range**	**Inverted Model**
ρ1(Ω-m)	0.1 - 6	4.22
ρ2(Ω-m)	2 - 32	10.5
ρ3(Ω-m)	2 - 40	15.5
ρ4(Ω-m)	0.1 - 8	4.49
h1(m)	0.1 - 6	1.45
h2(m)	0.5 - 15	7.46
h3(m)	10 - 80	42
Response Misfit (RRMSE)	0.046	

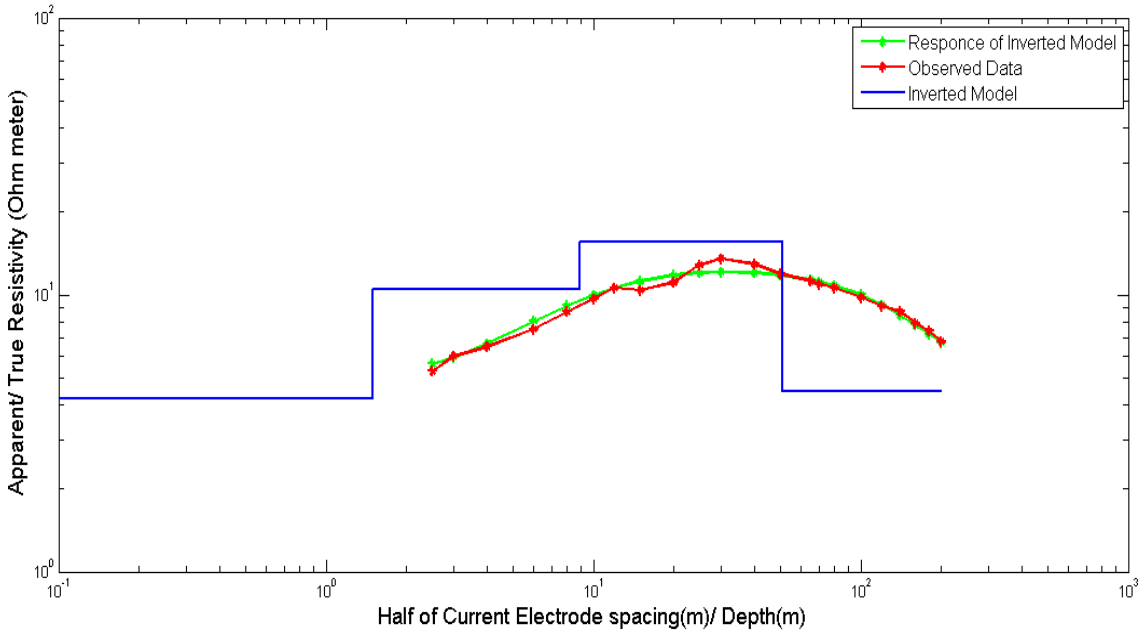

Figure 5.37 Inversion of VES-26.

Table 5.36 Result of VES-27

Parameter	**Search Range**	**Inverted Model**
ρ1(Ω-m)	5 - 60	33.7
ρ2(Ω-m)	0.1 - 8	3.2
ρ3(Ω-m)	5 - 40	16.3
ρ4(Ω-m)	5 - 55	27.6
h1(m)	0.1 - 6	2.5
h2(m)	2 - 32	16.5
h3(m)	10 - 75	39.5
Response Misfit (RRMSE)	0.083	

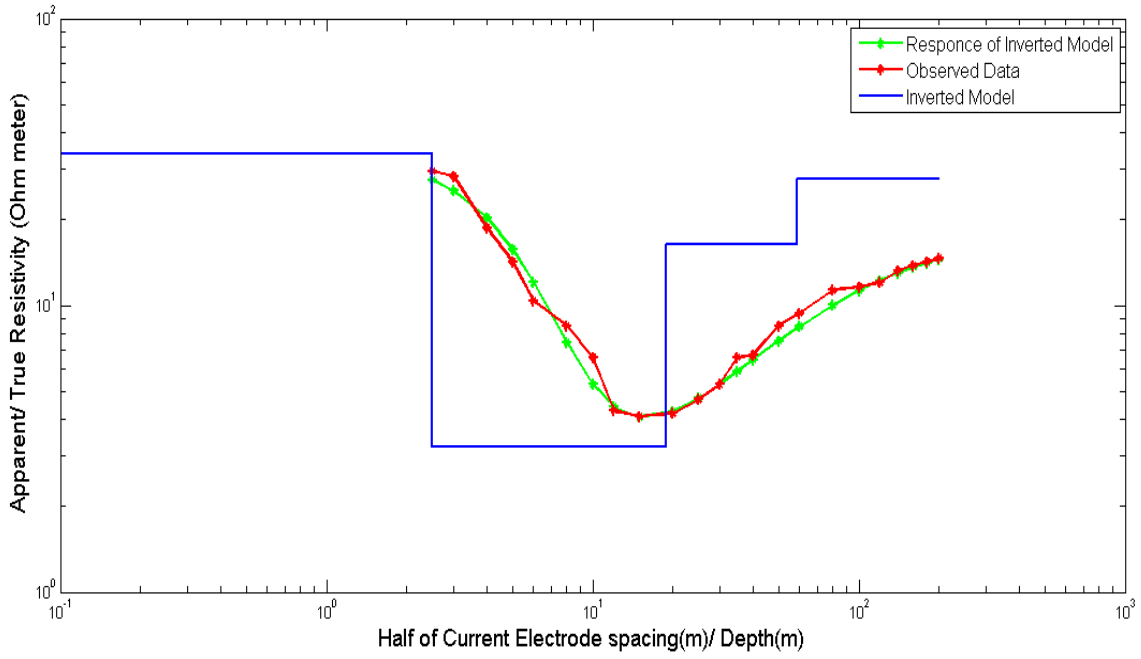

Figure 5.38 Inversion of VES-27.

Table-5.37 Result of VES-28

Parameter	**Search Range**	**Inverted Model**
ρ1(Ω-m)	5 - 55	33.2
ρ2(Ω-m)	2 - 32	9.5
ρ3(Ω-m)	5 - 55	17.1
ρ4(Ω-m)	4 - 40	26.1
h1(m)	0.1 - 6	2.3
h2(m)	2 - 32	16.1
h3(m)	5 - 60	35.9
Response Misfit (RRMSE)	0.058	

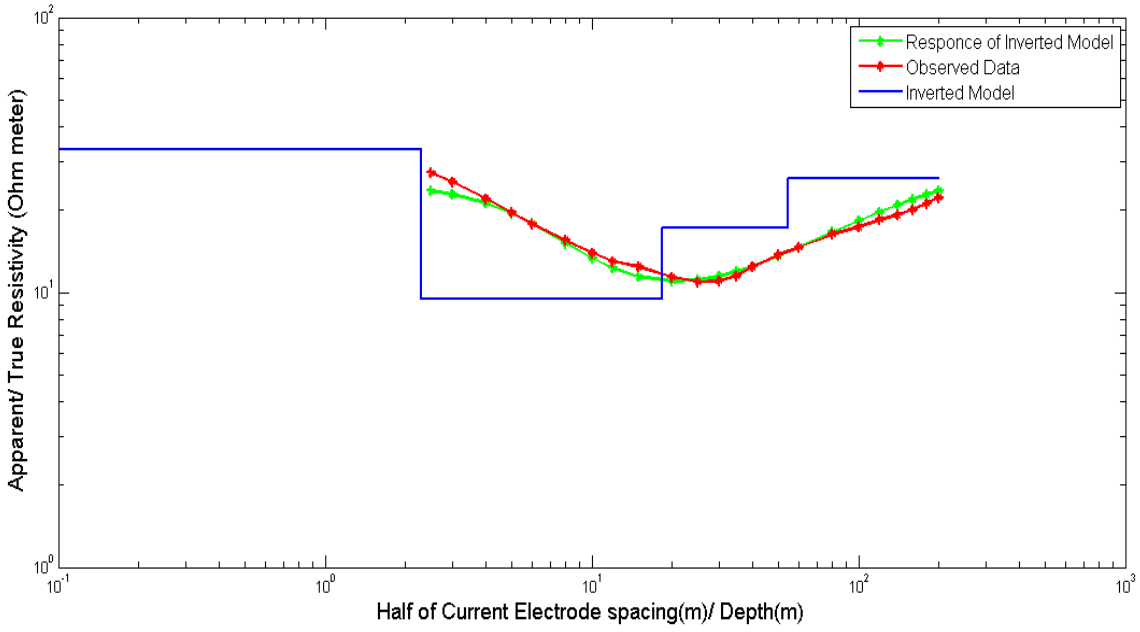

Figure 5.39 Inversion of VES-28.

Table 5.38 Result of VES-29

Parameter	**Search Range**	**Inverted Model**
ρ1(Ω-m)	5 - 60	35.6
ρ2(Ω-m)	1 - 16	3.7
ρ3(Ω-m)	2 - 32	13.8
ρ4(Ω-m)	5 - 50	24
h1(m)	0.1 - 6	2.5
h2(m)	2 - 32	17.6
h3(m)	5 - 70	30.1
Response Misfit (RRMSE)	0.094	

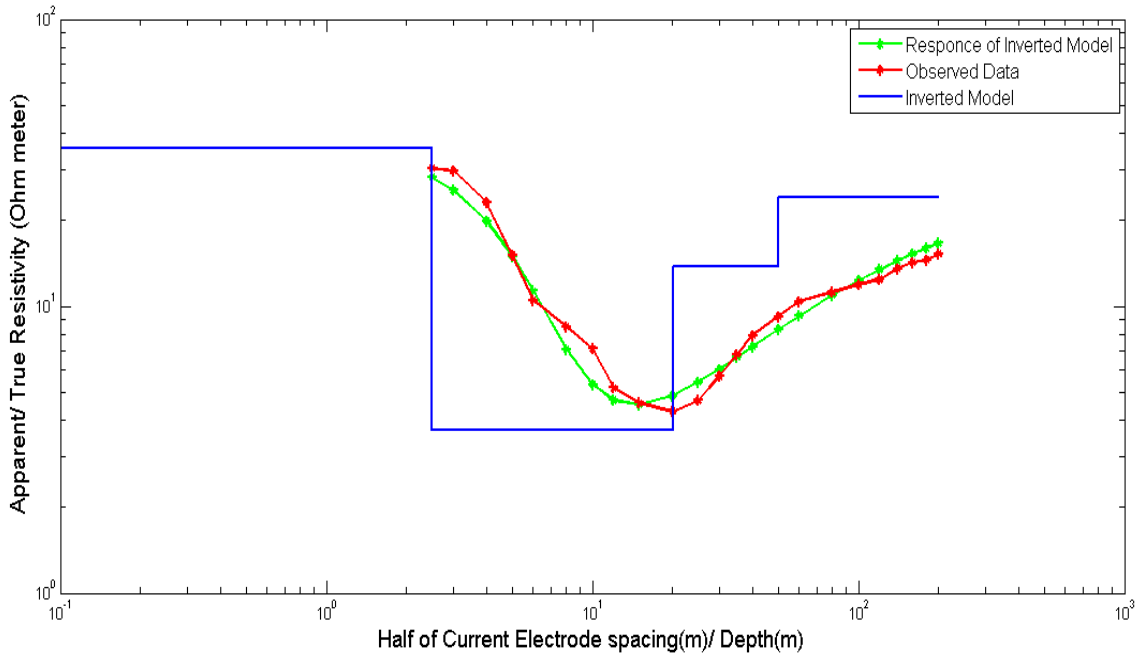

Figure 5.40 Inversion of VES-29.

