

**Genetic Geotomography:
An Application of Genetic
Algorithms to Electromagnetic Tomography**

JOSE RICARDO ARCE

GENETIC GEOTOMOGRAPHY: AN APPLICATION OF GENETIC
ALGORITHMS TO ELECTROMAGNETIC TOMOGRAPHY

by

José Ricardo Arce

A Thesis Submitted to the Faculty of the

DEPARTMENT OF MINING AND GEOLOGICAL ENGINEERING

In Partial Fulfillment of the Requirements
For the Degree of

MASTER OF SCIENCES

WITH A MAJOR IN GEOLOGICAL AND GEOPHYSICAL ENGINEERING

In the Graduate College

THE UNIVERSITY OF ARIZONA

1993

STATEMENT BY AUTHOR

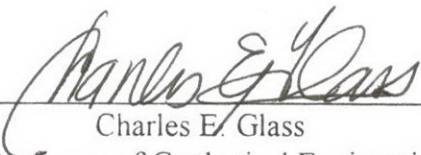
This thesis has been submitted in partial fulfillment of requirements for an advanced degree at the University of Arizona and is deposited in the University Library to be made available to borrowers under rules of the Library.

Brief quotations from this thesis are allowable without special permission, provided that accurate acknowledgment of source is made. Requests for permission for extended quotation from or reproduction of this manuscript in whole or in part may be granted by the head of the major department or the Dean of the Graduate College when in his or her judgment the proposed use of the material is in the interests of scholarship. In all other instances, however, permission must be obtained from the author.

SIGNED: 

APPROVAL BY THESIS DIRECTOR

This thesis has been approved on the date shown below:



Charles E. Glass
Professor of Geological Engineering

10-5-93

Date

ACKNOWLEDGMENTS

I wish to start by thanking Dr. Charles E. Glass, my thesis advisor and friend, for all the time he dedicated to helping me with this research. Every time I entered his office, Dr. Glass helped me in every way he could; he is the best advisor I have had so far, and it would only be fair to say that this research is our work. I also wish to thank Dr. Mary M. Poulton and Dr. John M. Kemeny, my other thesis committee members, for their support and very helpful ideas. I want to thank Dr. Ben K. Sternberg for all the support I got from him throughout my time spent in his department.

I would like to thank my friends from the L.A.S.I. Lab, in particular Mr. Clark Fitzsimmons, for all their help and thoughtful insight on my research. I would also like to thank my girlfriend, Lesley A. Perg, and all my friends from outside the department: Joseph Zerboni, Harlan Loomas, Manus Sweeney, Heidi Adriance, Marla Goldstein, Ginger Rothwell, Eric Wagner, Mark McNeil and Richard Cornish, among others, for all the help I got from them as well as the good times we spent together. And not to be excluded in the least, I must thank my feline companion and friend Amadeus.

Finally, I want to finish up by thanking my family back in Perú. They have been very supportive throughout my college years, and to whom I cannot express how thankful I am. They are the best family I could ever have.

DEDICATION

I wish to dedicate this thesis to my parents, Dr. José E. Arce Helberg and Mrs. Mirella Alleva de Arce for their continuous support throughout my entire life. I could have never achieved this without their help and love.

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS.....	7
LIST OF TABLES.....	7
ABSTRACT.....	8
1. INTRODUCTION.....	9
1.1. General Overview of Inversion Methods.....	9
1.1.1. Singular Value Decomposition (SVD).....	9
1.1.2. Genetic Algorithms.....	10
1.2. Scope of Study.....	12
2. WAVE DIFFUSION GEOTOMOGRAPHY.....	13
2.1. Background.....	13
2.2. The Wave Diffusion Approach.....	14
2.3. Inverting Equation 2-3.....	16
2.3.1. Singular Value Decomposition Approach.....	18
3. GENETIC ALGORITHMS AND GENETIC GEOTOMOGRAPHY.....	20
3.1. Genetic Algorithms.....	20
3.1.1. Preceding Research on Genetic Algorithms in Geophysics.....	22
3.2. Genetic Geotomography.....	22
3.2.1. Random Initial Guesses.....	23
3.2.2. Main Genetic Loop.....	25
3.2.2.1. The Fitness Function.....	25
3.2.2.2. Conductivity and Permittivity Scaling.....	28
3.2.2.2.1. Z-Scaling.....	28
3.2.2.2.2. Range Scaling.....	29
3.2.2.2.3. Fitness Scaling.....	30
3.2.2.3. String Selection.....	31
3.2.2.4. String Encoding and Decoding.....	31
3.2.2.5. Mate Crossover.....	32
3.2.2.6. Mutations.....	33
4. RESULTS AND CONCLUSIONS.....	35
4.1. Results.....	35
4.1.1. Inversions of Various Grids with One Anomalous Patch.....	35
4.1.1.1. The Nine Patch Grid.....	36
4.1.1.2. The Sixteen Patch Grid.....	39
4.1.1.3. The Twenty Five Patch Grid.....	39
4.1.2. The Random Number Generator.....	44

TABLE OF CONTENTS - *Continued*

4.1.3. The Multiple Anomalous Patches Case.....	46
4.2. Conclusions	47
5. FOLLOW-UP RESEARCH.....	49
APPENDIX A: PROGRAM FLOWCHART AND LISTING	50
A.1. Initial Values.....	51
A.2. Program Listing	54
APPENDIX B: OPTIONAL PROGRAM SUBROUTINES AND FUNCTIONS.....	79
REFERENCES CITED	82

LIST OF ILLUSTRATIONS

FIGURE 1-1. Inter-Borehole Tomographic Geometry	11
FIGURE 1-1a. Geotomography Ray Path Geometry	11
FIGURE 1-1b. Inter-Borehole Anomaly Geometry	11
FIGURE 2-1. Patch Geometry Describing the Wave Diffusion Geotomography Approach.....	15
FIGURE 3-1. Example of a Simple Nine Patch Mating Pair, Each Having a Respectable Fitness.....	24
FIGURE 3-2. Plots of Some Initial Models Generated with Subroutine RANGEN for a 3 by 3 Patch Grid.....	26
FIGURE 3-3. Roulette Wheel Selection Method as Proposed by Goldberg (1989).....	34
FIGURE 4-1. Geometry of the Nine Patch Cases.....	37
FIGURE 4-2. Improvement Charts for Nine Patch Inversions	38
FIGURE 4-3. Geometry of the 16 Patch Inversions.....	40
FIGURE 4-4. Improvement Charts for the Sixteen Patch Case.....	41
FIGURE 4-5. Geometry of the 25 Patch Grids.....	42
FIGURE 4-6. Improvement Charts for the 25 Patch Cases.....	43
FIGURE 4-7. Improvement Charts for the Fifth Grid on the 25 Patch Case Using Subroutine RAN1 as the Random Number Generator.....	45
FIGURE A-1. Flowchart Diagram of Program GEOTOM	53

LIST OF TABLES

TABLE 4-1. Converged Values for the Nine Patch Inversions	36
TABLE 4-2. Convergence Values for the Sixteen Patch Cases.....	39
TABLE 4-3. Converged Values for the 25 Patch Cases.....	44
TABLE 4-4. Errors on the Inversion of the Fifth Grid on the 25 Patch Case Using Subroutine RAN1 as the Random Number Generator with Various Initial Seeds.....	46

ABSTRACT

The need for economical, high precision methods to produce time-lapse images of pollution movement in a rock or soil mass has led rather naturally to investigation of cross-borehole electromagnetic (EM) remote sensing approaches. Success of medical tomography (CAT scans) using X-rays, has encouraged investigators to apply this same approach to cross-borehole EM measurements in the earth. The success of applying these techniques to measurements in the earth relies on careful calculation of the EM wave field in the receiver borehole and skill in inverting the equations describing the scattering of EM waves.

This thesis presents a preliminary test of the efficacy of genetic algorithms to solve the general least-square inversion of wave diffusion geotomography.

Results of this work provide a "proof of principle" that genetic algorithms can be used to invert simple geotomography problems using synthetic data, and sets the stage for more thorough investigations later.

An important secondary outcome of this work is the discovery that for this type of inversion to converge, it is necessary to have only four receivers and one transmitter for each row of patches in the interrogation region, an outcome that may aid future field survey design.

CHAPTER 1

INTRODUCTION

Inversion methods have been widely used in geophysics. These techniques are useful in the field of electrical soundings, and necessary for geotomography.

Sheriff (1989) defines tomography as the reconstruction of an object from its set of projections. In geology and geophysics, tomography, also known as geotomography, usually involves the reconstruction of a geological cross section from cross-borehole seismic or electrical data (see Figure 1).

1.1 General Overview of Inversion Methods

The main purpose of geophysical interpretation is to use measurements of some anomalous field to determine the location, shape, dimensions and physical properties of subsurface bodies. This process of interpretation is often called an *inverse* problem, because it is necessary to determine what distribution of physical properties in the earth is responsible for the measured anomalous field.

Given a distribution of physical properties (electrical conductivity within a set of subsurface patches, for example), it is relatively straightforward to compute the resulting anomalous field. A general equation to do this is given below,

$$\underline{\underline{A}}\underline{x} = \underline{y}, \quad (1-1)$$

where \underline{x} is a vector containing the physical property of interest for each patch, \underline{y} is a vector of anomalous field measurements and $\underline{\underline{A}}$ is a transformation matrix which

transforms the physical property into an anomalous field. Inversion involves solving equation 1-1 for now unknown physical properties, \underline{x} , by finding an inverse for matrix \underline{A} .

1.1.1. Singular Value Decomposition (SVD)

Conventional inversion approaches include techniques such as Gaussian elimination, projections (ART, SIRT, etc.) and singular value decomposition. Glass (1990) used singular value decomposition successfully for inverting the volume-current wave diffusion geotomography of Howard et. al., (1983) and Howard and Kretzschmar (1986). This thesis presents an application of genetic algorithms (GA's) to invert the wave diffusion geotomography data.

1.1.2. Genetic Algorithms

Over the last few years, there has been an increasing number of science fields that have successfully used genetic algorithms as a computer search method. Goldberg (1989) defined genetic algorithms as search methods based on the mechanics of natural selection and natural genetics. They have proved useful in fields that range from medicine to political science.

The popularity of GA's is mainly due to their simple mathematical construction as well as their robustness, which is a great advantage over conventional methods such as singular value decompositions. They are also relatively simple to implement in a computer program, and their robustness allows them to search for parameters in large, noisy and complex spaces in a short time. Thus genetic algorithms are also cost-efficient based on computer time.

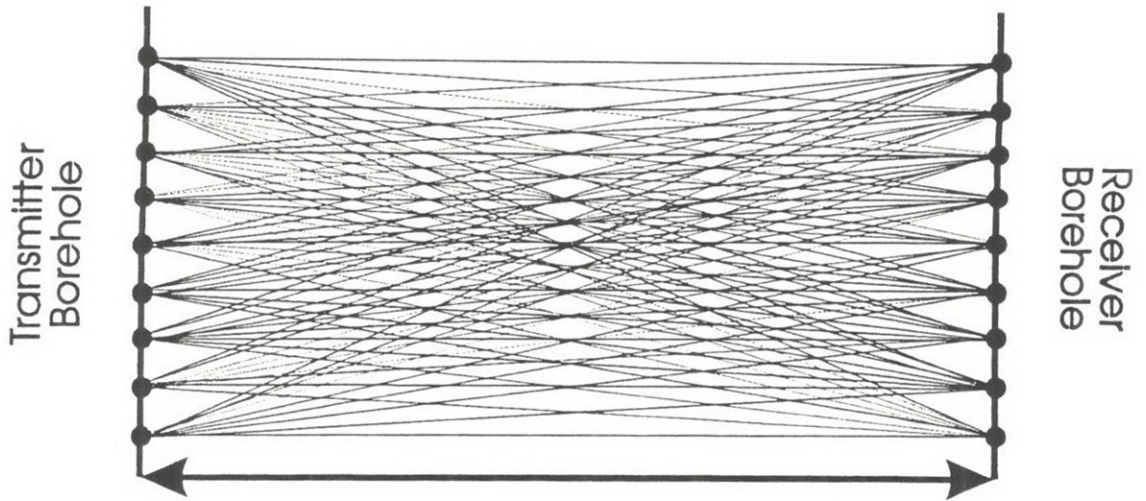


Figure 1-1a. Geotomography Ray Path Geometry.

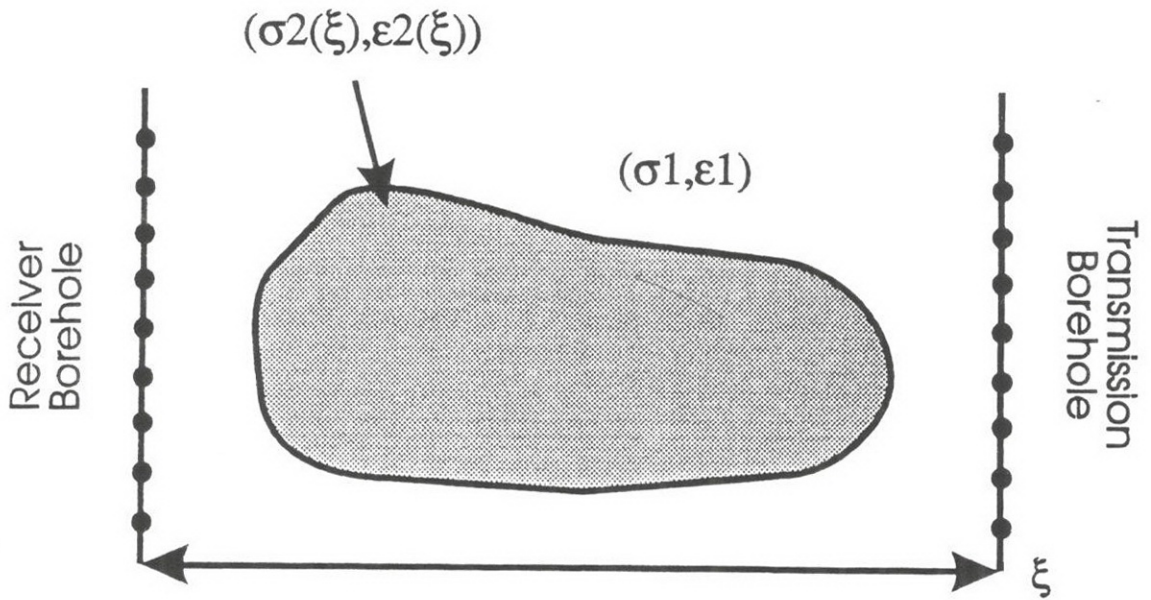


Figure 1-1b. Inter-Borehole Anomaly Geometry. (σ_n and ϵ_n are the conductivity and permittivity at medium n , respectively).

Figure 1-1. Inter-Borehole Tomographic Geometry

1.2. Scope of Study

The material presented in the following chapters discuss the construction, implementation and results of using a genetic algorithm for wave diffusion geotomographic reconstructions.

The genetic algorithm implemented here uses techniques similar to those presented by Goldberg (1989). Only two of the software routines from Goldberg (1989), however, were used in this research. These two software routines include the selection routine and the fitness scaling routine. These routines were translated from Pascal into Fortran 77, corresponding with the geotomography software.

CHAPTER 2
WAVE DIFFUSION GEOTOMOGRAPHY

2.1 Background

The need for economical, high precision methods to produce time-lapse images of pollution movement in a rock or soil mass has led to investigation of cross-borehole electromagnetic (EM) remote sensing approaches. Success of medical tomography (CAT scans) using X-rays, has encouraged investigators to apply this same approach to cross-borehole EM measurements in the earth. The following integral equation describes the general scattering problem in a simple lossy earth (see Howard et. al., 1983 and Howard and Kretzchmar, 1986 for derivation details):

$$\phi_2(\underline{x}) = \phi_1(\underline{x}) - \varpi^2 \mu_0 \int_A g(\underline{x}; \underline{x}') O(\underline{x}') \phi_s(\underline{x}') d\underline{x}' \quad (2-1)$$

In equation 2-1, $g(\underline{x}; \underline{x}')$ is the Green's function for the outward cylindrical wave solution to:

$$(\nabla^2 + k_1^2)g = -\delta^2(\underline{x} - \underline{x}'),$$

where k_1^2 is the background complex wavenumber, δ is the Dirac delta function and g is defined as

$$g(\underline{x}, \underline{x}') = -\frac{i}{4} H_0^{(1)}(k_1 R) \quad (2-2),$$

where $R = \sqrt{(x - x')^2 + (y - y')^2}$ and $H_0^{(1)}$ is the zero order Hankel function of the first kind. Also, in equation 2-1, $\phi_s(\underline{x}')$ is the scattered electric field from the anomaly surface

at location \underline{x}' , and the objective function $O(\underline{x}')$ provides in our case the electrical contrast $(k_x^2 - k_1^2)$ at the scattering location.

Equation 2-1 provides the general departure point for the different approaches to geotomography presented in the literature. Depending, for example, on the choice for $O(\underline{x}')$, one could apply the equation to the scattering of seismic or EM waves. Hence, it is the approximation one chooses for the integral, and the terms therein, that determines the solution strategy. If one assumes an EM object function $O(\underline{x}')$, and lets $\phi_2(\underline{x}') \approx \phi_1(\underline{x}')$, or $\partial\phi_2(\underline{x}')/\partial\phi_1(\underline{x}')$ be small (an aerosol anomaly), the problem reduces to that of diffraction geotomography (see several publications by Devaney; A.J. Devaney, 1982, 1984 and 1985 are three). If one further assumes that the integral of equation 2-1 can be approximated using a line integral (along a ray path, for example), the approach reduces to the ray-optics formulation developed by Lager and Lytle (see, for example, Lager and Lytle, 1977; Dines and Lytle, 1979; Ramirez, 1986; and others).

2.2. The Wave Diffusion Approach

To avoid the restrictive a priori assumptions on the anomaly characteristics or the interrogation wavefield, Howard et. al., 1983 proposed to solve equation 2-1 using a wave diffusion approach, which involves solving the integral analytically. If one selects circular patches to replace square patches (see Figure 2-1) such that the diameter of the new patch, a , is equal to $\Delta/\sqrt{\pi}$, where Δ is the patch width, equation 2-1 becomes

$$\phi_2(\underline{x}) = \phi_1(\underline{x}) + \sum_{n=1}^N (k_{2z_n}^2 - k_1^2) \phi_s(\underline{x}_n) \cdot \frac{i\pi a^2}{2} \frac{J_1(k_1 a)}{k_1 a} H_0^{(1)}(k_1 R_n), \quad (2-3)$$

where J_1 is a Bessel function of the first kind.

