# *Transmission-Line Matrix (TLM) modelling of neural fields*

*By*

*Momcilo Prodanovic, MEng.*

A thesis submitted for the Degree of
Master of Engineering Science (Research)

Department of Electrical and Computer Systems Engineering
Monash University
Melbourne, Victoria 3800
Australia

December, 2017

# Copyright notice

# Abstract

Brain structure and dynamics have been the focus of a vast number of recent scientific studies. Numerous models have been developed for the purpose to describe the neural processes at different scales, ranging from the substructures of the individual neurons, through neural network models at the microscopic level describing the interconnections between the individual neurons with varying degrees of idealisation, up to the mesoscopic models explaining how the neural populations interact to the macroscopic neural field models informing us about the whole brain dynamics and the interactions between the large-scale neural systems such as the cortical regions, the thalamus, and the brain stem. In the early days of neuroscience, models had to be kept simpler so that results could be obtained analytically. The recent development of powerful computers has allowed researchers to create more realistic, but also more complex models based on numerical simulation methods, avoiding limitations and simplified assumptions usually built in analytical solutions.

Depending on the model concept used, there are many ways the equations can be solved numerically. One approach is to replace the equations by analogue models and probably the best-known example is the usage of electrical network to mimic the physical problem where solution could be obtained using conventional circuit analysis techniques in either the time or frequency domains. The most elegant electrical equivalent network numerical method is the Transmission-Line Matrix method (TLM) that leads to a simple numerical discretisation scheme.

In this thesis, the feasibility to numerically solve the inhomogeneous damped wave equations using TLM techniques is explored. The equations are used in a multiscale neural field brain model, called NeuroField, to represent axonal propagation of activity through the cortex. The hypothesis tested was if the usage of TLM leads to more understandable and efficient brain modelling and what the cost in computer resources for those benefits is. This approach differs from the currently used Finite Difference (FD) numerical method in NeuroField by providing the electrical equivalent network where all the NeuroField model parameters have analogues in electrical elements of the TLM node, thus enabling better understanding of the physical implications of discretisation and of the model.

The numerical approximations of NeuroField damped wave equations developed and solved in this thesis by TLM simulations show a great compatibility with FD. In future, the developed, TLM based NeuroField model, can be used for building a brain-on-the-chip for in-silico brain experimentation, which will greatly help the advancement of neuroscience.

# Declaration

This thesis contains no material which has been accepted for the award of any other degree or diploma at any university or equivalent institution and that, to the best of my knowledge and belief, this thesis contains no material previously published or written by another person, except where due reference is made in the text of the thesis.

Momcilo Prodanovic

Melbourne, Australia

December, 2017

# Acknowledgements

I would like to thank my main supervisor, Professor Arthur Lowery, for his guidance and encouragement throughout my research work. Without his supervision and expert advice this thesis would not have been possible.

I would also like to thank my associate supervisor, Professor Thomas Drummond, for his valuable suggestions.

Special thanks to Professor Peter Robinson and Dr Paula Sanz-Leon from the School of Physics at the University of Sydney for providing me the access to their NeuroField software and their invaluable support and comments whenever I had some doubts about the program.

Sincere thanks to Monash University, Faculty of Engineering and the Department of Electrical and Computer Systems Engineering (ECSE) for supporting me academically and financially for the duration of my Master research.

I also want to thank to The Australian Research Council (ARC) Centre of Excellence for Integrative Brain Function (CIBF) for the financial support to travel to the Centre's meetings and the opportunity to collaborate with the great scientists in the neuroscience field within the Centre.

Thanks to all my new friends and colleagues at ECSE (Harish Vangala, Jignesh Jokhakar, Dilpreet Buxi and Callum Laurenson) for their emotional support and the productive technical discussions throughout my candidature.

Finally, I must express my very profound gratitude to my parents and to my spouse Emma, for providing me with unfailing support and continuous encouragement throughout my years of study and through the process of researching and writing this thesis. This accomplishment would not have been possible without them. Thank you.

Momcilo Prodanovic

# Contents

# 1. Introduction

Brain structure and dynamics have been the focus of a vast number of recent scientific studies (Bower & Beeman, 1998; Breakspear, Jirsa, & Deco, 2010; Dayan & Abbott, 2001; Deco, Jirsa, Robinson, Breakspear, & Friston, 2008; Friston & Dolan, 2010; Northrop, 2001). Numerous models have been developed for the purpose to describe the neural processes at different scales, starting from the substructures of the individual neurons (Brette et al., 2007; Carnevale & Hines, 2006; Dayan & Abbott, 2001; Deco et al., 2008; Gerstner & Kistler, 2002; Gewaltig & Diesmann, 2007; Northrop, 2001). Neural network models at the microscopic level describe the interconnections between the individual neurons with varying degrees of idealisation (Bower & Beeman, 1998; Brette et al., 2007; Carnevale & Hines, 2006; Gerstner & Kistler, 2002; Goodman & Brette, 2009; McLaughlin, Shapley, Shelley, & Wielaard, 2000; Tapson et al., 2013), whereas the mesoscopic models explain how the neural populations interact (Freeman, 2012; Ritter, Schirner, McIntosh,

& Jirsa, 2013), for example, in cortical columns. Finally, at the macroscopic scale, the neural field models inform us about the whole brain dynamics and the interactions between the large-scale neural systems such as the cortical regions, the thalamus, and the brain stem (Beurle, 1956; S. Coombes, 2010; Deco et al., 2008; Nunez, 1974).

In the early days of neuroscience, models had to be kept simpler so that results could be obtained analytically. The recent development of high-performance computers has allowed researchers to create and visualise more realistic, but also more complex models based on numerical simulation methods, avoiding limitations and simplified assumptions usually built in analytical solutions (Hoefer, 2012).

Depending on the model concept used, there are many ways the physical processes could be solved numerically (Cogan, O'Connor, & Pulko, 2005). One approach is to replace the equations by analogue models and probably the best-known example is the usage of electrical network to mimic the physical problem where solution could be obtained using conventional circuit analysis techniques in either the time or frequency domains. The most elegant electrical equivalent network numerical method is the Transmission-Line Matrix method (TLM) (P. B. Johns & Beurle, 1971) that leads to a simple and natural numerical discretisation scheme for electromagnetic field problems. The main difference between the TLM and other numerical methods, such as, the widely used, Finite Difference (FD) (Thom, 1961), is a discretisation approach. To use the FD method, the physical problem which is to be solved must have two levels of approximations: first it should be modelled by differential or integral equations and then this model is solved by numerical methods using purely mathematical discretisation approach (Cogan et al., 2005). On the other side, the TLM has a physical approach based on Huygens' principle

(Hoefer, 1985), where a continuous system is replaced by a network of transmission lines. The major advantage of the TLM over the FD method is that all the required discretisation is built-in in the initial model, which is then solved without any further approximation avoiding many anomalous effects that can arise in FD (Cogan et al., 2005). That makes TLM perfect method for analysing even the most complicated structures, boundaries and material properties. When simulating unstable systems, such as brain dynamics, numerical stability is particularly important. In TLM there are no problems with convergence, stability or spurious solutions and the method is limited only by the amount of memory storage required, which depends on the complexity of the TLM mesh (Hoefer, 1985). However, according to (Sadiku, 2009), the TLM method sometimes may have one important disadvantage over FD: programs using finite-difference time-domain (FDTD) method, introduced by Yee (Kane, 1966) in solving the electromagnetic (EM) field problems can almost be two times faster in CPU time than equivalent TLM programs under identical conditions and require less memory.

As a part of The Australian Research Council (ARC) Centre of Excellence for Integrative Brain Function (CIBF), which has been established in 2013 as an Australia-wide team of neuroscientists with their primary focuses on understanding how the human brain interacts with the world, through a collaboration with Prof. Peter Robinson and his Brain Dynamics group at The University of Sydney we gain access to their multiscale neural field brain model, called NeuroField (P. Sanz-Leon, 2017; Robinson, Rennie, Rowe, O'Connor, & Gordon, 2005). NeuroField models the interactions of spatially extended populations of neurons and can predict the spectral and time characteristics of brain electrical activity observable by electroencephalography (EEG), magnetoencephalography (MEG), functional magnetic resonance imaging (fMRI), electrocorticography (ECoG) and other

non-invasive imaging modalities (P. Sanz-Leon, 2017). The governing neural field equations in NeuroField are expressed as the Partial Differential Equations (PDE) for the inhomogeneous damped wave equations. These equations represent the axonal propagation of activity through the cortex and are numerically solved in NeuroField by applying the FD method (Robinson, Rennie, & Wright, 1997).

In this thesis, the feasibility to numerically solve the inhomogeneous damped wave PDEs using TLM techniques is explored. The hypothesis tested was whether the usage of TLM leads to more understandable and efficient brain modelling and, what the cost in computer resources for those benefits is. This approach differs from the currently used FD numerical method by providing the electrical equivalent network where all the NeuroField model parameters have analogues in electrical elements of TLM node, thus enabling better interpretation of the physical implications of discretisation and of the model. In order to compare the cost in computer resources of both methods, the main algorithm of NeuroField program, along with the FD approximation of the governing wave PDEs was translated from C++ into MATLAB.

The numerical approximations of NeuroField damped wave equations developed and solved in this thesis by TLM simulations show a great compatibility with FD method. Being a viable solution, the computational efficiency the TLM method is discussed. In the future, developed TLM based NeuroField model can be used for building a brain-on-the-chip for in-silico brain experimentation, which will greatly help the advancement of neuroscience.

The thesis is structured as follows: introduction to computational neuroscience and different types of neural models, including the neural field models is presented in Chapter 2. The in-depth explanation of the NeuroField algorithm is given in Chapter 3. The

theoretical background to TLM modelling is given in Chapter 4. In Chapter 5 we present the FD approach to numerically solve the hyperbolic PDEs and discuss the differences between five- and nine-point stencils used to approximate the Laplacian operator in 2D. Section 6.1 of Chapter 6 is the explanation of FD numerical method used to solve the governing PDEs in NeuroField. Then the proposed 2D TLM node which can be used to solve the PDEs is presented and its parameters are calculated. Within the same Chapter, the space and time discretisation, some methods' constraints and boundary conditions used in both numerical methods are discussed. Simulation comparisons and the discussion of the results are shown in Chapter 7, followed by the conclusion and recommendations for future work in Chapter 8. The MATLAB code for One-population NeuroField model using the TLM method is presented in Appendix A, and for the FD method in Appendix B.

# 2. Computational Neuroscience

Theoretical analysis and computational modelling in neuroscience are, according to (Dayan & Abbott, 2001), important tools for determining the functioning of nervous systems and in-depth understanding why they operate in particular ways. In this Chapter, the overview of some of the most important computational neuroscience models is presented, ranging from a single neuron models to the large scale neural simulators.

## 2.1. Models of neurons

Computational neuroscience has a long history, starting with the ground-breaking conductance-based, mathematical model of Hodgkin and Huxley, back in 1952, for the generation of the nerve action potential (Hodgkin & Huxley, 1952a, 1952b, 1952c, 1952d). The Hodgkin-Huxley (H-H) model dealt with events at the molecular and ionic levels on unit area of a giant squid axon membrane (Northrop, 2001). Unfortunately, the computational complexity of H-H like neuron models, such as Wilson-Cowan model (Hugh

R. Wilson, 1999; H. R. Wilson & Cowan, 1973), prevents us from using them for the simulation of even modestly-sized neural networks of a few hundred neurons because of the amount of computer time required. To combat this, while at the same time trying to preserve as much of the complex dynamics as possible, a variety of simpler phenomenological models has been developed (Eckhorn, Reitboeck, Arndt, & Dicke, 1990; Rulkov, Timofeev, & Bazhenov, 2004), which aim at keeping most of the dynamical effects produced by voltage-gated channels using the equations that are pushing the models further away from the physiological mechanisms (Wells, 2005).

Models that describe the membrane potential of a neuron by a single variable $V$ are called single-compartment models (Dayan & Abbott, 2001). The simplest integrate-and-fire neuron model, proposed by Lapicque in 1907 (a good overview of the spiking neuron models can be found in Chapter 4 of (Gerstner & Kistler, 2002)) and the H-H model fall within this category. Although single-compartment models give a good approximation of a neuron, the membrane potentials can vary considerably over the surface of the cell membrane, especially for neurons with long and narrow processes, or rapidly changing membrane potentials. In those cases, the cable theory (Rall, 2011) must be used for the mathematical analysis of signal propagation within neurons. The problem is that the cable equation can be solved analytically only in relatively simple cases, but when the complexity of real membrane conductances are included, the membrane potential must be calculated numerically. This is done by splitting the neuron into separate regions or compartments, and approximating the continuous membrane potential by a discrete set of values representing the potentials with the different compartment. Each compartment should be small enough so that there is negligible variation of the membrane potential across it. These models are called multi-compartment models (Dayan & Abbott, 2001).

In the 1960s, digital computers were not user-friendly as tools for interactive modelling, thus early neural modellers like Leon Harmon at Bell Labs developed dedicated, compact transistor circuits to emulate spike generation, and various nonlinear RC low-pass networks to model the generation of excitatory and inhibitory postsynaptic potentials and signal conduction dendrites called neuromimes. Neuromimes offered the experimenter two advantages: they ran in real time and they could be easily interconnected with patch cords. Also, the modeller could listen to their spike outputs on headphones or speaker and detect the subtle changes in phase between two spike outputs, frequency changes, bursting, etc. More about the neuromimes can be found in the Chapter 3 of (Northrop, 2001).

## 2.2. Computational models of the brain

In a recent Special Issue of NeuroImage (Breakspear et al., 2010) authors tried to classify models into relevant categories, but they admitted that it wasn't an easy task. A term "computational model of the brain", as they say, is usually used for a range of computational techniques for the analysis of functional and anatomical neuroimaging data, but it also includes biophysical forward models that allow mapping between models and experimental data, as well as the models that address activity at smaller scales.

The big impact that computational neuroscience has had on neuroimaging over the past years is discussed by (Friston & Dolan, 2010) where they draw the distinction between models of the brain as a computational machine and computational models of neuronal dynamics. Computational machine models focus on optimal control and decision (game) theory to illustrate the role of functional models in imaging neuroscience. In terms of biophysical modelling, they are investigating dynamic causal modelling, with a special

emphasis on recent advances in neural-mass models for hemodynamic and electrophysiological time series. The neural field models, which are used for modelling the brain at large scales, which is necessary for interpreting EEG, fMRI, MEG and optical imaging data, are reviewed by Coombes (S. Coombes, 2010). The conclusion is that neural field models provide a good framework for unifying data from different imaging modalities. Starting with a description of neural mass models, they spatially extended cortical models of layered two-dimensional sheets with long range axonal connections mediating synaptic interactions. The models, based on differential, brain wave, equations are described and techniques for the analysis of such models, including how to determine the onset of spatio-temporal pattern forming instabilities, are reviewed. An overview of the open challenges for the development of multi-scale models that can integrate macroscopic models at large spatial scales with models at the microscopic scale is presented.

Furthermore, (Deco et al., 2008) have reviewed and integrated, in a unifying framework, a variety of computational approaches that have been used to characterize the dynamics of the cortex, as evidenced at different levels of measurement (scales). Modelling at the single neuron level is necessary because this is the level at which information is exchanged between the computing elements of the brain; the neurons. The mesoscopic models explain how the neural populations interact in cortical columns, while the macroscopic models can inform us about whole brain dynamics and interactions between large-scale neural systems such as the cortical regions, the thalamus, and the brain stem. Each level of description relates uniquely to neuroscience data, from single-unit recordings, through local field potentials to fMRI, EEG, and MEG (Deco et al., 2008).

### 2.2.1. Spiking neuron models

One of the current brain models is “Spaun” (Semantic Pointer Architecture Unified Network), a large-scale spiking neural network model of the functioning brain (Eliasmith et al., 2012; Terrence C. Stewart, Bekolay, & Eliasmith, 2012; T. C. Stewart & Eliasmith, 2014; Tapson et al., 2013), which consists of 2.5-million-neurons that bridges the gap between neural activity and biological function by exhibiting many different behaviours. The model is presented only with visual image sequences, and it draws all its responses with a physically modelled arm. Although simplified, the model captures many aspects of neuroanatomy, neurophysiology, and psychological behaviour, which are demonstrated via diverse tasks. The network implementing the “Spaun” model consists of three hierarchies (visual system, motor and the working memory), an action-selection mechanism, and five subsystems. Components of the model communicate using spiking neurons that implement neural representations that is called “semantic pointers,” using various firing patterns. The number of cells in the visual hierarchy gradually decreases from the primary visual cortex (V1) to the inferior temporal cortex (IT), meaning that the information has been compressed from a higher dimensional (image-based) space into a lower dimensional (feature) space. However, the “Spaun” has many limitations that distinguish it from developed brains. For one, “Spaun” is not as adaptive as a real brain, as the model is unable to learn completely new tasks. In addition, both attention and eye position of the model is fixed, making “Spaun” unable to control its own input. Anatomically, many areas of the brain are missing from the model. Those that are included have too few neurons and perform only a subset of functions found in their respective areas. Physiologically, the variability of spiking in the model is not always reflective of the variability observed in real brains. However, as available computational power increases,

many of these limitations can be overcome via the same methods as those used to construct “Spaun”.

More about spiking neuron models can be found in (Dayan & Abbott, 2001; Gerstner & Kistler, 2002).

### 2.2.2. Neuronal network models

Neuronal network model with circuitry that is based on the anatomy has been built for macaque primary visual cortex (McLaughlin et al., 2000) with 4 orientation hypercolumns. Also, a comparison of models of orientation and ocular dominance columns in the visual cortex was given by (Erwin, Obermayer, & Schulten, 1995). But Bednar argues in (Bednar, 2012) that the approaches researchers have used to help understand mammalian visual systems tend to have quite different assumptions, strengths, and weaknesses. Computational models of the visual cortex have typically implemented either a proposed circuit for part of the visual cortex of the adult, assuming a very specific wiring pattern based on findings from adults, or else attempted to explain the long-term development of a visual cortex region from an initially undifferentiated starting point. He adds that previous models of adult V1 have been able to account for many of the measured properties of V1 neurons, while not explaining how these properties arise or why neurons have those properties. Moreover, previous developmental models have been able to reproduce the overall organization of specific feature maps in V1, such as orientation maps, but are generally formulated at an abstract level that does not allow testing with real images or analysis of detailed neural properties relevant for visual function. Thus, Bednar shows in this review how these models could represent a single, consistent explanation for a wide body of experimental evidence, and

form a compact hypothesis for much of the development and behaviour of neurons in the visual cortex. The models proposed are the first developmental models with wiring consistent with V1, the first to have realistic behaviour with respect to visual contrast, and the first to include all the demonstrated visual feature dimensions.

### 2.2.3. Neural mass models and neural field models

Models of the cortex can establish which types of large-scale neuronal networks can perform computations and characterize their emergent properties (Deco et al., 2008). Neural mass models (Stephen Coombes & Byrne, 2016; David & Friston, 2003; Moran et al., 2007; Pinotsis, Robinson, beim Graben, & Friston, 2014; Schellenberger Costa et al., 2016) are used for studying the temporal dynamics of whole brain dynamics and may explain how the neuronal activity unfolds on the spatially continuous cortical sheet (Deco et al., 2008). They can model the coarse-grained activity of large populations of neurons and synapses and have proven especially useful in understanding brain rhythms (Stephen Coombes & Byrne, 2016). In neural mass models, the properties of a large population of spiking neurons are averaged into a single population, and it is assumed that all neurons in a population are located at the same point (Pinotsis et al., 2014).

Neural field models fall under the same category as the neural mass models and are called mean field models of neural activity; but compared to neural mass models, which characterise activity over time only, neural field models retain spatial information (Pinotsis et al., 2014). This means that neuronal activity depends on its current state as well as spatial gradients, which allow its spread horizontally across the cortical surface.

Some of the neural field models for modelling the brain at the large scales were developed by Nunez (Nunez, 1974; Nunez & Srinivasan, 2006). Nunez solved this model

analytically for a 1D loop cortex, and for two-dimensional cortex with periodic and with spheroidal boundary conditions ignoring the more complicated convoluted form of the real cortex, and the inhomogeneity of cortical connections, interpreting observed cortical wave frequencies in terms of discrete Eigen frequencies with the alpha rhythm being at the fundamental cortical Eigen frequency (Robinson et al., 1997). Wright and Liley (Wright & Liley, 1995, 1996) introduced a spatially discretised model in which the cortex is treated as 2D and divided into patches, each of which is parametrised by the mean densities of excitatory and inhibitory neurons, their mean firing rates, and their mean densities of interconnections. Nonlinear effects and axonal and dendritic delays were all included, with a Green-function formulation describing the interconnections between patches as a function of their spatial and temporal separation. This model incorporated all relevant effects mentioned above, except convolutions and nonuniformities in cortical connectivity, while allowing for the imposition of a variety of boundary conditions. Moreover, its parameters were largely physiologically measurable, a significant advantage when comparing its predictions with measurements. However, simulations based on it have been limited to very small systems, or very coarse resolution in larger systems, due to its formulation in terms of Green functions, which are very slow to evaluate, and a numerically intensive treatment of dendritic lags (Robinson et al., 1997). Robinson (Robinson et al., 1997) introduced a model of cortical electrical activity which includes nonlinearities, axonal and dendritic time lags, variable geometries and boundary conditions in 2D, and which permits analytic studies of wave properties and stability, while speeding computation to the point that whole-cortex simulations are possible with good resolution. This lead to a series of papers using the Robinson's "NeuroField" model predicting steady states, stability, waves, spectra, coherence, correlations, EEG, ERP, SSEP,

ECoG, fMRI, Seizures, Parkinson's, Arousal Dynamics (normal, abnormal, jetlag, drugs), vision, neural plasticity, connection matrices (Abeysuriya, Rennie, & Robinson, 2014; Abeysuriya, Rennie, Robinson, & Kim, 2014; Kerr, Rennie, & Robinson, 2011; Rennie, Robinson, & Wright, 2002; Roberts & Robinson, 2012; Robinson, 2014; Robinson & Kim, 2012; Robinson, Rennie, Rowe, & O'Connor, 2004a; Robinson et al., 2005; Robinson et al., 2001; Robinson, Sarkar, Pandejee, & Henderson, 2014; van Albada, Gray, Drysdale, & Robinson, 2009; van Albada, Kerr, Chiang, Rennie, & Robinson, 2010; van Albada & Robinson, 2009; Wu & Robinson, 2007; Yamaguchi, Ogawa, Nakao, Jimbo, & Kotani, 2014).

### 2.2.4. Composite neural models

One composite, network/field neural model (Kerr et al., 2013) was created for the purpose of exploring how the basal ganglia influences cortical information flow and how that influence becomes pathological in Parkinson's disease (PD). The basal ganglia plays a crucial role in the execution of movements, as demonstrated by the severe motor deficits that accompany PD. The network model consisted of 4950 spiking neurons, divided into 15 excitatory and inhibitory cell populations in the thalamus and cortex. The field model consisted of the cortex, thalamus, striatum, subthalamic nucleus, and globus pallidus. Compared to the network driven by the healthy model, the PD-driven network had lower firing rates, a shift in spectral power toward lower frequencies, and higher probability of bursting, which was consistent with empirical data on PD.

### 2.2.5. Neural simulators

Neural simulators provide tools for conveniently building, managing, and using models in a way that is numerically sound and computationally efficient. These simulators

implement computationally efficient algorithms and are widely used for large-scale modelling and complex biophysical models. They tend to be well-suited to problems that are closely linked to experimental data, especially those that involve cells with complex anatomical and biophysical properties. The main goal of the neural simulators is to minimise the development time for a neural model, and, in particular, the time spent writing code, so that scientists can spend their time on the details of their model rather than the details of its implementation.

Several successful neural simulators are used today (Brette et al., 2007), such as Neuron (Carnevale & Hines, 2006) and Genesis (Bower & Beeman, 1998) for compartmental modelling, and NEST (Gewaltig & Diesmann, 2007) and Brian (Goodman & Brette, 2009) for large scale network modelling. A review of network simulators is given by (Brette et al., 2007) and the up-to-date comparison of neural network simulators is given by (Mingus, 2014).

## 2.3. Conclusion

In this Chapter, the overview of some of the most important computational neuroscience models was presented, ranging from a single neuron models to the large scale neural simulators. The neural field models, which are the focus of this thesis, are used for modelling the brain at large scales. They provide a good framework for unifying data from different imaging modalities (EEG, fMRI, MEG, optical imaging data) and fall under the same category as the neural mass models; but compared to neural mass models, which characterise activity over time only, neural field models retain spatial information, which allow its spread horizontally across the cortical surface.

# 3. “NeuroField” Program

Prof. Peter Robinson and his Brain Dynamics group at The University of Sydney have developed a multiscale neural field brain model, called NeuroField (P. Sanz-Leon, 2017; Robinson et al., 1997). NeuroField models the interactions of spatially extended populations of neurons and can predict the spectral and time characteristics of brain electrical activity observable by EEG, MEG, fMRI, ECoG and other non-invasive imaging modalities (P. Sanz-Leon, 2017). It models brain activity by averaging firing rates, soma voltages and incoming activities over many neurons and is capable of modelling both the large numbers of neurons, as well the fine structures in the brain and its activities (P. Sanz-Leon, 2017).

In this Chapter, the algorithm of NeuroField, implemented as a C++ program that solves the neural field model of (Rennie et al., 2002; Robinson et al., 2005; Robinson et al., 1997), and the main macroscopic variables are explained.

## 3.1. NeuroField algorithm

In modelling a brain system, the neuronal populations and the connections between them should be specified first. The macroscopic variables that describe the activity of each neural population $a$ and its interaction with other populations $b$ are the mean soma potential $V_a\left(\vec{r},t\right)$, the mean firing rate $Q_a\left(\vec{r},t\right)$, and the propagating axonal spike-rate field $\varphi_{ab}\left(\vec{r},t\right)$ that arrives at population $a$ from population $b$ (P. Sanz-Leon, 2017). The main dynamic process of a generic neural field model with three populations can be seen in Figure 3.1. Geometrically, the cortical sheet is represented by a 2D grid. Each square element of this grid represents a node with a certain extent $\Delta x$. On this grid, a given position, defined with the position vector $\vec{r}$ in the 2D Cartesian coordinate system, is assumed to be the actual position in the neuronal population 1. The second population is linked to the first population via a primary topographic one-to-one map. The same value of $\vec{r}$ is assigned to such points.

The axonal spike-rate field, $\varphi_{23}\left(\vec{r},t\right)$, from the stimulation population 3, $Q_3\left(\vec{r},t\right)$ propagates to the thalamic population 2, $Q_2\left(\vec{r},t\right)$. It is weighted by the synaptic coupling strength $\upsilon_{23}$

$$P_{23} = \upsilon_{23}\varphi_{23} \tag{3.1}$$

These weighted inputs are then temporally summed via convolution with a dendritic response function and evoke postsynaptic potentials and produce the soma potential $V_2$:

$$D_{23}(t)V_{23}\left(\vec{r},t\right)=\upsilon_{23}\varphi_{23}\left(\vec{r},t-\tau_{23}\right)$$
$$D_{23}(t)=\frac{1}{\alpha_{23}\beta_{23}}\frac{d^2}{dt^2}+\left(\frac{1}{\alpha_{23}}+\frac{1}{\beta_{23}}\right)\frac{d}{dt}+1 \tag{3.2}$$

In these equations, $1/\beta_{23}$ and $1/\alpha_{23}$ are the rise and the fall time-constants, respectively, of the response at the cell body, and $\tau_{23}$ is the long-range time delay between populations (for example between thalamus and the cortex) (Robinson et al., 2005).

Action potentials are generated at the axonal hillock when the soma voltage exceeds a threshold $\theta$ and the firing rate $Q_2\left(\vec{r},t\right)$ of the population is obtained via the nonlinear sigmoidal activation function:

$$Q_2\left(\vec{r},t\right)=S\left[V_2\left(\vec{r},t\right)\right]=\frac{Q_{\max}}{1+\exp\left(-\frac{V_2\left(\vec{r},t\right)-\theta_2\left(\vec{r},t\right)}{\sigma_2'\left(\vec{r},t\right)}\right)} \tag{3.3}$$

where $Q_{\max}$ is the maximum attainable firing rate, $\theta_2$ is the mean firing threshold, and $\sigma_2=\sigma_2'\pi/\sqrt{3}$ is the standard deviation of the threshold distribution in the neural population. $V_2\left(\vec{r},t\right)$ can be calculated as a sum of all the contributions from the potentials coming from other populations $b$ at a particular location $a$. In this example $V_2\left(\vec{r},t\right)=V_{23}\left(\vec{r},t\right)$, but the general formula is:

$$V_a\left(\vec{r},t\right)=\sum_b V_{ab}\left(\vec{r},t\right) \tag{3.4}$$

Lastly the transformation of $Q_2\left(\vec{r},t\right)$ into $\varphi_{12}\left(\vec{r},t\right)$ embodies the spatiotemporal propagation of pulses generated in Population 2 to other locations (like cortical Population 1) through the axon fibres:

$$\begin{aligned}&\mathcal{D}_{12}\left(\vec{r},t\right)\varphi_{12}\left(\vec{r},t\right)=Q_2\left(\vec{r},t\right)=S\left[V_2\left(\vec{r},t\right)\right]\\&\mathcal{D}_{12}\left(\vec{r},t\right)=\frac{1}{\gamma_{12}^2}\frac{\partial^2}{\partial t^2}+\frac{2}{\gamma_{12}}\frac{\partial}{\partial t}+1-r_{12}^2\nabla^2\end{aligned}\tag{3.5}$$

where $r_{12}$ is the mean range of axons between populations 2 and 1, $\gamma_{12}=v_{12}/r_{12}$ is the temporal damping rate of pulses in axons governing the dispersion of propagating waves, the speed of propagation of the field $\varphi_{12}\left(\vec{r},t\right)$ is $v_{12}$ and $\nabla^2$ is the Laplacian operator (P. Sanz-Leon, 2017).

The axonal spike-rate $\varphi_{12}\left(\vec{r},t\right)$ will then propagate to cortical population 1.

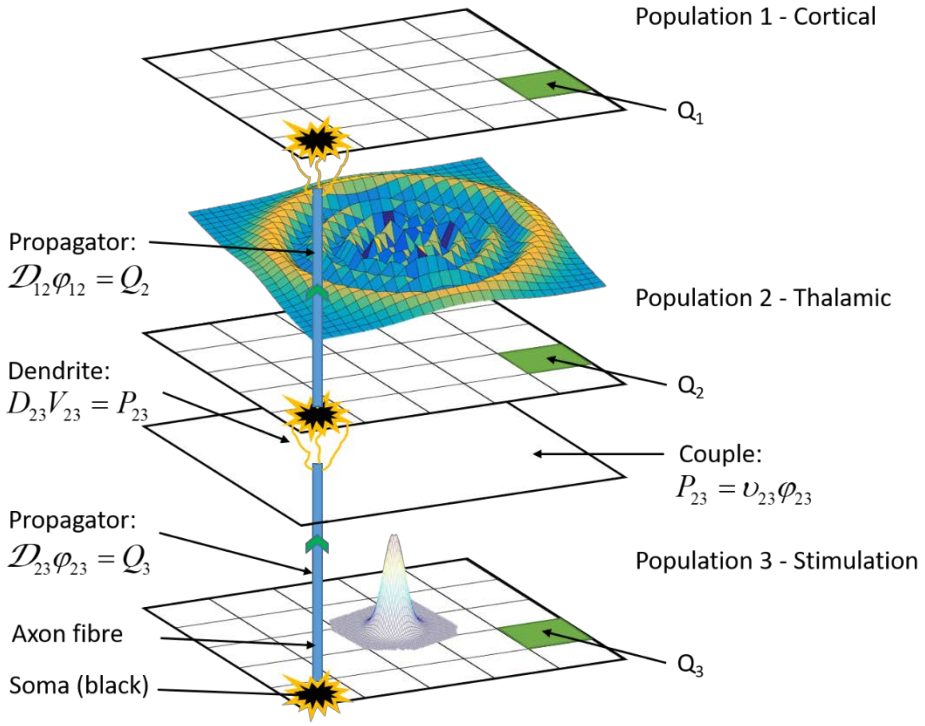


*Figure 3.1 Diagram of the dynamical processes that occur within and between neural populations in NeuroField program*

The biophysical processes described above make the main algorithm of NeuroField and each process is handled by one of the main classes in NeuroField program (P. Sanz-Leon, 2017):

$$\mathcal{D}_{ab}\varphi_{ab} = Q_b \qquad \textit{Propagator}$$

$$P_{ab} = \upsilon_{ab}\varphi_{ab} \qquad \textit{Couple}$$

$$D_{ab}V_{ab} = P_{ab} \qquad \textit{Dendrite}$$

$$Q_a = S_a\left[\sum_b V_{ab}\right] \qquad \textit{QResponse}$$

*Propagator* computes and determines the form of the axonal propagation of the presynaptic neural population Eq. (3.5). The connections between two populations are represented by an object of the class *Couple*, Eq. (3.1). The dendritic response of the postsynaptic population Eq. (3.2) is handled by *Dendrite*. Finally, each neural population is associated with *QResponse* which produces the soma response Eq. (3.3) (P. Sanz-Leon, 2017).

The more, in detailed explanation of the NeuroField algorithm can be found in (P. Sanz-Leon, 2017). Some of the models solved by NeuroField, along with many of their applications are described in more details in (Abeysuriya, Rennie, & Robinson, 2014; Abeysuriya, Rennie, Robinson, et al., 2014; Kerr et al., 2011; Rennie et al., 2002; Robinson et al., 2004a; Robinson et al., 2001; van Albada et al., 2010).

## 3.2. Conclusion

In this Chapter, the algorithm of NeuroField program, which solves the multiscale neural field brain model of (Rennie et al., 2002; Robinson et al., 2005; Robinson et al., 1997), was explained on a generic neural field model with three populations. The

macroscopic variables that describe the activity of each neural population and its interaction with other populations were defined. Finally, the main classes in NeuroField program, which are handling the biophysical processes that occur in this model, were presented.

# 4. Transmission-line matrix method

The introduction to Transmission-Line Matrix (TLM) method, with a literature review is presented in this Chapter. We also discuss the advantages of TLM to finite difference (FD) method and the possibility to use TLM to model neural activity. The last Section describes the basic algorithm for simulating the propagation of fields using TLM method.

## 4.1. TLM literature review

The Transmission-Line Matrix (TLM) method (or Transmission-Line Modelling method, as it is sometimes called) is one of the best-known examples of analogue models used to numerically solve the equations modelling a physical phenomenon. In TLM, an electrical network is used to mimic the physical problem where solution can be obtained using conventional circuit analysis techniques in either time or frequency domains. As a network model of Maxwell's equations formulated in terms of the scattering of impulses, it

possesses exceptional versatility, numerical stability, robustness and isotropic wave properties (Russer, 2000).

TLM was developed and first published in 1971 by Johns and Beurle (P. B. Johns & Beurle, 1971) as a physical approach based on Huygens' principle (Huygens, 1690). In TLM, a continuous system is replaced by a network of transmission lines and the space is discretised by a subdivision into cells. The electromagnetic field is modelled by wave pulses propagating between adjacent cells and scattered within the cells. In TLM the discretised field state is represented by a state vector summarizing the states of all TLM cells. One single computation of a pulse response produces a large amount of information. The frequency characteristics may be evaluated over the entire frequency range of interest by Fourier transform of the transient time–domain results. The versatility of the TLM method allows straightforward calculation of complicated structures, boundaries and material properties. There are no problems with convergence, stability or spurious solutions[1] in TLM and the method is limited only by the amount of memory storage required, which depends on the complexity of the TLM mesh (Sadiku, 2009). In general, the smallest feature in the structure should contain at least three nodes for good resolution (Hoefer, 1985).

TLM is mostly used in computational electromagnetics but its flexibility and the simplicity of formulation and programming also extend it to other fields of research where the wave equations need to be solved numerically. Some of the examples are:

[1] Although, accuracy decreses for high frequencies.

- 2D scattering problems in rectangular waveguides (field distribution of propagating and evanescent modes, wave impedance, scattering parameters of discontinuities) (P. B. Johns & Beurle, 1971);
- 2D eigenvalue problems (Sina Akhtarzad, 1975; P. B. Johns, 1972; Yi-Chi & Hoefer, 1980);
- 3D eigenvalue and hybrid field problems (dispersion characteristics of planar transmission lines, wave impedances, losses, Eigen frequencies, mode fields, Q factors of resonators, modelling of discontinuities) (S. Akhtarzad & Johns, 1975);
- Lumped network analysis (P. B. Johns & Brien, 1980);
- Diffusion problems (Amri, Saidane, & Pulko, 2011; Cogan et al., 2005; Desai et al., 1992);
- Acoustic propagation (Portí & Morente, 2001);
- Modelling of semiconductor lasers (Lowery, 1989);
- Induced currents in biological bodies exposed to EM fields (Deford & Gandhi, 1985);
- Ultrasound non-destructive testing of materials (Ciocan & Ida, 2003);
- Fast simulation of fluid flow dynamics (Velut & Tummescheit, 2011);
- Modelling of various mechanical processes (Cogan et al., 2005).

A field theoretical derivation of TLM was presented in (Krumpholz & Russer, 1994) with 3D TLM method with condensed symmetric node directly derived from Maxwell's equations using Method of Moments. The main difference between TLM and other numerical methods, such as, the widely used, Finite Difference (FD) (Thom, 1961), is its

discretisation approach. To use the FD method, the physical problem that should be solved must have two levels of approximation: first it should be modelled by differential or integral equation and then this model is solved by numerical method using purely mathematical discretisation approach, while the TLM has a physical approach as mentioned above. The major advantage of the TLM over the FD method is that all the required discretisation is built into the initial model, which is then solved without any further approximation avoiding many anomalous effects that can arise in FD (Cogan et al., 2005). A field theoretical comparison of the Finite-difference finite-time (FDTD)[2], and the 3D TLM methods was conducted by (Krumpholz, Huber, & Russer, 1995). They concluded that the 3D TLM exhibits some disadvantages in comparison to the FDTD from field theoretical point of view, mainly in the number of parameters needed for the TLM simulation, which some of them are nonphysical. Although the TLM is a very flexible analysis strategy similar to the FDTD in capabilities, more codes tend to be available with the FDTD method because, according to (Sadiku, 2009), the FDTD has a simpler algorithm, it can almost be two times faster in CPU time than equivalent TLM programs under identical conditions and requires less memory. However, according to Johns (P. B. Johns, 1987), the two methods complement each other rather than compete with each other. Hoefer in (Hoefer, 2012) gives a historical overview of development of TLM and FDTD in parallel. The various sources of error and the limitations of the TLM method are given, and methods for error correction or reduction, as well as improvements of numerical efficiency, are discussed in (Hoefer, 1985).

[2] Finite-difference finite-time (FDTD) method was introduced by Yee (Kane, 1966). It uses the FD method in solving the electromagnetic (EM) field problems.

An universal 3D TLM FORTRAN computer program was written by (Sina Akhtarzad, 1975) and the ease of application, versatility and accuracy of the TLM method is demonstrated by analysing a wide variety of microwave resonators. The surface mode phenomenon of microstrip is also investigated in this reference.

One popular commercial software package exists for solving TLM. It is called The TLM solver of CST MICROWAVE STUDIO® (CST MWS), and it is based on the 3D time-domain TLM method (Cst.com, 2015).

## 4.2. Analogy of TLM method and neurological activity

The TLM method can be applied to problems in other areas, such as thermodynamics, optics, and acoustic wave motion. Aside from the area of physics, however, there is a branch in the biological sciences to which the TLM method appears to have a natural affinity. The possible application is in neuroscience, specifically, in modelling the brain functions as speculated by Nunez in (Nunez & Srinivasan, 2006), where he talks about EEG dynamic behaviour that is similar to the transmission line theory. Weiner in (M. Weiner, 2010) speculates how the TLM method may be used as a framework to describe neurological activity of the brain, since it relies on a vast array of nerve fibres and synapses, analogous to the transmission lines and nodes of the TLM matrix.

As Weiner says (Maurice Weiner, 2010), in the area of nerve cells, the nerve fibres and synapses appear to play a role similar to transmission lines and nodal scatterers in the TLM model. Nerve impulses are conveyed along the fibres. The synapses exist at the juncture of two or more fibres and they serve to control the flow of the impulses from one fibre to another. The nature of the impulse propagation along the fibres is discussed in (Nunez & Srinivasan, 2006; Ray & Roy, 2010). Needless to say, the nerve fibres do not

form neat geometrical shapes, such as cubes or hexagons, as we assume in TLM analysis. The actual fibres appear as a tangled array with irregular shapes and with varying fibre lengths. Despite these differences, the same type of analysis may be applied to nerve impulses, taking into account the random nature of the fibre shape and length. In some ways the irregularity of the fibres is an advantage since it removes the anisotropy associated with the symmetry elements, where the energy is constrained to flow in only certain directions. With an irregular cell matrix, we are not bound to a preferred direction (Maurice Weiner, 2010). TLM has been used to model the propagation of action potentials along the axon in myelinated nerve in (Villapecellin-Cid, Rao, & Reina-Tosina, 2003; Villapecellin-Cid, Roa, & Reina-Tosina, 2001, 2002).

In case of using TLM as a brain model, one must obtain predictions of TLM model and compare these with experimental observations. To our knowledge, the use of TLM method to model neural fields has never been done.

## 4.3. TLM algorithm

The TLM algorithm consists of the propagation of the wave amplitudes from the mesh nodes to the neighbouring nodes and the scattering of the wave amplitudes in the mesh nodes. The propagation and the scattering of the wave amplitudes may be expressed by operator equations.

The two-dimensional TLM method is suitable for the analysis of electromagnetic fields with the electric field components oriented normal and the magnetic field parallel to a certain plane of reference (Transverse Electric (TE) case), or - vice versa - the magnetic field components oriented normal and the electric field parallel to the plane of reference (Transverse Magnetic (TM) case). Figure 4.1.A shows a TE arrangement with two parallel

conducting plates. This arrangement may be modelled by a two-dimensional mesh of lines as depicted in Figure 4.1.B.

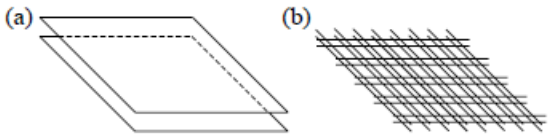


*Figure 4.1 – TE arrangement with two parallel conducting plates. A) Parallel plates; B) 2D mesh. Taken from (Russer, 2000)*

The two-dimensional mesh of lines may be modelled by interconnected four ports shown in Figure 4.2 (Russer, 2000). The lossless 2D TLM cell with lumped elements is shown in Figure 4.3.

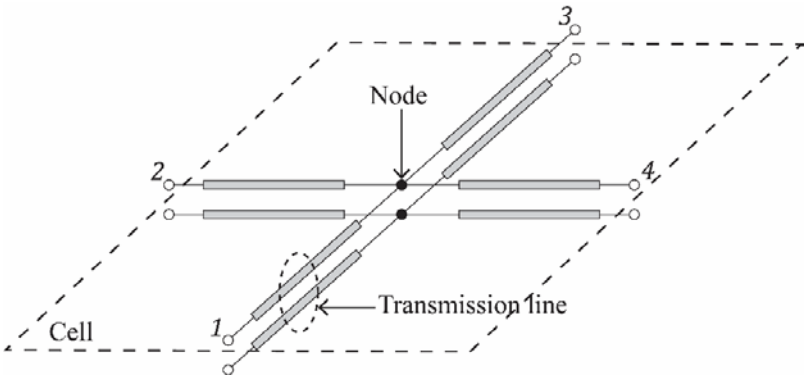


*Figure 4.2 – 2D TLM shunt cell (Russer, 2000)*

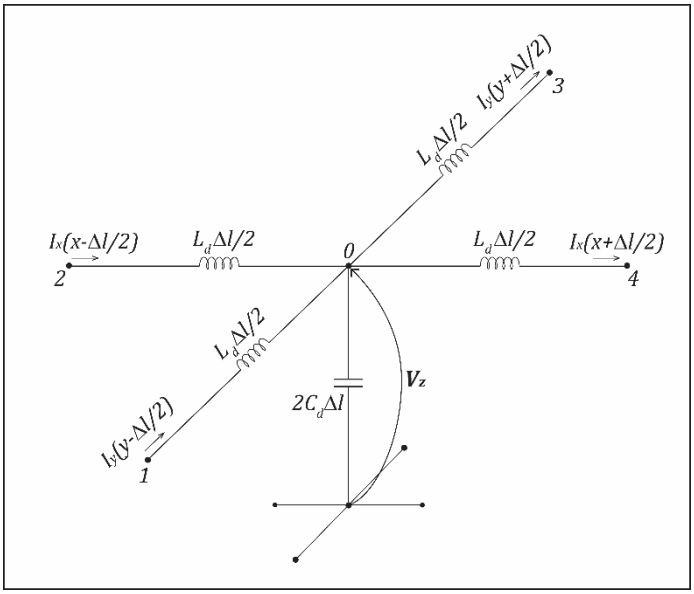


*Figure 4.3 – Lossless TLM cell represented by lumped elements (Sadiku, 2009)*

If a voltage pulse of amplitude $1V$ is incident on the central node (Figure 4.4), this pulse will be partially reflected and transmitted according to the transmission-line theory. If we assume that each line has a characteristic impedance $Z_0 = \sqrt{\frac{L_d}{2C_d}}$ (Figure 4.5), then the incident pulse sees effectively three transmission lines in parallel, with a combined impedance of $Z_0/3$. The reflection coefficient and the transmission coefficient are given by:

$$R = \frac{Z_0/3 - Z_0}{Z_0/3 + Z_0} = -\frac{1}{2}$$
$$T = \frac{2(Z_0/3)}{Z_0/3 + Z_0} = +\frac{1}{2} \qquad (4.1)$$

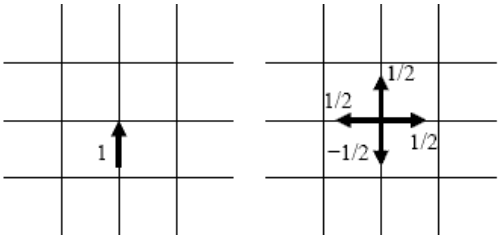


*Figure 4.4 – An incident voltage pulse and scattering at the node*

If we assume that in general formulation there are pulses incident from all four directions, we can calculate the nodal voltage ${}_kV_n(x,y)$:

$${}_kV_n(x,y) = \frac{\left({}_kV_1^i + {}_kV_2^i + {}_kV_3^i + {}_kV_4^i\right)}{2} \qquad (4.2)$$

where $k = t/\Delta t$ is the iteration number. Superscript $i$ in ${}_kV_j^i$ denotes the incident pulse coming from a port denoted with the subscript $j$ ( Figure 4.5).

