İSTANBUL TECHNICAL UNIVERSITY ★ INSTITUTE OF SCIENCE AND TECHNOLOGY

RECONSTRUCTION OF BINARY ELECTRICAL CONDUCTIVITY DISTRIBUTIONS USING GENETIC ALGORITHMS

M.Sc. Thesis by
Çetin GÜREL

Department : Mechatronics Engineering

Programme : Mechatronics Engineering

JANUARY 2010

İSTANBUL TECHNICAL UNIVERSITY ★ INSTITUTE OF SCIENCE AND TECHNOLOGY

RECONSTRUCTION OF BINARY ELECTRICAL CONDUCTIVITY DISTRIBUTIONS USING GENETIC ALGORITHMS

M.Sc. Thesis by
Çetin GÜREL
518071006

Date of submission : 24 December 2009
Date of defence examination: 29 January 2010

Supervisor (Chairman) : Asst. Prof. Dr. Levent OVACIK (ITU)
Members of the Examining Committee : Prof. Dr. Tayfun GÜNEL (ITU)
Asst. Prof. Dr. Canbolat UÇAK (YU)

JANUARY 2010

İSTANBUL TEKNİK ÜNİVERSİTESİ ★ FEN BİLİMLERİ ENSTİTÜSÜ

İKİLİ ELEKTRİK İLETKENLİK DAĞILIMLARININ GENETİK ALGORİTMALAR İLE YENİDEN OLUŞTURULMASI

YÜKSEK LİSANS TEZİ
Çetin GÜREL
518071006

Tezin Enstitüye Verildiği Tarih : 24 Aralık 2009
Tezin Savunulduğu Tarih : 29 Ocak 2010

Tez Danışmanı : Y. Doç. Dr. Levent OVACIK (İTÜ)
Diğer Jüri Üyeleri : Prof. Dr. Tayfun GÜNEL (İTÜ)
Doç. Dr. Canbolat UÇAK (YÜ)

OCAK 2010

FOREWORD

I wish to thank my parents for their endless love, support and patience. This thesis would not have been possible without the support and encouragement of my parents.

I would like to express my deep appreciations to my thesis supervisor Asst. Prof. Dr. Levent Ovacık for his warm encouragement and thoughtful guidance.

Finally, I thank TÜBİTAK for their support.

January 2010

Çetin GÜREL

Mechanical Engineer

TABLE OF CONTENTS

Page

ABBREVIATIONS

CPU **:** Central Processing Unit
EIT **:** Electrical Impedance Tomography
EII **:** Electrical Impedance Imaging
EC **:** Evolutionary Computation
FDM **:** Finite Difference Method
FEM **:** Finite Element Method
GA **:** Genetic Algorithm
RAM **:** Random Access Memory
SNR **:** Signal to Noise Ratio

LIST OF TABLES

Page

LIST OF FIGURES

Page

LIST OF SYMBOLS

Y	: Admittance matrix
υ_1	: Base selection pressure factor
$b(m)$	: Binary value of m-th bit
$\alpha_0, \alpha_1, \alpha_2, \alpha_3$	: Coefficients for transformation of x-plane
$\beta_0, \beta_1, \beta_2, \beta_3$	: Coefficients for transformation of y-plane
ξ, η	: Coordinates of the square plane
θ	: Crossover rate adaptation band
P_c	: Crossover rate
J	: Current density
J_E	: Current density vector of injected excitation
q^e	: Current flux vector in element-e
C	: Current Matrix
λ	: Diversity of the population
Ω_e	: Domain of the element-e
σ	: Electrical conductivity
$\bar{E}$	: Electrical field
ω	: Electrical Frequency
ε	: Electrical Permittivity
ϕ	: Electrical potential
V_i	: Electrical voltage on i-th node
D	: Electric flux density
e_f	: Error value of the individuals
φ	: Error value of the individuals after application of weight function
w_f	: Error values used in weight function
f	: Fitness string
F_{ij}	: Function used in (i,j)-th entry of the global admittance matrix
ξ_m, η_n	: Gauss points
w_m, w_n	: Gauss weights
w_{max}	: Highest error function value in weight function
Ω_H	: Homogeneous zones of the boundary condition
Ω	: Imaging Domain
J	: Jacobean matrix
Y_{ij}	: (i,j)-th entry of the element admittance matrix
g_m	: m-th bit of an individual
H	: Magnetic field density
P_{max}	: Maximum limit value for mutation probability
φ_{mean}	: Mean value of error values of all individuals
P_{min}	: Minimum limit value for mutation probability
τ_1, τ_2	: Mutation adaptation factors
P_m	: Mutation probability
I_m	: Net current flowing into the m-th element
l	: Normalized mutation parameter
r_n	: Normalized rank of the individuals

P : Number of current excitations
E : Number of electrodes
n : Outward normal vector on the boundary surface
I : Population matrix
r : Rank of the individuals
υ_2 : Selection pressure band factor
β : Selection pressure parameter
p : Selection probability of the individuals
s : Selection string
N_i : Shape function for i-th node
Ω_E : Surface of the electrodes of the boundary condition
i : String of an individual
R : Total number of individuals in a population
N : Total number of nodes
V : Voltage matrix
V_0 : Voltage values from actual measurement
V_{ind} : Voltage values from numerical simulation
H_d : Voltage values using homogeneous distribution
T_d : Voltage values using target distribution
α : Weight function parameter
W_f : Weight function scaling factors
W : Weighting function of Galerkin's weighted-residual method

RECONSTRUCTION OF BINARY ELECTRICAL CONDUCTIVITY DISTRIBUTIONS USING GENETIC ALGORITHMS

SUMMARY

Electrical impedance imaging is a noninvasive technique to determine the internal conductivity distribution of an object based on electrical measurements obtained on its outer boundary. This technique has been increasingly used in recent decades for monitoring industrial processes for safe, reliable, and optimal operating conditions. The wide acceptance of electrical impedance imaging technique is mainly due to its safety, unique portability, and its dependence on sufficiently inexpensive data acquisition hardware. However, the problem of image reconstruction to calculate the unknown electrical properties inside the object is extremely ill conditioned due to the nonlinear relationship between the measured data and the unknown conducivity parameters. In addition to the ill conditioning, search space of candidate solutions to the image reconstruction problem is excessively large, making the problem largely dependent on the computational efficiency of the solution algorithm. Although deterministic optimization algorithms based on differential search directions are widely used in image reconstructions, there are several new promising studies linking stochastic algorithms and the impedance imaging method in recent years. Genetic algorithms are stochastic search and optimization methods that are inspired by the principles of biological evolution to achieve convergence to a population of candidate solutions by using genetic operators such as selection, crossover and mutation. Particularly for ill-conditioned problems, genetic algorithms have significant advantages over the deterministic methods due to their stochastic nature, parallel searching capabilities and robustness in avoiding local minima.

In this thesis, an improved genetic algorithm is developed for the reconstruction of two-dimensional and binary conductivity distributions in electrical impedance imaging. The electrical impedance imaging method used in this thesis is based on the minimization of the discrepancies between measured and computed electrode voltages in a least-square sense. The electrode voltages are obtained from two-dimensional 16-electrode and 32-electrode phantoms, modeled by the finite element method using 9x9 and 17x17 quadrilateral elements, respectively. The voltage response on the boundary electrodes induced by the electrostatic field for a known conductivity distribution and injected electrode currents is simulated by the finite element method. The problem of ill conditioning due to the relatively weak voltage response to the targets that are located far away from the boundary electrodes is surmounted by a new special weight function developed in this thesis. This weight function calculates the scale factors for each current excitation pattern to equalize the contribution of different regions of the conductivity distribution to the fitness values of the candidate solutions.

The genetic algorithm developed for image reconstructions consists of two stages, each having different objectives and different genetic operators. The aim of the first stage is to make the population converge near the optimal solution. Because the

initial population of a genetic algorithm is randomly created, the diversity of the population is very rich at the beginning of the algorithm. Therefore, high convergence speeds can be achieved in the first stage with high selection pressures and low mutation probabilities. Convergence speed of a genetic algorithm generally becomes slower as the population converges near the optimal solution. As the convergence is achieved, the diversity of the population dramatically decreases. However, for the final iterations of the algorithm, diversity must be forced to increase by using high mutation probabilities and low selection pressures. The aim of the second stage is eventually to attain the true conductivity distribution by increasing the mutation probability and decreasing the selection pressure.

Four new mutation operators are developed in this thesis. Two of the mutation operators work in a shape searching mentality to aid the algorithm to attain the true conductivity distribution in the second stage. The other two mutation operators work in both stages of the algoritm to help the algorithm to avoid the premature convergence. An improved ranked proportionate selection operator is developed to prevent any candidate solution from dominating over others. Uniform crossover method is used in the algorithm as recombination operator to ensure an effective mixture of genes among the population.

Two most important factors for a genetic algorithm are the diversity of the population and the convergence speed of the algorithm. Genetic algorithms achieve convergence at the expense of diversity. Increasing the convergence speed decreases the diversity of the population. On the other hand, rich diversity provides robustness to a genetic algorithm. With a less diverse population, genetic algorithms are more likely to be trapped in local minima. Therefore, efficiency of the genetic algorithm is maximized when the convergence speed and the diversity are optimally balanced. By using parameter adaptation operator, which is developed to achieve efficiency from the start to the end of the algorithm, important parameters of the genetic algorithm, such as the selection pressure, the mutation probability and the crossover probability are controlled adaptively to maintain the diversity of the population at an efficient level.

A series of tests is conducted to observe the genetic algorithms performance on various conditions. Measurement process of each test is simulated using the finite element model with the optional addition of Gaussian white noise. The genetic algorithm performed well by attaining the true conductivity distribution in most of the tests for both 16-electrode and 32-electrode model without noise. The algorithm achieved convergence in all the tests with noise and attained the true conductivity distribution up to a certain noise level, showing robust characteristics. In all tests, it is observed that the adaptive parameter control effectively helps maintaining the diversity of population as the process converges.

İKİLİ ELEKTRİK İLETKENLİK DAĞILIMLARININ GENETİK ALGORİTMALAR İLE YENİDEN OLUŞTURULMASI

ÖZET

Elektriksel empedans görüntüleme, bir nesnenin içsel iletkenlik dağılımının dış sınırlarından elde edilen elektriksel ölçümlere dayanarak belirlendiği girişimsel olmayan bir görüntüleme yöntemdir. Bu yöntem; güvenli, güvenilir ve optimal çalışma koşullarının sağlanması için endüstriyel proseslerin görüntülenmesinde son on yıllarda artan bir oranla kullanım alanı bulmaktadır. Elektriksel empedans görüntüleme tekniğinin geniş uygulama alanlarında kabul görmesinin başlıca nedenleri yöntemin güvenliği, kendine özgü taşınabilirliği ve yeterince ucuz veri toplama donanımına bağımlı olmasıdır. Ancak, görüntülenen nesnenin içerisinin elektriksel özelliklerinin hesaplandığı görüntü oluşturma problemi, ölçülen veri ile bilinmeyen iletkenlik parametreleri arasındaki doğrusal olmayan ilişki nedeniyle son derece kötü koşullu bir problemdir. Bununla birlikte, görüntü oluşturma probleminin aday çözümlerini kapsayan arama uzayı aşırı ölçüde büyüktür ve bu durum problemi çözüm algoritmasının etkinliğine oldukça bağımlı duruma getirir. Deterministik prensibe sahip görüntü oluşturma algoritmalarının yaygın kullanımına karşın, son yıllarda stokastik algoritmalar ile elektriksel empedans görüntüleme yöntemini birleştiren ümit verici çalışmalar yapılmıştır. Genetik algoritma, biyolojik evrimin prensiplerinden esinlenerek, aday çözümlerin oluşturduğu bir popülasyonda yakınsama sağlanması amacıyla seçilim, çaprazlama ve mutasyon gibi genetik operatörlerin kullanıldığı stokastik arama ve optimizasyon yöntemidir. Genetik algoritmalar; stokastik yapıları, paralel arama kapasiteleri ve yerel minimum noktalardan kurtulmadaki dayanıklılık nitelikleri sayesinde özellikle kötü koşullu problemlerin çözümünde deterministik yöntemlere göre önemli avantajlara sahiptir.

Bu tezde, elektriksel empedans görüntüleme prensibi kullanılarak iki boyutlu ve ikili iletkenlik dağılımlarının yeniden oluşturulması amacıyla iyileştirilmiş bir genetik algoritma geliştirilmiştir. Tezde kullanılan elektriksel empedans görüntüleme yöntemi; ölçülen ve hesaplanan elektrot gerilim değerlerinin farklılıklarının en küçük kareler yaklaşımıyla minimizasyonuna dayanmaktadır. Elektrot gerilimleri, sırasıyla 9x9 ve 17x17 dörtgen eleman kullanılarak sonlu elemanlar yöntemi ile modellenen iki boyutlu 16 ve 32 elektrotlu fantomlardan elde edilir. Bilinen bir iletkenlik dağılımına sahip ve elektrotlarından akım uygulanan elektrostatik bir alan tarafından uyarılan sınır elektrotlarındaki gerilimler sonlu elemanlar metodu kullanılarak simüle edilir. Sınır elektrotlarından uzakta bulunan hedeflerin elektrotlarda göreceli olarak düşük bir gerilim değişimine neden olmasından kaynaklanan kötü koşulluluk sorunu, bu tezde yeni olarak geliştirilen özel bir ağırlık fonksiyonu ile aşılmıştır. Ağırlık fonksiyonu, iletkenlik dağılımındaki değişik bölgelerin aday çözümlerin uygunluk değerlerine olan katkısını eşitlemek üzere her akım uygulama kalıbı için oranlama faktörlerini hesaplamaktadır.

Görüntü oluşturma probleminin çözümü için geliştirilen genetik algoritma, her bir aşaması farklı hedeflere ve farklı genetik operatörlere sahip olmak üzere iki

aşamadan oluşmaktadır. İlk aşamanın amacı optimal çözüme yaklaşılacak şekilde yakınsama sağlamaktır. Genetik algoritmanın İlk popülasyonu rastgele oluşturulduğu için, başlangıçta popülasyonun çeşitliliği oldukça zengindir. Bu nedenle algoritmanın ilk aşamasında yüksek seçilim baskısı ve düşük mutasyon olasılıkları kullanılarak yüksek yakınsama hızları sağlanabilir. Genetik algoritmaların yakınsama hızları genel olarak optimal çözüme doğru yakınsama sağlandıkça azalır. Yakınsama gerçekleştikçe popülasyonun çeşitliliği çarpıcı bir biçimde düşer. Ancak, algoritmanın son iterasyonlarında, yüksek mutasyon olasılıkları ve düşük seçilim baskıları ile çeşitlilik yükselmeye zorlanmalıdır. Algoritmanın ikinci aşamasının amacı mutasyon olasılığını artırıp, seçilim baskısını azaltarak tam iletkenlik dağılımına ulaşmaktır.

Bu tez çalışmasında dört yeni mutasyon operatörü geliştirilmiştir. Bu mutasyon operatörlerinden ikisi algoritmanın ikinci aşamasında tam iletkenlik dağılımına ulaşılmasını kolaylaştırmak için biçim arama anlayışı ile çalışmaktadır. Diğer iki mutasyon operatörü ise prematüre yakınsama durumundan kaçınmak için her iki aşamada da çalışmak üzere düzenlenmiştir. Bir diğer geliştirme de, herhangi bir aday çözümün diğerleri üzerinde baskın hale gelmesini önlemek amacıyla gerçekleştirilen sıra orantılı seçilim operatörünün iyileştirilmesidir. Popülasyon içindeki genlerin etkin karışımını sağlamak için de algoritmanın yeniden birleştirim operatörü olarak birörnek çaprazlama yöntemi kullanılmıştır.

Genetik algoritmalar için en önemli iki faktör popülasyonun çeşitliliği ve algoritmanın yakınsama hızıdır. Genetik algoritmalar çeşitlilik kaybederek yakınsama sağlar. Yakınsama hızının artırılması popülasyonun çeşitliliğinin düşmesine neden olur. Diğer yandan, yüksek çeşitlilik algoritmaya dayanıklılık özellikleri kazandırır. Popülasyonun çeşitliliği düştükçe, genetik algoritmanın yerel minimum noktalarında takılma olasılığı yükselir. Bu nedenle, genetik algoritmanın verimi ancak yakınsama hızı ile popülasyonun çeşitliliği optimal olarak dengelendiğinde maksimum değere ulaşır. Baştan sona verimliliğin sağlanması için geliştirilen parametre adaptasyon operatörünün yardımıyla; seçilim baskısı, mutasyon olasılığı ve çaprazlama olasılığı gibi genetik algoritmanın önemli parametreleri, popülasyonun çeşitliliğini verimli bir düzeyde korunmak için uyarlamalı olarak kontrol edildi.

Genetik algoritmanın farklı koşullardaki performansının gözlemlenmesi için denemeler gerçekleştirildi. Her denemenin ölçüm işlemi sonlu elemanlar modeli kullanılarak ve seçime bağlı Gaussian beyaz gürültü eklenerek simüle edildi. Genetik algoritma, 16 elektrotlu ve 32 elektrotlu model üzerinde gürültü içermeyen veri kullanılarak yapılan çoğu denemede, gerçek iletkenlik dağılımına ulaşarak oldukça iyi bir performans gösterdi. Algoritmanın gürültü içeren veri kullanılarak yapılan denemelerde yakınsama sağladığı ve belirli bir gürültü düzeyine kadar gerçek iletkenlik dağılımına ulaşarak dayanıklı bir karakteristik sergilediği gözlemlendi. Bütün denemelerde, uyarlamalı parametre kontrolünün, bir yandan yakınsama sağlanırken, çeşitliliğin korunmasına etkin bir biçimde yardım ettiği saptandı.

1. INTRODUCTION

Recent technological advances in computerized tomography have led to many useful and inspiring results for visualization of inaccessible objects or media. Despite the high quality of images obtained with X-ray, positron emission and nuclear magnetic resonance tomography, the use of highly sophisticated equipment for such imaging modalities under uncontrollable harsh environmental conditions (due to excessive heat, pressure, electromagnetic interference, etc.) is problematic. For various industrial processes, particularly in heat exchangers, natural gas pumping systems and underwater petroleum pipelines, determination of spatial and temporal distribution of phase boundaries within two-phase flow fields is a critical task to assess safety and optimality in the process design. The imaging systems used for these processes depend on sufficiently fast, portable, inexpensive, and sensitive measurement and data acquisition instrument (Ceccio and George, 1996).

Alternatively, some other nondestructive testing techniques based on acoustical and electrical impedance measurements offer great advantages to overcome the drawbacks of these imaging techniques. They require a relatively simple sensing hardware, but intrinsically suffer from an ill conditioning problem due to integral/differential operators relating the measured data to the properties of sensed field (Rolnik and Seleghim, 2006).

Acoustical sensing methods benefit from the principles of reflection and attenuation of ultrasonic waves propagating in a medium with different sonic features. The reflection and attenuation coefficients of the medium are determined from acoustical sensing devices placed on the outer boundary of the flow medium. This technique is prone to some imaging artifacts due to scattering and diffraction of incident waves at liquid-gas interfaces whenever the wavelength of the ultrasonic sound is close to the size of the phase boundary (Atkinson and Kytomaa, 1992). A crucial resolution problem arises with particularly small-sized gas bubbles when their size is significantly smaller than the wavelength of the ultrasonic wave. As the wave frequency is increased to a 10-30 MHz range to annihilate this resolution problem, the high-frequency ultrasonic waves heavily attenuate and slowly propagate into the

flow medium as a result of multiple reflections of the ultrasonic wave in gas phases. This natural behavior of ultrasonic waves degrades the measurement speed, impeding the use of ultrasonic methods for real-time visualization of dynamic flow regimes (Atkinson and Kytomaa, 1993).

Imaging techniques based on electrical impedance measurements, however, do not severely suffer from the constraints arising from slow propagation of applied current waves. The gas phase has virtually no electrical conductivity and its permittivity is about two orders of magnitude lower than that of the liquid phase. Thus, the gas phase behaves as a purely capacitive medium into which propagation of electrical currents within 10 kHz to 1 MHz range is insignificant. If the conductivity of the liquid phase is significantly larger than its capacitive component at this frequency range, the electrical time constant of the medium is negligibly small, behaving a purely conductive medium. Since early 1990s, electrical impedance tomography (EIT) has been considered as a new visualization tool for two-phase flows. As the electrical conductivity (and/or permittivity) of the liquid phase significantly differ from that of the gas phase, the phase boundaries can be identified from the spatial variation of electrical properties without disturbing the flow field. Therefore, distribution of electrical properties is inferred from surface measurements of electric potentials resulting from independent electric current patterns repetitively injected into the outer surface of the flow. The hardware used for EIT systems is relatively inexpensive, portable and fast enough to acquire data in reasonable time and accuracy. This imaging modality is accepted as the most suitable one compared with other known modalities.

1.1 Background

The idea of impedance methods originated in geophysics in early 1930s (Langer, 1933) when Slichter (Slichter, 1933) attempted to determine the electrical resistivity of horizontally uniform geological structures from potential measurements observed on the earth's surface. This problem was first identified by Calderon (Calderon, 1980) in 1980 as the "inverse conductivity problem" in mathematics literature. The mathematical study inverse conductivity problem has had a great impetus to many mathematicians and scientists until mid 1990s with the proof of uniqueness by Kohn

and Vogelius (Kohn and Vogelius, 1984), Sylvester and Uhlmann (Sylvester and Uhlmann, 1987) and later by Nachman (Nachman, 1995).

Active research in EIT has started in the late 1970s when Henderson and Webster developed an EIT system as a medical imaging tool (Henderson and Webster, 1978). In about the same period Little and Dynes and independently presented a new EIT system for geophysical surveying (Dynes and Lytle, 1981). Since 1980s the computing power has incresed drastically and many large scale EIT systems are developed by different research groups worldwide.

The reconstruction algorithms for EIT can be sorted into noniterative and iterative algorithms. Noniterative algorithms are based on linear approximations relying on the assumption that the conductivity does not differ very much from a constant. Most important noniterative reconstruction algorithms are backprojection method (Barber and Brown, 1984) and Newton's one-step reconstruction algorithm (Cheney et al., 1990). These methods generally operate by using simplifying assumptions that limit the accuracy and the scope of their application to few problems of practical interest.

Iterative algorithms are based on the premises that conductivity of the visualized body differs slightly from a known conductivity distribution. They require relatively fewer assumptions, therefore yielding better approximate solutions to the image reconstruction problem than using noniterative methods. Furthermore, iterative methods have a wider range of application since the construction of the solution algorithms is relatively easy.

Iterative reconstruction algorithms can be categorized into two groups, consisting of deterministic and stochastic algorithms. The most popular deterministic reconstruction algorithms are based on regularized Newton-Raphson method and their variations are widely used in industrial applications (Yorkey et al., 1987). Jones, Lin, Ovacık and Shu created an algorithm named "block decomposition method" by combining the finite element and Newton-Raphson methods in the early 1990s. This new method allowed the number of elements used in the finite element model to decrease by applying locally analytic solutions as the shape functions inside the elements (Jones et al., 1993).

In recent decades, stochastic reconstruction algorithms are started to gain popularity due to their robustness and ability to avoid local minima more effectively. Genetic

algorithms (Cheng et al., 1996) and Monte Carlo method (Kaipio et al., 2000) are among the stochastic methods applied to the EIT image reconstruction problem. Of these studies, most promising results are obtained using the genetic algorithm method due to its parallel processing capability. In early 2000s, Olmi, Bini and Priori created an improved genetic algorithm for reconstruction of high-resolution images by using three successive genetic algorithms, each with different parameters (Olmi et al., 2000). Kim, Moon and their co-workers improved the efficiency of the image reconstruction process by developing a two-stage genetic algorithm (Kim et al., 2002).

1.2 Contributions

First contribution of this thesis is the development of a new special weight function to overcome the problem of ill conditioning due to the relatively weak voltage response to the targets that are located far away from the boundary electrodes. Another improvement is the two-stage genetic algorithm structure, which is designed to provide more efficient search strategies to the algorithm by using different genetic operators and parameters for each stage. Four new mutation operators are developed in this thesis, two of the them helping the algorithm to avoid the premature convergence and the other two working in a shape searching mentality to aid the algorithm to attain the true conductivity distribution in the second stage. Another contribution is the development of an improved ranked proportionate selection operator to achieve more efficient selection process. Final contribution of this thesis is the parameter adaptation operator, which adaptively controls the important parameters of the genetic algorithm, such as the selection pressure, the mutation probability and the crossover probability to maintain the diversity of the population at an efficient level.

1.3 Thesis Overview

This thesis consists of five chapters. Chapter 1 provides an introduction on the non-invasive imaging methods and includes a literature survey on the image reconstruction problem of EII method. Contributions of this thesis are also discussed in the first chapter.

Chapter 2 presents the finite element model developed for the solution of the forward conductivity problem. After stating the governing field equation, the finite element method formulation and the developed numerical simulation algorithm are presented in chapter 2.

Chapter 3 states the image reconstruction problem of EII method, providing knowledge on the ill conditioning nature and the search space of the problem. This chapter gives a general overview of evolutionary computation methods and genetic algorithms including their historical background. The general structure and each individual component of the genetic algorithm developed for the solution of the image reconstruction problem are also explained in detail in this chapter.

Chapter 4 is devoted to the numerical simulations conducted to test the genetic algorithm for the image reconstruction problem. Results of the various tests obtained by the genetic algorithm using the data from the numerical simulations are presented in chapter 4.

Finally, chapter 5 concludes the thesis, by briefly summarizing the study, discussing the results obtained from the numerical simulations presented in the fourth chapter. Performance of the genetic algorithm and its individual components are discussed in detail. This chapter ends with suggestions for future work for the development of genetic algorithms for the solution of the EII reconstruction problem.

2. SOLUTION OF FORWARD PROBLEM

2.1 Conductivity Problem

The concept of EII method involves a body with an unknown field of electrical conductivity distribution that is surrounded by electrodes placed on the boundary surface. The electrodes on the boundary surface are excited using different patterns, and the responding voltages on the electrodes are measured. A schematic representation of a typical electrical impedance tomography system is shown is Figure 2.1. Solution of the forward problem of EII is the simulation of the actual measurement of the imaging process. Because the inverse solution of the forward problem is impossible, the solution of the forward problem has to be obtained in order to solve the inverse problem. This situation makes the solution of the conductivity problem necessary for imaging process. Solution of the conductivity problem consists of calculation of measured voltage response values on the boundary electrodes of a body with a known conductivity distribution induced by known injected currents. To simulate the behavior of an electrostatic field with a variable conductivity distribution, solution of the Poisson's equation for a variable complex conductivity field has to be solved with the appropriate boundary conditions.

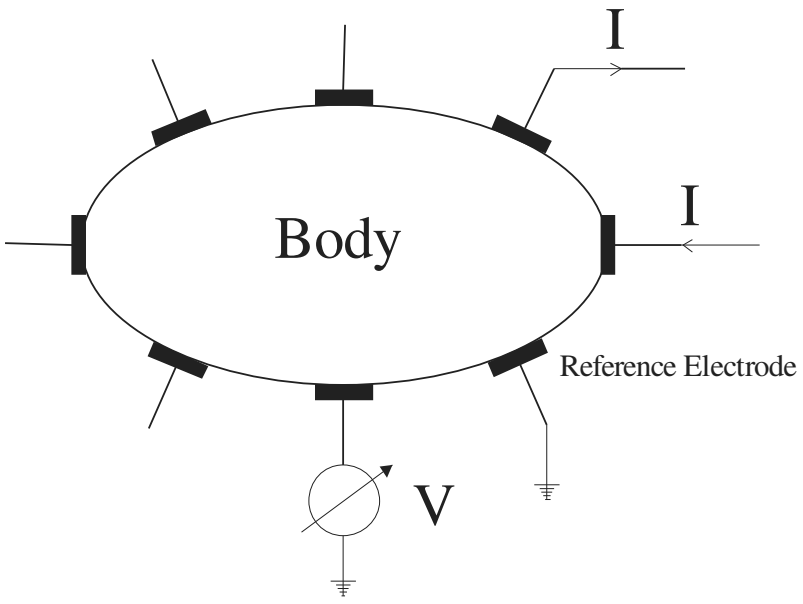


Figure 2.1: Schematic representation of an EIT system.

Necessity of a forward mathematical model of the imaging process arises with the need for the solution of the inverse problem of EII method. Since the direct inverse solution of the imaging problem is impossible, solution of the inverse problem must use the forward problem formulation alongside a parameter estimation method, which in our case is a genetic algorithm. Thus, the solution of the forward problem plays a crucial role in imaging process.

Since an analytic solution of the governing equations for EII is often impossible excluding some special cases, numerical methods must be used for the mathematical modeling of the imaging process. The most popular numerical method for solving electrostatic problems is the finite element method. Because the image reconstruction is a minimization problem, an objective function is required to simulate voltage values on the electrodes for conductivity distributions using the mathematical model of the process and compare the simulated results with the actual data from the phantom, calculating the error values. The objective function includes the finite element model of the imaging process and an error function with a least-squares approximation. The genetic algorithm minimizes the objective function to reconstruct the conductivity distribution of the phantom body.

Electrodes can be used for injecting current into the body and measuring the voltage on the boundary electrodes at the same time. However, because of a reported phenomenon that causes the electrode-skin contact surface to have a resistive behavior, different electrodes used for excitation and measurement purposes in medical imaging applications (Hua et al., 1993). Because of this phenomenon, High frequency alternating currents are used for excitation in medical applications. However, in this thesis, two-phased flows are imaged, therefore allowing the same electrodes to be used for both excitation and measurement purposes. Because every electrode is excited, it is ideal to use low frequency currents (1-10 kHz range) with multiple excitation patterns. Using low frequency excitation currents cause the imaging equipment to be less complex and less expensive.

For an E-electrode imaging system, where E represents the electrode number of the imaging system, $(E - 1)$ number of independent excitations can be applied. Linear relationship between current and voltage values can be represented by an operator matrix with the dimensions of $(E - 1)$ x $(E - 1)$. The operator matrix is symmetric and has $E\,(E - 1) / 2$ degrees of freedom (The number of upper diagonal entries of

the operator matrix). These upper diagonal entries of the operator matrix represent the admittance values between the boundary electrodes. Thus, Unknown element conductivities of the system should be selected as equal or less than E (E – 1) / 2 to avoid making the problem under defined. Number of independent measurements the equation system requires to obtain an appropriate solution is at least E (E – 1) / 2. Therefore, if the number of the unknown element conductivities is chosen as the number of the degrees of freedom, all element conductivities are uniquely determined (Ovacık, 1998b).

2.2 Governing Field Equation

The electromagnetic field induced by an applied current density to the surface of a conductive body is governed by Maxwell's equations. Ampere's circuit law with Maxwell's correction is stated below.

$$\nabla \times H = J + \frac{\partial D}{\partial t} \qquad \textbf{(2.1)}$$

Where H represents the magnetic field density, J represents the current density, and D represents the electric flux density. Multiplying both sides with the divergence operator, conservation of current density statement is expressed by,

$$\nabla \cdot \left(J + \frac{\partial D}{\partial t} \right) = 0 \qquad \textbf{(2.2)}$$

Current density is obtained by using the Ohm's law.

$$J = \sigma \cdot \overline{E} \qquad \textbf{(2.3)}$$

Where σ is the electrical conductivity and $\bar{E}$ is the electrical field. Time derivative of the electric field displacement vector is stated as,

$$\frac{\partial D}{\partial t} = -j\omega\varepsilon \cdot \overline{E} \qquad \textbf{(2.4)}$$

Electric field intensity can be stated in terms of the gradient of the electric potential for a conservative field.

$$\overline{E} = -\nabla\phi \tag{2.5}$$

Where ϕ is the electrical potential. Substituting Equation (2.5) into Equations (2.3) and (2.4), and inserting these equations into Equation (2.2), the governing equation for the electrical properties in the domain Ω is reached.

$$\nabla \cdot (\sigma + j\omega\varepsilon)\nabla\phi = 0 \tag{2.6}$$

Boundary conditions for the imaging domain Ω,

$$\begin{aligned} -(\sigma + j\omega\varepsilon)\cdot\frac{\partial\phi}{\partial n} &= \pm J_E \qquad \text{on} \quad \partial\Omega_E \\ -(\sigma + j\omega\varepsilon)\cdot\frac{\partial\phi}{\partial n} &= 0 \qquad\quad \text{on} \quad \partial\Omega_H \end{aligned} \tag{2.7}$$

Where the term ($\sigma + j\omega\varepsilon$) represents the complex conductivity, consisting of the conductivity and the permittivity terms. σ is the electrical conductivity and ε is the electrical permittivity. n represents the outward normal vector on the boundary surface. Ω_E and Ω_H represent the surface of the electrodes and the homogeneous zones of the boundary condition, respectively. J_E denotes the current density vector of injected excitation.

An electrically excited medium consists of both conductive and dielectric properties. Conductive property of materials is influenced by term the σ, while dielectric property of materials is influenced by the term $\omega\varepsilon$. The ratio of $\omega\varepsilon$ / σ proportionally increases with the increasing excitation frequency. In low frequency excitations, the effect of dielectric constant becomes negligible compared to high conductivity values. Thus, the phase shift between current and voltage measurement is very small when real part of the complex conductivity dominates the imaginary part ($\omega\varepsilon << \sigma$). Considering this effect, dielectric property will be neglected ($\omega\varepsilon \approx 0$) in further analytical developments for low frequency applications in 1-10 kHz range. Thus, Equation (2.6) is reduced to Laplace's equation, which is the governing equation for low frequency electrical impedance imaging system (Ovacık, 1998b).

$$\nabla \cdot (\sigma\nabla\phi) = 0 \tag{2.8}$$

Laplace's equation becomes Poisson's equation when right side of the equation is nonzero. Equation (2.8) can be expanded into the form below.

$$\nabla \cdot (\sigma \nabla \phi) = \sigma \nabla^2 \phi + \nabla \sigma \cdot \nabla \phi = 0 \qquad \textbf{(2.9)}$$

Expanded form of Laplace's equation exhibits a convection-diffusion phenomenon. This well-known behavior is often seen in heat transfer, mass transfer and fluid flow problems. An analytical solution can only be obtained for some very special cases such as existence of symmetry in geometry or boundary conditions, using conformal mapping, series expansion methods or integral methods.

In expanded form of Equation (2.9), the first term represents diffusion and the second term represents convection using an analogy from mass transfer and fluid mechanics. Numerical solution of this equation is very difficult to obtain because of the numerical instabilities that occurs when the transport of ϕ is dominated by the convection term (Ovacık, 1998b).

The most popular methods for solving the electrostatic field equations are finite difference and finite element methods. In this thesis, finite element method is chosen to solve the governing field equation for the forward problem. The quality of a FEM approximation is often higher then FDM approximation, therefore causing less numerical errors. In addition, FEM has the ability to handle complex geometries and boundaries. However, numerical solution of the governing field equation is very difficult to solve using common finite element methods directly (Ovacık, 1989). To overcome this difficulty, Galerkin's weighted-residual method is applied to the finite element model of the system. Galerkin's method makes numerical solution of Laplace's equation relatively less complex. It is also easy to apply to Laplace's equation, providing a satisfactory outcome.

2.3 Finite Element Formulation

Galerkin's weighted-residual method includes an arbitrary and continuous weighting function W into the governing field equation (Equation (2.8)). The parameters of the approximation are determined such that the governing field equation is valid for every choice of the weighting function W. After multiplying Equation (2.8) with weighting function W, equation is integrated over the domain of the element-e, Ω_e.

$$\int_{V(\Omega_e)} W\nabla\cdot(\sigma\nabla\phi)dV = 0 \tag{2.10}$$

Using product rule of a gradient on Equation (2.10),

$$\nabla(W\sigma\nabla\phi) = W\nabla\cdot(\sigma\nabla\phi) + \sigma\nabla\phi\cdot\nabla W \tag{2.11}$$

Equation (2.11) is inserted into Equation (2.10).

$$\int_{V(\Omega_e)} W\nabla\cdot(\sigma\nabla\phi)dV = \int_{V(\Omega_e)} \nabla(W\sigma\nabla\phi)dV - \int_{V(\Omega_e)} \sigma\nabla\phi\cdot\nabla W\cdot dV \tag{2.12}$$

The second integral term in Equation (2.12) becomes a surface integral using Gauss's Divergence Theorem.

$$\int_{V(\Omega_e)} \nabla(W\sigma\nabla\phi)dV = \oint_{S(\Omega_e)} (W\sigma\nabla\phi)\cdot dA \tag{2.13}$$

Inserting Equation (2.13) into Equation (2.12),

$$\int_{V(\Omega_e)} W\nabla\cdot(\sigma\nabla\phi)dV = \oint_{S(\Omega_e)} (W\sigma\nabla\phi)\cdot dA - \int_{V(\Omega_e)} \sigma\nabla\phi\cdot\nabla W\cdot dV \tag{2.14}$$

Using Equation (2.10), the first integral term in Equation (2.14) equals to zero.

$$\oint_{S(\Omega_e)} (W\sigma\nabla\phi)\cdot dA - \int_{V(\Omega_e)} \sigma\nabla\phi\cdot\nabla W\cdot dV = 0 \tag{2.15}$$

The current flux on the element boundary,

$$q^e \equiv -\sigma\nabla\phi \tag{2.16}$$

Substituting Equation (2.16) into Equation (2.15),

$$\int_{V(\Omega_e)} \sigma\nabla\phi\cdot\nabla W\cdot dV = -\oint_{S(\Omega_e)} W\cdot q^e\cdot dA \tag{2.17}$$

Where q^e is the current flux vector in element-e. Expanding the first integral term in Equation (2.17),

$$\int_{V(\Omega_e)} \sigma\nabla\phi\cdot\nabla W\cdot dV =$$

$$\int_{V(\Omega_e)} \left(\frac{\partial W}{\partial x}\bar{i} + \frac{\partial W}{\partial y}\bar{j} + \frac{\partial W}{\partial z}\bar{k}\right)\cdot\left(\sigma\frac{\partial\phi}{\partial x}\bar{i} + \sigma\frac{\partial\phi}{\partial y}\bar{j} + \sigma\frac{\partial\phi}{\partial z}\bar{k}\right)\cdot dxdydz \tag{2.18}$$

Combining two terms together,

$$\int_{V(\Omega_e)} \sigma \nabla \phi \cdot \nabla W \cdot dV =$$

$$\int_{V(\Omega_e)} \left(\frac{\partial W}{\partial x} \sigma \frac{\partial \phi}{\partial x} + \frac{\partial W}{\partial y} \sigma \frac{\partial \phi}{\partial y} + \frac{\partial W}{\partial z} \sigma \frac{\partial \phi}{\partial z} \right) \cdot dxdydz \qquad \textbf{(2.19)}$$

In this thesis, two-dimensional finite element approximation is used, so the volume element has unity depth, dz=1, and the terms that includes the derivatives of z vanishes.

$$\int_{V(\Omega_e)} \sigma \nabla \phi \cdot \nabla W \cdot dV = \int_{V(\Omega_e)} \left(\frac{\partial W}{\partial x} \sigma \frac{\partial \phi}{\partial x} + \frac{\partial W}{\partial y} \sigma \frac{\partial \phi}{\partial y} \right) \cdot dxdy \qquad \textbf{(2.20)}$$

2.3.1 Quadrilateral elements

Using bilinear quadrilateral elements with four straight sides, the potential inside the element-e is represented in terms of the potentials on the nodes at the corners of the element with the relationship below (Jones et al., 1992). Node numbers of an element are shown in Figure 2.2.

$$\phi(x, y) = \sum_{i=1}^{4} N_i(x, y) \cdot V_i \qquad \textbf{(2.21)}$$

$$\nabla \phi(x, y) = \sum_{i=1}^{4} \nabla N_i(x, y) \cdot V_i \qquad \textbf{(2.22)}$$

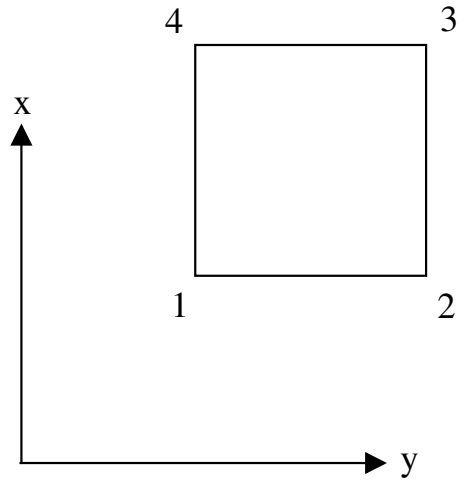


Figure 2.2: Node numbers of elements.