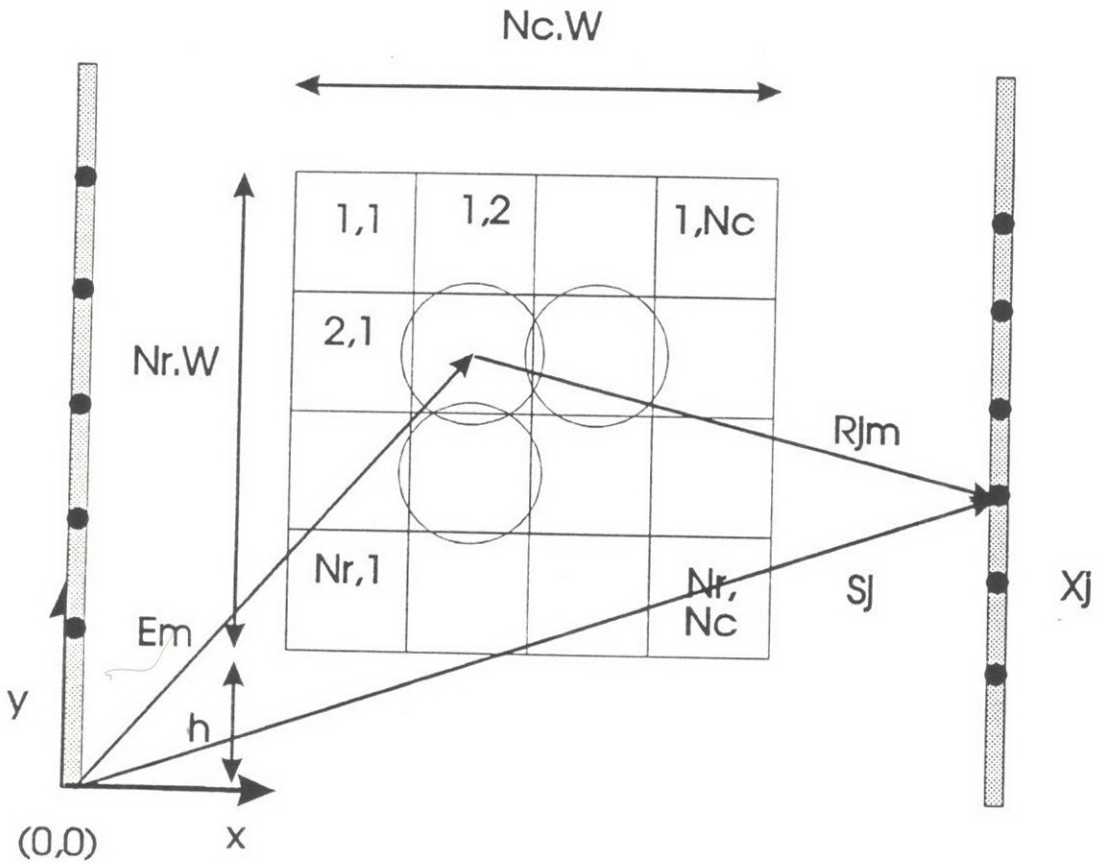


Figure 2-1. Patch Geometry Describing the Wave Diffusion Geotomography Approach. (where $E_m = \xi_m$, $W = \Delta$ (width of square patch), N_r = number of rows, and N_c = number of columns.)

The goal of wave diffusion geotomography is to solve equation 2-3 for the unknown wave number $k_{2\xi_n}$, then from $k_{2\xi_n}$ compute the electrical properties σ and ϵ . There are two complicating problems. First, $k_{2\xi_n}$ is complex so the wavefield amplitude and phase must be measured at the receiver borehole. Second, the scattered field at patch ξ_n due to scattering by patch $\xi_i \forall i$, ($\phi_s(\xi_n)$), is also unknown.

2.3. Inverting Equation 2-3

The inverse solution to equation 2-3 can be stated in the following way, "*find the distribution of anomalous eddy currents in the interborehole region that produces the measured field in the borehole.*"

Equation 2-3 can be simplified to yield

$$y_j^* = \underline{\underline{A}}_{jm} x_m, \quad (2-4)$$

where

$$y_j^* = \frac{-2i}{\pi a^2} \left(\frac{k_1 a}{J_1(k_1 a)} \right) (\phi_2(\underline{x}_j) - \phi_1(\underline{x}_j)),$$

$$\underline{\underline{A}}_{jm} = H_0^{(1)}(k_1 R_{jm}); \quad R_{jm} = |\xi_m - \underline{x}_j|, \text{ and}$$

$$x_m = (k_{2\xi_m}^2 - k_1^2) \phi_s(\xi_m).$$

Glass (1986) demonstrated that the unknown electrical properties can be calculated by finding the pseudo inverse of $\underline{\underline{A}}_{jm}$ in equation 2-4 using an iterative method (Pan and Reif, 1985). Since then (Glass, 1990) the inversion has been accomplished using a singular value decomposition technique. The solution for the electrical properties of the interborehole rock mass proceeds as follows. Since $x_m = (k_{2\xi_m}^2 - k_1^2) \phi_s(\xi_m)$, then from the relationship for the wavenumber k , one obtains

$$\omega^2 \mu_0 \left(\epsilon_{2\xi_m} + \frac{i\sigma_{2\xi_m}}{\omega} \right) = \omega^2 \mu_0 \left(\epsilon_1 + \frac{i\sigma_1}{\omega} \right) + \frac{x_m}{\phi_s(\xi_m)}. \quad (2-5)$$

Equating real and imaginary parts of equation 2-5 yields

$$\varepsilon_{2\xi_m} = \varepsilon_1 + \frac{\varepsilon_0}{k_0^2} \operatorname{Re}\left(\frac{x_m}{\phi_S(\xi_m)}\right), \text{ and} \quad (2-6a)$$

$$\sigma_{2\xi_m} = \sigma_1 + (\omega\mu_0)^{-1} \operatorname{Im}\left(\frac{x_m}{\phi_S(\xi_m)}\right). \quad (2-6b)$$

In equation 2-6a and 2-6b, $k_0 = \sqrt{\mu_0\varepsilon_0}$ is the free space wave number, and $\phi_S(\xi_m)$ is given by

$$\phi_S(\xi_m) = \sum_{n=1}^N \underline{G}_{mn} x_n. \quad (2-7)$$

The \underline{G} matrix in equation 2-7 is a patch interaction matrix, which describes the influence of patches on one another; it is given by

$$\underline{G} = \frac{i\pi\alpha^2 J_1(k_1\alpha)}{2\Delta^2 k_1\alpha} * \underline{L},$$

where $\underline{L} = \frac{4i}{k_1^2} \left[1 - \frac{i\pi k_1\alpha}{2} H_1^{(1)}(k_1\alpha)\right]$, along the diagonal (single interaction), and

$$\underline{L} = \frac{2\pi\alpha^2 J_1(k_1\alpha)}{k_1\alpha} H_0^{(1)}(k_1 R_{mn}), \text{ for off diagonal elements (a nearby patch}$$

interaction), and

$$R_{mn} = |\xi_m - \xi_n| \text{ with } n \text{ cycling through all patches.}$$

Equation 2-7, then, can be written as

$$\underline{\phi}_S(\xi) - \underline{\phi}_S(\xi) \underline{G} (k_{2\xi}^2 - k_1^2) = 0.$$

Letting $(k_{2\xi}^2 - k_1^2)$ be represented by VV , we have

$$\begin{aligned} \underline{\phi}_S(\xi) [\underline{I} - \underline{VV} \underline{G}] &= 0, \text{ or} \\ \underline{\phi}_S(\xi) &= [\underline{I} - \underline{VV} \underline{G}]^{-1}. \end{aligned} \quad (2-8)$$

This completes the inverse solution for the unknown electrical properties $\varepsilon_{2\zeta_m}$ and $\sigma_{2\zeta_m}$. An idea of the reconstruction error can be obtained by substituting the completed values for ε and σ into equation 2-4 and comparing the computed borehole measurement \underline{y}^* with the actual measurement \underline{y} .

The GA approach to reconstructing the interborehole electrical properties is attractive, because the electrical properties can be easily constrained, it is iterative, thus conserves memory, and the matrix inversion (at least the inversion of the ill conditioned matrix A) is eliminated.

2.3.1. Singular Value Decomposition Approach

Equation 2-4 may be inverted with a singular value decomposition. The independent variables to be found are the wave numbers k_m in vector \underline{x}_m ; to compute this vector, we start by transforming equation 2-4 into its following equivalent:

$$\underline{x}_m = \underline{y}_j [\underline{A}_{jm}]^{-1} \quad (2-9)$$

The A matrix and the y vector contain values that are known or can be computed. Therefore, all we need is to invert matrix A. First, matrix A needs to be decomposed into the following matrices:

$$\underline{A} = \underline{U}_p \underline{\Lambda}_p \underline{V}_p^T,$$

where \underline{U}_p , $\underline{\Lambda}_p$, and \underline{V}_p^T can be computed with the following equations:

$$\underline{A} \underline{A}^T - \underline{\Lambda}_p^2 \underline{I} = 0, \quad (2-10a)$$

$$\underline{A} \underline{A}^T - \underline{\Lambda} \underline{I} \underline{U}_p = 0, \text{ and} \quad (2-10b)$$

$$\underline{A}^T \underline{U} - \underline{\Lambda} \underline{V}_p = 0. \quad (2-10c)$$

In equations 2-10, \underline{I} is the identity matrix and $\underline{\Lambda}$ is a diagonal matrix where the diagonal terms are called singular values; the off-diagonal elements are zero, while the eigenvalues may be either zero or non-zero values.

The generalized inverse of matrix A now be computed as follows:

$$\underline{A}_g^{-1} = \underline{V}_p \underline{\Lambda}_p^{-1} \underline{U}_p^T \quad (2-11)$$

This procedure has been widely used in various scientific fields for many years now. Before new tools such as genetic algorithms were explored and tested, singular value decomposition was the main inversion technique used in science.

CHAPTER 3

GENETIC ALGORITHMS AND GENETIC TOMOGRAPHY

3.1. Genetic Algorithms

Genetic algorithms describe a broad class of optimization algorithms loosely based on the principle of natural selection. In all optimization problems, including geotomography, one is presented with a single function f (call it a fitness function), which depends on one, or a number of, independent variables. The objective is to search for values for those independent variables that result in an extremum for the fitness function. Although there are several standard approaches for satisfying this search, the most common is to randomly choose a point on f (select specific, though random, values for the independent variables), then move up **or** down gradient until the gradient in the neighborhood of the test point is zero (an extremum).

Two new approaches to optimization, simulated annealing and genetic algorithms, differ from standard approaches in two fundamental ways. First, a large number of points are chosen randomly so that the fitness function is thoroughly sampled. Second, new points are sought, in a massively parallel search both up **and** down gradient, that incrementally optimize f . Of these two new approaches, genetic algorithms are more flexible for geotomography, because their use does not require a rigid, discrete system configuration (values for rock mass electrical properties, for example, are not constrained to discrete values but may vary continuously between logical limits).

In a nutshell, genetic algorithms proceed in the following manner:

1. Randomly select a large number of possible values for the independent variables. Each unique combination, or set, of independent variables constitutes one population. So there may be several hundred, or several thousand populations needed to fully sample the richness of the fitness function.
2. Encode each population into a string of binary units. The genetic analogue to these strings are *chromosomes*, and to the independent variables constituting the strings is *genes*, although the complexity of chromosomes and genes relative to our simple strings significantly trivializes the analogy. In addition, although we have encoded the strings as binary, this is not a rigid requirement.
3. Each string has associated with it a *fitness*, obtained by substituting the randomly selected values for the independent variables into the fitness function. In anticipation of discussing some problems related to the geotomography fitness function, to be presented a bit later, it is fair to state here that it is advantageous if the fitness function is sensitive to meaningful perturbations of the independent variables.
4. Natural selection preserves the traits of those individuals who mate the most. We ensure that, on average, we move efficiently toward an extremum by allowing only our most fit strings to mate the most. This is usually accomplished using a weighted roulette wheel selection. The weighting favors the most fit strings, but occasionally, as luck would have it, a low fitness string enters the mating pool. In this way, genetic algorithms permit movement throughout the surface of the fitness function and avoid local extrema.
5. Mating proceeds through crossover and mutation of the independent variables in each mating pair, producing two offspring having traits different from the parents. At this stage, the offspring are returned to step 3 and the process repeated until the desired fitness is achieved, at which time the optimization problem is considered solved.

3.1.1. Preceding Research on Genetic Algorithms in Geophysics

There have been a few successful applications of genetic algorithms in geophysical exploration. Only a handful of papers were found, though. These papers are all applications of GA's in seismic inversions. Sambridge and Drijkoningen (1991), as well as three papers by Sen and Stoffa (1991), have successfully inverted seismograms, and thus obtained accurate velocity models. In these papers, the GA converged more accurately than other methods used such as Simulated Annealing and the Monte Carlo search, and Sen and Stoffa (1992) improved the GA performance further by "*stretching*" the fitness function using an annealing temperature. Berg (1990 and 1991) presents successful inversions of multiparameter data as well. Kennett and Sambridge (1992) used GA's to invert earthquake epicenters successfully. Finally, Wilson and Vasudevan (1991) presented a paper on the use of GA's as an optimizing method to compute residual statics in seismic data processing.

All of these authors found genetic algorithms to be a faster and more accurate way of doing a variety of seismic inversions, but all examples dealt with simple geometries, and they did not show how exactly their programs differ from standard genetic algorithms such as the one proposed by Goldberg (1989). No previous research was found on genetic geotomography.

3.2. Genetic Tomography

In our application of genetic algorithms, we wish to solve equation 2-4 for the unknown rock mass electrical properties σ and ϵ . This goal turned out to present several non-trivial challenges to GA's. First, the complex nature of the measurements challenged our ability to form binary encoding. Second, solution of equation 2-4, requires a matrix inversion (the $[I - \underline{VV} \underline{G}]$ matrix of equation 2-8). If we need to use SVD to invert a

matrix anyway (even if it is a square matrix), why use GA's?. Third, in the wave diffusion algorithm, the computed electrical field at each receiver vector y_j^* in equation 2-4, is a function of the scattered field from all of the interborehole patches (see Figure 2-1, and equations 2-4). Hence, a meaningful perturbation in the electrical properties of a single patch does not necessarily produce an unequivocal perturbation in the fitness function (here chosen as $\| y_j - y_j^* \|$).

The following presents our approach.

3.2.1. Random Initial Guesses

Genetic algorithms require an initial population of genetic strings. Each string comprises NPATCH pairs of electrical properties (where NPATCH is the number of patches constituting the interborehole rock mass). Each pair of electrical properties comprises a random selection of conductivity and permittivity between some predetermined maximum and minimum values (reasonable values are $1 \leq \epsilon \leq 80$, and $0 \leq \sigma$). These values are generated in subroutine RANGEN, which generates an initial population of 50 (variable NPOP) strings. RANGEN starts by selecting the patch in the input grid having the most significant influence on the fitness function. This is achieved by sequentially setting each path in a test string to one of the following pairs at a time:

$$(\sigma_{\max}, \epsilon_{\max}),$$

$$(\sigma_{\min}, \epsilon_{\min}),$$

$$(\sigma_{\max}, \epsilon_{\min}),$$

$$(\sigma_{\min}, \epsilon_{\max}),$$

while all the other patches are set to the background σ and ϵ values. The fitness of the test strings is computed for each trial. Function ONEFIT returns this fitness. If subroutine RANGEN obtains a fitness of four times the fitness of the background ($FIT_{\text{background}} =$

1.000) for any of the trials, that patch is assumed to be anomalous. Four times the background fitness was chosen (variable INC=4) empirically by observing the behavior of RANGEN with various grids, and was found to be correct for the case of only one anomalous patch in the grid.

The purpose for this rather unusual procedure relates to the third "excuse" mentioned above. During experimentation with the GA's, we discovered a lack of patch-specific sensitivity in the fitness function and mating process. For example, Figure 3-1 shows a simple pair of genetic strings for nine patches, but considering only ϵ as the independent variable.

Patch 1	Patch 9
[3 3 3 14 3 3 3 3 3]:	Fitness = 41.2
[32 3 3 3 3 3 3 3 3]:	Fitness = 35.7

Figure 3-1. Example of a Simple Nine Patch Mating Pair, Each Having a Respectable Fitness.

Note in Figure 3-1 that both strings have a different anomalous patch, but high fitness. When they mate, as we shall see in a moment, both anomalous patches will mate with background patches, thus diluting the mating pool and lowering the fitness of both. Convergence always occurs, but to background electrical values and low fitnesses. To avoid this, we need to search for the most promising anomalous patch first.

Once the anomalous (most promising) patches have been determined, RANGEN populates these patches with random conductivities and permitivities between the

predetermined ranges, while the non-anomalous patches are left with background conductivity and permittivity values for their entire population.

Using this approach, RANGEN generates populations that are more representative of our problem (Figure 3-2).

3.2.2. Main Genetic Loop

The main genetic loop starts by calculating the fitnesses for the individual genetic strings, followed by the scaling procedures, and concluding with the mating of the strings. Each of these individual processes is explained below. The genetic loop will stop if the inverse of the maximum fitness does not exceed the stopping criterion.

