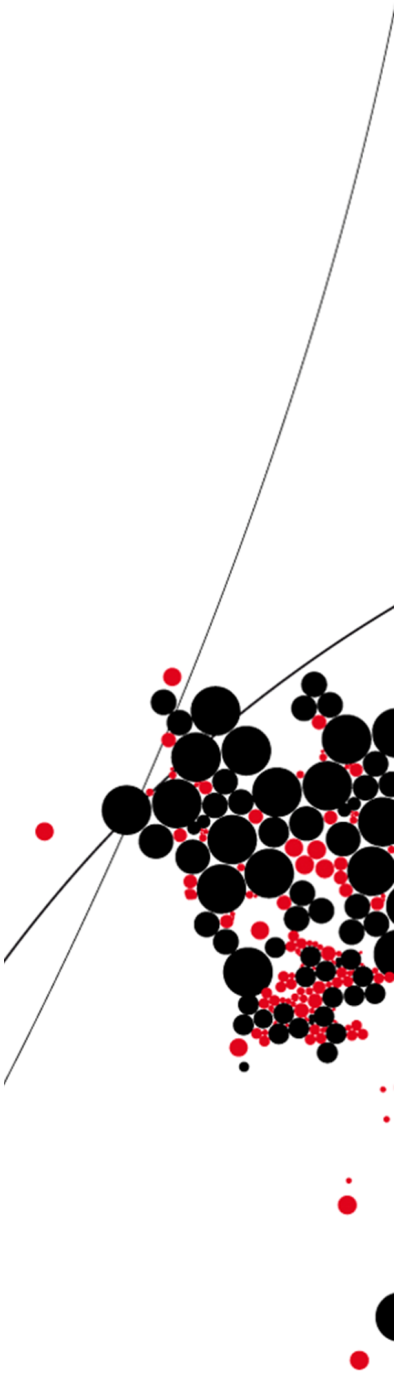
UNIVERSITY OF TWENTE.

Faculty of Electrical Engineering, Mathematics & Computer Science



Solution Techniques for Inverse Problems in Neural Field Theory

Nick Luiken

MSc Thesis in Applied Mathematics
October 2016

Assessment Committee:
Dr. C. Brune
Prof. dr. S.A. van Gils
Dr. P.K. Mandal

Supervisor:
Dr. C. Brune

Applied Analysis Group
Department of Applied Mathematics
Faculty of Electrical Engineering,
Mathematics and Computer Science
University of Twente
P.O. Box 217
7500 AE Enschede
The Netherlands

Abstract

In this thesis we present new solution techniques for inverse problems in neural field theory. Neural fields are a continuum limit of neural networks and describe the spatiotemporal evolution of neural activity in the brain. This evolution is described by an integro-differential equation called the Amari equation. One of the inverse problems in neural field theory is to describe the connections between neurons based on this spatiotemporal evolution. The inverse problem is ill-posed. In other work, this ill-posedness was dealt with using Tikhonov regularization. We present three methods that reduce the ill-posedness of the problem and improve the quality of the reconstruction. We compare these methods to the use of Tikhonov regularization and also show what happens when we combine these methods with Tikhonov regularization. The first method we use is parameter optimization. We show that parameter optimization is necessary when dealing with data generated for fixed parameters. The second method we introduce is subsampling. We show that subsampling is a tool to reduce the error of the reconstruction and reduce the ill-posedness. At some point we reach a trade-off between accuracy and stability. We furthermore show that a combination of subsampling and Tikhonov regularization is sometimes the best method. The third method we present is combining data. Sometimes we are dealing with insufficiently informative data. To overcome this, we can combine data that is qualitatively different.

Keywords: neural fields, Amari equation, inverse problem, integro-differential equation, regularization, subsampling

Acknowledgements

This thesis is my final work as a student Applied Mathematics at the University of Twente. With it, my student life comes to an end and there is a number of people I would like to thank for their support during this time.

First of all I would like to thank my supervisors Dr. Christoph Brune and Prof. dr. Stephan van Gils for helping me the past 6 months. They have done a great job supporting me, motivating me and advising me on this work. We have had many long discussions about my research and they have always been very helpful. I would like to thank Christoph as well for recommending me for the PhD position in Utrecht which I will take next year. I am sure we will work together at some point during my PhD research as well and I look forward to it.

I would also like to thank Prof. dr. Roland Potthast. He was my supervisor during my time as an intern at DWD in Germany. He has done a tremendous job of supervising me there and advised me to pursue a PhD position. This has led me to pursuing this myself and I am very grateful to him. He is also the one who has laid the groundwork for inverse problems in neural field theory, on which we have based this work. It is safe to say I owe him a great deal.

I thank Yoeri Boink for the discussions we have had and the collaboration over the last two years.

I thank Andre and Martin for their friendship and support. A special thanks goes out to Andre, who motivates me to work hard and perform to the best of my abilities.

Last but not least, I thank my parents and my sister. Without them, none of this would have been possible.

Contents

Chapter 1

Introduction

1.1 Introduction

In this thesis we study inverse problems in neural field theory. Neural fields are the continuum limits of neural networks. A neural network consists of a large amount of neurons. These neurons are cells that are electrically excitable and transmit information in our brain. They do so via synapses, which are structures that permit a neuron to pass this signal. Due to the large amount of neurons in the brain, it is tempting to derive a continuum limit of these neural networks [9], [10], [11]. This means that we assume there are infinitely many neurons connected to each other. The work was later extended by Wilson and Cowan [12], [13] to include excitation and inhibition as well as refractoriness. This work was later extended by Amari [3], [4], who introduced the *Amari equation*.

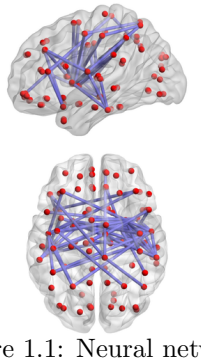


Figure 1.1: Neural network

The interactions within the neural field form a pattern. A description of these patterns is given by the *synaptic weight kernel*. The inverse problem in neural field theory is to determine the synaptic weight kernel based on observations of the neural field. The first study of the inverse problem was done by Potthast and beim Graben, [1], [2]. In this thesis, we build upon their work and study the inverse problem.

1.1.1 Mathematical description of the inverse problem

In this work we study the inverse problem in the *Amari equation*, given by:

$$\tau \frac{\partial u(x,t)}{\partial t} + u(x,t) = \int_{\Omega} w(x,y) f(u(y,t)) dy, \qquad x, y \in \Omega, t > 0, \tag{1.1}$$

for a given initial condition $u(x,0) = u_0(x)$. In the Amari equation the variable $u(x,t)$ describes the spatiotemporal evolution of the activity of the neurons. The function $f(u)$ is called the *firing rate function* and is usually given by:

$$f(u) = \frac{1}{1 + e^{-\beta(u-\eta)}} - c. \tag{1.2}$$

The constant c is either 0 or $\frac{1}{2}$. Choosing $c = \frac{1}{2}$ ensures that $f(0) = 0$, which means that the background is set to 0 [6]. In this case the parameter $\eta = 0$. If $c = 0$ then the parameter η represents a thresholding parameter. The parameter β is a measure for the variability. The function $w(x,y)$ is called the *synaptic weight kernel* and describes the connectivity between points $x, y \in \mathbb{R}^m$ in a domain $\Omega \in \mathbb{R}^m$, with $m \in \mathbb{N}$. Positive values in the weight kernel correspond to excitatory connections and negative values correspond to inhibitory connections.

1.1.2 Forward and inverse problem

A solution to the neural field equation given a certain initial condition, firing rate function and synaptic weight kernel and studying it's properties is known as the **forward problem**. Here, the **forward problem** is to generate u via equation 1.1 given 1.2. In an **inverse problem** the objective is to estimate the unknown parameters or variables in the **forward problem**, given certain data. We now state the **inverse problem** for the neural field equation, as described in [1].

> **The inverse problem**. Given a function $u(x,t) \in BC(D) \times BC^1([0,T])$ for some $T \in \mathbb{R}^+ \cup \infty$ and with $u(x,0) = u_0(x)$, the inverse problem is to construct a kernel $w(x,y)$ in X such that the solution of the neural field equation 1.1 is satisfied.

In order to solve the inverse problem, we first rewrite the neural field equation. Define the *synaptic weight operator* as:

$$(W\varphi)(x) := \int_D w(x,y)\varphi(y) dy, \qquad x, y \in D.$$

Then by defining the functions:

$$\psi(x,t) := \tau \frac{\partial u}{\partial t}(x,t) + u(x,t),$$
$$\varphi(x,t) := f(u(x,t)),$$

we can rewrite the neural field equation as:

$$\psi = W\varphi \tag{1.3}$$

Here, W is an integral operator with kernel w. The discretized version of 1.3 is given by:

$$\boldsymbol{\psi}^j = \mathbf{W}\boldsymbol{\varphi}^j. \tag{1.4}$$

Here, $\mathbf{W} \in \mathbb{R}^{n\times n}$, $\boldsymbol{\psi}^j, \boldsymbol{\varphi}^j \in \mathbb{R}^n$ for $j \in J \subset \mathbb{N}$ and $|J| = n$.
In [1] Potthast and beim Graben have shown that this problem is generally ill-posed. They have shown through several examples how regularization can be used to overcome some of the difficulties. In these examples we have to distinguish two different cases.

1. **Prescribed sequences.** In this case a certain sequence $u(x,t)$ is prescribed and we reconstruct the kernel using this sequence. After that we use the reconstructed kernel W_α to reconstruct the sequence $u_\alpha(x,t)$. In reconstructing $u_\alpha(x,t)$ we change the discretization over time to avoid the inverse crime. We then compare the sequences $u(x,t)$ and $u_\alpha(x,t)$ to verify the quality of W_α.

2. **Prescribed kernels.** In this case we prescribe a kernel W and generate a sequence $u(x,t)$ for a given $u_0(x)$. We use this data to reconstruct the kernel W_α and compare it to W.

In this work we will build upon the work of Potthast and beim Graben in [1]. We will use the solution procedure described by them and reconstruct their examples. We will show new solution techniques that we have developed to improve the quality of the reconstruction, be it sequence or kernel, and reduce the ill-posedness. Furthermore, we have included more examples illustrating the difficulties that arise in inverse problems for neural fields, and how our solution techniques help to overcome them.

Chapter 2

Mathematical framework

In this section we will describe the mathematical framework of the problem. We start by discussing the construction of solutions and the solution procedure we apply, following [1]. Then we present our solution techniques in addition to this and comment on the validation of our methods.

2.1 Construction of solutions

In this section we describe the construction of solutions following [1]. The goal of this section is to use biorthogonal sets to construct solutions and to investigate the ill-posedness of the problem. We recite the construction of biorthogonal sets from [1] and comment on the ill-posedness of the problem.
We assume we have a Hilbert space X with scalar product $\langle \cdot, \cdot \rangle$. Two linearly independent sets of functions $Q = \{\varphi_1, \varphi_2, ...\}$ and $R = \{\rho_1, \rho_2, ...\}$ are biorthogonal if

$$\langle \rho_i, \varphi_k \rangle = 0 \quad \forall \quad k \neq i, \quad \langle \rho_i, \varphi_i \rangle = c_i, \quad c_i \neq 0, \quad i \in \mathbb{N}.$$

Define

$$V_k = \{\varphi_1, \dots, \varphi_{k-1}, \varphi_{k+1}, \dots\}, \quad k \in \mathbb{N}. \tag{2.1}$$

We then have $X = \overline{V_k} \oplus V_k^{\perp}$. Denote the orthogonal projection of φ_k onto $V_k^{\perp}$ by $\tilde{\rho}_k$. The biorthogonal elements are then given by

$$\rho_k := \frac{\tilde{\rho}_k}{\|\tilde{\rho}_k\|^2}. \tag{2.2}$$

Writing $\bar{\varphi}_k = \tilde{\rho}_k + \varphi_k$ with $\varphi_k \in V_k$ we find

$$\langle \tilde{\rho}_k, \bar{\varphi}_k \rangle = \langle \tilde{\rho}_k, \tilde{\rho}_k + \varphi_k \rangle = \langle \tilde{\rho}_k, \tilde{\rho}_k \rangle = \|\tilde{\rho}_k\|^2. \tag{2.3}$$

If $\sum_{j=1}^{n} \beta_j \rho_j = 0$, then we have

$$0 = \left\langle \sum_{j=1}^{n} \beta_j \rho_j, \varphi_k \right\rangle = \sum_{j=1}^{n} \beta_j \langle \rho_j, \varphi_k \rangle = \beta_k, \quad k = 1, \dots, n \tag{2.4}$$

and hence the set $R = \{\rho_1, \rho_2, ...\}$ is linearly independent. Q is called a Riesz basis in a Hilbert space H if there are constants $c_1, c_2 > 0$ such that

$$c_1 \sum_{j=1}^{\infty} |\alpha_j|^2 \leq \left\| \sum_{j=1}^{\infty} \alpha_j \varphi_j \right\|^2 \leq c_2 \sum_{j=1}^{\infty} |\alpha_j|^2. \tag{2.5}$$

for all $\alpha = (\alpha_j)_{j\in\mathbb{N}} \in \ell^2$. In this case the mapping

$$A : \ell^2 \to X, \quad \alpha \mapsto \sum_{j=1}^{\infty} \alpha_j \varphi_j \tag{2.6}$$

is a bounded and invertible mapping from ℓ^2 onto $A(\ell^2)$. The dual operator is then given by

$$A' : X \to \ell^2, \quad \psi \mapsto \left(\langle \varphi_j, \psi \rangle_X\right)_{j\in\mathbb{N}}. \tag{2.7}$$

The following estimate then holds:

$$\langle \alpha, A'A\alpha \rangle_{\ell^2} = \langle A\alpha, A\alpha \rangle_X \geq c_1 \|\alpha\|_{\ell^2}^2 \tag{2.8}$$

According to the Lax-Milgram theorem $A'A$ is boundedly invertible in ℓ^2 with a lower bound given by $\dfrac{1}{c_1}$. The operation of $A'A$ on α is given by

$$A'A\alpha = \left(\langle \varphi_k, A\alpha \rangle_X\right)_{k\in\mathbb{N}} = \left(\sum_{j=1}^{\infty} \langle \varphi_k, \varphi_j \rangle_X \alpha_j \right)_{k\in\mathbb{N}} \tag{2.9}$$