Table 5.39 Result of VES-30

Parameter	**Search Range**	**Inverted Model**
ρ1(Ω-m)	100 - 400	237
ρ2(Ω-m)	4 - 32	14
ρ3(Ω-m)	15 - 75	35.4
ρ4(Ω-m)	350 - 700	496
ρ5(Ω-m)	650 - 1100	818
h1(m)	0.1 - 6	0.6
h2(m)	0.1 - 6	1
h3(m)	1 - 32	10.1
h4(m)	10 - 80	50.5
Response Misfit (RRMSE)	0.046	

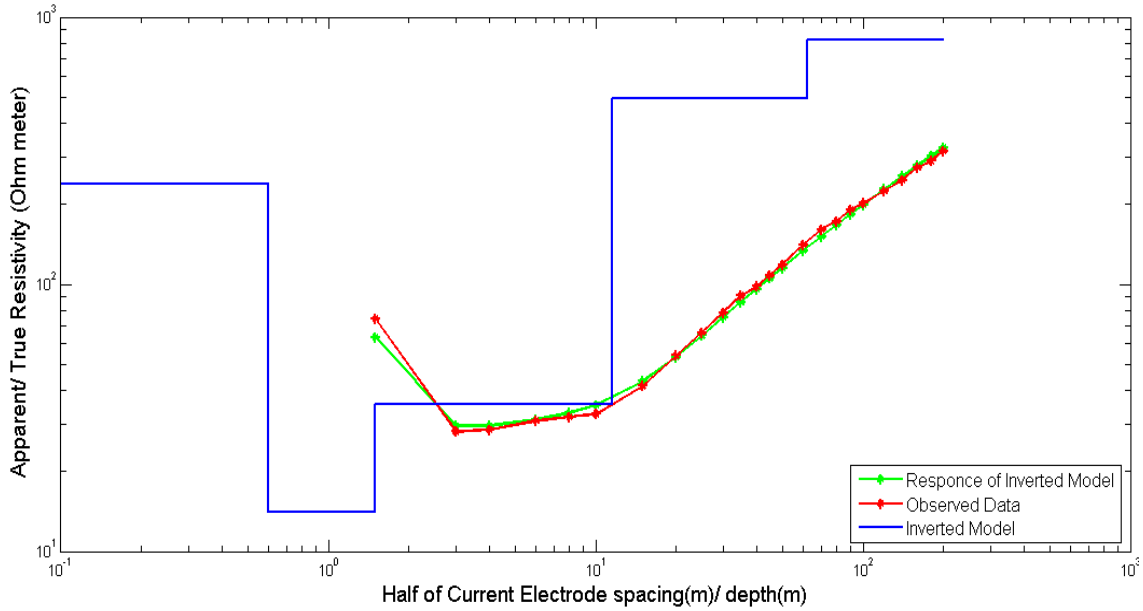

Figure 5.41 Inversion of VES-30.

Table 5.40 Result of VES-31

Parameter	**Search Range**	**Inverted Model**
ρ1(Ω-m)	30 - 150	99.1
ρ2(Ω-m)	120 - 320	248
ρ3(Ω-m)	2000 - 7500	3925
ρ4(Ω-m)	30 - 150	82.8
h1(m)	0.1 - 8	5.9
h2(m)	0.1 - 8	6.8
h3(m)	30 - 150	59.5
Response Misfit (RRMSE)	0.041	

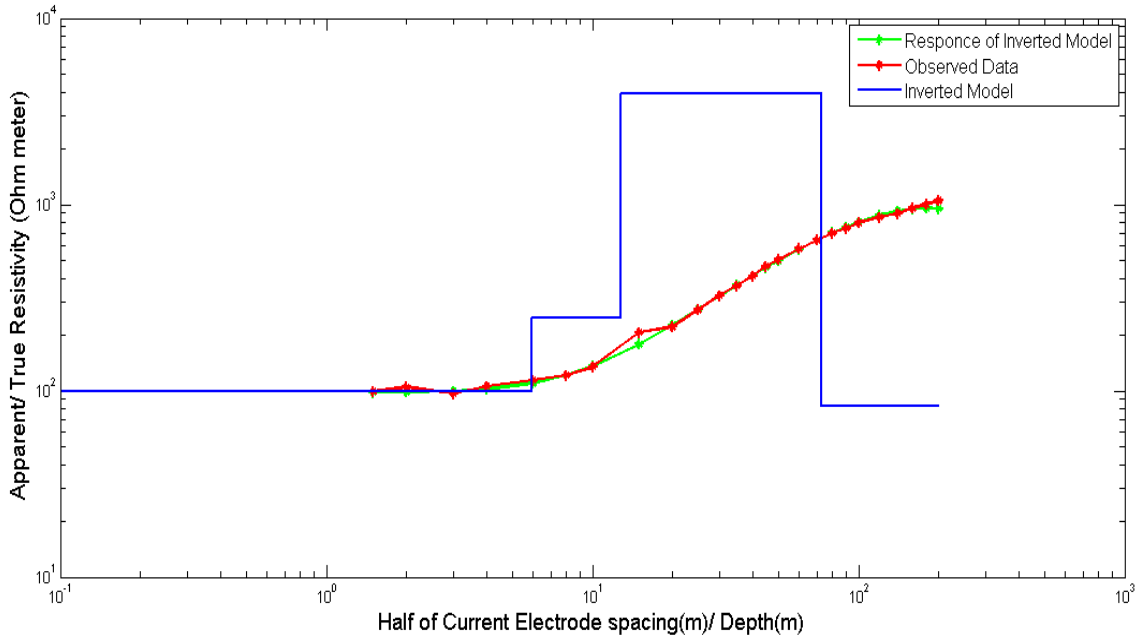

Figure 5.42 Inversion of VES-31.

Table 5.41 Result of VES-32

Parameter	**Search Range**	**Inverted Model**
ρ1(Ω-m)	20 - 110	77.4
ρ2(Ω-m)	5 - 50	16.5
ρ3(Ω-m)	5 - 50	11.6
ρ4(Ω-m)	4 - 40	24.6
ρ5(Ω-m)	20 - 80	56.1
h1(m)	0.1 - 6	0.4
h2(m)	1 - 32	8.5
h3(m)	1 - 32	8.3
h4(m)	15 - 150	79.5
Response Misfit (RRMSE)	0.056	

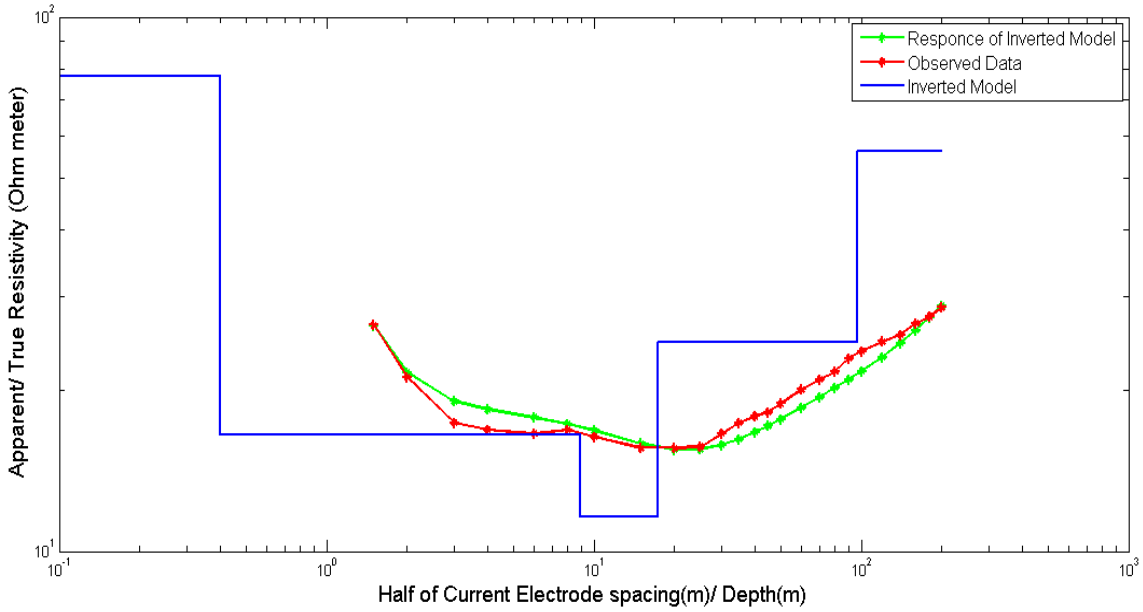

Figure 5.43 Inversion of VES-32.

Table 5.42 Result of VES-33

Parameter	**Search Range**	**Inverted Model**
ρ1(Ω-m)	2 - 32	16.3
ρ2(Ω-m)	2 - 32	8.9
ρ3(Ω-m)	2 - 56	19.4
ρ4(Ω-m)	10 - 100	55.3
h1(m)	0.1 - 6	0.9
h2(m)	0.1 - 8	3.6
h3(m)	5 - 55	17.9
Response Misfit (RRMSE)	0.048	

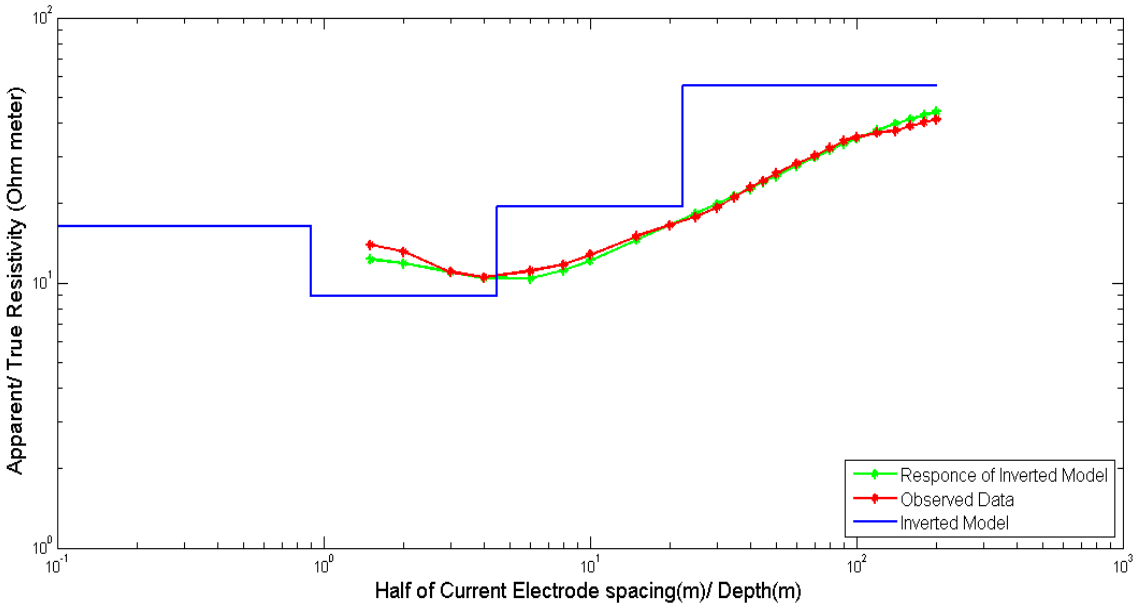

Figure 5.44 Inversion of VES-33.