3.2.2.1. The Fitness Function

The fitness function (object function) is given by equation 3-1, which states that the scattered field measured in a receiver borehole can be found as follows:

$$\phi_2(x) - \phi_1(x) = \sum_{n=1}^N (k_{2n}^2 - k_1^2) \phi_2(\xi_n) \cdot \frac{i\pi a^2}{2} \frac{J_1(k_1 a)}{k_1 a} H_0^{(1)}(k_1 R_n) \quad (3-1)$$

where $\phi_2(x)$ is the total horizontal electric field measured in the receiver borehole,

$\phi_1(x)$ is the calculated incident field at the receiver borehole for a wave traveling in the background medium (vector F1 in the program),

$(\phi_2(x) - \phi_1(x))$ is the scattered field at the receiver borehole,

k_{2n} is the anomalous wave number for patch n ,

k_1 is the background wave number,

ξ_n is the coordinate at the center of patch n ,

a is the diameter of a circular patch having the same area as the square patch

(see Figure 2-1),

Figure 3-2a. Plot of Real Components of some Initial Models

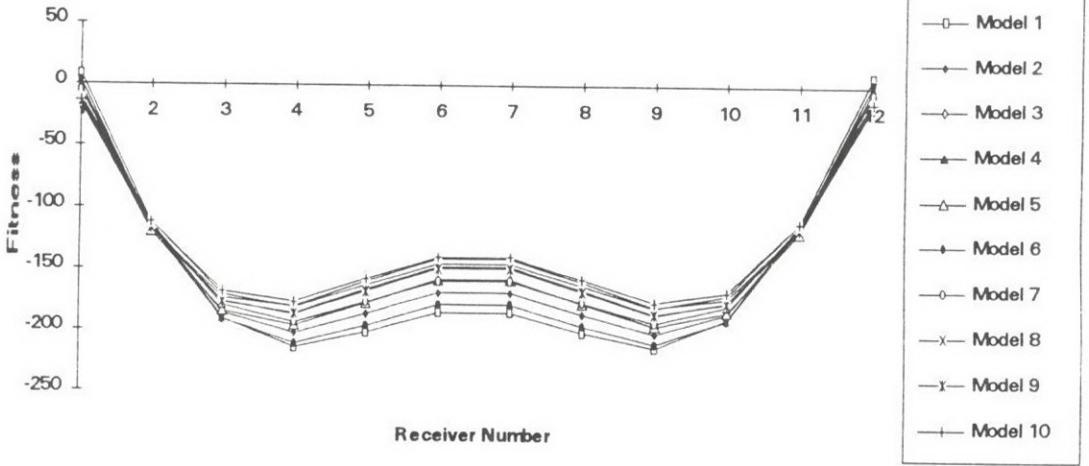


Figure 3-2b. Plot of Some Imaginary Initial Models

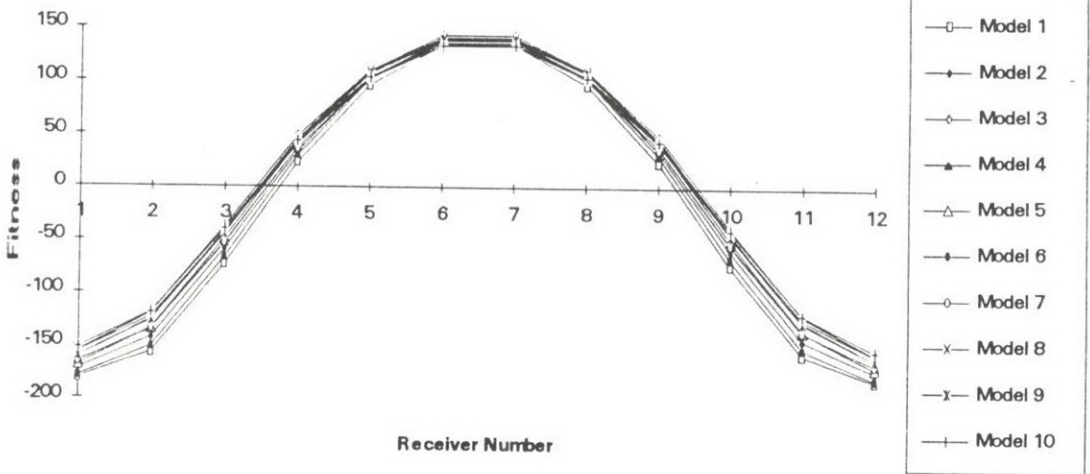


Figure 3-2. Plots of Some Initial Models Generated with Subroutine RANGEN for a 3 by 3 Patch Grid.

J_1 is a Bessel function of the first kind,

$H_0^{(1)}$ is a Hankel function of type 1 and order 0 (matrix A in the program),

R_n is the distance from patch n to each individual receiver, and

$\phi_2(\xi_n)$ is the scattered field due to patch ξ_n .

The geometry for this method is shown in figure 1-1b.

The wave numbers (k_g) can be estimated as follows:

$$k_g^2 = \omega^2 \mu_0 \left(\tau_g + \frac{i\sigma_g}{\omega} \right) \quad (3-2)$$

where g equals one (background wave number) or two (anomalous wave number),

ω is the circular frequency in radians per second,

μ_0 is the magnetic permeability of vacuum ($4\pi \cdot 10^{-7}$ Henries/meter),

τ_g is the electric permittivity in Farads per meter, and

σ_g is the conductivity in mhos per meter.

The wave numbers are computed in subroutine VSETUP and returned in variable VV as follows:

$$VV = (k_{2n}^2 - k_1^2) \quad (3-3)$$

An estimate of the scattered field vector $\phi_2(\xi_n)$ is computed using equation 2-8 and assuming that $[\underline{I} - \underline{VV}\underline{G}]_{ij}, \forall i=j \gg \gg [\underline{I} - \underline{VV}\underline{G}]_{ij}, \forall i \neq j$. Hence $[\underline{I} - \underline{VV}\underline{G}]_{ij}^{-1} \approx 1.0/[\underline{I} - \underline{VV}\underline{G}]_{ij}, \forall i=j$. The efficacy of the estimation was assessed by comparing GA convergence using the the estimation and using SVD inversion. Howard and Kretschmar (1986) proposed the following linear systems of equations for computing the scattered field y_j in the receiver borehole:

$$y_j^* = \sum_{m=1}^M A_{jm} x_m \quad (3-4)$$

where M is the total number of non-overlapping subregions that compose the complete anomalous subsurface region A ,

$$A_{jm} = H_0^{(1)}(k_1 R_{jm}), \quad (3-5)$$

$$R_{jm} = |\xi_m - S_j|, \quad (3-6)$$

where ξ_m is a vector to patch m , and

S_j is a vector to receiver j .

$$y_j^* = \frac{-2i}{\pi \alpha^2} \left(\frac{k_1 \alpha}{J_1(k_1 \alpha)} \right) (\phi_2(S_j) - \phi_1(S_j)) \quad (3-7), \text{ and}$$

$$x_m = (k_{2m}^2 - k_1^2) \phi_s(\xi_m) \quad (3-8).$$

This last set of equations are used in our fitness subroutine as well. The fitness function is then the average error $\| \underline{y} - \underline{y}^* \|$, where \underline{y} is the receiver vector measured in the borehole during a geotomography study. It will be shown in chapter 4 that this algorithm proved to be an adequate fitness function for our genetic algorithm.

3.2.2.2. Conductivity and Permittivity Scaling

3.2.2.2.1. Z-Scaling

Genetic algorithms seem to pay attention to the absolute magnitude and variability of the independent variables. Permittivity ranges from 0 to 80, whereas conductivity may range over several orders of magnitude. To accommodate these disparities in magnitude and variability, we scaled the independent variables' z scores, rather than the variables themselves. The z score is the number of standard deviations above or below the mean for each independent variable.

Z-scaling allows the (σ, ε) pairs to change during mating at proportional rates to their scaling. This method consists of scaling parameters according to their mean and standard deviation. The mean and standard deviation for conductivities and permittivities are computed in the main program, and the actual Z-scaling and descaling is achieved in subroutine UMATEM. To Z-scale the conductivity and permittivity values, the following operations are performed:

$$\sigma_{scaled} = \frac{(\sigma_i - \bar{\sigma})}{\sigma_{std}} \quad (3-13), \text{ and}$$

$$\varepsilon_{scaled} = \frac{(\varepsilon_i - \bar{\varepsilon})}{\varepsilon_{std}} \quad (3-14).$$

Once mating is complete, descaling is performed. This is done as follows:

$$\sigma_{descaled} = \frac{1}{(\sigma_{scaled} * \sigma_{std} + \bar{\sigma})} \quad (3-15), \text{ and}$$

$$\varepsilon_{descaled} = (\varepsilon_{scaled} * \varepsilon_{std} + \bar{\varepsilon}) \quad (3-16).$$

Z-scaling proved useful in the inversions tried.

3.2.2.2.2. Range Scaling

Genetic algorithms perform selection, crossover and mutations on strings. This program achieves crossover and mutation using binary strings. Hence, the conductivities and permittivities are scaled from 2^0 to 2^n , where n is the maximum number of bits on the string. Since σ and ε have different ranges, their strings will have a different number of bits. Therefore, the program finds appropriate scaling parameters for the strings in each generation.

The number of bits necessary to construct the strings (variables MEGABIT1 and MEGABIT2, corresponding to ϵ and σ respectively), are set to 6.

In order to linearly scale all the conductivities and permittivities, the slope (A) and the intercept (B) of the linear transformation need to be found. This is done as follows,

$$A = \frac{2^{(Mbit)}}{P_{max} - P_{min}} \quad (3-9)$$

$$B = A * P_{min} \quad (3-10).$$

For more convenient scaling, resistivities are used in place of conductivities. These parameters are calculated in subroutine SCALE, and are used to scale the selected mates before crossover in subroutine UMATEM by calling function SCALIT, which performs the following operation:

$$\text{ScaledMate} = \text{Mate} * A + B \quad (3-11).$$

Once crossover has been completed, the new strings are de-scaled in function DSCALIT, which operates as follows:

$$\text{DescaledMate} = (\text{ScaledMate} - B) / A \quad (3-12).$$

3.2.2.2.3. Fitness Scaling

Goldberg (1989) proposed a fitness scaling algorithm. The purpose of this algorithm is twofold. First, to ensure that an exceptionally fit string will not overwhelm most of the crossovers on early generations. Second, to ensure that negative fitnesses are scaled to zero. After a few tests, it was concluded that this algorithm (subroutine FITSCALE) was not necessary for this application. The call to this subroutine was commented out.

3.2.2.3. String Selection

In order to select the most fit populations for mating, another random process is followed. This is known as a weighted "Roulette-Wheel" selection, as described by Goldberg (1989). A graphical representation of the method is presented for 6 fitnesses in figure 3-3.

This procedure is done in function SELECT. To start the procedure, a random number between 0 and 1 is selected (function RANDOM), and multiplied by the sum of all the fitnesses (variable SUMFITNESS, which is equivalent to the circumference length of the roulette); this value is stored in variable RAND. This places a "pointer" anywhere on the roulette. Following this, variable PARTSUM is introduced with an initial value of zero, and each of the fitnesses are added to PARTSUM one at a time. When PARTSUM equals or exceeds variable RAND, the last fitness added corresponds to the (σ, ϵ) pair that will be selected for mating.

Two (σ, ϵ) pairs need to be selected for mating, so function SELECT is called twice in subroutine UMATEM. This process is repeated until a new population of strings is obtained.

3.2.2.4. String Encoding and Decoding

In order to perform crossover and mutations on the genetic strings, the conductivities and permittivities are converted to binary strings before, and converted back to their equivalent values in base 10, after these processes.

Subroutine ENCODE returns a binary string equivalent to the value sent to it. This is done with the following formulation,

$$x_j = INT\left(\frac{V - \sum_{J=1}^{J-1} (x_J \cdot 2^{(Mbit-J)})}{2^{(Mbit-J)}}\right) \quad (3-17)$$

where V is the value sent to the subroutine, and

X_I is the bit to be found.

The bits in the string are found in the following order: $x_1, x_2, x_3, \dots, x_{Mbit}$, represented by $2^{Mbit}, 2^{Mbit-1}, \dots, 2^1$, respectively.

After crossover and mutation of the mates, function DECODE transforms the binary string to its equivalent in base 10. The decoded value is then returned in the variable DECODE, which is computed as follows:

$$Decode = \sum_{I=1}^{Mbit} (x_I \cdot 2^{(Mbit-I)}) \quad (3-18).$$

3.2.2.5. Mate Crossover

Once the mate strings have been encoded into their binary equivalents, they are ready to be crossed. A random number between 1 and the length of the string (Mbit, set to 6 as a fixed value for both ϵ and σ) is selected separately for conductivity and permittivity in subroutine UMATEM. These values are stored in variables IICROSS and IRCROSS for conductivity and permittivity, respectively. Subroutine CROSSOVR then mates the strings. The mating process can be illustrated with the following diagram:

Parent Strings	Child Strings
[1 0 1 0 0 0 1 1]	[1 0 1 0 0 1 0 1]
[0 0 0 1 1 1 0 1]	[0 0 0 1 1 0 1 1]

The double lines represent the cross over points (IICROSS or IRCROSS) for a pair of mates. The offspring will contain the same bits as the parents except for the positions to

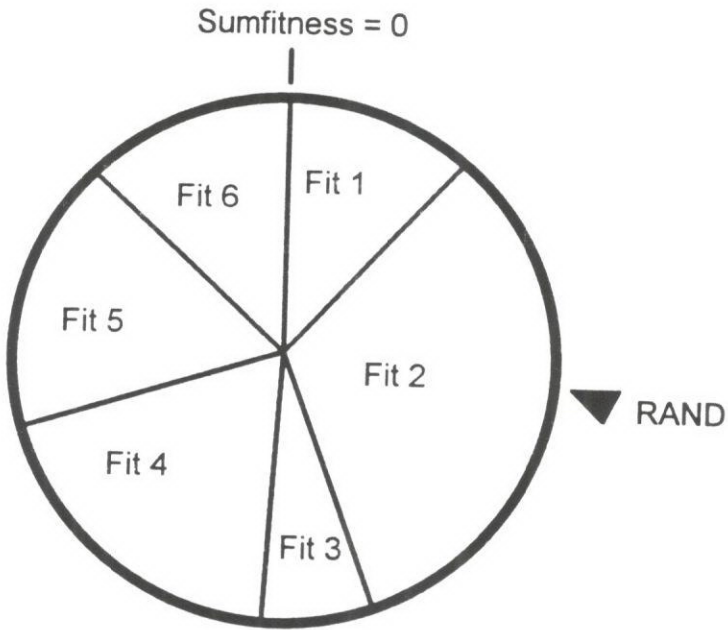


Figure 3-3. Roulette Wheel Selection Method as Proposed by Goldberg (1989).

CHAPTER 4

RESULTS AND CONCLUSIONS

4.1 Results

The genetic algorithm successfully inverted simple synthetic tomographic sections. Convergence was achieved after only a few generations of the genetic algorithm for most inversions. This chapter will show some inversions and their resulting tomographic sections.

4.1.1. Inversions of Various Grids with One Anomalous Patch.

This section will show eleven cases of synthetic data that were inverted using GA's to invert equation 2.4. The synthetic sections used to compute the forward models were chosen randomly, and they all produced satisfactory results. For all of these grids, a frequency of 100 MHz and the following values for conductivities and permittivities were used:

$$\sigma_{\text{background}} = 1 \cdot 10^{-3} \text{ mhos,}$$

$$\sigma_{\text{anomaly}} = 1 \cdot 10^{-2} \text{ mhos,}$$

$$\epsilon_{\text{background}} = 3 \text{ Farads/meter, and}$$

$$\epsilon_{\text{anomaly}} = 10 \text{ Farads/meter.}$$

All of these cases were under determined problems, since the genetic algorithm inverted more unknowns (σ - ϵ pairs for each patch) than available data points (number of receivers). In all of these cases, there were more unknowns than equations.

An important outcome of this work is the discovery that for this type of inversion to converge, it is necessary to have only four receivers and one transmitter for each row of patches in the interrogation region. This will ensure that the receivers will measure the effects of every single patch, and the patch detection algorithm in subroutine RANGEN will work properly. This outcome also has ramifications for field survey design.

4.1.1.1. The Nine Patch Grid

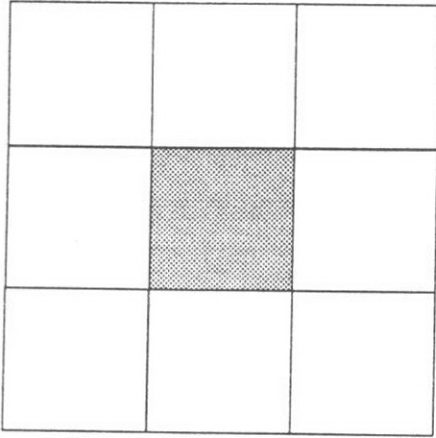
Figure 4-1 presents the geometry of the three inversions for this case. The sections were inverted in the first few generations and the results were almost identical to the desired values. For this problem, 12 receivers and 3 transmitters were used, so there were 24 known data values (amplitude and phase measured in receiver borehole) and 18 unknowns (9 conductivities and 9 permittivities). Table 4-1 shows the converged values.

	FIRST GRID	SECOND GRID	THIRD GRID
BACKGROUND CONDUCTIVITY	1.00E-03	1.00E-03	1.00E-03
ANOMALOUS CONDUCTIVITY	9.65E-03	9.65E-03	9.65E-03
BACKGROUND PERMITTIVITY	3	3	3
ANOMALOUS PERMITTIVITY	10	10	10

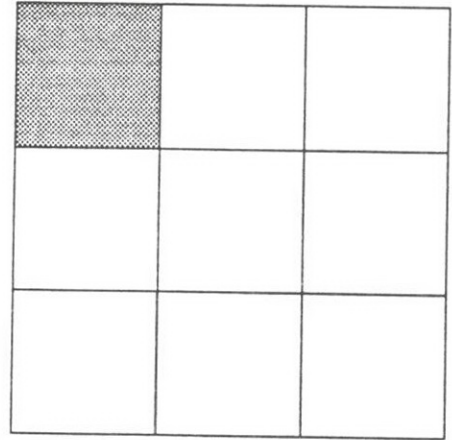
Table 4-1. Converged Values for Nine Patch Inversions.

The convergence process through various generations is shown in figure 4-2. Here, it is obvious that the average and maximum fitnesses have a substantial increase throughout the generations, while the minimum fitness increases, but not significantly for the different grids. This is a typical behavior for a genetic algorithm, as Goldberg (1989) described. The convergence criterion in program GEOTOM uses the maximum fitness, and stops the program when this exceeds a value of 220. This value was determined empirically.

First Grid:



Second Grid:



Third Grid:

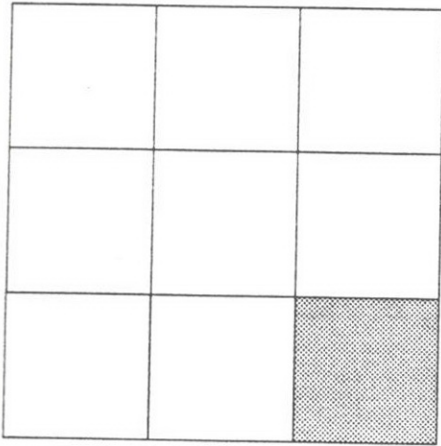


Figure 4-1. Geometry of the Nine Patch Cases. (The shaded patches represent the anomalous patch, whereas the rest are the background patches.)