The operation $A'A$ on ℓ^2 can then be expressed as a matrix multiplication with M, where M is defined as

$$M := \left(\langle \varphi_k, \varphi_j \rangle\right)_{k,j\in\mathbb{N}} \tag{2.10}$$

If we define

$$\rho_k := \sum_{j=1}^{\infty} (M^{-1})_{k,j} \varphi_j, \quad k \in \mathbb{N}, \tag{2.11}$$

we get

$$\langle \rho_k, \varphi_i \rangle_X = \left\langle \sum_{j=1}^{\infty} (M^{-1})_{k,j} \varphi_j, \varphi_i \right\rangle = \sum_{j=1}^{\infty} (M^{-1})_{k,j} \langle \varphi_j, \varphi_i \rangle = (M^{-1}M)_{k,i} = \delta_{ki}. \tag{2.12}$$

Hence we have a constructive method for calculating the biorthogonal set. We can now define operators V_n and V

$$V_n \varphi := \sum_{i=1}^{n} \psi_i \langle \rho_i, \varphi \rangle, \qquad V\varphi := \sum_{i=1}^{\infty} \psi_i \langle \rho_i, \varphi \rangle \tag{2.13}$$

such that

$$V_n\varphi_i = \begin{cases} \psi_i & \text{if } i = 1, .., n \\ 0, & \text{if } i > n \end{cases} \tag{2.14}$$

$$V\varphi_i = \psi_i, \quad i \in \mathbb{N} \tag{2.15}$$

2.1.1 Ill-posedness of the problem

In [1] it's stated that the division by $\|\tilde{\rho}_k\|^2$ in 2.3 leads to the ill-posedness of the problem. We have that:

$$\|\rho_k\| = \frac{\|\tilde{\rho}_k\|}{\|\tilde{\rho}_k\|^2} = \frac{1}{\|\tilde{\rho}_k\|}. \tag{2.16}$$

Using 2.11 we get that:

$$\begin{aligned}
\|\rho_k\|^2 &= \langle \rho_k, \rho_k \rangle \\
&= \left\langle \sum_{j=1}^{\infty} (M^{-1})_{k,j}\varphi_j, \sum_{l=1}^{\infty} (M^{-1})_{k,l}\varphi_l \right\rangle \\
&= \sum_{j=1}^{\infty} (M^{-1})_{k,j} \left\langle \varphi_j, \sum_{l=1}^{\infty} (M^{-1})_{k,l}\varphi_l \right\rangle \\
&= \sum_{j=1}^{\infty} (M^{-1})_{k,j} \sum_{l=1}^{\infty} (M^{-1})_{k,l} \langle \varphi_l, \varphi_j \rangle \\
&= \sum_{j=1}^{\infty} (M^{-1})_{k,j} \sum_{l=1}^{\infty} (M^{-1})_{k,l} M_{l,j} \\
&= \sum_{j=1}^{\infty} (M^{-1})_{k,j} (M^{-1}M)_{k,j} \\
&= \sum_{j=1}^{\infty} (M^{-1})_{k,j} \delta_{kj} \\
&= (M^{-1})_{k,k}
\end{aligned}$$

We now have that:

$$\|\tilde{\rho}_k\| = \frac{1}{\sqrt{(M^{-1})_{k,k}}} \tag{2.17}$$

From 2.2 we see that:

$$\rho_k = \frac{\tilde{\rho}_k}{\|\tilde{\rho}_k\|^2} = \tilde{\rho}_k \cdot \left(M^{-1}\right)_{k,k} \tag{2.18}$$

Hence we see that the elements ρ_k are dependent on the invertibility of M. If M is not invertible then the ρ_k are not well-defined and we are dealing with

an ill-posed inverse problem, because we can no longer construct a stable and unique solution.

2.2 Solution procedure

In this section we will briefly reiterate the solution procedure used by Potthast and beim Graben in [1].
In the examples we employ an explicit Euler scheme for the time derivative and the rectangular rule for the integral. For the spatial domain $D = [a_1, b_1] \times [a_2, b_2] \times ... \times [a_N, b_N] \subset \mathbb{R}^N$ we can define a grid by setting

$$h_k := \frac{b_k - a_k}{n_k - 1}, \qquad k = 1, ..., N,$$

and

$$x_{k_1,...,k_N} := (a_1 + k_1 \cdot h_1, ..., a_N + k_N \cdot h_N), \quad k_i = 0, ..., n_i - 1, \quad i = 1, ..., N.$$

All $x_{k_1,...,k_N}$ can be rearranged into a large vector $\mathbf{x} \in \mathbb{R}^{\Pi_{i=1}^N n_i}$, defined as

$$x_\xi := x_{k_1,...,k_N}, \quad \xi = \sum_{i=1}^{N-1} \prod_{j=i+1}^{N} n_j k_i + k_N, \quad k_j = 0, ..., n_j - 1, \quad j = 1, ..., N.$$

The operator $\mathbf{W}$ is discretized using a collocation scheme. The values $\mathbf{W}_{\xi,\eta}$ are given by

$$\mathbf{W}_{\xi,\eta} = w(\mathbf{x}_\xi, \mathbf{x}_\eta), \qquad \xi, \eta = 0, ..., \prod_{i=1}^{N} n_i - 1$$

Assume we have ℓ time discretization points. Adopting the notation

$$\boldsymbol{\varphi}^{(s)} := \boldsymbol{\varphi}(t_s), \quad \boldsymbol{\psi}^{(s)} := \boldsymbol{\psi}(t_s), \quad s = 1, ..., l,$$

we can define the following matrices:

$$\mathbf{A} := \left(\boldsymbol{\varphi}^{(1)}, \ldots, \boldsymbol{\varphi}^{(\ell)}\right), \quad \mathbf{B} := \left(\boldsymbol{\psi}^{(1)}, \ldots, \boldsymbol{\psi}^{(\ell)}\right). \tag{2.19}$$

We can now rewrite 1.4 as:

$$\mathbf{B} = \mathbf{WA} \tag{2.20}$$

Although it looks like it, 2.20 is not a standard least squares problem. Firstly, the unknown variable $\mathbf{W}$ in the product $\mathbf{WA}$ is on the left side. This can be overcome by taking the transpose on both sides. We then have the equivalent problem

$$\mathbf{B^T} = \mathbf{A^T W^T}. \tag{2.21}$$

If we write $\mathbf{B} = [\mathbf{b}_1\ \mathbf{b}_2 \cdots\ \mathbf{b}_\ell]$ and $\mathbf{A} = [\mathbf{a}_1\ \mathbf{a}_2 \cdots\ \mathbf{a}_\ell]$, we see that 2.21 can be seen as the system of equations:

$$\begin{aligned} \mathbf{b_1^T} &= \mathbf{a_1^T W^T} \\ \mathbf{b_2^T} &= \mathbf{a_2^T W^T} \\ \vdots &= \quad \vdots \\ \mathbf{b}_\ell^\mathbf{T} &= \mathbf{a}_\ell^\mathbf{T}\mathbf{W}^\mathbf{T} \end{aligned} \tag{2.22}$$

Here we see that the columns of $\mathbf{A}$ and $\mathbf{B}$ constitute the variables of the system and the rows constitute the number of equations used to solve the system. In this case, the variable is $\mathbf{W}$. It is not a vector but a matrix, in contrast to a standard least squares problem.
We furthermore observe that there is no clear separation between model and data. The measured data are in the matrix $\mathbf{B}$. In a standard least squares problem, $\mathbf{A}$ would be the model operator and would be independent of the data. In this case however, $\mathbf{A}$ also contains the data.

The discretized operator $\mathbf{W}$ is calculated by:

$$\mathbf{W} = \mathbf{B}\mathbf{A}^\dagger, \tag{2.23}$$

where $\mathbf{A}^\dagger = (\mathbf{A^T A})^{-1}\mathbf{A^T}$, the existence of which is discussed in [18], [19]. The product $(\mathbf{A^T A})^{-1}\mathbf{A^T}$ is the discretized version of 2.11. The matrix $\mathbf{A^T A}$ is the discretized version of M and thus the ill-posedness of the problem is related to the invertibility of the discretized overlap matrix $\mathbf{A^T A}$. We see that for increasing ℓ the vectors φ, i.e. the columns of $\mathbf{A^T A}$, become linearly dependent and the matrix $\mathbf{A^T A}$ becomes non-invertible. Therefore, to find a solution to the problem we have to regularize $(\mathbf{A^T A})^{-1}\mathbf{A^T}$. In [1] Tikhonov regularization was used, giving the equation:

$$\mathbf{R}_\alpha = (\alpha\mathbf{I} + \mathbf{A^T A})^{-1}\mathbf{A^T}. \tag{2.24}$$

The regularized kernel $\mathbf{W}_\alpha$ is now given by

$$\mathbf{W}_\alpha = \mathbf{B}(\alpha\mathbf{I} + \mathbf{A^T A})^{-1}\mathbf{A^T}. \tag{2.25}$$

The examples done by Potthast and beim Graben [1] are carried out by the following steps.

1. Given a certain field u generated by the Amari equation compute the matrices $\mathbf{A}$ and $\mathbf{B}$.
2. Calculate $\mathbf{R}_\alpha$ using 2.24 and use it to calculate $\mathbf{W}_\alpha$ using 2.25.
3. In the case of a prescribed kernel we can directly compare the ground truth $\mathbf{W}$ and the estimated $\mathbf{W}_\alpha$. In the case of a prescribed sequence we compare u and u_α where u_α is reconstructed using $\mathbf{W}_\alpha$ in 2.20.

The reconstructed solution u_α is generated using a different time discretization than the solution u. This is done to avoid an *inverse crime*, as described in [5], and specifically for this problem but in less detail in [1].

2.3 Solution techniques

In the previous sections we have shown that we are dealing with an ill-posed inverse problem. To resolve this, Potthast and beim Graben have used Tikhonov regularization and have shown in their work that this leads to good results [1]. In this section we discuss new methods to reduce the ill-posedness and improve the quality of reconstruction.

2.3.1 Parameter optimization

We have shown in section 2.1 that the ill-posedness of the problem is due to the fact that the matrix $\mathbf{A^T A}$ is not invertible. However, even if $\mathbf{A^T A}$ is invertible we may not be able to reconstruct the kernel due to other unknown parameters in the equation. These are the parameters β and η in the firing function. In [1] the parameters were chosen beforehand, and all examples were done using these fixed values. The prescribed sequences are generated using certain parameters of the firing function and we need to ensure that we use the same parameters in the inverse problem. We therefore try to determine the parameters β and η that produce the lowest error and condition number (see section 2.4) and assume these are the parameters that were used in the forward problem.

2.3.2 Subsampling

We have shown through the construction of biorthogonal sets that the ill-posedness in our problem is due to a very fine discretization in the temporal domain. To overcome this, we could take a very coarse discretization, i.e. very large time steps, but his way we risk missing important features in the solution that can be very informative. This problem of course only arises if we observe sequences that have very little transients. Transients are changes in time over space. If the sequence has many transients then φ_i and $\varphi_j, j \neq i$ are very different and the overlap matrix will be well-defined. This of course gets harder for decreasing Δt, or equivalently, increasing ℓ. If the sequence has very little transients then we could choose a very coarse discretization over time, but we would like to develop methods that solve the inverse problem independent of the structure of the sequence. A very fine discretization in space would also be an option to resolve this problem. If $N \gg \ell$ then the overlap matrix will be well-defined as well. However, in practice, we can never demand a certain discretization over space, especially since we are dealing with micro-scale problems in the brain. We look for a method of regularization that is both independent of the discretization in space and time and the structure of the sequence.
To reduce the ill-posedness in the problem, Potthast and beim Graben have used Tikhonov regularization. We propose **regularization through subsampling**. The idea is that we select exactly that data that adds information to our problem, while cutting out data that doesn't. By adding information we mean that the data we want to use is independent of all other data we are using. In a discretized sense, this boils down to reducing the matrix $\mathbf{A}$ to *reduced*

row echelon form. We then select all pivot columns and use these columns to form the new matrix $\tilde{\mathbf{A}}$. We cut out the same columns in $\mathbf{B}$, yielding a matrix $\tilde{\mathbf{B}}$. We can now calculate $(\tilde{\mathbf{A}}^{\mathbf{T}}\tilde{\mathbf{A}})^{-1}\tilde{\mathbf{A}}^{\mathbf{T}}$, which we will still denote by $\tilde{\mathbf{A}}^{\dagger}$. We furthermore write $\tilde{\mathbf{W}} = \tilde{\mathbf{B}}\tilde{\mathbf{A}}^{\dagger}$.
The idea behind this method is to use data that is actually necessary to solve the problem. If two different columns in the matrix $\mathbf{A}$ are very similar then they do not give any extra insight into the matrix $\mathbf{W}$. When data is actually giving information different from all other data we can infer new information about $\mathbf{W}$.
To compute the reduced row echelon form we use the standard algorithm in MATLAB, shown in algorithm 1.

Data: matrix A and a tolerance tol.
Result: [A,jb] = rref(A,tol) returns A in reduced row echelon form and returns the indices of the pivot columns in jb.
[m,n] = size(A);
while $i \leq m$ && $j \leq n$ **do**
  find the value p and the index k of the largest absolute value in column j;
  if $p < tol$ **then**
    Set the column to zero;
  else
    Create a pivot element;
  end
  i++;
  j++;
end

Algorithm 1: Reduced Row Echelon Form algorithm

An important parameter in this algorithm is the tolerance, **tol**. This functions like a measure for the linear dependence of two columns. A very low tolerance means that we impose a more strict measure for linear independence between columns. We will show in the examples how it influences the quality of the reconstruction. The algorithm used to transform $\mathbf{A}$ and $\mathbf{B}$ to $\tilde{\mathbf{A}}$ and $\tilde{\mathbf{B}}$ respectively is described in algorithm 2.

Data: matrices A and B and a tolerance tol.
Result: $\tilde{A}$ = rref_cut(A,tol) returns the pivot columns of A to $\tilde{A}$
[A,jb] = rref(A,tol)
A = A(:,jb)
B = B(:,jb)

Algorithm 2: Algorithm that transforms A and B to $\tilde{A}$ and $\tilde{B}$

2.3.3 Combining data

In some cases one sequence may not suffice to reconstruct the kernel. This happens in two cases:

1. The sequence is zero in some parts of the spatial domain for the full time window. If the sequence is zero in some parts of the spatial domain we do not have any information about the influence of these points on the other points in the domain. Therefore the corresponding points in the synaptic weight kernel will be zero.

