Circuits and Systems
Mekelweg 4,
2628 CD Delft
The Netherlands
https://cas.tudelft.nl/

CAS-2023-4480163

M.Sc. Thesis

Deep Learning Enhanced Contrast Source Inversion And Phase Error Based Conductivity Correction For Electrical Properties Tomography

Jory Edelman B.Sc.

Abstract

Magnetic resonance electrical properties tomography is a type of quantitative magnetic resonance imaging that aims to reconstruct the conductivity and permittivity of biological tissue. These electrical properties of the tissue can be used to compute the specific absorption rate, to differentiate tumours from healthy tissue and for hyperthermia treatment planning. Several methods to reconstruct these electrical properties exist with different degrees of success. Combining analytical reconstruction methods with deep learning methods is left relatively unexplored in the field of magnetic resonance electrical properties tomography. Hence, this work explores such hybrid methods in which deep learning is embedded in an analytical reconstruction method. A recurrent inference machine is integrated into the iterative reconstruction scheme called Contrast Source Inversion, in an attempt to decrease its high computational load. Additionally, a U-net is trained to correct reconstructed conductivity maps using discrepancies in measured- and reconstructed phase data, which is based on the relation between conductivity and phase in the Helmholtz equation. The recurrent inference machine embedded version of contrast source inversion failed to achieve a desirable reconstruction quality with its current implementation. However, the large amount of potential improvements to its implementation motivates further research into its application before discarding it. The conductivity correction U-net is able to correct conductivity errors as small as 0.13 S/m when used iteratively or 0.05 S/m when used a single time when noiseless data is used. Further research in its capabilities of handling noisy data is required to assess practical usage.

TU Delft
Faculty of Electrical Engineering, Mathematics and Computer Science
Delft University of Technology

Deep Learning Enhanced Contrast Source Inversion And Phase Error Based Conductivity Correction For Electrical Properties Tomography

Exploration Of Physics-Assisted Deep Learning Methodology For Magnetic Resonance Based Electrical Properties Tomography

---

THESIS

submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE

in

ELECTRICAL ENGINEERING

by

Jory Edelman B.Sc.
born in Heinenoord, Netherlands

This work was performed in:

Circuits and Systems Group
Department of Microelectronics
Faculty of Electrical Engineering, Mathematics and Computer Science
Delft University of Technology

Delft University of Technology

Copyright © 2023 Circuits and Systems Group
All rights reserved.

Delft University of Technology
Department of
Microelectronics

The undersigned hereby certify that they have read and recommend to the Faculty of Electrical Engineering, Mathematics and Computer Science for acceptance a thesis entitled **"Deep Learning Enhanced Contrast Source Inversion And Phase Error Based Conductivity Correction For Electrical Properties Tomography"** by **Jory Edelman B.Sc.** in partial fulfillment of the requirements for the degree of **Master of Science**.

Dated: 5 December 2023

Chairman:	dr.ir. R.F. Remis
Advisor:	
Committee Members:	dr. O.A. Krasnov
	prof. C.A.T. van den Berg
	dr. S. Mandija

Abstract

Magnetic resonance electrical properties tomography is a type of quantitative magnetic resonance imaging that aims to reconstruct the conductivity and permittivity of biological tissue. These electrical properties of the tissue can be used to compute the specific absorption rate, to differentiate tumours from healthy tissue and for hyperthermia treatment planning. Several methods to reconstruct these electrical properties exist with different degrees of success. Combining analytical reconstruction methods with deep learning methods is left relatively unexplored in the field of magnetic resonance electrical properties tomography. Hence, this work explores such hybrid methods in which deep learning is embedded in an analytical reconstruction method. A recurrent inference machine is integrated into the iterative reconstruction scheme called Contrast Source Inversion, in an attempt to decrease its high computational load. Additionally, a U-net is trained to correct reconstructed conductivity maps using discrepancies in measured- and reconstructed phase data, which is based on the relation between conductivity and phase in the Helmholtz equation. The recurrent inference machine embedded version of contrast source inversion failed to achieve a desirable reconstruction quality with its current implementation. However, the large amount of potential improvements to its implementation motivates further research into its application before discarding it. The conductivity correction U-net is able to correct conductivity errors as small as 0.13 S/m when used iteratively or 0.05 S/m when used a single time when noiseless data is used. Further research in its capabilities of handling noisy data is required to assess practical usage.

Acknowledgments

I would like to thank Rob and Stefano for their guidance and assistance throughout my thesis project. Their insightful suggestions have helped me push forward when results were far from desirable and when I got stuck in debugging my code. I would also like to thank my friends and family for their continued support, for giving me an outlet when things did not look great and for celebrating with me when things looked up. I am thankful for the opportunity to work on this project with the resources provided by both the TU Delft and the UMC Utrecht. Furthermore, working on this thesis has given me more insight into my strong and weak points. It has been a journey that has taught me much.

Jory Edelman B.Sc.
Delft, The Netherlands
5 December 2023

Contents

List of Figures

List of Tables

Introduction 1

Examining the internal state of a patient's body for diagnoses, treatment planning or research has become common practice in modern medicine. Medical imaging modalities allow professionals in the medical field to assess visual presentations of the insides of their patients. An assortment of different medical imaging modalities exist, such as Computed Tomography (CT), Magnetic Resonance Imaging (MRI), Positron Emission Tomography (PET) and X-ray. Each modality has its advantages and disadvantages and provides us with different types of information.

MRI features a broad spectrum of imaging modalities, from imaging soft tissue with T1- or T2-weighted MRI[1] to measuring brain activity with functional MRI (fMRI) [2]. An MRI scanner leverages magnetic fields to influence the magnetic orientation of protons in the subject. These are typically hydrogen protons as they have an abundant presence in the human body. These protons resemble a spinning top with a magnetic north- and south pole, gyrating along an axis. In regular circumstances these axes are aligned randomly, therefore yielding a net magnetic moment of approximately zero. Applying a strong static magnetic field to the subject aligns the axis of the protons. The static magnetic field of an MRI scanner is typically 1.5T, 3T or 7T and is referred to as the $\mathbf{B}_0$ field. Aligned with the magnetic field, the protons gyrate around their axis at a frequency dependent on the strength of the magnetic field and the element used. This frequency is called the Larmor frequency. A radio frequency (RF) magnetic field, called the B_1 field, provides a pulse of energy to the protons, which flips their alignment. The B_1 field is applied at a frequency equal to the Larmor frequency as the resonance ensures the most efficient energy transfer. Additional coils apply gradients to the magnetic field, this causes only select locations to resonate. As the protons restore their alignment to the B_0 field, they emit a RF signal which is measured. Hence, using the gradient coils to apply resonance to a single spot specifies where the measured signal comes from. Methodically alternating the resonance location allows the imaging of the region of interest of the subject.

The electrical properties (EPs) of the imaged subject create distortions in the measured RF signal [3]. Typically, these RF signal distortions are measured using mapping techniques [4] to compensate for these RF signal distortions. Alternatively, the distortions in the RF signal can be used to reconstruct the EPs of the subject's tissue. The MRI modality that is concerned with reconstructing these EPs from MRI measurements is referred to as MR-based Electrical Properties Tomography (MR-EPT or EPT).

The EPs refer to the conductivity (σ) and permittivity (ϵ) of tissue. Conductivity is a measure of how well a material conducts an electric current, while permittivity is a measure of how much a dielectric medium affects and is affected by an electric field. The EPs of tissue depend on the frequency of the electromagnetic (EM) waves, the ionic concentrations and the permeability of cellular membranes [5]. Subjecting biological tissue to the EM waves of an MRI scanner at MRI frequencies (64-300 MHz) causes this dependency to mostly be dominated by ionic concentrations [5].

The EPs can be used in a multitude of clinical applications. EPs are essential when trying to determine the Specific Absorption Rate of tissue [6]. The SAR is a measure of how much power is transferred to a mass of tissue by a RF field. Heating of tissue, induced by RF fields, can be limited by leveraging knowledge of that tissue's SAR. Such heating is a major safety concern in high- and ultrahigh-field MRI [7]. The EPs can be used as a biomarker for tumours [8, 9], since the EPs of tumours tend to be very different from their surrounding benign tissue [10, 11]. Additionally, knowledge of the EPs can be leveraged for hyperthermia treatment planning of tumours [12, 13].

Reconstruction methods for MR-EPT can be subdivided into 3 categories. Firstly, direct methods, which compute the EPs directly from the measured RF field. Secondly, forward methods, which iteratively improve on an initial guess of the EPs based on discrepancies between the measured field and the field computed from the initial- or updated EPs. Lastly, deep learning methods, which leverage deep learning to reconstruct the EPs. These methodologies are illustrated in Figure 1.1 and will be explained further in the following sections.

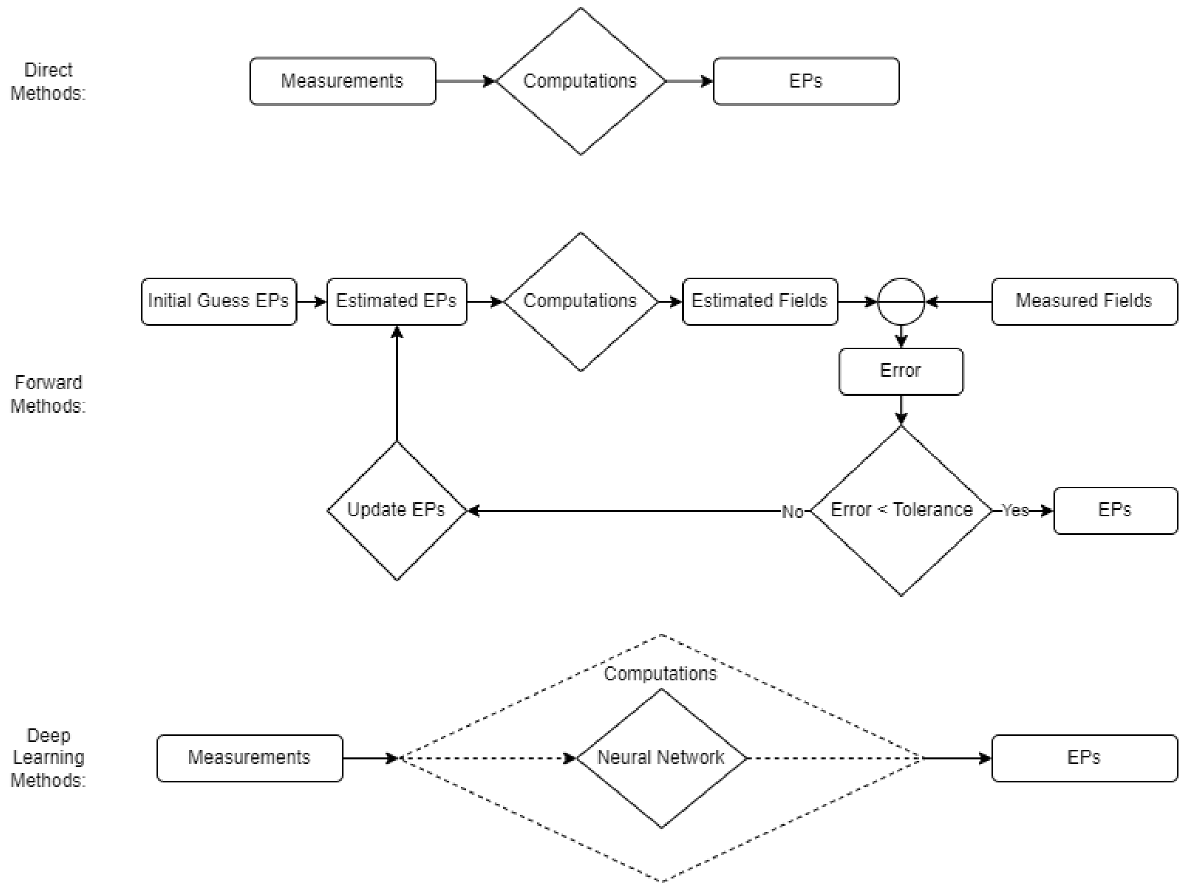


Figure 1.1: A generalised schematic overview of direct-, forward- and deep learning EPT methods.

1.1 Direct EPT Methods

A wide range of different MR-EPT methods have been developed. Yet few have reached the stage of being tested in clinical studies. A method that has reached this point [14], sometimes referred to as Simplified Helmholtz EPT (SH-EPT), is characterised by its simplicity, a low computational time and short acquisition times [14]. SH-EPT's simplicity makes it easy to implement, while its low computational time prevents excessive waiting for results. Since both cost and patient comfort are highly correlated to the duration of an MRI scan [15], having a short acquisition time increases cost-effectiveness and reduces discomfort. The short acquisition time of SH-EPT originates from only requiring either the phase or magnitude of the measured RF field depending on whether the conductivity or permittivity is to be reconstructed. Since SH-EPT assumes that changes in the phase are primarily caused by the conductivity, while it assumes that changes in the magnitude are primarily caused by the permittivity. Hence the reconstructed conductivity does not take the magnitude into account and the reconstructed permittivity does not take the phase into account. The validity of this assumption is observed to be reliant on the ratio between the conductivity and permittivity multiplied by the angular frequency [16]. The ratio between conductivity and permittivity is tissue-dependent, meaning that the extent to which the assumption will hold differs per type of tissue imaged. The frequency dependency means that the validity of the assumption is reliant on the MRI's field strength of its B_0 field, due to its direct relation to the Larmor frequency. The extent to which the assumption holds influences the accuracy of the EP's reconstruction. For example, overestimation of conductivity reconstruction has been observed when using 7T MRI scanners when compared to 1.5T MRI-scanner results [17].

Note that SH-EPT's acquisition time is short if only one of the two EPs is reconstructed. When both EPs are reconstructed, the acquisition time becomes similar to a method called Helmholtz EPT (H-EPT). Helmholtz EPT [18] is the method on which SH-EPT is based. It predates SH-EPT in the sense that its solution is based on the complete RF field. Hence not neglecting the magnitude and phase component when reconstructing conductivity and permittivity, respectively. Compared to SH-EPT, H-EPT has less stringent validity conditions at the expense of requiring more information. Meaning longer acquisition times for acquiring both magnitude and phase information, while not introducing additional inaccuracies because of SH-EPT's additional assumption.

Both SH-EPT and H-EPT assume that the EPs being imaged are locally homogeneous. The simplicity of these methods is largely attributed to this assumption [14]. However, the use of this assumption gives rise to significant errors at the boundaries of different tissue types. Since different tissue types typically have different EPs, thereby violating the local homogeneity assumption. Figure 2 in the work of Mandija et al. [19] displays a great example of the significance of boundary errors in H-EPT. Boundary errors are especially prevalent when imaging complex tissue structures such as the human brain, where grey matter, white matter and Cerebral Spinal Fluids (CSF) intertwine. To mitigate the boundary errors to an extent, the different tissue types could be segmented.

Alternatively, Local Maxwell Tomography (LMT) [20] uses an array of coils to measure the RF field through which a system of equations is constructed. If a sufficiently large amount of RF field maps have been acquired (e.g. 16), then the locally homogeneous assumption is not required [21]. Due to the need to acquire a higher amount of RF field maps with the coil array, LMT's acquisition time is significantly higher compared to H-EPT and SH-EPT. As aforementioned, longer acquisition times lead to increased cost and patient discomfort. Furthermore, such an array of coils is typically not readily available in clinical settings [14].

Similar to LMT, Modified Dual-Excitation EPT (MDE-EPT) avoids the locally homogeneous assumption by acquiring multiple RF field maps with a coil array. MDE-EPT requires a minimum of 5 RF field maps [22], which is typically less than what LMT requires. Hence, comparatively shortening acquisition time. MDE-EPT assumes that there are no variations in the longitudinal component of the RF magnetic field. Within the MRI setting, the longitudinal direction is defined as the direction in which the static magnetic field B_0 travels in. Errors in the reconstruction have been observed to increase the further away one moves from the central slice in the longitudinal direction, as a consequence of this assumption [22]. Hence, when compared to LMT, MDE-EPT features shorter acquisition times at the cost of increased errors outside of central regions in the longitudinal direction.

The aforementioned methods all make use of spatial differential operations to reconstruct the EPs directly from RF field data. Hence, as a group, they will be referred to as direct methods. The usage of spatial derivatives makes these direct methods rather sensitive to noise and other spatial fluctuations [19]. As noise can feature rapid changes, its gradients tend to be large and differentiation will therefore amplify the noise. This is further aggrandised by the finite differential kernels used to approximate the differential operation. Using large differential kernels decreases the sensitivity to noise but also exacerbates any boundary errors. Since a larger kernel size takes into account more spatial information and therefore has an increased likelihood of encountering a tissue boundary. Instead of adjusting kernel sizes, multiple different filtering or denoising methods can be used to mitigate some of the noise's impact, at the cost of additional computation.

1.2 Forward EPT Methods

Opposed to the direct methods which directly compute EPs from measured RF data, another set of methods starts with an initial guess of EPs. This initial guess is typically an average value typical to the tissue type inherent to the imaged location. From this initial guess, the corresponding magnetic (and often electric) fields are computed. These computed fields are compared with measured RF field data. Depending on the discrepancy of the measured and computed fields, the EPs are updated. This process repeats iteratively until the measured and computed fields are in agreement within a specified tolerance range. These methods implement a forward model, from EPs to EM fields and are therefore referred to as forward methods.

Inversely to the direct methods, forward methods leverage integral operations rather than differential ones. Using integration is inherently more noise robust than using differentiation [14]. Integrating or summing a sufficiently large amount of noise cancels each other out when the noise is zero mean. For zero mean noise, one expects a similar amount of positive- and negative magnitudes, adding those up nets zero. Note that this reasoning assumes that the noise has equal probabilities of being positive or negative for a similar magnitude. Integrating noise with a nonzero mean still smooths the noise but introduces a bias towards the noise's mean in the reconstruction.

A limiting factor to the forward methods originates from assuming that the RF EM fields in the MRI scanner when no object/subject is present, are known. These EM fields are referred to as the incident fields. However, these incident fields cannot be measured directly [14]. Instead, the incident fields are simulated based on the known RF coil geometry or referenced from a scan with an object with known EPs. These solutions still introduce inaccuracies to the reconstructions, as the presence of an object or subject loads the coils changing the incident fields [23]. Furthermore, when working with simulated incident fields, slight deviations in simulated and actual coil location or geometry have been observed to lead to reconstruction inaccuracies [24].

The Variational Born Iterative Method EPT (VBIM-EPT) [25, 26] is a forward method which expresses the difference between the measured magnetic RF field and the from EPs derived counterpart, as a residual. The magnetic RF field residual is a function of the difference in the guessed EPs and the actual EPs; the EPs residual. The initial guess of the EPs is updated by adding the residual of the EPs. Based on these updated EPs, the updated RF EM fields are computed. This process is repeated iteratively until the magnetic RF field residual is smaller than a user-defined threshold. The computation of the residual EPs from the residual magnetic RF field is an inverse problem which has to be numerically solved. Solving such an inverse problem is computationally heavy, which is especially prohibitive considering that this computation is performed in every iteration. Hence, the computational load of VBIM-EPT is significantly high.

Global Maxwell Tomography (GMT) [27, 28, 29] poses the forward problem as an optimisation problem. A cost function is defined based on the discrepancy between the measured RF magnetic field and the RF magnetic field computed from the (guessed) EPs. By taking the gradient of the cost function with respect to the EPs, an update for the EPs is formulated. The computed field resulting from the updated EPs, is then used to update the cost function. Repeating this process until the cost function falls within a specified tolerance range effectively minimises the cost function, with the EPs acting as the optimisation variable. GMT reformulates the relation between the EPs and the computed field used in VBIM-EPT into a forward problem and thereby eliminates the computationally heavy inverse problem. The computational load of GMT is still very high relative to direct methods [28] due to the high amount of iterations necessary for optimising the EPs.

Similar to GMT, Contrast Source Inversion EPT (CSI-EPT) [30, 23] reformulates the inversion problem, circumventing the computationally heavy inverse problem. CSI considers the distortions that the EPs apply to the RF field to have originated from a source. This source is introduced as a function of the EPs and the total electric field. Whereas the electric field and magnetic RF field are approximated as a function of this source. CSI also poses the forward problem as an optimisation problem and solves it iteratively. Its cost function contains two parts; each part assigns a cost to a discrepancy in the approximation of the electric field and magnetic RF field respectively. This cost function is minimised by optimising both the source and the EPs. Updating both variables is done by fixing the value of one while computing the update of the other. The source is updated first and the EPs second. CSI, like the other forward methods, also suffers from a high computational load caused by the high amount of iterations required for convergence.

1.3 Deep Learning EPT Methods

Like in many scientific fields, the potential of using Deep Learning (DL) to solve ill-posed problems is also being explored in the field of MR-EPT. Deep Learning is a subset of machine learning, the field of algorithms that learn from experience[31]. Deep Learning then refers to machine learning algorithms with multiple layers with each their own transformation. Together, these layers form a network referred to as a Neural Network (NN). These neural networks are trained from end-to-end, the layers jointly learn parameters from an input-output pair. The output of the input-output pair refers to the 'true' output that the NN should ideally compute from the corresponding input. Hence, the output of the input-output pair is referred to as the ground truth. When training, the NN computes an output based on the provided input, which is then compared to the ground truth. This comparison is performed mathematically by a function referred to as a loss function or objective function. Such an objective function is a mathematical definition of what constitutes an improvement in the learning process of the NN; objective functions are typically minimised.

Deep learning methods for EPT are usually trained with simulated data, as in-vivo data usually lacks ground truth data. Reconstructing EPs from data bearing little resemblance to the training data is bound to feature artefacts. This may occur when in-vivo data [32] is presented as an input when the model was trained with noiseless, simulated data. Alternatively, evaluating a model with inputs that feature tumours, while tumours were excluded from training could lead to unexpected behaviour [33]. More generally, such unexpected behaviour can occur whenever a NN is presented with unseen data; data that has not been included in training. Hence, in the field of deep learning we want neural networks to learn the general patterns that allow us to make accurate predictions on new unseen data [31].

To categorise the way DL is applied to EPT methodology, the definitions of direct deep learning, learning-assisted objective function deep learning and physics-assisted deep learning introduced by Chen et al. [34] will be applied to DL EPT methods. These categories are specifically introduced for inverse scattering problems (ISP). ISPs are a subfield of wavefield problems in which a domain of interest is illuminated with wavefields in an attempt to learn the characteristics of a penetrable object within the domain. The receivers in an ISP are outside of the domain of interest and their received wavefield is considered known. The wavefield produced by the source is similarly known, leaving the media of the object and its surroundings as the unknown quantity. MRI-EPT is also considered an ISP, with the subject the penetrable object and the source and receivers embodied by the coils of the MRI scanner.

1.3.1 Direct Deep Learning

The direct deep learning category contains methods in which a desired output is to be directly learned from the measurements by a NN. For MR-EPT this constitutes a NN learning how to reconstruct the EPs from the measured RF magnetic field. This category disregards any knowledge of the wave physics that governs the interaction between the wavefields and the subject. These physics have to be implicitly learned by the NN, which essentially wastes computational power on known information. Although learning these implicitly based on data, has the potential to better deal with deviations caused by any noise, biases or assumption-based artefacts that might have otherwise occurred. On the flip side, training the NN on data with a bias that is not inherently present in measured data from other sources, might instead degrade the quality of reconstructions. Methods within this category require relatively little knowledge of the user on the ISP [34]. Direct deep learning methods have the added benefit of having less stringent requirements of input data such as not having to assume knowledge of the incident fields.

Mandija et al. [35] proposed a direct deep learning EPT method in which the EPs are directly learned from the measured magnetic RF field using clinically available settings. They reported more noise-robust results than traditional MR-EPT methods. Additionally, it was observed that most predictive features for the reconstruction of conductivity originate from the phase maps, which is congruent with the phase-based conductivity reconstruction of SH-EPT.

Similarly, Gavazzi et al. [36] applied direct deep learning to reconstruct EPs in the pelvic area. The network was trained in two configurations, one with a RF magnetic field as input and one with only the corresponding phase as input. They reported comparable results for both configurations, once again providing support for SH-EPT's assumption of variations in phase being predominantly conductivity-related. When tested in vivo, the median values of the DL method were comparable to those of H-EPT used in comparison, only the precision of the DL method was higher [36].

1.3.2 Learning-Assisted Objective Function Deep Learning

Learning-assisted objective function deep learning retains the framework of solving an ISP by optimising the objective function iteratively. The difference lies in replacing a part of the framework with deep learning. In the field of EPT, this would refer to using a forward method with a NN replacing a part of the process. An example of such a method would be to have a NN generate an initial guess for the iterative solver. Initial guesses provided to iterative solvers are typically homogeneous guesses or guesses based on known tissue geometries or segmentations. Assuming that a NN provides an improved initial guess, the number of iterations taken by the solver will be reduced, in turn reducing the high computational load that is typical to forward methods.

Leijsen et al.[33] proposed a learning-assisted objective function deep learning method for EPT. They generate an initial guess with a NN, which is used by CSI-EPT to reconstruct the EPs. They report an improvement in quality and accuracy over standard CSI-EPT. Additionally, the data consistency that CSI-EPT provides, relaxes the need for exhaustive data sets to train the NN.

An alternate example of learning-assisted objective function deep learning is shown in a microwave imaging reconstruction method proposed by Guo et al.[37]. Here the average descent direction of the gradient descent is learned by a NN. Hence, a different part of the framework is assisted by a NN compared to the previous example.

1.3.3 Physics Assisted Deep Learning

In physics-assisted deep learning, domain knowledge is included in the deep learning. This is either done by incorporating it into the architecture of the NN or by including it in the input. When included in the NN architecture, this could entail using sparse connections in the NN when solving sparse problems. Including domain knowledge in the input of the NN refers to having the input of the NN being transformed from the measurement domain to the output domain. Meaning that for EPT the input to the network is in the form of an EP estimate instead of the measured RF magnetic field.

An example of physics-assisted deep learning is shown by Khoshdel et al. [38]. They proposed a method in which CSI is applied to ultrasound measurements to provide a rough reconstruction of the permittivity in breast imaging. This reconstruction is then fed to a NN which attempts to reconstruct the true permittivity. Hence CSI is used to impart domain knowledge into the NN's input. They report improvement in the performance of tumour segmentation and a reduction in reconstruction errors when compared to using CSI only.

1.4 Project Goal

As aforementioned, deep learning methods for MR-EPT are mostly restricted to the direct deep learning category. Even though the direct deep learning methods are the most straightforward, they waste computational power on learning the already known wave physics underlying the problem. Unlike in other medical imaging fields, physics-assisted deep learning for MR-EPT has not yet been attempted. In other medical image construction modalities physics assisted deep learning has mostly been implemented by embedding a NN in an iterative reconstruction scheme [39, 40, 41, 42]. These methodologies yielded improved generalisation to unseen data in training, while also decreasing the computational time. Similarly, MR-EPT forward methods could potentially benefit from similar improvements when combined with DL. In particular, CSI has already been combined with DL in such a manner in the field of ultrasound [38].

Direct EPT methods could potentially be used in a physics-assisted deep learning scheme as well. The direct method would be used to provide an initial reconstruction of EPs, which are then to be used as input to a NN. The NN could then potentially resolve issues typical to direct methods such as boundary errors.

This project aims to explore two different avenues of physics-assisted deep learning for MR-EPT in which both a direct- and a forward method are combined with deep learning to offset their disadvantages.

The forward method 3D-CSI will be combined with deep learning, with the goal of speeding up the computational time significantly. Additionally, potential methods of circumventing the requirement for known incident fields will be investigated.

A deep learning method will be designed to improve reconstructed estimates of conductivity, based on the working principles of SH-EPT.

1.5 Outline

In Chapter 2 mathematical background information is provided on two important assumptions in MR-EPT and a mathematical description is given of SH-EPT and CSI-EPT. The aim of Chapter 2 is to provide the reader with sufficient background knowledge to understand the subsequent chapters.

In Chapter 3 programming methodologies and implementations are reported. The chapter discusses the porting of 3D-CSI-EPT to Python, structure and code design choices and the debugging process.

In Chapter 4 potential methods of combining DL with CSI are discussed. The chapter first introduces relevant neural network types, afterwards deep learning strategies for CSI are discussed and lastly, potential topologies for combining DL with CSI are proposed.

In Chapter 5 the implementation and results of the hybrid DL-CSI method are reported. Covering the neural network design implementation, training strategies, results and neural network variations.

In Chapter 6 a hybrid method for improvement of conductivity reconstructions based on phase measurements is proposed. The method combines DL with the working principles of the direct MR-EPT method SH-EPT. Chapter 6 covers the methodology, neural network design, training strategies and results of the proposed hybrid method.

In Chapter 7 the results of both hybrid methods are discussed. Covering potential improvements and future work.

Chapter 7.4 concludes this work.

2 Mathematical Background

This chapter provides the reader with explanations, mathematical notations and derivations of assumptions and EPT methods directly relevant to the project. First, a mathematical representation of the electromagnetic fields for MRI is provided and the transceive phase assumption (TPA) is introduced. This is followed by the derivation of SH-EPT from Maxwell's equations, including how the local homogeneity assumption influences the solution mathematically and how the magnitude- and phase dominance assumptions segregate SH-EPT from H-EPT. The following section introduces the mathematical representation of the incident fields and discusses a method to simulate them. The chapter is concluded by an abbreviated derivation of CSI for EPT.

2.1 Field Representation and Transceive Phase Assumption

The following field representations follow the methodologies and notations of [14]. The earlier introduced RF magnetic field is expressed as

$$\mathbf{B}_1 = |\mathbf{B}_1|e^{j\phi}, \tag{2.1}$$

with j denoting the imaginary unit, the $|\cdot|$ operator denoting the amplitude and ϕ denoting the field's phase. The RF magnetic field is typically described by its phasor, which we introduce as

$$\mathbf{B}_1(\mathbf{x}, t) = \mathrm{Re}[\tilde{\mathbf{B}}_1(\mathbf{x}, j\omega)e^{j\omega t}] \tag{2.2}$$

[14]. Here $\mathbf{x}$ denotes the position vector, t time and ω the angular frequency. The $\mathrm{Re}[\cdot]$ operator takes the real part of a complex number. The time convention e^{jwt} is used.

In this work the field quantities are considered to be in phasor notation and for convenience the tilde will be dropped. Meaning that $\mathbf{B}_1$ from here on represents $\tilde{\mathbf{B}}_1(\mathbf{x}, j\omega)$.

Note that the RF magnetic field is expressed by a bold letter, indicating that it is a vector quantity, which can be broken down into its x, y and z components as:

$$\mathbf{B}_1 = B_{1;x}\mathbf{i}_x + B_{1;y}\mathbf{i}_y + B_{1;z}\mathbf{i}_z. \tag{2.3}$$

Where $B_{1;x}$, $B_{1;y}$ and $B_{1;z}$ denote the scalar x-, y- and z field components, respectively. The $\mathbf{i_x}$, $\mathbf{i_y}$ and $\mathbf{i_z}$ denote the unit vectors in the x-, y- and z-direction, respectively.

The static magnetic $\mathbf{B}_0$ field in the MRI scanner is considered to be aligned with the z-axis. Whereas we consider the RF magnetic $\mathbf{B}_1$ to solely consist of x- and y-components. Hence, the $B_{1;z}$ component in Equation 2.3 can be eliminated as it equates to 0. The $\mathbf{B}_1$ field can be further split into two circularly polarised field components

$$\mathbf{B}_1 = \mathbf{B}_1^+ + \mathbf{B}_1^-, \tag{2.4}$$

with

$$\mathbf{B}_1^+ = B_1^+(\mathbf{i}_x - j\mathbf{i}_y) \tag{2.5}$$

and

$$\mathbf{B}_1^- = \left[B_1^-\right]^* (\mathbf{i}_x + j\mathbf{i}_y). \tag{2.6}$$

These scalar field components are referred to as the transmit- (B_1^+) and receive (B_1^-) RF magnetic fields. The transmit and receive field components are defined as

$$B_1^+ = \frac{B_{1;x} + jB_{1;y}}{2} \tag{2.7}$$

and

$$B_1^- = (\frac{B_{1;x} - jB_{1;y}}{2})^*. \tag{2.8}$$

Here * denotes the complex conjugate. Alternatively, we introduce the transmit and receive unit vectors as

$$\mathbf{i}^+ = \frac{1}{2}(\mathbf{i}_x + j\mathbf{i}_y) \tag{2.9}$$

and

$$\mathbf{i}^- = \frac{1}{2}(\mathbf{i}_x - j\mathbf{i}_y)^*, \tag{2.10}$$

respectively. Using these vectors, the transmit and receive field relate to $\mathbf{B}_1$ following

$$B_1^+ = \mathbf{i}^+ \cdot \mathbf{B}_1 \tag{2.11}$$

and

$$B_1^{-;*} = \mathbf{i}^- \cdot \mathbf{B}_1. \tag{2.12}$$

In the time domain, these field components trace a circle and are hence referred to as circularly polarised. The transmit and receive fields travel along this circle in opposite directions. The direction in which the circularly polarised fields travel, is determined by the direction which the static $\mathbf{B}_0$ field travels in. Following the most frequently used convention, the $\mathbf{B}_0$ field's direction is defined as the positive z-direction. This results in the transmit field and receive field being defined as right- and left-hand circularly polarised, respectively. The transmit field rotates in the same direction as protons gyrating along the $\mathbf{B}_0$ field's alignment. Hence, the transmit field can efficiently transfer energy to flip the protons' orientation.

The quantities measured with the MRI scanner are limited to the magnitude of the transmit field ($|B_1^+|$) and the combined phase of the transmit and receive fields ($\phi^{\pm}$), referred to as the transceive phase. The transceive phase is a superposition of the trasmit and receive phases

$$\phi^{\pm} = \phi^+ + \phi^-. \tag{2.13}$$

Typically, the MR-EPT methods require both transmit magnitude and transmit phase in order to use the complex transmit field ($B_1^+ = |B_1^+|e^{j\phi^+}$) as input. Most often the gap in knowledge between the transceive and transmit phase, is bridged by the transceive phase assumption (TPA). The TPA assumes that the transmit- and receive phase are equal in contribution, therefore deriving the transmit phase as

$$\phi^+ = \frac{\phi^\pm}{2}. \tag{2.14}$$

The TPA is, however, only valid for low field strength and symmetrical objects [43], introducing errors otherwise. Circumventing the use of the TPA can be achieved by using multi-element arrays to acquire relative phases. However, these arrays are not widely available and prolong acquisition time significantly [14].

2.2 Simplified Helmholtz Electrical Properties Tomogaphy

Simplified Helmholtz EPT (SH-EPT) is a direct approach with a very low amount of complexity. It has a low computation time and is easily implementable [14]. SH-EPT relates conductivity and permittivity to the measured phase and magnitude of the B_1^+ field respectively. This reduces the need for obtaining both phase and magnitude if only a single EP is required instead of both.