Figure 4-2a. Maximum Fitness Improvement Plot of 3X3 Grids

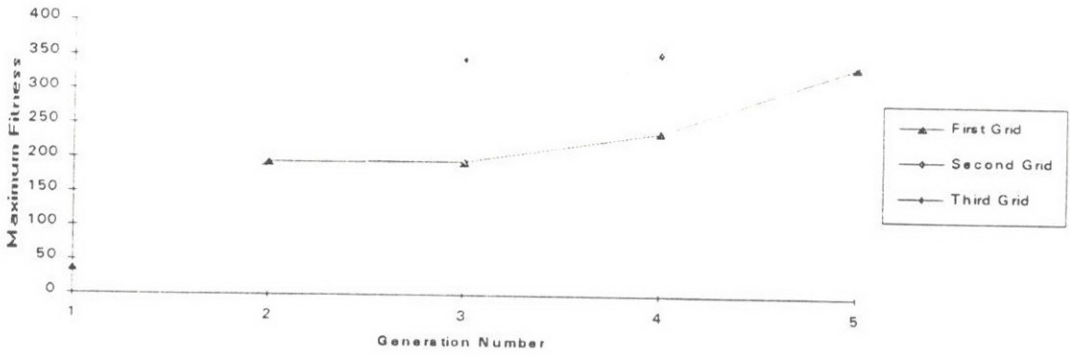


Figure 4-2b. Average Fitness Improvement Plot of 3X3 Grids

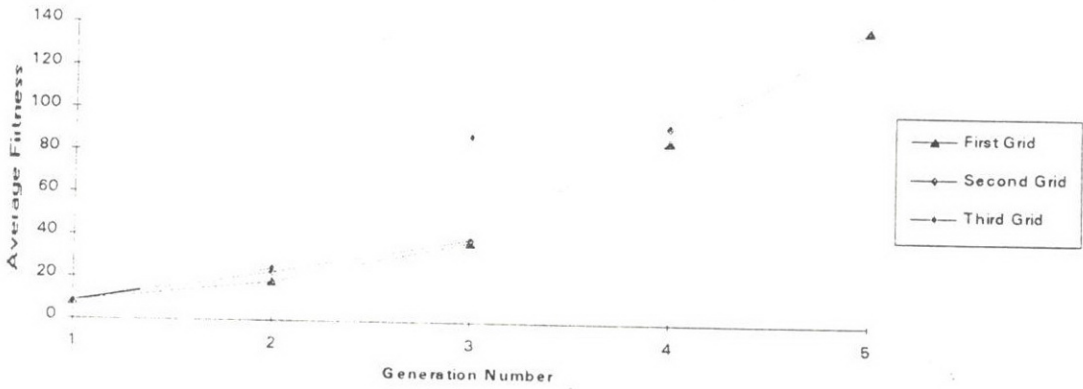


Figure 4-2c. Minimum Fitness Improvement Plot of 3X3 Grids

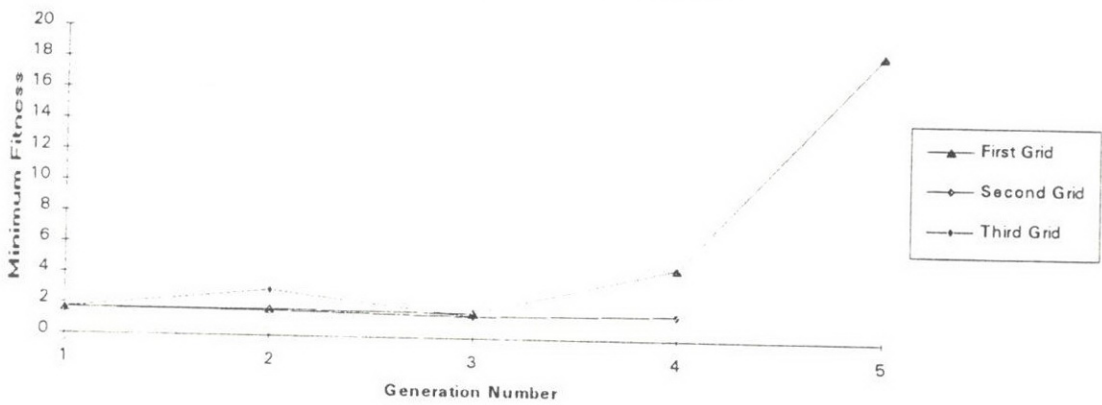


Figure 4-2. Improvement Charts for the Nine Patch Inversions.

4.1.1.2. The Sixteen Patch Grid

The inversions for the 16-patch case converged quickly and accurately. For this case, 16 receivers and 4 transmitters were used.

Three different patch grids were inverted. The geometry and location of these grids is shown in figure 4-3, where the anomalous patches are shown as shaded.

The conductivity and permittivity values to which convergence was achieved are shown in table 4-2. They are very satisfactory.

Improvement of the average and maximum fitnesses through the generations was substantial, while the minimum fitness improvement was acceptable (see Figure 4-4). The 16 patch case caused the same behavior as the nine patch case in the genetic algorithm.

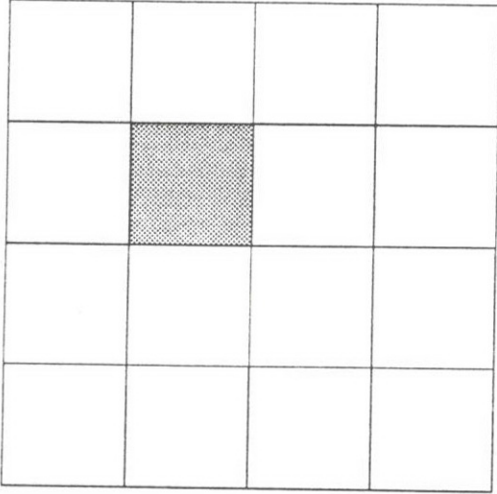
	FIRST GRID	SECOND GRID	THIRD GRID
BACKGROUND CONDUCTIVITIES	1.01E-03	1.01E-03	1.00E-03
ANOMALOUS CONDUCTIVITIES	9.77E-03	9.56E-03	9.65E-03
BACKGROUND PERMITTIVITIES	3	3	3
ANOMALOUS PERMITTIVITIES	10	10.11	9.989

Table 4-2. Convergence Values for the Sixteen Patch Cases.

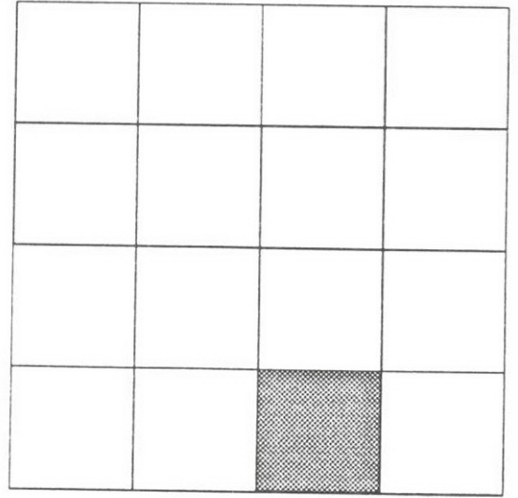
4.1.1.3. The Twenty Five Patch Case

Running inversions with program GEOTOM for a 25 patch grid proved no more difficult than the previous cases. The patch detection algorithm in subroutine RANGEN worked perfectly for all the cases. On the fifth grid, the program started with good average and maximum fitnesses, but never converged. For this case, the program was able to invert the grid when the initial seed for the random number generator (function RANDOM) was changed from 100003 to 100013 (see section 4.1.2.). For all other seeds the convergence was rapid.

First Grid:



Second Grid:



Third Grid:

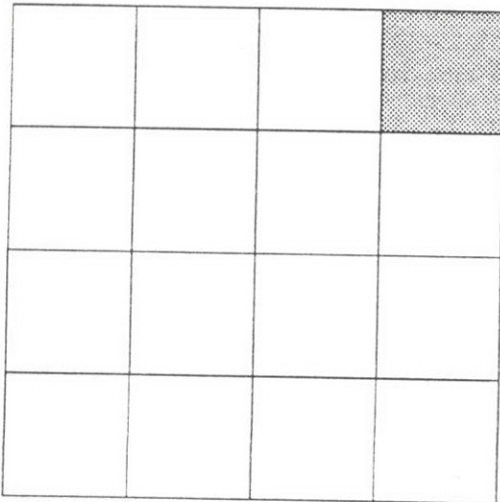


Figure 4-3. Geometry of the 16 Patch Inversions.

Figure 4-4a. Maximum Fitness Improvement Plot of 4X4 Grids

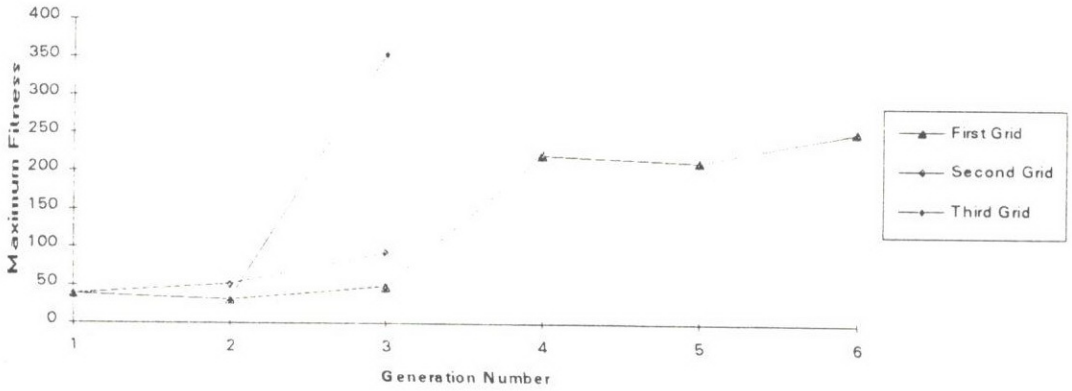


Figure 4-4b. Average Fitness Improvement Plot of 4X4 Grids

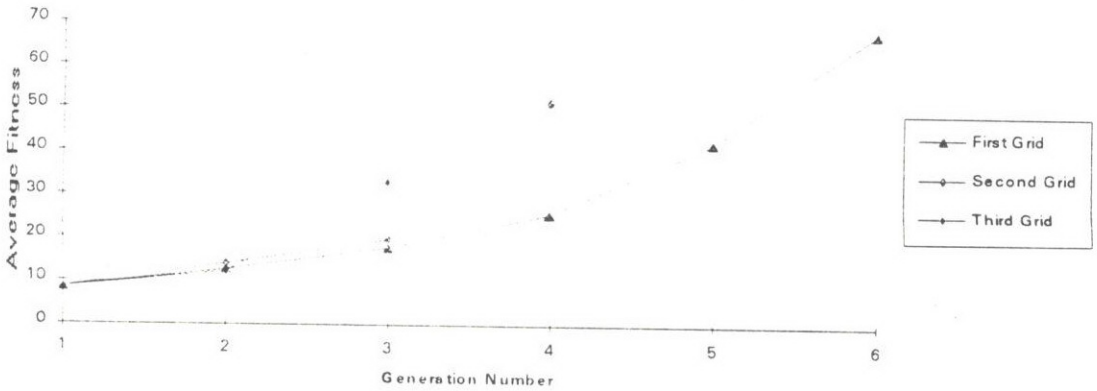


Figure 4-4c. Minimum Fitness Improvement Plot of 4X4 Grids

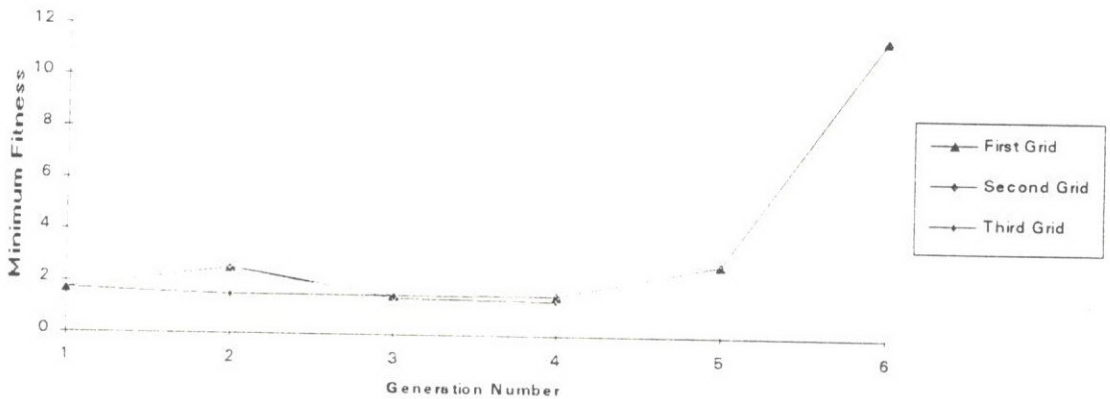
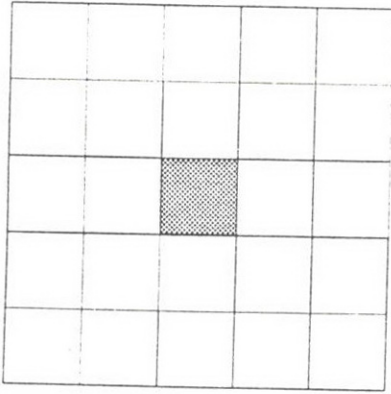
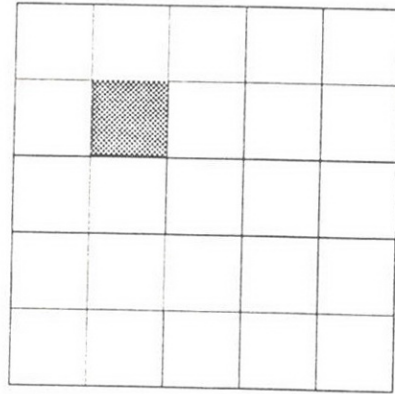


Figure 4-4. Improvement Charts for the Sixteen Patch Case.

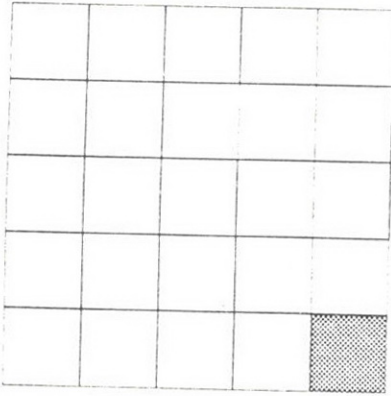
First Grid:



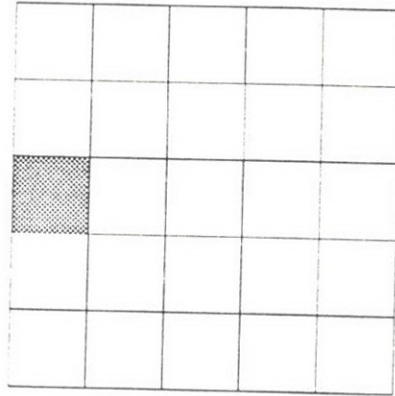
Second Grid:



Third Grid:



Fourth Grid:



Fifth Grid:

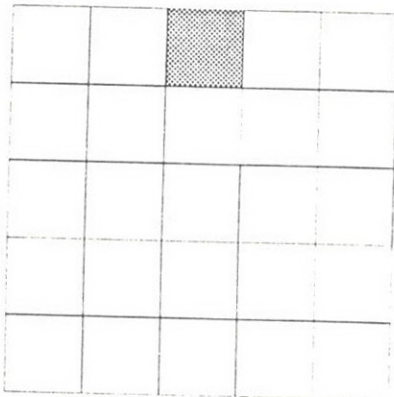


Figure 4-5. Geometry of the 25 Patch Grids.

Figure 4-6a. Maximum Fitness Improvement Plot of 5X5 Grids

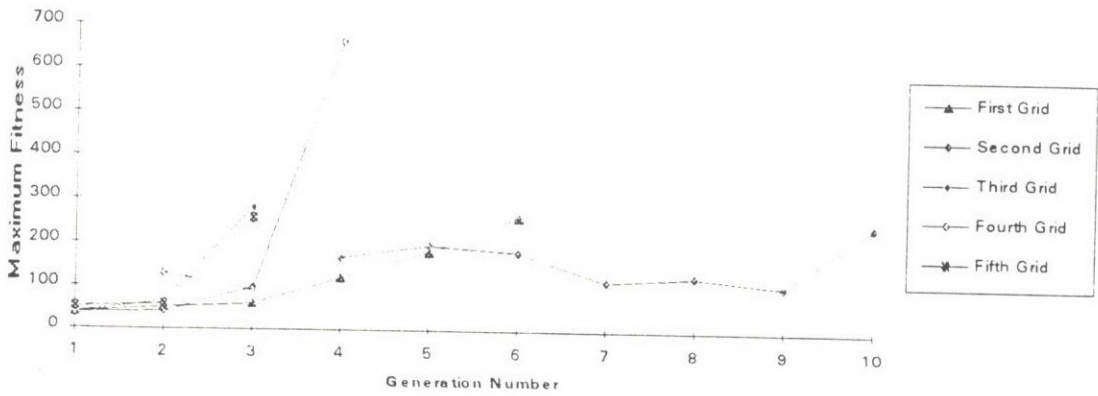


Figure 4-6b. Average Fitness Improvement Plot of 5X5 Grids

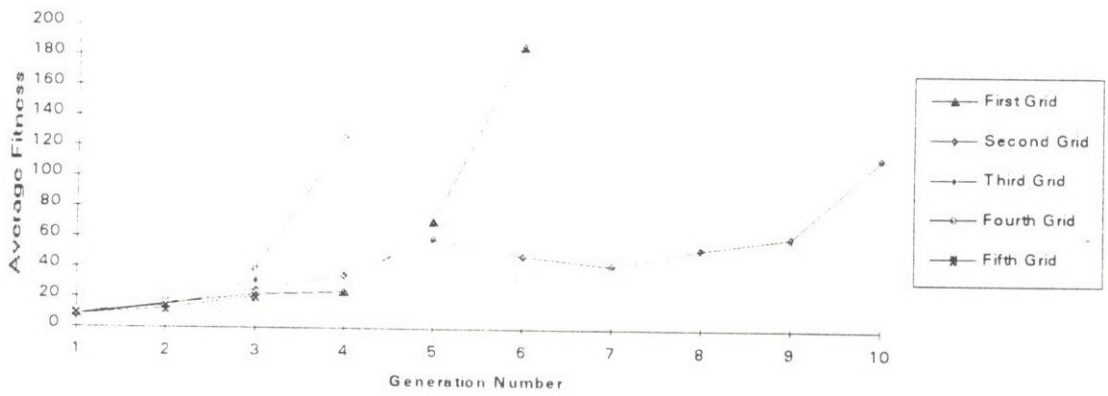


Figure 4-6c. Minimum Fitness Improvement Plot of 5X5 Grids

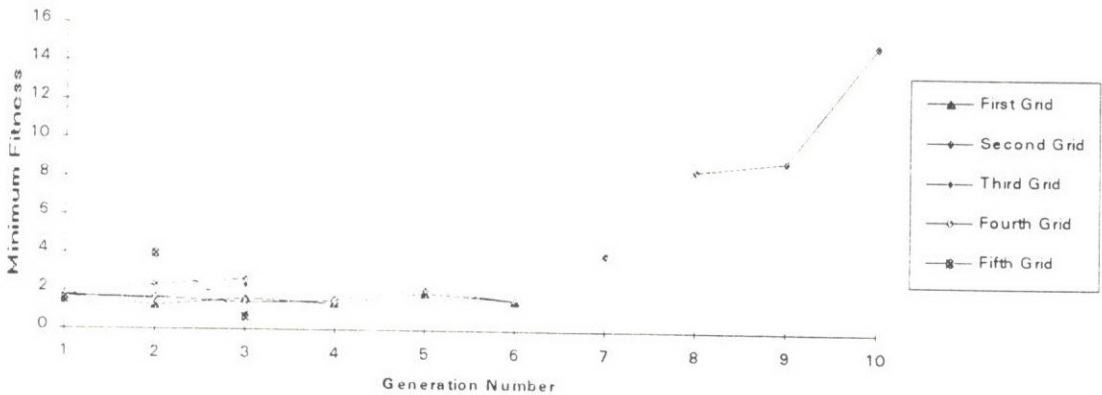


Figure 4-6. Improvement Charts for the 25 Patch Cases.