2. The sequence is not informative enough. From a continuous point of view this can be seen from 2.13. If n is a small number compared to the dimension of the φ_i, we have very little information about the φ_i. We thus see that when we use algorithm 2 the rank of the matrix is actually a measure of the informativeness of the sequence. For low rank matrices we have a low number of equations solving system 2.22, whereas for high rank matrices the converse is true. Sequences with very few transients will have a low rank. When $\mathbf{A}$ has low rank, we cannot expect a good reconstruction of $\mathbf{W}$. One possibility to overcome this is to combine sequences generated by the same kernel. This can be done by choosing different initial conditions and combining the sequences generated by them. We have to ensure that these sequences are qualitatively very different. We can combine sequences in our procedure by simply concatenating the matrices $\mathbf{A}$ and $\mathbf{B}$ that are generated by them. After that we apply algorithm 2 to the resulting matrix $\mathbf{A}$.

2.3.4 Localization

In some cases we have a priori knowledge about the connections we expect to see. When we know that we expect to see more local than non-local connections we can use a technique called localization, which has previously been used for kernel reconstruction in [20]. We solve the equation $\mathbf{W} = \mathbf{B}\mathbf{R}_\alpha$ on a localized domain Ω_{loc}. We have that $x \in \Omega_{\text{loc}}$ if $\|x - y\| < \rho$, $\forall y \in \Omega$. Denote $\mathbf{K} = \mathbf{A}\big|_{\Omega_{\text{loc}}}$ and $\mathbf{b} = \mathbf{B}\big|_{\Omega_{\text{loc}}}$. We then solve the equation $\mathbf{W}_{\text{loc}} = \mathbf{b}(\alpha + \mathbf{K}^\mathbf{T}\mathbf{K})^{-1}\mathbf{K}^\mathbf{T}$ and equate $\mathbf{W}\big|_{\Omega_{\text{loc}}} = \mathbf{W}_{\text{loc}}$. We apply this procedure for all $x \in \Omega$, or in the discretized setup, for all x_i, $i = 1, ..., n$. In the discretized version we use the following procedure. For a given ρ and index i, we select all indices j s.t. $\|x_i - x_j\| < \rho$ and store them in I_0. We then select all indices k s.t. $f(u(x_j, t_k)) > \nu$ and store them in I_1. We then build the matrix $\mathbf{K}$ and the vector $\mathbf{b}$ using $\mathbf{K} = \mathbf{A}(I_0, I_1)$ and $\mathbf{b} = \mathbf{B}(x_i, I_1)$. Then we solve the equation $\mathbf{W}_{\text{loc}} = \mathbf{b}(\alpha + \mathbf{K}^\mathbf{T}\mathbf{K})^{-1}\mathbf{K}^\mathbf{T}$ and store it in $\mathbf{W}_\alpha(x_i, I_1)$.

2.4 Validation

In this section we discuss how we validate the quality of the reconstruction of $\mathbf{W}$. In inverse problems, people usually look at the error between the ground truth (if available) and the reconstruction and the condition number. The condition number is a measure of the stability with respect to errors in the data.

2.4.1 Error

When calculating the error we again have to make a distinction between prescribed sequences and prescribed kernels. If we have a prescribed kernel we can determine the error by calculating the norm between $\mathbf{W}$ and $\mathbf{W}_\alpha$ (or $\tilde{\mathbf{W}}$), using the formula:

$$\text{Error} = \|\mathbf{W} - \mathbf{W}_\alpha\|_F, \tag{2.26}$$

where $\|\cdot\|_F$ denotes the *Frobenius norm*, which is defined as:

$$\|A\|_F = \sqrt{\sum_{i=1}^{m}\sum_{j=1}^{n}|a_{ij}|^2}. \tag{2.27}$$

We note that the matrix $\mathbf{W}$ is not the true kernel, but the kernel multiplied by the grid constants. In [1] the matrix $\mathbf{W}$ was shown in the figures and we have done the same to compare our results. Starting from section 3.6 we will show the true kernel $\mathbf{w}$ and use $\mathbf{w}$ instead of $\mathbf{W}$ in 2.26.
When we want to validate the quality of the reconstructed kernel for a prescribed sequence, we first calculate $\mathbf{W}_\alpha$. We then reconstruct the sequence u_α using the Amari equation with $\mathbf{W}_\alpha$. We then calculate the error between $u(x,t)$ and $u_\alpha(x,t)$ using the formula:

$$\text{Error} = \|\mathbf{u} - \mathbf{u}_\alpha\|_F. \tag{2.28}$$

This way, we calculate the error between u and u_α at every points in space per time step. This is not an optimal measure for all sequences. For instance, if we study traveling pulses the absolute error per time step is not optimal. We also have to take into account the shape of the pulse and the orientation, as well as the velocity of the pulse. It is very hard to determine an optimal measure for the error between two arbitrary sequences. We therefore use formula 2.28 as a good indication, keeping in mind that it is not optimal.

2.4.2 Condition number

The condition number is a measure of the stability of a matrix. The quantity arises naturally when studying the standard linear least squares equation $Ax = b$. Let e denote the error in b. The objective is to study the relative error in the solution induced by the relative error in b. This is calculated by:

$$\frac{\dfrac{\|A^{-1}e\|}{\|A^{-1}b\|}}{\dfrac{\|e\|}{\|b\|}} = \frac{\|A^{-1}e\|}{\|e\|}\frac{\|b\|}{\|A^{-1}b\|} \tag{2.29}$$

The condition number κ is defined as the maximum of 2.29 and is given by [21]:

$$\kappa(A) = \max_{e,b\neq 0} \frac{\|A^{-1}e\|}{\|e\|} \frac{\|b\|}{\|A^{-1}b\|} = \|A\| \cdot \|A^{-1}\|. \tag{2.30}$$

In case where the matrix is not invertible, we use the Moore-Penrose pseudo-inverse, $A^{\dagger}$, instead, giving:

$$\kappa(A) = \|A\| \cdot \|A^{\dagger}\|. \tag{2.31}$$

If the Singular Value Decomposition of A can be computed then the condition number is given by:

$$\kappa(A) = \frac{\sigma_1}{\sigma_N}. \tag{2.32}$$

As explained in section 2.2, we are not dealing with a standard linear least squares problem. However, when we use equation 2.21, we see that the solution of our problem is given by:

$$\mathbf{W^T} = \mathbf{A}(\mathbf{A^T A})^{-1}\mathbf{B^T}. \tag{2.33}$$

If we write $\mathbf{W^T} = [\bar{\mathbf{w}}_1\ \bar{\mathbf{w}}_2 \cdots\ \bar{\mathbf{w}}_N]$ and $\mathbf{B^T} = [\bar{\mathbf{b}}_1\ \bar{\mathbf{b}}_2 \cdots\ \bar{\mathbf{b}}_N]$, this can be rewritten as:

$$\begin{aligned} \bar{\mathbf{w}}_\mathbf{1} &= \mathbf{A}(\mathbf{A^T A})^{-1}\bar{\mathbf{b}}_\mathbf{1} \\ \bar{\mathbf{w}}_\mathbf{2} &= \mathbf{A}(\mathbf{A^T A})^{-1}\bar{\mathbf{b}}_\mathbf{2} \\ \vdots &= \quad\vdots \\ \bar{\mathbf{w}}_\mathbf{N} &= \mathbf{A}(\mathbf{A^T A})^{-1}\bar{\mathbf{b}}_\mathbf{N} \end{aligned} \tag{2.34}$$

This is the solution to the system 2.22. We see that when we study the condition number of $\mathbf{A}(\mathbf{A^T A})^{-1}$ we study the stability with respect to the measured data in $\mathbf{B}$. Since $\kappa(\mathbf{A}) = \kappa(\mathbf{A^T})$ and $(\mathbf{A}(\mathbf{A^T A})^{-1})^T = (\mathbf{A^T A})^{-1}\mathbf{A^T} = \mathbf{A}^{\dagger}$ we see that $\kappa(\mathbf{A}^{\dagger})$ is a measure for the stability.
In our case $\mathbf{A}^{\dagger}$ has the data in it as well, in contrast to the matrix A^{-1} in 2.29. This means that an error in the data has an effect on the quantity $\mathbf{A}^{\dagger}$ as well. The effect this has is captured by the condition number as well. The effect of a change in the entries on the pseudo-inverse of a matrix A is measured by $\frac{dA^{\dagger}}{d\alpha}$, where α denotes the entries of A. Let I denote the identity matrix. We then derive an expression for this by using:

$$0 = \frac{dI}{d\alpha} \tag{2.35}$$

$$= \frac{dA^{\dagger}A}{d\alpha} \tag{2.36}$$

$$= \frac{dA^{\dagger}}{d\alpha}A + A^{\dagger}\frac{dA}{d\alpha}. \tag{2.37}$$

Rewriting equation 2.37 we get:

$$\frac{dA^\dagger}{d\alpha} = -A^\dagger \frac{dA}{d\alpha} A^\dagger \tag{2.38}$$

Taking norms on both sides we get:

$$\left\| \frac{dA^\dagger}{d\alpha} \right\| = \left\| -A^\dagger \frac{dA}{d\alpha} A^\dagger \right\| \tag{2.39}$$

$$\leq \|A^\dagger\|^2 \left\| \frac{dA}{d\alpha} \right\| \tag{2.40}$$

$$\frac{\left\| \frac{dA^\dagger}{d\alpha} \right\|}{\|A^\dagger\|} \leq \|A^\dagger\| \|A\| \frac{\left\| \frac{dA}{d\alpha} \right\|}{\|A\|} \tag{2.41}$$

$$= \kappa(A) \frac{\left\| \frac{dA}{d\alpha} \right\|}{\|A\|}, \tag{2.42}$$

where in the last line we have used 2.31. This means that the condition number is an upper bound for the relative change in $A^\dagger$ with respect to the relative change in the entries of A.
We thus see that the condition number is a measure of the change in the solution with respect to a certain input as well as an upper bound for the the error we get in the quantity $\mathbf{A}^\dagger$ with respect to an error in $\mathbf{A}$.

Chapter 3

Application to the Neural Field Equation

In this chapter we present examples illustrating the difficulties that arise in inverse problems in neural field theory. We use prescribed sequences and prescribed kernels found in different articles on neural fields [1], [7], [8]. We start with the examples introduced in [1]. We reproduce their results to verify if our results are correct and then compare them to the results we get when using our solution techniques. The order in which we present the examples from [1] is chosen according to the difficulties that arise in solving the inverse problem. In section 3.2 we start with a relatively easy example where $\mathbf{A^T A}$ is still invertible. We only have to optimize over the parameters. In sections 3.3 and 3.4 we deal with traveling pulses. Here, the matrix $\mathbf{A^T A}$ is no longer invertible and we introduce our subsampling method. In section 3.5 we deal with a prescribed kernel. There we have the additional problem that we need to combine sequences to make them informative enough. In section 3.6 we extend to a prescribed kernel where we introduce excitatory and inhibitory behavior. In section 3.7 we use simplified versions of kernels described in [8]. Here, we will demonstrate how localization can help to solve the inverse problem and gain some more insight to the problems arising in the inverse problem. We start out with a summary of the work in this chapter as clarification for the reader.

3.1 Summary

Table 3.1 refers to all examples and the solution techniques used for that example. A white cell indicates that a certain solution technique is not used for that particular example. The shade of green shows how well the techniques improves with respect to Tikhonov regularization. The last column shows the overall quality of the reconstruction for the best case scenario. Red means a bad reconstruction. We seperate the Compexity, Entropy and Integration kernel via double line. Above, we have arranged the examples in order of increasing

complexity. The overall quality column is separated because in that column the color indicates the quality of the overall reconstruction and is not with respect to the case where we use Tikhonov regularization.

Example \ Solution Techniques	Validation 2.4	Parameter optimization	Subsampling	Subsampling + Tikhonov	Combining data	Localization	Overall quality
Sine 3.2	Sequence						
Traveling pulse 3.3	Sequence						
XOR 3.4	Sequence						
Kernel pulse 3.5	Kernel						
Mexican hat 3.6	Kernel						
Complexity 3.7	Kernel						
Entropy 3.7	Kernel						
Integration 3.7	Kernel						

Table 3.1: Table showing the solution techniques used for each example. The shade of green indicates the improvement in terms of error and condition number with respect to Tikhonov regularization. Red means a bad reconstruction.

3.2 Sine wave with order parameter dynamics

In this example we study a sine wave transitioning continuously between a finite number of linearly independent states [1]. This prescribed sequence $v(x,t)$ is given by:

$$v(x,t) = \sum_{i=1}^{Q} \lambda_i(t) \sin(ix)$$

The functions λ_i are so called tent functions. These functions are compactly supported and non-zero on the interval $[t_i, t_{i+1}]$, where the t_i are defined such that $0 = t_1 < t_2 < \cdots < t_Q = T$, and $|t_{i+1} - t_i| = \frac{T}{Q-1}\ \forall i$. The functions λ_i are given by:

$$\lambda_i(t) = \begin{cases} 0 & \text{if } t < t_{i-1} \\ \frac{Q-1}{T}(t - t_{i-1}) & \text{if } t_{i-1} \leq t < t_i \\ 1 - \frac{Q-1}{T}(t - t_i) & \text{if } t_i \leq t < t_{i+1} \\ 0 & \text{if } t \geq t_{i+1} \end{cases}$$

The spatial domain is $[0, 2\pi]$. The other parameters are $N = 320$, $T = 7$, $\ell = 100$ and $\tau = 2$. In order to verify our results, we have reproduced the matrices $\mathbf{A}$, $\mathbf{B}$, $\mathbf{R}_\alpha$ and $\mathbf{W}_\alpha$ that are shown in [1]. The matrices are shown in Figure 3.1.