Table 5.43 Result of VES-34

Parameter	**Search Range**	**Inverted Model**
ρ1(Ω-m)	20 - 60	43.9
ρ2(Ω-m)	3 - 30	14.5
ρ3(Ω-m)	2 - 32	11.1
ρ4(Ω-m)	10 - 80	39.7
ρ5(Ω-m)	20 - 140	62
h1(m)	0.1 - 6	1
h2(m)	0.1 - 8	4.7
h3(m)	1 - 32	6.7
h4(m)	10 - 100	47.6
Response Misfit (RRMSE)	0.023	

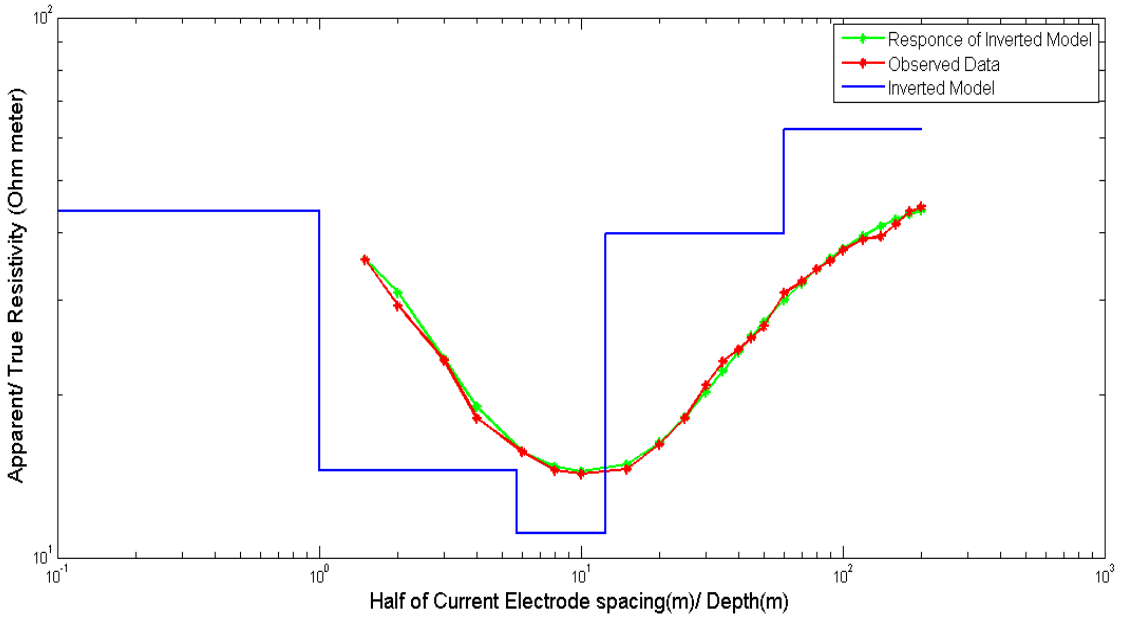

Figure 5.45 Inversion of VES-34.

Table 5.44 Result of VES-35

Parameter	**Search Range**	**Inverted Model**
ρ1(Ω-m)	20 - 120	69.4
ρ2(Ω-m)	2 - 32	24.3
ρ3(Ω-m)	110 - 320	194
ρ4(Ω-m)	650 - 1500	928
h1(m)	0.1 - 6	0.8
h2(m)	1 - 32	2.4
h3(m)	10 - 100	42.9
Response Misfit (RRMSE)	0.059	

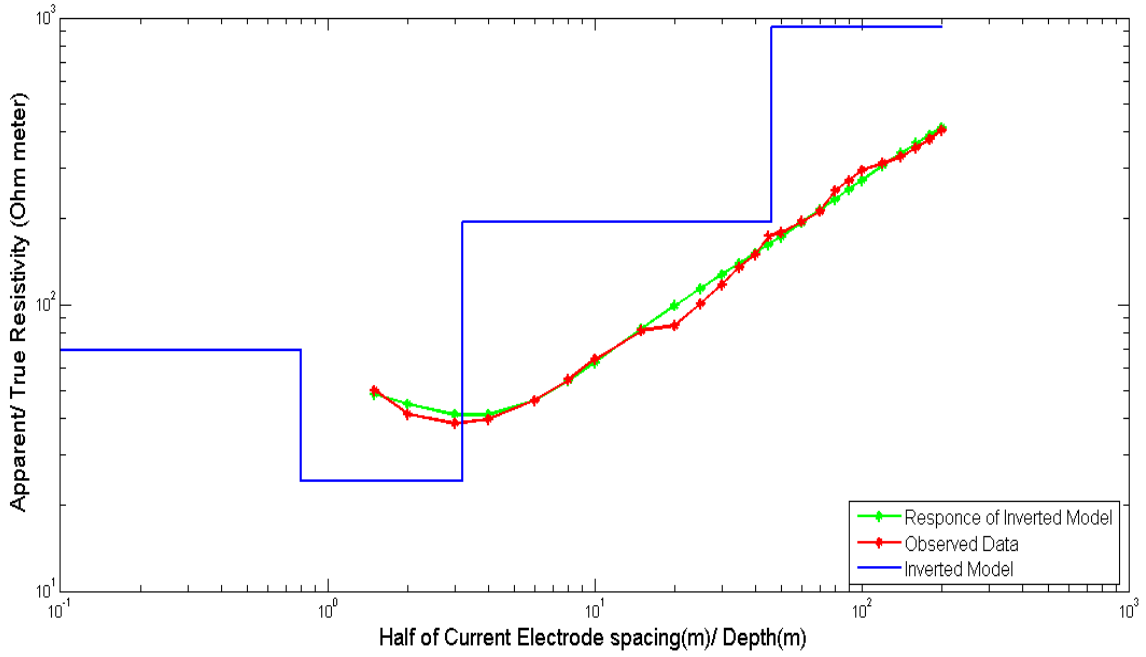

Figure 5.46 Inversion of VES-35.

Table 5.45: Result of VES-36

Parameter	**Search Range**	**Inverted Model**
ρ1(Ω-m)	5 - 40	19
ρ2(Ω-m)	50 - 200	112
ρ3(Ω-m)	200 - 500	384
ρ4(Ω-m)	1 - 20	5.8
ρ5(Ω-m)	15 - 75	36.4
h1(m)	0.1 - 6	0.9
h2(m)	1 - 32	8.5
h3(m)	1 - 32	7
h4(m)	10 - 60	24.2
Response Misfit (RRMSE)	0.081	

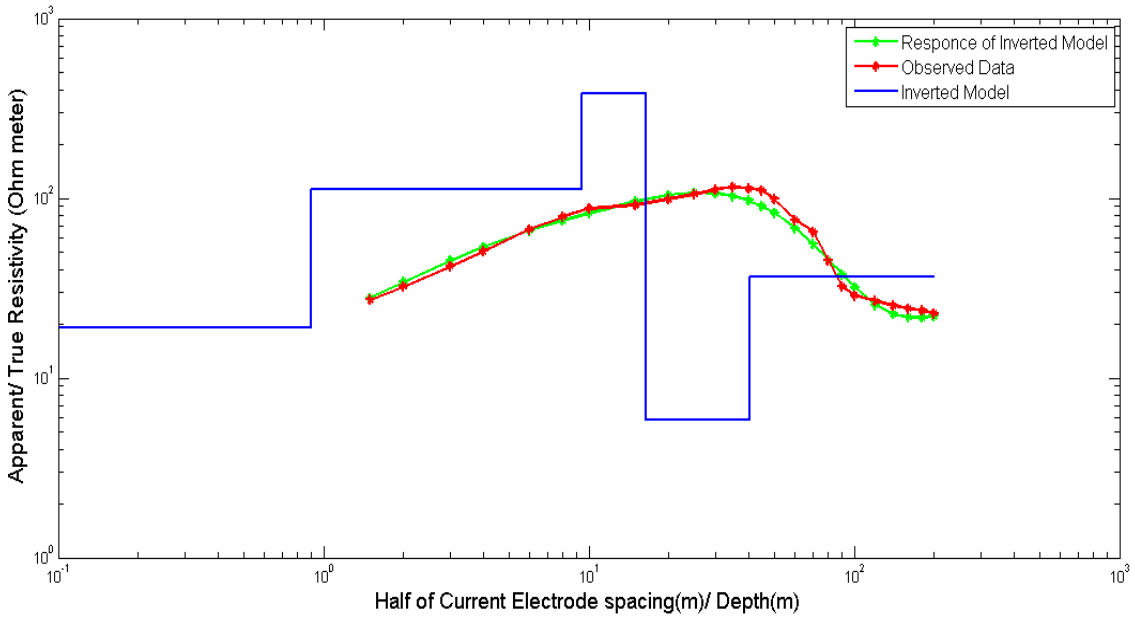

Figure 5.47 Inversion of VES-36.

Table 5.46 Result of VES-37

Parameter	Search Range	Inverted Model
ρ1(Ω-m)	2 - 32	11.7
ρ2(Ω-m)	50 - 200	122
ρ3(Ω-m)	25 - 100	43.2
ρ4(Ω-m)	100 - 400	228
ρ5(Ω-m)	25 - 150	58.4
h1(m)	0.1 - 6	0.9
h2(m)	1 - 24	2.9
h3(m)	1 - 32	6.1
h4(m)	20 - 120	54.7
Response Misfit (RRMSE)	0.052	

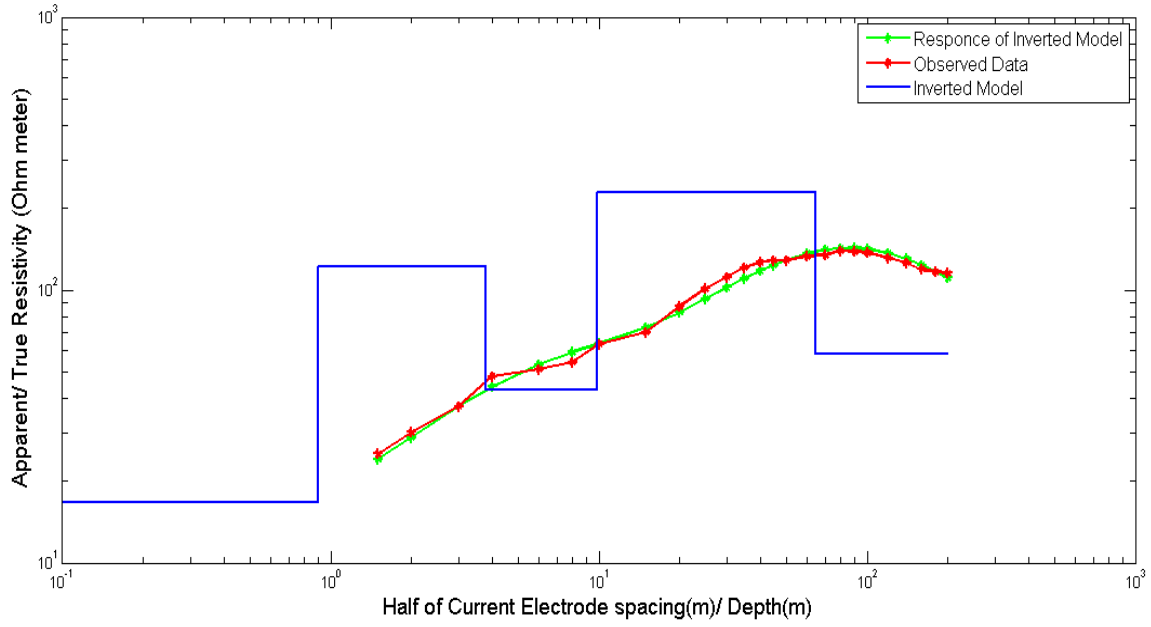

Figure 5.48 Inversion of VES-37.