Where V's are voltages and N's are shape functions of the nodes. Shape functions are used to map the element from the physical x-y plane to a standard square in the

parametric ξ-η plane confined between $-1 \leq \xi \leq 1$ and $-1 \leq \eta \leq 1$ as shown in Figure 2.3. The first node is mapped to the point $\xi = -1$, $\eta = -1$, the second to the point $\xi = 1$, $\eta = -1$, the third to the point $\xi = 1$, $\eta = 1$, and the fourth to the point $\xi = -1$, $\eta = 1$. The mapping from the *x*-*y* plane to the ξ-η plane is mediated using Equations (2.23) and (2.24).

$$x(\xi,\eta) = \sum_{i=1}^{4} x_i N_i(\xi,\eta) \tag{2.23}$$

$$y(\xi,\eta) = \sum_{i=1}^{4} y_i N_i(\xi,\eta) \tag{2.24}$$

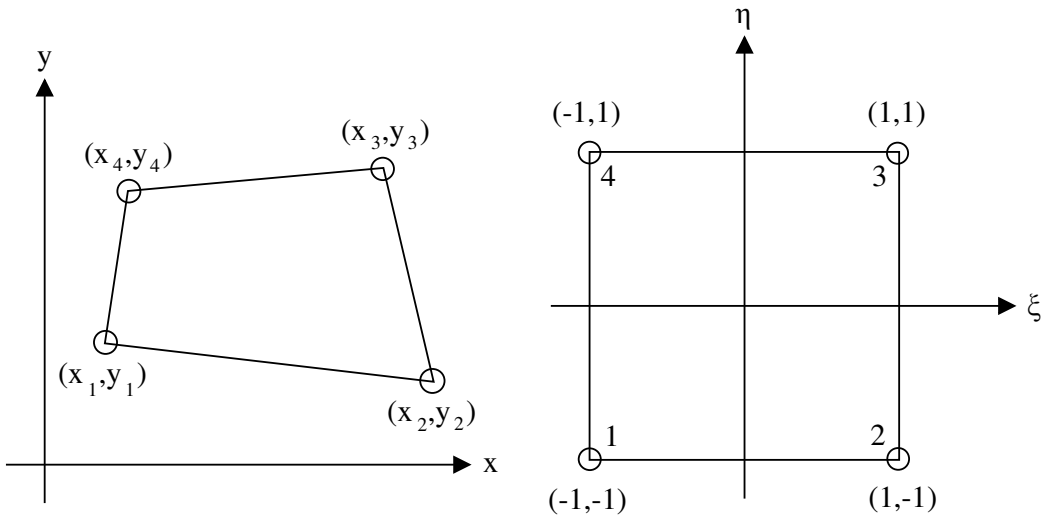


Figure 2.3: The mapping from the *x*-*y* plane to the ξ-η plane

The shape functions for bilinear quadrilateral elements,

$$N_1 = \frac{1}{4} \cdot (1-\xi) \cdot (1-\eta) \tag{2.25}$$

$$N_2 = \frac{1}{4} \cdot (1+\xi) \cdot (1-\eta) \tag{2.26}$$

$$N_3 = \frac{1}{4} \cdot (1+\xi) \cdot (1+\eta) \tag{2.27}$$

$$N_4 = \frac{1}{4} \cdot (1-\xi) \cdot (1+\eta) \tag{2.28}$$

A bilinear expansion form is utilized to transform coordinates between planes.

$$x(\xi,\eta) = \alpha_0 + \alpha_1\xi + \alpha_2\eta + \alpha_3\xi\eta \tag{2.29}$$

$$y(\xi,\eta) = \beta_0 + \beta_1\xi + \beta_2\eta + \beta_3\xi\eta \tag{2.30}$$

An infinite-small area is transformed as following,

$$dx \cdot dy = |J| \cdot d\xi \cdot d\eta \tag{2.31}$$

Using Equations (2.29) and (2.30), transformation Jacobean is written,

$$J = \begin{bmatrix} \frac{\partial x}{\partial \xi} & \frac{\partial x}{\partial \eta} \\ \frac{\partial y}{\partial \xi} & \frac{\partial y}{\partial \eta} \end{bmatrix} = \begin{bmatrix} \alpha_1 + \alpha_3\eta & \alpha_2 + \alpha_3\xi \\ \beta_1 + \beta_3\eta & \beta_2 + \beta_3\xi \end{bmatrix} \tag{2.32}$$

The equations for α's and β's is determined by writing Equations (2.29) and (2.30) for all nodes and solving the equation system. Writing Equations (2.29) and (2.30) for four nodes, two sets of four equations are solved to determine the mathematical statements for α's and β's for computation of Jacobean matrix to be used in the simulation algorithm.

$$\alpha_1 = \frac{1}{4} \cdot ((x_2 + x_3) - (x_1 + x_4)) \tag{2.33}$$

$$\alpha_2 = \frac{1}{4} \cdot ((x_3 + x_4) - (x_1 + x_2)) \tag{2.34}$$

$$\alpha_3 = \frac{1}{4} \cdot ((x_1 + x_3) - (x_2 + x_4)) \tag{2.35}$$

$$\beta_1 = \frac{1}{4} \cdot ((y_2 + y_3) - (y_1 + y_4)) \tag{2.36}$$

$$\beta_2 = \frac{1}{4} \cdot ((y_3 + y_4) - (y_1 + y_2)) \tag{2.37}$$

$$\beta_3 = \frac{1}{4} \cdot ((y_1 + y_3) - (y_2 + y_4)) \tag{2.38}$$

Using Equation (2.20) and (2.22) together,

$$\int_{V(\Omega_e)} \sigma \nabla\phi \cdot \nabla W \cdot dV =$$

$$\int_{V(\Omega_e)} \left(\sigma \frac{\partial W}{\partial x} \sum_{l=1}^{4} \left(\frac{\partial N_l}{\partial x} V_l \right) + \sigma \frac{\partial W}{\partial y} \sum_{l=1}^{4} \left(\frac{\partial N_l}{\partial y} V_l \right) \right) \cdot dxdydz \tag{2.39}$$

An admittance matrix Y is defined, which allows Equation (2.39) to be rewritten as,

$$\int_{V(\Omega_e)} \sigma \nabla \phi \cdot \nabla W \cdot dV = \sum_{l=1}^{4} Y_{ml} \cdot V_l \qquad \textbf{(2.40)}$$

Where Y_{ml} is defined by comparison to be,

$$Y_{ml} = \int_{V(\Omega_e)} \left(\sigma \frac{\partial W}{\partial x} \frac{\partial N_l}{\partial x} + \sigma \frac{\partial W}{\partial y} \frac{\partial N_l}{\partial y} \right) \cdot dxdy \qquad \textbf{(2.41)}$$

In the Rayleight-Ritz method, the weighting function W is determined to minimize the numerical error in Equation (2.41) (Jones et al., 1993). The weighting function W is selected as N_m.

$$Y_{ml} = \int_{V(\Omega_e)} \left(\sigma \frac{\partial N_m}{\partial x} \frac{\partial N_l}{\partial x} + \sigma \frac{\partial N_m}{\partial y} \frac{\partial N_l}{\partial y} \right) \cdot dxdy \qquad \textbf{(2.42)}$$

Relationship between voltage and current for element-e is expressed by,

$$\sum_{l=1}^{4} Y_{ml} \cdot V_l = I_m \qquad \textbf{(2.43)}$$

Net current I_m at the nodes into the element-e,

$$I_m \equiv - \oint_{S(\Omega_e)} N_m \cdot \bar{q}^e \cdot dA \qquad \textbf{(2.44)}$$

Y_{ml} is the (m,l)-th entry of the element stiffness matrix. Element-e also satisfies the Kirchhoff's law with voltages V_l of the nodes at the corner of the element and the net current I_m at the nodes into the element.

Equations (2.31) and (2.32) are substituted into Equation (2.42) to write admittance matrix elements in terms of variables ξ and η.

$$Y_{ij} = \sigma_e \int_{-1}^{1} \int_{-1}^{1} F_{ij}(\xi, \eta) \cdot d\xi d\eta \qquad \text{for} \quad \text{i = 1,...,4} \quad \text{and} \quad \text{j = 1,...,4} \qquad \textbf{(2.45)}$$

Where σ_e is the conductivity of element-e, which is regarded as constant inside the element. F_{ij} is expressed by,

$$F_{ij}(\xi, \eta) = \left[\begin{array}{l} \left(J_{11}^{-1} \frac{\partial N_i}{\partial \xi} + J_{12}^{-1} \frac{\partial N_i}{\partial \eta} \right) \left(J_{11}^{-1} \frac{\partial N_j}{\partial \xi} + J_{12}^{-1} \frac{\partial N_j}{\partial \eta} \right) + \ldots \\ + \left(J_{21}^{-1} \frac{\partial N_i}{\partial \xi} + J_{22}^{-1} \frac{\partial N_i}{\partial \eta} \right) \left(J_{21}^{-1} \frac{\partial N_j}{\partial \xi} + J_{22}^{-1} \frac{\partial N_j}{\partial \eta} \right) \end{array} \right] \qquad \textbf{(2.46)}$$

Double integration in Equation (2.45) is numerically computed using gauss quadrature formula with 4 x 4 points grid structure (Fish and Belytschko, 2007).

$$\int_{-1}^{1}\int_{-1}^{1} F_{ij}(\xi,\eta)\cdot d\xi d\eta = \sum_{m=1}^{4}\sum_{n=1}^{4} F_{ij}(\xi_m,\eta_n)\cdot w_m \cdot w_n \qquad \textbf{(2.47)}$$

Where ξ_m and η_n are gauss points, and w_m and w_n are their corresponding gauss weights. Gauss points and Gauss weights are shown in table 2.1 for 4 x 4 points grid. After computation of local admittance matrix entries for every element, the next step is to assemble the global admittance matrix using the local admittance matrices.

Table 2.1: Gauss points and their corresponding Gauss weights for 4 x 4 points grid.

Grid Points (m,n)	ξ and η	w
1	- 0.8611363116	0.3478548451
2	- 0.3399810435	0.6521451548
3	0.3399810435	0.6521451548
4	0.8611363116	0.3478548451

2.3.2 Construction of admittance matrix

Computed local element admittance matrix entries are assembled to form the global admittance matrix *Y*. Global admittance matrix has the dimensions of (*N*,*N*) and is used in solving global system of equations.

$$Y_{NxN}\cdot V_{NxP} = C_{NxP} \qquad \textbf{(2.48)}$$

Where *N* is the total number of nodes in the mesh structure and *P* is the number of current excitements into the system. *V* matrix is the voltage matrix that consists of the nodal voltages for each excitation and C matrix is the current matrix, which includes the nodal current values for each excitation.

There are six local admittance matrix entries for each element. These entries are added to the corresponding points of the global admittance matrix and the entries between same nodes are summed together, forming the global admittance matrix. Constructed global admittance matrix is a banded sparse matrix. Positions of six local admittance matrix entries of an element are illustrated on Figure 2.4.

Two rectangular shaped mesh structures are used in FEM model, one with 9x9 grid consisting of total 81 elements and the other with 17x17 grid consisting of total 289 elements. 9x9 grid model includes 16 electrodes and 17x17 model includes 32

electrodes. In Figure 2.5, two mesh structures used in this thesis are shown with node numbers at the corners. Elements and nodes are numbered starting from the bottom-left corner of the mesh grid, ending in the top-right corner. Elements next to the boundary surface are smaller than the other elements, therefore modeling the area near the boundary surface with higher accuracy. Because the gradients are higher at the boundaries than the middle sections of the grid, FEM approximation errors are higher on the boundary area, where high precision modeling is very important.

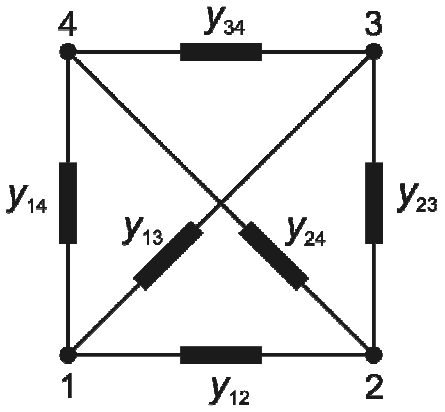


Figure 2.4: Positions of local admittance matrix entries of an element.

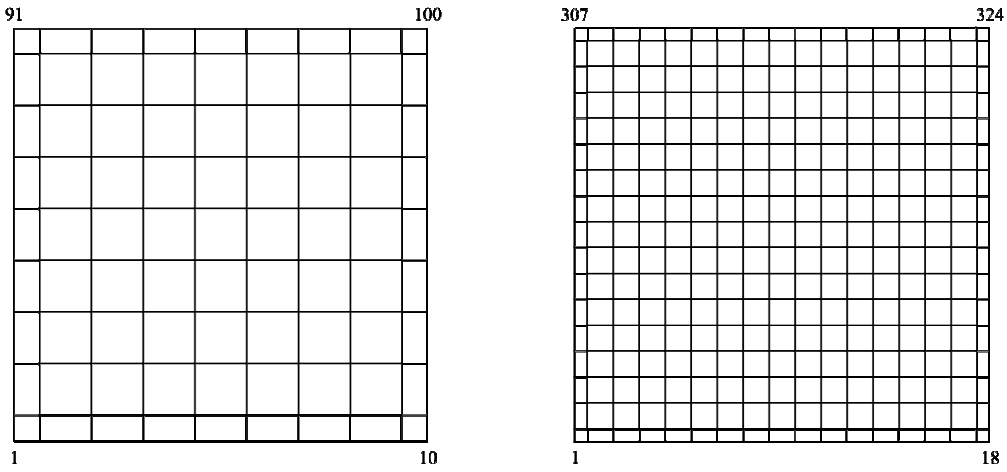


Figure 2.5: Mesh structures and node numbering used in the FEM model: **(a)** Mesh structure of the 9x9 grid model. **(b)** Mesh structure of the 17x17 grid model.

2.3.3 Modeling of electrodes

Electrodes can be modeled in two ways. First is the rod electrode approximation, where the electrodes are modeled like single points. In the rod electrode model, current is injected into the system from a single node in FEM model. The Rod electrode model approximation is reported to cause modeling errors in FEM model

because the injected currents are far from being uniformly distributed. Thus, the current gradient near the electrodes becomes extremely high and the sensitivity of the FEM model dramatically decreases. For a more efficient modeling of the electrodes, plate electrode model is introduced.

In plate electrode model, electrodes modeled as an area where the current is uniformly injected. Size of the electrodes must be enough to cover two nodes at minimum. To achieve this uniform current injection, voltage values of the nodes that have a contact with the same electrode are forced to be equal by adding electrode admittance entries between the nodes in the global admittance matrix.

Electrode admittance entries can be between the magnitude of 10^6 and 10^{10} depending on the electrode material. Using the plate electrode model, current flow at the electrodes are more smoothly distributed than in the rod electrode model, thus decreasing the modeling errors and increasing the sensitivity at the area near the electrodes (Ovacık et al., 1998a). In this thesis, electrode admittance elements are forced to be 10^{10}. Electrode numbers and positions of the FEM model structures are shown in Figure 2.6.

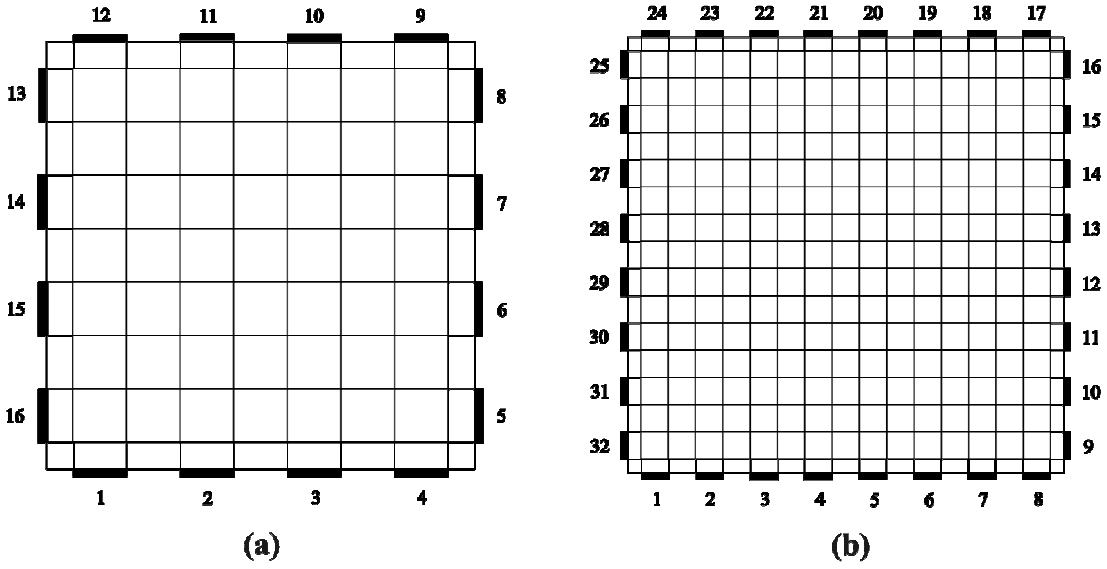


Figure 2.6: Electrode numbers and positions for the FEM model: **(a)** 16-electrode model. **(b)** 32-electrode model.

2.3.4 Boundary conditions

In impedance imaging measurement process, one of the electrodes is selected as the reference electrode to be used as a reference point in relative voltage measurement process. Therefore, the rank of the global admittance matrices rank decreases to (*N* -

1), where N is total node number of the model. In order to have a fully defined linear equation system, (E - 1) number of measurements are necessary, where E is the number of the electrodes. Thus, (E - 1) number of excitations is required to solve the linear equation system.

First boundary condition to apply to model is the reference electrode. Because reference electrode is grounded, voltage on the reference electrode must be zero. In order to apply the reference electrode boundary condition to the model, following tasks are performed. Keeping in mind that R is the number of the node that the reference electrode stands on; first, all the elements on the R-th column and R-th row of the global admittance matrix Y is set to zero. Second, diagonal element of the global admittance matrix Y_{RxR} is set to unity. Finally, all the elements on the R-th row of the current matrix C is set to zero (Pozrikidis, 2005). The other boundary conditions are the injected current values from the electrodes, which is applied by setting corresponding entries of the current matrix C.

2.3.5 Linear system solution

Linear equation system, which is shown in Equation (2.46), is a symmetric and sparse system. Admittance matrix Y is a banded matrix that has the bandwidth of $BW=m+2$, where m is the number of the horizontal index of the nodes in the finite element mesh structure (Ovacık, 1998b). Solution of the linear system is the determination of nodal voltages in the mesh structure, using the global admittance matrix and the current matrix. Equation (2.46) can be represented as following,

$$V_{NxP} = Y_{NxN}^{-1} \cdot C_{NxP} \qquad \textbf{(2.49)}$$

A basic method for solving the linear system is computing the inverse of the admittance matrix and then multiplying with the current matrix. However, this method is not efficient in terms of computing resources. For linear systems with relatively large admittance matrices, inversion procedure requires extremely high computing resources. Because the objective function is executed frequently, performance of the image reconstruction algorithms depends on its forward solution algorithm. Majority of the computing time needed for the simulation algorithm is consumed by the linear system solver; thus making speed of linear system solver algorithm crucial for the overall image reconstruction performance. MATLAB's

built-in ‘linsolve’ algorithm, which uses LU factorization method, is used as the linear solver operator in this thesis.

2.4 Test Phantom Simulation Algorithm

Simulation of the measurement system starts with reading the data files, including excited current values, conductivity distribution of the body and the mesh structure data. Then, the local admittance elements of FEM model are computed and assembled together to form the global admittance matrix. Next step is to solve the equation system using linear solver operator. Artificial noise is added to the computed voltage values if required. Last step of the algorithm is to write the voltage data for every excitement to the output file. In Figure 2.7, flowchart of the simulation algorithm is shown.

Simulation algorithm is coded using MATLAB language; approximately, it takes 0.001 second to run the forward model with 9x9 mesh grid and 0.01 second to run the forward model with 17x17 mesh grid using a computer with a 2.0 GHz dual-core CPU. Because the memory consumption of the algorithm is under 1 MB, performance of the algorithm is not affected by the amount of RAM the system has.

The algorithm developed in this thesis uses Walsh functions as injected current patterns. Because only two levels of current values (-1 and +1) are used, Walsh functions simplify the design of the data acquisition hardware (See Figure B.1 and Figure B.2 in Appendix B for Walsh patterns used for excitation of 16-electrode and 32-electrode model phantoms respectively.). Compared to other excitation patterns, Walsh function current injection is reported to provide the most efficient excitation in terms of the useful information collected about the interior conductivity distribution of the body (Woo et al., 1990). Walsh functions also have the computational simplicity advantage. Number of required measurements to fully define the solution of the equation system is (E - 1), where E is the total number of the electrodes on the boundary surface. Therefore, one Walsh function pattern (more specifically, the pattern that continuously equals to unity.) is discarded and the remaining patterns are used as the injected excitation current values for all of the electrodes.

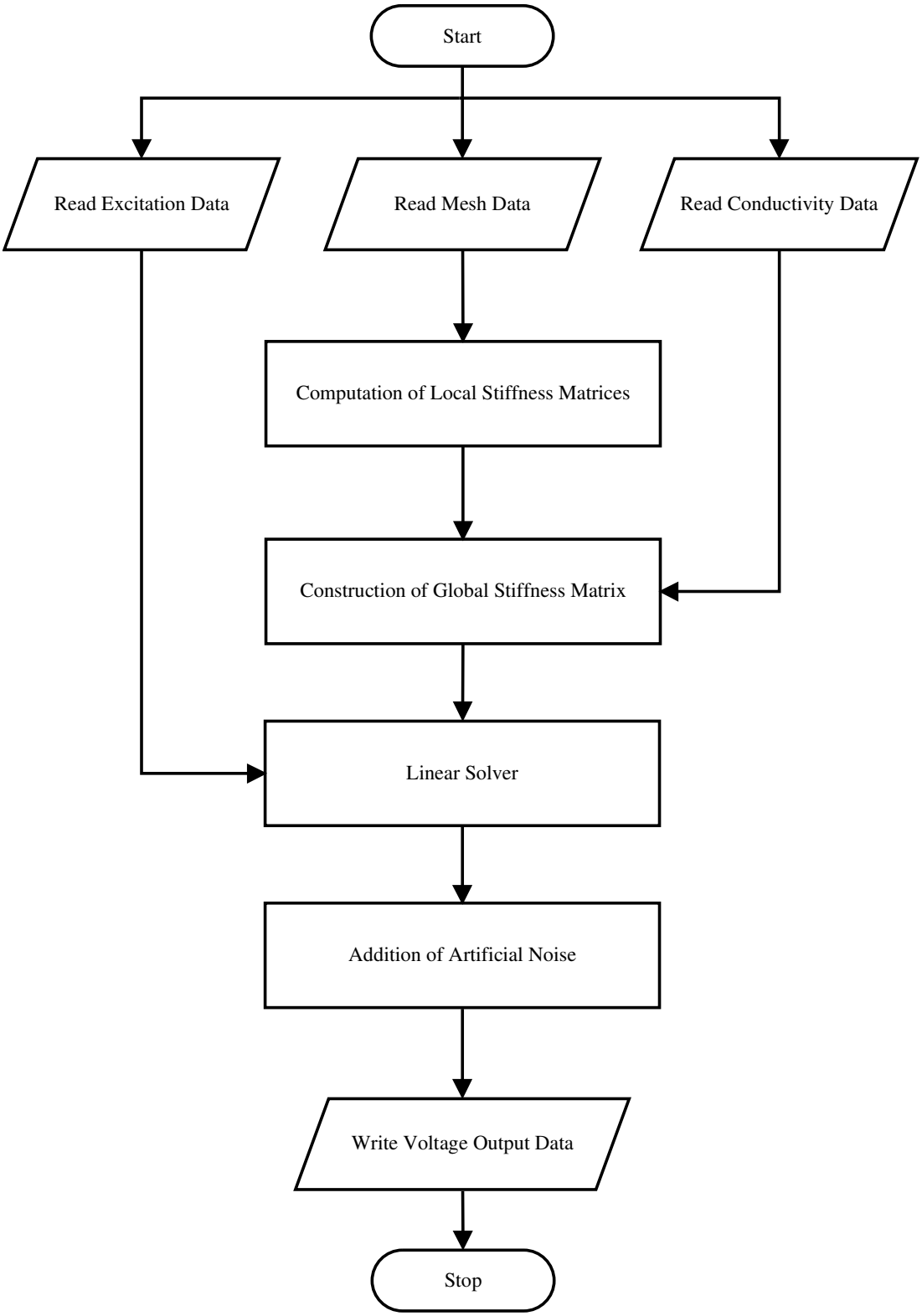


Figure 2.7: Flowchart of the test phantom simulation algorithm to generate test data.

3. SOLUTION OF INVERSE PROBLEM

3.1 Image Reconstruction Problem

Inverse problem of the electrical impedance imaging consists of estimation of electrical conductivity distribution of the body that is being imaged. Using the measured voltage amplitudes at the electrodes and the current values that are injected from electrodes, conductivity distribution along the body is calculated. Flowchart of the solution of the inverse problem is shown in Figure 3.1. The inverse problem of the EII method can be considered as an image reconstruction problem because the result of the problem is the image of the electrical conductivity distribution of the body. As discussed earlier, the inverse solution of the EII forward imaging problem is impossible to obtain analytically, unless the geometry of the body has some special properties. As a result, direct solution of the inverse problem is impossible for most cases, which makes the implementation of an additional inverse solution method necessary. However, due to the non-linear and ill-conditioned nature of the EII problem, solution of the inverse problem is very difficult to obtain.

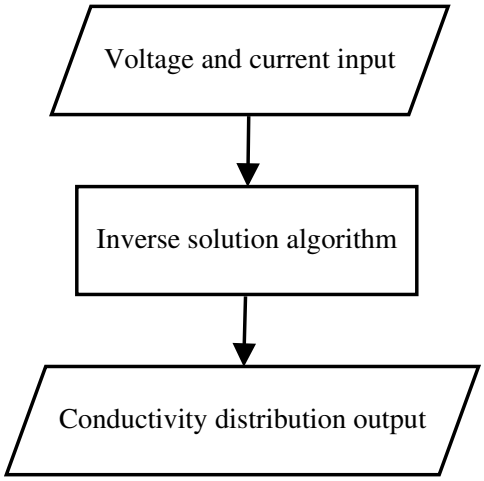


Figure 3.1: Flowchart of the solution of the inverse problem in EII.

Optimization, search and parameter estimation methods can be used to solve the image reconstruction problem of EII method. In this thesis, a genetic algorithm is developed to solve the image reconstruction problem. The inverse problem of EII has multiple local minimum solutions and only one global minimum solution, which is

the true conductivity distribution of the body. This problematic nature puts an extra emphasis on inverse solution method that is being used. Some parameter estimation methods that have a tendency to fall into the local minima instead of the global minimum solution are not ideal for the image reconstruction problem because they are likely to obtain premature results by converging into the local minima. Best results are achieved by using inverse solution methods that possesses robust characteristics.

Before genetic algorithms are started to be used as the image reconstruction algorithms, the most widespread method to solve this problem was Newton-Raphson minimization. Even today, Newton-Raphson method is very popular in the image reconstruction problem of EIT method due to its practical yet efficient nature. Newton-Raphson algorithm is fast and efficient when used with a good starting point. Nonetheless, it fails to achieve convergence when used with an inappropriate initial guess. Therefore, initial guess is extremely important in Newton-Raphson algorithm for a successful convergence. However, as the inverse problem of the electrical impedance imaging method is ill-conditioned and the search space has multiple local minimum solutions, selection of the initial guess is extremely difficult. Any initial guess that is chosen too distant to global minimum point may cause Newton-Raphson algorithm to converge into one of the local minimum solutions, resulting in a prematurely incorrect conductivity distribution. This phenomenon prevents Newton–Raphson method from being the ideal method for the solution of the image reconstruction problem in EII. The search for finding much efficient methods to solve the image reconstruction problem of EII continues. Evolutionary computation methods, which are being increasingly applied to similar problems in recent years, are between the methods that are being tested for this task. Among these evolutionary computing methods, the most popular one is the genetic algorithm method.

Genetic algorithms are starting to be applied to many optimization and search applications, including the image reconstruction problem in electrical impedance imaging, in recent years. Several studies (including Cheng et al. (1996), Meng et al. (1999), Olmi et al. (2000), Kim et al. (2002), Kim et al. (2006), Rolnik and Seleghim (2006)) claim to obtain optimistic results using GAs with the EII method. These studies point out that the stochastic nature of the genetic algorithms helps to

overcome the premature convergence problem by preventing the algorithm from converging into the local minima. Stochastic optimization methods, including the GAs, are not dependent to initial guesses, which makes these methods ideal for the problems with multiple local minima. According to the studies, the only drawback of GAs is their high computing time requirements, thus making them impossible to use them in real-time imaging applications.

3.1.1 Ill-conditioning

Solution of an inverse problem is ill-conditioned if the system has low sensitivity to its parameters. Conductivities of every pixel in the body have their effects on the boundary response. Because the measurements are obtained from the boundary in EII method, the conductivity values of the pixels in the interior region of the body have a reduced effect on the measured data. The image reconstruction problem of EII method is extremely ill-conditioned because due to the nonlinear relationship between the measured data and the unknown conducivity parameters. In ill-conditioned problems, noise on the measurements may be critical for the stability of the solution.

3.1.2 Search space

In optimization and search problems, the space of all feasible solutions is called search space. The search space consists of the set of all solutions that the desired solution resides. Every point in the search space represents one possible solution of the problem. As the search space becomes larger, the number of candidate solutions increases. Thus, a larger state space results in a more difficult search problem (Sivanandam and Deepa, 2008).

Search spaces of image reconstruction problems are generally quite large because of the number of variables. Every pixel of the reconstructed image is represented with an independent variable, which increases the degree of freedom of the system dramatically. Even for binary image reconstruction problems, search spaces are massively large. As mentioned in the previous chapter, two finite element models are used in this thesis, one with a 9x9 mesh grid and the other with a 17x17 mesh grid. For 9x9 mesh grid, the reconstructed image has the dimensions of 9x9 pixels, thus, the total number of pixels is 81. Therefore, the search space of the image reconstruction problem is 2^{81}, which is equal to $2{,}418.10^{24}$, an extraordinary large

number for a small grid of 9x9. For 17x17 mesh grid, the reconstructed image has the dimensions of 17x17 pixels, thus making the total number of pixels 289. For 17x17 pixels, the search space of the image reconstruction problem becomes 2^{289}, which is equal to $9{,}946.10^{86}$, a gigantically large number. From these numbers, we can see that the search space of 17x17 pixels is 4.10^{62} times larger than the search space of 9x9 grid; an extraordinary big difference between search spaces of 9x9 and 17x17 mesh grid structures. Therefore, it shows the complexity difference between two grid structures, image reconstruction problem becomes dramatically more difficult as the pixel number increases. Taking into account that the estimated number of atoms in the observable universe is in the magnitude of 10^{80}, we can see that the search space of 17x17 pixels is ten million times of the estimated number of atoms in the observable universe. This gives an impression of how big the search space is for the image reconstruction problem for 17x17 pixels. Thus, solving the inverse problem of EII is extremely difficult and it becomes dramatically more difficult as the number of pixels increases. Thinking that the forward simulation takes 0.01 second of computing time to calculate the voltage response values, using the blind search method, which is a search method that every possible solution is evaluated until the correct result is found, it would take nearly 10^{77} years to find the true conductivity distribution on a modern computer, which is practically impossible. Therefore, to find the true result in much lesser evaluations and practical computing times, a stronger inverse solution algorithm is required.

3.2 Introduction to Evolutionary Computation

Among the nature, “survival of the fittest” principle can be observed. According to this principle, a number of organisms that co-exist in the same environment compete over natural resources. The organisms that gather more resources than the others have an increased chance of reproducing themselves for the next generations. Ability of surviving and reproducing of an organism can be described as the fitness of the organism. The organisms that can adapt to their environment gain an edge over their competitors, therefore the fitter individuals have a higher chance to reproduce themselves for next generations. Individuals in a population are selected for their fitness to form the future generations, evolving the population to a fitter state in the process. Thus, future generations carry the characteristics of today’s fittest

individuals. This process continuously shapes tomorrows generations towards a better adaptation to the environment. These principles form the modern idea of biological evolution by natural selection.

Evolutionary computation techniques summarize these evolutionary principles into the algorithms that search for optimal solutions to a problem. In a search algorithm, the task is to find the best solution between a number of possible solutions to a problem in a practical amount of time. For a search space with a limited number of possible solutions, all the solutions can be checked out in a reasonable amount of time to find the optimal one. This comprehensive search (which is called blind search), however, becomes impractical as the search space gets larger. Traditional search algorithms sample the search space one solution at a time to find the optimal solution. The key aspect that distinguishes an evolutionary search algorithm from traditional search methods is its population-based nature. By the adaptation of successive generations for a number of individuals, evolutionary algorithms are efficient direct search tools. The evolutionary computation concept can be applied to the problems where heuristic solutions are not possible or leading to unsatisfactory results. Therefore, evolutionary algorithms are becoming increasingly popular, particularly for solving practical optimization and search problems. Genetic Algorithms, genetic programming, evolutionary strategies and evolutionary programming methods are among the most popular EC techniques (Sivanandam and Deepa, 2008).

3.2.1 Historical background

Several computer scientists studied evolutionary systems independently in the 1950s and the 1960s, with the idea that the evolutionary principles can be used as an optimization tool for the engineering problems. The idea of evolutionary computation was to evolve a population of candidate solutions to a given problem, using operators inspired by genetics and natural selection.

Evolution strategies firstly introduced by Rechenberg in 1960s as a method to optimize real-valued parameters for devices such as airfoils and the idea was further developed by Schwefel in 1970s. Although recently the field of evolution strategies have begun to interact with the field of genetic algorithms, evolution strategies has remained an active area of research. Evolutionary programming is developed by

Fogel, Owens and Walsh in 1966. Evolutionary programming is a technique in which the candidate solutions are represented as finite-state machines, evolving by mutating randomly their state–transition diagrams and selecting the fittest of the solutions. Evolution strategies, evolutionary programming, and genetic algorithms together form the backbone of the field of evolutionary computation (Mitchell, 1999).

Genetic algorithms were invented by John Holland in the 1960s and were developed by Holland and his students at the University of Michigan in the 1960s and the 1970s. Unlike evolution strategies and evolutionary programming, Holland's original idea was to study the phenomenon of adaptation as it happens in nature and to develop algorithms to import the mechanisms of natural adaptation into computer systems, rather than to design algorithms to solve optimization problems.

Holland's GA was a method for evolving from one population of chromosomes to a new population by using a kind of natural selection together with the genetics-inspired operators of crossover and mutation. Each chromosome consisted of genes, which was represented by bits in the algorithm, each representing a property. The selection operator was used to choose the chromosomes that will be allowed to reproduce, specifically, fitter chromosomes reproduced more than the less fit ones. Crossover operator combined two chromosomes in an analogy to biological recombination between two single-chromosome organisms. Mutation operator randomly changed the contents of genes on the chromosomes (Mitchell, 1999).

Holland's invention of a population-based algorithm with selection, crossover and mutation was a major innovation. Compared to Rechenberg's evolution strategies, which used a population of only two individuals, one parent and one offspring derived from the parent by being subject to mutation, Holand’s GA was a more realistic implementation of biological evolution to the world of computation with its solid theoretical foundation. Today, the boundaries between genetic algorithms, evolution strategies, evolutionary programming, and other evolutionary approaches have broken down to some range. Genetic algorithm term is often used for different evolutionary algorithms very far from Holland’s original concept.

3.2.2 Advantages of evolutionary computation

Evolutionary computation techniques offer practical advantages to the optimization problems. A key advantage of evolutionary computation is that it has a simple

concept. Quite satisfactory results can be achieved with relatively simple algorithms. In particular, the algorithm does not require any gradient information to operate. Because no gradient information is required, for mathematically complex problems, it is relatively easy to apply evolutionary computing to the problem than the traditional optimization methods. Search problems over discontinuous domains, where no gradient information is available, can also be solved with evolutionary computation techniques. Because gradient operator amplifies noise, evolutionary computation methods are more successful with the problems where noise presence is high.

Another advantage of the evolutionary computation is that it can be applied to a very wide range of problems. Any problem that can be formulated as a function optimization problem is solvable using the evolutionary computation methods. Evolutionary algorithms can be combined with traditional optimization techniques. It may be by the use of a conjugate-gradient minimization after primary search with an evolutionary algorithm. It may also involve simultaneous application of evolutionary algorithm with gradient-based search methods.

