

**DEVELOPMENT OF IMAGE
RECONSTRUCTION ALGORITHMS FOR
THREE DIMENSIONAL MAGNETIC
RESONANCE - ELECTRICAL IMPEDANCE
TOMOGRAPHY**

A THESIS

SUBMITTED TO THE DEPARTMENT OF ELECTRICAL AND
ELECTRONICS ENGINEERING

AND THE INSTITUTE OF ENGINEERING AND SCIENCE
OF BILKENT UNIVERSITY

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
MASTER OF SCIENCE

By

Serkan Onart

September, 2003

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

Prof. Dr. Y. Ziya İder (Supervisor)

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

Prof. Dr. Ayhan Altıntaş

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

Asst. Prof. Dr. Vakur B. Ertürk

Approved for the Institute of Engineering and Science:

Prof. Dr. Mehmet B. Baray
Director of the Institute Engineering and Science

ABSTRACT

DEVELOPMENT OF IMAGE RECONSTRUCTION ALGORITHMS FOR THREE DIMENSIONAL MAGNETIC RESONANCE - ELECTRICAL IMPEDANCE TOMOGRAPHY

Serkan Onart

M.S. in Electrical and Electronics Engineering

Supervisor: Prof. Dr. Y. Ziya İder

September, 2003

The electrical resistivity of biological tissues differ among various tissue types. Human body has a large resistivity contrast between a wide range of its tissues. The aim of this study is to reconstruct conductivity images of three dimensional objects with higher resolution and better accuracy than existing conductivity imaging techniques. In order to achieve our goal, we proposed a technique named as Magnetic Resonance - Electrical Impedance Tomography (MR-EIT) which combines the peripheral voltage measurements of classical Electrical Impedance Tomography (EIT) technique with magnetic flux density measurements acquired using a Magnetic Resonance Imaging (MRI) scanner. Five reconstruction algorithms are proposed and computer simulations are made. The proposed algorithms fall in two categories those that utilize current density data and those that utilize magnetic flux density data directly. The first group of algorithms get the current density data from magnetic flux density by Ampere's law. For calculation of current density with Ampere's law, we need to all three components of magnetic flux density but that is not possible to get all of them in one measurement phase. Total of three measurement phases are needed for getting all of them but this is not practical because, for measurement of each component the object has to be rotated appropriately in the MRI scanner. The algorithms in the second group suggest an exit to this difficulty and achieve the conductivity reconstruction by using only the data which was acquired in one measurement phase. As can be seen in the results, conductivity reconstruction of three dimensional objects on tomographic planes are made successfully with all of the algorithms. They also work fine against to the measurement noise up to an acceptable level.

Keywords: Magnetic Resonance - Electrical Impedance Tomography, Magnetic Resonance Imaging, Current Density Imaging, Impedance Imaging, Image Reconstruction, Finite Element Method.

ÖZET

MANYETİK REZONANS - ELEKTRİKSEL EMPEDANS GÖRÜNTÜLEMEDE ÜÇ BOYUTLU NESNELER İÇİN GÖRÜNTÜ GERİÇATMA ALGORİTMALARININ GELİŞTİRİLMESİ

Serkan Onart

Elektrik ve Elektronik Mühendisliği, Yüksek Lisans

Tez Yöneticisi: Prof. Dr. Y. Ziya İder

Eylül, 2003

Biyolojik dokuların elektriksel dirençleri çeşitli doku tipleri arasında çeşitlilik göstermektedir. Bu çalışmanın amacı, üç boyutlu nesnelere ait iletkenliklerinin varolan iletkenlik görüntüleme tekniklerinden daha yüksek çözünürlük ve daha iyi kesinlik ile geriçatılmasıdır. Amacımıza ulaşmak için Manyetik Rezonans - Elektriksel Empedans Tomografi (MR-EET) tekniği önerilmiştir. Bu teknik, klasik iletkenlik görüntüleme tekniği olan Elektriksel Empedans Tomografi'nin (EET) kullandığı yüzeysel voltaj ölçümlerine ek olarak Manyetik Rezonans Görüntüleme (MRG) tarayıcısı ile elde edilmiş manyetik akı yoğunluğunu da kullanmaktadır. Beş geriçatma algoritması önerilmiş ve bilgisayar simülasyonları yapılmıştır. Önerilen algoritmalar akım yoğunluğu dağılımını kullananlar ve direk manyetik akı yoğunluğu dağılımını kullananlar olarak iki kategoride incelenebilir. İlk kategorideki algoritmalar, akım yoğunluğu verisini Amper kuralı ile manyetik akı yoğunluğu verisinden elde ederler. Akım yoğunluğunun Amper kuralı ile elde edilebilmesi için manyetik akı yoğunluğunun her üç bileşenine de ihtiyacımız vardır. Ancak tüm bileşenlerin tek ölçüm safhasında elde edilmesi mümkün değildir. Tüm bileşenlerin elde edilebilmesi için toplam üç ölçüm yapılması gereklidir ki bu da pratik değildir. Çünkü her bileşenin ölçümü için obje, MRG sistemi içinde uygun şekilde döndürülmelidir. İkinci kategorideki algoritmalar, iletkenliği sadece tek ölçüm safhasında elde edilmiş veri ile geriçatarak bu probleme bir çıkış yolu önerirler. Sonuçlardan da görülebildiği gibi tüm önerilen algoritmalar, üç boyutlu objelerin iletkenliğini tomografik düzlemlerde başarılı bir şekilde geriçatabilmektedirler. Dahası, kabul edilebilir bir seviyeye kadar olan ölçüm gürültüsüne karşı bile gayet iyi çalışmaktadırlar.

Anahtar sözcükler: Manyetik Rezonans - Elektriksel Empedans Tomografi,

Manyetik Rezonans Görüntüleme, Akım Yoğunluğu Görüntüleme, Empedans Görüntüleme, Görüntü Geriçatma, Sonlu Elemanlar Yöntemi.

Acknowledgement

I am greatly indebted to my supervisor Prof. Dr. Y. Ziya İder for his instructive comments in the supervision of the thesis and for encouragement made throughout my graduate study.

I would like to express my special thanks and gratitude to Prof. Dr. Ayhan Altıntaş and to Asst. Prof. Dr. Vakur B. Öztürk the members of my jury, for showing their keen interest to the subject matter, reading and commenting on the thesis.

Finally, I wish to express my deep gratitude to my family for their support and patience.

To my family

Contents

| | |
|---|------------|
| List of Figures | xii |
| List of Tables | xv |
| 1 Introduction | 1 |
| 1.1 A Summary of Previous Studies on MR-EIT | 5 |
| 1.2 The Objective and Scope of the Thesis | 7 |
| 1.3 Organization of the Thesis | 8 |
| 2 The Forward Problem of MR-EIT | 9 |
| 2.1 Formulation of the Forward Problem | 10 |
| 2.2 Numerical Solution of the Forward Problem | 14 |
| 2.2.1 Finite Element Method | 14 |
| 2.2.2 Computation of Magnetic Flux Density | 18 |
| 3 The Inverse Problem of MR-EIT | 21 |

| | | |
|----------|---|-----------|
| 3.1 | Formulation of the Inverse Problem | 22 |
| 3.2 | Classification of the Reconstruction Algorithms | 26 |
| 4 | Current Density based Reconstruction | 28 |
| 4.1 | Reconstruction by Integration along Equipotential Lines | 29 |
| 4.1.1 | Method of Characteristic Curves | 29 |
| 4.2 | Reconstruction by Integration Along Cartesian Grid Lines | 34 |
| 4.3 | Reconstruction by Solution of Linear Equation System | 36 |
| 4.3.1 | Finite Difference Formulation | 36 |
| 5 | Magnetic Flux Density based Reconstruction | 40 |
| 5.1 | Reconstruction by Solution of Linear Equation System | 42 |
| 5.2 | Reconstruction by Sensitivity Matrix | 45 |
| 5.3 | Region of Interest Reconstruction for Iterative Algorithms | 51 |
| 6 | Simulation Results | 53 |
| 6.1 | Conductivity Models | 54 |
| 6.2 | Electrode models | 55 |
| 6.3 | Measurement Noise | 61 |
| 6.4 | Simulation Results for Current Density based Algorithms | 62 |
| 6.5 | Simulation Results for Magnetic Flux Density based Algorithms | 65 |

| | |
|---|------------|
| 6.6 Computational Cost of Algorithms | 81 |
| 7 Conclusions and Future Work | 93 |
| Bibliography | 96 |
| A Finite Element Formulation for 3D Objects | 102 |
| A.1 Weighted Residuals Method | 102 |
| A.2 Element Assembly | 104 |
| B Reconstruction of R on an Equipotential Surface | 107 |
| C Determination of the Scalar Factor for Absolute Imaging | 109 |
| D Source Codes for Matlab | 111 |

List of Figures

| | | |
|-----|--|----|
| 1.1 | EIT imaging scheme for a cylindrical object. | 3 |
| 1.2 | MIT imaging scheme for a cylindrical object. | 3 |
| 2.1 | MR-EIT imaging scheme for a cubical object. | 11 |
| 2.2 | Replacement of band type electrodes with point electrodes. | 13 |
| 2.3 | The mesh structure of cubical object consisting of hexahedrons. | 16 |
| 3.1 | Placement of the object in an MRI system in order to measure all three components of magnetic flux density. | 25 |
| 3.2 | Conductivity reconstruction on perpendicular planes. | 26 |
| 4.1 | Equipotential lines of two opposite current injection patterns. | 31 |
| 4.2 | Moving between equipotential surfaces of an injection pattern. | 33 |
| 4.3 | Finite difference approximation of derivatives on a slice. | 38 |
| 6.1 | Some selected slices of the first conductivity model. | 56 |

| | | |
|------|---|----|
| 6.2 | Some selected slices of the second conductivity model. | 57 |
| 6.3 | Some selected slices of the third conductivity model. | 58 |
| 6.4 | Electrode models used in simulations. | 60 |
| 6.5 | Results for the reconstruction along equipotential lines. | 68 |
| 6.6 | Results for the reconstruction along Cartesian grid lines. | 71 |
| 6.7 | The effect of electrode size on the interior current flow for two current injection profiles. | 72 |
| 6.8 | The effect of measurement noise on the interior current flow for two current injection profiles. | 73 |
| 6.9 | Results of reconstruction by solution of linear equation system (J based). | 76 |
| 6.10 | Assigned and reconstructed conductivity for model S3 after the first iteration. | 82 |
| 6.11 | Frequency response of the low-pass FIR blur filter. | 82 |
| 6.12 | Effect of iteration for, reconstruction by solution of linear equation system (B based). | 85 |
| 6.13 | Effect of RoI size for reconstruction by solution of linear equation system (B based). | 87 |
| 6.14 | Effect of SNR and injected current amount for, reconstruction by solution of linear equation system (B based). | 90 |
| 6.15 | Singular values of combined system matrix for different number of current injection profiles. | 91 |

| | |
|--|-----|
| 6.16 Results for the reconstruction by sensitivity matrix. | 92 |
| 6.17 Singular value plot of sensitivity matrices. | 92 |
| A.1 A pentahedron and three tetrahedrons constituting it. | 103 |

List of Tables

| | | |
|-----|--|----|
| 1.1 | Typical resistivity values of some biological tissues. | 2 |
| 6.1 | Regions conductivities of the conductivity models. | 54 |
| 6.2 | Electrode models used in simulations. | 55 |
| 6.3 | Simulations models. | 61 |

Chapter 1

Introduction

The electrical *resistivity* of biological tissues differ among various tissue types. Human body has a large resistivity contrast between a wide range of its tissues [1]. A brief summary of approximate resistivity values for important tissue types of human body are given in Table 1.1. Also, the physiological and pathological states of tissues reflect as resistivity variations [2]-[4]. Therefore, reconstructing the resistivity distribution of body would yield diagnostically valuable information about anatomy, physiological processes and pathology.

The *Electrical Impedance Tomography* (EIT) is a novel imaging technique to reconstruct resistivity distribution of the body [5]. It is technically based on generating a current distribution inside of the body either by injecting with surface electrodes or inducing by coils placed around the body and simultaneously measuring surface potential changes and/or outside magnetic field produced by internal current distribution. Measured field quantities contain information about the resistivity distribution of the body and can be extracted by suitable reconstruction algorithms. The found *image* of the resistivity is unique for noise-free complete boundary data [6]. A current-injected and voltage-measured EIT

| Tissue | Resistivity (Ωcm) |
|---------------------------------------|--------------------------------------|
| Blood masses | 15 |
| Bone | 17000 |
| Fat | 2000 |
| Heart fat | 2000 |
| Heart muscle | 450 |
| Human Body (avg.) | 460 |
| Left lung, Right lung | 1325 |
| Liver | 600 |
| Gray matter, White matter | 220 |
| Skeletal muscle (longitudinal fibers) | 300 |
| Skeletal muscle (transverse fibers) | 1500 |
| Skull | 17760 |
| Stomach | 400 |

Table 1.1: Typical resistivity values of some biological tissues.

configuration is given in Figure 1.1 with typical surface electrode positions. Generally, current injections and surface voltage measurements are made with the same electrode set. In a different type of EIT, the current is generated using surface electrodes again and the resulting magnetic field is measured with magnetometers outside the object [7]. A similar imaging technique, called *Magnetic Induction Tomography* [8] (MIT), uses a coil placed around the object, for both current induction and magnetic field measurement. Figure 1.2 demonstrates the MIT scheme with possible induced current path and direction for a cylindrical object.

When it is compared with other tomographic techniques like *Computerized X-ray Tomography* [9] (CT) and *Positron Emission Tomography* [10] (PET), EIT is about a thousand times cheaper, a thousand times smaller and requires no ionizing radiation. Further, EIT can in principle produce thousands of images per second. However due to noise and low sensitivity of boundary voltages to inner

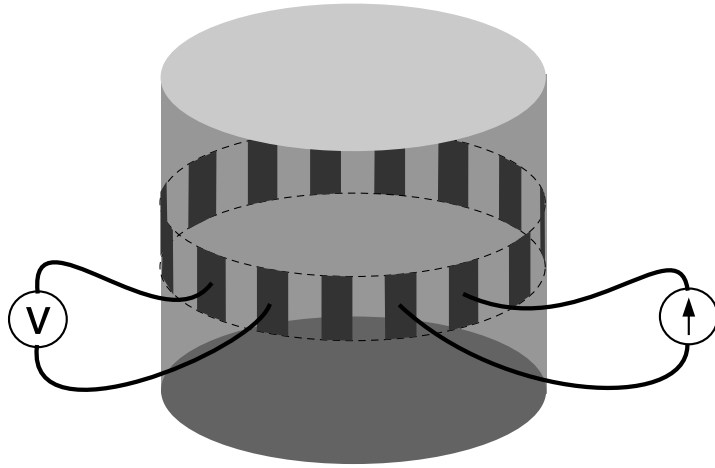


Figure 1.1: EIT imaging scheme for a cylindrical object. The surface electrodes are used for both current injection and voltage measurement.

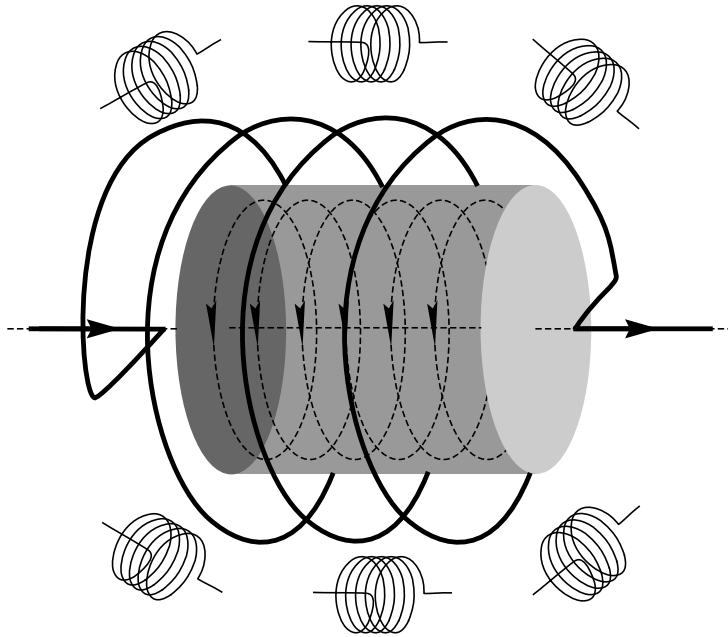


Figure 1.2: MIT imaging scheme for a cylindrical object. The current induction is made by the main coil and surrounding coils are used for magnetic field measurement. Induced circular currents are shown in dashed lines.

*conductivity** perturbations, and also due to practical problems with electrodes which allow for only a limited number of boundary voltage measurements, EIT can only yield inaccurate low resolution [11] images. The sensitivity and resolution degrade as the distance to the surface increases [12, 13]. Fixing of electrodes on the body is one of the remaining problems in the clinical use of EIT.

A solution to the position dependency problem of EIT is using a data set which is obtained from directly inside of the object. Making voltage measurements inside of the object is not possible non-invasively. Fortunately, it is possible to measure the *magnetic flux density* throughout the imaging region using a *Magnetic Resonance Imaging* (MRI) system with appropriate phase encoding sequences. Measurements can be made with high spatial sampling and also with high sensitivity even to inner conductivity perturbations. Reconstruction of conductivity images using measured magnetic flux density data with MRI system is named as *Magnetic Resonance - Electrical Impedance Tomography* (MR-EIT).

In other words, MR-EIT has been proposed to provide high resolution conductivity images by making use of an additional set of measurements which are made directly inside of the object. It uses an MRI scanner to obtain the distribution of induced magnetic flux density inside the object due to the internal current density distribution created by either injection with surface electrodes or induced with surrounded coils like EIT. Then, the reconstruction algorithms try to reconstruct the conductivity and/or *current density* images.

Inside current density distribution is also dependent on the size, shape and positions of surface electrodes in addition to its dependence on conductivity. Current injection with surface electrodes can be made in many different ways by using different electrode sets. In this study, two oppositely or diagonally

*Conductivity is the multiplicative inverse of resistivity.

placed electrodes are used as an electrode set. Each different electrode set and the amount of applied current is called a *current injection profile*.

MR-EIT makes use of the measurement techniques developed for *Magnetic Resonance Current Density Imaging* [14]-[18] (MR-CDI). In MR-CDI, magnetic flux density, \mathbf{B} , is measured using an MRI system, and the internal current density, \mathbf{J} , is obtained by $\mathbf{J} = \nabla \times \mathbf{B}/\mu_0$, the point form of Ampere's Law. Then, the MR-EIT reconstruction algorithms utilize either \mathbf{J} or \mathbf{B} in addition to peripheral voltage measurements to obtain high resolution conductivity images.

1.1 A Summary of Previous Studies on MR-EIT

The concept of MR-EIT has been introduced in 1992 by the MSc thesis [19] of N. Zhang. Zhang developed an algorithm, which is capable of reconstructing the correct image using internal current density and also the boundary voltage variation. Method is based on the fact that the potential difference between any two points on the boundary is the integral of *electrical field intensity*, \mathbf{E} , along any path connecting the points. Using the point form of Ohm's Law, $\mathbf{E} = \rho\mathbf{J}$, where ρ is the resistivity, and the measured \mathbf{J} on different paths connecting the two points, and also for different boundary point pairs, a linear system of equations can be obtained. Solution of this equation set, yields an image, which is unique and correct. The method is valid for 3D reconstructions, as well as for single slice imaging. A drawback of this method is the requirement of many boundary voltage measurements to improve the accuracy and resolution of reconstruction.

Eyüboğlu *et.al.* [20], Özdemir and Eyüboğlu [21] and Kwon *et.al.* [22] have proposed algorithms based on constructing the equipotential lines in the object using peripheral voltages and current density distribution. Current density inside the object is measured and it is known that the equipotential lines and current

lines are orthogonal. The potential and thus the electrical field distributions inside the object can be found by equipotential lines and projecting the peripheral voltage measurements into the *Field of View* (FoV) along these lines. Using the calculated electric field distribution and measured current density distribution, conductivity can be found for the entire FoV using Ohm's law. These methods, similar to the method of Zhang, are also non-iterative, and require peripheral voltage measurements and a single current injection pattern.

Woo *et.al.* [23] have proposed a reconstruction algorithm whereby the error between the current density measured by MR-CDI technique and the current density calculated by the *Finite Element Method* (FEM) is minimized as a function of the resistivity distribution. Kwon *et.al.* [24] have expanded on this idea to develop the *J-substitution Algorithm*, which uses at least two injected current patterns and a single voltage measurement to reconstruct the correct image. They also claim that at any point in the object, current densities measured for the two current injection patterns must not be parallel. This requirement of at least two injected current patterns is rigorously proven later by Kim *et.al.* [25]. Khang *et.al.* [26] have applied the J-substitution algorithm successfully to data obtained from saline phantoms. Both methods in [23] and [24] are iterative. Another iterative method which is proposed by Eyüboğlu *et.al.* [27], is based on minimizing the error between measured and calculated current densities and peripheral voltages simultaneously. Recently Birgül *et.al.* [28] have proposed another iterative method in which a single voltage measurement and eight current injection patterns are used. Their method is similar to the J-substitution algorithm in concept, but they have also studied its performance under opposite drive and cosine injection patterns.

İder *et.al.* [29, 30] have developed and applied to real data a method for reconstructing from the measured magnetic field without having to calculate the current density, using an iterative sensitivity matrix approach. Seo *et.al.* [31] have

also proposed a method which makes use of \mathbf{B} only. In both of these methods only a single component of \mathbf{B} is used. This provides a major practical advantage over the methods utilizing \mathbf{J} because to obtain \mathbf{J} by taking the curl of \mathbf{B} requires the measurement of its all three components.

1.2 The Objective and Scope of the Thesis

Almost all medical imaging systems concentrate on only the slice of interest of the body when trying to reconstruct their unknown parameter as image. At this time the other regions of body don't have any effect on reconstructed slice or slices. For example, when taking x-ray tomography image of breast, the projection of attenuation coefficient along selected directions are found and the attenuation coefficients of other tissues, *e.g.* stomach, have no effect on this process. Unfortunately, MR-EIT and in general EIT don't provide this useful feature as a consequence of their structure, because the injected or induced current in the slice of interest can easily change its magnitude and direction if resistivity of a part near the slice of interest changes slightly. Also the mesh generation and solution of the differential equation on this mesh which describe the voltage distribution in the object, requires big computational effort for 3D objects. For circumventing these difficulties, MR-EIT has been formulated and implemented for 2D objects.

This thesis is devoted to develop MR-EIT image reconstruction algorithms for 3D objects which are insensitive to off-slice effects and require less computation times for reconstruction. None of the proposed algorithms are implemented on real objects, instead only computer simulations are made. Also the required magnetic flux density data are generated by computer. When doing this, it is always assumed that current is injected to the object with electrodes attached to

the surface of object. Implementation of the algorithms using current induction methods and developing reconstruction algorithms specific to them are outside of the scope of this thesis.

1.3 Organization of the Thesis

The thesis is organized as follows: Chapter 2 describes the mathematical basis of MR-EIT and for a known conductivity distribution, all required processes to calculate field quantities starting from potential distribution to the magnetic flux density distribution which are generated as a result of current injection. In Chapter 3, basic formulations leading to the reconstruction algorithms are derived. This is done by assuming that only the measured magnetic flux density, boundary shape of the object and amount of injected current with size and positions of the electrodes are known. Chapter 4 includes the explanation of novel reconstruction algorithms which utilize the current density distribution which can be obtainable from measured magnetic flux density by a curl operation. Chapter 5, describes the other type of reconstruction algorithms, utilizing the magnetic flux density directly. Also these algorithms need only one component of magnetic flux density so they are more practical than the previous ones. Chapter 6 introduces the conductivity models used for computer simulations, describes how measurement noise can be simulated and gives all simulation results. Finally, Chapter 7 concludes the thesis.

Chapter 2

The Forward Problem of MR-EIT

In MR-EIT, a direct current is injected to the object with surface electrodes and current distributes inside as a function of conductivity distribution. If a non-alternating current flows on a conductive media then static potential and magnetic flux density distributions accompany it. With today's technology, the only measurable field quantity inside the object is magnetic flux density. This gives us the idea of reaching the conductivity by using magnetic flux density. For achieving this, first we have to understand and formulate what happens in the object when current is injected. This Chapter describes the formulation of field quantities, by starting from potential distribution to magnetic flux density distribution, for a known conductivity distribution. The formulation is called the *Forward Problem* of MR-EIT.

Forward problem is also a useful tool especially for iterative reconstruction algorithms. Generally this type of algorithms start with an initially taken conductivity, solves the forward problem and checks for errors between calculated and measured field quantities. Iterations continue by updating the conductivity in a manner and solving the forward problem again.

2.1 Formulation of the Forward Problem

The injection of direct current I into an isotropic nonmagnetic and conductive object, occupying volume of Ω in space with boundary $\partial\Omega$, generates a static and conductivity related current density distribution inside of the object. Current injection is made for a finite time duration by surface electrodes which are attached to some part of $\partial\Omega$. An illustration of this is given in Figure 2.1. The injected current always leaves the object by a second surface electrode. So there is no current source or sink point inside of Ω . The time interval of current injection process is adequately short so that conductivity distribution can be assumed to be time independent in this duration. With these assumptions, right hand sides of Maxwell's first equation and current conservation law vanish and reduce to the following equations respectively,

$$\nabla \times \mathbf{E} = 0 \tag{2.1}$$

$$\nabla \cdot \mathbf{J} = 0 \tag{2.2}$$

where \mathbf{E} is electric field intensity and \mathbf{J} is current density inside Ω . A curl-free vector field can be written as divergence of a unique scalar field. This rule is valid for electric field intensity too, and therefore, it is equal to the negative gradient of the potential field ϕ as shown by the following equation

$$\mathbf{E} = -\nabla\phi \tag{2.3}$$

Furthermore there is a relation between electric field intensity and current density known as Ohm's Law,

$$\mathbf{J} = \sigma\mathbf{E} \tag{2.4}$$

or

$$\mathbf{E} = \rho\mathbf{J} \tag{2.5}$$

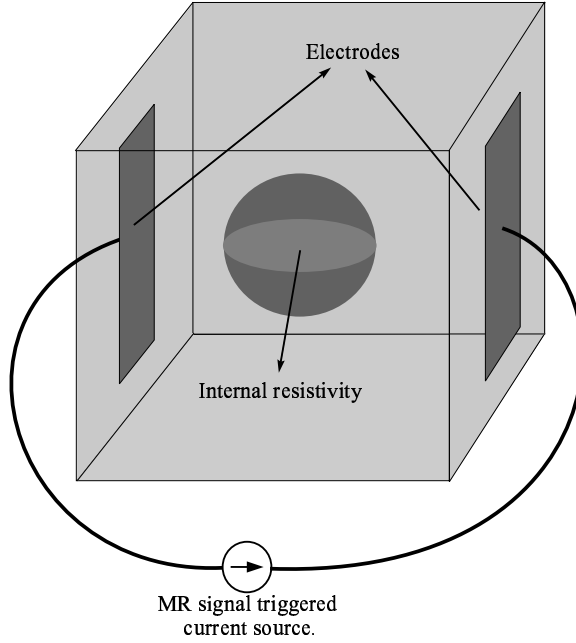


Figure 2.1: A cubical object with a spherical resistivity region different from background resistivity. Current is injected with surface electrodes and current source is triggered with MR signal appropriately.

where σ and ρ are conductivity and resistivity distributions respectively. Substitution of Eq. 2.3 into (2.4) and substitution of the resultant equation into (2.2) gives a nonlinear elliptic *Partial Differential Equation* (PDE),

$$\nabla \cdot \sigma \nabla \phi = 0 \quad (2.6)$$

which is a relation between conductivity and potential distribution. Solution of this equation for a known conductivity distribution gives a potential field.

Eq. 2.6 is defined on a finite domain Ω so it is a *Boundary Value Problem*. This means that, infinitely many solutions exist for a known conductivity distribution, without boundary conditions. In order to find the unique and correct solution, a proper boundary value condition has to be known in addition to the conductivity. The two types of boundary conditions are *Dirichlet* and *Neumann* boundary conditions. Dirichlet boundary condition requires us to know the value of potential at the boundary. Neumann boundary condition requires us to know the

normal derivative of potential at the boundary. In our case, Neumann boundary conditions apply. From Eq. 2.3, the normal derivative of potential at the boundary is the component of electric field aligned with, the outside directed surface normal vector of $\partial\Omega$, with a minus sign in front of it. Using this fact and Eq. 2.4 we obtain,

$$\frac{\partial\phi(\mathbf{s})}{\partial\vec{n}} = -\frac{\mathbf{n} \cdot \mathbf{J}}{\sigma} \quad (2.7)$$

where \mathbf{s} is the boundary vector and \vec{n} or \mathbf{n} are the unit outward normal vector all defined on the three dimensional object. By this equation we need the normal component of current density to find normal derivative of the potential. We know the quantity of applied current I , and also know shape, position, and the total area of surface electrodes where the current is applied. Using them and assuming every point of electrodes behaves like a current source, surface current densities can be defined as,

$$\mathbf{J}_{\mathbf{s}+} = \frac{I}{A_+} \quad (2.8)$$

and

$$\mathbf{J}_{\mathbf{s}-} = -\frac{I}{A_-} \quad (2.9)$$

where A_+ and A_- are the areas of two surface electrodes, current entered and left, $J_{\mathbf{s}+}$ and $J_{\mathbf{s}-}$ are the corresponding surface current densities. Therefore Neumann boundary condition can be written as,

$$\frac{\partial\phi(\mathbf{s})}{\partial\vec{n}} = \begin{cases} -J_{\mathbf{s}+}/\sigma(\mathbf{s}), & \text{on current enterece electrode,} \\ -J_{\mathbf{s}-}/\sigma(\mathbf{s}), & \text{on current exit electrode,} \\ 0, & \text{elsewhere.} \end{cases} \quad (2.10)$$

In practice, generally the assumption expressed above not hold because of nonconstant conductivity near to the electrodes. So surface current densities on the electrodes vary with position. This situation changes the boundary condition a little but solution may change completely. In order to get rid of this handicap, replacing the place of band type electrodes with finite number of

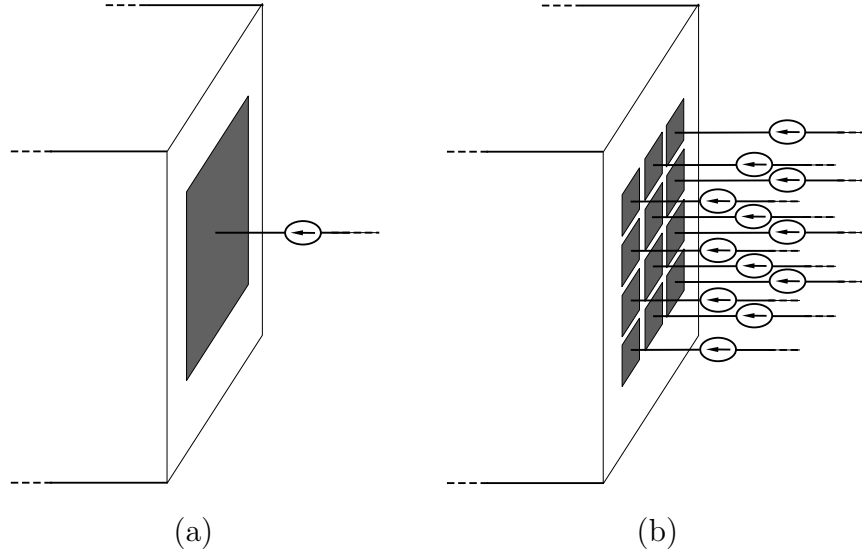


Figure 2.2: Replacement of band type electrodes with point electrodes prevents the current density inhomogeneity problem on the electrodes. (a) A band type electrode, driven with a single current source. (b) Point type electrodes driven with separate current sources.

point electrodes and driving each of them with different current sources may be useful. In Figure 2.2 such a replacement is shown.

Note that, for Neumann boundary value problem a reference potential is necessary because adding a constant to the found potential field also satisfies Eq. 2.6 and Neumann boundary condition. It is sufficient to select a node and set its potential to zero before solving Eq. 2.6.

After potential field is found, electric field intensity and current densities can be found easily using Eqs. 2.3 and 2.4 consecutively.

The current passing through object also generates an observable physical field quantity known as magnetic flux density \mathbf{B} , inside and outside of the object. Magnetic flux density can be measured by using an MRI scanner with current density imaging techniques. So it is an important field quantity for some MR-EIT image reconstruction algorithms and has to be a member of the forward problem.

It is meaningless in MR-EIT to find \mathbf{B} outside of the object because the mentioned technique is only capable of measuring inside magnetic flux density. Therefore as a last step of the forward problem internal \mathbf{B} has to be found. The relation between current density and magnetic flux density is Biot-Savart Rule,

$$\mathbf{B}(\mathbf{r}) = \frac{\mu_0}{4\pi} \int_{\Omega} \mathbf{J}(\mathbf{r}') \times \frac{\mathbf{r} - \mathbf{r}'}{|\mathbf{r} - \mathbf{r}'|^3} dv' \quad (2.11)$$

where μ_0 is the permeability of free space and also nonmagnetic objects, \mathbf{r} is the unit field vector defined from origin to source point (x, y, z) and \mathbf{r}' is the unit field vector defined from origin to field point (x', y', z') . Both the source and field points are elements of region Ω .

2.2 Numerical Solution of the Forward Problem

The complete solution of the forward problem requires the successive solution of Eqs. 2.6, 2.3, 2.4 and 2.11. The first equation of this solution chain is the most difficult one. Generally it is not possible to find an analytical solution to this equation. This situation obligates us to try numerical solution methods. One of the most popular numerical solution method utilized for approximate solution of the differential equations in two or more higher dimensions is *Finite Element Method* or simply *FEM*. A brief description of the FEM used in this study is given in next section. Details and mathematical formulations are given in Appendix A.

2.2.1 Finite Element Method

In FEM, the exact value of the solution is found only at finite number of sampling points, as called the *nodes*, instead of at every point of the domain. Nodes are also the corners of the *finite elements* which subdivide the solution domain into small

closed regions. The solution domain for a geometry which contains elements and nodes is called the *finite element mesh*.

It is easy to write an approximation for the solution inside of an element by using interpolating functions and node potentials. The degree of the approximation depends on the used interpolating functions. For most cases a first order approximation is enough to replace the exact solution with the approximated one. To ensure this, adequately small finite elements have to be used. Using small elements provide us a smooth variation for the approximation and decrease the error. Increasing the number of finite elements and nodes in Ω decreases the size of elements and gives a finer mesh, but increases the time for computing the approximated solution.