We observe that there is a small difference between the matrices that we have generated and the matrices shown in [1]. When we change the parameter

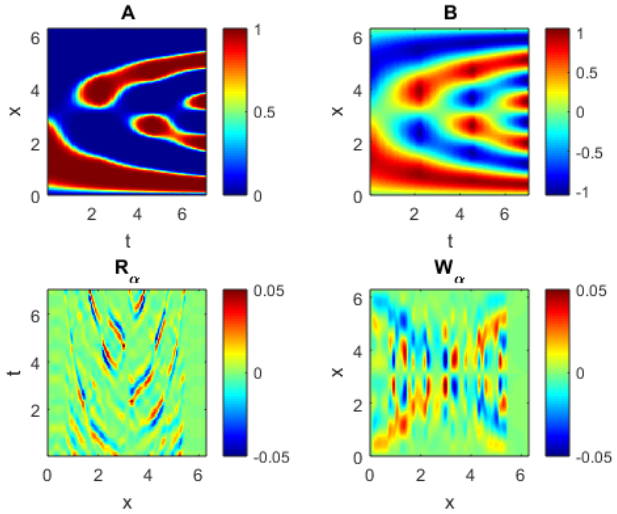


Figure 3.1: Matrices $\mathbf{A}$, $\mathbf{B}$, $\mathbf{R}_\alpha$ and $\mathbf{W}_\alpha$ for $\beta = 10$.

to $\beta = 100$ we observe that we obtain the same results as Potthast and beim Graben in [1]. We show the results in 3.2.

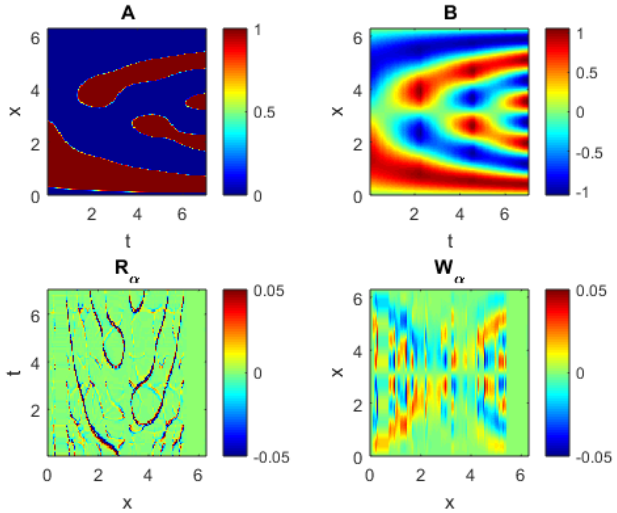


Figure 3.2: Matrices $\mathbf{A}$, $\mathbf{B}$, $\mathbf{R}_\alpha$ and $\mathbf{W}_\alpha$ for $\beta = 100$.

For the matrices shown in Figure 3.1 $Q = 4$. The prescribed and reconstructed sequences shown in [1] are generated using $Q = 8$. To compare our

results to the results in [1] we use the same parameters.

3.2.1 Tikhonov regularization

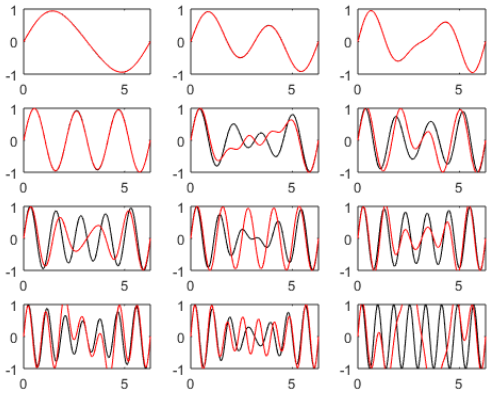
In [1] Potthast and beim Graben have shown that for the parameters $\beta = 10$ and $\eta = 0.3$ the problem is unstable. They have used Tikhonov regularization to overcome this instability. We show the results for $\alpha = 0,\ 1,\ 30$ in Figure 3.3. The black line is the prescribed sequence and the red line is the reconstructed sequence.

We see that if we don't regularize, i.e. $\alpha = 0$, after some time the reconstructed sequence strongly deviates from the prescribed sequence and shows some unstable behavior. If the regularization of the connectivity is too strong, i.e. $\alpha = 30$, the reconstructed sequence eventually damps out. For $\alpha = 1$ we observe a decent reconstruction. These are all observations we expect when dealing with an ill-posed problem, and can be overcome by regularization.

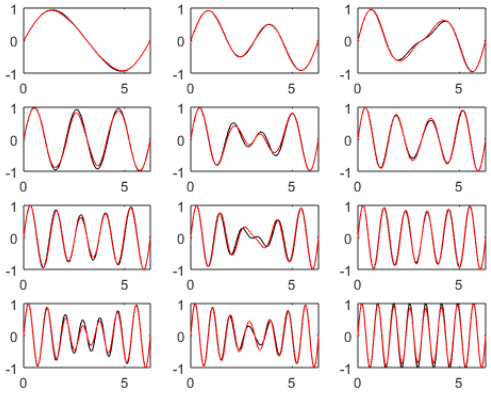
3.2.2 Influence of the parameters β and η

We have shown in section 3.2.1 that we observe behavior that is typical for ill-posed probems. From a theoretical viewpoint however, this should not be the case. In this example $N \gg \ell$ to ensure that $\mathbf{A^T A}$ is invertible. We have shown in section 2.1 that if the matrix $\mathbf{A^T A}$ is invertible then the problem is not ill-posed and we should be able to reconstruct the kernel without any regularization.
What was not taken into account in [1] is that the parameters of the firing function have an influence as well. The parameters are however unknown. Since the solution was generated by the Amari equation using a specific firing function the parameters have an influence on the inverse problem as well. To determine the right parameters we have set up an experiment where we vary the parameter β in the range $5 \leq \beta \leq 115$ and we vary the parameter η in the range $0.2 \leq \eta \leq 0.6$. We then investigate the error of the reconstruction and the condition number of $\mathbf{R}_\alpha$. We vary α from $\alpha = 1 \cdot 10^{-5}$ to $\alpha = 10$ where α is multiplied by 10 in each step. Furthermore we include $\alpha = 0$ to see what happens if we don't regularize. For each α we look at the parameters of the firing function that give the lowest error.

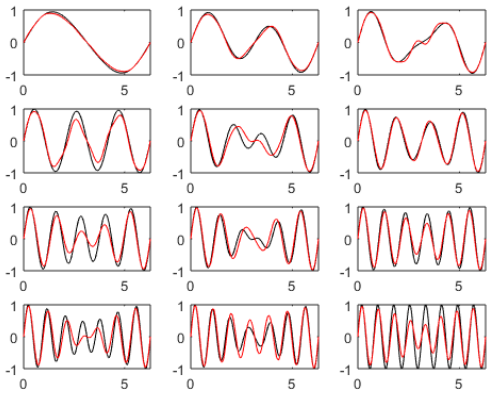
From table 3.2 we see that it is not strictly necessary to regularize the solution, as long as we are able to find the right parameters for the firing function. The smallest error for the reconstructed solution is for $\alpha = 1 \cdot 10^{-4}$, but the error nor the condition number for $\alpha = 0$ are that much bigger. For $\alpha = 1$ we do find a much smaller condition number, but here the error is bigger and also the parameter β gets very big. From the table it is clear that the parameters $\beta = 18$ and $\eta = 0.3$ appear the most and are probably the parameters that generated the sequence. We do see however that for $\beta = 17$ and $\eta = 0.3$ we have the smallest error. For completeness we have added a plot of the largest singular values, smallest singular values, condition numbers, and the error for varying α for both these parameters. They are shown in figures 3.4 and 3.5.



Solution for $\alpha = 0$.



Solution for $\alpha = 1$.



Solution for $\alpha = 30$.

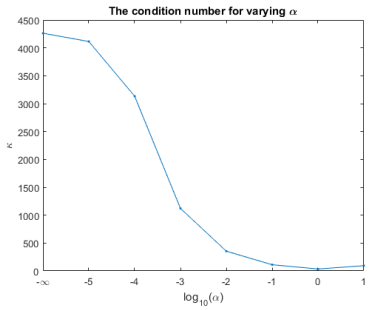
Figure 3.3: Instability resolved using Tikhonov regularization.

α	Optimal parameters	Smallest error	condition number
$1 \cdot 10^{-5}$	$\beta = 18$ and $\eta = 0.3$	1.166	$3.314 \cdot 10^3$
$1 \cdot 10^{-4}$	$\beta = 17$ and $\eta = 0.3$	1.097	$3.133 \cdot 10^3$
$1 \cdot 10^{-3}$	$\beta = 18$ and $\eta = 0.3$	1.143	$1.123 \cdot 10^3$
$1 \cdot 10^{-2}$	$\beta = 29$ and $\eta = 0.2$	1.421	$3.952 \cdot 10^2$
$1 \cdot 10^{-1}$	$\beta = 40$ and $\eta = 0.2$	1.728	$1.253 \cdot 10^2$
0	$\beta = 18$ and $\eta = 0.3$	1.187	$3.389 \cdot 10^3$
1	$\beta = 101$ and $\eta = 0.4$	3.853	$3.168 \cdot 10^1$
10	$\beta = 10$ and $\eta = 0.3$	15.19	$2.009 \cdot 10^3$

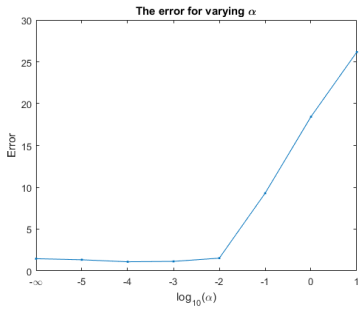
Table 3.2: Smallest error for varying parameters of the firing function.

Here we see that although we have in principle a well-posed problem, since $\mathbf{A}$ is linearly independent, we do benefit from regularizing the solution. The result for $\beta = 17$ and $\alpha = 1 \cdot 10^{-4}$ is shown in Figure 3.6.

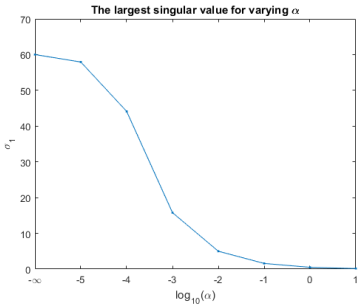
It is also clear that it is not strictly necessary to regularize the operator $(\mathbf{A^T A})^{-1}\mathbf{A^T}$ and that optimizing with respect to the parameters β and η reduces the ill-posedness. We get additional benefits from using Tikhonov regularization in terms of a lower condition number, but the error gets bigger. For the parameter setup in [1] the error is $1.08 \cdot 10^1$ and the condition number is $6.4 \cdot 10^2$. The singular values are shown in Figure 3.7. Although the condition number is smaller, the error is also much larger. If we compare the case where $\beta = 18$ and $\alpha = 0$ we see that the condition number for $\beta = 10$ and $\alpha = 1$ is a factor 0.19 smaller, but the error is a factor 9.08 bigger. If we then use $\beta = 18$ and $\alpha = 1 \cdot 10^{-3}$ the condition number is a factor 0.57 smaller, but the error is a factor 9.47 bigger.



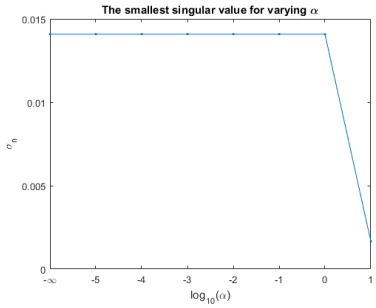
Condition number



Error

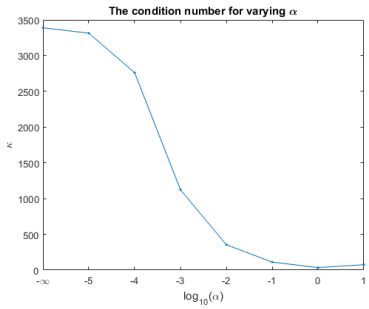


Largest singular value

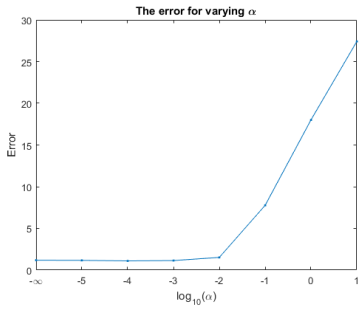


Smallest singular value

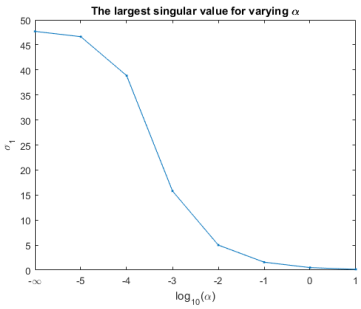
Figure 3.4: Results for $\beta = 17$ and $\eta = 0.3$.



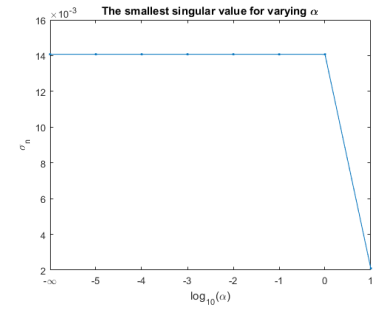
Condition number



Error



Largest singular value



Smallest singular value

Figure 3.5: Results for $\beta = 18$ and $\eta = 0.3$.

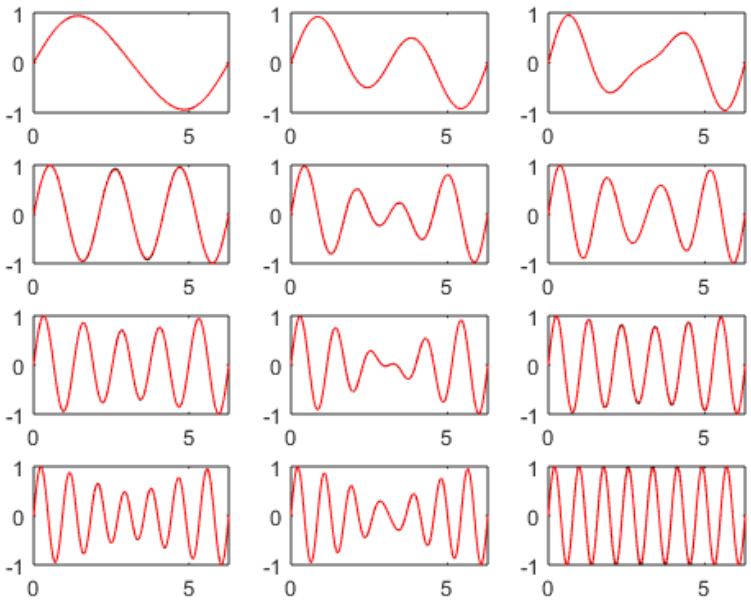


Figure 3.6: Result for $\beta = 17$, $\eta = 0.3$ and $\alpha = 1 \cdot 10^{-4}$.