Table 5.47 Result of VES-38

Parameter	**Search Range**	**Inverted Model**
ρ1(Ω-m)	20 - 120	70.4
ρ2(Ω-m)	5 - 100	49
ρ3(Ω-m)	2 - 32	12.4
ρ4(Ω-m)	3000 - 9000	5989
h1(m)	0.1 - 6	1.3
h2(m)	1 - 24	6.8
h3(m)	1 - 32	14.1
Response Misfit (RRMSE)	0.097	

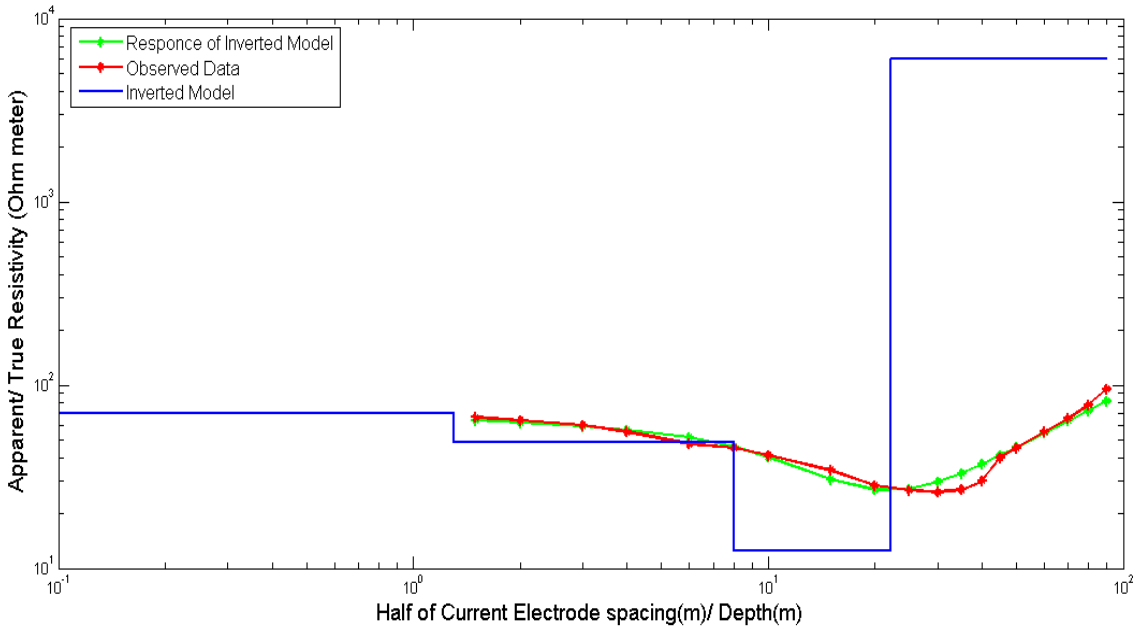

Figure 5.49 Inversion of VES-38.

Table 5.48 Result of VES-39

Parameter	**Search Range**	**Inverted Model**
ρ1(Ω-m)	500 - 1500	953
ρ2(Ω-m)	100 - 400	192
ρ3(Ω-m)	10 - 60	23
ρ4(Ω-m)	100 - 300	217
ρ5(Ω-m)	250 - 1000	722
ρ6(Ω-m)	600 - 2500	1268
h1(m)	0.1 - 6	0.2
h2(m)	1 - 24	11.1
h3(m)	1 - 32	5.8
h4(m)	0.1 - 8	2
h5(m)	2 - 32	8.4
Response Misfit (RRMSE)	0.082	

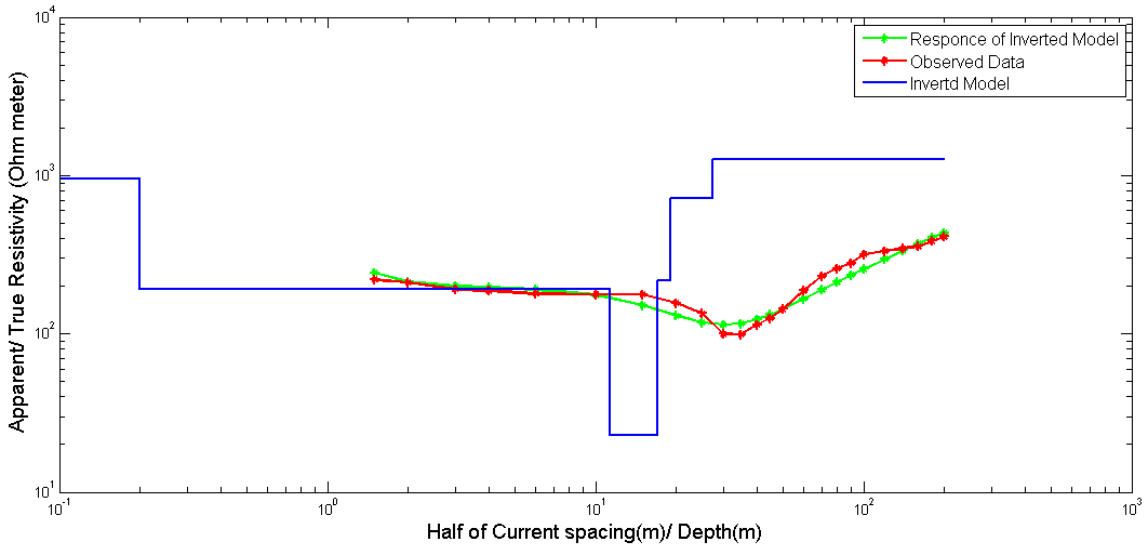

Figure 5.50 Inversion of VES-39.

Table 5.49 Result of VES-40

Parameter	**Search Range**	**Inverted Model**
ρ1(Ω-m)	2 - 22	20.1
ρ2(Ω-m)	50 - 155	114
ρ3(Ω-m)	150 - 450	291
ρ4(Ω-m)	2 - 32	12.7
ρ5(Ω-m)	5 - 50	26.1
h1(m)	0.1 - 6	1
h2(m)	1 - 32	8.3
h3(m)	5 - 40	8
h4(m)	15 - 75	30.4
Response Misfit (RRMSE)	0.098	

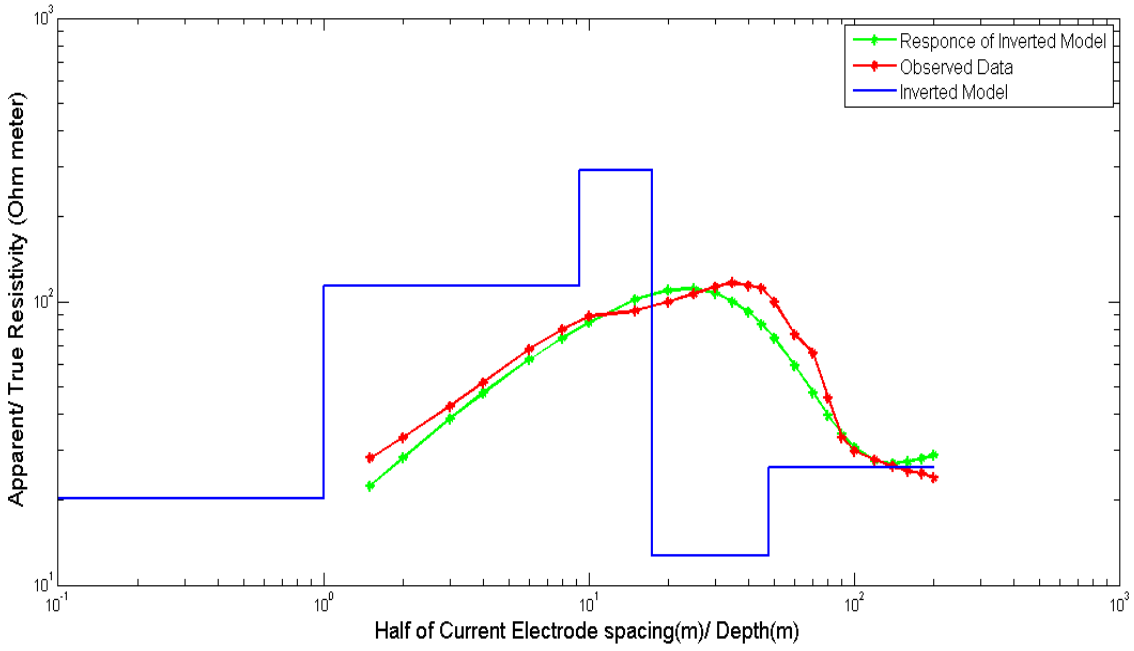

Figure 5.51 Inversion of VES-40.