The pulse, which is scattered back to port 2, for example, is:

$${}_{k+1}V_2^r = {}_kV_n - {}_kV_2^i \qquad (4.3)$$

which is the same as the sum of reflected and transmitted pulses from all other arms. The entire scattering process of a lossless TLM node can be given in matrix form by:

$$\begin{pmatrix} V_1^r \\ V_2^r \\ V_3^r \\ V_4^r \end{pmatrix}_{k+1} = S \cdot \begin{pmatrix} V_1^i \\ V_2^i \\ V_3^i \\ V_4^i \end{pmatrix}_{k} \tag{4.4}$$

where $S$ is the scattering matrix:

$$S = \frac{1}{2}\begin{pmatrix} -1 & 1 & 1 & 1 \\ 1 & -1 & 1 & 1 \\ 1 & 1 & -1 & 1 \\ 1 & 1 & 1 & -1 \end{pmatrix} \tag{4.5}$$

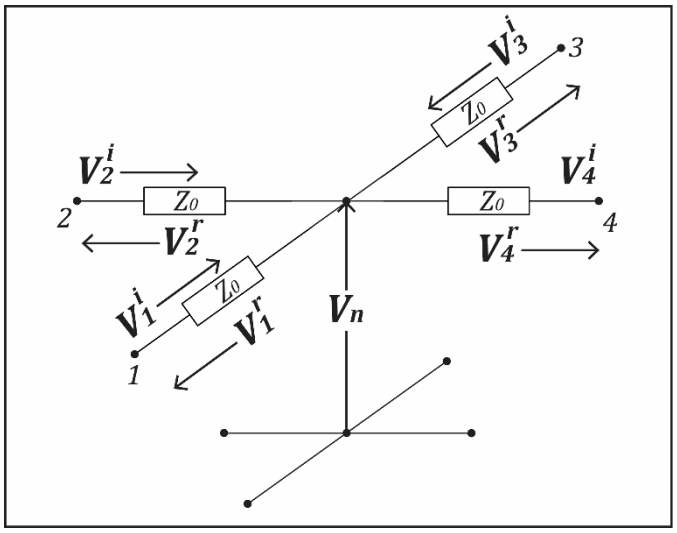


*Figure 4.5 – Lumped equivalent circuit for a 2D lossless TLM cell (Cogan et al., 2005)*

Furthermore, each impulse travels the discretisation distance $\Delta x$ during the discretisation time $\Delta t$ automatically becoming an incident impulse on the neighbouring node (Cogan et al., 2005). The connections to other nodes as seen at node $(x, y)$ can be expressed in terms of space and time-step, $k+1$ as:

$$
\begin{aligned}
{}_{k+1}V_1^i(x,y) &= {}_{k+1}V_3^r(x,y-1) \\
{}_{k+1}V_2^i(x,y) &= {}_{k+1}V_4^r(x-1,y) \\
{}_{k+1}V_3^i(x,y) &= {}_{k+1}V_1^r(x,y+1) \\
{}_{k+1}V_4^i(x,y) &= {}_{k+1}V_2^r(x+1,y)
\end{aligned} \tag{4.6}
$$

The repeated iteration of processes of scattering (Eq. (4.4)), connection (Eq. (4.6)) and summation (Eq. (4.2)) for every time step forms the basic algorithm of the TLM method for a 2D TLM network (Cogan et al., 2005).

The impulse response of the network is then found by initially fixing the magnitudes, directions and positions of all impulses at $t = 0$ and then calculating the state of the network at successive time intervals. Three consecutive scatterings are shown in Figure 4.6, visualizing the spreading of the injected voltage across the 2D network.

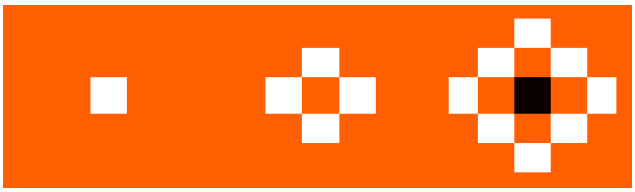

*Figure 4.6 – Heatmap of three consecutive scatterings in 2D TLM network created in MATLAB. Left image - the initial impulse; middle - first iteration; right - second iteration. White – positive values, Orange – zero, Black – negative values.*

## 4.4. Conclusion

The introduction to TLM method, with a literature review was presented in this Chapter. The advantages of TLM to FD method and the possibility to use TLM to model neural activity were discussed in Section 4.2. In Section 4.3 the lossless 2D TLM cell with lumped elements was introduced. Furthermore, the equations for basic algorithm of the TLM method, which consists of three main processes (incident, scattering, connection) for every time step, for a 2D TLM network, were presented.

# 5. Numerical solution of hyperbolic equations

In Chapter 3 was mentioned that NeuroField is coded in the C++ programming language. For testing the feasibility to numerically solve the governing inhomogeneous damped wave PDEs, used in NeuroField program, using TLM techniques, main algorithm of NeuroField, with its FD method of numerical approximations of governing wave equations, had to be reprogramed into MATLAB (Appendix B). Furthermore, the "old" version of NeuroField program (Robinson et al., 2005) used the so called nine-point stencil in FD method which was not compatible with TLM method because of the extra diagonal terms.

In this Chapter, the possibility of using the five-point stencil to numerically solve the undamped and damped wave PDEs, instead of the nine-point stencil is explored.

## 5.1. Analytical solution of the 1D undamped wave equation

The equation used in NeuroField to represent the axonal propagation of activity through the cortex is inhomogeneous (or forced) damped wave equation (Robinson et al., 1997), but in order to find the numerical solution to that equation, first the analytical solution to 1D undamped wave equation PDE is presented.

The 1D undamped wave equation for a lossless plucked string (Cogan et al., 2005):

$$\frac{\partial^2 u}{\partial t^2} = \beta^2 \frac{\partial^2 u}{\partial x^2},\ \beta^2 = \frac{T}{\rho} \tag{5.1}$$

where $u(x,t)$ denotes the vertical displacement of the string at position $x$ at time $t > 0$, $T$ denotes the tension of the string in $kg \cdot m/s^2$ and $\rho$ is the mass per unit length of the segment of the string in $kg/m$; thus the constant $\beta^2$ has the units $m^2/s^2$, which means that $\beta$ can be thought of as a velocity with which a small transverse disturbance moves along the string.

If Eq. (5.1) is rearranged as $\frac{\partial^2 u}{\partial t^2} - \beta^2 \frac{\partial^2 u}{\partial x^2} = 0$, it can be seen that it is a hyperbolic equation, since $A = 1, C = -\frac{T}{\rho}, B = 0$ and therefore $B^2 - 4AC = 4\frac{T}{\rho} > 0$.

The initial conditions for this problem are in the form of an initial position function $u(x,0) = f(x)$ and an initial velocity function $\frac{\partial u}{\partial t}(x,0) = g(x)$.

Boundary conditions: we will assume that the two ends of the string are fixed for every $t$; that is, $u(0,t) = u(L,t) = 0$ for all $t > 0$.

The analytical solution for this wave equation can be found using the method called Separation of Variables (Olsen-Kettle, 2011).

If we let $u(x,t) = X(x)T(t)$ and substitute that into the Eq. (5.1) we get:

$$XT'' = \beta^2 X''T \tag{5.2}$$

where ' denotes differentiation in respect to time for $T$ and differentiation in respect to displacement for $X$. Dividing Eq. (5.2) by $\beta^2 XT$ yields:

$$\frac{XT''}{\beta^2 XT} = \frac{\beta^2 X''T}{\beta^2 XT} \Rightarrow \frac{T''}{\beta^2 T} = \frac{X''}{X} = -\lambda \tag{5.3}$$

This results in the two ordinary differential equations:

$$\begin{aligned} X'' + \lambda X &= 0 \\ T'' + \lambda\beta^2 T &= 0 \end{aligned} \tag{5.4}$$

Once those two ordinary differential equations are solved using boundary conditions, we can get the general solution of the 1D undamped wave equation in the form:

$$u(x,t) = \sum_{n=1}^{\infty} C_n X_n(x) T_n(t) = \sum_{n=1}^{\infty} \sin\left(\frac{n\pi x}{L}\right)\left[A_n \cos\left(\frac{n\pi\beta}{L}t\right) + B_n \sin\left(\frac{n\pi\beta}{L}t\right)\right] \tag{5.5}$$

If we want to find the complete solution of the 1D wave equation with zero boundary conditions, then using first initial condition we can determine the constants $A_n$. Once we find them, we can use the second initial condition to find $B_n$.

## 5.2. Analytical solution of the 1D damped wave equation

Consider the equation:

$$\frac{\partial^2 u}{\partial t^2} + 2c\frac{\partial u}{\partial t} = \beta^2 \frac{\partial^2 u}{\partial x^2} \tag{5.6}$$

where $c$ is a small positive constant. The term $2c\frac{\partial u}{\partial t}$ represents a damping force proportional to the velocity $\frac{\partial u}{\partial t}$.

Using the separation of variables method to solve Eq. (5.6) (Arfken, Weber, & Harris, 2013) we have:

$$\frac{XT'' + 2cXT''}{\beta^2 XT} = \frac{\beta^2 X''T}{\beta^2 XT} \Rightarrow \frac{T'' + 2cT''}{\beta^2 T} = \frac{X''}{X} = -\lambda \tag{5.7}$$

Again, this can be separated into two ordinary differential equations:

$$\begin{aligned} X'' + \lambda X &= 0 \\ T'' + 2cT' + \lambda\beta^2 T &= 0 \end{aligned} \tag{5.8}$$

In order to solve Eq. (5.8) we will assume that the length of the string is $L = \pi$, the constant $\beta^2 = 1$, and $c < 1$. The solution of the 1D damped wave equation is then:

$$u(x,t) = \sum_{n=1}^{\infty} \sin(nx) e^{-ct} \left[ A_n \cos\left(\sqrt{n^2 - c^2}\, t\right) + B_n \sin\left(\sqrt{n^2 - c^2}\, t\right) \right] \tag{5.9}$$

## 5.3. Numerical solution of plucked string equation – 1D undamped wave equation

In Finite Difference (FD) approximation, the derivatives in the PDEs are approximate by linear combination of function values at the grid points.

For 1D case, we can replace the temporal $\frac{\partial^2 u}{\partial t^2}$ and spatial $\frac{\partial^2 u}{\partial x^2}$ derivatives with the finite central difference approximation on a discretised domain (Figure 5.1):

$$\begin{aligned} &\Delta t = \frac{T}{m},\ \Delta x = \frac{L}{n+1} \\ &t_k = k \cdot \Delta t,\ 0 \le k \le m \\ &x_j = j \cdot \Delta x,\ 0 \le j \le n+1 \end{aligned} \tag{5.10}$$

where $\Delta t$ is the time step, $T$ is the total time of the simulation, $m$ is the number of time steps, $\Delta x$ is the mesh size, $L$ is the length of the string and $n+1$ is the number of segments in which the string is divided into. $t_k$ is the time at time step $k$, and $x_j$ a grid point at position $j$.

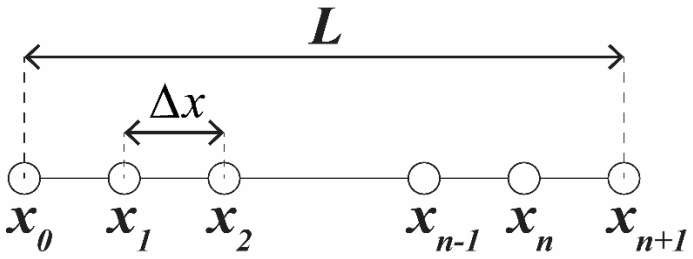


*Figure 5.1 – Discretisation of string, length L, for numerical solution of 1D wave equation. $x_j$ are grid points and $\Delta x$ is the mesh size.*

The finite central difference approximations (Olsen-Kettle, 2011) are:

$$\begin{aligned} \frac{\partial^2 u}{\partial t^2} &= \frac{u_j^{k+1} - 2u_j^k + u_j^{k-1}}{\Delta t^2} \\ \frac{\partial^2 u}{\partial x^2} &= \frac{u_{j+1}^k - 2u_j^k + u_{j-1}^k}{\Delta x^2} \end{aligned} \tag{5.11}$$

where $u_j^k$ denotes the value of $u$ at time point $k$ and grid point $j$.

The 1D undamped wave equation is then:

$$\frac{u_j^{k+1} - 2u_j^k + u_j^{k-1}}{\Delta t^2} = \beta^2 \cdot \frac{u_{j+1}^k - 2u_j^k + u_{j-1}^k}{\Delta x^2} \tag{5.12}$$

The equation to find $u_j$ for the next time step $k+1$ is given by:

$$u_j^{k+1} = -u_j^{k-1} + 2(1-s)u_j^k + s\left(u_{j+1}^k + u_{j-1}^k\right) \tag{5.13}$$

where $s = \dfrac{\beta^2 \Delta t^2}{\Delta x^2}$ .

The solution for the first step, $\vec{u}^1$ can be found in a matrix form from the Eq. (5.13). If we let the number of segments, in which the string from Figure 5.1 is divided into, to be four, $n+1=4$ , there will only be three interior points and $\vec{u}^1$ can be expressed in a matrix form as:

$$\vec{u}^1 = \begin{bmatrix} u_1^1 \\ u_2^1 \\ u_3^1 \end{bmatrix} = \underbrace{\begin{bmatrix} 2(1-s) & s & 0 \\ s & 2(1-s) & s \\ 0 & s & 2(1-s) \end{bmatrix}}_{A} \cdot \begin{bmatrix} u_1^0 \\ u_2^0 \\ u_3^0 \end{bmatrix} + \underbrace{s \cdot \begin{bmatrix} u_0^0 \\ 0 \\ u_4^0 \end{bmatrix}}_{\vec{b}} - \begin{bmatrix} u_1^{-1} \\ u_2^{-1} \\ u_3^{-1} \end{bmatrix} \tag{5.14}$$

where $A$ is the block tridiagonal coefficient matrix and $\vec{b}$ is the vector of boundary conditions multiplied by the coefficient $s$ . Using the boundary conditions $u(0,t) = u_0^k$ and $u(L,t) = u_4^k$ and the initial conditions $u_j^0 = f_j$ and $u_j^{-1} = u_j^1 - 2\Delta t \cdot g(x_j)$ we can find the solution for $\vec{u}^1$ for this case as:

$$\vec{u}^1 = \begin{bmatrix} u_1^1 \\ u_2^1 \\ u_3^1 \end{bmatrix} = \frac{1}{2}\begin{bmatrix} 2(1-s) & s & 0 \\ s & 2(1-s) & s \\ 0 & s & 2(1-s) \end{bmatrix} \cdot \begin{bmatrix} f_1 \\ f_2 \\ f_3 \end{bmatrix} + \frac{1}{2} \cdot s \cdot \begin{bmatrix} u_0^0 \\ 0 \\ u_4^0 \end{bmatrix} + \underbrace{\Delta t \cdot \begin{bmatrix} g_1 \\ g_2 \\ g_3 \end{bmatrix}}_{\vec{d}}$$

Or:

$$\vec{u}^1 = \frac{1}{2} A \cdot \vec{u}^0 + \frac{1}{2}\vec{b} + \vec{d} \tag{5.15}$$

where $\vec{d}$ is the vector of the initial conditions.

The general 2nd order iterative scheme for all other steps is then:

$$\vec{u}^{k+1} = A \cdot \vec{u}^k + \vec{b} - \vec{u}^{k-1} \tag{5.16}$$

Finally, we assumed that that the two ends of the string are fixed, thus the boundary conditions are:

$$u(0,t) = u_0^k = 0$$
$$u(L,t) = u_4^k = 0$$

which will make all the values in the vector $\vec{b}$ be zero and the equation for the 1st time step and the iterative formula will be reduced to:

$$\vec{u}^1 = \frac{1}{2} A \cdot \vec{u}^0 + \vec{d}$$
$$\vec{u}^{k+1} = A \cdot \vec{u}^k - \vec{u}^{k-1} \tag{5.17}$$

The derivation presented here solves the wave equations by using explicit methods, which means that the next value of $u$ will be computed from the known past values and, equivalently, all the future time terms appear on one side of the equation (Olsen-Kettle, 2011).

## 5.4. Numerical solution of the 1D damped wave equation

The numerical solution of the 1D damped wave equation is very similar to the solution of the undamped equation. We start again with replacing the temporal and spatial

derivatives with the finite central difference approximation (Olsen-Kettle, 2011) on the same discretised domain (Figure 5.1), which now gives us the equation:

$$\frac{u_j^{k+1} - 2u_j^k + u_j^{k-1}}{\Delta t^2} + 2c \cdot \frac{u_j^{k+1} - u_j^{k-1}}{2\Delta t} = \beta^2 \cdot \frac{u_{j+1}^k - 2u_j^k + u_{j-1}^k}{\Delta x^2} \tag{5.18}$$

The equation for $u_j$ for the future time step $k+1$ is now:

$$u_j^{k+1} = -\frac{1 - c\Delta t}{1 + c\Delta t} u_j^{k-1} + \frac{1}{1 + c\Delta t} 2(1 - s) u_j^k + \frac{1}{1 + c\Delta t} s\left(u_{j+1}^k + u_{j-1}^k\right) \tag{5.19}$$

From the initial and boundary conditions we can calculate the first iteration as:

$$\vec{u}^1 = \frac{1 + c\Delta t}{2} A \cdot \vec{u}^0 + \frac{1 + c\Delta t}{2} \vec{b} + \vec{d} \tag{5.20}$$

And the general iterative formula is:

$$\vec{u}^{k+1} = A \cdot \vec{u}^k + \vec{b} - e \cdot \vec{u}^{k-1} \tag{5.21}$$

where $e = \frac{1 - c\Delta t}{1 + c\Delta t}$ .

Finally, using the boundary conditions for the string with fixed edges all the values in the vector $\vec{b}$ will be zero and the equation for the 1st time step and the iterative formula will be reduced to:

$$\begin{aligned} \vec{u}^1 &= \frac{1 + c\Delta t}{2} A \cdot \vec{u}^0 + \vec{d} \\ \vec{u}^{k+1} &= A \cdot \vec{u}^k - e \cdot \vec{u}^{k-1} \end{aligned} \tag{5.22}$$

## 5.5. Analytical solution of the 2D undamped wave equation

The 2D wave equation can be visualised as a vibration of a thin elastic membrane stretched tightly over a rectangular frame with the edges firmly fixed at its walls (Figure 5.2):

$$\frac{\partial^2 u}{\partial t^2} = \beta^2 \left( \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} \right), \; 0 \le x \le a, \; 0 \le y \le b, \; 0 \le t \le T \tag{5.23}$$

where $u(x, y, t)$ denotes the displacement of the membrane at position $(x, y)$ at time $t > 0$.

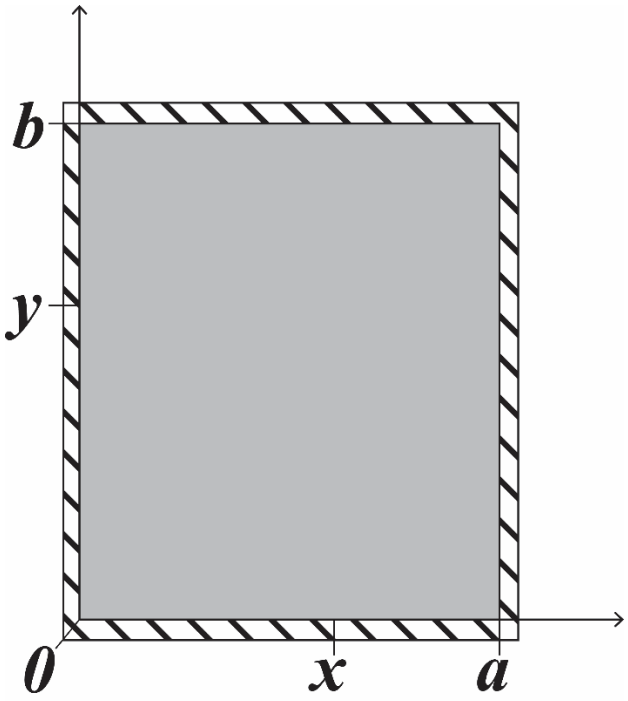


*Figure 5.2 – A thin elastic membrane stretched tightly over a rectangular frame dimensions a × b, with the edges fixed to a rigid frame.*

The boundary conditions for the membrane with fixed edges can be expressed as:

$$\begin{aligned} u(0, y, t) &= u(a, y, t) = 0 \\ u(x, 0, t) &= u(x, b, t) = 0 \end{aligned} \tag{5.24}$$

The initial deformation of the membrane and how is it set to motion can be described with the initial conditions:

$$\begin{aligned} u(x,y,0) &= f(x,y) \\ \frac{\partial u}{\partial t}(x,y,0) &= g(x,y) \end{aligned} \tag{5.25}$$

The analytical solution for the 2D wave equation Eq. (5.23) can be found using the Separation of Variables method in the similar way as for the 1D case described in the Section 5.1.

If we let $u(x,y,t) = X(x)Y(y)T(t)$ and substitute that into the Eq. (5.23) we get:

$$XYT'' = \beta^2 \left( X''YT + XY''T \right) \tag{5.26}$$

Dividing Eq. (5.26) by $XYT$ yields:

$$\frac{XYT''}{XYT} = \frac{\beta^2 \left( X''YT + XY''T \right)}{XYT} \Rightarrow \frac{T''}{T} = \beta^2 \frac{X''}{X} + \beta^2 \frac{Y''}{Y} = -\omega^2 \tag{5.27}$$

which can be separated into three differential equations:

$$\begin{aligned} T'' + \omega^2 T &= 0 \\ X'' + k_x^2 X &= 0 \\ Y'' + k_y^2 Y &= 0 \end{aligned} \tag{5.28}$$

Once these differential equations are solved using boundary conditions, we can get the general solution of the 2D wave equation in the form:

$$u(x,y,t) = \sum_{n=1}^{\infty} \sum_{m=1}^{\infty} \left[ A_{n,m} \cos\left( \omega_{n,m} \cdot t \right) + B_{n,m} \sin\left( \omega_{n,m} \cdot t \right) \right] \sin\left( k_x \cdot x \right) \cdot \sin\left( k_y \cdot y \right) \tag{5.29}$$

where $n=1,2,3\ldots$, $m=1,2,3\ldots$, $k_x=\frac{n\pi}{a}$, $k_y=\frac{m\pi}{b}$, $\omega_{n,m}=\beta\sqrt{k_x^2+k_y^2}$ and $A_{n,m}$ and $B_{n,m}$ are the constants that can be determined from the initial conditions as:

$$\begin{aligned} A_{n,m} &= \frac{4}{a\cdot b}\int_0^b\left[\int_0^a f(x,y)\cdot\sin(k_x\cdot x)\,dx\right]\sin(k_y\cdot y)dy \\ B_{n,m} &= \frac{4}{\omega_{n,m}\cdot a\cdot b}\int_0^b\left[\int_0^a g(x,y)\cdot\sin(k_x\cdot x)\,dx\right]\sin(k_y\cdot y)dy \end{aligned} \tag{5.30}$$

## 5.6. Analytical solution of the 2D damped wave equation

The 2D damped wave equation is:

$$\frac{\partial^2 u}{\partial t^2}+2c\frac{\partial u}{\partial t}=\beta^2\left(\frac{\partial^2 u}{\partial x^2}+\frac{\partial^2 u}{\partial y^2}\right) \tag{5.31}$$

where the term $2c\frac{\partial u}{\partial t}$ represents a damping force proportional to the velocity $\frac{\partial u}{\partial t}$.

Using the separation of variables method to solve Eq. (5.31) we have:

$$\frac{XYT''+2cXYT''}{XYT}=\frac{\beta^2\left(X''YT+XY''T\right)}{XYT}\Rightarrow\frac{T''+2cT''}{T}=\beta^2\frac{X''}{X}+\beta^2\frac{Y''}{Y}=-\omega^2 \tag{5.32}$$

Again, this can be separated into three ordinary differential equations:

$$\begin{aligned} T''+2cT'+\omega^2T&=0 \\ X''+k_x^2X&=0 \\ Y''+k_y^2Y&=0 \end{aligned} \tag{5.33}$$

The last two differential equations in Eq. are trivial to solve using boundary conditions, but the first one can have different solutions depending of the value of $\omega$. If we assume that $\omega^2>c^2$, then the solution of the 2D damped wave equation is:

$$u(x,y,t)=\sum_{n=1}^{\infty}\sum_{m=1}^{\infty}e^{-ct}\cdot\left[A_{n,m}\cos\left(\sqrt{\omega_{n,m}^2-c^2}\cdot t\right)+B_{n,m}\sin\left(\sqrt{\omega_{n,m}^2-c^2}\cdot t\right)\right]\cdot\sin\left(k_x\cdot x\right)\cdot\sin\left(k_y\cdot y\right) \tag{5.34}$$

where $A_{n,m}$ and $B_{n,m}$ can be determined from Eq. (5.30).

## 5.7. Numerical solution of 2D undamped wave equation

The 2D undamped wave equation Eq. (5.23) can be numerically solved using FD approximation on a discretised domain (Figure 5.3):

$$\begin{aligned}&\Delta t=\frac{T}{m},\ \Delta x=\frac{a}{n+1},\ \Delta y=\frac{b}{p+1}\\&t_k=k\cdot\Delta t,\ 0\le k\le m\\&x_i=i\cdot\Delta x,\ 0\le i\le n+1\\&y_j=j\cdot\Delta y,\ 0\le j\le p+1\end{aligned} \tag{5.35}$$

Let's assume that our membrane is divided into 16 cells, or that $n=3,\ p=3$ (Figure 5.3).

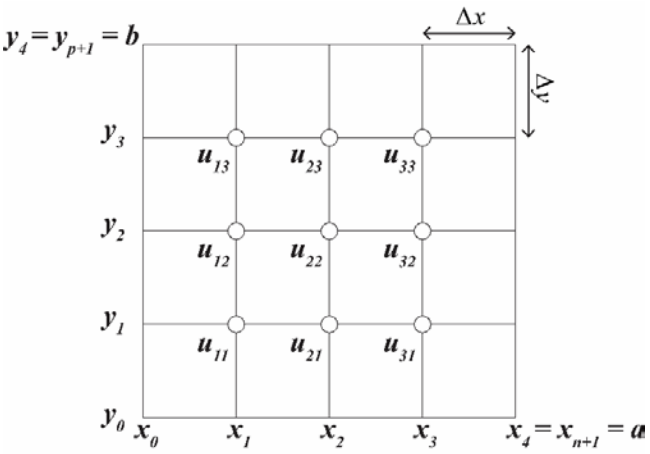


*Figure 5.3 – Discretisation of a thin elastic membrane stretched tightly over a rectangular frame dimensions $a \times b$, with the edges fixed to the frame, for FD solution of the 2D wave equation. $u_{i,j}$ denotes the displacement of the membrane at $(x_i, y_j)$ position. $\Delta x$ and $\Delta y$ are the mesh sizes in $x$ and $y$ directions, respectively.*

The boundary conditions are:

$$u(0,y,t) = u_{0,j}^{k}$$
$$u(a,y,t) = u_{4,j}^{k}$$
$$u(x,0,t) = u_{i,0}^{k}$$
$$u(x,b,t) = u_{i,4}^{k}$$

The initial conditions can be described as:

$$u(x,y,0) = f(x,y) = f_{i,j}$$
$$\frac{\partial u}{\partial t}(x,y,0) = g(x,y) = g_{i,j}$$

We can use the finite central difference approximation again to approximate the temporal and spatial derivatives. The approximate 2D wave equation will then be:

$$\frac{u_{i,j}^{k+1} - 2u_{i,j}^{k} + u_{i,j}^{k-1}}{\Delta t^2} = \beta^2 \cdot \frac{u_{i+1,j}^{k} - 2u_{i,j}^{k} + u_{i-1,j}^{k}}{\Delta x^2} + \beta^2 \cdot \frac{u_{i,j+1}^{k} - 2u_{i,j}^{k} + u_{i,j-1}^{k}}{\Delta y^2} \qquad (5.36)$$

where $u_{i,j}^{k}$ denotes the value of $u$ at time point $k$ and grid point $x_i, y_j$.

If we introduce the substitutions $s_x = \frac{\beta^2 \Delta t^2}{\Delta x^2}$ and $s_y = \frac{\beta^2 \Delta t^2}{\Delta y^2}$, we get the the equation for $u_{i,j}$ for the future time step $k+1$:

$$u_{i,j}^{k+1} = -u_{i,j}^{k-1} + 2\left(1 - s_x - s_y\right)u_{i,j}^{k} + s_x\left(u_{i+1,j}^{k} + u_{i-1,j}^{k}\right) + s_y\left(u_{i,j+1}^{k} + u_{i,j-1}^{k}\right) \qquad (5.37)$$

From the initial and boundary conditions we can calculate the first iteration as:

$$u_{i,j}^{1} = \left(1 - s_x - s_y\right)u_{i,j}^{0} + \frac{s_x}{2}\left(u_{i+1,j}^{0} + u_{i-1,j}^{0}\right) + \frac{s_y}{2}\left(u_{i,j+1}^{0} + u_{i,j-1}^{0}\right) + \Delta t g_{i,j} \qquad (5.38)$$

Both the iterative formula and the equation for the 1st time step can be calculated using the similar matrix method as for the 1D case.

For $\vec{u}^1$:

$$\vec{u}^1 = \frac{1}{2} A \cdot \vec{u}^0 + \frac{1}{2}\vec{b} + \vec{d} \tag{5.39}$$

And for all other steps, the general iterative scheme is then:

$$\vec{u}^{k+1} = A \cdot \vec{u}^k + \vec{b} - \vec{u}^{k-1} \tag{5.40}$$

where $A$ and $\vec{b}$ are the matrix and vector, respectively, shown in Figure 5.4, with $\lambda = 2\left(1 - s_x - s_y\right)$.

| | $u_{11}$ | $u_{12}$ | $u_{13}$ | $u_{21}$ | $u_{22}$ | $u_{23}$ | $u_{31}$ | $u_{32}$ | $u_{33}$ |
|---|---|---|---|---|---|---|---|---|---|
| $u_{11}$ | λ | $S_y$ | 0 | $S_x$ | 0 | 0 | 0 | 0 | 0 |
| $u_{12}$ | $S_y$ | λ | $S_y$ | 0 | $S_x$ | 0 | 0 | 0 | 0 |
| $u_{13}$ | 0 | $S_y$ | λ | 0 | 0 | $S_x$ | 0 | 0 | 0 |
| $u_{21}$ | $S_x$ | 0 | 0 | λ | $S_y$ | 0 | $S_x$ | 0 | 0 |
| $u_{22}$ | 0 | $S_x$ | 0 | $S_y$ | λ | $S_y$ | 0 | $S_x$ | 0 |
| $u_{23}$ | 0 | 0 | $S_x$ | 0 | $S_y$ | λ | 0 | 0 | $S_x$ |
| $u_{31}$ | 0 | 0 | 0 | $S_x$ | 0 | 0 | λ | $S_y$ | 0 |
| $u_{32}$ | 0 | 0 | 0 | 0 | $S_x$ | 0 | $S_y$ | λ | $S_y$ |
| $u_{33}$ | 0 | 0 | 0 | 0 | 0 | $S_x$ | 0 | $S_y$ | λ |

$A$

| | |
|---|---|
| $u_{11}$ | $S_x \cdot u_{01} + S_y \cdot u_{10}$ |
| $u_{12}$ | $S_x \cdot u_{02}$ |
| $u_{13}$ | $S_x \cdot u_{03} + S_y \cdot u_{14}$ |
| $u_{21}$ | $S_y \cdot u_{20}$ |
| $u_{22}$ | 0 |
| $u_{23}$ | $S_y \cdot u_{24}$ |
| $u_{31}$ | $S_x \cdot u_{41} + S_y \cdot u_{30}$ |
| $u_{32}$ | $S_x \cdot u_{42}$ |
| $u_{33}$ | $S_x \cdot u_{43} + S_y \cdot u_{34}$ |

$\vec{b}$

*Figure 5.4 – Matrix A and vector $\vec{b}$ used for matrix method for numerical solution of 2D undamped wave equation*

Finally, the boundary conditions for the membrane with fixed edges can be expressed as:

$$u\left(0, y, t\right) = u^k_{0,j} = 0$$
$$u\left(a, y, t\right) = u^k_{4,j} = 0$$
$$u\left(x, 0, t\right) = u^k_{i,0} = 0$$
$$u\left(x, b, t\right) = u^k_{i,4} = 0$$

Considering these boundary conditions, all the values in the vector $\vec{b}$ will be zero and the equation for the 1st time step and the iterative formula will be reduced to:

$$\vec{u}^1 = \frac{1}{2} A \cdot \vec{u}^0 + \vec{d}$$
$$\vec{u}^{k+1} = A \cdot \vec{u}^k - \vec{u}^{k-1} \tag{5.41}$$

## 5.8. Numerical solution of 2D damped wave equation

We start again by replacing the temporal and spatial derivatives in Eq. (5.31) with the finite central difference approximation on the same discretised domain as the 2D undamped wave equation (Figure 5.3), with the same boundary and the initial conditions. This now gives us the equation:

$$\frac{u_{i,j}^{k+1} - 2u_{i,j}^k + u_{i,j}^{k-1}}{\Delta t^2} + 2c \cdot \frac{u_{i,j}^{k+1} - u_{i,j}^{k-1}}{2\Delta t} = \beta^2 \cdot \frac{u_{i+1,j}^k - 2u_{i,j}^k + u_{i-1,j}^k}{\Delta x^2} + \beta^2 \cdot \frac{u_{i,j+1}^k - 2u_{i,j}^k + u_{i,j-1}^k}{\Delta y^2} \tag{5.42}$$

Using the same substitutions as for the undamped wave, we can find the equation for $u_{i,j}^{k+1}$ :

$$u_{i,j}^{k+1} = -\frac{1 - c\Delta t}{1 + c\Delta t} u_{i,j}^{k-1} + \frac{1}{1 + c\Delta t} 2\left(1 - s_x - s_y\right) u_{i,j}^k + \frac{1}{1 + c\Delta t} s_x \left(u_{i+1,j}^k + u_{i-1,j}^k\right) + \frac{1}{1 + c\Delta t} s_y \left(u_{i,j+1}^k + u_{i,j-1}^k\right) \tag{5.43}$$

Both the iterative formula and the equation for the 1st time step can be calculated using the similar matrix method as for the undamped 2D case.

For $\vec{u}^1$:

$$\vec{u}^1 = \frac{1 + c\Delta t}{2} A_d \cdot \vec{u}^0 + \frac{1 + c\Delta t}{2} \vec{b}_d + \vec{d} \tag{5.44}$$

and for all other steps, the general iterative scheme is then:

$$\vec{u}^{k+1} = A_d \cdot \vec{u}^k + \vec{b}_d - e \cdot \vec{u}^{k-1} \tag{5.45}$$

where $A_d = \frac{1}{1+c\Delta t} \cdot A$, $\vec{b}_d = \frac{1}{1+c\Delta t} \cdot \vec{b}$ and $e = \frac{1-c\Delta t}{1+c\Delta t}$ . $A$ and $\vec{b}$ are the matrix and vector, respectively, shown in Figure 5.4.

Finally, using the boundary conditions for the membrane with fixed edges all the values in the vector $\vec{b}$ will be zero and the equation for the 1st time step and the iterative formula will be reduced to:

$$\begin{aligned} \vec{u}^1 &= \frac{1+c\Delta t}{2} A_d \cdot \vec{u}^0 + \vec{d} \\ \vec{u}^{k+1} &= A_d \cdot \vec{u}^k - e \cdot \vec{u}^{k-1} \end{aligned} \tag{5.46}$$

The discrete approximation to the Laplacian operator $\nabla^2 = \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}$ used until this point is known as the five-point Laplacian (Abramowitz, 1974) or five-point stencil and it is a second-order accurate scheme. There are two possible forms of the five-point Laplacian operator, one making use of points adjacent to the centre point, and another one employing points diagonally adjacent. These two forms of the operator may be linearly combined to yield a so-called nine-point stencil which is also a second-order accurate scheme (Barkley Rosser, 1975; P. Sanz-Leon, 2017).

In Sections 5.7 and 5.8 the faster, matrix method of numerical solution of the 2D wave equations using five-point stencil is derived. This method calculates the wave propagation computationally faster than the iterative five-point stencil method, but is limited by computer memory, since the whole wave space is calculated at once.

In the next Section (5.9), we will show the nine-point Laplacian matrix method for numerical solution of the 2D damped wave equation (Barkley Rosser, 1975).

## 5.9. The nine-point stencil method for numerical solution of the 2D wave equation

The previous Sections (5.7 and 5.8) described the five-point stencil numerical scheme for numerical solution of the 2D wave equation. The difference between five-point and nine-point stencils are shown in Figure 5.5.

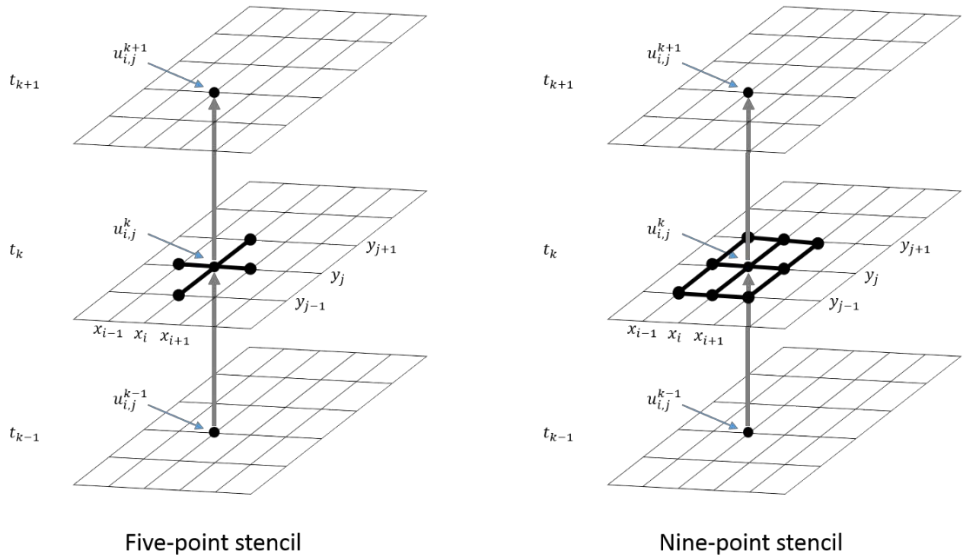


*Figure 5.5 – The five-point and the nine-point stencils*

In order to find the numerical solution for the 2D damped wave equation, with the same number of cells, boundary and initial conditions as used in the previous Chapters, using the nine-point stencil, we should start with the Eq. (5.31):

$$\frac{\partial^2 u}{\partial t^2} + 2c\frac{\partial u}{\partial t} = \beta^2\left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}\right)$$

As we know, $\nabla^2 = \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}$ is called Laplacian. Thus for the nine-point stencil we can write:

$$\frac{\partial^2 u}{\partial t^2} + 2c\frac{\partial u}{\partial t} = \beta^2 \nabla_9^2 u \tag{5.47}$$

where $\nabla_9^2$ is the nine-point Laplacian, and $\nabla_9^2 = \frac{\Delta_9}{\alpha}$ (Barkley Rosser, 1975). The operator $\Delta_9$ is:

$$\Delta_9 u(x,y) = \begin{bmatrix} 1 & c & 1 \\ b & -20 & b \\ 1 & c & 1 \end{bmatrix} u(x,y);\ b = \frac{10\Delta y^2 - 2\Delta x^2}{\Delta x^2 + \Delta y^2};\ c = \frac{10\Delta x^2 - 2\Delta y^2}{\Delta x^2 + \Delta y^2}.$$

Each coefficient in the operator $\Delta_9$ is multiplied with the corresponding point in mesh, for example, if a coefficient is $m$ units above the horizontal centre line and $n$ units to the right of the vertical centre line, we will have a product of that coefficient with $u(x + n\cdot\Delta x, y + m\cdot\Delta y)$. The entire operator denotes the sum of these products; thus we will have:

$$\Delta_9 u(x,y) = -20u_{i,j} + \frac{10\Delta y^2 - 2\Delta x^2}{\Delta x^2 + \Delta y^2}\left(u_{i+1,j} + u_{i-1,j}\right) + \frac{10\Delta x^2 - 2\Delta y^2}{\Delta x^2 + \Delta y^2}\left(u_{i,j+1} + u_{i,j-1}\right) + \left(u_{i+1,j+1} + u_{i+1,j-1} + u_{i-1,j+1} + u_{i-1,j-1}\right) \tag{5.48}$$

The coefficient $\alpha$ is:

$$\alpha = \frac{12\Delta x^2 \Delta y^2}{\Delta x^2 + \Delta y^2} \tag{5.49}$$

When we go back to the damped wave equation, Eq. (5.47), and solve it for $u_{i,j}^{k+1}$, we get:

$$u_{i,j}^{k+1} = -\frac{1-c\Delta t}{1+c\Delta t}u_{i,j}^{k-1} + \frac{1}{1+c\Delta t}\left(2-20s_m\right)u_{i,j}^{k} + \frac{s_x}{1+c\Delta t}\left(u_{i+1,j}^{k}+u_{i-1,j}^{k}\right) + \frac{s_y}{1+c\Delta t}\left(u_{i,j+1}^{k}+u_{i,j-1}^{k}\right) + \frac{s_m}{1+c\Delta t}\left(u_{i+1,j+1}^{k}+u_{i+1,j-1}^{k}+u_{i-1,j+1}^{k}+u_{i-1,j+1}^{k}\right) \quad (5.50)$$

where $s_m = C_1 s$, $s_x = C_2 s$, $s_y = C_3 s$, and $s = \frac{\beta^2 \Delta t^2}{12\Delta x^2 \Delta y^2}$, while $C_1 = \Delta x^2 + \Delta y^2$, $C_2 = 10\Delta y^2 - 2\Delta x^2$, and $C_3 = 10\Delta x^2 - 2\Delta y^2$.

Both the iterative formula and the equation for the 1st time step are exactly the same as for five-point stencil damped wave solution.

For $\vec{u}^1$:

$$\vec{u}^1 = \frac{1+c\Delta t}{2}A_d \cdot \vec{u}^0 + \frac{1+c\Delta t}{2}\vec{b}_d + \vec{d} \quad (5.51)$$

and for all other steps, the general iterative scheme is then:

$$\vec{u}^{k+1} = A_d \cdot \vec{u}^k + \vec{b}_d - e \cdot \vec{u}^{k-1} \quad (5.52)$$

where $A_d = \frac{1}{1+c\Delta t} \cdot A$, $\vec{b}_d = \frac{1}{1+c\Delta t} \cdot \vec{b}$ and $e = \frac{1-c\Delta t}{1+c\Delta t}$. $A$ and $\vec{b}$ are the matrix and vector, respectively, shown in Figure 5.6.

The update equations for the undamped wave equation using the nine-point stencil are also exactly the same as for the five-point stencil.

For $\vec{u}^1$:

$$\vec{u}^1 = \frac{1}{2}A \cdot \vec{u}^0 + \frac{1}{2}\vec{b} + \vec{d} \quad (5.53)$$

and for all other steps, the general iterative scheme is then:

$$\vec{u}^{k+1} = A \cdot \vec{u}^{k} + \vec{b} - \vec{u}^{k-1} \tag{5.54}$$

but now, $A$ and $\vec{b}$ are the matrix and vector, respectively, shown in Figure 5.6, with $\lambda = 2 - 20s_m$.

| | $u_{11}$ | $u_{12}$ | $u_{13}$ | $u_{21}$ | $u_{22}$ | $u_{23}$ | $u_{31}$ | $u_{32}$ | $u_{33}$ |
|---|---|---|---|---|---|---|---|---|---|
| $u_{11}$ | $\lambda$ | $S_y$ | 0 | $S_x$ | $S_m$ | 0 | 0 | 0 | 0 |
| $u_{12}$ | $S_y$ | $\lambda$ | $S_y$ | $S_m$ | $S_x$ | $S_m$ | 0 | 0 | 0 |
| $u_{13}$ | 0 | $S_y$ | $\lambda$ | 0 | $S_m$ | $S_x$ | 0 | 0 | 0 |
| $u_{21}$ | $S_x$ | $S_m$ | 0 | $\lambda$ | $S_y$ | 0 | $S_x$ | $S_m$ | 0 |
| $u_{22}$ | $S_m$ | $S_x$ | $S_m$ | $S_y$ | $\lambda$ | $S_y$ | $S_m$ | $S_x$ | $S_m$ |
| $u_{23}$ | 0 | $S_m$ | $S_x$ | 0 | $S_y$ | $\lambda$ | 0 | $S_m$ | $S_x$ |
| $u_{31}$ | 0 | 0 | 0 | $S_x$ | $S_m$ | 0 | $\lambda$ | $S_y$ | 0 |
| $u_{32}$ | 0 | 0 | 0 | $S_m$ | $S_x$ | $S_m$ | $S_y$ | $\lambda$ | $S_y$ |
| $u_{33}$ | 0 | 0 | 0 | 0 | $S_m$ | $S_x$ | 0 | $S_y$ | $\lambda$ |

$A$

| | |
|---|---|
| $u_{11}$ | $S_x \cdot u_{01} + S_y \cdot u_{10} + S_m \cdot (u_{20} + u_{01} + u_{00})$ |
| $u_{12}$ | $S_x \cdot u_{02} + S_m \cdot (u_{03} + u_{01})$ |
| $u_{13}$ | $S_x \cdot u_{03} + S_y \cdot u_{14} + S_m \cdot (u_{24} + u_{04} + u_{02})$ |
| $u_{21}$ | $S_y \cdot u_{20} + S_m \cdot (u_{30} + u_{10})$ |
| $u_{22}$ | 0 |
| $u_{23}$ | $S_y \cdot u_{24} + S_m \cdot (u_{14} + u_{34})$ |
| $u_{31}$ | $S_x \cdot u_{41} + S_y \cdot u_{30} + S_m \cdot (u_{42} + u_{40} + u_{20})$ |
| $u_{32}$ | $S_x \cdot u_{42} + S_m \cdot (u_{43} + u_{41})$ |
| $u_{33}$ | $S_x \cdot u_{43} + S_y \cdot u_{34} + S_m \cdot (u_{44} + u_{42} + u_{24})$ |

$\vec{b}$

*Figure 5.6 – Matrix A and vector $\vec{b}$ used for matrix method for numerical solution of 2D undamped wave equation using nine-point stencil.*

Finally, using the boundary conditions for the membrane with fixed edges all the values in the vector $\vec{b}$ will be zero and the equations for the 1st time step (Eq.(5.51) and Eq.(5.53)) and the iterative formulae (Eq.(5.52) and Eq.(5.54)) will be reduced to:

$$\begin{aligned} \vec{u}^{1} &= \frac{1 + c\Delta t}{2} A_d \cdot \vec{u}^{0} + \vec{d} \\ \vec{u}^{k+1} &= A_d \cdot \vec{u}^{k} - e \cdot \vec{u}^{k-1} \end{aligned} \tag{5.55}$$

for the damped wave, and:

$$\begin{aligned} \vec{u}^{1} &= \frac{1}{2} A \cdot \vec{u}^{0} + \vec{d} \\ \vec{u}^{k+1} &= A \cdot \vec{u}^{k} - \vec{u}^{k-1} \end{aligned} \tag{5.56}$$

for the undamped wave.