Because evolution is a parallel process, evolutionary algorithms can benefit very much from parallel computing techniques. As distributed processing computers become popular, application of evolutionary computation to highly complex problems are being possible. In a typical evolutionary computation algorithm, the individual solutions are evaluated independently of the competing solutions. To decrease the computing time required to solve the problem, evaluation of each solution can be handled by a single processor. The computing time required for an evolutionary application can be nearly inversely proportional with the number of processors used in parallel.

Traditional optimization methods are generally not robust to dynamic changes in the environment. On the other hand, evolutionary computation can be used to adapt to changing situations. Because of their nature, evolutionary computation techniques exhibit robust properties and they can easily adapt to dynamic changes of the parameters. As the population of candidate solutions continuously evolves to adapt the environment, robustness can be achieved and it is not necessary to reinitialize the algorithm at any stage for any change in the circumstances. Robustness of the

evolutionary computation methods is a key advantage compared to the traditional optimization methods (Sivanandam and Deepa, 2008).

3.2.3 Genetic algorithms

In nature, every individual in a population competes with each other for resources like food, shelter and reproduction. The individuals that are better adapted to their environments have more chance of surviving. Individuals that survive longer have a higher chance to attract a mate for reproduction than the less surviving ones, producing a relatively large number of offspring. Through the recombination of the genes, child individuals carry both parents characteristics. A child individual tends to have good characteristics than its ancestors, because of the increased chance that the offspring will carry the combination of the good genes of both parents. After generations, species evolve spontaneously to become more and more adapted to their environment. In 1975, Holland described how to apply the principles of natural evolution to optimization problems and built the first GA. After further development until today, GAs became a powerful tool for solving search and optimization problems (Holland, 1975).

GAs, which are based on the principles of genetics and evolution, possess a variety of important features. First, GAs are stochastic algorithms. Randomness of a GA plays an essential role. Both selection and reproduction needs random procedures. Its stochastic properties are among the most important features of GAs, preventing the algorithm to stall into local minima. Another very important feature of the GAs is that, a population of candidate solutions is evaluated instead of a solution. Evaluating more than a single solution in every iteration offers many advantages. Recombining different candidate solutions helps achieving better results. Population-based methods are superior to single-point methods in terms of robustness and they are also very applicable for parallelization. The robustness of the algorithm is also an essential property for the algorithms success. Robustness refers to the ability to perform consistently well on changing conditions for a large range of problem types. Robustness of the algorithm is the outcome of the stochastic and the population-based properties of GAs. Application range of the genetic algorithms is another important feature. Genetic algorithms can be applied to solve any problem that can be represented with a fitness function. All these features make GAs very powerful search and optimization tools. However, it is also important to mention the

limitations of GAs. Like the most stochastic methods, GAs does not guarantee to find the global optimum solution to a problem at every time the algorithm is executed, showing unpredictable characteristics.

3.2.3.1 Comparison with other optimization methods

Most significant difference between the genetic algorithms and the conventional optimization methods is that, the GAs are stochastic methods while the conventional optimization methods are generally deterministic. A stochastic method can be disadvantageous in some situations but for ill-conditioned problems with multiple local minimum points, they become advantageous because they are less likely to fall into local minimums. Most of the conventional optimization methods require a good initial guess to be used as a starting point for convergence. On the other hand, GAs do not require any initial guesses and because of their stochastic nature, GA can achieve convergence by starting from any point in the search space.

Another difference is that, GAs use fitness function to evaluate the candidate solutions, while the conventional optimization algorithms use derivative information. This is a major advantage of the GAs, because they can be applied to both continuous and discrete problems while the conventional methods suffer difficulties in adapting to discrete problems. Because of this property, GAs can solve any problem that is stated with an objective function. Another very important difference is that, the GAs operate on a whole population of points while the conventional methods search from only a single-point. This population-based structure of the GAs is one of their most significant advantages for achieving robustness. It improves the chance of reaching the global optimum solution by helping to avoid the local minimum points. GAs are more suited to parallel computing than the conventional methods. Genetic algorithms also operate better on the problems with large search spaces. Most important disadvantage of the genetic algorithms is their relatively high computing time requirements. However, as the parallel computing systems become widespread, this drawback of GAs becomes less important. Another drawback of the GAs is that, the determination of parameters is a difficult process and the success of the algorithm strongly depends on its parameters.

3.2.3.2 General structure of a genetic algorithm

The most important stages of a genetic algorithm are objective function, selection, crossover and mutation stages. A typical GA starts with the creation of initial population, which is followed by the evaluation of fitness function for all individuals of the initial population. Results of the fitness function are checked to see if the criteria for convergence are met to end the algorithm. Individuals are selected for reproduction in the selection stage according to their fitness values. In crossover stage, selected individuals are recombined to form the population of the next generation. Mutation operator randomly changes bits of individuals by a small percentage, hoping to achieve improvement. Next step is to evaluate the fitness function for all the individuals in the child generation. This loop continues until the required convergence criteria are met. Flowchart of a common genetic algorithm is shown in Figure 3.2. Encoding is also an important aspect of a genetic algorithm.

Encoding is the process of representing the individual genes. Encoding of the genes can be done in binary, octal, hexadecimal or real numbers, depending on the problem. Individuals of the population are possible solutions of the problem. Individuals are represented by the strings of bits that carry the properties of the corresponding solutions. These properties can be values or characteristics. Genes are encoded in the way that every possible individual represents one candidate solution in the search space and every candidate solution in the search space must be represented by a possible individual. Encoding stage of the GA directly alters the complexity of the process. For example, encoding a parameter estimation problem with binary parameters, simplifies crossover and mutation processes while increasing number of variables in the problem. Initial population of a GA is created randomly.

GAs, unlike conventional optimization methods, do not depend on starting point for convergence. Convergence can be achieved by starting from any point in the search space. However, having an initial population with a rich diversity increases the convergence speeds. As a result, all of the genetic algorithms start with random initial population. Fitness function is evaluated for all individuals of the population in every iteration. Fitness values of the individuals are checked for the convergence criteria to stop the algorithm. There can be multiple conditions to end the program and these examinations must be done in every iteration. If any of these criteria is met, the fittest individual is selected as the result and the algorithm ends. If not, the fitness

values of the individuals are stored to be recalled in the selection routine and the algorithm advances into the breeding process.

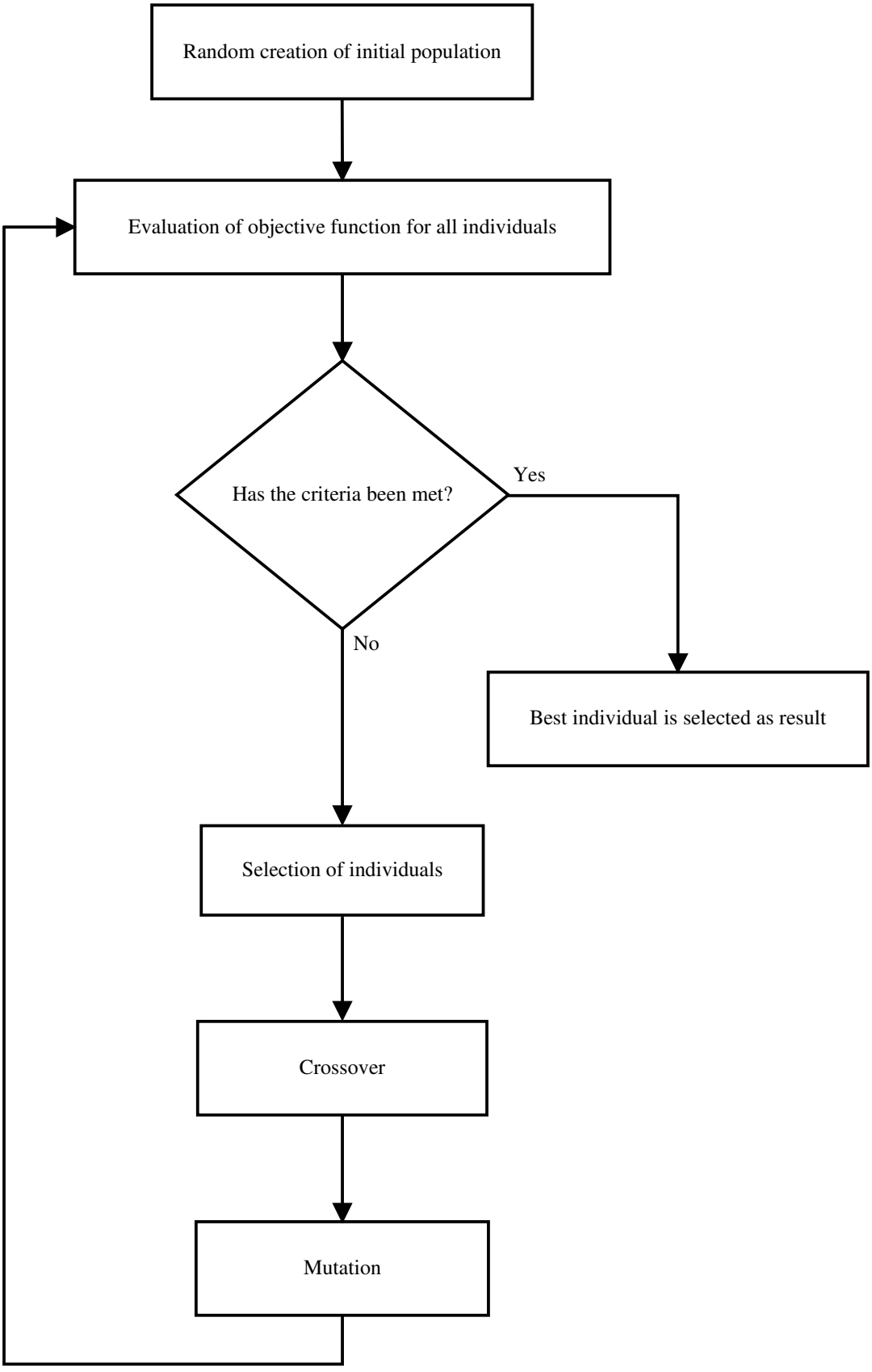


Figure 3.2: Flowchart of a common genetic algorithm.

Breeding process, which consists of selection and recombination operations, is the core of all genetic algorithms. In selection operation, individuals are selected for reproducing according to their fitness values. Selection operator is an extremely important operator for GAs and it must include randomness for a successful operation. The individuals with relatively higher fitness values must have a higher chance of selection than the individuals with lower fitness values. Even the least fit individual of the population might have a chance of being selected and this behavior adds stochastic properties to the genetic algorithms. If there is no randomness in the selection operation, the GA loses most of its stochastic properties, which often results in failure of converging to the true solution of the problem.

There are numerous selection algorithms for GAs to choose from. Most important ones are tournament selection, fitness proportionate selection and ranked selection. In tournament selection, a predetermined number of individuals are randomly picked from the population and the one with the highest fitness value is selected for reproducing. Another selection method is fitness proportionate selection, in which the individual's chances to be selected are in proportionate to their fitness values. Fitness values can also be normalized for a more balanced selection operation. In ranked selection, all the individuals are sorted in respect to their fitness values and they are given chances for selection in proportionate to their ranks. With rank selection method, selection pressure can be applied to the individuals according to their relative fitness values.

Selection operation is followed by recombination, which is called the crossover operation in genetic terminology. Crossover operator is the recombination of the selected individuals into the individuals of next generation. Population of the next generation is formed in crossover operation. The simplest crossover method is one−point crossover, where the strings of bits are cut from a randomly chosen point and remaining parts are exchanged between the individuals. However, better results can be achieved with more advanced crossover methods like N-point crossover and uniform crossover methods.

After the population of next generation is formed by breeding process, the individuals are subjected to mutation operation. Mutation is the random changes applied to individuals with a predetermined probability. However, the mutation probability must have a very small value, typically below one percent. Setting the

mutation probability too high decreases the algorithms convergence speeds, even at some point, preventing the convergence and causing the algorithm to stall. After the mutation stage, objective function is evaluated for all the individuals of newly created population and this process continues for every generation until the required criteria to stop are met.

3.2.3.3 Search strategies

Main target of a GA is to find the best possible solution to a problem by achieving convergence. However, there are numerous parameters in a GA that influence the performance and the results of the algorithm directly. Therefore, when designing a GA for a specific problem, having a search strategy is crucial for convergence.

There are important factors for GAs to consider when adapting a search strategy. Most important of these are convergence speed and diversity of the population. Convergence speed is how fast the algorithm converges to the best solution over successive iterations. Diversity is defined as the distance between the individual solutions in a population. Diversity increases with the variety of the individual solutions in a population. By increasing the selection pressure for fitter individuals, better convergence speeds can be achieved. However, increasing the selection pressure of better individuals causes those individuals to be dominant in the populations of the future generations; thus, it dramatically decreases the diversity of the population. If the fittest individuals of the early populations become dominant, there is a risk of premature convergence, which leads to converging into wrong solutions by falling into a local minimum point. Therefore, diversity among the population is very important in terms of stability and convergence of the algorithm. Decreasing the selection pressure increases the diversity, in the cost of convergence speed. However, too much diversity may even lead to a point where no convergence can be achieved at all. Thus, the right strategy is to set the selection pressure to an optimal point, where convergence speed and diversity of the population stays in balance. With the right strategy, reasonably good convergence speeds can be achieved, maintaining enough diversity among population at the same time.

Changing parameters in the genetic algorithm adaptively is a good strategy. Using adaptive parameters can be a solution to convergence speed-diversity dilemma. The selection pressure parameter adaptively changing with the diversity of the

population, the algorithm performs better on changing situations by keeping the diversity at reasonable levels while achieving optimal convergence speeds.

Another search strategy is to employ a multi-stage genetic algorithm. Every stage may serve to a different purpose, each having different parameters and even different operators. It is a good strategy to use this technique on problems with large search spaces. In problems where GAs become inefficient, they can be combined with another optimization method in hybrid scheme. There are studies that report to be successful by using hybrid algorithms including GAs and conventional optimization methods (Hsiao, 2001).

Applying elitist selection can be a good strategy to keep the best solution of the population safe. In elitist selection, a small number of the fittest individuals survive through the next generation. A good strategy would be to employ elitist selection to prevent the algorithm from divergence by losing the best individual of the population. Elitist selected members should also be protected from mutation operators.

3.3 Genetic Algorithm for Image Reconstruction Problem

Genetic algorithm method is very applicable to image reconstruction problem because of its many properties. Among these properties, the most significant one is the genetic algorithms ability to operate relatively better on ill-conditioned problems with multiple local minima due to its stochastic nature. Another important property is that the GAs are relatively successful on problems with large search spaces than the conventional optimization methods. Lastly, because no derivative information is needed, genetic algorithms work relatively well with noisy data. Thanks to these properties, GAs are starting to gain importance in electrical impedance imaging field. However, all the studies combining genetic algorithms and EII method are in development stage; no commercial impedance imaging system using GA is available as today. This thesis covers the application of GA method to electrical impedance imaging for reconstruction of binary conductivity distributions. Due to its high performance when working with matrices and its fast built-in linear equation solver, the genetic algorithm for the image reconstruction problem is developed using MATLAB 7.7 (MATLAB R2008b) programming environment. MATLAB

programming language is very suitable to technical applications and its performance with vector and matrix operations is quite high.

Focus of the GA for the image reconstruction problem is to find the true conductivity distribution of the body; or for some cases the closest possible solution to the true conductivity distribution. For some cases, where the electrical currents path is blocked, it may be impossible to determine the true conductivity distribution of the body. In situations like these, algorithm should find the best possible solution, which is the closest distribution possible to the actual distribution. To achieve these results, there are two important objectives to consider before the development of the genetic algorithm. First of these objectives is to achieve convergence to a population of solutions and the second one is to find the true conductivity distribution from the population of the solutions that are close to the exact result. Considering these two objectives, a strategy of a two-staged genetic algorithm is developed. A GA is developed with two stages, each stage being a genetic algorithm with different parameters and operators for two different objectives. Objective of the first stage of the GA is to achieve convergence in the population into a state that the distance between the individuals of the population and the actual conductivity distribution is at minimum. Objective of the second stage is to find the exact conductivity distribution using the population from the first stage of the algorithm as the initial population.

General overview of the genetic algorithm is seen in Figure 3.3. Algorithm starts with the reading of measurement data and GA parameters from the disk and saving to the memory. Next step of the algorithm is the calculation of local admittance matrices for the FEM model. These local admittance matrices don't depend on the conductivity distribution, therefore, they are calculated before the algorithm starts to prevent the same matrices from being computed every time the fitness function is evaluated. This pre-calculation step speeds up the evaluation of the fitness function later on. After the calculation of local admittance matrices, weight function factors are computed. Weight function is applied because of the ill-conditioning nature of the imaging system. Weight functions purpose is to increase the sensitivity of the pixels that are located in the center of the body. Details of the weight function are discussed later in this chapter. First stage of the algorithm runs until the desired convergence criteria are satisfied. After the first stage, the second stage of the GA

runs until an exact or acceptably good result is reached. Algorithm ends with displaying the results.

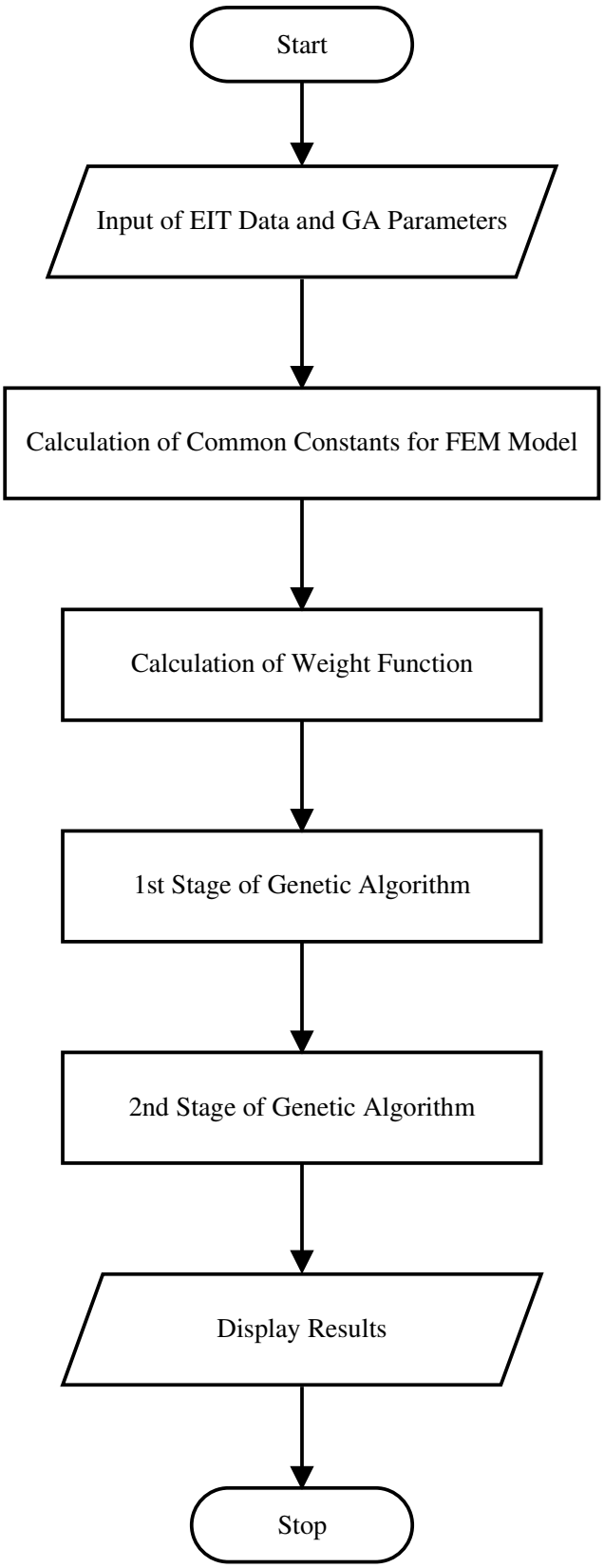


Figure 3.3: Flowchart of the two-stage genetic algorithm for image reconstruction problem.

3.3.1 Structure of two-stage genetic algorithm

Genetic algorithm for the image reconstruction problem consists of two stages, each having a separate objective. Objective of the first stage is to converge the population closer to the exact solution of the problem. Because the first stage of the algorithm starts with a random initial population, diversity among the population is very high at the beginning. Therefore, no additional operation to increase the diversity is necessary and the mutation rate is kept at a minimum value. Selection pressure is applied sensitively to increase the convergence speed of the algorithm. In the second stage, diversity among the population becomes lower than the first stage of the algorithm. At this stage, mutation rate must be increased in order to maintain rich diversity levels. High mutation rates also help the algorithm to evade the local minimum points to reach the true result. Selection pressure is also reduced to prevent the diversity from falling beneath a low limit. In the second stage of the algorithm, rich diversity is maintained mostly by the mutation operator and keeping a rich diversity level among the population is vital for algorithms success.

Flowcharts of the first and the second stages of the genetic algorithm are shown in Figure 3.4 and Figure 3.5 respectively. Main differences between the first and the second stages are the mutation operators and the application of fitness memory for objective function. Fitness memory is an additional function to speed up the execution of the objective function. Fitness memory function stores population of the previous generation and their fitness values. During the execution of the objective function, all members of the current population are compared to the previous population. For the individuals that present in the previous population, the algorithm uses the fitness value from the last generation, thus speeding up the whole process. However, because the distances between the individuals are relatively bigger in the first stage of the algorithm, only a very little amount of individuals survives exactly to the next generation. Therefore, it is only beneficial to use fitness memory in the second stage of the algorithm. Neighborhood shift mutation and center fill mutation operators are also used only in the second stage of the algorithm. These mutation operators help finding the exact solution with a shape search mentality. Neighborhood shift mutation randomly moves a foreground pixel to one of its neighbor pixels with a predefined probability. Center fill mutation turns a

background pixel that which has three foreground neighbor pixels into a foreground pixel with a fixed probability.

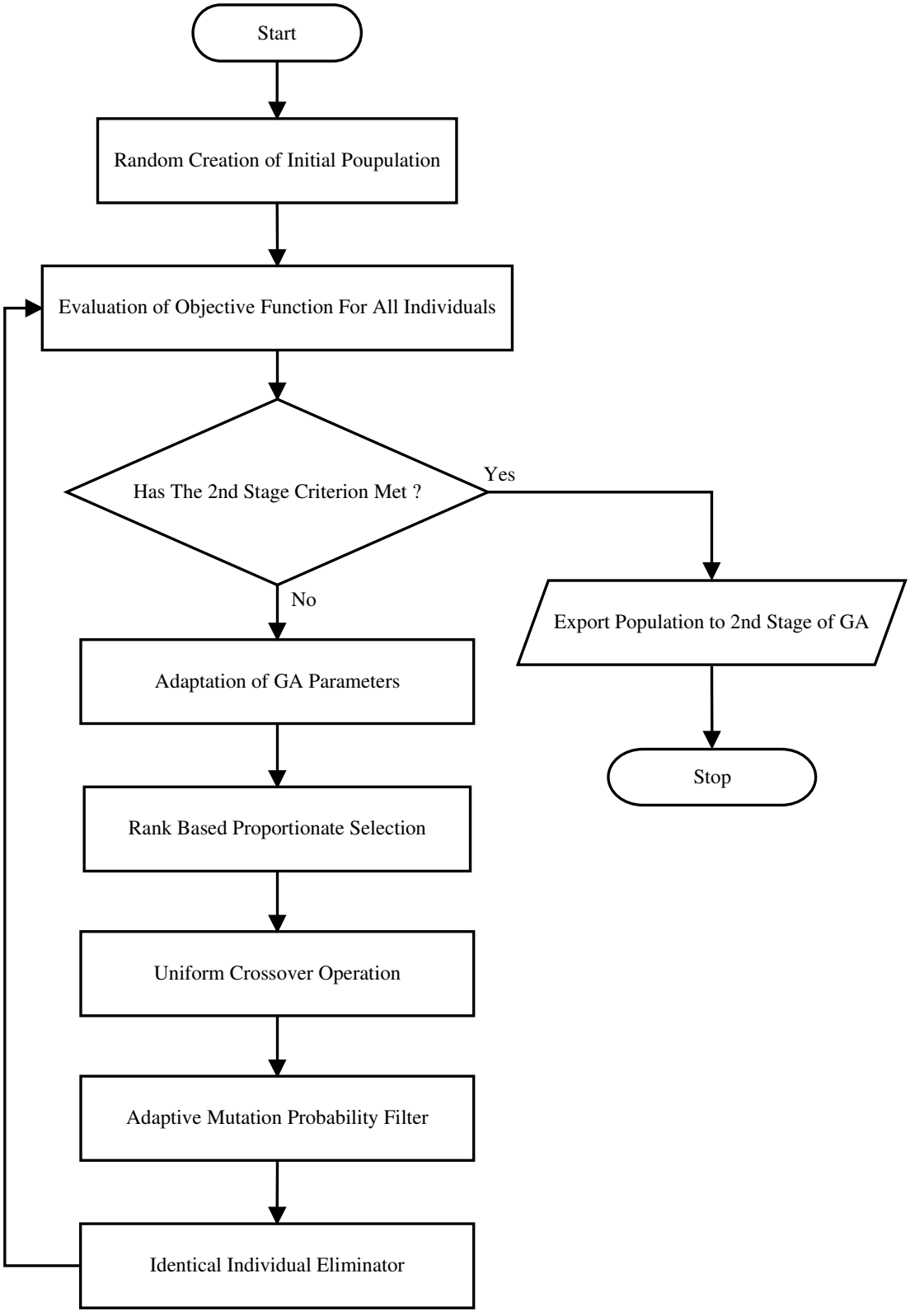


Figure 3.4: Flowchart of first stage of the genetic algorithm.

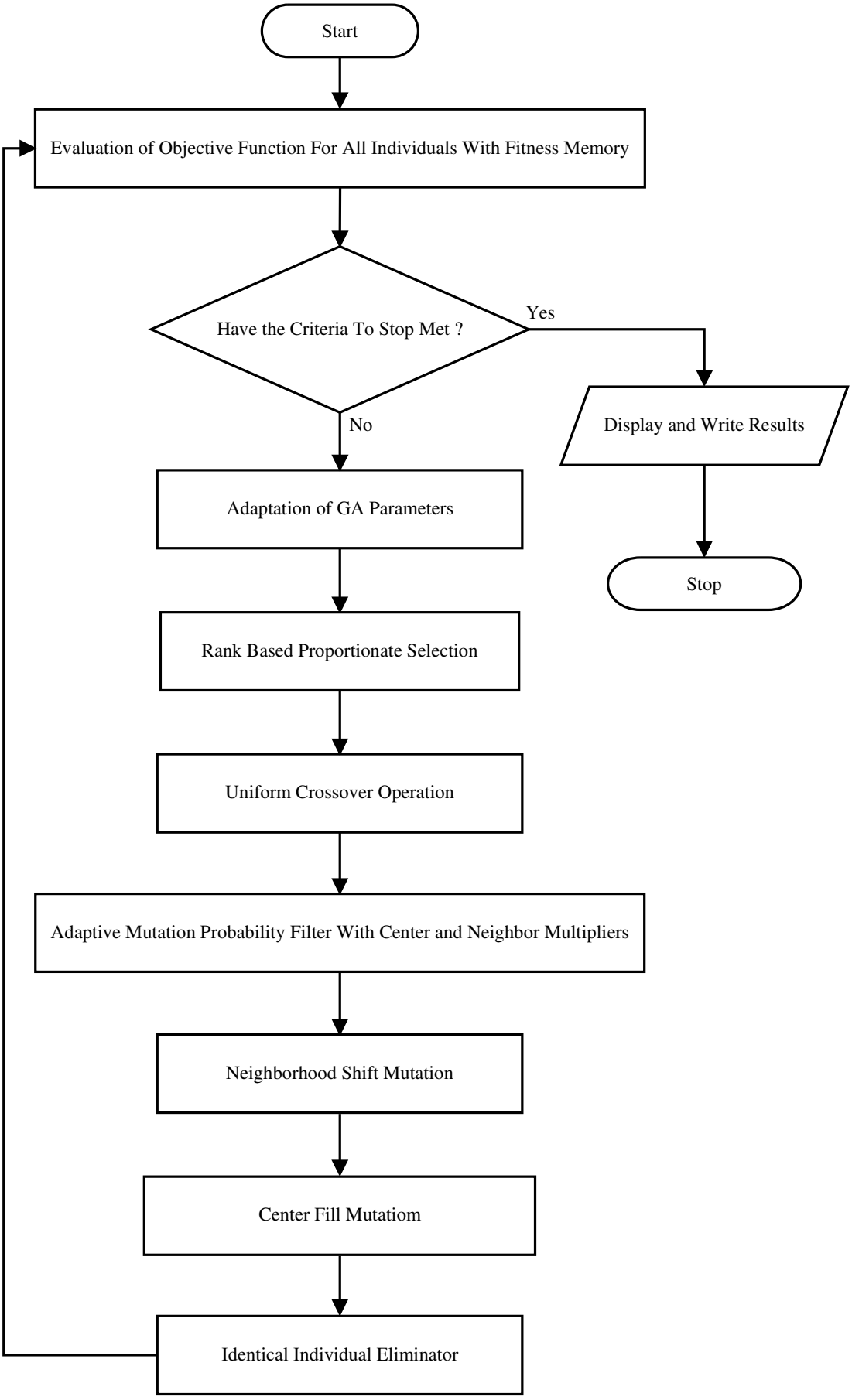


Figure 3.5: Flowchart of second stage of the genetic algorithm.

3.3.2 Population encoding

Encoding process of the population is the first stage of any genetic algorithm. Any property, which represents a quality or a quantity of a possible solution in the search space, must be encoded as bits in a GA. Bits represents genes in analogy with genetic science; as every property of organisms is represented by genes, every distinctive property of a possible solution is defined by bits. Every bit of a candidate solution is combined to form an individual. An individual is a string of bits that carries the information of a candidate solution. All the individuals of a generation form the population of the corresponding generation.

In the image reconstruction problem, every pixel of the conductivity distribution is represented by a variable. Because only the binary conductivity distributions are imaged in this thesis, reconstructed distributions includes two levels of conductivity values, a background and a foreground conductivity level. Thus, every pixel is represented by a binary variable, taking the value of one for the foreground conductivity value, zero for the background conductivity value. Any solution in the search space is fully defined using the number of binary variables that equals to the number of pixels of the reconstructed image. For the 16-electrode model, image dimensions are 9x9 pixels; the total number of pixels is 81 and therefore, any possible solution in the search space is encoded by 81 bits. For 32-electrode model, image dimensions of 17x17 sums up to a total of 289 pixels; therefore, any candidate solution in the search space is encoded by 289 bits. Bits are numbered starting from the pixel at the top-left corner to the pixel at the downright corner. Numbering of the bits is illustrated in Figure 3.6. All the bits that belong to a possible solution form a string, which is called an individual of the population.

$$i = \{g_1, g_2, ..., g_m, ..., g_M\} \qquad \textbf{(3.1)}$$

Where g represents a bit, i represents an individual and M is the total number of the bits in an individual. Strings of all individuals in a generation form the population of the corresponding generation. In the GA, the population is specified with a population matrix. The population matrix consists of all the individuals and it has the dimensions of R and M, where R is the number individuals in a generation and M is the total number of bits of an individual. A population matrix is shown in Equation 3.2.

1	2	3	4	5	6	7	8	9
10	11	12	13	14	15	16	17	18
19	20	21	22	23	24	25	26	27
28	29	30	31	32	33	34	35	36
37	38	39	40	41	42	43	44	45
46	47	48	49	50	51	52	53	54
55	56	57	58	59	60	61	62	63
64	65	66	67	68	69	70	71	72
73	74	75	76	77	78	79	80	81

Figure 3.6: Numbering of the bits on their corresponding pixels for 16-electrode model.

$$I_{RxM} = \begin{bmatrix} g_{11} & g_{12} & \cdot\cdot & g_{1M} \\ g_{21} & g_{22} & \cdot\cdot & g_{2M} \\ : & : & : & : \\ g_{R1} & g_{R2} & \cdot\cdot & g_{RM} \end{bmatrix} \quad \textbf{(3.2)}$$

Where *I* represents the population matrix.

Initial population is randomly created at the start of the algorithm. In terms of efficiency of the algorithm, rich diversity among the initial population is an important factor. It also contributes to the stochastic nature of the algorithm. Random creation operation ensures a diverse starting population and therefore, increases efficiency of the genetic algorithm.

3.3.3 Weight function

As mentioned earlier, the image reconstruction problem of EII is an extremely ill-conditioned problem. Interior pixels cause lesser response on the boundaries, where the measuring electrodes are located, than the pixels that are located near the boundary surface. This phenomenon causes sensitivity of the interior pixels to drop. Therefore, conductivity of the pixels near the boundaries dominates the error function and the pixels located near the center of the body have a lesser impact on the fitness values of the individuals. This situation prevents the convergence in the central region of the conductivity distribution.

To overcome the low sensitivity issue, error function must be modified to decrease the ill conditioning of the problem. There are several methods to improve the conditioning of the inverse problems. However, most of these methods are not very applicable to the image reconstruction problem. Therefore, a special weight function is developed specifically for the image reconstruction problem of impedance imaging to increase the sensitivity of the interior pixels, thus, reducing the ill conditioning of the problem. Main idea behind the weight function approach is that every excitation focuses different areas of the body; therefore, the data from different excitations amplify the information from different regions of the conductivity distribution. However, magnitudes of this amplification vary dramatically because of the ill-conditioned nature of the system. Therefore, extremely important data are neglected because of this difference. Aim of the weight function is to scale the data from each excitation in respect to the magnitudes of the error function value each excitation contributes. Weight function determines the scaling factors for each excitation by comparing the data from the actual measurement to the data from the numerical simulation using homogeneous background distribution.

A series of numerical simulations is conducted to show the sensitivity drop in the central area of the body by using the conductivity distribution including a foreground pixel moving in the horizontal direction on a homogeneous background. Results are shown in Figure 3.7.

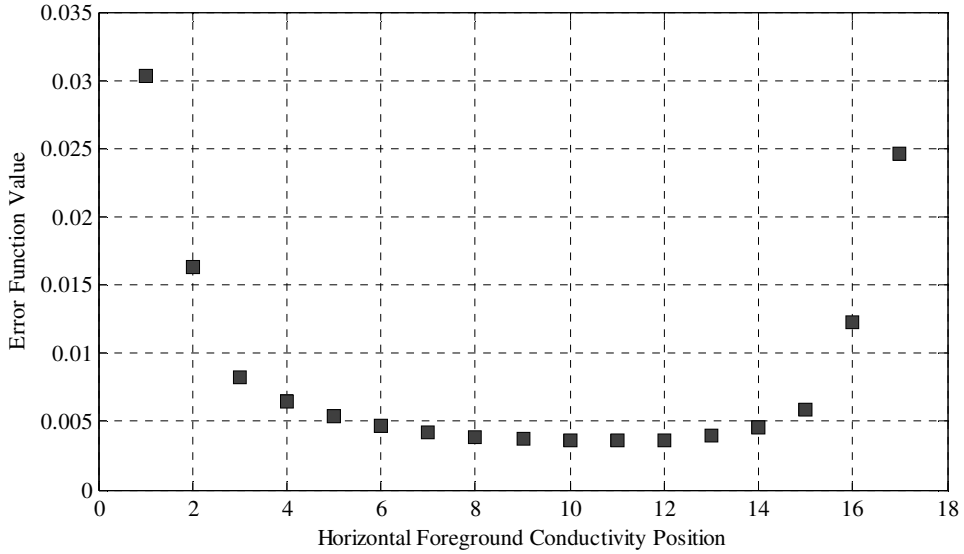


Figure 3.7: Error function values for a foreground pixel moving horizontally in a homogeneous distribution.

As the foreground pixel moves to the central region of the body, its impact on the error function is dramatically reduced. A pixel near the boundary is more than six times sensitive than a pixel in the central region.

In the execution of weight function, numerical simulation of the imaging system using the homogeneous conductivity distribution, where all the pixel conductivities set equal to the background conductivity value, is computed. For the numerical simulation, parameters are selected as same as the actual measurement of the imaging process. After the simulation, the difference between the actual voltage data and the voltage output of the homogeneous simulation is calculated. Error values are obtained using the difference data for each excitation. The weight function then analyses the error values for every excitation and calculates the factors to reduce the difference between the error value contributions of each excitation. Weight function factors range from zero to one. Excitations that provide a higher error value are multiplied with smaller factors in the execution of the objective function; thus, closing the gap between the magnitudes of data from the different excitations. Weight function prevents some pixels from becoming too dominant and increases the sensitivity of the pixels that resides in the center of the body. In Figure 3.8, flowchart of calculation of the weight function is illustrated.

Error values for each excitation are calculated using Equation (3.3).

$$w_f(i) = \sum_{e=1}^{E} \left(H_d(i,e) - T_d(i,e)\right)^2 \qquad \text{for } i = 1,2,3, \ldots ,P \tag{3.3}$$

Where w_f denotes error values, H_d denotes the voltage values on the corresponding electrodes from the numerical simulation data using homogeneous distribution, T_d denotes the voltage values on the corresponding electrodes from the actual measurement data, e index represents the electrode numbers and i index represents the excitation numbers. E is the total number of electrodes and P equals to the total number of excitations of the imaging system. Weight function scaling factors are calculated for all excitations using Equation (3.4).

$$W_f(i) = 1 - erf\left(\alpha \cdot \frac{w_f(i)}{w_{\max}}\right) \qquad \text{for } i = 1,2,3, \ldots ,P \tag{3.4}$$

Where W_f represents the weight function scaling factors for each excitation, w_{max} represents the highest error function (w_f) value that is computed in Equation (3.3),

and α represents the weight function parameter that controls the sharpness of the scaling operation.

Increasing α parameter increases the presence of the weight function, setting the scaling factors more radically, causing more amount of information to be cut out from the data and increasing sensitivity of interior pixels. Decreasing α parameter reduces weight functions effects and limits weight functions scaling factors from decreasing beyond a certain level.