Table 5.50 Result of VES-41

Parameter	**Search Range**	**Inverted Model**
ρ1(Ω-m)	1 - 18	10
ρ2(Ω-m)	5 - 50	23.2
ρ3(Ω-m)	1 - 24	6.5
ρ4(Ω-m)	20 - 100	55.5
h1(m)	0.1 - 6	1.6
h2(m)	1 - 24	2.7
h3(m)	2 - 32	10.5
Response Misfit (RRMSE)	0.083	

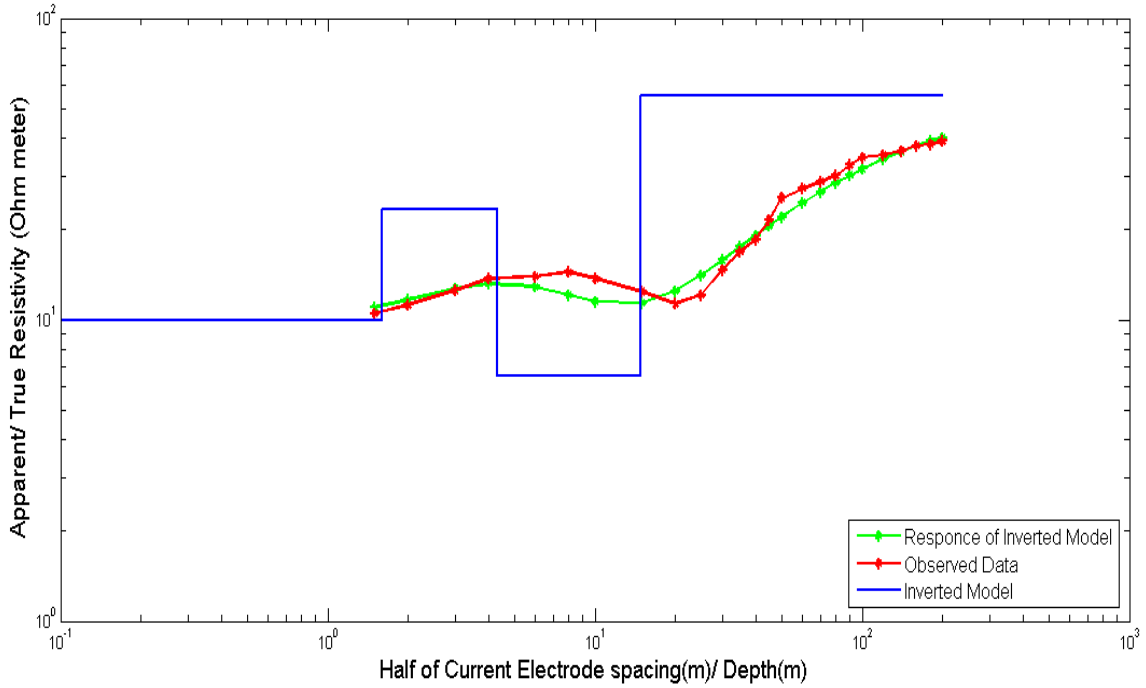

Figure 5.52 Inversion of VES-41.

Table 5.51 Result of VES-42

Parameter	**Search Range**	**Inverted Model**
ρ1(Ω-m)	10 - 75	53.1
ρ2(Ω-m)	2 - 30	9.2
ρ3(Ω-m)	20 - 140	61.8
ρ4(Ω-m)	1 - 20	3.9
ρ5(Ω-m)	10 - 100	37.7
h1(m)	0.2 - 6	0.9
h2(m)	1 - 24	2.1
h3(m)	2 - 32	4.3
h4(m)	10 - 50	13.9
Response Misfit (RRMSE)	0.14	

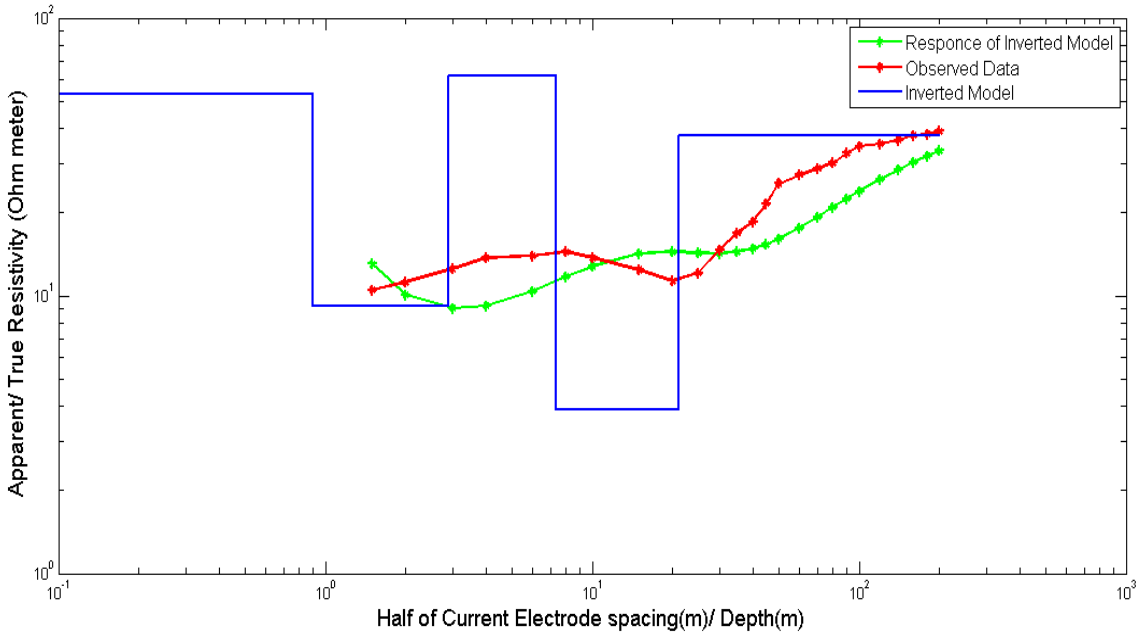

Figure 5.53 Inversion of VES-42.

Table 5.52 Result of VES-43

Parameter	Search Range	Inverted Model
ρ1(Ω-m)	5 - 40	11
ρ2(Ω-m)	6 - 50	14.3
ρ3(Ω-m)	10 - 60	29.7
ρ4(Ω-m)	2 - 30	11.3
ρ5(Ω-m)	600 - 2000	1225
h1(m)	0.1 - 6	0.9
h2(m)	2 - 32	12.6
h3(m)	5 - 50	27.1
h4(m)	20 - 100	43.6
Response Misfit (RRMSE)	0.04	

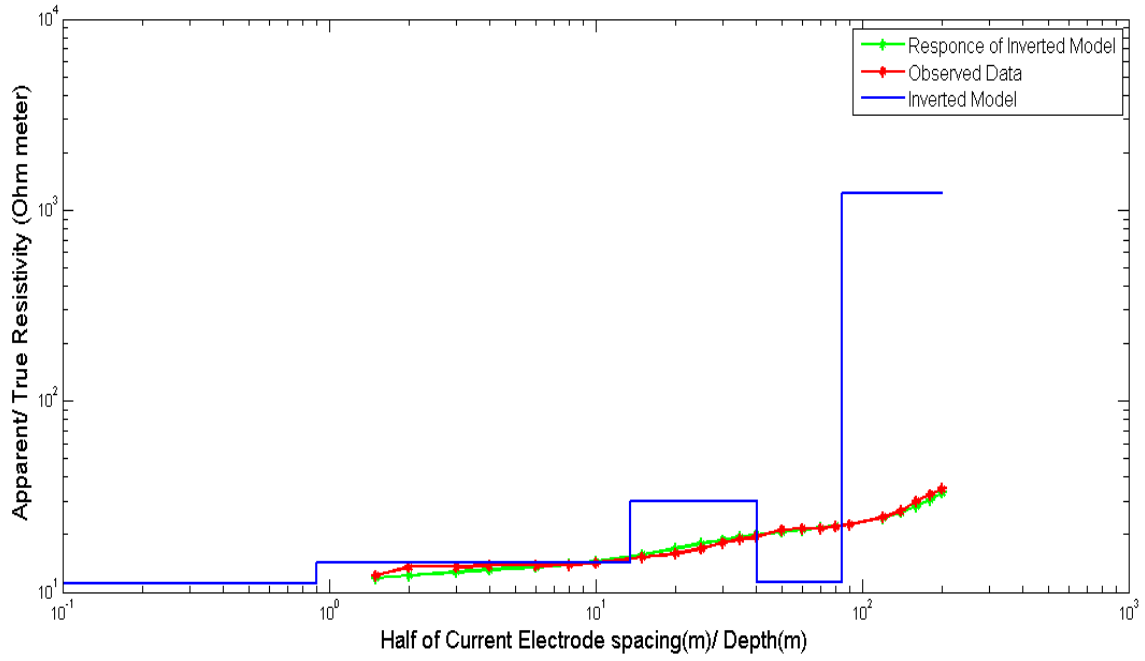

Figure 5.54 Inversion of VES-43.

Table 5.53 Result of VES-44

Parameter	**Search Range**	**Inverted Model**
ρ1(Ω-m)	60 - 250	151
ρ2(Ω-m)	2 - 30	19.4
ρ3(Ω-m)	150 - 500	212
ρ4(Ω-m)	7000 - 18000	10499
h1(m)	0.1 - 6	0.5
h2(m)	2 - 30	11.5
h3(m)	20 - 150	78.5
Response Misfit (RRMSE)	0.063	

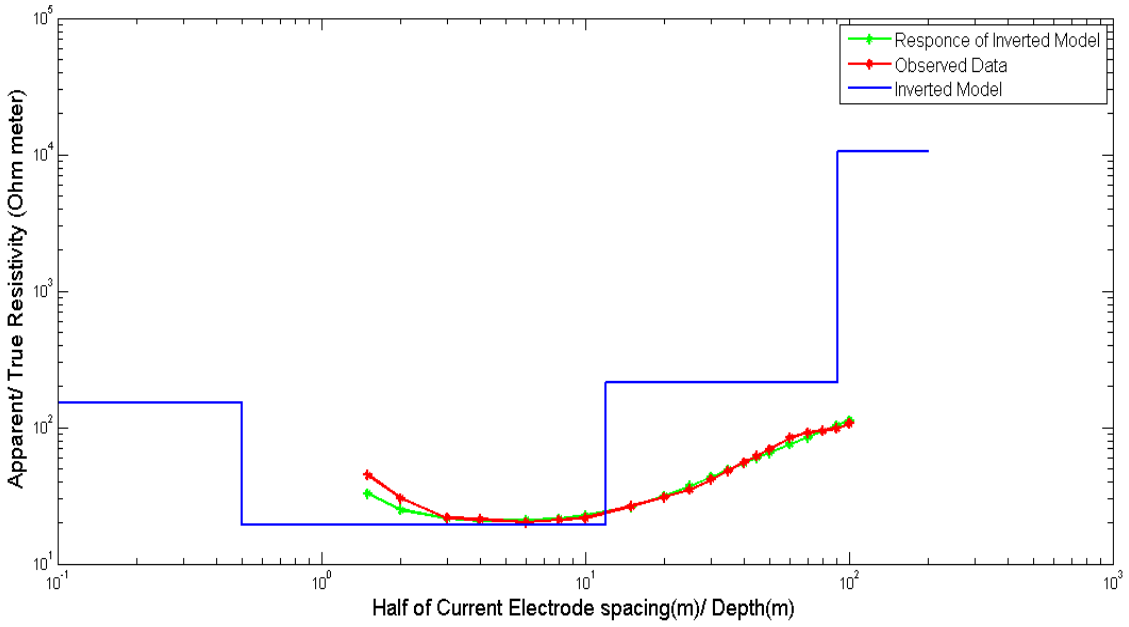

Figure 5.55 Inversion of VES-44.