## 5.10. Numerical comparisons between five-point and nine-point stencils

In order to compare the five-point and the nine-point stencils discrete approximation methods of the Laplacian operator, the MATLAB programs for both methods were built (Appendix C and Appendix D).

The simulations were all run in MATLAB R2017a running on a laptop with Intel i7 core processor and 16GB RAM. All the simulations had the same grid size, duration of the simulation and the step size:

$$T = 12s,\ m = 2400;\ \ a = b = 2m,\ n = 199,\ p = 199$$

The mesh was initialised with the 2D Gaussian function spread over the whole mesh with the peak of the function in the central node, amplitude of 5 and variance $\sigma^2 = 4 \cdot 10^{-5} m^2$. The time of the code execution for each simulation is given in Table 5-1.

*Table 5-1 – Time of the code execution of the simulations using 5- and 9-point stencil methods to solve the 2-D wave PDE*

| Simulation: | Time [s]: |
|---|---|
| 9-point stencil – undamped wave | 2.123368 |
| 9-point stencil – damped wave | 2.233060 |
| 5-point stencil – undamped wave | 1.485464 |
| 5-point stencil – damped wave | 1.600054 |

A comparison was made between 3 points in a mesh, as shown in Figure 5.7.

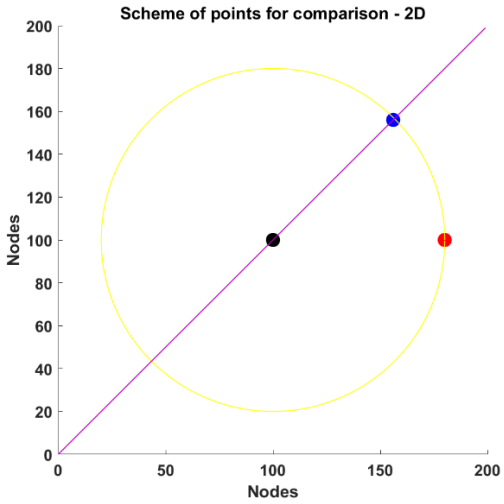


*Figure 5.7 – Three points in the mesh that are monitored in the comparisons between the 9-point and 5-point stencils. The mesh consists of 200x200 cells. Black dot represents the central point in mesh, where the stimulus is applied. Red dot is the central-right point, while blue dot is a diagonal point. Both red and blue points are distanced the same length from the central point, which is demonstrated by the yellow circle.*

First, the Figure 5.8 shows the nodes' traces when the 9-point stencil is used for solving the damped and the undamped waves. It is clear that the damped wave's oscillations are getting weaker with time and will eventually disappear.

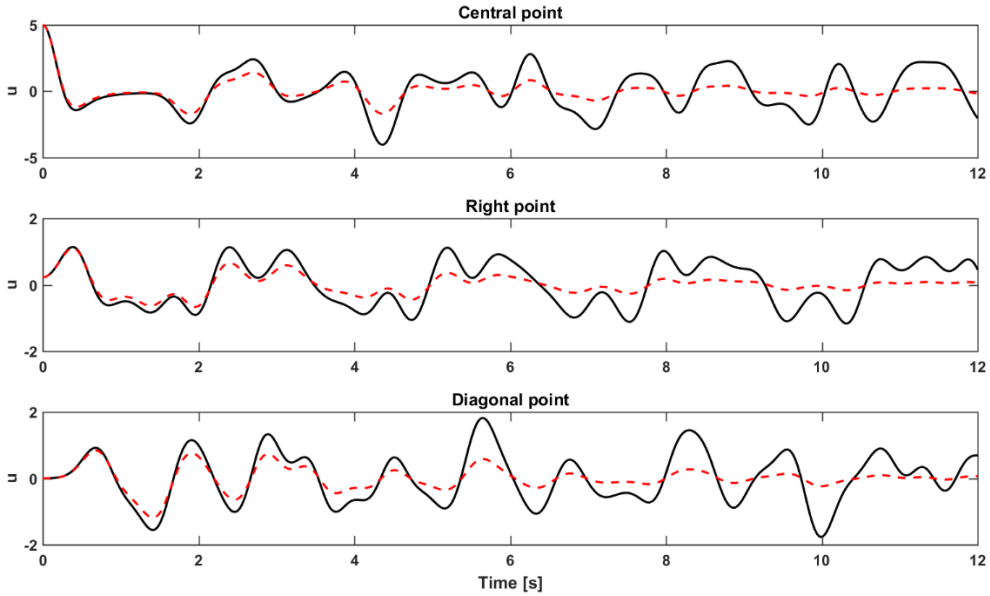


*Figure 5.8 – The effect of damping using the 9-point stencil method for the three locations specified in Figure 5.7. The red lines are damped and black lines are undamped waves.*

Next, we show the nodes' traces for 9-point and 5-point stencils in time domain. Figure 5.9 and Figure 5.10 show the undamped waves, while Figure 5.11 and Figure 5.12 are showing the damped waves. From these figures we calculated that the maximum difference is less than 0.5% between 9-point and 5-point stencils in time domain.

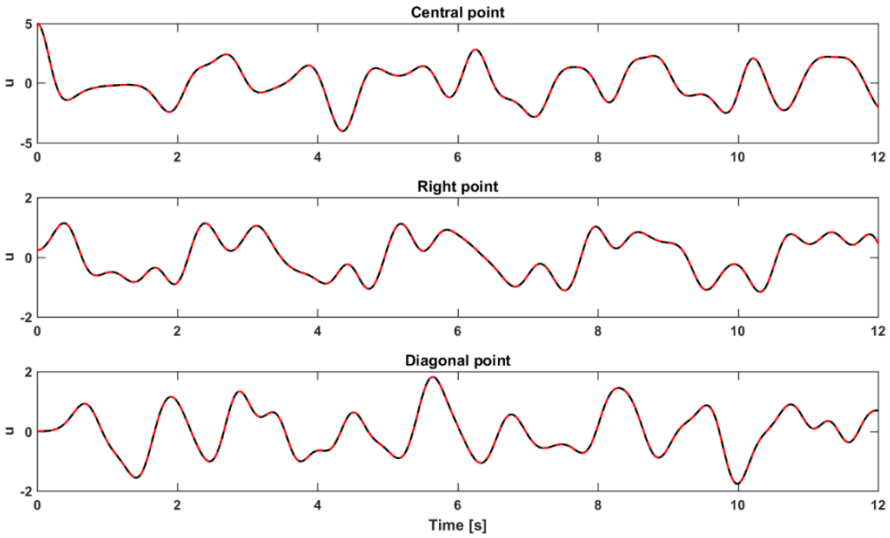


*Figure 5.9 – The undamped waveforms generated by the 9-point (red line) and 5-point (black line) stencils for the three locations specified in Figure 5.7.*

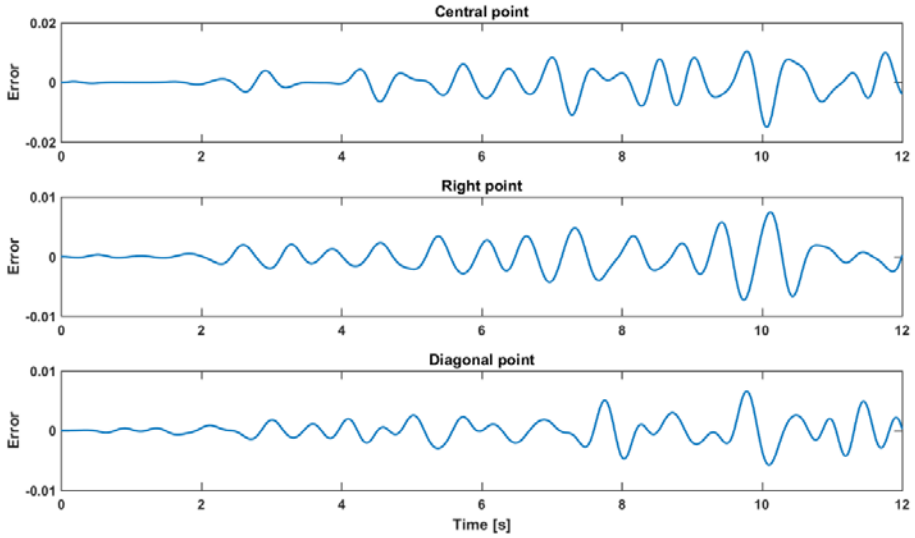


*Figure 5.10 – The accumulation of error between the undamped waveforms generated by the 9-point and 5-point stencils the three locations specified in Figure 5.7.*

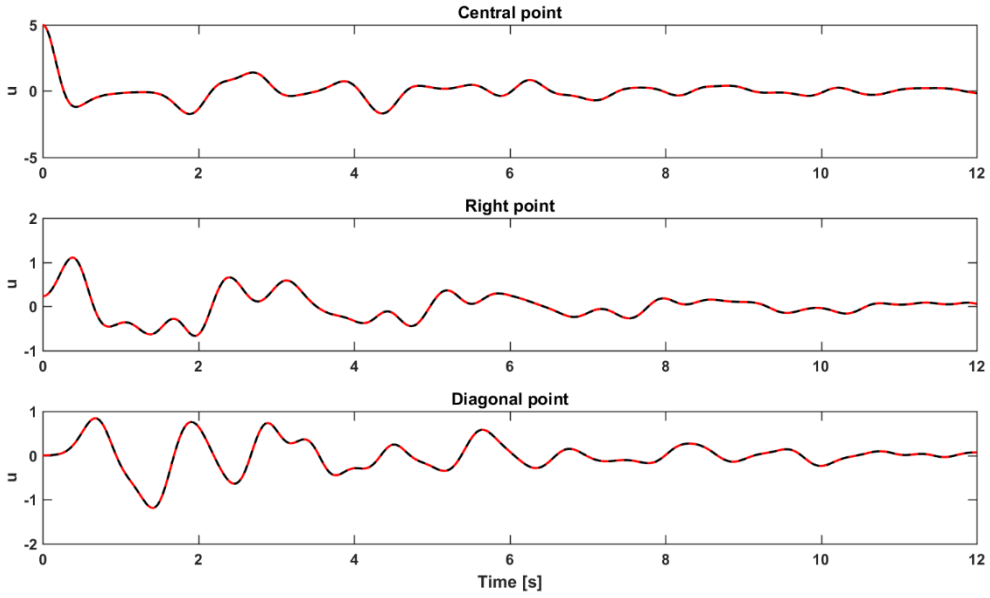


*Figure 5.11 – Damped waveforms generated by the 9-point (red line) and 5-point (black line) stencils for the three locations specified in Figure 5.7.*

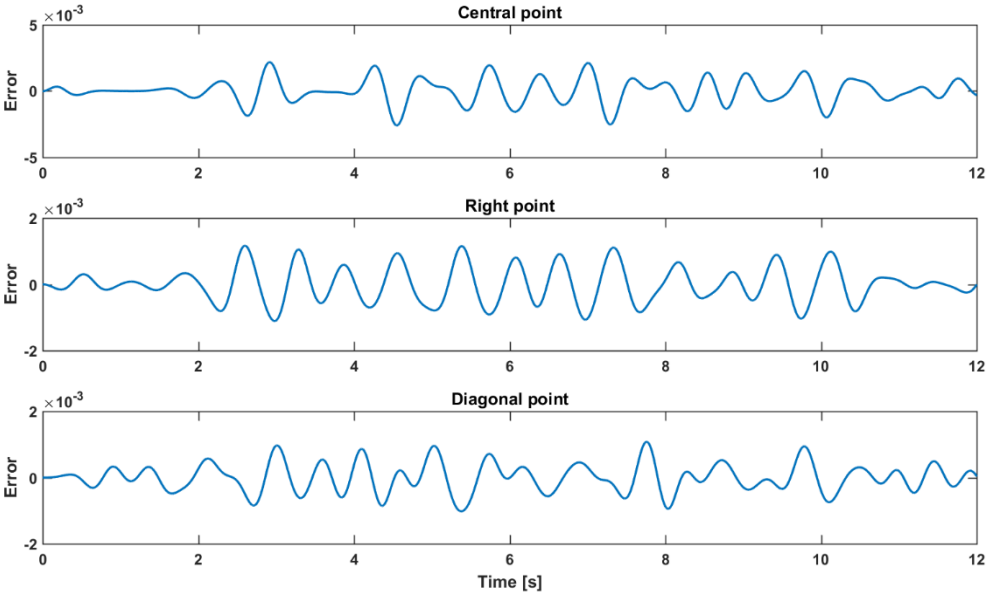


*Figure 5.12 – The accumulation of error between the damped waveforms generated by the 9-point and 5-point stencils the three locations specified in Figure 5.7.*

After that, we compared both stencils for undamped waves in frequency domain as well, Figure 5.13. There, the error between 9-point and 5-point stencils frequency spectra is multiplied by 100 so we can observe it on the same plots as the signals we are comparing.

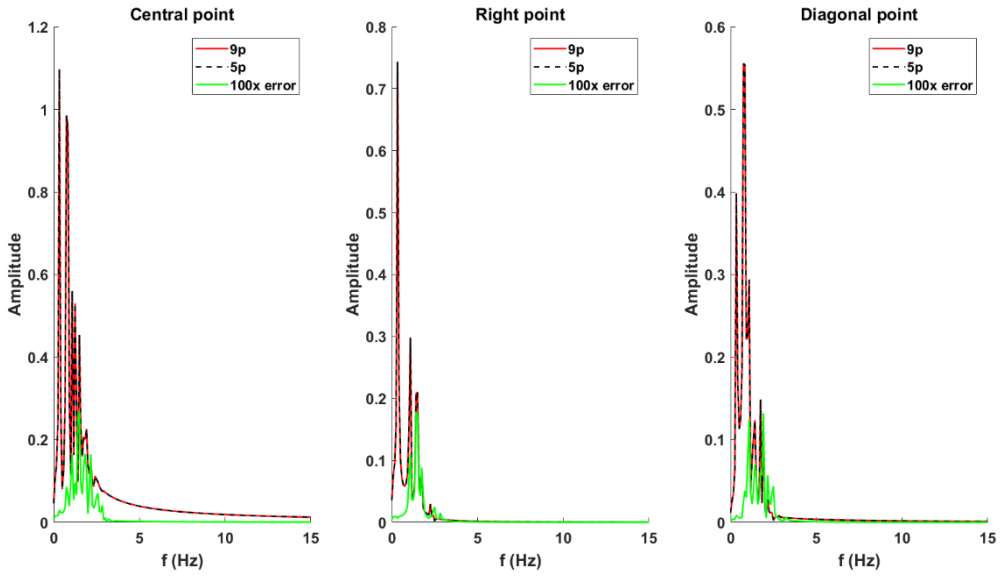


*Figure 5.13 – The frequency spectra for the undamped waveforms generated by the 9-point (red line) and 5-point (black line) stencils for the three locations specified in Figure 5.7. The error (multiplied by 100) is shown with green line.*

We also compared both stencils over the whole mesh using the correlation (*corr* function built in MATLAB, which computes $p$-values for Pearson's correlation using a Student's $t$ distribution for a transformation of the correlation), Figure 5.14, and Nash–Sutcliffe Efficiency Index (Zachary, Richard, & Cutter, 2006), Figure 5.15. Nash–Sutcliffe Efficiency Index is a common measure of model accuracy, calculated as:

$$E = 1 - \frac{\sum_{t=1}^{T} \left( Q_m^t - Q_0^t \right)^2}{\sum_{t=1}^{T} \left( Q_0^t - \overline{Q_0} \right)^2}$$

where $Q_m^t$ is the predicted (modelled) value at time $t$ , $Q_0^t$ is the observed (measured) value at time $t$ and $\overline{Q_0}$ is the mean of the observed values.

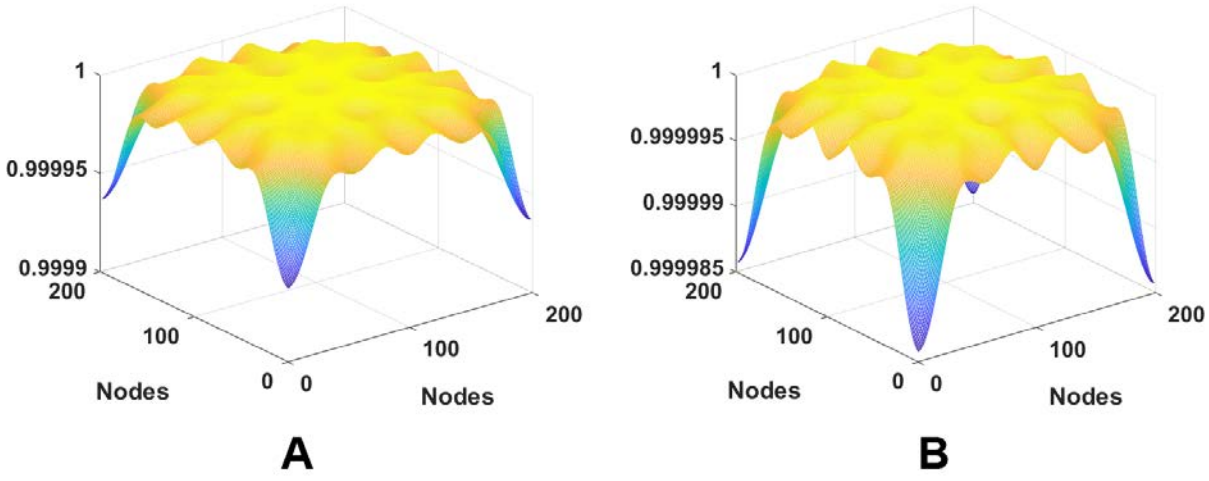


*Figure 5.14 – The correlation between 9-point and 5-point stencils for: A) undamped, B) damped, waves*

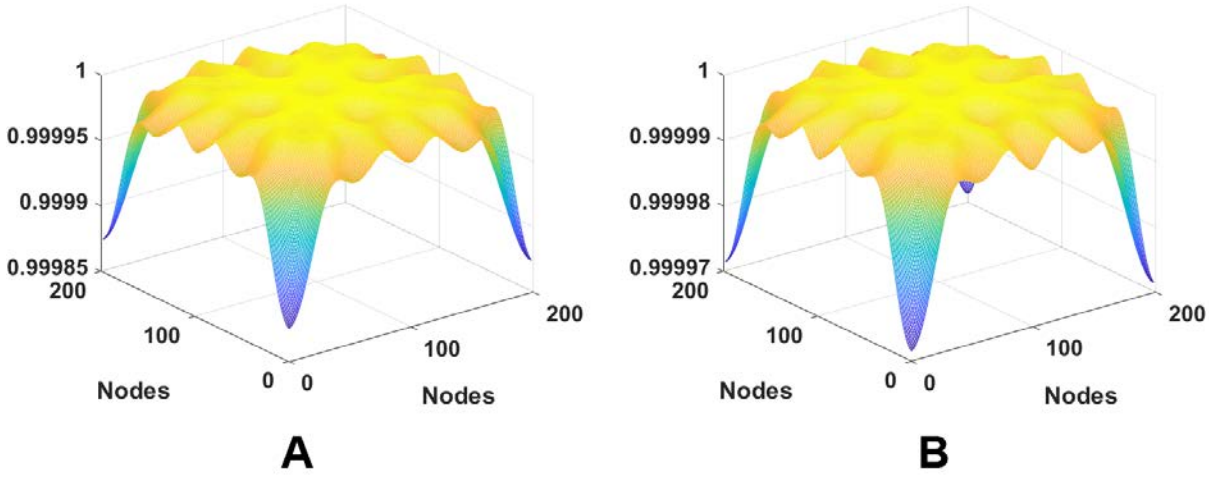


*Figure 5.15 – The Nash–Sutcliffe Efficiency Index between 9-point and 5-point stencils for: A) undamped, B) damped, waves*

Both methods show that, there is a slight difference in diagonal parts of the mesh, which was expected considering that 9-point stencil has the diagonal terms, whereas 5-point stencil doesn't. But when we inspect the numbers on the figures, we see that these

differences are so small, that we can pick 5-point stencil over 9-point stencil in our calculations without having problems with the numerical error.

## 5.11. Conclusion

In this Chapter, the possibility of using the five-point stencil to numerically solve the undamped and damped wave PDEs, instead of the nine-point stencil is explored. The results show that the five-point stencil could indeed be used instead of the nine-point stencil in NeuroField model to solve the governing damped wave equation, providing a significant speed up in code execution, without losing accuracy.

This result was discussed with the creators of NeuroField from The University of Sydney and was discovered that they were also working towards changing the method of solving the wave equation from nine-point to five-point stencil for easier understanding of the code. The presented results of our tests assured them that, by using five-point stencil, there will be significant speed up in the computation, the code will be easier to understand and all without losing accuracy. This investigation also helped us to translate NeuroField from C++ to MATLAB (presented in Appendix B) by better understanding some of the processes in NeuroField program.

# 6. Mapping NeuroField parameters to TLM

In the first part of this Chapter the numerical approximations of governing wave PDEs in NeuroField using FD method and the iterative formula are presented. The proposed TLM cell for numerically solving the inhomogeneous damped wave PDEs is presented and the PDE equivalent to the analytical solution in NeuroField is derived in the second part of this Chapter (6.2). In the same Section, the TLM node parameters are calculated to match the NeuroField parameters and finally the scattering algorithm and calculating nodal voltage for Link-Line and Link-Resistor TLM node are derived.

The last Section of this Chapter discusses the space and time discretisation, some methods' constraints and boundary conditions used in both numerical methods.

## 6.1. NeuroField wave equation in FD

The governing differential equation that represents axonal propagation of activity through the cortex in the NeuroField model is the inhomogeneous (or forced) damped wave equation relating the field $\varphi_{ab}(\vec{r},t)$ to the driving signal $Q_b(\vec{r},t)$:

$$\frac{1}{\gamma_{ab}^2}\frac{\partial^2\varphi_{ab}(\vec{r},t)}{\partial t^2}+\frac{2}{\gamma_{ab}}\frac{\partial\varphi_{ab}(\vec{r},t)}{\partial t}+\varphi_{ab}(\vec{r},t)=r_{ab}^2\nabla^2\varphi_{ab}(\vec{r},t)+Q_b(\vec{r},t) \tag{6.1}$$

This equation can be simplified by converting it into the undamped wave equation simply by introducing substitutions $u=\varphi_{ab}\exp(\gamma_{ab}t)$ and $\omega=Q_b\exp(\gamma_{ab}t)$ (P. Sanz-Leon, 2017), which gives:

$$\frac{1}{\gamma_{ab}^2}\frac{\partial^2 u(\vec{r},t)}{\partial t^2}=r_{ab}^2\nabla^2 u(\vec{r},t)+\omega(\vec{r},t) \tag{6.2}$$

This PDE is similar to the 2D undamped wave PDE, Eq. 5.1 from Chapter 5, and therefore can be solved numerically using the same five-point stencil explicit method (Olsen-Kettle, 2011). After a derivation, which is presented in detail in (P. Sanz-Leon, 2017), the explicit solution to compute future values of $u$ is:

$$\begin{aligned}u_{m,l}^{n+1}=&\left(2-4p^2\right)u_{m,l}^n+p^2\left(u_{m,l+1}^n+u_{m,l-1}^n+u_{m+1,l}^n+u_{m-1,l}^n\right)-u_{m,l}^{n-1}\\&+\frac{k^2\gamma_{ab}^2}{12}\Big[\left(10-4p^2\right)\omega_{m,l}^n+\left(\omega_{m,l}^{n+1}+\omega_{m,l}^{n-1}\right)\\&+p^2\left(\omega_{m,l+1}^n+\omega_{m,l-1}^n+\omega_{m+1,l}^n+\omega_{m-1,l}^n\right)\Big]\end{aligned} \tag{6.3}$$

where superscript $n$ is the index in time units of $k=\Delta t$, centred at current time $t$ and the future and previous states are $+1$ and $-1$ step away respectively. That means that the current state is indexed by $n$, $n+1$ denotes $0+\Delta t$ and $n-1$ denotes $0-\Delta t$. In Eq (6.3) $p=v\frac{\Delta t}{\Delta x}$ is the Courant number (see Section 6.3.2).

Finally, when the axonal field $\varphi_{ab}(\vec{r},t)$ and the driving signal $Q_b(\vec{r},t)$ are re-introduced in Eq. (6.3), the iterative formula for propagation of the axonal fields $\varphi_{ab}(\vec{r},t)$ is (P. Sanz-Leon, 2017):

$$\begin{aligned}\varphi_{m,l}^{n+1} = e^{-\gamma_{ab}\Delta t}\Big\{&\left(2-4p^2\right)\varphi_{m,l}^{n} + p^2\left(\varphi_{m,l+1}^{n}+\varphi_{m,l-1}^{n}+\varphi_{m+1,l}^{n}+\varphi_{m-1,l}^{n}\right) - \varphi_{m,l}^{n-1}\,e^{-\gamma_{ab}\Delta t}\\ &+\frac{k^2\gamma_{ab}^2}{12}\Big[\left(10-4p^2\right)Q_{m,l}^{n} + \left(Q_{m,l}^{n+1}e^{\gamma_{ab}\Delta t} + Q_{m,l}^{n-1}e^{-\gamma_{ab}\Delta t}\right)\\ &+p^2\left(Q_{m,l+1}^{n}+Q_{m,l-1}^{n}+Q_{m+1,l}^{n}+Q_{m-1,l}^{n}\right)\Big]\Big\}\end{aligned} \tag{6.4}$$

## 6.2. TLM method for inhomogeneous (or forced) damped wave equation

### 6.2.1. TLM equivalent network

In order to make a TLM model that can simulate the same inhomogeneous (forced) damped wave equation, we needed the TLM equivalent network, as shown in Figure 6.1, where $V_z$ is the voltage at a node, $I_k$ is the current generator and $R_d$ (resistance), $C_d$ (capacitance), $L_d$ (inductance), $G_d$ (conductance) are the distributed electrical parameters per unit length of the individual transmission lines making up the mesh (Desai et al., 1992). The length between two TLM nodes is $\Delta l$ .

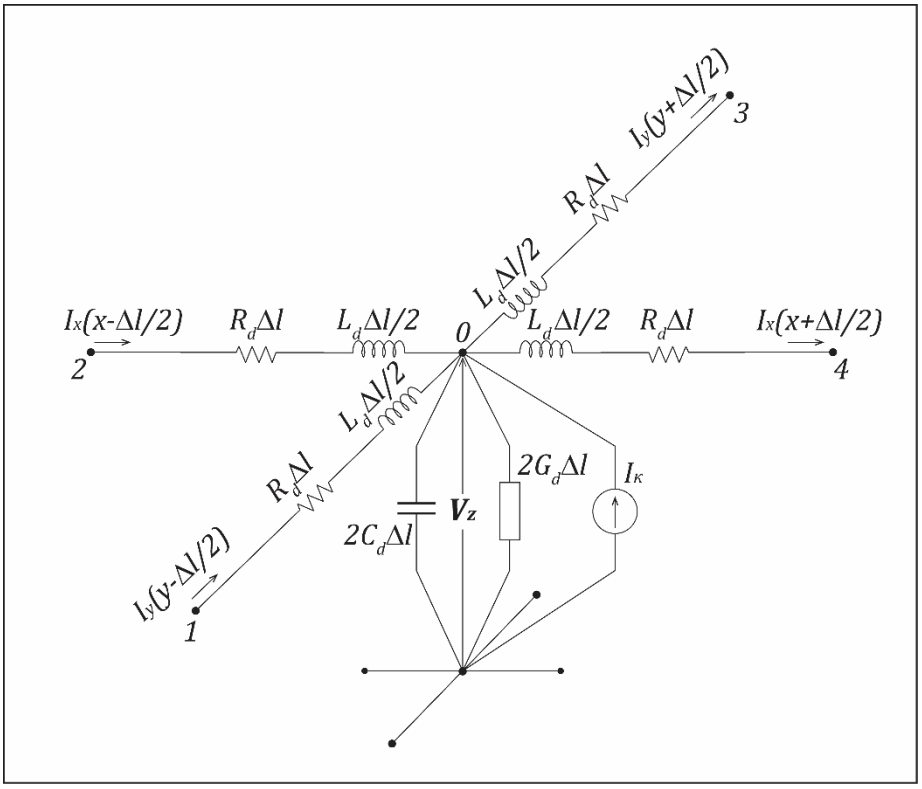


*Figure 6.1 – One cell of the 2-D TLM mesh for the model of damped wave equation represented as lumped components, similar to (Amri et al., 2011)*

After applying Kirchhoff's current law at node 0 we get:

$$-\frac{\partial I_y}{\partial y}-\frac{\partial I_x}{\partial x}+\frac{I_k}{\Delta l}=2G_dV_z+2C_d\frac{\partial V_z}{\partial t} \tag{6.5}$$

Then, applying the Kirchhoff's voltage law around the loop in y-z plane we get:

$$\frac{\partial V_z}{\partial x}=-2R_dI_y-L_d\frac{\partial I_y}{\partial t} \tag{6.6}$$

And if we do similar for x-z plane we have:

$$\frac{\partial V_z}{\partial z}=-2R_dI_x-L_d\frac{\partial I_x}{\partial t} \tag{6.7}$$

To form a wave equation, Eq. (6.5) needs to be differentiated with respect to $t$, Eq. (6.6) with respect to $y$, and Eq. (6.7) with respect to $x$:

$$
\begin{aligned}
-\frac{\partial^2 I_y}{\partial y \partial t} - \frac{\partial^2 I_x}{\partial x \partial t} + \frac{\partial I_k / \Delta l}{\partial t} &= 2G_d \frac{\partial V_z}{\partial t} + 2C_d \frac{\partial^2 V_z}{\partial t^2} \\
\frac{\partial^2 V_z}{\partial y^2} &= -2R_d \frac{\partial I_y}{\partial y} - L_d \frac{\partial^2 I_y}{\partial y \partial t} \\
\frac{\partial^2 V_y}{\partial x^2} &= -2R_d \frac{\partial I_x}{\partial x} - L_d \frac{\partial^2 I_x}{\partial x \partial t}
\end{aligned}
\tag{6.8}
$$

After substitution we get:

$$
2L_d C_d \frac{\partial^2 V_z}{\partial t^2} + \left(4R_d C_d + 2L_d G_d\right)\frac{\partial V_z}{\partial t} + 4R_d G_d V_z = \nabla^2 V_z + 2R_d \frac{I_k}{\Delta l} + \frac{L_d}{\Delta l}\frac{\partial I_k}{\partial t} \tag{6.9}
$$

In some earlier works by (Desai et al., 1992) and (Amri et al., 2011) the derivative over time for current source $I_k$ in Eq. (6.8) was set to zero, because the driving force for the diffusion equations was constant over time, but in our case the driving force, the mean firing rate $Q_b(\vec{r},t)$, varies over time and is also dependent of surrounding brain activities that are connected to the particular neural population $b$.

If we divide the NeuroField equation (6.1) by $r_{ab}^2$, we get:

$$
\frac{1}{\gamma_{ab}^2 r_{ab}^2}\frac{\partial^2 \varphi_{ab}(\vec{r},t)}{\partial t^2} + \frac{2}{\gamma_{ab} r_{ab}^2}\frac{\partial \varphi_{ab}(\vec{r},t)}{\partial t} + \frac{1}{r_{ab}^2}\varphi_{ab}(\vec{r},t) = \nabla^2 \varphi_{ab}(\vec{r},t) + \frac{1}{r_{ab}^2} Q_b(\vec{r},t) \tag{6.10}
$$

From Equations (6.9) and (6.10), the following equivalences between the TLM lumped parameters and NeuroField parameters can be drawn:

$$
\begin{aligned}
V_z &= \varphi_{ab}\left(\vec{r},t\right) \\
2L_dC_d &= \frac{1}{\gamma_{ab}^2 r_{ab}^2} \\
4R_dC_d + 2L_dG_d &= \frac{2}{\gamma_{ab} r_{ab}^2} \\
4R_dG_d &= \frac{1}{r_{ab}^2} \\
2R_d\frac{I_k}{\Delta l} + \frac{L_d}{\Delta l}\frac{\partial I_k}{\partial t} &= \frac{1}{r_{ab}^2}Q_b\left(\vec{r},t\right)
\end{aligned}
\tag{6.11}
$$

### 6.2.2. Calculation of TLM cell parameters to match NeuroField and the units analysis

To match TLM node parameters to NeuroField, the units analysis is used. First, we need to write all the parameters with their S.I. units, where $\underline{X}$ is numerical quantity of the parameter and the corresponding dimensional unit is in square brackets:

$$
\begin{aligned}
R_d &= \underline{R_d}\cdot\left[\frac{\Omega}{m}\right] = \underline{R_d}\cdot\left[\frac{s}{F\cdot m}\right] \\
C_d &= \underline{C_d}\cdot\left[\frac{F}{m}\right] = \underline{C_d}\cdot\left[\frac{s^2}{H}\right] \\
L_d &= \underline{L_d}\cdot\left[\frac{H}{m}\right] = \underline{L_d}\cdot\left[\frac{s^2}{F\cdot m}\right] \\
G_d &= \underline{G_d}\cdot\left[\frac{1}{\Omega\cdot m}\right] = \underline{G_d}\cdot\left[\frac{F}{s\cdot m}\right] = \underline{G_d}\cdot\left[\frac{s}{H\cdot m}\right] \\
V_z &= \underline{V_z}\cdot\left[V\right] \\
I_k &= \underline{I_k}\cdot\left[A\right] \\
\gamma_{ab} &= \underline{\gamma_{ab}}\cdot\left[1/s\right] \\
r_{ab} &= \underline{r_{ab}}\cdot\left[m\right] \\
\varphi_{ab}\left(\vec{r},t\right) &= \underline{\varphi_{ab}}\cdot\left[1/s\right] \\
Q_b\left(\vec{r},t\right) &= \underline{Q_b}\cdot\left[1/s\right]
\end{aligned}
$$

This way, while calculating the parameters for TLM node, we can immediately check if the units on both sides of the equations are the same.

Starting from the impedance of the transmission line $Z_0$ we have the following:

$$Z_0 = \sqrt{\frac{L_d}{2C_d}} \Rightarrow \underline{Z_0} \cdot [\Omega] = \sqrt{\frac{\underline{L_d} \cdot \left[\frac{s^2}{\cancel{F} \cdot m}\right]}{2\underline{C_d} \cdot \left[\frac{\cancel{F}}{m}\right]}} = \sqrt{\frac{\underline{L_d}}{2\underline{C_d}}} \cdot [\Omega]$$

$$\Rightarrow \ \underline{L_d} = 2\underline{C_d}\,\underline{Z_0}^2 \tag{6.12}$$

The speed of the wave across the 2-D mesh can be expressed as:

$$v = \gamma_{ab} r_{ab} = \frac{1}{\sqrt{2L_d C_d}}$$

$$\underline{v} \cdot \left[\frac{m}{s}\right] = \underline{\gamma_{ab}} \cdot [1/s] \cdot \underline{r_{ab}} \cdot [m] = \frac{1}{\sqrt{\underline{L_d} \cdot \left[\frac{s^2}{\cancel{F} \cdot m}\right] \cdot 2\underline{C_d} \cdot \left[\frac{\cancel{F}}{m}\right]}}$$

$$\underline{v} \cdot \left[\frac{m}{s}\right] = \underline{\gamma_{ab} r_{ab}} \cdot \left[\frac{m}{s}\right] = \frac{1}{\sqrt{2\underline{L_d}\,\underline{C_d}}} \cdot \left[\frac{m}{s}\right]$$

Here we can substitute the expression for inductance Eq. (6.12) and calculate $\underline{L_d}$ and $\underline{C_d}$ as:

$$\underline{\gamma_{ab} r_{ab}} = \frac{1}{\sqrt{2\underline{C_d}\,\underline{Z_0}^2 \cdot 2\underline{C_d}}} \Rightarrow \underline{\gamma_{ab} r_{ab}} = \frac{1}{2\underline{C_d}\,\underline{Z_0}}$$

$$\underline{C_d} = \frac{1}{2\underline{Z_0}\,\underline{\gamma_{ab}}\,\underline{r_{ab}}} \ , \ \underline{L_d} = \frac{\underline{Z_0}}{\underline{\gamma_{ab}}\,\underline{r_{ab}}} \tag{6.13}$$

Now we should check the units for equivalence 3 from Eq. (6.11):

$$4R_d C_d + 2L_d G_d = \frac{2}{\gamma_{ab} r_{ab}^2}$$

$$4\underline{R_d} \cdot \underline{C_d} \cdot \left[\frac{\cancel{F}}{m} \cdot \frac{s}{\cancel{F} \cdot m}\right] + 2\underline{L_d} \cdot \underline{G_d} \cdot \left[\frac{\cancel{H}}{m} \cdot \frac{s}{\cancel{H} \cdot m}\right] = \frac{2}{\underline{\gamma_{ab} r_{ab}^2}} \cdot \left[\frac{s}{m^2}\right]$$

$$\left(4\underline{R_d} \cdot \underline{C_d} + 2\underline{L_d} \cdot \underline{G_d}\right) \cdot \left[\frac{s}{m^2}\right] = \frac{2}{\underline{\gamma_{ab} r_{ab}^2}} \cdot \left[\frac{s}{m^2}\right] \tag{6.14}$$

From Eq. (6.13) and (6.14) we can find the relation between $\underline{R_d}$ and $\underline{G_d}$:

$$\underline{R_d} + \underline{Z_0}^2 \underline{G_d} = \frac{Z_0}{\underline{r_{ab}}} \Rightarrow \underline{G_d} = \frac{1}{\underline{Z_0 r_{ab}}} - \frac{R_d}{\underline{Z_0}^2} \tag{6.15}$$

We should also check the units for the equivalence 4 from Eq. (6.11):

$$4R_d G_d = \frac{1}{r_{ab}^2}$$

$$4\underline{R_d} \cdot \underline{G_d} \cdot \left[ \frac{\cancel{\Omega}}{m} \cdot \frac{1}{\cancel{\Omega} \cdot m} \right] = \frac{1}{\underline{r_{ab}^2}} \cdot \left[ \frac{1}{m^2} \right]$$

$$4\underline{R_d} \cdot \underline{G_d} \cdot \left[ \frac{1}{m^2} \right] = \frac{1}{\underline{r_{ab}^2}} \cdot \left[ \frac{1}{m^2} \right] \tag{6.16}$$

From Eq. (6.15) and (6.16) we can finally calculate $\underline{R_d}$ and $\underline{G_d}$:

$$4\underline{R_d} \cdot \left( \frac{1}{\underline{Z_0 r_{ab}}} - \frac{R_d}{\underline{Z_0}^2} \right) = \frac{1}{\underline{r_{ab}^2}}$$

$$\frac{4R_d^2}{\underline{Z_0}^2} - \frac{4R_d}{\underline{Z_0 r_{ab}}} + \frac{1}{\underline{r_{ab}^2}} = 0$$

$$\frac{2R_d}{\underline{Z_0}} - \frac{1}{\underline{r_{ab}}} = 0$$

$$\underline{R_d} = \frac{Z_0}{2\underline{r_{ab}}} \ , \ \underline{G_d} = \frac{1}{2\underline{Z_0 r_{ab}}} \tag{6.17}$$

Finally, from equivalences 1 and 5, Eq. (6.11), we have:

$$\underline{V_z} \cdot [V] = \underline{\varphi_{ab}} \cdot \left[ \frac{1}{s} \right] \tag{6.18}$$

$$2\underline{\underline{R_d}}\frac{\underline{\underline{I_k}}}{\Delta l}\cdot\left[\frac{\Omega}{\cancel{m}}\cdot\frac{A}{\cancel{m}}\right]+\frac{\underline{\underline{L_d}}}{\Delta l}\frac{\partial \underline{\underline{I_k}}}{\partial t}\cdot\left[\frac{H}{\cancel{m}}\cdot\frac{1}{\cancel{m}}\cdot\frac{A}{s}\right]=\frac{1}{\underline{\underline{r_{ab}^2}}}\underline{\underline{Q_b}}\cdot\left[\frac{1}{s\cdot\cancel{m^2}}\right]$$

$$2\underline{\underline{R_d}}\frac{\underline{\underline{I_k}}}{\Delta l}\cdot[V]+\frac{\underline{\underline{L_d}}}{\Delta l}\frac{\partial \underline{\underline{I_k}}}{\partial t}\cdot[V]=\frac{1}{\underline{\underline{r_{ab}^2}}}\underline{\underline{Q_b}}\cdot\left[\frac{1}{s}\right] \qquad (6.19)$$

Since TLM is a numerical method, we are working with constant time steps $\Delta t$ instead of $\partial t$, thus $\frac{\partial \underline{\underline{I_k}}}{\partial t}=\frac{\Delta \underline{\underline{I_k}}}{\Delta t}=\frac{\underline{\underline{I_k^n}}-\underline{\underline{I_k^{n-1}}}}{\Delta t}$, where $n$ is the current iteration step of the TLM. If we divide Eq. (6.19) with $\frac{2\underline{\underline{R_d}}}{\Delta l}$ and change the derivative of $\underline{\underline{I_k}}$ with $\frac{\underline{\underline{I_k^n}}-\underline{\underline{I_k^{n-1}}}}{\Delta t}$ we get the iterative formula for calculating the driving force in our TLM model:

$$\underline{\underline{I_k^n}}=\frac{\Delta l}{\underline{\underline{Z_0}}\,\underline{\underline{r_{ab}}}}\underline{\underline{Q_b^n}}-\frac{1}{\underline{\underline{\gamma_{ab}}}}\frac{\underline{\underline{I_k^n}}-\underline{\underline{I_k^{n-1}}}}{\Delta t}$$

$$\underline{\underline{I_k^n}}=\frac{\frac{\Delta l}{\underline{\underline{Z_0}}\,\underline{\underline{r_{ab}}}}\underline{\underline{Q_b^n}}+\frac{1}{\underline{\underline{\gamma_{ab}}}\Delta t}\underline{\underline{I_k^{n-1}}}}{1+\frac{1}{\underline{\underline{\gamma_{ab}}}\Delta t}} \qquad (6.20)$$

When $n=0$ we have: $\underline{\underline{I_k^0}}=\frac{\Delta l}{\underline{\underline{Z_0}}\,\underline{\underline{r_{ab}}}}\underline{\underline{Q_b^0}}$, where $\underline{\underline{Q_b^0}}$ is the steady-state value of the neural population $b$.

From the dimensional analysis of Eq. (6.18) and Eq. (6.19) it is obvious that the units do not agree, thus to achieve full dimensional homogeneity in equivalences between NeuroField and TLM parameters we should multiply voltage $V_z$ and current $I_k$ with the appropriate unity constant $U_V=1\cdot\left[\frac{A\cdot s^2}{kg\cdot m^2}\right]$.

Finally, here is the summary of the numerical quantities for all the TLM node parameters should we want to match them to NeuroField:

$$\underline{V_z} = \underline{\varphi_{ab}}$$

$$\underline{C_d} = \frac{1}{2\underline{Z_0}\,\underline{\gamma_{ab}}\,\underline{r_{ab}}}$$

$$\underline{L_d} = \frac{\underline{Z_0}}{\underline{\gamma_{ab}}\,\underline{r_{ab}}}$$

$$\underline{R_d} = \frac{\underline{Z_0}}{2\underline{r_{ab}}}$$

$$\underline{G_d} = \frac{1}{2\underline{Z_0}\,\underline{r_{ab}}}$$

$$\underline{I_k^n} = \frac{\frac{\Delta l}{\underline{Z_0}\,\underline{r_{ab}}}\underline{Q_b^n} + \frac{1}{\underline{\gamma_{ab}}\Delta t}\underline{I_k^{n-1}}}{1+\frac{1}{\underline{\gamma_{ab}}\Delta t}}, \underline{I_k^0} = \frac{\Delta l}{\underline{Z_0}\,\underline{r_{ab}}}\underline{Q_b^0} \tag{6.21}$$

### 6.2.3. Electrical equivalent for lumped TLM cell

The lumped network can now be converted into a 2D electrical network with the set impedance $Z_0$.

Impedance $Z_0$ can be set to any arbitrary value, as shown in Figure 6.2, while values for resistance $R$ and conductance $G$ can be calculated as:

$$R = dx \cdot \underline{R_d}$$
$$G = 2 \cdot dx \cdot \underline{G_d}$$

The electrical equivalent for lumped TLM cell for numerical solution of inhomogeneous damped wave equation is shown in Figure 6.3.