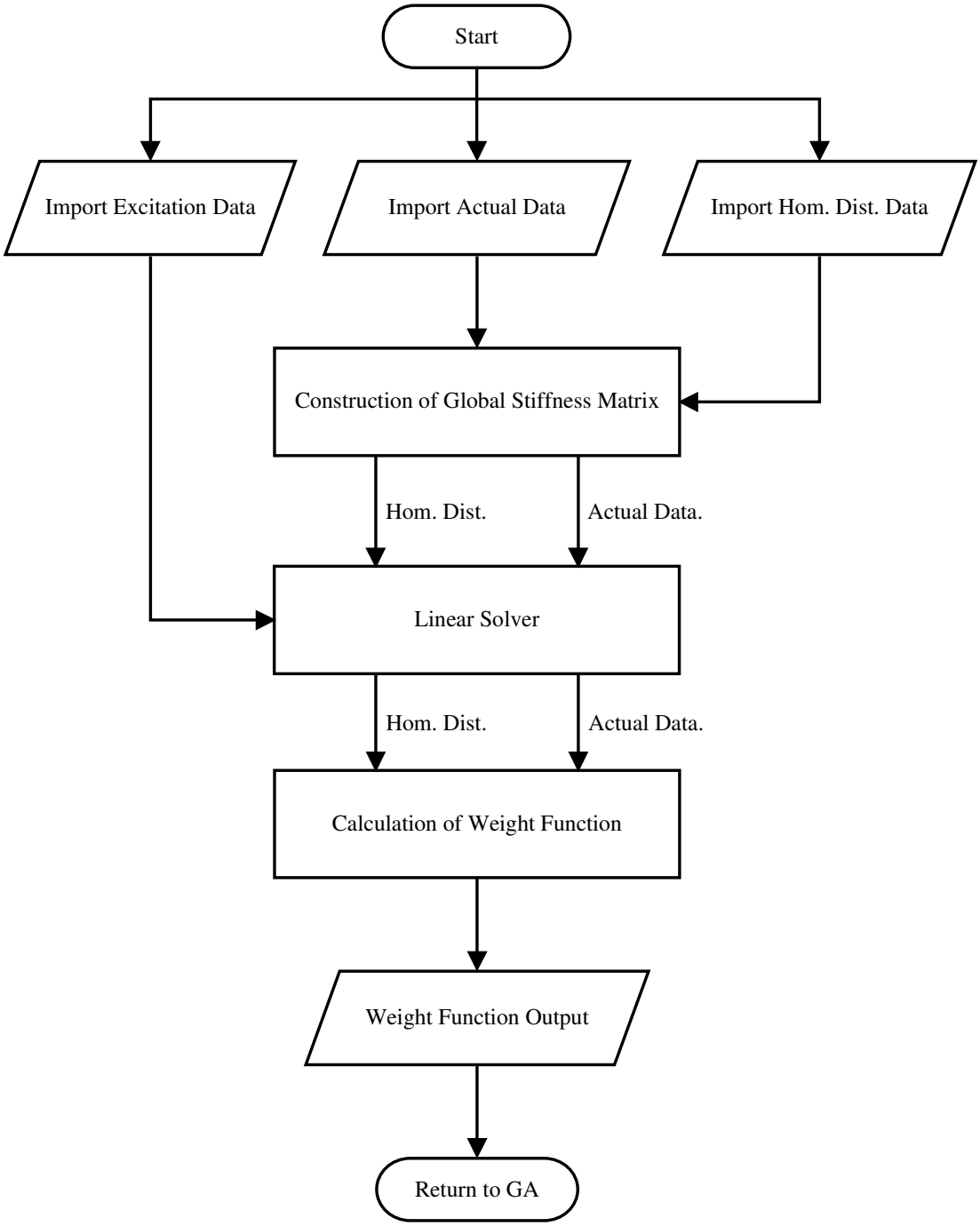
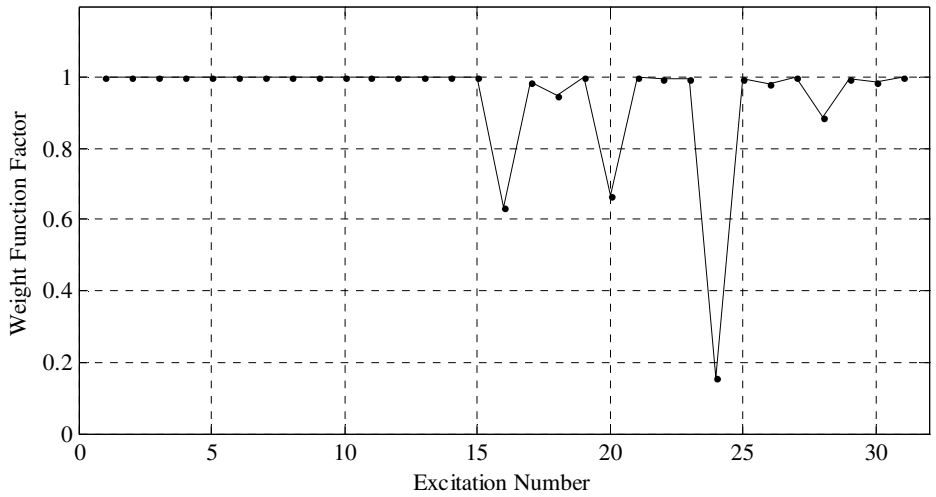
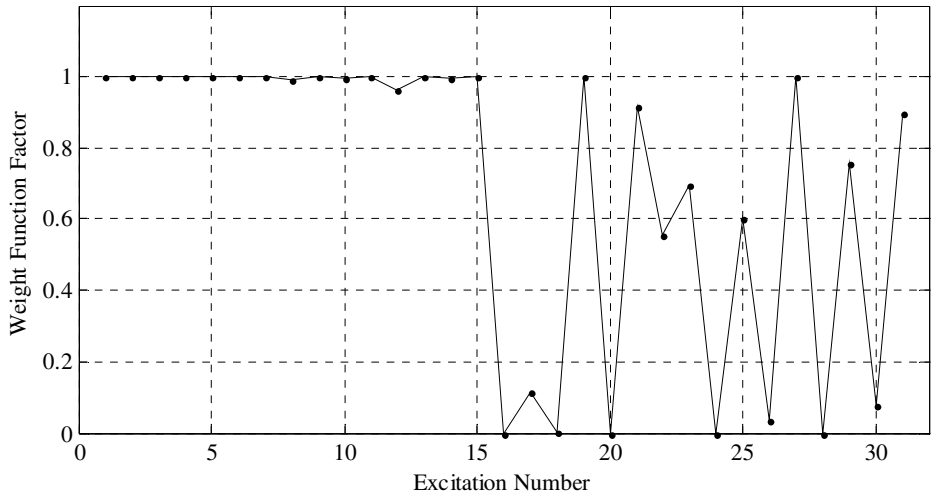


Figure 3.8: Flowchart of calculation of the weight function.

To demonstrate the effects of α parameter, weight function scaling factors is plotted using different α values for a fixed conductivity distribution on 32-electrode model. Figure 3.9 (a) shows the weight function factors for each excitation for α parameter equal to one. Figure 3.9 (b) shows the weight function factors for α parameter equal to 100 using the same conductivity distribution. In Figure 3.9 (b), we can see that more data are discarded from the excitations starting from the excitation number 15. Comparing two plots, a higher α parameter causes the weight function to behave more aggressively, having freedom to discard more data from the excitations. However, increasing α parameter keeps the error values closer to each other in terms of contributions of every excitation.



(a)



(b)

Figure 3.9: Weight function scaling factors for each excitation: **(a)** Weight function factors for $\alpha = 1$. **(b)** Weight function factors for $\alpha = 100$.

3.3.4 Objective function

Aim of the objective function (also called "fitness function") is to measure the fitness of the individuals. Genetic algorithms need a mathematical function to trial the individuals according to their fitness. To select the better individuals for recombination, first, it is crucial to measure their fitness levels numerically. After this numerical representation of fitness, different individuals can be compared for their distance to the exact solution. Objective functions aim is to represent the fitness of the individuals using numerical values.

Objective function of the image reconstruction problem is very similar to an error function. However, in contrast with an error function, smaller error value points to a fitter individual. To determine the fitness of an individual, which is the distance to the true solution in this case, firstly an error measuring function is created. An error function with a least squares approximation is used in the algorithm. Error function is shown in Equation (3.5).

$$e_f = \frac{1}{2}\left(V_{ind} - V_0\right)^T \left(V_{ind} - V_0\right) \qquad \textbf{(3.5)}$$

Where e_f is the error value of the individual, V_{ind} is the voltage output values from the simulation using the conductivity distribution of the individual and V_0 is the voltage output values from the actual measurement. After the combination with the weight function factors, least squares error function stated in Equation (3.6).

$$\varphi = \sum_{i=1}^{P} \sum_{e=1}^{E} W_f(i) \cdot \left[V_{ind}(i,e) - V_0(i,e)\right]^2 \qquad \textbf{(3.6)}$$

Where φ represents the error value of the corresponding individual. Objective function numerically simulates the voltage values on the electrodes for the individual's conductivity distributions using the FEM model that is introduced in the second chapter of the thesis. Results of this simulation are compared to the results from the actual measurement of the imaging process using the error function stated above.

Results of the objective function are stored as a string in the algorithm. This fitness string carries the error values for every individual in the population. Error values are inverse proportional with the fitness of the individuals. Fitness string is shown in Equation (3.7).

$$f = [\varphi_1, \varphi_2, \varphi_3, ..., \varphi_n, ..., \varphi_N] \quad \textbf{(3.7)}$$

Where f represents the fitness string and φ represents the calculated error values of the individuals. Fitness string is used by the selection operator for deciding the individuals that will be selected for recombination process later in the algorithm. Flowchart of the objective function is shown in Figure 3.10.

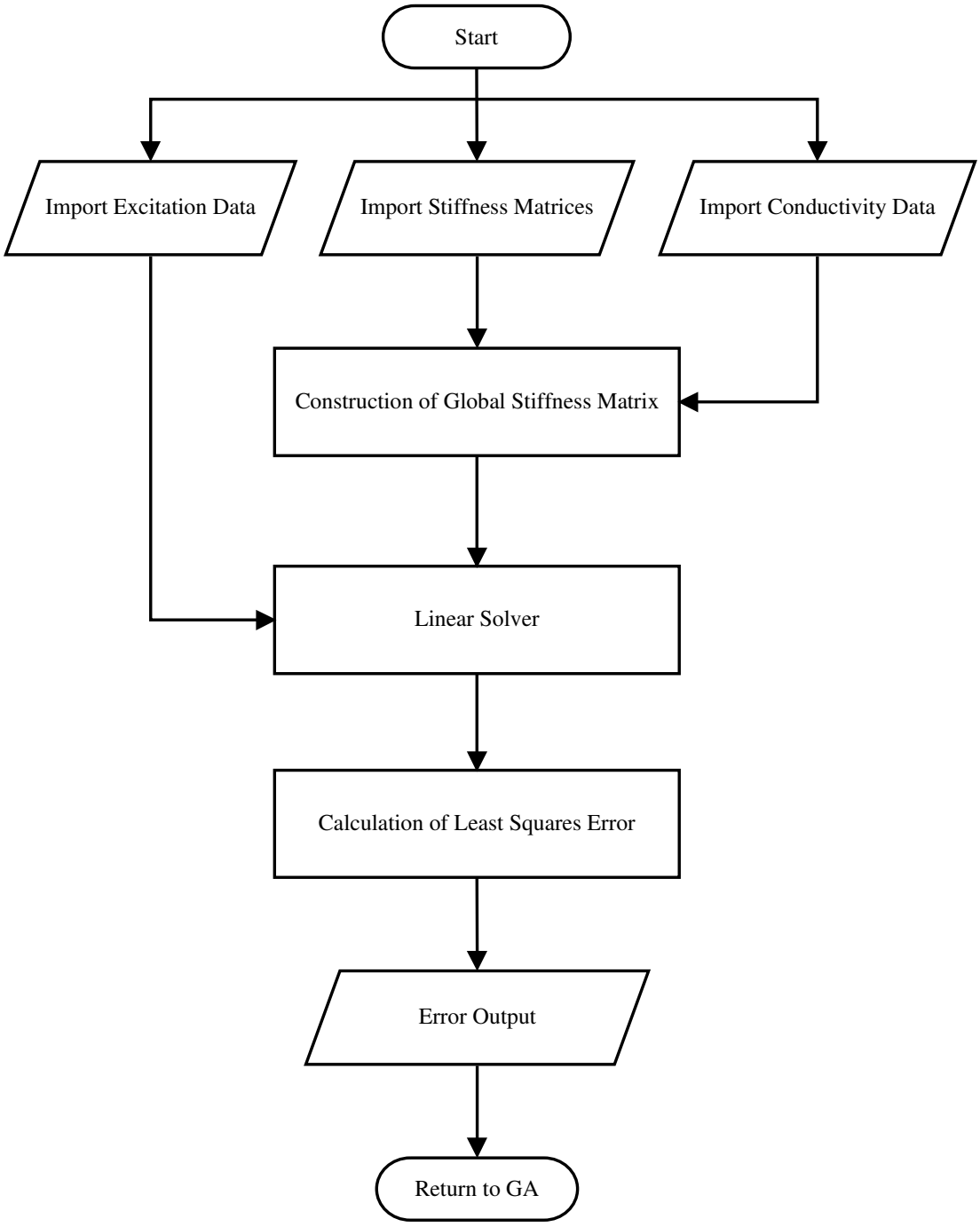


Figure 3.10: Flowchart of the objective function.

3.3.5 End criteria of genetic algorithm

Like every other search and optimization method, genetic algorithms also need to be stopped when the desired criteria are met. End criteria of a GA are the group of conditions that directs the GA to stop working and output the results to the user. There can be single or multiple ending conditions depending on the structure of the GA.

There are three conditions to stop for the genetic algorithm for the image reconstruction problem; first is to reach the maximum desired error value, second is the end of the convergence and the last is to reach the maximum iteration limit. In the first condition, the algorithm stops when an individual's error value falls below a certain threshold and the individual is selected as the result of the algorithm. However, value of this threshold must be selected small enough to prevent the algorithm from stopping before attaining the true solution. Generally, values between 10^{-5} and 10^{-4} are observed to be suitable for low-noise imaging conditions. When working with data that contain heavy noise, this value should be increased further. An optimal value for the threshold should be selected experimentally.

Second condition is activated when the number of successive iterations without an improvement reaches to a certain limit. Criterion for no improvement is triggered when the best individuals of predetermined number of successive generations have the same error values. The limit must be set high enough to ensure that the convergence is stopped completely to prevent any premature results. Generally, values above 200 are suitable for this limit. Value of this limit is safely selected as 250 for the genetic algorithm. Activation of the second condition means that the algorithm has stopped without reaching the exact conductivity distribution. Presence of this condition is crucial to prevent the algorithm from running pointlessly when no convergence can be achieved.

The last condition stops the algorithm when the maximum iterations limit is reached. Maximum iterations limit is selected high enough (around 1000-2000) to prevent the algorithm from stopping too early before finding the true solution. There is also the condition to stop for the first stage of the algorithm.

When a predetermined number of successive iterations pass without any improvement, the first stage of the algorithm ends and the second stage of the

algorithm starts. The number of successive iterations requires for the initiation of the second stage of the algorithm can be selected between 10 and 50; for this study, it is experimentally selected as 20.

3.3.6 Selection

Selection is the process of choosing parents from the population for recombination. The purpose of selection is to emphasize fitter individuals of the population to form a fitter population in the next generations. Selection algorithm is built in a mentality that the fitter individuals of the population have a higher chance of selection than the less fit ones. These selection chances are regulated by applying selection pressure. The selection pressure is defined as the degree that the fitter individuals are favored in terms of selection chances. The higher selection pressure means the better individuals will be favored more. Main factor behind the convergence of a genetic algorithm is selection pressure. The convergence speed of a GA is mostly influenced by the magnitude of the selection pressure. Higher selection pressures often result in higher convergence speeds. However, if the selection pressure is set too high, because of the reduced diversity among the population, there is a big chance that the algorithm prematurely converges into an incorrect solution. If the selection pressure is set too low, the convergence rate decreases dramatically and the time required for reaching a solution unnecessarily increases. Exact solution can only be reached within minimal computing times by using optimum selection pressure values.

There are three most common selection methods for GAs, Tournament selection, fitness proportionate selection, and rank-based proportionate selection. In tournament selection, a group of individuals is chosen randomly from the population and the individual with the highest fitness value is selected to be a parent for the next generation's population. Selection pressure can be controlled by the number of the individuals selected to the group, which is also called tournament size. Despite being a simple and efficient method, it is very difficult to control the characteristics of the selection pressure in this method. Another selection method is fitness proportionate selection, where each individual is given a probability of selection in proportionate to its fitness level. Selection of parents is executed randomly using the probabilities calculated according to the individuals' fitness values. Fitness proportionate selection is a popular method; however, controlling the selection pressure is impossible without using a scaling function.

Rank-based proportionate selection method uses ranks of the individuals rather than their raw fitness levels. In this method, all the individuals of the population are sorted according to their fitness levels. The individuals are given selection probabilities in proportionate to their position (their rank) in the sorted population. This method is very flexible and efficient, because it allows the distribution of the selection probabilities to have any predetermined characteristic and shape. Diversity is preserved more efficiently with the rank-based proportionate selection than the other methods due to its nature that prevents any individual from becoming too dominant. It is also very easy to apply the selection pressure with this method, because the selection probabilities curve is determined before the algorithm starts. Due to these advantages, rank-based proportionate selection with an elitist selection scheme is developed for the genetic algorithm for the image reconstruction problem.

3.3.6.1 Elitist selection

Elitist selection is a selection method, where the best individuals of the population survive to the next generation. Elitist selection is used in combination with the other selection methods. After a constant number of individuals with the highest fitness values are selected directly to the next generation by the elitist selection, an another stochastic selection method choses other parents to fill the remaining spaces in the population of the next generation. Number of individuals that are selected by the elitist selection, (in other words, “elitist selection quota”) must be rather small compared to the population size. Generally, values between %0.5 and %3 of the population size is suitable for GAs. Elitist selection quota is determined to take the value of two for the first stage and four for the second stage of the GA for the image reconstruction problem.

Because the elitist selected individuals are not subjected to recombination, setting the elitist selection quota too high diminishes the stochastic nature of the genetic algorithm and keeps algorithm from converging to the true solution. Elitist selected individuals are also not affected by the mutation filters. While reducing the diversity of the population, elitist selection increases the convergence speed. Elitist selection is a useful tool for GAs, because it prevents the algorithm from diverging by losing the best individual of the population. Always preserving the best individual of the population to the next generation, elitist selection is an insurance to prevent the divergence.

3.3.6.2 Rank-based proportionate selection

First step of the rank-based proportionate selection method is to construct the selection probabilities string, which includes the selection probability values of all individuals of the population. There is a wide range of mathematical functions that can be used as the selection probability curve; however, exponential functions are proved to be the most efficient ones for the genetic algorithm, because exponentially increasing selection probabilities fits the stochastic nature of the GAs better. Before the calculation of selection probabilities, individuals of the population are sorted according to their fitness values and every individual are given a rank value in respect of the individual's sorted position. Rank values are integers ranging from one to R, where R is the population size, and the rank value of one refers to the best individual. Normalized rank of the individuals is calculated by dividing the individuals rank value with the population size.

$$r_n(i) = \frac{r(i)}{R} \qquad \text{for } i=1,\ldots,R \tag{3.8}$$

Where $r(i)$ represents the i-th individuals rank and $r_n(i)$ represents normalized rank of the i-th individual. Normalized rank values ranges from zero to one. Selection probabilities of the individuals are computed using Equation (3.9).

$$p(i) = \frac{e^{-\beta \cdot r_n(i)}}{\sum_{n=1}^{R} e^{-\beta \cdot r_n(n)}} \qquad \text{for } i=1,\ldots,R \tag{3.9}$$

In Equation (3.9), $p(i)$ represents the selection probability of i-th individual and β is the selection pressure parameter. The sum of selection probabilities of all individuals in a population is equal to unity. Increasing the selection pressure parameter increases the selection probability of the fitter individuals. Selection pressure of zero means every individual has equal selection probability, which would prevent the convergence of the algorithm. Therefore, the selection pressure must be selected greater than zero in order to achieve convergence. To show the effects of the selection pressure parameter β, Selection probabilities for a population size of twenty individuals are plotted using β values of two and five in Figure 3.11. It is seen from these figures that increasing β value increases the selection probabilities of the individuals with high fitness values, while decreasing the selection probabilities of

the individuals with low fitness values. Individuals with mid-range fitness values are not remarkably affected from β.

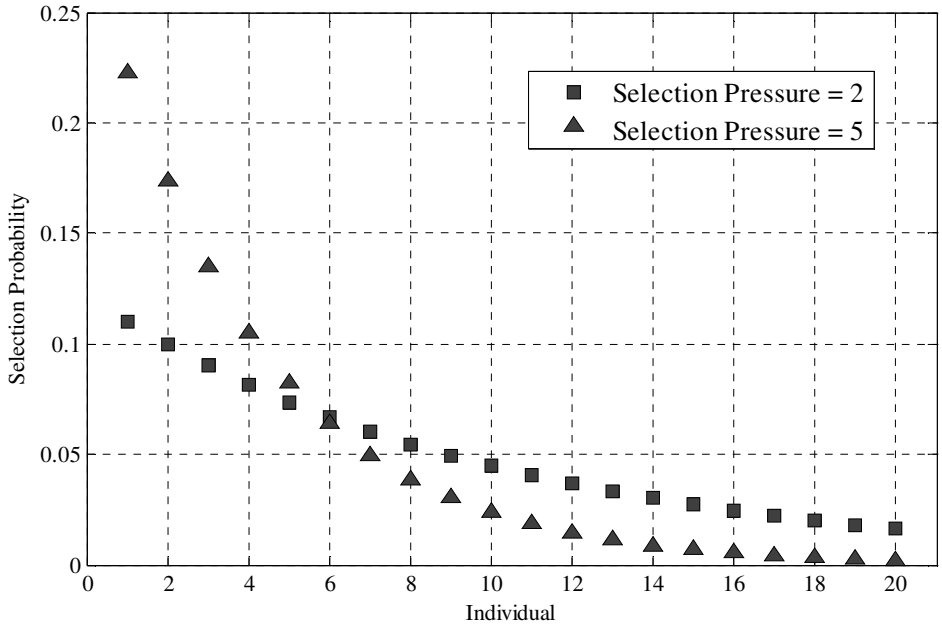


Figure 3.11: Selection probabilities of the individuals for a population size of 20 using selection pressure value of 2 and 5.

After the calculation of the selection probabilities, selection string is formed. The selection string is the series of sums of the probabilities from first to each individual of the population. Starting with zero, it has (R + 1) terms. The selection string is shown in Equation (3.10).

$$s = \left\{0, p(1), \sum_{n=1}^{2} p(n), \sum_{n=1}^{3} p(n), \sum_{n=1}^{4} p(n), ..., \sum_{n=1}^{R} p(n)\right\} \qquad \textbf{(3.10)}$$

Selection string is used in the algorithm by creating a random real number between zero and one, and checking the selection string for the condition that the random number is greater than the n-th term and it is less than the (n+1)-th term. The n value that satisfies this condition is selected as the parent individual by the selection algorithm.

In this routine, there is a chance that the same individual is selected for both parents. If this situation occurs, selection routine is repeated until different individuals are selected as parents. This selection process continues until the desired amount of parents is selected for the recombination. Pseudocode of the selection algorithm is shown in Figure 3.12.

```
Sort the individuals according to their fitness values.
For i=1:(elitist selection quota);
   Select i-th best individual as a parent.
Assign rank numbers to individuals according to sorted positions.
For i=1:population size;
   Calculate selection probabilities of individuals.
Construct the selection string.
For i=1:(required number of parents - elitist selection quota)/2;
   Create a random number between (0,1)
   For j=1:(population size);
      If the random number > j-th term of the selection string and
        the random number < (j+1)-th term of the selection string;
        Select j-th individual as a parent.
   Create a random number between (0,1)
   For j=1:(population size);
      If the random number > j-th term of the selection string and
        the random number < (j+1)-th term of the selection string;
        Select j-th individual as a parent.
   If the first parent is the same as the second parent;
      Repeat selection routine with different random numbers
```

Figure 3.12: Pseudocode of the selection algorithm.

3.3.7 Crossover

Crossover is the process of recombining two parent individuals and creating two child individuals using the parents' genes. Crossover process enriches the population with better individuals by recombining the fitter individuals of the previous generation. After recombining two selected individuals, there is a chance that the offspring will carry the good characteristics of both parents. Crossover operator recombines the individuals that are chosen as parents by the selection. Because the selection operator increases the selection probabilities of the better individuals, the individuals with good characteristics tend to mix with each other more frequently than the others, increasing the fitness of the next generation's populations and achieving convergence in the process. Shaping the breeding process of a genetic

algorithm with the selection operator, crossover operation is vital for convergence of any GA.

During the crossover operation, strings of both parents are mixed together, forming two child individuals. This exchange of the bits can be done with different methods. The most simple crossover method is one-point crossover, where the strings of both parents are cut from a random position into two pieces and one of the pieces is exchanged between parents. However, this method does not provide a good mixture. In two-point crossover technique, strings are cut from two random points, providing a better mixture rate. The crossover technique that provides the best mixture is uniform crossover. In uniform crossover technique, crossover is applied to the strings of bits using a random crossover mask. Exchange of the bits is independently decided using the crossover mask for each bit in the string. Uniform crossover method is the most advanced crossover method to use with the binary variables and because the image reconstruction problem demands good mixture rates for success, this method is ideal for the GA for the image reconstruction problem.

3.3.7.1 Uniform crossover

Uniform crossover is a technique where each gene in the offspring is created by copying the corresponding gene from the parent that is chosen according to a random generated binary crossover mask that has the same length as the chromosome size of the individuals. Crossover between parents is executed for each bit if the corresponding entry of the crossover mask equals to one. An Illustration of the uniform crossover method is shown in Figure 3.13.

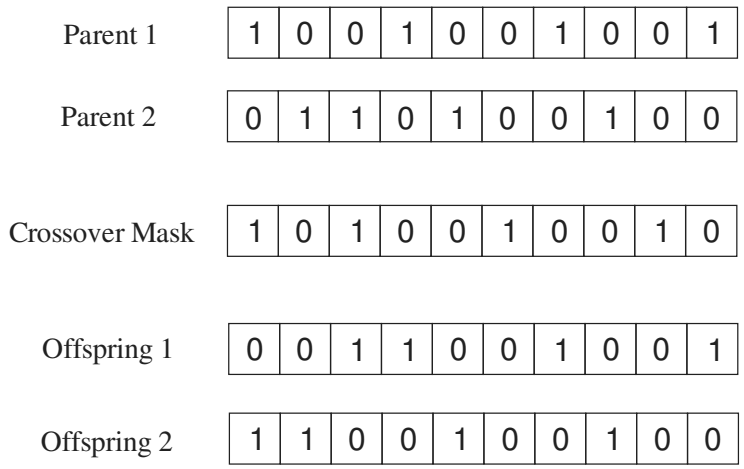


Figure 3.13: Illustration of the uniform crossover method.

Randomized crossover mask is a random binary string with the length of the bit number of the individuals. For every bit of the individuals, algorithm checks the corresponding term of the crossover mask. If the corresponding entry of the crossover mask equals to zero, first offspring takes the corresponding bit from the first parent and second offspring from the second parent. If the crossover mask equals to one, first offspring takes the corresponding bit from the second parent and the second offspring from the first parent. Because the entries of the crossover mask are created randomly using equal probability for each binary value, perfect mixture rates can be achieved using the uniform crossover method. Pseudocode of the crossover algorithm is shown in Figure 3.14.

```
For i=1:(population size – elitist selection quota)/2;
   Create a random number between (0,1).
   If the random number < crossover rate;
      Create the crossover mask as a random logical string.
   If the random number > crossover rate;
      Set all the entries of crossover mask to zero.
   For j=1:(individual string length);
      If j-th term of crossover mask is zero;
        Assign j-th bit of the first parent to first offspring.
        Assign j-th bit of the second parent to second offspring.
      If j-th term of crossover mask is one;
        Assign j-th bit of the first parent to second offspring.
        Assign j-th bit of the second parent to first offspring.
```

Figure 3.14: Pseudocode of the crossover algorithm.

Crossover is applied to the selected individuals with a probability called crossover rate (crossover probability). Crossover rate determines whether the selected parent individuals will be subjected to the crossover operation or they will be transferred to the next generation without being mixed with each other. Before execution of the crossover operation, a random real number is generated between zero and one. If the generated number is greater than the crossover probability, parent individuals are transferred to the next generation's population without any changes, otherwise the crossover routine is applied to the parent individuals. Because the genetic algorithms operate more efficiently with good mixture rates, crossover probability should be

selected higher than 0.5. Values around 0.9 are ideal for crossover probability for the image reconstruction algorithm.

3.3.8 Mutation

After crossover, newly formed population is subjected to mutation. Main purpose of the mutation operator is to prevent the algorithm from being trapped in a local minimum. Mutation maintains diversity in the population and it is an insurance against the irreversible loss of the genetic material. Mutation operator randomly changes the bits of the individuals. These random changes help the GA to move forward when a better solution is not available in the gene pool. When the algorithm is stalled due to the poor diversity in the population, mutation is the only way to improve the population. In binary strings, mutation is executed by inverting the value of the bit that is subjected to mutation. An illustration of mutation for binary strings is shown in Figure 3.15.

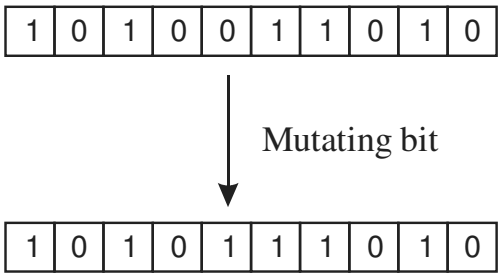


Figure 3.15: Illustration of mutation for binary strings.

Success of the genetic algorithm for the image reconstruction problem relies heavily on the mutation operator. For the first stage of the algorithm, diversity among the population is rich enough to maintain good convergence speeds. Therefore, intensive mutation rates are not required for the convergence of the algorithm. Mutation probabilities for the first stage of the algorithm are kept around minimal values.

Two mutation operators are used in the first stage of the algorithm, Adaptive mutation probability filter and identical individual eliminator mutation filter. Both of the operators, which are first introduced in this thesis, are developed specifically for the GA for the image reconstruction problem.

Aim of the second stage of the GA is to find the exact conductivity distribution of the body using a population of converged solutions. Because the population is converged close to the true solution in the first stage, diversity among the population is very low

in the second stage of the algorithm. To enrich the diversity of the population in the second stage, mutation probabilities are radically increased. Two additional mutation types are developed to provide additional diversity to the population. Neighborhood shift mutation filter and center fill mutation filter operators, which are first introduced in this thesis, are developed specifically for the second stage of the algorithm using a shape searching mentality. These two mutation operators are used intensively with high mutation probabilities with the aim of reaching the exact conductivity distribution image by searching in the population of solutions.

3.3.8.1 Adaptive mutation probability filter

Adaptive mutation probability filter is a modified version of the conventional mutation operator that is developed exclusively for two-dimensional image reconstruction problem. This mutation filter adaptively alters the mutation probability of each bit according to the diversity of the population for the corresponding bit. As the algorithm advances through the generations, diversity of some bits may fall radically where certain bits of the most individuals of the population carries the same value. This reduction of diversity for certain bits slows down the convergence of the algorithm and it may stall the algorithm completely as well. The idea behind the adaptive mutation probability filter is that increasing the mutation probability for these bits increases the diversity of the corresponding bits, increasing the efficiency of the overall algorithm and the convergence speed. In adaptive mutation probability filter, mean values of each bit of the population is calculated and the mutation probability for each bit is computed adaptively using the mean values independent from each other. Pseudocode of the adaptive mutation probability filter for the first stage of the genetic algorithm is shown in Figure 3.16. Diversity of the bits is defined by the normalized mutation parameter that is shown in Equation (3.11).

$$l(m) = \left| \frac{2}{R} \sum_{n=1}^{R} b(m) - 1 \right| \qquad \text{for } m=1,2,3, \ldots ,M \tag{3.11}$$

Where $l(m)$ represents the normalized mutation parameter for m-th bit, $b(m)$ represents the binary value of m-th bit, R represents the population size and M represents the string length of the individuals. Indexes n and m represent the individual and bit numbers respectively. Normalized mutation parameter of the bits

ranges between zero and one, taking the value of zero in the case of the richest diversity. Normalized mutation parameter changes linearly with the diversity for each bit. Mutation probability for *m*-th bit, which is represented by $P_m(m)$, is stated in Equation 3.12.

$$P_m(m) = l(m) \cdot \left(P_{\max} - P_{\min}\right) + P_{\min} \qquad \text{for } m=1,2,3, \ldots, M \qquad \textbf{(3.12)}$$

Where P_{max} and P_{min} are the maximum and the minimum limit values for the mutation probability. Mutation probability varies linearly between these values depending on the diversity of *m*-th bit in the population. After the calculation of the mutation probabilities for each bit in the string, each bit of the individuals is subjected to mutation with the mutation probability of the corresponding bit. Mutation is operated by inverting the binary value of the bit if a randomly created number is lesser than the mutation probability of the corresponding bit.

```
Calculate adaptive mutation probabilities for all bits.
For i=(elitist selection quota):(population size);
   For j=1:(individual string length);
      Create a random number between (0,1).
      If the random number < mutation probability for j-th bit;
         Mutate j-th bit of i-th individual.
```

Figure 3.16: Pseudocode of the adaptive mutation probability filter for the first stage of the genetic algorithm.

Adaptive mutation probability filter is used in both stages of the algorithm; however, additional routines are included to the operator for the second stage of the algorithm. Because of the sensitivity drop in the interior region of the conductivity distribution, convergence is achieved faster for the pixels that reside in the outer region of the image, than the pixels that resides in the center. Generally, the pixels in the outer areas of the body are determined before the end of the first stage of the algorithm. Considering this situation, if a pixel is located in the central half of the image, its mutation probability value is multiplied with a predefined factor called the central bit factor. This routine, which is only used in the second stage of the algorithm, helps finding the true conductivity distribution in the center of the body by increasing the mutation probability of the central pixels.

Another additional routine is that, if a pixel has any neighbor pixels with foreground conductivity value, its mutation probability is multiplied with a predefined factor called the neighborhood factor. Main idea behind this routine is that, the population is already converged to a state that the population of solutions is very close to the true solution of the image until the second stage; therefore, increasing the mutation probability of the neighbor pixels may help finding the exact shape of the conductivity distribution. This routine is also exclusive to the second stage of the algorithm. Pseudocode of the adaptive mutation probability filter with the additional routines for the second stage of the genetic algorithm is shown in Figure 3.17.

```
Calculate adaptive mutation probabilities for all bits.
For i=(elitist selection quota):(population size);
   For j=1:(individual string length);
      If j-th bit has a neighbor bit with foreground conductivity;
         Multiply mutation probability with neighborhood factor.
      If j-th is located in the interior of the image;
         Multiply mutation probability with central bit factor.
      Create a random number between (0,1).
      If the random number < mutation probability for j-th bit;
         Mutate j-th bit of i-th individual.
```

Figure 3.17: Pseudocode of the adaptive mutation probability filter for the second stage of the genetic algorithm.

3.3.8.2 Neighborhood shift mutation filter

When the second stage of the algorithm is initiated, population is already converged closer to the exact result. Most of the individuals in the population evolve shapes that are very similar to the exact solution of the problem until the second stage. However, finding the exactly true solution from these close candidates can be a very difficult task in the case of poor diversity among the population. As the GAs converge closer to the exact solution, convergence speed decreases and the recombination process becomes ineffective because of the reduced diversity among the population. At this stage, mutation becomes the main force behind the convergence of the algorithm. Finding the true solution, which is the main purpose of the second stage of the algorithm, can be accelerated by using mutation filters that work in a shape searching

mentality. To increase the efficiency of the GA, neighborhood shift mutation filter is developed to be used only in the second stage of the algorithm with the purpose of attaining the true solution of the problem. Neighborhood shift mutation filter moves a foreground pixel to one of its eight neighbor pixels with a predetermined probability. Neighborhood shift mutation is only applied to the pixels with foreground conductivity. Individuals of the population are subjected to neighborhood shift mutation with a fixed probability called the individual mutation probability and selected individual's foreground bits are moved to one of its eight neighbors with another probability called bit mutation probability. Neighborhood shifting mutation is used intensively in the second stage of the algorithm, bit mutation and individual mutation probabilities being around twenty percent. Process of neighborhood shift mutation is illustrated in Figure 3.18.

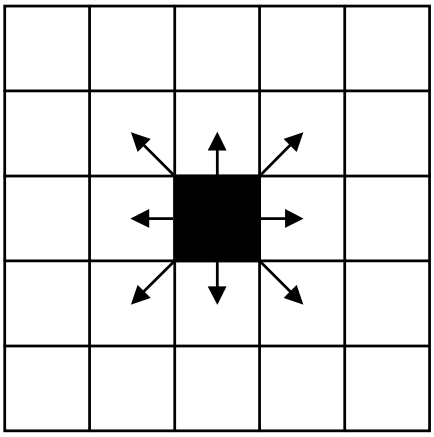


Figure 3.18: Illustration of neighborhood shift mutation process.

In neighbor shift mutation technique, a random real number between zero and one is created for each individual of the population, if the random number is lesser than the individual mutation probability, corresponding individual is selected to apply the mutation. Every bit of the selected individual is checked for the bit mutation probability by using another random number and the selected bits are moved randomly to one of its neighbor pixels in the eight directions. Choice of the direction is made by using a random integer between zero and eight, integers one to eight representing the direction numbers and zero representing that the corresponding bit mutated directly to the background conductivity. Pseudocode of the neighborhood shift mutation filter is shown in Figure 3.19.

Neighborhood shift mutation filter is one of the most important components of the genetic algorithm for the image reconstruction problem. It vastly improves the results

of the GA by accelerating the process of reaching the true solution; even sometimes, presence of the neighborhood shift mutation filter determines the difference between ending the algorithm with a close solution and attaining the true solution. In the experiments, neighborhood shift mutation filter is observed to improve the results of the GA dramatically. Neighborhood shift mutation filter helped to achieve the true solution of the problem for some complex conductivity distributions that the true result cannot be reached before.

```
For i=(elitist selection quota):(population size);
  Create a random number between (0,1).
  If the created number < nsmf individual mutation probability;
    For j=1:(individual string length);
      If j-th bit has foreground conductivity;
        Create a random number between (0,1).
        If the created number < nsmf bit mutation probability;
          Create a random integer between (0,8).
          Move j-th bit in the direction specified by the integer.
          Set the j-th bit to background conductivity.
```

Figure 3.19: Pseudocode of the neighborhood shift mutation filter.

3.3.8.3 Center fill mutation filter

Center fill mutation filter is developed exclusively for the second stage of the genetic algorithm with the aim of supporting the algorithm for attaining the true solution of the problem. Like neighborhood shift mutation filter, center fill mutation filter works in a shape searching mentality, by mutating a pixel that is surrounded by at least three foreground neighbor pixels with a predefined probability. An illustration of the center fill mutation is shown in Figure 3.20.