Table 5.54 Result of VES-45

Parameter	Search Range	Inverted Model
ρ1(Ω-m)	50 - 300	187.4
ρ2(Ω-m)	20 - 200	110.5
ρ3(Ω-m)	5 - 75	32.3
ρ4(Ω-m)	250 - 800	484.2
ρ5(Ω-m)	5000 - 15000	7819
h1(m)	0.1 - 6	0.3
h2(m)	1 - 32	5.3
h3(m)	2 - 40	8.7
h4(m)	50 - 200	105.1
Response Misfit (RRMSE)	0.045	

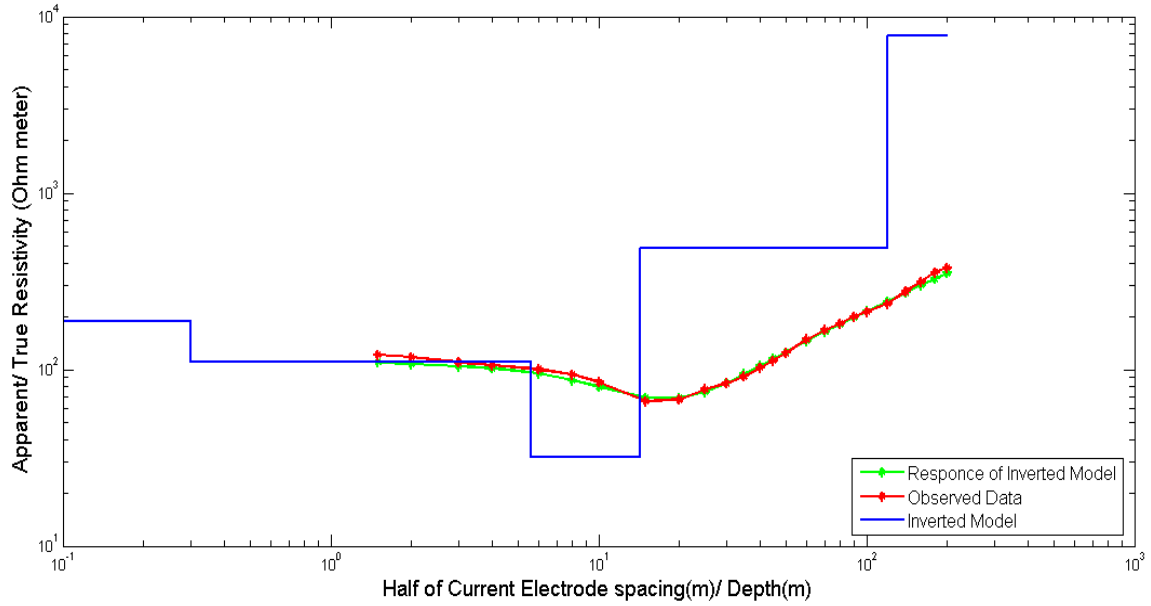

Figure 5.56 Inversion of VES-45.

Table 5.55 Result of VES-46

Parameter	**Search Range**	**Inverted Model**
ρ1(Ω-m)	150 - 500	273
ρ2(Ω-m)	20 - 150	76.1
ρ3(Ω-m)	100 - 500	198
ρ4(Ω-m)	400 - 1200	769
h1(m)	0.5 - 6	0.8
h2(m)	1 - 20	2.5
h3(m)	10 - 80	35.1
Response Misfit (RRMSE)	0.067	

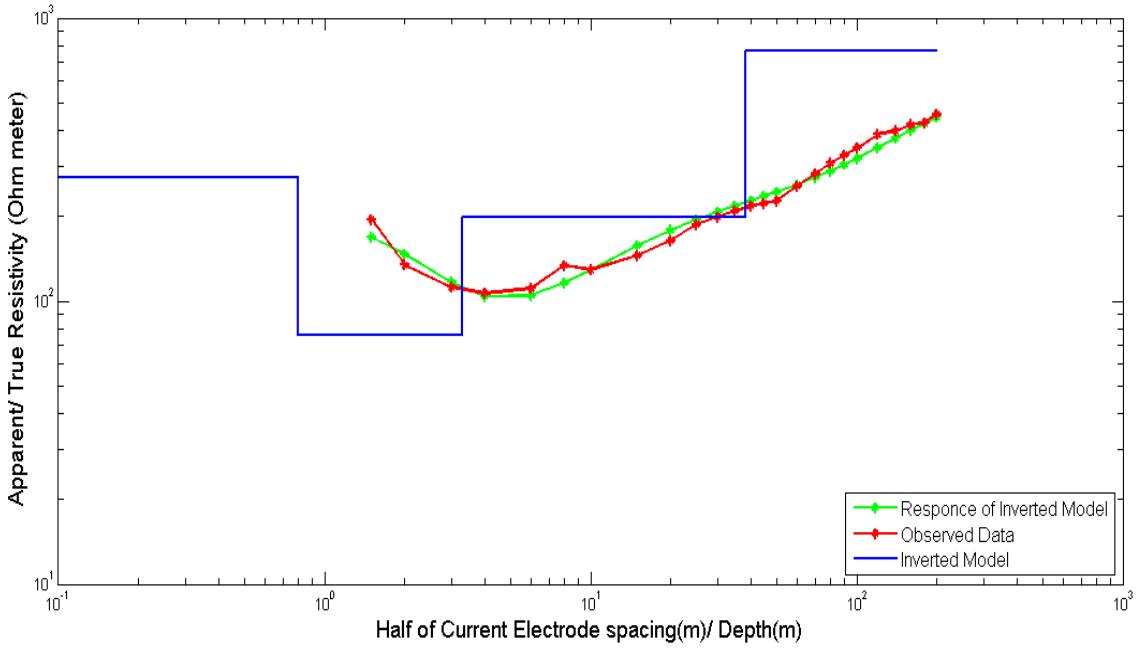

Figure 5.57 Inversion of VES-46.

Table 5.56 Result of VES-47

Parameter	**Search Range**	**Inverted Model**
ρ1(Ω-m)	150 - 500	293
ρ2(Ω-m)	20 - 100	59.1
ρ3(Ω-m)	1000 - 2200	1530
ρ4(Ω-m)	250 - 1000	678
h1(m)	0.5 - 6	1
h2(m)	1 - 32	20.6
h3(m)	20 - 120	55.2
Response Misfit (RRMSE)	0.077	

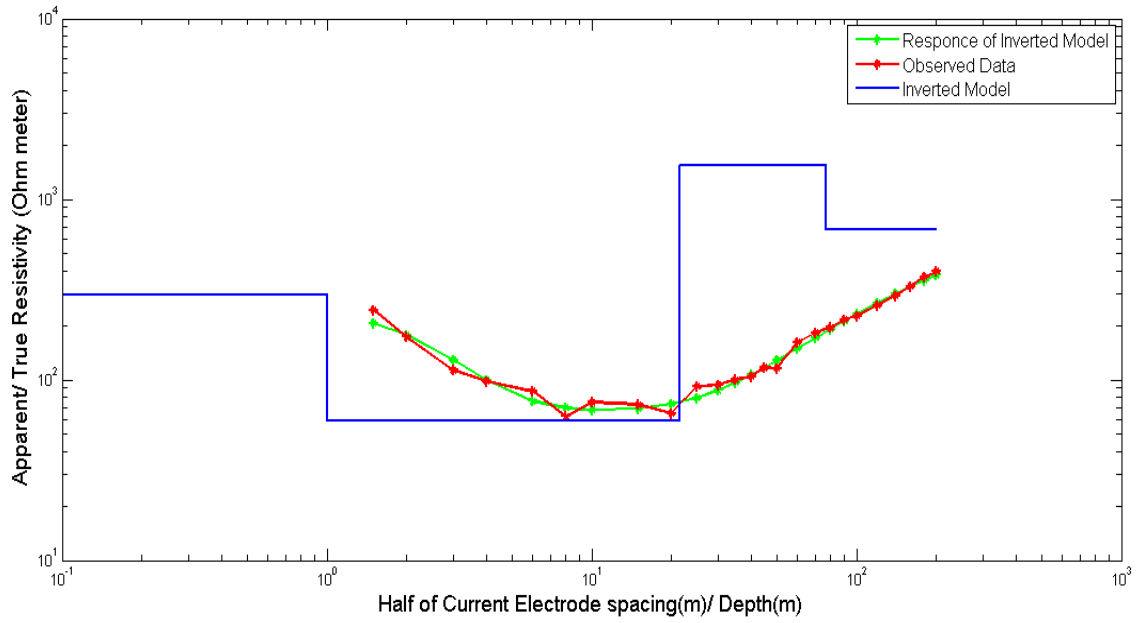

Figure 5.58 Inversion of VES-47.

Table 5.57 Result of VES-48

Parameter	**Search Range**	**Inverted Model**
ρ1(Ω-m)	20 - 120	73.7
ρ2(Ω-m)	5 - 60	27.1
ρ3(Ω-m)	10 - 100	38.3
ρ4(Ω-m)	2000 - 7000	4340
ρ5(Ω-m)	50 - 250	100.9
h1(m)	0.1 - 6	0.5
h2(m)	1 - 30	2.6
h3(m)	5 - 50	18.9
h4(m)	20 - 100	33.7
Response Misfit (RRMSE)	0.031	

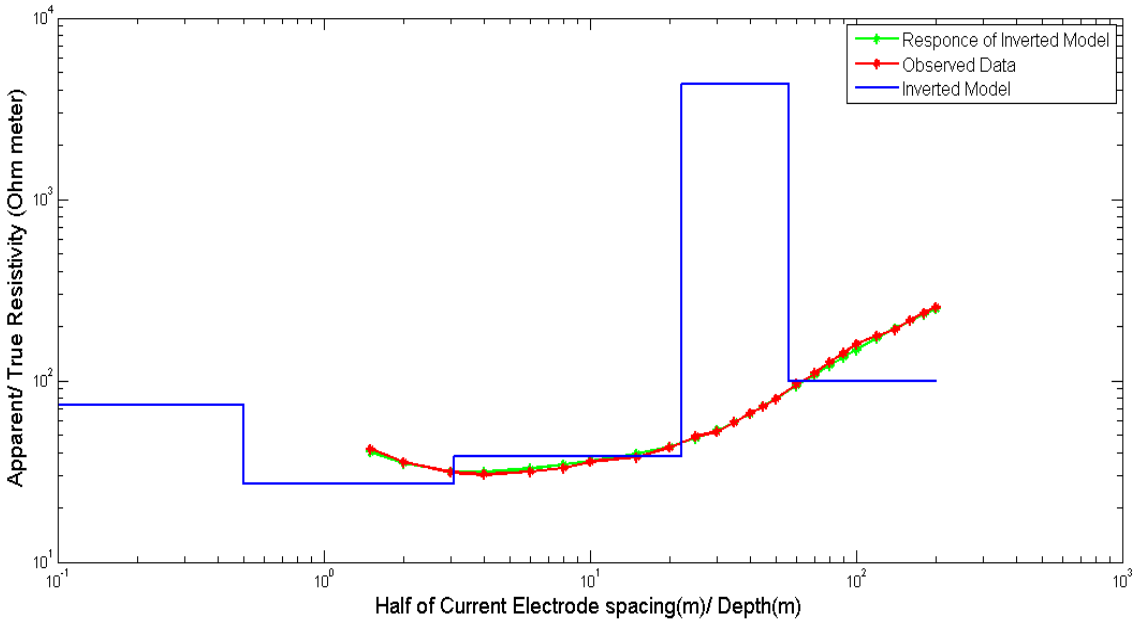

Figure 5.59 Inversion of VES-48.