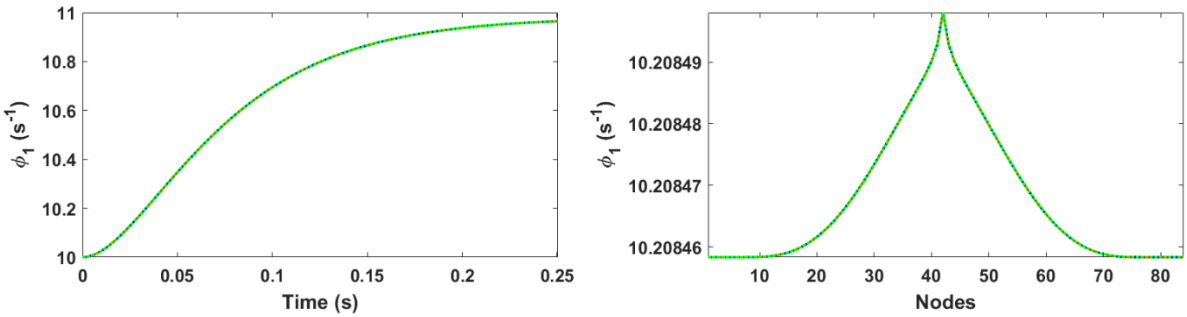


*Figure 6.2 – The influence of impedance $Z_0$ on profiles in One-population NeuroField model based on the TLM method: $Z_0 = 10^{-4}\Omega$ red line, $Z_0 = 1\Omega$ blue line, $Z_0 = 10^{4}\Omega$ green line. Left figure - central node traces for propagation field $\varphi_1$, Right figure - equatorial profiles of the same axonal fields for iteration step 49. Simulation was driven by a sine wave of amplitude $1s^{-1}$ and frequency of 20Hz, applied at the centre of the mesh.*

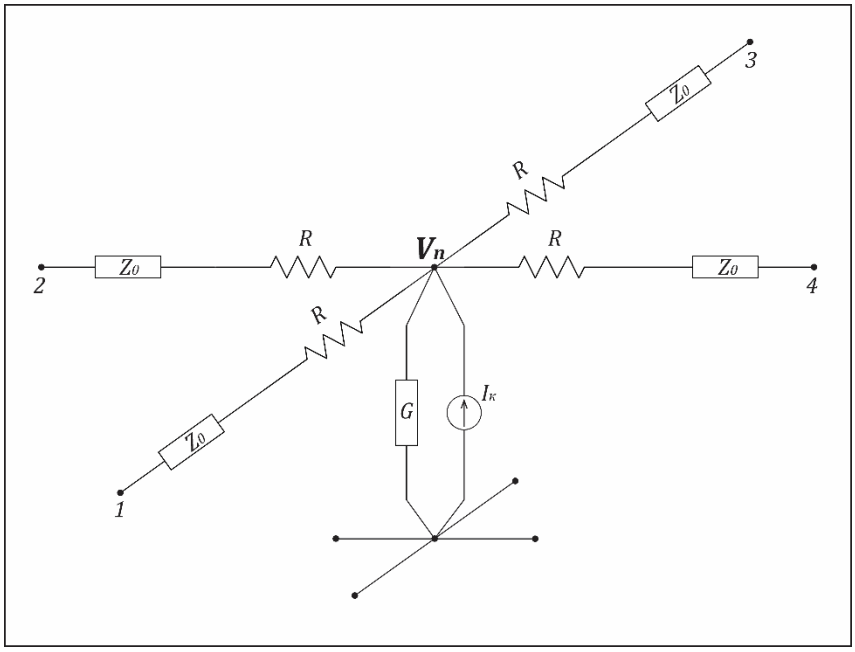


*Figure 6.3 – Lumped electrical equivalent of a TLM cell with calculated parameters to match the NeuroField model*

There are two TLM implantations in lossy formulation, depending on the relative placement of the transmission lines and resistors within a node. If we are making observations at the interface between two resistors, than that is called link-line TLM node (Figure 6.4), where as if the observations are made at the centre of the transmission line that is called link-resistor TLM node (Figure 6.6) (Cogan et al., 2005).

### 6.2.3.1. Link-Line TLM node

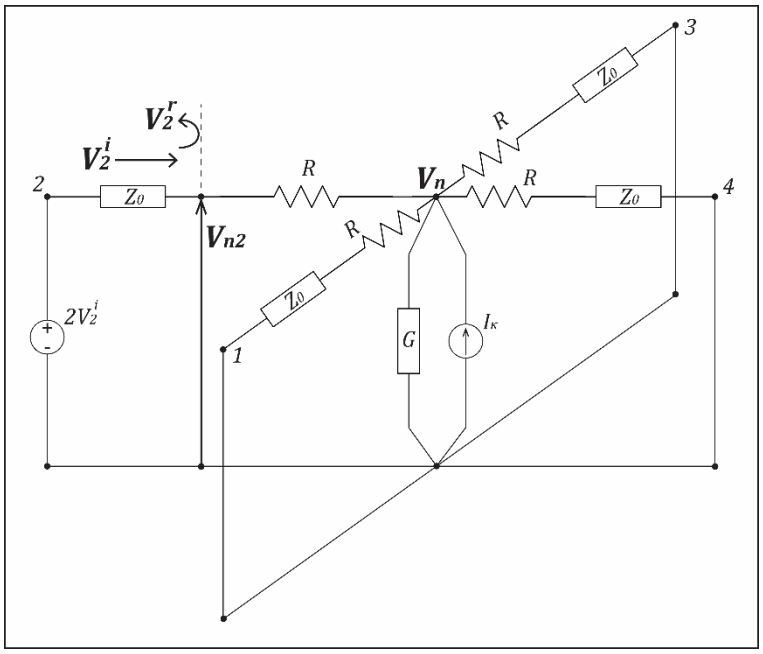


*Figure 6.4 – Lumped electrical equivalent of a TLM cell – Link-Line configuration*

We need to find the impulse scattering matrix for a proposed Link-Line TLM node as well as voltage at the node centre $V_n$.

We can calculate voltage at any point on the transmission line as $V_{n2} = V_2^i + V_2^r$, where $V_2^i$ is the voltage impulse entering a TLM node from the west, at port 2, and $V_2^r$ is the scattered (reflected) pulse from a node to port 2. In lossless TLM $V_{n2}$ is equal to node voltage $V_n$, but in link-line arrangement they are not the same and we need to calculate both separately.

First, we can calculate $V_{n2}$, which will also give us the equation for the reflected voltage $V_2^r$.

Using superposition method, we get:

$$V_{n2} = \frac{2V_2^i \cdot R + V_n \cdot Z_0}{R + Z_0} \tag{6.22}$$

Then we can find the reflected pulse:

$$\begin{aligned} V_2^r &= V_{n2} - V_2^i \\ V_2^r &= \frac{V_2^i \cdot (R - Z_0) + V_n \cdot Z_0}{R + Z_0} \end{aligned} \tag{6.23}$$

Test:

If $R = 0$ we expect that $V_{n2} = V_n$ and $V_2^r = V_n - V_2^i$:

$$V_2^r = \frac{V_2^i \cdot (0 - \cancel{Z_0}) + V_n \cdot \cancel{Z_0}}{0 + \cancel{Z_0}} = V_n - V_2^i$$

$$V_{n2} = \frac{2V_2^i \cdot 0 + V_n \cdot \cancel{Z_0}}{0 + \cancel{Z_0}} = V_n$$

The same method can be applied to calculate the reflected pulses for other ports ($V_1^r$, $V_3^r$ and $V_4^r$). Next, we need to calculate the Nodal voltage $V_n$. Let's start with the nodal voltage for impulse from direction 2 approaching the resistors at the centre of node from left (Figure 6.5).

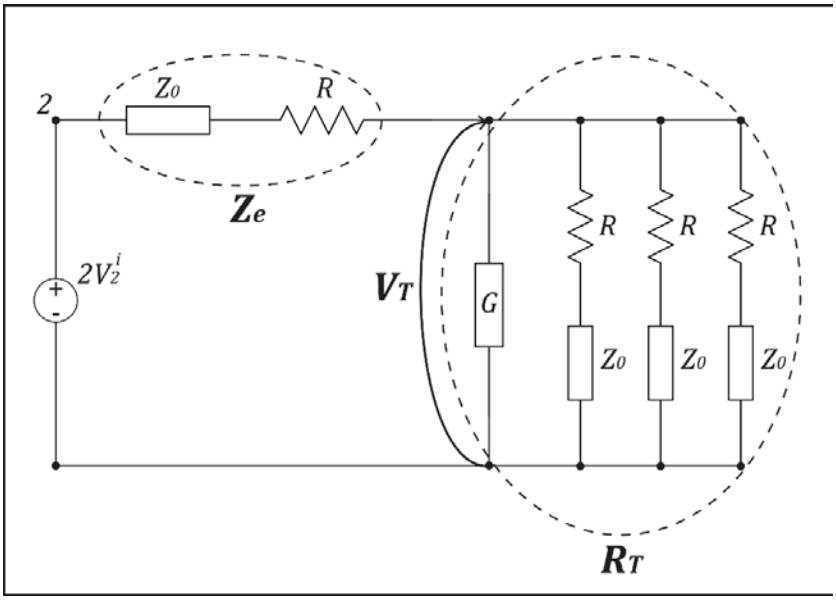


*Figure 6.5 – The Link-Line configuration – Thevenin circuit or calculation of nodal voltage*

The Thevenin equivalent circuit assumes that this pulse has originated from voltage source $2_k V_2^i$ . Using simple potential divider formula, we can calculate the contribution to the voltage at the centre of the node:

$$V_T = 2_k V_2^i \cdot \frac{R_T}{Z_e + R_T}$$

where $Z_e = R + Z_0$ and $R_T = \frac{Z_e}{3 + G \cdot Z_e}$. The incident wave from one direction, for example $V_2^i$, will contribute to the voltage at the centre of the node with:

$$V_{T2} = \frac{2_k V_2^i}{4 + (R + Z_0) \cdot G} \tag{6.24}$$

The current source (Figure 6.4) also contributes to the voltage at the node centre with:

$$V_I = \frac{I_k \cdot (R + Z_0)}{4 + (R + Z_0) \cdot G} \tag{6.25}$$

Now, total voltage at the node centre can be calculated as the sum of all contributions from all four directions and voltage from the current source:

$$ {}_kV_n = \frac{2\cdot\left({}_kV_1^i + {}_kV_2^i + {}_kV_3^i + {}_kV_4^i\right)}{4+\left(R+Z_0\right)\cdot G} + \frac{I_k\cdot\left(R+Z_0\right)}{4+\left(R+Z_0\right)\cdot G} \tag{6.26} $$

#### 6.2.3.2. Link-Resistor TLM node

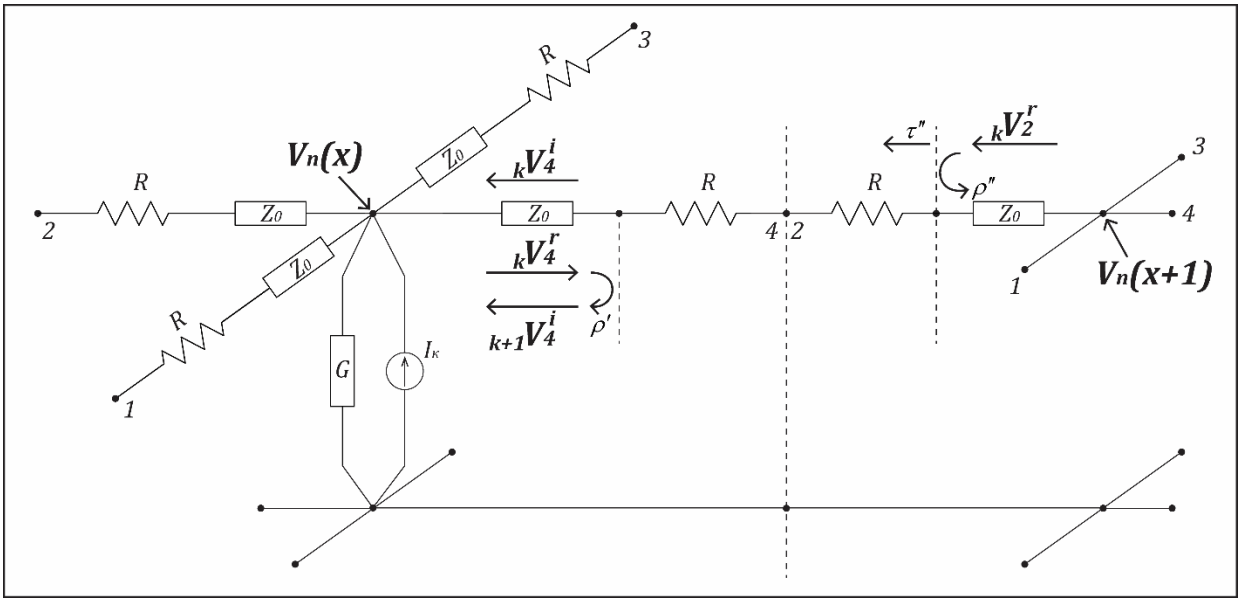


*Figure 6.6 – Lumped electrical equivalent of a TLM cell – Link-Resistor configuration*

In a 2D link-resistor node, the transmission lines are connected through two resistors, as shown in Figure 6.6. The scattered waves at discrete time intervals are identical to those in lossless TLM and can be calculated from:

$$ V_4^r = V_n - V_4^i \tag{6.27} $$

However, the presence of linking resistors will set a second scattering event that occur at the half-time intervals. This is because the pulse reflected from the node at position $(x)$, $V_4^r$, and traveling along a line, sees its resistor and the resistor and the transmission

line of the target node at $(x+1)$ as a miss-matching load. Connection equations then need to include this as:

$$\begin{aligned}
{}_{k+1}V_1^i(x,y) &= \rho' \cdot {}_kV_1^r(x,y) + \tau'' \cdot {}_kV_3^r(x,y-1) \\
{}_{k+1}V_2^i(x,y) &= \rho' \cdot {}_kV_2^r(x,y) + \tau'' \cdot {}_kV_4^r(x-1,y) \\
{}_{k+1}V_3^i(x,y) &= \rho' \cdot {}_kV_3^r(x,y) + \tau'' \cdot {}_kV_1^r(x,y+1) \\
{}_{k+1}V_4^i(x,y) &= \rho' \cdot {}_kV_4^r(x,y) + \tau'' \cdot {}_kV_2^r(x+1,y)
\end{aligned} \tag{6.28}$$

where $\rho' = R/(R+Z_0)$ and $\tau'' = 1-\rho'' = Z_0/(R+Z_0)$.

The nodal potential is then:

$${}_kV_n = \frac{2\cdot\left({}_kV_1^i + {}_kV_2^i + {}_kV_3^i + {}_kV_4^i\right)}{4+G\cdot Z_0} + \frac{I_k \cdot Z_0}{4+G\cdot Z_0} \tag{6.29}$$

In the link-line TLM nodal formulation, the “jumps-to-zero” effects can be observed as a sawtooth effect during a single-shot excitation of a spatial mesh (Cogan et al., 2005). It is a well-known anomaly in a range of numerical models when the frequencies are approaching $1/2\Delta t$. For the heat-flow and particle diffusion simulations following single-shot injection into a TLM the problem can be solved using a link-resistor TLM nodal formulation for lossy TLM (Cogan et al., 2005).

## 6.3. Discretisation and Boundary conditions

### 6.3.1. Space and time discretisation in FD method

The cortex is modelled in NeuroFeild as a 2D rectangular sheet, with edges of length $w$ and $h$ $[m]$. For all the simulations in this thesis, squared sheet, dimensions: $w = h = 0.5[m]$, is used. The number of nodes $N_x$ can be specified in the configuration

file for NeuroField model (P. Sanz-Leon, 2017), which determines the discretisation along $x$ and $y$ axes:

$$\Delta x = \Delta y = \frac{w}{N_x} \tag{6.30}$$

If cortex model is 2D square sheet, then $N_y = N_x$, Figure 6.7. Otherwise, number of nodes in $y$ axis has to be calculated from the total number of nodes specified in the configuration file of NeuroField model (P. Sanz-Leon, 2017).

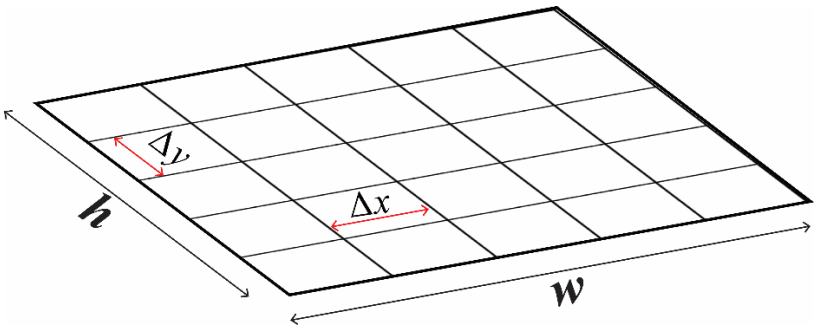


*Figure 6.7 – Discretised space for a generic cortical model in NeuroField simulations.*

### 6.3.2. Courant condition for FD numerical method

The Courant condition (Courant–Friedrichs–Lewy (CFL) condition) is a necessary condition for convergence when solving the hyperbolic PDEs numerically by the explicit FD method (Courant, Friedrichs, & Lewy, 1967). Once the length interval $\Delta x$ has been chosen and the speed of the propagating wave $v$ is known, the time step, $\Delta t$, can be evaluated using CFL in order to obtain a stable solution and a specified accuracy. In the 2D case (Press, Teukolsky, Vetterling, & Flannery, 1992) the value of the CFL number $p$ must be below:

$$p_{\max} = v\frac{\Delta t}{\Delta x} \leq \frac{1}{\sqrt{2}} \tag{6.31}$$

where $v$ is the speed of the wave and $\Delta t$ is the time step size.

This gives the maximum CFL number for NeuroField as:

$$p_{\max} = v_{ab} \frac{\Delta t}{\Delta x} \leq \frac{1}{\sqrt{2}} \tag{6.32}$$

where the speed of propagation of the field $\varphi_{ab}\left(\vec{r},t\right)$ is calculated as $v_{ab} = \gamma_{ab} r_{ab}$.

From a physical point of view, the CFL condition ensures that the propagation speed of any physical perturbation is always smaller than the numerical one which is $v_n = \Delta x / \Delta t$ (Rezzolla & Zanotti, 2014).

In (Robinson et al., 1997) a stricter condition was imposed, where the CFL number is $p = 0.1$. In some other NeuroField models (Abeysuriya, Rennie, & Robinson, 2014) $p \approx 0.06$, or in (van Albada et al., 2009) $p = 0.028$.

Although the solution of the hyperbolic PDEs is numerically stable using FD method when the CFL conditions are met, there is still a question if the discretisation domain is optimal for simulations of all the frequencies of interest. When the discretisation domain is not optimal, then there is a good chance that either space or time is poorly sampled in the simulation. In FDTD numerical simulations of EM fields, for example, there is usually one more condition that needs to be fulfilled in order to be certain that the optimal space discretisation is achieved: the grid resolution $\Delta x$ depends on the shortest wavelength $\lambda_{\min}$ of the highest frequency of interest $f_{\max}$. For a good space discretisation, it is suggested that $\frac{\Delta x}{\lambda_{\min}} \leq 0.1$.

We will give an example of the NeuroField model with the poorly sampled space in the simulation. Let the speed of the axonal field propagation in the model be $v = \gamma_{ab} r_{ab} = 6 m/s$. That is the minimal axonal velocity according to (Robinson, Rennie, Rowe, & O'Connor, 2004b). If the number of nodes in the mesh is $N_x = 30$, and the time step is $\Delta t = 10^{-4} s$ (Abeysuriya, Rennie, & Robinson, 2014), we can find the maximum frequency that can be simulated.

Firstly, the discretisation along $x$ and $y$ axes is the same and is: $\Delta x = \Delta y = \frac{w}{N_x} = 0.0167 m$. Then, if we say that the maximum frequency of interest for brain modelling is $f_{\max} = 100 Hz$, which is the maximum frequency of gamma brain waves (Hughes, 2008), we get the shortest wavelength to be $\lambda_{\min} = 0.06 m$. The ratio between and is then $\frac{\Delta x}{\lambda_{\min}} \approx 0.28$. That means we are modelling the highest frequency of interest with only $3.6\Delta x$. At the same time, we have an oversampling in time domain.

If the space discretisation fulfils the condition that $\frac{\Delta x}{\lambda_{\min}} \leq 0.1$ then the model from this example will be able to simulate the frequencies only up to $36 Hz$, which might not show some of the processes in brain during the simulation.

### 6.3.3. Space and time discretisation in TLM

The TLM method is explicit, unconditionally stable numerical method for the solution of differential equations (Peter B. Johns, 1977). The propagation velocity on a rectangular mesh depends on frequency and direction, the phenomenon called numerical dispersion

(Cogan et al., 2005). To reach the diagonal node, a wavefront needs to travel the distance of $\sqrt{2}\Delta x$, but it can only do that in $2\Delta t$, so the propagation velocity is actually:

$$v_{prop} = \frac{1}{\sqrt{2}} \frac{\Delta x}{\Delta t} \tag{6.33}$$

Due to the dispersion, in TLM velocity drops to zero when $\Delta x / \lambda = 0.25$. That means that it is not possible to propagate a wave if the discretisation is equal to four nodes per wavelength (Cogan et al., 2005). As in FDTD method, when the $\frac{\Delta x}{\lambda_{\min}} \leq 0.1$, the propagation velocity in TLM is approximated to $1/\sqrt{2}$ of the free space speed and it is considered to be almost constant (Figure 6.8). Once the speed and the space discretisation are set in TLM, time discretisation is then easily calculated from Eq. (6.33).

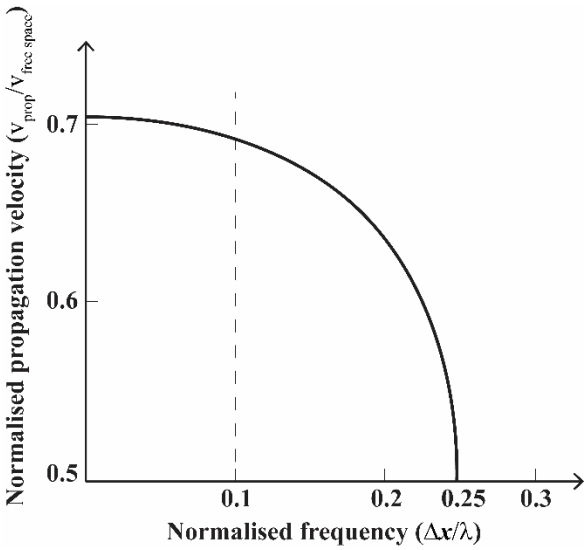


*Figure 6.8 – Normalised propagation velocity plotted versus normalised frequency to show dispersion. The maximum velocity is 70.7% of the free-space velocity (Cogan et al., 2005).*

In our MATLAB simulations of FD and TLM methods in NeuroField models, the speed of the axonal propagation was set to be the same for both methods. The CFL condition in

FD method was set to $p = \frac{1}{\sqrt{2}}$ which yielded the same discretisation in space and time for both methods.

### 6.3.4. Boundary conditions

The periodic boundary conditions (PBC) are implemented in NeuroField program (P. Sanz-Leon, 2017). When the signal reaches the far-right edge of the cortical sheet, it will emerge on the far-left side with the same velocity and continue its propagation to the right. The opposite applies for the signals travelling in the left direction. Similarly, the signals travelling towards top of the sheet will reappear again on the bottom with the same velocity, and vice versa. This produces the effect that the waves propagate on a sphere, but in topological terms, the space made by PBC can be thought of as being mapped onto a torus (Figure 6.9).

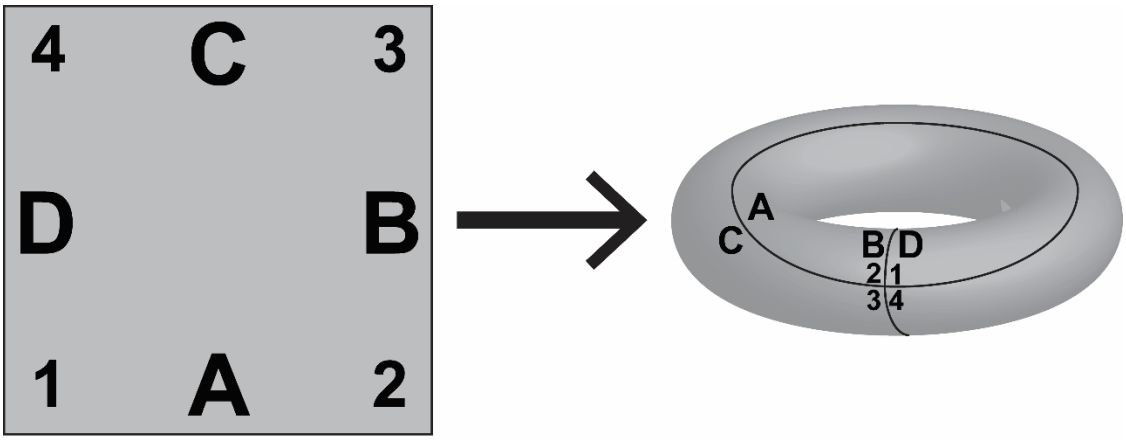


*Figure 6.9 – The illustration of periodic boundary condition (PBC) implemented in NeuroField. The left image is the 2D cortical sheet, which is mapped onto a torus (right image).*

## 6.4. Conclusion

The goal in this Chapter was to develop the TLM equivalent network capable of solving the inhomogeneous damped wave equations used in NeuroField, which was presented in Section 6.2. The electrical equivalent parameters for TLM cell were calculated in the same

Section. In Section 6.1 the equations describing the FD numerical approach in solving the damped wave equations, built into the NeuroField program, were presented. The analysis of space and time discretisation for both methods (Section 6.3) showed that TLM is unconditionally stable method, compared to FD, where the length of the cell and the time step need to be picked carefully, so that they can meet the Courant condition. Finally, the boundary conditions, implemented in NeuroField program, were discussed in subsection 6.3.4.

# 7. Comparison of the FD and TLM simulations

In this Chapter, results from MATLAB simulations are compared. The first Section (7.1) presents the comparison between FD method, using five-point stencil approach (described in Chapter 5), and TLM method to numerically solve the 2D undamped and damped wave PDEs. Two initial conditions are examined: the Dirac impulse, applied to the central node in the mesh, and the 2D Gaussian function spread over the whole mesh with the peak of the function in the central node of the mesh. Both numerical methods are compared with the analytical solutions for 2D undamped and damped wave PDEs shown in Chapter 5.

NeuroField simulations using the FD method, reprogrammed in MATLAB (Appendix B), are compared against the TLM method with the matched parameters (Appendix A) in the last Section (7.2) of this Chapter. These methods are compared in simulations of three NeuroField models consisting of One-, Two- and Four-populations. For the One-

population NeuroField model (subsection 7.2.1), simulations with 7 different stimuli are compared: Pulse, Sine waves with frequencies 12Hz and 40Hz and Gaussian waves with two different variances $\sigma^2$ depending on the same frequencies as used for Sine waves stimuli. The effects of changing the temporal damping coefficient $\gamma_{ab}$ are also examined in this subsection. In the last subsection (7.2.4) the results of the NeuroField simulation comparisons are discussed.

## 7.1. Comparison between FD and TLM methods to numerically solve the 2D wave PDEs

In Chapter 5, we derived the iterative method for numerically solving 2D wave equations, both undamped and damped, using five-point stencil method for discrete approximation of the Laplacian operator. Now, we will use the same programs built for numerical comparisons between five- and nine-point stencils in Chapter 5 (Appendix C and Appendix D) to compare their outputs with the TLM model presented in Chapter 6. The parameters for the 2D FD wave modes are calculated to match the TLM parameters from NeuroField model. It is expected that this TLM model will be able to simulate undamped and damped waves just by turning off certain electrical elements in the TLM node.

### 7.1.1. Undamped wave PDEs

Undamped wave is a lossless wave, thus in order to numerically find a solution using TLM techniques, the lossless TLM node shown in Figure 4.3 can be employed.

After solving this TLM node for voltage $V_z$, we get the Helmholtz wave equation (Sadiku, 2009) in 2D space:

$$2L_dC_d\frac{\partial^2 V_z}{\partial t^2} = \nabla^2 V_z \tag{7.1}$$

If we divide Eq. (7.1) with $2L_dC_d$ and if we recall equation for 2D undamped wave Eq. (5.23), we get the following:

$$\frac{\partial^2 u}{\partial t^2} = \beta^2 \nabla^2 u$$
$$\frac{\partial^2 V_z}{\partial t^2} = \frac{1}{2L_dC_d} \nabla^2 V_z$$

From here we can notice the equivalences:

$$u = V_z$$
$$\beta^2 = \frac{1}{2L_dC_d} \Rightarrow \beta = \frac{1}{\sqrt{2L_dC_d}}$$

where $\beta$ is the speed of the wave.

To compare the TLM model, presented in Chapter 6, against the FD method for solving the undamped wave we need to turn off all the losses and external sources, which is the current source in our case. If we set $R_d = G_d = 0$ and remove the current source $I_k$ from the Eq. (6.9) we get exactly the same equation as Eq. (7.1).

In Chapter 5 we calculated the speed of the TLM wave using NeuroField parameters as $v = \gamma_{ab} r_{ab} = \frac{1}{\sqrt{2L_dC_d}}$. Comparing two expressions for wave speed, we get that $\beta = v$.

#### *7.1.1.1. Comparison of the simulations for 2D undamped wave*

Simulations of 2D undamped wave using FD and TLM techniques were run in MATLAB R2017a running on a laptop with Intel i7 core processor and 16GB RAM. Simulation parameters were the same for both programs and are shown in the Table 7-1. All boundaries were set to be perfectly reflective and two different initial conditions (I.C.) were examined: Dirac impulse (Figure 7.1) and 2D Gaussian spread (Figure 7.2).

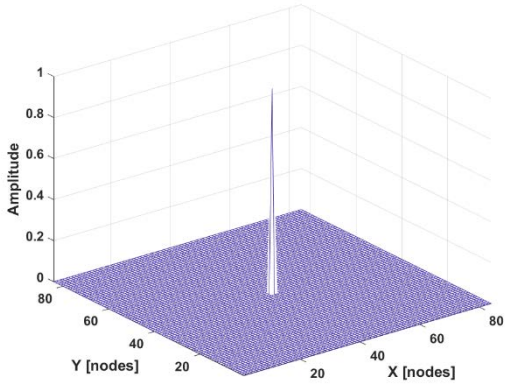


*Figure 7.1 – Dirac impulse initial condition.*

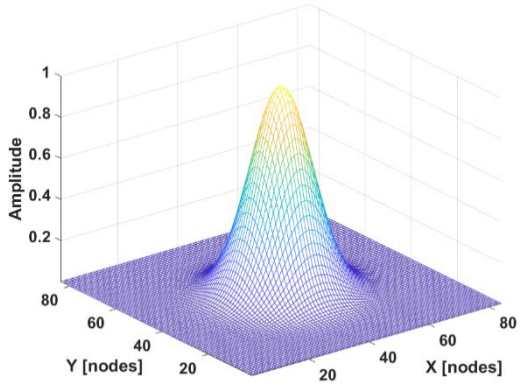


*Figure 7.2 – 2D Gaussian spread initial condition*

*Table 7-1 – Simulation parameters for 2D undamped wave*

| $r_{ab}$ | $\gamma_{ab}$ | $v = \beta$ | $f_{\max}$ | $w = h$ | $N_x = N_y$ | $sim\_time$ | $Steps$ |
|---|---|---|---|---|---|---|---|
| $0.2m$ | $30s^{-1}$ | $6\,m/s$ | $100Hz$ | $0.5m$ | 84 | $0.06s$ | 89 |

#### 7.1.1.1.1. Dirac impulse I.C.

For this simulation, we applied Dirac impulse (Figure 7.1) to the central node in the mesh with the amplitude of 1 and other nodes are set to 0. Time-series of the middle nodes for both approaches are inspected and their traces are compared with the analytical solution in Figure 7.3.

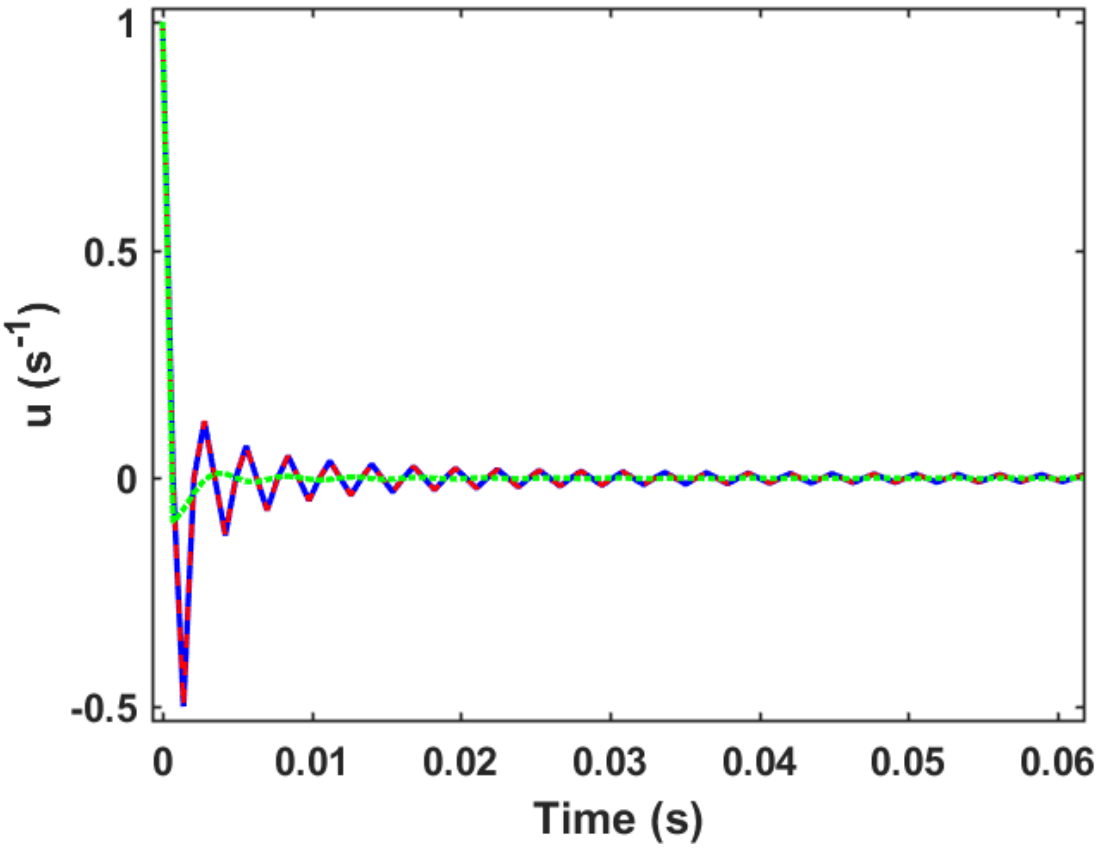


*Figure 7.3 – Central node traces for 2D undamped waves with Dirac Impulse I.C. Analytical solution – green dotted line, FD – blue line, TLM – red dashed line*

The voltages across the central horizontal lines of both meshes were inspected and compared with the analytical solution, which showed us the spreading of the waves in time Figure 7.4.

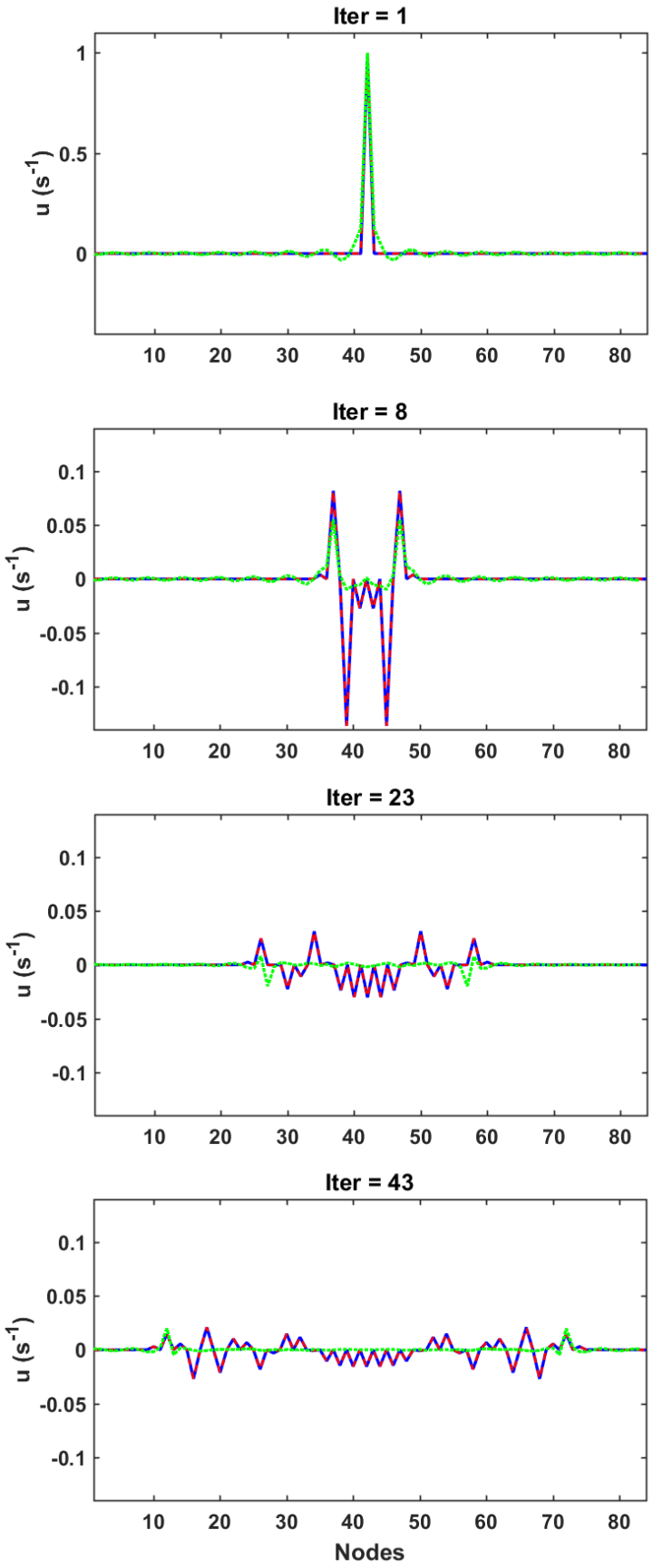


*Figure 7.4 – Equatorial profiles for 2D undamped waves with Dirac Impulse I.C. in four iteration steps. Analytical solution – green dotted line, FD – blue line, TLM – red dashed line*

#### 7.1.1.1.2. Gaussian spread I.C.

In order to excite the mesh with the broad frequency range, but to avoid the high frequencies, the following simulations were run with the 2D Gaussian function spread (Figure 7.2) over the whole mesh with the peak of the function in the central node, amplitude of 1 and variance $\sigma^2 = 0.025m^2$ . Figure 7.5 shows the traces of the middle nodes for both approaches and Figure 7.6 are the equatorial profiles of both meshes. Analytical solution for the Gaussian spread I.C. wasn't found due to its complexity.

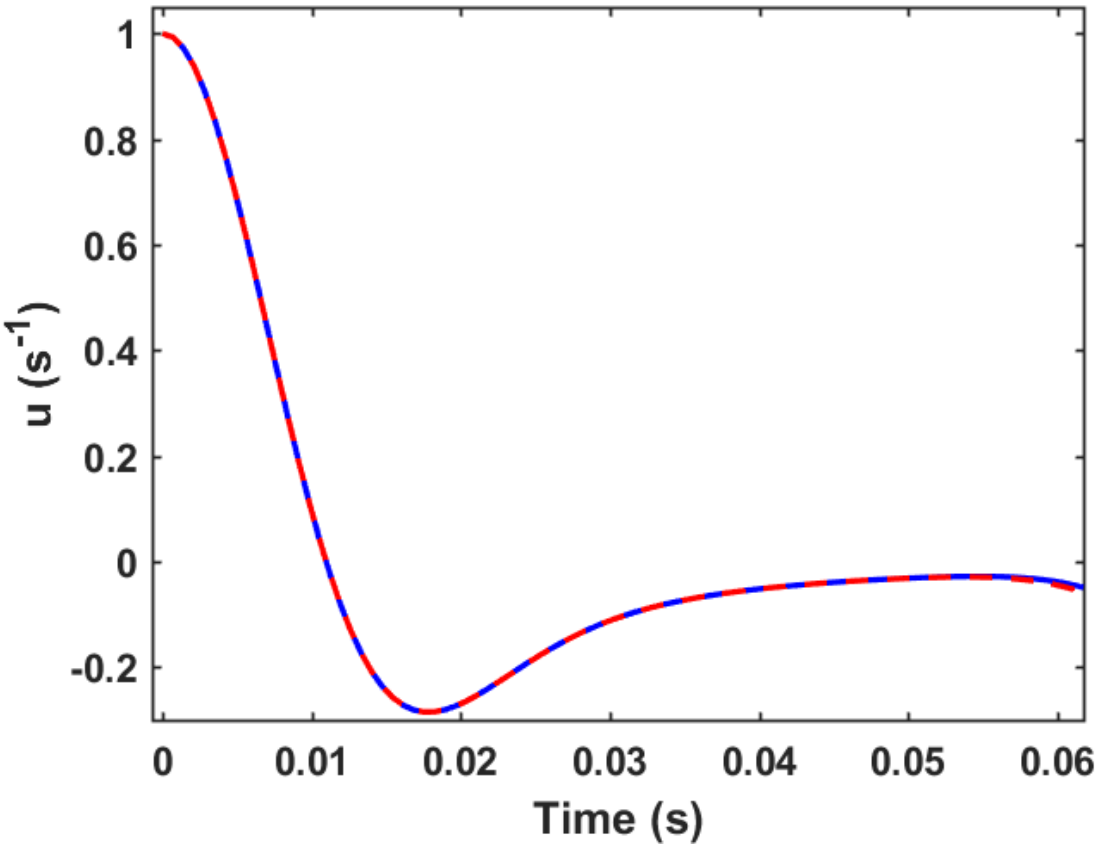


*Figure 7.5 – Central node traces for 2D undamped waves with Gaussian I.C. FD – blue line, TLM – red dashed line.*

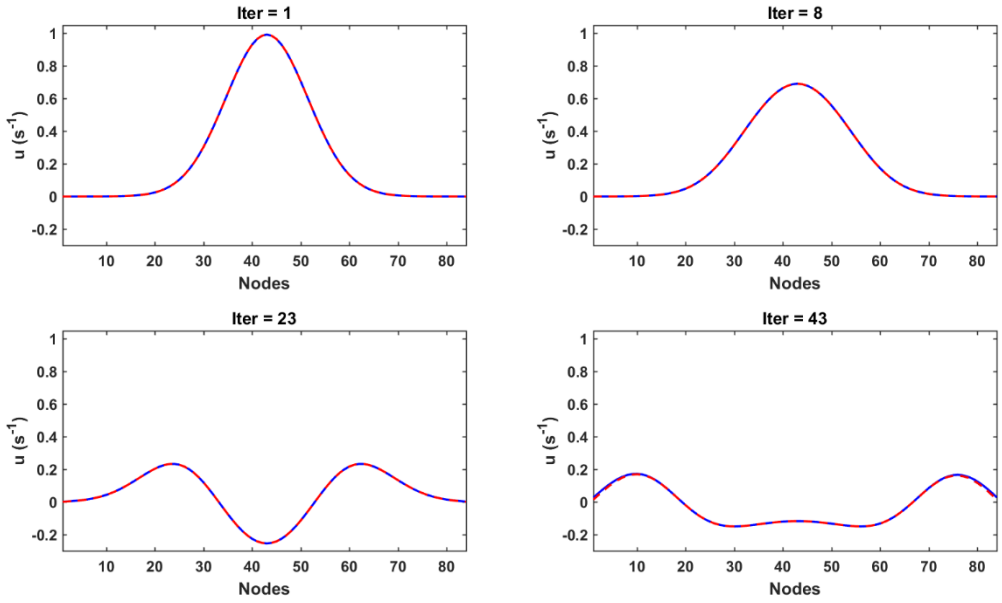


*Figure 7.6 – Equatorial profiles for 2D undamped waves with Gaussian I.C. in four iteration steps. FD – blue line, TLM – red dashed line.*

#### *7.1.1.2. Discussion of the results*

It can be observed from Figure 7.3 and Figure 7.4 that the numerical approximation of the analytical solution of the 2D undamped wave PDE using FD and TLM methods is the same for Dirac impulse I.C.. When comparing FD and TLM methods for Gaussian I.C we may notice slight differences towards the ends of the traces in Figure 7.5 and for the iteration step 43 in Figure 7.6. To quantitatively describe differences in the central traces, we used Nash-Sutcliffe Efficiency Index (Zachary et al., 2006) and the results are shown in Table 7-2:

*Table 7-2 – Nash-Sutcliffe Efficiency Index showing how similar are the central node traces between analytical solution and numerical methods and between FD and TLM when simulating 2D undamped wave*

| | **Dirac Impulse I.C.** | **Gaussian I.C.** |
|---|---|---|
| **FD vs. analytical** | 0.7441 | N/A |
| **TLM vs. analytical** | 0.7441 | N/A |
| **TLM vs. FD** | 1 | 0.999964 |

We can conclude from the Table 7-2 that for the simulations when the Dirac impulse is set as the I.C. we get the perfect match between two numerical methods. There is a slight difference in methods, of the order of $10^{-5}$ when the Gaussian is set as the I.C.

### 7.1.2. Damped wave PDEs

To compare 2D damped wave numerical solutions using FD and TLM methods, we should recall the differential equation for 2D damped wave from Chapter 5, Eq. (5.31) and TLM equation for inhomogeneous damped wave equation from Chapter 6, Eq. (6.9). For easier explanation, we will repeat these two equations here:

$$\frac{\partial^2 u}{\partial t^2} + 2c\frac{\partial u}{\partial t} = \beta^2 \nabla^2 u$$

$$2L_d C_d \frac{\partial^2 V_z}{\partial t^2} + \left(4R_d C_d + 2L_d G_d\right)\frac{\partial V_z}{\partial t} + 4R_d G_d V_z = \nabla^2 V_z + 2R_d \frac{I_k}{\Delta l} + \frac{L_d}{\Delta l}\frac{\partial I_k}{\partial t} \tag{7.2}$$

We notice that for these to equations to be equivalent, we should set $G_d = 0$ and turn off the current source $I_k$ in the TLM equation. That way we get the lossy wave equation, also known as the telegrapher's equation:

$$2L_d C_d \frac{\partial^2 V_z}{\partial t^2} + 4R_d C_d \frac{\partial V_z}{\partial t} = \nabla^2 V_z \tag{7.3}$$

If we divide the 2D damped wave equation by $\beta^2$ we get:

$$\frac{1}{\beta^2}\frac{\partial^2 u}{\partial t^2} + \frac{2c}{\beta^2}\frac{\partial u}{\partial t} = \nabla^2 u \tag{7.4}$$

From Eq. (7.3) and Eq. (7.4) we can find the following equivalences:

$$u = V_z$$
$$\beta^2 = \frac{1}{2L_dC_d} \Rightarrow \beta = v$$
$$\frac{2c}{\beta^2} = 4R_dC_d \Rightarrow c = \frac{\gamma_{ab}}{2}$$

Electrical equivalent TLM node in this case can be either link-line or link-resistor type node. For our simulations, we picked link-line lossy node, shown in Figure 6.4.

The reflected pulse in this case can be calculated using the same equation as Eq. (6.23) and nodal voltage can be found by removing $I_k$ term from the Eq. (6.26), which gives us the same expression for nodal voltage as Eq. (4.2).