As a direct approach, SH-EPT reconstructs the EPs directly from the measured B_1^+ data. To understand how the EPs interact with the B_1^+ field, its relation is derived from Maxwell's equations for time-harmonic fields following the methodology presented in [14]. Maxwell's equations for time-harmonic fields are introduced as

$$-\nabla \times \mathbf{H} + \eta \mathbf{E} = -\mathbf{J}^{ext}, \tag{2.15}$$

$$\nabla \times \mathbf{E} + \zeta \mathbf{H} = \mathbf{0}. \tag{2.16}$$

Here, $\mathbf{H}$ denotes the magnetic field strength. The admittance and impedance of the medium are denoted by $\eta = \sigma + j\omega\epsilon$ and $\zeta = j\omega\mu$, respectively. Of which, μ represents the permeability. The spatial variations of the permeability of biological tissue tend to be small [23], therefore μ is assumed constant and equal to the permeability of vacuum, $\mu = \mu_0$. Due to the B_1^+ that the MRI scanner measures, we are more interested in the magnetic flux density $\mathbf{B}$ than the magnetic field strength $\mathbf{H}$. Therefore, the $\mathbf{H}$ is substituted by $\mathbf{B}$ using the constitutive relation $\mathbf{B} = \mu_0 \mathbf{H}$. For convenience, the magnetic flux density will be referred to as the magnetic field. The external current density distribution $\mathbf{J}^{ext}$ in MR-EPT, is located on the RF-coil of the MRI scanner, which is used to generate EM fields. The coil is located outside the imaging domain, allowing us to equate $\mathbf{J}^{ext}$ to $\mathbf{0}$. The Maxwell equations in the imaging domain for biological tissue then equate to

$$-\nabla \times \mathbf{B} + \mu_0 \eta \mathbf{E} = \mathbf{0}, \tag{2.17}$$

$$\nabla \times \mathbf{E} + j\omega \mathbf{B} = \mathbf{0}. \tag{2.18}$$

Since the quantity measured by an MRI scanner consists of a magnetic field component, the electric field component is eliminated from the system of equations. To achieve this, the following two curl identities are leveraged

$$\nabla \times \nabla \times \mathbf{F} = \nabla\nabla \cdot \mathbf{F} - \nabla^2 \mathbf{F}, \tag{2.19}$$

$$\nabla \times (v\mathbf{F}) = \nabla v \times \mathbf{F} + v\nabla \times \mathbf{F}, \tag{2.20}$$

where $\mathbf{F}$ and v represent a vector field and scalar function, respectively. The curl of the first Maxwell Equation 2.17 is taken to make use of these identities.

$$-\nabla \times \nabla \times \mathbf{B} + \mu_0 \nabla \times (\eta \mathbf{E}) = 0. \tag{2.21}$$

According to the curl identities, the additive parts of Equation 2.21 can now be rewritten as

$$\nabla \times \nabla \times \mathbf{B} = \nabla\nabla \cdot \mathbf{B} - \nabla^2 \mathbf{B} \tag{2.22}$$

and

$$\nabla \times (\eta \mathbf{E}) = \nabla \eta \times \mathbf{E} + \eta \nabla \times \mathbf{E}. \tag{2.23}$$

Now to eliminate the electric field from Equation 2.23, Equations 2.17 and 2.18 are rewritten as

$$\mathbf{E} = \frac{1}{\mu_0 \eta}(\nabla \times \mathbf{B}) \tag{2.24}$$

and

$$\nabla \times \mathbf{E} = -j\omega \mathbf{B} \tag{2.25}$$

respectively, and substituted in Equation 2.23.

$$\nabla \eta \times \mathbf{E} + \eta \nabla \times \mathbf{E} = \frac{\nabla \eta}{\mu_0 \eta} \times (\nabla \times \mathbf{B}) - j\omega\eta \mathbf{B} \tag{2.26}$$

is now obtained, which can be combined with Equation 2.22 to rewrite Equation 2.23

$$-\nabla\nabla \cdot \mathbf{B} + \nabla^2 \mathbf{B} + \frac{\nabla \eta}{\eta} \times (\nabla \times \mathbf{B}) - \eta\zeta \mathbf{B} = \mathbf{0}. \tag{2.27}$$

For the second Maxwell Equation 2.18 a divergence operation is applied to eliminate the electric field. As the divergence of a curl $\nabla \cdot (\nabla \times \mathbf{E})$ equals 0,

$$\nabla \cdot (\nabla \times \mathbf{E} + j\omega \mathbf{B}) = \nabla \cdot \mathbf{B} = 0 \tag{2.28}$$

remains. This means that in this system of equations, the divergence of the magnetic field is zero. The leftmost term in Equation 2.27 therefore equals 0 and can be removed. Introducing the complex wave number $k = (-\eta\zeta)^{\frac{1}{2}} = (\omega^2\mu_0\epsilon - j\omega\mu_0\sigma)^{\frac{1}{2}}$ and substituting it into Equation 2.27

$$\nabla^2 \mathbf{B} + \frac{\nabla \eta}{\eta} \times (\nabla \times \mathbf{B}) + k^2 \mathbf{B} = \mathbf{0}, \tag{2.29}$$

results into the generalised Helmholtz equation. Since the complex wave number is a function of the EPs, a relation between the magnetic field and EPs has been established. Dotting the generalised Helmholtz equation with the transmit unit vector $\mathbf{i}^+$ yields the generalised Helmholtz equation for the B_1^+ field component

$$\nabla^2 B_1^+ + \mathbf{i}^+ \cdot \left[\frac{\nabla \eta}{\eta} \times (\nabla \times \mathbf{B})\right] + k^2 B_1^+ = 0. \tag{2.30}$$

Most direct EPT methods solve this generalised Helmholtz equation or a variation thereof. Hence, the differential relation in Equation 2.30 represents the cause of the direct methods their sensitivity to noise, as explained in Section 1.1. Here the differential relation is embodied by the Laplacian operator (∇^2) acting on the B_1^+ field. Equation 2.30 contains additional differential operators, these are however typically eliminated from the equation by assuming the medium to be locally homogeneous.

Under the local homogeneity assumption, meaning η being constant, the generalised Helmholtz equation can be reduced to

$$\nabla^2 B_1^+ + k^2 B_1^+ = 0. \tag{2.31}$$

Rewriting Equation 2.31

$$\frac{\nabla^2 B_1^+}{B_1^+} = -k^2 \tag{2.32}$$

and using the wave number's definition, yields

$$\sigma = \frac{1}{\omega\mu_0}\mathrm{Im}\left(\frac{\nabla^2 B_1^+}{B_1^+}\right) \tag{2.33}$$

and

$$\epsilon = \frac{-1}{\omega^2\mu_0}\mathrm{Re}\left(\frac{\nabla^2 B_1^+}{B_1^+}\right). \tag{2.34}$$

This solution to the generalised Helmholtz equation for locally homogeneous media defines H-EPT. To further simplify this methodology into SH-EPT, the polar decomposition of the B_1^+ field is substituted in Equation 2.31 and the real and imaginary parts are split

$$\sigma = \frac{1}{\omega\mu_0}(2\frac{\nabla|B_1^+|\cdot\nabla\phi^+}{|B_1^+|} + \nabla^2\phi^+) \text{ and } \epsilon = \frac{-1}{\omega^2\mu_0}(\frac{\nabla^2|B_1^+|}{|B_1^+|} - |\nabla\phi^+|^2). \tag{2.35}$$

It is then assumed that the term with the curvature of the transmit phase is dominant when reconstructing the conductivity

$$\nabla^2\phi^+ \gg 2\frac{\nabla|B_1^+|\cdot\nabla\phi^+}{|B_1^+|} \tag{2.36}$$

and that the term with the curvature of the magnitude divided by the magnitude is dominant when reconstructing the permittivity

$$\frac{\nabla^2|B_1^+|}{|B_1^+|} \gg |\nabla\phi^+|^2. \tag{2.37}$$

Yielding SH-EPT, a reconstruction method where the conductivity and permittivity are reconstructed based on only the phase and magnitude, respectively:

$$\sigma = \frac{1}{\omega\mu_0}\nabla^2\phi^+, \tag{2.38}$$

$$\epsilon = \frac{-1}{\omega^2\mu_0}\frac{\nabla^2|B_1^+|}{|B_1^+|}. \tag{2.39}$$

2.3 Incident Fields

The EM fields present in the imaging domain of the MRI scanner can be described as a superposition of incident- and scatter fields. The incident fields introduced in Section 1.2 represent the EM fields when no object is present in the imaging domain. Inversely, scatter fields represent the EM fields that arise from the presence of an object in the imaging domain. Applying said superposition to the electric field and the transmit magnetic field

$$B_1^+ = B_1^{+;\mathrm{inc}} + B_1^{+;\mathrm{sc}}, \tag{2.40}$$

$$\mathbf{E} = \mathbf{E}^{\mathrm{inc}} + \mathbf{E}^{\mathrm{sc}}. \tag{2.41}$$

Here, the superscript 'inc' refers to the incident field component, with the RF coils as its source. The superscript 'sc' refers to the scattering field component, with the scatter object as its source. **E** denotes the electric field. The scatter field components relate to how the EPs interact with the EM fields. Forward MR-EPT methods consider only the scatter field components as the measurable field of interest. Considering that (under the TPA) only the B_1^+ field is truly measurable, these methods assume the incident fields to be known quantities. This allows one to directly acquire the $B_1^{+;sc}$ field through superposition. In practice, however, it is difficult to acquire these incident fields as they are not directly measurable. Oftentimes the incident fields are simulated using the known geometry of the RF coils that act as their source. Simulated incident fields are unfortunately not an ideal solution. Since placing a scatter object in the MRI scanner loads its RF coils, influencing the incident fields it produces.

2.4 Contrast Source Inversion

Contrast Source Inversion is a forward method and optimisation scheme that iteratively improves the estimated electrical properties. As an inverse method, it starts with an initial guess of the EPs and compares the corresponding magnetic transmit scatter field with the measured counterpart. If minimised, this difference in estimated and measured fields, logically, yields a corresponding optimised estimate of EPs.

The following derivation is based on Leijsen et al.'s 3D CSI-EPT [23].

In CSI the incident fields are assumed to be known, using the magnetic transmit scatter field as input.

Combining the superposition of the B_1^+ field and the criterion of minimising the difference between estimated and measured fields, gives rise to a residual:

$$\rho = B_1^{+;\mathrm{sc}} - \hat{B}_1^{+;\mathrm{sc}}, \tag{2.42}$$

where $\hat{B}_1^{+;\mathrm{sc}}$ represents the estimated counterpart, which is further expressed as:

$$\hat{B}_1^{+;\mathrm{sc}} = \mathcal{G}_{\mathrm{data}}\{\mathbf{w}\}. \tag{2.43}$$

The field perturbations caused by the EPs, represented by the scatter fields, can be considered to have originated from a source. This source is referred to as the contrast source and is denoted by $\mathbf{w}$. $\mathcal{G}_{\text{data}}\{\cdot\}$ is the data operator that approximates the relation between a source and its B_1^+ field component. The data operator is defined as

$$\mathcal{G}_{\text{data}}\{\mathbf{w}\}(\mathbf{x}) = \frac{\omega}{c_0^2}\tilde{\nabla}\cdot\int_{\mathbf{x}'\in\mathbb{D}^{\text{obj}}} G(\mathbf{x}-\mathbf{x}')\mathbf{w}(\mathbf{x}')\,\mathrm{d}V. \tag{2.44}$$

Here, c_0 represents the electromagnetic wave speed in vacuum and $\mathbb{D}^{\text{obj}}$ is the object domain.

$$\tilde{\nabla} = -\frac{1}{2}(\mathbf{i}_x + j\mathbf{i}_y)\partial_z + \mathbf{i}_z\partial^+, \tag{2.45}$$

with

$$\partial^+ = \frac{1}{2}(\partial_x + j\partial_y) \tag{2.46}$$

and ∂_x, ∂_y, ∂_z are the spatial derivatives with respect to the Cartesian x, y and z directions. Likewise, $\mathbf{i}_x$, $\mathbf{i}_y$ and $\mathbf{i}_z$ denote the Cartesian unit vectors. G represents the 3-dimensional Green's function defined as

$$G(\mathbf{x}) = \frac{\exp(-\mathrm{j}k_{\mathrm{b}}|\mathbf{x}|)}{4\pi|\mathbf{x}|}, \quad \mathbf{x}\neq\mathbf{0}. \tag{2.47}$$

The residual can now be re-expressed as

$$\rho = B_1^{+;\text{sc}} - \mathcal{G}_{\text{data}}\{\mathbf{w}\}. \tag{2.48}$$

Hence, optimizing the contrast source minimises this residual. The aim, however, is to optimise for a set of EPs.

The contrast source relates to the EPs as follows:

$$\mathbf{w} = \chi\mathbf{E}, \tag{2.49}$$

where χ is the contrast function, which combines the EPs in a single variable.

$$\chi = \frac{\sigma + j\omega\epsilon}{j\omega\epsilon_0} - 1, \tag{2.50}$$

with σ the conductivity, ϵ the permittivity, ϵ_0 the permittivity of free space, ω the angular frequency and j the imaginary number.

As shown in Equation 2.49, the electric field is required to relate the EPs to the residual. Since the incident electric field is assumed to be known, we require the electric scatter field component to acquire $\mathbf{E}$ by superposition. The integral representation of the electric scatter field is defined as

$$\mathbf{E}^{\text{sc}} = \mathcal{G}_{\text{obj}}\{\mathbf{w}\}, \tag{2.51}$$

with $\mathcal{G}_{\text{obj}}\{\cdot\}$ the object operator, which approximates the relation between a source and its electric field component. The object operator is introduced as

$$\mathcal{G}_{\text{obj}}\{\mathbf{w}\}(\mathbf{x}) = (k_{\mathrm{b}}^2 + \nabla\nabla\cdot)\int_{\mathbf{x}'\in\mathbb{D}^{\text{obj}}} G(\mathbf{x}-\mathbf{x}')\mathbf{w}(\mathbf{x}')\,\mathrm{d}V, \tag{2.52}$$

with $k_b = \frac{\omega}{c_0}$ the wave number in vacuum.

Substituting Equation 2.51 into Equation 2.49 by means of superposition yields

$$\mathbf{w} = \chi\mathbf{E}^{\text{inc}} + \chi\mathcal{G}_{\text{obj}}\{\mathbf{w}\}. \tag{2.53}$$

This shows the contrast source to be a function of itself. Instead of computing the contrast source directly, a gradient descent scheme is introduced in order to incrementally update its value to optimise the underlying EPs. Equation 2.53 is rewritten as an additional residual to punish diverging contrast source values

$$\mathbf{r} = \chi\mathbf{E}^{\text{inc}} - \mathbf{w} + \chi\mathcal{G}_{\text{obj}}\{\mathbf{w}\}. \tag{2.54}$$

Combining both residuals and adding normalisation factors leads to the cost functional for the CSI algorithm:

$$F(\mathbf{w}, \chi) = \frac{||\mathbf{r}||^2_{\text{obj}}}{||\chi\mathbf{E}^{\text{inc}}||^2_{\text{obj}}} + \frac{||\rho||^2_{\text{obj}}}{||B_1^{+;\text{sc}}||^2_{\text{obj}}}, \tag{2.55}$$

where $||\cdot||_{\text{obj}}$ denotes the $\mathbf{L}^2$ norm on the domain of the scatter object.

Updating the contrast source using the gradient descent scheme at an iteration $[i]$ is performed using the following update formula

$$\mathbf{w}^{[i]} = \mathbf{w}^{[i-1]} + \alpha^{[i]}\mathbf{v}^{[i]}. \tag{2.56}$$

The superscript $[i]$ denotes the current iteration's value, while $[i-1]$ expresses the use of the previous iteration's result. Here, α represents the step length and $\mathbf{v}$ the update direction. Polak-Ribière update directions are typically used. The gradient of the cost function with respect to the contrast source is required to compute the update directions. Computing the gradient requires the contrast function, which for iteration $[i]$ is still unknown. Hence, the previous iteration's result of the contrast function is used $\chi^{[i-1]}$.

Using the updated contrast source, the total electric field can now be computed

$$\mathbf{E}^{[i]} = \mathbf{E}^{inc} + \mathcal{G}_{\text{obj}}\mathbf{w}^{[i]}. \tag{2.57}$$

The contrast function representing the EPs can now be directly computed as

$$\chi^{[i]} = \frac{\mathbf{w}^{[i]} \cdot \mathbf{E}^{[i];*}}{|\mathbf{E}^{[i]}|^2}, \tag{2.58}$$

which solves the least squares problem

$$||\chi\mathbf{E} - \mathbf{w}||^2. \tag{2.59}$$

Alternatively, the contrast function can be updated using a gradient descent scheme similar to the contrast source

$$\chi^{[i]} = \chi^{[i-1]} + \beta^{[i]}u^{[i]}. \tag{2.60}$$

Here β denotes the step length and u represents the Polak-Ribière update direction as a function of the gradient of the cost function with respect to the contrast function. The latter method of updating the contrast source with gradient descent is assumed to be used in this work.

By iteratively updating the initial guessed EPs using the aforementioned method, CSI minimises the cost function leading to optimised EP values. The iterative process is terminated when a user-specified tolerance for the cost function has been achieved or if a maximum number of iterations have passed.

The specifics and further details on 3D-CSI can be found in [23]. A summary of the major steps in CSI is shown in the diagram below.

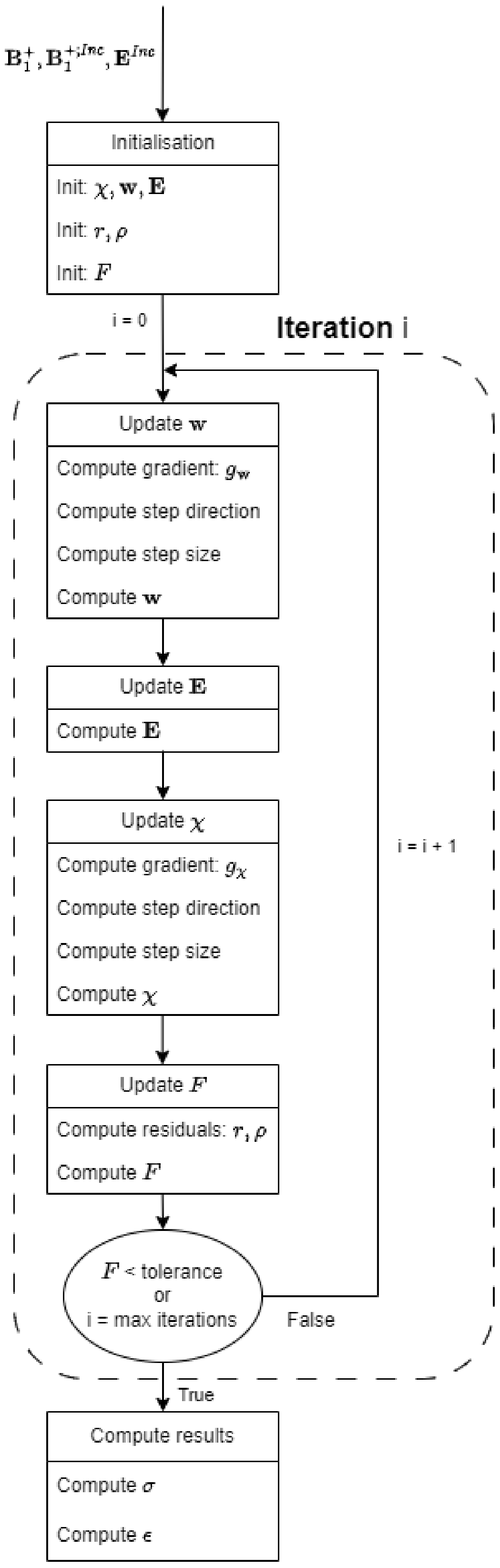


Figure 2.1: A diagram displaying the steps taken within the CSI algorithm.

3 Programming Methodology

3.1 Implementation Language

The starting point of the project is the 3D-CSI-EPT algorithm implemented in MATLAB by Leijsen based on the paper [23]. Instead of continuing in MATLAB, the code has been ported to Python. Using Python allows for easier usage of external GPU servers for neural network training, as no proprietary licenses are required. Python also features a wider range of deep learning tools to choose from. Python features more, or more straightforward, customisation options when it comes to neural networks, supporting seamless integration of (customised) neural networks with the CSI algorithm. The major disadvantage of porting to Python lies in the major time consumption required.

The PyTorch library has been chosen as a framework for DL. This choice was made due to its customization options and because of familiarity with the library.

3.2 Structure and Code Design

Besides porting the functionality of the CSI code and combining it with a Neural Network, the code is also refactored using several software design principles from [44].

The initial CSI code has been altered into an object-oriented design, splitting its main components into different classes. The structure of these classes is depicted below in Figure 3.1. Unlike regular design principles in object-oriented design, the code has been split into a relatively small amount of classes. This has several reasons to it.

Firstly, to retain a relatively high amount of similarity to the original code. This allows for more direct comparisons and ease in implementation and debugging.

Secondly, CSI has a very linear and sequential flow to it. Meaning that most parts are used in a single way, in a specific order and with a single goal in mind.

Lastly, a large amount of data is shared by multiple different methods and manipulated in different places. Overly fragmenting these methods yields highly coupled objects, conflicting with one of the main design principles of object-oriented design. Namely, loosely coupled objects, as opposed to highly coupled ones. Loose coupling promotes the ease of changing or extending a module without influencing the operations of others. This could be counteracted by building elaborate interfaces between the different objects but this would yield limited returns considering the unnecessary amounts of time and effort invested.

The advantages of still using object-oriented design to a certain extent, lie in the interchanging of modules and the sharing of particular methods and operators. Having these different objects communicate through unchanging interfaces, allows for switching out regular CSI methods for DL methods, without having to alter other parts of the code. Having specific structures for shared data, operators and methods, ensures that only the intended modules have access to them. This potentially allows for efficient construction and destruction of said structures, yielding optimised memory usage. Additionally, using classes that inherit from PyTorch's base class: nn.module, eases the process of constructing a model with NNs that can be moved to a GPU's memory for efficient parallel computation.

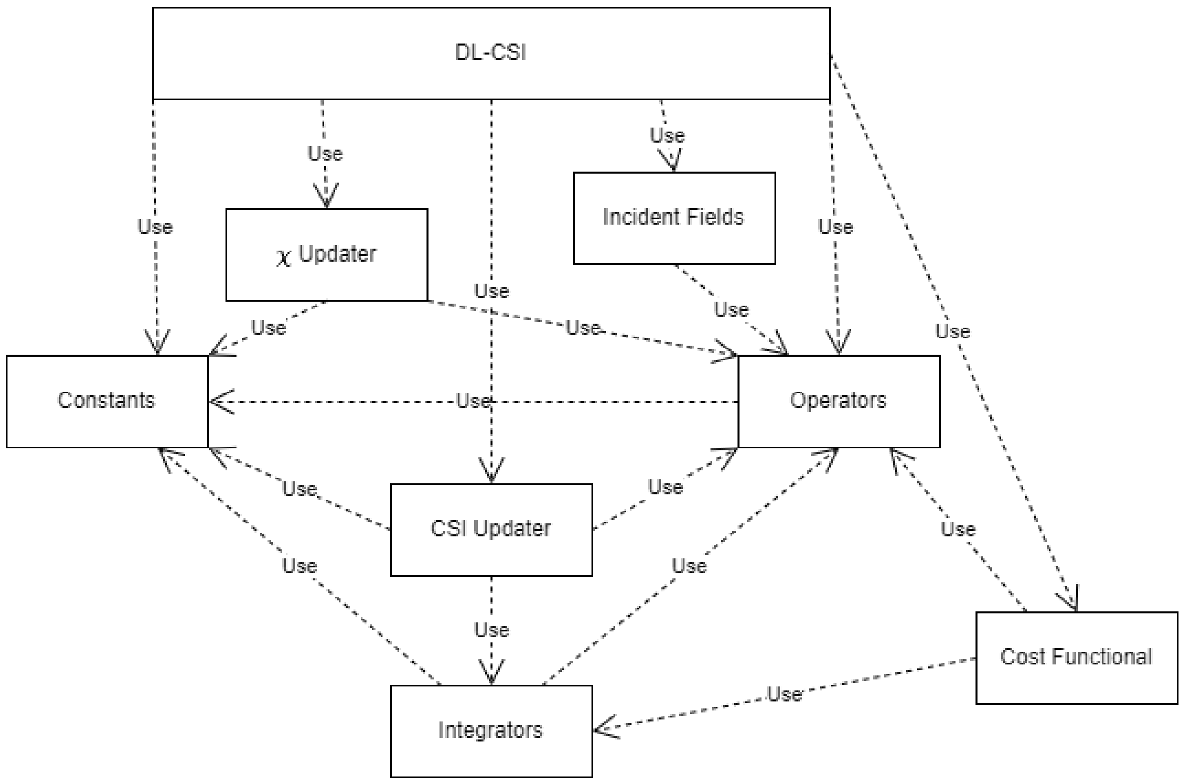


Figure 3.1: A representation of the class structure of the DL-CSI program.

3.2.1 DL-CSI Class

The largest displayed block represents the DL-CSI class. It functions as a manager, constructing the other classes that make up the building blocks of CSI and managing the data flow between them. The class interfaces different CSI functions and variables while allowing CSI functions to be replaced by a NN, essentially embedding a NN into the CSI algorithm. To facilitate the training process in DL for the entire module, regardless of the NN(s) embedded in the CSI algorithm, this class inherits from PyTorch's 'nn.module' class. Its main methods, therefore, coincide with those of such a module. Namely, the __init__() and forward() methods. The __init__() method, initialises the class and its corresponding submodules. The forward() method defines the linear forward operation of the class, applied to the model's input. In this case, this method contains the entirety of the (DL-)CSI algorithm. Though, not explicitly linear due to

CSI's iterations managed by a for-loop, unfolded it can be considered as such.

As much data as possible is hidden from this class by its submodules to decrease data dependencies. This allows for easier modification of each individual submodule while minimising its impact on this class. Additionally, this should make it easier to isolate errors.

3.2.2 Constants Class

The Constants class can be considered as a container for shared constant data. As shown in Figure 3.1, the Constants class is used directly by 6 out of the 8 classes and therefore, has major merit by being a shared container compared to each object containing its own instance of data. The constants within the class represent the physical constants used in CSI, such as the permittivity in vacuum.

3.2.3 Operators Class

The Operators class has a very similar goal to the Constants class. It is shared by many of the other classes. It contains the variety of operators used by the CSI algorithm, the majority being numerical representations of discretized operations. An example of such a discretized operator would be the discrete implementation of the partial derivatives in Equation 2.45. Ideally, it would consist of methods that apply these operations, barring any process outside of the module to alter the underlying operators. However, such an implementation would be too time-consuming for such a benefit within the scope of this project. Therefore, it currently functions as a simple container for these operators instead.

3.2.4 Integrators Class

The Integrators class is another shared resource type, class. It houses the integration methods used in CSI. These are the major components in implementing the $\mathcal{G}_{\text{data}}\{\cdot\}$ and $\mathcal{G}_{\text{obj}}\{\cdot\}$ operators introduced in Chapter 2.4 as Equations 2.44 and 2.52, respectively. This class is less heavily shared among the other modules, yet essential for those who do use it.

3.2.5 Cost Functional Class

This class computes and updates the residuals and cost functions of CSI. Hence, it has the sole purpose of assessing the current progress of the running CSI. Its methods are limited to initialization and update functions. The residual computations implement Equations 2.48 and 2.54. The cost functional implementation of Equation 2.55 is split into the left- and right-hand additive terms and are referred to as FE and FB.

3.2.6 χ Updater Class

The χ Updater class is responsible for computing and updating the new estimate of χ. It has been split from the other parts of CSI with the aim of interchangeability of its implementation while maintaining a constant interface. This allows seamless switching between regular CSI, conjugate gradient descent methods, and neural network implementations. This module is the essence of this project and, therefore subject to many changes. Splitting it from other parts of the code, with a constant interface, is essential to limit perpetual changes to other parts of the code.

3.2.7 Incident Fields Class

The Incident Fields class is separated from the main CSI module for similar reasons to the χ Updater class. As mentioned earlier, potentially circumventing the assumption of known incident fields through indirect DL was an avenue to potentially be explored in this project. This means that this part of the code was likely to change and should be able to interchange regular and DL methods. Ultimately, the use of DL to improve incident fields was dropped to limit the scope of the project.

3.2.8 CSI Updater Class

The CSI Updater class contains all other elements of CSI not yet implemented in the other classes. This means failing to adhere to the principle of classes having a single responsibility. Though as expressed above, maintaining a higher resemblance to the original 3D-CSI code has been favoured for comparability and ease of translation.

3.3 Debugging

Translation and refactoring of the code requires efficient and exhaustive testing to ensure identical (or improved) behaviour compared to the original code. This section describes three methods that were used to rid the code of bugs.

3.3.1 Unit Test

A unit test refers to testing the smallest testable element of code. This is the lowest level of testing to ensure a function/class/method behaves as intended. Unit tests were for example used to validate if the operators in the Operators class, were created correctly. Unit testing helped isolate hard-to-detect problems, such as a small output difference in computing the step length determined to update χ, caused by a capital letter instead of a lowercase one.

Unit testing, however, has its own fallacies. The small, inherent deviations in floating point variables make it more difficult to check for equality when the typically used data consists of small numbers. The main drawback lies in designing the unit test. The test should be exhaustive, meaning that it should cover all possible avenues of using the piece of code. Accounting for all options can be difficult and time-consuming. Failing to properly design a unit test, caused a bug to slip through in the translation process of the CSI algorithm. When testing the use of PyTorch's dot function for computing the dot product of two vectors in the same way as MATLAB's version, only real numbers were used. While for real numbered vectors the operations acted equally, for complex numbers they did not.

Not all functions in this project could be unit-tested in isolation. This issue arises due to dependencies in the code, such as a function from the CSI Updater class, making use of a function from the Integrators class, which in turn uses an operator from the Operators class.

3.3.2 Integration Tests

Integration testing concerns the interaction between different modules and interfaces. Unlike writing unit tests, writing integration tests is done without considering how the modules have been implemented. It is meant to test the interaction between interfaces rather than the functionality of specific methods within a module. In the case of this project, integration testing mostly revolves around the dependency between classes. An example of this would be testing the computation of a residual in the Cost Functional class, which depends on the usage of the functions of the Integrators class.

The CSI algorithm has quite a sequential structure in terms of operation. This makes integration testing more challenging to achieve without having to resort to executing prior parts of the sequence up until the current interface that has to be tested. Especially, as not having a thorough enough understanding of how the data in between steps typically looks like, limits the extent of artificial inputs representing real data.

To circumvent these issues, an interface between the to-be-tested modules and the original MATLAB CSI implementation was set up. In this manner, replacing equivalent MATLAB functionalities with the module undergoing testing essentially integrates the new module in the old code. This allows the detection of discrepancies between old and new. The interface between MATLAB and Python was implemented using the MATLAB engine, a feature that allows the user to run and interact with an instance of MATLAB within the Python environment.

Integration testing with the MATLAB interface has been performed sequentially on the CSI pipeline. This means that the MATLAB functions were replaced one at a time by the Python counterpart until the result deviated. A deviation was detected when passing the updated gradient of the contrast source to the next function. The data structure was incorrectly interpreted by the next function, giving rise to an error.

3.3.3 System Test

The system test tests the program as a whole. It functions much like a complete version of the integration test, testing the interactions between all modules. In this case, a system test was performed to validate whether the translated CSI implementation functioned in the same manner as the original MATLAB implementation. A simulated magnetic transmit field, which corresponds to the model of a human head, was used as an input for both the Python and MATLAB codes. The validity was checked by comparing the cost functional's value every 100 iterations. As long as the values were within a tolerance range of each other (0.1%) the test was considered to have been passed.

The system test resulted in the detection of an issue with the generation of the simulated incident fields. A unit test did not detect this issue due to the function working as designed. However, a design flaw was found when the simulated incident fields were used in the system test and Python's CSI failed to converge as in the MATLAB implementation. The system test was passed after the correction of the design flaw.

4 Deep Learning for Contrast Source Inversion

This chapter covers strategies for embedding DL into the CSI methodology. Initially, background information is provided on different neural network types which are typically prevalent in the field of (medical) image reconstruction. Afterwards, strategies for embedding DL in CSI are presented. The chapter is concluded by the introduction of DL-embedded CSI topologies.

4.1 Neural Network Types

The amount of different neural network types has grown in the last years. Therefore, this section aims to introduce three different NN types prevalent in (medical) image reconstruction. Most of the information provided comes from the book Dive into Deep Learning[31].

4.1.1 Convolutional Neural Networks

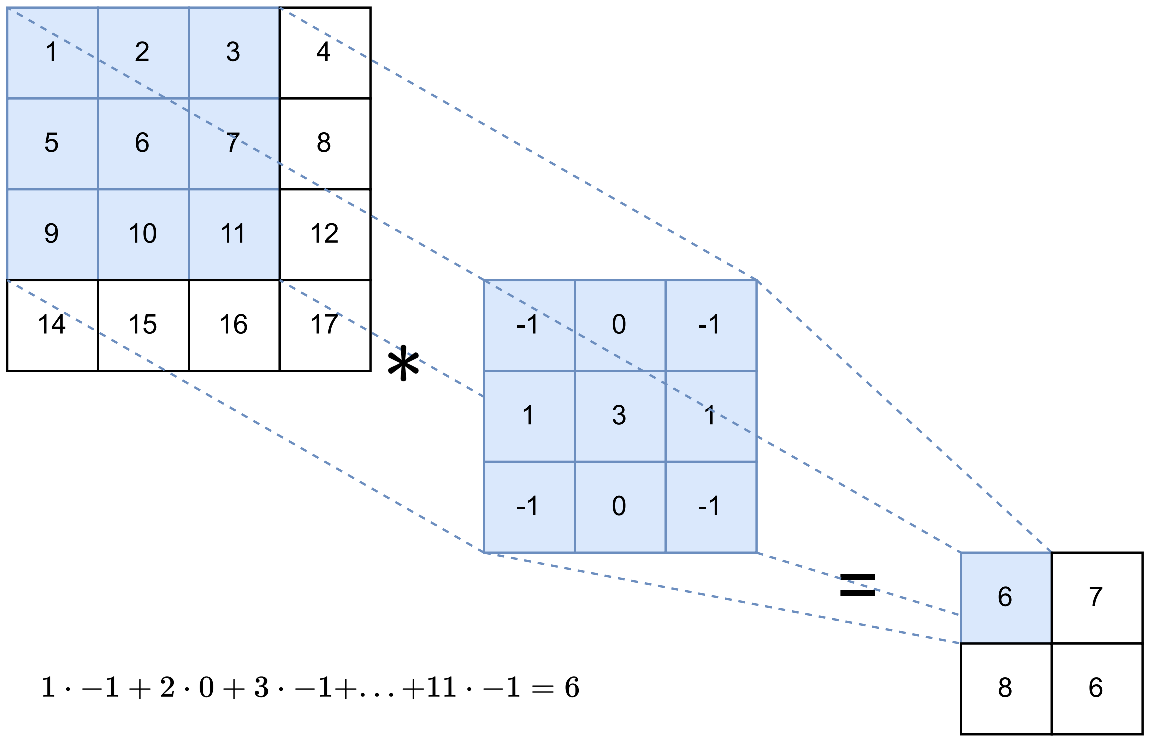
A Convolutional Neural Network (CNN) is a NN that features convolution operations to interact with the input data. More specifically, convolution layers in the CNN make use of a convolution operation on the input data; to combine information on multiple data points to construct its output. The (discrete) convolution operation functions as follows:

$$(x * y)[n] = \sum_{m=-\infty}^{\infty} x[m]y[n-m] \tag{4.1}$$

Here x can be treated as an input signal and y as a kernel. While the kernel shifts over the input, the product at every shift $[m]$ is summed to produce the output. In image processing a kernel refers to the small matrix used in convolutions; different kernel values produce different effects such as blurring with a Gaussian kernel. In the case of convolution layers in DL, the kernel's values are weights to be learned by the network. A visualisation example of a 2D convolution is provided in Figure 4.1.

The convolution operation can be extended to higher dimensions, such as in our case of 3D data. This allows a convolution layer to operate on image data without the need to flatten them into vectors. This allows the input to retain its spatial relations.

Convolution layers feature multiple options to tweak the convolution operation to the user's needs; padding being the most relevant. Padding refers to appending values to the borders of the input data and is used to provide the convolution kernel values around the border of the input data. Padding the input with an equal amount as half the kernel size minus one ensures the centre voxel of the kernel starts on the original border voxels. This ensures that the size of our image domain remains constant.



1
2
3
4
5
6
7
8
9
10
11
12
14
15
16
17
*
-1
0
-1
1
3
1
-1
0
-1
=
6
7
8
6
$1 \cdot -1 + 2 \cdot 0 + 3 \cdot -1 + \ldots + 11 \cdot -1 = 6$

Figure 4.1: Visualisation of a 2D convolution with a kernel of size 3

A convolution layer tends to act locally on its input by using relatively small kernel sizes. This prevents the number of computations from becoming infeasible for large input sizes. Additionally, this reduces the amount of memory required to store the layer's weights. The disadvantage of using small kernels is the loss of information on global relations within the data. This loss, however, is reduced by using the second characteristic layer of a CNN, the pooling layer.

The pooling layer condenses data locally by either averaging or finding the maximum value of a local patch of data. This essentially pools the information in small areas into singular values, resulting in a sort of down-sampled version of the data. Combining this layer with a new convolution layer afterwards gives the convolution access to slightly more global relations in the data. Cascading the combination of convolutional- and pooling layers can be compared to going from learning higher frequency to learning lower frequency information. A pooling example is visualised in Figure 4.2.

More recent CNN topologies also feature skip/residual connections, which break with the normally sequential pipeline of a CNN. A skip connection retains an intermediate output and concatenates it later on with a more recent output, skipping forward. This allows the CNN to combine information from before and after pooling layers. Meaning that more local and more global information can be combined. The first CNN with residual connections is called ResNet and was introduced by He et al. [45]. Such a CNN with residual connections will be used in Chapter 6.

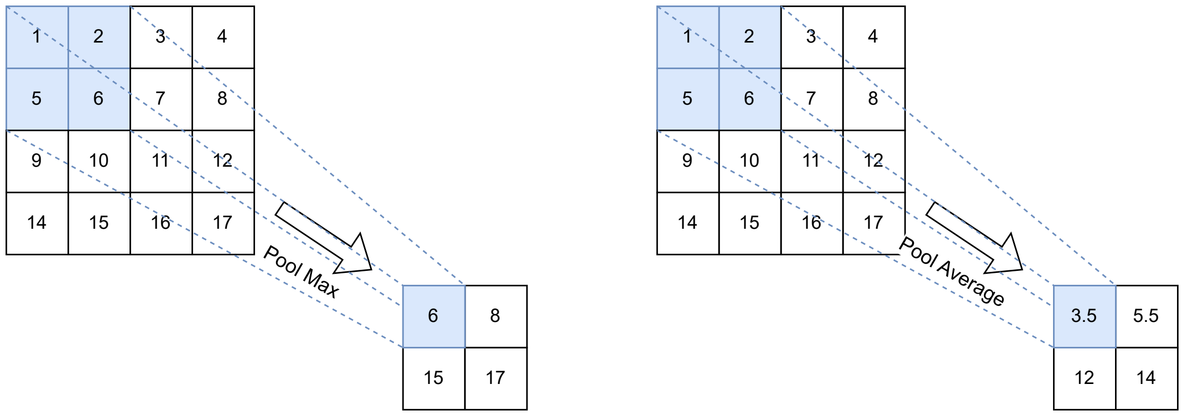


Figure 4.2: Visualisation of a max pooling on the left, where the highest value is propagated to the result and an average pooling on the right, where the average value is propagated to the result.

4.1.2 Constrained Generative Adversarial Network

A Generative Adversarial Network (GAN) can be split up in two parts. A generator network and a discriminator network; the adversaries. The generator's task is to forge an output that is indistinguishable to the discriminator from real data. The discriminator's task is to distinguish between real data from the fake data generated by the generator. The feedback from the discriminator lets the generator learn when its output is good enough to fool the discriminator or if it needs to be adjusted. The discriminator is fed either fake or real data to learn to distinguish between the two. So in essence, these two adversaries compete with each other, pushing the network to a higher performance. The generator of a GAN is fed random noise as a starting point.

A constrained GAN (CGAN) on the other hand, starts with structured input data (a low-resolution version of the intended output of sorts) to push the output further into a certain direction. If properly tuned, a cGAN can for example be used to increase the sizes of data sets. Since a perfectly functioning cGAN would be able to generate new data that is indistinguishable from the data in the real data set.