In this study tetrahedral elements are used as finite elements. In three dimensional space, tetrahedron is the simplest volume element containing the minimum number of corners in order to make a volume. The object and solution domain are assumed as rectangular prism. It is first divided into 64 slices in z direction. Then every slice is divided by 32×32 hexahedron elements. Conductivity is assumed constant in a hexahedron. As will be described in the next section, magnetic flux density is found in hexahedron centers. Also all proposed reconstruction algorithms run on this mesh. This structure of the mesh is shown in Figure 2.3. For FEM solution, a mesh structure constructed with tetrahedrons is needed. Therefore, every hexahedron of mesh is subdivided diagonally into two pentahedrons and every pentahedron is subdivided into three tetrahedrons. In Figure A.1, detailed drawings for a pentahedron and its tetrahedrons can be found. First order linear interpolating functions are used and potential is assumed as varying linearly between four nodes in each tetrahedron. The solution is same at the common surfaces of tetrahedrons so there is no gap or discontinuity at the potential. For each tetrahedron, internal approximated potential field is expressed in terms of node potential values and

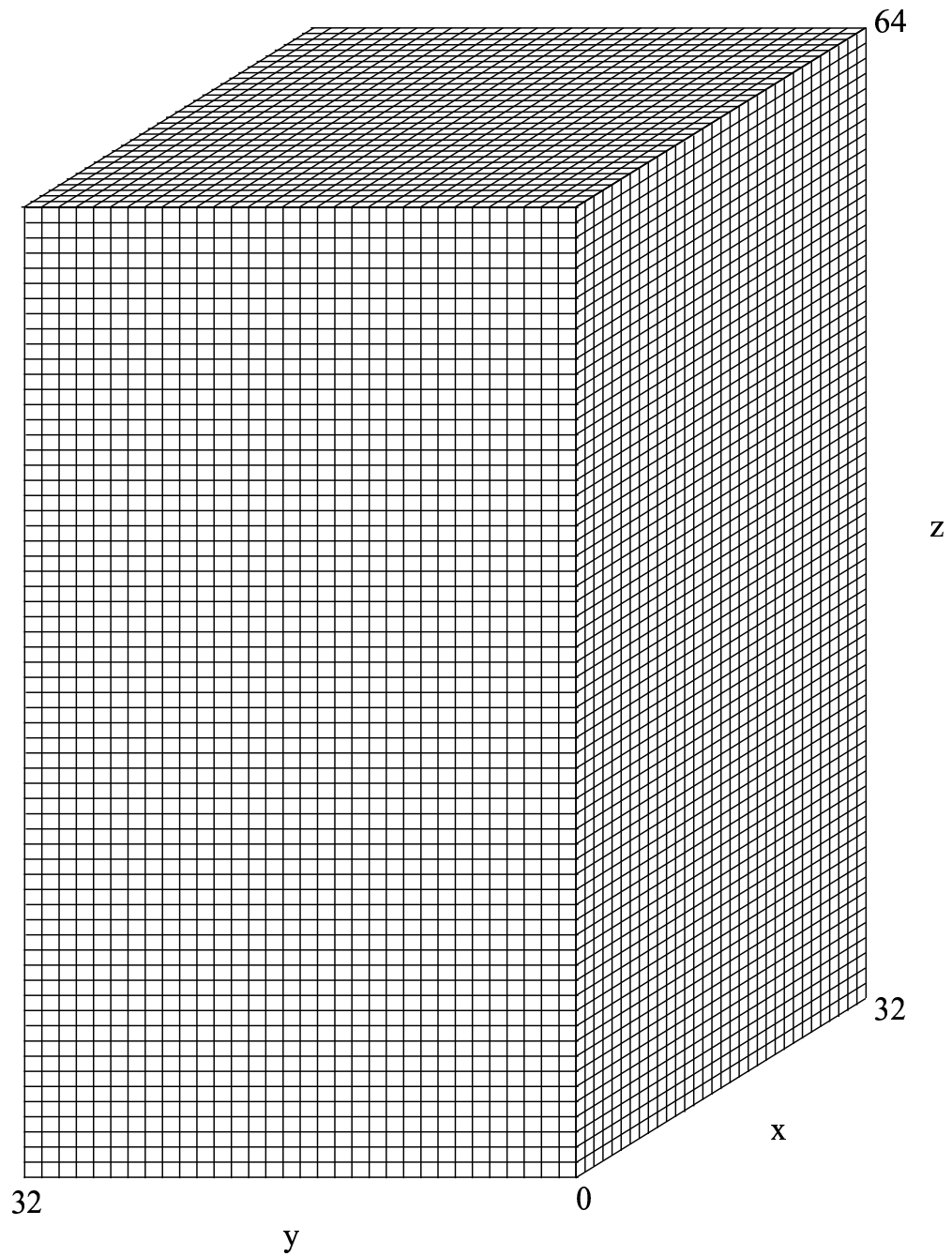


Figure 2.3: The object is first divided into 64 slices in z direction. Then each slice is subdivided into 32×32 hexahedrons. Relative node numbers are also shown. All edges of the hexahedrons are equal and assumed as in 1cm length. Physical dimensions of object are important for units of field quantities.

spatial coordinates, with a linear equation. Eq. 2.6 is converted to a matrix relation by combining the linear equations obtained from all tetrahedrons. The form of this relation is shown below,

$$\mathbf{A}_{\text{con}}\boldsymbol{\phi} = \mathbf{b} \quad (2.12)$$

where \mathbf{A}_{con} is the sparse connected coefficient matrix depending on the conductivity and mesh structure, $\boldsymbol{\phi}$ is the node potentials vector to be found, \mathbf{b} is the boundary condition vector. For n nodes in mesh, the size of coefficient matrix \mathbf{A}_{con} is $(n \times n)$ and the size of column vectors $\boldsymbol{\phi}$ and \mathbf{b} is $(n \times 1)$. Node potentials vector $\boldsymbol{\phi}$ can be calculated as,

$$\boldsymbol{\phi} = \mathbf{A}_{\text{con}}^{-1} \mathbf{b} \quad (2.13)$$

by inversion and multiplication operations.

The square matrix \mathbf{A}_{con} has to be nonsingular in order to be invertible. Singularity means that the rank of \mathbf{A}_{con} is lower than the number of rows, n . As explained in the previous section, for uniqueness, Neumann type boundary value problem needs a node with reference potential. The value of the reference is not limited by any number, but with common sense it is accepted as zero. The rank of \mathbf{A}_{con} can be at most $n - 1$ unless the reference potential is set. For numerically setting the reference potential, some manipulations have to be done on the entries of coefficient matrix \mathbf{A}_{con} so that its rank is equal to n . For example to set the potential of i^{th} node to zero as reference potential, i^{th} row of \mathbf{A}_{con} have to be set to zero except the $(i, i)^{\text{th}}$ entry which is set to unity and also i^{th} element of vector \mathbf{b} is set to zero.

The next steps of numerical forward problem solution include the calculations of electric field intensity, current density and magnetic flux density. Numerical calculation of first two items is easier than the last item in terms of computation time. The approximated potential field varies linearly in all three dimensions

inside of the elements as,

$$\Phi_i = \begin{cases} A_i x + B_i y + C_i z + D_i, & \text{Inside of the } i. \text{ element,} \\ 0, & \text{Outside of the } i. \text{ element.} \end{cases} \quad (2.14)$$

where A_i , B_i , C_i and D_i are the constant coefficients effected by physical coordinates and node potentials of i^{th} element. From Eqs. 2.3 and 2.14 it can be seen that electric field intensity is constant inside of the tetrahedrons. For the i^{th} tetrahedron its form is,

$$\mathbf{E}_i = -(A_i \mathbf{x} + B_i \mathbf{y} + C_i \mathbf{z}) \quad (2.15)$$

where \mathbf{x} , \mathbf{y} and \mathbf{z} are basis unit vectors. As stated earlier conductivity is also constant inside of the tetrahedrons. This assumption makes easier the calculation of current density. From Eq. 2.4, to calculate the current density, simply three components of the electric field are divided by the corresponding conductivity,

$$\mathbf{J}_i = -\frac{1}{\sigma_i}(A_i \mathbf{x} + B_i \mathbf{y} + C_i \mathbf{z}). \quad (2.16)$$

\mathbf{J}_i is constant too throughout the i^{th} tetrahedron like \mathbf{E}_i .

2.2.2 Computation of Magnetic Flux Density

Biot-Savart relation analytically formulates the contributions of all differential current elements to the magnetic flux density at every point of space. The numerical and discrete calculation of magnetic flux density requires the discretization of Biot-Savart relation which was given in Eq. 2.11. Generally the reconstruction algorithms utilizing magnetic flux density, necessitate the knowledge of \mathbf{B} on a cartesian grid for simplicity. For this reason magnetic flux density will be found at the centers of the hexahedrons which are ordered regularly in the cubical object.

The piece of induced magnetic flux density at the center of j^{th} hexahedron by

currents inside of the i^{th} tetrahedron can be written as,

$$\mathbf{B}_{ji} = \frac{\mu_0}{4\pi} \int_{i.tetra} \mathbf{J}_i \times \frac{\mathbf{r} - \mathbf{r}'}{|\mathbf{r} - \mathbf{r}'|^3} dv'. \quad (2.17)$$

As stated earlier \mathbf{J}_i is constant throughout the integration limits but cannot be taken outside of integration because of the curl operation which is still a function of \mathbf{r}' . If the tetrahedron is small enough and field points are adequately far from its gravity center then source points can be replaced with gravity center point of tetrahedron as below,

$$\mathbf{r} - \mathbf{r}' \approx \mathbf{r} - \mathbf{r}'_g \triangleq \mathbf{R} \quad (2.18)$$

where \mathbf{R} is the distance vector between field point \mathbf{r} and center of gravity point \mathbf{r}_g of tetrahedron. Using this approximation, Eq. 2.17 can be written as,

$$\mathbf{B}_{ji} = \frac{\mu_0}{4\pi} \mathbf{J}_i \times \frac{\mathbf{R}_{ji}}{|\mathbf{R}_{ji}|^3} V_i. \quad (2.19)$$

where \mathbf{R}_{ji} is distance vector between center of gravity of the i^{th} tetrahedron and geometric center of the j^{th} hexahedron, V_i is volume of tetrahedron. To calculate magnetic flux density at a point, contributions of all tetrahedrons must be summed up,

$$\mathbf{B}_j = \frac{\mu_0}{4\pi} \sum_{i=1}^T \mathbf{J}_i \times \frac{\mathbf{R}_{ji}}{|\mathbf{R}_{ji}|^3} V_i. \quad (2.20)$$

where $j = 1, 2, \dots, H$, H is number of hexahedrons and T is number of tetrahedrons. All three components of \mathbf{B} can be calculated independently by evaluating the curl operation,

$$\mathbf{B}_x^j = \frac{\mu_0}{4\pi} \sum_{i=1}^T \frac{\mathbf{J}_y^i R_z^{ji} - \mathbf{J}_z^i R_y^{ji}}{|\mathbf{R}_{ji}|^3} V_i \quad (2.21)$$

$$\mathbf{B}_y^j = \frac{\mu_0}{4\pi} \sum_{i=1}^T \frac{\mathbf{J}_z^i R_x^{ji} - \mathbf{J}_x^i R_z^{ji}}{|\mathbf{R}_{ji}|^3} V_i \quad (2.22)$$

$$\mathbf{B}_z^j = \frac{\mu_0}{4\pi} \sum_{i=1}^T \frac{\mathbf{J}_x^i R_y^{ji} - \mathbf{J}_y^i R_x^{ji}}{|\mathbf{R}_{ji}|^3} V_i \quad (2.23)$$

geometric centers of hexahedrons and gravity centers of tetrahedrons never coincide in object so there is no any possibility of singularity chance for calculation of \mathbf{B} with the equations given.

Computation of \mathbf{B} takes most of the time in forward problem solution especially if number of elements are increased more. For example computation of only one component of \mathbf{B} for a cubic object consisting of 64 slices with 32×32 hexahedrons in every slice takes several hours. To reduce the computation time, pentahedrons can be used instead of tetrahedrons without increasing error much. The constant current value inside of a pentahedron can be assigned as the mean of its total current which is the summation of the currents of its three tetrahedrons. So required computation time is reduced by a factor of three but the error is not increased much. To check this, a comparison between correct and computed solutions is required. Symbolic solution packets can find an analytic solution of \mathbf{B} if conductivity is constant in the object. It is seen that for constant conductivity, maximum and mean proportional differences and ℓ^2 norm of the difference between analytical and numerical solutions are 2.47%, 0.05% and 0.056% respectively.

Chapter 3

The Inverse Problem of MR-EIT

The aim of MR-EIT is to reconstruct the unknown interior resistivity distribution of three dimensional objects. In the previous chapter it is explained how some measurable field quantities can be calculated numerically from a known conductivity distribution and boundary conditions. The calculation of these physical quantities by starting with a known conductivity and boundary conditions and solving elliptic equation is named as the *Forward Problem*, and the computer program or tool realizing this process numerically is a *Forward Solver*. Instead, the *Inverse Problem* includes the formulation of the field and conductivity relations and solution of the unknown conductivity distribution from them.

In spite of the well-defined formulation of forward problem, it is not easy to find simple equations describing the inverse problem sufficiently clearly. Furthermore, they don't have analytic solutions, includes highly nonlinear relations, and require non-standard, equation-specific solution techniques. A systematic solution method of resistivity from an equation is named as a *Reconstruction Algorithm*. Reconstruction algorithms may use inside measured magnetic flux

density, some peripheral voltage measurements and boundary information which includes shape and position of boundary and electrodes.

An inverse problem relation may include some of the field quantities as equation elements. But no method has been developed yet to directly measure all field quantities inside of the object. Only the magnetic flux density can be measured by calculating the phase difference of magnetic resonance images obtained when a current exist and when not exist in the object. As a consequence of this, in MR-EIT, all reconstruction algorithms have to utilize magnetic flux density first. Once magnetic flux density is found with all of its components, *e.g.* current density can be calculated by a simple curl operation.

3.1 Formulation of the Inverse Problem

Assume that, an object Ω is in same conditions with the object which was described in the beginning of previous chapter and satisfies all assumptions we have made for it. So electric field is curl-free inside of it,

$$\nabla \times \mathbf{E} = 0. \quad (3.1)$$

As given before, Ohm's Law states that $\mathbf{E} = \rho \mathbf{J}$. Substituting this in previous one yields,

$$\nabla \times \rho \mathbf{J} = 0 \quad (3.2)$$

which can be written in an equivalent form as,

$$\nabla \rho \times \mathbf{J} + \rho \nabla \times \mathbf{J} = 0 \quad (3.3)$$

by using vector identity $\nabla \times \psi \mathbf{A} = \nabla \psi \times \mathbf{A} + \psi \nabla \times \mathbf{A}$ for scalar field ψ and vector field \mathbf{A} . Divide both sides with ρ and pass over the curl of \mathbf{J} to right hand side,

$$\frac{\nabla \rho}{\rho} \times \mathbf{J} = -\nabla \times \mathbf{J}. \quad (3.4)$$

For simplicity, rewrite this equation in terms of the natural logarithm of resistivity as,

$$\nabla R \times \mathbf{J} = -\nabla \times \mathbf{J} \quad (3.5)$$

where $R = \ln \rho$.

If \mathbf{J} is known, we can interpret Eq. 3.5 as yielding information on the gradient of R . Inside current density can be calculated by $\mathbf{J} = \nabla \times \mathbf{B}/\mu_0$, the point form of Ampere's Law. Instead of calculating \mathbf{J} by Ampere's law, substituting it into (3.5) and using a vector identity may produce a direct relation between magnetic flux density and logarithmic resistivity. Substitution of Ampere's law in place of \mathbf{J} , at the right-hand side of Eq. 3.5 yields,

$$\nabla R \times \mathbf{J} = -\nabla \times (\nabla \times \mathbf{B}/\mu_0). \quad (3.6)$$

Using vector identity $\nabla \times (\nabla \times \mathbf{A}) = \nabla(\nabla \cdot \mathbf{A}) - \nabla^2 \mathbf{A}$ for any vector field \mathbf{A} gives that,

$$\nabla R \times \mathbf{J} = -\nabla(\nabla \cdot \mathbf{B}/\mu_0) + \nabla^2 \mathbf{B}/\mu_0. \quad (3.7)$$

Magnetic flux density is a solenoidal field and its divergence vanishes always. In the result we obtain,

$$\nabla R \times \mathbf{J} = \nabla^2 \mathbf{B}/\mu_0. \quad (3.8)$$

This equation utilizes the Laplacian of \mathbf{B} but it still requires us to know the current density. Furthermore calculating the Laplacian of magnetic flux density requires taking the second order derivatives of \mathbf{B} which will increase the measurement noise and so decreases quality of the reconstructed resistivity. Obviously \mathbf{J} can be calculated by Ampere's law and in that case, using Eq. 3.5 is more sensible because it does not include a "noise increasing" Laplacian operation.

Despite its poor sides, Eq. 3.8 has a considerable big advantage against to (3.5). In order to see this, a technical problem about the measurement of \mathbf{B} with

an MRI scanner has to be known. Calculation of current density from magnetic flux density by Ampere's law, requires us to measure \mathbf{B} in $(\mathbf{x}, \mathbf{y}, \mathbf{z})$ directions, because curl operation needs the all three components of \mathbf{B} . As explained before, measurement of \mathbf{B} by an MRI scanner is made by calculating the phase shifts between two images obtained by an appropriate pulse sequence. The problem is that, MRI system is capable of detecting phase shifts only for one component of \mathbf{B} which is aligned with its main magnets magnetic field direction. Measuring the other components necessitates rotating the object appropriately and repeating the same pulse sequence for other different orientations. Perhaps that is not a big problem for experimental small objects but for a human body is not possible. Placement of the object in the MRI system to measure all three components of magnetic flux density is given in Figure 3.1. In order to handle this difficulty we need a reconstruction algorithm which utilize only one component of magnetic flux density and cancel the measurement of other two components of \mathbf{B} . The advantage of (3.8) appears right here.

Lets now return to Eq. 3.5. To gain further insight about it, expand its components in all three directions,

$$\begin{bmatrix} 0 & J_z & -J_y \\ -J_z & 0 & J_x \\ J_y & -J_x & 0 \end{bmatrix} \begin{bmatrix} \frac{\partial R}{\partial x} \\ \frac{\partial R}{\partial y} \\ \frac{\partial R}{\partial z} \end{bmatrix} = - \begin{bmatrix} \frac{\partial J_z}{\partial y} - \frac{\partial J_y}{\partial z} \\ \frac{\partial J_x}{\partial z} - \frac{\partial J_z}{\partial x} \\ \frac{\partial J_y}{\partial x} - \frac{\partial J_x}{\partial y} \end{bmatrix} \quad (3.9)$$

which provides an equation system consisting of three first order quasi-linear* partial differential equations. Each row of this system conveys information about gradient of R on two dimensional domains which are planes actually. For example first row deals with the gradient of R on constant x planes. Similarly second and third rows are related with constant y and z planes respectively. Therefore planes of different equations are perpendicular to each other. An illustration of perpendicular planes in the object is given in Figure 3.2. It can be seen

*A special form of the nonlinear partial differential equations [32].

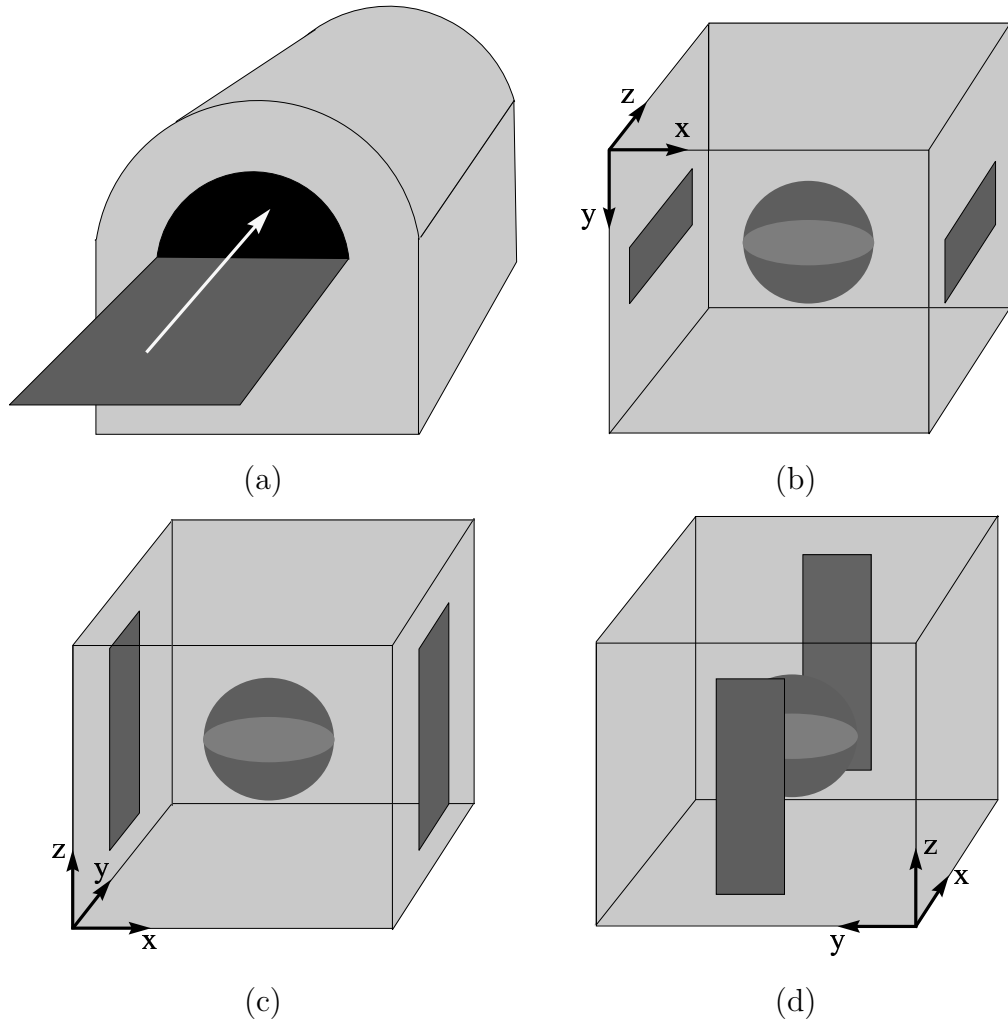


Figure 3.1: MRI system is sensible to the variation of magnetic flux density only along its main magnets direction. To measure all three components of \mathbf{B} , object has to be rotated properly and measurement process has to be restarted. (a) An MRI system. Its main magnets direction is shown with a white arrow. (b) Object placement to measure \mathbf{B}_z . (c) Object placement to measure \mathbf{B}_y . (d) Object placement to measure \mathbf{B}_x .

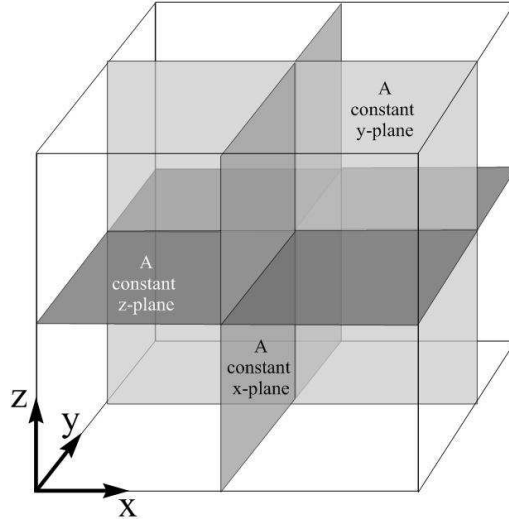


Figure 3.2: If the \mathbf{J} is known, conductivity can be reconstructed on this planes by solving each row of Eq. 3.9 separately.

clearly that, if \mathbf{J} is known on a plane then reconstructing the conductivity of that plane is easy. It also gives us the ability of single slice reconstruction, and even, reconstructing only some partial area of slice instead of whole object.

Note that, still we do reconstruction for a 3D object but find resistivity only on the region of interest. So computation time for reconstruction reduces significantly. If desired, whole object reconstruction can be made by selecting any one row of (3.9) and making reconstruction on parallel planes consisting of the object. All proposed algorithms in this study are capable of making whole or partial conductivity reconstruction and also slice reconstruction.

3.2 Classification of the Reconstruction Algorithms

A reconstruction algorithm is a systematic way to find the resistivity by solving equations which define the inverse problem clearly. Types of the reconstruction

algorithms depend on the field elements of the equation system they try to solve. Generally two types of reconstruction algorithms exist; which those that current density and those that utilize magnetic flux density directly. Then, the reconstruction algorithms trying to solve Eqs. 3.5 and 3.8 are named as *Current Density based* and *Magnetic Flux Density based* algorithms, respectively. There may be also iterative and non-iterative forms of both algorithm types. Our proposed current density based algorithms are non-iterative which means running them once is enough. As a cost, they necessitate the measurement of all three components of magnetic flux density. The magnetic flux density based algorithm proposed for the solution of Eq. 3.8 requires only one component of magnetic flux density but needs some iteration steps to approach to the true conductivity distribution.

Another magnetic flux density based reconstruction algorithm described in this study is the *Sensitivity Matrix Method*. In this method we try to linearize the forward problem around a conductivity distribution point and find a linear matrix relation mapping the magnetic flux density deviations for small variations of conductivity around the selected linearization point. This algorithm can be implemented for single or all components of magnetic flux density and also it needs iteration steps too. We made computer simulations for a few iterations and for single component case only .

Chapter 4

Current Density based Reconstruction

This Chapter is devoted to description of some reconstruction algorithms utilizing inside current density distribution which, is calculated from measured magnetic flux density by a curl operation. Formulation of the relation between current density and logarithmic resistivity has already been made in previous Chapter and Eq. 3.5 was obtained. In this Chapter, three new reconstruction algorithms are proposed for solution of any row of the Eq. 3.9 which are first order partial differential equations.

In the first section, the application of a standard solution method to the third row of Eq. 3.9 is described. In the next section, the gradient of logarithmic resistivity is calculated on a Cartesian grid and reconstruction is made by integrating ∇R along Cartesian lines. The third and last section discretizes the third row with finite differences and obtains a matrix relation by also utilizing a few current injection profiles. The results of computer simulations and comments will be given together for all reconstruction algorithms in Chapter 6.

4.1 Reconstruction by Integration along Equipotential Lines

As stated in previous Chapter, each row of Eq. 3.9 can be used separately by reconstruction algorithms to solve the resistivity on a slice. The *Method of Characteristic Curves* is a standard technique for the solution of a single first order, linear or quasi-linear partial differential equations. A brief review of this technique will be given in the following Part for convenience.

4.1.1 Method of Characteristic Curves

For a vector field $\mathbf{A}(\mathbf{x})$ and scalar field $b(\mathbf{x})$, where $\mathbf{x} = [x \ y \ z]^T$, a first order linear or quasi-linear partial differential equation has the form,

$$\mathbf{A} \cdot \nabla u = b. \quad (4.1)$$

A curve in the solution domain is called a *characteristic curve* for the PDE, if at each point (x_0, y_0, z_0) on the curve, the vector $\mathbf{A}(x_0, y_0, z_0)$ is tangent to the curve

$$c \frac{d\mathbf{x}}{ds} = \mathbf{A}(\mathbf{x}(s)) \quad (4.2)$$

where s is an independent variable parameterizing the characteristic curve and c is a constant. The s parameter can be scaled and nothing changes for the characteristic curve. To remove constant c replace s with $s' = s/c$

$$c \frac{d\mathbf{x}(s')}{ds} = \frac{d\mathbf{x}(s')}{ds'} = \mathbf{A}(\mathbf{x}(s')). \quad (4.3)$$

Therefore the characteristic curve passing through any point $\mathbf{x}(s_0)$, for which s is assigned to be s_0 , can be found by the integral

$$\mathbf{x}(s) = \mathbf{x}(s_0) + \int_{s_0}^s \mathbf{A}(\mathbf{x}(t)) dt. \quad (4.4)$$

The derivative of u along a characteristic curve *w.r.t.* s gives us the value of scalar field b along that characteristic curve,

$$\frac{d}{ds}u(\mathbf{x}(s)) = \nabla u \cdot \mathbf{x}'(s) = b(\mathbf{x}(s)) \quad (4.5)$$

and thus we can recover the solution u along that characteristic curve given its value at one point, say for $s = s_1$, on the curve from the integral

$$u(s) = u(s_1) + \int_{s_1}^s b(\mathbf{x}(t))dt. \quad (4.6)$$

Each row of Eq. 3.9 can be written in a form, similar to (4.1) as below,

$$\tilde{\mathbf{J}}_i \cdot \nabla R = -(\nabla \times \mathbf{J})_i \quad i = 1,2,3 \quad (4.7)$$

where,

$$\tilde{\mathbf{J}}_1 = \begin{bmatrix} 0 \\ J_z \\ -J_y \end{bmatrix} \quad \tilde{\mathbf{J}}_2 = \begin{bmatrix} -J_z \\ 0 \\ J_x \end{bmatrix} \quad \tilde{\mathbf{J}}_3 = \begin{bmatrix} J_y \\ -J_x \\ 0 \end{bmatrix} \quad (4.8)$$

Note that $\tilde{\mathbf{J}}_i \cdot \mathbf{J} = 0$ for $i = 1, 2, 3$ and rank of $\tilde{\mathbf{J}} = \begin{bmatrix} \tilde{\mathbf{J}}_1 & \tilde{\mathbf{J}}_2 & \tilde{\mathbf{J}}_3 \end{bmatrix}$ is two for nonzero \mathbf{J} . At any given point the span of the columns of $\tilde{\mathbf{J}}$ is just the plane normal to \mathbf{J} and therefore this is the tangent plane at that point to an equipotential surface. This means that, the characteristic surfaces are coincides with the equipotential surfaces of the system (4.7). Cauchy data for this problem is the specification of R along a non-characteristic curve, that is a curve nowhere tangent to the family of equipotential surfaces, and specification of R at any point on an equipotential determines R on the connected component of the equipotential surface containing that point. This can be achieved of course only for all equipotential surfaces intersecting the non-characteristic curve. Appendix B explains how R can be determined in an equipotential surface if it is specified at one point in it.

Let us consider for simplicity the case where current is injected via a pair of point electrodes – a source and sink of current at the boundary $\partial\Omega$. For a simply

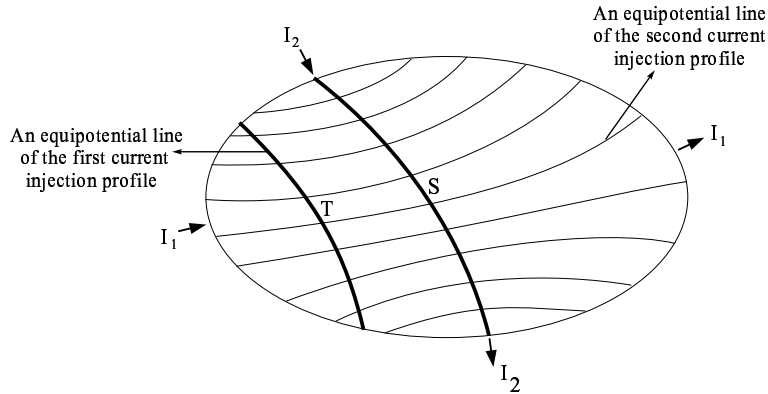


Figure 4.1: I_1 and I_2 denote two opposite current injection patterns which are applied one at a time. The lines S and T are two of the equipotential lines corresponding to the I_1 injection pattern, and the other lines in the domain are equipotential lines corresponding to the I_2 injection pattern. Positions of the electrodes for the I_2 injection pattern are chosen to be at the ends of S which intersects all equipotential lines of the I_2 injection pattern. Note that T does not intersect all equipotential lines of the I_2 injection pattern.

connected domain Ω it is clear that the equipotential surfaces are connected, and that a curve in Ω joining the source and sink intersects every equipotential. An example to such a curve can be a current streamline, in which case it is orthogonal to each equipotential surface and intersects each exactly once. Specifying R at all points of this current streamline, allows us to determine R at all equipotential surfaces and hence in all of Ω . In order to see if R can be determined in all of Ω by specifying it at only one point, now consider two such current injection pairs with all four point electrodes are different, for which the interior current density is known. A 2D illustration of this is given in Figure 4.1. Suppose we specify R at one point on an equipotential surface* of the first injection pattern, S , and hence R is known on all of that equipotential surface. If S is a non-characteristic surface† for the equipotential surfaces of the second injection pattern we have that R is determined on all equipotential surfaces for the second injection pattern intersecting S . However for this example, *i.e.* two pairs of point current injections,

*Equipotential curve for Figure 4.1.

†Non-characteristic curve for Figure 4.1.

we can choose the electrode positions so that the equipotential surfaces of the second injection pattern intersecting S cover the entire domain Ω . For example this happens when the second electrode pair lies on the intersection points of S with $\partial\Omega$. Figure 4.1 is illustrated for this selection of electrode positions. In this case, we see that R is determined completely by the interior current densities for two suitably chosen patterns up to one unknown constant. Consider now that R is specified on a point not on S . We can then move from this point along the corresponding equipotential curve of the second injection up to S , and then proceed to determine all of R in the whole domain.

The examples given above are simple cases which are used to illustrate some of the concepts. In order to derive the conditions for uniqueness in general, let us again consider two injection patterns for which \mathbf{J}^1 and \mathbf{J}^2 are measured. If at a given point $\mathbf{J}^1 \times \mathbf{J}^2 \neq 0$, then the two equipotential lines passing through that point corresponding to the two injection patterns are non-characteristic to each other, *i.e.* they are not “aligned” or “transverse” to each other. We call this condition the *transversality condition*. This allows us, *e.g.* for first injection pattern, moving to its nearby equipotential surface along the equipotential line of the second injection pattern. This situation is illustrated in Figure 4.2. Thus, if the transversality condition holds for at least one point on each equipotential surfaces of an injection pattern, then we can move in between any two equipotential surfaces of that current injection. This is then sufficient to reconstruct R in all of Ω given its value at a single point. The transversality condition may not hold at sufficient number of points by only two injection patterns, and therefore more than two injection patterns may be necessary so that for each equipotential surface there will be at least one injection pattern holding the transversality condition on an intersection point. If this condition is still not met on a sufficiently many number of points then R can only be determined in the biggest set of points which can be reached from the point at which it is specified

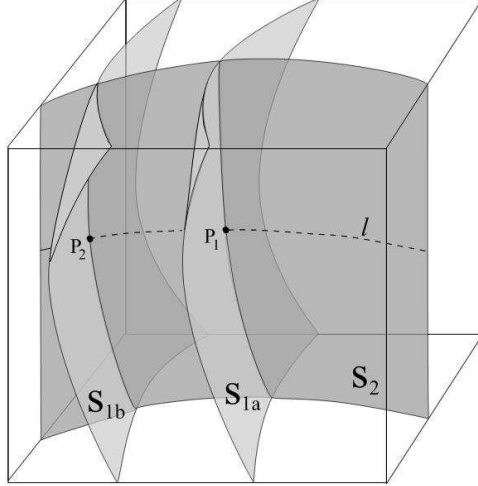


Figure 4.2: S_{1a} and S_{1b} are two nearby equipotential surfaces of first current injection pattern and S_2 is the equipotential surface of second current injection pattern. For S_{1a} and S_2 , transversality condition holds on point P_1 . If the value of R is known at P_1 , then the value of R can be found at point P_2 , and so at all points on S_{1b} , by moving along the equipotential line l or on any curve of S_2 which is connecting the two points.

by a chain of smooth curves each contained in the equipotential surface of one injection.

Let us now concentrate on the third row of Eq. 3.9, which is

$$\frac{\partial R}{\partial x} J_y - \frac{\partial R}{\partial y} J_x = -\left(\frac{\partial J_y}{\partial x} - \frac{\partial J_x}{\partial y}\right), \quad (4.9)$$

or an alternative form of its can be written as $\tilde{\mathbf{J}}_3 \cdot \nabla R = -(\nabla \times \mathbf{J})_3$ by putting $i = 3$ in (4.7). Note that this equation has characteristic curves defined by $\mathbf{x}'(s) = \tilde{\mathbf{J}}_3(\mathbf{x}(s))$ which stay in the same constant z plane as their starting points because third entry of $\tilde{\mathbf{J}}_3$ vanishes. In fact, the characteristic curve found for a starting point is the intersection of a constant z plane with the characteristic surface including the same point.

Consider a $z = c$ plane where c is a constant. Intersection of this plane with Ω is denoted by Ω_{xy}^c . In Ω_{xy}^c , $\left[\frac{\partial R}{\partial x} \quad \frac{\partial R}{\partial y} \right]^T$ is the projection of the gradient of R on

Ω_{xy}^c , and the left-hand side of Eq. 4.9 can be interpreted as the projection of this two dimensional gradient on the $\begin{bmatrix} J_y & -J_x \end{bmatrix}^T$ direction which is perpendicular to the current direction. Thus the characteristic curves are perpendicular to current streamlines and are in fact the equipotential lines. Therefore R can be obtained in Ω_{xy}^c by integrating along the characteristic curves in Ω_{xy}^c provided R is known for at least one point in each characteristic curve.

Assume now that two different injected current patterns are used and two internal current density distributions \mathbf{J}^1 and \mathbf{J}^2 are measured. Let \mathbf{J}_{xy}^1 and \mathbf{J}_{xy}^2 be the projections of \mathbf{J}^1 and \mathbf{J}^2 in Ω_{xy}^c onto Ω_{xy}^c . If $\mathbf{J}_{xy}^1 \times \mathbf{J}_{xy}^2 \neq 0$ for at least one point on each equipotential line of one injection pattern, then, it is enough to specify R on only a single point in Ω_{xy}^c . If this transversality condition holds for all points in Ω_{xy}^c the same conclusion can be drawn, but this is an over specification.

Similarly one can obtain slice images for Ω_{yz}^c and Ω_{xz}^c using the 1st and 2nd rows of (3.9) respectively.

4.2 Reconstruction by Integration Along Cartesian Grid Lines

For practical reasons integration along a Cartesian grid may be preferred to integration along equipotential lines.

If the gradient of logarithmic resistivity is known in Ω , then the logarithm of resistivity can be found, by integrating its gradient along cartesian grid lines, except for an additive constant which is equivalent to specifying the resistivity at a single point in Ω .

The determinant of the coefficient matrix in Eq. 3.9 is zero. Therefore the

gradient of R cannot be found if a single injected current profile is employed. Let us then assume that there are two experimentally measured current densities \mathbf{J}^1 and \mathbf{J}^2 which correspond to two different injection patterns. The third row of Eq. 3.9 can then be written twice to obtain,

$$\begin{bmatrix} J_y^1 & -J_x^1 \\ J_y^2 & -J_x^2 \end{bmatrix} \begin{bmatrix} \frac{\partial R}{\partial x} \\ \frac{\partial R}{\partial y} \end{bmatrix} = \begin{bmatrix} \frac{\partial J_x^1}{\partial y} - \frac{\partial J_y^1}{\partial x} \\ \frac{\partial J_x^2}{\partial y} - \frac{\partial J_y^2}{\partial x} \end{bmatrix}. \quad (4.10)$$

From this set of equations one can calculate $\left[\frac{\partial R}{\partial x} \quad \frac{\partial R}{\partial y} \right]^T$ for any point (x, y, z) provided that for that point the determinant, $-J_y^1 J_x^2 + J_y^2 J_x^1$, is not zero, or equivalently

$$\mathbf{J}_{xy}^1 \times \mathbf{J}_{xy}^2 \neq 0, \quad (4.11)$$

where, \mathbf{J}_{xy}^1 and \mathbf{J}_{xy}^2 are the projections of \mathbf{J}^1 and \mathbf{J}^2 onto the xy plane.

Once $\left[\frac{\partial R}{\partial x} \quad \frac{\partial R}{\partial y} \right]^T$ is found, then, using the first or the second row of Eq. 3.9, $\frac{\partial R}{\partial z}$ can be found easily if at least one of the conditions, $(J_y^1 \neq 0 \text{ or } J_y^2 \neq 0)$, or $(J_x^1 \neq 0 \text{ or } J_x^2 \neq 0)$ is satisfied respectively. One of these conditions will hold anyway, because condition in Eq. 4.11 is already required.