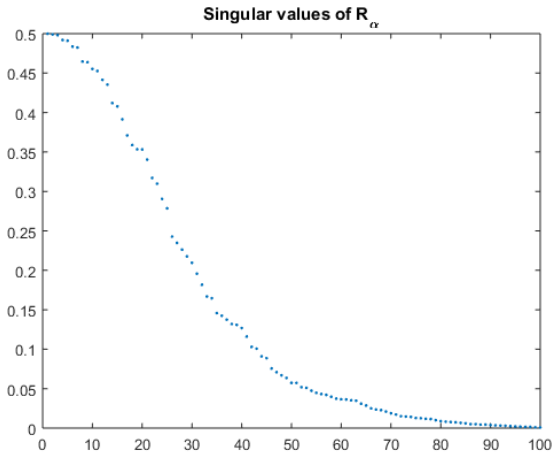


Figure 3.7: Singular values for $\beta = 10$, $\eta = 0.3$ and $\alpha = 1$. The horizontal axis denotes the ordering of the singular values from largest to smallest.

3.3 Traveling pulse

For this example we consider a traveling pulse which follows a prescribed parabolic path [1]. This is a prescribed sequence given by:

$$u(x,t) := e^{-R|x-x_0(t)|^2}, \quad t \in [0,T] \tag{3.1}$$

where

$$x_0(t) := (q_1 t, q_2 t - q_3 t^2).$$

In [1] the parameters for the pulse and the discretization were not explicitly given. We were not able to get the same reconstruction as in [1]. However, we do get a reasonable reconstruction. We have tried to estimate the parameters used in [1] based on the figures shown. We have estimated that the parameters chosen were $q_1 = 1$, $q_2 = 0.8$, $q_3 = 5q_2$ and $R = 5$ for the parabolic path and the size of the pulse. We have chosen $n_1 = 60$, $n_2 = 60$, $x_1 \in [0,5]$, $x_2 \in [0,5]$ and $\ell = 800$. The parameters for the firing function in [1] are $\beta = 10$ and $\eta = 0.3$. We show twelve snapshots of the prescribed sequence in 3.8.

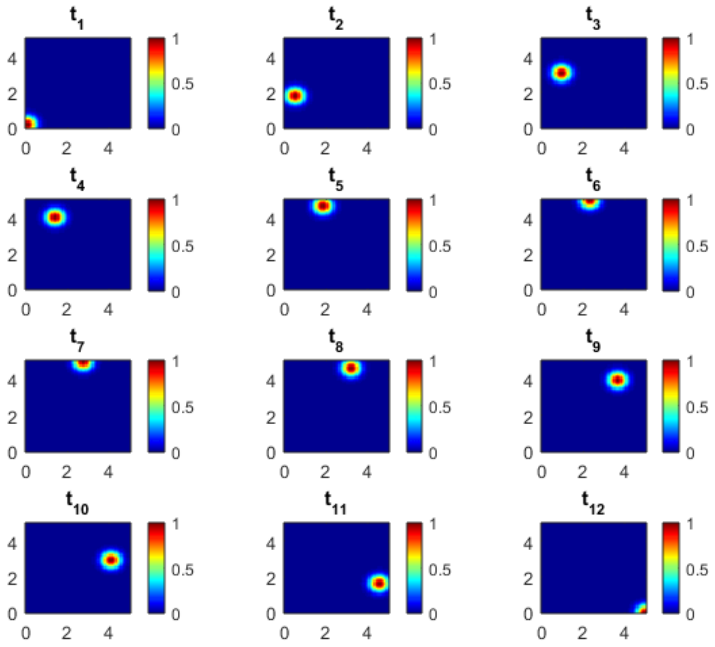


Figure 3.8: Prescribed traveling pulse.

3.3.1 Tikhonov regularization

For this example the matrix $\mathbf{A}^\mathbf{T}\mathbf{A}$ is not invertible. That means that in this case the problem is ill-posed and that we need to regularize. The resulting reconstructed sequence after solving the problem without regularization is shown in Figure 3.9.

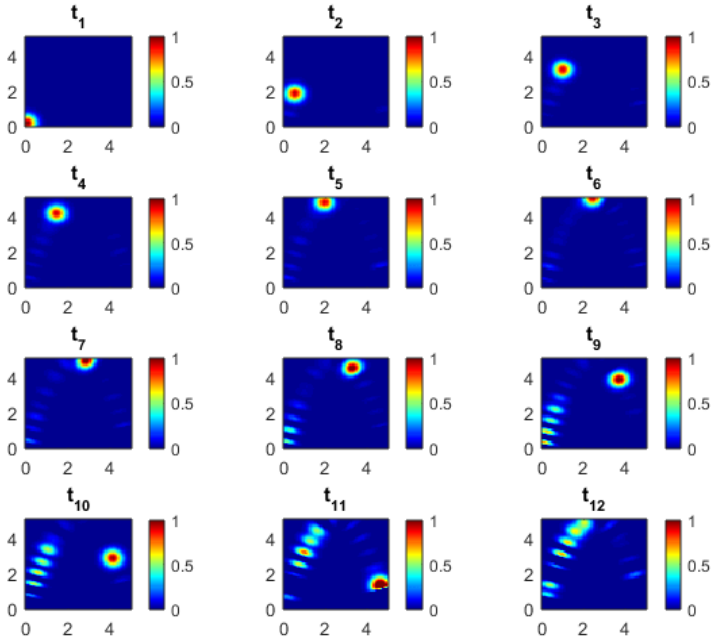


Figure 3.9: Reconstructed pulse without regularization.

We compare the results for Tikhonov regularization used in [1] and the subsampling method we described in section 2.3.2. As this is a prescribed sequence, we again optimize over the parameters of the firing function. For the parameters $\beta = 10$ and $\eta = 0.3$ we have found that $\alpha = 800$ yields the best results. The reconstructed sequence is shown in 3.10. The error calculated using formula 2.28 is 87.76. The condition number as calculated by equation 2.32 is $2.03 \cdot 10^{16}$.

Concerning the parameters of the firing function, we have found that the reconstruction gets better as $\beta \to \infty$: it seems to control the propagation speed of the pulse. We have therefore chosen:

$$f(u;\eta) = \begin{cases} 1 & \text{if } u - \eta > 0 \\ 0 & \text{otherwise} \end{cases} \tag{3.2}$$

The result is shown in Figure 3.11. The error is 65.83 and the condition

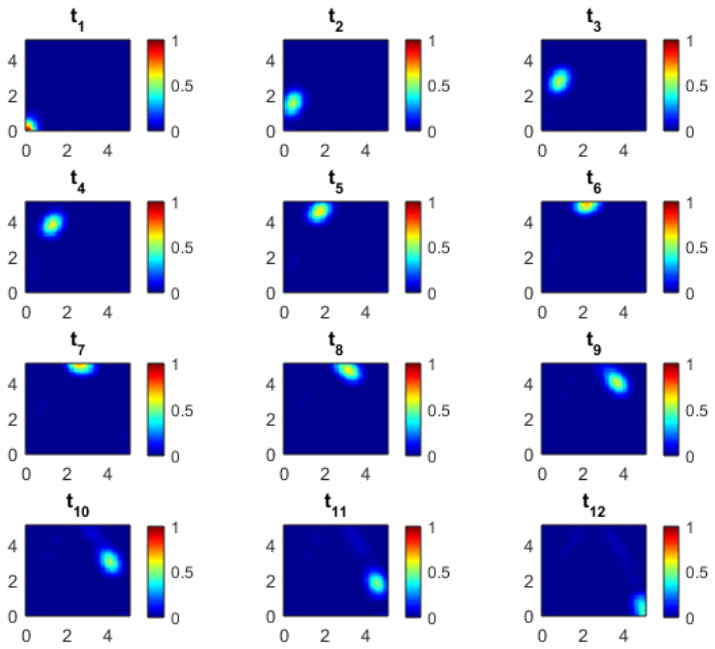


Figure 3.10: Reconstructed traveling pulse with $\beta = 10$.

number is $7.64 \cdot 10^{16}$.

We observe that the pulse reconstructed using the firing function 3.2 preserves it's shape better, and that the trail that the pulse leaves behind is less substantial.

3.3.2 Subsampling

In this section we present the results using the subsampling procedure described in 2.3.2. Through this procedure we have cut out 92 timesteps, reducing the size of the matrices $\mathbf{A}$ and $\mathbf{B}$ from 3600×800 to 3600×708. We estimate that the optimal parameter for η is $\eta = 0.4$. The error is 47.09 and the condition number is $4.33 \cdot 10^2$.

We show the distribution of the singular values for both methods in Figure 3.13. What's very interesting when we compare the figures is that we have precisely removed those columns that yield very small singular values. This means that this method functions somewhat like a spectral cut-off method. When we remove these small singular values from 3.13 the quality of the reconstruction does not improve, but gets really bad. We conclude that for this example the subsampling method cuts out precisely the information that is redundant and

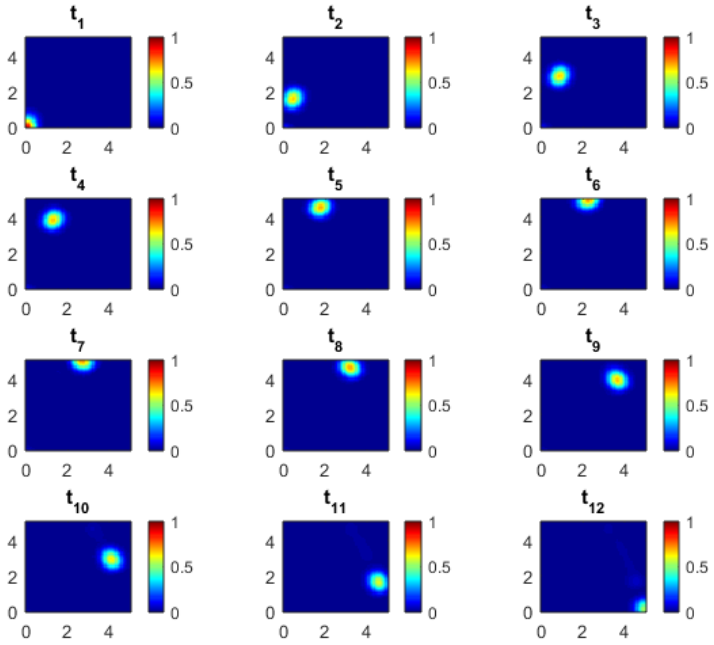


Figure 3.11: Reconstructed traveling pulse with the firing rate function 3.2.

leads to the ill-posedness in the problem.

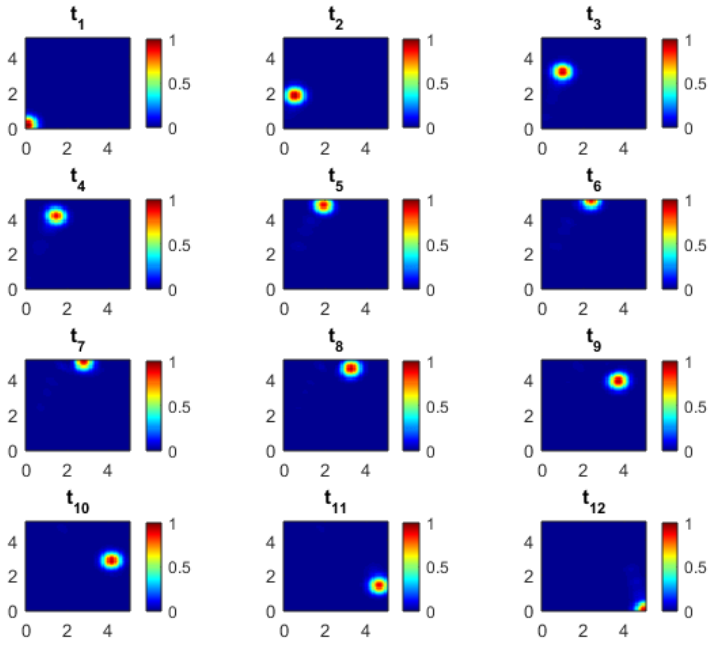


Figure 3.12: Reconstructed traveling pulse using subsampling.

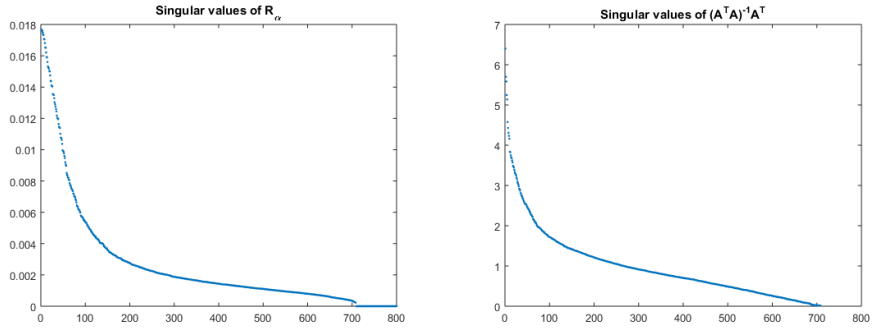


Figure 3.13: Singular values. Left for Tikhonov regularization and right for subsampling. The horizontal axis denotes the ordering of the singular values from largest to smallest.

3.4 XOR gate

For this example we investigate the inverse problem for an XOR gate [1]. There are two gates placed at the positions $(0, 3)$ and $(0, -3)$. We consider pulses

traveling through the gates to the position $(10, 0)$ on the domain $[0, 10] \times [-5, 5]$. When there are pulses at both gates they will die out. If there is only one pulse at either one of the gates it will travel to $(10, 0)$.
We have again found that we get the best results when we use a step function for the firing function, with $\eta = 0.3$. Using the subsampling method we find that the condition number is 47.33. The error for the reconstruction for a pulse at either $(0, 3)$ or $(0, -3)$ is 29.28. For a pulse at both gates the error is 9.05. We show the reconstruction in 3.14. When using Tikhonov regularization we find that the error for a pulse at $(0, 3)$ or $(0, -3)$ is 29.30 and the error for a pulse at both gates is 9.03. The condition number is much larger: $1.2 \cdot 10^{16}$. When looking at the singular values, we again see that the subsampling method functions somewhat like a spectral cut-off method, as described in section 3.3 for the traveling pulse as well. This is shown in Figure 3.15.