A typical issue with GANs is when one of the two adversaries starts to greatly outperform the other. Leaving no more room for the other to grow.

A potential cGAN-embedded CSI topology will be introduced in Subsection 4.3.3.

4.1.3 Recurrent Neural Networks

Recurrent Neural Networks (RNN) are typically used to deal with sequences of data. They feature hidden states that retain information from previous inputs. This allows previous inputs to influence the outputs of the current part of the sequence. The following paragraphs will introduce three different kinds of RNN elements; the fully connected element, the Long Short Term Memory and the Gated Recurrent unit. The final paragraph introduces a Recurrent Inference Machine, which is a specific type of RNN relevant to this work.

The simplest form of a RNN features a fully connected layer, which also takes a hidden state as input. It then produces an output that is also used as the hidden state for the next part of the sequence (or the next iteration). This implementation has the disadvantage of producing exploding or vanishing gradients while backpropagating. A fully connected layer refers to a layer with an arbitrary amount of weights, where all input values are multiplied with every weight to produce an output. This RNN implementation uses the least amount of memory out of the three options but also features the lowest amount of complexity.

Long Short Term Memory (LSTM) is an implementation that features multiple gates, an input node and an internal state in its layer. The output of the LSTM is shaped by a combination of the input-, hidden- and the internal state. The extent to which the output is comprised of the input-, hidden- or the internal state is managed by the output gate. The internal state is shaped by the previous internal state, the input state and the hidden state. The forget gate determines how much of the previous internal state should be forgotten, while the input gate determines how much of the input- and hidden state should be saved into the internal state. The LSTM learns what information it should store for the future and when to reset its memory using the different gates. In this manner, it prevents the vanishing and exploding gradient problem. The disadvantage of this implementation comes in the form of higher computational load and higher memory usage due to the additional gates.

The Gated Recurrent Units (GRU) tries to achieve similar results to the LSTM while using less computational power. Therefore, it uses fewer gates while still influencing how much of the past information should be used and how much should be stored for the future.

A GRU does not use an internal state to retain past information. Instead, the hidden state is used as the source of past information. It features a reset gate that determines a candidate hidden state based on the previous hidden state and the input. This is followed by an update gate that determines how much of the previous hidden state and the candidate hidden state is to be used to produce the new hidden state. The GRU has a reduced computational load and memory usage compared to the LSTM and higher adaptability to the use of past information than the fully connected RNN. Hence, the GRU is chosen as most suitable for the potential RNN-based topologies in Section 4.3.

Recurrent Inference Machines are a form of RNNs that are used to solve inverse inference problems iteratively. They are typically considered as an alternative to maximum a posteriori (MAP) based solutions. Like in MAP-based solutions, the inference problem is mathematically described using a (negative) log-likelihood function and a prior. A likelihood function expresses how likely a certain outcome is based on a given observation, such as an input. The prior provides information on the distribution of the observed variable and is typically unknown for inverse inference problems. A RIM takes as input the estimated variable to be improved and the gradient of the log-likelihood function with respect to the estimated variable. A RIM iteratively improves the estimation of the variable using recurrent layers. The same network architecture is used for each iteration but memory states are passed on internally. RIM is a framework that does not require explicit models of a prior(/regularisation) or inference procedure instead their gradient is implicitly learned by the NN.

4.2 Deep Learning Strategies for Contrast Source Inversion

Before diving into the choice of what kind of neural network to embed into CSI, first an understanding of what variables can be trained on needs to be established. As we aim to use physics-assisted deep learning, domain knowledge is preferably leveraged in the input of the NN. This means that instead of directly using the measured transmit magnetic field as input to the NN to compute EPs, a different variable derived from the measured field using CSI's regular operations is used as input. The variable has to fulfil two conditions; a ground truth of the variable needs to be available for training and the variable needs to be closely related to the EPs.

The dataset available for training is generated from the model of a human head. The different samples vary in geometry and EP values. Electromagnetic fields have been simulated from these EP head models, providing us with a paired set of EPs and their corresponding EM fields. More specifically, the magnitude of the magnetic transmit field and the transceive phase are combined as input for the CSI algorithm. The dataset provides three potential variables to train our NN on.

Firstly, a ground truth contrast function can be directly computed from the ground truth EPs using Equation 2.50. The direct relation between the EPs and the contrast function makes training on the contrast function instead of directly on the EPs slightly more efficient by requiring one less step while backpropagating. Backpropagating refers to the process of adjusting the weights of the NN depending on the loss computed. The contrast function is normally updated using either Equation 2.58 or Equation 2.60. Replacing these update equations with a NN could potentially converge its values faster, by relying on its learning ability and the non-linearity of the NN. As the regular update equations either solve a least squares problem or leverage a gradient descent scheme, replacing these computations with a NN does not lead to the NN having to relearn known physics.

Secondly, the contrast source can be used as a training variable. With both the electric field and EPs being available in the dataset, a ground truth for the contrast source can be computed using Equation 2.49. The contrast source is only removed from the EPs by a single additional step compared to the contrast function. Similar to the contrast function, replacing the regular update Equation 2.56 with a NN does not discard a known physics relation and can potentially speed up convergence.

Thirdly, the electric field can be used as a training variable. The electric field's update Equation 2.57 does relate to the EPs through the contrast source. However, the update relation is a simple physics-based computation. A reason for using the electric field as (an additional) training variable could be its importance in determining the SAR, which is one of the main applications of using EPT. However, using a NN over the physics-based computation is unlikely to yield improvement to CSI as a whole. Additionally, using the electric field would not classify as a physics-assisted DL method as it essentially relearns known physics.

Multiple training variables could potentially be used as well. This might improve the convergence rate but it could potentially decrease data consistency as more of

CSI is replaced by neural networks. Using multiple training variables in the case of a hybrid CSI approach would require the use of more neural networks. More neural networks lead to higher memory usage and their implementation and tuning become more time-consuming.

Ultimately, only the contrast function is selected as a training variable. It requires slightly less data than using the contrast source and is marginally more directly related to the EPs. Using the electric field as a training variable is discarded due to its non-physics-assisted nature. No additional training variable is chosen to limit the scope of the project to a manageable degree. Hence the network will only be trained with the contrast function as the training variable.

In addition to the contrast function, gradient information of the contrast function will be leveraged by the NN when computing an update. By using both the contrast function and its gradient as input, the network attempts to implicitly learn gradient descent steps. This way the NN attempts to learn the step length and direction in a data-driven manner, rather than analytically.

4.3 Neural Network Embedded CSI Topologies

This section outlines possible topologies for combining the CSI algorithm with different types of NNs. These hybrid approaches are inspired by literature and use the contrast function and its gradient as inputs for the NN. First CNN CNN-based topologies are presented, followed by a cGAN topology. Subsequently, RNN-based topologies are proposed.

4.3.1 Cascaded CNNs

The first design is based on the principle of algorithm unrolling. Algorithm unrolling is the principle of replacing/embodying each iteration of an iterative algorithm with one or a small amount of NN layers. Algorithm unrolling has been gaining ground in the fields of signal- and image processing [46]. Typically, each iteration is unrolled as a single network layer and training the complete network end-to-end. This would, however, enforce a direct deep learning method rather than a physics-assisted one. So instead of replacing iterations entirely by NN layers, this potential topology replaces only the contrast functions' update step with a small NN. Algorithm unrolling is typically implemented using CNNs when used to solve image processing problems. CNNs are generally good at image processing tasks due to making use of local spatial relations. Hence the proposed network will make use of small CNNs. A schematic representation of the proposed topology is shown in Figure 4.3.

Changing from a single complex CNN for unrolling the iterations into a multitude of smaller ones, resembles cascaded CNNs. Cascaded CNNs have been proposed before as a way to reduce a single, large and complex CNN into a multitude of smaller, less complex CNNs[47]. However by combining it with algorithm unrolling it is not entirely similar to this situation; the advantage of smaller, less complex CNNs compared to other single network topologies, should still hold.

In terms of network architecture, possible candidates are a patch-based approach as in [36] or an up- and down-sampling approach like in a U-net [48]. These approaches both leverage information at progressively differing levels of spatial locality or resolution.

An expected disadvantage of this design is its unknown quantity of CNNs that is necessary for convergence. This would be determined experimentally. Possibly requiring the training of a lot of networks, making it a time-consuming task. Hence, this topology is deemed unfavourable.

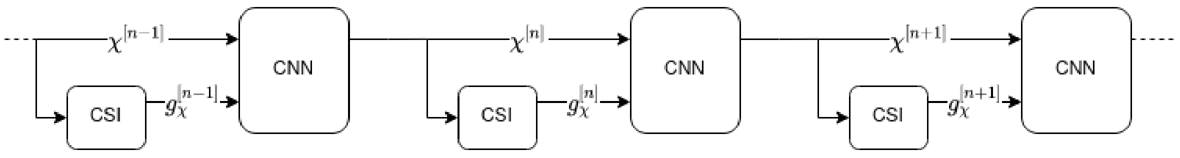


Figure 4.3: A CSI topology using a cascade of convolutional neural networks; one for each iteration. $\chi^{[n]}$ represents the n^th iteration's contrast and $g_\chi^{[n]}$ its corresponding gradient.

4.3.2 Coarse and Refiner CNNs

The second design is rather similar to the first; the cascaded unrolling of CSI is retained but the CNNs are reduced in number and split up into two categories. The first category will be dubbed a Coarse CNN. The first (few) CNNs are expected to change the estimate to the greatest extent, making coarse changes as it were. The later CNNs are expected to make finer adjustments to the estimate, hence they will be called Refiner CNNs. As the changes these Refiner CNNs apply to the estimate, are expected to be smaller and probably quite similar, they might be used in multiple iterations. This design concept is illustrated in Figure 4.4, with a single coarse NN and a single refiner NN used. However, an arbitrary number of coarse and refiner NNs can be used in succession. The NN does not differ much from the cascaded CNN approach implementation-wise and can be easily adjusted from one to the other. The ease of adjustment allows for potential joint investigation and comparisons.

Reducing the Cascade of CNNs to a few Coarse and a few (or possibly even a single) Refiner CNNs limits the amount of NNs in the topology and the time it would take to manage them. Eliminating the uncertainty in CNN amount of the Cascade approach.

Its downside is the possibility of the Refiner CNN(s) not increasing the convergence when used iteratively. Likening it to gradient descent, the update steps of the refiner might learn to continuously apply its steps in a similar direction. Hence, if the update overshoots the lowest point of convergence, the refiner CNN might be unable to adjust its step direction and continue diverging instead.

This approach could be considered a more elaborate case of improving an initial guess with DL as in [49], but in this case, the iterative improvement is done with NNs as well.

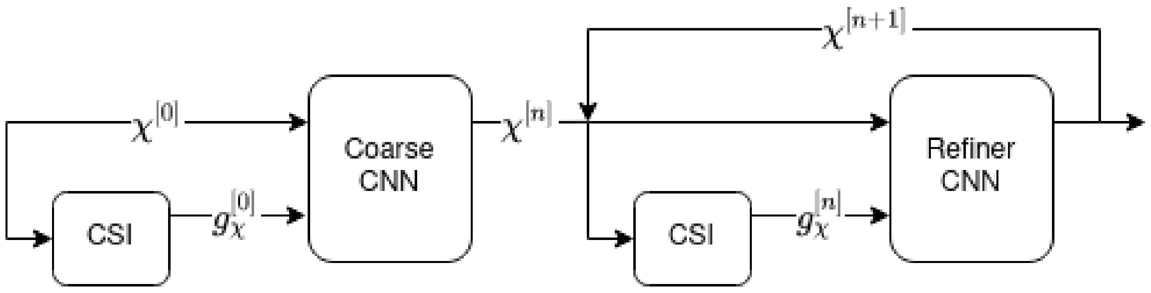


Figure 4.4: A CSI topology using a coarse CNN once, followed by the iterative use of a refiner CNN. $\chi^{[n]}$ represents the nth iteration's contrast and $g_\chi^{[n]}$ its corresponding gradient.

4.3.3 cGAN

This topology, as shown in Figure4.5, uses a conditional generative adversarial network (cGAN) in a less conventional way. When learning, the generator will produce a new estimate based on its previous estimate as well as the corresponding gradient. Its estimates should improve based on the discriminator's feedback. The discriminator learns from differentiation estimates and ground truths.

When evaluating, the generator will iteratively produce new estimates until the discriminator deems it accurate.

This method might suffer from difficulties in tuning. The discriminator should be rather strict, as it determines if the proposed estimate is adequate or not. However, if the discriminator does not learn to recognise some estimates with significant changes such as tumours, it would seriously impair the approach.

The generator could also learn to alter the estimates in a way that does not converge to the ground truth but rather to something that consistently passes the discriminator's check.

These difficulties in tuning and the uncertainties in convergence lead to this topology being an unfavourable solution.

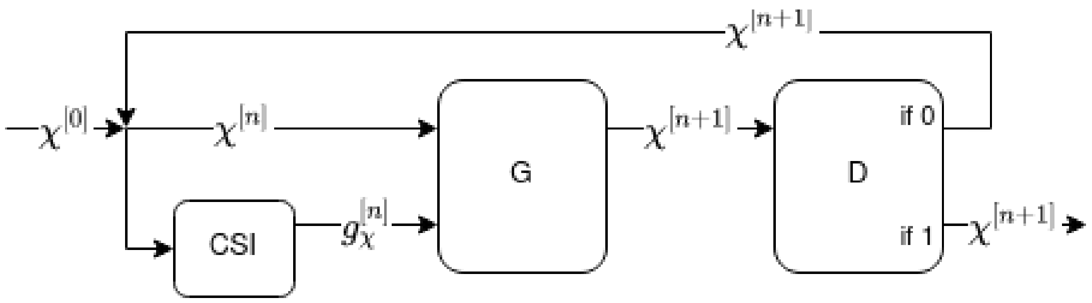


Figure 4.5: A CSI topology using a cGAN's generator to improve the contrast's estimate every iteration, until the discriminator deems it sufficient. $\chi^{[n]}$ represents the nth iteration's contrast and $g_\chi^{[n]}$ its corresponding gradient.

4.3.4 Convolutional Recurrent Neural Network

In the field of accelerated dynamic MRI reconstruction, Qin et al. [40] introduced a novel network design to replace typical unrolling approaches. The network is referred to as a convolutional recurrent neural network (CRNN). A CRNN is a variant of the RNN that uses convolutions as operators instead of elementwise multiplications. This combination can be applied to all the different kinds of memory cells of an RNN, like an LSTM or GRU.

Using convolutional operators in the memory cells allows for local spatial relations to be taken into account, even across iterations. Using a convolution approach rather than a fully connected kind of approach to using the memory cells, alleviates much of its burden on the memory for large 3D image data.

For the topology, a single CRNN would be used iteratively, as shown in Figure 4.6.

Opting to use a single network is likely to require less tuning than using multiple NNs. However, the network itself is expected to be very large and quite complex considering it will have a large amount of memory states. Implementation-wise this approach is expected to be relatively more time-consuming due to the convolution-based RNN layers requiring manual construction. This topology is expected to be a very feasible option logistics-wise. Because the small amount of weights used by convolution operators limits excessive memory usage, allowing the use of more complex network implementations.

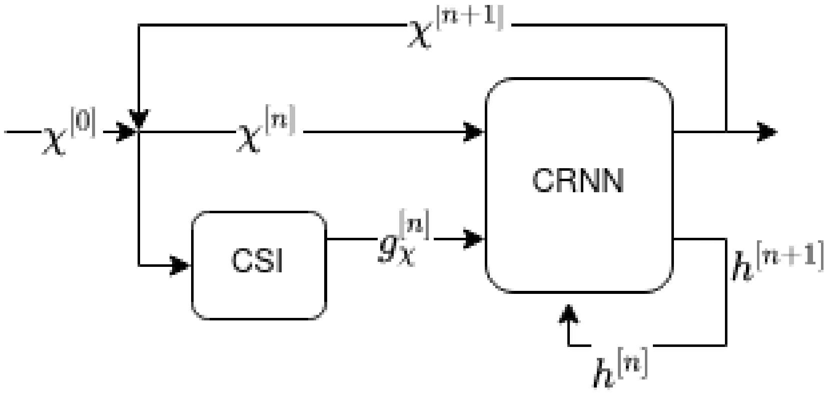


Figure 4.6: A CSI topology using a convolutional recurrent neural network iteratively. $\chi^{[n]}$ represents the n^th iteration's contrast, $g_\chi^{[n]}$ its corresponding gradient and $h^{[n]}$ the corresponding memory state.

4.3.5 RIM

The recurrent inference machine (RIM) topology acts rather similarly to the CRNN, replacing only the CRNN block with an RIM block in Figure 4.6. Instead of combining RNN layers with convolutions, it alternates CNN and RNN layers instead. As aforementioned a RIM takes in both the contrast function and the gradient of the log-likelihood function as input. Rather than explicitly deriving the gradient of the log-likelihood function, we leverage the contrast function's gradient used in CSI's gradient descent scheme. This substitution of gradients should be warranted due to the similarities between the (negative) log-likelihood and CSI's cost functional. The log-likelihood function is a measure of how likely the measured transmit magnetic field is caused by the estimated contrast function. Similarly CSI's cost functional assigns a cost to the extent that the magnetic transmit field computed from the estimated contrast function corresponds to the measured magnetic transmit field. Both functions punish deviations from the intended result and intend to be minimised. Hence, using the gradient derived from the cost functional should achieve a similar effect when used for an RIM.

RIMs have also been successfully used in other MRI reconstructions before [39].

The major advantage of this topology is its inherent use of gradients. However, the implementation of this concept might be applicable to some of the other topologies as well. Like the CRNN, the RIM uses spatial information as well as memory states. Though they are less intertwined. In terms of memory usage, the RIM requires substantially more than a CRNN. The implementation of an RIM is significantly easier, as no custom layers need to be designed. It also requires less deep of a network to take into account data covering the entire imaging domain. Hence, the RIM topology is deemed most advantageous and is used in an attempt to improve CSI.

5 Implementation of Deep Learning Enhanced Contrast Source Inversion

This chapter covers the implementation of the hybrid deep learning CSI method, with the RIM topology chosen in chapter 4. The hybrid, physics-assisted deep learning method for CSI is dubbed Deep Learning Enhanced Contrast Source Inversion (DLE-CSI). DLE-CSI aims to achieve a speed-up over CSI while achieving a similar level of reconstruction quality, offsetting the forward method's issue of long computational times.

The start of the chapter covers the prerequisites necessary to work with DLE-CSI. Afterwards, a description of the RIM topology implementation chosen from Chapter 4 is given. Which is followed by an elaboration on the training strategies used. The chapter is concluded with the tests performed and their corresponding results.

5.1 Prerequisites

In order to use and train DLE-CSI some preliminary steps are required. These steps are visualised in Figure 5.1 and will be elaborated on in the following subsections.

5.1.1 Dataset

As aforementioned in Chapter 4, the provided dataset consists of simulated electromagnetic fields of human head models. The field maps used from the dataset are the magnitude of the transmit magnetic field, the transceive phase and the EPs. The dataset contains 180 sets of 3D field maps with voxels that have an edge length of 2 mm. Each map is of the dimensions $128, 128, 80$ in the x, y, z directions respectively.

The contrast function is computed from the EPs using Equation 2.50, which is to be used as Ground Truth (GT) for training the network. The transmit phase is computed from the transceive phase using the TPA, as per Equation 2.14 and combined with the magnitude of the transmit magnetic field to produce the complex transmit magnetic field B_1^+.

5.1.2 Incident Fields

In order to derive the transmit magnetic scatter field $B_1^{+;sc}$ and to initialise the electric field E, the incident fields are required. As described in section 2.3, these fields are not actually known. Hence, they have been simulated using the MRI scanner's RF birdcage coil's known geometry.

The incident fields have been simulated following Stijnman's methodology [50]. The legs of a birdcage coil are treated as infinitely long line sources. These line sources are implemented as a line along the z-axis with a single voxel of thickness in the x,y-plane.

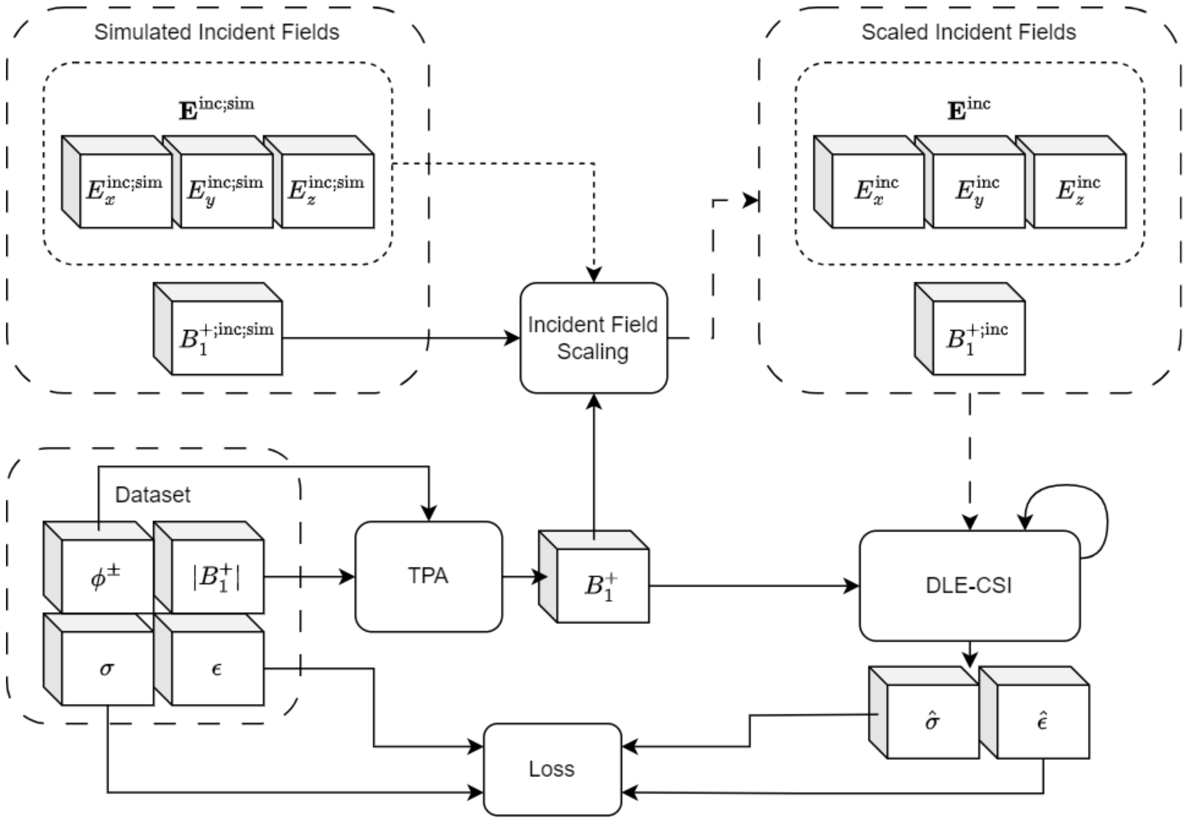


Figure 5.1: Visualisation of the preliminary steps required to train DLE-CSI starting with a dataset and a set of simulated incident fields. Cubes represent 3D data and dashed arrows represent inputs/outputs from data grouped with the corresponding dash-style.

The simulated legs are distributed equidistantly in a circular fashion, resembling the cross-section in the middle of a birdcage coil. The amplitude of all the line sources is set to unity, whereas the phase is shifted by $\frac{2\pi}{16}$ according to operation in quadrature mode. In order to take into account the presence of RF shields in MRI machines, mirror sources are leveraged. Mirror sources have their amplitude reversed in polarity and are placed at such a distance from the line sources that at the location of the RF shield the field contributions should equate to 0. This distance is defined in [50] as:

$$d = \frac{r_{shield}^2}{r_{coil}} \tag{5.1}$$

from the centre. Where r are the radii of the shield and coil at the equivalent subscript.

To compute the incident fields from this collection of simulated leg sources and mirror sources, functions inherent to the CSI algorithm are leveraged. These are the $GE(s)$ and $GB(s)$ functions. In CSI, these functions produce the electric and magnetic field produced by a source s. These functions implement the data- and object operator described by equations 2.44 and 2.51, in which case they are applied to the contrast source $\mathbf{w}$ to produce the scatter field components. For the incident fields, $\mathbf{w}$ is replaced by the coil representation of line sources.

The amplitude of the simulated current sources is set to unity due to the actual amplitude of the current being unknown, which leads to errors in the resulting incident fields. The magnitude of the current is a multiplicative factor and can be considered to scale the incident field. Hence, the simulated incident fields can be scaled to correct the errors originating from setting the current's amplitude to unity. Therefore, the simulated fields are scaled depending on the B_1^+ field used as input to DLE-CSI, depicted in Figure 5.1 as the Incident Field Scaling block. For scaling, first, an average value of the B_1^+ field outside the object domain is computed. Secondly, the average of the simulated $B_1^{+;inc}$ outside the object domain is computed. Limiting the average value computation to be outside of the object domain should ensure minimal inclusion of scatter field values caused by the object. From these averages, a scaling factor is computed by division of the average B_1^+ field value by the average of the $B_1^{+;inc}$ field value. Subsequently, the simulated incidents are scaled with this scaling factor to yield the incident fields used as input for DLE-CSI. Though likely not accurate, this scaling method should at least produce an estimate with the magnitude of the incident fields at a similar scale of the actual fields.

5.2 Neural Network Implementation

As was explained in Chapter 4, the neural network is to replace CSI's update step of the contrast χ. More specifically, the update Equation 2.60 is replaced by an update from the NN. Two different approaches to updating the contrast function with a NN have been implemented. The first approach, the replacement update manner, updates the estimated contrast by replacing it entirely with the output of the NN:

$$\chi^{[i+1]} = \mathrm{NN}(\chi^{[i]}, g_\chi). \tag{5.2}$$

Where $\mathrm{NN}(\chi^{[i]}, g_\chi)$ is the neural network's output as a function of the current contrast estimate and its respective gradient. The second approach, the additive update manner, updates the estimated contrast by adding the output of the NN to the current estimate:

$$\chi^{[i+1]} = \chi^{[i]} + \mathrm{NN}(\chi^{[i]}, g_\chi). \tag{5.3}$$

The explanation of the implementation of the NN will be given alongside the pipeline shown in Figure 5.2.

The design is limited to 3 layers, the minimum complexity that constitutes a RIM. This should limit the expected, large memory usage.

The neural network layers consist of two 3D-convolution layers surrounding a GRU. These NN layers are shown in red in Figure 5.2. Before any of the layers can be applied, the data needs to be manipulated. The data is indicated by the white blocks and the manipulation functions are indicated by the blue blocks.

5.2.1 Neural Network Inputs

This subsection discusses the manipulation of the inputs of the NN, which covers everything up to the Conv3D block in Figure 5.2. The input of the neural network is

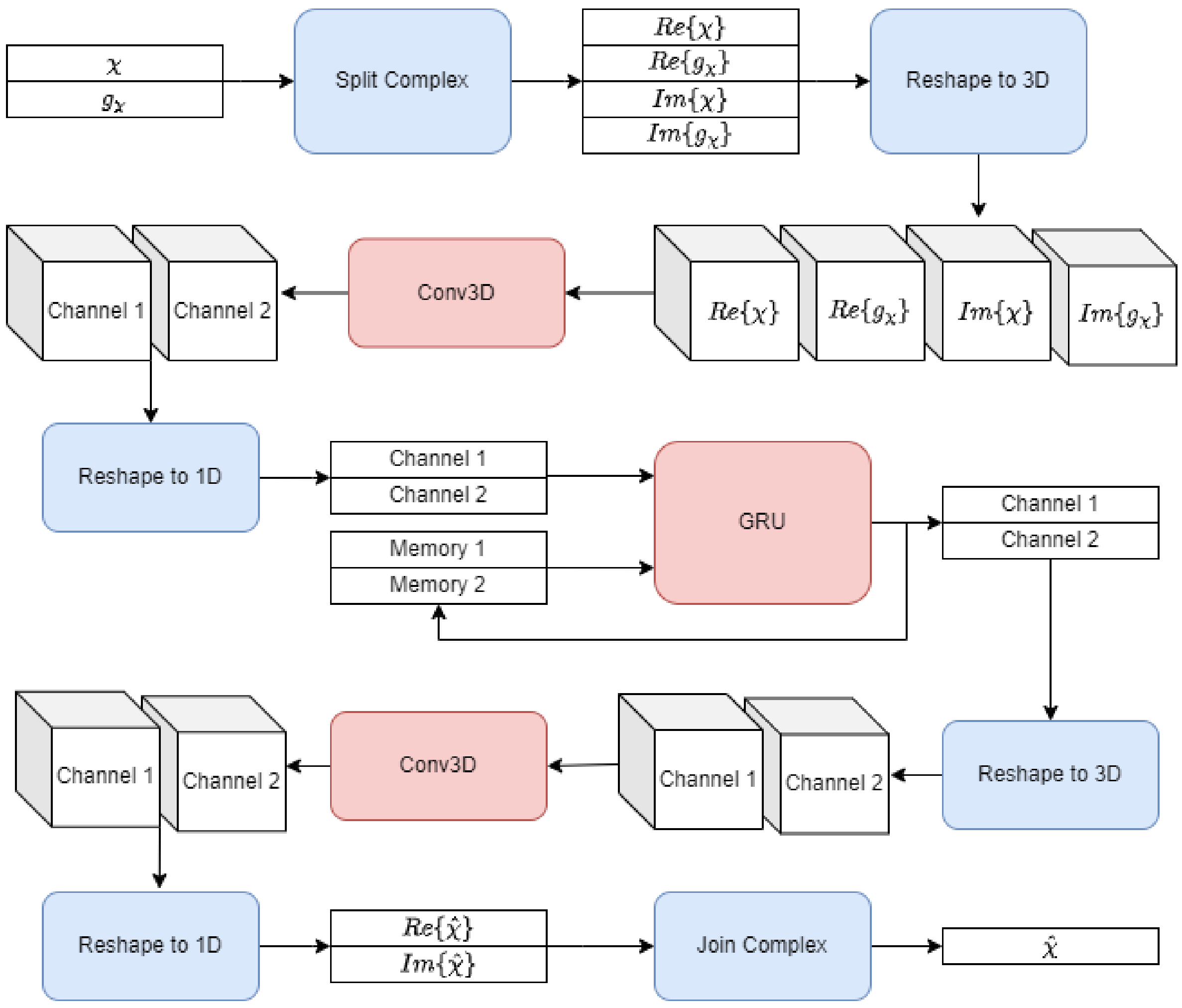


Figure 5.2: Pipeline of the deep learning part of DLE-CSI. Red blocks indicate neural network layers, blue blocks indicate data manipulation operators and white blocks depict data. Bars of data represent vectors and cubes represent 3D data.

comprised of the current estimate of the contrast function χ and its gradient g_χ. At this stage, both the estimated contrast function χ and its gradient g_χ are separate, complex, flattened vectors. Hence, represented as two thin bars of data. These vectors are first concatenated, and then split into their real and imaginary parts. This results in 4 vectors, equally split into real and imaginary data. Lastly, the 4 vectors are reshaped into a stack of 4 blocks of 3D data, represented by cubes. This stack can be considered as a 3D image with 4 channels. The complex data is split into real and imaginary components due to complex numbers not being fully supported by some network modules and/or activation functions.

5.2.2 3D Convolution Layer

The 3D convolution layers are shown in Figure 5.2 as the red Conv3D blocks. Both convolution layers have a kernel size of 3, as to take into account only very local information. Using small kernel sizes has the advantage of limiting the amount of weights and therefore the amount of memory used. Small kernel sizes are expected to still yield sufficient results due to working in the domain of the contrast function. The contrast function is dependent on the EPs, which are largely dependent on tissue type and ionic concentrations. Both these dependencies are typically local, supporting the use of small kernel sizes. The data is zero-padded before applying the convolution; to retain its original size. The first convolution layer combines the split estimate and gradient information to output two channels. The hyperbolic tangent function is used for introducing non-linearity into the network.

5.2.3 Convolution to GRU

The difference in formatting between convolution layers and GRU layers requires some data manipulation. Convolution layers are designed to make use of spatial data, hence being applied on multi-dimensional arrays. GRUs typically operate on sequences of data, which are expressed as vectors. Therefore, the 2 channels of 3D data need to be vectorised (flattened). These steps are visualised in Figure 5.2 between the first red Conv3D block and the red GRU block.

5.2.4 GRU Layer

The GRU layer is indicated in Figure 5.2 by a red GRU block. Besides the input channels fed from earlier parts of the network, the GRU also uses internally updated memory states. These memory states are also referred to as hidden states and are shown as the vectors memory 1 and memory 2 in the figure. Three different GRU layer implementations have been tested.

The first implementation features a GRU with the input size and hidden size chosen to be equivalent to the number of voxels of the imaging domain. This ensures that each voxel is updated individually based on prior results in prior iterations and the newly provided input. The two channels are presented to the GRU as a batch instead of providing both with individual weights in order to limit memory usage. Each channel does have its own corresponding memory state to ensure custom results per channel. Henceforth this implementation will be referred to as the per voxel implementation.

The second implementation is meant to further limit the GRU layer's memory burden. The 3-dimensional data before vectorisation is presented as a batch of 2D data instead. Additionally, the 2D data is split into quadrants. These quadrants are mirrored as to resemble the first quadrant as much as possible. This way, the small degree of symmetry of a human head can be leveraged to reduce the amount of weights necessary. Each quadrant per 2D plane from the batch is provided with its own memory state to ensure that distinctly different results for each quadrant can be possible. The amount of weights within a GRU primarily consists of 6 sets of size [input size $\times$ hidden size]. The change to 2D data reduces these sizes by a factor of 80 and the splitting into quadrants by another factor of 4. Hence, leading to a memory reduction of approximately $(80 * 4)^2 = 102400$ times. Henceforth this implementation will be referred to as the quadrant-based implementation.

The third implementation reduces the GRU layer's memory burden in a different manner. Instead of using a single GRU with large sets of weights, many smaller GRUs are used. The third implementation also reduces the 3D data to a batch of 2D data instead. Assuming identical input- and hidden size, the amount of weights is dominated by the quadratic relation [input size $\times$ hidden size]. Dividing the inputs over f GRUs changes this relation to $f \times (\frac{\text{input size}}{f} \times \frac{\text{hidden size}}{f})$. Note that such a division can not be performed endlessly due to additional linear relations between memory and input size, which increases in significance as the quadratic relation decreases in size. The downside of this approach is the loss of global information taken into account during the GRUs' operation. This is caused by each GRU not being able to access voxel information outside the provided input. Henceforth this implementation will be referred to as the local patches implementation.

5.2.5 GRU to Convolution

The data manipulation done between the convolution layer and GRU has to be reversed in order to provide an adequate output to the next convolution layer. For the per voxel implementation this entails reshaping the vectors back into two channels of 3D image data, as visualised in Figure 5.2. For the quadrant-based implementation, this entails reshaping the vector back into a stack of 2D data, the quadrants getting reverse-mirrored and joined up into entire 2D planes. The stack of 2D data is then treated as 3D data again for the convolution layer input. For the local patches implementation, this entails rejoining the GRU outputs and reshaping them into two channels of 3D image data. In all cases, it leads to two channels of 3D-shaped data which the 3D convolution layers can operate on.

5.2.6 Final convolution layer

In the case that the network is used in the replacement update manner instead of the additive update manner, the final convolution layer is no longer followed by a hyperbolic tangent function. The hyperbolic tangent function would limit the possible values of the updated contrast $\hat{\chi}$ to -1 and 1, while the true contrast may lie outside this range.

5.2.7 Output

The output of the last convolution layer needs to be processed so it conforms with the inputs of the CSI algorithm. The real and imaginary channels rejoin into a single complex-valued channel which is then vectorised (flattened). It is then passed on to the rest of the CSI algorithm as the new estimate $\hat{\chi}$.

5.3 Training Strategies

For training the network, the dataset is split into 140 training sets and 40 sets for evaluation.

The number of iterations run by DLE-CSI is limited to 10 and the amount of epochs is set to 3. These amounts should limit training time to a manageable amount, while expected to be enough to assess training progress.

Initially, the network is trained and evaluated on noiseless data. This enables us to assess the network's feasibility before extending its assessment to more realistic inputs.

The contrast is masked throughout the use of DLE-CSI. Meaning that the contrast outside the object domain or where the head is not present is ignored. This ensures that no weight is given to learning reconstructed EPs outside of the geometry of the head.

An ADAM optimizer and a mean square error (MSE) loss, are used for training the network, with a learning rate of $1e-3$. The MSE loss is defined as follows:

$$MSE(x,\hat{x}) = \frac{1}{N}\sum_{i=0}^{n}(x_i - \hat{x}_i)^2. \tag{5.4}$$

Here x and $\hat{x}$ denote the true and the network's output respectively. The subscript i denotes the individual voxel, whereas N represents the total amount of voxels.

The MSE loss is applied to the algorithm's final estimate of the contrast, χ, and its ground truth value. The ground truth of χ is constructed using the ground truth values of the conductivity and permittivity present in the data sets.

5.4 Tests & Results

The first two tests are performed using the replacement update manner. The subsequent tests use the additive update manner instead. Note that the permittivity results displayed in any of the figures represent the relative permittivity and are therefore without unit.

5.4.1 Per Voxel Implementation Assessment

The Per Voxel implementation was observed to be infeasible for the size of input data used within the project. The amount of memory usage of a GRU layer with input- and memory states per voxel was severely underestimated. A GRU has multiple sets of weights and biases. Considering that the memory state size and input size are equal, the GRU features 6 sets of weights of the shape $[inputsize, inputsize]$. The image domain is of the shape $[128, 128, 80]$. Meaning, that the imaging domain features roughly $128*128*80 \approx 1.3*10^6$ voxels. Using the Per Voxel implementation would mean using more than $6*1.3*10^6*1.3*10^6$ weights. Accounting for more than 10 TB of data. This makes this approach infeasible.