Table 5.58 Result of VES-49

Parameter	**Search Range**	**Inverted Model**
ρ1(Ω-m)	80 - 300	182
ρ2(Ω-m)	20 - 80	42.7
ρ3(Ω-m)	80 - 400	185
ρ4(Ω-m)	500 - 2000	916
h1(m)	1 - 6	1.9
h2(m)	1 - 32	8.4
h3(m)	10 - 120	52.6
Response Misfit (RRMSE)	0.063	

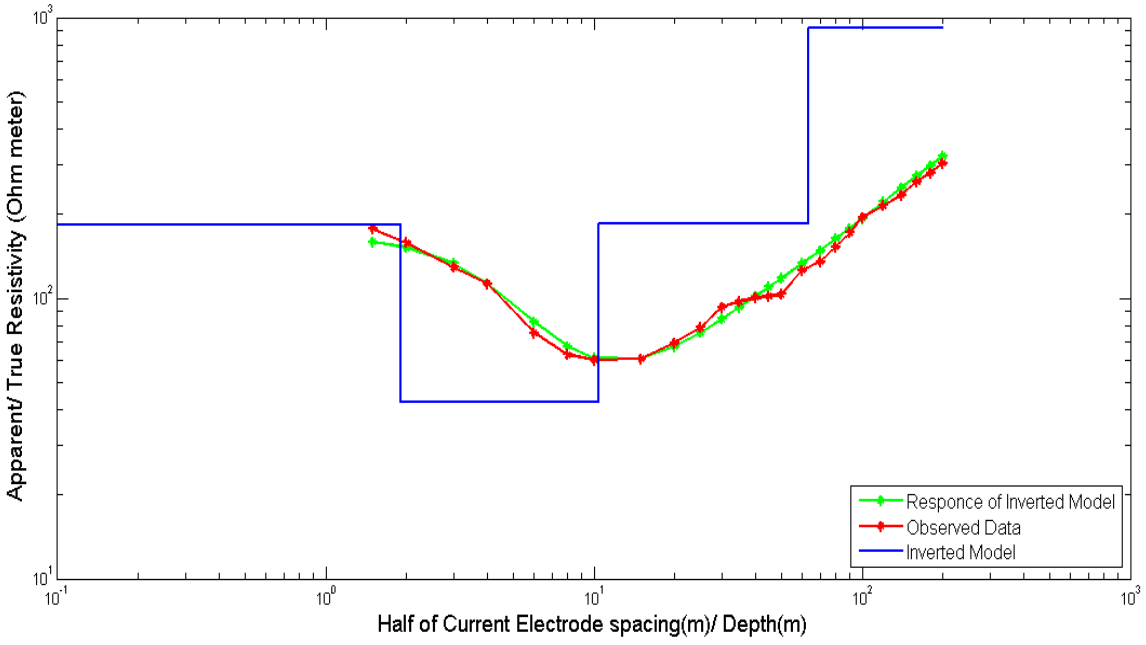

Figure 5.60 Inversion of VES-49.

Table 5.59 Result of VES-50

Parameter	**Search Range**	**Inverted Model**
ρ1(Ω-m)	40 - 200	104
ρ2(Ω-m)	5 - 60	26.2
ρ3(Ω-m)	350 - 1200	704
ρ4(Ω-m)	1500 - 5000	2688
h1(m)	0.1 - 6	5
h2(m)	1 - 32	9.1
h3(m)	20 - 150	70.9
Response Misfit (RRMSE)	0.043	

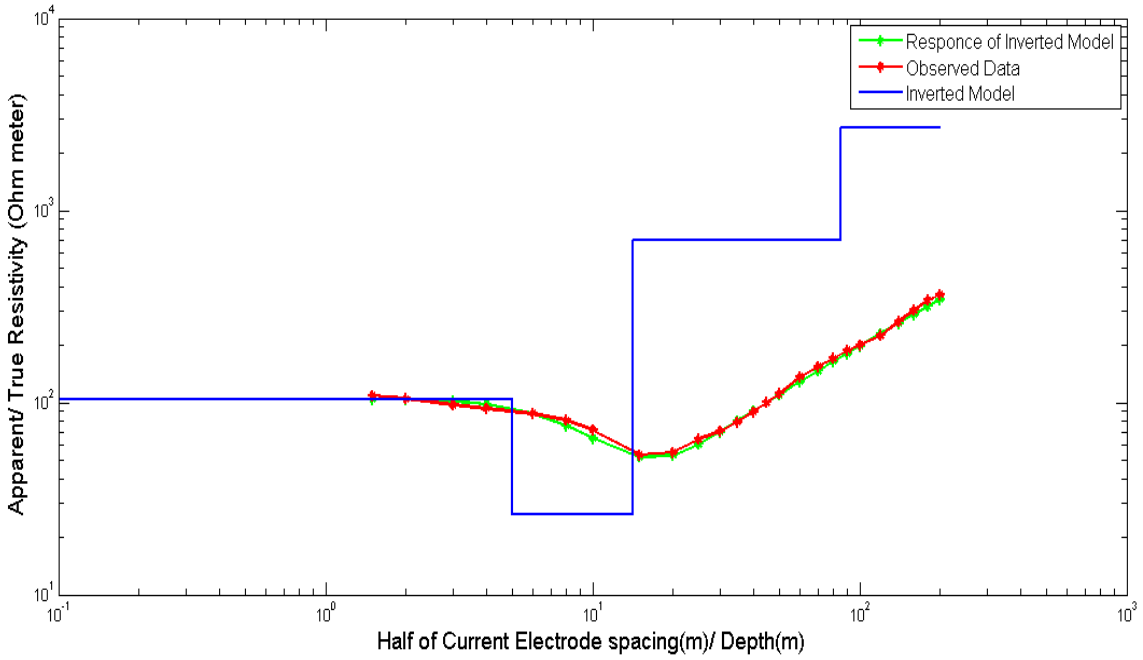

Figure 5.61 Inversion of VES-50.

Table 5.60 Result of VES-51

Parameter	**Search Range**	**Inverted Model**
ρ1(Ω-m)	50 - 200	108
ρ2(Ω-m)	10 - 80	22.3
ρ3(Ω-m)	300 - 800	449
ρ4(Ω-m)	1100 - 2500	1713
h1(m)	0.1 - 6	5.5
h2(m)	1 - 32	6.2
h3(m)	20 - 150	60.5
Response Misfit (RRMSE)	0.038	

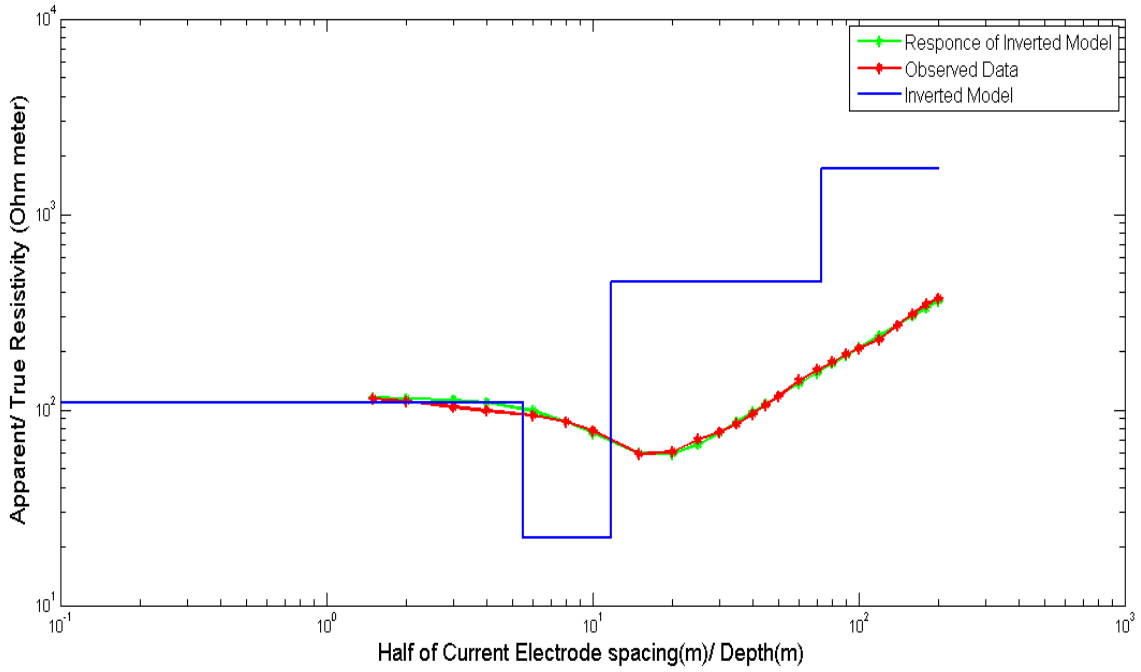

Figure 5.62 Inversion of VES-51.

Table 5.61 Result of VES-52

Parameter	**Search Range**	**Inverted Model**
ρ1(Ω-m)	20 - 120	66.2
ρ2(Ω-m)	1 - 32	11.6
ρ3(Ω-m)	10 - 100	42.2
ρ4(Ω-m)	200 - 1200	645
h1(m)	0.1 - 6	0.7
h2(m)	1 - 32	4.9
h3(m)	10 - 100	16.4
Response Misfit (RRMSE)	0.076	

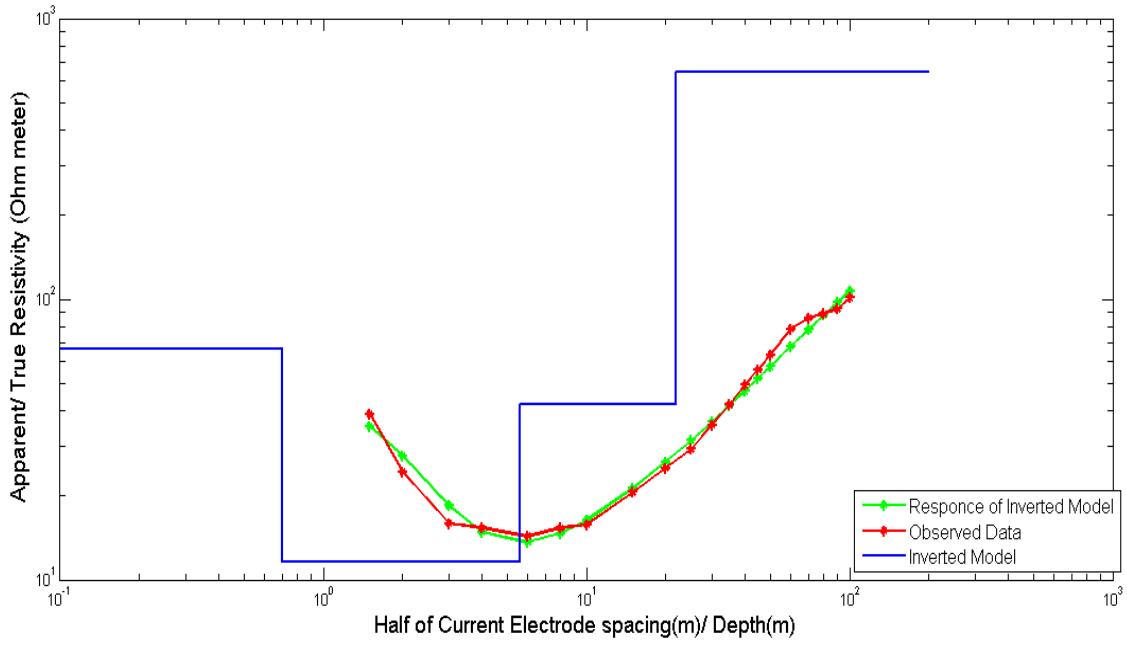

Figure 5.63 Inversion of VES-52.