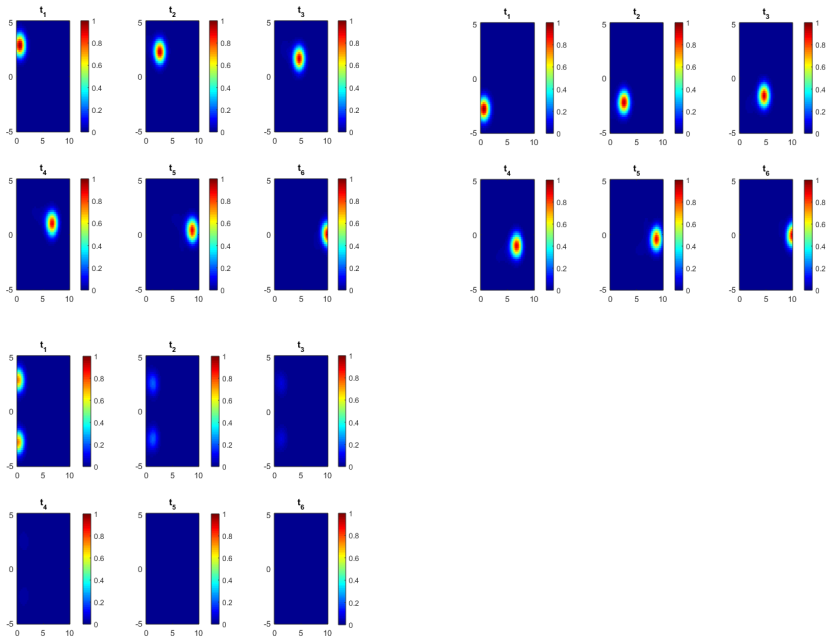
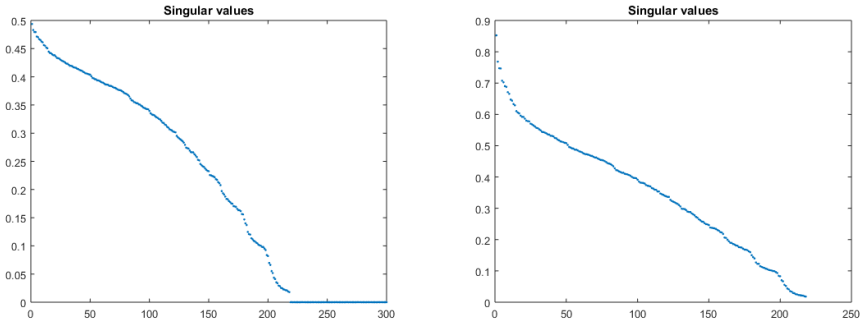


Figure 3.14: Reconstructed pulses placed at the XOR gates using subsampling.



Tikhonov regularization Subsampling

Figure 3.15: Singular values. Left for Tikhonov regularization and right for subsampling. The horizontal axis denotes the ordering of the singular values from largest to smallest.

3.5 Two-dimensional pulse with a prescribed kernel

This example shows the reconstruction of a prescribed kernel [1]. Here, we do not have a prescribed sequence, but a prescribed kernel and we generate our sequence using the Amari equation. The prescribed kernel is given by:

$$w(x,y) := ce^{-R|x-y|^2} \cdot \chi_{x \geq y}(x,y) - \chi_{x_1 < y_1}(x,y) - \chi_{x_2 < y_2}(x,y), \quad x, y \in \Omega \quad (3.3)$$

Here χ is given by:

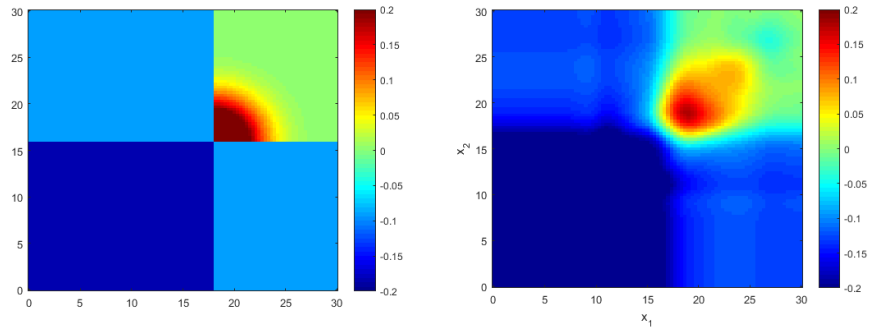
$$\chi_{\text{statement}} = \begin{cases} 1 & \text{if statement is true} \\ 0 & \text{if statement is false} \end{cases} \quad (3.4)$$

The initial condition is given by:

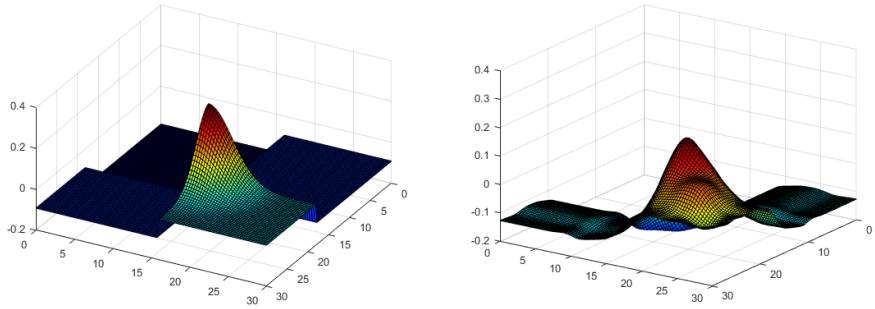
$$u(x,0) = \begin{cases} 1 & \text{if } |x - x_0| \leq c \\ 0 & \text{if otherwise} \end{cases} \quad (3.5)$$

We have included the constant c in formula 3.3 because we have found that the pictures shown in [1] are not in accordance with the formula if $c = 1$. The full kernel times the grid constants h_1, h_2 is shown in Figure 3.17 with $c = 2$ and this figure is accordance with the one in [1]. The kernel times the grid constants h_1, h_2 for fixed x_2 is given in Figure 3.16. Here, we have chosen $c = 4$ so that the figure is the same as in [1]. The reconstructed kernel slightly from the one shown [1], which is likely due to a difference in the discretization of the spatial and temporal domain and the regularization parameter.

The full kernel shown in [1] is not the full kernel for the kernel in the previous example for fixed x_2. There, the spatial domain was discretized using the domain



2D kernel 2D Reconstructed kernel



3D kernel 3D reconstructed kernel

Figure 3.16: Various plots of the kernel for fixed y.

$[0, 30] \times [0, 30]$ and $n_1 = 91$ and $n_2 = 91$. For the full kernel the spatial domain is discretized using the domain $[0, 10] \times [0, 10]$ and $n_1 = 31$ and $n_2 = 31$. For the full kernel the constant in 3.3 is $c = 2$.

Due to the fact that we have a prescribed kernel we do not optimize over the parameters of the firing function and use the same parameters as in [1], namely $\beta = 10$ and $\eta = 0.5$. The parameters do have some influence in terms of the informativeness of the sequence, due to the fact that different parameters for the firing function generate different sequences. This influence is not substantial and therefore we omit results for other parameters of the firing function.
There are two things that influence the reconstruction of the kernel. The first is the constant c in 3.3. If this constant is too small then the initial pulse from 3.5 will not be propagated enough and the sequence will not be informative. We could also choose a constant smaller than 1 in 3.5: this will have the same effect as choosing a larger c in 3.3. The second influence is the width of the pulse. If the pulse is too narrow we again have a very non-informative sequence. In the

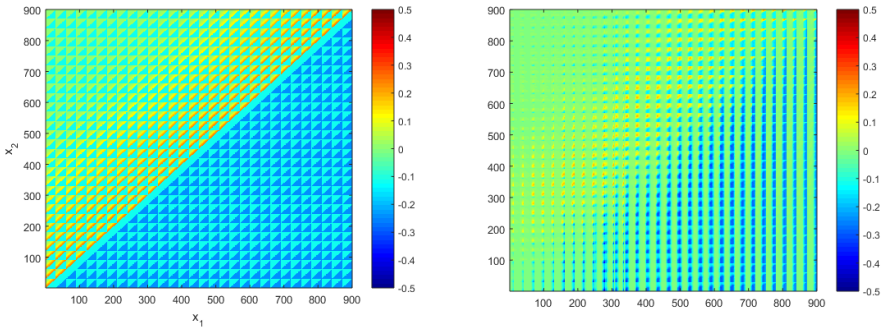


Figure 3.17: Kernel and reconstructed kernel. The axes show the discretization points and not the actual domain, in accordance with the figure in [1].

example shown in 3.17 we have chosen c in 3.5 $c = 4$. When we look at the reconstructed kernel in 3.17 we see bars that alternate between bars that show a reconstruction of the prescribed kernel and bars that are 0. This is due to the path that is traversed by the pulse. We illustrate this in Figure 3.18 for the first 12 timesteps and for the full time window in 3.19. We see clearly in Figure 3.18 that the pulse doesn't traverse the entire domain. The parts of the domain that are not traversed correspond to the bars that are 0 in 3.17.

3.5.1 Combining sequences

It is clear from 3.17 that additional information is required to reconstruct the kernel fully. In this case we do this by combining sequences with different initial conditions. We see from Figure 3.18 that the initial pulse travels diagonally to the right part of the domain. If we use several pulses located at different parts of the domain we can add more information to $\mathbf{A}$ and reconstruct the kernel better. We choose two different set-ups and compare the two cases.

1. **Setup 1**. We choose pulses at the locations $\begin{bmatrix} k-1 \\ 0 \end{bmatrix}$, $k = 1, ..., 10$ and $\begin{bmatrix} 0 \\ m-1 \end{bmatrix}$, $m = 2, ..., 10$ with radius 1 and compare the results when we use the matrices $\mathbf{A}$ and $\tilde{\mathbf{A}}$ and what the effects of regularization are. In this case there are pulses placed along the axes x_1 and x_2. Due to the structure of the kernel these pulses will traverse the entire domain and reconstruct the full kernel.
2. **Setup 2**. We choose pulses at $\begin{bmatrix} k-1 \\ m-1 \end{bmatrix}$, $k, m = 1, ..., 10$. In this case the pulses cover the entire domain.

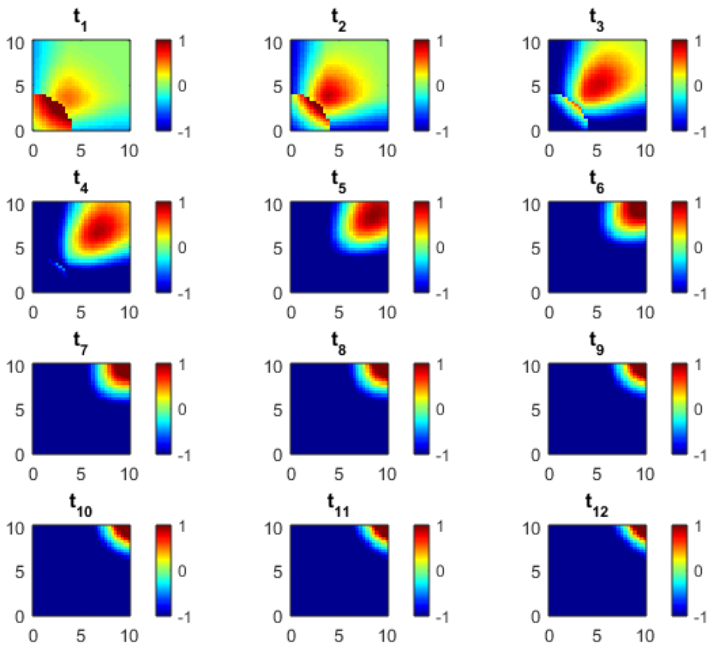


Figure 3.18: Snapshots of the first twelve timesteps of the solution generated by the kernel in 3.3 using initial condition 3.5 with $c = 4$.

3.5.2 Numerical accuracy of the Reduced Row Echelon Form algorithm

In the previous example the reduction of $\mathbf{A}$ to $\tilde{\mathbf{A}}$ was very straightforward. For this example however, the tolerance of the algorithm has an influence. The tolerance determines if a certain column is zeroed out: if the largest element in a certain column is smaller than the tolerance, then the column is zeroed out. In the previous example we have varied the tolerance and we have found no difference between using the default tolerance and taking a tolerance as large as tol $= 1 \cdot 10^{-1}$. For this example we have adjusted the tolerance because the default tolerance still leads to bad numerical results and a relatively high condition number. We compare the results for Tikhonov regularization versus the subsampling method as well as the subsampling method combined with Tikhonov regularization for all three cases. We show the condition number and the error for varying tolerance in Figure 3.20 for the case where we have only 1 sequence. We show the results for the two cases where we use multiple sequences for setup 1 respectively setup 2 in figures 3.21 and 3.22.