#### *7.1.2.1. Comparison of the simulations for 2D damped wave*

Simulations of 2D damped wave using FD and TLM methods were run on the same platform as the simulations for undamped wave using the same parameters as shown in the Table 7-1, with one extra parameter for damping, $c = \frac{\gamma_{ab}}{2} = 15s^{-1}$. All boundaries were set to be perfectly reflective and two different I.C. were examined again: Dirac impulse (Figure 7.1) and 2D Gaussian spread (Figure 7.2).

##### 7.1.2.1.1. Dirac impulse I.C.

For this simulation, we again applied the Dirac impulse (Figure 7.1) to the central node in the mesh with the amplitude of 1 and other nodes are set to 0. Time-series of the middle nodes for both approaches are inspected and their traces are compared with the analytical solution in Figure 7.7.

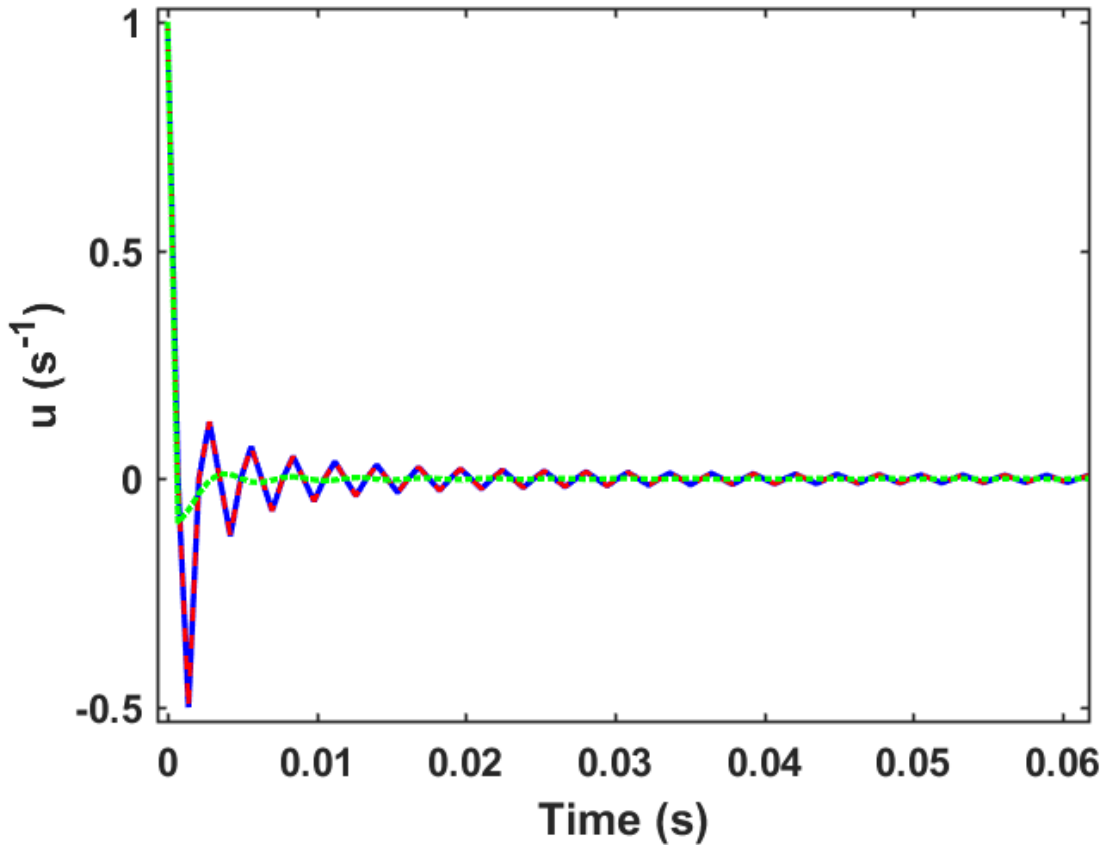


*Figure 7.7 – Central node traces for 2D damped waves with Dirac Impulse I.C. Analytical solution – green dotted line, FD – blue line, TLM – red dashed line*

We also inspected equatorial profiles of both meshes and compared them with the analytical solution, which showed us the spreading of the waves in time Figure 7.8.

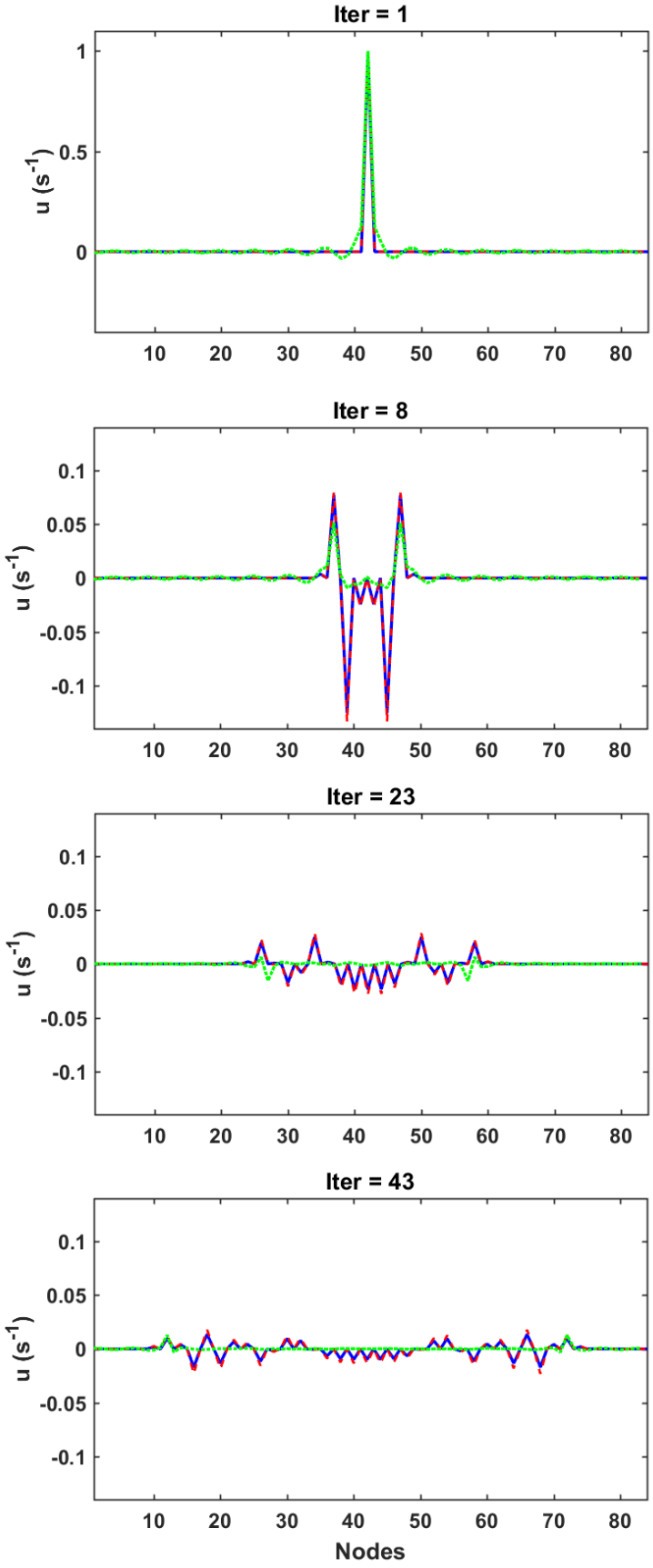


*Figure 7.8 – Equatorial profiles for 2D damped waves with Dirac Impulse I.C. in four iteration steps. Analytical solution – green dotted line, FD – blue line, TLM – red dashed line*

#### 7.1.2.1.2. Gaussian spread I.C.

These simulations were run with the 2D Gaussian function spread (Figure 7.2) over the whole mesh with the peak of the function in the central node, with the same amplitude of 1 and variance $\sigma^2 = 0.025m^2$ as for undamped wave simulations. Figure 7.9 shows the traces of the middle nodes for both approaches and Figure 7.10 are the equatorial profiles of both meshes. Analytical solution for the Gaussian spread I.C. wasn't found due to its complexity.

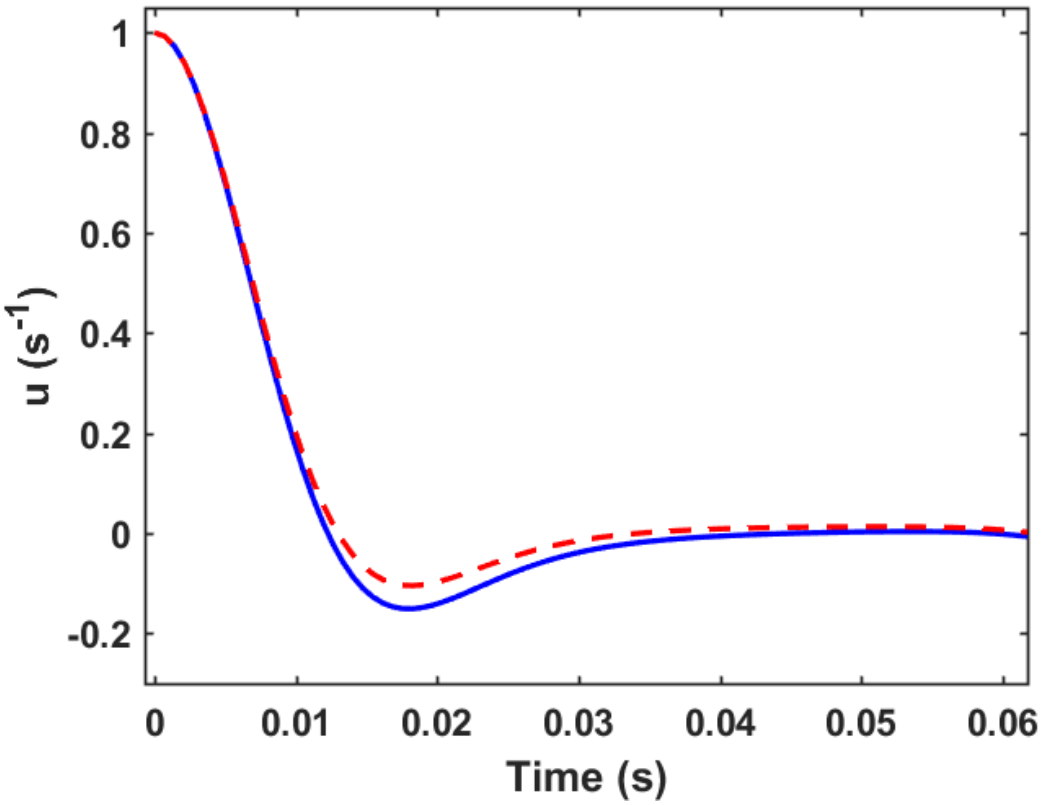


*Figure 7.9 – Central node traces for 2D damped waves with Gaussian I.C. FD – blue line, TLM – red dashed line*

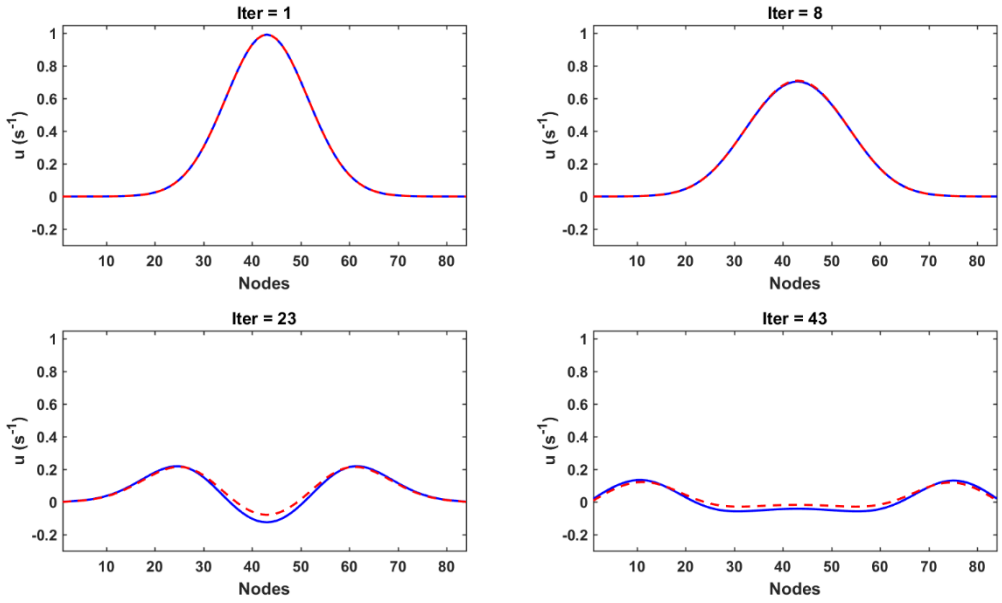


*Figure 7.10 – Equatorial profiles for 2D damped waves with Gaussian I.C. in four iteration steps. FD – blue line, TLM – red dashed line*

#### *7.1.2.2. Discussion of the results*

Visual observations of Figure 7.7 and Figure 7.8 show that the numerical approximation of the analytical solution for the damped wave PDE using FD and TLM methods should be the same for Dirac impulse I.C., but for Gaussian I.C. we can notice slight differences in the traces in Figure 7.9 and in Figure 7.10 for the iteration steps 8 to 43. We compared the central traces again using Nash-Sutcliffe Efficiency Index and the results are shown in Table 7-3:

*Table 7-3 – Nash-Sutcliffe Efficiency Index showing how similar are the central node traces between analytical solution and numerical methods and between FD and TLM when simulating 2D damped wave*

| | **Dirac Impulse I.C.** | **Gaussian I.C.** |
|---|---|---|
| **FD vs. analytical** | 0.767604 | N/A |
| **TLM vs. analytical** | 0.749382 | N/A |
| **TLM vs. FD** | 0.999869 | 0.992915 |

We can notice from the Table 7-3 that for the simulations when the Dirac impulse is set as the I.C. FD method slightly better approximates the analytical solution. When the

Gaussian is set as the I.C. the difference between numerical methods is an order of magnitude larger than for Dirac impulse and 2 orders of magnitude larger than for Gaussian I.C. in undamped wave. Nevertheless, this is still considered an excellent match between FD and TLM methods.

## 7.2. Comparing the NeuroField simulations using FD and TLM methods for solving the governing wave equation in MATLAB

In this Section, the NeuroField simulations using the FD method, reprogrammed in MATLAB, are compared against the TLM method with the matched parameters. These methods are compared in simulations of three NeuroField models consisting of One-, Two- and Four- neuronal populations.

### 7.2.1. One-population NeuroField model

One-population NeuroField model is the simplest model that can be used to illustrate the neural dynamics in NeuroField. Although it is called One-population model, it effectively consists of two populations: one is stimulus and the other one is excitatory neural population, Figure 7.11.

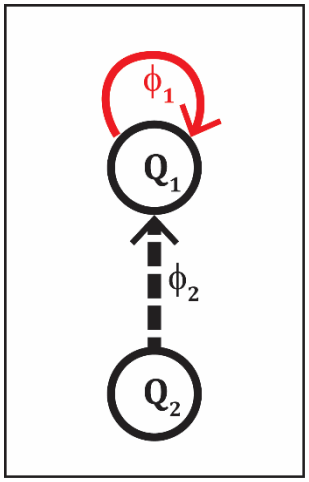


*Figure 7.11 – Block diagram of One-population NeuroField model. Red arrow indicates inhomogeneous damped wave propagation; thick, dashed black arrow indicates stimulus propagation.*

Parameters for this model were taken from (Robinson et al., 2004b), with the only difference in the axonal propagation parameters which are chosen, for illustrative purposes, as the maximum, $r_{ab} = 0.2m$, and the minimum, $\gamma_{ab} = 30s^{-1}$, values of the given range, obtainable without effecting model-based constraints. The rest of the parameters used in these simulations are shown in Table 7-4, where $Q_1^0$ is the steady state, low firing rate in which the system was initially in.

*Table 7-4 – Simulation parameters for One-population NeuroField model*

| $v = \beta$ | $N_x = N_y$ | $sim\_time$ | $Steps$ | $Q_1^0$ | $\upsilon_1$ | $\upsilon_2$ |
|---|---|---|---|---|---|---|
| $6\,m/s$ | 84 | $0.15s$ | 357 | $10s^{-1}$ | 0 | $10^{-4}Vs$ |

In order to compare FD and TLM methods, the simulations with five different stimuli are run: Pulse, Sine waves with frequencies 12Hz and 40Hz and Gaussian pulses with two different variances $\sigma^2$ depending on the same frequencies as used for Sine waves stimuli. The MATLAB code for the TLM method is presented in Appendix A, and for the FD method in Appendix B.

#### *7.2.1.1. Pulse stimulus*

For this simulation, the model was driven by a pulse applied in the centre of the grid, with the amplitude of $1s^{-1}$ and the duration of $10dt$. Time-series of the central nodes for both methods were inspected and their traces are showed in Figure 7.12. Neural activity distributions are shown in Figure 7.13 for FD and Figure 7.14 for TLM methods. Equatorial profiles of axonal fields are compared in Figure 7.15, and their power spectrums were compared in Figure 7.16.

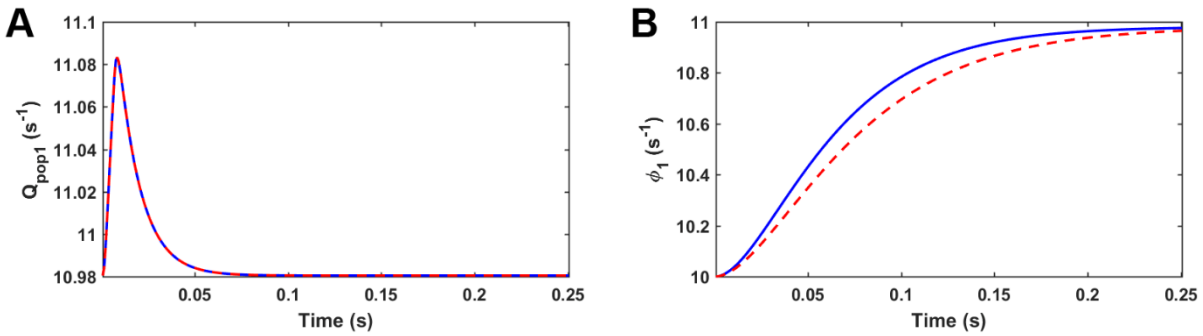


*Figure 7.12 – Central node traces for One-population NeuroField model driven by a pulse. A) Mean firing rate for population 1, $Q_1$ ; B) propagation field $\varphi_1$. FD – blue line, TLM – red dashed line*

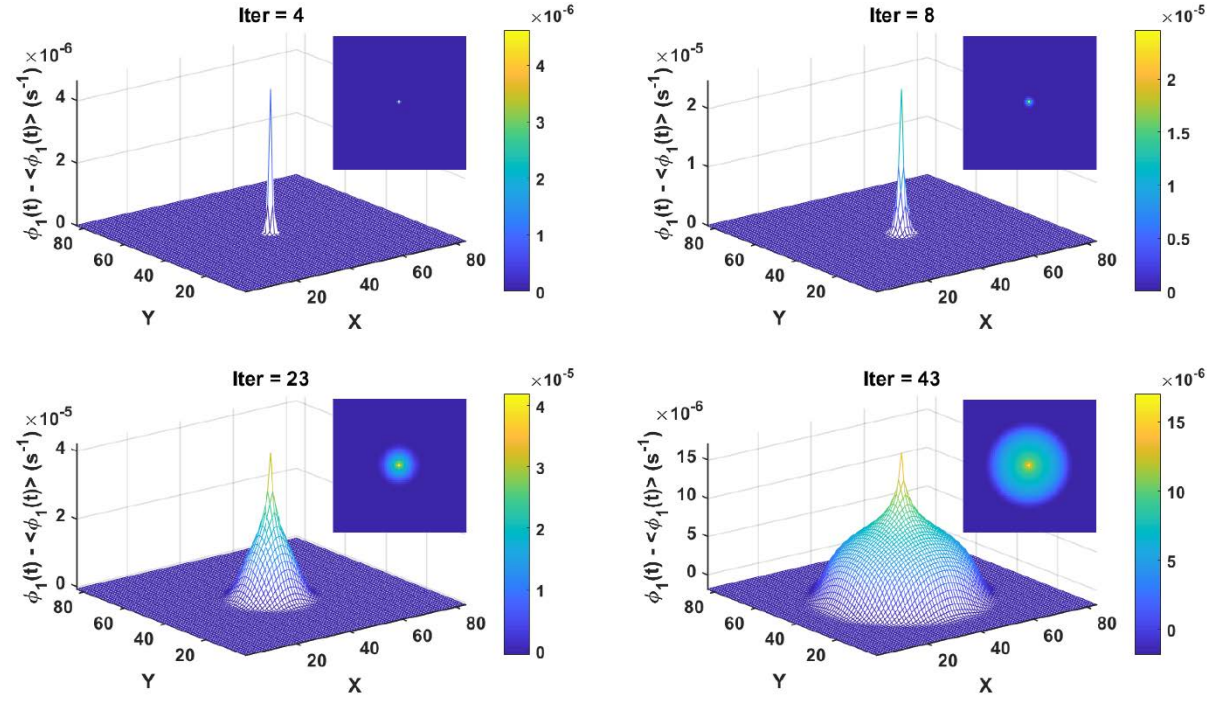


*Figure 7.13 – Neural activity distribution of the One-population model at four different iteration steps in a simulation using FD method. Axonal fields propagate radially outwards. The model was driven by a pulse applied at the centre of a grid. In each timeframe, the mean has been subtracted, so the colour shows deviations from the average amplitude at that iteration step.*

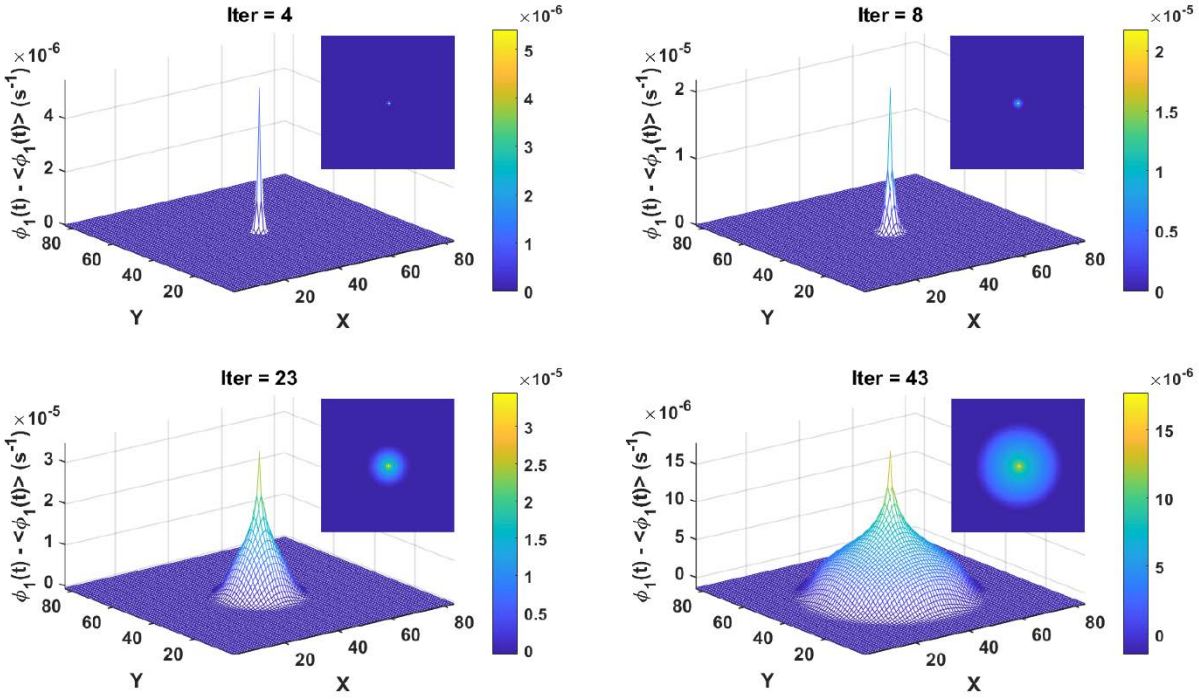


*Figure 7.14 – Neural activity distribution of the One-population model at four different iteration steps in a simulation using TLM method. Axonal fields propagate radially outwards. The model was driven by a pulse applied at the centre of a grid. In each timeframe, the mean has been subtracted, so the colour shows deviations from the average amplitude at that iteration step.*

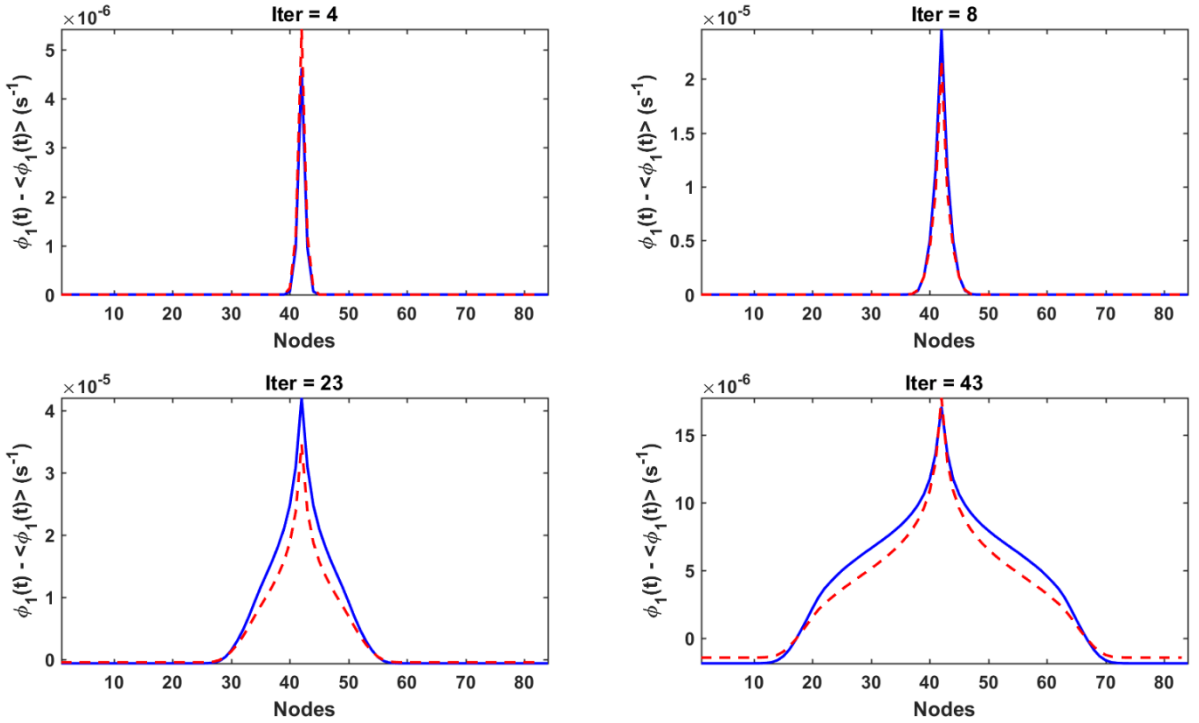


*Figure 7.15 – Equatorial profiles of axonal fields, with the subtracted mean values, of the One-population model at four different iteration steps in the simulations. The model was driven by a pulse applied at the centre of a grid. FD – blue line, TLM – red dashed line.*

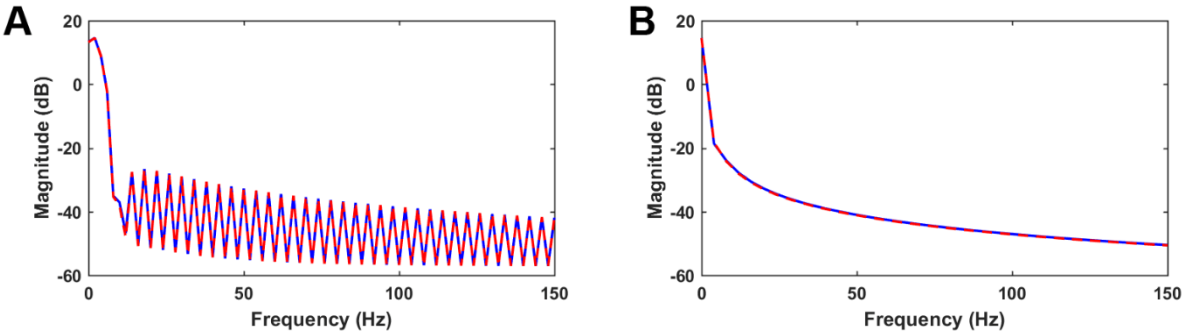


*Figure 7.16 – Power spectrum of the axonal fields in the One-population model with pulse drive. A) Power spectral density estimate of the central trace using the standard MATLAB function pwelch; B) Spatially summed spectrum using NeuroField MATLAB module. FD – blue line, TLM – red dashed line.*

#### *7.2.1.2. Sine wave stimulus*

For this simulation, the model was driven by a sine wave applied in the centre of the grid of amplitude $1s^{-1}$ and two different frequencies: 12Hz and 40Hz, corresponding to frequencies of alpha and gamma brain waves, respectively.

Time-series of the central nodes for both methods were inspected and their traces are showed in Figure 7.17. Neural activity distributions are shown in Figure 7.18 for FD and Figure 7.19 for TLM methods. Equatorial profiles of axonal fields are compared in Figure 7.20, and their power spectrums were compared in Figure 7.21.

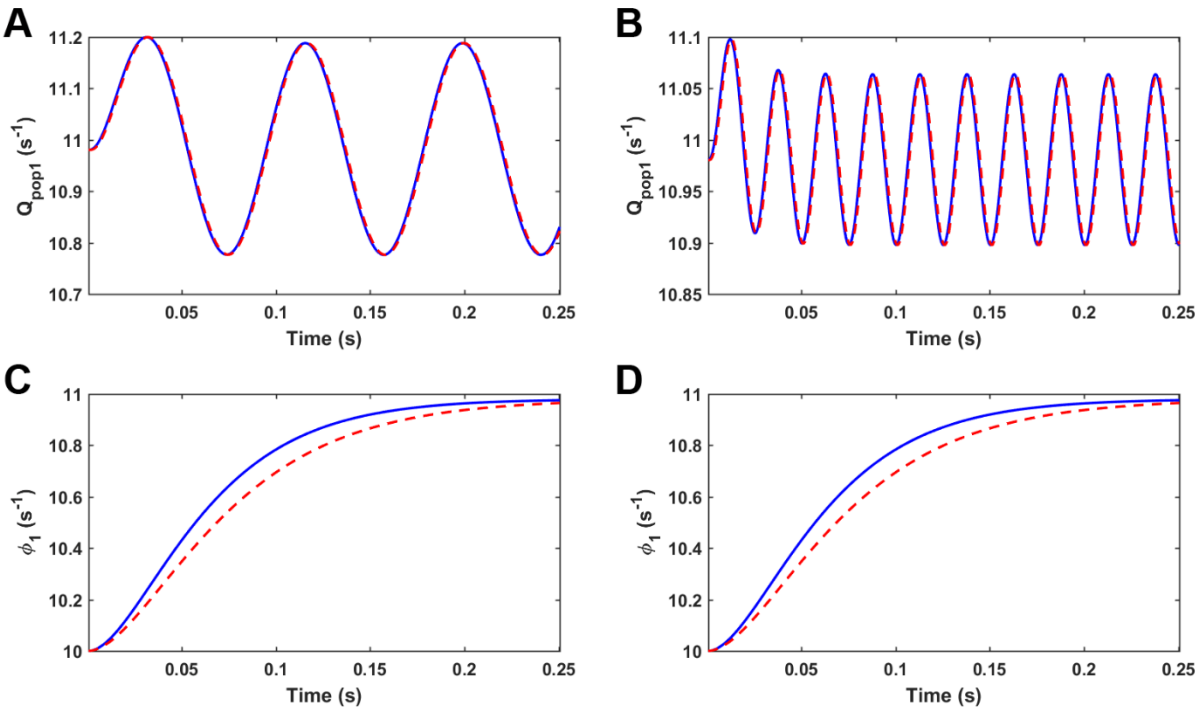


*Figure 7.17 – Central node traces for One-population NeuroField model driven by a sine wave of amplitude $1s^{-1}$ with two different frequencies: 12Hz left figures, 40Hz right figures. A) and B) Mean firing rates for populations 1, $Q_1$ , C) and D) propagation fields $\varphi_1$. FD – blue line, TLM – red dashed line.*

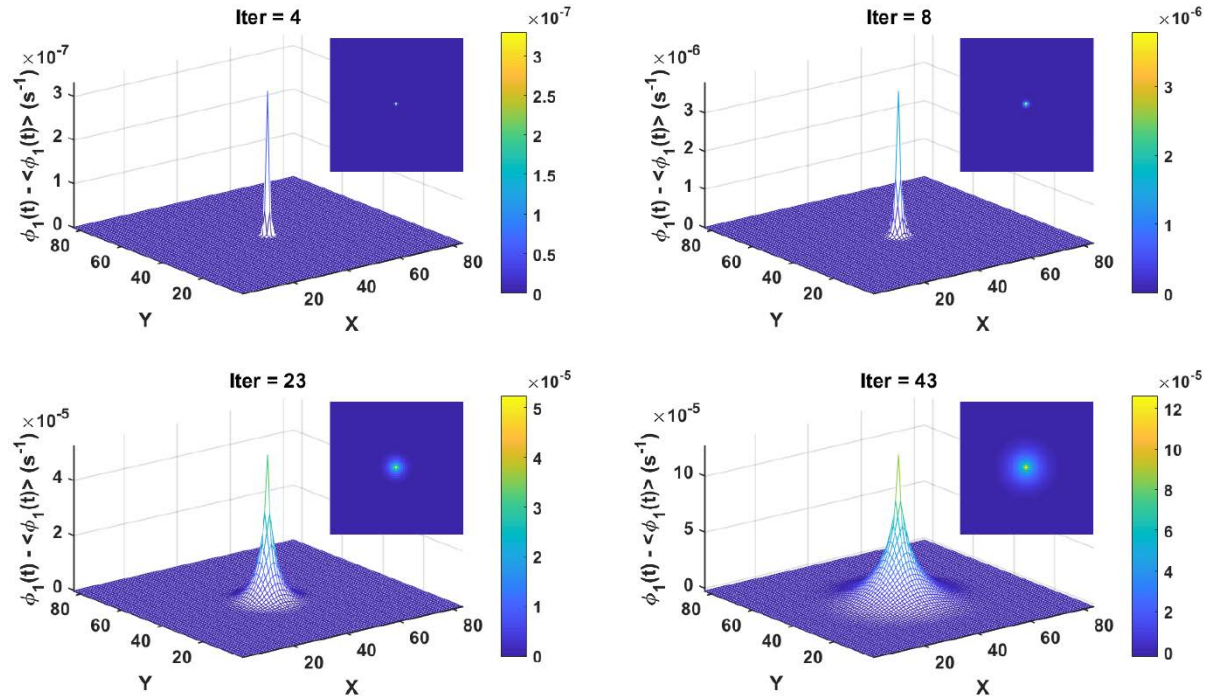


*Figure 7.18 – Neural activity distribution of the One-population model at four different iteration steps in a simulation using FD method. Axonal fields propagate radially outwards. The model was driven by a sine wave of amplitude $1s^{-1}$ and frequency of 12Hz applied at the centre of a grid. In each timeframe, the mean has been subtracted, so the colour shows deviations from the average amplitude at that iteration step.*

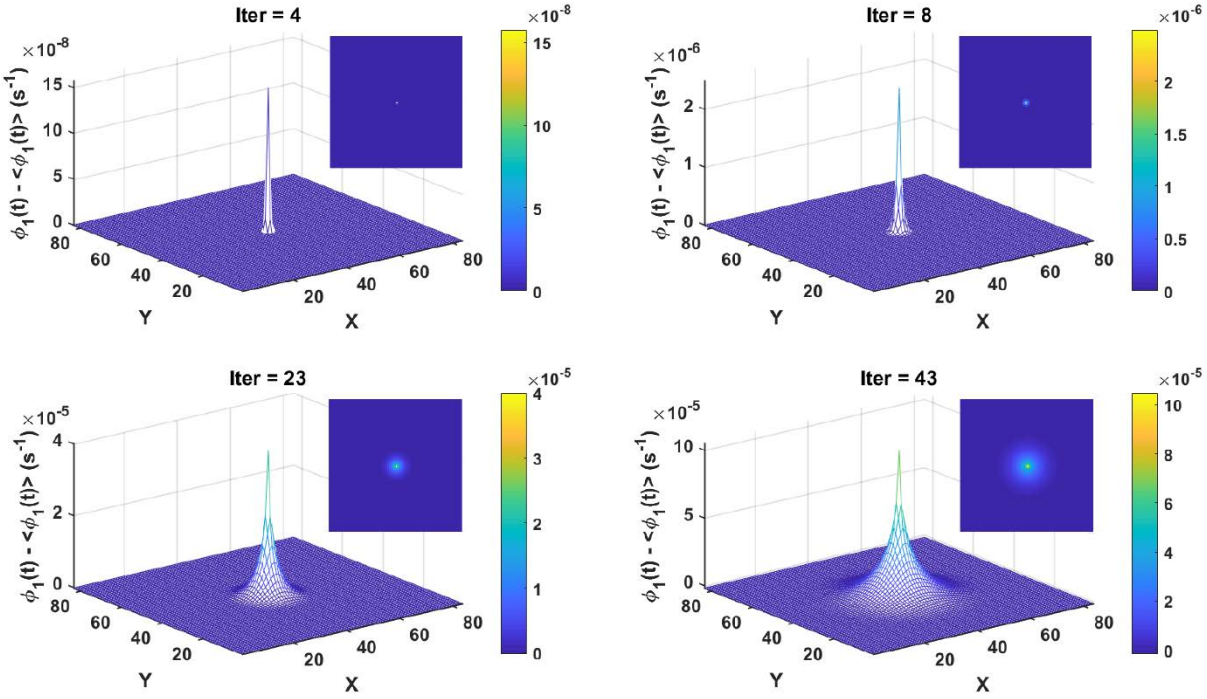


*Figure 7.19 – Neural activity distribution of the One-population model at four different iteration steps in a simulation using TLM method. Axonal fields propagate radially outwards. The model was driven by a sine wave of amplitude $1s^{-1}$ and frequency of 12Hz applied at the centre of a grid. In each timeframe, the mean has been subtracted, so the colour shows deviations from the average amplitude at that iteration step.*

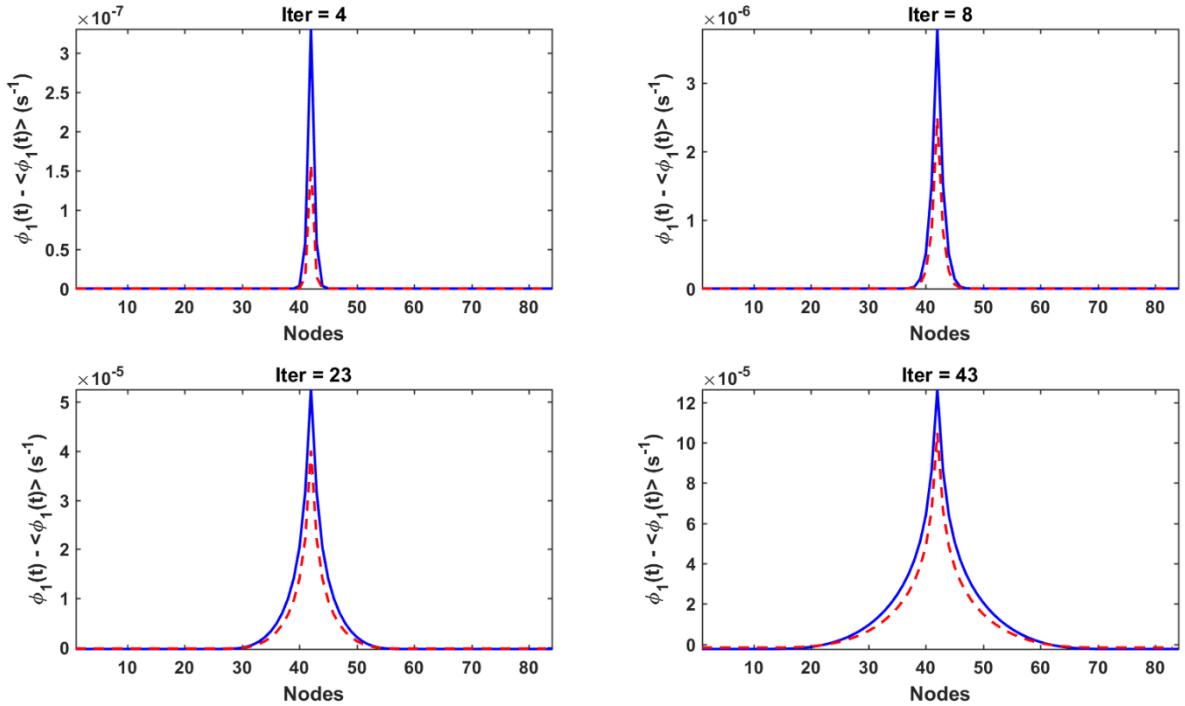


*Figure 7.20 – Equatorial profiles of axonal fields, with the subtracted mean values, of the One-population model at four different iteration steps in the simulations. The model was driven by a sine wave of amplitude $1s^{-1}$ and frequency of 12Hz applied at the centre of a grid. FD – blue line, TLM – red dashed line.*

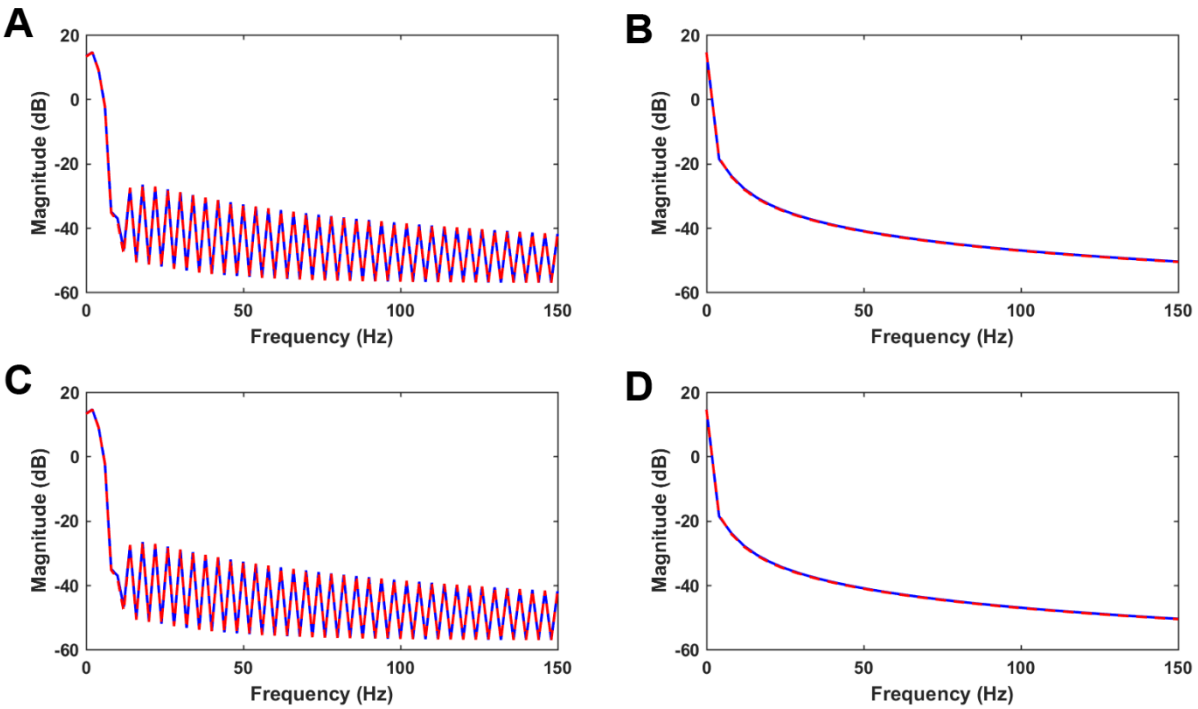


*Figure 7.21 – Power spectrum of the axonal fields in the One-population model with sine wave drive of amplitude $1s^{-1}$ and two different frequencies: 12Hz top figures, 40Hz bottom figures. A) and C) Power spectral density estimates of the central traces using the standard MATLAB function pwelch; B) and D) Spatially summed spectrum using NeuroField MATLAB module. FD – blue line, TLM – red dashed line.*

#### *7.2.1.3. Gaussian stimulus*

In order to excite the mesh with the broad frequency range, but to avoid the high frequencies, for this simulation the model was driven by a Gaussian pulse applied in the centre of the grid, with the amplitude of $1s^{-1}$ and two different variances $\sigma^2$ depending on frequencies: 12Hz and 40Hz. Variances were calculated as: $\sigma^2 = \frac{f^2}{8}$, where $f$ is the frequency of interest.

Time-series of the central nodes for both methods were inspected and their traces are showed in Figure 7.22. Neural activity distributions are shown in Figure 7.23 for FD and Figure 7.24 for TLM methods. Equatorial profiles of axonal fields are compared in Figure 7.25, and their power spectrums were compared in Figure 7.26.