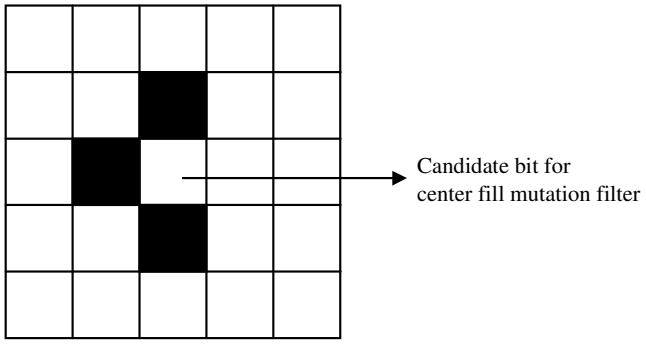


Figure 3.20: Illustration of center fill mutation filter.

In EII method, detection of pixels that are surrounded by other foreground pixels is a difficult operation. The impact that these surrounded bits causes on the boundaries is very low comparing to the other pixels. In these situations, electrical currents path often becomes blocked and very little information can be obtained from the voltage data. Therefore, reconstruction of images that includes large target objects becomes particularly difficult. To overcome this difficulty, center fill mutation filter is developed in this thesis. Center fill mutation scans whole strings of individuals for any pixels that has at least three orthogonal neighbor pixels with foreground conductivity and mutates the bit that represents the corresponding pixel with a predefined probability. Pseudocode of the center fill mutation filter is shown in Figure 3.21.

```
For i=(elitist selection quota):(population size);
   For j=1:(individual string length);
      If j-th bit has at least three neighbor foreground bits;
         Create a random number between (0,1).
         If the random number < mutation probability for cfmf;
            Mutate j-th bit of i-th individual.
```

Figure 3.21: Pseudocode of the center fill mutation filter.

3.3.8.4 Identical individual eliminator mutation filter

In GAs, there is a chance that two or more individuals have exactly the same genes in a population. When the selection pressure is set too high, fitter individuals of the early generations becomes dominant over the next generations, copying themselves through the next generations more frequently than the other individuals. Therefore, there may be more than one copy of these dominant individuals in the late generations. This situation hinders the progress of the algorithm by slowing the convergence and reducing the diversity among the population. Identical individual eliminator mutation filter is developed to prevent the presence of multiple identical individuals in a generation. Because using this filter in every iteration of the algorithm would not be beneficial in terms of the computing resources, this mutation filter is applied in both stages of the GA once in an interval of five generations. Identical individual eliminator mutation compares all individual's bits with each other. If two identical individuals are found, filter mutates one of the individual's bits

with a predefined probability, which is around one to five percent. Pseudocode of this mutation filter is shown in Figure 3.22.

```
If the generation number is exact multiple of iiemf interval;
  For i=(elitist selection quota):(population size);
    For j=i:(population size);
      If i-th and j-th individuals is exactly identical;
        For k=1:(individual string length);
          Create a random number between (0,1).
          If the random number < mutation probability for iiemf;
            Mutate k-th bit of j-th individual.
```

Figure 3.22: Pseudocode of the identical individual eliminator mutation filter.

3.3.9 Adaptation of parameters

There are two important factors for the genetic algorithms, which are the convergence speed and the diversity. These two factors influence GAs on a large scale. Genetic algorithms efficiency becomes maximum when the convergence speed and the diversity are optimally balanced.

Increasing the diversity of the population increases robustness of the GA. With a diverse population, GAs are less likely to be trapped in a local minimum. However, increasing the diversity also decreases the convergence speed. Decreasing the diversity produces higher convergence speeds, while increasing the chance that the algorithm falls in a local minimum. This phenomenon reveals the need for an optimization in the algorithms parameters.

At the early stages of a GA, diversity is rich enough to allow high selection pressures. Therefore, using relatively high selection pressure values produces fast convergence rates. Because the population is quite diverse, mutation probabilities can be kept at a small value. However, as the algorithm converges, diversity in the population is dramatically reduced. Thus, in the later generations of the algorithm, selection pressure should be decreased to help increasing the diversity of the population. Higher mutation probabilities also help to enrich the diversity of the population. This different parameter requirements of different stages of the genetic algorithms rises the need for the adaptation of the parameters. Therefore, an adaptive parameter control method is developed in this thesis. In the GA for the image

reconstruction problem; selection pressure, mutation probability and crossover probability parameters are adaptively controlled. Adaptive control system uses diversity of the population as input and calculates the selection pressure, the crossover probability and the mutation probabilities adaptively to keep the diversity at efficient levels. Adaptation of the parameters requires a numerical representation of the diversity of the population. Standard deviation approach is used in representing the diversity among the population as shown in Equation 3.13.

$$\lambda = \sqrt{\frac{1}{R}\sum_{n=1}^{R}\left(\varphi(n) - \varphi_{mean}\right)^2} \qquad \textbf{(3.13)}$$

Where λ is the diversity of the population, $\varphi(n)$ is the error value of the n-th individual, and φ_{mean} is the mean value of the error values of all individuals in the population. Standard deviation is an effective criterion for measuring the diversity of the population, as it is used for similar purposes in statistics. Adaptation of parameters brings additional robustness to the genetic algorithm and gives the algorithm the ability to perform well on changing situations.

3.3.9.1 Adaptation of selection pressure

Selection pressure parameter, which controls the convergence speed of the algorithm, has an adverse effect on the diversity. Convergence is achieved in the algorithm by consuming the diversity of the population. Therefore, when the diversity is rich among the population, high selection pressure is applied to the algorithm to increase the convergence speed. However, when diversity drops below a certain limit, selection pressure is reduced. This situation often results in oscillation of the diversity. This oscillation behavior helps the convergence of the algorithm. Diversity is very high at the start of the algorithm, thus the selection pressure is at its maximum value. After the early generations, population enters a steady state where the diversity is controlled with the selection pressure and the mutation probabilities. In this state, when the diversity bears to the upper limit of the oscillation band, selection pressure increases and the mutation probability decreases to achieve convergence. When the diversity falls to the lower limit, the selection pressure decreases and the mutation probability increases to enrich the diversity. To improve the efficiency of the algorithm, oscillation of the diversity should be kept in an optimal band.

Selection pressure parameter is adapted using an exponential function of the diversity.

$$\beta = \upsilon_1 - \upsilon_2 \; e^{-\lambda} \qquad \textbf{(3.14)}$$

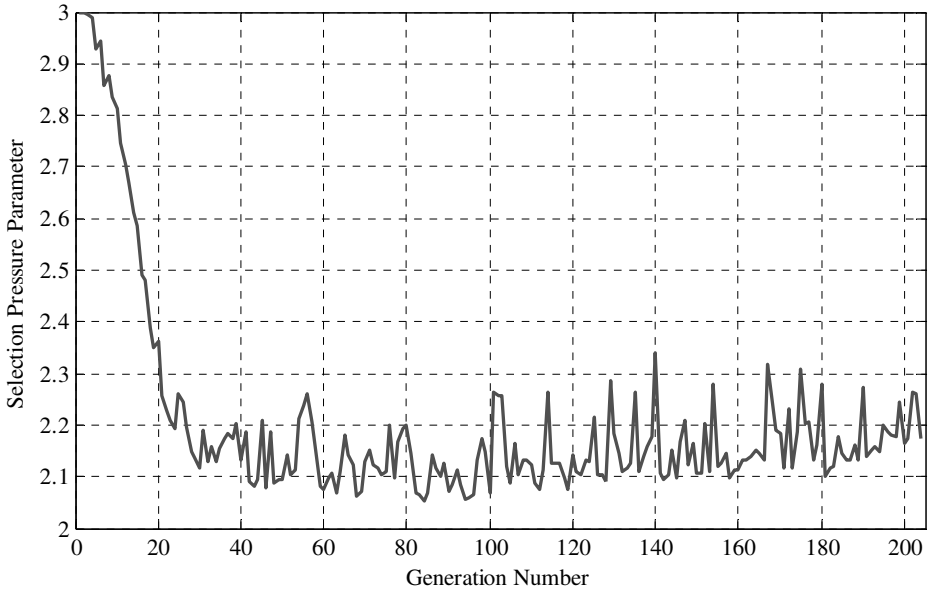
Where β represents the selection pressure parameter, υ_1 represents the base selection pressure factor, υ_2 represents the selection pressure band factor and λ represents the diversity. The base selection pressure and the selection pressure band factors define the way that the selection pressure parameter changes. The base selection pressure factor determines the highest value that the selection pressure can take during the algorithm. It should be set as the highest safe value of the selection pressure parameter without losing efficiency of the algorithm. The selection pressure band factor determines the interval that the selection pressure can vary between. It restricts the adaptation to prevent the selection pressure from decreasing below a lower limit, which may result in losing the convergence. Experimentally, values around three for υ_1 and values around one for υ_2 are observed to be optimal.

GA is executed to see the effects of adaptation of parameters using the 32-electrode model. Variation of selection pressure parameter and diversity is plotted versus generation number in Figure 3.23 (a) and (b) respectively. The diversity at the start of the algorithm is very high because of the random creation of initial population; therefore, applied selection pressure is also very high. As the convergence is achieved, the diversity falls very sharply and it is followed by the selection pressure parameter. After the algorithm reaches the steady state operation, diversity is controlled in a band by the selection pressure parameter and the mutation probability.

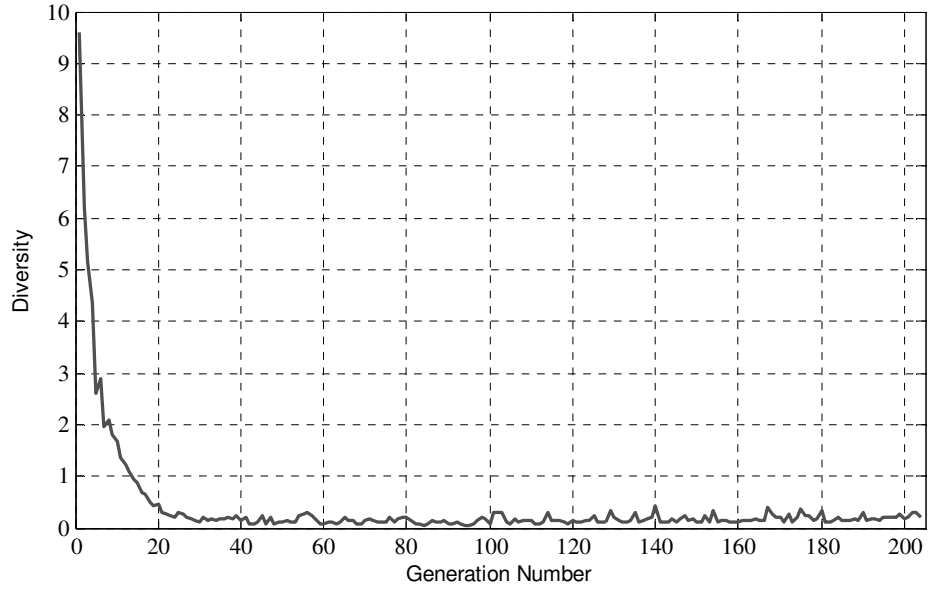
3.3.9.2 Adaptation of crossover probability

An ideal crossover probability should change with the diversity, decreasing slightly with the reducing diversity. At the early generations of the algorithm, where the mixture rate is very important, using crossover probabilities near one is ideal. However, as the algorithm achieves progress, decreasing the crossover probability at small scale slightly improves the results of the algorithm.

$$P_c = 1 - \theta \; e^{-\lambda} \qquad \textbf{(3.15)}$$



(a)



(b)

Figure 3.23: Variation of parameters versus generation number for 32-electrode model: **(a)** Variation of selection pressure parameter. **(b)** Variation of diversity.

Where P_c is the crossover probability, θ is the crossover probability adaptation band and λ is the diversity. The crossover probability adaptation band determines the change interval for the crossover probability. Highest value for crossover probability is always one and the adaptation controls the crossover probability value inside the crossover adaptation band. Values around 0.1 are observed to be suitable for the crossover probability band. Variation of crossover probability versus generation

number is plotted for 32-electrode model in Figure 3.24. Corresponding variation of the diversity plot can be seen in Figure 3.23 (b).

3.3.9.3 Adaptation of mutation probability

Mutation is an important tool to maintain the diversity in the genetic algorithms, especially at the later stages. Mutation probability should ideally increase with the reducing diversity in the population. As mentioned earlier, mutation probability plays a very important role to maintain the diversity in cooperation with selection pressure.

The minimum and the maximum mutation probability values are adaptively calculated by using the statements in Equations (3.16) and (3.17).

$$P_{\min} = \tau_1 \, e^{-\lambda} \tag{3.16}$$

$$P_{\max} = \tau_2 \, e^{-\lambda} \tag{3.17}$$

Where $P_{\min}$ and $P_{\max}$ is the minimum and the maximum mutation probabilities respectively, τ_1 and τ_2 are the mutation adaptation factors and λ is the diversity. Variation of the mutation probabilities versus generation number is plotted in Figure 3.25 (a) and (b). Corresponding variation of the diversity plot is in Figure 3.23 (b).

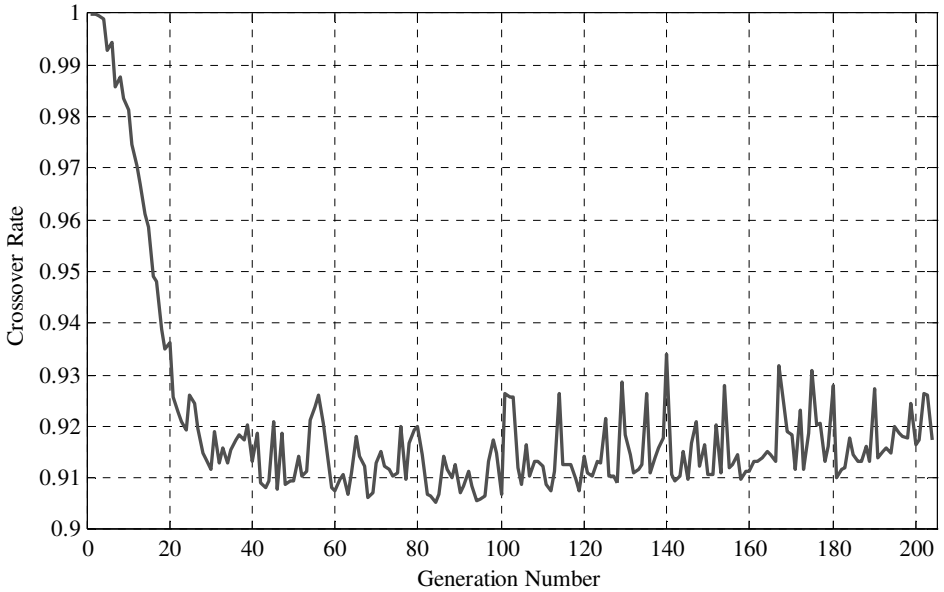
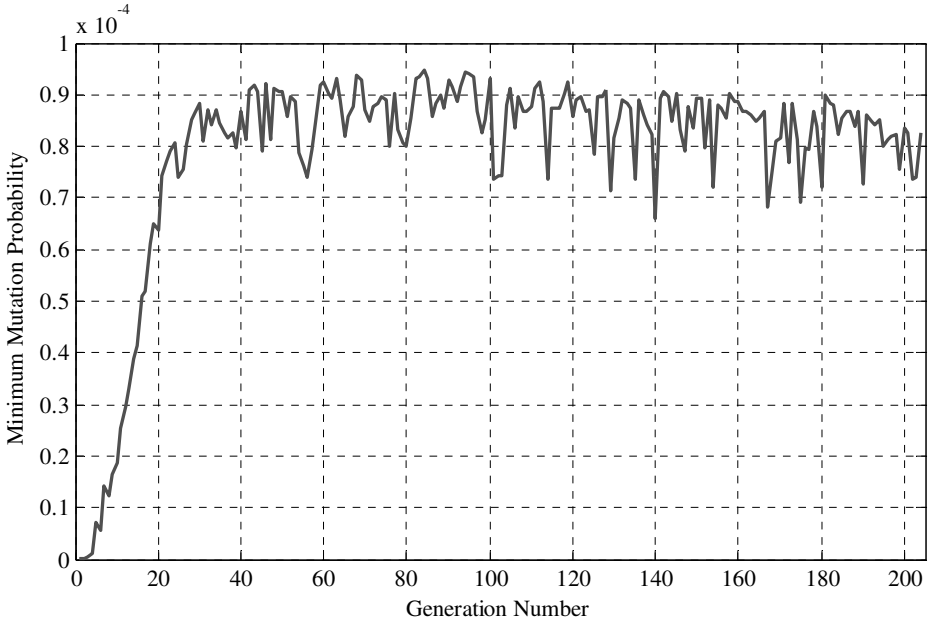
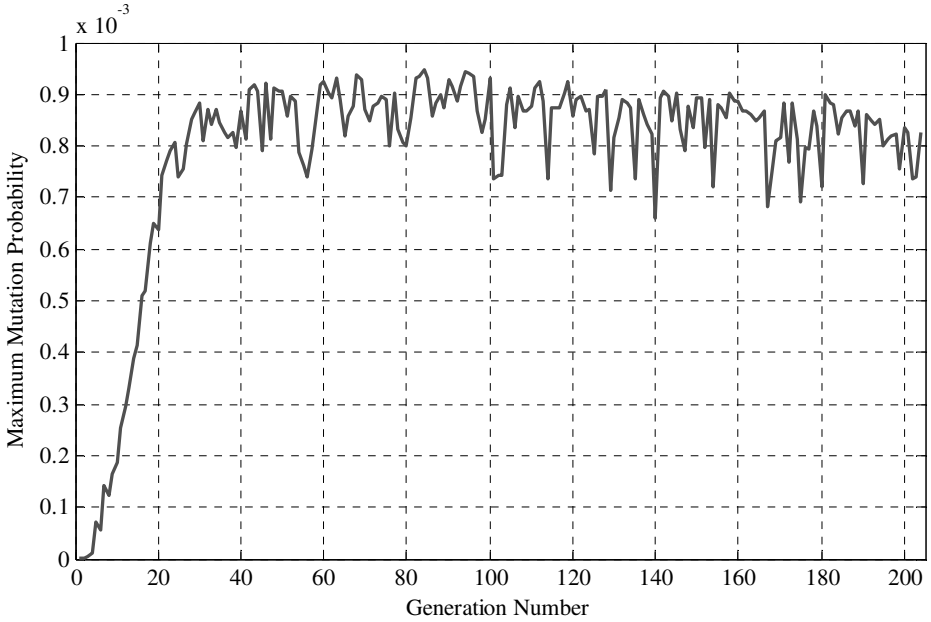


Figure 3.24: Variation of crossover probability versus generation number for 32-electrode model.



(a)



(b)

Figure 3.25: Variation of the mutation probabilities versus generation number for 32-electrode model: **(a)** Variation of the minimum mutation probability. **(b)** Variation of the maximum mutation probability.

4. NUMERICAL SIMULATIONS

Purpose of the fourth chapter is to demonstrate the performance of the genetic algorithm for the image reconstruction problem. A series of experiments is carried out to test the efficiency of the GA. Measurement stage of these experiments are simulated using the FEM model, which is introduced in the second chapter, and to reach the actual conductivity distribution of the body, the GA uses the synthetic data that are produced by this model. GA parameters used in the tests are shown in table 4.1. GA is tested with the data from the numerical simulation that is produced using distinctive conductivity distributions that examine various properties of the GA.

Table 4.1: Genetic algorithm parameters used in the tests.

Genetic algorithm Parameter	16 Electrode Model	32 Electrode Model
Population size	200	250
Generation limit	500	2000
Maximum acceptable error	10^{-3}	10^{-4}
Successive iterations criterion	200	300
Second stage criterion	15	20
Background conductivity	1	1
Foreground conductivity	0.1	0.1
Weight function parameter	10	10
Selection pressure base	3	3
Selection pressure band	1	1
Elitist quota for first stage	2	2
Elitist quota for second stage	4	4
Crossover probability band	0.1	0.1
Minimum mutation factor	0.0001	0.0001
Maximum mutation factor	0.01	0.001
Neighborhood mutation probability for individuals	0.2	0.2
Neighborhood mutation probability for pixels	0.1	0.1
Center fill mutation prob.	0.1	0.1

First part of the fourth chapter consists of the experiments that are conducted using the 16-electrode model. Second part covers the experiments that utilize the 32-

electrode model. Third part consists of the experiments that are conducted by using data with additive synthetic noise to test the algorithms efficiency with noisy data.

4.1 Results for 16-Electrode Model

First test that is conducted on the 16-electrode model is the moving object test, which is the series of experiments that a target object is moved to a different region in the conductivity distribution of the body for each single experiment. Aim of this experiment is to test the algorithm's ability to detect objects in different regions of the body. Figures 4.1 - 4.5 shows the results of the experiment for a plus-shaped foreground object that is placed in the left, up, right, down and center of the conductivity distribution respectively. In the figures, the image on the left side, which is labeled as the actual distribution, is the true conductivity distribution that is used in the simulation of the measurement process; while the image on the right side, which is labeled as the calculated distribution, is the result that is obtained by the genetic algorithm. In the case of an exact result, two images must be identical. From Figures 4.1 - 4.5, we can see that the algorithm attains the true conductivity distribution, regardless of where the object is located on the conductivity distribution.

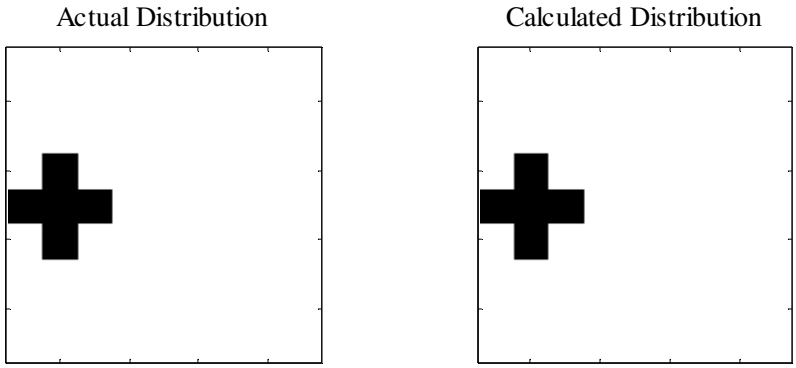


Figure 4.1: Comparison of the actual conductivity distribution and the result of the genetic algorithm for an object located on the left region of the body.

Conductivity distributions that include numerous objects that stand close to each other are often very difficult to reconstruct correctly in impedance imaging method. These multiple objects are sometimes detected incorrectly as a single object. To test the algorithms ability to detect objects that are close to each other correctly, a second test was conducted using a conductivity distribution that includes two objects that stands close to each other. After the simulation, the genetic algorithm was used to

reconstruct the conductivity distribution of the body. Comparison of the actual conductivity distribution and the result of the GA is shown in Figure 4.6. As seen in Figure 4.6, exact result was achieved by the GA. It took 46 seconds for the algorithm to reconstruct the conductivity distribution of the body.

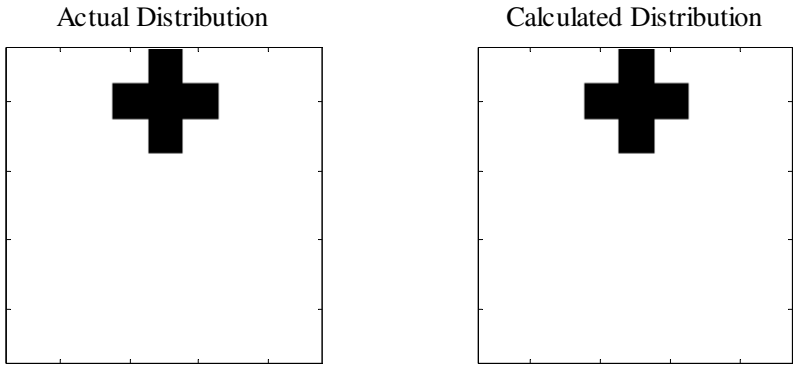


Figure 4.2: Comparison of the actual conductivity distribution and the result of the genetic algorithm for an object located on the upper region of the body.

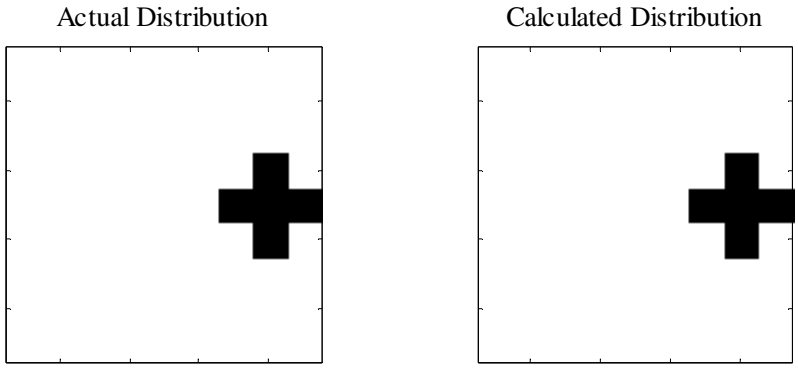


Figure 4.3: Comparison of the actual conductivity distribution and the result of the genetic algorithm for an object located on the right region of the body.

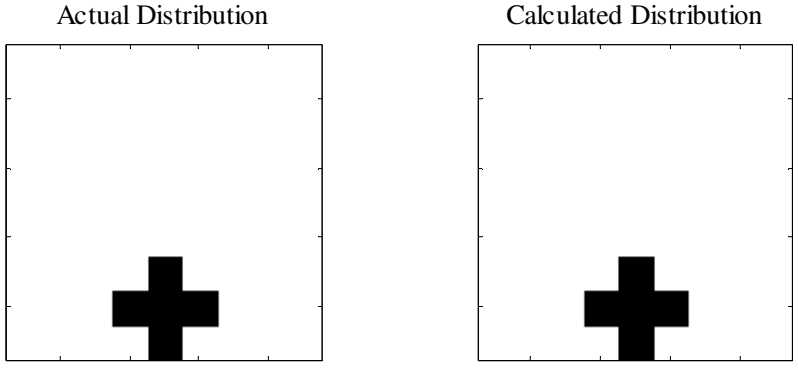


Figure 4.4: Comparison of the actual conductivity distribution and the result of the genetic algorithm for an object located on the lower region of the body.

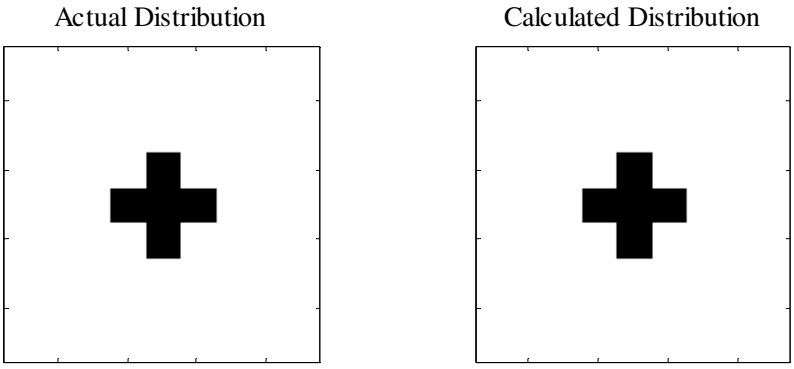


Figure 4.5: Comparison of the actual conductivity distribution and the result of the genetic algorithm for an object located on the center of the body.

In Figure 4.7, the error values of the best individuals of each generation are plotted versus generation number. Diversity of the population versus generation number is shown in Figure 4.8.

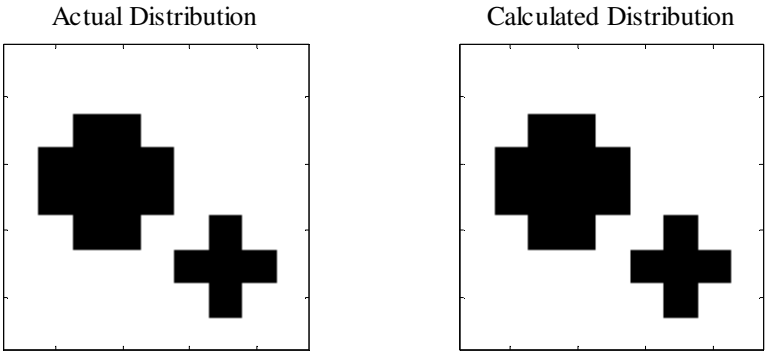


Figure 4.6: Comparison of the actual conductivity distribution and the result of the genetic algorithm for the second test of the 16-electrode model.

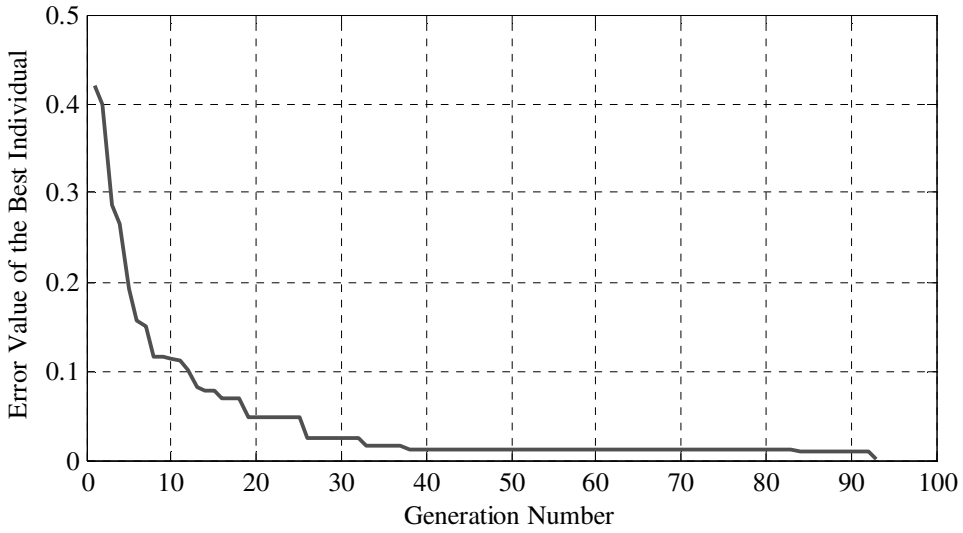


Figure 4.7: Error values of the best individuals of each generation versus generation number for second test of 16-electrode model.

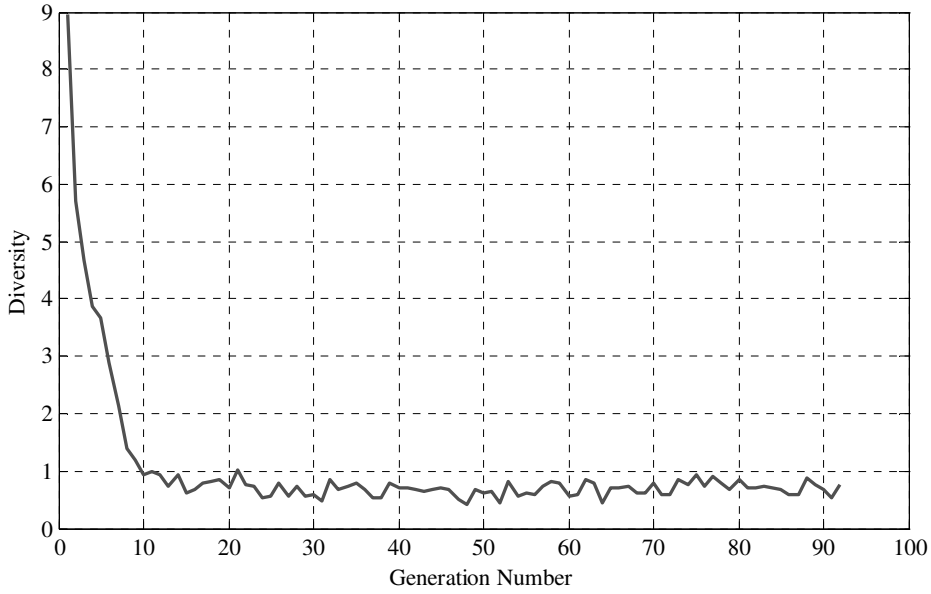


Figure 4.8: Diversity versus generation number for the second test of 16-electrode model.

To demonstrate the convergence steps of the GA, best individuals of the selected generations of the reconstruction process are shown in Figure 4.9 for the conductivity distribution that is calculated in the second test of the 16-electrode model. Figure 4.9 (a) shows the best individual of the first generation, which is randomly generated. Figure 4.9 (b), (c), (d) and (e) are the best individuals of the generations number ten, twenty, thirty and forty respectively. Figure 4.9 (f) shows the best individual of the last generation, which is the result of the algorithm. In Figure 4.9, it can be seen that the convergence is achieved by evolving the candidate solutions of the population and continuously improving the resemblance of the individuals to the actual distribution until the exact result is reached.

The third experiment was carried out by using a conductivity distribution of ten foreground pixels that are randomly positioned on the body. Conductivity distribution of the body was reconstructed using the genetic algorithm. Comparison of the actual conductivity distribution and the resulted conductivity distribution from the GA is shown in Figure 4.10. As seen in Figure 4.10, exact result was attained by the GA. During the image reconstruction process, the GA consumed 103 seconds of computing time. Convergence of the algorithm is plotted in Figure 4.11 where the error values of the best individuals of each generation are plotted versus generation number. Diversity of the population versus generation number is plotted in Figure 4.12.

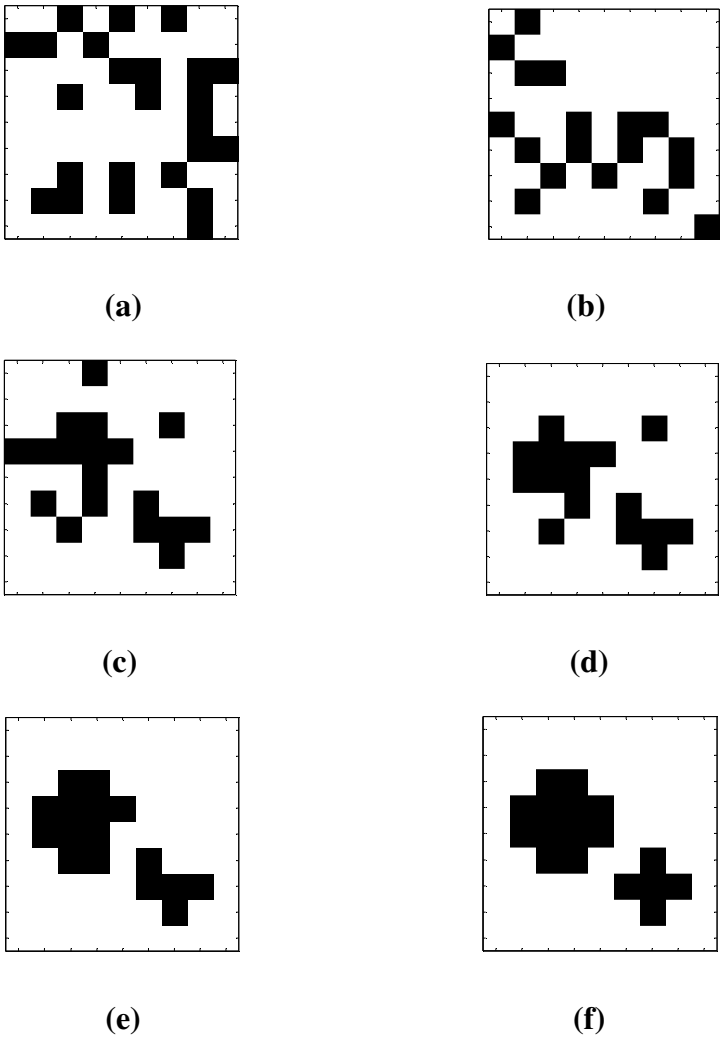


Figure 4.9: Best individuals of the selected generations of the reconstruction process for the second test of the 16-electrode model : **(a)** Best individual of the first generation. **(b)** Best individual of the 10th generation. **(c)** Best individual of the 20th generation. **(d)** Best individual of the 30th generation. **(e)** Best individual of the 40th generation. **(f)** Best individual of the last generation.

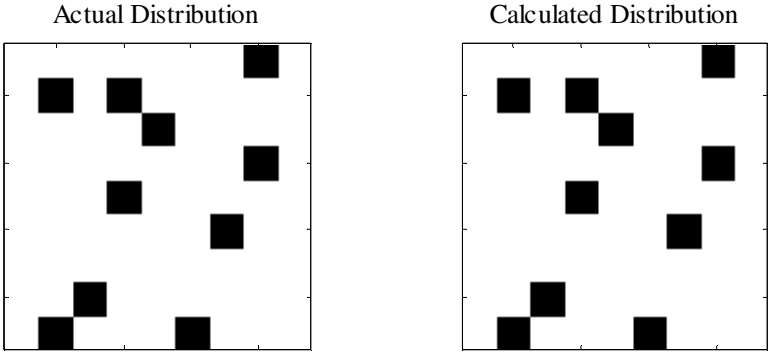


Figure 4.10: Comparison of the actual conductivity distribution and the result of the genetic algorithm for the third test of 16-electrode system.

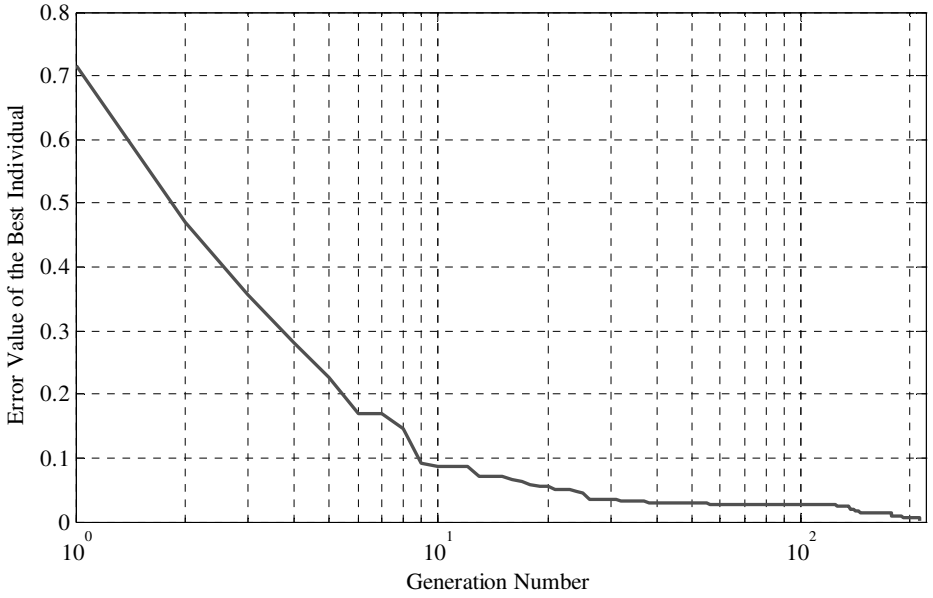


Figure 4.11: Error values of the best individuals of each generation versus generation number for the third test of 16-electrode model.