Handling the rows of Eq. 3.9 in different orders, one can show that to find the gradient of R at any point, it is also sufficient to have $(\mathbf{J}_{zx}^1 \times \mathbf{J}_{zx}^2) \neq 0$ or $(\mathbf{J}_{yz}^1 \times \mathbf{J}_{yz}^2) \neq 0$, at that point. In general if,

$$\mathbf{J}^1 \times \mathbf{J}^2 = \mathbf{J}_{yz}^1 \times \mathbf{J}_{yz}^2 + \mathbf{J}_{zx}^1 \times \mathbf{J}_{zx}^2 + \mathbf{J}_{xy}^1 \times \mathbf{J}_{xy}^2 \neq 0, \quad (4.12)$$

at a certain point, then the gradient at that point can be calculated because at least one of the terms in (4.12) will not vanish. In practice it may be necessary to employ more than two injection patterns because the condition in (4.12) may not be satisfied at all points by a single pair of injection patterns.

Note that by finding $\left[\frac{\partial R}{\partial x} \quad \frac{\partial R}{\partial y} \right]^T$ for only one xy plane, logarithmic resistivity can be found at only that plane *i.e.* slice, apart from an additive constant, without

having to be concerned about finding the gradient at other xy slices. Similarly for xz and yz slices.

4.3 Reconstruction by Solution of Linear Equation System

Any row of Eq. 3.9, *e.g.* the third[‡] one, can be discretized using *finite differences* on a rectangular mesh. This is made for each node of the mesh and also for every different current profiles. Then all obtained equations are combined into a matrix equation form where R can be found with a matrix inversion in least-square sense.

4.3.1 Finite Difference Formulation

The finite difference is the discrete analog of the derivative. If the P values of function $f(x)$ are tabulated at spacings h , then the notation

$$f(x) \equiv f_p \triangleq f(x_0 + ph) \quad \text{for } p = 1, 2, \dots, P \quad (4.13)$$

is used. The *finite forward difference* of function f_p is defined as,

$$\Delta f_p = f_{p+1} - f_p \quad \text{for } p = 1, 2, \dots, P - 1, \quad (4.14)$$

and the *finite backward difference* is defined as,

$$\nabla f_p = f_p - f_{p-1} \quad \text{for } p = 2, 3, \dots, P. \quad (4.15)$$

Note that, the forward difference of last, and the backward difference of first items of the f_n are not defined. A first order approximation to $f'(x)$ is,

$$f'(x) = \frac{\Delta f_p}{h} + O(h), \quad (4.16)$$

[‡]By selecting the third row, R can be reconstructed only on an xy slice.

with forward finite differences and,

$$f'(x) = \frac{\nabla f_p}{h} + \mathcal{O}(h), \quad (4.17)$$

with backward finite differences. In both case, the approximation error is in order of h .

A more accurate discrete approximation for $f'(x)$ can be achieved by the *central finite differences* of function f_p which is defined as,

$$f_p = \frac{\delta f_p}{2h} + \mathcal{O}(h^2) \quad (4.18)$$

where $\delta f_p = f_{p+1} - f_{p-1}$ for $p = 2, 3, \dots, P - 1$. Approximation error is in order of h^2 for central difference formulation. This means that halving the h decreases the error to a quarter. Central finite difference is not defined at the edges of the array. It can only be useful for points between first and last points.

Finite difference formulations given up to there are useful for approximating ordinary derivatives. For partial derivatives the same formulations can be used separately in each derivation direction. For example the third row of Eq. 3.9 can be approximated by central differences on a Cartesian grid of a slice consisting of $N \times M$ hexahedrons and for the i^{th} inner hexahedron as shown in Figure 4.3 like below,

$$\begin{aligned} & \frac{R_{(i+1,j)} - R_{(i-1,j)}}{2\Delta x} J_{y(i,j)} - \frac{R_{(i,j+1)} - R_{(i,j-1)}}{2\Delta y} J_{x(i,j)} \\ & = -\frac{J_{y(i+1,j)} - J_{y(i-1,j)}}{2\Delta x} - \frac{J_{x(i,j+1)} - J_{x(i,j-1)}}{2\Delta y} \end{aligned} \quad (4.19)$$

where Δx and Δy are the discretization steps in x and y directions, i and j are the indices of mesh element centers in x and y directions, and index k representing z dependence is omitted for ease of representation. For points lying near the boundary of Ω , backward and/or forward differences can be used where appropriate.

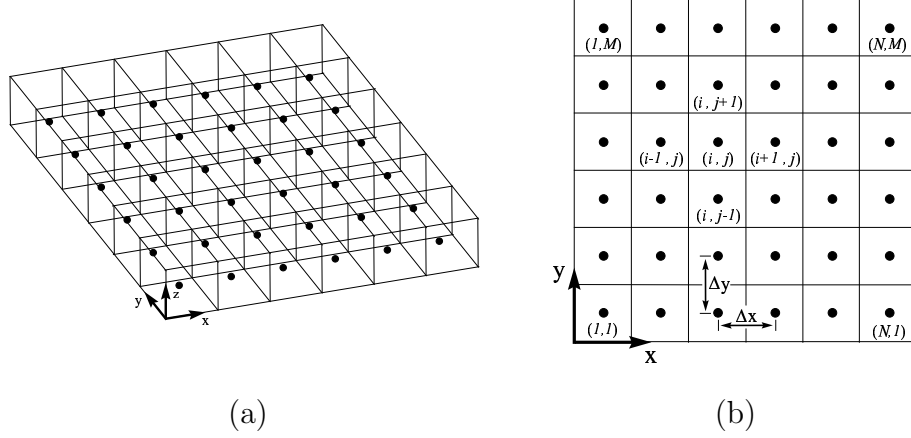


Figure 4.3: To reconstruct R on a slice, the third row of Eq. 3.9 is discretized by finite differences on the Cartesian grid consisting of center points of the slice hexahedrons. (a) 3D illustration of a slice with center points of its hexahedrons. (b) 2D illustration of a slice and indices of some hexahedrons used in Eq. 4.19.

Rearranging the finite difference equations one can obtain a linear set of equations

$$\mathbf{C}\mathbf{R} = \mathbf{b} \quad (4.20)$$

where $\mathbf{R} = [R_1, R_2 \dots R_{NM}]^T$ and NM is the number of unknown logarithmic resistivities. The sizes of \mathbf{C} matrix and \mathbf{b} vector are $NM \times NM$ and $NM \times 1$, respectively.

For K different injected current profiles, coefficient matrices and the right-hand side vectors can be concatenated to obtain the combined set of equations,

$$\begin{bmatrix} \mathbf{C}_1 \\ \mathbf{C}_2 \\ \vdots \\ \mathbf{C}_K \end{bmatrix} \mathbf{R} = \begin{bmatrix} \mathbf{b}_1 \\ \mathbf{b}_2 \\ \vdots \\ \mathbf{b}_K \end{bmatrix}. \quad (4.21)$$

It must be noted of course that for any point (x, y, z) there must be at least two injected current profiles such that the condition expressed in Eq. 4.12 is satisfied. Still, the rank of this equation will be $NM - 1$ because we know that, from its gradients, a function can be reconstructed only apart from an additional constant.

Therefore one can specify one of the R 's and solve the reduced set of equations, to find the remaining R 's.

Note again that this method can be used to obtain the logarithmic resistivities of only one slice without having to be concerned with other slices.

The algorithms given in this Chapter assume that the current density distribution is calculated by using all components of magnetic flux density distribution and formulates the solution of any one row of Eq.3.9. Actually if current density is an unknown too like resistivity, then each row equation becomes a nonlinear partial differential equation and it is not easy to solve them in one step. But when the current density is known, their coefficients and right-hand sides are known too so these nonlinear equations reduce to simple first order linear partial differential equations.

Measurement of magnetic flux density with all of its components is not practical so in practice we cannot calculate the current density, instead, we can measure only one component of magnetic flux density. In the next Chapter, we will concentrate on two algorithms which utilize only this measured data. Because this data not enough to reduce nonlinear equations into linear equations, they try solve resistivity distribution in an iterative manner by starting from an initially taken resistivity distribution and tries to approach an acceptable solution point instead of the exact true resistivity distribution.

Chapter 5

Magnetic Flux Density based Reconstruction

In this Chapter two iterative magnetic flux density based image reconstruction algorithms will be presented. Both of these algorithms can be used to reconstruct resistivity by using only one component of measured magnetic flux density. This removes the necessity of object rotations in MRI system in order to acquire other two components of magnetic flux density. This is important for keeping the practical situation of MR-EIT. Furthermore, proposed algorithms are suitable for slice reconstruction. This is also important to reduce the overall computation time significantly.

The first algorithm formulates the resistivity reconstruction as iterative solution of one component of quasi-linear PDE (3.8) which was derived in Chapter 3. For achieving this, Eq. 3.8 is expanded in its three components and for reconstructing the resistivity, *e.g.* on a constant z slice, its third component is discretized by finite differences on the Cartesian mesh consisting of hexagon centers of that slice and a matrix relation is obtained again by also employing

more than one current profiles. Because of the inner current density distribution is unknown, the unique solution of R from this matrix relation cannot be made in one step. Instead of calculating the true R vector, its an adequately near successor is calculated iteratively. The iterations start with solving the forward problem for obtaining current density distribution for an initially assumed resistivity distribution. Forward solver also uses exactly the same electrode shapes and boundary conditions of the object when magnetic flux density was acquired by MRI scanner. Using this calculated current density distribution, an R vector is found by solving the matrix relation. This found R vector is not true but surprisingly similar to the exact distribution of R as can be seen in the simulation results. To approximate further, the found R is substituted for the initial distribution, forward problem is solved again and a new R vector is obtained by the solution of matrix equation. As iterations proceed, solution converges to the true distribution if the starting point is not so far.

The second algorithm implements the linearization idea on the forward problem. Linearization is a classical approach in engineering which is used for the solution of nonlinear equations in a limited region. As derived in Chapter 2, the relation between conductivity and magnetic flux density is nonlinear. This nonlinear relation can be written around a conductivity distribution point using Taylor series expansion and by eliminating adequately many number of higher order terms a linear matrix equation can be obtained from conductivity perturbation to magnetic flux density perturbation domains. In this case, the inverse problem reduces only to the inversion of the transformation matrix and multiplication of it with the difference magnetic flux density vector. Nevertheless, reconstruction of true conductivity from this linear relation depends on how it is close to the selected the linearization point. If it is not close enough, then with a few iterations solution may approach to the true conductivity.

5.1 Reconstruction by Solution of Linear Equation System

Actually Eqs. (3.5) and (3.8) are the same except for their right-hand sides, so they have common properties and may be solved in a similar manner. In order to make a decision for which equation has to be used for resistivity reconstruction, one has to consider the number of measured magnetic flux density components. If all three components of magnetic flux density are known, then both of the equations can be solved by single run algorithms. In this case, solving Eq 3.5 is preferred because it includes only first order derivatives and this prevents more growth of the present measurement noise. If only one component of magnetic flux density is known, then current density is also an unknown like logarithmic resistivity so we cannot find the unique solution of both equations. However, Eq. 3.8 still includes some information about the conductivity with the Laplacian of measured magnetic flux density in its right-hand side and it may be possible to extract this information using the proposed iterative algorithm.

The algorithm uses only one component of measured magnetic flux density and tries to solve only the related row of the matrix form of Eq. 3.8 which was given below for convenience,

$$\begin{bmatrix} 0 & J_z & -J_y \\ -J_z & 0 & J_x \\ J_y & -J_x & 0 \end{bmatrix} \begin{bmatrix} \frac{\partial R}{\partial x} \\ \frac{\partial R}{\partial y} \\ \frac{\partial R}{\partial z} \end{bmatrix} = \frac{1}{\mu_0} \begin{bmatrix} \nabla^2 B_x \\ \nabla^2 B_y \\ \nabla^2 B_z \end{bmatrix}. \quad (5.1)$$

If it is assumed that only the \mathbf{z} component of magnetic flux density is measured, then we have to concentrate on the third row of this matrix equation,

$$\frac{\partial R}{\partial x} J_y - \frac{\partial R}{\partial y} J_x = \nabla^2 B_z / \mu_0. \quad (5.2)$$

which is a first order quasi-linear* PDE which relates the partial derivatives

*Equation is linear in the derivatives $\partial R/\partial x$ and $\partial R/\partial y$, but not for R .

of R to the Laplacian of B_z/μ_0 . It formulates the inverse problem on slices perpendicular to the \mathbf{z} direction. If $\nabla^2 B_z$ is calculated from measured B_z on a constant z slice, then the solution of this equation gives us the R distribution for that slice apart from an additive constant. As described in Appendix C, for absolute imaging making a single voltage measurement on the boundary is enough.

In general Eq. 5.2 may not have a unique solution and therefore as will be demonstrated in the Results Chapter, more than one current injection profiles must be used in order to obtain a unique solution for R .

An iterative algorithm has been used to solve this equation on a slice of cubical object which was shown before in Figure 4.3. Each slice includes $N \times M$ hexahedrons, where N is the number of hexahedrons in the x-direction, and M is the number of hexahedrons in the y-direction.

To reconstruct the logarithmic resistivity distribution for a slice, Eq. 5.2 is discretized on a square Cartesian mesh. Nodes of this mesh coincide with the hexahedron centers of the slice. Discretization is achieved by replacing the derivatives with finite difference equivalents. Finite difference approximation was derived in the previous Chapter for forward, backward or central differences. On edges and corners, appropriate forward or backward differences; and in the interior regions, central differences are used. As an example, assuming also that discretization steps Δx , Δy and Δz are all equal to unity, the discretized equation for the $(i, j)^{th}$ hexahedron, sitting in the interior part of the cartesian mesh, *i.e.* $2 < i < N - 1$ and $2 < j < M - 1$, is

$$\frac{R_{(i+1,j)} - R_{(i-1,j)}}{2} J_{y(i,j)} - \frac{R_{(i,j+1)} - R_{(i,j-1)}}{2} J_{x(i,j)} = (\nabla^2 \frac{B_z}{\mu_0})_{(i,j)}. \quad (5.3)$$

where the discretized version of the Laplacian (∇^2) operator for a scalar function

U for the k^{th} slice (not on the boundary) is

$$\begin{aligned} (\nabla^2 U)_{(i,j)} = & U_{(i+2,j,k)} + U_{(i-2,j,k)} + U_{(i,j+2,k)} + U_{(i,j-2,k)} \\ & + U_{(i,j,k+2)} + U_{(i,j,k-2)} - 6U_{(i,j,k)}. \end{aligned} \quad (5.4)$$

All together there are NM equations. We rearranged and combined all equations into matrix form as,

$$\mathbf{C}\mathbf{R} = \mathbf{b} \quad (5.5)$$

where \mathbf{C} is the $(NM \times NM)$ coefficient matrix, \mathbf{R} is the $(NM \times 1)$ vector of all hexahedron R values and \mathbf{b} is the $(NM \times 1)$ vector of $\nabla^2 B_z / \mu_0$.

Equation (5.5) is obtained for a single current injection profile. If there are K current injection profiles, *i.e.* K internal current distributions, \mathbf{J}_ℓ , renaming their \mathbf{C} matrices and \mathbf{b} vectors as \mathbf{C}_ℓ and \mathbf{b}_ℓ , where $\ell = 1, 2, \dots, K$, we obtain, by concatenation, the combined system equation

$$\mathcal{C}\mathbf{R} = \mathcal{B} \quad (5.6)$$

where $\mathcal{C} = [\mathbf{C}_1^T \ \mathbf{C}_2^T \ \dots \ \mathbf{C}_K^T]^T$ and $\mathcal{B} = [\mathbf{b}_1^T \ \mathbf{b}_2^T \ \dots \ \mathbf{b}_K^T]^T$.

Steps of the iterative reconstruction algorithm are then,

Step 1: Measure B_z and calculate $\nabla^2 B_z / \mu_0$ for all current profiles. Obtain \mathbf{b}_ℓ vectors for $\ell = 1, 2, \dots, K$.

Step 2: Assume an initial \mathbf{R}_0 vector. For the first iteration, it is taken as to correspond to a uniform distribution.

Step 3: Calculate \mathbf{J}_ℓ using forward solver and obtain \mathbf{C}_ℓ for $\ell = 1, 2, \dots, K$.

Step 4: Concatenate \mathbf{C}_ℓ matrices and \mathbf{b}_ℓ vectors to obtain combined system equation $\mathcal{C}\mathbf{R}_i = \mathcal{B}$. The indice of \mathbf{R}_i vector indicates the iteration number.

Step 5: Solve combined system equation to find \mathbf{R}_i .

Step 6: Check for the stopping criterion and stop if it is met. Else, go to *Step 3* and use the \mathbf{R}_i found in *Step 5* as the initial vector.

Solution of the combined system equation in *Step 5* requires that \mathcal{C} is not singular. This issue is discussed in the Chapter 6, which is devoted to simulation results. For stopping criterion one may check if the ℓ_2 norm of the change of \mathbf{R}_i vector between two consecutive iterations is below a preselected threshold in order to terminate the iterative algorithm. In our simulation studies we have looked at the relative ℓ_2 error of the difference between the reconstructed image and the actual resistivity distribution in order to follow the convergence of the algorithm as iterations proceed. Relative ℓ_2 error is defined as,

$$R\epsilon_i \triangleq 100 \frac{\|(\mathbf{R}_i - \mathbf{R})\|_{\ell_2}}{\|\mathbf{R}\|_{\ell_2}} \quad (5.7)$$

where \mathbf{R} is the original resistivity vector.

5.2 Reconstruction by Sensitivity Matrix

As explained in the Chapter 2 which formulates the forward problem of MR-EIT, the measured magnetic flux density is a non-linear function of the conductivity distribution. If we can write this relation approximately in a linear equation system form then we can solve it by using matrix algebra.

The non-linear relation between conductivity and magnetic flux density can be linearized around a conductivity distribution. This can be achieved by eliminating the higher order terms of Taylor series expansion which is written for the relation between conductivity and magnetic flux density around an initially taken conductivity distribution,

$$\mathbf{B}_z(\boldsymbol{\sigma}) \approx \mathbf{B}_z(\boldsymbol{\sigma}_0) + \left. \frac{\partial \mathbf{B}_z}{\partial \boldsymbol{\sigma}} \right|_{\boldsymbol{\sigma}=\boldsymbol{\sigma}_0} \Delta \boldsymbol{\sigma}. \quad (5.8)$$

Here $\Delta\boldsymbol{\sigma}$ is the difference vector between $\boldsymbol{\sigma}$ and $\boldsymbol{\sigma}_0$, which are the true and initially taken conductivity distribution vectors, respectively. The measured z component of magnetic flux density is the $\mathbf{B}_z(\boldsymbol{\sigma})$ vector and the calculated z component of magnetic flux density for conductivity distribution $\boldsymbol{\sigma}_0$ is the $\mathbf{B}_z(\boldsymbol{\sigma}_0)$ vector. For n conductivity elements and m magnetic flux density measurement points, the derivative of $\mathbf{B}_z(\boldsymbol{\sigma}_0)$ with respect to $\boldsymbol{\sigma}$ is an $m \times n$ matrix. Denoting this matrix by \mathbf{S} and moving the term $\mathbf{B}_z(\boldsymbol{\sigma}_0)$ to left-hand side, the linearized equation can be written in matrix form,

$$\Delta\mathbf{B}_z = \mathbf{S}\Delta\boldsymbol{\sigma} \quad (5.9)$$

where,

$$\Delta\mathbf{B}_z = \left[\Delta B_{z_1} \quad \Delta B_{z_2} \quad \dots \quad \Delta B_{z_m} \right]^T \quad (5.10)$$

$$\Delta\boldsymbol{\sigma} = \left[\Delta\sigma_1 \quad \Delta\sigma_2 \quad \dots \quad \Delta\sigma_n \right]^T \quad (5.11)$$

$$\mathbf{S} = \begin{bmatrix} \frac{\partial B_1}{\partial \sigma_1} & \frac{\partial B_1}{\partial \sigma_2} & \dots & \frac{\partial B_1}{\partial \sigma_n} \\ \frac{\partial B_2}{\partial \sigma_1} & \frac{\partial B_2}{\partial \sigma_2} & \dots & \frac{\partial B_2}{\partial \sigma_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial B_m}{\partial \sigma_1} & \frac{\partial B_m}{\partial \sigma_2} & \dots & \frac{\partial B_m}{\partial \sigma_n} \end{bmatrix}. \quad (5.12)$$

For simplicity, in the entries of the \mathbf{S} matrix, z and $\boldsymbol{\sigma}_0$ indices of magnetic flux density values are dropped. Eq. 5.9 relates the change in conductivity to change in magnetic flux density. This is a linear transformation from conductivity perturbation domain to magnetic flux density perturbation domain. The transformation matrix, \mathbf{S} , is the *Sensitivity Matrix*.

As seen from Eq. 5.12, each column of the sensitivity matrix includes the derivatives of B_z with respect to a conductivity element, for each measurement point of magnetic flux density. In order to form the sensitivity matrix, all conductivity elements are changed a little one by one and the difference between corresponding \mathbf{B}_z and $\mathbf{B}_z(\boldsymbol{\sigma}_0)$ is divided by the amount of conductivity change. This gives us the numerically calculated columns of sensitivity matrix. In other

words, the numerical calculation of sensitivity matrix is made by approximating the derivatives of magnetic flux density around $\boldsymbol{\sigma}_0$ point. The numerical calculation of sensitivity matrix is easy but it needs more computation time.

Singular values of the sensitivity matrix give us an insight about the quality of the imaging system. Therefore, a singular value decomposition (SVD) analysis of sensitivity matrix is required. A real $m \times n$ \mathbf{S} matrix can be written as

$$\mathbf{S} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^T \quad (5.13)$$

where

$$\mathbf{U} = \begin{bmatrix} \mathbf{u}_1 & \mathbf{u}_2 & \dots & \mathbf{u}_m \end{bmatrix} \in R^{m \times m} \quad (5.14)$$

$$\mathbf{V} = \begin{bmatrix} \mathbf{v}_1 & \mathbf{v}_2 & \dots & \mathbf{v}_n \end{bmatrix} \in R^{n \times n} \quad (5.15)$$

$$\boldsymbol{\Sigma} = \text{diag}(\lambda_1, \lambda_2, \dots, \lambda_p) \in R^{m \times n}, p = \min\{m, n\}. \quad (5.16)$$

The singular values of \mathbf{S} are λ_i and $\lambda_1 \geq \lambda_2 \geq \dots \geq \lambda_p \geq 0$. The number of nonzero singular values, r , is also the rank of matrix \mathbf{S} . The \mathbf{u}_i and \mathbf{v}_i are the i^{th} left and right singular vectors respectively.

The condition number is basically a measure of stability or sensitivity of a matrix (or the linear system it represents) to numerical operations. It is defined as

$$\kappa = \frac{\max\{\lambda_i\}}{\min\{\lambda_j\}} \quad i, j = 1, 2, \dots, p. \quad (5.17)$$

Matrices with condition numbers near 1 are said to be *well-conditioned*. Matrices with condition numbers much greater than one are said to be *ill-conditioned*, *e.g.* for a singular \mathbf{S} matrix $\kappa \rightarrow \infty$. In other words, we may not be able to trust the results of computations on an ill-conditioned matrix.

The computed sensitivity matrix for one current profile is ill-conditioned, in general. By using more than one current profile and combining their calculated sensitivity matrices for $\boldsymbol{\sigma}_0$, we may obtain a well-conditioned sensitivity matrix

\mathcal{S} . This new sensitivity matrix will have a smaller condition number and bigger rank than \mathbf{S} . For K different current profiles, sensitivity matrices and magnetic flux density difference vectors can be combined as below,

$$\begin{bmatrix} \Delta\mathbf{B}_{z_1} \\ \Delta\mathbf{B}_{z_2} \\ \vdots \\ \Delta\mathbf{B}_{z_K} \end{bmatrix} = \begin{bmatrix} \mathbf{S}_1 \\ \mathbf{S}_2 \\ \vdots \\ \mathbf{S}_K \end{bmatrix} \Delta\boldsymbol{\sigma} \quad (5.18)$$

and Eq. 5.9 can be written again for the multiple current profile case,

$$\Delta\mathcal{B}_z = \mathcal{S}\Delta\boldsymbol{\sigma}. \quad (5.19)$$

For obtaining $\Delta\boldsymbol{\sigma}$ from Eq. 5.9 we must find the inverse of \mathbf{S} but it is generally not a square matrix. Therefore, instead of direct inversion, pseudo-inverse of combined sensitivity matrix can be found by SVD which was given in Eq. 5.13. Since \mathbf{U} and \mathbf{V} are square and orthonormal, their inverses exist and are equal to the transpose of themselves. The inverse of diagonal singular values matrix $\boldsymbol{\Sigma}$ can be found simply by replacing the diagonal entries with their multiplicative inverses. So the pseudo-inverse of \mathbf{S} can be written as,

$$\mathbf{S}^\dagger = \mathbf{V}\boldsymbol{\Sigma}^{-1}\mathbf{U}^T. \quad (5.20)$$

Pseudo-inverse of the combined sensitivity matrix \mathcal{S} can also be found in a similar manner. Conductivity difference vector is found from Eq. 5.9 as,

$$\Delta\boldsymbol{\sigma} = \mathbf{S}^\dagger\Delta\mathcal{B}_z. \quad (5.21)$$

Note that, if Neumann boundary condition is assumed, then scaling the conductivity with a non-zero positive constant[†] does not change the amount of injected current and inside current density distribution. Furthermore, the magnetic flux density will remain the same too. This means that, any positively scaled version of

[†]Conductivity was defined as non-zero and positive.

σ_0 will also satisfy Eq. 5.8. Therefore the reconstruction of absolute conductivity cannot be achieved unless the scaling factor has been determined. In Appendix C, a procedure is described for determination of the scalar factor by using only a single surface voltage measurement.

The steps of the sensitivity matrix image reconstruction algorithm can be summarized as below,

Step 1: Measure $\mathbf{B}_{z_i}(\sigma)$ for all current profiles.

Step 2: Assume an initial conductivity distribution vector σ_0 . For the first iteration, it could be taken to correspond to a uniform distribution.

Step 3: Calculate the sensitivity matrices \mathbf{S}_ℓ around the initial conductivity distribution, magnetic flux density vectors $\mathbf{B}_{z_\ell}(\sigma_0)$ and difference magnetic flux density vectors $\Delta\mathbf{B}_{z_\ell}$ using measured data for each current profile by the formula $\Delta\mathbf{B}_{z_\ell} = \mathbf{B}_{z_\ell}(\sigma) - \mathbf{B}_{z_\ell}(\sigma_0)$ for $\ell = 1, 2, \dots, K$.

Step 4: Combine all sensitivity matrices and difference magnetic flux density vectors as shown in Eq. 5.18 in order to find \mathcal{S} and $\Delta\mathcal{B}_z$.

Step 5: Find \mathcal{S}^\dagger , the pseudo-inverse of combined sensitivity matrix, by SVD as described above. Solve the combined system equation as $\Delta\sigma_i = \mathcal{S}^\dagger\Delta\mathcal{B}_z$.

Step 6: Check for the stopping criterion and stop if it is met. Else, set $\sigma_0 \leftarrow \sigma_i$ and go to *Step 3*.

Step 7: For absolute imaging, calculate the scaling factor k as described in the Appendix C. Find absolute conductivity by Eq. C.2.

If our initial conductivity, σ_0 , is sufficiently near the original conductivity σ , then with iterations $\Delta\sigma_i$ will approach to zero. This means that the conductivity σ_i in *Step 3* is the true conductivity or algorithm has converged to another

solution point. For stopping criterion, we checked if the ℓ_2 norm of $\Delta\sigma_i$ is below a preselected threshold value.

In this Chapter we proposed two reconstruction algorithms which are utilize only one measured component of magnetic flux density. Therefore these algorithms are practical and suitable for clinical application of MR-EIT. Their drawback is that they require iterations and at the end of iterations its not guaranteed whatever the true conductivity distribution is obtained.

In our simulations we saw that by starting from an initially constant conductivity distribution, the first algorithm successfully detects the conductivity regions and produces a similar image in its first iteration step but the pixels of this priori image are not within the true values when we forced for absolute imaging. Then, each iteration contributes to the image as correction of its pixels. Practically after five iterations it converges to a very close conductivity distribution with a relative norm error below than 10%.

The second algorithm which linearizes the forward problem, also converges to an acceptable solution point in a few iterations if its initial conductivity distribution is not very far from the true conductivity point. For sensitivity matrix method, after some iteration steps the relative norm error begins to increase because making more iterations send away the solution from the true conductivity.

Because of their iterative structure, these algorithms need considerably more computation time compared to the current density based reconstruction algorithms. As the time consuming parts, the first algorithm solves the forward problem and combined linear equation system and, the second algorithm solves the forward problem, calculates the sensitivity matrix and finds its pseudo-inverse in each iteration step for a three dimensional object.

5.3 Region of Interest Reconstruction for Iterative Algorithms

Both iterative algorithms explained in this Chapter are start to iteration with an initial conductivity distribution, make their specific process and obtain a solution to the conductivity which is similar but not equal to the exact one. To approximate further, they go on to the next iteration by assigning the found conductivity as initial. But the found conductivity may include only some portion of slices by depending on how many slices the magnetic flux density was measured. In other words, if we measured B_z on a region including only a few slices then, after first iteration we cannot find any solution for outside of the measurement region. In this case, the second iteration cannot solve the forward problem accurately because of the unknown conductivity regions. In a real application of MR-EIT it is not practical to measure B_z for all slices of the object. As will be demonstrated in Chapter 6, this may not even be necessary if the image of a certain *Region-of-Interest* (RoI) is sought for.

Now let's assume that we try to reconstruct only a single slice of an object. Then we select a RoI which cover our single slice as its middle slice and measure B_z for only its slices. After giving the decision of which algorithm will be used, start the algorithm as usual and find conductivity for RoI. Before starting to the second iteration, Assign the blurred versions of top and bottom slices of RoI for upper and lower slices of RoI, respectively. For example, take the top slice of RoI, convolve it by the impulse response of a blurring filter and assign it to the upper slice. Repeat this process by assigning the reblurred versions of top slice for consecutive upper slices. Make same process for lower slices by blurring bottom slice of RoI. In the result, for second iteration we can calculate current density and also magnetic flux density more accurately unless the upper or lower slices include very high or low conductivity regions.

In Chapter 6, we investigate for the size of RoI and what happens if some unusually high or low conductivity regions are exists in upper or lower of the RoI. As will be seen in simulations, slice reconstruction can be made successfully even for small RoI sizes.

Simulation results of all proposed algorithms are given in together in Chapter 6 to be able to made comparison between different algorithms. Simulations are made for different combinations of conductivity and electrode sizes as far as possible.

Chapter 6

Simulation Results

This Chapter is reserved for computer simulation results of the proposed algorithms. All required data are generated by computer and no measurements were made. The simulated object has a rectangular prism shape with edge lengths $32 \times 32 \times 64cm$. These dimensions are similar to human body dimensions and, are enough for simulation studies. For assigning a conductivity model, object is subdivided into hexahedrons with $1cm$ edge length and the conductivity of each hexahedron is assumed to be constant.

In order to understand the features of algorithms better, nine simulation models are constructed from three different conductivity models with three different electrode sizes and current injection profiles. Simulations are made for these nine models to investigate the effects of electrode sizes and positions, number of current injection profiles and measurement noise. Also for iterative algorithms the effect of number of iterations and size of RoI are investigated.

It is assumed that our slice of interest is the 32^{nd} (middle) slice in z direction and all simulations are made for reconstruction of this single slice.

| Conductivity Region | Conductivity Model 1 (C1) | Conductivity Model 2 (C2) | Conductivity Model 3 (C3) |
|------------------------|------------------------------|------------------------------|------------------------------|
| Sphere (slices 9-56) | 1 | × | × |
| Sphere (slices 20-45) | × | 1 | × |
| Sphere (slices 20-39) | × | 3.3 | × |
| Sphere (slices 5-15) | × | 2.2 | × |
| Sphere (slices 50-60) | × | 1.7 | × |
| Cylinder | × | 2.5 | × |
| Lungs | × | × | 1.81-0.61 |
| Heart | × | × | 40-22.2 |
| Backbone | × | × | 1.43-0.55 |
| Background | 2 | 2 | 2 |

Table 6.1: Regions conductivities of the conductivity models ($mS \times cm^{-1}$).

6.1 Conductivity Models

Three conductivity models with different complexities are used for simulations. Depending on their complexity, each model includes some regions with different conductivity from background. The background conductivity is taken to be $2mS \times cm^{-1}$ which is close to the average body conductivity. For each conductivity model, 30 slices are selected out of the 64 slices and these are given in Figures 6.1, 6.2 and 6.3, respectively.

The first model includes a simple spherical region in the center, more resistive than background. Spherical regions have special importance in 3D MR-EIT because their shape change in all three dimensions so the size of conductivity region varies on different slices. In the second model, there are different three dimensional conductivity regions such as sphere, cylinder and rectangular prism. The big sphere and rectangular prism are more resistive and other regions, which occur in different places, are more conductive than the background. The third conductivity model is a realistic one which represents the conductivity distribution of human thorax. In thorax model, lungs and backbone are more

| Electrode Model | Electrode size in x direction | Electrode size in y direction | Electrode size in z direction | Electrode placement |
|-----------------|---------------------------------|---------------------------------|---------------------------------|---------------------|
| E1 | × | 100% | 100% | × |
| E2 | 100% | × | 100% | × |
| E3 | × | 40% | 100% | middle |
| E4 | 40% | × | 100% | middle |
| E5 | × | 40% | 100% | diagonal |
| E6 | 40% | × | 100% | diagonal |
| E7 | × | 30% | 35% | middle |
| E8 | 30% | × | 35% | middle |
| E9 | × | 30% | 35% | diagonal |
| E10 | 30% | × | 35% | diagonal |

Table 6.2: Electrode models used in simulations.

resistive and heart is more conductive than background body conductivity. In Table 6.1, region conductivities are given for all models.

6.2 Electrode models

The optimum size and positions of the electrodes are important in MR-EIT. In general, for each different conductivity distribution they may change. Three different electrode sets in shape and size are used in nine simulation models and their different placements are given in Figure 6.4. To simulate the electrodes, the amount of injected current is divided to the number of tetrahedrons which have a triangular face at the electrode surface. Furthermore, the total current of each triangular surface is equally distributed among its nodes. In Tables 6.2 and 6.3 details of used electrode sets, and simulation models are given.

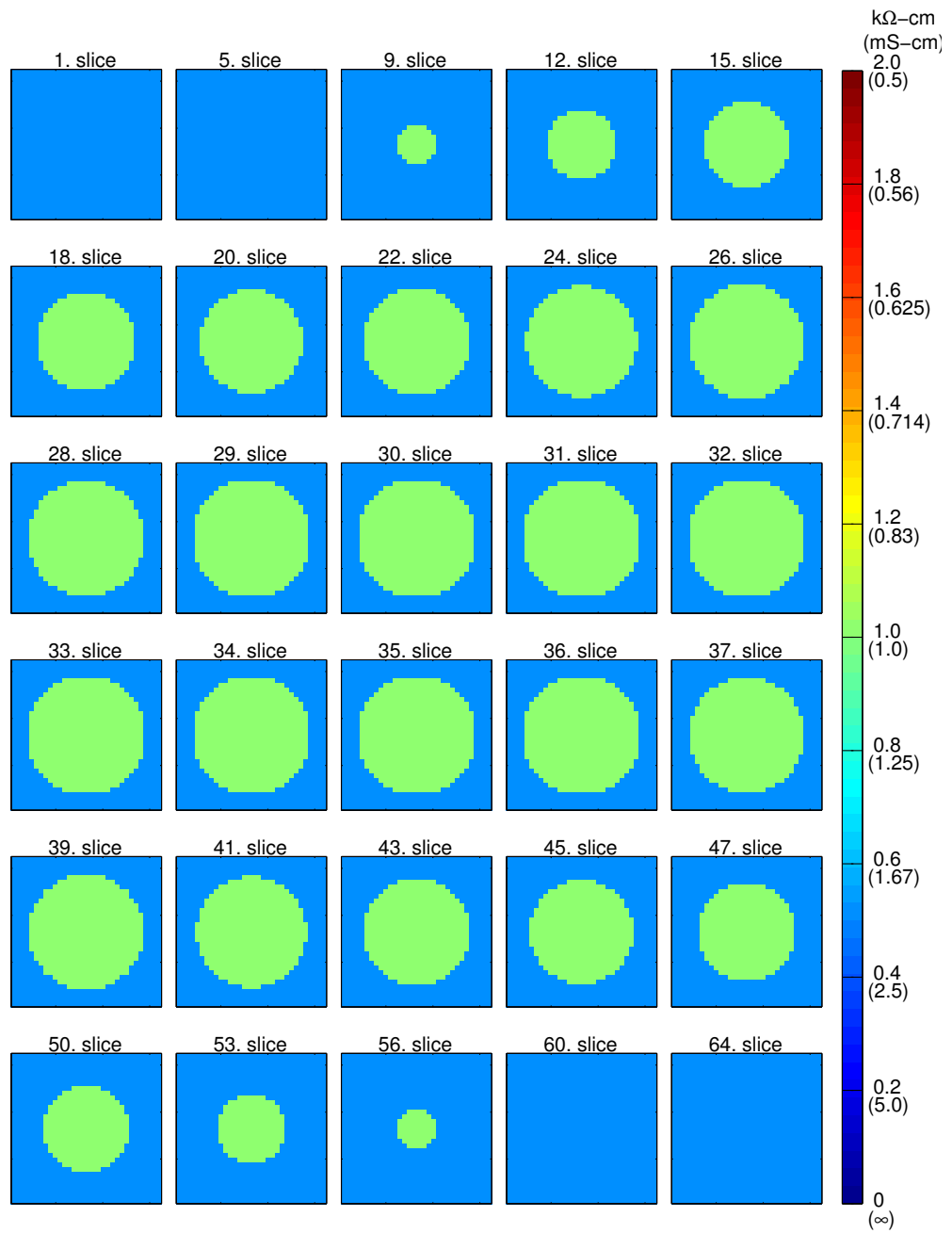


Figure 6.1: Some selected slices of the first conductivity model.