In Figure 3.20 we see that the error is barely influenced by adding Tikhonov

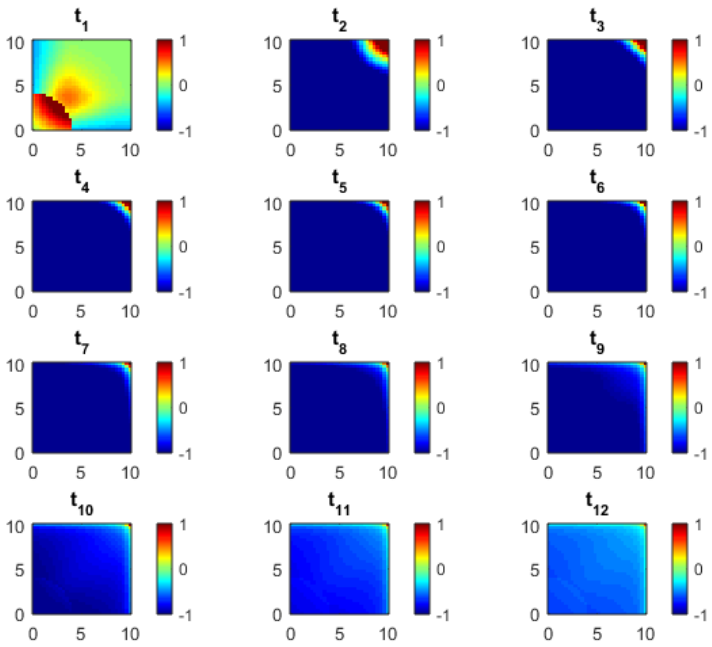
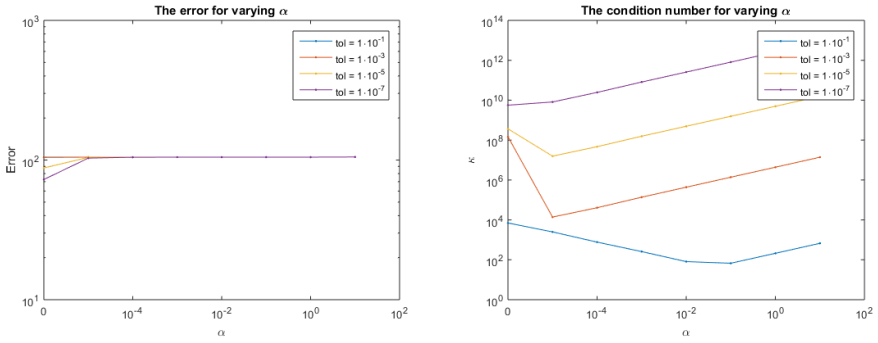
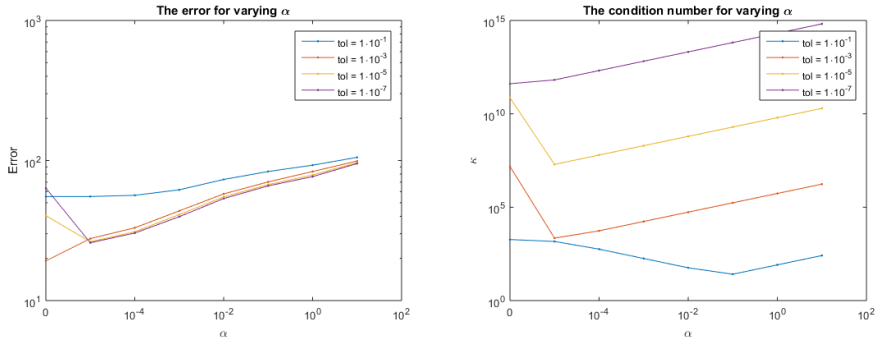


Figure 3.19: Snapshots of the full time window of the solution generated by the kernel in 3.3 using initial condition 3.5 with $c = 4$.



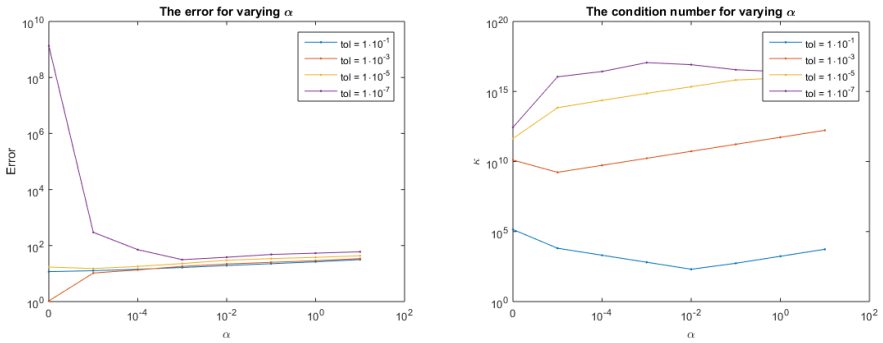
Error Condition number

Figure 3.20: Influence of the tolerance in the RREF algorithm for 1 sequence. Each line corresponds to a certain tolerance as indicated by the legend.



Error Condition number

Figure 3.21: Influence of the tolerance in the RREF algorithm for setup 1. Each line corresponds to a certain tolerance as indicated by the legend.

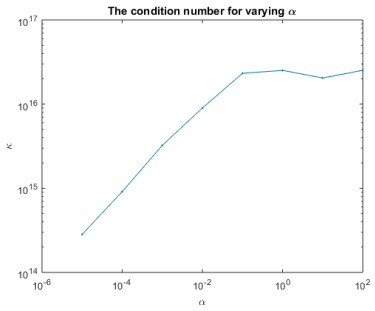


Error Condition number

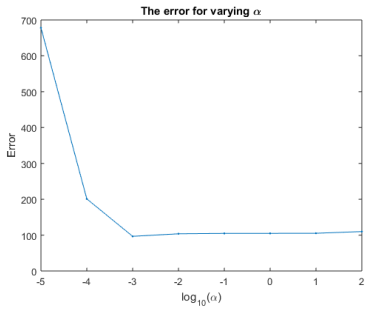
Figure 3.22: Influence of the tolerance in the RREF algorithm for setup 2. Each line corresponds to a certain tolerance as indicated by the legend.

regularization. The error is lower for a lower tolerance. This means that by increasing the tolerance we cut out information that is of importance for the reconstruction. When we look at the condition number we see a drastic decrease when we increase the tolerance. We also see a benefit in using Tikhonov regularization in addition to the subsampling method.
In Figure 3.21 we see the benefits of combining sequences to reconstruct the kernel. The error is overall lower than the errors shown in 3.20. It can also be seen that as we increase the tolerance, initially, the error decreases. However, when the tolerance becomes too big, in this case tol $= 10^{-1}$, the error increases again. This means that we cut out too much information. When looking at the condition number, we see that for tol $= 10^{-1}$ and tol $= 10^{-3}$ the condition num-

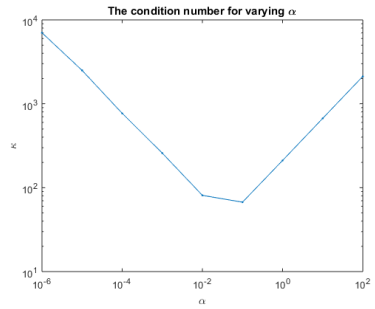
ber is lower than in 3.20 whereas the condition number for tol $= 10^{-5}$ and tol $= 10^{-7}$ is higher. This is not unexpected, as using multiple sequences in principle increases the ill-posedness. This is due to the fact that ℓ (discretization in time) becomes almost as large, or even larger than N (discretization in space). We again see benefits in using Tikhonov regularization in addition to subsampling in terms of the condition number. For the error we see that for sufficiently large tolerance, i.e. 10^{-1} and 10^{-3}, this is not the case. When the tolerance is too small, i.e. 10^{-5} and 10^{-7}, additional Tikhonov regularization does benefit. We conclude that choosing the right tolerance is essential in this case. From our results it is however not straightforward what tolerance to choose. This strongly depends on the preference of accuracy over stability or vice versa.
Figure 3.22 shows the results when we combine sequences that are generated from initial pulses that cover the entire domain. Here we have $\ell \gg N$ and the problem is severely ill-posed. We see again that increasing the tolerance strongly reduces the condition number, where in the case that tol $= 10^{-1}$ and tol $= 10^{-3}$ additional Tikhonov regularization decreases the condition number even more. We can see that the problem becomes severely ill-posed when we look at the condition number, which is much larger than in 3.20 and 3.21 for low tolerances. When we look at the error we again see that when the tolerance gets too large, in this case tol $= 10^{-1}$, the error increases. The lowest error we get is when we use tol $= 10^{-3}$ and $\alpha = 0$. The error for this case is 1.05. We do however get a very large condition number, namely $\kappa = 1.27 \cdot 10^{10}$. The result is shown in figure 3.26.
For this example the results are not conclusive. We see that using subsampling makes the problem solvable and reduces the ill-posedness. However, for all setups there is a strong trade-off between error and stability, in terms of the condition number. In figures 3.23, 3.24 and 3.25 we compare the solution using Tikhonov regularization versus subsampling, and subsampling in combination with Tikhonov regularization for a given tolerance.



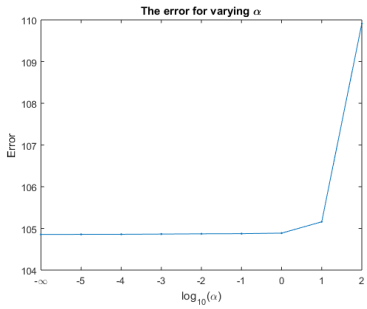
$\kappa(\mathbf{R}_\alpha)$ for varying α



Error for $\mathbf{R}_\alpha$

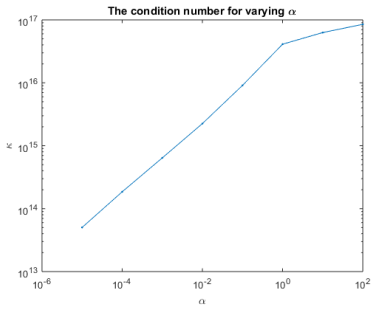


$\kappa(\tilde{\mathbf{R}}_\alpha)$ for varying α

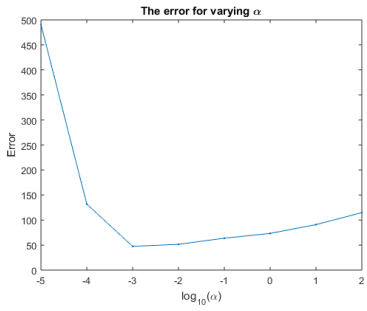


Error for $\tilde{\mathbf{R}}_\alpha$

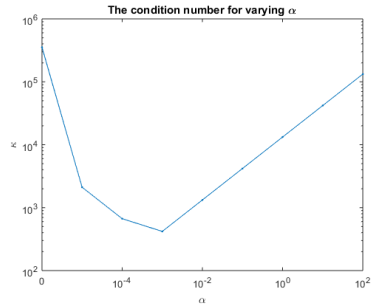
Figure 3.23: Error and condition number for varying α for reconstruction with 1 sequence and tol = 10^{-1}.



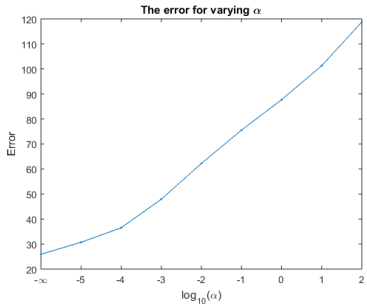
$\kappa(\mathbf{R}_\alpha)$ for varying α



Error for $\mathbf{R}_\alpha$

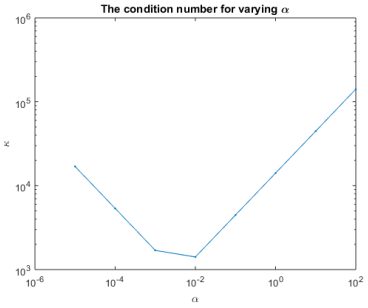


$\kappa(\tilde{\mathbf{R}}_\alpha)$ for varying α

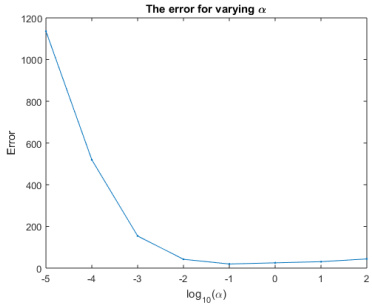


Error for $\tilde{\mathbf{R}}_\alpha$

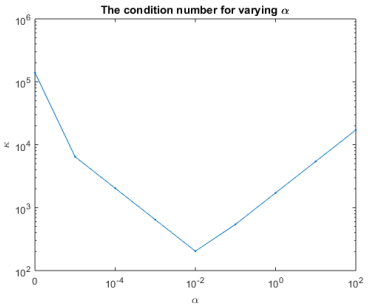
Figure 3.24: Error and condition number for varying α for reconstruction with multiple sequences using setup 1 and tol = 10^{-2}.



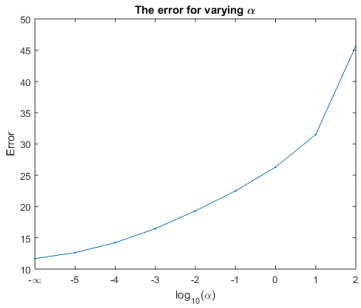
$\kappa(\mathbf{R}_\alpha)$ for varying α



Error for $\mathbf{R}_\alpha$



$\kappa(\tilde{\mathbf{R}}_\alpha)$ for varying α



Error for $\tilde{\mathbf{R}}_\alpha$

Figure 3.25: Error and condition number for varying α for reconstruction with multiple sequences using setup 2 and tol $= 10^{-1}$.

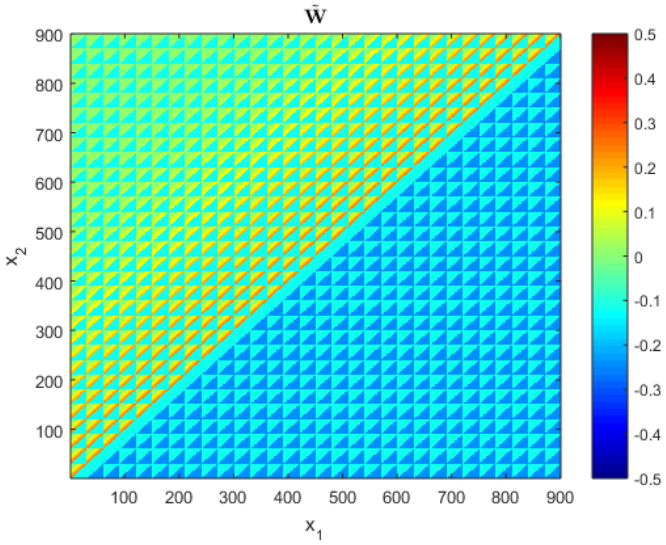


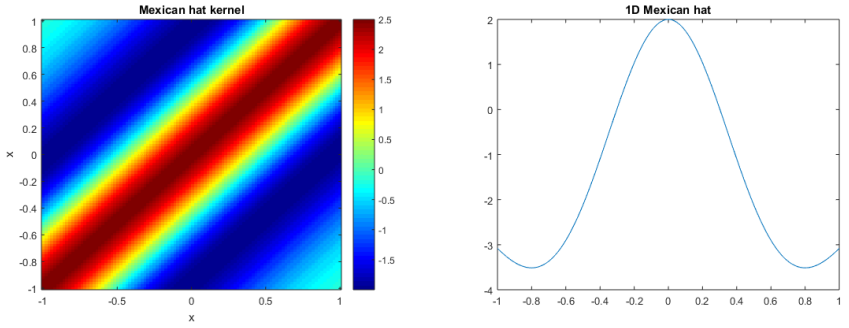
Figure 3.26: The reconstructed kernel using multiple sequences and setup 2 with $\alpha = 0$ and tol $= 10^{-3}$. Compare figure 3.17

3.6 Mexican hat

In this section we extend to a kernel where we have two populations of neurons, namely excitatory and inhibitory. A typical example of a kernel where we have two populations is called the Mexican hat. This is a 1D example where the kernel has the form shown in Figure 3.27. We use the work of Dijkstra et al. [7] where bifurcations for this type of connectivity have been studied. One big difference is that we do not have any delays, so that our simulation of the Amari equation differs from theirs. For example, without any delays, we are not able to generate solutions that oscillate over the temporal domain. We have however used their setup, i.e. we use the same spatial domain, the same connectivity function, the same firing function and use their initial conditions. We use the parameter $\beta = 3$ and the firing function $f(x) = \dfrac{1}{1+\exp(-\beta x)} - \dfrac{1}{2}$. Here, we have included the constant c, which is chosen $c = \dfrac{1}{2}$ to ensure that $f(0) = 0$, see [6]. We try to reconstruct the following prescribed kernel:

$$w(x,y) = 12.5e^{-2|x-y|} - 10e^{-|x-y|} \tag{3.6}$$

The kernel is shown in Figure 3.27.