For these grids, 20 receivers and 5 transmitters were used. Figure 4-5 shows the geometry of the grids and the location of the anomalous patches in each grid.

The results of the inversions were once again successful. Table 4-3 lists the converged values.

	FIRST GRID	SECOND GRID	THIRD GRID	FOURTH GRID	FIFTH GRID
BACKGROUND CONDUCTIVITIES	1.03E-03	1.03E-03	1.00E-03	1.01E-03	1.00E-03
ANOMALOUS CONDUCTIVITIES	1.04E-02	9.88E-03	9.65E-03	9.76E-03	9.92E-03
BACKGROUND PERMITTIVITIES	3	3	3	3	3
ANOMALOUS PERMITTIVITIES	10.09	10.33	10.05	10.05	9.913

Table 4-3. Convergence values for the 25 patch cases.

Figure 4-6 shows the improvement rates for the average, maximum and minimum fitnesses. They reveal similar patterns of convergence as the previous cases. They converge to similar conductivity and permittivity values and usually in less than 10 iterations.

4.1.2. The Random Number Generator

The random number generator included in the Fortran 77 compiler for the Sun Sparc 1 stations has been known to be problematic. This brought up the dilemma that the trouble with the initial seed of the random number generator for the fifth grid on the 25 patch case (see section 4.1.1.3) was due to a problem with this function.

To try to answer this question, function RANDOM was replaced with the first random number generator routine (function RAN1) for Fortran as described by Press, Flannery, Teukolsky and Vetterling (1987). Please refer to Appendix B under the name of FUNCTION RANDOM for a program listing of RAN1. Five initial seeds were used and the results are shown on table 4-4.

Figure 4-7a. Maximum Fitness Improvement for Various Random Seeds

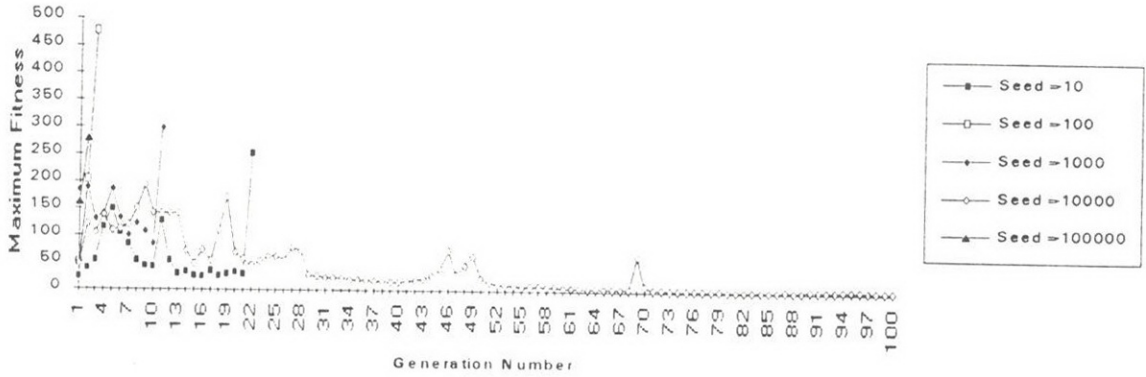


Figure 4-7b. Plot of Average Fitness Improvement for Multiple Seeds

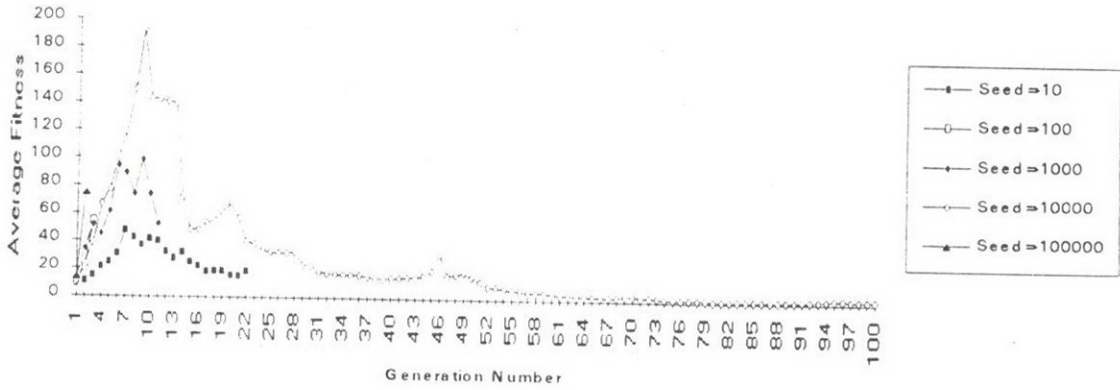


Figure 4-7c. Minimum Fitness Improvement for Various Random Seeds

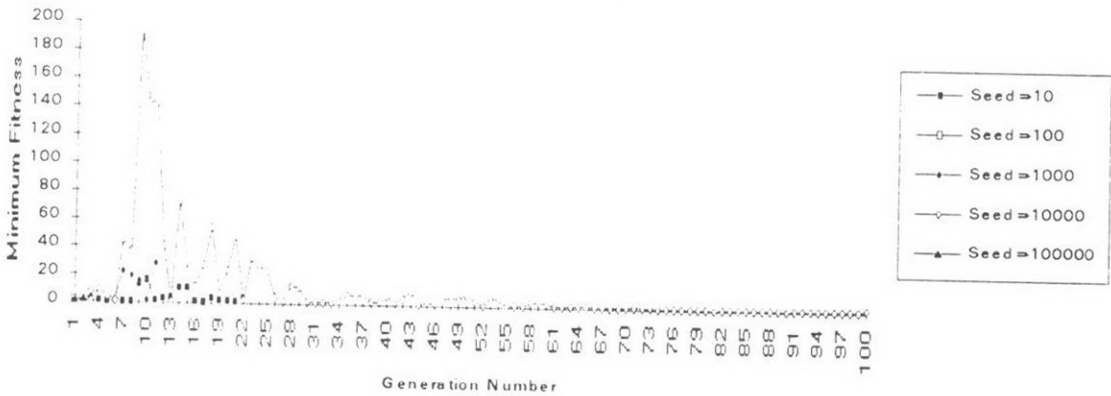


Figure 4-7. Improvement Charts for the Fifth Grid on the 25 Patch Case Using Subroutine RAN1 as the Random Number Generator.

ABSOLUTE DIFFERENCES	SEED =-10	SEED =-100	SEED =-1000	SEED =-10000	SEED =-100000
BACKGROUND CONDUCTIVITY	2.87E-04	0.00E+00	6.00E-05	2.03E-02	1.47E-05
ANOMALOUS CONDUCTIVITY	2.18E-03	1.18E-04	0.00E+00	3.80E-03	8.00E-05
BACKGROUND PERMITTIVITY	0.00E+00	0.00E+00	0.00E+00	5.36E-01	0.00E+00
ANOMALOUS PERMITTIVITY	5.20E-01	0.00E+00	1.00E-01	5.00E-01	1.00E-01
# OF GENERATIONS TO CONVERGENCE	22	3	11	100	2

Table 4-4. Errors on the Inversion of the Fifth Grid on the 25 Patch Case Using Subroutine RAN1 as the Random Number Generator with Various Initial Seeds.

Table 4-4 shows that different seeds provide different convergence speeds and accuracy. The results were good for all but the fourth seed used (seed=-10000). This indicates that different initial seeds should provide good inversions for most cases. It would be advisable for the user of our inversion program GEOTOM that if one initial seed does not function properly, the seed should be changed. This also proves that the random number generator provided with our Fortran compiler proved to be effective in our inversions. Figure 4-7 shows the improvement rates for the fitnesses using the different seeds stated in table 4-4. This figure shows that the fourth seed may be doing a poor job in choosing the random numbers for selection, crossover and mutation. To test this, the stopping criterion was reduced from 220 to 190. Now, the GA converged on generation 9 with the following values: $\epsilon_{\text{background}} = 3$, $\epsilon_{\text{anomalous}} = 10$, $\sigma_{\text{background}} = 1.029 \cdot 10^{-3}$, and $\sigma_{\text{anomalous}} = 0.9987 \cdot 10^{-2}$. This proves that convergence in this case is not a function of the seed in the random number generator, but the stopping criterion. A similar fitness function stretching approach to the one of Sen and Stoffa (1992) was attempted with subroutine FITSCALE (refer to Appendix B), but with no success.

4.1.3. The Multiple Anomalous Patches Case

When there is more than one anomalous patch, the problem becomes more complicated. The fitness function is not sensitive to each patch, but to a whole set of

patches in each grid. Here, the anomaly detection algorithm does not detect any anomalous patches, so the genetic algorithm does not converge.

It was thought at first that this may be due to the approximation of $[\underline{I} - \underline{V}\underline{V}\underline{G}]^{-1}$ as the inverse of the diagonal elements, so subroutine ONEFIT was modified, so that it would actually invert this matrix, to try to get the anomalous patch detection algorithm to be sensitive to multiple anomalous patches. The results were the same as with the approximation of the inverse of the matrix: non-convergence since no anomalous patches were detected, and all patches were set to background values of ϵ and σ . It is likely that an initial search strategy that includes a systematic evaluation of patch *combinations* rather than solely *individuals*, or a stepped inversion approach using sequentially higher resolution patch geometries may be more successful for multiple patch anomalies. This is a subject for future study.

4.2. Conclusions

- The genetic algorithms proved to be a new powerful tool to invert simple wave diffusion geotomography models. The models converged quickly and accurately. The models contained only one anomalous patch, since the multiple anomaly patch cases would not reach convergence.
- Approximating $[\underline{I} - \underline{V}\underline{V}\underline{G}]^{-1}$ as the inverse of the diagonal elements was acceptable, thus avoiding the need to invert the matrix $[\underline{I} - \underline{V}\underline{V}\underline{G}]$. This approximation seems to be quite accurate as well.
- It is necessary in the wave diffusion geotomography case to mate populations of strings where only the probable anomalous patches contain a wide range of values for the search parameters. If this is not achieved, the mating pool is diluted, and the fitnesses show no improvement throughout the generations.

- Synthetic data indicates that there is potential for the genetic algorithm approach to wave diffusion geotomography. Even though the cases tried were simple, they show that this approach may be a useful tool in the future for this type of tomographic inversions.
- The stage is now set to use genetic algorithm tomography on laboratory tank data during the next phase of research.

CHAPTER 5
FOLLOW-UP RESEARCH

Follow-up research should address the following:

1. A better method for detecting anomalous patches is needed. It is possible that considering combinations of patches instead of solely individuals in the anomalous patch recognition subroutine will be more successful for multiple anomalous patches
2. Push the GA to higher density patch networks. So far, the patch density used is relatively small.
3. Test the algorithm on tank and field data.

APPENDIX A
PROGRAM FLOWCHART AND LISTING

A.1. Initial Values

Subroutine INITIAL resets all the values for all the matrices and vectors to zero before the genetic algorithm proceeds with its computations.

Some constants need to be established for the program to use. This is done in subroutine CONST. These constants represent the following:

- SA - diameter of the circular patch equivalent to a square patch,
- OMEGA - circular frequency,
- RMU (or μ_0) - magnetic permeability of vacuum.
- EP0 (or ϵ_0) - electric permittivity of the medium,
- ZERO - complex value (0.0,0.0),
- ZONE - complex value (1.0,0.0),
- ZI - complex value (0.0,1.0),
- ZK1 - background wave number, and
- ZWT - is defined with the following equation:

$$ZWT = \frac{-2i}{\pi a^2} \left(\frac{k_1 a}{J_1(k_1 a)} \right).$$

Subroutine INPUT reads necessary variables from the input files. These are the following:

- FMHZ - probing frequency in megahertz,
- SIG1 - background conductivity of the medium,
- EP1 - background permittivity of the medium,
- NPATCH - number of patches in the interrogated section to be reconstructed,
- NREC - number of receivers,
- NTRA - number of transmitters,

- DELTA - patch dimension,
- IR, IP - dimensions of the A matrix,
- IA, JA - row and column position in A to be read,
- ZA - A matrix value to be placed in position (IA,JA),
- IP1, IP2 - dimensions of the G matrix,
- IG, JG - row and column position in G to be read,
- ZG - G matrix value to be placed in positions (IG,JG) and (JG,IG),
- IP, IF1 - dimensions of the incident field matrix F1,
- ZF1 - F1 matrix value to be placed at position (IP,IF1)
- NPOP - number of strings in the mating pool,
- SIG2MAX, SIG2MIN - maximum and minimum allowable values for conductivities,
- EP2MAX, EP2MIN - maximum and minimum allowable values for permittivities,
- NGEN - maximum number of generations on the genetic algorithm, and
- STOPIT - stopping criterion for the genetic algorithm.

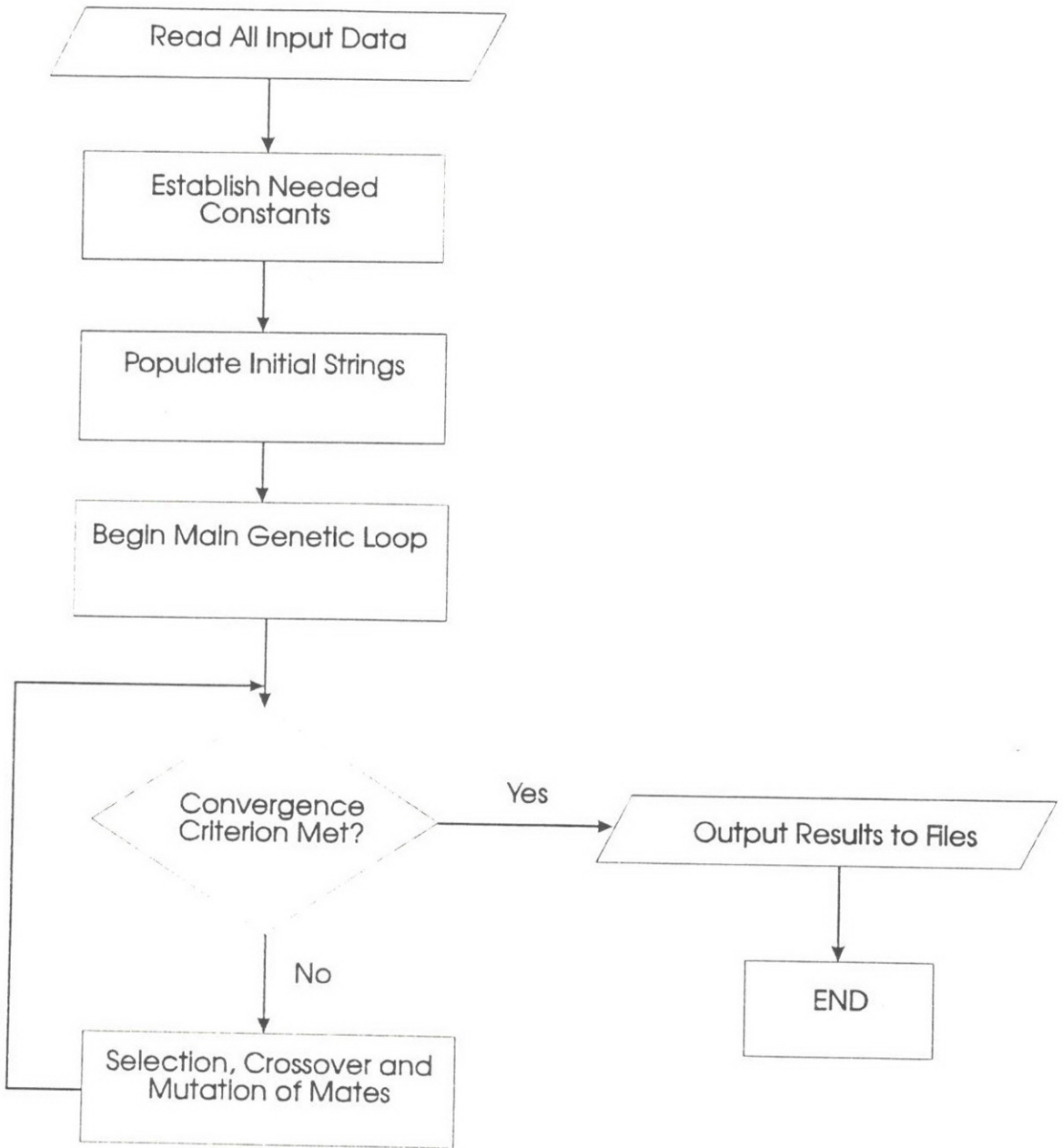


Figure A-1. Flowchart diagram of program GEOTOM.