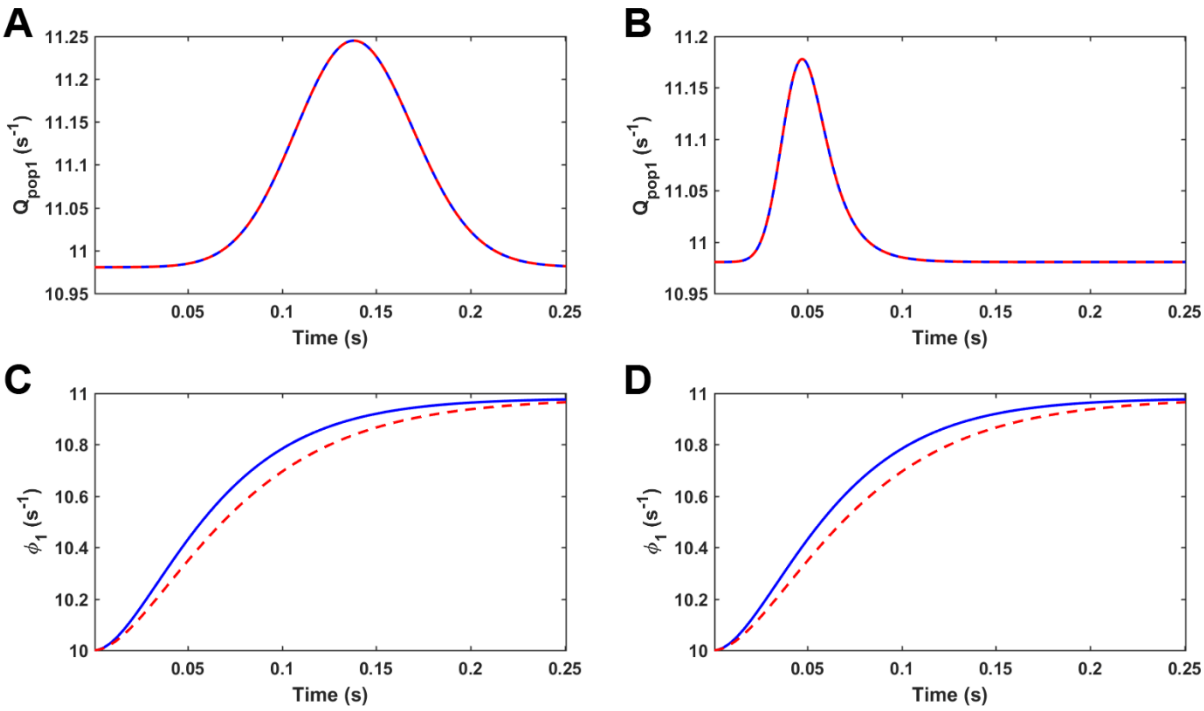


*Figure 7.22 – Central node traces for One-population NeuroField model driven by a Gaussian wave of amplitude $1s^{-1}$ and two different variances depending on frequencies: 12Hz left figures, 40Hz right figures. A) and B) Mean firing rates for populations 1, $Q_1$ , C) and D) propagation fields $\varphi_1$. FD – blue line, TLM – red dashed line.*

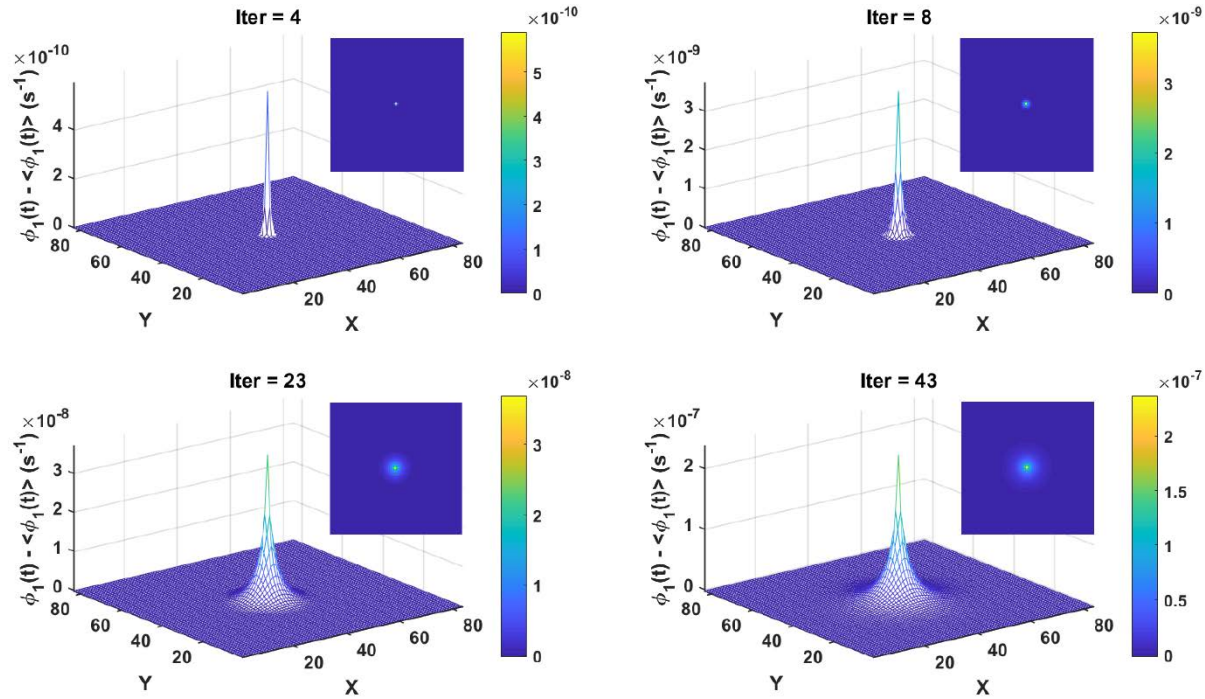


*Figure 7.23 – Neural activity distribution of the One-population model at four different iteration steps in a simulation using FD method. Axonal fields propagate radially outwards. The model was driven by a Gaussian wave of amplitude $1s^{-1}$ and the variance for 12Hz applied at the centre of a grid. In each timeframe, the mean has been subtracted, so the colour shows deviations from the average amplitude at that iteration step.*

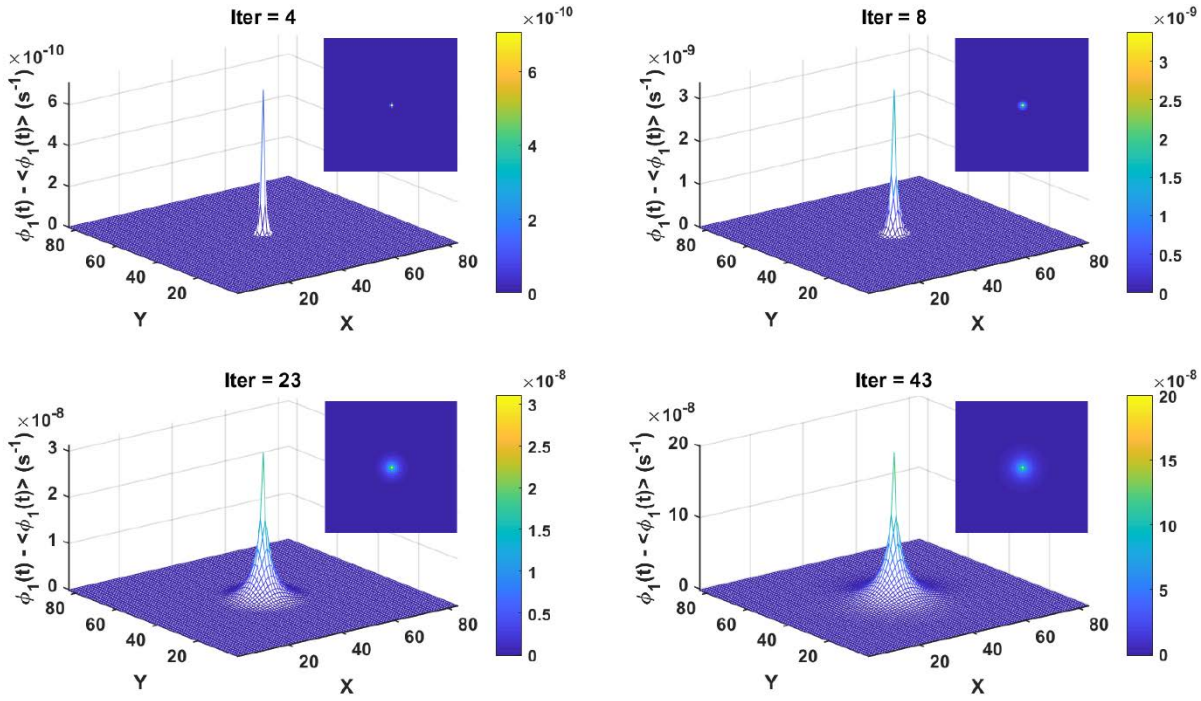


*Figure 7.24 – Neural activity distribution of the One-population model at four different iteration steps in a simulation using TLM method. Axonal fields propagate radially outwards. The model was driven by a sine wave of amplitude $1s^{-1}$ and frequency of 12Hz applied at the centre of a grid. In each timeframe, the mean has been subtracted, so the colour shows deviations from the average amplitude at that iteration step.*

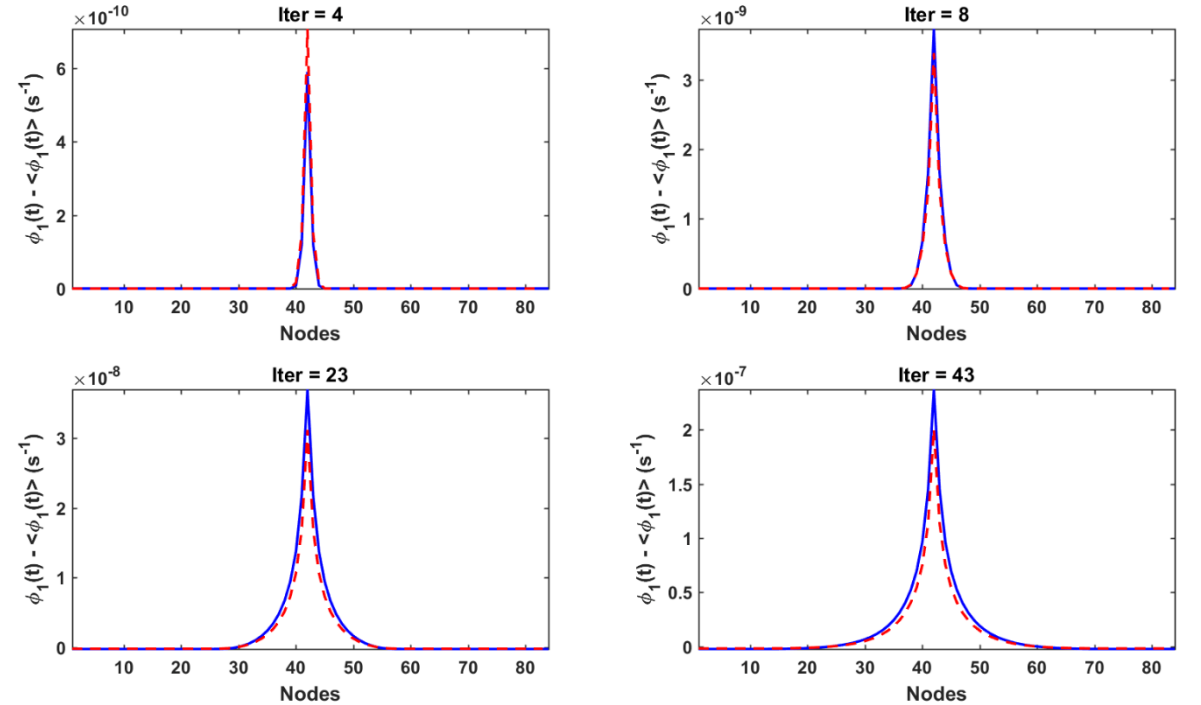


*Figure 7.25 – Equatorial profiles of axonal fields, with the subtracted mean values, of the One-population model at four different iteration steps in the simulations. The model was driven by a Gaussian wave of amplitude $1s^{-1}$ and the variance for 12Hz applied at the centre of a grid. FD – blue line, TLM – red dashed line.*

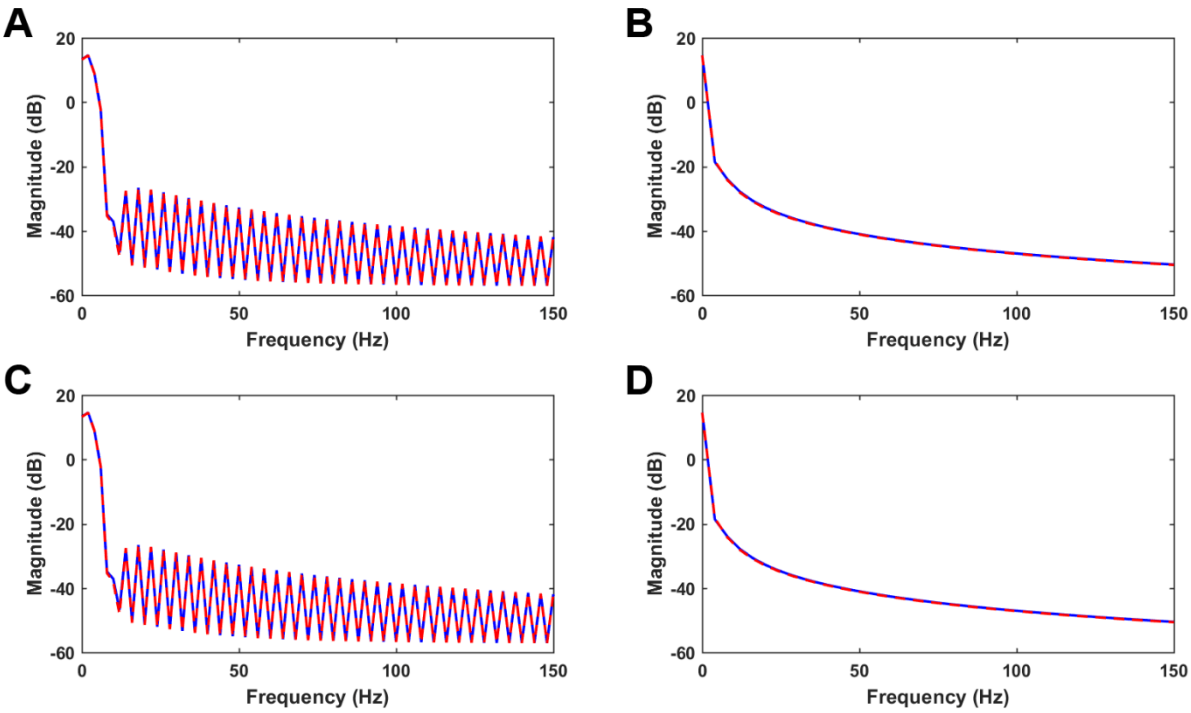


*Figure 7.26 – Power spectrum of the axonal fields in the One-population model with Gaussian wave of amplitude $1s^{-1}$ and two different variances depending on frequencies: 12Hz top figures, 40Hz bottom figures. A) and C) Power spectral density estimates of the central traces using the standard MATLAB function pwelch; B) and D) Spatially summed spectrum using NeuroField MATLAB module. FD – blue line, TLM – red dashed line.*

#### *7.2.1.4. Effects of changing the temporal damping coefficient*

To investigate the effects of changing temporal damping coefficient, $\gamma_{ab}$, on NeuroField models, we have used several values for $\gamma_{ab}$, that fall into the range showed in (Robinson et al., 2004b). Figure 7.27 and Figure 7.28 are central node traces and equatorial profiles for propagation fields $\varphi_1$, respectively, for three values: 30, 60 and 120 $\left[s^{-1}\right]$. In these simulations, we kept the wave speed constant, thus changing the axonal range, $r_{ab}$ ,parameter accordingly. The models were driven by the sine wave of amplitude $1s^{-1}$ and frequency of 20Hz applied at the centre of the grid.

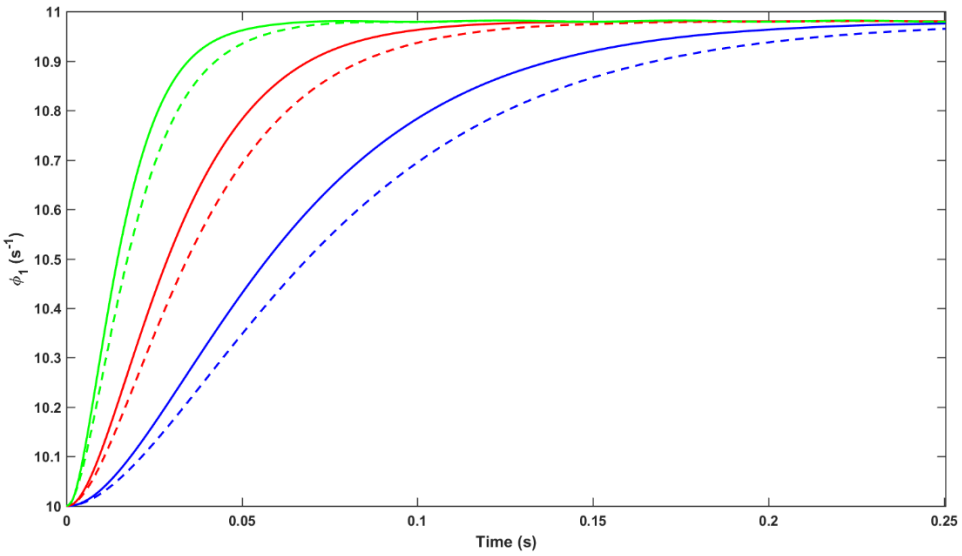


*Figure 7.27 – Central node traces for One-population NeuroField model driven by a sine wave of amplitude $1s^{-1}$ and frequency of 20Hz. $\gamma_{ab} = 30s^{-1}$ – blue lines; $\gamma_{ab} = 60s^{-1}$ – red lines; $\gamma_{ab} = 120s^{-1}$ – green lines. FD – solid lines, TLM – dashed lines.*

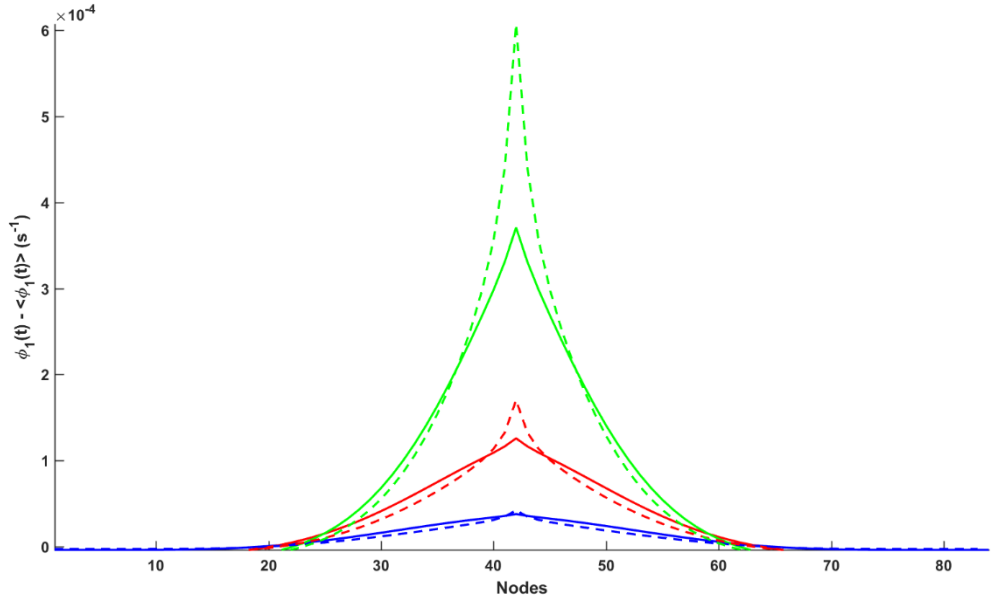


*Figure 7.28 – Equatorial profiles of axonal fields, with the subtracted mean values, of the One-population model at iteration step 49. The model was driven by a sine wave of amplitude $1s^{-1}$ and frequency of 20Hz. $\gamma_{ab} = 30s^{-1}$ – blue lines; $\gamma_{ab} = 60s^{-1}$ – red lines; $\gamma_{ab} = 120s^{-1}$ – green lines. FD – solid lines, TLM – dashed lines.*

### 7.2.2. Two-populations NeuroField model

A Two-population NeuroField model is the reduced Corticothalamic model from (Robinson et al., 2004b). It consists of three populations: excitatory cortical $Q_1$, relay nuclei $Q_2$ and stimulus $Q_3$. For illustration purposes, axonal propagation from relay nuclei to cortical population is also inhomogeneous damped wave, instead of "1-to-1" mapping used in the original corticothalamic model, Figure 7.29.

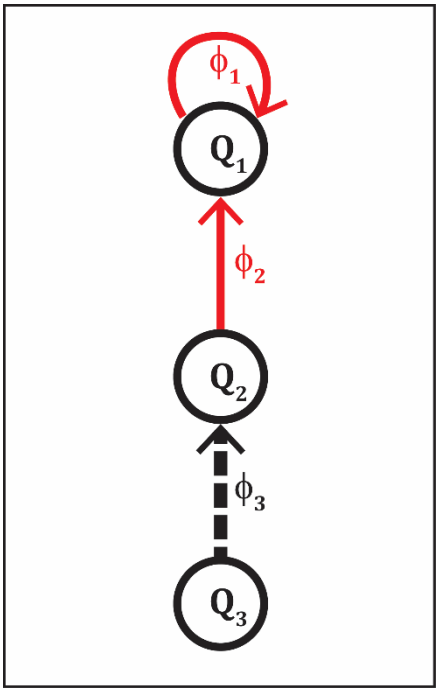


*Figure 7.29 – Block diagram of Two-populations NeuroField model. Red arrow indicates inhomogeneous damped wave propagation; thick, dashed black arrow indicates stimulus propagation.*

Parameters for this model were taken from (Robinson et al., 2004b). The initial steady state low firing rates for populations 1 (excitatory cortical) and 2 (relay nuclei) are $Q_1^0$ and $Q_2^0$ respectively. In contrast with One-population model, axonal propagation parameters are not changed here. The rest of the parameters used in these simulations are shown in Table 7-5.

*Table 7-5 – Simulation parameters for Two-population NeuroField model*

| | | | |
|---|---|---|---|
| $v = \beta [m/s]$ | 9.9760 | $Q_1^0 \left[ s^{-1} \right]$ | 5.2484 |
| $N_x = N_y$ | 51 | $Q_2^0 \left[ s^{-1} \right]$ | 8.7897 |
| $sim\_time[s]$ | 0.15 | $\upsilon_1 [Vs]$ | $1.525 \cdot 10^{-3}$ |
| *Steps* | 360 | $\upsilon_2 [Vs]$ | $5.675 \cdot 10^{-4}$ |
| | | $\upsilon_3 [Vs]$ | $3.593 \cdot 10^{-3}$ |

These models were driven by the sine wave stimulus of amplitude $1s^{-1}$ and frequency of 20Hz applied in the centre of the grid.

Time-series of the central nodes for both methods were inspected and their traces are showed in Figure 7.30. Neural activity distributions are shown in Figure 7.31 for FD and Figure 7.32 for TLM methods. Equatorial profiles of axonal fields are compared in Figure 7.33, and their power spectrums were compared in Figure 7.34.

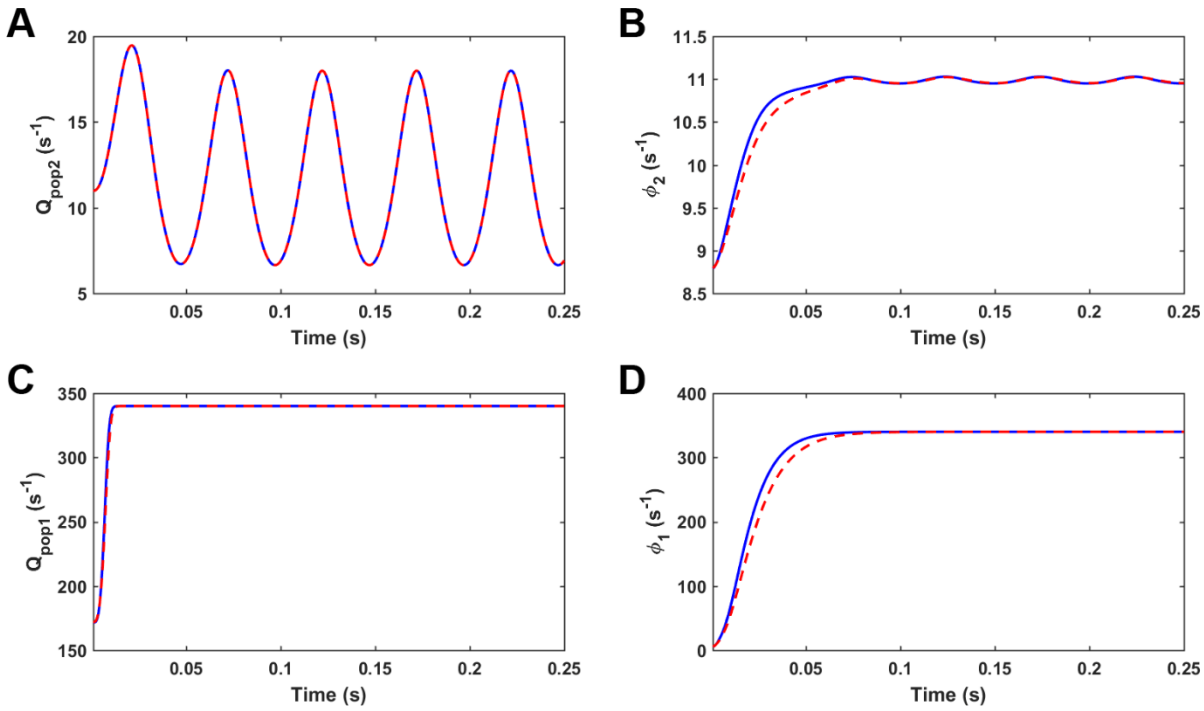


*Figure 7.30 – Central node traces for Two-population NeuroField model driven by a sine wave of amplitude $1s^{-1}$ and frequency of 20Hz. A) and C) Mean firing rates for populations 2 and 1, $Q_2$ and $Q_1$ respectively, B) and D) propagation fields $\varphi_2$ and $\varphi_1$ ,respectively, driven by the corresponding mean firing rates. FD – blue line, TLM – red dashed line.*

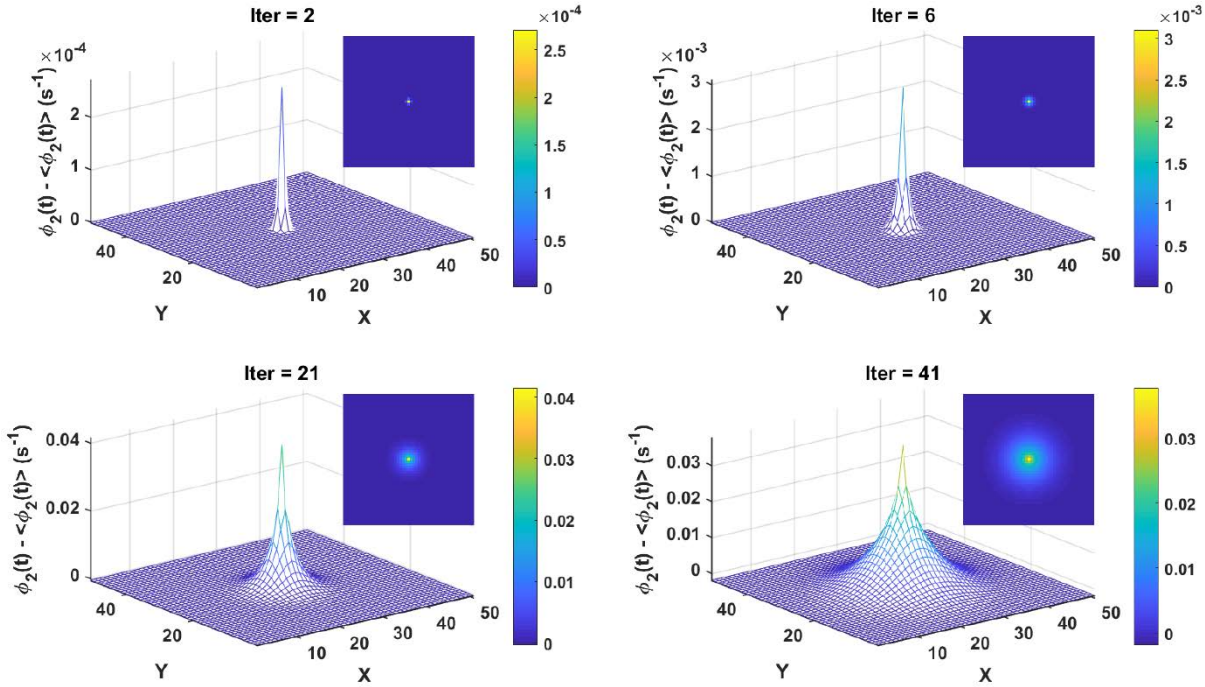


*Figure 7.31 – Distribution of the propagation field $\varphi_2$ from the Two-population model at four different iteration steps in a simulation using FD method. Axonal fields propagate radially outwards. The model was driven by a sine wave of amplitude$1s^{-1}$ and frequency of 20Hz applied at the centre of a grid. In each timeframe, the mean has been subtracted, so the colour shows deviations from the average amplitude at that iteration step.*

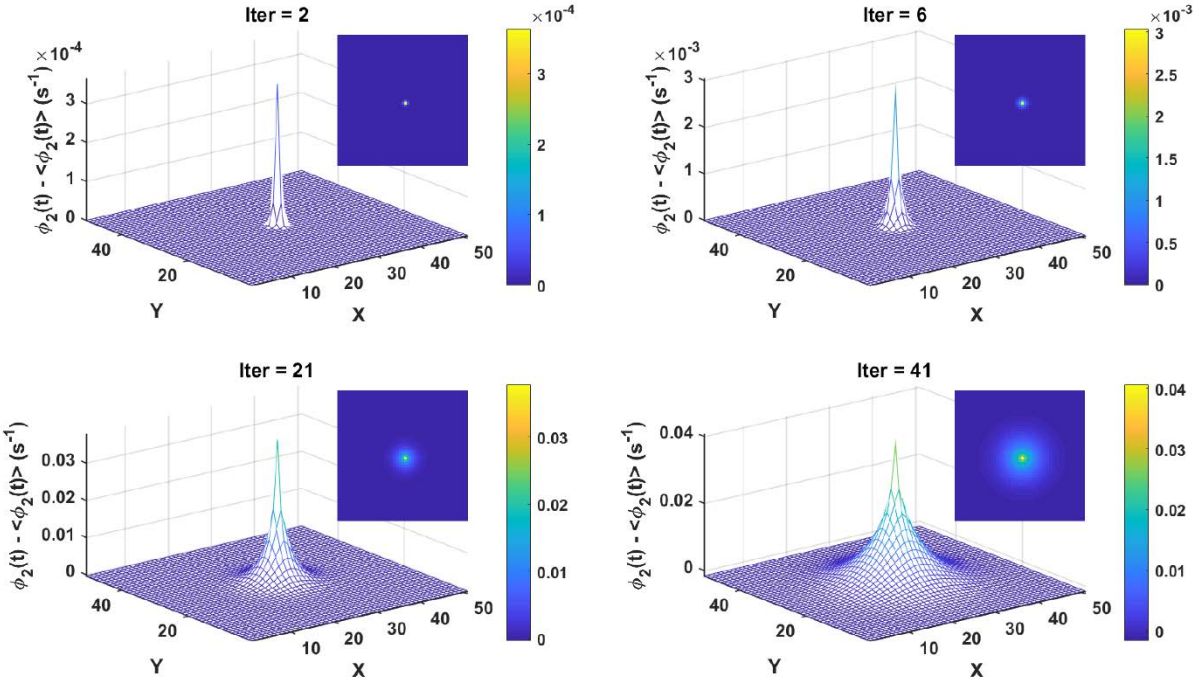


*Figure 7.32 – Distribution of the propagation field $\varphi_2$ from the Two-population model at four different iteration steps in a simulation using TLM method. Axonal fields propagate radially outwards. The model was driven by a sine wave of amplitude$1s^{-1}$ and frequency of 20Hz applied at the centre of a grid. In each timeframe, the mean has been subtracted, so the colour shows deviations from the average amplitude at that iteration step.*

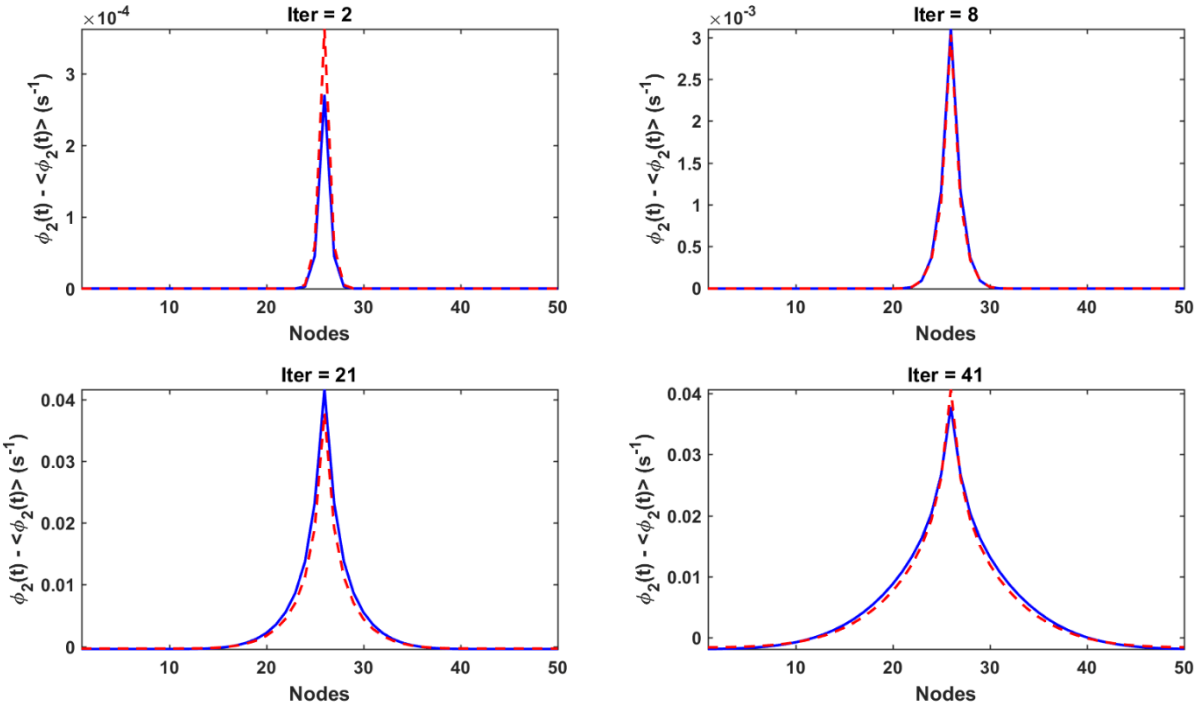


*Figure 7.33 – Equatorial profiles of axonal fields, $\varphi_2$ , with the subtracted mean values, of the Two-population model at four different iteration steps in the simulations. The model was driven by a sine wave of amplitude $1s^{-1}$ and frequency of 20Hz applied at the centre of a grid. FD – blue line, TLM – red dashed line.*

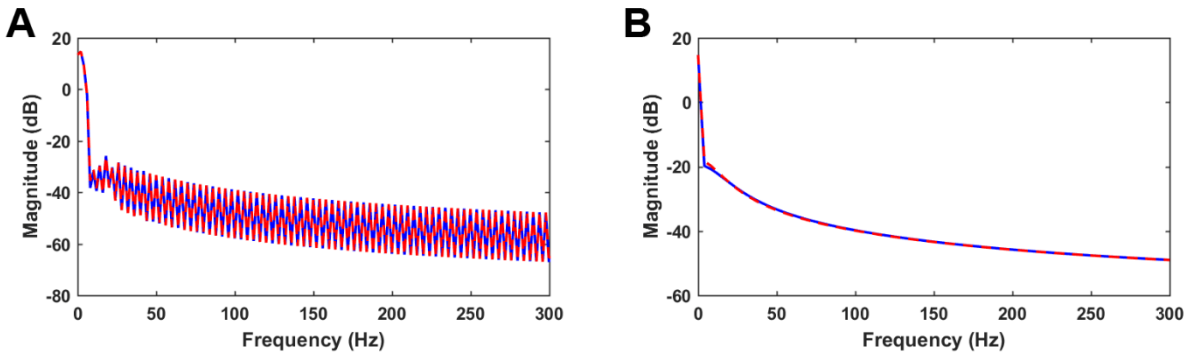


*Figure 7.34 – Power spectrum of $\varphi_2$ axonal fields in the Two-population model with sine wave of amplitude $1s^{-1}$ and frequency of 20Hz. A) Power spectral density estimates of the central traces using the standard MATLAB function pwelch; B) spatially summed spectrum using NeuroField MATLAB module. FD – blue line, TLM – red dashed line.*

### 7.2.3. Four-populations NeuroField model

Four-populations NeuroField model represents the complete Corticothalamic model from (Robinson et al., 2004b). It consists of five populations: excitatory and inhibitory cortical $Q_1$ and $Q_2$, reticular $Q_3$, relay nuclei $Q_4$ and stimulus $Q_5$, Figure 7.35. The main difference between this NeuroField model and Corticothalamic model in Robinson 2004 is that this model was driven by the sine wave stimulus of amplitude $1s^{-1}$ and frequency of 20Hz applied in the centre of the grid instead of the spatiotemporal white noise used to approximate external stimuli of the spontaneous EEG.

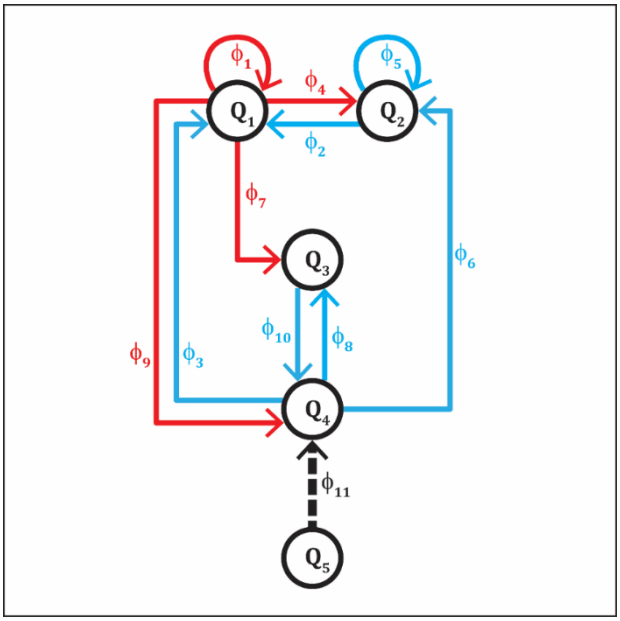


*Figure 7.35 – Block diagram of Four-populations NeuroField model. Red arrows indicate inhomogeneous damped wave propagations; thick, dashed black arrow indicates stimulus propagation; blue arrows are "1-to-1" mapping.*

Parameters for this model were also taken from (Robinson et al., 2004b). The initial steady state low firing rates for populations are denoted with the superscript 0 in the Table 7-6. Axonal propagation parameters are the same as for Corticothalamic model. In this model, some connections have included long range time delay $\tau = 10dt$. The rest of the parameters used in these simulations are the same as for Two-populations model, like the wave speed, number of cells in the grid, thus equal $dx$ as well, simulation time and number of steps, which gives the same $dt$ too.

*Table 7-6 – Simulation parameters for Four-population NeuroField model*

| | | | |
|---|---|---|---|
| $Q_1^0\left[s^{-1}\right]$ | 5.2484 | $\upsilon_4[Vs]$ | $1.525\cdot10^{-3}$ |
| $Q_2^0\left[s^{-1}\right]$ | 5.2484 | $\upsilon_5[Vs]$ | $-3.023\cdot10^{-3}$ |
| $Q_3^0\left[s^{-1}\right]$ | 15.3960 | $\upsilon_6[Vs]$ | $5.675\cdot10^{-4}$ |
| $Q_4^0\left[s^{-1}\right]$ | 8.7897 | $\upsilon_7[Vs]$ | $1.696\cdot10^{-4}$ |
| | | $\upsilon_8[Vs]$ | $5.070\cdot10^{-5}$ |
| $\upsilon_1[Vs]$ | $1.525\cdot10^{-3}$ | $\upsilon_9[Vs]$ | $3.447\cdot10^{-3}$ |
| $\upsilon_2[Vs]$ | $-3.023\cdot10^{-3}$ | $\upsilon_{10}[Vs]$ | $-1.465\cdot10^{-3}$ |
| $\upsilon_3[Vs]$ | $5.675\cdot10^{-4}$ | $\upsilon_{11}[Vs]$ | $3.593\cdot10^{-3}$ |

Time-series of the central nodes for both methods were inspected and their traces are showed in Figure 7.36 and Figure 7.37. Neural activity distributions are shown in Figure 7.38 for FD and Figure 7.39 for TLM methods. Equatorial profiles of axonal fields are compared in Figure 7.40, and their power spectrums were compared in Figure 7.41.