5.4.2 Test 1: Simultaneous Iteration Training

The simultaneous iteration training (SIT) uses a fixed amount of iterations per reconstruction for the entire duration of the training process. An arbitrary amount of iterations is performed every batch before the loss is calculated and the weights are updated by backpropagation. Hence, the network tries to learn optimal weights for every iteration simultaneously. Algorithm 1 provides an abbreviated pseudo-algorithm of the simultaneous iteration training.

Algorithm 1 A pseudo algorithmic description of Simultaneous Iteration Training

```
for $e$ in Epochs do
    for Batch in Training Set do
        for $i$ in Max Iterations do
            Compute $g_\chi$ according to CSI
            Update $\chi$ using the NN
            Update $F$ according to CSI
            if $F \leq$ tolerance then Break
            end if
        end for
        Compute MSE Loss
        Update NN weights by backpropagation
    end for
end for
```

The fixed amount of iterations has been set to 10. 10 iterations are expected to be sufficient to assess whether the network adjusts its behaviour to different iteration steps while being small enough to prevent excessive training times.

5.4.2.1 Results Test 1 using 2D Quadrant Based Implementation

The average MSE losses for training and testing the 2D Quadrant-based implementation under simultaneous iteration training for 5 epochs and 10 iterations are shown in Figure 5.3. The average MSE loss values are displayed on the y-axis and the epoch numbers are displayed on the x-axis. The blue line demarcates the average training losses and the orange line demarcates the average test losses.

The most notable result shown in Figure 5.3 pertains to the average test losses being lower than the average training losses at every epoch. This means that the trained network is able to reconstruct the unseen test data to a higher degree than the training data. Within a single epoch, this could be explained by the network performing to a similar degree on both training and test data; likely due to high similarity between the datasets. If the performance on training and test data at the end of training is similar, then a higher average training loss is to be expected, as the poorer performance at the beginning of the training phase skews the average loss upwards. However, one would then expect the training loss at the next epoch to be at least equal to or lower than the average test loss of the previous epoch. Instead, Figure 5.3 displays the average training loss reaching similar values to the initial average test loss after 5 epochs.

The second thing of note is the high MSE loss values. The average values of the contrast maps are typically single digit. Average MSE loss values around 3000 indicate

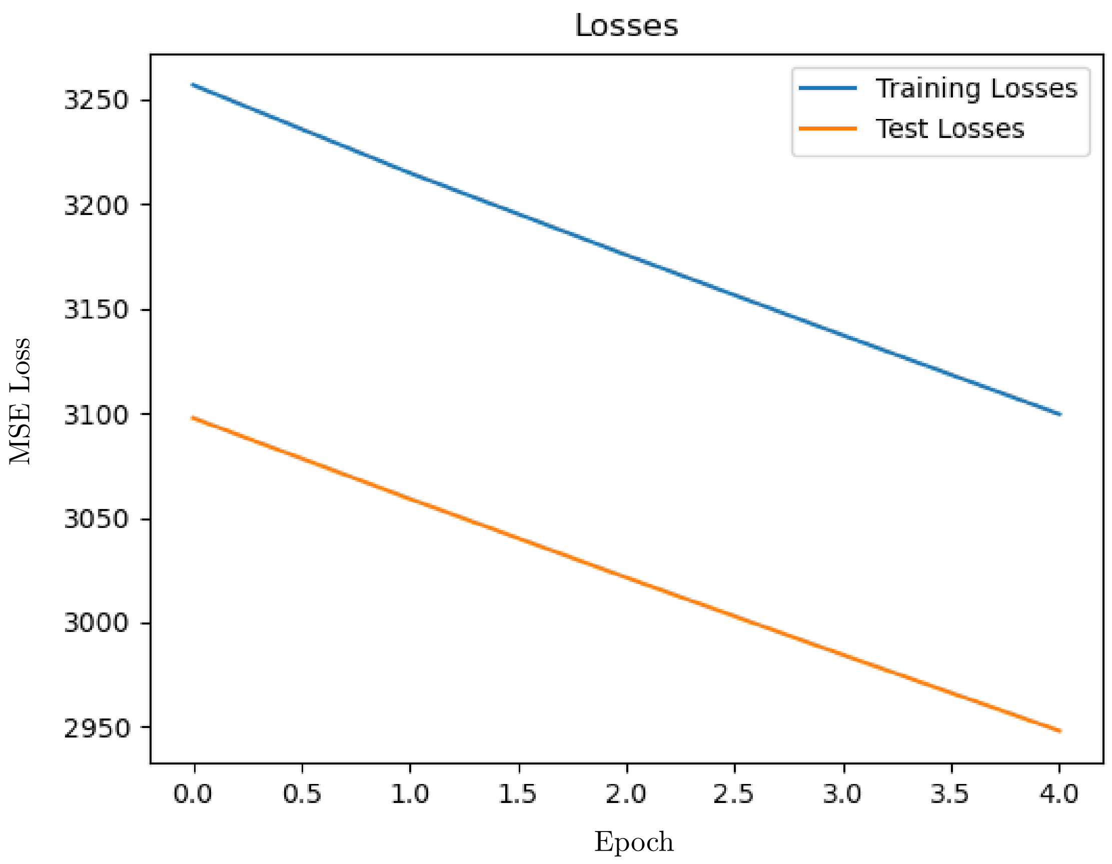


Figure 5.3: Average Training- and test losses of simultaneous iteration training using the 2D quadrant-based implementation. Max iterations is set to 10 and the training was performed for 5 epochs.

The leftmost plots in Figure 5.4 display CSI's cost functional (Equation 2.55) for Test 1 using the 2D Quadrant-based implementation. Whereas the MSE loss measures the performance of reconstructing the EPs through χ, the cost functional F measures the performance of updating both χ and w at every iteration. Hence, the cost functional provides insight into DLE-CSI's progression between iterations. The cost functional values are displayed on the y-axis and the epoch numbers are displayed on the x-axis. The different colours represent the iterations.

The upper row and bottom row of Figure 5.4 concern the training and testing, respectively. The dark blue line indicating Iteration 0, shows a strong decline over the 5 epochs; converging towards a value of 0.5 for both training and testing. A cost functional value of approximately 0.5 is also observed as the initial cost functional value of regular 3D-CSI. Hence, the first iteration's reconstruction performance after 5 epochs can be considered similar to the homogeneous guess used in both DLE-CSI and CSI.

Iteration 1 indicated by the orange line, shows a slight increase in value for both training and testing over the duration of 5 epochs. Indicating a decrease in reconstruction performance at Iteration 1 at increased training durations. Though at all epochs Iteration 1's cost functional value remains smaller than Iteration 0's value, maintaining an improvement between iterations.

The succeeding iterations, Iteration 2 up until Iteration 9, progressively decrease in cost functional values for both training and testing over the duration of 5 epochs. Each iteration also improves in value compared to its preceding iteration. Hence, DLE-CSI improves upon its reconstruction performance at every iteration. However, the values remain close to 0.5; yielding a similar performance as the homogeneous initial guess. Additionally, the cost functional values of the testing phase slightly outperform those from the training phase, which is in line with the behaviour of the MSE losses.

The cost functional F can be subdivided into sub-cost functionals FB and FE. FB and FE represent the right- and left additive terms of Equation 2.55, respectively. FB is a function of the contrast source FB(w), whereas FE is a function of both the contrast function and contrast source FE(χ, w). In Figure 5.4 the middle column shows FB and the rightmost column shows FE, effectively deconstructing F. Top and bottom still represents training and testing, respectively. The y-axis shows the (sub-)functional values and the x-axis shows the epoch numbers.

Figure 5.4 shows how after the first epoch, the cost functional F is dominated by the contrast source dependent FB. Hence, the contrast source's updates through gradient descent seem unable to converge as quickly as the NN computed updates of the contrast function. This is considered unexpected behaviour because the contrast source's gradient descent updates are computed based on the updated contrast function. It was therefore expected that the faster converged contrast function would propagate to a better gradient descent of the contrast source.

Iteration 0 in the middle column of Figure 5.4, displays a constant FB value of approximately 0.46 for both the training and testing phase at every epoch. Most notable in the plots of FB, is Iteration 0's lower value than the values of the succeeding iterations. Both of these behaviours originate from the contrast source's first update being based on the initial guess of the contrast function. As the initial guess is unaffected by the NN, FB's first value is independent of training the NN. The higher values of FB in successive iterations display the CSI part's reaction to the NN output updates of the contrast function.

After the initial spike in the value of FB at Iteration 1, the first decrease in value is observed at different iterations depending on the epoch number. The higher the epoch number the later this first decrease is observed. Despite this observed behaviour, the values of each iteration do decrease in every successive epoch, with the exception of Iteration 0. Hence, extended training does yield a general improvement in reconstruction.

The values of FE displayed in the rightmost column of Figure 5.4, show behaviours opposite to those of FB's values. In training, Iteration 0 starts with a relatively high value of more than 0.6, decreasing considerably to values below 0.1 as the epoch number increases. Testing displays similar behaviour but with lower values overall. Hence, training the NN effectively improves Iteration 0's updates.

The successive iterations increase in FE values as the number of epochs increases, worsening as the training progresses. The lowest average FE values are observed in the first epoch. The adverse effect of training to iterations succeeding Iteration 0, might be related to the NN having to learn optimal weights for every iteration simultaneously. Alternatively, it could be DLE-CSI working as intended. The gradients used to update both the contrast source and contrast function, are gradients of the cost functional with respect to the contrast source or contrast function. Meaning, that if FB dominates F, the gradient reflects this and pushes the contrast source and contrast function to decrease FB. These FB prioritised updates could lead to an increase in FE as long as F decreases. Hence, only if worsening results of FE persist when it dominates F, does it confirm to pose a problem.

For all epochs, the FE value initially decreases but starts to increase in later iterations. The point at which the first increase is observed occurs later as the amount of epochs used in training increases.

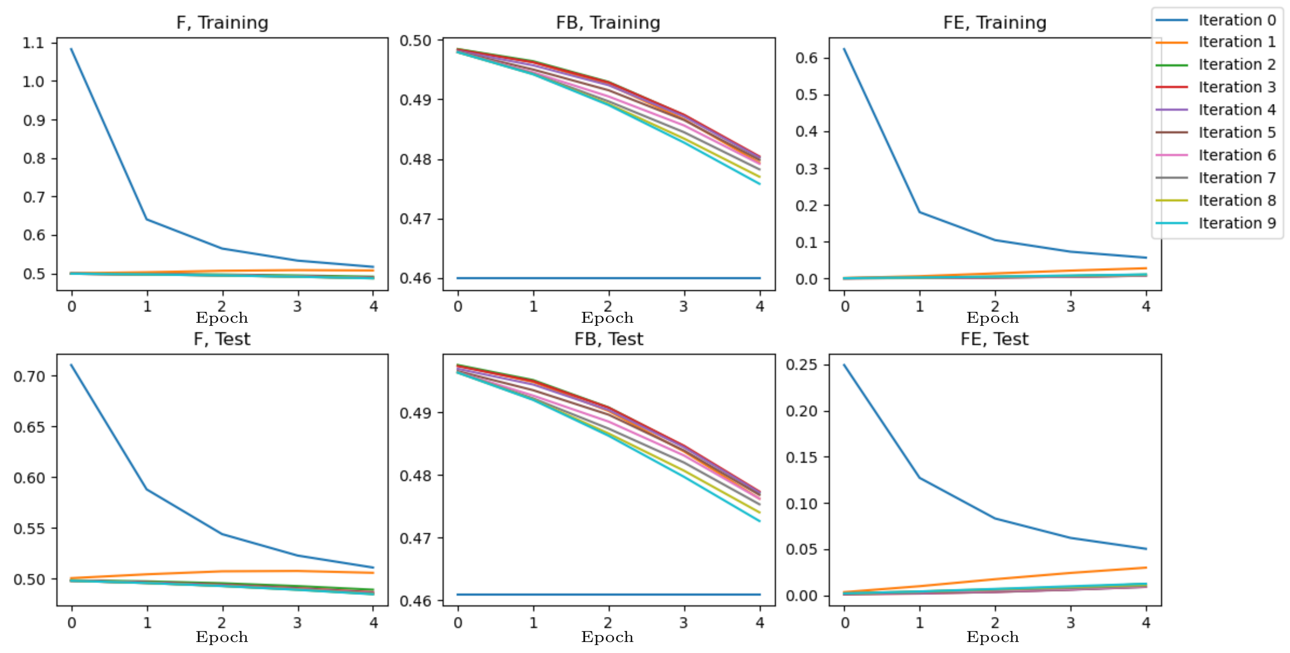
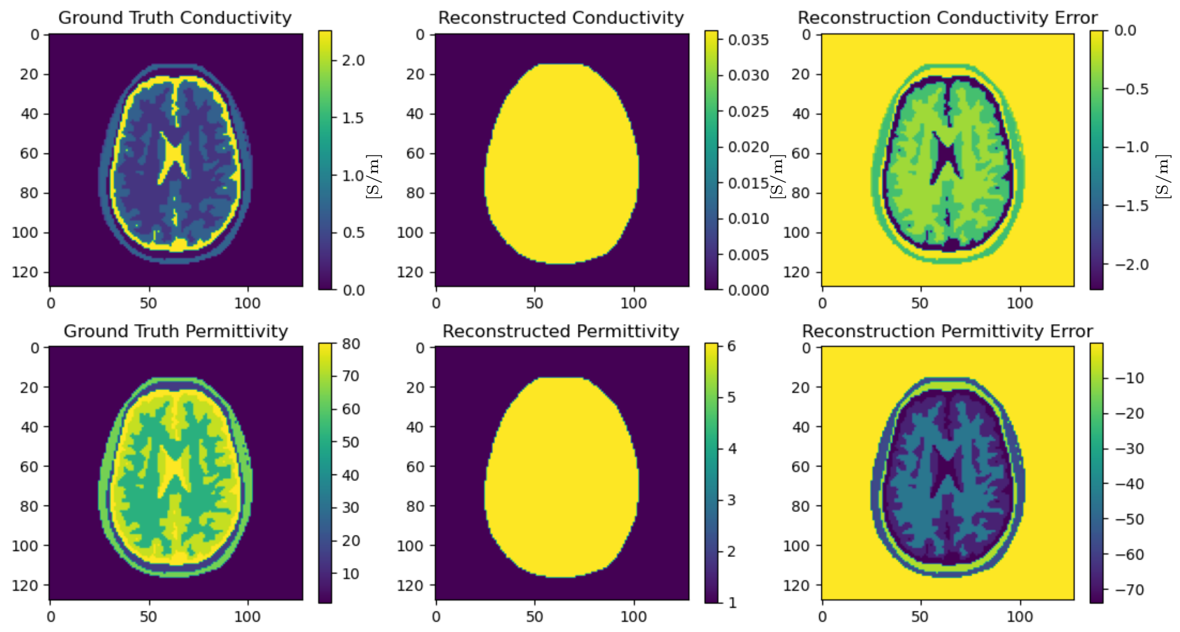


Figure 5.4: Average Cost Functional values at every iteration and epoch of simultaneous iteration training using the 2D Quadrant-based implementation. Max iterations is set to 10 and the training was performed for 5 epochs.

To further analyse possible causes for the high MSE losses and cost functional values, an inspection of reconstructed EP slices was performed. Figure 5.5 shows an example of such a reconstructed set of EPs in the middle column. The left- and rightmost columns show the ground truth and reconstruction error, respectively. The 2D example shows the transverse plane of the brain at the median slice.

The reconstructed EP images show a homogeneous EP value. The only difference between the reconstruction and the homogeneous initial guess is their amplitudes. This shows that this DLE-CSI implementation is currently unable to apply a different structure to the homogeneous map. Hence in training, the network tries to learn weights that find an optimal amplitude for the homogeneous map rather than applying structural changes. The MSE losses and cost functional values remain high due to the large error caused by the lack of reconstructing the head's internal structure.



Ground Truth Conductivity
Reconstructed Conductivity
Reconstruction Conductivity Error
Ground Truth Permittivity
Reconstructed Permittivity
Reconstruction Permittivity Error
[S/m]

Figure 5.5: An example of electrical properties reconstructed by DLE-CSI using the 2D quadrant-based implementation and trained for 5 epochs using the simultaneous iteration approach with a maximum of 10 iterations. The reconstruction is performed on a simulated model of the human head, of which a transversal slice is shown. The leftmost images show the ground truth EP values to be reconstructed, the middle column images show the reconstructed EPs and the rightmost images show the difference between the reconstruction and the ground truth.

5.4.2.2 Results Test 1 using Local Patches Implementation

The average MSE losses for training and testing the 2D Quadrant-based implementation under simultaneous iteration training for 5 epochs and 10 iterations are shown in Figure 5.6. The average MSE loss values are shown on the y-axis and the number of epochs is shown on the x-axis. The blue line and orange line represent the average training losses and the average test losses, respectively.

The Local Patches implementation shows the same behaviour as the 2D Quadrant-Based implementation. The Local Patches implementation only has slightly higher values by a value of approximately 25.

Under Simultaneous Iteration Training and using the Local Patches implementation, Figure 5.7 displays the resulting average cost functional and sub-cost functionals F, FB, and FE from left to right, respectively. The top row and bottom row differentiate between training and testing. The (sub-)cost functional values are shown on the y-axis

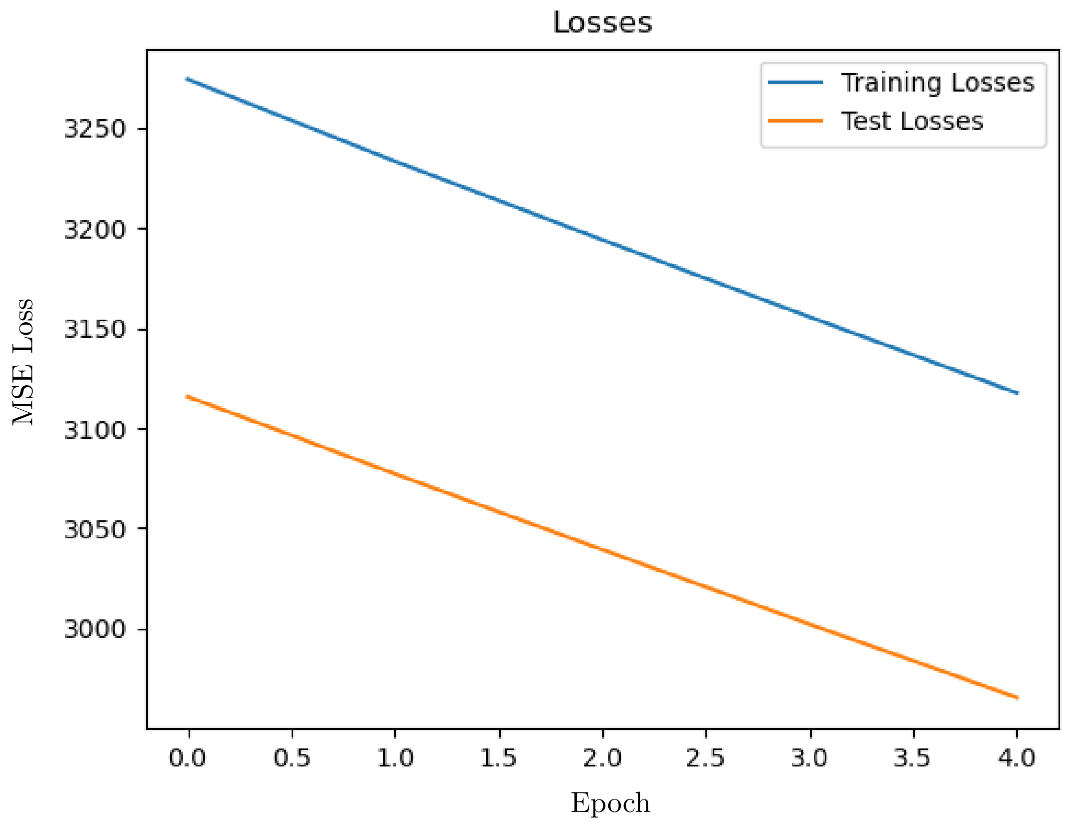


Figure 5.6: Average Training- and test losses of simultaneous iteration training using the Local Patches implementation. Max iterations is set to 10 and the training was performed for 5 epochs.

and the number of epochs is shown on the x-axis. The different colours demarcate the iteration number.

The Local Patches implementation results are nearly identical to those of the 2D Quadrant-based implementation. The only noticeable difference is observed in the starting value of the average cost functional of Iteration 0 which is slightly higher.

Figure 5.8 shows from left to right, an example set of the ground truth, reconstruction and reconstruction error of EPs under the simultaneous iteration training and local patches implementation. The 2D example shows the transverse plane of the brain at the median slice. The local patches implementation displays the same homogeneous reconstruction results as the 2D quadrant-based implementation. Only the reconstructed permittivity amplitude is slightly lower in value.

5.4.3 Test 2: Per Iteration Training

The Per Iteration Training (PIT) initially sets the max amount of iterations to 1 and increases it every subsequent epoch. This way the network does not require to learn optimal weights for every iteration all at once. The goal of this training approach is to improve upon the results of the Simultaneous Iteration Training. Simultaneous Iteration Training resulted in cost functional values at certain iterations not improving when trained. The first iteration(s) have shown to improve most significantly before. Having these iterations trained first, reducing their cost functional values, should bring them closer to the initial cost functional values of the latter iterations. Closer cost functional

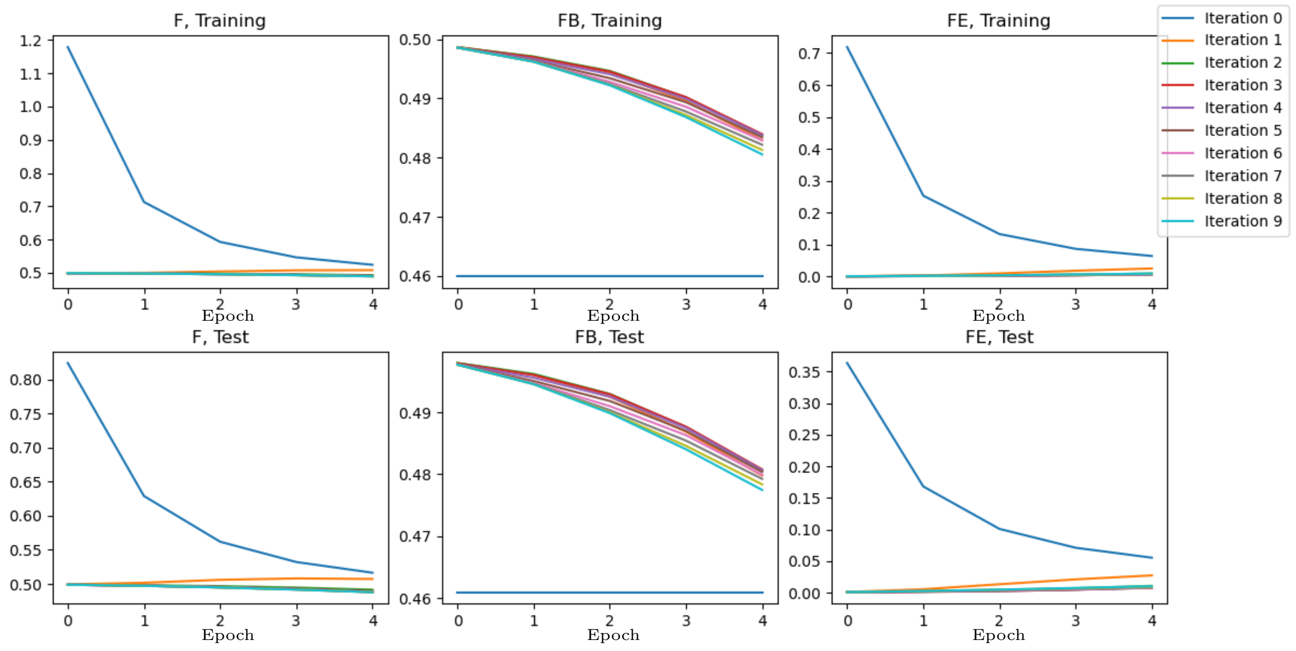


Figure 5.7: Average Cost Functional values at every iteration and epoch of simultaneous iteration training using the Local Patches implementation. Max iterations is set to 10 and the training was performed for 5 epochs.

values would mean more similar amounts of impact on the final reconstruction result. Hence, improving these latter iterations at that point is likely to be valued higher than when training them simultaneously with the first iterations which at that point can improve to a higher extent.

A pseudoalgorithmic description Per Iteration Training is shown in Algorithm 2.

Algorithm 2 A pseudo algorithmic description of Per Iteration Training

```
for $e$ in Epochs do
    for Batch in Training Set do
        for $i$ in $e+1$ do
            Compute $g_\chi$ according to CSI
            Update $\chi$ using the NN
            Update $F$ according to CSI
            if $F \leq$ tolerance then Break
            end if
        end for
        Compute MSE Loss
        Update NN weights by backpropagation
    end for
end for
```

The amount of epochs has been set to 5, resulting in a maximum amount of iterations of 5.

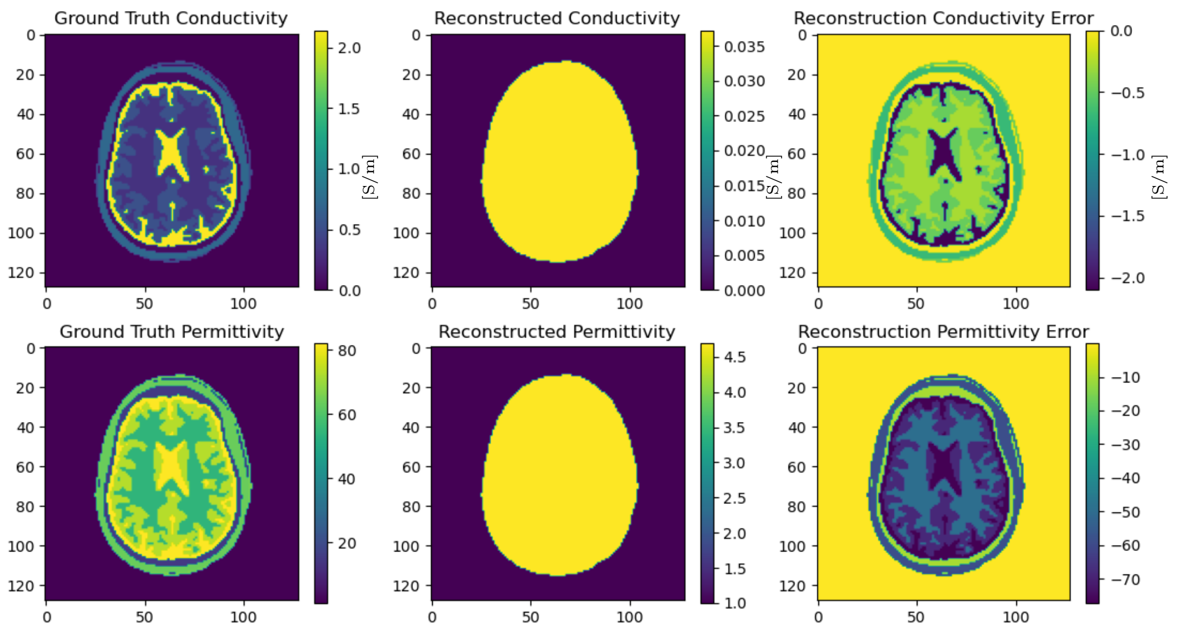

Ground Truth Conductivity
Reconstructed Conductivity
Reconstruction Conductivity Error
Ground Truth Permittivity
Reconstructed Permittivity
Reconstruction Permittivity Error
[S/m]

Figure 5.8: An example of electrical properties reconstructed by DLE-CSI using the Local Patches implementation and trained for 5 epochs using the simultaneous iteration approach with a maximum of 10 iterations. The reconstruction is performed on a simulated model of the human head, of which a transversal slice is shown. The leftmost images show the ground truth EP values to be reconstructed, the middle column images show the reconstructed EPs and the rightmost images show the difference between the reconstruction and the ground truth.

5.4.3.1 Results Test 2 using 2D Quadrant Based Implementation

The average MSE losses for training and testing the 2D Quadrant-based implementation under iteration per epoch training for 5 epochs and 1 up until 5 iterations are shown in Figure 5.9. The average MSE loss values are shown on the y-axis and the number of epochs is shown on the x-axis. The average training losses are shown in blue and the average test losses are shown in orange.

The behaviour of the average losses remains very linear and compared to the average losses in Figure 5.3, the amplitude of the losses is slightly higher. Hence in terms of losses, the iteration per epoch approach performs worse than the simultaneous iteration training for the 2D Quadrant-based implementation.

The average cost functional values and sub-cost functional values are shown in Figure 5.10 for both training and testing in the case of iteration per epoch training using the 2D quadrant-based implementation. The top row shows the (sub-)cost functionals during training and the bottom row during testing. The y-axis displays the (sub-)cost functional values and the x-axis the number of epochs. The different colours indicate the iteration numbers.

Iteration 0 in the leftmost plots displaying F, starts and ends at a higher value than in the results of simultaneous iteration training. Similarly, the subsequent it-

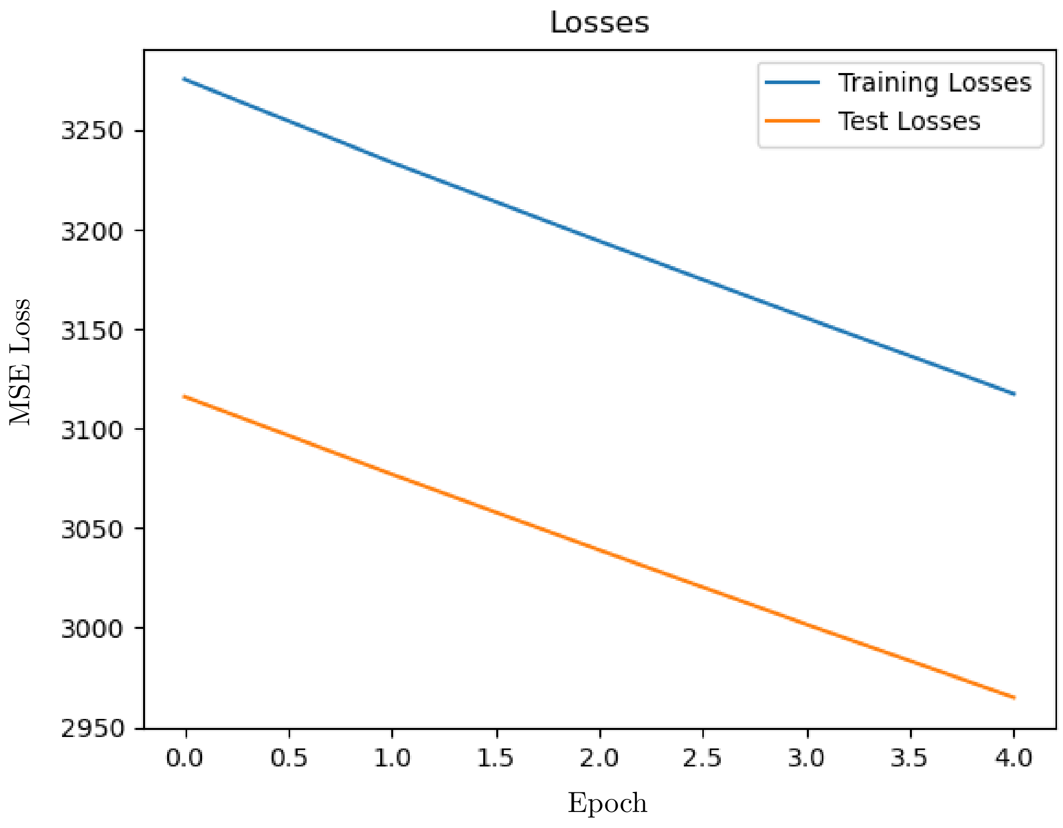


Figure 5.9: Average Training- and test losses of iteration per epoch training using the 2D quadrant-based implementation. Max iterations progresses to 5 and the training was performed for 5 epochs.

erations also perform comparatively worse than their simultaneous iteration training counterpart.

The sub-cost functionals FB and FE, display the same behaviour as in the simultaneous iteration training with 2D quadrant-based implementation but with slightly higher values.

Figure 5.11 shows from left to right: the ground truth, reconstruction and reconstruction error of the EPs of a 2D example of the transverse plane of the brain at the median slice. The reconstructed EPs in the middle column of Figure 5.11 show the same homogeneous results as the simultaneous iteration training, with a deviation in permittivity values of about 1.5. Hence, no meaningful improvement has been observed.

5.4.3.2 Results Test 2 using Local Patches Implementation

The results shown in Figures 5.12 5.13 and 5.14 follow the same principles as the 2D quadrant-based implementation. The average losses, cost functionals and reconstructions have a highly similar shape with a slightly worse amplitude.

The iteration per epoch training has a slightly worse performance than the simultaneous iteration training overall. Its only advantage is shorter training times due to the lower amount of iterations used in earlier epochs. The consistent absence of DLE-CSI's ability to reconstruct the brain's complex structure and move away from a homogeneous guess, deems these implementations to be severely inadequate. Therefore, a conclusion on the comparison of these two training approaches should not be

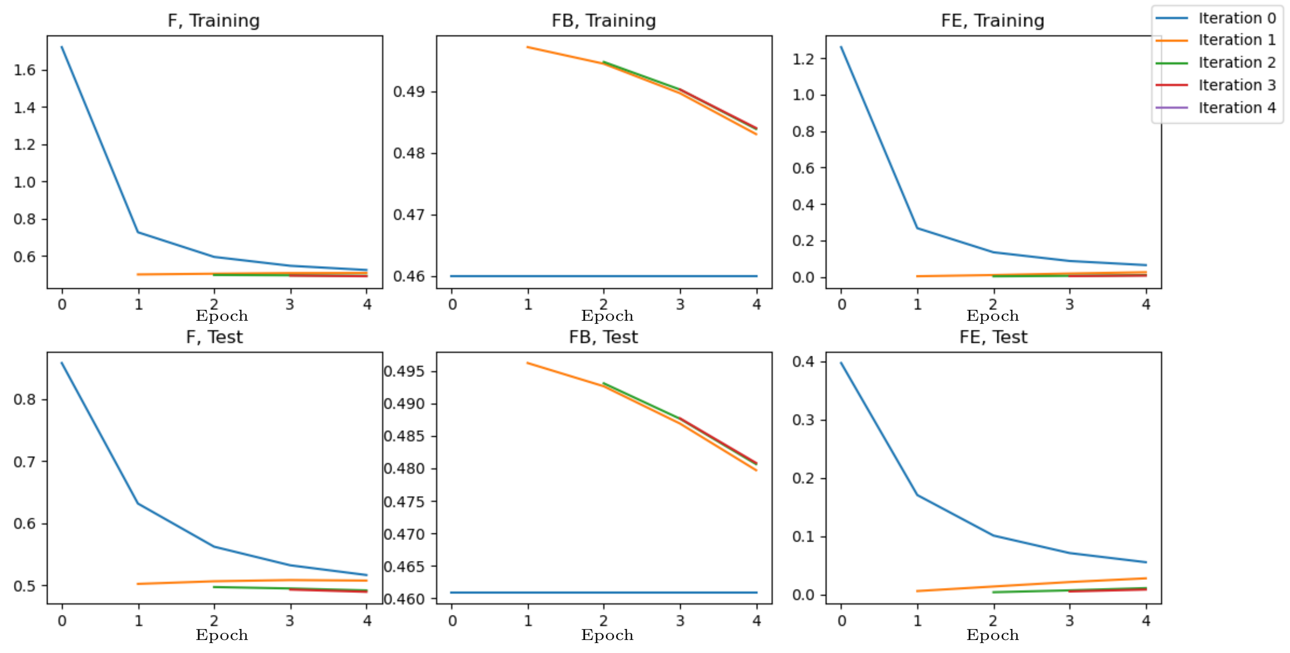


Figure 5.10: Average Cost Functional values at every iteration and epoch of iteration per epoch training using the 2D quadrant-based implementation. Max iterations progresses to 5 and the training was performed for 5 epochs.

limited to their effects on inadequate implementations.

5.4.4 Test 3: SIT With Additive Update Manner

Test 3 entails simultaneous iteration training with a switch between the update approach used in the integration of the NN into CSI. For Test 3, the additive update manner is used instead of the replacement update manner. Only the local patches-based implementation has been tested with the additive update manner due to time constraints and the higher potential for network expansion with the reduced size of this approach. The training duration was set to 10 epochs. Unfortunately, the training under Test 3 has been cut off early so the 10th epoch is missing from the results.

5.4.4.1 Results Test 3

Figure 5.15 shows the average training- and average test losses under Test 3. The average MSE loss values are shown on the y-axis and the number of epochs is shown on the x-axis. The blue line represents the average training losses and the orange line represents the average test losses. The losses no longer display the very linear behaviour, decreasing more sharply at the start instead. The amplitude of the losses is lower by a value of approximately 1000 compared to any implementation of the replacement update manner. The losses remain significant, considering a standard deviation of $\sqrt{1225} = 35$, while average contrast values remain single digit. The test losses remain lower than the training losses.