A.2 Program Listing

C....Program GEOTOM V. 3.0

```

INTEGER MAXP,MAXR,MAXBIT,MAXPOP
PARAMETER (MAXP=225)
PARAMETER (MAXR=225)
PARAMETER (MAXBIT=20)
PARAMETER (MAXPOP=225)

COMPLEX ZERO,ZONE,ZI,ZK1,ZWT
COMPLEX A(MAXR,MAXP),G(MAXP,MAXP),F1(MAXP)
COMPLEX Y(MAXR),X(MAXPOP,MAXP),YSTAR(MAXR)
COMPLEX V(MAXP),F2(MAXP)
REAL EMAG(MAXP),EPHASE(MAXP),SIGMA(MAXPOP,MAXP),MAXFIT
REAL EPSILON(MAXPOP,MAXP),SUMFITNESS,MINFIT,FIT(MAXPOP)
REAL SIG2(MAXPOP,MAXP),EPS2(MAXPOP,MAXP),ER,SMEAN,EMEAN
REAL AVG,SR,S,E,S2,E2,SSDEV,ESDEV,EN
INTEGER CODER1(MAXBIT),CODER2(MAXBIT),CODEI1(MAXBIT)
INTEGER CODEI2(MAXBIT),END,START,CHANGE(225)
INTEGER*4 NSEED

```

C---OPEN FILES

CALL TIMEIT(BEGIN)

```

OPEN(11,FILE='GEO.DAT',STATUS='OLD',FORM='FORMATTED')
OPEN(1,FILE='REC.DAT',STATUS='OLD',FORM='FORMATTED',ERR=2)
OPEN(10,FILE='AGF.DAT',STATUS='OLD',FORM='UNFORMATTED',ERR=3)
OPEN(9,FILE='REC.OUT',STATUS='NEW',FORM='FORMATTED')
OPEN(12,FILE='SIGMA.OUT',STATUS='NEW',FORM='FORMATTED')
OPEN(13,FILE='EPSILON.OUT',STATUS='NEW',FORM='FORMATTED')
OPEN(14,FILE='GEO.INP',STATUS='OLD',FORM='FORMATTED')

```

GO TO 4

```

2 WRITE(*,*) 'ERROR - FILE "REC.DAT" NOT FOUND'
  STOP
3 WRITE(*,*) 'ERROR - FILE "AGF.DAT" NOT FOUND'
  STOP
4 CONTINUE

```

C---INITIALIZE MATRICES

```

1 CALL INITAL(A,G,F1,Y,X,FIT,V,F2,MAXR,MAXP,ZERO,MAXPOP,YSTAR,
  CODER1,CODEI1,CODER2,CODEI2,MAXBIT)

```

C---INPUT DATA FROM FILE AGF.DAT

```

      CALL INPUT(FMHZ,SIG1,EP1,DELTA,NPATCH,NREC,A,G,F1,MAXP,MAXR,
1         NPOP,SIG2MAX,SIG2MIN,EP2MAX,EP2MIN,NGEN,STOPIT)

```

C---ESTABLISH SOME NEEDED CONSTANTS

```

      CALL CONST(DELTA,FMHZ,SIG1,EP1,SA,PI,OMEGA,RMU,EP0,
1         ZERO,ZONE,ZI,ZK1,ZWT)

```

C---READ AND MODIFY THE RECEIVER DATA

```

      WRITE(*,*) '(1)...READ RECEIVER DATA'
      WRITE(9,*) '(1)...READ RECEIVER DATA'
      CALL YDAT(Y,MAXR,NREC,ZWT)

```

C---CONSTRAIN THE SOLUTION AND ESTABLISH THE INITIAL BOUNDS ON THE
C---POPULATION. ALSO COMPUTE THE INITIAL X MATRIX.

C---SET THE INITIAL SEED FOR THE RANDOM NUMBER FUNCTION

```

      NSEED=100003

```

C---RANDOMLY GENERATE INITIAL POPULATION FOR CONDUCTIVITIES AND
C---PERMITIVITIES.

```

      CALL RANGEN(SIG2MAX,EP2MAX,SIG2MIN,EP2MIN,NPATCH,NPOP,
1         MAXPOP,NSEED,SIGMA,EPSILON,SIG1,EP1,MAXP,
2         MAXR,X,A,ZERO,Y, YSTAR,F1,G,OMEGA,ZI,RMU,
3         NREC,EP0,CHANGE)

```

C---BEGIN MAIN GENETIC LOOP

```

      MUTATIONS=0
      DO 900, IGEN=1,NGEN

```

C---COMPUTE FITNESSES OF EXISTING SIGMAS AND EPSILONS

```

      CALL FITNESS(X,FIT,NPOP,MAXPOP,NREC,NPATCH,A,ZERO,MAXP,MAXR,Y,
1         YSTAR,SIGMA,EPSILON,F1,G,EP0,SIG1,EP1,OMEGA,
2         ZI,RMU,IGEN)

```

C---COMPUTE THE SUM, MINIMUM AND MAXIMUM VALUES OF FITNESSES

```

      SUMFITNESS=0.0

```

```

MINFIT=FIT(1)
MAXFIT=FIT(1)
DO 898, KK=1,NPOP
  SUMFITNESS=SUMFITNESS+FIT(KK)
  MINFIT=AMIN1(FIT(KK),MINFIT)
  MAXFIT=AMAX1(FIT(KK),MAXFIT)
898  CONTINUE
  AVG=SUMFITNESS/FLOAT(NPOP)

C---BEGIN FITNESS SCALING

C.....      CALL FITSCALE(FIT,MAXFIT,MINFIT,AVG,NPOP,SUMFITNESS)

  WRITE(*,*) 'GENERATION = ',IGEN
  WRITE(9,*) 'GENERATION = ',IGEN
  WRITE(*,*) 'AVERAGE FITNESS = ',AVG
  WRITE(*,*) 'MOST FIT STRING = ',MAXFIT
  WRITE(*,*) 'LEAST FIT STRING = ',MINFIT
  WRITE(9,*) 'AVERAGE FITNESS = ',AVG
  WRITE(9,*) 'MOST FIT STRING = ',MAXFIT
  WRITE(9,*) 'LEAST FIT STRING = ',MINFIT

C.....COMPUTE THE MEAN AND STANDARD DEVIATIONS FOR CONDUCTIVITY
C.....AND PERMITTIVITY FOR Z-SCALING.
  EN=0.0
  SMEAN=0.0
  EMEAN=0.0
  DO 237, I=1,NPOP
    DO 237, J=1,NPATCH
      SR=1.0/SIGMA(I,J)
      SMEAN=SMEAN+SR
      ER=EPSILON(I,J)
      EMEAN=EMEAN+ER
      EN=EN+1.0
237  CONTINUE
  SMEAN=SMEAN/EN
  EMEAN=EMEAN/EN
  SSDEV=0.0
  ESDEV=0.0
  DO 238, I=1,NPOP
    DO 238, J=1,NPATCH
      SR=1.0/SIGMA(I,J)
      ER=EPSILON(I,J)
      S=SR-SMEAN
      E=ER-EMEAN
      S2=S*S

```



```

      E2=E*E
      SSDEV=SSDEV+S2
      ESDEV=ESDEV+E2
238  CONTINUE
      SSDEV=SQRT(SSDEV/(EN-1.0))
      ESDEV=SQRT(ESDEV/(EN-1.0))

```

C---FIND THE LIMITS FOR SCALING -> USE RESISTIVITY FOR MORE
C---CONVENIENT SCALING.

```

      SMAX = ((1.0/SIGMA(1,1))-SMEAN)/SSDEV
      EMAX = (EPSILON(1,1)-EMEAN)/ESDEV
      SMIN = SMAX
      EMIN = EMAX
      DO 100 I = 1,NPOP
        DO 100 J = 1,NPATCH
          SR = ((1.0/SIGMA(I,J))-SMEAN)/SSDEV
          ER = (EPSILON(I,J)-EMEAN)/ESDEV
          SMAX = AMAX1(SR.SMAX)
          EMAX = AMAX1(ER.EMAX)
          EMIN = AMIN1(ER.EMIN)
          SMIN = AMIN1(SR.SMIN)
100  CONTINUE

```

C---COMPUTE SCALING CONSTANTS TO SCALE SIGMA AND EPSILON
C---FROM 0 -> 2**MEGABIT.

```

      DIFF1 = EMAX - EMIN
      DIFF2 = SMAX - SMIN
      ANUM1 = ALOG10(DIFF1)
      ANUM2 = ALOG10(DIFF2)
      DENOM = ALOG10(2.0)
      BIT1 = ANUM1/DENOM
      BIT2 = ANUM2/DENOM
      MEGABIT1 = 6
      MEGABIT2 = 6
      ALIM1 = 2.0**MEGABIT1
      ALIM2 = 2.0**MEGABIT2
      CALL SCALE(AS.BS.SMAX.SMIN.ALIM2)
      CALL SCALE(AE.BE.EMAX.EMIN.ALIM1)

```

C---CHECK FITNESS AGAINST STOPPING CRITERION.

```

      IF((MAXFIT).GE.(220)) GO TO 950

```

C---MATE THE MOST FIT STRINGS.

```

      CALL UMATEM(NPOP,NPATCH,MAXPOP,MAXP,MEGABIT1,MEGABIT2,
1      AS,BS,AE,BE,CODER1,CODEI1,CODER2,CODEI2,R,NSEED,
2      SIGMA,EPSILON,SUMFITNESS,NREC,ZERO,MAXR,FIT,
3      SIG2,EPS2,MUTATIONS,SMEAN,EMEAN,SSDEV,ESDEV,
4      CHANGE,SIG1,EP1)

```

900 CONTINUE

950 CONTINUE

C---COMPUTE FITNESSES FOR LAST GENERATION

```

      CALL FITNESS(X,FIT,NPOP,MAXPOP,NREC,NPATCH,A,ZERO,MAXP,MAXR,Y,
1      YSTAR,SIGMA,EPSILON,F1,G,EP0,SIG1,EP1,OMEGA,
2      ZI,RMU)

```

WRITE(9,*) MUTATIONS. ' MUTATIONS OCCURED.'

C---SORT SIGMA AND EPSILON ACCORDING TO FITNESSES FROM BIT TO SMALL

```

      CALL SORTEM(X,SIGMA,EPSILON,FIT,NPOP,NPATCH,MAXPOP,MAXP)

```

C---COMPUTE THE F2 VECTOR

```

      CALL TIMEIT(START)

```

```

      WRITE(*,*) '(4) COMPUTE F2'

```

```

      WRITE(9,*) '(4) COMPUTE F2'

```

```

      CALL F2COMP(F1,G,X,MAXP,NPATCH,F2,MAXPOP)

```

```

      CALL TIMEIT(END)

```

```

      SPAN = END - START

```

```

      WRITE(*,*) 'TIME TO COMPUTE F2 = ',SPAN,' SECONDS'

```

```

      WRITE(9,*) 'TIME TO COMPUTE F2 = ',SPAN,' SECONDS'

```

C---COMPUTE THE CONDUCTIVITY AND DIELECTRIC CONSTANT

```

      WRITE(*,*) '(5)...COMPUTE ELECTRICAL PARAMETERS'

```

```

      WRITE(9,*) '(5)...COMPUTE ELECTRICAL PARAMETERS'

```

```

      CALL PARCOMP(X,F1,F2,MAXP,NPATCH,OMEGA,RMU,EP0,EP1,SIG1,

```

```
1      EMAG.EPHASE.SIGMA.EPSILON,MAXPOP,NPOP)
```

```
      STOP
      END
```

```
C-----
```

```
1      SUBROUTINE SORTEM(X,SIGMA,EPSILON,FIT,NPOP,
      NPATCH,MAXPOP,MAXP)
```

```
      COMPLEX X(MAXPOP,MAXP),TEMPC
      REAL FIT(1),SIGMA(MAXPOP,MAXP),EPSILON(MAXPOP,MAXP)
```

C---THIS SUBROUTINE SORTS ARRAY X FROM LARGEST FITNESS TO SMALLEST.

```
      LIM = NPOP - 1
```

C---INITIALIZE INT TO 1 IN THE EVENT ALL FIT(I) ARE IN ORDER.

```
1      INT = 1
      DO 200 I = 1,LIM
        IF(FIT(I+1).LE.FIT(I)) GO TO 200
        TEMP = FIT(I+1)
        DO 100 J = 1,NPATCH
          TEMPC = X(I+1,J)
          X(I+1,J) = X(I,J)
          X(I,J) = TEMPC
          TEMPS = SIGMA(I+1,J)
          SIGMA(I+1,J) = SIGMA(I,J)
          SIGMA(I,J) = TEMPS
          TEMPE = EPSILON(I+1,J)
          EPSILON(I+1,J) = EPSILON(I,J)
          EPSILON(I,J) = TEMPE
100     CONTINUE
        FIT(I+1) = FIT(I)
        FIT(I) = TEMP
        INT = I
200     CONTINUE
```

C---INT GIVES THE POSITION OF THE LAST INTERCHANGE.

```
      IF(INT.EQ.1) GO TO 300
      LIM = INT - 1
      GO TO 1
```

300 CONTINUE

RETURN
END

C-----

SUBROUTINE TIMEIT(RTVAL)
INTEGER*4 ARRAY(3)
CALL ITIME(ARRAY)

RTVAL = 3600*ARRAY(1) + 60*ARRAY(2) + ARRAY(3)

RETURN
END

C-----

1 SUBROUTINE INITIAL(A,G,F1,Y,X,FIT,V,F2,MAXR,MAXP,ZERO,MAXPOP,
1 YSTAR,CODER1,CODEI1,CODER2,CODEI2,
MAXBIT)

COMPLEX A(MAXR,MAXP),G(MAXP,MAXP),F1(MAXP),X(MAXPOP,MAXP)
COMPLEX Y(MAXR),ZERO,YSTAR(MAXR)
COMPLEX V(MAXP),F2(MAXP)
REAL FIT(MAXPOP)
INTEGER CODER1(MAXBIT),CODEI1(MAXBIT),CODER2(MAXBIT)
INTEGER CODEI2(MAXBIT)

C---SET INITIAL VALUES OF ARRAYS AND MATRICES TO ZERO

ZERO = CMPLX(0.0,0.0)

DO 100 I = 1,MAXR
Y(I) = ZERO
YSTAR(I) = ZERO
DO 100 J = 1,MAXP
100 A(I,J) = ZERO

DO 110 I = 1,MAXP
F1(I) = ZERO
V(I) = ZERO
F2(I) = ZERO

```

      DO 110 J = 1,MAXP
      G(I,J) = ZERO
110  CONTINUE

```

```

      DO 200 I = 1,MAXPOP
      FIT(I) = 0.0
      DO 200 J = 1,MAXP
      X(I,J) = ZERO
200  CONTINUE

```

```

      DO 300 I = 1,MAXBIT
      CODER1(I) = 0
      CODEI1(I) = 0
      CODER2(I) = 0
      CODEI2(I) = 0
300  CONTINUE

```

```

      RETURN
      END

```

C-----

```

      SUBROUTINE INPUT(FMHZ,SIG1,EP1,DELTA,NPATCH,NREC,A,G,F1,
1          MAXP,MAXR,NPOP,SIG2MAX,SIG2MIN,EP2MAX,EP2MIN,
1          NGEN,STOPIT)

```

```

C      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      COMPLEX A(MAXR,MAXP),G(MAXP,MAXP),F1(MAXP)
      COMPLEX ZA,ZG,ZF1

```

C---READ IN ALL NECESSARY INFORMATION FOR THE GA

```

      WRITE(*,*) 'PROGRAM VECTORGA (VER. 2, 5/93) C. GLASS & J. ARCE'
      WRITE(*,*) '-----'
      WRITE(*,*) 'RUN INFORMATION'
      WRITE(*,*) '-----'
      WRITE(*,*)
      WRITE(9,*) 'PROGRAM VECTORGA (VER. 2, 5/93) C. GLASS & J. ARCE'
      WRITE(9,*) '-----'
      WRITE(9,*) 'RUN INFORMATION'
      WRITE(9,*) '-----'

```

```

      READ(10) FMHZ
      WRITE(9,*) 'PROBING FREQUENCY = ',FMHZ
      WRITE(*,*) 'PROBING FREQUENCY = ',FMHZ

```

```

READ(10) SIG1.EP1
WRITE(9,*) 'BACKGROUND CONDUCTIVITY = ',SIG1
WRITE(*,*) 'BACKGROUND CONDUCTIVITY = ',SIG1
WRITE(9,*) 'BACKGROUND PERMITTIVITY = ',EP1
WRITE(*,*) 'BACKGROUND PERMITTIVITY = ',EP1

READ(10) NPATCH,NREC.NTRA
WRITE(9,*) 'NUMBER OF INTERROGATED PATCHES = ',NPATCH
WRITE(*,*) 'NUMBER OF INTERROGATED PATCHES = ',NPATCH
WRITE(9,*) 'NUMBER OF RECEIVERS      = ',NREC
WRITE(*,*) 'NUMBER OF RECEIVERS      = ',NREC
WRITE(9,*) 'NUMBER OF TRANSMITTERS   = ',NTRA
WRITE(*,*) 'NUMBER OF TRANSMITTERS   = ',NTRA
IF(NPATCH.GT.MAXP) THEN
WRITE(*,*) 'ERROR - THERE ARE TOO MANY PATCHES (NPATCH > MAXP)'
STOP
ENDIF
IF(NREC.GT.MAXR) THEN
WRITE(*,*) 'ERROR - THERE ARE TOO MANY RECEIVERS (NREC > MAXR)'
STOP
ENDIF

READ(10) DELTA
WRITE(9,*) 'THE PATCH DIMENSION = ',DELTA
WRITE(*,*) 'THE PATCH DIMENSION = ',DELTA

WRITE(9,*) 'READING THE A MATRIX'
WRITE(*,*) 'READING THE A MATRIX'
READ(10) IR.IP
IF(IR.NE.NREC) THEN
WRITE(*,*) 'ERROR - THE NUMBER OF RECEIVERS DOES NOT EQUAL NREC'
STOP
ENDIF
IF(IP.NE.NPATCH) THEN
WRITE(*,*) 'ERROR - THE NUMBER OF PATCHES DOES NOT EQUAL NPATCH'
STOP
ENDIF
WRITE(9,*) 'THE DIMENSION OF THE A MATRIX IS ',IR,' BY ',IP
WRITE(*,*) 'THE DIMENSION OF THE A MATRIX IS ',IR,' BY ',IP
DO 100 I = 1,NREC
DO 100 J = 1,NPATCH
READ(10) IA.JA.ZA
IF(IA.NE.I.OR.JA.NE.J) THEN
WRITE(*,*) 'ERROR - ELEMENTS OF THE A MATRIX ARE OUT OF ORDER'
STOP
ENDIF