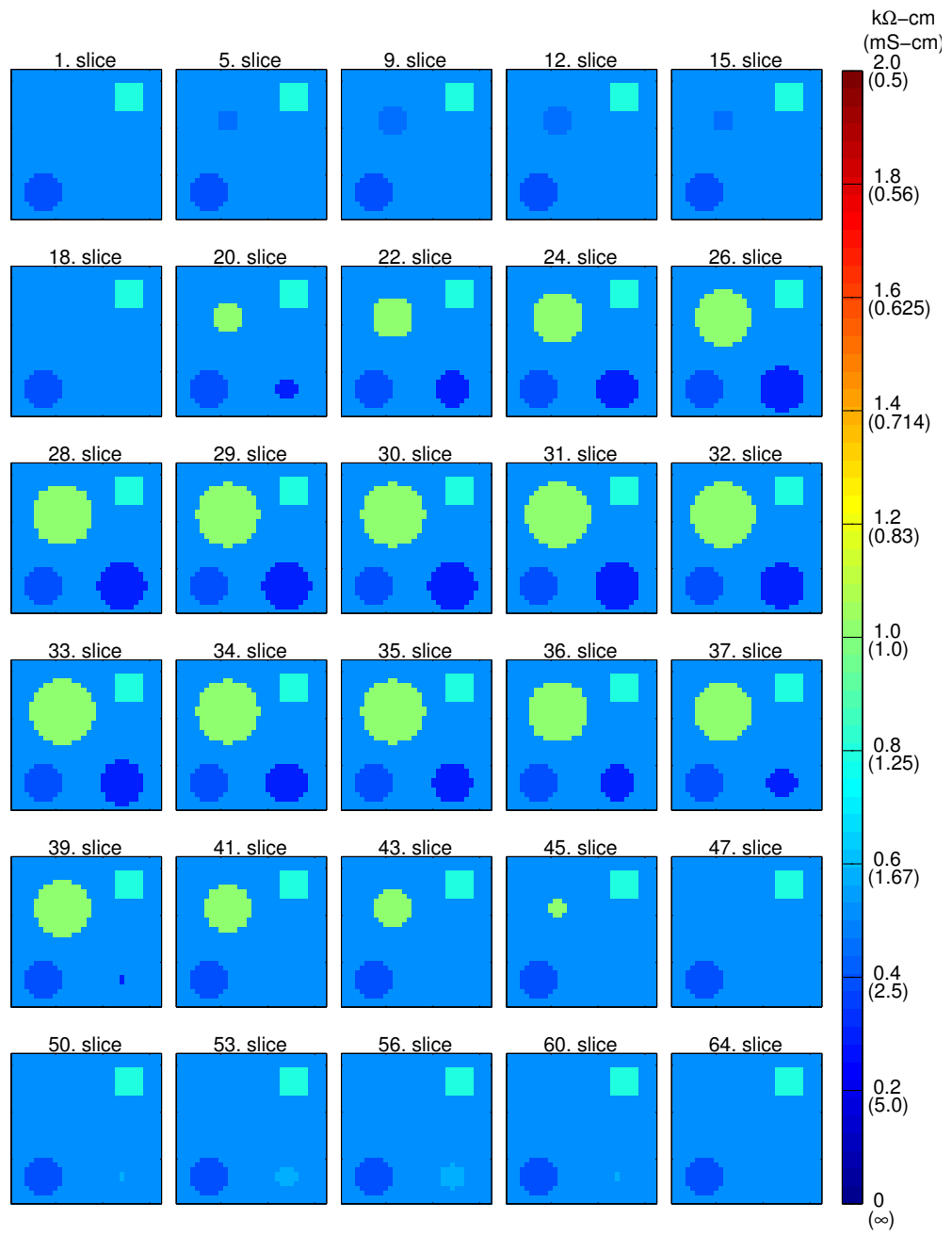


Figure 6.2: Some selected slices of the second conductivity model.

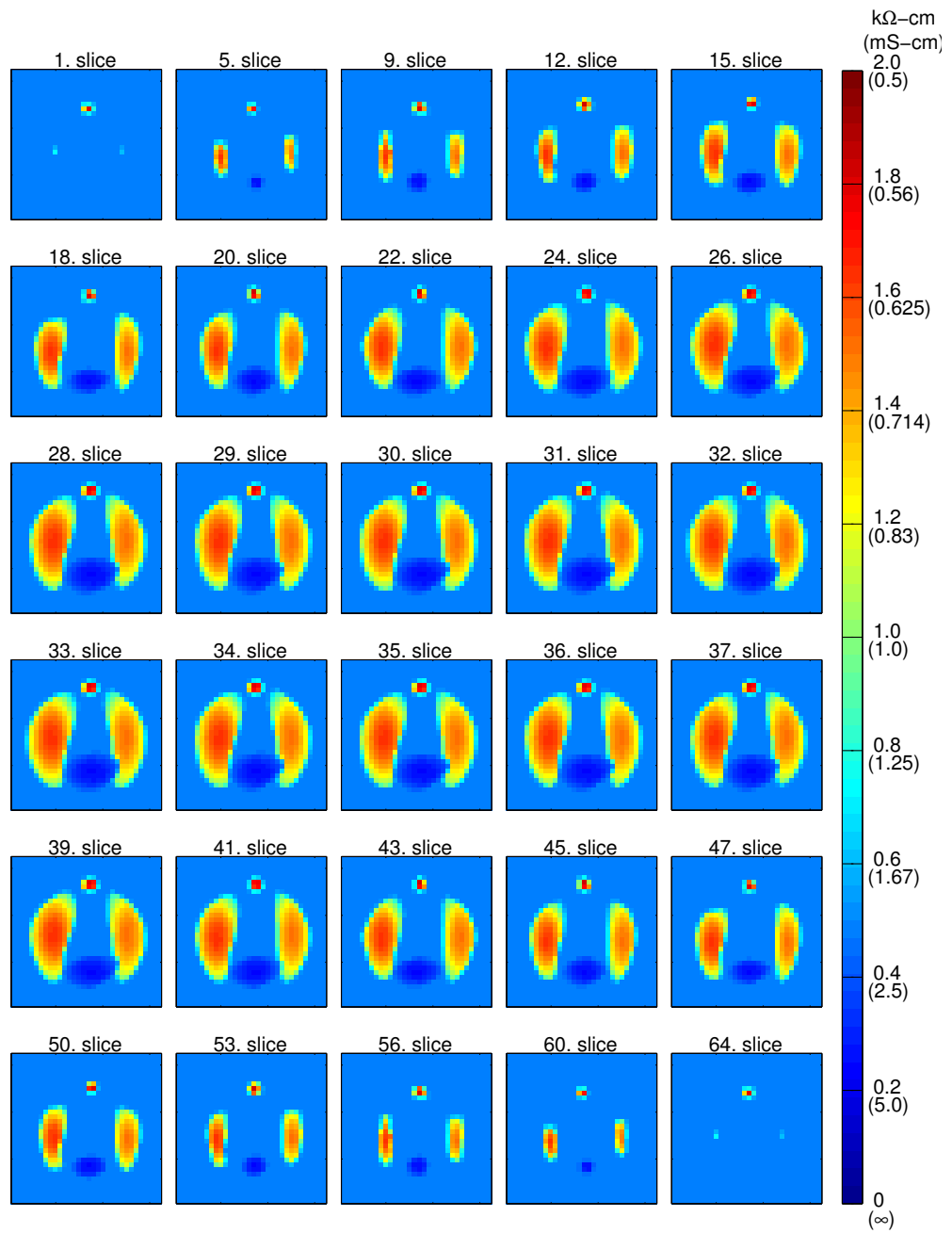
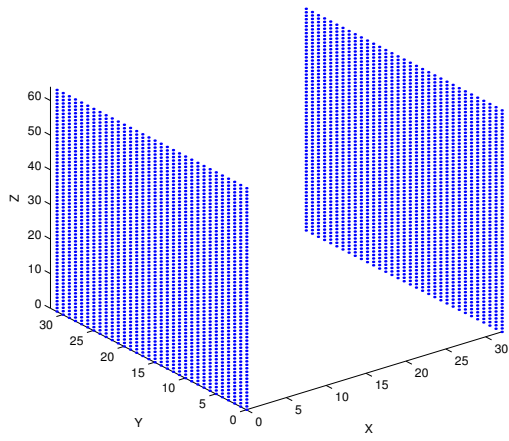
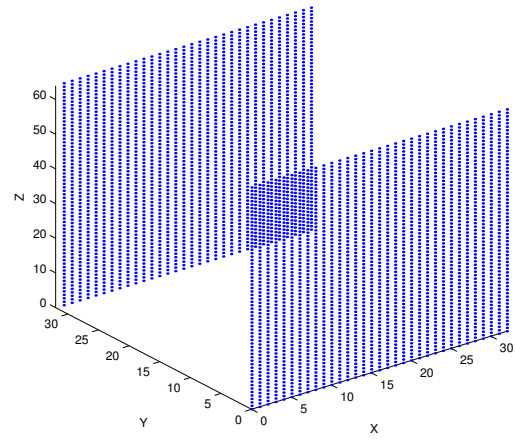


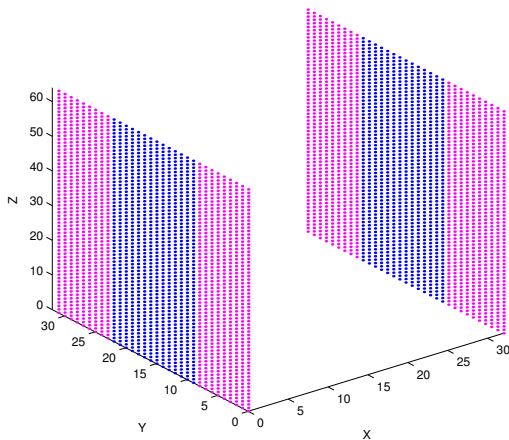
Figure 6.3: Some selected slices of the third conductivity model.



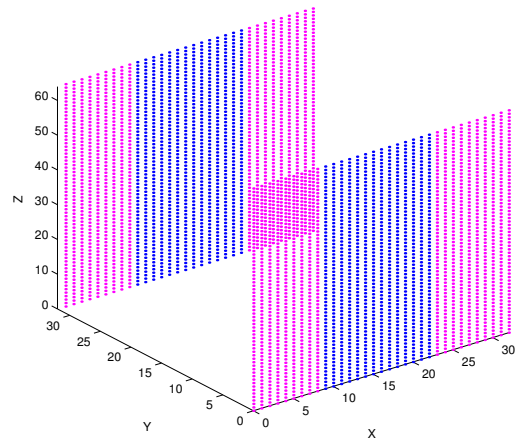
(a)



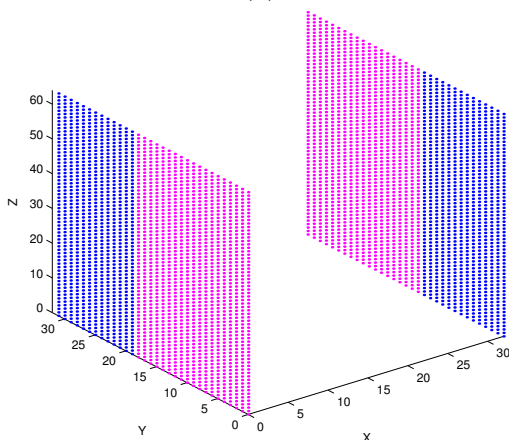
(b)



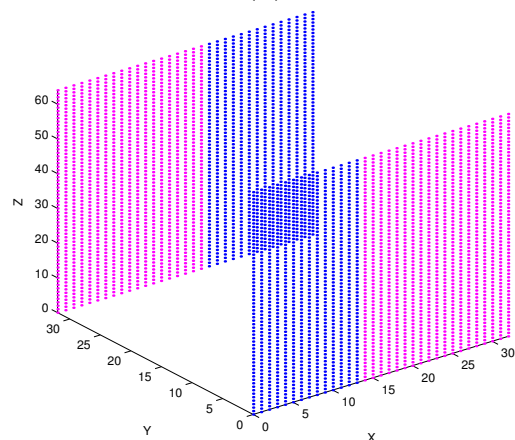
(c)



(d)



(e)



(f)

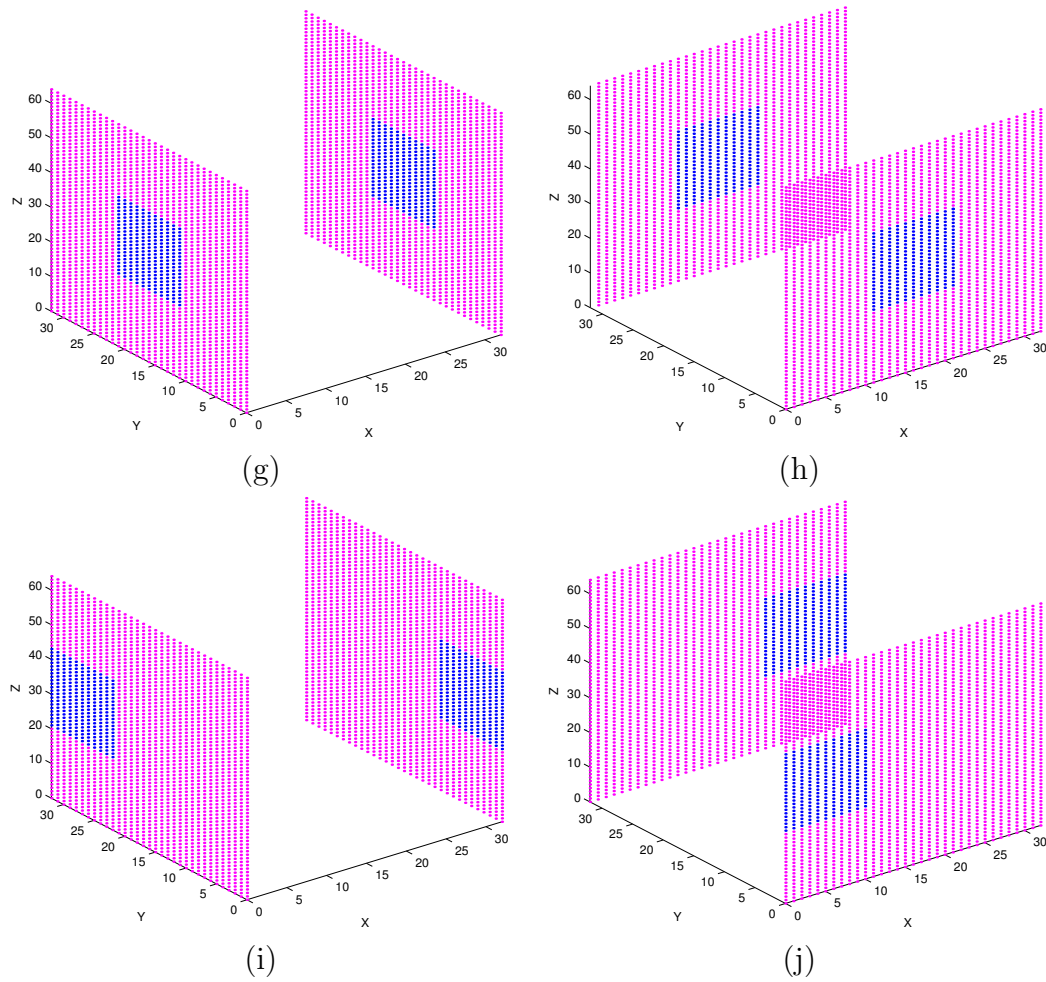


Figure 6.4: Electrode models used in simulations. Blue points indicate the current injected boundary nodes. In simulations, current injection is made with separate current sources for each node. Figures (a) to (j) correspond to electrode models E1 to E10 respectively which are defined in Table 6.2.

| Simulation Model No | Conductivity Model No | Electrode Models No | Number of Current Injection Profiles |
|---------------------|-----------------------|---------------------|--------------------------------------|
| S1 | C1 | E1,E2 | 2 |
| S2 | C1 | E3-E6 | 4 |
| S3 | C1 | E7-E10 | 4 |
| S4 | C2 | E1,E2 | 2 |
| S5 | C2 | E3-E6 | 4 |
| S6 | C2 | E7-E10 | 4 |
| S7 | C3 | E1,E2 | 2 |
| S8 | C3 | E3-E6 | 4 |
| S9 | C3 | E7-E10 | 4 |

Table 6.3: Simulation models.

6.3 Measurement Noise

In order to understand the behavior of the algorithms in the presence of measurement noise, simulated noise is added to both current density and magnetic flux density data. For current density imaging, the characteristics of measurement noise are analyzed by Scott *et.al.* [15] and its distribution is found as similar to Gaussian distribution with zero mean. The standard deviation of noise for one component of magnetic flux density is formulated as

$$\sigma_B = \frac{1}{2\gamma T_c \text{SNR}}, \quad (6.1)$$

where γ is gyro-magnetic ratio ($26.75 \times 10^7 \text{rad}/(\text{sec} \times \text{Tesla})$), T_c is the duration of applied current, and SNR is the signal-to-noise ratio of the MRI system. The standard deviation of noise for one component of current density is separately formulated as

$$\sigma_J = \frac{1}{2\gamma\mu_0 T_c \text{SNR}} \sqrt{\left(\frac{F_x}{\Delta x}\right)^2 + \left(\frac{F_y}{\Delta y}\right)^2}, \quad (6.2)$$

where $\Delta x, \Delta y$ are lengths of measurement grid in x, y directions and F_x, F_y are the noise weights depending on the used convolution template which is used for differentiation of phase images in CDI. Our measurement grid length is in 1cm

length in both directions and for the simplest template, noise weights are unity.

Noise generation is made for both algorithm type by first generating a Gaussian distributed noise and after multiplying it with the corresponding standard deviation. Current injection time, T_c , is taken as $100ms$. Simulations are made for three different noise levels by assigning 30, 60 and 90 to the SNR.

As seen from Eqs. 6.1 and 6.2, the range of noise is independent from the magnitudes of current density and magnetic flux density. This means that by increasing the amount of injection current we can increase the SNR* of the MR-EIT system. But this is not so simple for human objects because there is a safety limit for injected current and exceeding this amount may result in risk for the patient.

The safety limit for direct current is often accepted as $2mA$ [33]. By a simple calculation, for $2mA$ injection current, an average of $50nT$ induced magnetic flux density is generated in the body. If the SNR of our MRI system is 30 and current injection time is $100ms$, then we have a noise with $0.62nT$ standard deviation. For positive and negative numbers deviation is symmetric, so we can expect %0.62 deviation for the induced magnetic field.

6.4 Simulation Results for Current Density based Algorithms

Figure 6.5 shows the reconstructed images of the 32^{nd} slice for nine simulation models by integration along equipotential lines which are also shown just bottom of each image. As an advantage, this algorithm needs only one current injection profile and simulations are made for the first current injection profile of each

*This is the general system SNR and must not be confused with the MRI system SNR.

simulation model. The behavior of the algorithm is observed for three different measurement noise levels in addition to the noiseless case. The amount injected current is $100mA$ for noisy cases.

The algorithm works fine for all simulation models in the noiseless case. It finds the equipotential lines correctly and reconstructs the resistivity by integrating along them. But when noise is increased more, its performance drops dramatically because the noise also disturbs the equipotential lines and tracing them becomes harder.

For the same simulation models the method of reconstruction by integrating along Cartesian grid lines is also applied by first two current injection profiles. Again it is assumed that the conductivity of first elements is known for each line. Starting from these elements, the pixel values of the 32^{nd} slice are reconstructed along the x direction by integrating $\partial R/\partial x$. For integration, trapezoidal method is used for the 32×32 discretization where the integration points are the center points of the hexahedrons. The results are given in Figure 6.6 for nine simulation models. The edges of interior object's are reconstructed in a more blurred manner. Also errors are made along the integration if the derivative of a pixel is not found correctly. When reconstruction is repeated for noisy data with SNR of 30, 60 and 90 as explained previously, it is seen that the method does not blow up at all but if the value of a derivative has high error at a point then, this error propagates along the integration. Also it is seen from the figure that increasing injected current reduces the effect of noise on reconstructions.

Derivative may be found incorrectly in some pixels because, the transversality condition does not hold at those pixels. In other words, the currents of two current injection profiles become more aligned and this leads to a near singular current

density matrix

$$\begin{bmatrix} J_y^1 & -J_x^1 \\ J_y^2 & -J_x^2 \end{bmatrix} \quad (6.3)$$

which was in Eq. 6.3 and given here for convenience. For models S1, S4 and S7 which use all-surface current injection electrodes, the transversality condition holds better than the other models, because their currents flow directly from one surface to the opposite surface of the object and cross more perpendicularly in the pixels. This is shown in Figure 6.7 for models S1-S3 and S7-S9 by the quiver plot of their current densities for first two current injection profiles.

In Figure 6.8 the quiver plot of current densities of models S1, S4 and S7 are given for noiseless and noisy cases. This shows clearly why noise distorts the reconstructed image only on some integration lines. On some pixels, because of the noise, currents of two injection profiles become more aligned and transversality condition is not held at those points. Effect of this goes on along those integration lines because the integration cumulates the incorrectly found derivatives.

The last one of current density based algorithms achieves reconstruction by solution of a linear equation system which is obtained by finite difference formulation of the third row of Eq. 3.9. We made simulations for models *S1* to *S9* again and reconstructed the 32nd slice for noiseless and for noisy cases by assuming that one pixels conductivity is known. This is required for reconstruction of absolute imaging. The absolute imaging can be achieved by the addition of a single voltage measurement on the boundary as described in Appendix C.

This algorithm exhibits the best performance against noise. We see again that increasing the amount of injected current decreases the effects of noise. As opposite to the other two algorithms, it keeps the mutual relations of all pixels in the system matrix and reconstructs all conductivity by solution of linear equation system in the least-squares sense. This provides some kind of filtering of the noise

and makes the algorithm the best one against noise.

In order to see the noise filtering effect of least-square sense solution, let us make a simple experiment in MATLAB and try to reconstruct a four pixel image. Our algorithm solves a linear equation system in the form of $\mathbf{CR} = \mathbf{b}$. First, generate a full rank coefficient matrix \mathbf{C} with random entries,

$$\mathbf{C} = \begin{bmatrix} 44 & 32 & 74 & 68 \\ 50 & 96 & 27 & 21 \\ 21 & 73 & 44 & 84 \\ 64 & 41 & 93 & 63 \end{bmatrix}. \quad (6.4)$$

Next, assign four numbers as the conductivities of pixels and which are near in value, like that,

$$\mathbf{R} = \begin{bmatrix} 1 & 2 & 3 & 2 \end{bmatrix}^T. \quad (6.5)$$

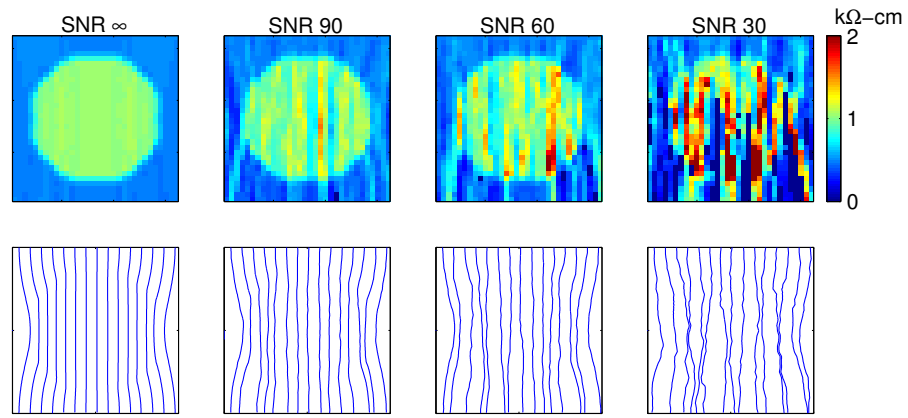
For the given case, left-hand side of linear equation system becomes

$$\mathbf{CR} = \begin{bmatrix} 466 & 365 & 467 & 551 \end{bmatrix}^T. \quad (6.6)$$

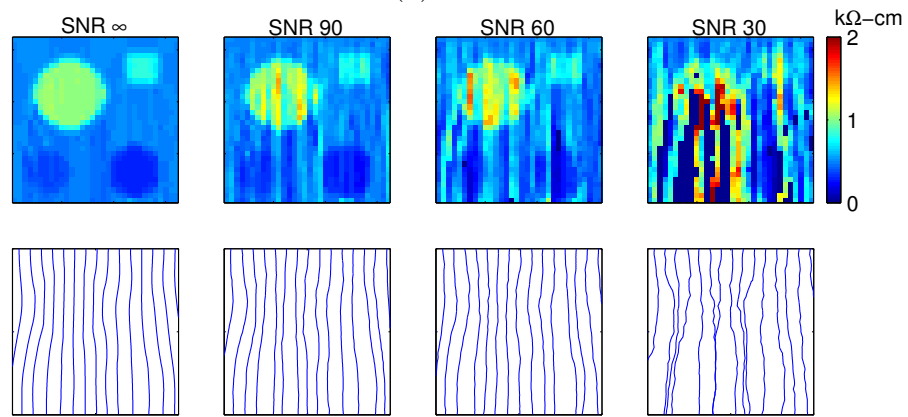
Then, add some noise to \mathbf{C} and \mathbf{b} about %5 of their entries, and solve for \mathbf{R} . Result changes by depending on the added noise but it is very near to the exact one. A sample result is $\begin{bmatrix} 1.0492 & 1.9446 & 2.9495 & 2.0015 \end{bmatrix}^T$ which has a relative norm error of %2.11.

6.5 Simulation Results for Magnetic Flux Density based Algorithms

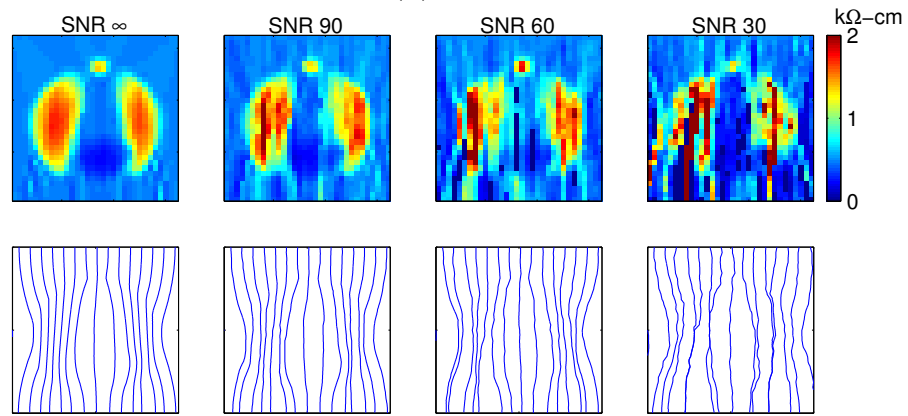
The results for algorithms which utilize current density data were given up to this point. Getting the current density data from MR images is not practical, therefore the use of these algorithms is limited. From now on, we will examine



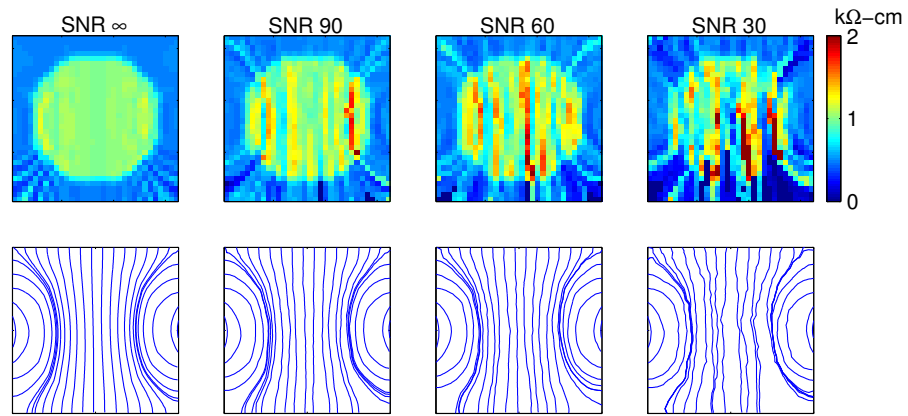
(a)



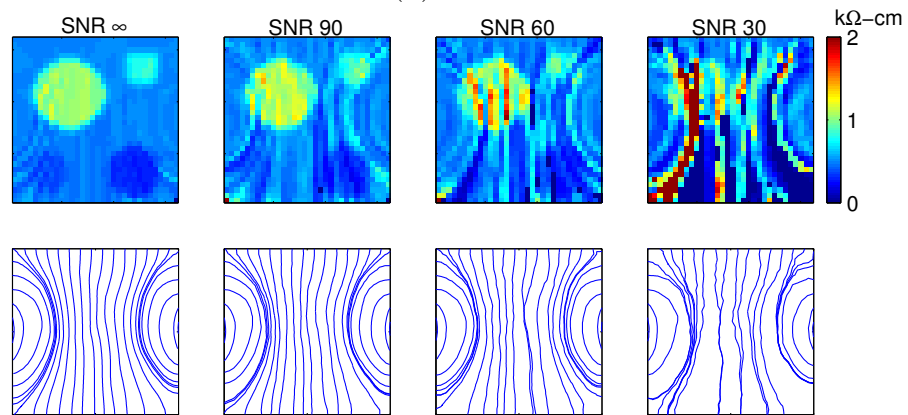
(b)



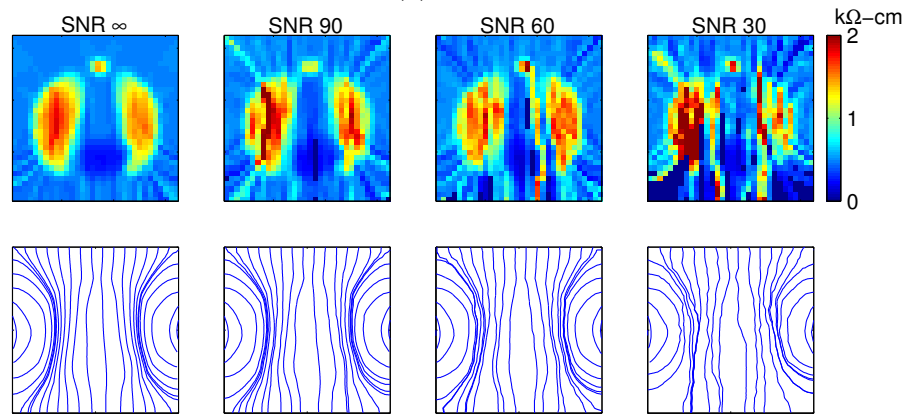
(c)



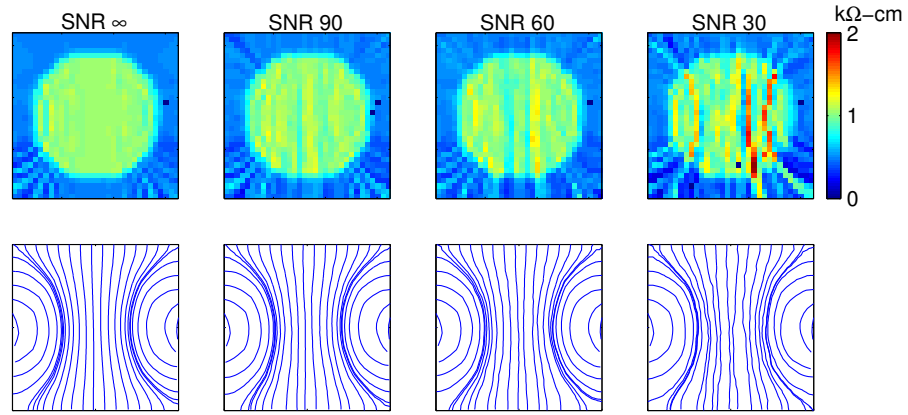
(d)



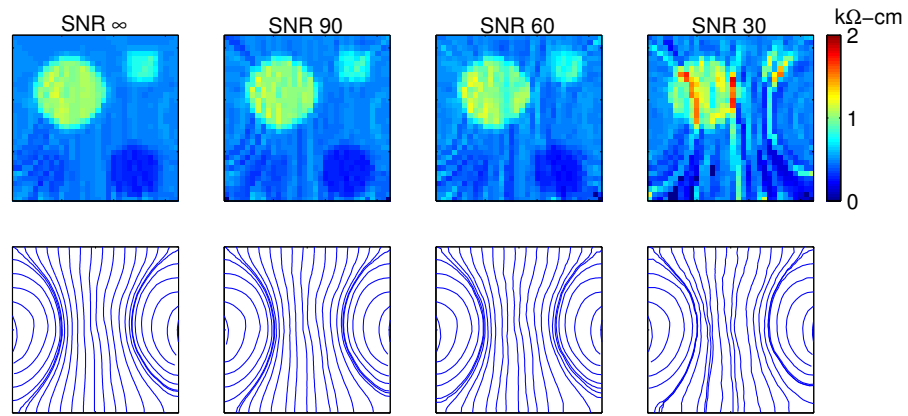
(e)



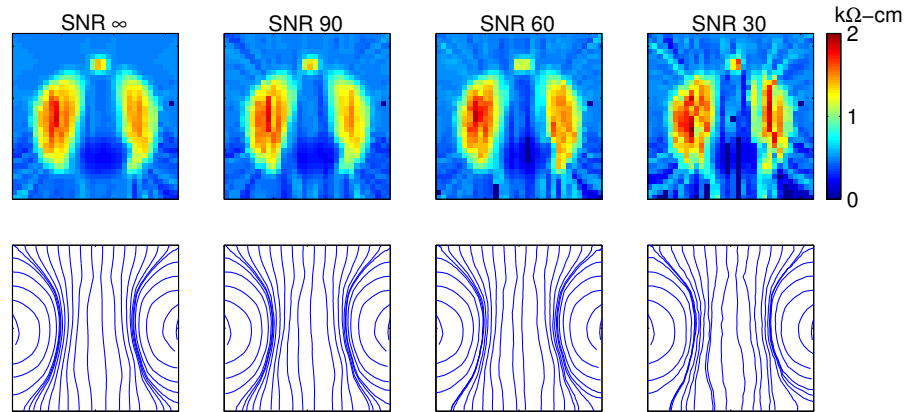
(f)



(g)

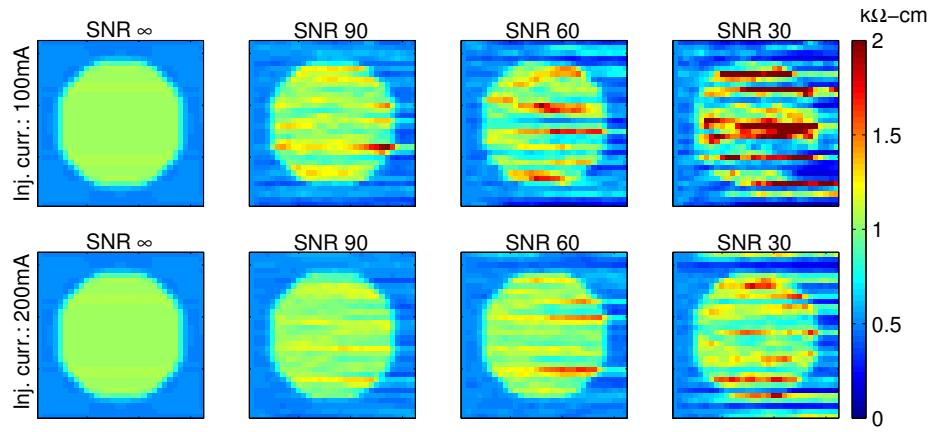


(h)

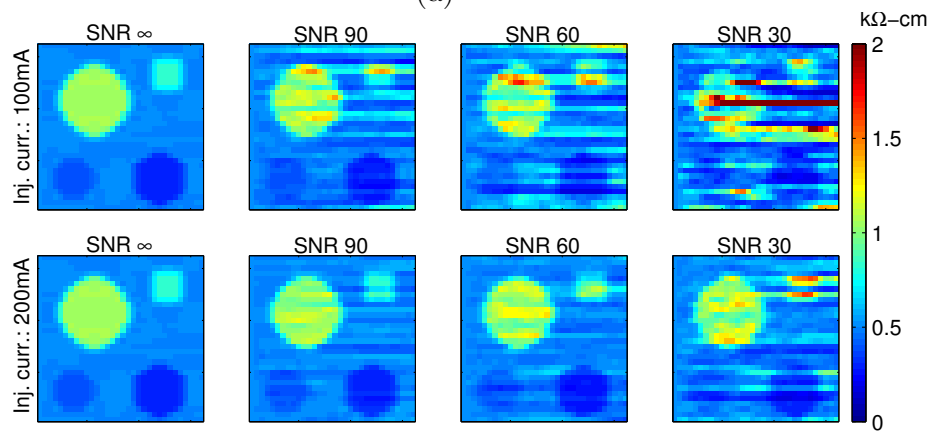


(i)

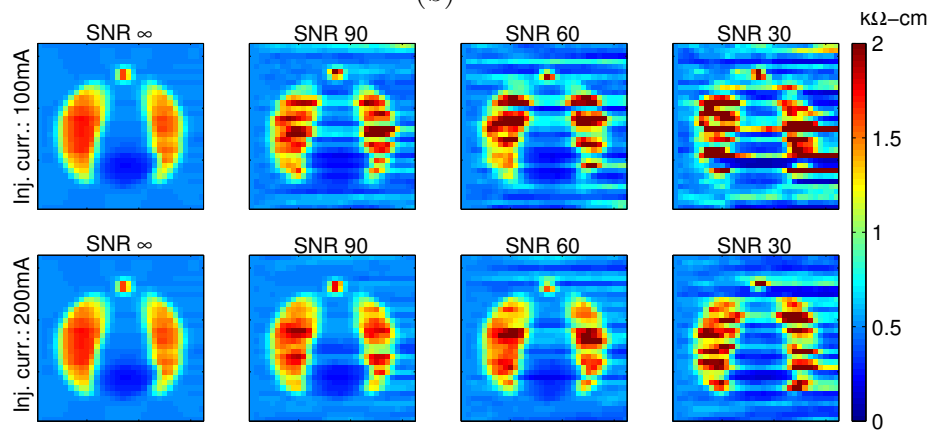
Figure 6.5: Results for the reconstruction along equipotential lines for 32^{nd} slices of simulation models S1 to S9 are given in (a) to (i) respectively. Some selected equipotential lines used in reconstruction were given just below of the each image. For noisy cases $100mA$ current was injected. The amount of current is not important for noiseless case.