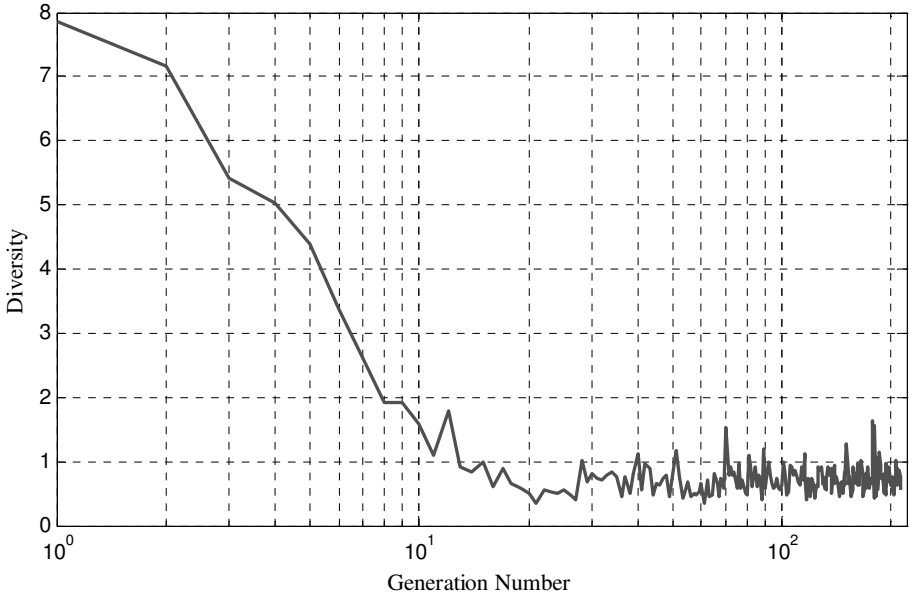


Figure 4.12: Diversity versus generation number for the third test of 16-electrode model.

4.2 Results for 32-Electrode Model

The aim of the first test of 32-electrode model is to observe the algorithm's ability to reconstruct large objects with exact details, which is a difficult task for image reconstruction algorithms in general because the details of large objects cause a small influence on the boundary electrodes. In this test, true conductivity distribution was

successfully reconstructed and the process lasted 772 seconds. Comparison of the actual conductivity distribution and the result of the genetic algorithm for the first test of 32-electrode model is shown in Figure 4.13. Error values of the best individuals of each generation and the diversity of the population are plotted versus generation number in Figures 4.14 and 4.15 respectively.

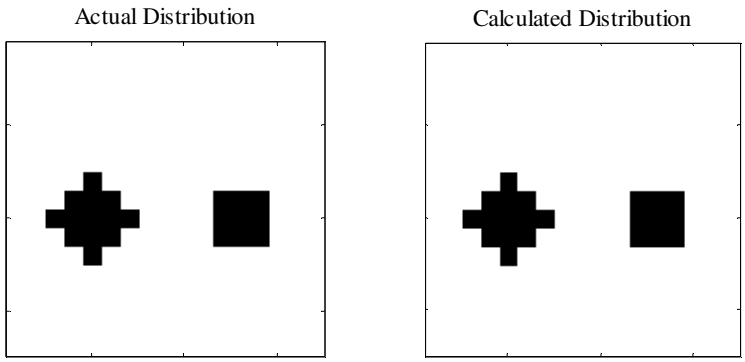


Figure 4.13: Comparison of the actual conductivity distribution and the result of the genetic algorithm of first test of 32-electrode model.

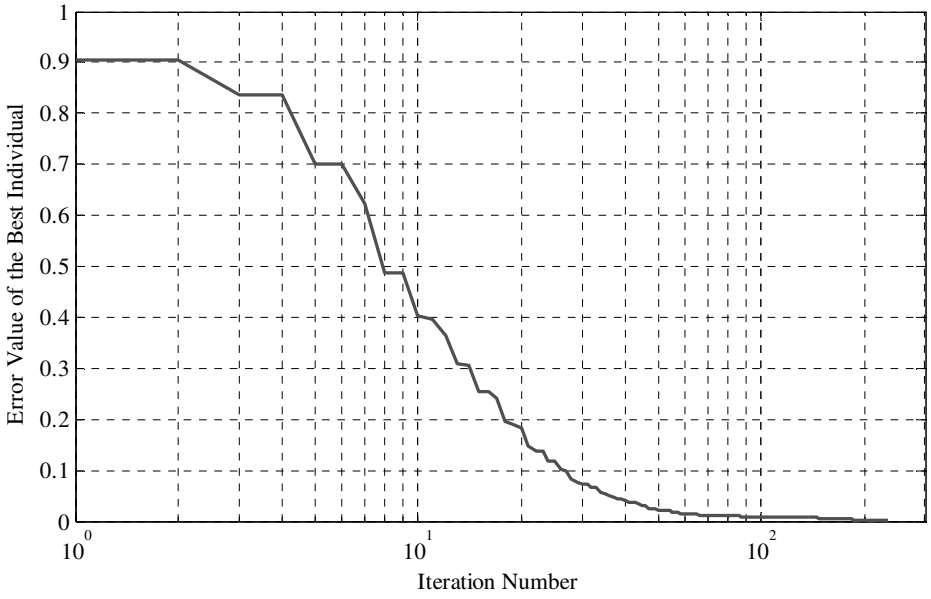


Figure 4.14: Error values of the best individuals of each generation versus generation number for the first test of 32-electrode model.

Best individuals of the selected generations are shown in Figure 4.16 to demonstrate the convergence steps of the image reconstruction process. Figures 4.9 (a), (b), (c), (d), (e) and (f) show the best individual of the first, 50th, 100th, 150th, 200th and the last generation respectively. It can be seen that the algorithm constantly improves the population of candidate solutions by increasing their resemblance to the true

conductivity distribution starting from the area near the boundary. Ill-conditioned nature of the problem causes the sensitivity of the central region of the conductivity distribution to drop. As a direct result of this phenomenon, the boundary region of the conductivity distribution is reconstructed during early iterations and the central region is reconstructed during late iterations of the solution process. Reconstruction of the boundary region also requires lesser iterations than reconstruction of the central region.

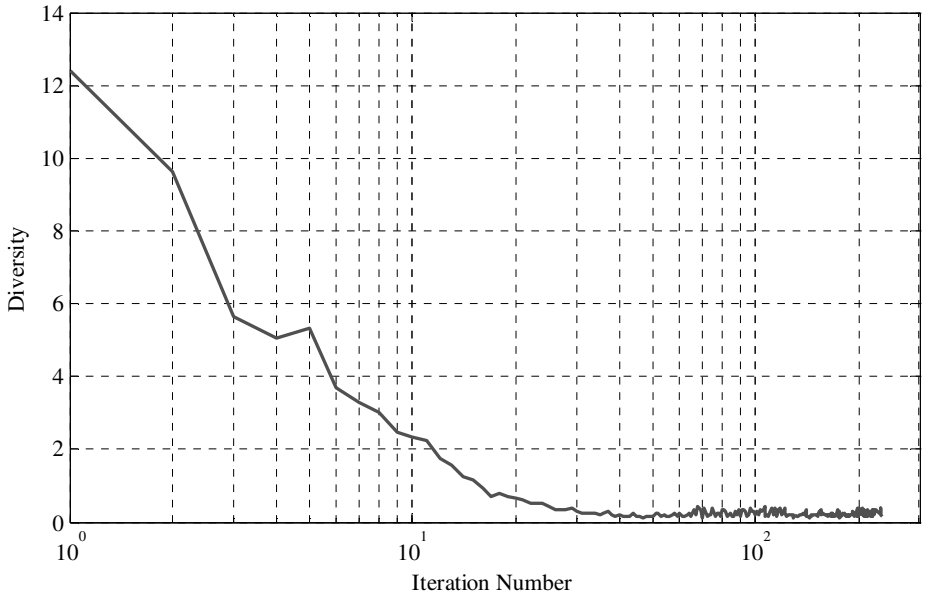


Figure 4.15: Diversity versus generation number for the first test of 32-electrode model.

As mentioned earlier, exact reconstruction of large objects is very difficult due to the reduced effect of each pixel of the object on the boundaries. Reconstruction of these target objects becomes increasingly difficult if it is located in the central region of the body because of the sensitivity drop of the pixels in the center of the conductivity distribution.

To test the algorithm's ability to reconstruct large and complex-shaped objects located in the central region of the conductivity distribution with exact details, another simulation is conducted using a conductivity distribution that consists of a large foreground object located in the central region of the homogeneous body. Comparison of the actual conductivity distribution and the resulted conductivity distribution from the genetic algorithm is shown in Figure 4.17.

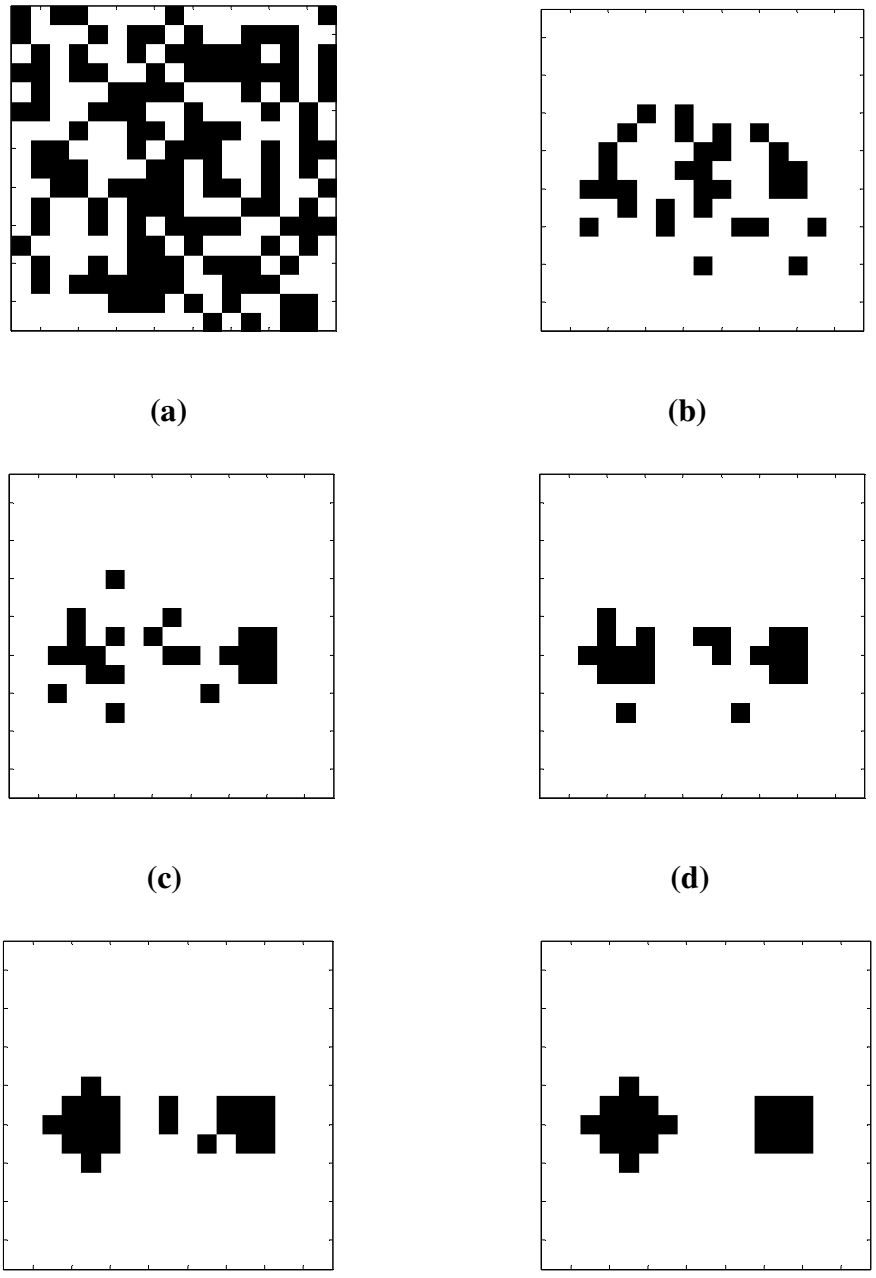


Figure 4.16: Best individuals of the selected generations of the reconstruction process for the first test of 32-electrode model: **(a)** Best individual of the first generation. **(b)** Best individual of the 50th generation. **(c)** Best individual of the 100th generation. **(d)** Best individual of the 150th generation. **(e)** Best individual of the 200th generation. **(f)** Best individual of the last generation.

As observed in Figure 17, the genetic algorithm reconstructed the exact conductivity distribution successfully in 1038 seconds. The error values of the best individuals of each generation are plotted versus generation number in Figure 4.18. Diversity of the population versus generation number is plotted in Figure 4.19.

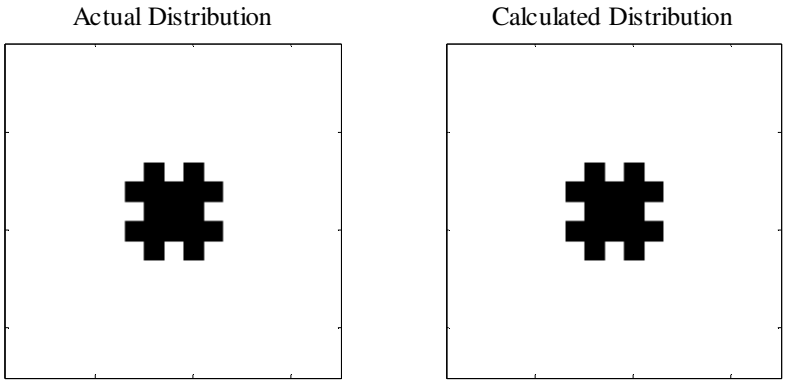


Figure 4.17: Comparison of the actual conductivity distribution and the result of the genetic algorithm for the second test of 32-electrode system.

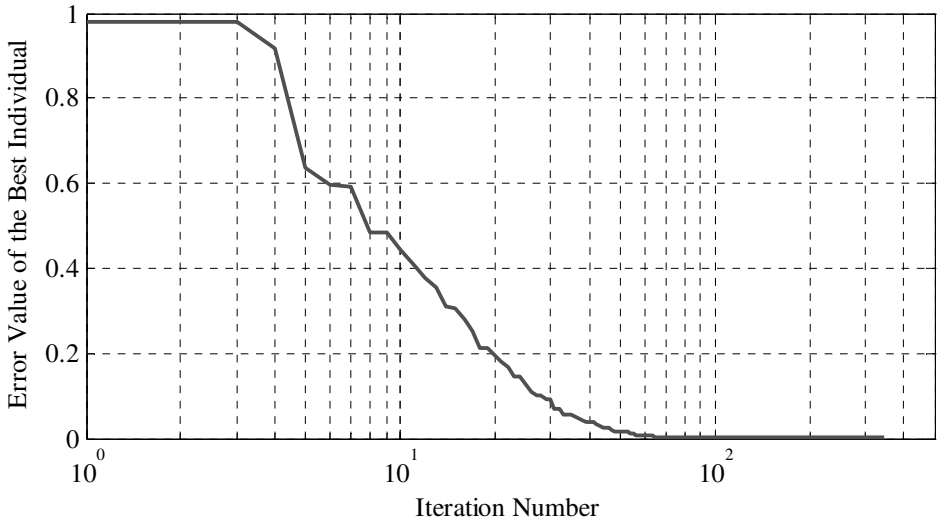


Figure 4.18: Error values of the best individuals of each generation versus generation number for the second test of 32-electrode model.

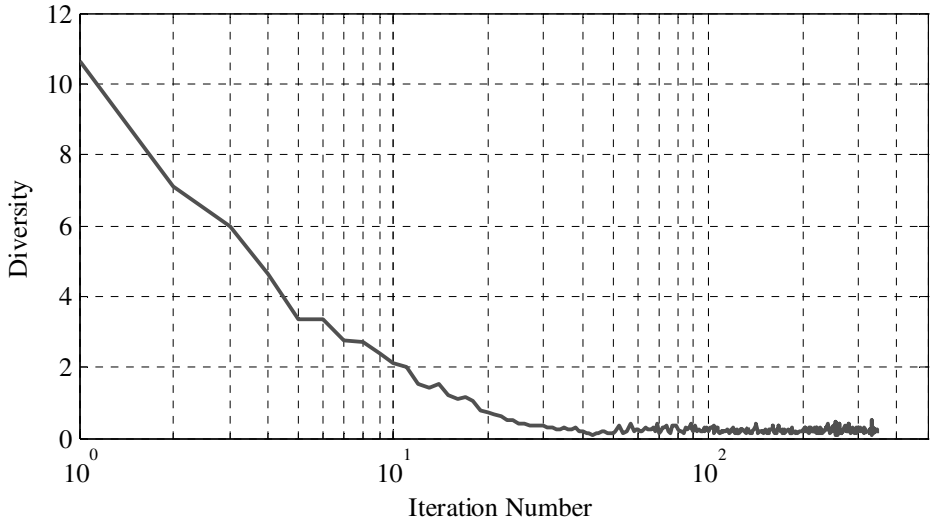


Figure 4.19: Diversity versus generation number for the second test of 32-electrode model.

The third test of the 32-electrode model was conducted using a conductivity distribution contains two small foreground target objects that are placed on the homogeneous background. Comparison of the actual conductivity distribution and the resulted conductivity distribution from the genetic algorithm is shown in Figure 4.20. Exact result was attained successfully in 720 seconds. The error values of the best individuals of each generation and the diversity of the population are plotted versus generation number in Figures 4.21 and 4.22 respectively.

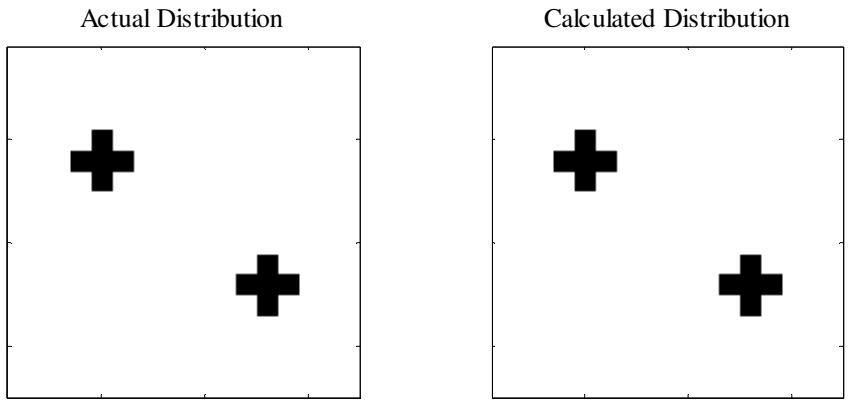


Figure 4.20: Comparison of the actual conductivity distribution and the result of the genetic algorithm for the third test of 32-electrode system.

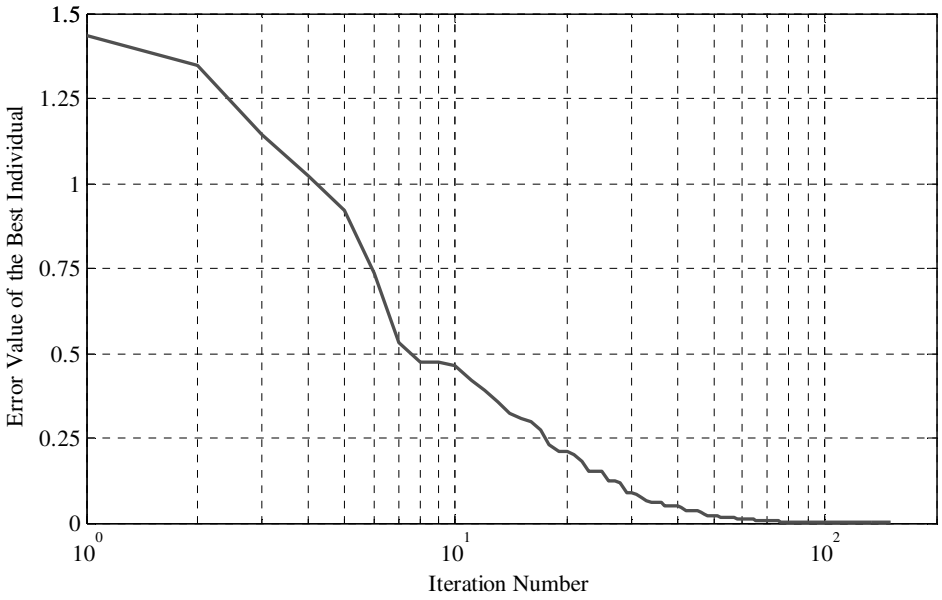


Figure 4.21: Error values of the best individuals of each generation versus generation number for the third test of 32-electrode model.

The most difficult conductivity distributions for the image reconstruction problem are the distributions that include numerous small objects spread to the entire body.

Small objects cause lesser impact on the electrodes at the boundary, reducing the sensitivity of the pixels located in the central region of the body. There is also a blocking problem with the conductivity distributions that contain multiple objects that resides close to each other when the path of the electrical current flowing from the electrodes to an object is blocked by another object; therefore, preventing the useful data from being collected.

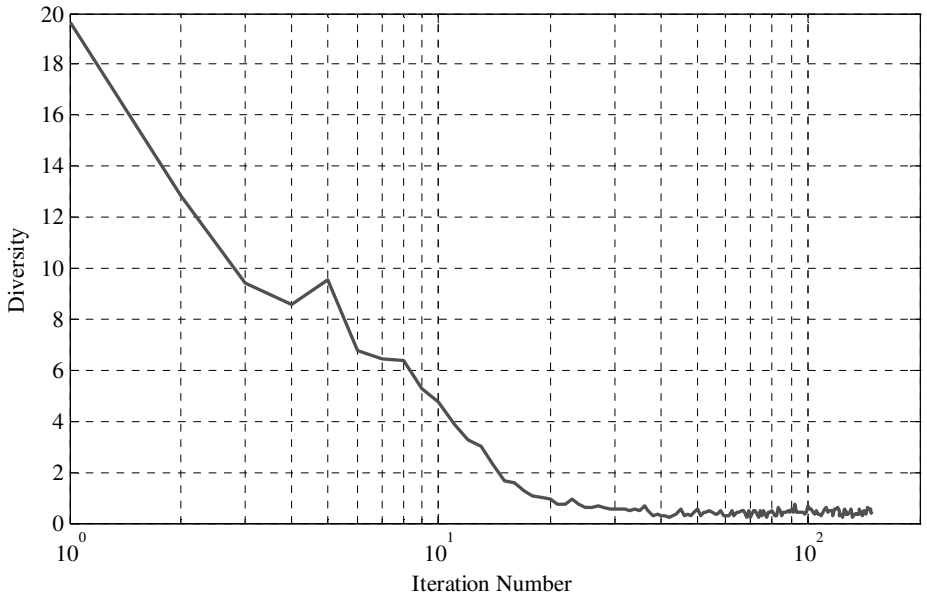


Figure 4.22: Diversity versus generation number for the third test of 32-electrode model.

The ill-conditioning nature of the problem may even rise to a point where the exact image reconstruction is impossible because the data provided to the genetic algorithm is insufficient. To demonstrate the performance of the GA in extremely ill-conditioned situations, two simulations were conducted with a conductivity distribution that contain ten and twenty random foreground pixels. Next, GA reconstructed the conductivity distributions using the data from the simulations. Comparison of the actual conductivity distribution and the resulted conductivity distribution from the GA for the distribution of ten foreground pixels is shown in Figure 4.23.

The exact conductivity distribution was successfully obtained in 399 seconds of computing time. Error values of the best individuals of each generation and the diversity of the population are plotted versus generation number in Figures 4.24 and 4.25 respectively.

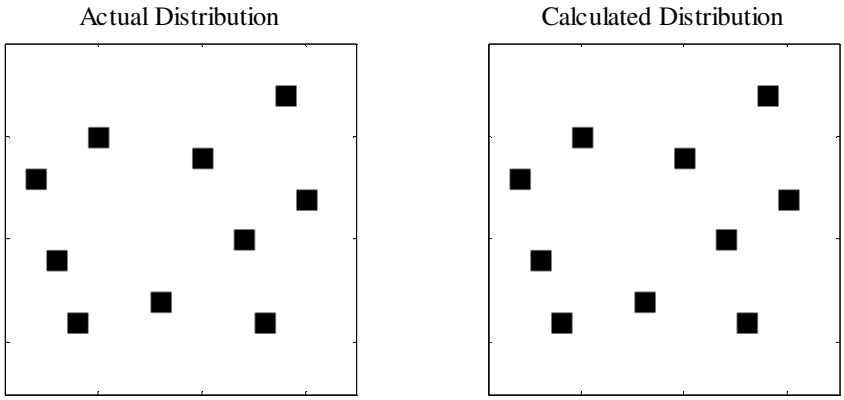


Figure 4.23: Comparison of the actual conductivity distribution and the result of the genetic algorithm for the fourth test of 32-electrode system.

The fifth test of the 32-electrode model was carried out by using a conductivity distribution of twenty foreground pixels that are randomly positioned on the homogeneous background. Comparison of the actual conductivity distribution and the resulted conductivity distribution from the genetic algorithm is shown in Figure 4.26. Because no improvement was achieved in the last two hundred generations, the end criterion of the genetic algorithm was met and the algorithm ended without reaching the exact result. A similar conductivity distribution was obtained in 2173 seconds. The error values of the best individuals of each generation and the diversity of the population are plotted versus generation number in Figures 4.27 and 4.28 respectively.

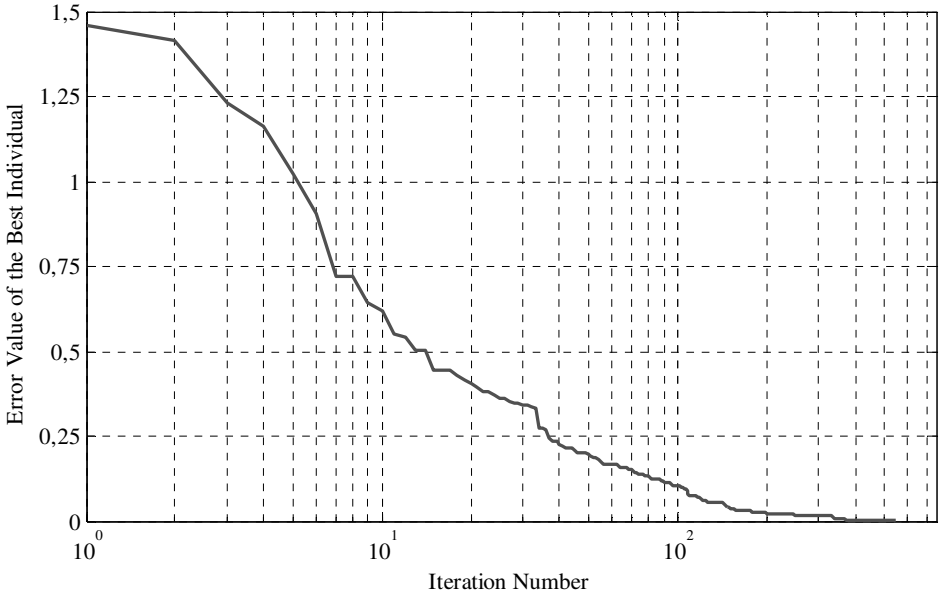


Figure 4.24: Error values of the best individuals of each generation versus generation number for the fourth test of 32-electrode model.

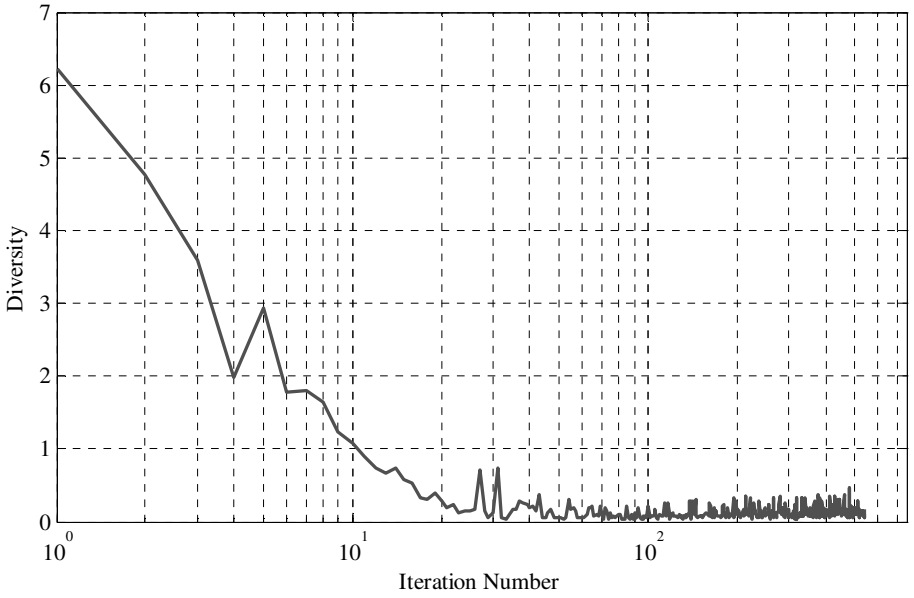


Figure 4.25: Diversity versus generation number for the fourth test of 32-electrode model.

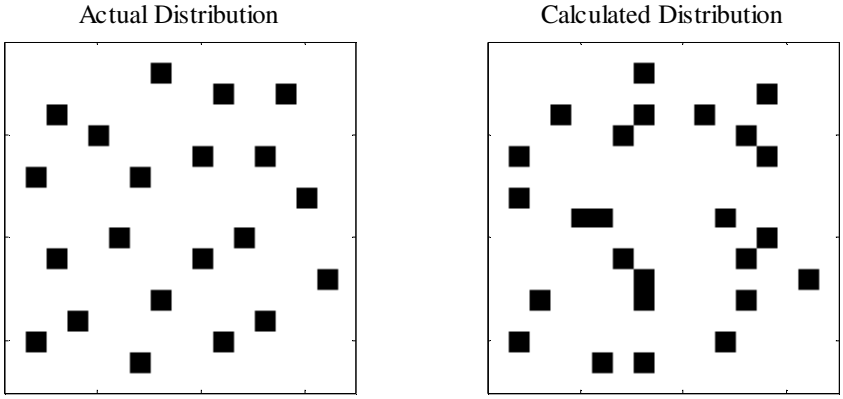


Figure 4.26: Comparison of the actual conductivity distribution and the result of the genetic algorithm for the fifth test of 32-electrode system.

The reason that the algorithm has failed to reconstruct the true conductivity distribution for twenty small foreground pixels is the sensitivity drop in the central area of the body. As the number of the objects in the conductivity distribution increases, sensitivity of each object decreases, especially for the objects that are located in the center of the conductivity distribution. Combined with the blocking problem mentioned before, reduced sensitivity increases the ill conditioning of the problem dramatically. However, despite the ill-conditioned nature of the problem, genetic algorithm successfully converges near the optimal solution where the exact result cannot be obtained, demonstrating the effectiveness of the genetic algorithm.

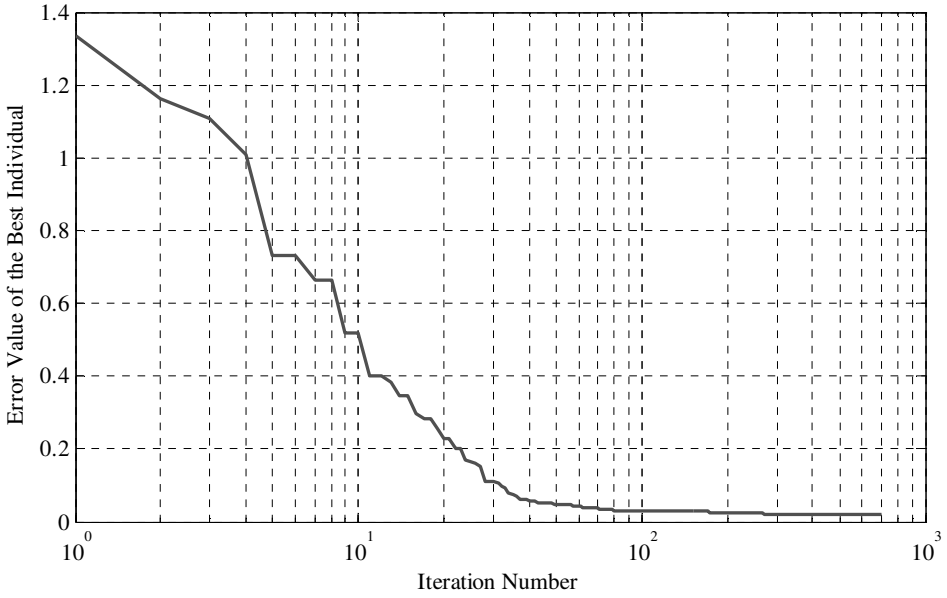


Figure 4.27: Error values of the best individuals of each generation versus generation number for the fifth test of 32-electrode model.

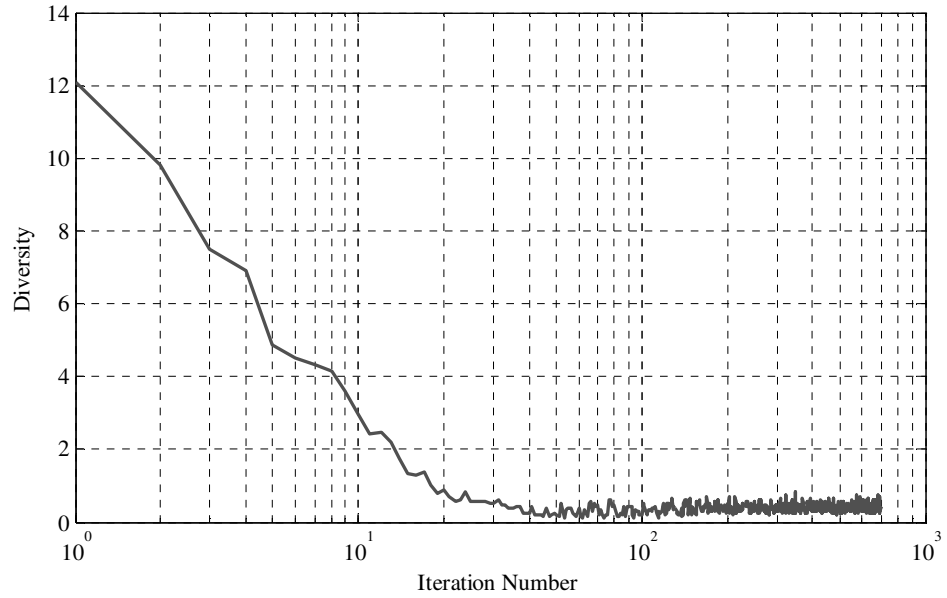


Figure 4.28: Diversity versus generation number for the fifth test of 32-electrode model.

As mentioned before, conductivity distributions that contain small objects located near a large object are very difficult to reconstruct exactly in electrical impedance imaging method, often incorrectly detecting these objects as a single object. Therefore, a test was conducted to analyze the genetic algorithms performance with conductivity distributions that contain large and small objects located closer to each other.

In the sixth test of the 32-electrode model, the algorithm was used to reconstruct a conductivity distribution with a large square object in the center and four small pixels reside close the large object. The GA successfully reconstructed the exact conductivity distribution in 684 seconds. Comparison of the actual conductivity distribution and the resulted conductivity distribution from the GA is shown in Figure 4.29.

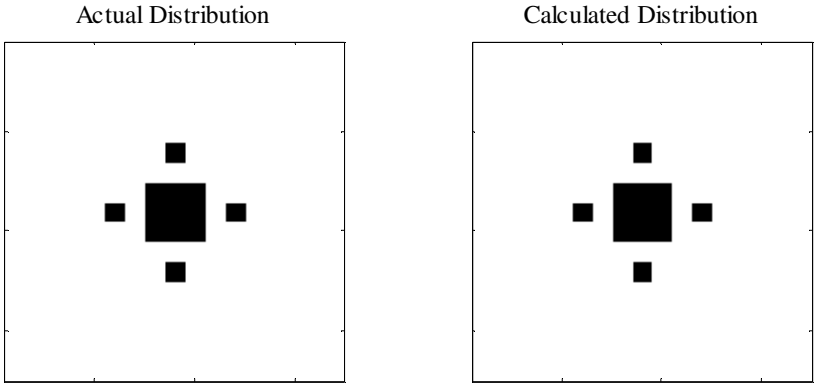


Figure 4.29: Comparison of the actual conductivity distribution and the result of the genetic algorithm for the sixth test of 32-electrode system.

Error values of the best individuals of each generation and the diversity of the population are plotted versus generation number in Figures 4.30 and 4.31 respectively.

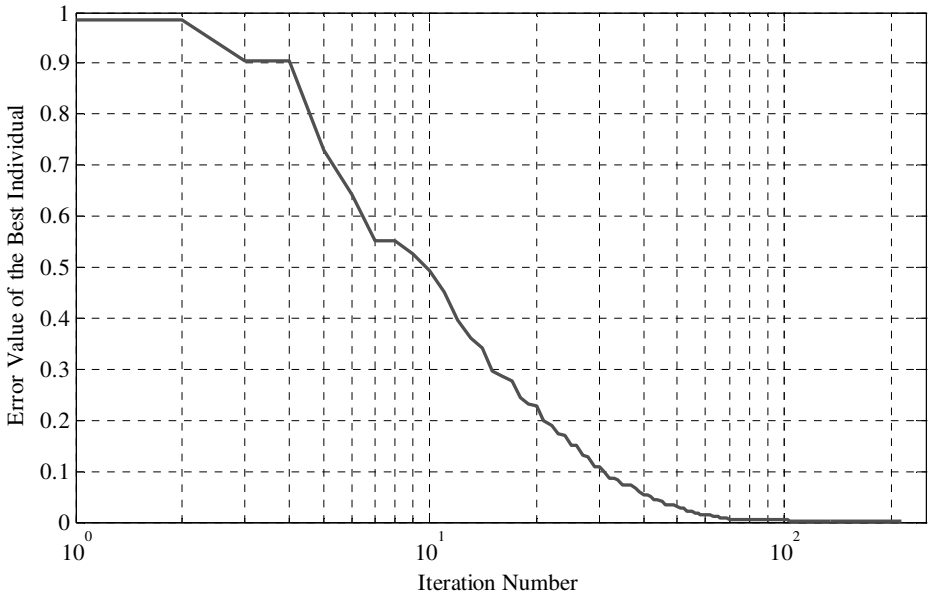


Figure 4.30: Error values of the best individuals of each generation versus generation number for the sixth test of 32-electrode model.