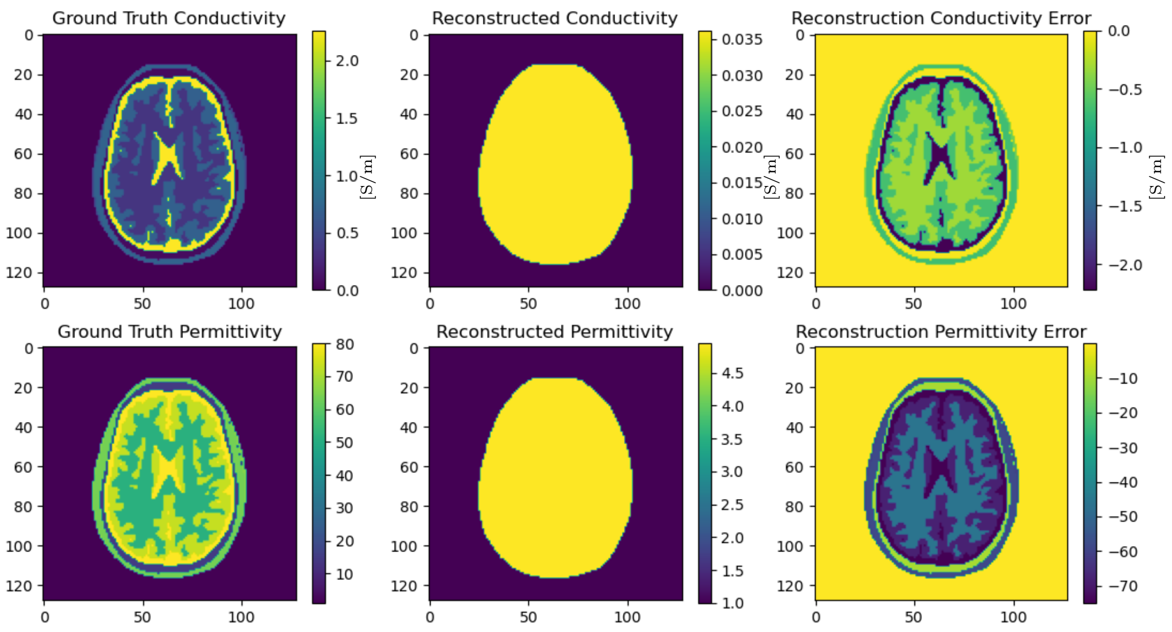

Ground Truth Conductivity
Reconstructed Conductivity
Reconstruction Conductivity Error
Ground Truth Permittivity
Reconstructed Permittivity
Reconstruction Permittivity Error
[S/m]

Figure 5.11: An example of electrical properties reconstructed by DLE-CSI using the 2D quadrant-based implementation and trained for 5 epochs using the iteration per epoch approach with a progressive maximum of 5 iterations. The reconstruction is performed on a simulated model of the human head, of which a transversal slice is shown. The leftmost images show the ground truth EP values to be reconstructed, the middle column images show the reconstructed EPs and the rightmost images show the difference between the reconstruction and the ground truth.

Figure 5.16 shows the average (sub-)cost functional values at every iteration and epoch under Test 3. The top and bottom rows differentiate between training and testing respectively. From left to right the (sub-)cost functionals F, FB and FE are shown, with the number of epochs on the x-axis and the colour indicating the iteration number. Unlike the previous tests' F values, these F values decrease far below the 0.5 point. Indicating that the additive update manner allows improvement beyond the homogeneous initial guess. The extent to which the F values decline decreases every iteration. Such behaviour is similar to the regular CSI algorithm. The FB values follow the pattern of F values, declining at a reduced rate as iterations progress. The FE values on the other hand, initially increase and only decrease from iteration 4 onwards. This initial increase is likely caused by the update's priority of reducing the relatively higher FB values. Since the gradients used in the update computations reflect these imbalances.

The (sub-)cost functional values improve at a much smaller degree per epoch in comparison to each iteration step.

Figure 5.17 shows an example of an EP reconstruction using DLE-CSI, test 3. The top images show the conductivity and the bottom images the permittivity. From left to right the images represent the ground truth, reconstruction and reconstruction error.

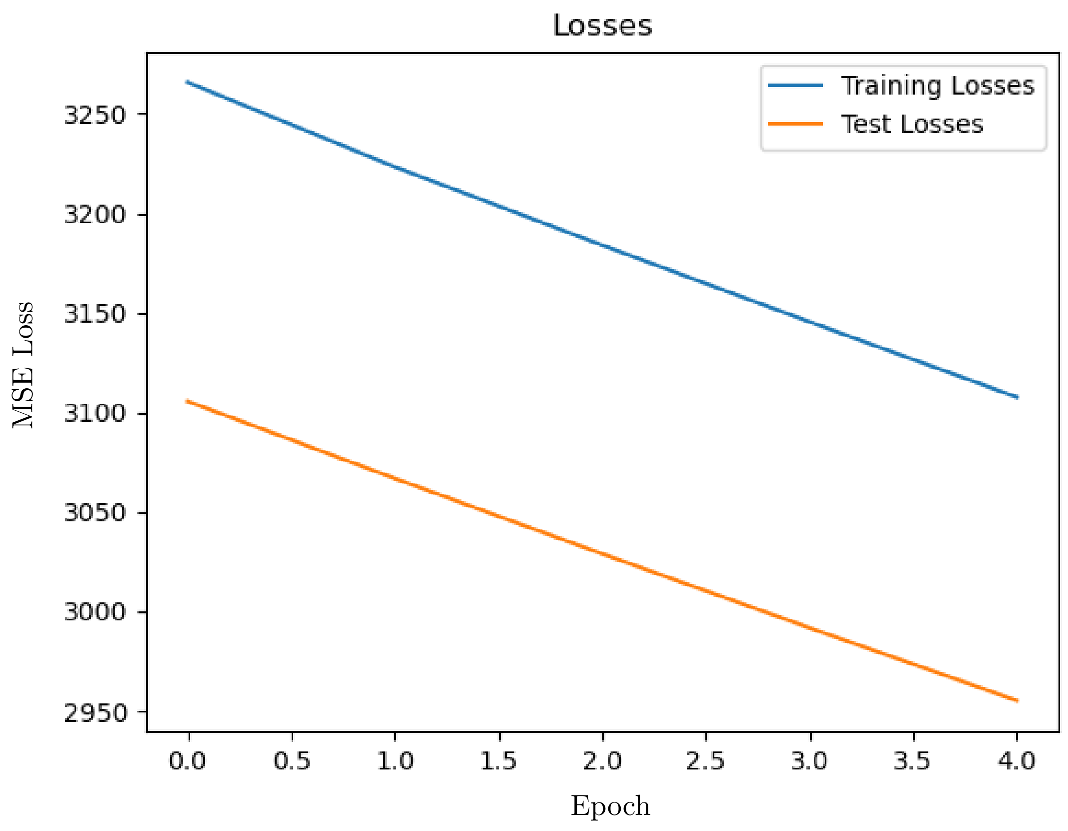


Figure 5.12: Average Training- and test losses of iteration per epoch training using the local patches implementation. Max iterations progresses to 5 and the training was performed for 5 epochs.

The 2D example shows the transverse plane of the median slice of a brain. The EP reconstruction images showcase how under Test 3 the geometry is changed beyond the uniformity of the results under previous tests. The results, however, still show major differences in terms of tissue geometry and amplitude values.

5.4.5 Test 4: PIT with Additive Update Manner

In Test 4 per iteration training is performed with the additive update manner instead of the replacement update manner. The training duration was set to 10 epochs and each epoch has the corresponding amount of iterations as the epoch number.

5.4.5.1 Results Test 4

Figure 5.18 shows the average training- (in blue) and average test losses (in orange) under Test 4. The x- and y-axis display the number of epochs and the average MSE loss values, respectively. Compared to the results under Test 3, the initial losses are higher. However, at the 9th epoch the losses are slightly lower for the training phase and approximately equivalent for the testing phase. The higher losses at lower epoch amounts are likely attributed to the smaller amount of iterations used at these epochs. A higher amount of iterations is likely to improve the reconstruction and hence, reduces the losses. The higher rate of improvement in losses at the final epochs indicates that PIT might outperform SIT at higher training volumes. However, extended training or other metrics are required to truly validate this hypothesis.

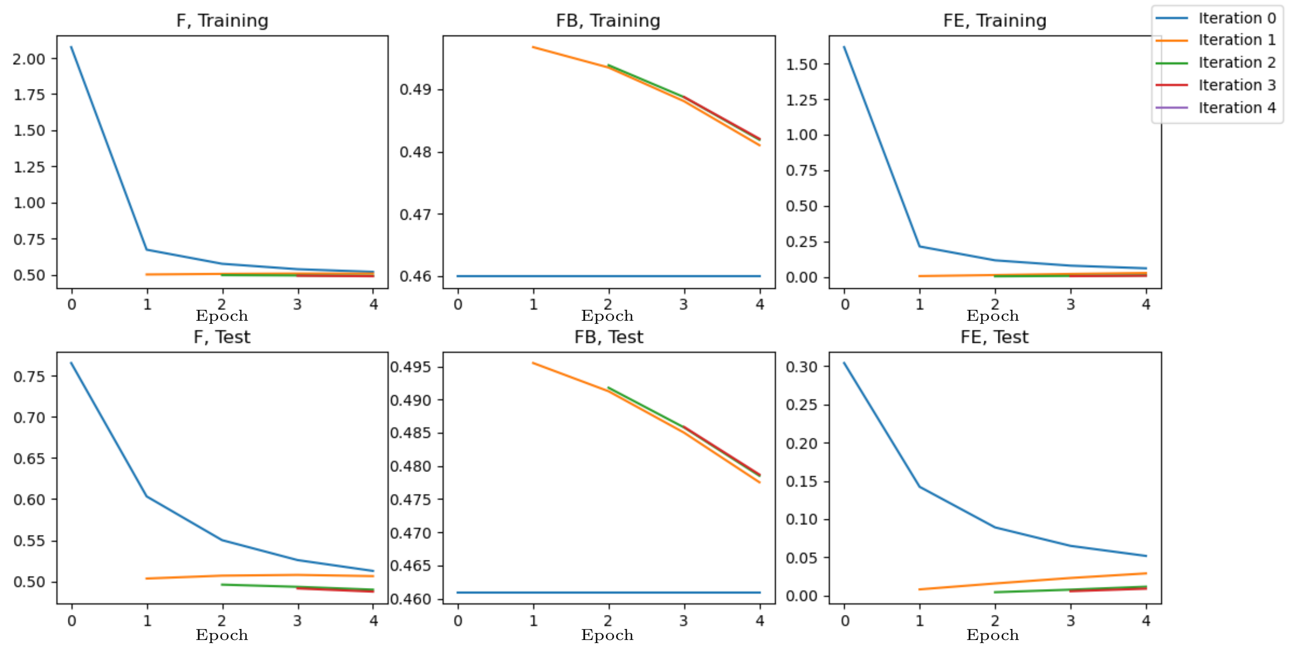


Figure 5.13: Average Cost Functional values at every iteration and epoch of iteration per epoch training using the local patches implementation. Max iterations progresses to 5 and the training was performed for 5 epochs.

Figure 5.19 shows from left to right the average cost functional F, average sub-cost functional FB and average sub-cost functional FE. With the top row showing these (sub-)cost functionals in the training phase and the bottom row during the testing phase. The number of epochs is shown on the x-axis and the (sub-)cost functional values are shown on the y-axis, with the different colours indicating the iteration number. Compared to Test 3, these cost functional results under Test 4, show no significant difference. Still, the improvement in values is dominated by iterations rather than the improvements per epoch. This indicates that an increased iteration amount is likely to contribute more to an improved reconstruction compared to an increased epoch amount.

Figure 5.17 shows an example of an EP reconstruction using DLE-CSI, test 4. The top images show the conductivity and the bottom images the permittivity. From left to right the images represent the ground truth, reconstruction and reconstruction error. This 2D example shows the transverse plane of the median slice of a brain. Compared to Test 3's results, an increased amount of patches is observed in the reconstructed conductivity. Whereas, in the permittivity, a decreased amount of patches is observed. Hence, the amount of tissue structure reconstructed within the conductivity map has increased, while within the permittivity it has decreased.

The small improvements in both Test 3 and Test 4 in terms of reconstructing tissue structure, indicate the additive update manner's superiority over the replacement update manner.

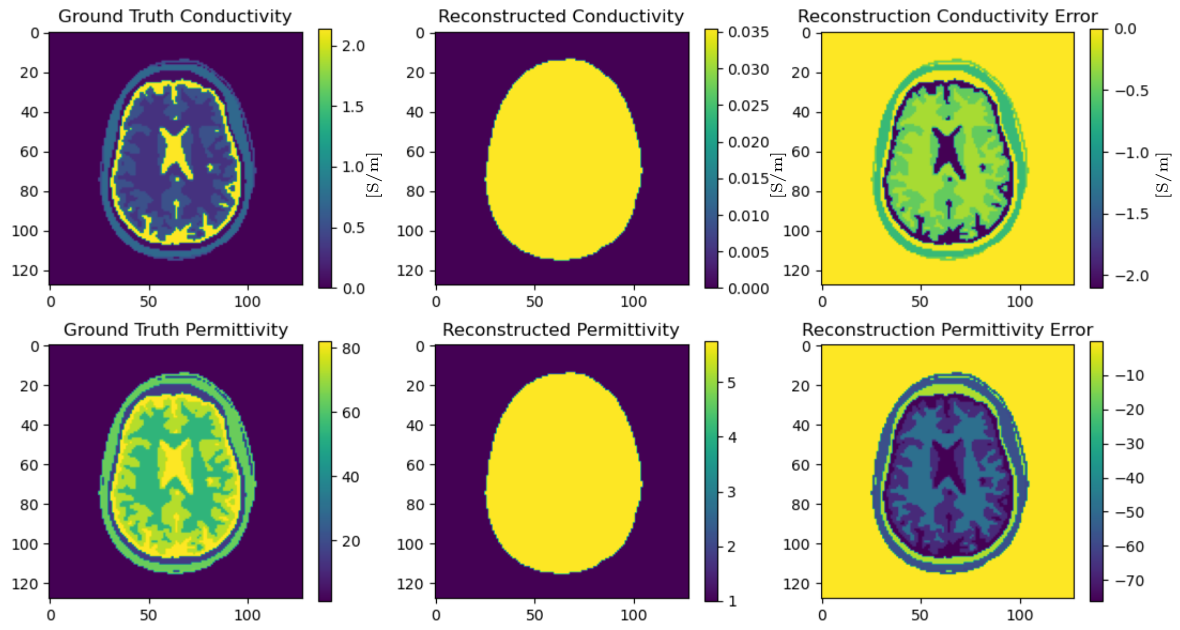


Figure 5.14: An example of electrical properties reconstructed by DLE-CSI using the local patches implementation and trained for 5 epochs using the iteration per epoch approach with a progressive maximum of 5 iterations. The reconstruction is performed on a simulated model of the human head, of which a transversal slice is shown. The leftmost images show the ground truth EP values to be reconstructed, the middle column images show the reconstructed EPs and the rightmost images show the difference between the reconstruction and the ground truth.

5.4.6 Test 5: Network Depth Expansion

In Test 5 the neural network depth has been expanded to investigate whether this has any positive impact on the EP reconstruction. Essentially the previous network design is doubled, consisting of the following layers:

- 3D Convolution Layer
- Local Patches GRU Layer
- 3D Convolution Layer
- 3D convolution Layer
- Local Patches GRU Layer
- 3D convolution Layer.

The data manipulation in between layers is duplicated with the exception of the output manipulations after the second convolution layer, which has naturally been delayed until after the final convolution layer. Additionally, the data in between the second and third convolution layers is kept as is.

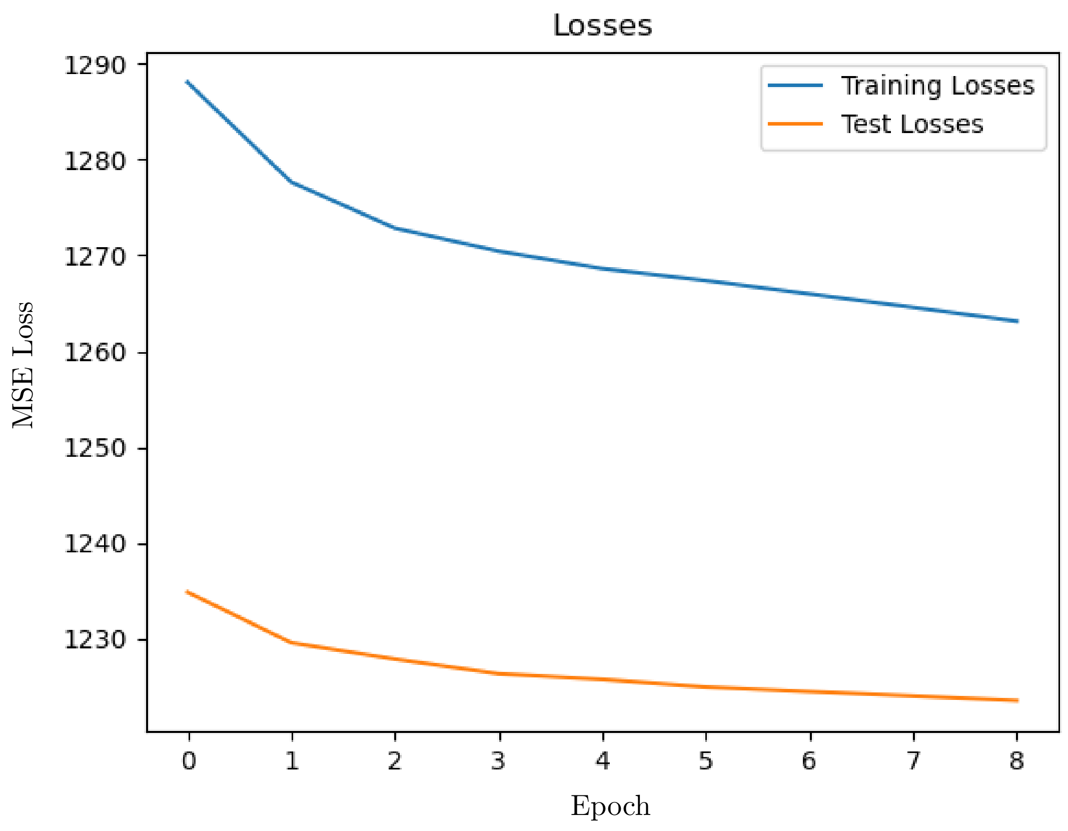


Figure 5.15: Average Training- and test losses of simultaneous iteration training using the Local Patches implementation and the additive update manner. Max iterations is set to 10 and the training was performed for 9 epochs.

5.4.6.1 Results Test 5: Simultaneous Iteration Training

Figure 5.21 shows the average training- and average test losses resulting from Test 5 with SIT. The average training losses are shown in blue, the average test losses are shown in orange and the average loss values are displayed on the y-axis. The x-axis displays the number of epochs. Compared to the results of Test 3, the average training losses of Test 5 have decreased by almost 20 at epoch 8. The average test losses have decreased by about 5 at epoch 8. Hence, the losses have improved by increasing the network's depth but not by a significant margin.

Figure 5.22 shows, from left to right, the average (sub-)cost functionals F, FB and FE for training and testing under Test 5. The y-axis displays the (sub-)cost functional values and the x-axis the number of epochs. The different colours indicate the iteration number. The cost functionals still predominantly change per iteration rather than per epoch. Interestingly, the cost functionals increases as the training progresses. Hence, a decline in performance is observed according to the cost functionals. Therefore, conflicting results have been observed between the MSE losses and the cost functionals.

Figure 5.23 shows an example of a DLE-CSI reconstruction under Test 5 with simultaneous iteration training along with the corresponding ground truth- and error maps. The 2D example shows the transverse plane of the middle slice of a brain. The top row displays the conductivity maps. The reconstructed conductivity has background values corresponding to the ground truth's grey matter tissue. A structure resembling a blurred or low-frequency version of the white matter tissue is taking shape around the middle parts of the brain. The CSF is entirely missing in terms of structure and am-

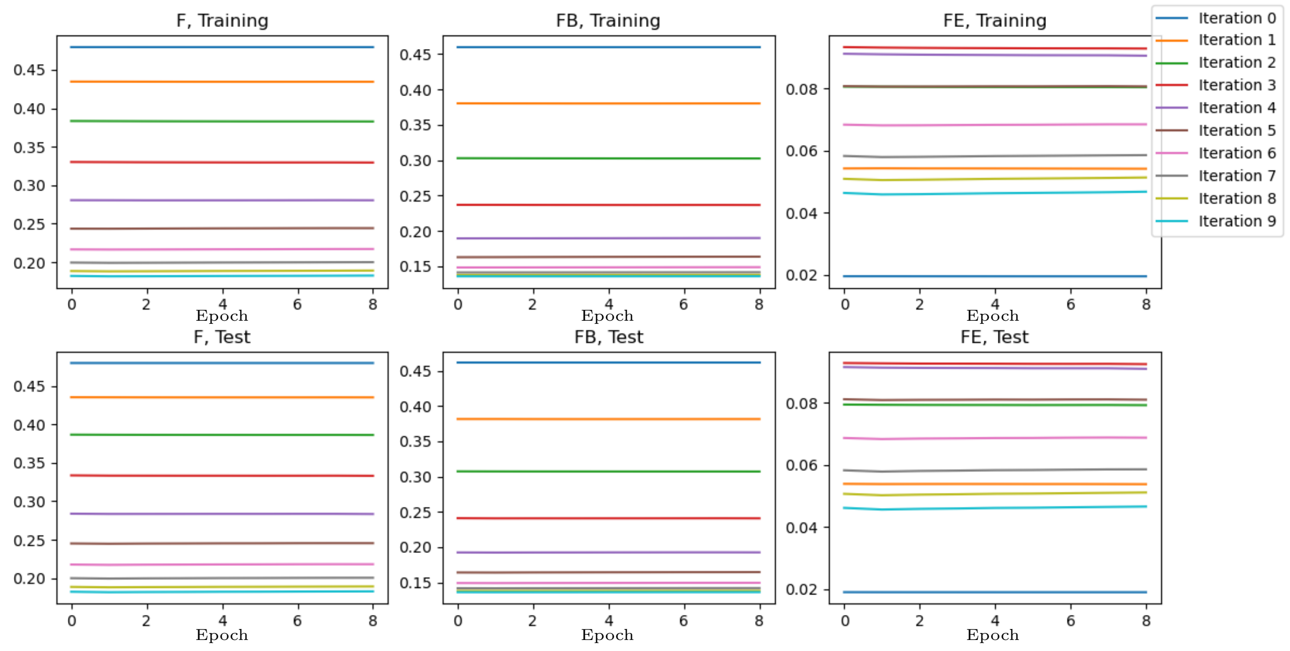


Figure 5.16: Average Cost Functional values at every iteration and epoch of simultaneous iteration training using the Local Patches implementation and the additive update manner. Max iterations is set to 10 and the training was performed for 9 epochs.

plitude. Compared to Figure 5.17 under Test 3, the amount of white matter structure has significantly increased.

The bottom row displays the permittivity maps. The reconstructed permittivity is almost entirely uniform and has a value resembling the white matter. The only structure to be observed is the outline of tissue outside of the brain. Compared to Figure 5.17 the very small amount of the non-background patches has disappeared.

5.4.6.2 Results Test 5: Per Iteration Training

Figure 5.24 shows the average training- and average test losses resulting from Test 5 with PIT. The training and testing losses are displayed as blue and orange lines, respectively. The average MSE loss values are shown on the y-axis and the epoch numbers are shown on the x-axis. At epoch 0 the average training- and average test losses have a value of approximately 1355 and 1290, respectively. In this respect, Test 5 with PIT and Test 4 show near identical loss values at the first epoch. At the final epoch, however, the average training- and average test losses are lower by approximately 20 and 5; displaying a marginal improvement. Compared to Test 5 with SIT, the PIT losses are initially higher but end up at approximately the same losses.

Figure 5.25 shows the average (sub-)cost functionals F, FB and FE for training and testing under Test 5 with PIT. Training and testing is shown on the top and bottom, respectively. The (sub-)cost functional values are displayed on the y-axis and the number of epochs is displayed on the x-axis. The different colours refer to the

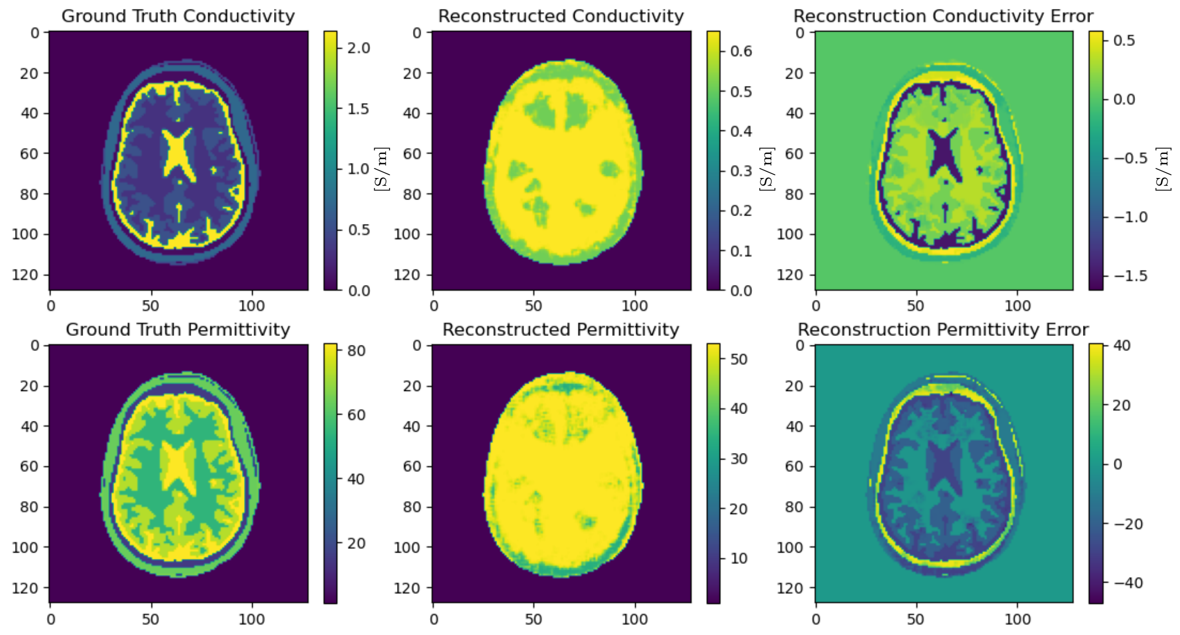

Ground Truth Conductivity
Reconstructed Conductivity
Reconstruction Conductivity Error
Ground Truth Permittivity
Reconstructed Permittivity
Reconstruction Permittivity Error
[S/m]
[S/m]
[S/m]

Figure 5.17: An example of electrical properties reconstructed by DLE-CSI using the local patches-based implementation and additive update manner. The NN was trained for 9 epochs using the simultaneous iteration approach with a maximum of 10 iterations. The reconstruction is performed on a simulated model of the human head, of which a transversal slice is shown. The leftmost images show the ground truth EP values to be reconstructed, the middle column images show the reconstructed EPs and the rightmost images show the difference between the reconstruction and the ground truth.

different iterations, as shown in the legend. The cost functionals still predominantly change per iteration rather than per epoch. Unlike Test 5 with SIT, the cost functionals of Test 5 with PIT decrease as the training progresses. Hence, indicating improving results as training progresses in agreement with the average losses.

Figure 5.26 shows a reconstruction example resulting from Test 5 with PIT. The top and bottom rows display the conductivity and permittivity respectively. From left to right, the ground truth, reconstruction and reconstruction error are shown. The 2D example shows the transverse plane of the middle slice of a brain. The difference in Test 5's reconstruction with PIT versus with SIT, lies in the tissue structures being subtly different. However, neither can be considered better or worse than the other. Conductivity-wise both show a crude white matter structure which lacks the finer structures. Whereas permittivity-wise, both barely show more than a homogeneous value.

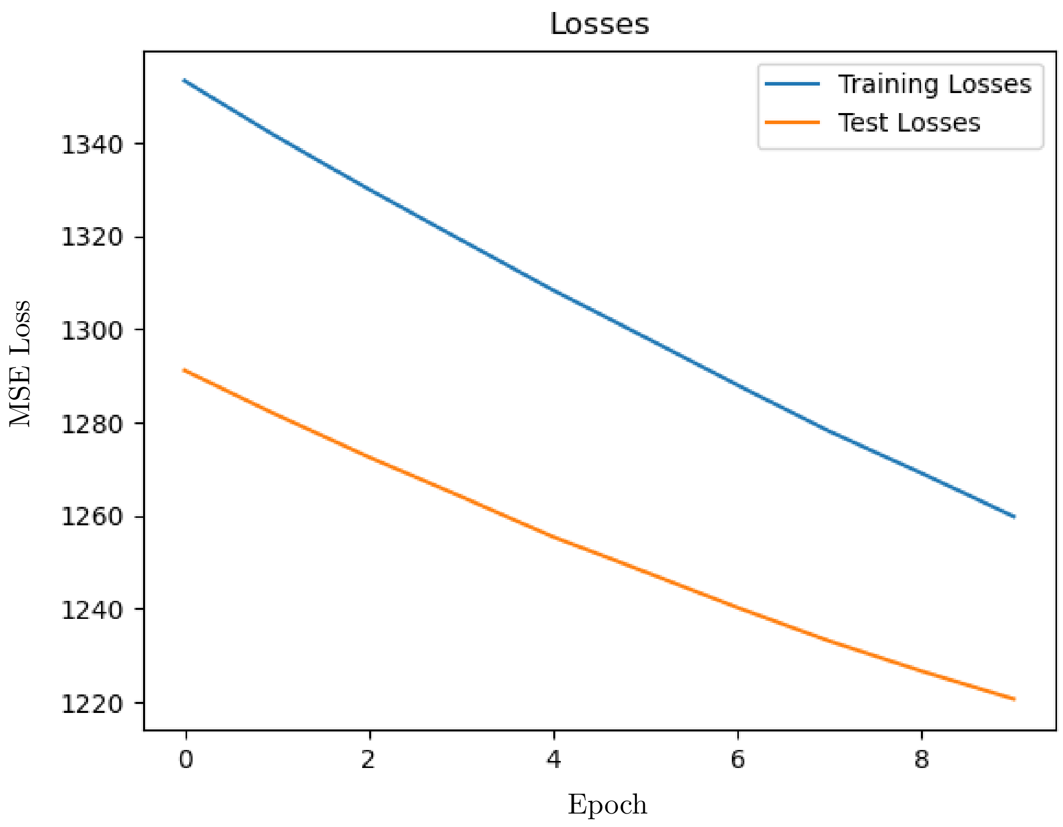


Figure 5.18: Average Training- and test losses of per iteration training using the Local Patches implementation and the additive update manner. Max iterations is set to 10 and the training was performed for 10 epochs.

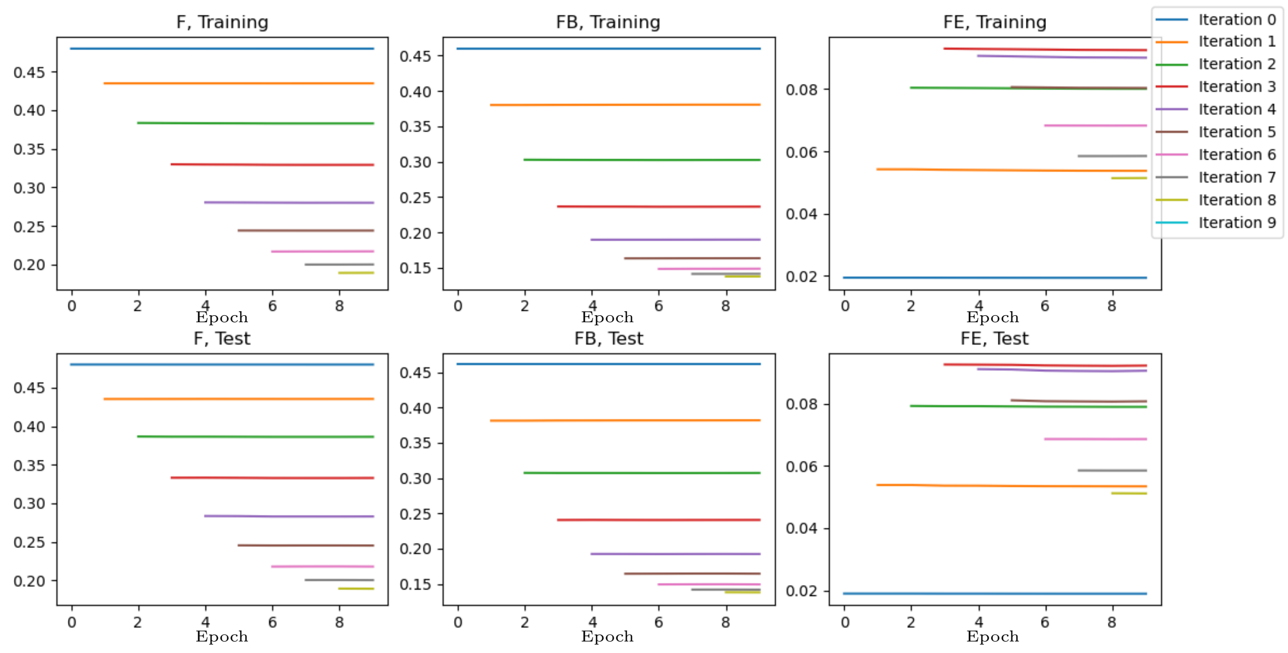


Figure 5.19: Average Cost Functional values at every iteration and epoch of per iteration training using the Local Patches implementation and the additive update manner. Max iterations is set to 10 and the training was performed for 10 epochs.

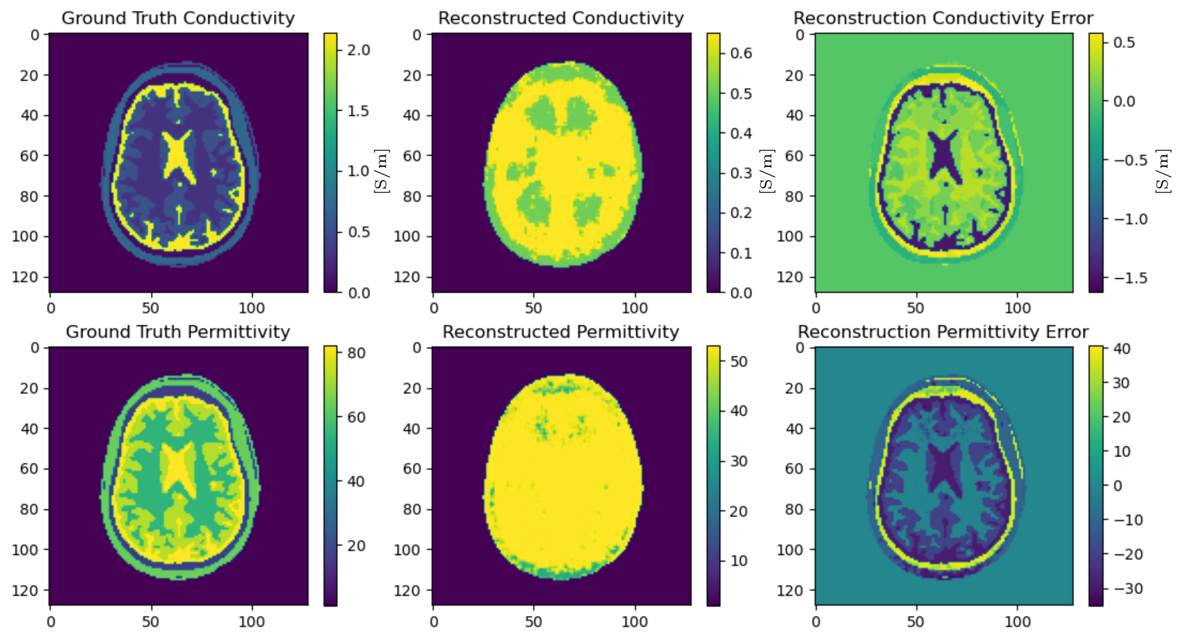

Ground Truth Conductivity
Reconstructed Conductivity
Reconstruction Conductivity Error
Ground Truth Permittivity
Reconstructed Permittivity
Reconstruction Permittivity Error
[S/m]
[S/m]
[S/m]

Figure 5.20: An example of electrical properties reconstructed by DLE-CSI using the local patches-based implementation and additive update manner. The NN was trained for 10 epochs using the per iteration training approach with a maximum of 10 iterations. The reconstruction is performed on a simulated model of the human head, of which a transversal slice is shown. The leftmost images show the ground truth EP values to be reconstructed, the middle column images show the reconstructed EPs and the rightmost images show the difference between the reconstruction and the ground truth.