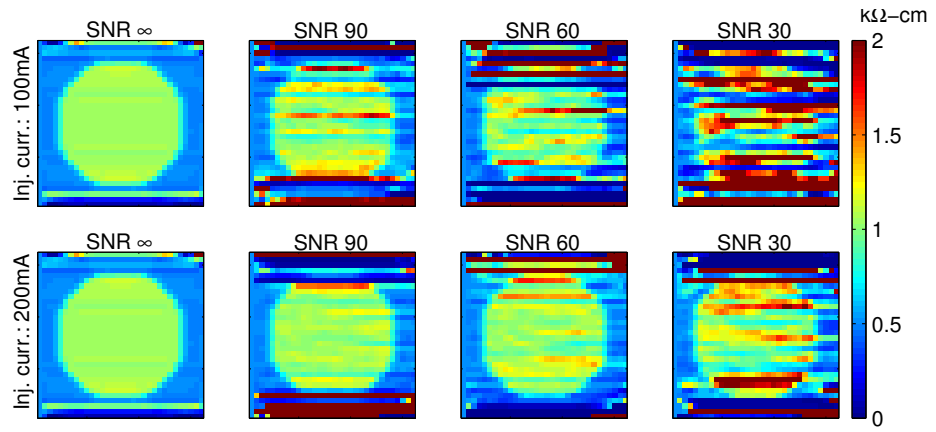
(a)



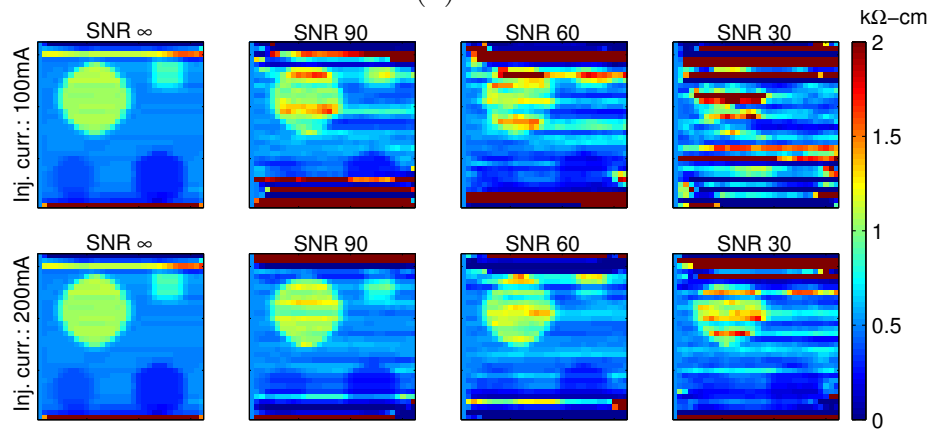
(b)



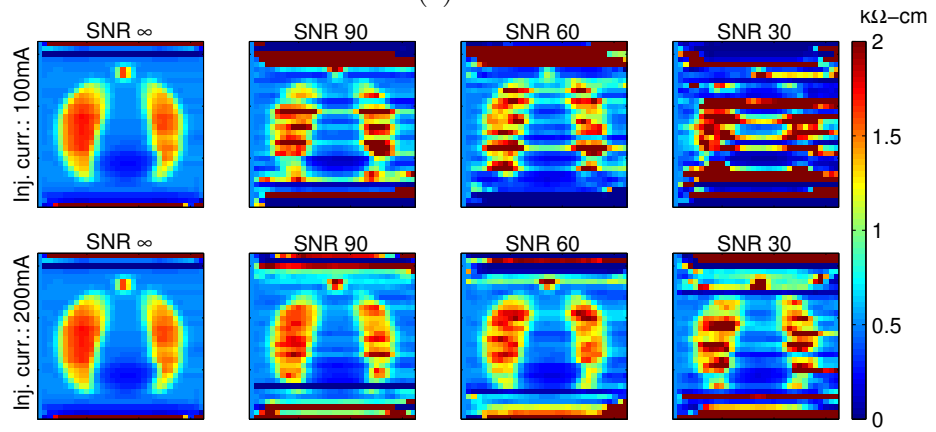
(c)



(d)



(e)



(f)

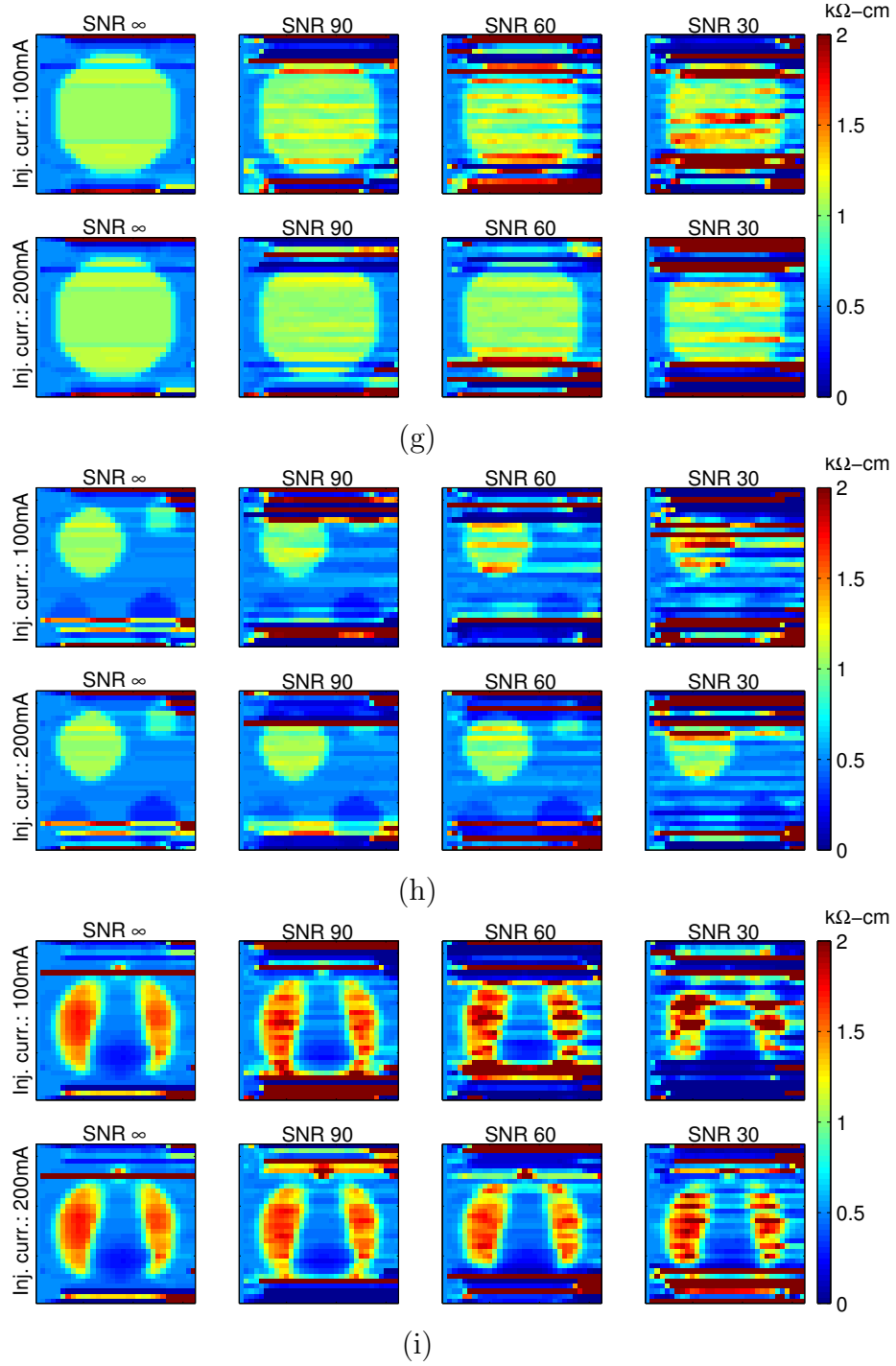


Figure 6.6: Results for the reconstruction along Cartesian grid lines for 32^{nd} slices of simulation models S1 to S9 are given in (a) to (i) respectively. Simulations are made for two different current injection levels and it is seen that increasing the current reduces the effects of noise. Some lines couldn't be reconstructed well because the transversality condition is not hold for their some pixels.

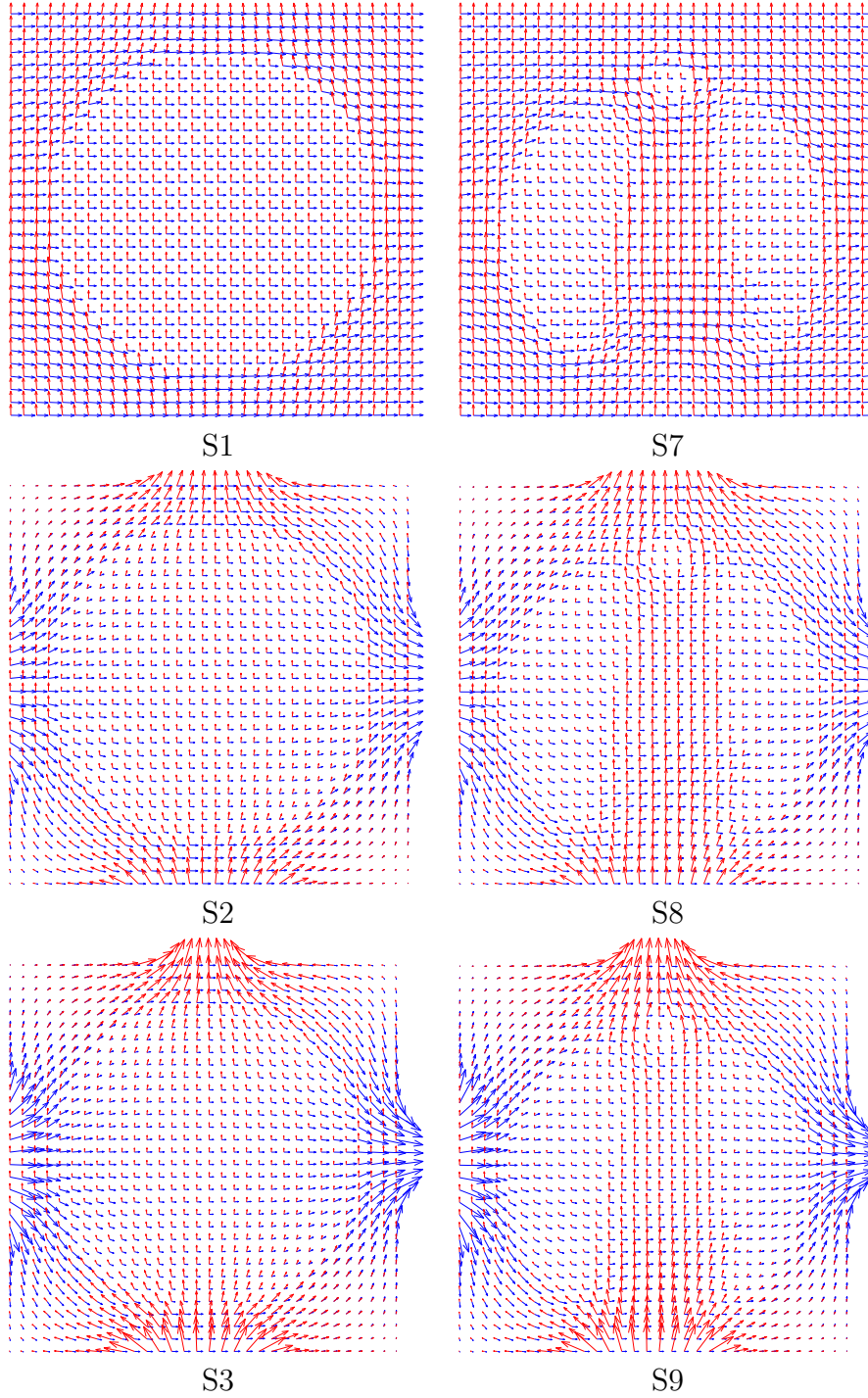


Figure 6.7: The effect of electrode size on the interior current flow is exhibited for two current injection profiles of models S1-S3 and S7-S9. The current flow is more straight for models S1,S7 and transversality condition holds better than others. The electrodes of S1 and S2 occupy the whole surface as shown in (a) and (b) of Figure 6.4.

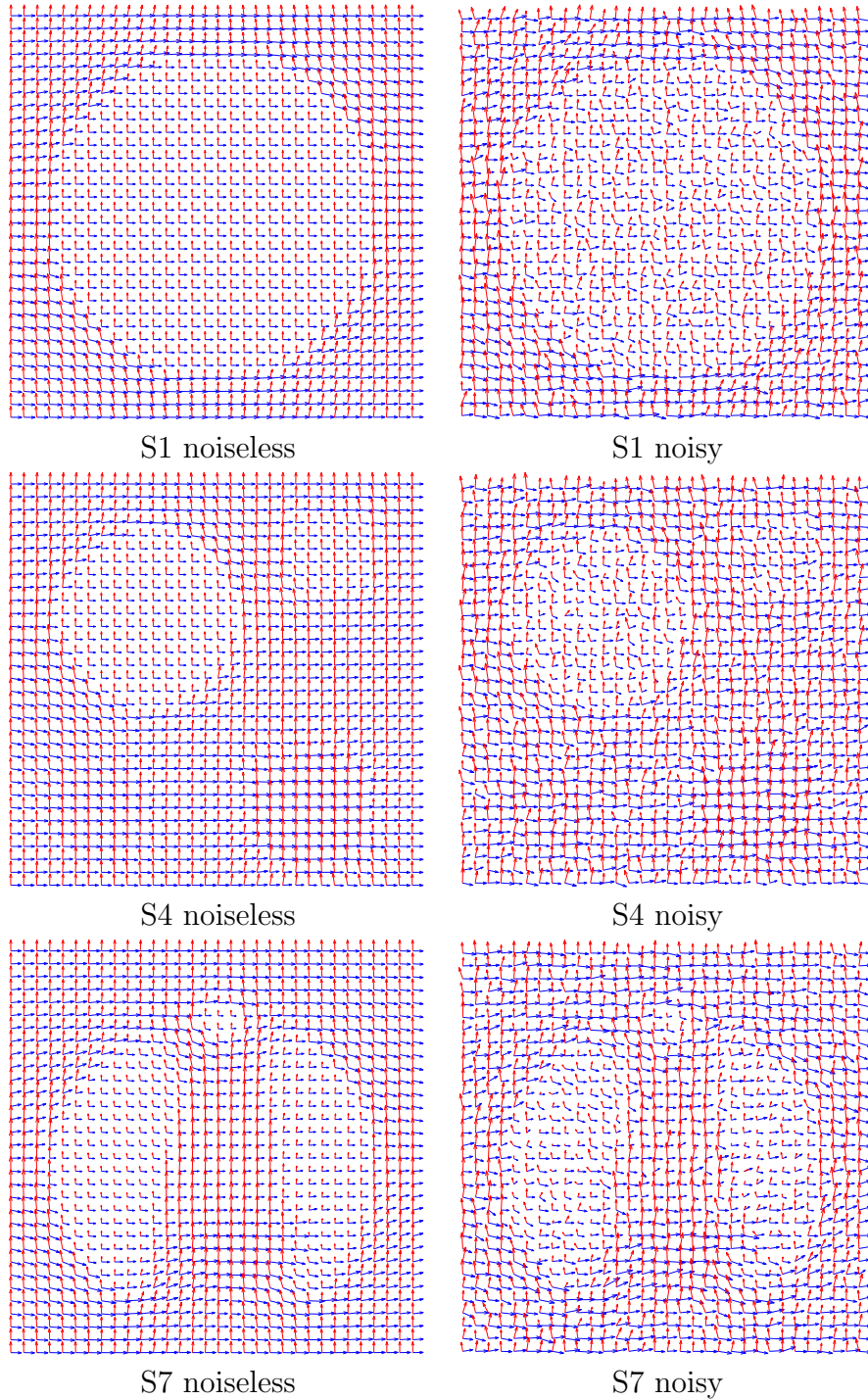
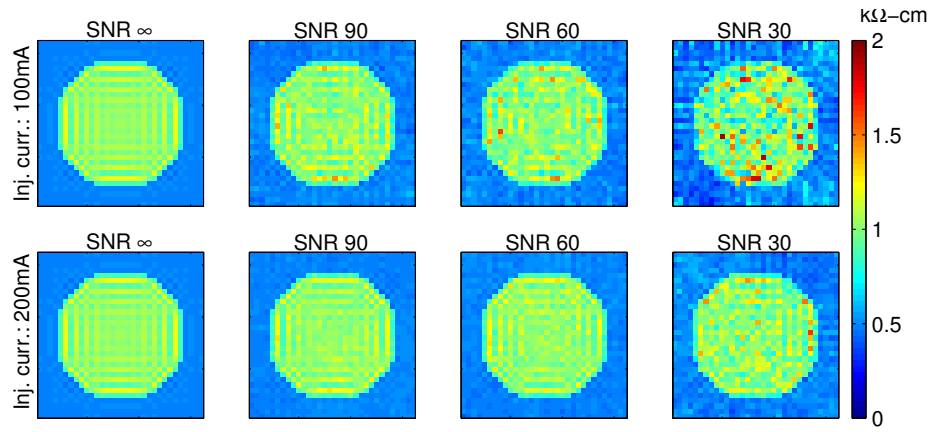
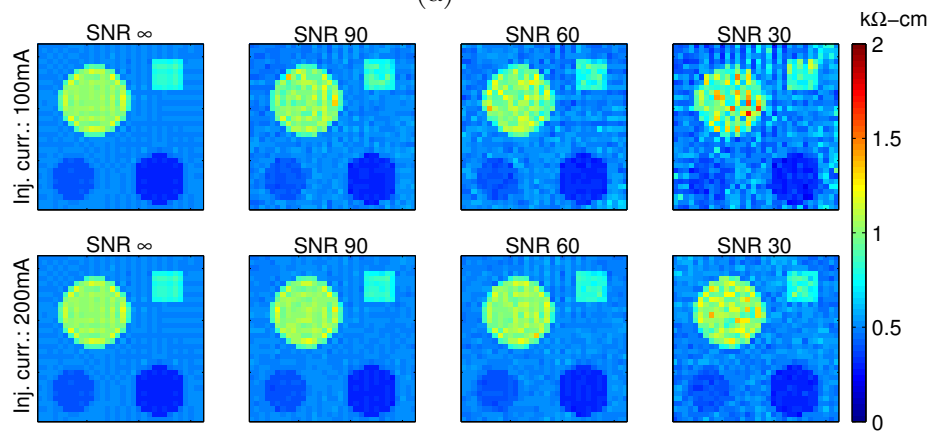


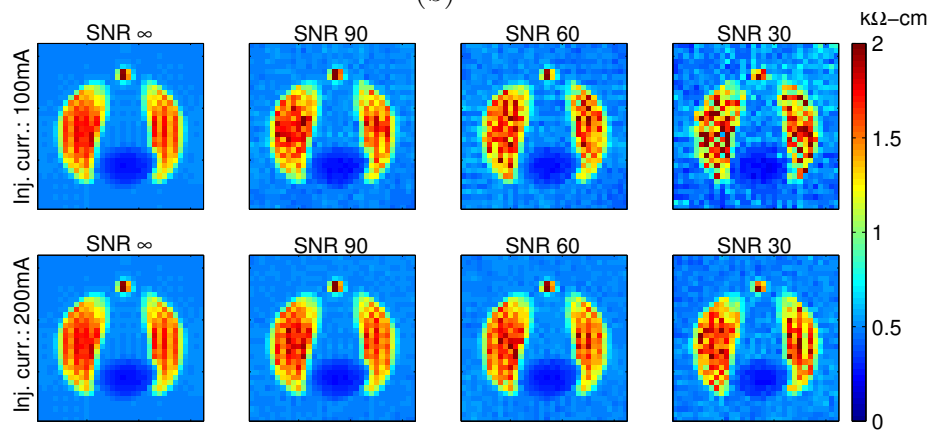
Figure 6.8: The effect of measurement noise on the interior current flow is exhibited for two current injection profiles of models S1, S4 and S7. The current flow is straight for noiseless cases. When a measurement noise was added with SNR of 30, the current directions deviate randomly and transversality condition is not held on some pixels.



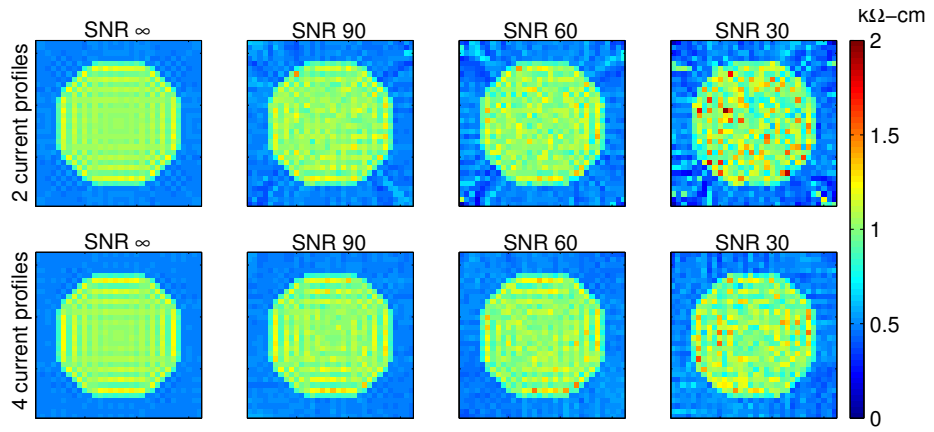
(a)



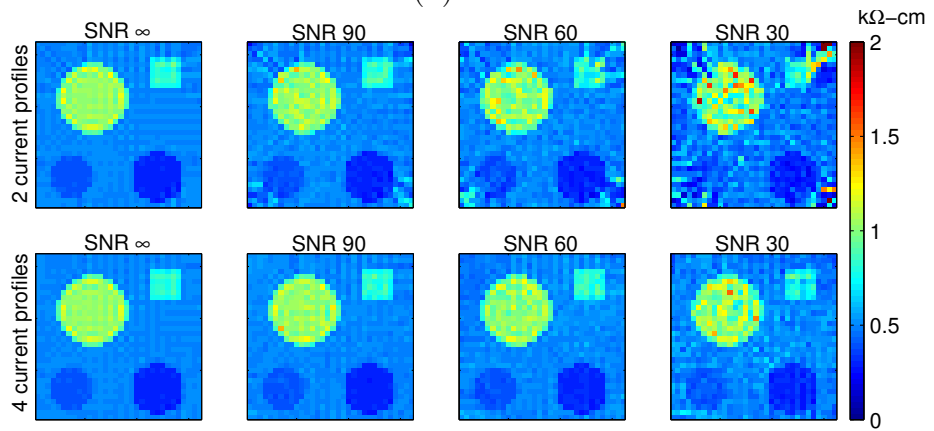
(b)



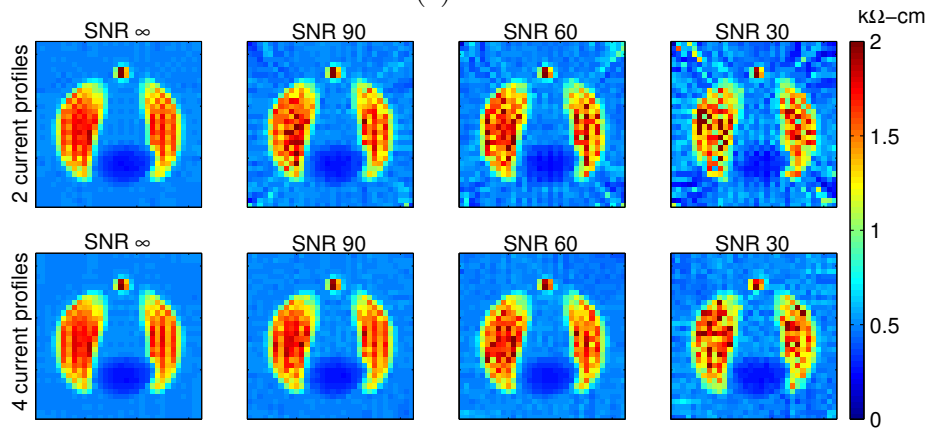
(c)



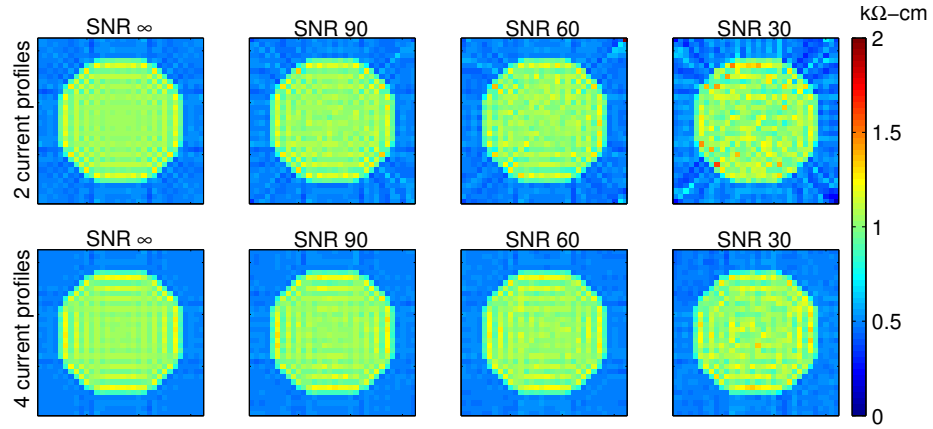
(d)



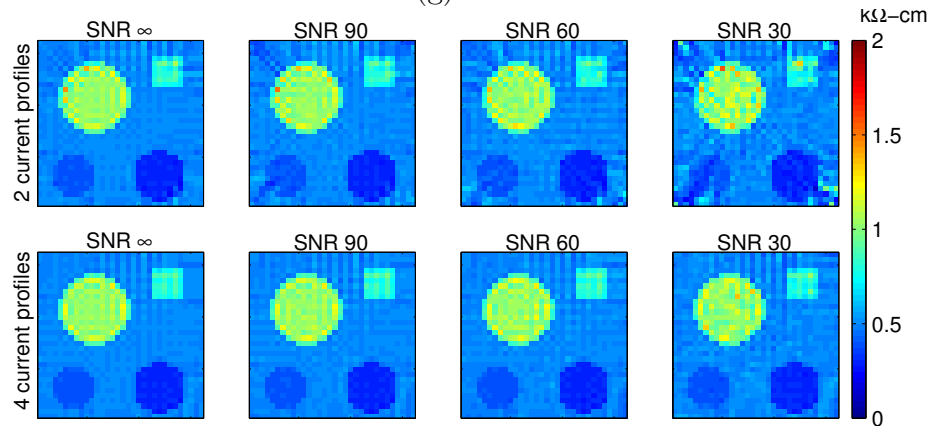
(e)



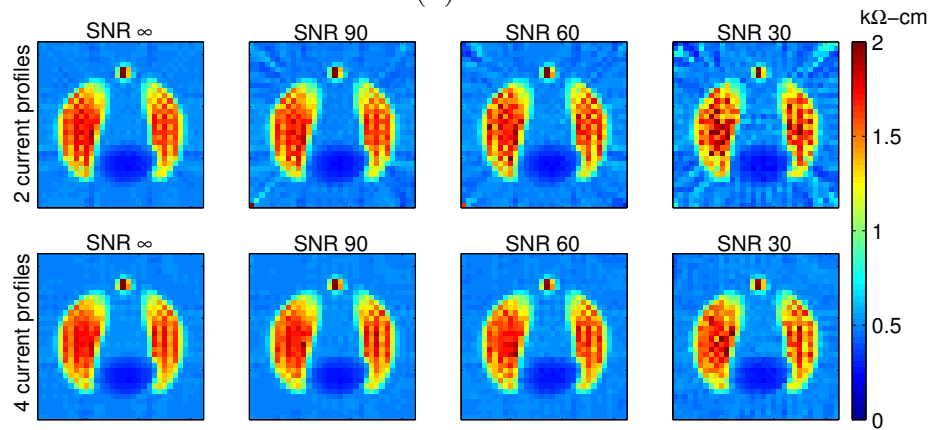
(f)



(g)



(h)



(i)

Figure 6.9: Results of reconstruction by solution of linear equation system for 32^{nd} slices of simulation models S1 to S9 are given in (a) to (i) respectively. Simulations are made for two different current injection levels and it is seen that increasing the current reduces the effects of noise. This algorithm exhibits the best performance against measurement noise. Reconstructed images are the same except for some small fluctuations.

the results for magnetic flux density based algorithms and see that they can be used satisfactorily instead of the current density based ones.

The first algorithm makes reconstruction like the third one of current density based algorithms. It solves a linear equation system in the least-square sense which was obtained by finite difference formulation of Eq 5.2. For absolute imaging, it is assumed that we know one conductivity value. The difference of this algorithm comes from its iterative structure. Algorithm starts with a constant conductivity distribution, like $R = 1$ everywhere. Then, the coefficient matrices and left-hand side vectors are calculated for each current injection profile and they formed in combined matrix form like in Eq. 5.6. Solution of this linear equation system gives a conductivity distribution and algorithm can be iterated further by assigning it as the initial conductivity.

Each iteration step solves the forward problem again for all current injection profiles when obtaining the linear system equations. Since the object is 3D, solution of the forward problem requires the knowledge of the whole 3D resistivity distribution. In a real application of MR-EIT it is not practical to measure B_z for all slices of the object. But the algorithm can reconstruct the conductivity only for slices on which B_z is measured. Then, for second and subsequent iterations we have to assign a conductivity distribution to outside of measurement region.

Assume that we want to reconstruct conductivity of one or a few consecutive slices which are in the middle part of the object. First we have to determine a region of interest and measure B_z for only this region. After first iteration we assigned blurred[†] versions of top and bottom slices of RoI to upper and lower regions of the RoI. For example to fill the upper side of RoI we first placed blurred version of the top slice of RoI to the next upper slice. Then we reblurred this slice and assigned it to the next upper slice. By this approach we assigned smoothly

[†]For blurring, a low-pass filter is applied to image.

varying resistivity distributions to out-of-RoI regions such that as the distance of a slice from RoI increases, the resistivity distribution of that slice approaches a uniform distribution with a value equal to the mean of the resistivity distribution of the starting slice, *i.e.* the top (or the bottom) slice of the RoI. This is shown in Figure 6.10 for model S3 after the first iteration. Our RoI is three slices of 31 to 33. Reblurred versions of bottom slice 31 and top slice 33 are the slices 1 to 30 and 34 to 64 respectively. Some selected slices outside of the RoI are also given.

The blurring filter is a 9×9 FIR filter with $0.18\text{cm}^{-1} -3\text{dB}$ cutoff frequency. Its frequency response is given in Figure 6.11. With this filter, after 6 slices above (or below) of the RoI, approximately uniform resistivity distributions are achieved.

In Figure 6.12 the reconstructed images of the 32^{nd} slice for simulation models S1 to S9 are given for the first five iterations. Also the profile plots of horizontal 16^{th} lines of images are added to the bottom of each image. Relative ℓ_2 norm errors between reconstructed and original images are also given.

It is seen surprisingly that after the first iteration an admissible image is found and subsequent iterations don't provide any further improvement. To the contrary they distort the image and increase relative ℓ_2 norm error. As a comment, the importance of iterations may appear when conductivity levels of regions are considered. In some of our previous simulations we observed that the first iteration reconstructs the edges successfully but conductivity levels are a bit away from the exact ones. Iterations may correct this problem and the conductivity levels approach to the exact levels.

An important question which is waiting for an answer is what size of RoI has to be taken for best performance in the practical sense, computation time and accuracy. It is clear that if we take the whole object as RoI then, this will give us the best accurate reconstruction but worst computation performance

and impracticality. On the other side, reducing its size decreases the reliability of the result. The answer of this question is depends heavily on the individual conductivity distribution and at most we can made experiments on some different RoI sizes.

In Figure 6.13 the effect of RoI size is shown for simulation models S1 to S9. It can be seen from the figure clearly that, the size of RoI is not very important for our simulation models which have conductivity values close to the human body, and RoI with one slice thickness may even be acceptable.

Like other algorithms, the most important problem of this algorithm is also the measurement noise. Unfortunately this algorithm needs two or three times more injected current amounts then previous algorithms to give a similar performance. It needs injected current amounts like $200mA$ or $300mA$ which exceeds a lot the $2mA$ current safety limit of human body. Simulations are made for $200mA$ and $300mA$ and for MRI scanners with SNR's 30, 60 and 90. Results are given in Figure 6.14. By increasing the SNR, we can decrease the amount of injected current and obtain the same noise performance. Manufacturing MRI scanners with high SNR is not very easy and it is still a research area.

Especially for the models which have big sized or long along z direction electrodes, injected current diffuses to everywhere in the object easily so the magnitude of current density $|\mathbf{J}|$ reduces in RoI slices. This makes the RoI more vulnerable to the noise because noise does not increase or decrease with the magnitude of current density. In fact, using small sized electrodes is very nice from the practical point of view. At this point, I have to give my insight about the sizes of electrodes and RoI. Best performance can be obtained if the electrode length is a few slice bigger than the RoI size along the z direction. Because in that case, current flows mostly in RoI, so we can reconstruct RoI with its all slices and with a more little effect of noise.

In order to see the effect of the number of current injection profiles, we looked for singular values of combined coefficient matrices which was formed by different number of current injection profiles. In Figure 6.15 singular values are plotted for simulation models S1 to S9 and for different number of current injection profiles. Used electrode sets are also shown in the legend. For example E1 means we applied current by using electrode set E1, obtained a coefficient matrix and calculated its singular values. If two or more electrode sets were used then the singular values of the combined coefficient matrix was evaluated. It is seen clearly that, adding more current profiles increases the values of singular values and also increases the condition number which was defined before as the ratio of maximum singular value to the minimum singular value. This means that by adding more current injection profiles we will get a well-conditioned coefficient matrix.

Finally, we investigated the performance of sensitivity matrix reconstruction algorithm and made simulations for some selected simulation models only. In fact, the sensitivity matrix approach is a powerful method for 2D objects but for 3D case it takes more computation time to compute the sensitivity matrix for conductivity elements of RoI slices and for each iteration step. In Figure 6.16, simulation results are given for models S1 and S4 to S7 for noiseless case. As Region of interest, slices 31 to 33 are selected and 32nd slice was reconstructed for the first five iterations. The singular values of sensitivity matrix was also calculated and given in Figure 6.17. The condition number of sensitivity matrix is not too high so we can accept it as a well-conditioned matrix. But the results are not as satisfactory as the previous algorithms and also they have a big computation time cost. By increasing the RoI size we may get better results but then computation time increases too. Under these conditions, we can say that this algorithm is not more suitable for 3D reconstruction and needs some fast methods for computation of sensitivity matrix or it may be evaluated analytically.

6.6 Computational Cost of Algorithms

The computation time for reconstruction of conductivity and required computer sources such as memory and software platform is important for clinical implementation of the MR-EIT system. Although there are too many system possibilities, computation times of algorithms for the system used in this thesis may give an insight to the reader.

For simulations, we used a general purpose PC. Its essential hardware consist of a Pentium IV CPU and 256 MB SD-RAM running at 1700 MHz system bus speed. The software platform of PC is Microsoft Windows 2000. Routines of reconstruction algorithms and forward solver are written with Matlab R12. Because Matlab is a command interpreter, coding is made more efficient by vectorizing [40] the loops as much as possible. Matlab supports to call of C++ routines in the m files which contains the source code of Matlab. For generation of magnetic flux density, a C++ code is written and called in m file in order to increase the performance. The source codes of routines are given in Appendix D.