```

```

A(I,J) = ZA
100 CONTINUE
WRITE(9,*) 'READING THE G MATRIX'
WRITE(*,*) 'READING THE G MATRIX'
READ(10) IP1,IP2
IF(IP1.NE.NPATCH) THEN
WRITE(*,*) 'ERROR - IP1 DOES NOT EQUAL NPATCH'
STOP
ENDIF
IF(IP2.NE.NPATCH) THEN
WRITE(*,*) 'ERROR - IP2 DOES NOT EQUAL NPATCH'
STOP
ENDIF
WRITE(9,*) 'THE DIMENSION OF THE G MATRIX IS ',IP1,' BY ',IP2
WRITE(*,*) 'THE DIMENSION OF THE G MATRIX IS ',IP1,' BY ',IP2
DO 200 I = 1,NPATCH
DO 200 J = 1,NPATCH
READ(10) IG,JG,ZG
IF(IG.NE.I.OR.JG.NE.J) THEN
WRITE(*,*) 'ERROR - ELEMENTS OF THE G MATRIX ARE OUT OF ORDER'
STOP
ENDIF
G(I,J) = ZG
G(J,I) = ZG
200 CONTINUE
WRITE(9,*) 'READING THE INCIDENT FIELD MATRIX, F1'
WRITE(*,*) 'READING THE INCIDENT FIELD MATRIX, F1'
READ(10) IP,IF1
IF(IP.NE.NPATCH) THEN
WRITE(*,*) 'ERROR - IP DOES NOT EQUAL NPATCH'
STOP
ENDIF
IF(IF1.NE.1) THEN
WRITE(*,*) 'ERROR - IF1 DOES NOT EQUAL 1'
STOP
ENDIF
WRITE(9,*) 'THE DIMENSION OF THE F1 MATRIX IS ',IP,' BY ',IF1
WRITE(*,*) 'THE DIMENSION OF THE F1 MATRIX IS ',IP,' BY ',IF1
DO 300 I = 1,NPATCH
READ(10) IF,JF,ZF1
IF(IF.NE.I.OR.JF.NE.I) THEN
WRITE(*,*) 'ERROR - ELEMENTS OF THE F1 MATRIX ARE OUT OF ORDER'
STOP
ENDIF
F1(I) = ZF1
300 CONTINUE

```

```

READ(10) INCLUD
DO 500 I = 1,INCLUD
  READ(10) I,ANLOC
500 CONTINUE
READ(10) NPOP,SIG2MAX,SIG2MIN,EP2MAX,EP2MIN,NGEN,STOPIT

NGEN=100
WRITE(9,*) 'TOTAL MATING POPULATION = ',NPOP
WRITE(9,*) 'RANGE FOR SIGMA = ',SIG2MIN,' TO ',SIG2MAX
WRITE(9,*) 'RANGE FOR EPSILON = ',EP2MIN,' TO ',EP2MAX
WRITE(9,*) 'NUMBER OF MATING GENERATIONS = ',NGEN
WRITE(9,*) 'MINIMUM ERROR = ',STOPIT
WRITE(*,*) 'TOTAL MATING POPULATION = ',NPOP
WRITE(*,*) 'RANGE FOR SIGMA = ',SIG2MIN,' TO ',SIG2MAX
WRITE(*,*) 'RANGE FOR EPSILON = ',EP2MIN,' TO ',EP2MAX
WRITE(*,*) 'NUMBER OF MATING GENERATIONS = ',NGEN
WRITE(*,*) 'MINIMUM ERROR = ',STOPIT

RETURN
END

```

C-----

```

SUBROUTINE CONST(Delta,FMHZ,SIG1,EP1,SA,PI,OMEGA,RMU,EP0,
1      ZERO,ZONE,ZI,ZK1,ZWT)

```

```

COMPLEX ZERO,ZONE,ZI,ZK1,ZARG,ZH0,ZH1,ZJ0,ZJ1,ZWT

```

C---DECLARE SOME NECESSARY CONSTANTS

```

SA = 0.5641895836*DELTA
PI = 4.0*ATAN(1.0)
OMEGA = 2.0E+6*PI*FMHZ
RMU = 4.0E-7*PI
EP0 = 8.8541853E-12

```

```

ZERO = CMPLX(0.0,0.0)
ZONE = CMPLX(1.0,0.0)
ZI = CMPLX(0.0,1.0)

```

```

ZK1 = CSQRT(OMEGA*OMEGA*RMU*(EP0*EP1 + ZI*SIG1/OMEGA))
ZARG = ZK1*SA
CALL BSJH(ZARG,ZJ0,ZJ1,ZH0,ZH1,0)
ZWT = -2.0*ZI*ZARG/(PI*SA*SA*ZJ1)
XLAM = 2.0*PI/REAL(ZK1)
WRITE(9,*) ' BACKGROUND WAVELENGTH = ',XLAM

```



```

RETURN
END

```

```

C-----

```

```

SUBROUTINE YDAT(Y,MAXR,NREC,ZWT)

```

```

COMPLEX ZWT,Y(MAXR)

```

```

C---READ RECEIVER VECTOR

```

```

READ(1,*) NR
IF(NR.NE.NREC) THEN
WRITE(*,*) 'ERROR - NR DOES NOT EQUAL NREC'
STOP
ENDIF

```

```

DO 99 I = 1,NREC
99 READ(1,*) Y(I)

```

```

C---COMPUTE EFFECTIVE Y

```

```

DO 100 I = 1,NREC
Y(I) = Y(I)*ZWT
100 CONTINUE

```

```

RETURN
END

```

```

C-----

```

```

SUBROUTINE F2COMP(F1,G,X,MAXP,NPATCH,F2,MAXPOP)

```

```

COMPLEX F1(MAXP),G(MAXP,MAXP),F2(MAXP),X(MAXPOP,MAXP)
COMPLEX ZSUM

```

```

C---COMPUTE F2 VECTOR

```

```

DO 200 I = 1,NPATCH
ZSUM = F1(I)
DO 220 J = 1,NPATCH
ZSUM = ZSUM + G(I,J)*X(I,J)
220 CONTINUE
F2(I) = ZSUM
200 CONTINUE

```

```
RETURN
END
```

```
C-----
```

```
1 SUBROUTINE PARCOMP(X,F1,F2,MAXP,NPATCH,OMEGA,RMU,EP0,EP1,SIG1,
    EMAG,EPHASE,SIGMA,EPSILON,MAXPOP,NPOP)
```

```
C---COMPUTE EDDY CURRENT PHASE AND MAGNITUDE, CONDUCTIVITIES, AND
C---PERMITIVITY PARAMETERS
```

```
COMPLEX X(MAXPOP,MAXP),F2(MAXP),DUM,F1(MAXP)
REAL EMAG(MAXP),EPHASE(MAXP),SIGMA(MAXPOP,MAXP)
REAL EPSILON(MAXPOP,MAXP)
READ(11,*) IPATCH
READ(11,*) DUMMY
```

```
READ(14,*) NROW,NCOL
WRITE(*,*) '*****'
WRITE(*,*)
WRITE(9,*)
WRITE(9,1010)
WRITE(9,*) ' EDDY CURRENT MAGNITUDE'
```

```
DO 300 I = 1,NPATCH
    WRITE(9,*) X(1,I)
    DUM = CLOG(X(1,I))
    DI = AIMAG(DUM)
    DR = REAL(DUM)
    EMAG(I) = EXP(DR)
    EPHASE(I) = 57.29577951*DI
    READ(11,*) IPATCH,XX,YY
    WRITE(12,*) XX,YY,EMAG(I)
    WRITE(13,*) XX,YY,EPHASE(I)
300 CONTINUE
REWIND(11)
READ(11,*) IPATCH
READ(11,*) DUMMY
```

```
C.....PRINT OUT EDDY CURRENT MAGNITUDE
```

```
1010 FORMAT(1H1)
DO 320 I = 1,NROW
    INDEX = (I - 1)*NCOL + 1
    IEND = I*NCOL
    WRITE(9,1020)
```

```

1020    FORMAT(////)
        WRITE(9,1030) (EMAG(J),J = INDEX,IEND)
1030    FORMAT(10(E12.4,2X))
320    CONTINUE

```

C.....PRINT OUT EDDY CURRENT PHASE

```

        WRITE(9,1010)
        WRITE(9,*) '          EDDY CURRENT PHASE'
        DO 330 I = 1,NROW
            INDEX = (I - 1)*NCOL + 1
            IEND = I*NCOL
            WRITE(9,1020)
            WRITE(9,1030) (EPHASE(J), J = INDEX,IEND)
330    CONTINUE

```

C.....PRINT OUT CONDUCTIVITY DISTRIBUTION

```

        WRITE(9,1010)
        WRITE(9,*) '          PATCH CONDUCTIVITY USING F2'
        DO 379 I = 1,NPATCH
            EPSILON(2,I) = REAL(X(1,I)/F2(I))/(OMEGA*OMEGA*RMU*EP0)+EP1
            SIGMA(2,I) = AIMAG(X(1,I)/F2(I))/(OMEGA*RMU) + SIG1
            WRITE(*,*) ' PATCH # ',I,' :SIG = ',SIGMA(1,I)
1          ' EPSILON = ', EPSILON(1,I)
            READ(11,*) IPATCH,XX,YY
            WRITE(12,*) XX,YY,SIGMA(1,I),SIGMA(2,I)
            WRITE(13,*) XX,YY,EPSILON(1,I),EPSILON(2,I)
379    CONTINUE
        WRITE(*,*)
        WRITE(*,*) '*****'
        DO 380 I = 1,NROW
            INDEX = (I - 1)*NCOL + 1
            IEND = I*NCOL
            WRITE(9,1020)
            WRITE(9,1030) (SIGMA(2,J),J = INDEX,IEND)
380    CONTINUE
        WRITE(9,1010)

```

C.....PRINT OUT THE PERMITTIVITY DISTRIBUTION

```

        WRITE(9,*) '          PERMITTIVITY USING F2'
        DO 385 I = 1,NROW
            INDEX = (I - 1)*NCOL + 1
            IEND = I*NCOL

```

```

        WRITE(9,1020)
        WRITE(9,1030) (EPSILON(2,J),J = INDEX,IEND)
385  CONTINUE
        REWIND(11)
        READ(11,*) IPATCH
        READ(11,*) DUMMY
        WRITE(9,1010)
        WRITE(9,*) '          PATCH CONDUCTIVITY USING F1'
        DO 390 I = 1,NPATCH
        EPSILON(2,I) = REAL(X(1,I)/F1(I))/(OMEGA*OMEGA*RMU*EP0) + EP1
        SIGMA(2,I) = AIMAG(X(1,I)/F1(I))/(OMEGA*RMU) + SIG1
        READ(11,*) IPATCH,XX,YY
        WRITE(12,*) XX,YY,SIGMA(2,I)
        WRITE(13,*) XX,YY,EPSILON(2,I)
390  CONTINUE
        DO 395 I = 1,NROW
            INDEX = (I - 1)*NCOL + 1
            IEND = I*NCOL
            WRITE(9,1020)
            WRITE(9,1030) (SIGMA(2,J),J = INDEX,IEND)
395  CONTINUE
        WRITE(9,1010)

```

C.....PRINT OUT THE PERMITTIVITY DISTRIBUTION USING F1'

```

        WRITE(9,*) '          PERMITTIVITY USING F1'
        DO 400 I = 1,NROW
            INDEX = (I - 1)*NCOL + 1
            IEND = I*NCOL
            WRITE(9,1020)
            WRITE(9,1030) (EPSILON(2,J),J = INDEX,IEND)
400  CONTINUE
        WRITE(9,1020)

        RETURN
        END

```

C-----

SUBROUTINE VSETUP(VV,EP0,SIG1,EP1,OMEGA,ZI,RMU,SIG,EPS)

COMPLEX ZI,VV

VR = EP0*(EPS-EP1)

VI = (SIG-SIG1)/OMEGA

VV = CMPLX(VR,VI)

VV = VV*OMEGA*OMEGA*RMU

```

RETURN
END

```

C-----

```

SUBROUTINE UMATEM(NPOP,NPATCH,MAXPOP,MAXP,MEGABIT1,
1     MEGABIT2,AS,BS,AE,BE,CODER1,CODEI1,CODER2,CODEI2,R,NSEED,
2     SIGMA,EPSILON,SUMFITNESS,NREC,ZERO,MAXR,FIT,
3     SIG2,EPS2,MUTATIONS,SMEAN,EMEAN,SSDEV,ESDEV,
4     CHANGE,SIG1,EP1)

```

C---THIS SUBROUTINE MATES THE STRINGS.

```

INTEGER CODER1(1),CODER2(1),CODEI1(1),CODEI2(1)
INTEGER MATE1,MATE2,SELECT,CHANGE(1),POS
INTEGER*4 NSEED
REAL SIGMA(MAXPOP,MAXP),EPSILON(MAXPOP,MAXP)
REAL SUMFITNESS,SIG2(MAXPOP,MAXP),EPS2(MAXPOP,MAXP)
REAL E1,E2,S1,S2,RANDOM

```

C---START MATING OF ANOMALOUS PATCHES.

```

DO 400, J=1,NPOP, 2

```

C---SELECT TWO MATES

```

MATE1=SELECT(NPOP,SUMFITNESS,FIT,NSEED)
MATE2=SELECT(NPOP,SUMFITNESS,FIT,NSEED)

```

```

DO 460,I=1,NPATCH

```

C---CONVERT TO RESISTIVITIES. SCALE ANOMALOUS SIGMA AND EPSILON.

```

S1=((1.0/SIGMA(MATE1,I))-SMEAN)/SSDEV
S1=SCALIT(S1,AS,BS)
S2=((1.0/SIGMA(MATE2,I))-SMEAN)/SSDEV
S2=SCALIT(S2,AS,BS)
E1=(EPSILON(MATE1,I)-EMEAN)/ESDEV
E1=SCALIT(E1,AE,BE)
E2=(EPSILON(MATE2,I)-EMEAN)/ESDEV
E2=SCALIT(E2,AE,BE)

```

C---ENCODE EACH SIGMA AND EPSILON INTO BINARY STRINGS.

```

CALL ENCODE(E1.MEGABIT1.CODER1)
CALL ENCODE(S1.MEGABIT2.CODEI1)
CALL ENCODE(E2.MEGABIT1.CODER2)
CALL ENCODE(S2.MEGABIT2.CODEI2)

```

C---FIND CROSS-OVER POINTS FOR STRINGS

```

BIT1=FLOAT(MEGABIT1)
BIT2=FLOAT(MEGABIT2)
IICROSS=RANDOM(BIT2,1.0,NSEED)
IRCROSS=RANDOM(BIT1,1.0,NSEED)

```

C---CROSS THE STRINGS AND CHECK FOR MUTATION.

```

      CALL CROSOVR(CODER1,CODER2.CODEI1,CODEI2,MEGABIT1,MEGABIT2,
1      IICROSS,IRCROSS,NSEED,MUTATIONS)

```

C---DECODE THE NEW STRING INTO REAL AND IMAGINARY VALUES

```

S1 = DECODE(CODEI1,MEGABIT2)
S2 = DECODE(CODEI2,MEGABIT2)
E1 = DECODE(CODER1,MEGABIT1)
E2 = DECODE(CODER2,MEGABIT1)

```

C---WRITE CHILD STRINGS IN SIG2 AND EPS2'S ANOMALOUS PATCHES AND
C---SET ALL OTHER PATCHES TO BACKGROUND.

```

      SIG2(J,1)=DSCALIT(S1,AS,BS)
      EPS2(J,1)=DSCALIT(E1,AE,BE)
      SIG2(J+1,1)=DSCALIT(S2,AS,BS)
      EPS2(J+1,1)=DSCALIT(E2,AE,BE)
460  CONTINUE
400  CONTINUE

```

C---STORE NEW POPULATION BACK IN OUR ORIGINAL ARRAYS.

```

      DO 500 I=1,NPATCH
      DO 500 J=1,NPOP
      SIGMA(J,1)=1.0/(SIG2(J,1)*SSDEV+SMEAN)
      EPSILON(J,1)=EPS2(J,1)*ESDEV+EMEAN
500  CONTINUE

      RETURN
      END

```

C-----

SUBROUTINE SCALE(A,B,BIG,SMALL,ALIM)

C---THIS SUBROUTINE SCALES A VECTOR RANGING FROM BIG TO SMALL TO
C---THE RANGE 2**MEGABIT TO 0 (A MEGABIT BIT VALUE).

DELTA=BIG-SMALL

A=ALIM/DELTA

B=-A*SMALL

RETURN

END

C-----

FUNCTION SCALIT(VALUE,A,B)

C---SCALE VALUE TO CALCULATED RANGE

SCALIT = VALUE*A + B

RETURN

END

C-----

FUNCTION DSCALIT(VALUE,A,B)

C---DESCALES VALUE FROM CALCULATED RANGE TO ORIGINAL RANGE

DSCALIT = (VALUE - B)/A

RETURN

END

C-----

SUBROUTINE ENCODE(XX,MEGABIT,ICODE)

INTEGER ICODE(1)

C---THIS SUBROUTINE CODES A VARIABLE INTO ITS BINARY STRING.

SUB = 0.0

```

DO 100, I = 1, MEGABIT
  JBIT = MEGABIT - I
  ID = 2**JBIT
  IBIT = (XX - SUB)/FLOAT(ID)
  ICODE(I) = INT(IBIT)
  SUB = SUB + FLOAT(ICODE(I)*ID)
100 CONTINUE

RETURN
END

```

C-----

```

REAL FUNCTION DECODE(AA, MEGABIT)

INTEGER AA(I)

```

C---THIS SUBROUTINE DECODES A BINARY STRING INTO REAL VALUES.

```

DECODE = 0.0
DO 100, I = 1, MEGABIT
  J = MEGABIT - I
  DECODE = DECODE + REAL(AA(I)*(2**J))
100 CONTINUE

RETURN
END

```

C-----

```

SUBROUTINE CROSOVR(CR1, CR2, C11, C12, MEGABIT1, MEGABIT2,
1          I, J, NSEED, MUTATIONS)

```

```

INTEGER CR1(I), CR2(I), C11(I), C12(I), C
INTEGER*4 NSEED

```

C---THIS SUBROUTINE CROSSES THE GENES OF THE MATING STRINGS.

```

DO 100, K = 1, MEGABIT1
  IF(K.GE.J) THEN
    C = CR1(K)
    CR1(K) = CR2(K)
    CR2(K) = C
  ENDIF
  CALL UMUTATE(CR1(K), CR2(K), NSEED, MUTATIONS)
100 CONTINUE

```



```

DO 200, K=1, MEGABIT2
  IF(K.GE.I) THEN
    C = CI1(K)
    CI1(K) = CI2(K)
    CI2(K) = C
  ENDIF
  CALL UMUTATE(CI1(K),CI2(K),NSEED,MUTATIONS)
200 CONTINUE

RETURN
END

```

C-----

```

SUBROUTINE UMUTATE(IA,IB,NSEED,MUTATIONS)

```

```

  INTEGER*4 NSEED

```

C---THIS SUBROUTINE PERFORMS A GENE MUTATION AT A PROBABILITY OF 0.001

```

  TEST = RANDOM(10000.0,0.0,NSEED)
  ITEST = INT(TEST)
  IF(ITEST.EQ.1) THEN
    MUTATIONS=MUTATIONS+1
    TEST = RANDOM(1000.0,0.0,NSEED)
    IF(TEST.GE.500.0) THEN
      IA = ABS(IA - 1)
    ELSE
      IB = ABS(IB - 1)
    ENDIF
  ENDIF
  RETURN
END

```

C-----

```

SUBROUTINE RANGEN(SMAX,EMAX,SMIN,EMIN,NPATCH,NPOP,MAXPOP,
1      NSEED,SIGMA,EPSILON,SIG1,EPI,MAXP,MAXR,
2      X,A,ZERO,Y,YSTAR,F1,G,OMEGA,ZI,RMU,
3      NREC,EP0,CHANGE)

```

```

  COMPLEX A(MAXR,MAXP),ZERO,F1(1),X(MAXPOP,MAXP)
  COMPLEX G(MAXP,MAXP),ZI,Y(1),YSTAR(1)
  REAL SIGMA(MAXPOP,MAXP),EPSILON(MAXPOP,MAXP)
  REAL SMAX,EMAX,SMIN,EMIN,SIG1,EPI,ONEFIT

```

```
REAL BACKFIT,NEWFIT,SVAL,EVAL,RANDOM
INTEGER CHANGE(1)
```

C---THIS SUBROUTINE POPULATES THE SIGMA AND EPSILON MATRICES WITH
C---RANDOM VALUES BETWEEN (SMAX,EMAX) AND (SMIN,EMIN).