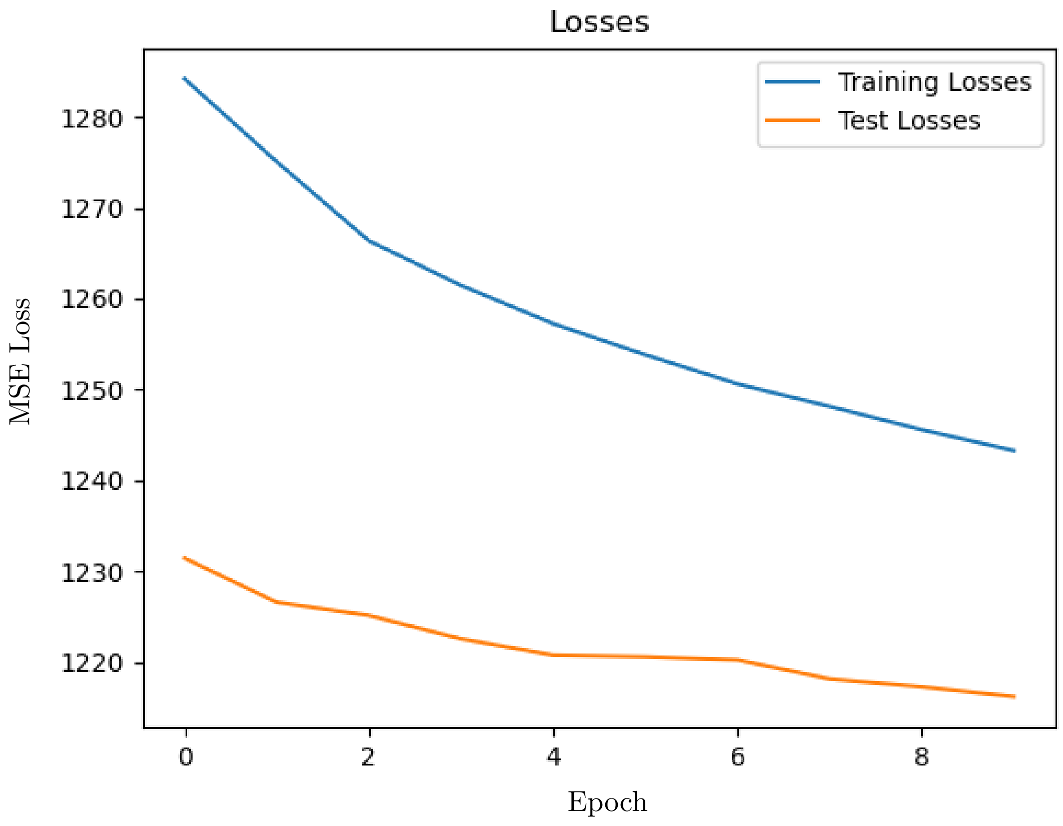


Figure 5.21: Average Training- and test losses of simultaneous iteration training using the Local Patches implementation, the additive update manner and doubled network depth. Max iterations is set to 10 and the training was performed for 10 epochs.

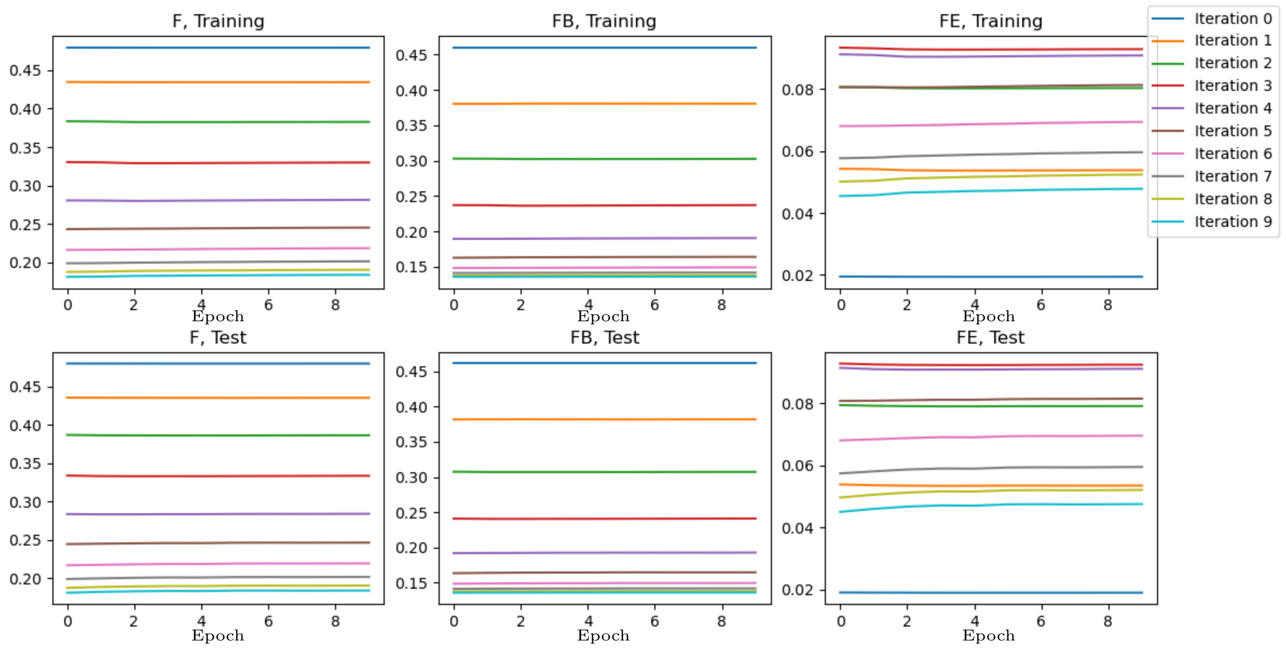


Figure 5.22: Average Cost Functional values at every iteration and epoch of simultaneous iteration training using the Local Patches implementation, the additive update manner and doubled network depth. Max iterations is set to 10 and the training was performed for 10 epochs.

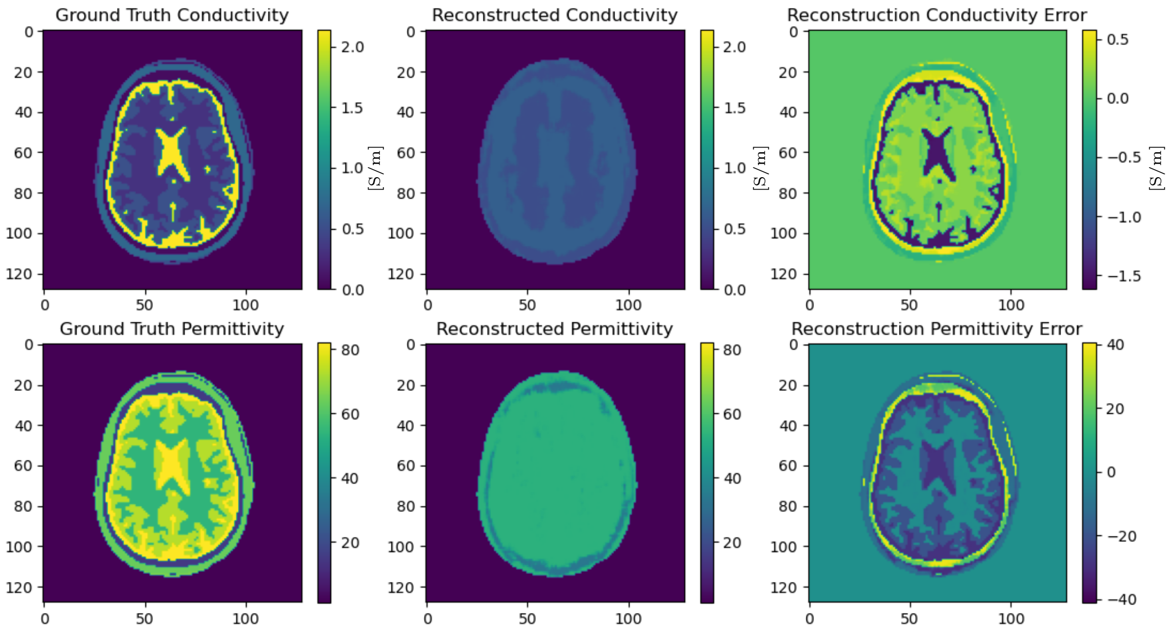

Ground Truth Conductivity
Reconstructed Conductivity
Reconstruction Conductivity Error
Ground Truth Permittivity
Reconstructed Permittivity
Reconstruction Permittivity Error
[S/m]

Figure 5.23: An example of electrical properties reconstructed by DLE-CSI using the local patches-based implementation, additive update manner and doubled network depth. The NN was trained for 10 epochs using the simultaneous iteration training approach with a maximum of 10 iterations. The reconstruction is performed on a simulated model of the human head, of which a transversal slice is shown. The leftmost images show the ground truth EP values to be reconstructed, the middle column images show the reconstructed EPs and the rightmost images show the difference between the reconstruction and the ground truth.

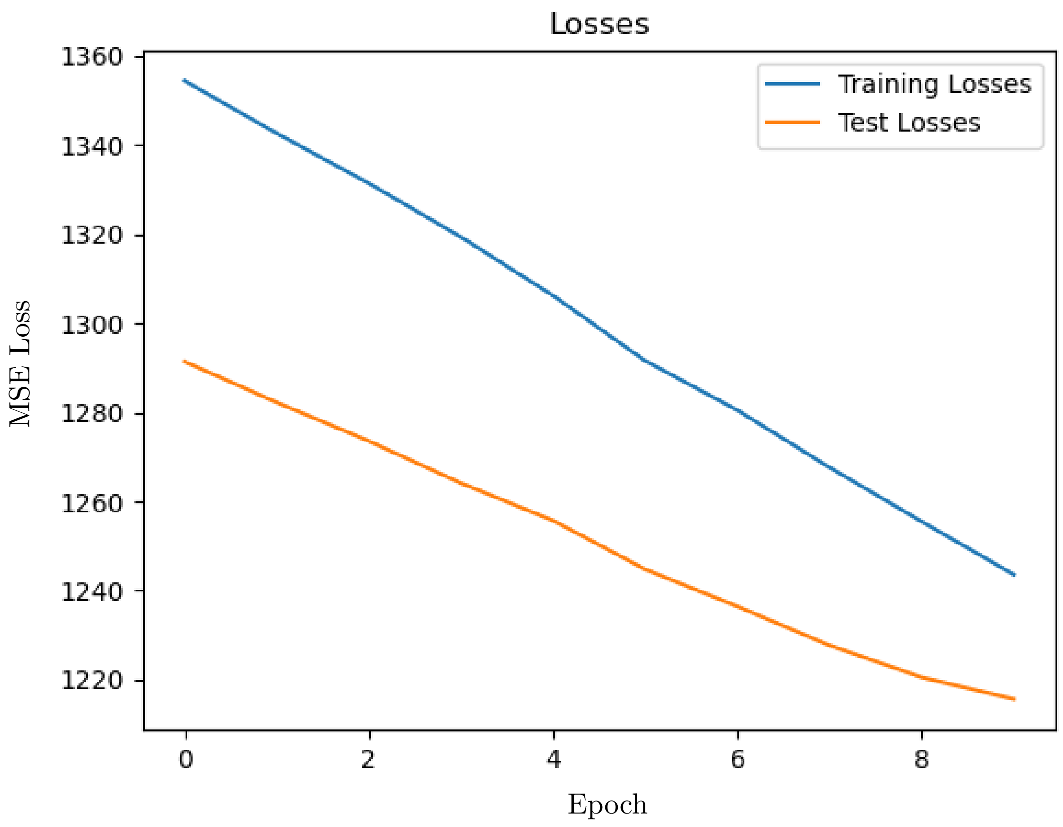


Figure 5.24: Average Training- and test losses of per iteration training using the Local Patches implementation, the additive update manner and doubled network depth. Max iterations is set to 10 and the training was performed for 10 epochs.

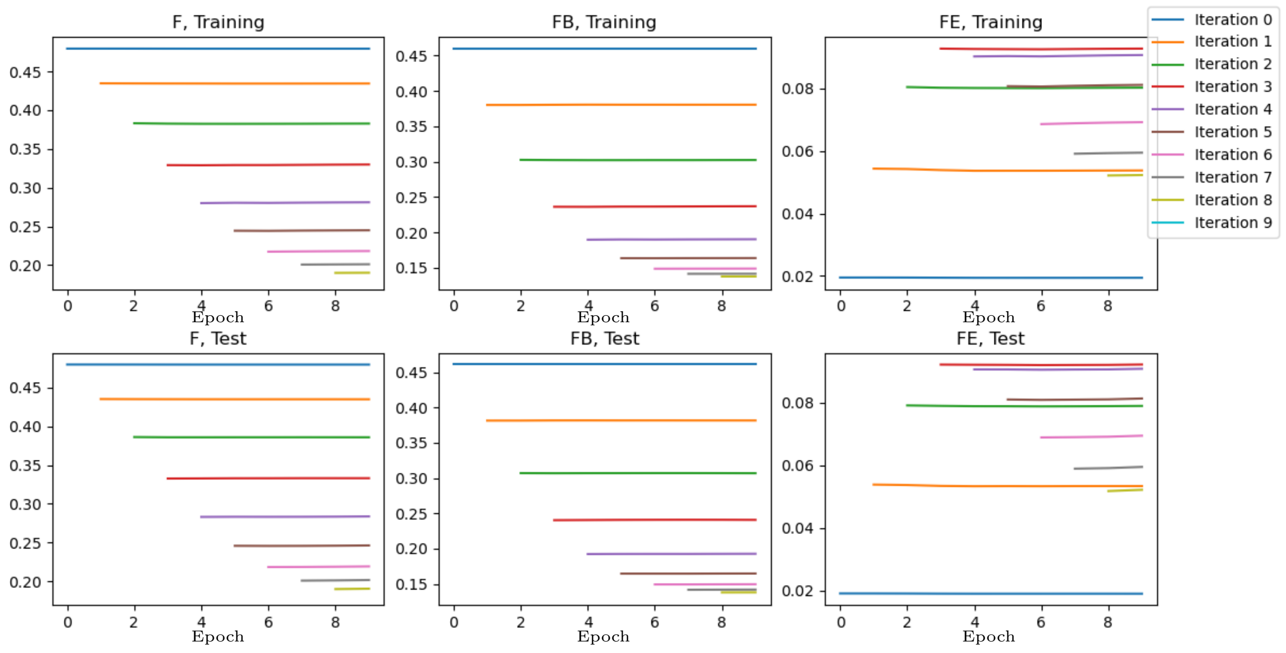


Figure 5.25: Average Cost Functional values at every iteration and epoch of per iteration training using the Local Patches implementation, the additive update manner and doubled network depth. Max iterations is set to 10 and the training was performed for 10 epochs.

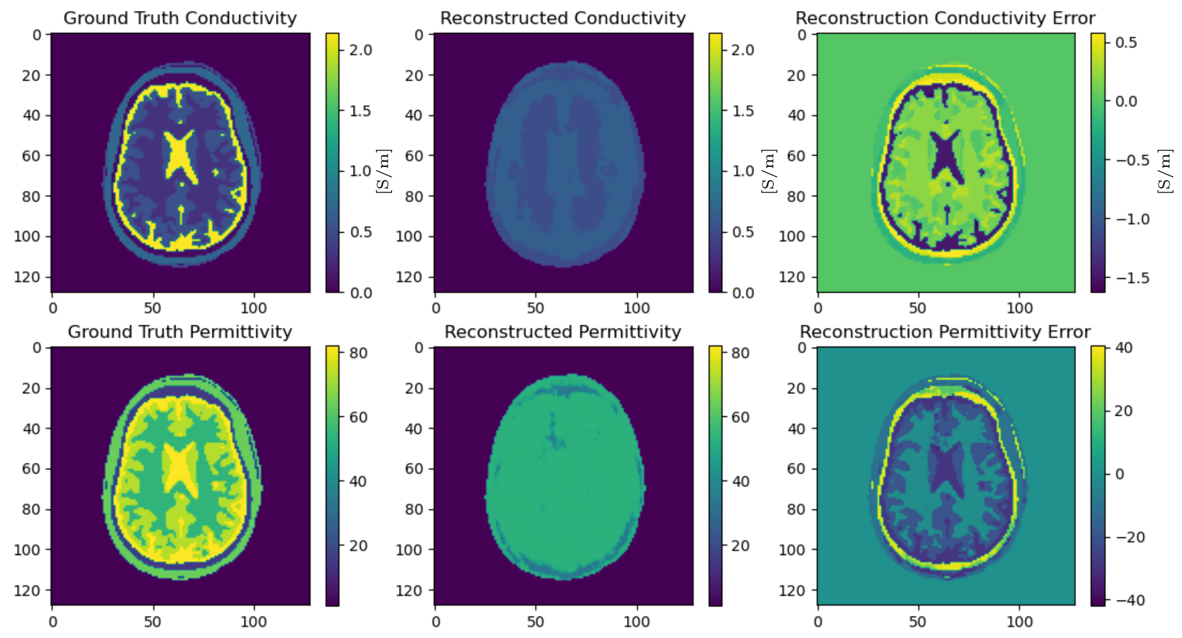

Ground Truth Conductivity
Reconstructed Conductivity
Reconstruction Conductivity Error
Ground Truth Permittivity
Reconstructed Permittivity
Reconstruction Permittivity Error
[S/m]
[S/m]
[S/m]

Figure 5.26: An example of electrical properties reconstructed by DLE-CSI using the local patches-based implementation, additive update manner and doubled network depth. The NN was trained for 10 epochs using the per iteration training approach with a maximum of 10 iterations. The reconstruction is performed on a simulated model of the human head, of which a transversal slice is shown. The leftmost images show the ground truth EP values to be reconstructed, the middle column images show the reconstructed EPs and the rightmost images show the difference between the reconstruction and the ground truth.

6 Phase Error Based Conductivity Enhancement

Suppose an arbitrary MR-EPT method is used to reconstruct a conductivity map from MR measurements. Depending on the reconstruction method used, different types of errors can occur in the reconstructed conductivity map. These errors can arise from assumptions used in the reconstruction methodology, such as boundary errors when assuming the tissue to have locally homogeneous conductivity values. Alternatively, if for example a DL reconstruction method was used, errors might be introduced due to reconstructing from unseen data (data that has not been used in training the NN).

Such errors in a reconstructed conductivity map, also alter the conductivity's relation to the transceive phase. Meaning, that errors in the conductivity reconstruction propagate to the corresponding transceive phase. This relation between the conductivity and the transceive phase is showcased in SH-EPT's methodology, Equation 2.38. Note that SH-EPT relies on assumptions to achieve this direct relation between the conductivity and the transceive phase. Hence, the extent of the errors being propagated to the transceive phase relies on the extent of the validity of SH-EPT's assumptions.

If the altered phase map can be reconstructed from the reconstructed erroneous conductivity map, it can be compared to the measured transceive phase to compute a phase error map. Assuming that the relation between the conductivity and the transceive phase holds to a large extent, the phase error map is likely to provide information on the difference between the reconstructed- and true conductivity.

Figure 6.1 visualises the process of determining such a phase error map. ϕ represents the measured transceive phase map, $\hat{\sigma}$ and $\hat{\phi}$ represent the erroneous reconstructed conductivity map and the corresponding erroneous transceive phase map, respectively. $\Delta\phi$ and $\Delta\sigma$ denote the phase- and conductivity error maps.

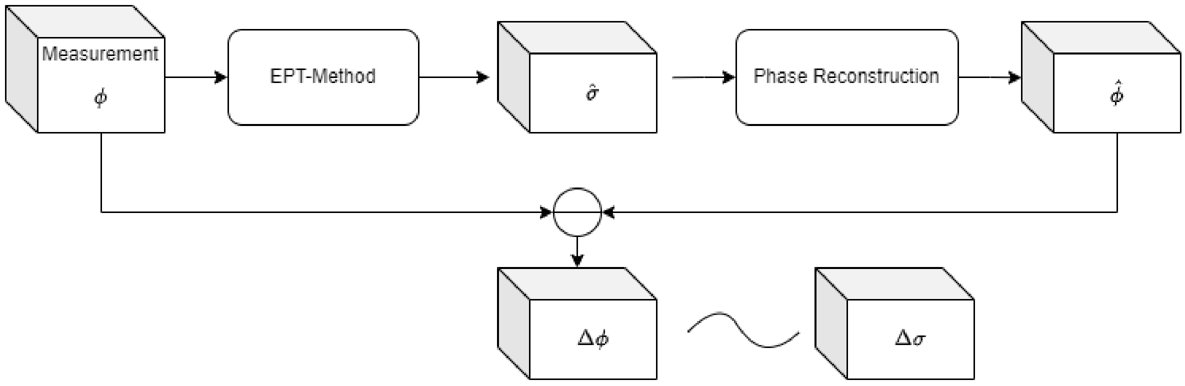


Figure 6.1: Diagram displaying the acquisition of a phase error map in relation to the error of a reconstructed conductivity map.

This chapter describes a method that leverages this phase error map in an attempt to correct reconstruction errors, dubbed phase error based conductivity correction (PEBCC). A DL methodology is proposed in an attempt to implicitly learn the conductivity error's relation to the phase error. A schematic overview of the neural network's in- and outputs are given by Figure 6.2.

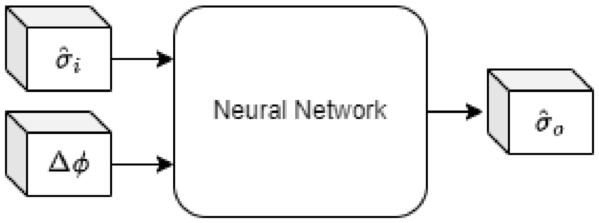


Figure 6.2: Schematic overview of phase error based conductivity correction

A DL method is deliberately chosen over the direct use of SH-EPT to potentially acquire a conductivity error map from a phase error map. The direct use of SH-EPT would subject the error maps to SH-EPT's issues such as boundary problems and high noise sensitivity, which might only further degrade the resulting conductivity map instead of achieving improvement. Additionally, the availability of a dataset with simulated human heads with the same anatomies but differing conductivity- and phase maps, allows for precise training and testing of a neural network. As two different sets of conductivity and corresponding phase with the same anatomy $\{S_1(\sigma_1, \phi_1), S_2(\sigma_2, \phi_2)\}$ can be assigned as inputs $\hat{\sigma_i} = \sigma_1$, $\Delta\phi = \phi_1 - \phi_2$ and output $\hat{\sigma_o}, = \sigma_2$ in order to train the network into correcting σ_1 into σ_2.

6.1 Methodology

PEBCC starts with a given conductivity map, which can either have been reconstructed using an MR-EPT method or have been artificially created based on literature values. As we aim to improve the conductivity map, we consider it to be an erroneous estimation of the true map. Hence, denoting it as $\hat{\sigma}$, with the hat signifying an estimation.

A phase map is to be obtained from the estimated conductivity map using some kind of mapping function $\phi(\sigma)$. Computing the phase map can be achieved by following the approach of Borsic et al. [51], rewriting Equation 2.38 as

$$\nabla^2\phi = \mu_0\omega\sigma \tag{6.1}$$

This equation can be treated as a linear matrix equation by expressing the derivatives as finite differential kernels. The phase map can then be computed using a forward solver. The disadvantage of using this method is its reliance on SH-EPT's assumptions potentially introducing new errors into the phase map. Alternatively, a neural network could be trained to compute the phase map from the corresponding conductivity map. However, similarly to the approach of Borsic et al., new errors can be introduced into the phase map by the neural network depending on its performance.

By comparing the obtained phase map with a measured phase ϕ_m acquired with MRI, a phase error map is established $\Delta\phi$. The estimated conductivity and the phase error are used as input for a neural network, which computes an updated conductivity estimate. A new phase map can be computed from the updated conductivity, resulting in an updated phase error. Hence, PEBCC can be iteratively used to improve reconstructed conductivity maps, or as a complete MR-EPT method by for example using a homogeneous initial guess for the conductivity. Figure 6.3 illustrates PEBCC's process, with the subscripts i and $i + 1$ denoting the current and next iteration. Additionally, the phase error can be used as an uncertainty measure of the reconstructed conductivity map, as a perfect reconstruction should yield a phase error of 0.

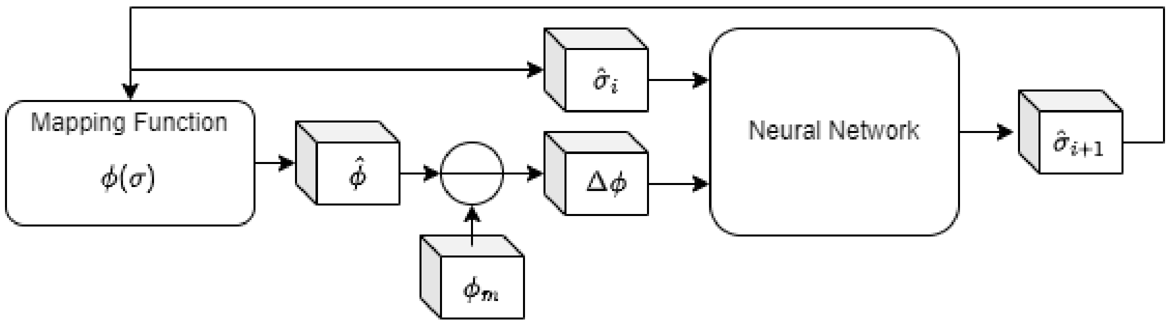


Figure 6.3: Schematic overview of using phase error based conductivity enhancement in an iterative manner.

6.2 Neural Network Design

The neural network chosen to compute the conductivity updates is a U-net. A U-net is a convolutional neural network, that is typically used for image-related tasks. U-nets have been shown to perform well on pixel-wise regression problems such as pansharpening in the field of remote sensing [48]. Considering that updating our conductivity estimates per pixel (or voxel) is similarly a pixel-wise regression problem, makes a U-net a suitable choice.

The U-net derives its name from the way the network and its data flow are typically depicted. The U-net can be split into two paths, connected at the bottom, giving it the U shape. The path on the left consists of encoder blocks and the data travels downwards. The path on the right consists of decoding blocks and the data travels upwards.

As one travels down the encoding blocks, the spatial resolution of the data is reduced, whereas the feature space is increased. The encoding block consists of one or more convolution layers and a pooling layer. The (first) convolution layer increases the amount of feature channels. The pooling layer reduces the spatial resolution.

As one travels up the decoding blocks, the spatial resolution is increased, while the feature space is decreased. A decoding block consists of an up-convolution layer and one or more convolution layers. The up-convolution layer increases the spatial resolution. The (first) convolution layer decreases the amount of feature channels.

The U-net features skip connections between the encoding and decoding blocks. These skip connections and the varying spatial resolution levels allow the network to learn and incorporate features at differing spatial resolutions. This can be considered as incorporating information from different frequency components within the image data.

The bridge or lower path of the U-shape typically functions like an encoding block without using a pooling layer.

6.2.1 Neural Network Implementation Details

The network implemented features 3 encoding and decoding blocks. Each convolution layer, within these blocks and those within the bridge, is followed by a ReLU activation function to introduce non-linearity into the network. A diagram of the implemented network is shown in Figure 6.4 below.

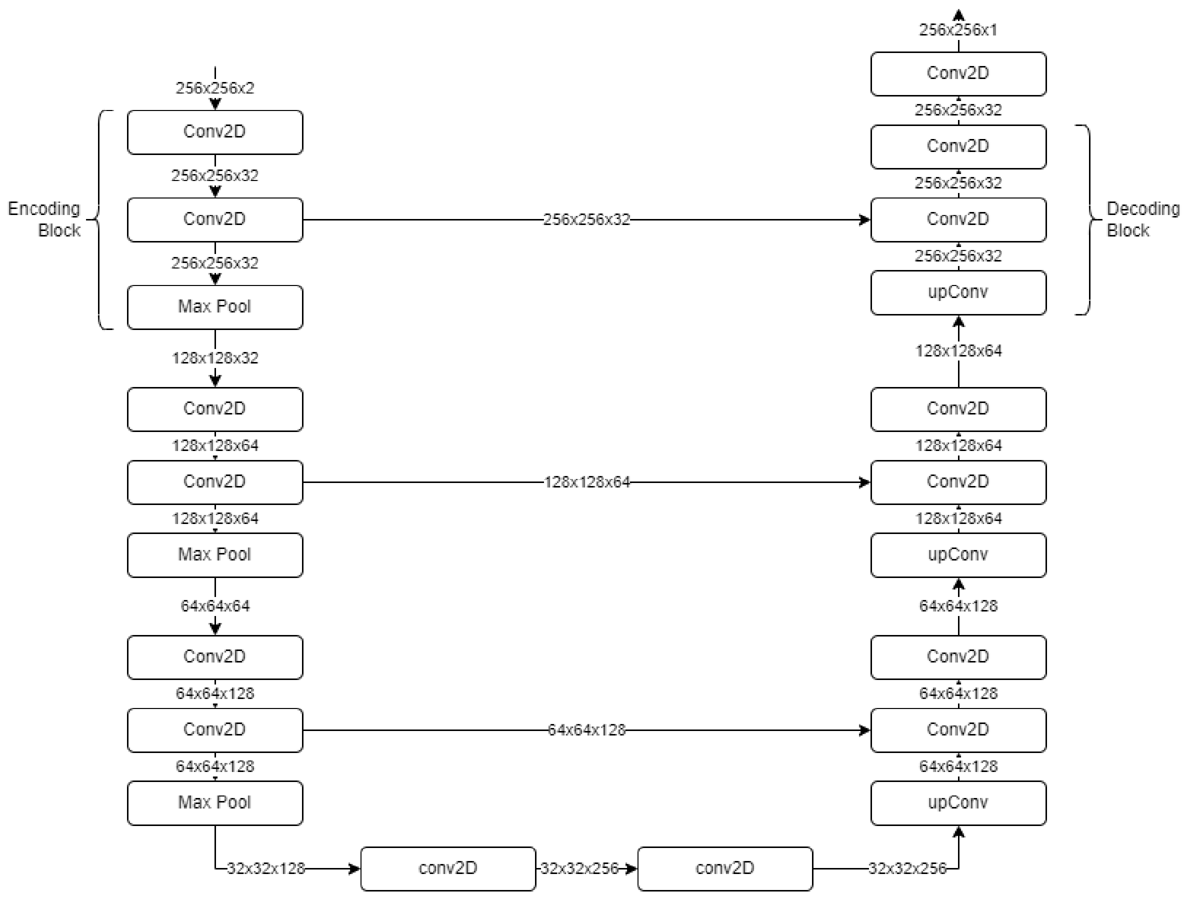


Figure 6.4: U-net design featuring 3 encoder and decoder blocks.

The input data consists of two channels of 256 by 256 pixels, in the figure shown as 256 x 256 x 2. The first channel contains the estimated conductivity map and the second channel contains the corresponding phase error map. Limiting the input data to 2D prevents the network's memory usage from becoming a bottleneck.

In the first encoding block, the amount of feature channels is increased from 2 to 32 by the first convolution layer of the network. Afterwards, each encoding block decreases the x,y-dimensions by 2 and increases the amount of feature channels by 2. Inversely, the decoding blocks increase the x,y-dimensions by 2 and decrease the feature channels by 2. The final layer produces the output of a single channel, a conductivity map.

Though chosen arbitrarily, using 3 blocks and 32 initial feature channels seemed small enough to not become a burden in terms of memory size and sufficiently large to yield sufficient results.

The convolution kernel sizes are chosen to be 3-by-3 pixels. The size is relatively small compared to the initial image size of 256-by-256 but the receptive field is expected to be sufficient due to the decreasing resolution travelling down the U-net.

6.3 Training Strategies

The database used provides 3D conductivity and phase maps of human heads. The dataset is split into 6 healthy subjects and 18 subjects with a brain tumour. Each healthy subject has 3 different geometry configurations and each geometry configuration has 4 variations in conductivity values.

The subjects with tumours only feature 2 different conductivity variations of the tumour tissue; the conductivity of healthy tissue does not vary.

The network is trained with 5 healthy test subjects while leaving out 1 healthy subject for testing. Hence, a training set size of $5 * 4 * 3 = 60$ and a test set of size $1 * 4 * 3 = 12$.

The tumour-bearing subjects are used in testing under the name of Out Of Distribution (OOD). As the tumours are structures that remain unseen by the network while training, they can be considered to be outside of the distribution of data. Tumours could possibly be detected by potential faulty reconstructions caused by tumours being OOD. Alternatively, tumours being reconstructed correctly would also enable tumour detection. However, tumour detection was not investigated due to time constraints.

When training, the dataset is split into batches containing two sets of conductivity maps with their corresponding phase maps $S_1 = \{\sigma_1, \phi_1\}, S_2 = \{\sigma_2, \phi_2\}$. Set 1 ($S_1$) is used as input variables and set 2 (S_2) as ground truth variables. Therefore, σ_1 represents the reconstructed conductivity map and σ_2 the ground truth conductivity map σ_{GT}. The ground truth conductivity map is compared to the output of the network using a loss function. Similarly, ϕ_1 represents the phase map normally acquired with a forward solver from the reconstructed conductivity and ϕ_2 represents the true or measured phase map, normally obtained with MRI. The phase error map computed from ϕ_1 and ϕ_2, and σ_1 are combined as input for the neural network. Based on the loss, the weights of the network are adjusted using back-propagation. This training process is visualised in Figure 6.5.

The MR-EPT methods to be used for reconstructing the conductivity estimations used as input are assumed to correctly reconstruct the conductivity's geometry. This assumption lets us limit the sets used in a batch to only the conductivity variations of the dataset. Meaning that a batch contains no variation in geometry or subject between the used sets. Not having to learn adjustments in geometry should reduce the

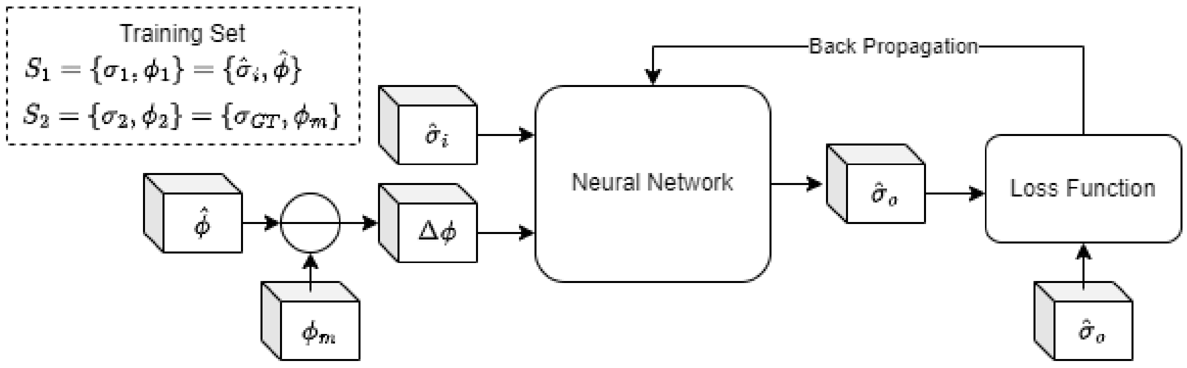


Figure 6.5: Schematic overview of phase error based conductivity enhancement in training

network's requirements as the operations should be less complex.

Of the 4 variations in conductivity for the healthy subject, the 4^{th} one is always used as the ground truth set (S_2). This means that per geometry variation, per healthy subject, there are 3 batches of data. Training with 5 healthy subjects, with their 3 geometry variations results in $5 * 3 * 3 = 45$ batches.

The conductivity and phase maps of each set consist of 3D data of size $\{256, 256, 160\}$ in the $\{x, y, z\}$ dimensions. Each voxel edge has a length of 1 mm in each dimension. To keep the neural network of a manageable size, it was implemented to process 2D data. Therefore, the input data is split into 160 x,y-plane slices. Resulting in each of the 45 batches containing 160 inputs of size $\{256, 256\}$.

6.3.1 Test 1: K-Fold Validation

For Test 1 K-fold validation is used to assess the training capabilities of the network. The healthy subject used for testing is switched every fold to implement the k-fold validation. The k-fold validation has been limited to 4 folds to prevent excessive training times. Implementation-wise, using 4 folds means alternating 4 of the healthy subjects for testing, while two healthy subjects are exclusively used for training.

A Mean Squared Error loss is chosen as a loss function for training the network. The MSE loss is defined in Equation 5.4:

The MSE loss was chosen due to its strong suppression of outliers, caused by the squaring of the estimation error. The network's capability to be used iteratively incentivises the improvement to be focused on a per pixel level rather than achieving a greater mean improvement. Since improving pixel values on average, does not necessarily ensure convergence for every pixel. Greatly punishing outliers prevents divergence of individual pixels. Hence the use of an MSE loss function.

6.3.2 Test 2: Hybrid Loss Function

In Test 2 K-fold validation was dropped in favour of a leave-one-out approach, where just like in K-fold validation 5 healthy subjects are used for testing and 1 healthy subject is used for testing. However, instead of repeating the process with a different subject used for testing, the process is executed only once. This has the advantage of saving time while testing variations in training the network. However, the network is less thoroughly validated on the generalisation of input data.

The loss function has been adjusted by including the Mean Absolute Error (MAE)

$$MAE(x, \hat{x}) = \frac{1}{N} \sum_{i=0}^{n} |x - \hat{x}|. \tag{6.2}$$

The new loss function is expressed as a combination of the MSE and MAE

$$Loss(x, \hat{x}) = \alpha MSE(x, \hat{x}) + (1 - \alpha) MAE(x, \hat{x}), \tag{6.3}$$

with α a parameter for balancing the loss functions. The MAE penalises outliers less hard compared to the MSE. Shifting from the MSE to the MAE moves the learning focus of the network away from only improving the large conductivity differences as much as possible. Large conductivity differences cause comparatively higher losses due to the squared factor. Ideally, all conductivity differences should yield improvement regardless of how much, as iterative usage would then minimise the loss over time. However, trying to improve marginally small errors is unrealistic and unnecessary. Hence, the larger errors should be punished most severely while slightly punishing small errors as well. Therefore, the MSE and MAE are combined with a balancing factor, in order to find a balanced output. The balancing factor α is initially set to 0.5.

The learning rate and epoch amount have been tuned for the aforementioned network parameters. Both are achieved by training with a large amount epochs; 100 epochs were eventually used. The learning rate was deemed too high if the losses showed oscillations and didn't converge. The learning rate was deemed too low if the losses decreased too slowly, such as taking 100 epochs. If the initial learning rate was deemed too high, the learning rate was halved till it was assessed as too low. From then on it was adjusted to halfway points between its nearest known points that are considered too high or low. Until a satisfactory learning rate was found. The number of epochs was determined by assessing when the network started to overfit, indicated by the point where test losses started increasing. The amount of epochs just before overfitting should provide optimal results.