Table 5.62 Result of VES-53

Parameter	Search Range	Inverted Model
ρ1(Ω-m)	50 - 200	167
ρ2(Ω-m)	30 - 200	97.5
ρ3(Ω-m)	10 - 60	17.1
ρ4(Ω-m)	100 - 600	276
ρ5(Ω-m)	800 - 2500	1357
h1(m)	0.1 - 6	1
h2(m)	1 - 32	1.6
h3(m)	2 - 40	7.4
h4(m)	30 - 150	52.9
Response Misfit (RRMSE)	0.053	

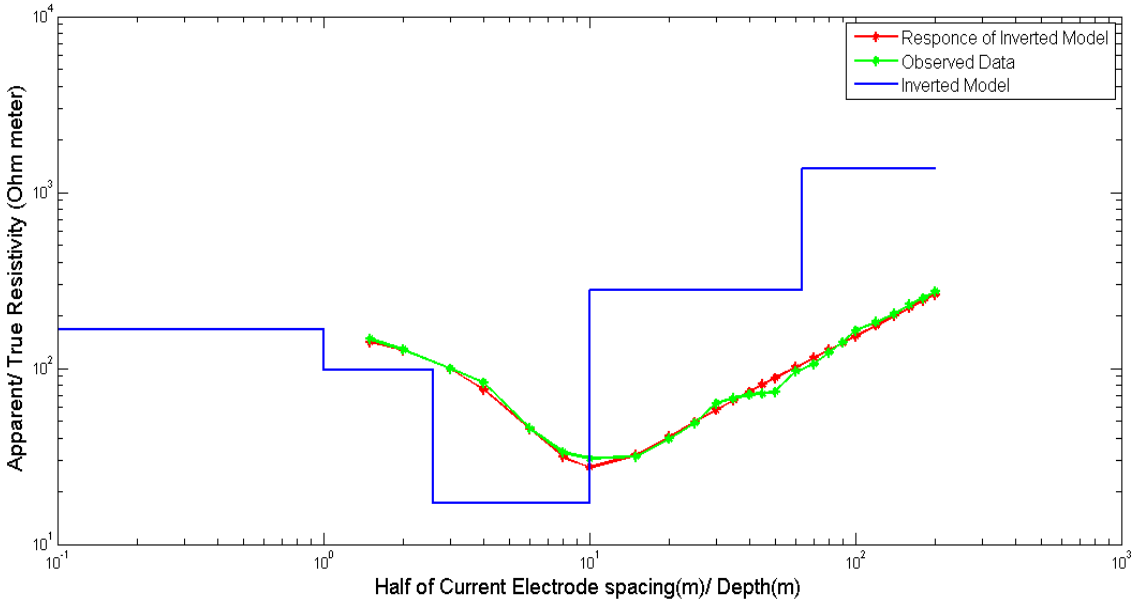

Figure 5.64 Inversion of VES-53.

Table 5.63 Result of VES-54

Parameter	**Search Range**	**Inverted Model**
ρ1(Ω-m)	5 - 40	17.7
ρ2(Ω-m)	5 - 50	10.5
ρ3(Ω-m)	10 - 100	22
ρ4(Ω-m)	10 - 120	36.7
ρ5(Ω-m)	30 - 200	59.9
h1(m)	0.1 - 6	0.9
h2(m)	1 - 32	3.4
h3(m)	10 - 80	15
h4(m)	30 - 150	44.7
Response Misfit (RRMSE)	0.034	

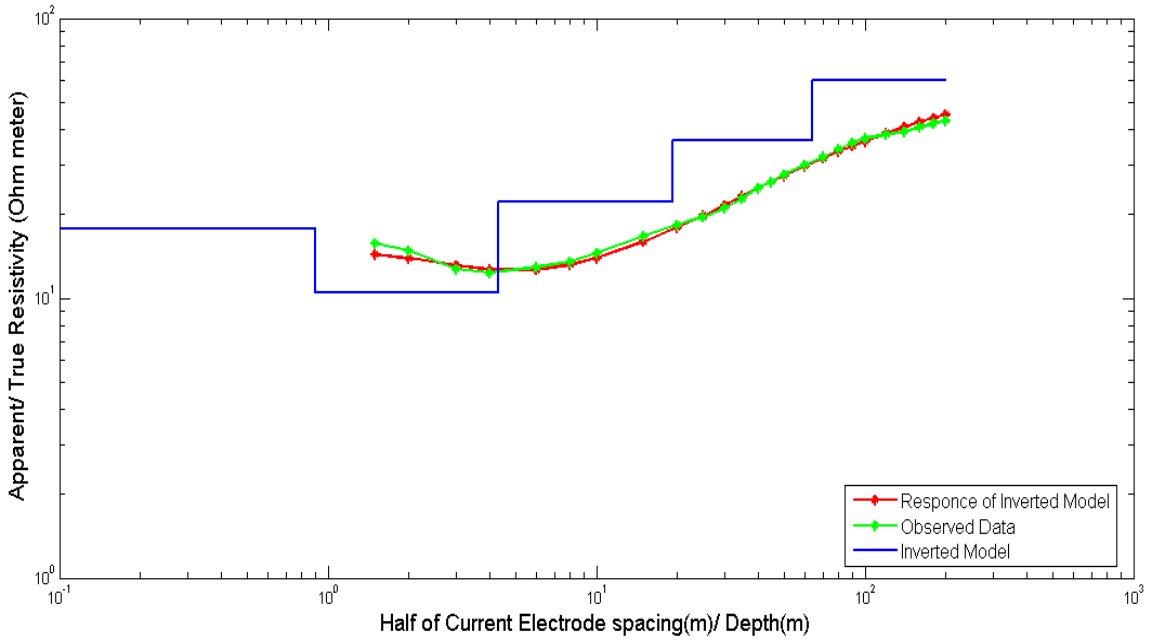

Figure 5.65 Inversion of VES-54.

Table 5.64 Result of VES-55

Parameter	Search Range	Inverted Model
ρ1(Ω-m)	50 - 200	92.3
ρ2(Ω-m)	10 - 100	21.8
ρ3(Ω-m)	10 - 80	16.9
ρ4(Ω-m)	20 - 80	30.5
ρ5(Ω-m)	20 - 120	45.9
h1(m)	0.1 - 6	0.4
h2(m)	1 - 32	8.6
h3(m)	5 - 60	8.5
h4(m)	30 - 250	76.4
Response Misfit (RRMSE)	0.039	

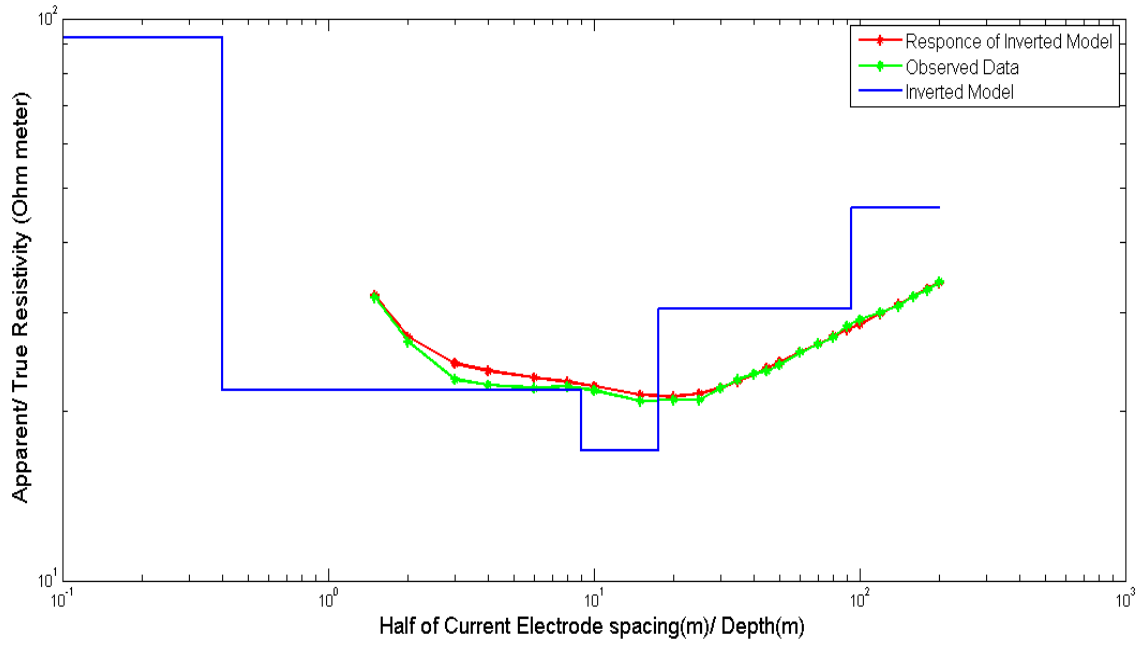

Figure 5.66 Inversion of VES-55.

The comprehensive study presented in this chapter indicated that developed MATLAB-based code for the inversion of 1-D Geoelectrical data using a Genetic Algorithm (GA) effectively inverts synthetic data. Further, its validity was checked on published VES data taken from the study of Mandal et al. (2021). It was found that the inverted model matches well with the model of the research paper. Further, it is used for the inversion of 55 acquired VES data in the Singrauli coalfield region to obtain layer parameters (true resistivity, thickness).

Analysis of the results of study area VES data reveals that out of 55 VES data sets, 2 data sets fitted to the three-layer model, 34 VES data sets to the four-layer model, 18 data sets to the five-layer and 1 data set to the six-layer model. The sub-surface layer parameters (true resistivity, thickness) obtained by the Genetic Algorithm (GA) technique for most of the VES are within the acceptable value of relative root mean square error (RRMSE). This finding is further confirmed by close agreement between the observed and calculated apparent resistivity curves at most VES stations. Thus, the developed program is efficient and reliable for determining sub-surface layer parameters from the VES data. In addition, the results obtained from the inversion of VES data correlated satisfactorily with borehole lithology, present at three nearby VES stations, described in detail in Chapter 6. The obtained layers parameters(true resistivity, thickness) is further used to calculate secondary parameter known as Dar-Zarrouk parameters for the characterization of the aquifers, described in detail in Chapter 6.