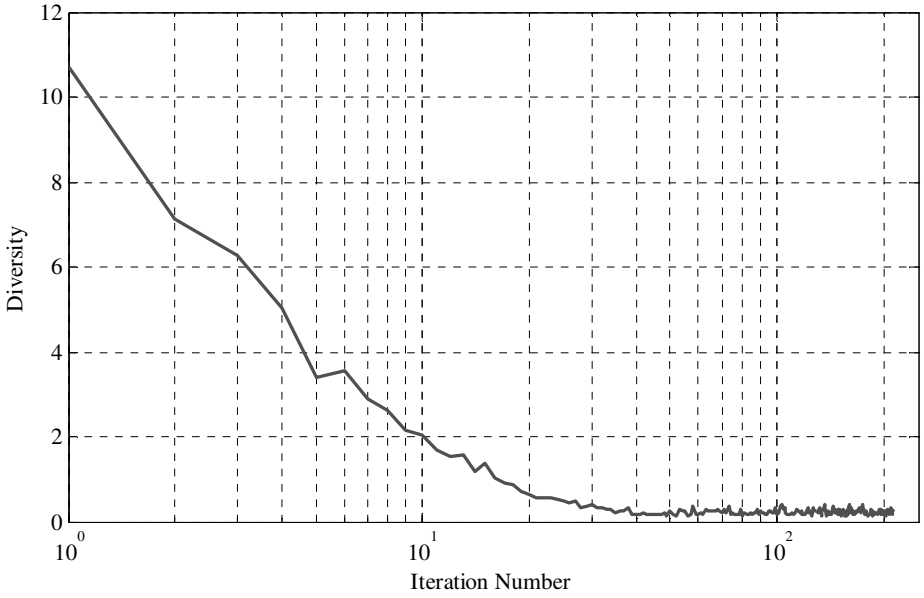


Figure 4.31: Diversity versus generation number for the sixth test of 32-electrode model.

4.3 Effects of Noise

Because the image reconstruction problem of EII method is an extremely ill-conditioned problem, noise affects the performance of the reconstruction algorithm dramatically. Presence of noise further reduces the sensitivity of each pixel in the conductivity distribution, raising the difficulty of the image reconstruction process. Noise on the measurement data may even prevent the exact image reconstruction if the noise dominates some important information in the data. To test the genetic algorithms performance with the noisy data, a series of numerical simulations is conducted using a fixed conductivity distribution, adding Gaussian white noise to the data using different standard deviation value for each test. After the simulations, the GA is subjected to reconstruct the conductivity distributions using the data from the simulations with additive white Gaussian noise.

Comparison of the actual conductivity distribution and the resulted conductivity distribution from the GA for uncontrolled noise test using Gaussian white noise with standard deviation of 0.0001 volts is shown in Figure 4.32 and the error values of the best individuals of each generation are plotted versus generation number in Figure 4.33. The algorithm successfully attained the true conductivity distribution, which shows that the algorithm can handle noise with the standard deviation of 0.0001 volts.

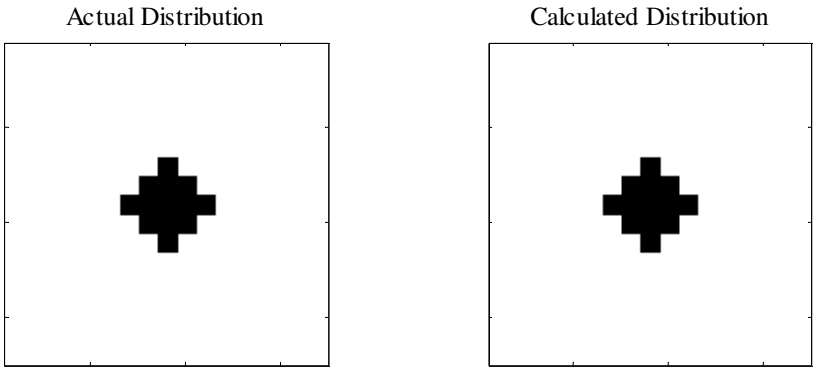


Figure 4.32: Comparison of the actual conductivity distribution and the result of the genetic algorithm for uncontrolled noise test using Gaussian white noise with standard deviation of 0.0001 volts.

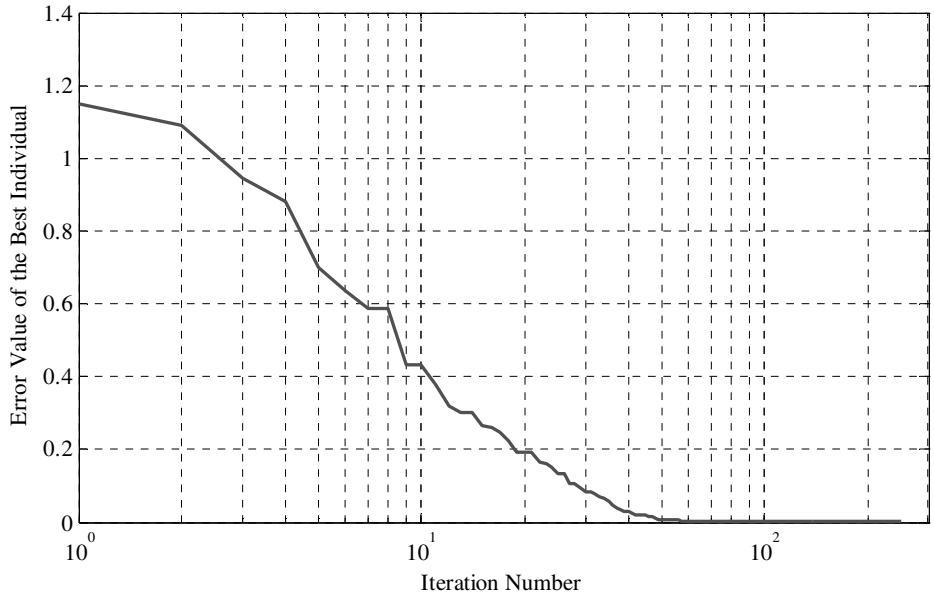


Figure 4.33: Error values of the best individuals of each generation versus generation number for uncontrolled noise test using Gaussian white noise with standard deviation of 0.0001 volts.

In Figure 4.34, comparison of the actual conductivity distribution and the result of the genetic algorithm for uncontrolled noise test using Gaussian white noise with standard deviation of 0.001 volts is shown. The error values of the best individuals of each generation are plotted versus generation number in Figure 4.35. From Figure 4.34, it is seen that the GA failed to reach the true conductivity distribution using the data with Gaussian white noise with the standard deviation of 0.001 volts; however, a very similar conductivity distribution was achieved. The voltage response on the boundary electrodes caused by the pixels located near the center of the conductivity distribution is lower than the voltage response caused by the pixels located near the

boundary. Because of this situation, central region of the conductivity distribution is more sensitive to the presence of noise than the boundary region. Therefore, errors in the reconstructed image first appear in the central region of the conductivity distribution with increasing noise levels.

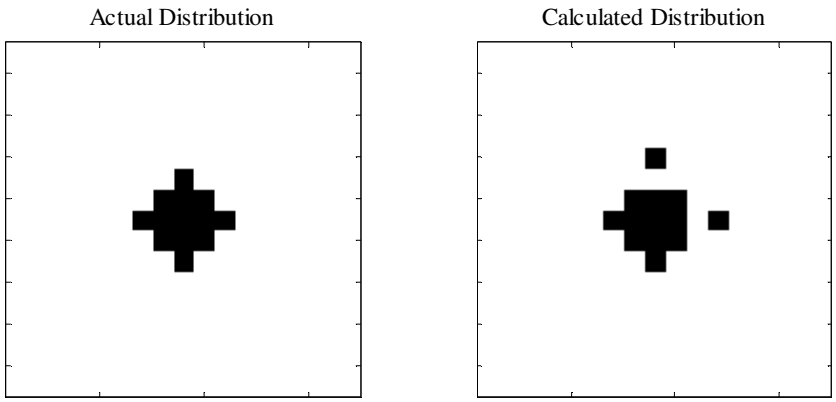


Figure 4.34: Comparison of the actual conductivity distribution and the result of the genetic algorithm for uncontrolled noise test using Gaussian white noise with standard deviation of 0.001 volts.

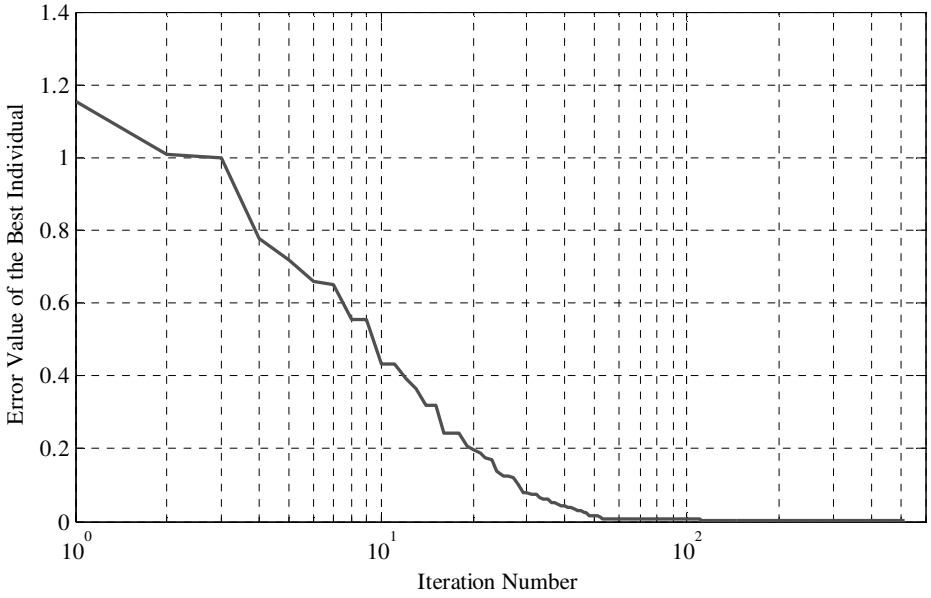


Figure 4.35: Error values of the best individuals of each generation versus generation number for uncontrolled noise test using Gaussian white noise with standard deviation of 0.001 volts.

The actual conductivity distribution and the result of the GA are compared for uncontrolled noise test using Gaussian white noise with standard deviation of 0.01 volts in Figure 4.36 and the error values of the best individuals of each generation are plotted versus generation number in Figure 4.37. We can see that the resemblance of

the result to the true conductivity distribution is reduced with the increasing noise. However, algorithm still achieves convergence successfully by reaching closer to the optimal solution of the problem with the noise standard deviation of 0.01 volts.

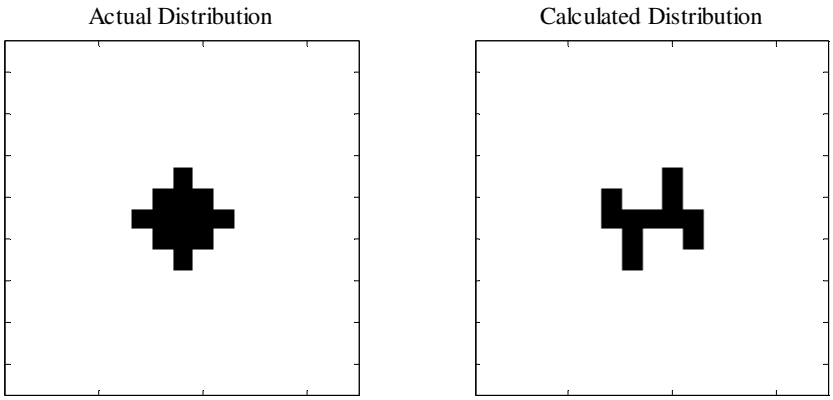


Figure 4.36: Comparison of the actual conductivity distribution and the result of the genetic algorithm for uncontrolled noise test using Gaussian white noise with standard deviation of 0.01 volts.

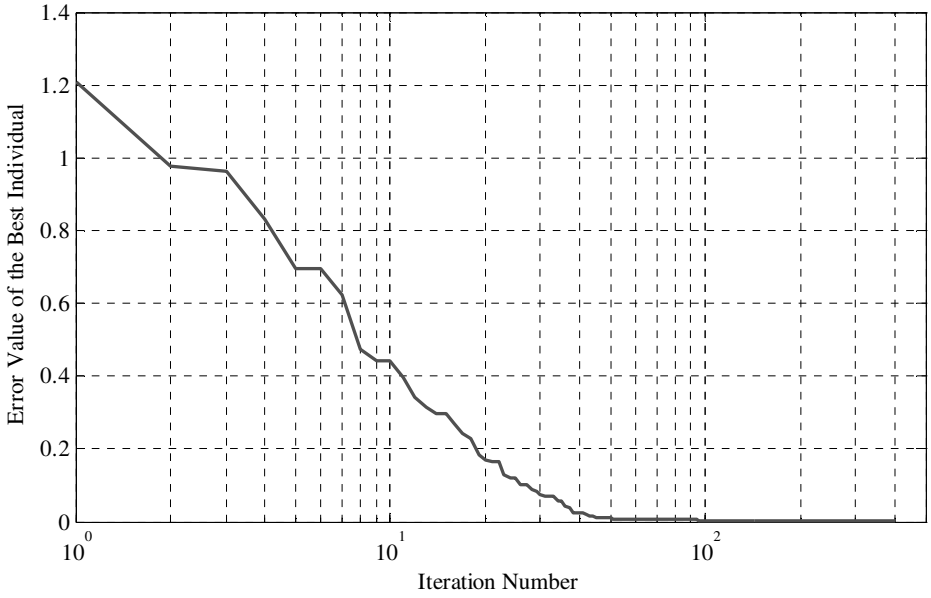


Figure 4.37: Error values of the best individuals of each generation versus generation number for uncontrolled noise test using Gaussian white noise with standard deviation of 0.01 volts.

In Figure 4.38, comparison of the actual conductivity distribution and the result of the GA are compared for uncontrolled noise test using Gaussian white noise with standard deviation of 0.1 volts. The error values of the best individuals of each generation are plotted versus generation number in Figure 4.39. As the noise level increases, the resulted conductivity distribution further worsens. Convergence of the

algorithm also gets worse with the increasing noise and the convergence speed decreases.

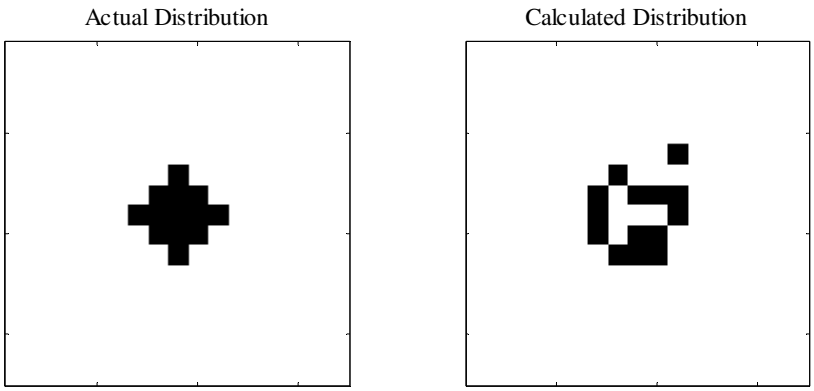


Figure 4.38: Comparison of the actual conductivity distribution and the result of the genetic algorithm for uncontrolled noise test using Gaussian white noise with standard deviation of 0.1 volts.

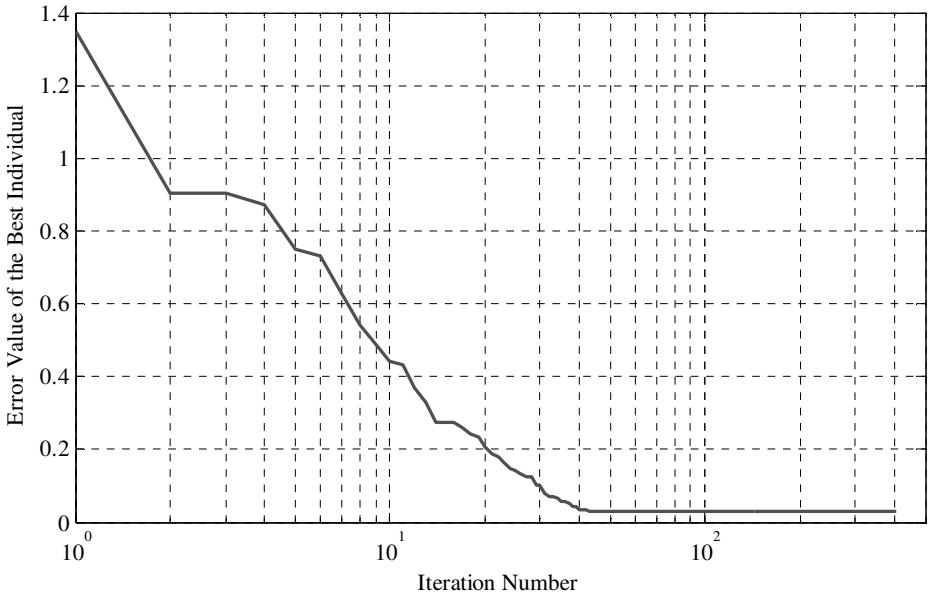


Figure 4.39: Error values of the best individuals of each generation versus generation number for uncontrolled noise test using Gaussian white noise with standard deviation of 0.1 volts.

The amount of Gaussian white noise that the genetic algorithm tolerates can be seen by analyzing the results of the noise tests. As the standard deviation of the noise increases up to 0.001 volts, possibility of reaching the exact result is dramatically decreases. If the amplitude of the noise is further increased, results of the image reconstruction algorithm worsen and the resemblance of the result and true conductivity distribution is reduced. Increasing amplitude of the noise also causes the

convergence speed of the GA to become slower and if it exceeds a certain level, convergence near the optimal solution becomes impossible.

White Gaussian noise function is considered as the most appropriate noise generator to simulate the noise on a typical EII system (Holder, 2005). In the numerical simulations, noise is generated by using MATLAB's Gaussian white noise function. Histogram of the Gaussian white noise generated by MATLAB with standard deviation of 0.01 volts for voltage data with 10000 elements is shown in Figure 4.40.

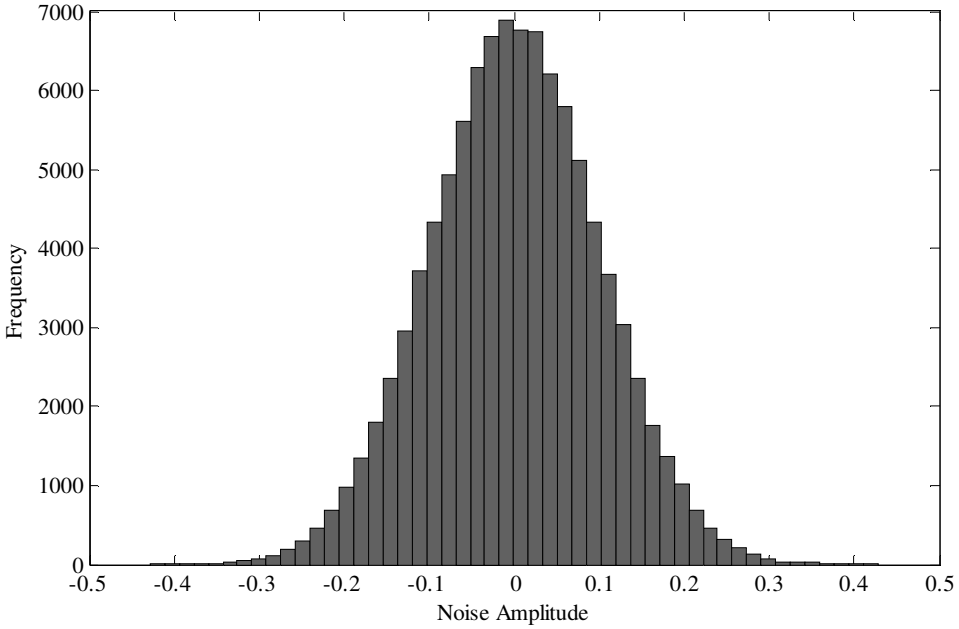
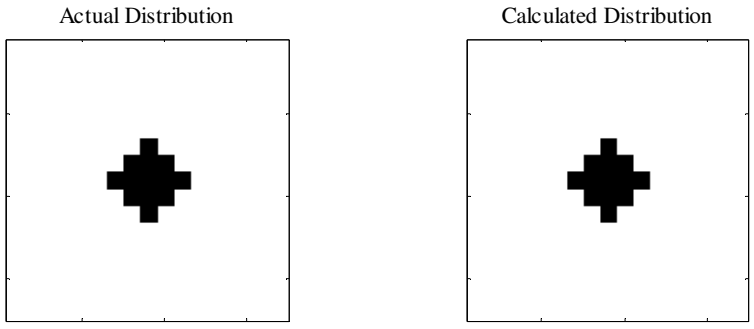


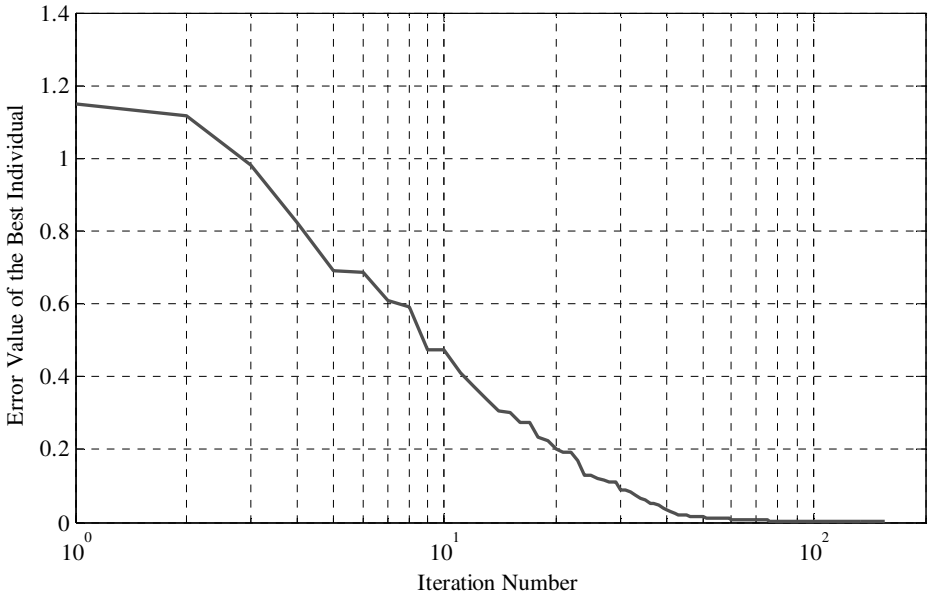
Figure 4.40: Histogram of Gaussian white noise with standard deviation of 0.01 volts for 10000 voltage data points.

Noise in the measurements data can be reduced by executing multiple measurements and averaging the collected. Because the EII is a static imaging method, it is possible to achieve multiple measurements for the same conductivity distribution in a short period of time. Changes in the electrical field for different measurements are small enough to neglect; therefore, the difference in the data from separate measurements is mainly due to the noise on the imaging system. As the noise on the data is randomly distributed, averaging the data from different measurements reduces the noise greatly by causing the collected voltage values from the electrodes to converge closer to their expected values, which are the ideal measured voltage values without the presence of the noise. This sequential averaging method controls the noise that is present on the measurement data and increases accuracy of the measurement process of EII (Ovacık, 1998b).

The next part of the tests covers the experiments that are conducted using the data with the noise controlled by averaging data that are collected from multiple measurements. Numerical simulations are conducted using the 32-electrode model and the same conductivity distribution as the previous noise tests for comparison. Aim of the controlled noise test is to determine the performance of the algorithm using noise reduction by averaging. In each controlled noise test, simulation of the measurement process is repeated 1000 times with the addition of random Gaussian white noise and the output data for each simulation are averaged element-wise. Next, the GA reconstructs the conductivity distribution. Results of the controlled noise test for noise standard deviation of 0.001, 0.01 and 0.1 volts are shown in Figures 4.41, 4.42 and 4.43 respectively.

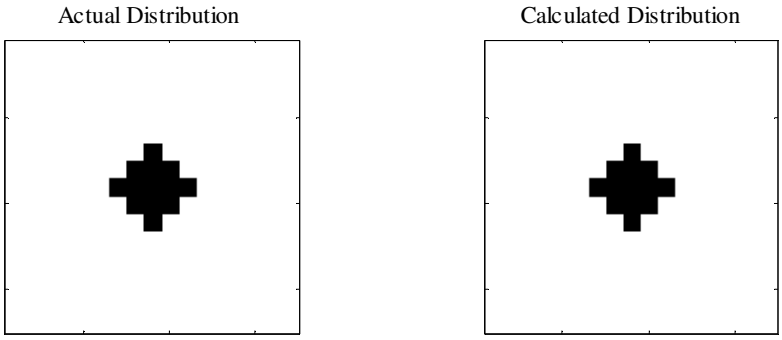


(a)

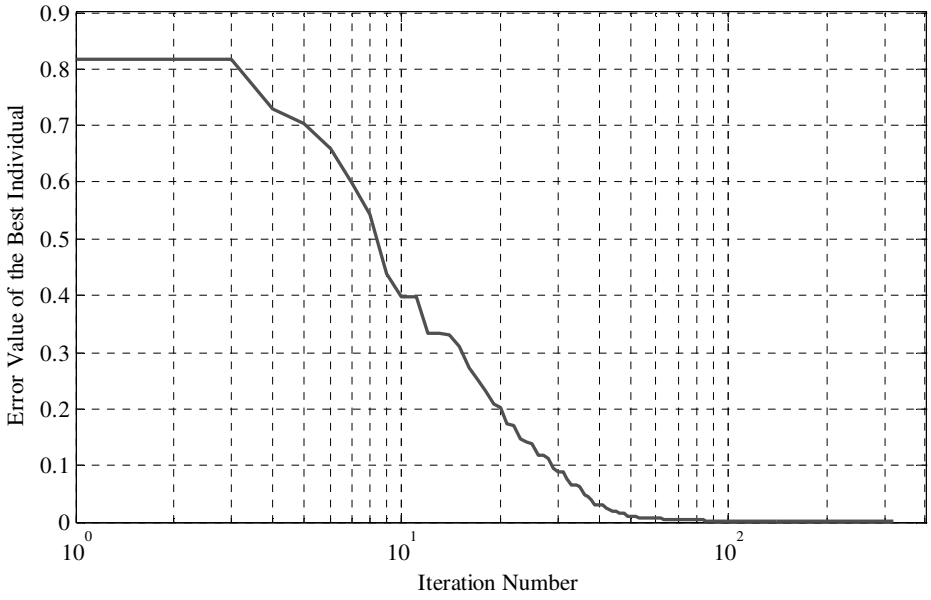


(b)

Figure 4.41: Results of the controlled noise test with 1000 measurements and noise standard deviation of 0.001 volts: **(a)** Comparison of the actual and resulted conductivity distributions. **(b)** Convergence of the algorithm.



(a)

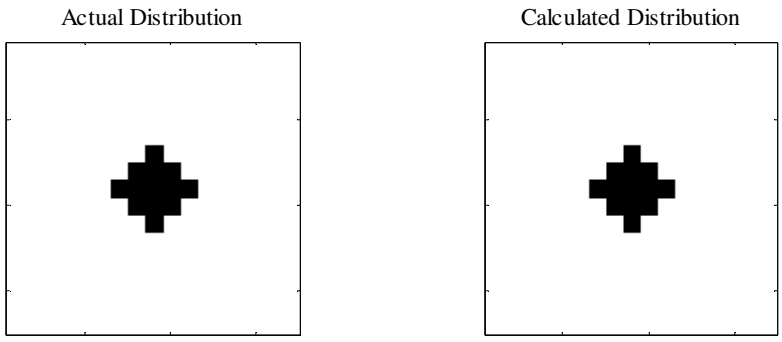


(b)

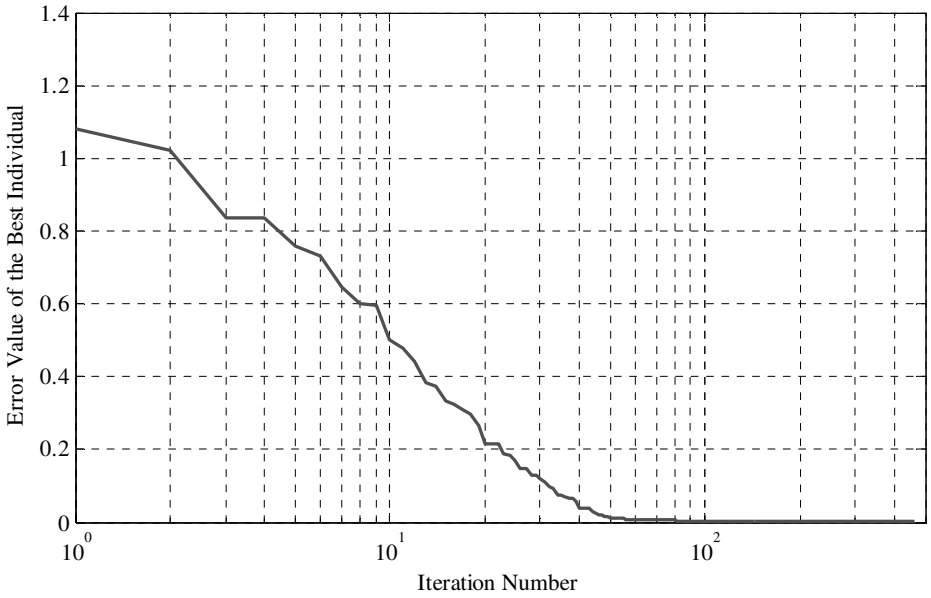
Figure 4.42: Results of the controlled noise test with 1000 measurements and noise standard deviation of 0.01 volts: **(a)** Comparison of the actual and resulted conductivity distributions. **(b)** Convergence of the algorithm.

After comparing the results from the controlled noise tests and the uncontrolled noise tests for the corresponding noise levels, the improvement in the results by using sequential averaging method is clear. GA attained the true conductivity distribution in the controlled noise tests using the data with the noise levels that the exact result could not be reached in the uncontrolled noise tests. Even with high presence of noise (standard deviation of 0.1 volts), the GA managed to attain the true conductivity distribution in the controlled noise tests. By controlling the noise on the data with averaging, the GAs noise tolerance is increased dramatically. Looking at the convergence plots, we can see that increasing noise level also increases the

number of iterations that the algorithm requires to reach the true conductivity distribution.



(a)

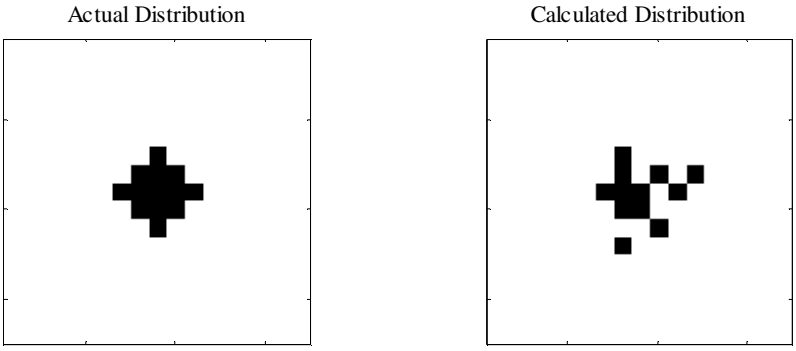


(b)

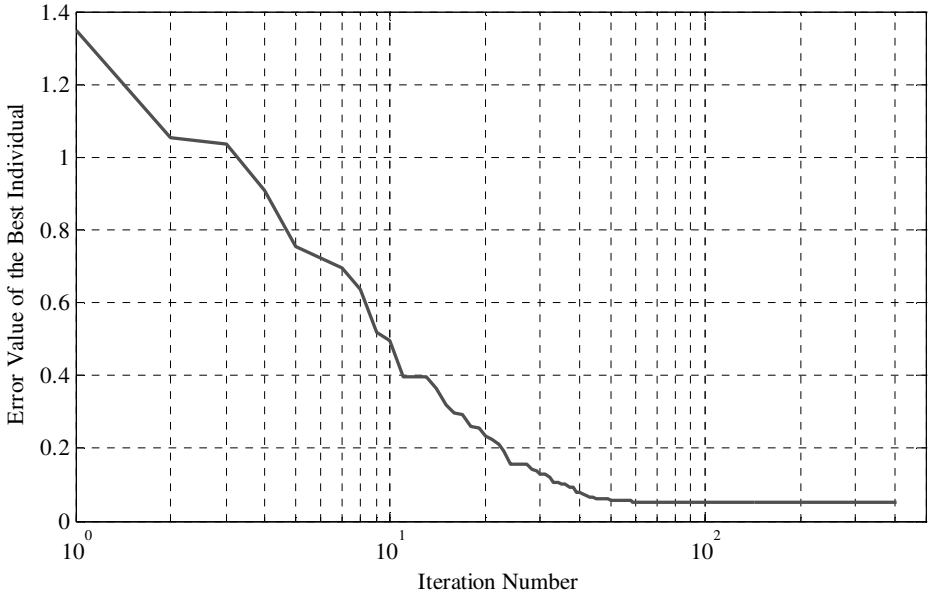
Figure 4.43: Results of the controlled noise test with 1000 measurements and noise standard deviation of 0.1 volts: **(a)** Comparison of the actual and resulted conductivity distributions. **(b)** Convergence of the algorithm.

In the last part of the tests, the GA is executed to reconstruct the conductivity distribution using the data that contain Gaussian white noise with a fixed signal to noise ratio. Fixing the SNR for every data point ensures that the noise on each point does not exceed a fixed fraction of the corresponding voltage value. For a better comparison, the measurement simulation is conducted using the same conductivity distribution as in the previous noise tests. The algorithm is tested with four levels of noise, specifically SNR of 25 dB, 50 dB, 75 dB and 100 dB.

Results of the noise test for the SNR of 25 dB are shown in Figure 4.44. Although the convergence was achieved up to a point, we can see that the algorithm failed to reach the true conductivity distribution for the noise level of 25 dB. In Figure 4.44 (b), error values of the best individuals of each generation are plotted versus generation number.



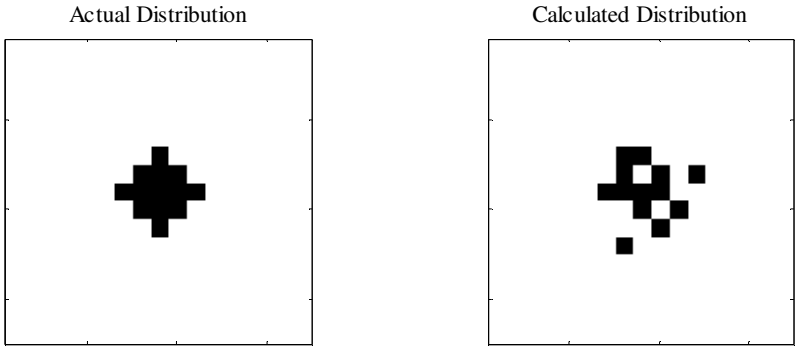
(a)



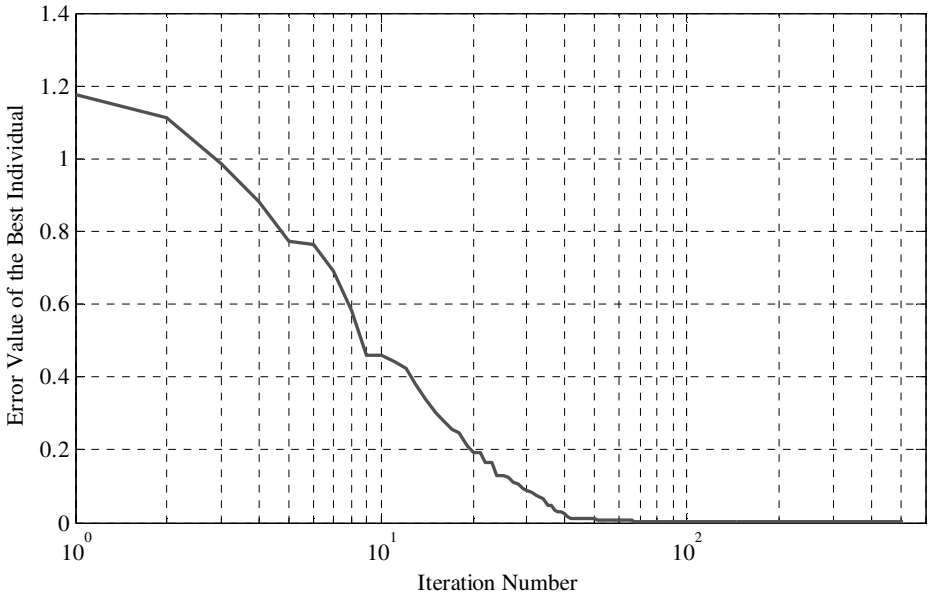
(b)

Figure 4.44: Results of the noise test for the noise level of 25 dB: **(a)** Comparison of the actual and resulted conductivity distributions. **(b)** Error values of the best individuals of each generation versus generation number.

Results of the noise test for the SNR of 50 dB are shown in Figure 4.45. Despite achieving convergence, the algorithm was not successful in attaining the true conductivity distribution for the noise level of 50 dB. In Figure 4.45 (b), error values of the best individuals of each generation are plotted versus generation number.



(a)



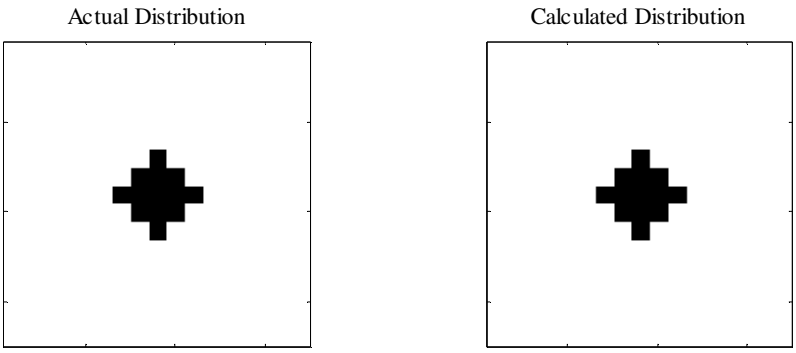
(b)

Figure 4.45: Results of the noise test for the noise level of 50 dB: **(a)** Comparison of the actual and resulted conductivity distributions. **(b)** Error values of the best individuals of each generation versus generation number.