6.3.3 Test 3: Convolution Kernel Tuning

Test 3 expands upon the changes in Test 2 by additionally tuning the convolution kernel size. The convolution kernel size is a tunable parameter that was investigated in an attempt to further increase the performance of the network. The initial kernel size was set to 3, meaning that the kernel has side lengths of 3 voxels in both the x and y directions. If SH-EPT's relation between phase and conductivity is reliable, then the network ought to learn operations resembling second-order special derivatives. However, if the kernel size is too small, the learned operator might rather resemble a linear approximation. This can likened to heavily undersampling a signal. Hence, larger kernel sizes might be beneficial in learning the appropriate operator. However, a U-net might not suffer from this issue due to operating on different spatial resolutions facilitated by the pooling layers.

The effect of kernel sizes on the losses has been investigated by training the network for 11 epochs at the same learning rate of $2.5e-4$, for kernel sizes of length $\{3, 5, 7, 9, 11\}$. In order to counteract boundary effects prevalent at larger kernel sizes, a segmentation of the input data has been performed beforehand. For every tissue type, a mask was created and the input data was split into a batch of every tissue type. This way all conductivity values outside the specific tissue type are set to zero. Effectively eliminating the influence of any other tissue types. At the end of the network, the segments are rejoined as the final output.

Segmentation is easy to implement due to the data set being simulated and providing these segmentations of the bat. In practice, these segmentations could be derived from different MRI modalities, such as T2-weighted imaging.

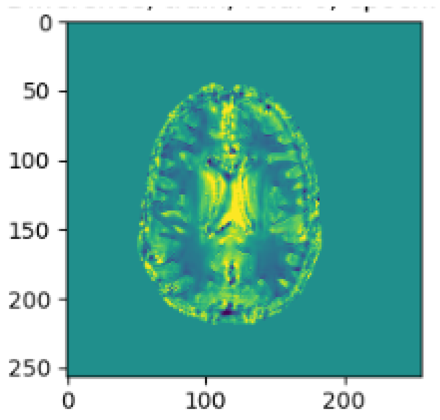


Figure 6.6: Updated conductivity map with a convolution kernel size of 11, after training for 11 epochs at a learning rate of 2..5e-4. An example of the boundary effect.

Figure 6.6 shows an example of the boundary effect, displaying the necessity of segmentation. The reconstruction has been produced with a kernel size of 11 without segmentation. The tissue surrounding boundaries are blurred. This is what we refer to as boundary errors. Boundary errors are caused by the larger kernels taking too much information into account from tissue types differing from the tissue type at the centre voxel. At boundaries, this smooths its surroundings.

6.3.4 Performance Metrics

To evaluate the performance of the network, two metrics are introduced. Considering the previously introduced sets of conductivity and phase $\{\hat{\sigma}_i, \phi_i, \sigma_{GT}, \phi_{GT}\}$ as well as the networks corresponding output $\hat{\sigma}_o$, we define the error in updating the conductivity as the difference between the ground truth and the network's output:

$$\Delta\sigma_o = \hat{\sigma}_o - \sigma_{GT}. \tag{6.4}$$

This error is henceforth referred to as the update error.

The to be corrected error between the input and the ground truth is expressed as:

$$\Delta\sigma_i = \hat{\sigma}_i - \sigma_{GT}, \tag{6.5}$$

which is henceforth referred to as the input error.

The first metric, the relative error, is introduced as this update error in relation to the difference in conductivity between the input and the ground truth:

$$e = \frac{\Delta\sigma_o}{\Delta\sigma_i} = \frac{\hat{\sigma}_o - \sigma_{GT}}{\hat{\sigma}_i - \sigma_{GT}}. \tag{6.6}$$

We define the second metric, the Sensitivity, as the smallest difference in conductivity that the network is able to improve. Meaning, a relative error lower than one, $e \leq 1$.

In order to validate the feasibility of the phase error-based conductivity enhancement, the training and testing settings are limited to ideal conditions. This entails that no noise is present and the geometry of both input and output conductivity maps are considered equal. When using input data of a human head, equal geometry then refers to the location and shape of different tissues being identical. Only the conductivity values per tissue, are changing.

6.4 Results

6.4.1 Results Test 1: K-Fold Validation

The average test- and OOD losses resulting from Test 1 are plotted in Figure 6.7, with the number of epochs displayed on the x-axis and the losses displayed on the y-axis. The solid lines represent the test losses, whereas the dashed lines represent the OOD losses. The training strategy consists of k-fold cross-validation for 4 folds and 3 epochs. The initial learning rate was set to $1e-3$ and only the MSE loss was used. Each colour represents the same fold for both the test losses and the OOD losses. Figure 6.7 shows two things of note. Firstly, the first fold performs significantly better compared to the other folds. Secondly, the OOD losses are lower than their respective test losses. The lower OOD losses are caused by the lack of difference in conductivity values outside of the tumour region between the two sets used for testing. Meaning that outside the tumour the network does not need to make any adjustments. Though the network is unlikely to keep the updated conductivity exactly the same, the error between the ground truth and update is expected to be very small. This skews the loss of evaluating the OOD data to the lower end. Therefore, comparing test and OOD losses has little meaning. Only including tumour regions in the loss would not result in a suitable comparison either. If conductivity updates outside the tumour region are neglected, trained networks that only perform well on tumour regions could outperform networks that achieve a better update for the entirety of the head.

Figure 6.8 shows an example of the network's results for the first fold at the third epoch at the middle slice of the brain. The top row shows from left to right: the ground truth conductivity σ_{GT}, the input conductivity $\hat{\sigma}_i$ and the updated output conductivity $\hat{\sigma}_o$. The bottom middle image shows the input error, which the network intends to resolve. It gives us insight into the amount of change required to perfectly update the conductivity. The bottom right image shows the update error. Its values should ideally reach as close to zero as possible. The bottom left image shows the relative error, which

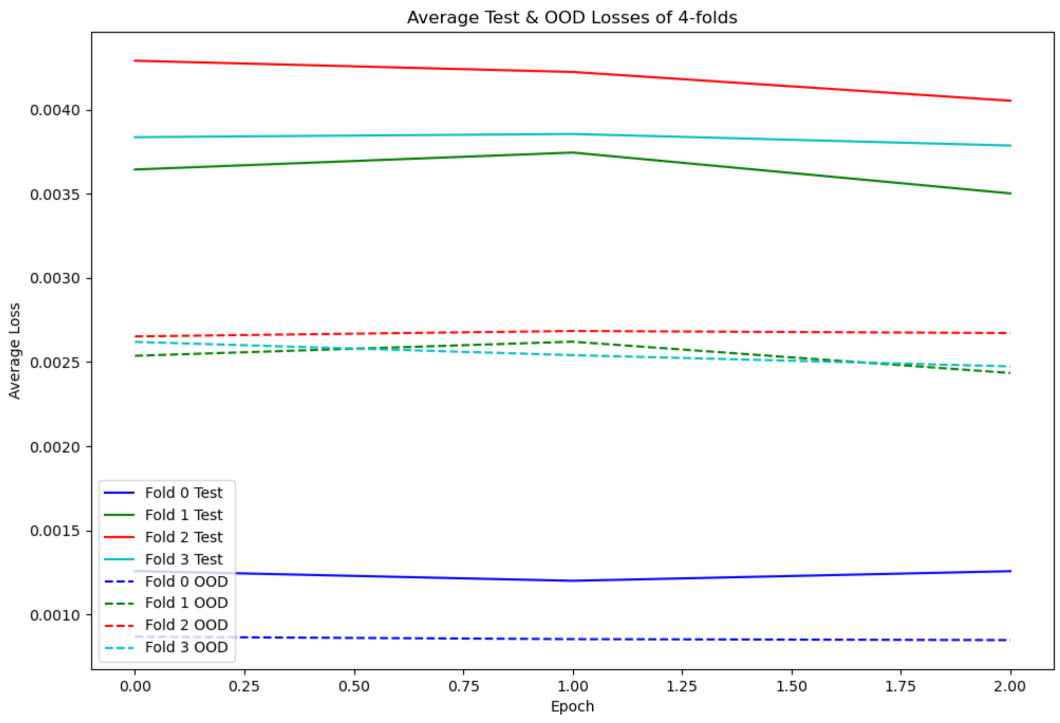


Figure 6.7: Average test- and out of distribution losses per epoch, for 4 folds of k-fold validation.

is the input error of the bottom middle image relative to the update error of the bottom right image (Equation 6.6). Ideally, all values should be lower than 1 and higher than 0. Note that the values of the relative error are capped at 1, as we are mostly interested in when values no longer improve and not to which extent.

As shown in the bottom right figure, the grey matter and white matter show a very similar level of discrepancy with the ground truth. The bottom middle figure, however, shows that the grey matter requires minimal change. The update of the grey matter shows decline, while the white matter has improved slightly. The CSF in the centre of the brain shows an improvement, as the CSF's amplitude in the figure on the right is about 0.1 smaller than in the figure in the middle. These findings are confirmed by the relative error on the bottom left, which shows relative errors below 1 for the CSF and white matter while the grey matter is valued at or above 1. Additionally, the relative error shows a lower value and therefore a larger improvement in the CSF than in the white matter.

The results in Figure 6.9 show quantitative results of the relative error for test 1. These results have been gathered from two healthy subjects, which have not been used during the training of the network. Two healthy subjects equate to 24 different conductivity input errors to be corrected.

Violin plots were chosen in order to visualise the large number of updated data points per input error value in an understandable representation. A violin is a visual

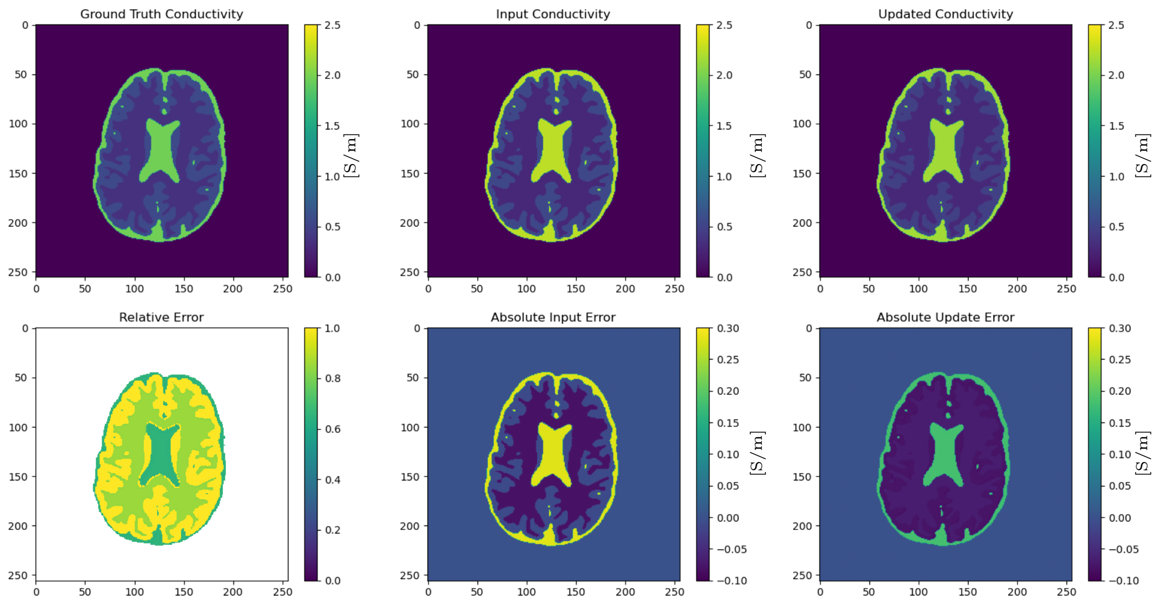


Figure 6.8: Evaluation results of training the first fold at the third epoch using an MSE loss. From the top left to the bottom right: the ground truth conductivity, the input conductivity, the updated conductivity output from the network, the relative error: the update error relative to the input error (Equation 6.6), the input error: the difference between the input and the ground truth, the update error: the difference between the updated conductivity and the ground truth.

representation of a distribution of data, which tends to be shaped like the musical instrument. A violin consists of an upper-, lower- and middle horizontal line demarcating the maximum value-, minimal value- and mean value of the data, respectively. The distribution of the data between the range of the minimum and maximum values is represented by a light blue outer shape. The wider the outer shape, the more data points are centred at that value.

The relative error results associated with an input error have been grouped as a violin. Hence, each input error value has its own distribution of the relative error.

The scope of Figure 6.9 has been restricted to values between -1 and 1 for the relative error shown on the y-axis. The 24 different input errors are shown on the x-axis in ascending order. Any relative error outside of the -1 to 1 range signifies deterioration in results, as the magnitude of the output error has exceeded the magnitude of the input error.

Due to the limited scope of the y-axis, no violins are displayed at most of the input error values below 0.02. Each x-axis label without a corresponding violin is located out of scope. For these missing data points the network worsens the reconstruction. These out-of-scope occurrences are almost entirely restricted to negative input error values. Only at the input error of -0.08 does a negative input error result in a relative error distribution entirely within scope. Hence, the network performs poorly when correcting

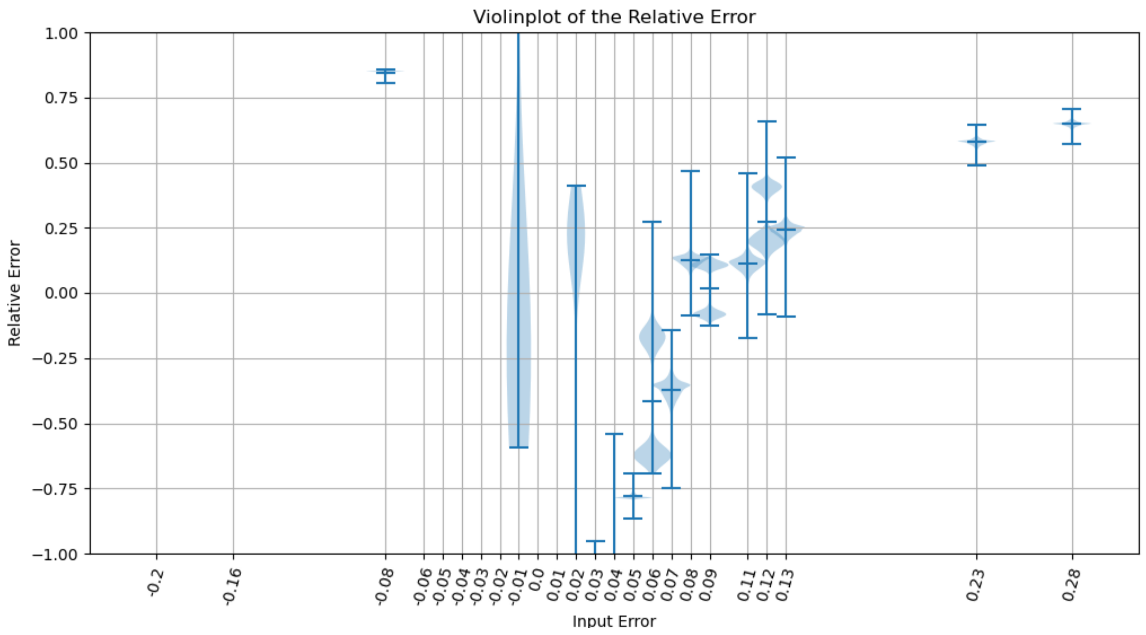


Figure 6.9: Violin plots displaying the relative error of the network under test 1

negative input errors.

From an input error value of 0.05 and above all violins are within scope. From a value of 0.08 and above all the mean relative error values are above 0. For all above 0.08, with the exception of 0.09, the bulk of the distribution is all positive in value. Only the input error values of 0.23 and 0.28 always result in positive relative error values within scope.

Negative relative errors above -1 can be considered an improvement in absolute terms since the difference between the reconstruction and the result has lessened. A negative relative error resulting from a positive input error means the output error is negative. As shown in Figure 6.9, the current network is unable to properly correct negative input errors. Hence, negative relative errors for positive input errors inhibits the successive use of the network for iterative improvement of the conductivity.

For a single conductivity correction, the input error sensitivity for test 1 equates to 0.05. For iterative use, the input error sensitivity equates to 0.08 when considering the mean and 0.23 in all cases.

The network seems to have prioritised diminishing the large differences in amplitude within the CSF. This, however, seems to degrade the performance of the network when it needs to resolve smaller changes in conductivity. This resembles strong outlier suppression, which is characteristic of using the MSE loss.

6.4.2 Results Test 2: Hybrid Loss Function

The network is intended to be used in an iterative manner to correct conductivity estimations. Therefore, the network should improve as wide a range of conductivity differences as possible. This allows us to converge the conductivity estimate close to its true value. Hence, the loss function adjustment introduced in test 2 6.3.2.

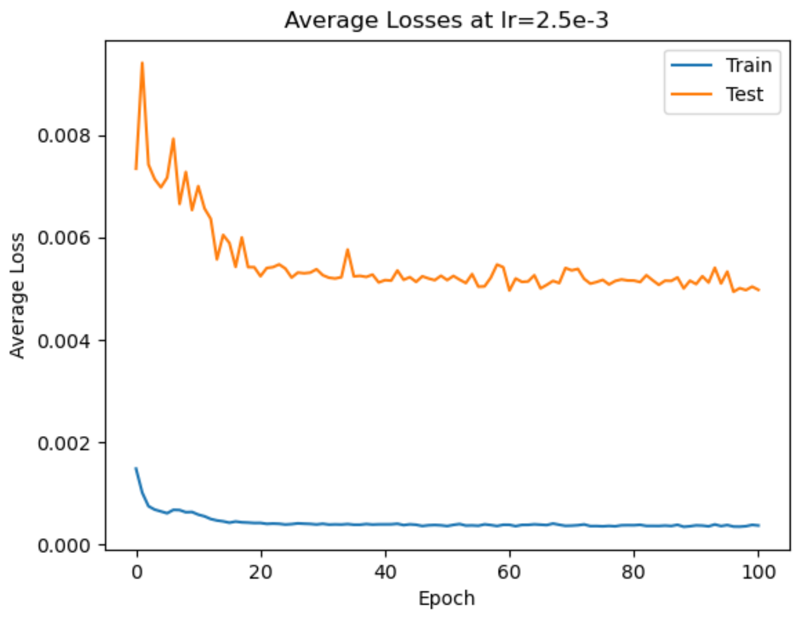


Figure 6.10: Average training and test losses per epoch, using the loss function L=0.5MSE+0.5MAE and a learning rate of 2.5e-3

Figure 6.10 shows the average losses for training and testing PEBCC at a learning rate of $2.5e-3$ for 101 epochs using the combined loss function with a balancing factor of $\alpha = 0.5$. The learning rate used has been approximated using the aforementioned methodology presented in Section 6.3.2. The average losses are displayed on the y-axis and the number of epochs on the x-axis. The blue line demarcates the average training losses and the orange line demarcates the respective test losses. The figure shows average losses that stabilise at around 20 epochs. Hence, 20 epochs should be an efficient amount at this learning rate. Note that the network does not seem overfitted after 100 epochs. As the test losses stay relatively constant after 20 epochs instead of increasing. The average losses are comparatively lower than those of Figure 6.7.

Figure 6.11 shows the conductivity reconstruction results after training PEBCC with the updated loss function at epoch 20. The figure shows from left to right, top to bottom, the ground truth-, input-, updated output conductivity maps and the relative error, the input error and the update error. The relative error presents the most direct measure of whether changing the loss function, tuning the learning rate and epoch amount improved the network. The relative error in the bottom left corner shows a highly similar improvement in the white matter (0.7) and CSF (0.8) areas of the brain. The grey matter, however, remains outside PEBCC's sensitivity.

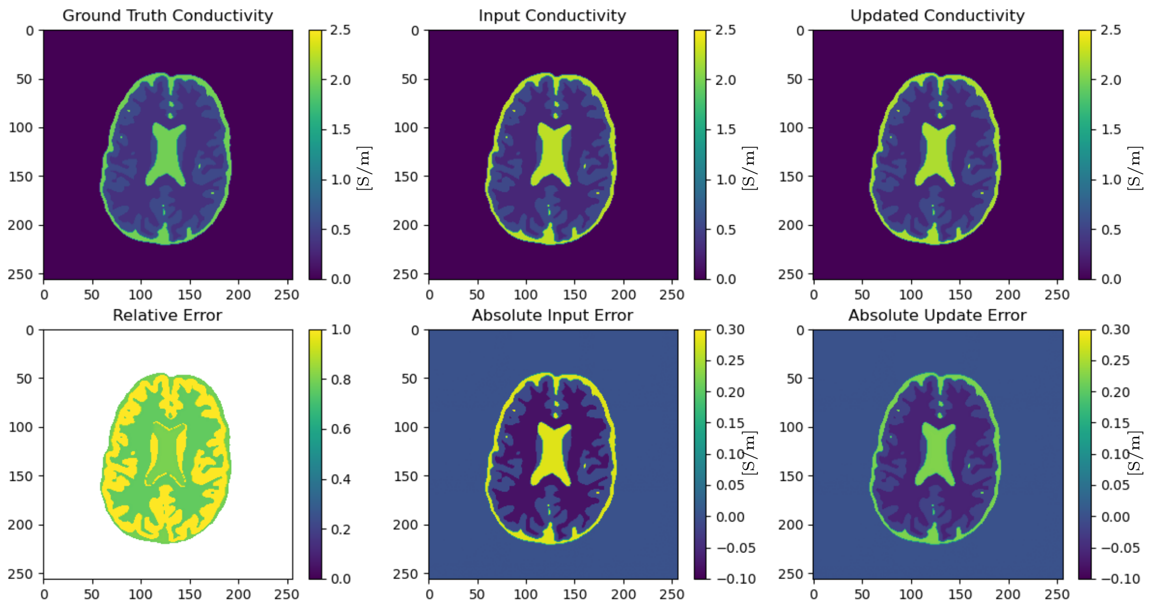


Figure 6.11: Evaluation results of training the PEBCC with the loss function L=0.5MSE+0.5MAE and a learning rate of 2.5e-3 at epoch 20. From the top left to the bottom right: the ground truth conductivity, the input conductivity, the updated conductivity output from the network, the relative error: the absolute update error relative to the absolute input error (Equation 6.6), the absolute input error: the difference between the input and the ground truth, the absolute update error: the difference between the updated conductivity and the ground truth.

Figure 6.12 displays the quantitative results of the relative error under test 2, with the relative error values displayed on the y-axis and the corresponding input error on the x-axis. The first notable difference between the results under test 1 (Figure 6.9) and test 2 (Figure 6.12) is the increased amount of violins present under test 2. Under test 2 more negative valued input errors are corrected within scope. However, all relative error violins with negative valued input error only lie partially in scope, with the exception of the input error -0.08. The mean relative error at an input error of -0.05 lies within the scope and therefore, on average yields improvement. The relative error violins with input errors -0.04 and -0.03 have the majority of their distribution outside of scope; generally causing a deterioration in results instead.

In general, the range of all violins has decreased compared to test 1's results, indicating a smaller spread in relative error distributions. Smaller ranges in relative errors are caused by smaller variations in corresponding output conductivities. Less variation in output conductivity translates to more predictable behaviour of the network, which is desirable. This overall improvement is likely attributed to the increase in training duration. Since longer training has shown an improvement in the average test losses in Figure 6.10.

The relative error violins of the higher input error values such as 0.13, 0.23 and 0.28,

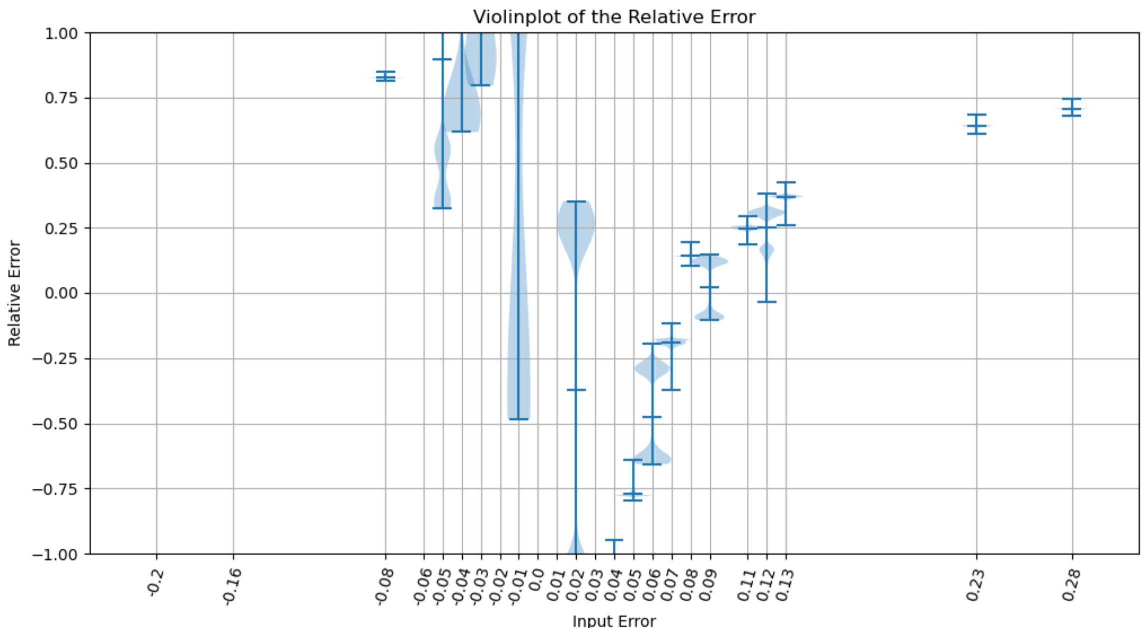


Figure 6.12: Violin plots displaying the relative error of the network under test 2

have a slightly higher mean than under test 1. In the middle ranges of 0.07 till 0.12 the mean has improved instead. The combination of these two results indicates that the adjusted loss function has the desired effect of providing a more overall improvement to the conductivity maps. Since more of the improvement has been shifted away from large input errors to smaller ones.

For a single conductivity correction, the input error sensitivity for test 2 equates to 0.05. For iterative use, the input error sensitivity equates to 0.07 when considering the mean and 0.13 in all cases. Hence, in iterative use, the adjustments of test 2 have yielded an improved sensitivity compared to test 1.

6.4.3 Results Test 3: Convolution Kernel Tuning

Figures 6.13, 6.14 and 6.15 show the average losses for the training, testing and out of distribution under Test 3, respectively, with the average loss values on the y-axis and the number of epochs on the x-axis. The training losses indicate that under the current settings, a convolution kernel size of 9, represented by the red line, performs the poorest out of the 5 tested sizes. The lowest losses are achieved with a convolution kernel size of 7, the green line, and the remaining options of 3 (blue line), 5 (orange line) and 11 (purple line) show a very similar result in average training losses.

The average test losses shown in Figure 6.14 are difficult to assess due to their wildly oscillating results.

Similarly, the average out of distribution losses shown in Figure 6.15 also show highly oscillatory results.

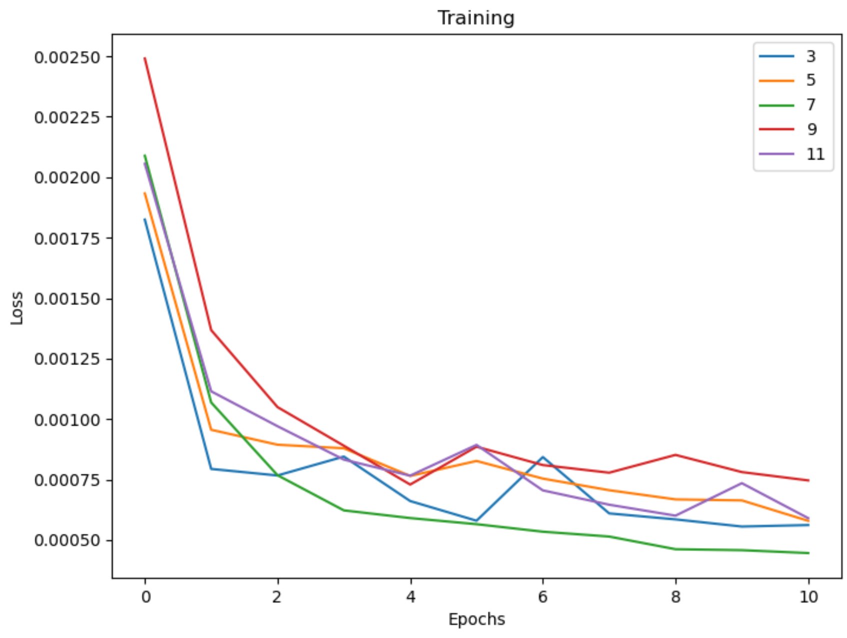


Figure 6.13: Average training losses at different convolution kernel sizes under Test 3

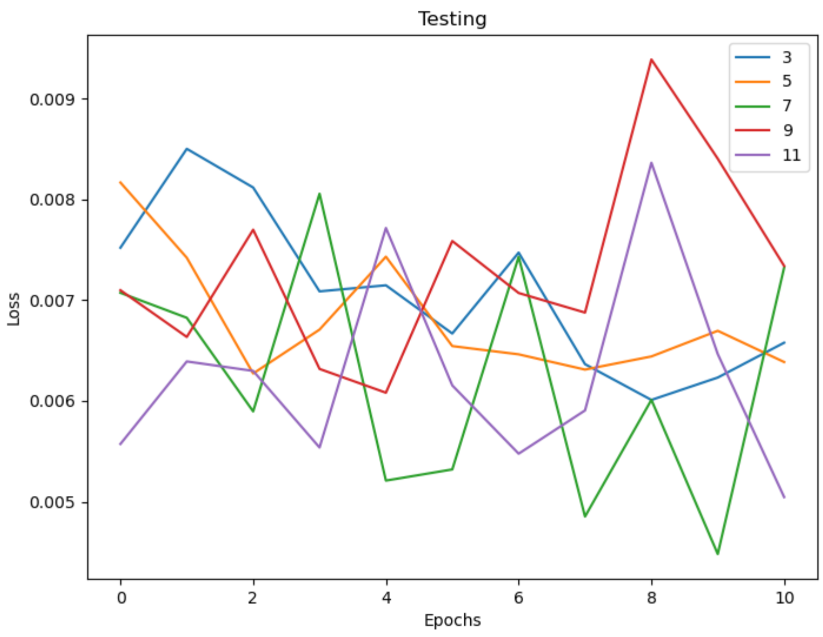


Figure 6.14: Average test losses at different convolution kernel sizes under Test 3

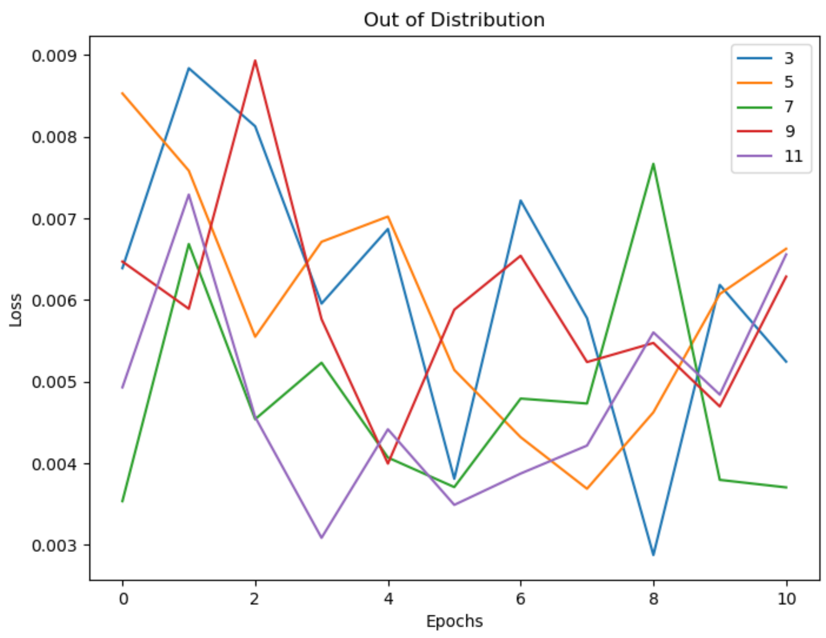


Figure 6.15: Average out of distribution losses at different convolution kernel sizes under Test 3

Hence, it is difficult to assess the performance of the different convolution kernel sizes by average losses alone and the learning rate needs to be adjusted to acquire insightful results.

Figures 6.16, 6.17, 6.18, 6.19 and 6.20 show the violin plots of the relative error of the network at convolution kernel sizes of 3, 5, 7, 9 and 11, respectively. For the assessment of their respective sensitivity, a violin has been included if the mean and the visible distribution are within 0 and 1. This means that at an assessed input error, the respective relative error is allowed to have outliers outside of the range between 0 and 1. Additional assessment criteria include the amount violins visible, including both partially and entirely visible violins. Lastly, the amount of violins completely within the scope of the figure has been reported. Table 6.1 reports these 3 criteria at every convolution kernel size between 3 and 11. A convolution kernel size of 3 yields the best sensitivity under the conditions of Test 3. Table 6.1 shows an increasing sensitivity value at higher kernel sizes, with size 7 being an outlier with a high jump in sensitivity to 0.28. Larger kernel sizes tend to feature a higher amount of visible violins but comparatively fewer of the visible violins are entirely within scope.