Current density based algorithms are the fastest algorithms. Reconstruction time of a single slice is lower than half a minute for them. Because the magnetic flux density based algorithms solve the forward problem in each iteration step, they need more computation time than the current density based algorithms. For first algorithm which solves a linear equation system, computation of single iteration takes 15 minutes approximately. Second algorithm which uses the sensitivity matrix approach, exhibits the worst performance. Computation of single iteration takes 10 hours by using it. These values belong to models which utilize four current injection profiles. At last, generation of magnetic flux density with all its components on the hexahedron centers takes 2 hours by using the compiled C++ routine.

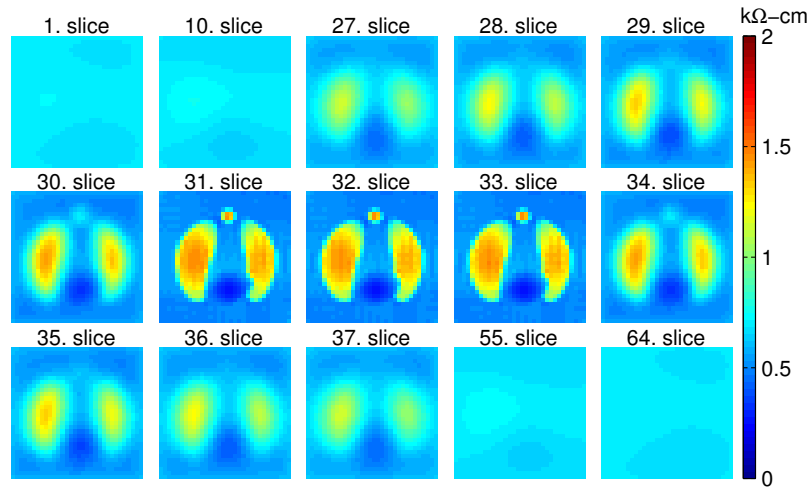


Figure 6.10: Assigned and reconstructed conductivity for model S3 after the first iteration of magnetic flux density based algorithms. The RoI is three slices of 31 to 33 and their conductivity distributions are reconstructed. Blurred conductivities are assigned to slices outside of the RoI and some selected ones are given.

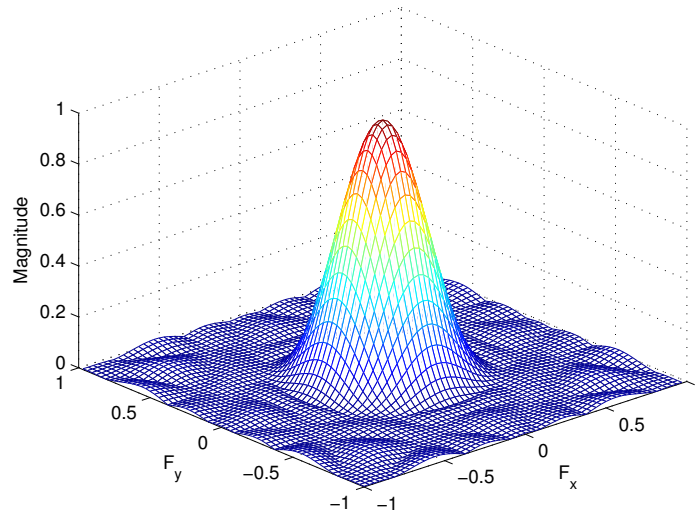
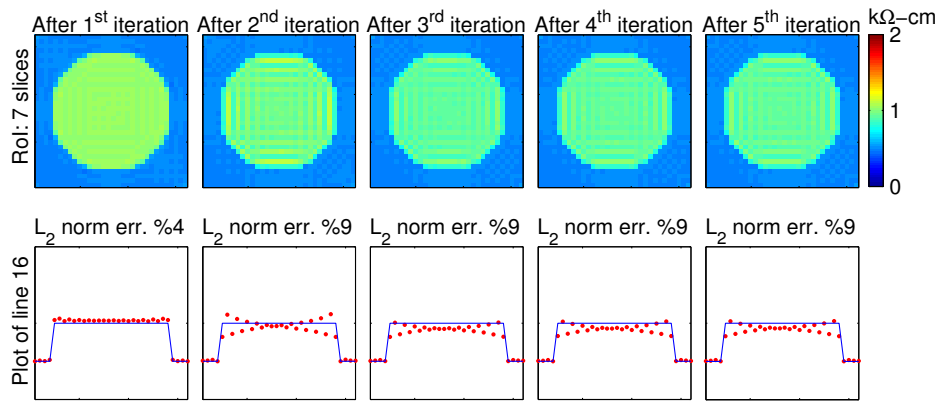
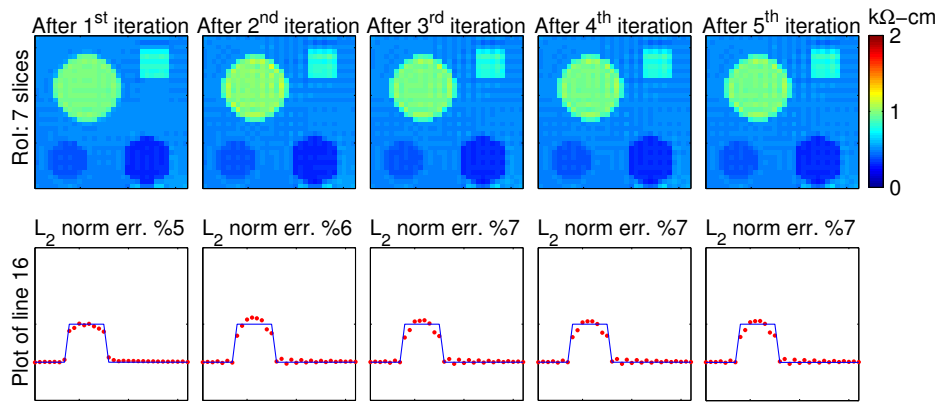


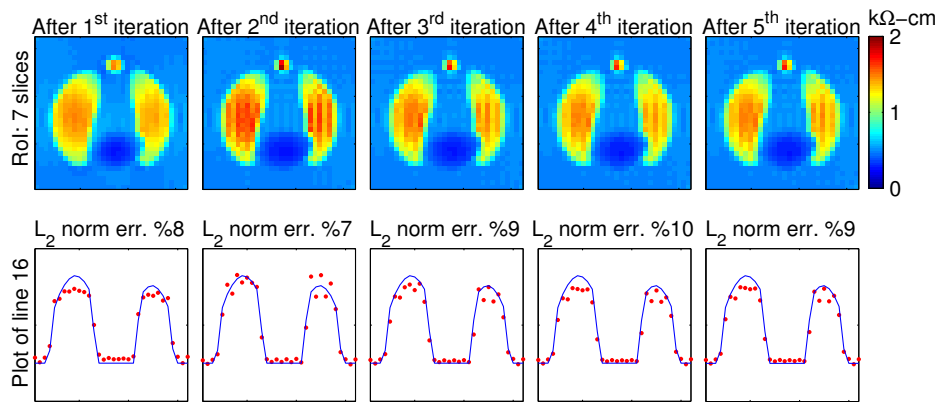
Figure 6.11: Frequency response of the low-pass FIR blur filter.



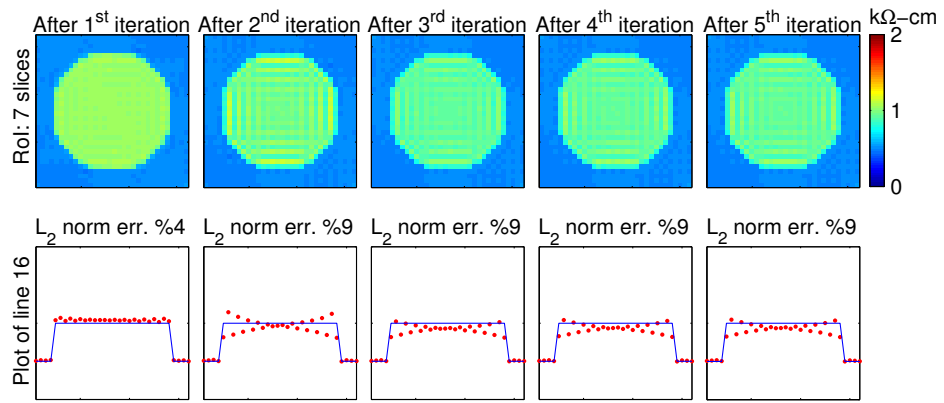
(a)



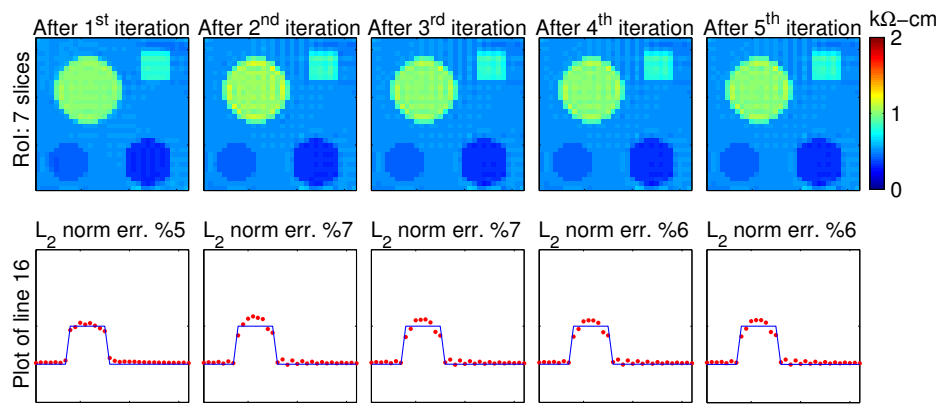
(b)



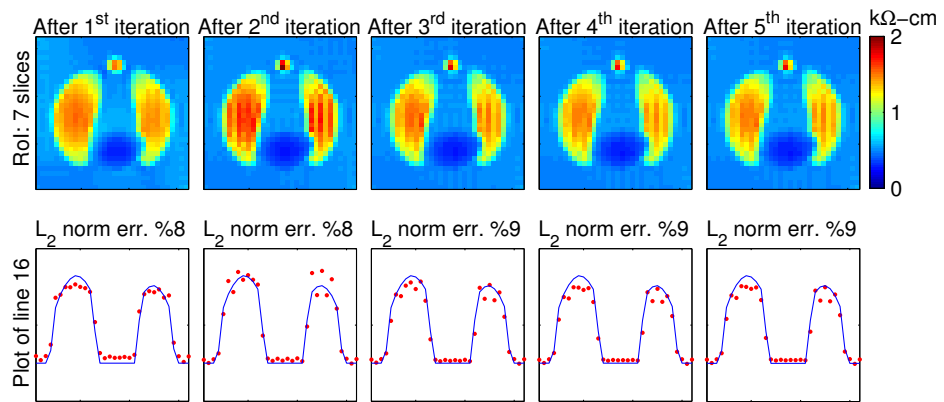
(c)



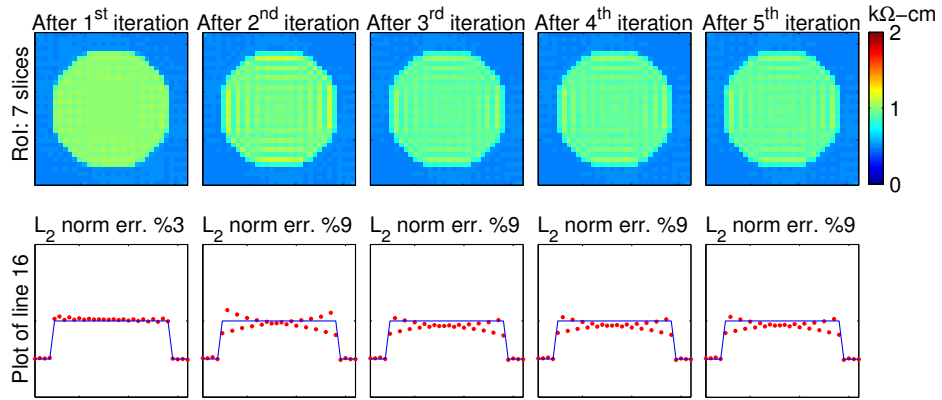
(d)



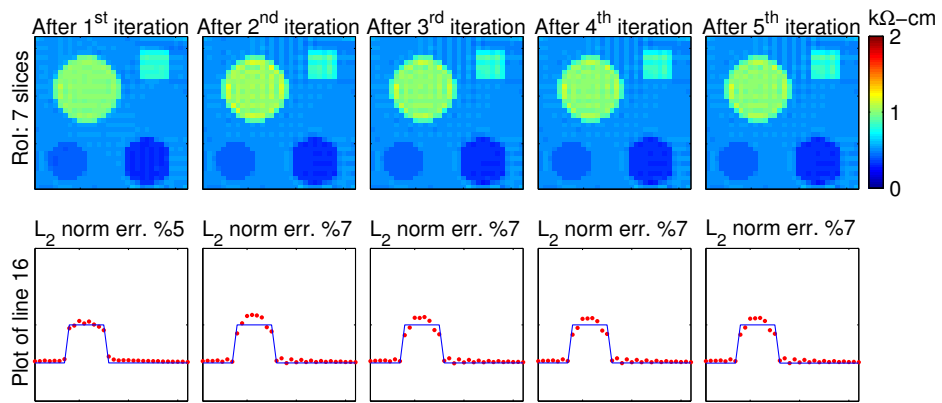
(e)



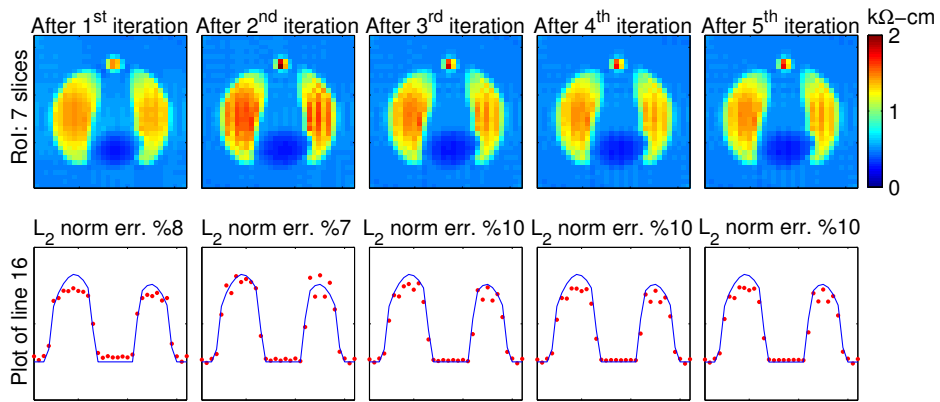
(f)



(g)

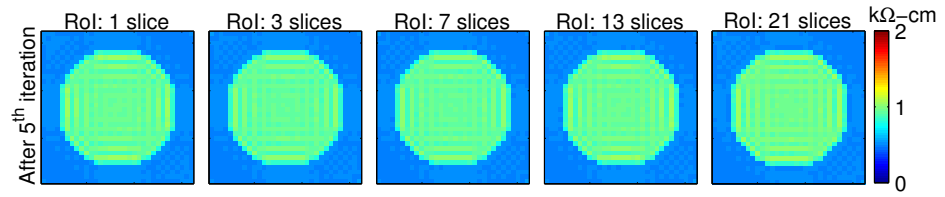


(h)

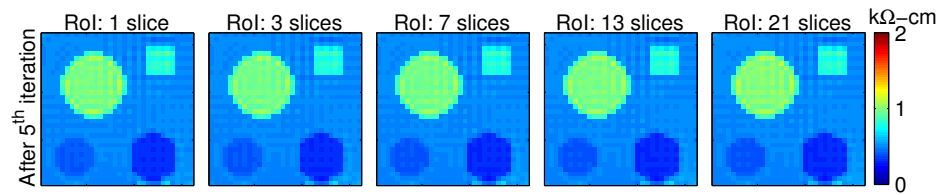


(i)

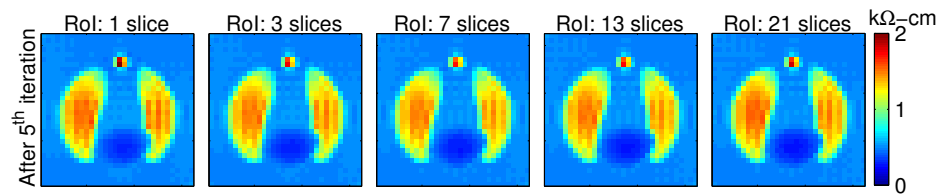
Figure 6.12: Effect of iteration for, reconstruction by solution of linear equation system (\mathbf{B} based). Reconstructed images of 32^{nd} slice for simulation models S1 to S9 are given in (a) to (i) respectively for the first five iterations. No measurement noise was added to the magnetic flux density. Also the profile plots of horizontal 16^{th} lines of images are at the bottom of each image. Relative ℓ_2 norm errors indicate the difference between reconstructed and original images.



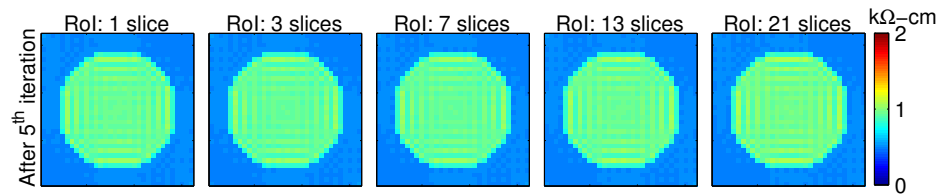
(a)



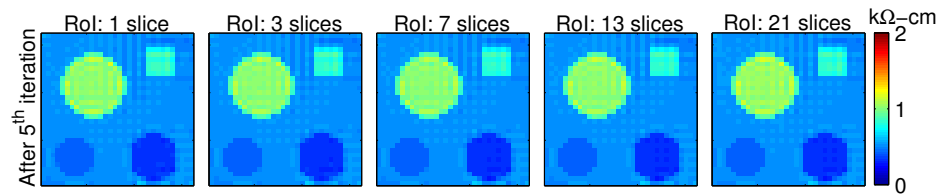
(b)



(c)



(d)



(e)

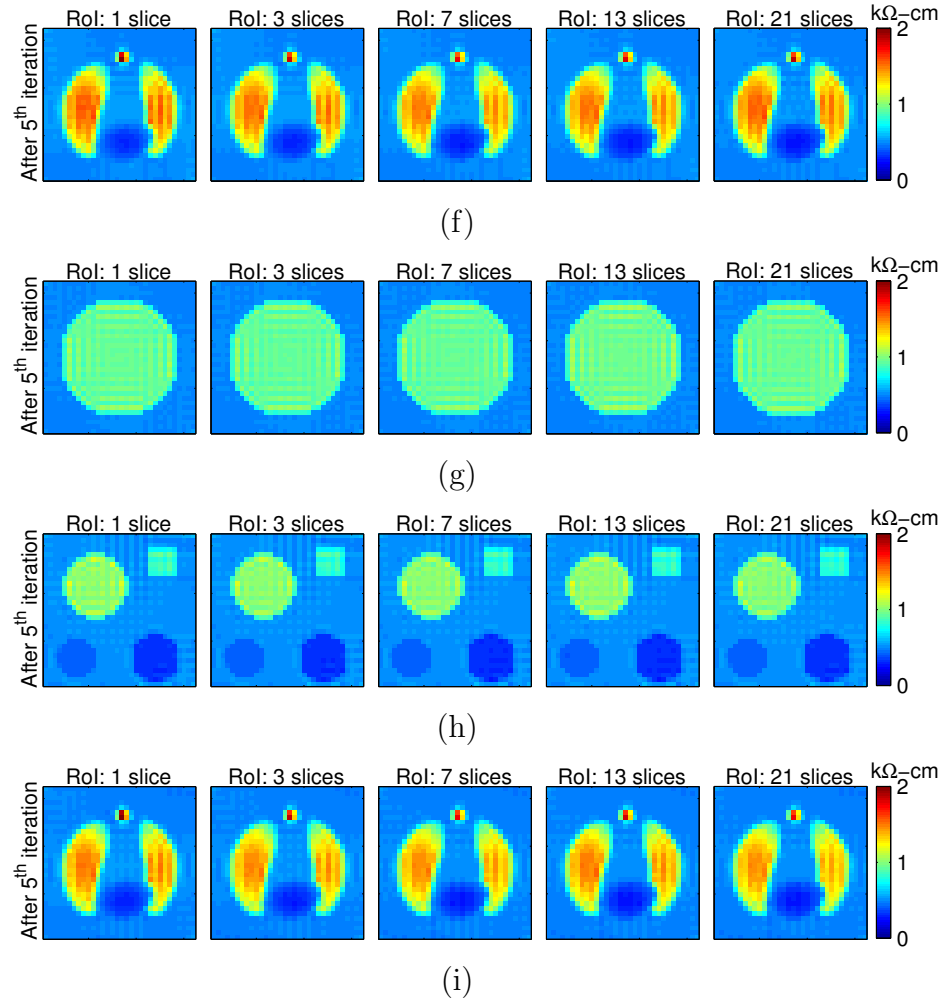
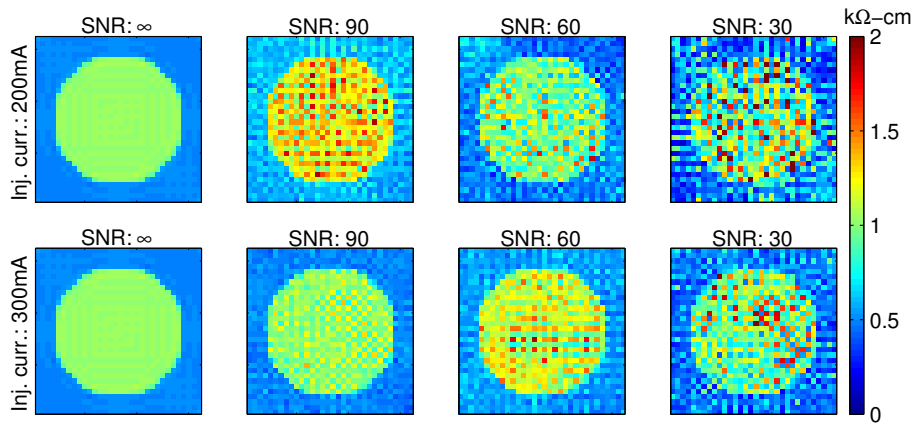
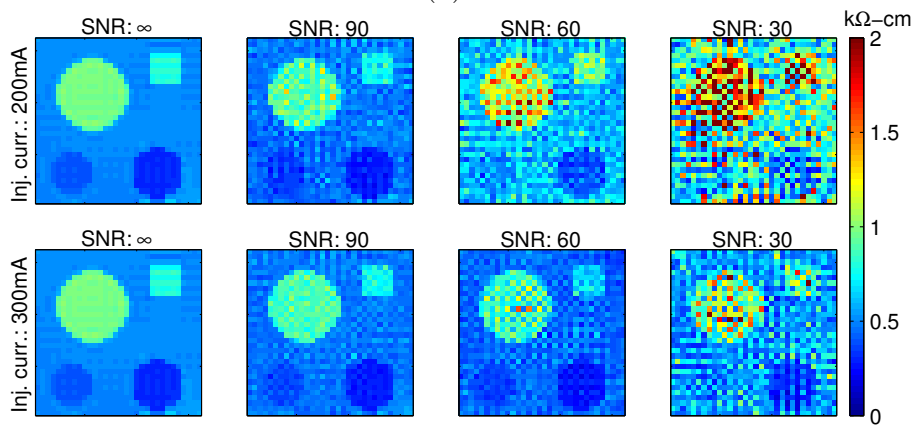


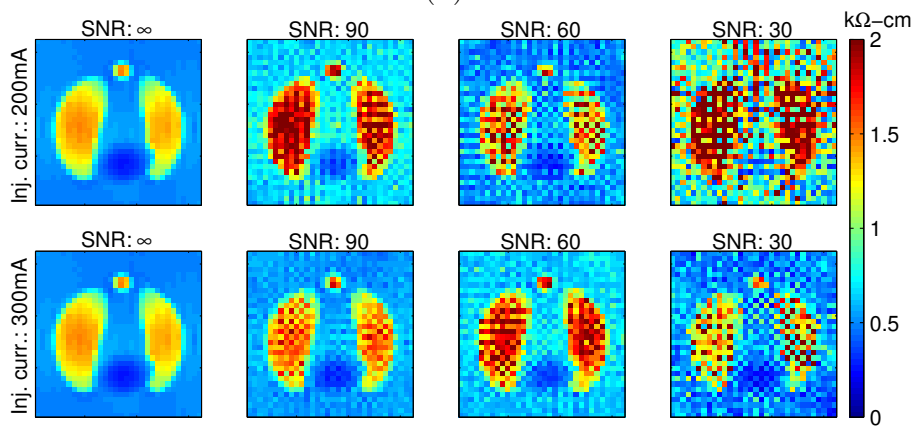
Figure 6.13: Effect of RoI size for, reconstruction by solution of linear equation system (\mathbf{B} based). Reconstructed images of 32nd slice for simulation models S1 to S9 are given in (a) to (i) respectively for five RoI sizes. No measurement noise was added to the current density. It can be seen clearly that, the size of RoI is not more important for our simulation models and almost it may be accepted as in one slice length.



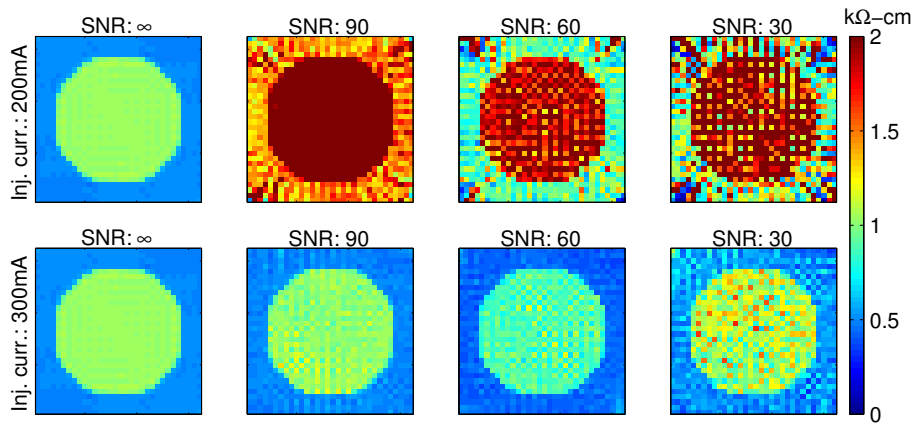
(a)



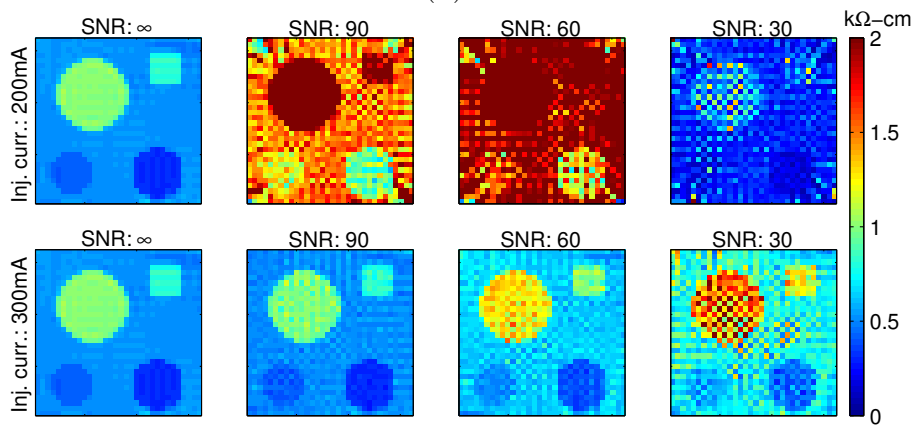
(b)



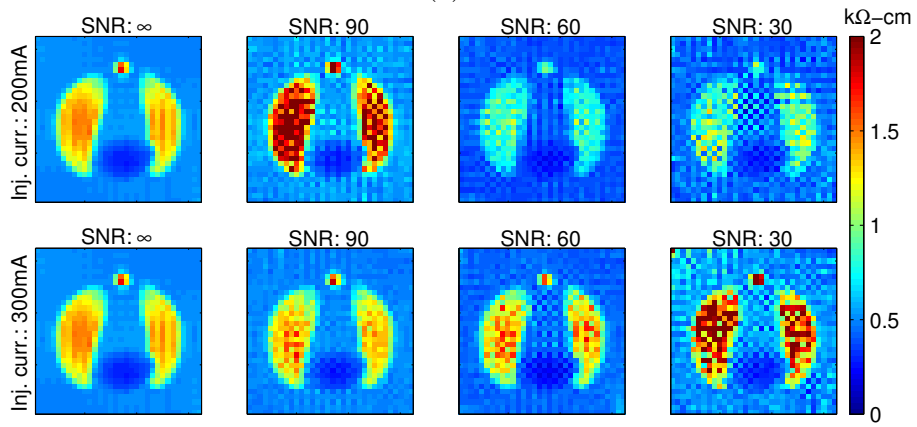
(c)



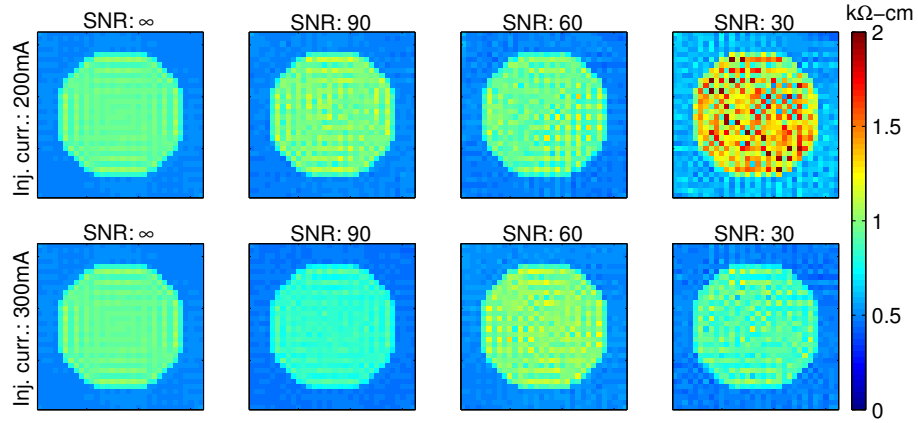
(d)



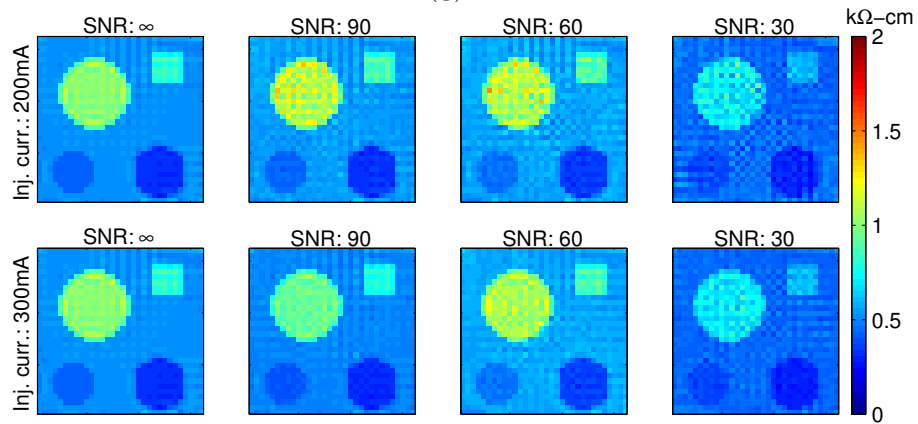
(e)



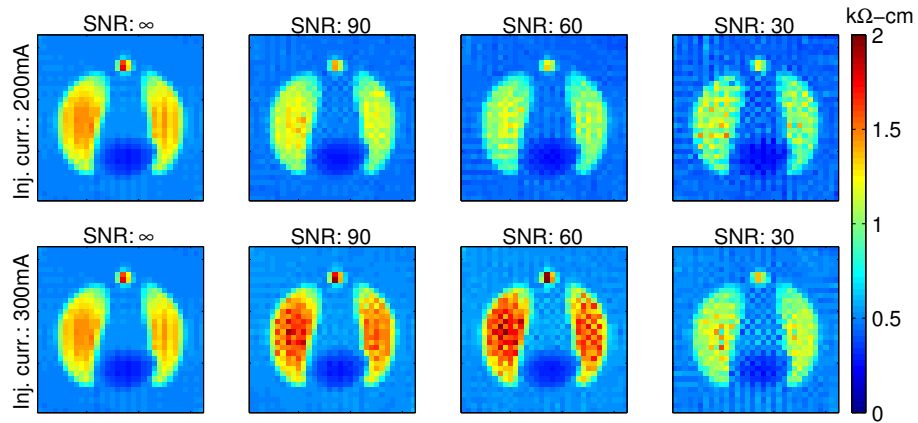
(f)



(g)



(h)



(i)

Figure 6.14: Effect of SNR and injected current amount for, reconstruction by solution of linear equation system (\mathbf{B} based). Reconstructed images of 32^{nd} slice for simulation models S1 to S9 after 5^{th} iteration are given in (a) to (i) respectively for three values of SNR and for two injected current amounts. This algorithm needs more injected current amount or equivalently higher value of SNR for admissible reconstruction against to noise.

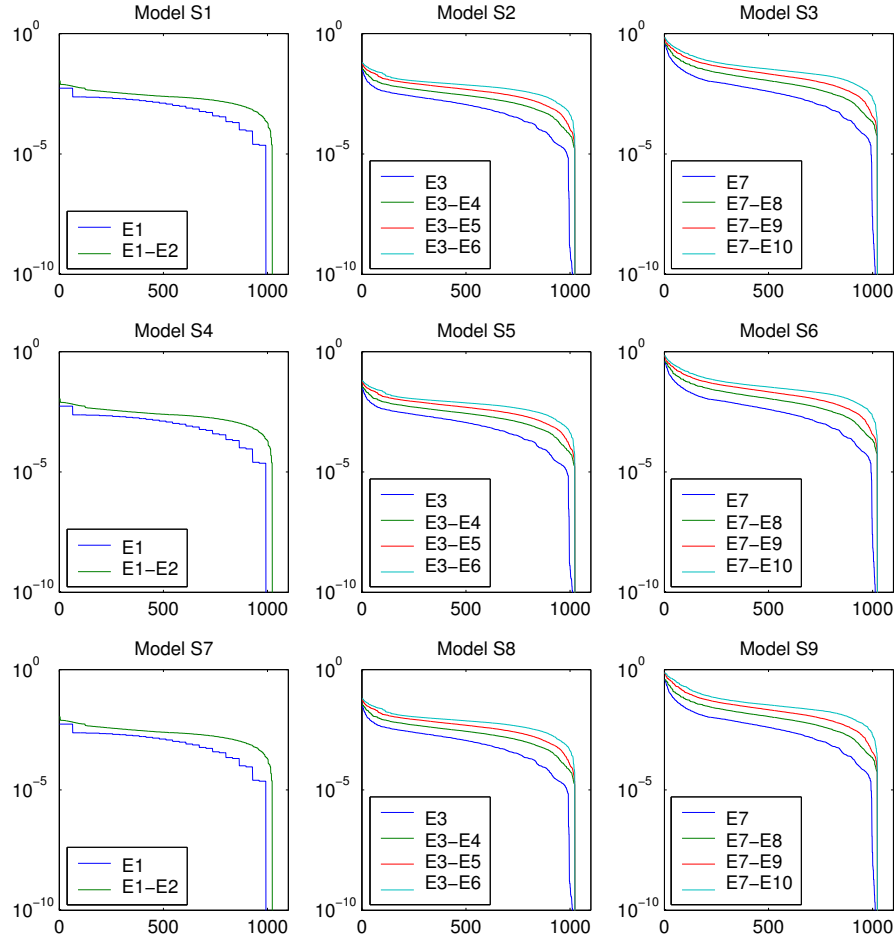


Figure 6.15: Singular values of combined system matrix for different number of current injection profiles are plotted for simulation models S1 to S9. Used electrode sets are also shown on legend. For example E1 means that we applied current by using electrode set E1, obtained a coefficient matrix and calculated its singular values. If two or more electrode sets are used, then the singular values are evaluated for the combined coefficient matrix. It is seen that, adding more current profiles increases the condition number of system matrix.