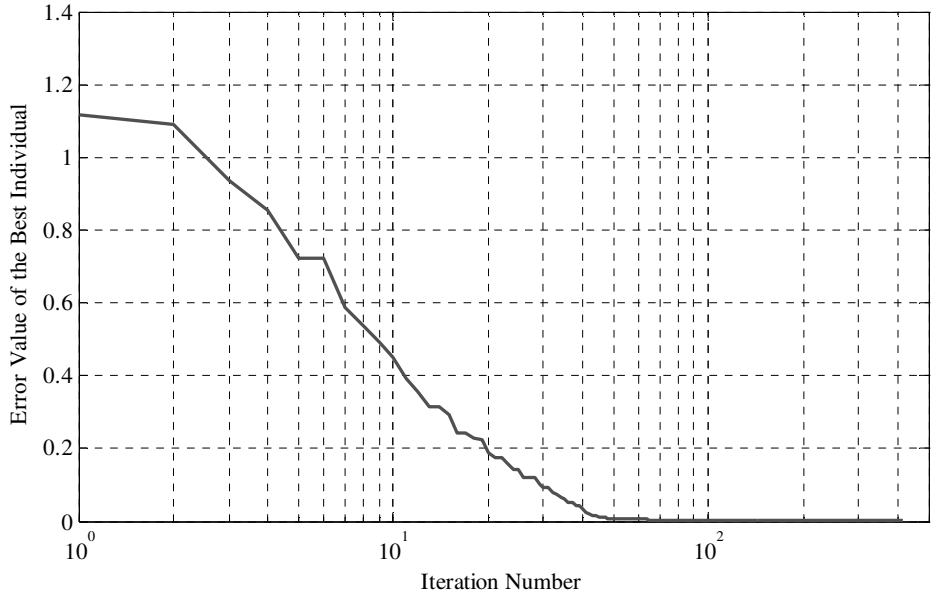
Comparing Figures 4.44 (b) and 4.45 (b), it is clear that the algorithm converged closer to the global optimum in this test than the previous test. As the noise level decreases, convergence speed of the genetic algorithm increases and the distance between the global optimum and the result of the algorithm becomes closer.

Comparison of the actual conductivity distribution and the result of the GA using Gaussian white noise with the SNR of 75 dB is shown in Figure 4.46 (a) and the error values of the best individuals of each generation are plotted versus generation number in Figure 4.46 (b). From the figure, we can see that the algorithm successfully attained the true conductivity distribution for the noise level of 75 dB.

Image reconstruction process lasted 1197 seconds of computation time and 413 iterations.



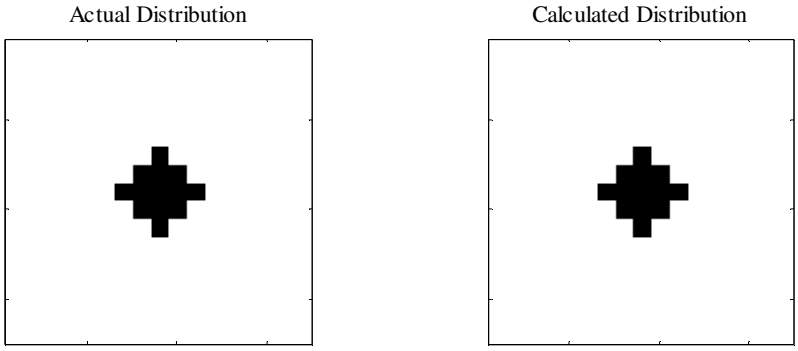
(a)



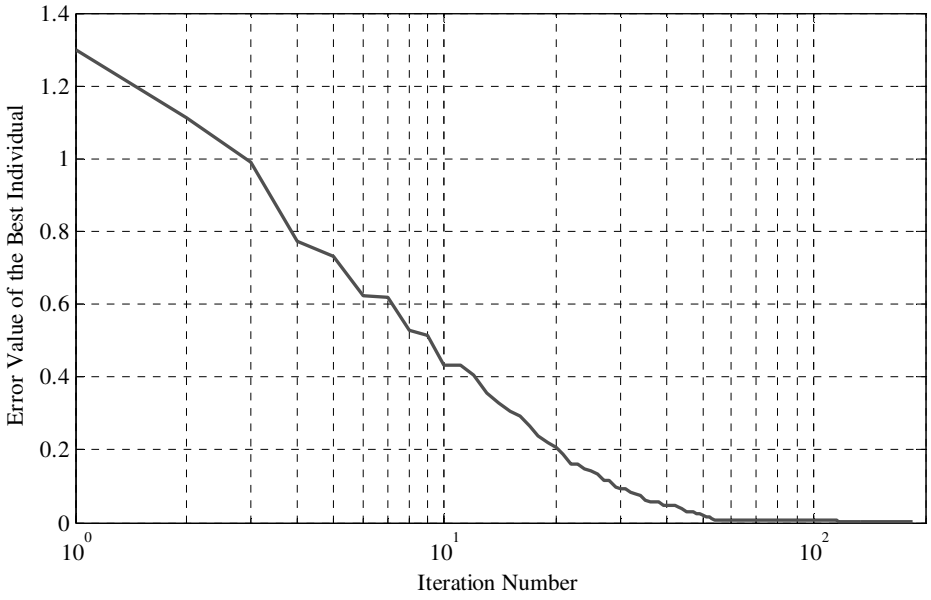
(b)

Figure 4.46: Results of the noise test for the noise level of 75 dB: **(a)** Comparison of the actual and resulted conductivity distributions. **(b)** Error values of the best individuals of each generation versus generation number.

In Figure 4.47, results of the noise test for the SNR of 100 dB are shown. In Figure 4.47 (b), it is seen that the algorithm successfully achieved the true conductivity distribution for the noise level of 100 dB. In Figure 4.47 (b), error values of the best individuals of each generation are plotted versus generation number. Total computing time of the image reconstruction was 603 seconds and 186 iterations. Comparing this numbers with the previous test, we can see that increasing noise also increases the computing time requirement of the algorithm.



(a)



(b)

Figure 4.47: Results of the noise test for the noise level of 100 dB: **(a)** Comparison of the actual and resulted conductivity distributions. **(b)** Error values of the best individuals of each generation versus generation number.

5. CONCLUSION AND DISCUSSIONS

An improved two-stage genetic algorithm was developed in this thesis for the reconstruction of two-dimensional and binary conductivity distributions using the concept of electrical impedance imaging method based on the minimization of the discrepancies between measured and computed electrode voltages in a least-square sense. Mathematical model of the imaging process was established using finite element method to obtain the voltage response on the boundary electrodes. The problem of ill conditioning due to the relatively weak voltage response to the targets that are located far away from the boundary electrodes was surmounted by the development of a new weight function. Four new mutation operators and an improved rank proportionate selection operator were introduced in this thesis. An adaptive parameter control operator was established to maintain the diversity of the population at an efficient level. A series of tests was conducted to observe the genetic algorithms performance on various conditions.

The genetic algorithm has shown an excellent performance by attaining the true conductivity distribution in all the tests with 16-electrode model. Despite the increasing vastness of the search space for the 32-electrode model, the algorithm still managed to attain the true conductivity distribution in most of the tests. GA successfully reconstructed target objects with complex details located in the center of the conductivity distribution, which is a particularly difficult task in EII method due to the reduced sensitivity of the region that is far away from the boundary. Reconstruction of the conductivity distributions that include numerous small objects spread to the entire body was observed to be the most demanding situation for the GA, where the objects located near the boundary becomes dominant, causing inaccuracy in the central region of the body.

In the tests using the data with noise, the algorithm reached the true conductivity distribution up to a certain noise level, Gaussian white noise up to a fixed standard deviation of 0,0001 V and down to a fixed SNR of 75 dB. Working with the data that contain noise exceeding these levels, the GA achieved convergence in the first stage

and reconstructed a similar image to the actual conductivity distribution of the body. By taking advantage of electrical impedance imaging method's high imaging speed, noise on the data was controlled by making numerous measurements and averaging the corresponding results to minimize the effects of the noise on the reconstruction algorithm. Although the noise tolerance of the GA has improved dramatically with the usage of this noise control technique, EII is still unsuitable for applications that contain excessive noise on the measurement data and low-noise data acquisition hardware is recommended for the stability of the results.

The weak voltage response of the pixels located far away from the boundary electrodes is a common problem of EII image reconstruction process. It can clearly be seen in Figure 3.7 that a pixel in the central region is nearly six times less sensitive than a pixel near the boundary. This problem hinders the reconstruction algorithm's progress of attaining the true conductivity distribution in the center region. The weight function developed in this thesis has improved the reconstruction of the central pixels of the conductivity distribution significantly. With the addition of this weight function, GA attained the exactly true conductivity distribution in most of the tests. In Figures 4.9 and 4.16, where the best individuals of the selected generations are shown chronologically, it can be seen that the GA reconstructs the conductivity distribution starting from the area near the boundary, completing the center of the distribution in the final generations. This is a direct result of the dominancy of the area near the boundary on the fitness values of the individuals. Because the selection operator favors the fitter individuals, GAs focus on the short-term gain. Therefore, convergence is first achieved in the properties of the individuals that contribute more to the fitness function. This characteristic of the genetic algorithms must be taken into account when developing a search strategy.

Multi-stage structure offers genetic algorithms very significant advantages in the development of a search strategy. More Efficient search strategies can be established by using different genetic operators in each stage to match the dynamic behavior of the problem. Multi-stage structure also allows altering the parameters of the GA to suit the characteristics of different eras of the solution process. Characteristic behavior of the image reconstruction problem of EII method changes dramatically throughout the solution process of the genetic algorithm. Convergence speed of a genetic algorithm generally becomes slower as the algorithm converges near the

optimal solution. Although the diversity of the population is very rich at the beginning of the algorithm due to the random creation of the initial population, it decreases as the algorithm achieves progress. Two-stage structure of the genetic algorithm, which is developed to overcome this changing behavior of the solution process, has improved the efficiency of the algorithm dramatically as it allows the use of shape-searching mutation filters in the final generations of the algorithm. The shape-searching mutation operators (neighborhood shift mutation filter and center fill mutation filter) are crucial for reaching the exact conductivity distribution. Because the diversity is low among the population in the final generations of the algorithm, genetic operators like selection and recombination becomes ineffective in the search for the true conductivity distribution. Neighborhood shift mutation filter and center fill mutation filter, which are introduced in this thesis, vastly decreases the time required to reach the true result by mutating the shapes of the targets located in the conductivity distributions of the individuals. However, these mutation operators should only be used in the second stage of the GA as they cause the convergence speed to drop. For the first stage of the algorithm, two new mutation operators (adaptive mutation probability filter and identical individual eliminator filter) are developed to increase the convergence speed.

Deciding which individuals carry their genes to the next population, selection process is the heart of a genetic algorithm,. By giving higher probability of selection to the fitter individuals, selection operators must possess stochastic properties in order to provide robustness to the GA. In this thesis, an improved ranked proportionate selection operator is developed to achieve robustness and acquire the ability to apply selection pressure to the population. Conventional fitness proportionate selection operators does not provide the option to apply selection pressure. Selection pressure is a critical parameter for controlling the diversity of the population and the convergence speed, which are the most important factors of a genetic algorithm. Increasing the convergence speed decreases the diversity of the population. On the other hand, rich diversity provides robustness to a genetic algorithm, giving the it the ability of avoiding the local minima more effectively. Therefore, efficiency of a genetic algorithm increases to a maximum level only when the convergence speed and the diversity of the population are optimally balanced. The only way of maintaining the balance between the diversity and the convergence

speed at an efficient level is to control the important parameters of the genetic algorithm adaptively. The selection pressure, the mutation probability and the crossover probability are controlled adaptively in the genetic algorithm depending on the diversity of the population. Applying higher selection pressure to the population increases the convergence speed. However, to enrich the population's diversity, the selection pressure should be decreased. Although controlling the crossover rate creates a weaker response on these two factors, an ideal crossover rate should start at the maximum level and decrease slightly with the reducing diversity. Mutation probability has an adverse effect on the convergence speed, while helping the algorithm to enrich the diversity of the population. By controlling these parameters efficiently, improving the dynamic behavior of the genetic algorithm becomes possible and robustness of the algorithm increases, avoiding the local minima more effectively.

Genetic algorithms are promising tools for the solution of image reconstruction problem of electrical impedance imaging method. Although genetic algorithms are expensive in terms of computing time and resources, which renders them inappropriate for the real-time applications, they are suitable for ill-conditioned problems thanks to their stochastic nature, parallel searching capabilities and robustness in avoiding local minima. As a response to the increasing demand from the industry for process monitoring applications, the usage of genetic algorithms in image reconstruction problem of EII has been an active area of research in the recent years. Future developments on this subject can be achieved in different areas including the mathematical model and the structure of the reconstruction algorithm. A more accurate numerical model of the measurement process may help to increase the imaging resolution. Real-time imaging can be possible with the introduction of parallel computing to EII method. Development of hybrid image reconstruction algorithms that combine genetic algorithms and gradient-based methods may lead to more efficient reconstruction processes, reducing the computation time requirements of the current-generation reconstruction algorithms.

REFERENCES

Atkinson, C. M., and Kytomaa, H. K., 1992. Acoustic wave speed and attenuation in suspensions, *International Journal of Multiphase Flow*, Vol. 18, No. 4, pp. 557–592.

Atkinson, C. M., and Kytomaa, H. K., 1993. Acoustic properties of solid-liquid mixtures and limits of ultrasound diagnostics-I: Experiments, *Journal of Fluids Engineering*, Vol. 115, pp. 665–675.

Barber, D. C., and Brown, B. H., 1984. Applied Potential Tomography, *Journal of Physics E: Scientific Instruments*, Vol. 17, pp. 723-733.

Calderon, A. P., 1980. On an inverse boundary value problem, *Seminar on Numerical Analysis and its applications to Continuum Physics*, Sociedade Brasileira de Matematica, Rio de Janeiro, pp. 65–73.

Ceccio, S. L., and George, D. L., 1996. A review of electrical impedance techniques for measurement of multiphase flows, *Journal of Fluids Engineering*, Vol. 118, pp. 391–399.

Cheney, M., Isaacson, D., Newell, J., Goble, J., and Simske, S., 1990. NOSER: An algorithm for solving the inverse conductivity problem, *International Journal of Imaging Systems and Technology*, Vol. 2, pp. 66-75.

Cheng, K. S., Chen, B. H., and Tong, H. S., 1996. Electrical impedance image reconstruction using the genetic algorithm, *18th Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, Amsterdam, Vol. 2, pp. 768-769.

Dynes, K. A., and Lytle, R. J., 1981. Analysis of electrical conductivity imaging, *Geophysics*, Vol. 46, pp. 1025–1036.

Fish, J., and Belytschko, T., 2007. *A first course in finite elements*, Wiley & Sons.

Henderson, R. P., and Webster, J. G., 1978. An impedance camera for spatially specific measurements of the thorax, *IEEE Transactions on Biomedical Engineering*, Vol. 25, No. 3, pp. 250–254.

Holder, D. S., 2005. Electrical impedance tomography: Methods, history and applications, IOP Publishing Ltd.

Holland, J. H., 1975. Adaptation in natural and artificial systems, *University of Michigan Press*.

Hsiao, C. T., Chahine, G., and Gumerov, N., 2001. Application of a hybrid genetic/powell algorithm and a boundary element method to electrical impedance tomography, *Journal of Computational Physics*, Vol. 173, pp. 433-454.

Hua, P., Woo, E. J., Webster, J. G., and Tompkins, W. J., 1993. Finite element modeling of electrode-skin contact impedance in electrical impedance tomography, *IEEE Transactions on Biomedical Engineering*, Vol. 40, No. 4, pp. 335-343.

Jones, O. C., Lin, J. T., and Ovacık, L., 1992. Investigation of electrical impedance imaging relative to two-phase, gas-liquid flows, *Chemical engineering communications,* Vol. 118, pp. 299-325.

Jones, O. C., Lin, J. T., Ovacık, L., and Shu, H., 1993. Impedance imaging relative to gas-liquid systems, *Nuclear engineering and design,* Vol. 141, No. 1-2, pp. 159-176.

Kaipio, J. P., Kolehmainen, V., Somersalo, E., and Vauhkonen, M., 2000. Statistical inversion and Monte Carlo sampling methods in electrical impedance tomography, *Inverse Problems*, Vol. 16, pp. 1487-1522.

Kim H. C., Moon, D. C., Kim, M. C., Kim, S., and Lee, Y. J., 2002. Improvement in EIT image reconstruction using genetic algorithm, *Proceedings of the American Control Conference*, Anchorage, Vol. 5, pp. 3858- 3863.

Kim, H. C., Boo, C. J., and Kang, M. J., 2006. Image reconstruction using genetic algorithm in electrical impedance tomography, *Lecture Notes in Computer Science*, Volume 4234, pp. 938-945.

Kohn, R. V., and Vogelius, M., 1984. Determining conductivity by boundary measurements, *Communications on Pure and Applied Mathematics*, Vol. 38, No. 5, pp. 643-667.

Langer R. E., 1933. An inverse problem in differential equations, *Bulletin of the American Mathematical Society*, Vol. 39, pp. 814–820.

Meng, Z. Q., Takenaka, T., and Tanaka, T., 1999. Image reconstruction of two-dimensional impenetrable objects using genetic algorithm, *Journal of Electromagnetic Waves and Applications*, Vol. 13, pp. 95-118.

Mitchell, M., 1999. An Introduction to genetic algorithms, *MIT Press*.

Nachman, A. I., 1995. Global uniqueness for a two-dimensional inverse boundary value problem, *Annual of Mathematics*, Vol. 142, No. 1, pp. 71–96.

Olmi, R., Bini, M., And Priori, S., 2000. A genetic algorithm approach to image reconstruction in electrical impedance tomography, *IEEE Transactions on Evolutionary Computation*, Vol. 4, No. 1, pp. 83-88.

Ovacık, L., 1989. Sonlu elemanlar yöntemi ile elektrostatik alanların sayısal olarak hesaplanması, Yüksek Lisans Tezi, İstanbul Teknik Üniversitesi, İstanbul.

Ovacık, L., Jones, O. C., Lin, J. T., and Shu, H., 1998a. A finite element electrical impedance imaging system for binary media: 1. Analytical and numerical methods, *Inverse Problems in Engineering*, Vol. 6, pp. 33-77.

Ovacık, L., 1998b. Extensions to the finite element method for the analysis of inverse problems in electromagnetic devices, PhD Thesis, Rensselaer Polytechnic Institute, New York.

Pozrikidis, C., 2005. *Introduction to finite and spectral element methods using MATLAB*, CRC Press.

Rolnik, V.P., and Seleghim, P., 2006. A specialized genetic algorithm for the electrical impedance tomography of two-phase flows, *Journal of the Brazilian Society of Mechanical Sciences and Engineering*, Vol. 28, No. 4, pp. 378-389.

Sivanandam, S.N., and Deepa, S.N., 2008. *Introduction to genetic algorithms*, *Springer*.

Slichter L. B., 1933. The interpretation of the resistivity prospecting method for horizontal structures, *Physics*, Vol. 4, pp. 309-322.

Sylvester, J., and Uhlmann, G., 1987. A global uniqueness theorem for an inverse boundary value problem, *Annual of Mathematics*, Vol. 125, No. 1, pp. 153-169.

Woo, E. J., Pallas-Areny, R., Webster, J. G., and Tompkins, W. J., 1990. Using Walsh functions in electrical impedance tomography, *Twelfth Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, Vol. 12, No. 1, pp. 124-125.

Yorkey, T. J., Webster, J. G., and Tompkins, W. J., 1987. Comparing reconstruction algorithms for electrical impedance tomography, *IEEE Transactions on Biomedical Engineering*, Vol. 34, No. 11, pp. 843-852.

APPENDICES

APPENDIX A : MATLAB Code of the Genetic Algorithm
APPENDIX B : Walsh patterns used for excitation

APPENDIX A

```
% Model parameters
mi=17; mj=17; vne=32; cne=32; nex=31; vrefelec=1; ni=mi+1; nj=mj+1;
m=mi*mj; n=ni*nj; backgcon=1; foregcon=0.1; wfalfa=10;

% General genetic algorithm parameters
maxpop=200; genmax=2000; errcrit=1e-4; itercrit=200; secinit=20;
selectco1=3; selectco2=1; crossco=0.1; ieeappgen=5;
ieemutprob=0.002; mutprob1max=0.0001; mutprob2max=0.001; elitist=1;

% GA parameters for second stage
sselitist=2;  midmutmul=2; nemutmul=2; midmutrange=[0.25 0.75];
nmutprobind=0.2; nmutprobbit=0.1; fillmutprob=0.1;

% Import data
fid=fopen('out.txt', 'r');
for i=1:(vne*nex)
out=fscanf(fid,'%f',[1,2]); vi(i)=out(1,1); ci(i)=out(1,2);
end, fclose(fid);
fidie=fopen('Ielec.txt','r'); ielecnod = fscanf(fidie,'%i',[2,inf]);
fclose(fidie);
fidve=fopen('Velec.txt','r'); velecnod = fscanf(fidve,'%i',[2,inf]);
fclose(fidve);
fidme=fopen('mesh.txt','r');
for i=1:n
xyco = fscanf(fidme,'%f',[1,2]); xco(i)=xyco(1); yco(i)=xyco(2);
end
fclose(fidme);
fidcur=fopen('current.txt','r');
for i=1:nex, for j=1:cne
cur(i,j) = fscanf(fidcur,'%f',[1,1]);
end, end
status = fclose(fidcur);

% Calculation of Local Admittance Elements
y2d=initmodel(mi,mj,xco,yco);

% Calculation of Weight Function
wf=wfunc(mi,mj,cne,vne,nex,vrefelec,ielecnod,velecnod,cur,vi,y2d,bac
kgcon,wfalfa);

% Initiation of First Stage of the Genetic Algorithm
stage=1; disp('First Stage of the Genetic Algorithm.');
clear minerr diver bestindhist selecthist crosshist mutprob1hist
mutprob2hist
pop=round(rand(maxpop,m)); pop=logical(pop);

% Main program loop
for gen=1:genmax
disp('Iteration Number: '); disp(gen);

% Evaluation of Fitness Fuction for Stage 1
if stage == 1
for ie=1:maxpop, for je=1:mj, for ke=1:mi
if pop(ie,((je-1)*mi+ke))==0
real(je,ke)=backgcon;
else, real(je,ke)=foregcon;
```

```
end, end, end
error=fit(mi,mj,cne,vne,nex,vrefelec,ielecnod,velecnod,real,cur,vi,w
f,y2d);
poperr(ie)=error;
end, end

% Evaluation of Fitness Fuction for Stage 2 with fitness memory
if stage == 2
for ie=1:maxpop, for je=1:mj, for ke=1:mi
if pop(ie,((je-1)*mi+ke))==0
real(je,ke)=backgcon;
else, real(je,ke)=foregcon;
end, end, end
for i=1:maxpop
found=all(pop(ie,:) == prevpop(i,:));
if found == 1
poperr(ie)=prevpoperr(i);
break
end, end
if found == 0
error=fit(mi,mj,cne,vne,nex,vrefelec,ielecnod,velecnod,real,cur,vi,w
f,y2d); poperr(ie)=error;
end, end, end

%Store the best individual of the generation
if rem(gen,10) == 0 || gen == 1, if gen == 1
[C,I]=min(poperr); bestindhist(1,:)=pop(I,:);
Else, [C,I]=min(poperr); bestindhist((gen/10)+1,:)=pop(I,:);
end, end

% Termination by Error Criteria
minerr(gen)=min(poperr); meanerr(gen)=mean(poperr);
if minerr(gen) <= errcrit
break
end

% Termination by maximum number of iterations without improvement
if itercrit < gen
for i=1:itercrit
itercritcheck(i) = minerr(gen-i+1) == minerr(gen);
end
else, itercritcheck=0;
end
if all(itercritcheck) == 1
break
end
disp('Best Individual: '); disp(minerr(gen));
diver(gen)=sqrt((meanerr(gen)-poperr)*(meanerr(gen)-poperr)');
disp('Diversity: '); disp(diver(gen));

% Initiation of Second Stage of the Genetic Algorithm
if secinit < gen
for i=1:secinit
secinitcheck(i) = minerr(gen-i+1) == minerr(gen);
end
else, secinitcheck=0;
end
if all(secinitcheck) == 1 && stage == 1
stage=2; secstageinitgen=gen; elitist=sselitist;
disp('Second Stage of the Genetic Algorithm.');
end
```

```
% Adaptation of Parameters
selectco=selectco1-selectco2*exp(-diver(gen));
crossrate=1-crossco*exp(-diver(gen));
mutprob1=mutprob1max*exp(-diver(gen));
mutprob2=mutprob2max*exp(-diver(gen));
mutprobpop=(mutprob2-mutprob1)*(abs(2*(mean(pop)-0.5)))+mutprob1;
selecthist(gen)=selectco; crosshist(gen)=crossrate;
mutprob1hist(gen)=mutprob1; mutprob2hist(gen)=mutprob2;

% Elitist Selection
if elitist ~= 0
[sorted,sortindex]=sort(poperr,'ascend');
for i=1:elitist
sel1(i)=sortindex(i); sel2(i)=sortindex(i);
end, end

% Rank-Based proportionate Selection
[sorted,sortindex]=sort(poperr,'ascend');
rank=1:maxpop;
for i=1:maxpop
normalrank(i)=rank(i)/maxpop;
select(i)=exp(-selectco*normalrank(i));
end
sumselect=sum(select);
selectprob(1)=0;
for i=2:(maxpop+1)
selectprob(i)=(select(i-1)/sumselect)+selectprob(i-1);
end
for i=(elitist+1):(round(maxpop/2)), for counter=1:1000
randselect1=rand(1,1); randselect2=rand(1,1);
for ii=1:maxpop
if randselect1 >= selectprob(ii) && randselect1 < selectprob(ii+1)
sel1(i)=sortindex(ii);
end
if randselect2 >= selectprob(ii) && randselect2 < selectprob(ii+1)
sel2(i)=sortindex(ii);
end, end
if all(pop(sel1(i),:) == pop(sel2(i),:)) == 0
break
end, end, end
newpop= false(maxpop,m);

% Crossover
for i=1:length(sel1)
cross=rand(1,1);
if cross > crossrate
crossmask=zeros(m,1);
else, crossmask=round(rand(m,1));
end
for j=1:m
if crossmask(j)==0
newpop(((2*i)-1),j)=pop(sel1(i),j); newpop((2*i),j)=pop(sel2(i),j);
else
newpop(((2*i)-1),j)=pop(sel2(i),j); newpop((2*i),j)=pop(sel1(i),j);
end, end, end

% Mutation For 2nd Stage
if stage == 2
for i=1:maxpop
randmut1=rand(1,1);
```

```
if randmut1 <= nmutprobind && i > (2*elitist)
for j=1:m
if newpop(i,j) == 1
randmut2=rand(1,1);
if randmut2 <= nmutprobbit
randbitpos=ceil(9*rand(1,1));
divremain=((j/mi)-floor(j/mi))*mi;
if j<=(mi+1) || j>(m-mi-1) || divremain<=1 || divremain>=(mi-1)
randbitpos=5;
end
switch randbitpos
case 1
if newpop(i,j-mi-1) == 0
newpop(i,j)=~newpop(i,j);
end
newpop(i,j-mi-1)=~newpop(i,j-mi-1);
case 2
if newpop(i,j-mi-1) == 0
newpop(i,j)=~newpop(i,j);
end
newpop(i,j-mi)=~newpop(i,j-mi);
case 3
if newpop(i,j-mi-1) == 0
newpop(i,j)=~newpop(i,j);
end
newpop(i,j-mi+1)=~newpop(i,j-mi+1);
case 4
if newpop(i,j-mi-1) == 0
newpop(i,j)=~newpop(i,j);
end
newpop(i,j-1)=~newpop(i,j-1);
case 5, newpop(i,j)=~newpop(i,j);
case 6
if newpop(i,j-mi-1) == 0
newpop(i,j)=~newpop(i,j);
end
newpop(i,j+1)=~newpop(i,j+1);
case 7
if newpop(i,j-mi-1) == 0
newpop(i,j)=~newpop(i,j);
end
newpop(i,j+mi-1)=~newpop(i,j+mi-1);
case 8
if newpop(i,j-mi-1) == 0
newpop(i,j)=~newpop(i,j);
end
newpop(i,j+mi)=~newpop(i,j+mi);
case 9
if newpop(i,j-mi-1) == 0
newpop(i,j)=~newpop(i,j);
end
newpop(i,j+mi+1)=~newpop(i,j+mi+1);
end, end, end, end
elseif i > (2*elitist)
for j=1:m
fillcount=[0, 0, 0, 0];
if j < (m-mj)
fillcount(1)=newpop(i,j+mj);
end
if rem(j,mj) ~= 0
fillcount(2)=newpop(i,j+1);
```

```
end
if j > mj
fillcount(3)=newpop(i,j-mj);
end
if rem(j,mj) > 1
fillcount(4)=newpop(i,j-1);
end
fillsum=sum(fillcount);
if rand(1,1) <= fillmutprob && fillsum >= 3 && newpop(i,j) == 0
newpop(i,j)=1;
end
bithorpos=((j/mi)-floor(j/mi))*mi;
bitverpos=ceil(j/mi);
if bithorpos>(midmutrange(1)*mi) && bithorpos<(midmutrange(2)*mi) &&
bitverpos>(midmutrange(1)*mj) && bitverpos<(midmutrange(2)*mj)
mutprob=midmutmul*mutprobpop(j);
else, mutprob=mutprobpop(j);
end
if any(fillcount) == 1
mutprob=mutprob*nemutmul;
end
mutmask=rand(1,1);
if mutmask <= mutprob
if newpop(i,j) == 0
newpop(i,j)=1;
else, newpop(i,j)=0;
end, end, end, end, end, end

% Mutation For 1st Stage
if stage == 1
for i=1:maxpop
for j=1:m
if i > (2*elitist)
mutmask=rand(1,1);
else, mutmask=1;
end
if mutmask <= mutprobpop(j)
if newpop(i,j) == 0
newpop(i,j)=1;
else, newpop(i,j)=0;
end, end, end, end, end

% Identical individual eliminator mutation filter
if (gen/ieeappgen) == floor(gen/ieeappgen)
for i=1:maxpop
for j=1:maxpop
if i ~= j && j > i && all(pop(i,:) == pop(j,:))
for k=1:m
randiie=rand(1,1);
if randiie <= ieemutprob
pop(j,m)=~pop(j,m);
end, end, end, end, end, end
prevpop=pop; prevpoperr=poperr; pop=newpop; clear newpop
end

% Export output data
[sorted,sortindex]=sort(poperr,'ascend');
result=pop(sortindex(1),:); dispim;
```

APPENDIX B

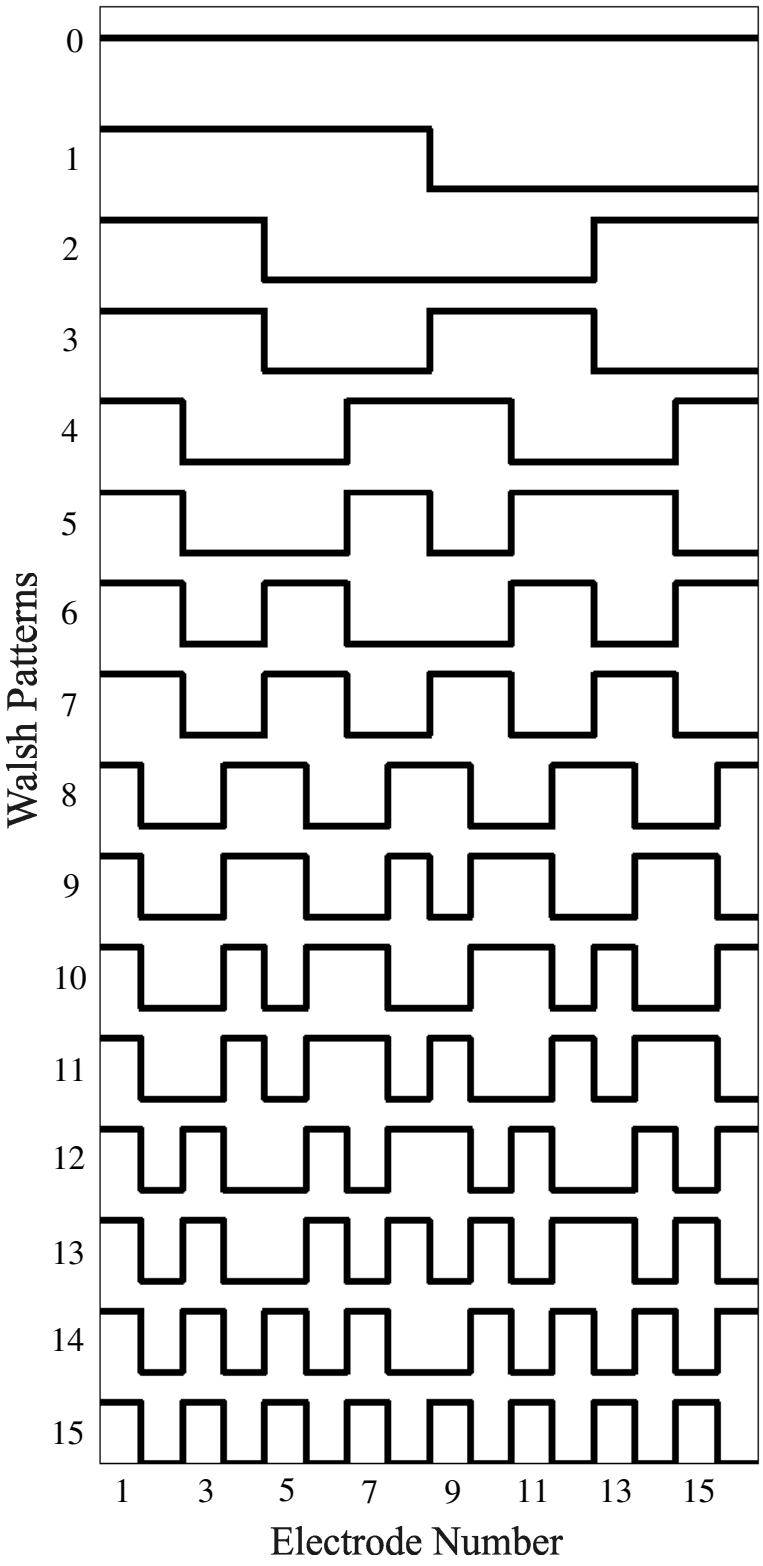


Figure B.1: Walsh patterns used for excitation of a 16-electrode system.

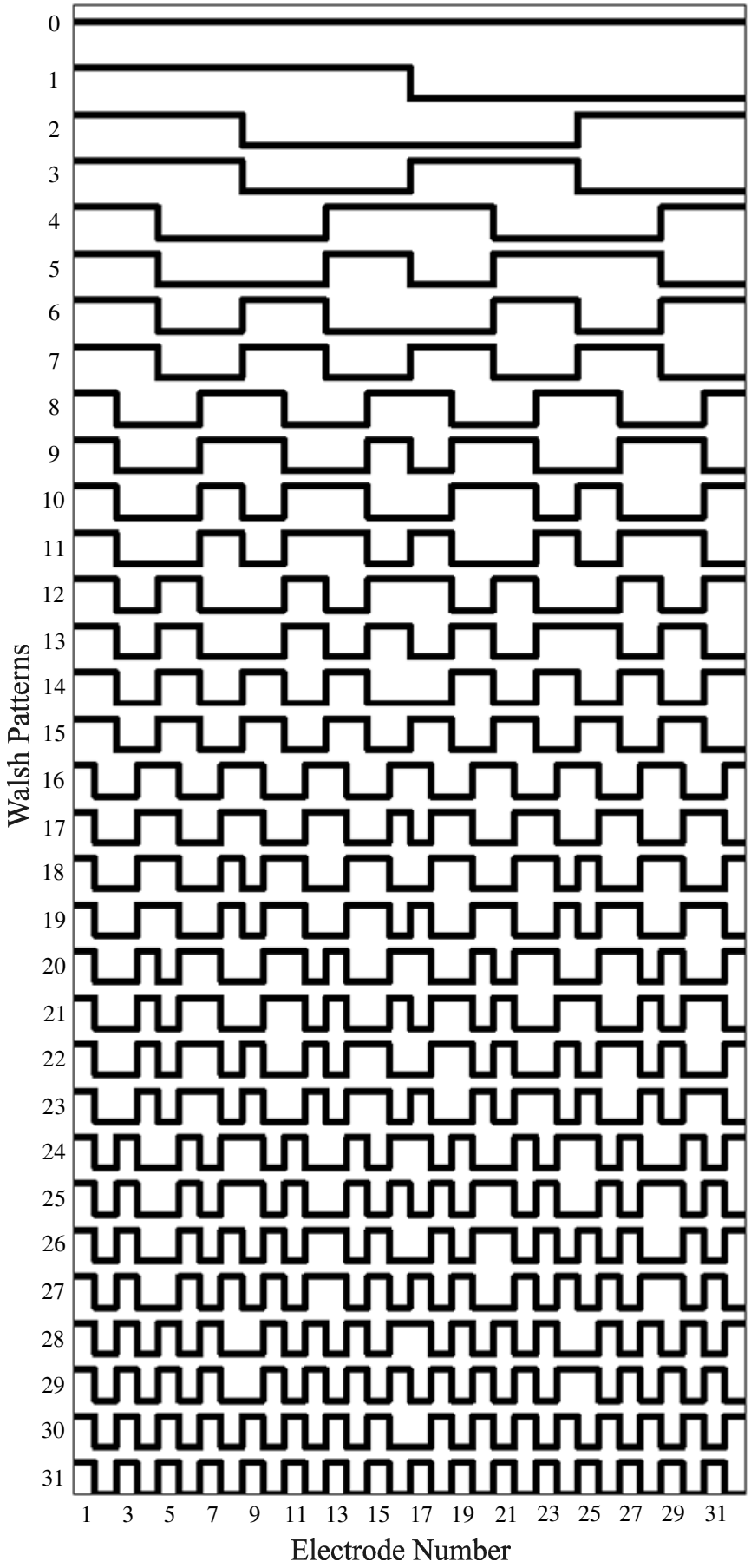
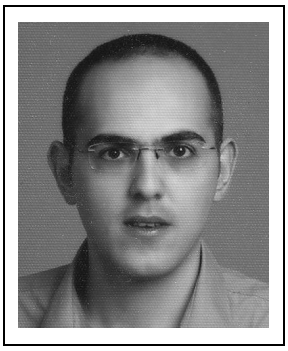


Figure B.2: Walsh patterns used for excitation of a 32-electrode system.

CURRICULUM VITA



Candidate's full name: **Çetin GÜREL**

Place and date of birth: **İstanbul, 24.03.1982**

Permanent Address: **İnönü Cad. No: 8 / 2 Tuna Apt. 34844 Maltepe - İSTANBUL**

Universities attended: **Yıldız Technical University – Mechanical Engineering, BSc, 2002-2006.**

Publications:

- **Çetin Gürel,** 2006. Development and Optimization of Condenser Design Algorithm using MATLAB, BSc Thesis. Yıldız Technical University, İstanbul.