C---POPULATE ALL STRINGS WITH BACKGROUND SIGMA AND EPSILON

```
      DO 100,I = 1,NPOP
          DO 50,J = 1,NPATCH
              SIGMA(I,J) = SIGI
              EPSILON(I,J) = EPI
50          CONTINUE
100       CONTINUE
          INC=4
          DO 300,I=1,NPATCH
              WRITE (9,*) 'PATCH = ',I
              CHANGE(I)=0
              BACKFIT=ONEFIT(X,MAXPOP,NREC,NPATCH,A,ZERO,MAXP,
1              MAXR,Y,YSTAR,SIGMA,EPSILON,F1,G,EP0,
2              SIGI,EPI,OMEGA,ZI,RMU)
              WRITE (9,*) 'BACKFIT=',INC*BACKFIT
              SIGMA(1,I)=SMAX
              EPSILON(1,I)=EMAX
              NEWFIT=ONEFIT(X,MAXPOP,NREC,NPATCH,A,ZERO,MAXP,
1              MAXR,Y,YSTAR,SIGMA,EPSILON,F1,G,EP0,
2              SIGI,EPI,OMEGA,ZI,RMU)
              WRITE (9,*) 'NEW=',NEWFIT
              IF (NEWFIT.GT.(INC*BACKFIT)) CHANGE(I)=1
              SIGMA(1,I)=SMIN
              EPSILON(1,I)=EMIN
              NEWFIT=ONEFIT(X,MAXPOP,NREC,NPATCH,A,ZERO,MAXP,
1              MAXR,Y,YSTAR,SIGMA,EPSILON,F1,G,EP0,
2              SIGI,EPI,OMEGA,ZI,RMU)
              WRITE (9,*) 'NEW=',NEWFIT
              IF (NEWFIT.GT.(INC*BACKFIT)) CHANGE(I)=1
              SIGMA(1,I)=SMAX
              EPSILON(1,I)=EMIN
              NEWFIT=ONEFIT(X,MAXPOP,NREC,NPATCH,A,ZERO,MAXP,
1              MAXR,Y,YSTAR,SIGMA,EPSILON,F1,G,EP0,
2              SIGI,EPI,OMEGA,ZI,RMU)
              WRITE (9,*) 'NEW=',NEWFIT
              IF (NEWFIT.GT.(INC*BACKFIT)) CHANGE(I)=1
              SIGMA(1,I)=SMIN
              EPSILON(1,I)=EMAX
              NEWFIT=ONEFIT(X,MAXPOP,NREC,NPATCH,A,ZERO,MAXP,
1              MAXR,Y,YSTAR,SIGMA,EPSILON,F1,G,EP0,
```

```

2          SIG1,EPI,OMEGA,ZI,RMU)
WRITE (9,*) 'NEW=',NEWFIT
IF (NEWFIT.GT.(INC*BACKFIT)) CHANGE(I)=1
SIGMA(1,I)=SIG1
EPSILON(1,I)=EP1
300 CONTINUE

```

C-----PATCH CONSTRUCTION BASED ON RESULTS

```

DO 400,J=1,NPOP
SVAL=RANDOM(SMAX,SMIN,NSEED)
EVAL=RANDOM(EMAX,EMIN,NSEED)
DO 400,I=1,NPATCH
IF (CHANGE(I).EQ.1) THEN
SIGMA(J,I)=SVAL
EPSILON(J,I)=EVAL
ENDIF
400 CONTINUE

WRITE (9,*) 'ANOMALY DETECTION PATTERN'
DO 450, I=1,NPATCH
IF (CHANGE(I).EQ.1) THEN
WRITE (9,*) 'PATCH#'.I
ENDIF
450 CONTINUE

RETURN
END

```

C-----

```

SUBROUTINE FITNESS(X,FIT,NPOP,MAXPOP,NREC,NPATCH,A,ZERO,MAXP,
1          MAXR,Y,YSTAR,SIGMA,EPSILON,F1,G,EP0,
2          SIG1,EPI,OMEGA,ZI,RMU,IGEN)

COMPLEX A(MAXR,MAXP),ZERO,Y(1),YSTAR(1),SUMP,X(MAXPOP,MAXP)
COMPLEX G(MAXP,MAXP),ZONE,F1(1),VV,ZI,DIAG,ZINV,ZF2
REAL FIT(1),SIGMA(MAXPOP,MAXP),EPSILON(MAXPOP,MAXP)
REAL SUMY,B,SUMZ

```

C---THIS SUBROUTINE COMPUTES THE FITNESS FOR EACH GENETIC STRING
C---AND PLACES IT IN ARRAY FIT(NPOP).

```

ZONE = CMPLX(1.0,0.0)
RECEIVER = FLOAT(NREC)

```

```

DO 300 I = 1,NPOP
  SUMY = 0.0
  SUMZ = 0.0
  B=0.0
    DO 200 J = 1,NREC
      SUMP = ZERO
      DO 100 K = 1,NPATCH

```

C---FOR EACH SIGMA AND EPSILON, COMPUTE A V VALUE.

```

      CALL VSETUP(VV,EP0,SIG1,EP1,OMEGA,ZI,RMU,
I          SIGMA(I,K),EPSILON(I,K))

```

C---COMPUTE AN ESTIMATE OF F2 BY APROXIMATING THE (I-GV) INVERSE
C---AS A DIAGONAL MATRIX COMPRISING 1/(I-GV) TERMS ON THE DIAGONAL.

```

          DIAG = ZONE - G(K,K)*VV
          ZINV = ZONE/DIAG
          ZF2 = F1(K)*ZINV
          X(I,K) = VV*ZF2
          SUMP = SUMP + X(I,K)*A(J,K)
100      CONTINUE
          YSTAR(J) = SUMP
          YDIF=CABS(Y(J)-YSTAR(J))
          YY=CABS(Y(J))
          DUMMY=YY/YDIF
          B=AMAX1(DUMMY,B)
          SUMZ=SUMZ+DUMMY
200      CONTINUE
          SUMY=SUMZ/RECEIVER
          FIT(I) = SUMY
300      CONTINUE

      RETURN
      END

```

C-----

```

REAL FUNCTION ONEFIT(X,MAXPOP,NREC,NPATCH,A,ZERO,MAXP,
1          MAXR,Y,YSTAR,SIGMA,EPSILON,F1,G,EP0,
2          SIG1,EP1,OMEGA,ZI,RMU)

```

```

COMPLEX A(MAXR,MAXP),ZERO,Y(1),YSTAR(1),SUMP,X(MAXPOP,MAXP)
COMPLEX G(MAXP,MAXP),ZONE,F1(1),VV,ZI,DIAG,ZINV,ZF2
REAL SIGMA(MAXPOP,MAXP),EPSILON(MAXPOP,MAXP)

```

REAL SUMY,B,SUMZ

C---THIS SUBROUTINE COMPUTES THE FITNESS FOR EACH GENETIC STRING
C---AND PLACES IT IN ARRAY FIT(NPOP).

```

ZONE = CMPLX(1.0,0.0)
RECEIVER = FLOAT(NREC)
SUMY = 0.0
SUMZ = 0.0
B=0.0
    DO 200 J = 1,NREC
        SUMP = ZERO
        DO 100 K = 1,NPATCH

```

C---FOR EACH SIGMA AND EPSILON, COMPUTE A V VALUE.

```

1          CALL VSETUP(VV,EP0,SIG1,EP1,OMEGA,ZI,RMU,
                SIGMA(I,K),EPSILON(I,K))

```

C---COMPUTE AN ESTIMATE OF F2 BY APROXIMATING THE (I-GV) INVERSE
C---AS A DIAGONAL MATRIX COMPRISING 1/(I-GV) TERMS ON THE DIAGONAL.

```

                DIAG = ZONE - G(K,K)*VV
                ZINV = ZONE/DIAG
                ZF2 = F1(K)*ZINV
                X(1,K) = VV*ZF2
                SUMP = SUMP + X(1,K)*A(J,K)
100        CONTINUE
                YSTAR(J) = SUMP
                YDIF=CABS(Y(J)-YSTAR(J))
                YY=CABS(Y(J))
                DUMMY=YY/YDIF
                B=AMAX1(DUMMY,B)
                SUMZ=SUMZ+DUMMY
200        CONTINUE
                SUMY=SUMZ/RECEIVER
                ONEFIT = SUMY

```

```

RETURN
END

```

C-----

FUNCTION SELECT(NPOP,SUMFITNESS,FIT,NSEED)

REAL RAND,PARTSUM,FIT(1),SUMFITNESS,RANDOM

```

INTEGER J,NPOP
INTEGER*4 NSEED,SELECT

```

C----- THIS SUBROUTINE SELECTS A MATE FOR CROSSOVER

```

PARTSUM = 0.0
J=0
10  RAND=RANDOM(1.0,0.0,NSEED)*SUMFITNESS
    IF (RAND.EQ.0.0) GO TO 10
    DO 30,J=1,NPOP
        PARTSUM=PARTSUM+FIT(J)
        IF (PARTSUM.GE.RAND) GO TO 50
30  CONTINUE

50  SELECT = J

    RETURN
    END

```

C-----

```

REAL FUNCTION RANDOM(RMAX,RMIN,NSEED)

```

```

REAL DIFF,RMAX,RMIN,RAN1
INTEGER*4 NSEED

```

C---THIS SUBROUTINE GENERATES A RANDOM NUMBER BETWEEN RMAX AND RMIN

```

NSEED=NSEED+10
DIFF=RMAX-RMIN
RAN1=RAN(NSEED)
RANDOM=RMIN+RAN1*DIFF

```

```

RETURN
END

```

C-----

APPENDIX B

OPTIONAL PROGRAM SUBROUTINES AND FUNCTIONS

C-----

REAL FUNCTION RANDOM(RMAX,RMIN,IDUM)

REAL DIFF,RMAX,RMIN,RANI

INTEGER*4 IDUM

DIMENSION R(97)

PARAMETER (M1=259200,IA1=7141,IC1=54773,RM1=1./M1)

PARAMETER (M2=134456,IA2=8121,IC2=28411,RM2=1./M2)

PARAMETER (M3=243000,IA3=4561,IC3=51349)

DATA IFF /0/

C---THIS SUBROUTINE GENERATES A RANDOM NUMBER BETWEEN RMAX AND RMIN

DIFF=RMAX-RMIN

IF (IDUM.LT.0.OR.IFF.EQ.0) THEN

IFF=1

IX1=MOD(IC1-IDUM,M1)

IX1=MOD(IA1*IX1+IC1,M1)

IX2=MOD(IX1,M2)

IX1=MOD(IA1*IX1+IC1,M1)

IX3=MOD(IX1,M3)

DO 11, J=1,97

IX1=MOD(IA1*IX1+IC1,M1)

IX2=MOD(IA2*IX2+IC2,M2)

R(J)=(FLOAT(IX1)+FLOAT(IX2)*RM2)*RM1

11 CONTINUE

IDUM=1

ENDIF

IX1=MOD(IA1*IX1+IC1,M1)

IX2=MOD(IA2*IX2+IC2,M2)

IX3=MOD(IA3*IX3+IC3,M3)

J=1+(97*IX3)/M3

IF (J.GT.97.OR.J.LT.1) PAUSE

RANI=R(J)

RANDOM=RMIN+RANI*DIFF

R(J)=(FLOAT(IX1)+FLOAT(IX2)*RM2)*RM1

RETURN

END

C-----

```
SUBROUTINE FITSCALE(FIT,MAXFIT,MINFIT,AVG,NPOP,SUMFITNESS)
```

```
REAL FIT(1),MAXFIT,MINFIT,AVG,SUMFITNESS,FM,XMAX,XMIN
```

```
REAL DELTA,CA,CB
```

```
INTEGER NPOP
```

C---THIS SUBROUTINE SCALES THE FITNESSES ACCORDING TO
C---GOLDBERG'S ALGORITHM

```
IF (MAXFIT.GE.(2.0*AVG)) GO TO 235
```

```
FM=2.0
```

```
XMAX=FLOAT(MAXFIT)
```

```
XMIN=FLOAT(MINFIT)
```

```
TEST=(FM*AVG-XMAX)/(FM-1.0)
```

```
IF (XMIN.GT.TEST) THEN
```

```
    DELTA=XMAX-AVG
```

```
    CA=(FM-1.0)*AVG/DELTA
```

```
    CB=AVG*(XMAX-FM*AVG)/DELTA
```

```
ELSE
```

```
    DELTA=AVG-XMIN
```

```
    CA=AVG/DELTA
```

```
    CB=-XMIN*AVG/DELTA
```

```
ENDIF
```

```
SUMFITNESS=0.0
```

```
DO 234, II=1, NPOP
```

```
    FIT(II)=CA*FIT(II)+CB
```

```
    SUMFITNESS=SUMFITNESS+FIT(II)
```

```
234 CONTINUE
```

```
235 CONTINUE
```

```
RETURN
```

```
END
```

C-----

REFERENCES

- Berg, E., 1990, "Simple Convergent Genetic Algorithm for Inversion of Multiparameter Data." Society of Exploration Geophysicists Sixtieth Annual International Meeting and Exposition - SEG Abstracts, Vol. 60, pp. 1126-1128.
- Berg, E., 1991, "Convergent Genetic Algorithm for Inversion." Society of Exploration Geophysicists Sixty-First Annual International Meeting and Exposition - SEG Abstracts, Vol. 61, pp. 948-950.
- Devaney, A. J., 1982, "Inverse Scattering as a Form of Computed Tomography." Applications of Mathematics in Modern Optics, Vol. 358, pp. 10-16.
- Devaney, A. J., 1984, "Geophysical Diffraction Tomography." IEEE Transactions on Geoscience and Remote Sensing, Vol. GE-22, No. 1, pp. 3-13.
- Devaney, A. J., 1985, "Generalized Projection-Slice Theorem for Fan Beam Diffraction Tomography." Ultrasonic Imaging, Vol. 7, pp. 264-275.
- Dines, K. A. and Lytle, R. J., 1979, "Computerized Geophysical Tomography." Proceedings of the IEEE, Vol. 67, No. 7, pp. 1065-1073.
- Glass, C. E., 1986, "Final Report-High Resolution Geotechnical Exploration Using Wave-Diffusion Geotomography." USAF Ballistic Missile Office, Contract No. F04704-85-C-0121.
- Glass, C. E., 1990, "A Wave Diffusion Geotomography Approach to Providing Subsurface Images of Mining Environment." Colorado School of Mines - Rock Mechanics Contributions and Challenges: Proceedings of the 31st. U.S. Symposium, pp. 863-869.
- Goldberg, D. E., 1989, "Genetic Algorithms in Search, Optimization and Machine Learning.", Addison-Wesley Publishing Co., Inc..
- Howard, A. Q. JR., Glass C. E., Henry, D. M. N'Guessan and Siemers, D. M., 1983, "Indirect Rock Mass Investigations for Optimizing Borehole Drilling Programs."

Office of Nuclear Regulatory Research, Division of Health, Siting and Waste Management, Contract No. NRC 04-78-269.

- Howard, A. Q. JR. and Kretzschmar, J. L., 1986, "Synthesis of EM Geophysical Tomographic Data. " Proceedings of the IEEE, Vol. 74, No. 2, pp. 353-360.
- Kennett, B. L. N. and Sambridge, M. S., 1992, "Earthquake Location - Genetic Algorithms for Teleseisms." Physics of the Earth and Planetary Interiors, Vol. 75, pp. 103-110.
- Lager, D. L. and Lytle, R. J., 1977, "Determining a Subsurface Electromagnetic Profile from High-Frequency Measurements by Applying Reconstruction-Technique Algorithms." American Geophysical Union - Radio Science, Vol. 12, No. 2, pp. 249-260.
- Pan, V. and Reif, J., 1985, "Efficient Parallel Solutions of Linear Systems." Proc. 17th Annual ACM Symp. of the Theory of Computing, Rov. R.I., pp. 143-152.
- Ramirez, A. L., 1986, "Recent Experiments Using Geophysical Tomography in Fractured Granite." Proceedings of the IEEE, Vol. 74, No. 2, pp. 347-352.
- Sambridge, M. and Drijkoningen, G., 1992, "Genetic Algorithms in Seismic Waveform Inversion." Geophysical Journal Int., Vol. 109, pp. 323-342.
- Sen, M. K. and Stoffa, P. L., 1991, "Simulated Annealing, Genetic Algorithms and Seismic Waveform Inversion." Soc. of Explor. Geoph. Sixty-First Annual International Meeting and Exposition - SEG Abstracts, Vol. 61, pp. 945-947.
- Sen, M. K. and Stoffa, P. L., 1992, "Rapid Sampling of Model Space Using Genetic Algorithms: Examples from Seismic Waveform Inversion." Geophysical Journal Int., Vol. 108, pp. 281-292.
- Sheriff, R. E., 1991, "Encyclopedic Dictionary of Exploration Geophysics", Soc. of Exp. Geophys. Press, 3rd. Edition.
- Stoffa, P. L. and Sen, M. K., 1991, "Nonlinear Multiparameter Optimization Using Genetic Algorithms: Inversion of Plane-Wave Seismograms." Geophysics, Vol. 56, No. 11, pp. 1794-1810.
- Wilson, W. G. and Vasudevan K., 1991, "Application of the Genetic Algorithm to Residual Statics Estimation." Geophysical Research Letters, Vol. 18, No. 12, pp. 2181-2184.