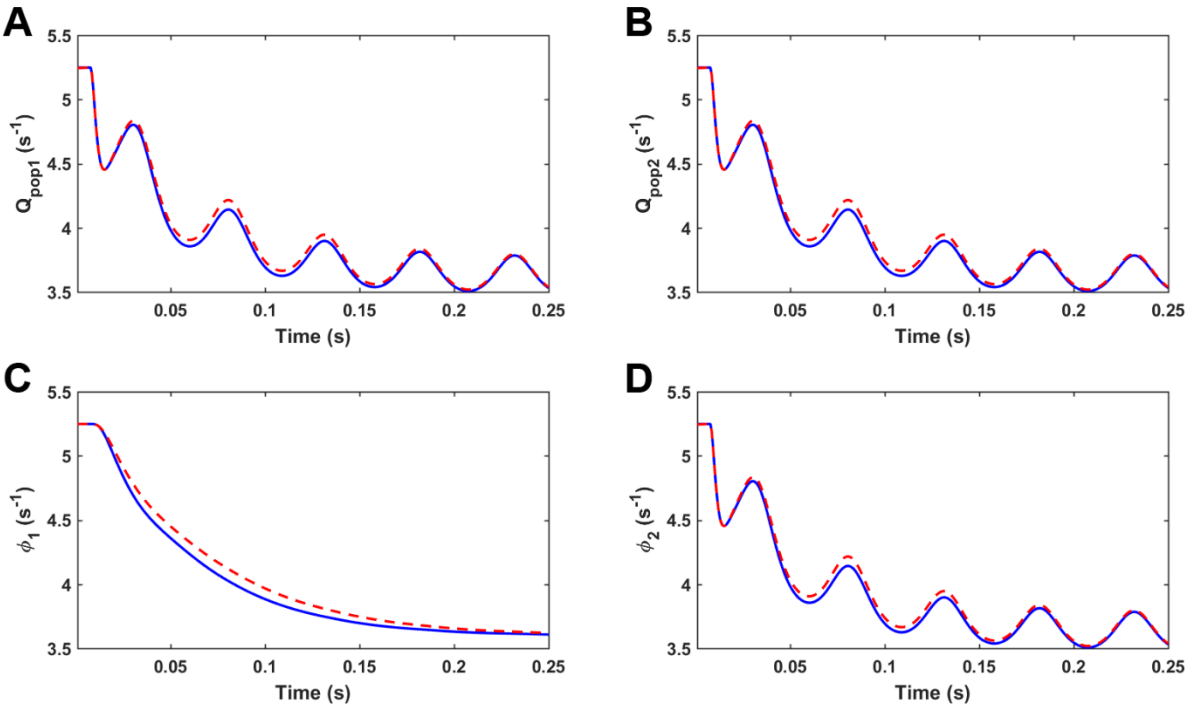


*Figure 7.36 – Central node traces for Four-population NeuroField model driven by a sine wave of amplitude $1s^{-1}$ and frequency of 20Hz. A) and B) Mean firing rates for populations 1 and 2, $Q_1$ and $Q_2$ respectively, C) and D) propagation fields $\varphi_1$ and $\varphi_2$ ,respectively, driven by the corresponding mean firing rates shown above them. FD – blue line, TLM – red dashed line.*

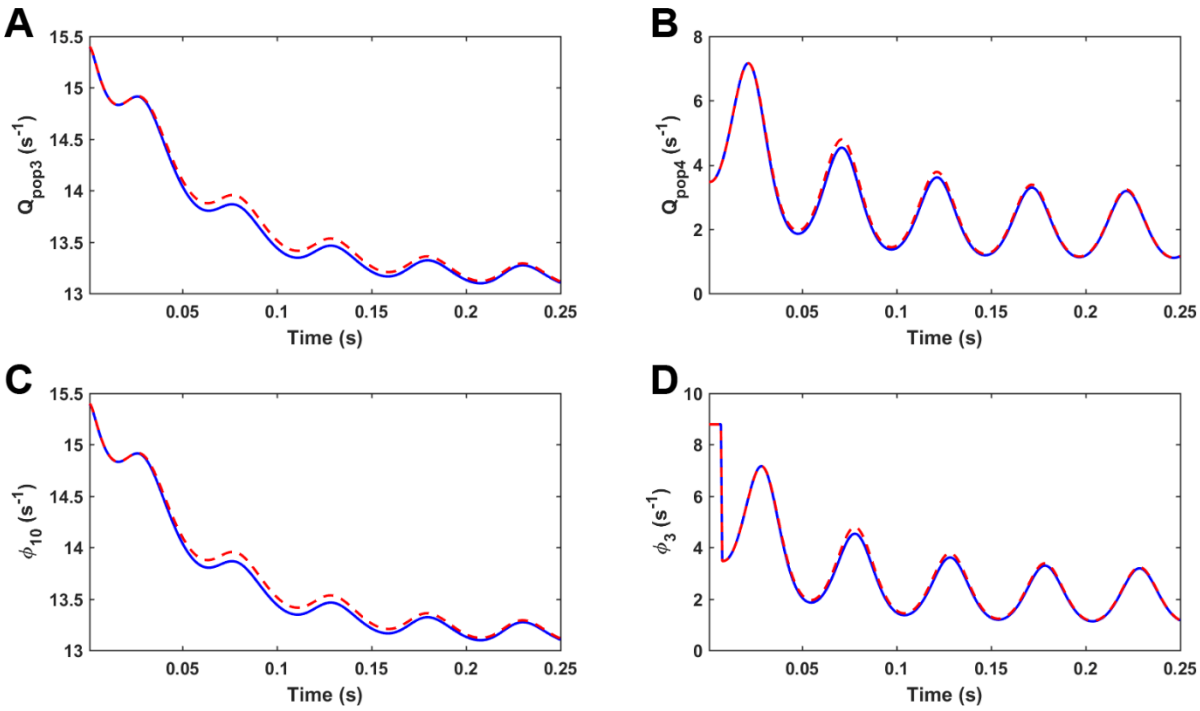


*Figure 7.37 – Central node traces for Four-population NeuroField model driven by a sine wave of amplitude $1s^{-1}$ and frequency of 20Hz. A) and B) Mean firing rates for populations 3 and 4, $Q_3$ and $Q_4$ respectively, C) and D) propagation fields $\varphi_{10}$ and $\varphi_3$ ,respectively, driven by the corresponding mean firing rates shown above them. FD – blue line, TLM – red dashed line.*

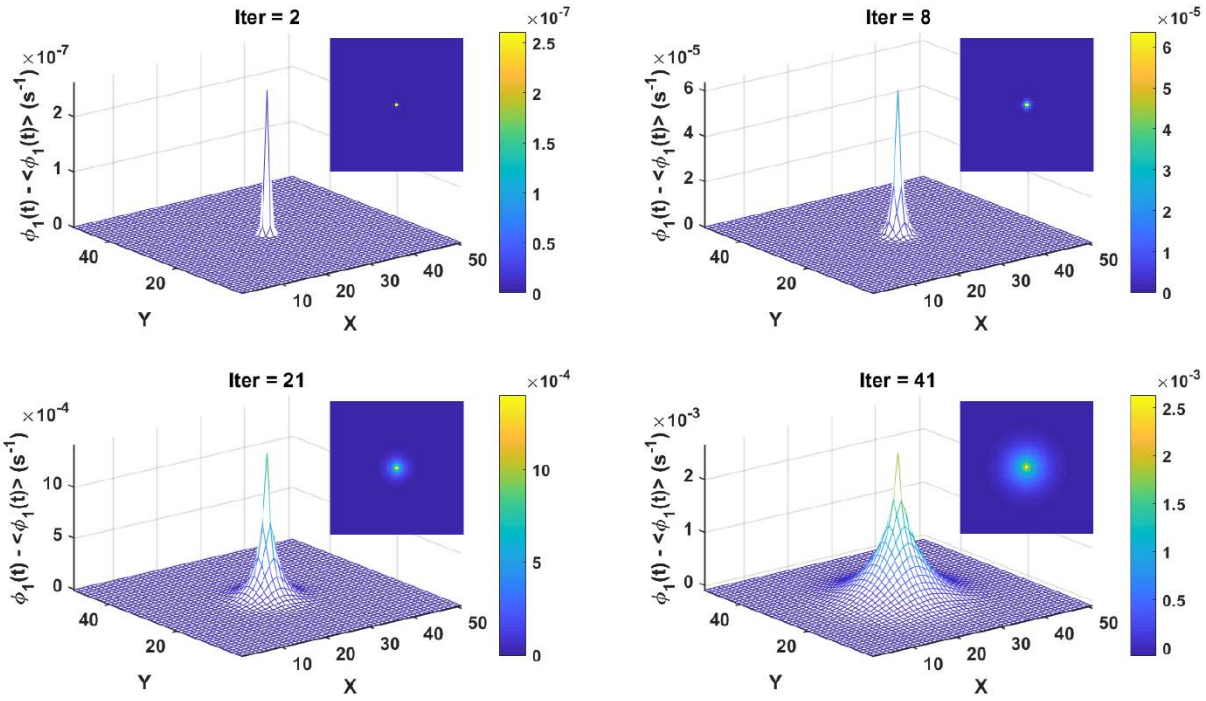


*Figure 7.38 – Distribution of the propagation field $\varphi_1$ from the Four-population model at four different iteration steps in a simulation using FD method. Axonal fields propagate radially outwards. The model was driven by a sine wave of amplitude $1s^{-1}$ and frequency of 20Hz applied at the centre of a grid. In each timeframe, the mean has been subtracted, so the colour shows deviations from the average amplitude at that iteration step.*

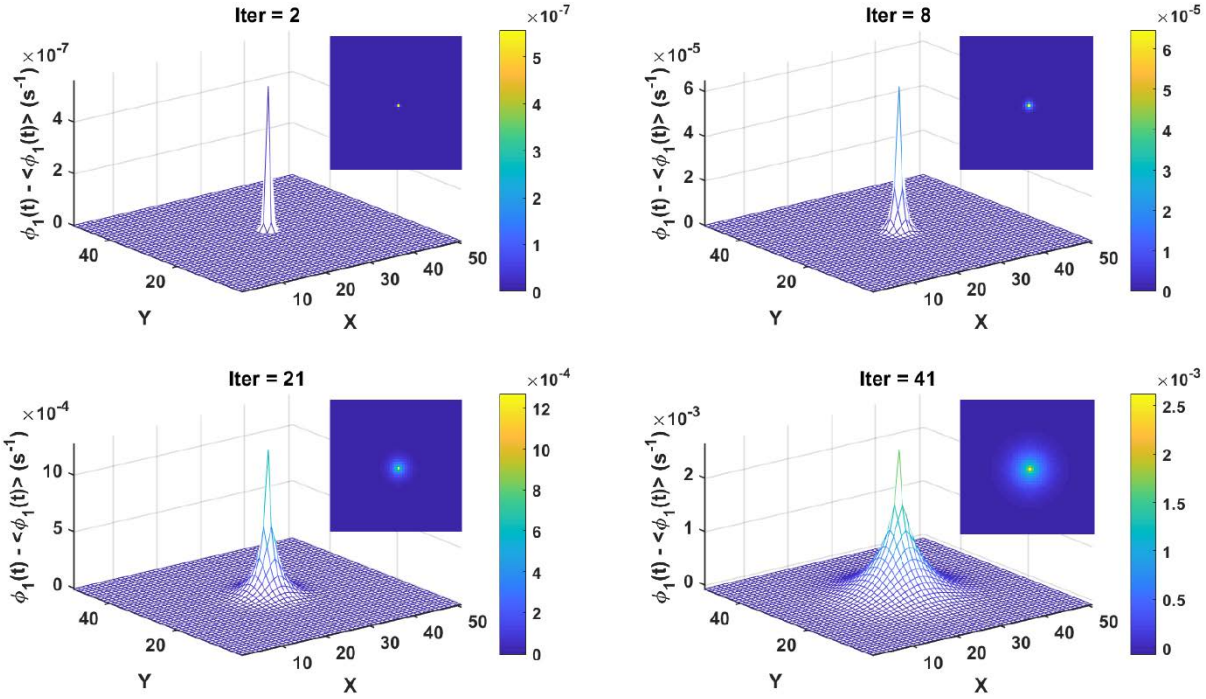


*Figure 7.39 – Distribution of the propagation field $\varphi_1$ from the Four-population model at four different iteration steps in a simulation using TLM method. Axonal fields propagate radially outwards. The model was driven by a sine wave of amplitude $1s^{-1}$ and frequency of 20Hz applied at the centre of a grid. In each timeframe, the mean has been subtracted, so the colour shows deviations from the average amplitude at that iteration step.*

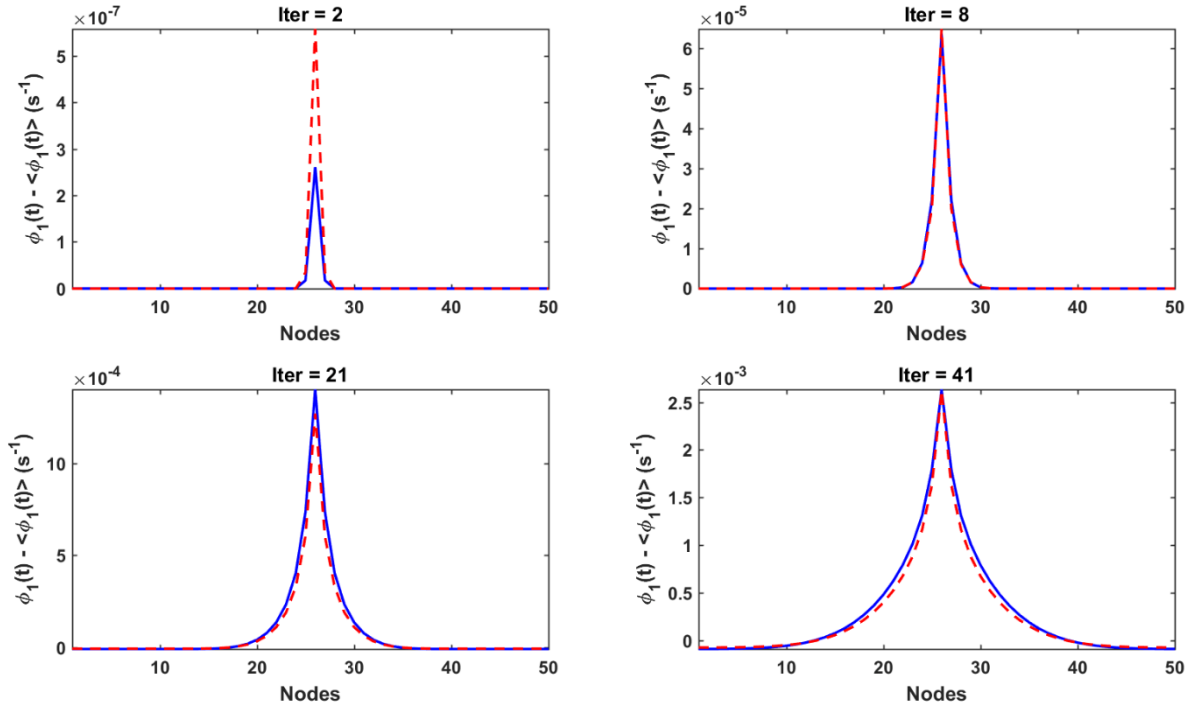


*Figure 7.40 – Equatorial profiles of axonal fields, $\varphi_1$, with the subtracted mean values, of the Four-population model at four different iteration steps in the simulations. The model was driven by a sine wave of amplitude $1s^{-1}$ and frequency of 20Hz applied at the centre of a grid. FD – blue line, TLM – red dashed line.*

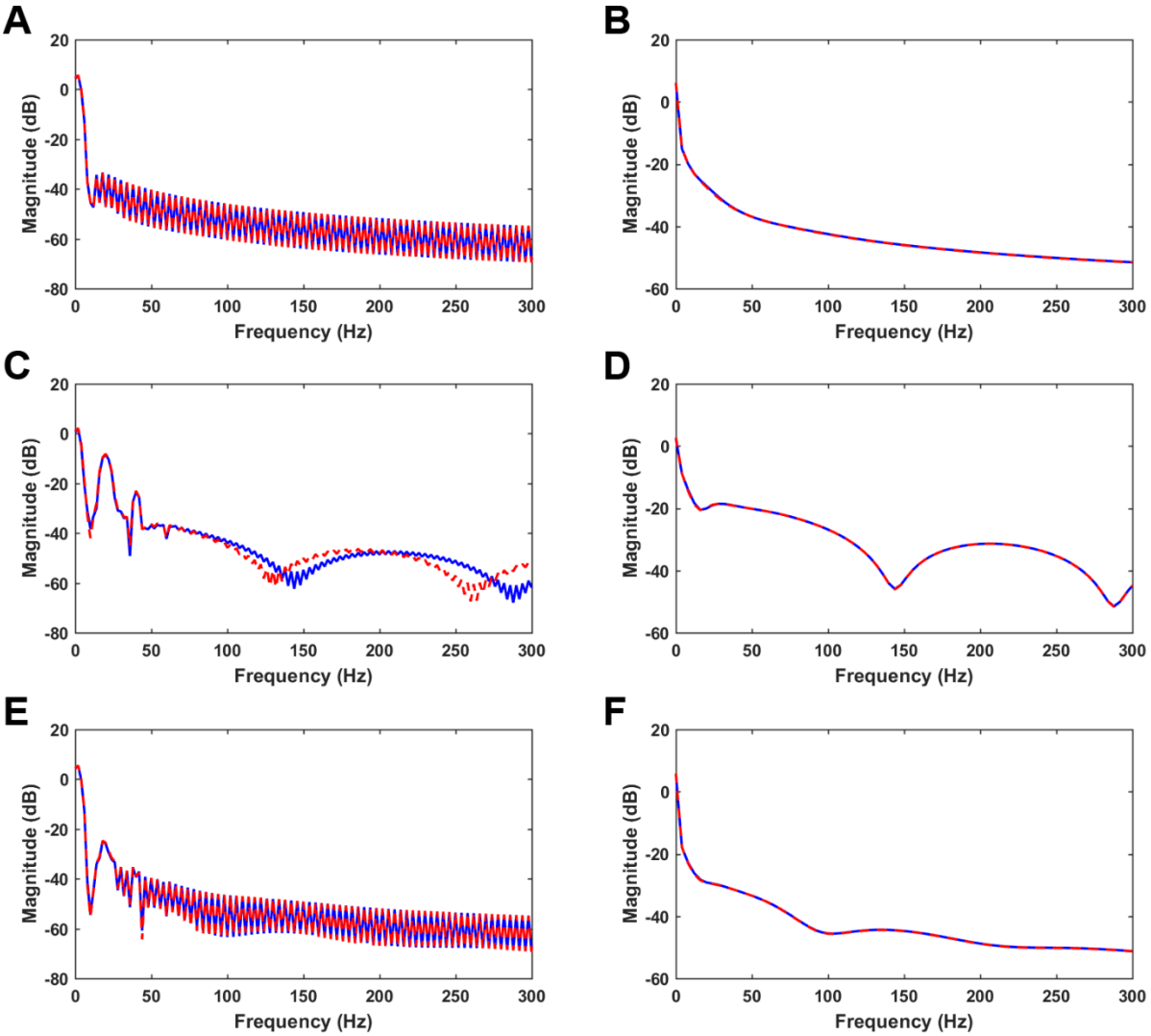


*Figure 7.41 – Power spectrum of some axonal fields in the Four-population model with sine wave of amplitude $1s^{-1}$ and frequency of 20Hz. Top pair corresponds to $\varphi_1$ , middle to $\varphi_3$ and bottom pair to $\varphi_5$ . A), C) and E) are Power spectral density estimates of the central traces using the standard MATLAB function pwelch; B), D) and F) are spatially summed spectrums using NeuroField MATLAB module. FD – blue line, TLM – red dashed line.*

### 7.2.4. Discussion of the results

In this Section, we compared two numerical methods for solving the governing inhomogeneous damped wave differential equation in NeuroField, FD and TLM. NeuroField models with three levels of complexity (Figure 7.11, Figure 7.29 and Figure 7.35) were simulated for both methods. It is noticeable from inspecting the figures in Section 7.2 for all three NeuroField models that there are some slight differences between FD and TLM methods when the axonal propagation fields are compared (Table 7-7). In One-

population models, mean firing rates are the same, because they don't depend on damped wave propagation fields. However, in other models, some differences can be observed (Table 7-7).

*Table 7-7 – Nash-Sutcliffe Efficiency Index (NSEI) for the central node traces of mean firing rates and axonal propagation fields showing how similar they are between FD and TLM methods for all three NeuroField models presented in Figures 7.12 to 7.37.*

| | | | **NSEI** |
|---|---|---|---|
| **One-population model** | Figure 7.12 | $Q_1$ | 1 |
| | | $\varphi_1$ | 0.9616 |
| | Figure 7.17 | $Q_1$ | 1 |
| | | $\varphi_1$ | 0.9974 |
| | Figure 7.22 | $Q_1$ | 1 |
| | | $\varphi_1$ | 0.9616 |
| **Two-populations model** | Figure 7.30 | $Q_1$ | 0.9965 |
| | | $Q_2$ | 1 |
| | | $\varphi_1$ | 0.9786 |
| | | $\varphi_2$ | 0.9743 |
| **Four-populations model** | Figure 7.36 | $Q_1$ | 0.9924 |
| | | $Q_2$ | 0.9924 |
| | | $\varphi_1$ | 0.9834 |
| | | $\varphi_2$ | 0.9924 |
| | Figure 7.37 | $Q_3$ | 0.9929 |
| | | $Q_4$ | 0.9949 |
| | | $\varphi_{10}$ | 0.9929 |
| | | $\varphi_3$ | 0.9965 |

The rapid alterations in the power spectrum of a single traces (central trace), observable in Figures, are the artefacts that occur due to windowing, considering that only 5 periods were used for the sine stimulus. When the power spectrums of all the traces are spatially summed the artefacts are gone.

#### 7.2.4.1. Time of code execution

According to (Sadiku, 2009), the FD methods can be up to two times faster in CPU time than equivalent TLM programs under identical conditions. In order to compare the CPU times required for both methods, the main algorithm of NeuroField program, along with the FD approximation of the governing wave PDEs, was translated from C++ into MATLAB (Appendix B) and then the execution times for each simulation were measured. The results are shown in the Table 7-8.

*Table 7-8 – Average execution times for FD and TLM simulations of NeuroField One- and Four-populations models ran in MATLAB 2017a and C++*

| | **MATLAB 2017a** | | **C++** |
|---|---|---|---|
| | FD [s] | TLM [s] | FD [s] |
| One-population model: 8 simulations/method | 3048.70 | 3048.67 | 8.504 |
| Four-population model: 1 simulation/method | 12469.76 | 12671.11 | 12.347 |

From the Table 7-8 we can see that both methods' execution times in MATLAB, even though they are equally fast, are significantly worse than compared to NeuroField program built in C++, where the same Four-population model takes only 12 seconds to execute. The biggest slowdown in both MATLAB programs is the calculation of differential equations required to find soma potentials $V_{ab}$, using standard MATLAB function *ode45*. Finding $V_{ab}$ is necessary for calculation of firing rates for each population.

The real CPU time required for FD and TLM methods in MATLAB was obscured because of the calculation of differential equations required to find soma potentials $V_{ab}$, thus we ran another comparison of CPU time of code execution for both numerical methods by bypassing the slow pre-processing algorithm.

*Table 7-9 – Average execution times for FD and TLM simulations of NeuroField One- and Four-populations models ran in MATLAB 2017a when the slow pre-processing algorithm is bypassed*

| | FD [s] | TLM [s] | Ratio (TLM/FD) |
|---|---|---|---|
| One-population model | 0.08569 | 0.18695 | 2.1817 |
| Four-population model | 0.15972 | 0.30451 | 1.9065 |

This time, we can see from Table 7-9 that for the One-population model the FD method is two times faster than the equivalent TLM program, which agrees with the literature (Sadiku, 2009). The result isn't surprising because the TLM method has four commands to execute in each iteration (calculate current $I_k$, calculate scattered pulses, connect to the next nodes, calculate new nodes' voltages), where in the FD method the 5-point stencil is sliding across the whole mesh in two "*for*" loops. When the complexity of the model is increasing, it is noticeable that the CPU time difference is lowering. This is probably due to the matrix implementation of TLM method, which is significantly faster to execute in MATLAB than "*for*" loops used in FD method.

Perhaps the code execution times for TLM could be reduced more by more careful code optimisation, but FD method run times could also be lowered by using the matrix approach, as described in Chapter 5.

#### *7.2.4.2. Fitting parameters in TLM method to better correspond to FD*

While we were testing the effects of changing temporal damping coefficient, $\gamma_{ab}$, on NeuroField models, Figure 7.27, we found that central node traces reach maximum firing rate faster as $\gamma_{ab}$ is rising. That behaviour was observed for both methods and it shows that governing inhomogeneous damped wave differential equation is acting like a response from an overdamped RLC Low Pass Filter. When $\gamma_{ab}$ is rising, damped wave

equation will reach critically damped response, which is the rise with the fastest possible time without getting into oscillation (unstable) state. If we would push $\gamma_{ab}$ above critical stage, we would get the underdamped oscillatory response. From the same figure, it is also noticeable that there is almost constant difference between two methods as the damping coefficient is changing.

Equatorial profiles in Figure 7.28 show that there is a difference in wave spreading as temporal damping coefficient is changing, but the radius of the spread remains the same, which is expected as the wave speed is kept constant.

Analysing Figure 7.27 we found that in order to have the same axonal propagation fields from both numerical methods, we should make $\gamma_{ab}$ parameter slightly bigger in TLM, which is probably due to the stray inductances and capacitances in TLM nodes. We used Nash-Sutcliffe Efficiency Index to compare $\varphi_1$ from both methods and find the optimal value for $\gamma_{ab}$ parameter in TLM, Table 7-10. If we would plot the original versus optimal parameters we would get the linear function, shown in Figure 7.42.

*Table 7-10 – Temporal damping coefficients $\gamma_{ab}$ used in NeuroField models with TLM method, original and optimally fitted and Nash-Sutcliffe Efficiency Index (NSEI) comparing axonal propagation fields when using TLM to FD method with original $\gamma_{ab}$.*

| **Original $\gamma_{ab}$** | **NSEI for original $\gamma_{ab}$** | **Optimal $\gamma_{ab}$** | **NSEI for optimal $\gamma_{ab}$** |
|---|---|---|---|
| **One-population model** | | | |
| 30 | 0.9616204 | 36 | 0.9999256 |
| 40 | 0.9675656 | 48 | 0.9999242 |
| 60 | 0.9722288 | 72 | 0.9999093 |
| 100 | 0.9754292 | 120 | 0.9998594 |
| 120 | 0.9762378 | 144 | 0.9998289 |
| 150 | 0.9771014 | 181 | 0.9997777 |
| **Four-population model** | | | |
| 116 | 0.9951895 | 140 | 0.9999839 |

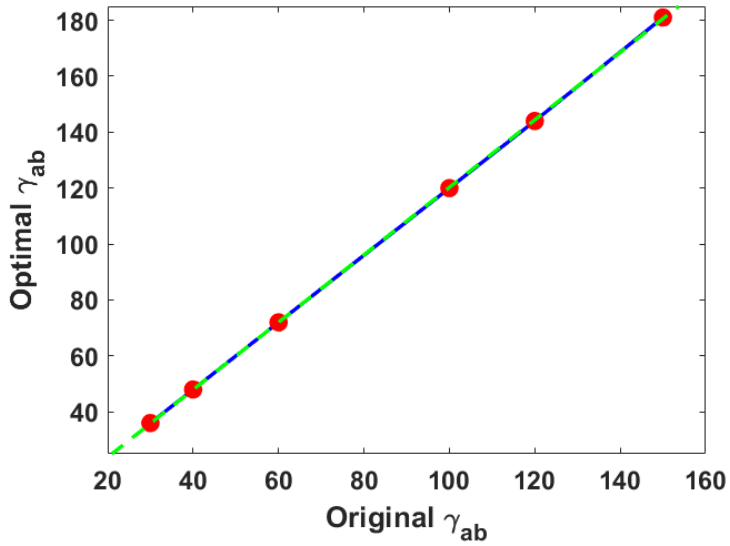


*Figure 7.42 – The relationship between original versus optimal $\gamma_{ab}$ parameters after fitting TLM method to FD for One-population model. Blue line – straight line connection between data points (Red dots), Green dashed line – linear fit through data points.*

Finally, we used the optimal $\gamma_{ab}$ parameters found for One- and Four-populations models to run the TLM simulations again and compared them to FD simulations with the original $\gamma_{ab}$ parameters. The results for One-population model driven by the sine wave of amplitude $1s^{-1}$ and frequency of 20Hz applied at the centre of the grid, for $\gamma_{ab} = 30s^{-1}$ are shown in Figure 7.43 for central node traces and Figure 7.44 for equatorial profiles, while the results for Four-population model with $\gamma_{ab} = 116s^{-1}$ driven by the same stimulus and are shown in Figure 7.45 and Figure 7.46.

From these figures we can see that TLM method is almost perfectly matched with FD when optimal value for $\gamma_{ab}$ parameter is used.

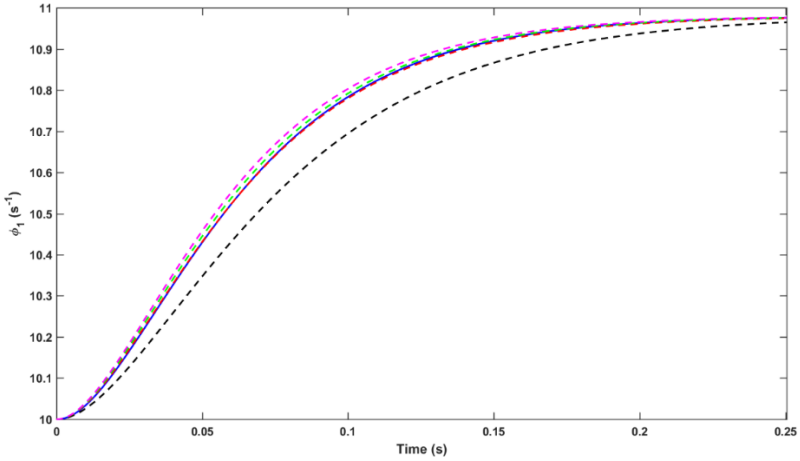


*Figure 7.43 – Fitting $\gamma_{ab}$ parameters for TLM method to match FD. Central node traces of propagation fields for One-population NeuroField model driven by a sine wave of amplitude $1s^{-1}$ and frequency of 20Hz. FD – solid blue line with $\gamma_{ab} = 30s^{-1}$, TLM – dashed lines: $\gamma_{ab} = 30s^{-1}$ – black; $\gamma_{ab} = 36s^{-1}$ – red; $\gamma_{ab} = 37s^{-1}$ – green; $\gamma_{ab} = 38s^{-1}$ – purple.*

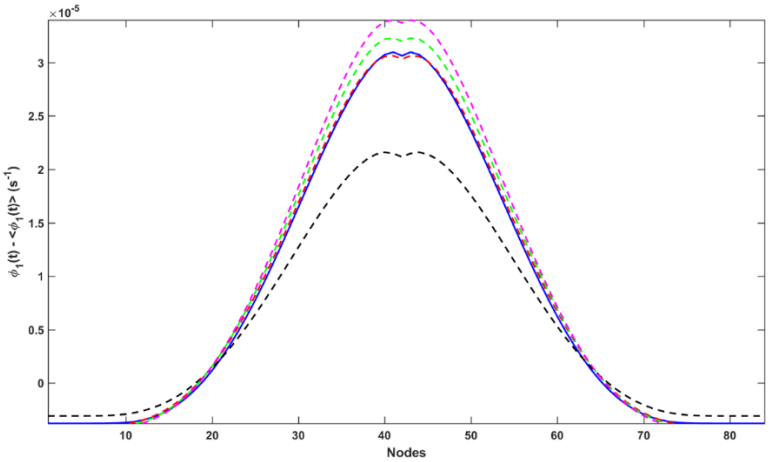


*Figure 7.44 – Fitting $\gamma_{ab}$ parameters for TLM method to match FD. Equatorial profiles of axonal fields, with the subtracted mean values, of the One-population model at iteration step 50. The model was driven by a sine wave of amplitude $1s^{-1}$ and frequency of 20Hz. FD – solid blue line with $\gamma_{ab} = 30s^{-1}$, TLM – dashed lines: $\gamma_{ab} = 30s^{-1}$ – black; $\gamma_{ab} = 36s^{-1}$ – red; $\gamma_{ab} = 37s^{-1}$ – green; $\gamma_{ab} = 38s^{-1}$ – purple.*

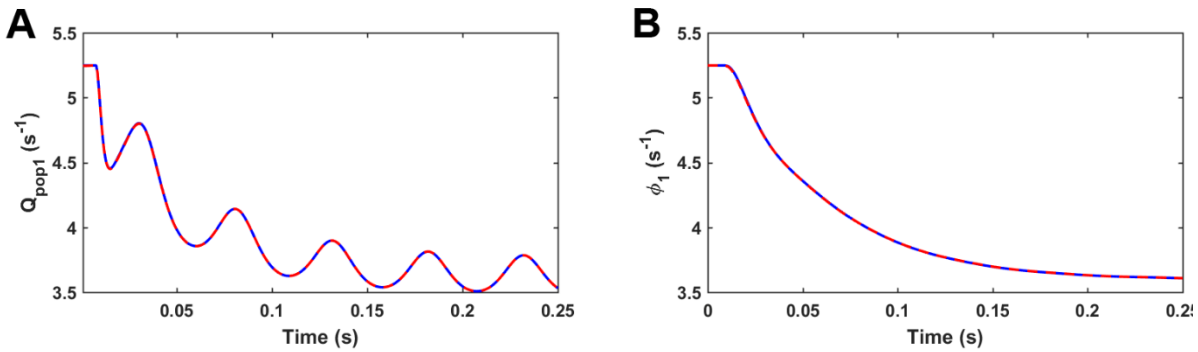


*Figure 7.45 – Fitting $\gamma_{ab}$ parameters for TLM method to match FD. Central node traces for Four-population NeuroField model driven by a sine wave of amplitude $1s^{-1}$ and frequency of 20Hz. A) Mean firing rates for population 1, $Q_1$, B) propagation fields $\varphi_1$. FD – solid blue line with $\gamma_{ab} = 116s^{-1}$, TLM – dashed red line with $\gamma_{ab} = 140s^{-1}$.*

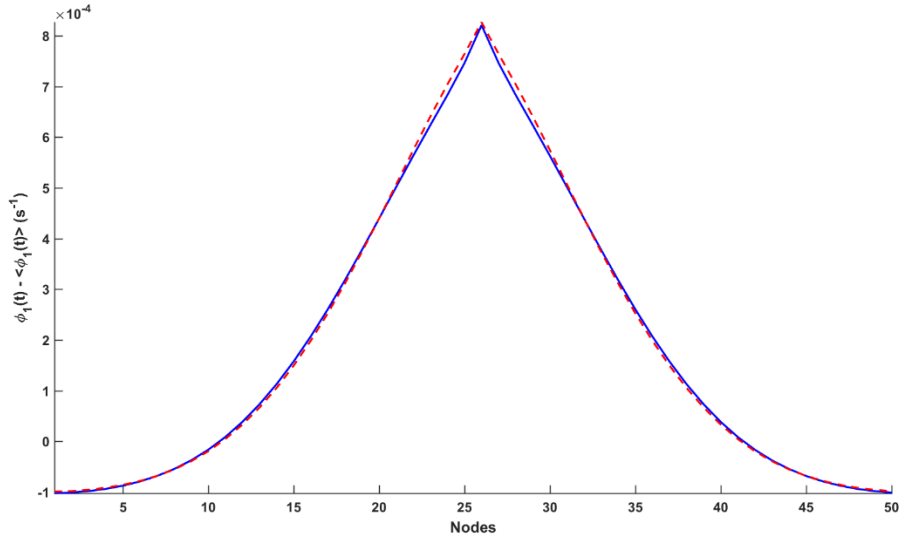


*Figure 7.46 – Fitting $\gamma_{ab}$ parameters for TLM method to match FD. Equatorial profiles of axonal fields, with the subtracted mean values, of the Four-population model at iteration step 49. The model was driven by a sine wave of amplitude $1s^{-1}$ and frequency of 20Hz. FD – solid blue line with $\gamma_{ab} = 116s^{-1}$, TLM – dashed red line with $\gamma_{ab} = 140s^{-1}$.*

## 7.3. Conclusion

The two numerical methods (TLM and FD) were compared in this Chapter for different levels of complexity of wave PDEs. First, they were compared by solving the least complicated case, the undamped wave PDEs (subsection 7.1.1). Afterwards, the methods were tested on damped wave PDEs (subsection 7.1.2). Finally, TLM and FD were compared for three cortical models (one-, two- and four-population), governed by the

inhomogeneous damped wave PDEs, and with the Dirac impulse, Gaussian wave and sine wave as the stimuli. The Dirac impulse was used to test the models' response to a single shot excitation. In order to excite the mesh with the broad frequency range, but to avoid the high frequencies, the Gaussian wave was chosen for a stimulus. Finally, to test the model for the specific frequencies that may occur in brain, the sine wave stimulus was used. The results of NeuroField simulations with TLM method show a great compatibility when compared to NeuroField numerically approximated by FD (Table 7-7). The slight differences in simulations may occur due to the stray inductances and capacitances in TLM nodes and it was shown that they could be minimised by changing the temporal damping coefficient $\gamma_{ab}$ in TLM based model (subsection 7.2.4.2).

The computational efficiency the TLM method was tested in subsection 7.2.4.1. It has been shown that for the One-population model the FD method is two times faster in CPU time than the equivalent TLM program, which agrees with the literature (Sadiku, 2009), but as the complexity of the model is increasing, it is noticeable that the difference in required CPU time is lowered due to the matrix implementation of TLM method, which is considerably faster to execute in MATLAB than "*for*" loops used in FD method.

# 8. Conclusions and recommendations for future work

The TLM method is one of the best-known examples of analogue models used to numerically solve the equations modelling a physical phenomenon. In TLM, an electrical network is used to mimic the physical problem, so the solutions could be obtained using conventional circuit analysis techniques in either the time or frequency domains. The versatility of the TLM method allows straightforward calculation of complicated structures, boundaries and material properties. There are no problems with convergence, stability or spurious solutions in TLM and the method is limited only by the amount of memory storage required, which depends on the complexity of the TLM mesh (Hoefer, 1985). Due to its simplicity of formulation and programming, it is used in variety of research fields where the wave equations should be solved numerically. The major advantage of the TLM over FD method is that all the required discretisation is built-in in the initial model, which is then solved without any further approximation avoiding many

anomalous effects that can arise in FD (Cogan et al., 2005). When compared to FDTD in EM models TLM is reported to sometimes be two times slower in CPU time and requires more memory space (Sadiku, 2009). That is mostly due to a simpler mathematical algorithm which the FDTD is based on and the fact that for 3D node FDTD requires only seven real memory stores compared to 22 stores per node in TLM for an isotropic dielectric medium (Sadiku, 2009). One of the possible applications of TLM method is in neuroscience, specifically, in modelling the brain functions as speculated by Nunez in (Nunez & Srinivasan, 2006) and Weiner in (M. Weiner, 2010), where the TLM method may be used as a framework to describe neurological activity of the brain, since it relies on a vast array of nerve fibres and synapses, analogous to the transmission lines and nodes of the TLM matrix.

In this thesis, the feasibility to numerically solve the governing inhomogeneous damped wave PDEs from neural field theory, used in NeuroField program, using TLM techniques has been explored. The hypothesis tested was whether the usage of TLM leads to more understandable and efficient brain modelling and what the cost in computer resources for those benefits is. This approach differs from the currently used FD numerical method by providing the electrical equivalent network where all the NeuroField model parameters have analogues in electrical elements of TLM node, thus enabling better interpretation of the physical implications of discretisation and of the model. In order to compare the cost in computer resources of both methods, the main algorithm of NeuroField program, along with the FD approximation of the governing wave PDEs was translated from C++ into MATLAB (Appendix B).

NeuroField is a multiscale neural field brain model developed by Prof. Peter Robinson and his Brain Dynamics group at The University of Sydney (P. Sanz-Leon, 2017; Robinson et al., 2005). It models the whole brain dynamics through the interactions of spatially extended populations of neurons and can predict the spectral and time characteristics of brain electrical activity observable by various non-invasive imaging modalities (P. Sanz-Leon, 2017). The governing neural field equations in NeuroField represent the axonal propagation of activity through the cortex and are numerically solved in NeuroField by applying the FD method (Robinson et al., 1997).

In Chapter 6 the numerical approximations of NeuroField damped wave equations were developed and solved using TLM method. The electrical equivalent parameters for TLM node are calculated in the same Chapter. The analysis of space and time discretisation for both methods showed that TLM is unconditionally stable method, compared to FD, where the length of the cell and the time step need to be picked cautiously so that they can meet the Courant condition.

Two methods were compared in Chapter 7 using three levels of complexity of cortical models (one-, two- and four-population) and with Dirac impulse, sine wave and Gaussian wave as the stimuli. The results of NeuroField simulations with TLM method show a great compatibility when compared to NeuroField numerically approximated by FD. The slight differences in simulations may occur due to the stray inductances and capacitances in TLM nodes and can be minimised by changing the temporal damping coefficient $\gamma_{ab}$ in TLM based model.

Being a viable solution, the computational efficiency the TLM method was tested. It has been shown that for the One-population model the FD method (Appendix B) is two

times faster in CPU time than the equivalent TLM program (Appendix A), which agrees with the literature (Sadiku, 2009), but as the complexity of the model is increasing, it is noticeable that the difference in required CPU time is lowered due to the matrix implementation of TLM method, which is considerably faster to execute in MATLAB than "*for*" loops used in FD method.

Encouraged by these results, we propose that the next step in TLM modelling of neural fields would be to translate the developed TLM code in C++ language, and plug it into the NeuroField code for further testing of the compatibility and speed of execution. When compared to FD method built in C++ NeuroField program, where the same Four-population model takes only couple of seconds to execute, both methods' execution times in MATLAB are significantly worse due to the slow pre-processing algorithm for calculation of differential equations required to find soma potentials $V_{ab}$, using standard MATLAB function *ode45*.

Replacing the FD method for numerically approximating governing wave equations in NeuroField with TLM should enable better interpretation of the physical implications of discretisation and of the model by modelling the physical problem with the electrical equivalent network where all the NeuroField model parameters have analogues in electrical elements of TLM node, thus opening a great possibility for a development of a brain-on-the-chip for in-silico multiscale brain experimentation, which will greatly help the advancement of neuroscience.

# Appendix A: MATLAB code for One-population model using TLM

```
function [Q_POP1, PHI1, Q_POP2, PHI2] = TLM_code_Neurofield_1popNFTMod_Sine
(freq,r_a,gamma_ab)

%% TLM NeuroField code
% Implementation of the lossy TLM cell with parameters matched with NeuroField:
% Zo=1; Ld=2*Cd*Zo^2=Zo/(gamma_ab*r_a); Rd = (Zo/(2*r_a)); Gd = 1/(Zo*2*r_a); V=Phi;
Ik=((dx/(Zo*r_a))*Q+(1/(gamma_ab*dt_tlm))*Ik(iter-1))/(1+1/(gamma_ab*dt_tlm));
% Simulation of "onepop.conf" with Sine wave stimulus and periodic BC

%% Set global variables
global alpha beta P_i
% UNITS
meters      = 1;
seconds     = 1;
hertz       = 1/seconds;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% DASHBOARD
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% NEUROFIELD PARAMETERS

% Grid size
Nx_nft = 30; % number of cells in X direction
Ny_nft = 30; % number of cells in Y direction

% Sigmoid parameters
Theta = 0.01292;
Sigma = 0.0038;
Qmax = 340;

% Dendrite parameters
alpha = 83;
beta = 769;

% Propagation parameters (wave)
% r_a = 0.2 * meters; % mean range of axons
% gamma_ab = 30 * hertz; % cortical damping rate
v_a = r_a * gamma_ab; % axonal velocity

% Coupling parameters
nu_1 = 0;
nu_2 = 1e-4;

% Initial firing rate for the whole population:
Qin = 10;

% SOURCE PARAMETERS
fmax = 100 * hertz; % max freq that we want to simulate. From fmax we calculate the
duration of our pulse source!
lam0 = v_a/fmax; % minimal freespace wavelength of our simulation

% DEVICE PARAMETERS
w = 0.5 * meters;
h = 0.5 * meters;

% GRID PARAMETERS
disp_fact = 0.1; % dispersion factor for TLM (when dx/lam <= 0.1 v_tlm = 1/sqrt(2)*v_a
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% COMPUTE OPTIMIZED GRID
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% NOMINAL RESOLUTION

dx_tlm  = lam0*disp_fact; % grid resolution resolving the shortest wavelength (lam0/nmax =
min wavelength)
dx_nft  = w/Nx_nft; % resolving the minimum dimension
dx      = min([dx_tlm dx_nft]);
dy      = dx;

% SNAP GRID TO CRITICAL DIMENSION
Nx = ceil(w/dx);
dx = w/Nx;
Ny = ceil(w/dy);
dy = w/Ny;


%% The rest of GRID parameters for TLM
v_tlm = v_a; % speed of the wave

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% BUILD DEVICE ON GRID
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% COMPUTE START AND STOP INDICES OF DEVICE
nx_dev = round(w/dx); % number of cells for width of the device = Nx in this case
x_start_dev = 2; % where does the device start on our grid
x_end_dev = x_start_dev + nx_dev - 1; % where does the device end on our grid
ny_dev = round(h/dy); % = Ny in this case
y_start_dev = 2;
y_end_dev = y_start_dev + ny_dev - 1;

% COMPUTE GRID SIZE
Nx = nx_dev + 2; % number of cells, including boundary regions
Sx = Nx*dx; % new physical size of the whole simulation grid

Ny = ny_dev + 2;
Sy = Ny*dy;

% COMPUTE GRID AXIS
xa = (0:Nx-1)*dx;
ya = (0:Ny-1)*dy;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% COMPUTE THE SOURCE
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% COMPUTE STABLE TIME STEP (dt)
dmin = min([dx dy]);
dt_tlm = dmin/(sqrt(2)*v_tlm);
% freq = 20*hertz;

% COMPUTE SOURCE PARAMETERS
tau = 0.5/freq; % duration needs to be sufficient so that includes enough power at max freq
t0 = 3*tau; % offset - if not given at the 1st step we will be in the middle of gaussian.
            % It's not good to turn on the source that fast, we should rather ease into it
            % and out of it.

% COMPUTE THE NUMBER OF TIME STEPS
STEPS = 357+1; % 5 periods

% COMPUTE THE SOURCE
% ta = (0: STEPS-1-1)*dt_tlm; % time array
ta = (1: STEPS-1)*dt_tlm; % time array - it should begin with 0,
                          % but this way is consistent with NeuralField Cpp program
```

```
% Gaussian source:
% stim = exp(-((ta - t0)/tau).^2);
% stim(ceil(2*t0/dt_tlm)+2:end)=0; % cut Gaussian to be symmetrical with respect to centre

% Sine wave source:
stim = 1*sin(2*pi*freq*ta);

% Pulse source:
% pulse_ON = 1;
% pulse_OFF = 9;
% stim = zeros(1,length(ta));
% stim(pulse_ON:pulse_OFF) = 1;

% POSITION OF THE SOURCE
nx_src = ceil(Nx/2);
ny_src = ceil(Ny/2);

%% Plot stimulus
% figure()
% plot(ta,stim)
% xlabel('Time (s)')
% ylabel('Amplitude')
% title('Stimulus')

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% INITIALIZE TLM PARAMETERS
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
Zo = 1;
Ro = dx*(Zo/(2*r_a));
G = 2*dx*(1/(Zo*2*r_a));

C1 = 8/(4+(Ro+Zo)*G);
C2 = (Ro+Zo)/(4+(Ro+Zo)*G);
C3 = dx/(Zo*r_a);
C4 = 1/(gamma_ab*dt_tlm);
C5 = 1+C4;

% set up connection matrices
Cn=[[zeros(Nx-1,1) eye(Nx-1)];zeros(1,Nx)];Cs=Cn';
Ce=[[zeros(Ny-1,1) eye(Ny-1)];zeros(1,Ny)];Cw=Ce';

% make room for incident and scattered voltages
mvi = zeros(Ny,Nx,4); % Matrix of Vis all points repeated 4x (4 ports ) for each
interaction
mvr = zeros(Ny,Nx,4); % Matrix of Vrs all points repeated 4x (4 ports ) for each
interaction
Ey = zeros(Ny,Nx); % Matrix of Ex for all the points for each interaction
Ik = zeros(Ny,Nx); % Current source generator

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% INITIALIZE POPULATIONS
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
Q_POP1 = zeros(ny_dev,nx_dev,STEPS); % Mean firing rate for Population 1
PHI1 = zeros(ny_dev,nx_dev,STEPS); % Axonal propagation field 1
Q_POP2 = zeros(ny_dev,nx_dev,STEPS); % Mean firing rate for Population 2
PHI2 = zeros(ny_dev,nx_dev,STEPS); % Axonal propagation field 2

Q_POP1(:,:,1:2) = Qin;
PHI1(:,:,1:2) = Qin;

% Initialize Nodal Voltages for TLM:
Ik(y_start_dev:y_end_dev,x_start_dev:x_end_dev) = C3*Q_POP1(:,:,1);
```

```
mvi(y_start_dev:y_end_dev,x_start_dev:x_end_dev,1)=(PHI1(:,:,1)-
C2*Ik(y_start_dev:y_end_dev,x_start_dev:x_end_dev))/C1;
mvi(y_start_dev:y_end_dev,x_start_dev:x_end_dev,2)=(PHI1(:,:,1)-
C2*Ik(y_start_dev:y_end_dev,x_start_dev:x_end_dev))/C1;
mvi(y_start_dev:y_end_dev,x_start_dev:x_end_dev,3)=(PHI1(:,:,1)-
C2*Ik(y_start_dev:y_end_dev,x_start_dev:x_end_dev))/C1;
mvi(y_start_dev:y_end_dev,x_start_dev:x_end_dev,4)=(PHI1(:,:,1)-
C2*Ik(y_start_dev:y_end_dev,x_start_dev:x_end_dev))/C1;

Ey(:,:)=(2*(mvi(:,:,1)+mvi(:,:,2)+mvi(:,:,3)+mvi(:,:,4)))/(4+(Ro+Zo)*G)+Ik(:,:)*((Ro+Zo)/(4
+(Ro+Zo)*G));
% Now Ey(:,:,1)=PHI_POP1(:,:,1)=Q_POP1(:,:,1)=Qin

% Initialize vars for diff eq to get some potential V:
timerange = [0 dt_tlm];
V_i = zeros(1,ny_dev*nx_dev);
dV_i_dt = zeros(1,ny_dev*nx_dev);

h = waitbar(0,'Please wait...'); % Initialise progress bar
tic

for iter = 2:STEPS % Main loop

    % Finding Q_POP1:
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    % Solving diff eq to get soma potential V:
    P = nu_1*PHI1(:,:,iter-1) + nu_2*PHI2(:,:,iter-1);
    P = reshape(P,[],1);
    for i = 1 : length(P)
        P_i = P(i);
        initalvalue = [V_i(i) dV_i_dt(i)];
        [~,V_temp] = ode45(@soma_potential,timerange,initalvalue);
        V_i(i)  = V_temp(end,1);
        dV_i_dt(i) = V_temp(end,2);
    end
    V = reshape(V_i,ny_dev,nx_dev);
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    Q_POP1(:,:,iter) = Qmax./( 1 + exp( -(V-Theta)./Sigma ) );


    % Calculate PHI1 (TLM scatter process):

    Ik(y_start_dev:y_end_dev,x_start_dev:x_end_dev) =
(C3*Q_POP1(:,:,iter)+C4*Ik(y_start_dev:y_end_dev,x_start_dev:x_end_dev))/C5;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Calculate scattered pulses
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    mvr(:,:,:) = (Ey(:,:).*Zo + mvi(:,:,:).*(Ro-Zo))./(Ro+Zo);
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Connection to next node - calculate incident pulses
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    mvi(:,:,1) = Cn*mvr(:,:,3);
    mvi(:,:,3) = Cs*mvr(:,:,1);
    mvi(:,:,2) = mvr(:,:,4)*Ce;
    mvi(:,:,4) = mvr(:,:,2)*Cw;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Boundary conditions
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    % Periodic BC - folded sheet simulating torus
    mvi(2,:,3) = mvr(Ny-1,:,1);
    mvi(Ny-1,:,1) = mvr(2,:,3);
    mvi(:,Nx-1,4) = mvr(:,2,2);
    mvi(:,2,2) = mvr(:,Nx-1,4);
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Outputs to a specific point of the mesh
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    Ey(:,:) = (2*(mvi(:,:,1)+mvi(:,:,2)+mvi(:,:,3)+mvi(:,:,4)))/(4+(Ro+Zo)*G) +
Ik(:,:)*((Ro+Zo)/(4+(Ro+Zo)*G));

    PHI1(:,:,iter) = Ey(y_start_dev:y_end_dev,x_start_dev:x_end_dev);

    % Calculate PHI2:
    Q_POP2(ny_src,nx_src,iter) = stim(iter-1);
    PHI2(:,:,iter) = Q_POP2(:,:,iter); % this is because the propagation function is "MAP"

    % update progress bar:
    waitbar(iter/STEPS,h,[num2str(iter) ' of ' num2str(STEPS) ' finished'])
end

toc
close(h) % close progress bar
```