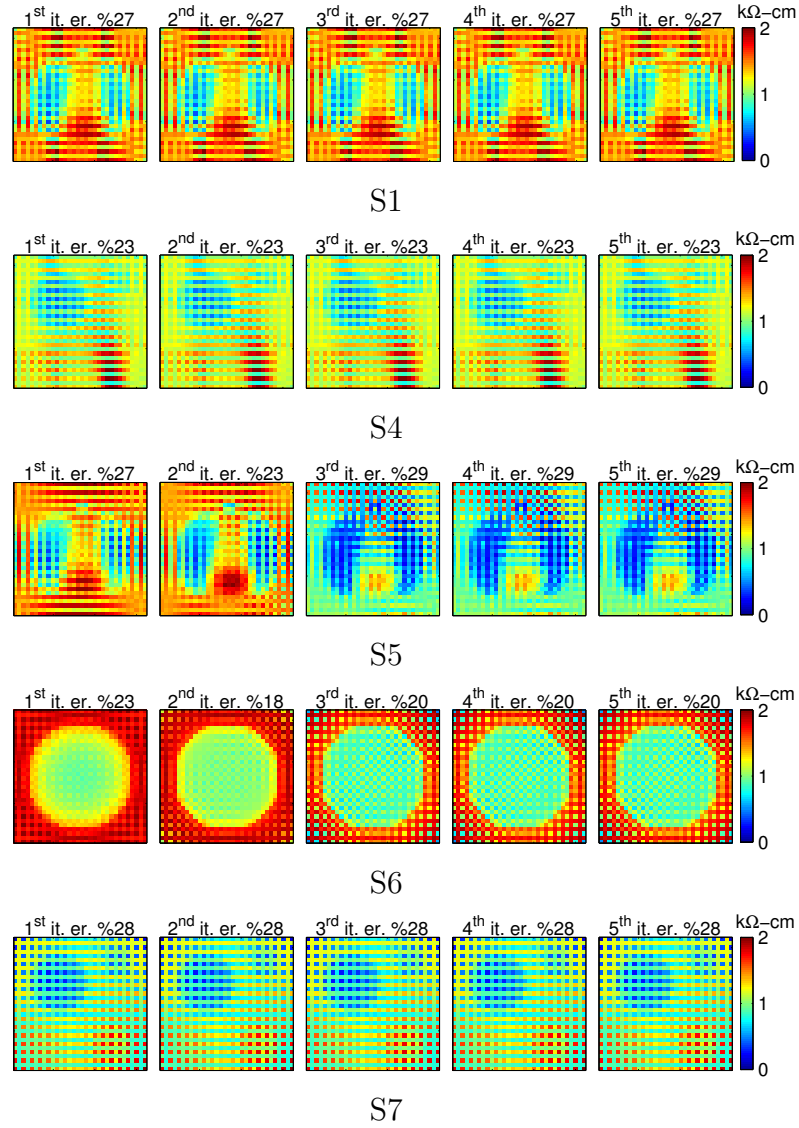


Figure 6.16: Results for the reconstruction by sensitivity matrix. 32^{nd} slices are reconstructed for first five iterations with three slice length RoI. Relative ℓ_2 norm errors are also given.

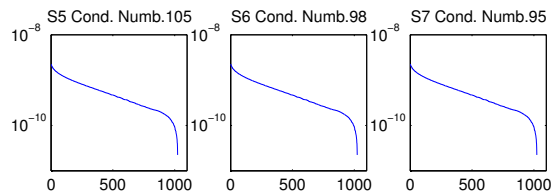


Figure 6.17: Singular value plot of sensitivity matrices for first iteration are given with the corresponding condition numbers.

Chapter 7

Conclusions and Future Work

In this thesis, five new reconstruction algorithms are proposed for a novel imaging technique named as Magnetic Resonance-Electrical Impedance Tomography. These algorithms are proposed for the reconstruction of conductivity of objects in three dimensions. The algorithms can be classified in two categories by depending on the input data they require; those that utilize current density distribution and those that utilize magnetic flux density distribution directly. The algorithms in the first category obtain their current density distributions from magnetic flux density data by a curl operation. But getting the current density in this way is a bit more difficult and in fact impractical because of the need for rotation of object in the MRI scanner.

All algorithms are only capable of reconstruction of relative conductivity images. For absolute imaging, making a single voltage measurement between two points on the boundary is enough as described in Appendix C. In many applications the true values of the pixel conductivities may not be required, and only the relative variation of conductivity in the imaging region may be of interest. In such cases there is no need for the additional single voltage

measurement, to identify the multiplicative constant. This significantly reduces the instrumentation requirements because then only the availability of a current source for current injection suffices.

The study of MR-EIT has two aspects: the forward problem of MR-EIT and the inverse problem of MR-EIT. Both parts are studied consequently and formulations are made under some assumptions. Forward problem deals with the generation of all field components from a known conductivity distribution with boundary conditions which are in fact the amount of injected current and electrodes. On the contrary, inverse problem deals with the reconstruction of conductivity from measured magnetic flux density data and boundary conditions. Once the forward problem is formulated, obtaining a relation for the solution of the inverse problem becomes clearer.

Three different formulations are proposed for the solution of inverse problem. First formulation is obtained by starting from the fact that the curl of static electric field vanishes. A first order nonlinear partial differential equation which relates the gradient of logarithmic resistivity two current density was obtained. This nonlinear equation reduces to a linear one if current density is known. Three different solution algorithms are proposed for solution of this reduced form.

Second formulation replaces the right-hand side of first formulation with the Laplacian of any one component of magnetic flux density. Therefore, the measurement necessity for all three components of magnetic flux density also disappears. But this partial differential equation is still nonlinear because now we don't know the coefficients of it which are in fact J_y and $-J_x$. The cost of this to us is that, we cannot solve it in one step like in current density based algorithms and we need an iterative algorithm. The iterative algorithms consume time and also necessitate the invention of some new concepts like region of interest reconstruction.

The third formulation, sensitivity matrix approach, follows a classical way and linearizes the nonlinear relation between conductivity and magnetic flux density by Taylor series expansion and elimination of higher order terms. This method is simple and reliable but it needs huge computation times for three dimensional objects.

Results of the algorithms of first two formulations are very satisfactory in the absence of noise. But unfortunately, when measurement noise was added, algorithms need either more amount of injection current or MRI scanners with higher SNR values. Injection of current above of the body safety limit is not possible for clinical usage of MR-EIT. Therefore we need much higher SNR values. But selecting the size of RoI small and length of the electrodes just a bit more larger than its length, we may prevent the more diffusing of current in the body and this reduces the effect of noise.

As future work, improvement of noise performance more, implementation of algorithms with real data and design of an induced current MR-EIT systems can be said.

Some priori results of this thesis were presented in a conference [36, 39] in UMIST. Studied current density based algorithms are published [34]. Also the magnetic flux density based algorithm formulating the inverse problem using the Laplacian of one component of magnetic flux density, is submitted [35] for publication.

Bibliography

- [1] Geddes L A and Baker L E, "The Specific Resistance of Biological Materials", *Med. Biol. Eng.*, vol. 5, pp. 271-293, 1967.
- [2] Pethig R, "Dielectric Properties of Biological Materials", *John Wiley and Sons Ltd.*, New York, 1979.
- [3] Pethig R, "Dielectric Properties of Body Tissues", *Clin. Phys. Physiol. Meas.*, vol. 8, suppl. A, pp. 5-12, 1987.
- [4] Pethig R and Kell D B, "The Passive Electrical Properties of Biological Systems: Their Significance in Physiology, Biophysics and Biotechnology", *Phys. Med. Biol.*, vol. 32, pp. 933-970, 1987.
- [5] Boone K, Barber D C and Brown B H, "Imaging with electricity: Report of the European concerned action on impedance tomography", *Journal of Medical Engineering and Technology*, vol. 21, no. 6, pp. 201-232, 1997.
- [6] Sylvester J and Uhlmann G, "A uniqueness theorem for an inverse boundary value problem in electrical prospecting", *Commun. Pure Appl. Math.*, vol. 17, pp. 91-112, 1986.
- [7] Ahlfors S and Ilmoniemi R, "Magnetic imaging of conductivity", *Proc. IEEE EMBS Conf.*, pp. 17178, 1992.

- [8] Korjenevsky A and Sapetsky S C, "Magnetic induction tomography: experimental realization", *Physiol. Meas.*, vol. 21, pp. 8994, 2000.
- [9] Kak A C, Slaney M, "Principles of computerized tomographic imaging", *SIAM*, New York, 1988.
- [10] Wahl R L, Buchanan J W, "Principles and Practice of Positron Emission Tomography", *Lippincott Williams & Wilkins*, Philadelphia, 2002.
- [11] Seagar A D, Baber D C and Brown B H "Theoretical Limits to Sensitivity and Resolution in Impedance Imaging", *Clin. Phys. Physiol. Meas.*, vol. 8, suppl. A, pp. 13-31, 1987. vol. 21, no. 6, pp. 201-232, 1997.
- [12] Köksal A and Eyüboğlu B M, "Determination of optimum injected current patterns in electrical impedance tomography", *Clin. Phys. Physiol. Meas.*, vol. 16, suppl. A, pp. 99-109, 1995.
- [13] Eyüboğlu B M, Köksal A and Demirbilek B M, "Distinguishability analysis of an induced current EIT system using discrete coils", *Phys. Med. Biol.* vol. 45, pp. 1997-2009, 2000.
- [14] Scott G C, Joy M L G, Armstrong R L and Hankelman R M, "Measurement of non-uniform current density by magnetic resonance", *IEEE Trans. on Medical Imaging*, vol. 10, pp. 362-374, 1991.
- [15] Scott G C, Joy M L G, Armstrong R L and Hankelman R M, "Sensitivity of magnetic resonance current density imaging", *Jour. of Mag. Res.*, vol. 97, pp. 235-254, 1992.
- [16] Scott G C, Joy M L G, Armstrong R L and Hankelman R M, "RF current density imaging in homogeneous media", *Magnetic Resonance in Medicine*, vol. 28, pp. 186-201, 1992.

- [17] Scott G C, Joy M L G, Armstrong R L and Hankelman R M, "Rotating frame RF current density imaging", *Magnetic Resonance in Medicine*, vol. 33, pp. 355-369, 1995.
- [18] Scott G C, Joy M L G, Armstrong R L and Hankelman R M, "Electromagnetic considerations for RF current density imaging", *IEEE Trans. on Medical Imaging*, vol. 14, pp. 515-524, 1995.
- [19] Zhang N, "Electrical impedance tomography based on current density imaging", Master's Thesis, University of Toronto, Dept. of Electrical and Electronics Eng., Toronto, Canada, 1992.
- [20] Eyüboğlu B M, Reddy R and Leigh J S, "Magnetic resonance - electrical impedance tomography" *patent no: US6,397,095 B1, provisional patent application on Mar.1 1999*, 2002.
- [21] Özdemir M and Eyüboğlu B M, "Novel fast reconstruction algorithm for magnetic resonance electrical impedance tomography", *Proc. of Biyomut 8th Annual Conference*, CD-ROM, Istanbul, Turkey, 2002.
- [22] Kwon O, Lee J Y and Yoon J R, "Equipotential line method for magnetic resonance electrical impedance tomography" *Inverse Probl.*, vol. 18, pp. 1089-1100, 2002.
- [23] Woo E J, Lee S Y , and Mun C W, "Impedance tomography using current density distribution measured by nuclear magnetic resonance", *SPIE*, vol. 2299, pp. 377-385, 1994.
- [24] Kwon O, Woo E J, Yoon J R, and Seo J K, "Magnetic Resonance Electrical Impedance Tomography (MREIT): Simulation Study of J-Substitution Algorithm", *IEEE Trans. on Biomed. Eng.*, vol. 49-2, pp. 160-167, 2002.

- [25] Kim S, Kwon O, Seo J K, and Yoon J-R, "On a Nonlinear Partial Differential Equation arising in Magnetic Resonance Electrical Impedance Tomography" *SIAM J. MATH. ANAL.*, vol. 34, pp. 511-526, 2002.
- [26] Khang H S, Lee B I, Oh S H, Woo E J, Lee S Y, Cho M H, Kwon O, Yoon J R and Seo J K, "J-substitution algorithm in magnetic resonance electrical impedance tomography (mreit): Phantom experiments for static resistivity images", *IEEE Trans. on Med. Imag.*, Vvol. 21, pp. 695-702, 2002.
- [27] Eyüboğlu B M, Birgül Ö, and İder Y Z, "Dual modality system for high-resolution true conductivity imaging", *XI Int. Conf. Elec. Bioimpedance*, Oslo, Norway pp. 409-413, 2001.
- [28] Birgül Ö, Eyüboğlu B M and İder Y Z, "Current constrained voltage scaled reconstruction (CCVSR) algorithm for MR-EIT and its performance with different probing current patterns", *Phys. Med. Biol.*, vol. 48 pp. 653-671, 2003.
- [29] İder Y Z and Müftüler T, "Measurement of ac magnetic field distribution using magnetic resonance imaging", *IEEE Trans. Med. Imaging*, vol. 16, pp. 617-22, 1997.
- [30] İder Y Z and Birgül Ö, "Use of magnetic field generated by the internal distribution of injected currents for electrical impedance tomography (mr-eit)", *Elektrik, Turkish J. of Elec. Eng. and Comp. Sci.*, vol. 6, pp. 215-225, 1998.
- [31] Seo J K, Kwon O, Yoon J-R, Lee B I and Woo E J, "Single Component of Magnetic Flux Density can produce Resistivity Images in Magnetic Resonance Electrical Impedance Tomography (MREIT)", abstract, *First Mummy Range Workshop on Electrical Impedance Imaging*, Pingree Park, Colorado, 2002.

- [32] B David, C George, "Basic partial differential equations", *Van Nostrand Reinhold Company*, New York, 1992.
- [33] Underwriters Laboratories, American National Standard for Leakage Current for Appliances, ANSI C101-1992, March. 1992.
- [34] İder Y Z, Onart S, Lionheart W R B, "Uniqueness and reconstruction in magnetic resonance-electrical impedance tomography (MR-EIT)", *Physiol. Meas.*, vol. 24, pp. 591-604, 2003.
- [35] İder Y Z, Onart S, "Algebraic Reconstruction for 3D MR-EIT using one component of magnetic flux density", submitted to *Physiol. Meas.*, August, 2003.
- [36] Lionheart W R B, İder Y Z, Onart S, "Method of Characteristics for Reconstruction in MR-EIT formulated as a system of First Order Hyperbolic Partial Differential Equations", 4th Conference on Biomedical Applications of Electrical Impedance Tomography, *UMIST*, April, 2003.
- [37] İder Y Z, Lionheart W R B, Onart S, "Finite Difference and Gradient Based solutions for MR-EIT formulated as a system of First Order Hyperbolic Partial Differential Equations", 4th Conference on Biomedical Applications of Electrical Impedance Tomography, *UMIST*, April, 2003.
- [38] Onart S, İder Y Z, "A new Iterative Reconstruction Algorithm for MR-EIT using the Laplacian of one component of Magnetic Field Intensity", 4th Conference on Biomedical Applications of Electrical Impedance Tomography, *UMIST*, April, 2003.
- [39] İder Y Z, Eyüboğlu B M, "Introduction to MR-EIT and Review of Reconstruction Algorithms ", 4th Conference on Biomedical Applications of Electrical Impedance Tomography, *UMIST*, April, 2003.

- [40] "Using Matlab Version 6", *The MathWorks, Inc.*, pdf file, Sixth Printing, pp. 22-3–22-5, August, 2002.

Appendix A

Finite Element Formulation for 3D Objects

In this study, finite element method is used to find the potential approximately by solving Neumann boundary value problem in Ω . To achieve this, the object is subdivided by tetrahedrons, given in Figure A.1 and potential within an element is approximated by a first order polynomial as,

$$\phi_j(x, y, z) = \alpha_{j1}\phi_{j1} + \alpha_{j2}\phi_{j2} + \alpha_{j3}\phi_{j3} + \alpha_{j4}\phi_{j4} \quad (\text{A.1})$$

where ϕ_{ji} is potential of the i^{th} node of the j^{th} tetrahedron, α_{ji} is the linear shape function defined on j^{th} tetrahedron which is equal to unity at i^{th} node, and vanishes at the rest of the three nodes.

A.1 Weighted Residuals Method

Using Galerkin Weighted Residuals Method, Neumann boundary value problem is to be satisfied on each element by multiplying both sides with the shape function

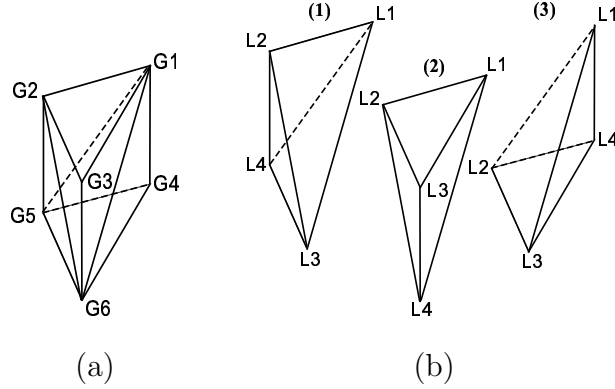


Figure A.1: The object is subdivided by tetrahedrons such that three tetrahedrons constitute a pentahedron.(a) A Pentahedron with global node numbers.(b) Tetrahedrons forming the pentahedron with local node numbers.

and integrating over the element volume as,

$$\int_{\Omega_j} \alpha_{ji} \nabla \cdot (\sigma \nabla \phi_j) dv = 0 \quad (\text{A.2})$$

for $i = 1, \dots, 4$ and $j = 1, 2, \dots, M$ with the assumption of M tetrahedrons are exist in solution domain. Using vector identity $\psi \nabla \cdot \mathbf{A} = \nabla \cdot (\psi \mathbf{A}) - \nabla \psi \cdot \mathbf{A}$ for scalar field ψ and vector field \mathbf{A} , Eq. A.2 becomes to the following form,

$$\int_{\Omega_j} \nabla \alpha_{ji} \cdot \sigma \nabla \phi_j dv = \int_{\Omega_j} \nabla \cdot (\alpha_{ji} \sigma \nabla \phi_j) dv. \quad (\text{A.3})$$

The volume integral on right hand side of Eq. A.3 is equivalent to a surface integral

$$\oint_{\partial \Omega_j} \alpha_{ji} \sigma \nabla \phi_j \cdot d\hat{\mathbf{s}} = - \oint_{\partial \Omega_j} \alpha_{ji} \mathbf{J}_j \cdot d\hat{\mathbf{s}} \quad (\text{A.4})$$

as a result of Divergence Theorem. Using fact $\mathbf{J}_j = -\sigma \nabla \phi_j$, the surface integral has also written in terms of element current density. Assuming σ is constant throughout integration volume and replacing ϕ_j with (A.1), left hand side of Eq. A.3 can be written in a more explicit form. Substitution of this results into left and right hand sides of Eq. A.3 yields,

$$\sigma_j \int_{\Omega_j} \nabla \alpha_{ji} \cdot (\nabla \alpha_{j1} \phi_{j1} + \nabla \alpha_{j2} \phi_{j2} + \nabla \alpha_{j3} \phi_{j3} + \nabla \alpha_{j4} \phi_{j4}) dv = - \oint_{\partial \Omega_j} \alpha_{ji} \mathbf{J}_j \cdot d\hat{\mathbf{s}}. \quad (\text{A.5})$$

The matrix form of Eq. A.5 is also given below after dropping element indices for simplicity,

$$\sigma_j \begin{bmatrix} s_{11} & s_{12} & s_{13} & s_{14} \\ s_{21} & s_{22} & s_{23} & s_{24} \\ s_{31} & s_{32} & s_{33} & s_{34} \\ s_{41} & s_{42} & s_{43} & s_{44} \end{bmatrix} \begin{bmatrix} \phi_1 \\ \phi_2 \\ \phi_3 \\ \phi_4 \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ b_4 \end{bmatrix} \quad (\text{A.6})$$

where $b_i = -\oint_{\partial\Omega_j} \alpha_{ji} \mathbf{J}_j \cdot d\hat{\mathbf{s}}$ is amount of the surface normal current weighted by corresponding shape function and $s_{ab} = \int_{\Omega_j} \nabla\alpha_{ja} \cdot \nabla\alpha_{jb} dv$ is an entry of element coefficient matrix which is determined by only physical positions of nodes.

A.2 Element Assembly

Derived linear matrix relation is valid for only one tetrahedron. For each element, weighted surface normal current vector and coefficient matrix must be calculated. Furthermore, to be consistent, potentials must be same at the shared nodes and surfaces among different elements. Be able to find all node potentials and of course approximated potential field, coefficient matrices and current vectors of each element must be combined into one linear equation. In order to demonstrate this process, assembling of three tetrahedrons in Figure A.1(b) will be described.

For j^{th} element, Eq. A.6 can be written again more briefly and with local node numbering as,

$$\sigma_j \mathbf{S}_j \boldsymbol{\phi}_{\mathbf{jL}} = \mathbf{b}_{\mathbf{jL}} \quad (\text{A.7})$$

where $\mathbf{b}_{\mathbf{jL}} = \left[b_{L1}^{(j)} \quad b_{L2}^{(j)} \quad b_{L3}^{(j)} \quad b_{L4}^{(j)} \right]^T$ and $\boldsymbol{\phi}_{\mathbf{jL}} = \left[\phi_{L1}^{(j)} \quad \phi_{L2}^{(j)} \quad \phi_{L3}^{(j)} \quad \phi_{L4}^{(j)} \right]^T$. After concatenating coefficient matrices and current vectors of three elements in one

equation we obtain,

$$\begin{bmatrix} \sigma_1 \mathbf{S}_1 & 0 & 0 \\ 0 & \sigma_2 \mathbf{S}_2 & 0 \\ 0 & 0 & \sigma_3 \mathbf{S}_3 \end{bmatrix} \begin{bmatrix} \phi_{1L} \\ \phi_{2L} \\ \phi_{3L} \end{bmatrix} = \begin{bmatrix} \mathbf{b}_1 \\ \mathbf{b}_2 \\ \mathbf{b}_3 \end{bmatrix}. \quad (\text{A.8})$$

When elements are assembled, local nodes will be replaced with global ones. by following Figure A.1 carefully it can be seen that the matrix relation between local and global node numbering is,

$$\begin{bmatrix} \phi_{L1}^{(1)} \\ \phi_{L2}^{(1)} \\ \phi_{L3}^{(1)} \\ \phi_{L4}^{(1)} \\ \phi_{L1}^{(2)} \\ \phi_{L2}^{(2)} \\ \phi_{L3}^{(2)} \\ \phi_{L4}^{(2)} \\ \phi_{L1}^{(3)} \\ \phi_{L2}^{(3)} \\ \phi_{L3}^{(3)} \\ \phi_{L4}^{(3)} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} \phi_{G1} \\ \phi_{G2} \\ \phi_{G3} \\ \phi_{G4} \\ \phi_{G5} \\ \phi_{G6} \end{bmatrix} \quad (\text{A.9})$$

or simply,

$$\phi_L = \mathbf{C} \phi_G. \quad (\text{A.10})$$

Replacing this into Eq. A.8 and multiplying both sides with transpose of \mathbf{C} we obtain,

$$\mathbf{C}^T \begin{bmatrix} \sigma_1 \mathbf{S}_1 & 0 & 0 \\ 0 & \sigma_2 \mathbf{S}_2 & 0 \\ 0 & 0 & \sigma_3 \mathbf{S}_3 \end{bmatrix} \mathbf{C} \phi_G = \mathbf{C}^T \begin{bmatrix} \mathbf{b}_1 \\ \mathbf{b}_2 \\ \mathbf{b}_3 \end{bmatrix} \quad (\text{A.11})$$

or in simple form,

$$\mathbf{A}_{\text{con}} \phi = \mathbf{b} \quad (\text{A.12})$$

where \mathbf{A}_{con} is 6×6 connected coefficient matrix, ϕ is 6×1 unknown node potential vector and \mathbf{b} is 6×1 current vector specified by boundary conditions.

Appendix B

Reconstruction of R on an Equipotential Surface

If R is specified at any single point then it can be found completely on the equipotential surface including that point.

One method for this is first calculation of R on the characteristic curve L passing from the specified point, which is obtained by using one of the rows of the hyperbolic system in (3.9). Then, by starting from points on L , complete the equipotential surface along characteristic curves of either one of the remaining rows of the hyperbolic system. L is a non-characteristic curve for the equipotential surface in the complete sense, *i.e.* it intersects all characteristic curves of either one of the remaining rows of the system. This completes the determination of R on that equipotential surface.

The other method is finding R at any other point in the equipotential surface by integrating along any path in the surface starting from the specified point. For the hyperbolic system

$$\tilde{\mathbf{J}}^T \nabla R = \underline{\mathbf{b}} \tag{B.1}$$

where $\underline{\mathbf{b}} = -\nabla \times \mathbf{J}$, let $\underline{\mathbf{x}}(s)$ be a curve on a characteristic surface joining the point we are interested in to $\underline{\mathbf{x}}(s_0)$ where R is known. Since $\underline{\mathbf{x}}(s)$ is on the characteristic surface, $\mathbf{x}'(s) \in \text{Range}(\tilde{\mathbf{J}}_i(s), i = 1, 2, 3)$ and $\mathbf{x}'(s) \neq 0$. Hence there is a non-zero vector field $\underline{\mathbf{c}}(s)$ along $\underline{\mathbf{x}}(s)$ such that

$$\mathbf{x}'(s) = \tilde{\mathbf{J}}(\mathbf{x}(s))\underline{\mathbf{c}}(s). \quad (\text{B.2})$$

Now, $\underline{\mathbf{c}}^T \tilde{\mathbf{J}}^T \nabla R = \underline{\mathbf{c}}^T \underline{\mathbf{b}}$ along the curve $\underline{\mathbf{x}}(s)$, and hence $(\tilde{\mathbf{J}}\underline{\mathbf{c}}) \cdot \nabla R = \underline{\mathbf{c}}^T \underline{\mathbf{b}}$, and $\mathbf{x}'(s) \cdot \nabla R = \underline{\mathbf{c}}^T \underline{\mathbf{b}}$. Therefore,

$$R(\underline{\mathbf{x}}(s)) = R(\underline{\mathbf{x}}(s_0)) + \int_{s_0}^s (\underline{\mathbf{c}}^T \underline{\mathbf{b}})(\underline{\mathbf{x}}(t)) dt. \quad (\text{B.3})$$

Appendix C

Determination of the Scalar Factor for Absolute Imaging

All reconstruction algorithms proposed in this study are capable of to find relative conductivity or resistivity distributions only. In fact, in the sense of diagnostically valuable information, there is no any difference between relative and absolute images. But if for special cases it is needed to the exact values of the images then this can be achieved by a single voltage measurement between any to points on the boundary. The value of this potential difference provides us the enough information to uniquely determine the scaling factor between exact and relative images.

As a definition, the potential difference between any two points on the boundary is equal to the negative path integral of electric field along on any path which connecting the specified points. For boundary points $p1$ and $p2$ by also using Ohm's law these path integral of electric field is like,

$$\nu = V_{p1} - V_{p2} = - \int_{p1}^{p2} \sigma \mathbf{J} \cdot d\boldsymbol{\ell} \quad (\text{C.1})$$

where σ represents the exact conductivity distribution and ν is the voltage

difference between the specified points. Now assume that k is the positive constant scaling factor such that,

$$\boldsymbol{\sigma}_r = k\boldsymbol{\sigma} \tag{C.2}$$

where $\boldsymbol{\sigma}_r$ is the relative conductivity distribution which is found by a reconstruction algorithm. Substituting this equation into previous one and solving the result one for k yields,

$$k = -\frac{1}{\nu} \int_{p1}^{p2} \boldsymbol{\sigma}_r \mathbf{J} \cdot d\boldsymbol{\ell}. \tag{C.3}$$

Appendix D

Source Codes for Matlab

In this part, source codes of algorithms and forward solver are given for running under Matlab. Forward solver uses the routines femsolver, formacon, elmatrprism, findEfield, findnodeBzfield, BzGenerator and findNodesForCurrent. Current density based algorithms uses the routines IntEquiPotLines, IntCart-GridLines and SolveLinEqSys. Magnetic flux density based algorithms uses the routines FindwithBz, SensMatAlg, SliceBlurringFunc, GenSensMat and ForProbSolv.

Listing D.1: femsolver

```
function [J] = femsolver(R,InjCurr,CurrDir,x_perc,y_perc,z_perc,x_off,y_off,z_off,ext,een,zen)
% Forward solver with FEM. Calculates phi, E and J.
% Define global variables.
global nodes
5 global nelmts
global x
global y
global z
global elmn
10 global sigma
% load mesh data.
load('mesh3dtri' ext);
% Form right-hand side
B = InjCurr * findNodesForCurrent(CurrDir,x_perc,y_perc,z_perc,x_off,y_off,z_off,ext);
15 % Find middle point to set its potential to zero.
```

```

zero_node = find(x==round(een/2) & y==round(een/2) & z==round(zen/2));%node to be taken as zero voltage
B(zero_node) = 0;
% This part is valid if logarithmic resistivity is given.
sigma_hex = exp(-R);
20 % sigma_hex is for hexahedrons, sigma is for pentahedrons.
sigma = repmat(sigma_hex',2,1);
sigma = sigma(1:end)';
% Form connected A matrix.
A = formacon;
25 % Set the reference potential.
A(zero_node,:) = 0;
A(zero_node,zero_node) = 1;
% Solve potential field.
phi = pcg(A,B,1e-6,10000,[],[],zeros(nodes,1));
30 % Find E field.
E = findEField(phi);
% Find J field.
J = [E(:,1).*sigma_hex E(:,2).*sigma_hex E(:,3).*sigma_hex];

```

Listing D.2: formacon

```

function [A] = formacon
% Forms connected A matrix.

global nodes
5 global nelmts
global x
global y
global z
global elmn
10 global sigma

[s1]=elmatrprism(1);
[s2]=elmatrprism(2);

15 s12 = sigma .* (repmat([s1(1,2);s2(1,2)],nelmts/2,1));
s13 = sigma .* (repmat([s1(1,3);s2(1,3)],nelmts/2,1));
s14 = sigma .* (repmat([s1(1,4);s2(1,4)],nelmts/2,1));
s15 = sigma .* (repmat([s1(1,5);s2(1,5)],nelmts/2,1));
s16 = sigma .* (repmat([s1(1,6);s2(1,6)],nelmts/2,1));
20 s23 = sigma .* (repmat([s1(2,3);s2(2,3)],nelmts/2,1));
s24 = sigma .* (repmat([s1(2,4);s2(2,4)],nelmts/2,1));
s25 = sigma .* (repmat([s1(2,5);s2(2,5)],nelmts/2,1));
s26 = sigma .* (repmat([s1(2,6);s2(2,6)],nelmts/2,1));
s34 = sigma .* (repmat([s1(3,4);s2(3,4)],nelmts/2,1));
25 s35 = sigma .* (repmat([s1(3,5);s2(3,5)],nelmts/2,1));
s36 = sigma .* (repmat([s1(3,6);s2(3,6)],nelmts/2,1));
s45 = sigma .* (repmat([s1(4,5);s2(4,5)],nelmts/2,1));
s46 = sigma .* (repmat([s1(4,6);s2(4,6)],nelmts/2,1));
s56 = sigma .* (repmat([s1(5,6);s2(5,6)],nelmts/2,1));
30
s11 = sigma .* (repmat([s1(1,1);s2(1,1)],nelmts/2,1));
s22 = sigma .* (repmat([s1(2,2);s2(2,2)],nelmts/2,1));
s33 = sigma .* (repmat([s1(3,3);s2(3,3)],nelmts/2,1));
s44 = sigma .* (repmat([s1(4,4);s2(4,4)],nelmts/2,1));

```

```

35 s55 = sigma .* (repmat([s1(5,5);s2(5,5)],nelmts/2,1));
s66 = sigma .* (repmat([s1(6,6);s2(6,6)],nelmts/2,1));

A=sparse(elmn(:,1),elmn(:,2),s12,nodes,nodes);
40 A=A+sparse(elmn(:,1),elmn(:,3),s13,nodes,nodes);
A=A+sparse(elmn(:,1),elmn(:,4),s14,nodes,nodes);
A=A+sparse(elmn(:,1),elmn(:,5),s15,nodes,nodes);
A=A+sparse(elmn(:,1),elmn(:,6),s16,nodes,nodes);
A=A+sparse(elmn(:,2),elmn(:,3),s23,nodes,nodes);
45 A=A+sparse(elmn(:,2),elmn(:,4),s24,nodes,nodes);
A=A+sparse(elmn(:,2),elmn(:,5),s25,nodes,nodes);
A=A+sparse(elmn(:,2),elmn(:,6),s26,nodes,nodes);
A=A+sparse(elmn(:,3),elmn(:,4),s34,nodes,nodes);
A=A+sparse(elmn(:,3),elmn(:,5),s35,nodes,nodes);
50 A=A+sparse(elmn(:,3),elmn(:,6),s36,nodes,nodes);
A=A+sparse(elmn(:,4),elmn(:,5),s45,nodes,nodes);
A=A+sparse(elmn(:,4),elmn(:,6),s46,nodes,nodes);
A=A+sparse(elmn(:,5),elmn(:,6),s56,nodes,nodes);
A=A+A.';
55 A=A+sparse(elmn(:,1),elmn(:,1),s11,nodes,nodes);
A=A+sparse(elmn(:,2),elmn(:,2),s22,nodes,nodes);
A=A+sparse(elmn(:,3),elmn(:,3),s33,nodes,nodes);
A=A+sparse(elmn(:,4),elmn(:,4),s44,nodes,nodes);
A=A+sparse(elmn(:,5),elmn(:,5),s55,nodes,nodes);
60 A=A+sparse(elmn(:,6),elmn(:,6),s66,nodes,nodes);

```

Listing D.3: elmatrprism

```

function [s] = elmatrprism(ie)
% Finds the element matrix for element ie.

global x
5 global y
global z
global elmn

stetra = zeros(12,12);
10 for i=1:3
tetramat = [1 x(elmn(ie, i)) y(elmn(ie, i)) z(elmn(ie, i))
            1 x(elmn(ie,1+i)) y(elmn(ie,1+i)) z(elmn(ie,1+i))
            1 x(elmn(ie,2+i)) y(elmn(ie,2+i)) z(elmn(ie,2+i))
            1 x(elmn(ie,3+i)) y(elmn(ie,3+i)) z(elmn(ie,3+i))];
15 tetraVOL = abs(det(tetramat)/6);
delalfa = [0 1 0 0;0 0 1 0;0 0 0 1] * inv(tetramat);
j=1+(i-1)*4;
stetra(j:j+3,j:j+3) = tetraVOL * delalfa' * delalfa;
end
20 c = [eye(4) zeros(4,2);zeros(4,1) eye(4) zeros(4,1);zeros(4,2) eye(4)];
s = c' * stetra * c;

```

Listing D.4: findEfield

```
function [E] = findEfield(phi)
% Finds electric field for every hexahedron from node voltages.

global nodes
5 global nelmts
global x
global y
global z
global elmn
10 global sigma

E = zeros(nelmts/2,3);
for i=1:2
    for j=1:3
15 % Find tetrahedron nodes.
        tn = elmn(i,j:j+3);
        abcd = inv([ones(4,1) x(tn) y(tn) z(tn)]) * phi(elmn(i:2:end,j:j+3)');
        E = E - abcd(2:4,:)';
    end
20 end
E = E/6;
```

Listing D.5: findnodeBzfield

```
#include "mex.h"
#include <math.h>

void findNodeBfield (double *Bz,
5 double *Jx, double *Jy, double *Jz,
double *x, double *y, double *z,
double *Tcentx, double *Tcenty, double *Tcentz,
int NodeNum, int TelmnNum, double suspend)
{
10 int i,j,k;
double dist,Xdiff,Ydiff,Zdiff,PercOne;

PercOne = (double) (NodeNum-1)/100;
k = 1;
15 for (i=0; i<NodeNum; i++) {

    *(Bz+i) = 0;

    for (j=0; j<TelmnNum; j++) {
20
        Xdiff = *(x+i) - *(Tcentx+j);
        Ydiff = *(y+i) - *(Tcenty+j);
        Zdiff = *(z+i) - *(Tcentz+j);

25
        dist = sqrt(Xdiff * Xdiff + Ydiff * Ydiff + Zdiff * Zdiff);
        dist = dist * dist * dist;
        /*(distmat+TelmnNum*i+j) = dist;

        *(Bz+i) = *(Bz+i) + ( Ydiff * *(Jx+j) - Xdiff * *(Jy+j) ) / dist;
30
```

```

    }

    *(Bz+i) = *(Bz+i) / 1.0e7;

35    if (i>=PercOne*k & suspend == 0)
    {
        printf("%d%% completed.\n",k);
        mexEvalString("drawnow;"); // to dump string.
        k = k++;
40    }
}

void mexFunction( int nlhs, mxArray *plhs[], int nrhs, const mxArray *prhs[] )
45 {
    double *Bz,*Jx,*Jy,*Jz,*x,*y,*z,*Tcentx,*Tcenty,*Tcentz, suspend;

    int NodeNum, TelmnNum;

50 /* Check for proper number of arguments. */

    if(nrhs!=10)
mexErrMsgTxt("Ten inputs required.");
    if(nlhs!=1)
55 mexErrMsgTxt("One output required.");

    /*Get input vector addresses*/
    Jx = mxGetPr(prhs[0]);
    Jy = mxGetPr(prhs[1]);
60 Jz = mxGetPr(prhs[2]);

    x = mxGetPr(prhs[3]);
    y = mxGetPr(prhs[4]);
    z = mxGetPr(prhs[5]);

65    Tcentx = mxGetPr(prhs[6]);
    Tcenty = mxGetPr(prhs[7]);
    Tcentz = mxGetPr(prhs[8]);

70    suspend = mxGetScalar(prhs[9]);

    /* Find number of tetrahedron elmts. */
    TelmnNum = mxGetM(prhs[0]);

75 /* Find number of nodes. */
    NodeNum = mxGetM(prhs[3]);

    /* Set the output pointer to the output matrix. */
    plhs[0] = mxCreateDoubleMatrix(NodeNum,1,mxREAL);
80 //plhs[1] = mxCreateDoubleMatrix(TelmnNum,NodeNum,mxREAL);

    /* Create a C pointer to a copy of the output matrix. */
    Bz = mxGetPr(plhs[0]);
    //distmat = mxGetPr(plhs[1]);

85 /* Call the C subroutine. */
    findNodeBfield (Bz, Jx, Jy, Jz, x, y, z, Tcentx, Tcenty, Tcentz, NodeNum, TelmnNum, suspend);
}