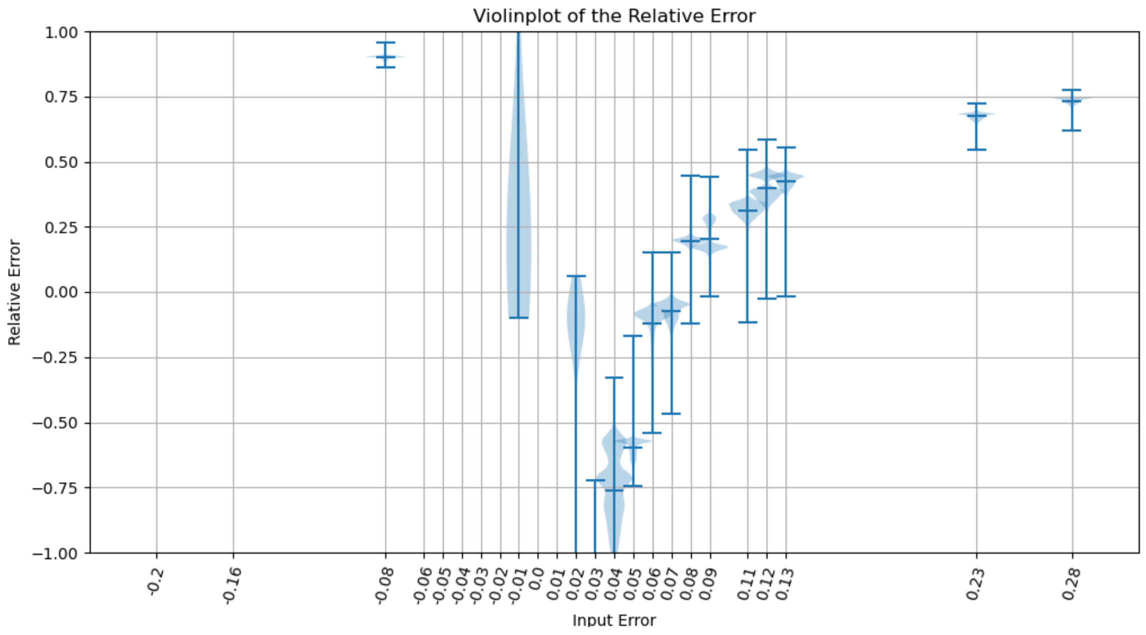


Figure 6.16: Violin plots of the relative error of the network under Test 3 and a convolution kernel size of 3

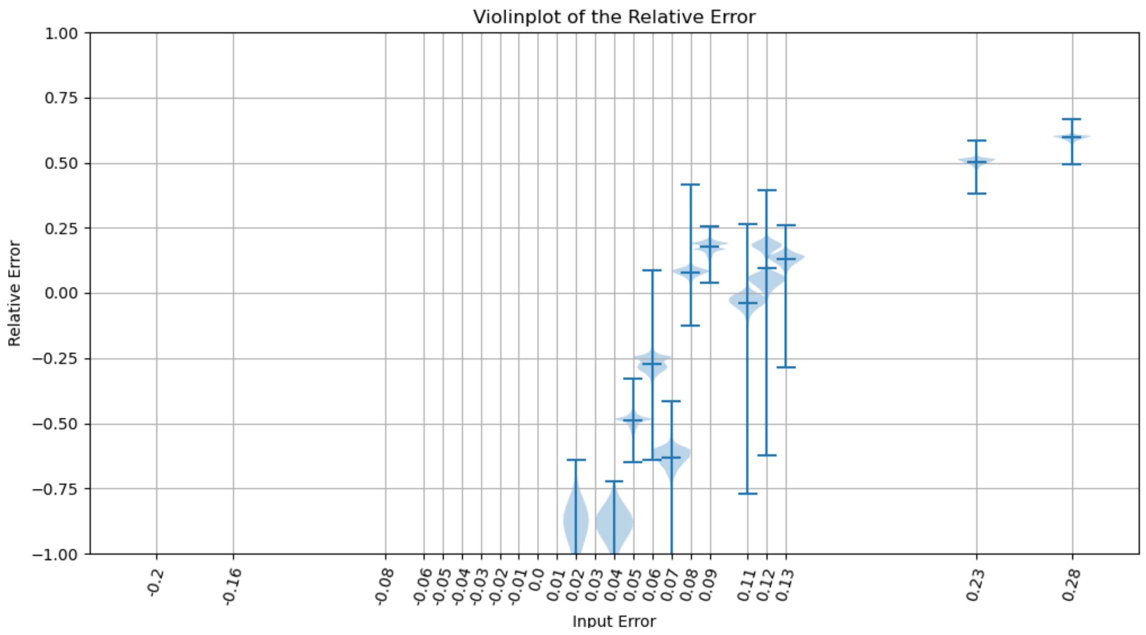


Figure 6.17: Violin plots of the relative error of the network under Test 3 and a convolution kernel size of 5

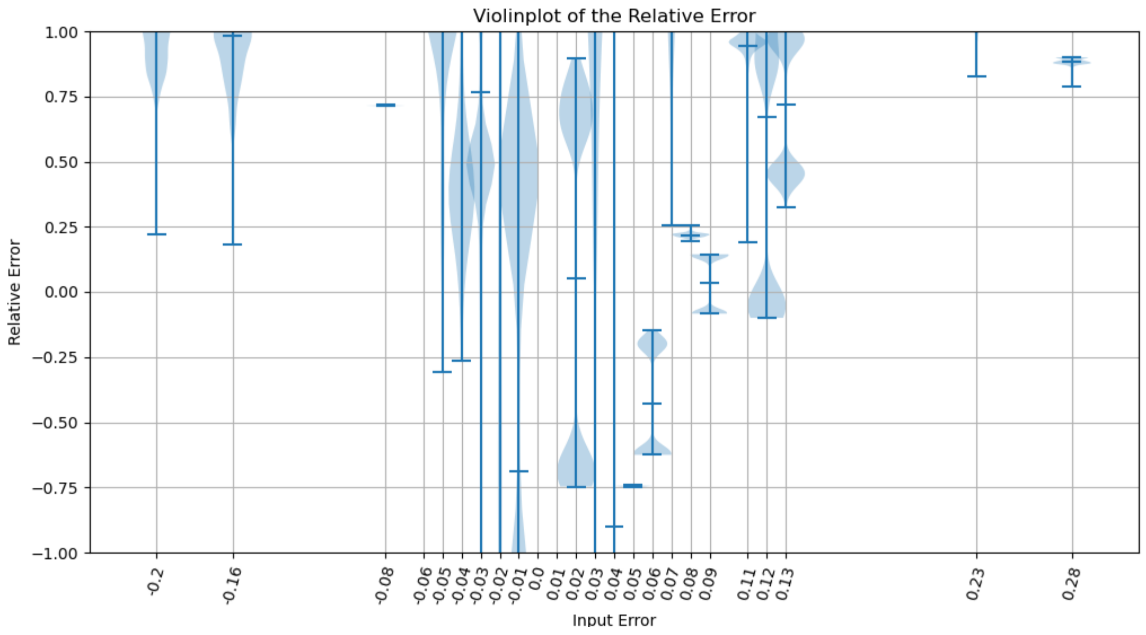


Figure 6.18: Violin plots of the relative error of the network under Test 3 and a convolution kernel size of 7

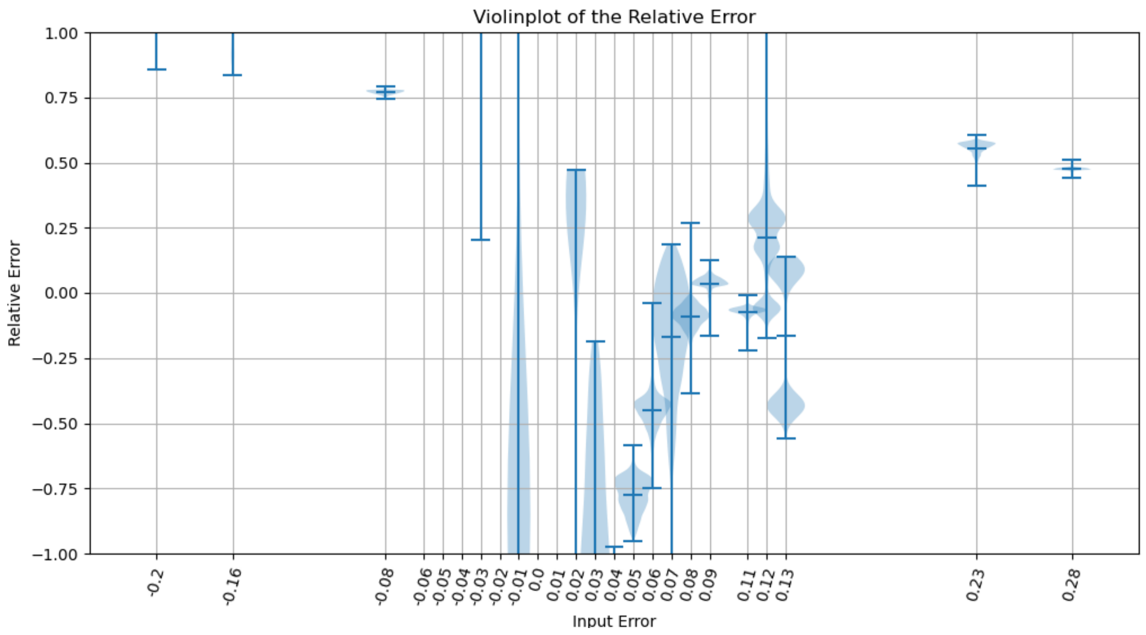


Figure 6.19: Violin plots of the relative error of the network under Test 3 and a convolution kernel size of 9

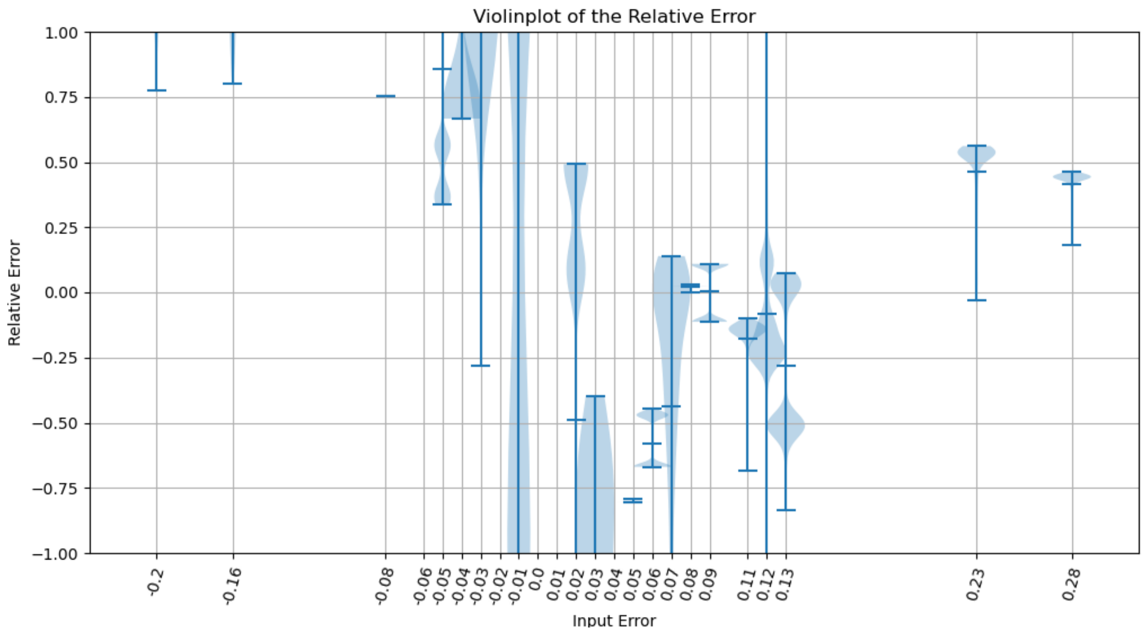


Figure 6.20: Violin plots of the relative error of the network under Test 3 and a convolution kernel size of 11

Kernel Size	Sensitivity	Visible Violins	Violins Within Scope
3	0.08	15	11
5	0.12	12	9
7	0.28	21	7
9	0.23	18	8
11	0.23	19	9

Table 6.1: Convolution kernel size assessment of relative error violin plots

Discussion And Conclusion 7

7.1 DLE-CSI Discussion

The discussion on DLE-CSI will be centred on the topic of image reconstruction quality rather than the intended goal of achieving a speedup over CSI. Discussing a speedup is deemed irrelevant if the reconstruction quality is still far from equivalent to CSI reconstructions.

The results from Section 5.4 which use the replacement update manner, have shown that using the replacement update manner results in a homogeneous reconstruction regardless of network implementation or training strategy. Hence, the replacement update manner is deemed an unsuitable implementation of DLE-CSI and specifically for use in consort with a RIM. Therefore, neural network implementations and training strategies will only be assessed under the use of the additive update manner. This means that the differences between the quadrant-based GRU implementation and the local patches-based GRU implementation cannot be properly assessed on reconstruction quality differences. Nonetheless, the local patches-based implementation was preferred due to its lack of reliance on any symmetry and its larger reduction in memory usage. The local patches implementation allows for a more controllable way of reducing memory usage by managing the amount of GRUs used. Hence, for extensions to the network, the local patches approach can more easily manage the additional memory usage.

The results from Test 3 and Test 4 showcase the differences between simultaneous iteration training and per iteration training. PIT performs slightly better in terms of average losses but these differences are minor in comparison to the range of loss values both training methods yield. The larger improvements of the PIT losses at the latter epochs do seem more promising when using a higher amount of iterations and epochs, for which validation is recommended. In terms of cost functional values, both training methodologies perform similarly, whereas in the reconstruction example, PIT shows a slightly larger amount of reconstructed tissue structure. Overall, PIT has a slight advantage over SIT as a training methodology. This is further accentuated in the results of Test 5, where the cost functional under SIT increases over the training duration, while for PIT it decreases. Besides the reconstruction quality advantages of PIT, PIT has a computational advantage in requiring a reduced amount of time to train in the earlier epochs due to using fewer iterations. In comparison, SIT's main advantage lies in the flexibility of not having the number of iterations coupled to the number of epochs used for training. An ideal training methodology would be to compute losses and backpropagate every iteration of DLE-CSI. This would train the network on each iteration independently of any previous iterations. This would also enable training for a much larger amount of iterations, as saved gradient information is limited to the current iteration. Hence, the memory usage would not accumulate according to the amount of iterations, as is currently the case. This methodology's disadvantage is its difficulty in implementation, having to decouple every DLE-CSI iteration.

Test 5 showcased the changes to the results when the network's depth and complexity doubles. The results showed improvement in all aspects with the exception of the cost functional when using SIT. Specifically when assessing the reconstruction example, a lot of information can be gained. Firstly, the white matter tissue structure starts to take shape in the conductivity reconstruction. It is still only a general outline of the area in which the white matter is present but a significant improvement has been achieved. The reconstructed white matter tissue can be considered a blurred or low frequency version of the ground truth. The finer details are missing and this could indicate the need to accommodate higher frequency components of the image into the network. The first recommended change would be to increase the amount of channels. Increasing the number of channels typically allows a neural network to learn a higher amount of different features. In addition, larger kernel sizes or increased network depth should be tested. These changes and especially the increase in the amount of channels, would drastically increase the amount of memory required. The increase in memory usage mostly arises from requiring larger or more GRUs to pass the additional information through. This issue could be (partially) mitigated by treating the additional data as a separate sequence and reusing the same GRU with the same weights on different data. Alternatively, skip connections could be used on an arbitrary amount of channels to bypass GRUs. These channels with skip connections would then only learn changes that are independent of specific iteration steps.

Secondly, the example EP reconstructions show that the network performs better at reconstructing conductivity than permittivity. The permittivity reconstructions rarely showcase significantly more than a homogeneous result. The conductivity is represented by the real part of the contrast and the permittivity is represented by the imaginary part of the contrast. Hence, keeping separate channels for real and imaginary data that do not interact, might improve the neural network's ability to improve both EPs to a similar extent. Alternatively, it could be investigated whether the real and imaginary parts can be decoupled completely by using two different neural networks specialised in reconstructing each EP.

Thirdly, the example conductivity reconstruction shows that the NN has a preference for reconstructing the white matter structures over the CSF structures. This is likely caused by the reconstruction achieving a smaller MSE loss when focusing on white matter, which is more abundant than CSF. The MSE should counteract this to a certain extent as it punishes outliers severely due to its squared relation to the absolute error. This indicates that the network is likely struggling to act on multiple features with finer details.

The cost functional values of Test 5 with SIT show that the cost functional values can increase while the average MSE losses decrease. This is reasonable as the MSE represents the variance of the error, which can be decreased while the mean error increases. However, the cost functional is used in the DLE-CSI algorithm as the metric to determine if the reconstruction is of sufficient quality. Hence, replacing the MSE loss with a different loss function that ensures a decrease in the cost functional is recommended. The cost functional cannot be used as a loss function in supervised learning due to the lack of ground truth incident fields required for a ground truth cost functional. The cost functional could be leveraged as a loss function for unsupervised learning, allowing DLE-CSI to directly learn from in-vivo data. However, this does sustain the reliance on simulated incident fields.

The cost functional values of Test 3, 4 and 5 are predominantly decreasing per iteration step and to a much lesser degree per epoch. Hence, using a larger amount of iterations should be investigated. More specifically, if we test the network with an arbitrarily large amount of iterations (e.g. 1000 iterations), whether training the network with 10 or 25 iterations achieves very different results. This teaches us whether training with larger iteration amounts is conducive to validating with a significantly higher iteration amount. If there is a limit to the usefulness of training with an increased iteration amount, then this would limit the increases in training time compared to training with a massive amount of iterations. Such knowledge is especially important since memory-wise the maximum number of iterations during training is about 40.

For all tests, the average test losses were lower than the average training losses. This could happen if the data in the training- and testing set are so similar that similar losses are achieved. The average training losses will then be higher due to the losses at the start of an epoch being higher than at the end. As the testing phase commences at the end of the epoch, it will have losses similar to the final losses of the training phase but without the higher losses at the start of the training phase. This is, however, unlikely to be the only cause since PEBCC uses the 1mm version of the same dataset and does not show this behaviour at all. Hence, the data should not be similar to such a degree that this behaviour is warranted. Considering the poor losses and reconstruction results, it is more likely that the DLE-CSI is not equipped to solve the presented problem in its current capacity. Meaning that the results are poor enough that the difference between a reconstructed set from training data and test data shows little difference in quality. Combined with the aforementioned reason of lower losses for testing with equivalent data should explain this behaviour.

A further point of investigation is concerned with typical use cases of RIMs. The assumption of CSI's gradient of the contrast function being an adequate replacement of the negative log-likelihood function's gradient should be researched. This assumption should be tested to verify its validity, which can be easily achieved by formulating a negative log-likelihood function and switching out the gradients.

The MSE of the reconstruction can be likened to the variance of an estimator, indicating the spread of reconstructed values from the ground truth values. Similarly, the root of the MSE (RMSE) can be interpreted as the standard deviation of the reconstructed values from the ground truth values. The observed RMSE of Test 3 at epoch 9 approximates to $\sqrt{1225} = 35$, a very large standard deviation value for ground truth values with single digits. The losses do not decrease much for the other tests, showing that in its current state DLE-CSI's performance is deemed far from sufficient. This is further accentuated by the reconstruction examples showing a severe lack of details in tissue structures and even the complete absence of reconstructed CSF. DLE-CSI still has potential with all the improvements suggested above. Most of these suggestions come at the cost of increased memory usage, which is a common theme when working with deep learning for 3D-MRI data. It is therefore quite important for the development of a technique such as DLE-CSI to optimise memory usage. It is recommended to continue researching the feasibility of DLE-CSI due to the many avenues of improvement available. If a RIM is concluded to not be fit for this type of problem, a different topology from Chapter 4 can be investigated. Such as a CRNN, which functions much like an RIM but uses convolution operations within the GRU, limiting the amount of weights required.

If for the current conditions, a functioning version of DLE-CSI can be achieved, the next step would be to validate DLE-CSI's performance on noisy data. In practice, data will never be noiseless and the ability of a MR-EPT method to deal with noise is of paramount importance.

If an adequate DLE-CSI can be achieved with the aforementioned improvements, the simulated incident fields will still limit its performance. Placing objects or patients in the MRI scanner loads the coils leading to inaccuracies when using simulated incident fields based on a free space. With a functioning DLE-CSI method, an attempt at indirectly improving the incident field simulations could be made with deep learning. However, no labels of 'true' incident fields are available for training the network directly in a supervised manner. Instead, the effect that a DL-produced incident field has on the output EPs of DLE-CSI could potentially be used to implicitly learn them. This could be considered indirect supervised learning. Somewhat similar concepts can be seen in other fields [52, 53, 54]. Instead of directly using simulated incident fields $B_{1;sim}^{+;inc}, E_{sim}^{inc}$ as input to DLE-CSI, they will be used as input to a NN together with the measured B_1^+ field to produce an updated version of the incident fields $B_{1;up}^{+;inc}, E_{up}^{inc}$. The NN aims to implicitly approximate the coil loading effects on the incident fields based on the measured B_1^+ field.

The feasibility of such a method is limited by three things. Firstly, the entirety of DLE-CSI needs to be incorporated as a NN to allow backpropagation to the point where the incident fields are simulated. Moving such a model to a GPU for accelerated computations might not be feasible due to increased memory usage. Secondly, the body outside the imaging domain influences the incident field. Hence, updating the simulated incident fields based on the image domain limited measurement of the B_1^+ field will limit its effectiveness. Thirdly, additional NN layers increase memory usage even further. Nonetheless, investigating improvements in incident field usage could be worthwhile.

7.2 PEBCC Discussion

The results of PEBCC shown in Chapter 6 displayed its capability to improve conductivity reconstructions to a certain extent for noiseless cases. For the best performing case of Test 2, a sensitivity of 0.05 when used a single time, 0.07 on average for iterative use and 0.13 at all times for iterative use, was observed. These results were observed for a kernel size of 3, and the use of a combined loss function consisting of both a MSE and MAE loss, with both carrying equal weight. However, these results require more extensive validation. The K-fold validation in Test 1, described in Subsection 6.3.1, resulted in a single fold outperforming the other 3 significantly. This displays that a leave one out approach might not properly represent the overall performance of the network. A K-fold validation of the optimally configured network is therefore recommended.

The results of Test 2 showed how adjusting the loss function from a pure MSE loss to a fifty-fifty of a MSE loss and MAE loss, improved the sensitivity but slightly degraded results for larger input errors. Further tuning of the balance between these different types of loss functions should be performed. A better sensitivity might still be achieved while keeping the relative error for larger input errors below 1.

A better sensitivity could be achieved with the same network settings if the network is trained to handle negative input errors. All relative error results have shown that within the current capabilities of the network, any negative input error is highly unlikely to yield an improved reconstructed conductivity value. Using the same dataset in a larger amount of combinations can already significantly increase the diversity in training data, including more negatively valued input errors. This means that within the same geometry, every set of conductivity- and phase maps are used as the ground truth in an alternating fashion, instead of only the 4^{th} set.

The training and testing assume that the geometry of the tissue is equivalent between input and ground truth. If in practice the input conductivity map was not reconstructed with the correct tissue geometry, errors might arise from the use of PEBCC. An example of altered tissue geometry in reconstructions could be the boundary errors occurring when using SH-EPT for reconstruction. Hence, PEBCC's response to altered tissue geometry should be investigated. Additionally, the effects of training PEBCC with different tissue geometry combinations of inputs and ground truths should be researched, as this would lead to wider applicability of PEBCC. If PEBCC can resolve tissue geometry differences relatively well, it could potentially be used to correct issues like boundary errors.

The effectiveness of iterative use of PEBCC is reliant on the mapping of the output conductivity to a corresponding phase map. Any newly introduced deficiencies could degrade the performance and worsen the sensitivity. Additionally, outputs from PEBCC outside of the sensitivity range should be ignored in order to use PEBCC iteratively. This requires sensitivity information based on the phase error rather than the input conductivity error, as the input conductivity error is not available in practice. After an iteration of PEBCC, the newly computed phase error for the next iteration should be compared with the previous phase error. Wherever the phase error has increased, the conductivity output should be replaced with its former value. Afterwards, the phase maps should be recomputed accordingly. Hence, a phase error sensitivity should be determined.

To further assess the capabilities of PEBCC, it should be tested on noisy data. The sensitivity should be determined for a wide range of noise levels in order to truly assess its usability. Similarly, any recommended improvements and adjustments should be assessed on noisy data rather than noiseless data. The kernel size of 3 turned out to be optimal in the noiseless case. With noise, this might differ, as a larger kernel could be able to average out noise better than a small kernel size due to accessing a larger sample size of the noise. Smaller kernel sizes do have the advantage of lower memory usage and no need for segmentation.

It is recommended to investigate the amount of encoder- and decoder blocks in the design of the U-net. If any additional blocks improve the sensitivity, it might be worth the increase in memory usage. Alternatively, if using fewer blocks barely degrades performance, it could result in a smaller more practical neural network.

It is difficult to properly assess PEBCC's performance on tumour reconstructions with the current results. In order to assess the average losses in comparison to usage on healthy tissue, data with EP differences in both healthy and tumour tissue should be used. Furthermore, it should be assessed whether the relative error of tumour tissues within the sensitivity range acts as an outlier. If so, such an outlier could potentially be used as a way of tumour detection.

7.3 Project And Programming Methodology

For similar future projects, where the program's operation is rather linear, it is recommended to take more of a functional programming approach over an object-oriented approach. Hiding variables and operators behind interfaces generally made it harder to track where changes occurred. This increased the difficulty of debugging and tracking the data flow. Object-orientated programming is more suited when a program has more variable ways in which it is run, compared to DLE-CSI in which the same order of actions is performed every time. Functional programming refers to using mainly functions and expressions which do not internally change data or states. Meaning that any changes are returned as a new variable by the function and no input variables are altered within. Hence, functional programming is more insightful to the user as the data flow can be easily tracked from the main function.

The long training times of DLE-CSI of more than a week, motivated efficient use of training time by preparing for the next training before the last one has been completed. However, a more thorough assessment of current results should have been performed first in order to not lose time on training a faulty network, which could have been predicted in advance.

7.4 Conclusion

We explored two methods of combining deep learning with direct- and inverse optimization EPT methods. The inverse optimization method, three-dimensional contrast source inversion, is combined with a neural network with the aim of reducing the computational load while maintaining reconstruction quality. The direct method, simplified Helmholtz EPT, is used as a basis for a neural network that improves reconstructions of conductivity by using measured phase data.

The results on deep learning enhanced contrast source inversion reported in this work, show a significantly lower reconstruction quality than the 3D-CSI it is based on. In its current capacity, DLE-CSI is unable to properly reconstruct the complex tissue structures of the human brain. The current version of DLE-CSI has many potential avenues of improvement which are recommended to be explored. These recommendations include an increase in feature channels, training the neural network independently per iteration, adjusting the loss function, increasing the iteration amount and increasing the depth of the neural network. If all avenues of using a RIM are deemed inadequate, other neural network topologies such as cascaded CNNs can still be investigated as potential DLE-CSI implementations. Only if an equivalent reconstruction quality to 3D-CSI's reconstructions can be achieved, can DLE-CSI's potential improvements in computational time be assessed. Furthermore, assessment of the noise robustness of DLE-CSI is an important next step in the validation process.

The implementation of phase error based conductance correction has shown to correct conductivity maps with a conductivity input error sensitivity of 0.05 for single time use and 0.13 for iterative usage. These results can potentially be improved further by tuning the various network parameters and by having the network learn to handle negative conductivity input errors. For practical use, an assessment of the phase error sensitivity should be provided, as outside of training the conductivity input error is unknown. Additionally, the influence of different mapping functions on the sensitivity should be researched and included in the performance of PEBCC. Ultimately, PEBCC needs to be assessed on noisy data to research its feasibility in practice.

Bibliography

[1] A. De Schepper, L. De Beuckeleer, J. Vandevenne, and J. Somville, “Magnetic resonance imaging of soft tissue tumors,” *European radiology*, vol. 10, no. 2, pp. 213–223, 2000.

[2] H. Van Duinen, R. Renken, N. Maurits, and I. Zijdewind, “Effects of motor fatigue on human brain activity, an fmri study,” *Neuroimage*, vol. 35, no. 4, pp. 1438–1449, 2007.

[3] U. Katscher and C. A. van den Berg, “Electric properties tomography: Biochemical, physical and technical background, evaluation and clinical applications,” *NMR in Biomedicine*, vol. 30, no. 8, p. e3729, 2017.

[4] R. Stollberger and P. Wach, “Imaging of the active b1 field in vivo,” *Magnetic resonance in medicine*, vol. 35, no. 2, pp. 246–251, 1996.

[5] E. Michel, D. Hernandez, and S. Y. Lee, “Electrical conductivity and permittivity maps of brain tissues derived from water content based on t1-weighted acquisition,” *Magnetic Resonance in Medicine*, vol. 77, no. 3, pp. 1094–1103, 2017. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/mrm.26193

[6] J. Liu, Y. Wang, U. Katscher, and B. He, “Electrical properties tomography based on b_1 maps in mri: Principles, applications, and challenges,” *IEEE Transactions on Biomedical Engineering*, vol. 64, no. 11, pp. 2515–2530, 2017.

[7] X. Zhang, J. Liu, and B. He, “Magnetic-resonance-based electrical properties tomography: a review,” *IEEE reviews in biomedical engineering*, vol. 7, pp. 87–96, 2014.

[8] T. Voigt, O. Vaterlein, C. Stehning, U. Katscher, and J. Fiehler, “In vivo glioma characterization using mr conductivity imaging,” in *Proceedings of the 19th Scientific Meeting of the International Society of Magnetic Resonance in Medicine (ISMRM’11)*, vol. 127, 2011.

[9] A. L. van Lier, J. M. Hoogduin, D. L. Polders, V. O. Boer, J. Hendrikse, P. A. Robe, P. A. Woerdeman, J. Lagendijk, P. R. Luijten, and C. Van Den Berg, “Electrical conductivity imaging of brain tumours,” in *Proceedings of the 19th Annual Meeting of ISMRM, Montreal, Canada*, 2011, p. 4464.

[10] A. Surowiec, S. Stuchly, J. Barr, and A. Swarup, “Dielectric properties of breast carcinoma and the surrounding tissues,” *IEEE Transactions on Biomedical Engineering*, vol. 35, no. 4, pp. 257–263, 1988.

[11] I. Hancu, J. C. Roberts, S. Bulumulla, and S.-K. Lee, “On conductivity, permittivity, apparent diffusion coefficient, and their usefulness as cancer markers at mri frequencies,” *Magnetic Resonance in Medicine*, vol. 73, no. 5, pp. 2025–2029, 2015. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/mrm.25309

[12] E. Balidemaj, A. L. van Lier, H. Crezee, A. J. Nederveen, L. J. Stalpers, and C. A. Van Den Berg, “Feasibility of electric property tomography of pelvic tumors at 3t,” *Magnetic resonance in medicine*, vol. 73, no. 4, pp. 1505–1513, 2015.

[13] E. Balidemaj, P. de Boer, H. Crezee, R. Remis, S. Lukas, A. Nederveen, and C. A. van den Berg, “In vivo reconstructed conductivity values of cervical cancer patients based on ept at 3t mri,” in *Proceedings of the 23rd Annual Meeting ISMRM*, 2013.

[14] R. Leijsen, W. Brink, C. van den Berg, A. Webb, and R. Remis, “Electrical properties tomography: A methodological review,” *Diagnostics*, vol. 11, no. 2, p. 176, 2021.

[15] K. G. Hollingsworth, “Reducing acquisition time in clinical mri by data undersampling and compressed sensing reconstruction,” *Physics in Medicine & Biology*, vol. 60, no. 21, p. R297, 2015.

[16] T. Voigt, U. Katscher, and O. Doessel, “Quantitative conductivity and permittivity imaging of the human brain using electric properties tomography,” *Magnetic Resonance in Medicine*, vol. 66, no. 2, pp. 456–466, 2011.

[17] A. L. van Lier, D. O. Brunner, K. P. Pruessmann, D. W. Klomp, P. R. Luijten, J. J. Lagendijk, and C. A. van den Berg, “B phase mapping at 7 t and its application for in vivo electrical conductivity mapping,” *Magnetic Resonance in Medicine*, vol. 67, no. 2, pp. 552–561, 2012. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/mrm.22995

[18] U. Katscher, T. Voigt, C. Findeklee, P. Vernickel, K. Nehrke, and O. Doessel, “Determination of electric conductivity and local sar via b1 mapping,” *IEEE transactions on medical imaging*, vol. 28, no. 9, pp. 1365–1374, 2009.

[19] S. Mandija, A. Sbrizzi, U. Katscher, P. R. Luijten, and C. A. van den Berg, “Error analysis of helmholtz-based mr-electrical properties tomography,” *Magnetic Resonance in Medicine*, vol. 80, no. 1, pp. 90–100, 2018. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/mrm.27004

[20] D. K. Sodickson, L. Alon, C. M. Deniz, R. Brown, B. Zhang, G. C. Wiggins, G. Y. Cho, N. B. Eliezer, D. S. Novikov, R. Lattanzi *et al.*, “Local maxwell tomography using transmit-receive coil arrays for contact-free mapping of tissue electrical properties and determination of absolute rf phase,” in *Proceedings of the 20th Annual Meeting of ISMRM.* ISMRM Concord California, USA, 2012, p. 388.

[21] D. K. Sodickson, L. Alon, C. M. Deniz, N. Ben-Eliezer, M. Cloos, L. A. Sodickson, C. M. Collins, G. C. Wiggins, and D. S. Novikov, “Generalized local maxwell tomography for mapping of electrical property gradients and tensors,” in *Proceedings of 21th Annual Meeting ISMRM, Salt Lake City, USA*, vol. 4175, 2013.

[22] X. Zhang, S. Schmitter, J. Liu, P.-F. Van de Moortele, and B. He, “Local sar estimation for human brain imaging using multi-channel transceiver coil at 7 t,” in *Proc. ISMRM*, 2013, p. 288.

[23] R. L. Leijsen, W. M. Brink, C. A. Van Den Berg, A. G. Webb, and R. F. Remis, "3-d contrast source inversion-electrical properties tomography," *IEEE transactions on medical imaging*, vol. 37, no. 9, pp. 2080–2089, 2018.

[24] L. Guo, J. Jin, C. Liu, F. Liu, and S. Crozier, "An efficient integral-based method for three-dimensional mr-ept and the calculation of the rf-coil-induced B_z field," *IEEE Transactions on Biomedical Engineering*, vol. 65, no. 2, pp. 282–293, 2018.

[25] R. Hong, S. Li, J. Zhang, Y. Zhang, N. Liu, Z. Yu, and Q. H. Liu, "3-d mri-based electrical properties tomography using the volume integral equation method," *IEEE Transactions on Microwave Theory and Techniques*, vol. 65, no. 12, pp. 4802–4811, 2017.

[26] S. Li, R. Hong, N. Liu, J. Zhang, L. Chen, Y. Zhang, Z. Yu, and Q. H. Liu, "Three-dimensional mr reconstruction of high-contrast magnetic susceptibility by the variational born iterative method based on the magnetic field volume integral equation," *Magnetic Resonance in Medicine*, vol. 79, no. 2, pp. 923–932, 2018. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/mrm.26760

[27] J. E. C. Serrallés, L. Daniel, J. K. White, D. K. Sodickson, R. Lattanzi, and A. G. Polimeridis, "Global maxwell tomography: A novel technique for electrical properties mapping based on mr measurements and volume integral equation formulations," in *2016 IEEE International Symposium on Antennas and Propagation (APSURSI)*, 2016, pp. 1395–1396.

[28] J. E. Serrallés, I. I. Giannakopoulos, B. Zhang, C. Ianniello, M. A. Cloos, A. G. Polimeridis, J. K. White, D. K. Sodickson, L. Daniel, and R. Lattanzi, "Noninvasive estimation of electrical properties from magnetic resonance measurements via global maxwell tomography and match regularization," *IEEE Transactions on Biomedical Engineering*, vol. 67, no. 1, pp. 3–15, 2019.

[29] I. I. Giannakopoulos, J. E. Serrallés, L. Daniel, D. K. Sodickson, A. G. Polimeridis, J. K. White, and R. Lattanzi, "Magnetic-resonance-based electrical property mapping using global maxwell tomography with an 8-channel head coil at 7 tesla: a simulation study," *IEEE Transactions on Biomedical Engineering*, vol. 68, no. 1, pp. 236–246, 2020.

[30] E. Balidemaj, C. A. van den Berg, J. Trinks, A. L. van Lier, A. J. Nederveen, L. J. Stalpers, H. Crezee, and R. F. Remis, "Csi-ept: A contrast source inversion approach for improved mri-based electric properties tomography," *IEEE transactions on medical imaging*, vol. 34, no. 9, pp. 1788–1796, 2015.

[31] A. Zhang, Z. C. Lipton, M. Li, and A. J. Smola, "Dive into deep learning," *arXiv preprint arXiv:2106.11342*, 2021.

[32] N. Hampe, U. Katscher, C. A. Van den Berg, K. K. Tha, and S. Mandija, "Investigating the challenges and generalizability of deep learning brain conductivity mapping," *Physics in Medicine & Biology*, vol. 65, no. 13, p. 135001, 2020.

[33] R. Leijsen, C. van den Berg, A. Webb, R. Remis, and S. Mandija, "Combining deep learning and 3d contrast source inversion in mr-based electrical properties tomography," *NMR in Biomedicine*, vol. 35, no. 4, p. e4211, 2022.

[34] X. Chen, Z. Wei, M. Li, and P. Rocca, "A review of deep learning approaches for inverse scattering problems (invited review)," *Progress In Electromagnetics Research*, vol. 167, pp. 67–81, 2020.

[35] S. Mandija, E. F. Meliadò, N. R. Huttinga, P. R. Luijten, and C. A. van den Berg, "Opening a new window on mr-based electrical properties tomography with deep learning," *Scientific reports*, vol. 9, no. 1, pp. 1–9, 2019.

[36] S. Gavazzi, C. A. van den Berg, M. H. Savenije, H. P. Kok, P. de Boer, L. J. Stalpers, J. J. Lagendijk, H. Crezee, and A. L. van Lier, "Deep learning-based reconstruction of in vivo pelvis conductivity with a 3d patch-based convolutional neural network trained on simulated mr data," *Magnetic resonance in medicine*, vol. 84, no. 5, pp. 2772–2787, 2020.

[37] R. Guo, X. Song, M. Li, F. Yang, S. Xu, and A. Abubakar, "Supervised descent learning technique for 2-d microwave imaging," *IEEE Transactions on Antennas and Propagation*, vol. 67, no. 5, pp. 3550–3554, 2019.

[38] V. Khoshdel, A. Ashraf, and J. LoVetri, "Enhancement of multimodal microwave-ultrasound breast imaging using a deep-learning technique," *Sensors*, vol. 19, no. 18, 2019. [Online]. Available: https://www.mdpi.com/1424-8220/19/18/4050

[39] K. Lønning, P. Putzky, J.-J. Sonke, L. Reneman, M. W. Caan, and M. Welling, "Recurrent inference machines for reconstructing heterogeneous mri data," *Medical image analysis*, vol. 53, pp. 64–78, 2019.

[40] C. Qin, J. Schlemper, J. Caballero, A. N. Price, J. V. Hajnal, and D. Rueckert, "Convolutional recurrent neural networks for dynamic mr image reconstruction," *IEEE transactions on medical imaging*, vol. 38, no. 1, pp. 280–290, 2018.

[41] F. Knoll, K. Hammernik, E. Kobler, T. Pock, M. P. Recht, and D. K. Sodickson, "Assessment of the generalization of learned image reconstruction and the potential for transfer learning," *Magnetic resonance in medicine*, vol. 81, no. 1, pp. 116–128, 2019.

[42] E. Sabidussi, S. Klein, M. Caan, S. Bazrafkan, A. den Dekker, J. Sijbers, W. Niessen, and D. Poot, "Recurrent inference machines as inverse problem solvers for mr relaxometry," *Medical Image Analysis*, vol. 74, p. 102220, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1361841521002656

[43] P. R. Stijnman, S. Mandija, P. S. Fuchs, C. A. van den Berg, and R. F. Remis, "Transceive phase corrected 2d contrast source inversion-electrical properties tomography," *Magnetic Resonance in Medicine*, vol. 85, no. 5, pp. 2856–2868, 2021.

[44] J. F. Dooley and K. Dooley, *Software Development, Design and Coding*. Springer, 2017.

[45] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[46] V. Monga, Y. Li, and Y. C. Eldar, "Algorithm unrolling: Interpretable, efficient deep learning for signal and image processing," *IEEE Signal Processing Magazine*, vol. 38, no. 2, pp. 18–44, 2021.

[47] K. Neshatpour, H. Homayoun, and A. Sasan, "Icnn: The iterative convolutional neural network," *ACM Transactions on Embedded Computing Systems (TECS)*, vol. 18, no. 6, pp. 1–27, 2019.

[48] W. Yao, Z. Zeng, C. Lian, and H. Tang, "Pixel-wise regression using u-net and its application on pansharpening," *Neurocomputing*, vol. 312, pp. 364–371, 2018. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0925231218307008

[49] Y. Sanghvi, Y. Kalepu, and U. K. Khankhoje, "Embedding deep learning in inverse scattering problems," *IEEE Transactions on Computational Imaging*, vol. 6, pp. 46–56, 2019.

[50] P. R. S. Stijnman, Stefano Mandija, P. S. Fuchs, C. A. T. van den Berg, and R. F. Remis, "Transceive phase corrected 2d contrast source inversion-electrical properties tomography," *Magnetic Resonance in Medicine*, vol. 85, no. 5, pp. 2856–2868, 2021. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/mrm.28619

[51] A. Borsic, I. Perreard, A. Mahara, and R. J. Halter, "An inverse problems approach to mr-ept image reconstruction," *IEEE transactions on medical imaging*, vol. 35, no. 1, pp. 244–256, 2015.

[52] Y. Zhang, N. Charoenphakdee, and M. Sugiyama, "Learning from indirect observations," *arXiv preprint arXiv:1910.04394*, 2019.

[53] J. K. V. Tan, I. Budvytis, and R. Cipolla, "Indirect deep structured learning for 3d human body shape and pose prediction," 2017.

[54] Y. Chen and D. Zhang, "Physics-constrained indirect supervised learning," *Theoretical and Applied Mechanics Letters*, vol. 10, no. 3, pp. 155–160, 2020.