# Appendix B: MATLAB code for One-population model using FD

```
function [Q_POP1, PHI1, Q_POP2, PHI2] = Neurofield_Matlab_1popNFTMod_Sine
(freq,r_a,gamma_ab)

%% NeuroField code in Matlab
% Reprogramed NFT code from C++ to Matlab for comparison with TLM Neurofield code
% Simulation of "onepop.conf" with Sine wave, Gaussian and Pulse stimulus and periodic BC

%% Set global variables
global alpha beta P_i
% UNITS
meters      = 1;
seconds     = 1;
hertz       = 1/seconds;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% DASHBOARD
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% NEUROFIELD PARAMETERS

% Grid size
Nx_nft = 30; % number of cells in X direction
Ny_nft = 30; % number of cells in Y direction

% Sigmoid parameters
Theta = 0.01292;
Sigma = 0.0038;
Qmax = 340;

% Dendrite parameters
alpha = 83;
beta = 769;

% Propagation parameters (wave)
% r_a = 0.2 * meters; % mean range of axons
% gamma_ab = 30 * hertz; % cortical damping rate
v_a = r_a * gamma_ab; % axonal velocity

% Coupling parameters
nu_1 = 0;
nu_2 = 1e-4;

% Initial firing rate for the whole population:
Qin = 10;

% SOURCE PARAMETERS
fmax = 100 * hertz; % max freq that we want to simulate. From fmax we calculate the
duration of our pulse source!
lam0 = v_a/fmax; % minimal freespace wavelength of our simulation

% DEVICE PARAMETERS
w = 0.5 * meters;
h = 0.5 * meters;

% GRID PARAMETERS
disp_fact = 0.1; % dispersion factor for TLM (when dx/lam <= 0.1 v_tlm = 1/sqrt(2)*v_a
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% COMPUTE OPTIMIZED GRID
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% NOMINAL RESOLUTION

dx_tlm  = lam0*disp_fact; % grid resolution resolving the shortest wavelength (lam0/nmax =
min wavelength)
dx_nft  = w/Nx_nft;% resolving the minimum dimension
dx      = min([dx_tlm dx_nft]);
dy      = dx;

% SNAP GRID TO CRITICAL DIMENSION
Nx = ceil(w/dx);
dx = w/Nx;
Ny = ceil(w/dy);
dy = w/Ny;

% Adjust wave speed:
v_tlm = v_a;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% COMPUTE THE SOURCE
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% COMPUTE STABLE TIME STEP (dt)
dmin = min([dx dy]);
dt_tlm = dmin/(sqrt(2)*v_tlm);
% freq = 20*hertz;

% COMPUTE SOURCE PARAMETERS
tau = 0.5/freq; % duration needs to be sufficient so that includes enough power at max freq
t0 = 3*tau; % offset - if not given at the 1st step we will be in the middle of gaussian.
            % It's not good to turn on the source that fast, we should rather ease into it
and out of it.

% COMPUTE THE NUMBER OF TIME STEPS
STEPS = 359; % 5 periods - 357+2(2 is added for initial steps t(-1) and t(0))

% COMPUTE THE SOURCE
% ta = (0: STEPS-1-2)*dt_tlm; % time array
ta = (1: STEPS-1-1)*dt_tlm; % time array - it should begin with 0,
                            % but this way is consistent with NeuralField Cpp program
% Gaussian source:
% stim = exp(-((ta - t0)/tau).^2);
% stim(ceil(2*t0/dt_tlm)+2:end)=0; % cut Gaussian to be symmetrical with respect to centre

% Sine wave source:
stim = 1*sin(2*pi*freq*ta);

% Pulse source:
% pulse_ON = 1;
% pulse_OFF = 9;
% stim = zeros(1,length(ta));
% stim(pulse_ON:pulse_OFF) = 1;

% POSITION OF THE SOURCE
nx_src = ceil(Nx/2);
ny_src = ceil(Ny/2);

%% Plot stimulus
% figure()
% plot(ta,stim)
% xlabel('Time (s)')
% ylabel('Amplitude')
% title('Stimulus')
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% INITIALIZE POPULATIONS
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Calculated constants (from NeuroField program - Eq. 6.4 in thesis)
p_tlm = v_tlm*dt_tlm/dx; %Courant condition - should be 1/sqrt(2)
A1 = 2 - 4*p_tlm^2;
A2 = dt_tlm^2*gamma_ab^2/12;
A3 = 10 - 4*p_tlm^2;
expfactneg = exp(-dt_tlm*gamma_ab);
expfactpos = exp(dt_tlm*gamma_ab);

Q_POP1 = zeros(Ny,Nx,STEPS); % Mean firing rate for Population 1
PHI1 = zeros(Ny,Nx,STEPS); % Axonal propagation field 1
Q_POP2 = zeros(Ny,Nx,STEPS); % Mean firing rate for Population 2
PHI2 = zeros(Ny,Nx,STEPS); % Axonal propagation field 2

Q_POP1(:,:,1:2) = Qin;
PHI1(:,:,1:2) = Qin;

% Initialize vars for diff eq to get soma potential V:
timerange = [0 dt_tlm];
V_i = zeros(1,Ny*Nx);
dV_i_dt = zeros(1,Ny*Nx);

h = waitbar(0,'Please wait...'); % Initialise progress bar
tic

for iter = 3:STEPS % Main loop
    % Finding Q_POP1:
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    % Solving diff eq to get soma potential V:
    P = nu_1*PHI1(:,:,iter-1) + nu_2*PHI2(:,:,iter-1);
    P = reshape(P,[],1);

    for i = 1 : length(P)
        P_i = P(i);
        initalvalue = [V_i(i) dV_i_dt(i)];
        [~,V_temp] = ode45(@soma_potential,timerange,initalvalue);
        V_i(i)  = V_temp(end,1);
        dV_i_dt(i) = V_temp(end,2);
    end
    V = reshape(V_i,Ny,Nx);
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    Q_POP1(:,:,iter) = Qmax./( 1 + exp( -(V-Theta)./Sigma ) );

    % Calculate PHI1 (FD method):
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    % Find Phi from Q using FD method from NeuroField Cpp code (Eq. 6.4 in thesis):
    for y=2:Ny-1
        for x=2:Nx-1
            PHI1(y,x,iter) = expfactneg*(A1*PHI1(y,x,iter-1) + p_tlm^2*(PHI1(y-1,x,iter-
1)+PHI1(y+1,x,iter-1)+PHI1(y,x+1,iter-1)+PHI1(y,x-1,iter-1)) - PHI1(y,x,iter-
2)*expfactneg...
                + A2*(A3*Q_POP1(y,x,iter-1) + (Q_POP1(y,x,iter)*expfactpos +
Q_POP1(y,x,iter-2)*expfactneg) + p_tlm^2*(Q_POP1(y-1,x,iter-1)+Q_POP1(y+1,x,iter-
1)+Q_POP1(y,x+1,iter-1)+Q_POP1(y,x-1,iter-1))));
        end
    end

    % Periodic BC
    norht = PHI1(2,2:Nx-1,iter);
    south = PHI1(Ny-1,2:Nx-1,iter);
    west = PHI1(2:Ny-1,2,iter);
    east = PHI1(2:Ny-1,Nx-1,iter);
    nwc = PHI1(2,2,iter);
    swc = PHI1(Ny-1,2,iter);
```

```
    nec = PHI1(2,Nx-1,iter);
    sec = PHI1(Ny-1,Nx-1,iter);
    PHI1(1,2:Nx-1,iter) = south;
    PHI1(Ny,2:Nx-1,iter) = norht;
    PHI1(2:Ny-1,1,iter) = east;
    PHI1(2:Ny-1,Nx,iter) = west;
    PHI1(1,1,iter) = nec;
    PHI1(Ny,1,iter) = sec;
    PHI1(1,Nx,iter) = nwc;
    PHI1(Ny,Nx,iter) = swc;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

    % Calculate PHI2:
    Q_POP2(ny_src,nx_src,iter) = stim(iter-2);
    PHI2(:,:,iter) = Q_POP2(:,:,iter); % this is because the propagation function is "MAP"

    % update progress bar:
    waitbar(iter/STEPS,h,[num2str(iter) ' of ' num2str(STEPS) ' finished'])
end

toc
close(h) % close progress bar
```

# Appendix C: MATLAB code for solving 2D wave PDEs using 5-point stencil FD method

```
function [U_Matrix,tvec] = Wave2D_5p_stencil (Damped,Pwidth,Gaussian)

%% Wave2D_5p_stencil
% using an explicit central difference method for the 2D wave equation:
% U_tt = c^2*(U_xx+U_yy)
% Input parameters:
% Damped <- 0 for undamped wave; 1 for damped wave
% Pwidth <- width of Gaussian pulse (0.2 was used in thesis)
% Gaussian <- 0 for Dirac IC; 1 for Gaussian IC

x1=0;
max_x = 2; % length of membrane in x-direction
x2=max_x;

y1=0;
max_y = 2; % length of membrane in y-direction
y2=max_y;

T = 12 ; % length of time for solution (period 1.2)

n = 199;   % no of grid points Xn
p = 199;   % no of grid points Yn
m = 2400;  % 120 per period

dx = max_x/(n+1); % grid spacing in x direction
dy = max_y/(p+1); % grid spacing in y direction

dt = T/m; % timestep size

t=0; % initial time = 0

c = 1; % wave speed

if Damped
    kappa=0.2;  % frictional coefficient
    C1 = 1+kappa*dt;
    C2 = 1-kappa*dt;
    e = C2/C1;
else
    C1 = 1;
    e = 1;
end

s_x = (c^2)*(dt^2)/(dx^2);  % gain parameter in x direction
s_y = (c^2)*(dt^2)/(dy^2);  % gain parameter in y direction

CourantCondition_x = c*dt/dx;  % Courant condition for x direction
CourantCondition_y = c*dt/dy;  % Courant condition for y direction

if CourantCondition_x > 1
   fprintf('Courant Condition in x direction is > 1 so central difference method is
unstable, please reduce time step size to gain stability');
   return
end
```

```
if CourantCondition_y > 1
   fprintf('Courant Condition in y direction is > 1 so central difference method is
unstable, please reduce time step size to gain stability');
   return
end

%% Build the A matrix to march finite difference solution forward in time

lambda = 2*(1-s_x-s_y);

temp_diag = lambda*ones(n,1);
temp_sub = s_y*ones(n,1);
temp_sup = temp_sub;

% Create n-by-n diagonal matrix block:
A_diag_block = spdiags([temp_sub,temp_diag,temp_sup],[-1 0 1],n,n);

% show matrix:
% test_A_t = full (A_t);
clear temp_diag temp_sub temp_sup

% Create diagonal matrix A_Sx:
temp_diag_X = s_x*ones(n*p,1);
A_Sx = spdiags([temp_diag_X, temp_diag_X],[-n n],n*p,n*p);

% show matrix:
% test_A_X = full (A_X);
clear temp_diag_X

% Create a block diagonal matrix:
A_diag = kron(eye(n),A_diag_block);
clear A_diag_block

% Finally create matrix A:
A = (1/C1)*(A_diag + A_Sx);

% show matrix:
% test_A = full (A);
clear A_diag A_Sx

%% Specify boundary conditions through vector b and by changing any rows in A matrix needed
- for Neumann boundary conditions
% we have u(0,t) = 0 = u(a,t)

b = (1/C1)*zeros(n*p,1); %if the boundary conditions are different, then "zeros(n,1)"
should be changed to the adequate vector

%% Initial conditions for du/dt
% in this case u_t(x,y,0) = 0.

d = zeros(n*p,1); %if the initial conditions are different, change "d" accordingly

%% Set up mesh
% in x-direction:
x = linspace(x1+dx,x2-dx, n)';
% in y-direction:
y = linspace(y1+dy,y2-dy, p)';

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Set up vector U^0 at time tk=0: U(x,y,0) = f(x,y)
% Formatting Gaussian Pulse:

% Setting Gaussian parameters
Eo=5; % Pulse amplitude
Gmu=max_x/2; % Centre of pulse
sigmaxSq = Pwidth*max_x; %pulse width x - used to be 0.05 for elongated gaussian
```

```
sigmaySq = Pwidth*max_y; %pulse width y -

if Gaussian
    ax = 2*sigmaxSq;
    ay = 2*sigmaySq;

    % Generating Gaussian
    Gauss_x = exp(-pi^2*((x-Gmu).^2/ax)); % Horizontal

    Gauss_y = exp(-pi^2*((y-Gmu).^2/ay)); % Vertical

    U_tk_last_Matrix = Eo*Gauss_y*Gauss_x'; % Gaussian in 2D

else % Dirac I.C.
    U_tk_last_Matrix = zeros(n,p);
    U_tk_last_Matrix (round(length(x)/2),round(length(y)/2)) = 5;
end

%{
% Cut mask - if we want to set just the mid part of the membrane to
% gaussian and the rest to 0 (or some other value):

% Cut_mask = zeros(n,p);
% for i = 1:n
%     for j = 1:p
%         if x(i) >= 0.3*max_x && x(i) <= 0.7*max_x
%             if y(j) >= 0.3*max_y && y(j) <= 0.7*max_y
%                 Cut_mask(i,j)= 1;
%             else
%                 Cut_mask(i,j) = 0;
%             end;
%         else
%             Cut_mask(i,j) = 0;
%         end;
%     end;
% end;

U_tk_last_Matrix = U_tk_last_Matrix.*Cut_mask;
%}

U_tk_last = reshape(U_tk_last_Matrix.',[],1);

%% First we initialise and find U^1 = U_tk at time k=1

t = t+dt;
U_tk = (C1/2)*A*U_tk_last + (C1/2)*b + d;

U_tk_Matrix = reshape(U_tk,n,[]);
U_tk_Matrix = U_tk_Matrix';

%{
figure(1)
mesh (x,y,U_tk_last_Matrix)
title ('Initial condition for plucked elastic membrane')
xlabel ('x')
ylabel ('y')
zlabel ('U')
axis ('tight')

figure(2)
mesh (x,y,U_tk_Matrix)
title ('Vibrations of elastic membrane after 1 time step')
xlabel ('x')
ylabel ('y')
zlabel ('U')
axis ('tight')
```

```
%}

clear x y d Gauss_x Gauss_y
%% Store solution for each time in matrix U(n_x_m):

U_Matrix = zeros(n,p,m);
tvec = zeros(m,1);

U_Matrix(:,:,1) = U_tk_last_Matrix;
tvec(1) = t-dt;
U_Matrix(:,:,2) = U_tk_Matrix;
tvec(2) = t;

clear U_tk_last_Matrix U_tk_Matrix
%% March solution forward in time using U_tk+1 = A*U_tk + b:

for k = 2:m % Main loop
    t = t+dt;
    % if boundary conditions vary with time you need to update b here
    U_tk_new = A*U_tk + b - e*U_tk_last;

    U_tk_new_Matrix = reshape(U_tk_new,n,[]);
    U_tk_new_Matrix = U_tk_new_Matrix';
    U_Matrix(:,:,k) = U_tk_new_Matrix;

    % for next time step:
    U_tk_last = U_tk;
    U_tk = U_tk_new;
    tvec(k) = t;

end

clear U_tk_last U_tk U_tk_new
```

# Appendix D: MATLAB code for solving 2D wave PDEs using 9-point stencil FD method

```
function [U_Matrix,tvec] = Wave2D_9p_stencil (Damped,Pwidth,Gaussian)

%% Wave2D_9p_stencil
% using an explicit central difference method for the 2D wave equation:
% U_tt = c^2*(U_xx+U_yy)
% Input parameters:
% Damped <- 0 for undamped wave; 1 for damped wave
% Pwidth <- width of Gaussian pulse (0.2 was used in thesis)
% Gaussian <- 0 for Dirac IC; 1 for Gaussian IC

x1=0;
max_x = 2; % length of membrane in x-direction
x2=max_x;

y1=0;
max_y = 2; % length of membrane in y-direction
y2=max_y;

T = 12 ; % length of time for solution (period 1.2)

n = 199;   % no of grid points Xn
p = 199;   % no of grid points Yn
m = 2400;  % 120 per period

dx = max_x/(n+1); % grid spacing in x direction
dy = max_y/(p+1); % grid spacing in y direction

dt = T/m; % timestep size

t=0; % initial time = 0

c = 1; % wave speed

C1 = dx^2 + dy^2;
C2 = 10*dy^2 - 2*dx^2;
C3 = 10*dx^2 - 2*dy^2;

s = (c^2)*(dt^2)/(12*dx^2*dy^2);

s_x = C2*s;  % gain parameter in x direction
s_y = C3*s;  % gain parameter in y direction
s_m = C1*s;  % gain parameter in xy direction

if Damped
    kappa=0.2;  % frictional coefficient
    C4 = 1+kappa*dt;
    C5 = 1-kappa*dt;
    e = C5/C4;
else % Undamped wave
    C4 = 1;
    e = 1;
end

CourantCondition_x = c*dt/dx;  % Courant condition for x direction
CourantCondition_y = c*dt/dy;  % Courant condition for y direction
```

```
if CourantCondition_x > 1
   fprintf('Courant Condition in x direction is > 1 so central difference method is
unstable, please reduce time step size to gain stability');
   return
end

if CourantCondition_y > 1
   fprintf('Courant Condition in y direction is > 1 so central difference method is
unstable, please reduce time step size to gain stability');
   return
end

%% Build the A matrix to march finite difference solution forward in time

lambda = 2*(1-10*s_m);

temp_diag = lambda*ones(n,1);
temp_sub = s_y*ones(n,1);
temp_sup = temp_sub;

% Create n-by-n diagonal matrix block:
A_diag_block = spdiags([temp_sub,temp_diag,temp_sup],[-1 0 1],n,n);

% % show matrix:
% test_A_t = full (A_diag_block);
clear temp_diag temp_sub temp_sup

% Create diagonal matrix A_Sx:
temp_diag = s_x*ones(n,1);
temp_sub = s_m*ones(n,1);
temp_sup = temp_sub;
% Create n-by-n diagonal matrix block:
A_Sx_block = spdiags([temp_sub,temp_diag,temp_sup],[-1 0 1],n,n);
Ones_sub_sup = spdiags([ones(n,1),ones(n,1)],[-1 1],n,n);

% % show matrix:
% test_A_X = full (A_Sx_block);
clear temp_diag temp_sub temp_sup

% Create a block diagonal matrix:
A_diag = kron(eye(n),A_diag_block);

% % show matrix:
% test_A_diag = full (A_diag);
clear A_diag_block

A_Sx = kron(Ones_sub_sup,A_Sx_block);

% % show matrix:
% test_A_X = full (A_Sx);
clear A_Sx_block Ones_sub_sup

% Finally create matrix A:
A = (1/C4)*(A_diag + A_Sx);

% % show matrix:
% test_A = full (A);
clear A_diag A_Sx

%% Specify boundary conditions through vector b and by changing any rows in A matrix needed
- for Neumann boundary conditions
% we have u(0,t) = 0 = u(a,t)

b = (1/C4)*zeros(n*p,1); %if the boundary conditions are different, then "zeros(n,1)"
should be changed to the adequate vector
```

```
%% initial conditions for du/dt
% in this case u_t(x,y,0) = 0.

d = zeros(n*p,1); %if the initial conditions are different, change "d" accordingly

%% Set up mesh
% in x-direction:
x = linspace(x1+dx,x2-dx, n)';
% in y-direction:
y = linspace(y1+dy,y2-dy, p)';

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Set up vector U^0 at time tk=0: U(x,y,0) = f(x,y)
% Formatting Gaussian Pulse:

% Setting Gaussian parameters
Eo=5; % Pulse amplitude
Gmu=max_x/2; % Centre of pulse
sigmaxSq = Pwidth*max_x; %pulse width x - used to be 0.05 for elongated gaussian
sigmaySq = Pwidth*max_y; %pulse width y -

if Gaussian
    ax = 2*sigmaxSq;
    ay = 2*sigmaySq;

    % Generating Gaussian
    Gauss_x = exp(-pi^2*((x-Gmu).^2/ax)); % Horizontal

    Gauss_y = exp(-pi^2*((y-Gmu).^2/ay)); % Vertical

    U_tk_last_Matrix = Eo*Gauss_y*Gauss_x'; % Gaussian in 2D

else % Dirac I.C.
    U_tk_last_Matrix = zeros(n,p);
    U_tk_last_Matrix (round(length(x)/2),round(length(y)/2)) = 5;
end


%{
% Cut mask - if we want to set just the mid part of the membrane to
% gaussian and the rest to 0 (or some other value):

% Cut_mask = zeros(n,p);
% for i = 1:n
%     for j = 1:p
%         if x(i) >= 0.3*max_x && x(i) <= 0.7*max_x
%             if y(j) >= 0.3*max_y && y(j) <= 0.7*max_y
%                 Cut_mask(i,j)= 1;
%             else
%                 Cut_mask(i,j) = 0;
%             end;
%         else
%             Cut_mask(i,j) = 0;
%         end;
%     end;
% end;

U_tk_last_Matrix = U_tk_last_Matrix.*Cut_mask;
%}

U_tk_last = reshape(U_tk_last_Matrix.',[],1);
```

```
%% First we initialise and find U^1 = U_tk at time k=1

t = t+dt;
U_tk = (C4/2)*A*U_tk_last + (C4/2)*b + d;

U_tk_Matrix = reshape(U_tk,n,[]);
U_tk_Matrix = U_tk_Matrix';

%{
figure(1)
mesh (x,y,U_tk_last_Matrix)
title ('Initial condition for plucked elastic membrane')
xlabel ('x')
ylabel ('y')
zlabel ('U')
axis ('tight')

figure(2)
mesh (x,y,U_tk_Matrix)
title ('Vibrations of elastic membrane after 1 time step')
xlabel ('x')
ylabel ('y')
zlabel ('U')
axis ('tight')
%}

clear x y d Gauss_x Gauss_y
%% Store solution for each time in matrix U(nxm):

U_Matrix = zeros(n,p,m);
tvec = zeros(m,1);

U_Matrix(:,:,1) = U_tk_last_Matrix;
tvec(1) = t-dt;
U_Matrix(:,:,2) = U_tk_Matrix;
tvec(2) = t;

clear U_tk_last_Matrix U_tk_Matrix
%% March solution forward in time using U_tk+1 = A*U_tk + b:

for k = 2:m % Main loop
    t = t+dt;
    % if boundary conditions vary with time you need to update b here
    U_tk_new = A*U_tk + b - e*U_tk_last;

    U_tk_new_Matrix = reshape(U_tk_new,n,[]);
    U_tk_new_Matrix = U_tk_new_Matrix';
    U_Matrix(:,:,k) = U_tk_new_Matrix;

    % for next time step:
    U_tk_last = U_tk;
    U_tk = U_tk_new;
    tvec(k) = t;

end

clear U_tk_last U_tk U_tk_new
```

# References


Abeysuriya, R. G., Rennie, C. J., & Robinson, P. A. (2014). Prediction and verification of nonlinear sleep spindle harmonic oscillations. *J Theor Biol, 344*, 70-77. doi: 10.1016/j.jtbi.2013.11.013

Abeysuriya, R. G., Rennie, C. J., Robinson, P. A., & Kim, J. W. (2014). Experimental observation of a theoretically predicted nonlinear sleep spindle harmonic in human EEG. *Clin Neurophysiol, 125*(10), 2016-2023. doi: 10.1016/j.clinph.2014.01.025

Abramowitz, M. (1974). *Handbook of Mathematical Functions, With Formulas, Graphs, and Mathematical Tables*: Dover Publications, Incorporated.

Akhtarzad, S. (1975). *Analysis of Lossy Microwave Structures and Microstrip Resonators by the TLM Method*. PhD thesis.

Akhtarzad, S., & Johns, P. B. (1975). Three-Dimensional Transmission-Line Matrix Computer Analysis of Microstrip Resonators. *IEEE Transactions on Microwave Theory and Techniques, 23*(12), 990-997. doi: 10.1109/TMTT.1975.1128732

Amri, A., Saidane, A., & Pulko, S. (2011). Thermal analysis of a three-dimensional breast model with embedded tumour using the transmission line matrix (TLM) method. *Computers in Biology and Medicine, 41*(2), 76-86. doi: http://dx.doi.org/10.1016/j.compbiomed.2010.12.002

Arfken, G. B., Weber, H. J., & Harris, F. E. (2013). Chapter 9 - Partial Differential Equations *Mathematical Methods for Physicists (Seventh Edition)* (pp. 401-445). Boston: Academic Press.

Barkley Rosser, J. (1975). Nine-point difference solutions for Poisson's equation. *Computers & Mathematics with Applications, 1*(3), 351-360. doi: http://dx.doi.org/10.1016/0898-1221(75)90035-8

Bednar, J. A. (2012). Building a mechanistic model of the development and function of the primary visual cortex. *Journal of Physiology-Paris, 106*(5-6), 194-211. doi: 10.1016/j.jphysparis.2011.12.001

Beurle, R. L. (1956). Properties of a Mass of Cells Capable of Regenerating Pulses. *Philosophical Transactions of the Royal Society of London. Series B, Biological Sciences, 240*(669), 55-94.

Bower, J. M., & Beeman, D. (1998). *The book of GENESIS (2nd ed.): exploring realistic neural models with the GEneral NEural SImulation System*: Springer-Verlag New York, Inc.

Breakspear, M., Jirsa, V., & Deco, G. (2010). Computational models of the brain: from structure to function. *Neuroimage, 52*(3), 727-730. doi: 10.1016/j.neuroimage.2010.05.061

Brette, R., Rudolph, M., Carnevale, T., Hines, M., Beeman, D., Bower, J. M., . . . Destexhe, A. (2007). Simulation of networks of spiking neurons: a review of tools and strategies. *Journal of Computational Neuroscience, 23*(3), 349-398. doi: 10.1007/s10827-007-0038-6

Carnevale, N. T., & Hines, M. L. (2006). *The NEURON Book*: Cambridge University Press.

Ciocan, R., & Ida, N. (2003). *Applications of Transmission Line Matrix Method For NDT* (Vol. 8 No.3).

NDT.net.

Cogan, D. d., O'Connor, W. J., & Pulko, S. (2005). *Transmission Line Matrix (TLM) in Computational Mechanics*: CRC Press, Inc.

Coombes, S. (2010). Large-scale neural dynamics: Simple and complex. *Neuroimage, 52*(3), 731-739. doi: http://dx.doi.org/10.1016/j.neuroimage.2010.01.045

Coombes, S., & Byrne, A. (2016). *Next generation neural mass models*: In A. Torcini and F. Corinth, editors, Lecture Notes in Nonlinear Dynamics in Computational Neuroscience: from Physics and Biology to ICT. Springer.

Courant, R., Friedrichs, K., & Lewy, H. (1967). On the partial difference equations of mathematical physics. *IBM J. Res. Dev., 11*(2), 215-234. doi: 10.1147/rd.112.0215

Cst.com. (2015). TLM Solver of CST MICROWAVE STUDIO®. Retrieved 05/11/2015, from https://www.cst.com/Products/CSTMWS/TLM-Solver

David, O., & Friston, K. J. (2003). A neural mass model for MEG/EEG:: coupling and neuronal dynamics. *Neuroimage, 20*(3), 1743-1755. doi: https://doi.org/10.1016/j.neuroimage.2003.07.015

Dayan, P., & Abbott, L. F. (2001). *Theoretical Neuroscience : Computational and Mathematical Modeling of Neural Systems*. Cambridge, Mass.: Massachusetts Institute of Technology Press.

Deco, G., Jirsa, V. K., Robinson, P. A., Breakspear, M., & Friston, K. (2008). The Dynamic Brain: From Spiking Neurons to Neural Masses and Cortical Fields. *PLoS Comput Biol, 4*(8), e1000092. doi: 10.1371/journal.pcbi.1000092

Deford, J. F., & Gandhi, O. P. (1985). An Impedance Method to Calculate Currents Induced in Biological Bodies Exposed to Quasi-Static Electromagnetic Fields. *IEEE Transactions on Electromagnetic Compatibility, EMC-27*(3), 168-173. doi: 10.1109/TEMC.1985.304281

Desai, R. A., Lowery, A. J., Christopoulos, C., Naylor, P., Blanshard, J. M. V., & Gregson, K. (1992). Computer modelling of microwave cooking using the transmission-line model. *IEE Proceedings A - Science, Measurement and Technology, 139*(1), 30-38. doi: 10.1049/ip-a-3.1992.0005

Eckhorn, R., Reitboeck, H., Arndt, M., & Dicke, P. (1990). Feature Linking via Synchronization among Distributed Assemblies: Simulations of Results from Cat Visual Cortex. *Neural Comput, 2*(3), 293-307. doi: 10.1162/neco.1990.2.3.293

Eliasmith, C., Stewart, T. C., Choo, X., Bekolay, T., DeWolf, T., Tang, Y., & Rasmussen, D. (2012). A Large-Scale Model of the Functioning Brain. *Science, 338*(6111), 1202-1205. doi: 10.1126/science.1225266

Erwin, E., Obermayer, K., & Schulten, K. (1995). Models of orientation and ocular dominance columns in the visual cortex: a critical comparison. *Neural Comput, 7*(3), 425-468.

Freeman, W. (2012). *Neurodynamics: An Exploration in Mesoscopic Brain Dynamics*: Springer London.

Friston, K. J., & Dolan, R. J. (2010). Computational and dynamic models in neuroimaging. *Neuroimage, 52*(3), 752-765. doi: http://dx.doi.org/10.1016/j.neuroimage.2009.12.068

Gerstner, W., & Kistler, W. M. (2002). *Spiking neuron models : single neurons, populations, plasticity*. Cambridge: Cambridge University Press.

Gewaltig, M., & Diesmann, M. (2007). NEST (NEural Simulation Tool). *Scholarpedia, 2*, 1430.

Goodman, D. F. M., & Brette, R. (2009). The Brian simulator. *Frontiers in Neuroscience, 3*. doi: 10.3389/neuro.01.026.2009

Hodgkin, A. L., & Huxley, A. F. (1952a). Currents carried by sodium and potassium ions through the membrane of the giant axon of Loligo. *The Journal of Physiology, 116*(4), 449-472.

Hodgkin, A. L., & Huxley, A. F. (1952b). The components of membrane conductance in the giant axon of Loligo. *The Journal of Physiology, 116*(4), 473-496.

Hodgkin, A. L., & Huxley, A. F. (1952c). The dual effect of membrane potential on sodium conductance in the giant axon of Loligo. *The Journal of Physiology, 116*(4), 497-506.

Hodgkin, A. L., & Huxley, A. F. (1952d). A quantitative description of membrane current and its application to conduction and excitation in nerve. *The Journal of Physiology, 117*(4), 500-544.

Hoefer, W. J. R. (1985). The Transmission-Line Matrix Method - Theory and Applications. *Microwave Theory and Techniques, IEEE Transactions on, 33*(10), 882-893. doi: 10.1109/TMTT.1985.1133146

Hoefer, W. J. R. (2012, 21-24 May 2012). *A history of time domain electromagnetics - a voyage back in time.* Paper presented at the Electromagnetic Compatibility (APEMC), 2012 Asia-Pacific Symposium on.

Hughes, J. R. (2008). Gamma, fast, and ultrafast waves of the brain: their relationships with epilepsy and behavior. *Epilepsy Behav, 13*(1), 25-31. doi: 10.1016/j.yebeh.2008.01.011

Huygens, C. (1690). *Traité de la lumière: Ou sont expliquées les causes de ce qui luy arrive dans la reflexion, et dans la refraction Et particulierement dans l'etrange refraction du cristal d'Islande ; Avec un discours de la cause de la pesanteur*: van der Aa.

Johns, P. B. (1972). Application of the transmission-line-matrix method to homogeneous waveguides of arbitrary cross-section. *Electrical Engineers, Proceedings of the Institution of, 119*(8), 1086-1091. doi: 10.1049/piee.1972.0206

Johns, P. B. (1977). A simple explicit and unconditionally stable numerical routine for the solution of the diffusion equation. *International Journal for Numerical Methods in Engineering, 11*(8), 1307-1328. doi: 10.1002/nme.1620110810

Johns, P. B. (1987). On the Relationship Between TLM and Finite-Difference Methods for Maxwell's Equations (Short Paper). *IEEE Transactions on Microwave Theory and Techniques, 35*(1), 60-61. doi: 10.1109/TMTT.1987.1133595

Johns, P. B., & Beurle, R. L. (1971). Numerical solution of 2-dimensional scattering problems using a transmission-line matrix. *Electrical Engineers, Proceedings of the Institution of, 118*(9), 1203-1208. doi: 10.1049/piee.1971.0217

Johns, P. B., & Brien, M. O. (1980). Use of the transmission-line modelling (t.l.m.) method to solve non-linear lumped networks. *Radio and Electronic Engineer, 50*(1.2), 59-70. doi: 10.1049/ree.1980.0006

Kane, Y. (1966). Numerical solution of initial boundary value problems involving maxwell's equations in isotropic media. *IEEE Transactions on Antennas and Propagation, 14*(3), 302-307. doi: 10.1109/TAP.1966.1138693

Kerr, C. C., Rennie, C. J., & Robinson, P. A. (2011). Model-based analysis and quantification of age trends in auditory evoked potentials. *Clin Neurophysiol, 122*(1), 134-147. doi: 10.1016/j.clinph.2010.05.030

Kerr, C. C., Van Albada, S. J., Neymotin, S. A., Chadderdon, G. L., Robinson, P. A., & Lytton, W. W. (2013). Cortical information flow in Parkinson's disease: a composite network/field model. *Front Comput Neurosci, 7*, 39. doi: 10.3389/fncom.2013.00039

Krumpholz, M., Huber, C., & Russer, P. (1995). A field theoretical comparison of FDTD and TLM. *Microwave Theory and Techniques, IEEE Transactions on, 43*(8), 1935-1950. doi: 10.1109/22.402284

Krumpholz, M., & Russer, P. (1994). A field theoretical derivation of TLM. *Microwave Theory and Techniques, IEEE Transactions on, 42*(9), 1660-1668. doi: 10.1109/22.310559

Lowery, A. J. (1989). Transmission-line modelling of semiconductor lasers: The transmission-line laser model. *International Journal of Numerical Modelling: Electronic Networks, Devices and Fields, 2*(4), 249-265. doi: 10.1002/jnm.1660020408

McLaughlin, D., Shapley, R., Shelley, M., & Wielaard, D. J. (2000). A neuronal network model of macaque primary visual cortex (V1): Orientation selectivity and dynamics in the input layer 4Cα. *Proceedings of the National Academy of Sciences, 97*(14), 8087-8092. doi: 10.1073/pnas.110135097

Mingus, B. (2014). Comparison of Neural Network Simulators. Retrieved 05/11/2015, from https://grey.colorado.edu/emergent/index.php/Comparison_of_Neural_Network_Simulators

Moran, R. J., Kiebel, S. J., Stephan, K. E., Reilly, R. B., Daunizeau, J., & Friston, K. J. (2007). A neural mass model of spectral responses in electrophysiology. *Neuroimage, 37*(3-3), 706-720. doi: 10.1016/j.neuroimage.2007.05.032

Northrop, R. B. (2001). *Introduction to dynamic modeling of neuro-sensory systems*. Boca Raton, FL: CRC Press.

Nunez, P. L. (1974). The brain wave equation: a model for the EEG. *Mathematical Biosciences, 21*(3–4), 279-297. doi: http://dx.doi.org/10.1016/0025-5564(74)90020-0

Nunez, P. L., & Srinivasan, R. (2006). *Electric fields of the brain : the neurophysics of EEG* (2nd ed. ed.). Oxford

New York: Oxford University Press.
Olsen-Kettle, L. (2011). *Numerical solution of partial differential equations and code*: Earth Systems Science Computational Centre Publications.
P. Sanz-Leon, P. A. R., S. A. Knock, R. G. Abeysuriya, P. M. Drysdale, P. K. Fung, C. J. Rennie and X. Zhao. (2017). NeuroField: A software for neural field simulations in C++. *In preparation for PlosCB*.
Pinotsis, D., Robinson, P., beim Graben, P., & Friston, K. (2014). Neural masses and fields: modeling the dynamics of brain activity. *Front Comput Neurosci, 8*, 149. doi: 10.3389/fncom.2014.00149
Portí, J. A., & Morente, J. A. (2001). TLM method and acoustics. *International Journal of Numerical Modelling: Electronic Networks, Devices and Fields, 14*(2), 171-183. doi: 10.1002/jnm.405
Press, W. H., Teukolsky, S. A., Vetterling, W. T., & Flannery, B. P. (1992). *Numerical recipes in C (2nd ed.): the art of scientific computing*: Cambridge University Press.
Rall, W. (2011). Core Conductor Theory and Cable Properties of Neurons *Comprehensive Physiology*: John Wiley & Sons, Inc.
Ray, K., & Roy, M. K. (2010, 7-9 July 2010). *A theoretical basis for brain waves with implications for a large scale integration required for cognitive processes.* Paper presented at the Cognitive Informatics (ICCI), 2010 9th IEEE International Conference on.
Rennie, C. J., Robinson, P. A., & Wright, J. J. (2002). Unified neurophysical model of EEG spectra and evoked potentials. *Biol Cybern, 86*(6), 457-471. doi: 10.1007/s00422-002-0310-9
Rezzolla, L., & Zanotti, O. (2014). *Relativistic hydrodynamics*. Oxford: Oxford University Press.
Ritter, P., Schirner, M., McIntosh, A. R., & Jirsa, V. K. (2013). The Virtual Brain Integrates Computational Modeling and Multimodal Neuroimaging. *Brain Connect, 3*(2), 121-145. doi: 10.1089/brain.2012.0120
Roberts, J. A., & Robinson, P. A. (2012). Corticothalamic dynamics: Structure of parameter space, spectra, instabilities, and reduced model. *Physical Review E, 85*(1), 011910.
Robinson, P. A. (2014). Determination of effective brain connectivity from functional connectivity using propagator-based interferometry and neural field theory with application to the corticothalamic system. *Physical Review E, 90*(4). doi: 10.1103/PhysRevE.90.042712
Robinson, P. A., & Kim, J. W. (2012). Spike, rate, field, and hybrid methods for treating neuronal dynamics and interactions. *J Neurosci Methods, 205*(2), 283-294. doi: 10.1016/j.jneumeth.2012.01.018
Robinson, P. A., Rennie, C. J., Rowe, D. L., & O'Connor, S. C. (2004a). Estimation of multiscale neurophysiologic parameters by electroencephalographic means. *Hum Brain Mapp, 23*(1), 53-72. doi: 10.1002/hbm.20032
Robinson, P. A., Rennie, C. J., Rowe, D. L., & O'Connor, S. C. (2004b). Estimation of multiscale neurophysiologic parameters by electroencephalographic means. *Hum Brain Mapp, 23*(1), 53-72. doi: 10.1002/hbm.20032
Robinson, P. A., Rennie, C. J., Rowe, D. L., O'Connor, S. C., & Gordon, E. (2005). Multiscale brain modelling. *Philosophical Transactions of the Royal Society B: Biological Sciences, 360*(1457), 1043-1050. doi: 10.1098/rstb.2005.1638
Robinson, P. A., Rennie, C. J., & Wright, J. J. (1997). Propagation and stability of waves of electrical activity in the cerebral cortex. *Physical Review E, 56*(1), 826-840.
Robinson, P. A., Rennie, C. J., Wright, J. J., Bahramali, H., Gordon, E., & Rowe, D. L. (2001). Prediction of electroencephalographic spectra from neurophysiology. *Physical Review E, 63*(2), 021903.
Robinson, P. A., Sarkar, S., Pandejee, G. M., & Henderson, J. A. (2014). Determination of effective brain connectivity from functional connectivity with application to resting state connectivities. *Physical Review E, 90*(1). doi: 10.1103/PhysRevE.90.012707
Rulkov, N. F., Timofeev, I., & Bazhenov, M. (2004). Oscillations in Large-Scale Cortical Networks: Map-Based Model. *Journal of Computational Neuroscience, 17*(2), 203-223. doi: 10.1023/B:JCNS.0000037683.55688.7e

Russer, P. (2000). The Transmission Line Matrix Method. In N. Uzunoglu, K. Nikita & D. Kaklamani (Eds.), *Applied Computational Electromagnetics* (Vol. 171, pp. 243-269): Springer Berlin Heidelberg.

Sadiku, M. N. O. (2009). *Numerical Techniques in Electromagnetics with MATLAB, Third Edition*: Taylor & Francis.

Schellenberger Costa, M., Weigenand, A., Ngo, H.-V. V., Marshall, L., Born, J., Martinetz, T., & Claussen, J. C. (2016). A Thalamocortical Neural Mass Model of the EEG during NREM Sleep and Its Response to Auditory Stimulation. *PLoS Comput Biol, 12*(9), e1005022. doi: 10.1371/journal.pcbi.1005022

Stewart, T. C., Bekolay, T., & Eliasmith, C. (2012). Learning to Select Actions with Spiking Neurons in the Basal Ganglia. *Frontiers in Neuroscience, 6*, 2. doi: 10.3389/fnins.2012.00002

Stewart, T. C., & Eliasmith, C. (2014). Large-Scale Synthesis of Functional Spiking Neural Circuits. *Proceedings of the IEEE, 102*(5), 881-898. doi: 10.1109/JPROC.2014.2306061

Tapson, J. C., Cohen, G. K., Afshar, S., Stiefel, K. M., Buskila, Y., Hamilton, T. J., & van Schaik, A. (2013). Synthesis of neural networks for spatio-temporal spike pattern recognition and processing. *Frontiers in Neuroscience, 7*. doi: 10.3389/fnins.2013.00153

Thom, A. (1961). *Field computations in engineering & physics [by] A. Thom and C.J. Apelt*. London, New York: Van Nostrand.

van Albada, S. J., Gray, R. T., Drysdale, P. M., & Robinson, P. A. (2009). Mean-field modeling of the basal ganglia-thalamocortical system. II Dynamics of parkinsonian oscillations. *J Theor Biol, 257*(4), 664-688. doi: 10.1016/j.jtbi.2008.12.013

van Albada, S. J., Kerr, C. C., Chiang, A. K., Rennie, C. J., & Robinson, P. A. (2010). Neurophysiological changes with age probed by inverse modeling of EEG spectra. *Clin Neurophysiol, 121*(1), 21-38. doi: 10.1016/j.clinph.2009.09.021

van Albada, S. J., & Robinson, P. A. (2009). Mean-field modeling of the basal ganglia-thalamocortical system. I Firing rates in healthy and parkinsonian states. *J Theor Biol, 257*(4), 642-663. doi: 10.1016/j.jtbi.2008.12.018

Velut, S., & Tummescheit, H. (2011). *Implementation of a transmission line model for fast simulation of fluid flow dynamics*.

Villapecellin-Cid, M. M., Rao, L., & Reina-Tosina, J. (2003, 17-21 Sept. 2003). *Ranvier nodes impedance match with internodal transmission lines of myelinated axons.* Paper presented at the Engineering in Medicine and Biology Society, 2003. Proceedings of the 25th Annual International Conference of the IEEE.

Villapecellin-Cid, M. M., Roa, L. M., & Reina-Tosina, J. (2001, 2001). *Transverse magnetic waves in myelinated nerves.* Paper presented at the Engineering in Medicine and Biology Society, 2001. Proceedings of the 23rd Annual International Conference of the IEEE.

Villapecellin-Cid, M. M., Roa, L. M., & Reina-Tosina, J. (2002, 23-26 Oct. 2002). *Electromagnetic fields induced in anisotropic tissues by myelinated axons.* Paper presented at the Engineering in Medicine and Biology, 2002. 24th Annual Conference and the Annual Fall Meeting of the Biomedical Engineering Society EMBS/BMES Conference, 2002. Proceedings of the Second Joint.

Weiner, M. (2010). *Electromagnetic Analysis Using Transmission Line Variables*: World Scientific.

Weiner, M. (2010). Introduction to Transmission Lines and Their Application to Electromagnetic Phenomena *Electromagnetic Analysis Using Transmission Line Variables* (2nd ed., pp. 1-63): WORLD SCIENTIFIC.

Wells, R. B. (2005). Cortical Neurons and Circuits: A tutorial introduction. LCNTR Tech Brief, Moscow: The University of Idaho, http://www.mrc.uidaho.edu/~rwells/techdocs/.

Wilson, H. R. (1999). Simplified Dynamics of Human and Mammalian Neocortical Neurons. *J Theor Biol, 200*(4), 375-388. doi: http://dx.doi.org/10.1006/jtbi.1999.1002

Wilson, H. R., & Cowan, J. D. (1973). A mathematical theory of the functional dynamics of cortical and thalamic nervous tissue. *Kybernetik, 13*(2), 55-80. doi: 10.1007/BF00288786

Wright, J. J., & Liley, D. T. J. (1995). Simulation of electrocortical waves. *Biological Cybernetics, 72*(4), 347-356. doi: 10.1007/BF00202790

Wright, J. J., & Liley, D. T. J. (1996). Dynamics of the brain at global and microscopic scales: Neural networks and the EEG. *Behavioral and Brain Sciences, 19*(02), 285-295. doi: doi:10.1017/S0140525X00042679

Wu, H., & Robinson, P. A. (2007). Modeling and investigation of neural activity in the thalamus. *J Theor Biol, 244*(1), 1-14. doi: 10.1016/j.jtbi.2006.07.016

Yamaguchi, I., Ogawa, Y., Nakao, H., Jimbo, Y., & Kotani, K. (2014). Linear Analysis of the Corticothalamic Model with Time Delay. *Electronics and Communications in Japan, 97*(8), 32-44. doi: 10.1002/ecj.11581

Yi-Chi, S., & Hoefer, W. J. R. (1980). Dominant and Second-Order Mode Cutoff Frequencies in Fin Lines Calculated with a Two-Dimensional TLM Program. *IEEE Transactions on Microwave Theory and Techniques, 28*(12), 1443-1448. doi: 10.1109/TMTT.1980.1130264

Zachary, K., Richard, H. M., & Cutter, A. G. (2006). Evaluation of the Nash–Sutcliffe Efficiency Index. doi: 10.1061/(ASCE)1084-0699(2006)11:6(597)