```

Listing D.6: BzGenerator

```

% BzGenerator
global nodes
global nelmts
global x
5 global y
global z
global elmn
global sigma
%32x32x64 -> 70785
10 ext = '70785';
    ext2 = '_1'; %sigma model ext
    ext3 = '_8'; %save ext

load(['mesh3dtri' ext]);
15 load(['OrgSigma' ext ext2]);

sigma = repmat(OrgSigma',2,1);
sigma = sigma(1:end)';

20 cpn = 4;
CurrDir = {'xDir'; 'yDir'; 'xDir'; 'yDir'};

x_perc=[0.4; 0.4; 0.4; 0.4];
y_perc=[0.4; 0.4; 0.4; 0.4];
25 z_perc=[1; 1; 1; 1];

x_off=[NaN NaN; 0 0; NaN NaN; 0.3 -0.3];
y_off=[ 0 0;NaN NaN;-0.3 0.3;NaN NaN ];
z_off=[ 0 0;0 0;0 0;0 0];
30
for cp = 1:cpn

B = 100 * findNodesForCurrent(CurrDir{cp},x_perc(cp),y_perc(cp),z_perc(cp),x_off(cp,:),y_off(cp,:),z_off(cp,:),ext);
zero_node = find(x==round(een/2) & y==round(een/2) & z==round(zen/2));%node to be taken as zero voltage
35 B(zero_node) = 0;

%-----View current applied nodes in blue and zero node in red.-----
B_all = 100 * findNodesForCurrent(CurrDir{cp},1,1,1,[0 0],[0 0],[0 0],ext);
figure(cp);
40 plot3(x(find(B_all)),y(find(B_all)),z(find(B_all)),'m.',x(find(B)),y(find(B)),z(find(B)),'.');
axis([min(x) max(x) min(y) max(y) min(z) max(z)]);
xlabel('X');ylabel('Y');zlabel('Z');
%-----
end
45 A = formacon;

A(zero_node,:) = 0;
A(zero_node,zero_node) = 1;

50 phi = zeros(nodes,1);
phi = pcg(A,B,1e-6,5000,[],[],phi);

E = findPentaEField(phi);
J = [E(:,1).*sigma E(:,2).*sigma E(:,3).*sigma];
55 load(['CentPntsOfPentahedrons' ext]);

```

```

load(['CentPtsOfHexahedrons' ext]);
Bz = findNodeBzfield(J(:,1),J(:,2),J(:,3),Hcentx,Hcenty,Hcentz,Pcentx,Pcenty,Pcentz,0)/2;

60 switch cp
    case 1
        Bz1=Bz;
        save(['Bz' ext ext3 '_1'], 'Bz1');
        clear Bz1;
65 case 2
        Bz2=Bz;
        save(['Bz' ext ext3 '_2'], 'Bz2');
        clear Bz2;
    case 3
70 Bz3=Bz;
        save(['Bz' ext ext3 '_3'], 'Bz3');
        clear Bz3;
    case 4
75 Bz4=Bz;
        save(['Bz' ext ext3 '_4'], 'Bz4');
        clear Bz4;
    end
    end
load(['Bz' ext ext3 '_1']);
80 load(['Bz' ext ext3 '_2']);
load(['Bz' ext ext3 '_3']);
load(['Bz' ext ext3 '_4']);
Bz=[Bz1 Bz2 Bz3 Bz4];
save(['Bz' ext ext3], 'Bz', 'CurrDir', 'x_perc', 'y_perc', 'z_perc', 'x_off', 'y_off', 'z_off', 'cpn');

```

Listing D.7: findNodesForCurrent

```

function [B] = findNodesForCurrent(CurrDir,x_perc,y_perc,z_perc,x_off,y_off,z_off,ext)
% Finds the nodes on surface constituting the electrodes.

global nodes
5 global nelmts
global x
global y
global z
global elmn
10 load(['CentPtsOfTetrahedrons' ext]);
B=zeros(nodes,1);

switch lower(CurrDir)
    case 'xdir'
15 NormTy = abs((Tcenty-min(Tcenty))/max(Tcenty-min(Tcenty))-0.5+y_off(1) );
        NormTz = abs((Tcentz-min(Tcentz))/max(Tcentz-min(Tcentz))-0.5+z_off(1) );

        n = find(Tcentx == min(Tcentx) & -y_perc/2 <= NormTy & NormTy <= y_perc/2 ...
                & -z_perc/2 <= NormTz & NormTz <= z_perc/2);

20 m = 1 + mod(n-1,3);
        ln = length(n);
        for i=1:ln
            k = elmn(ceil(n(i)/3),m(i):3+m(i));
            l = k(find(x(k) == min(x)));

```

```

25     B(l) = B(l) + 1/(3*ln);
      end

      NormTy = abs((Tcenty-min(Tcenty))/max(Tcenty-min(Tcenty)))-0.5+y_off(2) ;
      NormTz = abs((Tcentz-min(Tcentz))/max(Tcentz-min(Tcentz)))-0.5+z_off(2) ;

30     n = find(Tcentx == max(Tcentx) & -y_perc/2 <= NormTy & NormTy <= y_perc/2 ...
              & -z_perc/2 <= NormTz & NormTz <= z_perc/2);

      m = 1 + mod(n-1,3);
      ln = length(n);
35     for i=1:ln
          k = elmn(ceil(n(i)/3),m(i):3+m(i));
          l = k(find(x(k) == max(x)));
          B(l) = B(l) - 1/(3*ln);
      end

40     case 'ydir'
      NormTx = abs((Tcentx-min(Tcentx))/max(Tcentx-min(Tcentx)))-0.5+x_off(1) ;
      NormTz = abs((Tcentz-min(Tcentz))/max(Tcentz-min(Tcentz)))-0.5+z_off(1) ;

45     n = find(Tcenty == min(Tcenty) & -x_perc/2 <= NormTx & NormTx <= x_perc/2 ...
              & -z_perc/2 <= NormTz & NormTz <= z_perc/2);

      m = 1 + mod(n-1,3);
      ln = length(n);
      for i=1:ln
50         k = elmn(ceil(n(i)/3),m(i):3+m(i));
          j = k(find(y(k) == min(y)));
          B(j) = B(j) + 1/(3*ln);
      end

55     NormTx = abs((Tcentx-min(Tcentx))/max(Tcentx-min(Tcentx)))-0.5+x_off(2) ;
      NormTz = abs((Tcentz-min(Tcentz))/max(Tcentz-min(Tcentz)))-0.5+z_off(2) ;

      n = find(Tcenty == max(Tcenty) & -x_perc/2 <= NormTx & NormTx <= x_perc/2 ...
              & -z_perc/2 <= NormTz & NormTz <= z_perc/2);

60     m = 1 + mod(n-1,3);
      ln = length(n);
      for i=1:ln
          k = elmn(ceil(n(i)/3),m(i):3+m(i));
          j = k(find(y(k) == max(y)));
65         B(j) = B(j) - 1/(3*ln);
      end

      end

      case 'zdir'
      NormTx = abs((Tcentx-min(Tcentx))/max(Tcentx-min(Tcentx)))-0.5+x_off(1) ;
70     NormTy = abs((Tcenty-min(Tcenty))/max(Tcenty-min(Tcenty)))-0.5+y_off(1) ;

      n = find(Tcentz == min(Tcentz) & -x_perc/2 <= NormTx & NormTx <= x_perc/2 ...
              & -y_perc/2 <= NormTy & NormTy <= y_perc/2);

      m = 1 + mod(n-1,3);
75     ln = length(n);
      for i=1:ln
          k = elmn(ceil(n(i)/3),m(i):3+m(i));
          l = k(find(z(k) == min(z)));
          B(l) = B(l) + 1/(3*ln);
80     end

      end

      NormTx = abs((Tcentx-min(Tcentx))/max(Tcentx-min(Tcentx)))-0.5+x_off(2) ;
      NormTy = abs((Tcenty-min(Tcenty))/max(Tcenty-min(Tcenty)))-0.5+y_off(2) ;

```

```

85  n = find(Tcentz == max(Tcentz) & -x_perc/2 <= NormTx & NormTx <= x_perc/2 ...
      & -y_perc/2 <= NormTy & NormTy <= y_perc/2);

    m = 1 + mod(n-1,3);
    ln = length(n);
    for i=1:ln
90     k = elmn(ceil(n(i)/3),m(i):3+m(i));
        l = k(find(z(k) == max(z)));
        B(l) = B(l) - 1/(3*ln);
    end

95  otherwise
    disp('Error! Control the input parameters of method "findNodesForCurrent".');
end

```

Listing D.8: IntEquipotLines

```

% Current density based algorithm for reconstruction by
% integration along equipotential lines.
ext = '70785';
ext2 = '_1'; %for model.
5  ext3 = '_1'; %for sigma.

load(['J' ext ext2]);
load(['OrgSigma' ext ext3]);

10  een = 32;
    zen = 64;

    ren=een; jen=een;
    deltax = 1; deltax = 1;deltaL = 1e-4; %meter

15  Slice = 32;
    s1 = (Slice-1)*een*een+1;
    s2 = Slice*een*een;

20  %NP = [0 0.1 0.2 0.4];
    SNR = [inf 90 60 30];
    for JNL=1:4
        J1 = Jcell{1}(s1:s2,1:2);

25  InjCurr = 100e3; %microA
        J1=InjCurr*1e-2*J1; % 1e-2 : J was generated by 100 microA current. This is a correction factor.

        %-----Add Noise to J-----
        Tc=100e-3;
30  mu_0 = 4*pi*1e-7;
        sigmaJ = sqrt((1/1e-2)^2 + (1/1e-2)^2) / (2*42.6e6*2*pi*mu_0*Tc*SNR(JNL));
        N = randn(size(J1)) * sigmaJ;
        J1 = 1e2*N+J1; % 1e-2 : Unit of J=micA/cm^2 = 1e-2 A/m^2.
        %-----

35  [rhoX rhoY] = meshgrid(0.5:ren-0.5,0.5:ren-0.5);
        rhoX = rhoX/ren;
        rhoY = rhoY/ren;

```

```

[jX jY] = meshgrid(0.5:jen-0.5,0.5:jen-0.5);
40 jX = jX/jen;
    jY = jY/jen;

    % Adjust numer of starting points.
    pl=50;
45 pr=50;
    pm=100;
    spn = pl+pr+pm-2;
    sy = [linspace(0.5,1,pl) ones(1,pm-2) linspace(1,0.5,pr)];
    sx = [zeros(1,pl-1) linspace(0,1,pm) ones(1,pr-1)];
50 % Find equipotential lines passing through sstarting points.
    Lxy = cell(3,spn);
    for i=1:spn
        nxv = sx(i);
        nyv = sy(i);
55     j = 1;
        while 0 <= nxv & nxv <= 1 & 0 <= nyv & nyv <= 1
            Lxy{1,i}(j) = nxv;
            Lxy{2,i}(j) = nyv;

60         [v Ix] = min(min( abs( jX - Lxy{1,i}(j) ) ));
            [v Iy] = min(min( abs( jY - Lxy{2,i}(j) ) ));

            ind = Ix + (Iy-1)*jen;
            Lxy{3,i}(j) = ind;

65         nxv = Lxy{1,i}(j) + deltaL*Jl(ind,2);
            nyv = Lxy{2,i}(j) - deltaL*Jl(ind,1);

            j = j+1;
70     end
    end

    [J1x_x J1x_y] = gradient(reshape(J1(:,1),jen,jen)',1/jen);
    [J1y_x J1y_y] = gradient(reshape(J1(:,2),jen,jen)',1/jen);
75
    J1x_y = J1x_y.';
    J1y_x = J1y_x.';

    r = cell(1,spn);
80 for i=1:spn
        r{i}(1) = log(1/OrgSigma(Lxy{3,i}(1)));
        for j=1:length(Lxy{1,i})-1
            Ind = Lxy{3,i}(j);
            nInd = Lxy{3,i}(j+1);
85         r{i}(j+1) = r{i}(j) + (J1x_y(nInd)-J1y_x(nInd) + J1x_y(Ind)-J1y_x(Ind)) * deltaL / 2 ;
        end
        r{i} = exp(r{i});
    end

90 rho = zeros(ren,ren);
    count = zeros(ren,ren);
    for i=1:spn
        for j=1:length(Lxy{1,i})
            [v Ix] = min(min( abs( rhoX - Lxy{1,i}(j) ) ));
            [v Iy] = min(min( abs( rhoY - Lxy{2,i}(j) ) ));
95         rho(Iy,Ix) = rho(Iy,Ix) + r{1,i}(j);
            count(Iy,Ix) = count(Iy,Ix) + 1;
        end
    end

```

```

    end
end
100 ind = find(count);
    rho(ind) = rho(ind) ./ count(ind);
end

```

Listing D.9: IntCartGridLines

```

% Current density based algorithm for reconstruction by
% integration along Cartesian grid lines.
ext = '70785';
ext2 = '.1'; %for model.
5 ext3 = '.1'; %for sigma.

load(['J' ext ext2]);
load(['OrgSigma' ext ext3]);

10 een = 32;
    zen = 64;

    Slice = 32;
    s1 = (Slice-1)*een*een+1;
15 s2 = Slice*een*een;

ren=een; jen=een; deltax = 1e-2; deltay = 1e-2; %meter

SNR = [inf 90 60 30];
20 for JNL=1:4
    J1 = Jcell{1}(s1:s2,1:2);
    J2 = Jcell{2}(s1:s2,1:2);

    InjCurr = 200e3; %microA
25 J1=InjCurr*1e-2*J1; % 1e-2 : J was generated by 100 microA current. This is a correction factor.
    J2=InjCurr*1e-2*J2; % 1e-2 : J was generated by 100 microA current. This is a correction factor.

    % -----Add Noise to J-----
    Tc=100e-3;
30 mu_0 = 4*pi*1e-7;
    sigmaJ = sqrt((1/1e-2)^2 + (1/1e-2)^2) / (2*42.6e6*2*pi*mu_0*Tc*SNR(JNL));
    N = randn(size(J1)) * sigmaJ;
    J1 = 1e2*N+J1; % 1e-2 : Unit of J=micA/cm^2 = 1e-2 A/m^2.

35 N = randn(size(J2)) * sigmaJ;
    J2 = 1e2*N+J2; % 1e-2 : Unit of J=micA/cm^2 = 1e-2 A/m^2.
    % -----

    [J1x_x J1x_y] = gradient(reshape(J1(:,1), jen, jen)');
40 [J1y_x J1y_y] = gradient(reshape(J1(:,2), jen, jen)');

    J1x_x = J1x_x / deltax;
    J1x_y = J1x_y / deltay;
    J1y_x = J1y_x / deltax;
45 J1y_y = J1y_y / deltay;

    J1x_y = J1x_y.';

```



```

J1y_x = J1y_x.';

50 [J2x_x J2x_y] = gradient(reshape(J2(:,1), jen, jen)');
[J2y_x J2y_y] = gradient(reshape(J2(:,2), jen, jen)');

J2x_x = J2x_x / deltax;
J2x_y = J2x_y / deltax;
55 J2y_x = J2y_x / deltax;
J2y_y = J2y_y / deltax;

J2x_y = J2x_y.';
J2y_x = J2y_x.';

60
% Find GragR.
GradR=zeros(een*een,2);
DET = zeros(een*een,1);
for i=1:een*een
65 INV = inv([J1(i,2) -J1(i,1) ; J2(i,2) -J2(i,1)]);
DET(i) = abs(det([J1(i,2) -J1(i,1) ; J2(i,2) -J2(i,1)]));
if ~isnan(INV)
GradR(i,:) = (INV * [-(J1y_x(i) - J1x_y(i)) ; -(J2y_x(i) - J2x_y(i))])';
end
70 end

% Find image using integral along x direction.
R = zeros(ren*ren,1);
% Give first line for absolute imaging.
75 R(1:een:end) = -log(OrgSigma(s1:een:s2));

for i=1:ren
for j=2:ren
ind = ren*(i-1)+j-1;
80 Nind = ind + 1;
R(Nind) = R(ind) + deltax * (GradR(Nind,1) + GradR(ind,1)) / 2;
end
end
RecRho = exp(R);
85 end

```

Listing D.10: SolveLinEqSys

```

ext = '70785';
ext2 = '.1'; %for model.
ext3 = '.1'; %for sigma.

5 load(['J' ext ext2]);

een = 32;
zen = 64;

10 ren=een; jen=een; deltax = 1e-2; deltax = 1e-2; %meter

Slice = 32;
s1 = (Slice-1)*een*een+1;
s2 = Slice*een*een;

```

```

15 SNR = [inf 90 60 30];
    for JNL=1:4
        J1 = Jcell{1}(s1:s2,1:2);
        J2 = Jcell{2}(s1:s2,1:2);
20 J3 = Jcell{3}(s1:s2,1:2);
        J4 = Jcell{4}(s1:s2,1:2);

        InjCurr = 100e3; %microA
        J1=InjCurr*1e-2*J1; % 1e-2 : J was generated by 100 microA current. This is a correction factor.
25 J2=InjCurr*1e-2*J2; % 1e-2 : J was generated by 100 microA current. This is a correction factor.
        J3=InjCurr*1e-2*J3; % 1e-2 : J was generated by 100 microA current. This is a correction factor.
        J4=InjCurr*1e-2*J4; % 1e-2 : J was generated by 100 microA current. This is a correction factor.

        %-----Add Noise to J-----
30 Tc=100e-3;
        mu_0 = 4*pi*1e-7;
        sigmaJ = sqrt((1/1e-2)^2 + (1/1e-2)^2) / (2*42.6e6*2*pi*mu_0*Tc*SNR(JNL));
        N = randn(size(J1)) * sigmaJ;
        J1 = 1e2*N+J1; % 1e-2 : Unit of J=micA/cm^2 = 1e-2 A/m^2.
35
        N = randn(size(J2)) * sigmaJ;
        J2 = 1e2*N+J2; % 1e-2 : Unit of J=micA/cm^2 = 1e-2 A/m^2.

        N = randn(size(J3)) * sigmaJ;
40 J3 = 1e2*N+J3; % 1e-2 : Unit of J=micA/cm^2 = 1e-2 A/m^2.

        N = randn(size(J4)) * sigmaJ;
        J4 = 1e2*N+J4; % 1e-2 : Unit of J=micA/cm^2 = 1e-2 A/m^2.
        %-----

45 %-----
        [J1x_x J1x_y] = gradient(reshape(J1(:,1), jen, jen)');
        [J1y_x J1y_y] = gradient(reshape(J1(:,2), jen, jen)');

50 J1x_x = J1x_x / deltax;
        J1x_y = J1x_y / deltay;
        J1y_x = J1y_x / deltax;
        J1y_y = J1y_y / deltay;

55 J1x_y = J1x_y.';
        J1y_x = J1y_x.';
        %-----
        [J2x_x J2x_y] = gradient(reshape(J2(:,1), jen, jen)');
        [J2y_x J2y_y] = gradient(reshape(J2(:,2), jen, jen)');
60
        J2x_x = J2x_x / deltax;
        J2x_y = J2x_y / deltay;
        J2y_x = J2y_x / deltax;
        J2y_y = J2y_y / deltay;
65
        J2x_y = J2x_y.';
        J2y_x = J2y_x.';
        %-----
        [J3x_x J3x_y] = gradient(reshape(J3(:,1), jen, jen)');
70 [J3y_x J3y_y] = gradient(reshape(J3(:,2), jen, jen)');

        J3x_x = J3x_x / deltax;
        J3x_y = J3x_y / deltay;

```

```

J3y_x = J3y_x / deltax;
75 J3y_y = J3y_y / deltay;

J3x_y = J3x_y.';
J3y_x = J3y_x.';
%-----
80 [J4x_x J4x_y] = gradient(reshape(J4(:,1), jen, jen)');
[J4y_x J4y_y] = gradient(reshape(J4(:,2), jen, jen)');

J4x_x = J4x_x / deltax;
J4x_y = J4x_y / deltay;
85 J4y_x = J4y_x / deltax;
J4y_y = J4y_y / deltay;

J4x_y = J4x_y.';
J4y_x = J4y_x.';
90 %-----
B1 = -(J1y_x(:) - J1x_y(:));
B2 = -(J2y_x(:) - J2x_y(:));
B3 = -(J3y_x(:) - J3x_y(:));
B4 = -(J4y_x(:) - J4x_y(:));
95
for m=1:4
A = sparse(een*een, een*een);
switch m
case 1
100     J=J1;
case 2
        J=J2;
case 3
        J=J3;
105     case 4
            J=J4;
end

% Using central finite difference.
110 for i=2:een-1
    for j=2:een-1
        k = j+(i-1)*een;
        A(k,k-1) = A(k,k-1) - J(k,2)/(2*deltax);
        A(k,k+1) = A(k,k+1) + J(k,2)/(2*deltax);
115     A(k,k+een) = A(k,k+een) - J(k,1)/(2*deltay);
        A(k,k-een) = A(k,k-een) + J(k,1)/(2*deltay);
    end
end

120 % First row.
i=1;
for j=2:een-1
    k = j+(i-1)*een;
    A(k,k-1) = A(k,k-1) - J(k,2)/(2*deltax);
125     A(k,k+1) = A(k,k+1) + J(k,2)/(2*deltax);
    A(k,k+een) = A(k,k+een) - J(k,1)/deltay;
    A(k,k) = A(k,k) + J(k,1)/deltay;
end

130 % Last row.
i=een;
for j=2:een-1

```

```

    k = j+(i-1)*een;
    A(k,k-1) = A(k,k-1) - J(k,2)/(2*deltax);
135  A(k,k+1) = A(k,k+1) + J(k,2)/(2*deltax);
    A(k,k-een) = A(k,k-een) + J(k,1)/deltay;
    A(k,k) = A(k,k) - J(k,1)/deltay;
end

140  % First column.
    j=1;
    for i=2:een-1
        k = j+(i-1)*een;
        A(k,k+een) = A(k,k+een) - J(k,1)/(2*deltax);
145  A(k,k-een) = A(k,k-een) + J(k,1)/(2*deltax);
        A(k,k+1) = A(k,k+1) + J(k,2)/deltax;
        A(k,k) = A(k,k) - J(k,2)/deltax;
    end

150  % Last column.
    j=een;
    for i=2:een-1
        k = j+(i-1)*een;
        A(k,k+een) = A(k,k+een) - J(k,1)/(2*deltax);
155  A(k,k-een) = A(k,k-een) + J(k,1)/(2*deltax);
        A(k,k-1) = A(k,k-1) - J(k,2)/deltax;
        A(k,k) = A(k,k) + J(k,2)/deltax;
    end

160  i=1; j=1;
    k = j+(i-1)*een;
    A(k,k) = A(k,k) - J(k,2)/deltax + J(k,1)/deltay;
    A(k,k+1) = A(k,k+1) + J(k,2)/deltax;
    A(k,k+een) = A(k,k+een) - J(k,1)/deltay;
165
    i=1; j=een;
    k = j+(i-1)*een;
    A(k,k) = A(k,k) + J(k,2)/deltax + J(k,1)/deltay;
    A(k,k-1) = A(k,k-1) - J(k,2)/deltax;
170  A(k,k+een) = A(k,k+een) - J(k,1)/deltay;

    i=een; j=een;
    k = j+(i-1)*een;
    A(k,k) = A(k,k) + J(k,2)/deltax - J(k,1)/deltay;
175  A(k,k-1) = A(k,k-1) - J(k,2)/deltax;
    A(k,k-een) = A(k,k-een) + J(k,1)/deltay;

    i=een; j=1;
    k = j+(i-1)*een;
180  A(k,k) = A(k,k) - J(k,2)/deltax - J(k,1)/deltay;
    A(k,k+1) = A(k,k+1) + J(k,2)/deltax;
    A(k,k-een) = A(k,k-een) + J(k,1)/deltay;
    switch m
        case 1
185         A1=A;
        case 2
            A2=A;
        case 3
            A3=A;
190         case 4
            A4=A;

```

```

end

end

195 % For absolute imaging.
n=1;
A1(1:n,:) = 0;
for i=1:n
200     A1(i,i) = 1;
end
B1(1:n) = 0;

R = pcg([A1' A2' A3' A4']*[A1 ; A2 ; A3 ; A4],[A1' A2' A3' A4']*[B1 ; B2 ; B3 ; B4],1e-12,2000);
205
RecRho = exp(R);
end

```

Listing D.11: FindwithBz

```

% Magnetic flux density based algorithm for reconstruction by
% solution of linear equation system.
global nodes
global nelmts
5 global x
global y
global z
global elmn

10 % 32x32x64 -> 70785
ext = '70785';
ext2 = '.1'; %for conductivity and electrode model
ext3 = '.1'; %for sigma model

15 load(['mesh3dtri' ext]);
load(['Bz' ext ext2]);
load(['OrgSigma' ext ext3]);

OrgR = -log(OrgSigma);

20
SNR = 90;
InjCurr = 100; %microA
Tc = 100e-3;
N = randn(size(Bz)) / (2*42.6e6*Tc*SNR); %Tesla

25
Mu_zero = 4*pi*1e-7;
Hz = 1e4*(InjCurr*1e-2*1e-4*Bz+N) / Mu_zero; %with noise
% Hz = 1e4*(InjCurr*1e-2*1e-4*Bz) / Mu_zero; %without noise

30 % InjCurr : microA
% 1e-2 : Bz was generated by 100 microA current. This is a correction factor.
% 1e-4 : for units J=micA/cm and dist=cm function findNodeBzfield2 which uses
% Biot-Savart rule gives result in unit 1e-4 Tesla.
% Multiplying Bz with 1e-4 converts its unit to Tesla

35 % Find Laplacian of Hz by using Elemental Bz values.

```

```

    for i=1:cpn
        LapHz(:,i) = reshape(6*del2(reshape(Hz(:,i), een, een, zen)), nelmts/2, 1);
    end
40
    deltax = 1e-2; deltay = 1e-2; %meter
    S=zeros(een*een, cpn);

    % Selected slice for reconstruction.
45    SelSlc=32;
    ss1 = (SelSlc-1)*een*een+1;
    ss2 = SelSlc*een*een;

    R = zeros(size(OrgR));
50    RecRho = zeros(length(OrgR), 5);
    top_slc = 33;
    bot_slc = 31;

    for IterNum=1:5
55        Jcell = cell(1, cpn);
        for i=1:cpn
            Jcell{i} = femsolver(R, InjCurr, CurrDir{i}, x_perc(i), y_perc(i), z_perc(i), x_off(i,:), ...
                y_off(i,:), z_off(i,:), ext, een, zen); %(microA/cm^2)
60        Jcell{i} = 1e-2*Jcell{i}; %(A/m^2)
        end

        for Slice=bot_slc:top_slc
            ofs = (Slice-1)*een*een;
65            s1 = (Slice-1)*een*een+1;
            s2 = Slice*een*een;

            B = LapHz(s1:s2, 1:cpn);
70            Ccon = [];
            % Go on for all current profiles.
            for CurPro=1:cpn
                C = sparse(een*een, een*een);
                J = Jcell{CurPro};
75            % Use central finite difference.
            for i=2:een-1
                for j=2:een-1
                    k = j+(i-1)*een;
                    n = k+ofs;
80                    C(k, k-1) = C(k, k-1) - J(n, 2)/(2*deltax);
                    C(k, k+1) = C(k, k+1) + J(n, 2)/(2*deltax);
                    C(k, k+een) = C(k, k+een) - J(n, 1)/(2*deltay);
                    C(k, k-een) = C(k, k-een) + J(n, 1)/(2*deltay);
                end
85            end
            % First row.
            i=1;
            for j=2:een-1
                k = j+(i-1)*een;
90                n = k+ofs;
                C(k, k-1) = C(k, k-1) - J(n, 2)/(2*deltax);
                C(k, k+1) = C(k, k+1) + J(n, 2)/(2*deltax);
                C(k, k+een) = C(k, k+een) - J(n, 1)/deltay;
                C(k, k) = C(k, k) + J(n, 1)/deltay;
95            end

```

```

% Last row.
i=een;
for j=2:een-1
    k = j+(i-1)*een;
100    n = k+ofs;
        C(k,k-1) = C(k,k-1) - J(n,2)/(2*deltax);
        C(k,k+1) = C(k,k+1) + J(n,2)/(2*deltax);
        C(k,k-een) = C(k,k-een) + J(n,1)/deltay;
        C(k,k) = C(k,k) - J(n,1)/deltay;
105 end
% First column.
j=1;
for i=2:een-1
    k = j+(i-1)*een;
110    n = k+ofs;
        C(k,k+een) = C(k,k+een) - J(n,1)/(2*deltay);
        C(k,k-een) = C(k,k-een) + J(n,1)/(2*deltay);
        C(k,k+1) = C(k,k+1) + J(n,2)/deltax;
        C(k,k) = C(k,k) - J(n,2)/deltax;
115 end
% Last column.
j=een;
for i=2:een-1
    k = j+(i-1)*een;
120    n = k+ofs;
        C(k,k+een) = C(k,k+een) - J(n,1)/(2*deltay);
        C(k,k-een) = C(k,k-een) + J(n,1)/(2*deltay);
        C(k,k-1) = C(k,k-1) - J(n,2)/deltax;
        C(k,k) = C(k,k) + J(n,2)/deltax;
125 end
% C3
i=1; j=1;
k = j+(i-1)*een;
n = k+ofs;
130 C(k,k) = C(k,k) - J(n,2)/deltax + J(n,1)/deltay;
    C(k,k+1) = C(k,k+1) + J(n,2)/deltax;
    C(k,k+een) = C(k,k+een) - J(n,1)/deltay;
% C4
i=1; j=een;
135 k = j+(i-1)*een;
    n = k+ofs;
    C(k,k) = C(k,k) + J(n,2)/deltax + J(n,1)/deltay;
    C(k,k-1) = C(k,k-1) - J(n,2)/deltax;
    C(k,k+een) = C(k,k+een) - J(n,1)/deltay;
140 % C2
    i=een; j=een;
    k = j+(i-1)*een;
    n = k+ofs;
    C(k,k) = C(k,k) + J(n,2)/deltax - J(n,1)/deltay;
145 C(k,k-1) = C(k,k-1) - J(n,2)/deltax;
    C(k,k-een) = C(k,k-een) + J(n,1)/deltay;
% C1
i=een; j=1;
k = j+(i-1)*een;
150 n = k+ofs;
    C(k,k) = C(k,k) - J(n,2)/deltax - J(n,1)/deltay;
    C(k,k+1) = C(k,k+1) + J(n,2)/deltax;
    C(k,k-een) = C(k,k-een) + J(n,1)/deltay;
% Concatenate coefficient matrices for current profiles.

```

```

155 Ccon = [Ccon ; C];
    end
    % For absolute imaging set one pixels value.
    Ccon(1,:) = 0;
    Ccon(1,1) = 1;
160 B(1,1) = OrgR(1);
    % Solve the linear equation system in least-square sense.
    R(s1:s2) = inv(Ccon'*Ccon)*Ccon'*B(:);
    end
    RecRho(s1:s2,IterNum) = exp(R(s1:s2));
165 % Fill outside of RoI by blurred images.
    SliceBlurringFunc(RecRho,bot_slc,top_slc,een,zen);
    end

```

Listing D.12: SensMatAlg

```

% Magnetic flux density based algorithm for Sensitivity Matrix approach.
global nodes
global nelmts
global x
5 global y
global z
global een
global zen
global elmn
10 global sigma
    %32x32x64 -> 70785
    ext = '70785';
    ext2 = '_1';%for conductivity and electrode model
    ext3 = '_1';%for sigma model
15 load(['Bz' ext ext2]);
    load(['OrgSigma' ext ext3]);
    load(['mesh3dtri' ext]);
    %*****
    DelSigma = 0.001;
20 cpn = 4;
    top_slc = 33;
    bot_slc = 31;
    SelSlc = 32;
    %*****
25 sels1 = (SelSlc-1)*een*een+1;
    sels2 = SelSlc*een*een;
    % Sigma0 is 0.02 S/cm.
    sigma0 = 0.02*ones(nelmts/2,1);
    for Iter=1:5
30 for Slice = bot_slc:top_slc
        fs1 = (Slice-1)*een*een+1;
        fs2 = Slice*een*een;
        % Generate sensitivity matrix.
        [S Bz0] = GenSensMat(CurrDir,x_perc,y_perc,z_perc,x_off,y_off,z_off,cpn,ext,sigma0,DelSigma,fs1,fs2);
35 % Find pseudo inverse of sensitivity matrix.
        InvS = pinv(S);
        % Concatenate deltaBz vectors of all current profiles.
        deltaBz = [];
        for i=1:cpn

```



```

40 deltaBz = [deltaBz ; Bz(SR1:SR2,i) - Bz0(:,i)]; %BzMeas-BzCalc
end
% Find deltasigma.
DeltaSigma = DelSigma * InvS * deltaBz;
end
45 % Reconstruct sigma for RoI.
RecSigma = sigma0 + 0.1*DeltaSigma;
% Assign values outside of RoI by blurring.
RecSigma = SliceBlurringFunc(RecSigma,bot_slc,top_slc,een,zen);
% Assign found sigma as sigma0 for next iteration.
50 sigma0 = RecSigma;
end

```

Listing D.13: SliceBlurringFunc

```

function [RecSigma] = SliceBlurringFunc(RecSigma,bot_slc,top_slc,een,zen)
ms = 9;
[f1,f2] = freqspace(ms,'meshgrid');
Hd = ones(ms);
5 r = sqrt(f1.^2 + f2.^2);
Hd( r>0.01) = 0;
mask = fwind1(Hd,hamming(ms));
mask = mask/sum(mask(:));

10 s1=(top_slc-1)*een*een+1; s2=top_slc*een*een;
top_img = reshape(RecSigma(s1:s2),een,een);

top_img_max = max(max(top_img));
top_img_min = min(min(top_img));
15 next_top_img = top_img;
for Slc=top_slc+1:zen
%figure(Slc);imagesc(next_top_img',[top_img_min top_img_max]);axis xy;colorbar;
mean_img = mean(mean(next_top_img))*ones(een+ms-1);
mean_img((ms+1)/2:een+(ms-1)/2,(ms+1)/2:een+(ms-1)/2) = next_top_img;
20 next_top_img = filter2(mask,mean_img,'valid');

s1 = (Slc-1)*een*een+1;
s2 = Slc*een*een;
RecSigma(s1:s2) = reshape(next_top_img,een,een);
25 end

s1=(bot_slc-1)*een*een+1; s2=bot_slc*een*een;
bot_img = reshape(RecSigma(s1:s2),een,een);

30 bot_img_max = max(max(bot_img));
bot_img_min = min(min(bot_img));
next_bot_img = bot_img;
for Slc=bot_slc-1:-1:1
%figure(Slc);imagesc(next_bot_img',[bot_img_min bot_img_max]);axis xy;colorbar;
35 mean_img = mean(mean(next_bot_img))*ones(een+ms-1);
mean_img((ms+1)/2:een+(ms-1)/2,(ms+1)/2:een+(ms-1)/2) = next_bot_img;
next_bot_img = filter2(mask,mean_img,'valid');

s1 = (Slc-1)*een*een+1;
40 s2 = Slc*een*een;

```

```

RecSigma(s1:s2) = reshape(next_bot_img,een,een);
end

```

Listing D.14: GenSensMat

```

function [S,Bz0]=GenSensMat(CurrDir,x_perc,y_perc,z_perc,x_off,y_off,z_off,cpn,ext,sigma0,DelSigma,fs1,fs2)
% Sensitivity Matrix Generator. Automated current profile number feature. generates sens matr for fs1:fs2 ,
global nodes
global nelmts
5 global x
global y
global z
global een
global zen
10 global elmn
global sigma

sigma = repmat(sigma0',2,1);
sigma = sigma(1:end)';
15
S = [];
Bz0 = [];

for cp = 1:cpn
20 B = 100 * findNodesForCurrent(CurrDir{cp},x_perc(cp),y_perc(cp),z_perc(cp),x_off(cp,:),y_off(cp,:),z_off(cp,:),ext);
zero_node = find(x==round(een/2) & y==round(een/2) & z==round(zen/2));%node to be taken as zero voltage
B(zero_node) = 0;
% Solve forward problem.
A = formacon;
25 A(zero_node,:) = 0;
A(zero_node,zero_node) = 1;
phi = zeros(nodes,1);
[Bz phi] = ForProbSolv(A,B,phi,zero_node,ext);

30 SensMat = zeros(nelmts,length(fs1:fs2));
for i=fs1:fs2
% Change sigma a little.
sigma(2*i-1:2*i) = sigma(2*i-1:2*i) + DelSigma;
% Find Bz for selected slice.
35 [Bz1 phi] = ForProbSolv(A,B,phi,zero_node,ext);
% Form one column os Sens Mat.
SensMat(:,i-fs1+1) = Bz1 - Bz;
% Rechange sigma.
sigma(2*i-1:2*i) = sigma(2*i-1:2*i) - DelSigma;
40 end
% Concatenate S matrices.
S = [S ; SensMat];
% Concatenate Bz0.
Bz0 = [Bz0 Bz];
45 end

```

Listing D.15: ForProbSolv

```
function [Bz,phi] = ForProbSolv(A,B,phi,zero_node,ext)
global nodes
global nelmts
global x
5 global y
global z
global elmn
global sigma

10 [phi,FLAG,RELRES,ITER] = pcg(A,B,1e-7,5000,[],[],phi);

E = findPentaEField(phi);
J = [E(:,1).*sigma E(:,2).*sigma E(:,3).*sigma];

15 load(['CentPntsOfPentahedrons' ext]);
load(['CentPntsOfHexahedrons' ext]);
Bz = findNodeBzfield(J(:,1),J(:,2),J(:,3),Hcentx,Hcenty,Hcentz,Pcentx,Pcenty,Pcentz,1)/2;
```