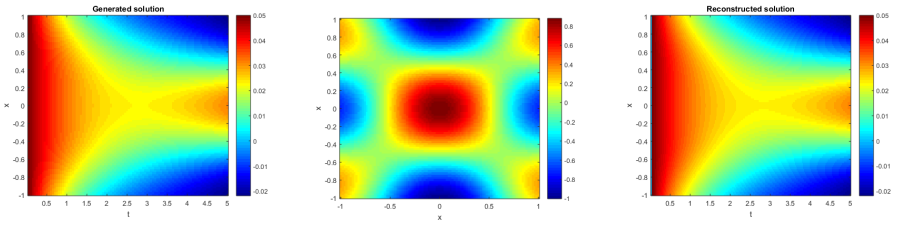
Mexican hat 1D Full kernel for the Mexican hat

Figure 3.27: Prescribed kernel $w(x,y)$.

We will use the following initial conditions, as used in [7]:

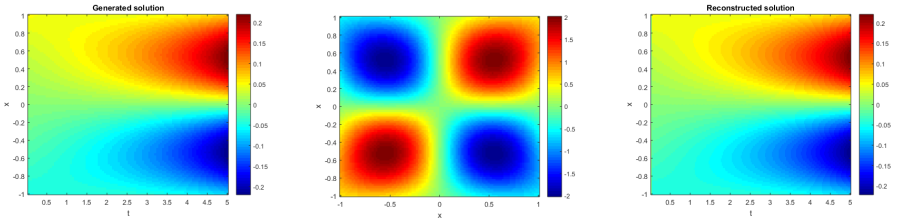
$$u(x,0) = 0.05 \tag{3.7}$$
$$u(x,0) = 0.05(0.99x + 0.01) \tag{3.8}$$
$$u(x,0) = 0.05(0.7\cos(4.3x) + 0.3\cos(2.04x)) \tag{3.9}$$

The spatial domain is given by $\Omega = [-1,1]$. Furthermore, we choose $T = 5$ and $\alpha = 1$. Here, our temporal domain is much smaller than the one in [7], but this doesn't influence the inverse problem. The results are shown in 3.28, 3.29 and 3.30.



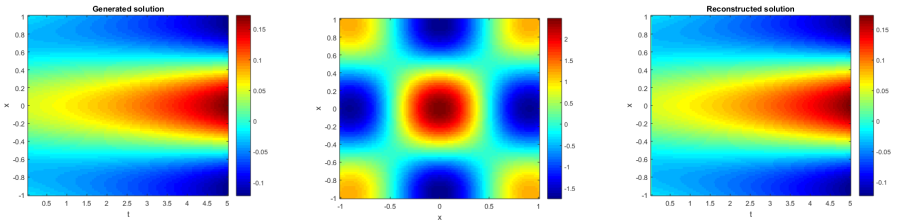
Generated sequence Reconstructed kernel Reconstructed sequence

Figure 3.28: Images for initial condition 3.7.



Generated sequence Reconstructed kernel Reconstructed sequence

Figure 3.29: Images for initial condition 3.8.



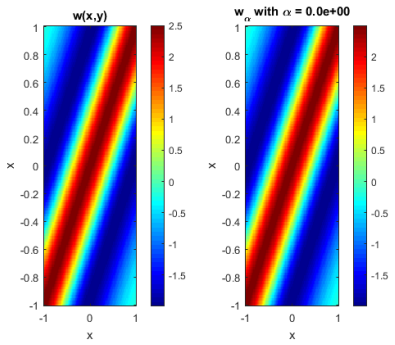
Generated sequence Reconstructed kernel Reconstructed sequence

Figure 3.30: Images for initial condition 3.9.

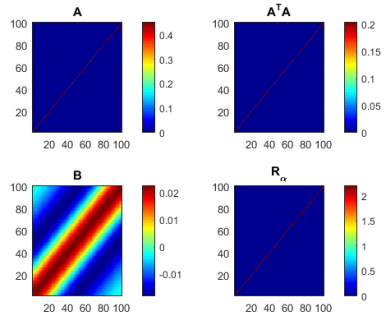
From these figures we see that we are not able to reconstruct the kernel. What is interesting is that although we are not able to reconstruct the kernel, we are able to reconstruct the generated sequence. This is due to ambiguity in the forward model.

The first thing we show is that we can nearly perfectly reconstruct the kernel when we use δ-pulses located at each point of the domain as initial conditions, and only observe one time step. This is very straightforward, since each pulse yields a column linearly independent of all other columns in the matrix $\mathbf{A}$. Since we use a δ-pulse on every point of the domain and only observe one time step, we have a bijection and can invert the kernel. The reconstructed kernel and the matrices $\mathbf{A}$, $\mathbf{B}$ and $\mathbf{R}_\alpha$ with $\alpha = 0$ are shown in Figure 3.31. The error between the prescribed kernel and the reconstructed kernel is $2.87 \cdot 10^{-12}$ and

$\kappa(\mathbf{R}_\alpha) = 1$.



Kernel and reconstructed kernel



Matrices $\mathbf{A}$, $\mathbf{A^T A}$, $\mathbf{B}$ and $\mathbf{R}_\alpha$

Figure 3.31: Reconstruction of the Mexican hat using δ-pulses.

We want to see next how well we can reconstruct the kernel when we use less data. We generate sequences using the initial conditions 3.7, 3.8 and 3.9. Again we compare the subsampling versus subsampling combined with Tikhonov regularization. We present the results for all initial conditions in Figure 3.33.

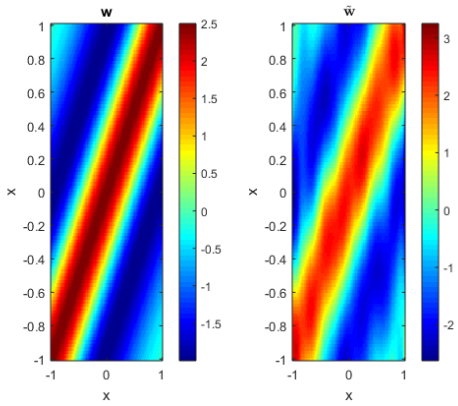


Figure 3.32: Reconstructed kernel using subsampling and tol $= 10^{-7}$ using initial condition 3.8.

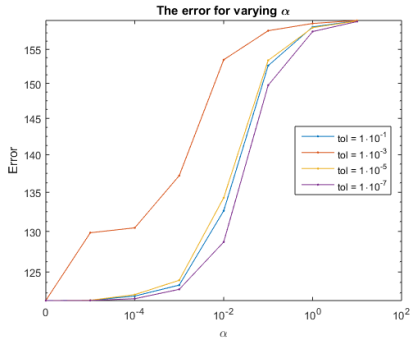
We see that we again reduce the condition number in all cases. For the error we see that combining subsampling and Tikhonov has adverse effects: the error only increases. For the condition number we see that combining subsampling and Tikhonov reduces the condition number compared to only using subsampling. For the sequences generated by initial conditions 3.7 and 3.9 the error is

roughly the same as using Tikhonov regularization. For initial condition 3.8 we see that for tol $= 10^{-7}$ and $\alpha = 0$ the error is much lower compared to using Tikhonov regularization. However, for this tolerance we still have a very high condition number. The reconstructed kernel is shown in 3.32.

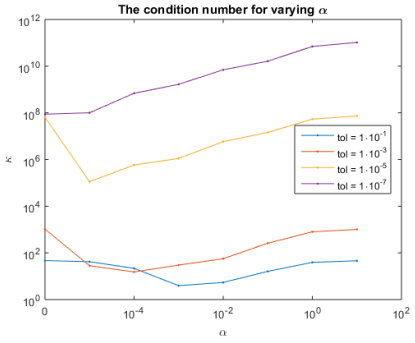
To check the informativeness of the sequences, we check the rank of the matrix $\mathbf{A}$. For the sequences generated by the initial conditions 3.7-3.9 the rank of the matrices $\mathbf{A}$ are 10, 13 and 11. Since we have $N = 100$, we see via 2.22 that we are solving a system of equations with 100×100 variables and 10, 13 or 11 equations. To come to a better reconstruction we combine sequences to make the matrix $\mathbf{A}$ more informative. From figures 3.28 and 3.30 we see that the sequences generated by initial conditions 3.7 and 3.9 are qualitatively very similar. Combining them will therefore not lead to a better reconstruction. Combining the sequences generated by combining initial conditions 3.7 and 3.8 or 3.7 and 3.9 will lead to better results. We look at the condition number and the error we get by combining these sequences in Figure 3.34.

We see that by combining sequences that are qualitatively different we reduce the error and the condition number, and have better results than when we use Tikhonov regularization. We still see that combining subsampling and Tikhonov regularization reduces the condition number, and that for lower tolerances the error is lower. The ranks for the matrices $\mathbf{A}$ corresponding to the sequences generated in Figure 3.34 are 14, 21 and 22. We see that the rank of the matrix $\mathbf{A}$ increases. We have seen in Figure 3.32 that for the matrix $\mathbf{A}$ with rank 13 we get a reasonable reconstruction. When we combine initial conditions 3.7 and 3.9 the matrix $\mathbf{A}$ has rank 14, but the reconstruction is worse. Hence, we cannot conclude that the rank is a sufficient measure for informativeness.
We see no benefit by combining subsampling and Tikhonov regularization in terms of the error. We show the reconstructed kernel for combining initial conditions 3.7 and 3.8 with tolerance tol $= 10^{-2}$ and $\alpha = 0$ in Figure 3.35 as well as the matrices $\mathbf{A}$, $\tilde{\mathbf{A}}$ and $(\tilde{\mathbf{A}}^{\mathbf{T}}\tilde{\mathbf{A}})^{-1}\tilde{\mathbf{A}}^{\mathbf{T}}$. Although we get fairly low condition number, we do see that there is very little structure in the matrix $(\tilde{\mathbf{A}}^{\mathbf{T}}\tilde{\mathbf{A}})^{-1}\tilde{\mathbf{A}}^{\mathbf{T}}$ and that the values get very big.

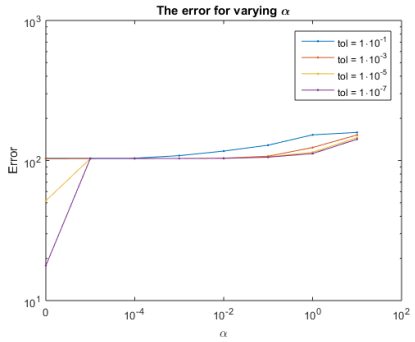
We conclude that we again improve the quality of the reconstruction in terms of error and stability when using the subsampling method instead of using Tikhonov regularization. We also see that increasing the tolerance yields a lower condition number, but a larger error. We thus have to choose between accuracy and stability.



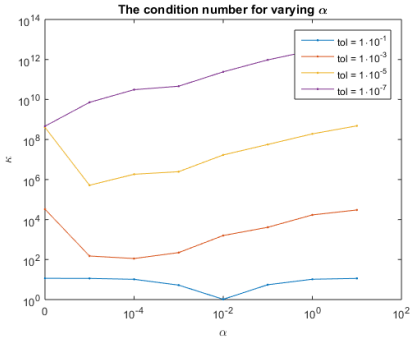
The error using $\mathbf{R}_\alpha$ with $\alpha = 10^{-3}$ is $1.2 \cdot 10^2$



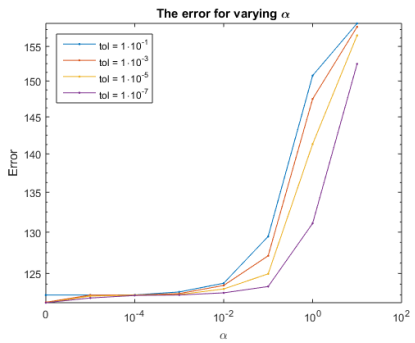
$\kappa(\mathbf{R}_\alpha)$ with $\alpha = 10^{-3}$ is $1.7 \cdot 10^{30}$



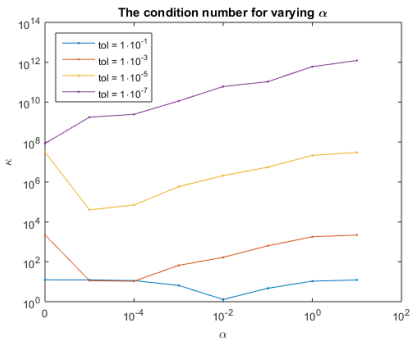
The error using $\mathbf{R}_\alpha$ with $\alpha = 10^{-1}$ is $1.0 \cdot 10^2$



$\kappa(\mathbf{R}_\alpha)$ with $\alpha = 10^{-1}$ is $1.4 \cdot 10^{17}$

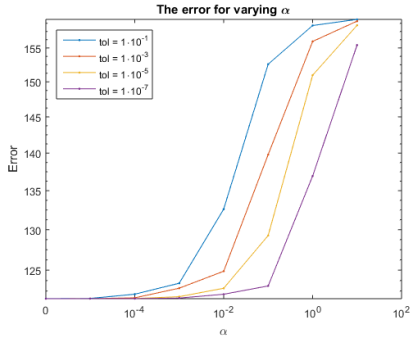


The error using $\mathbf{R}_\alpha$ with $\alpha = 10^{-1}$ is $1.2 \cdot 10^2$

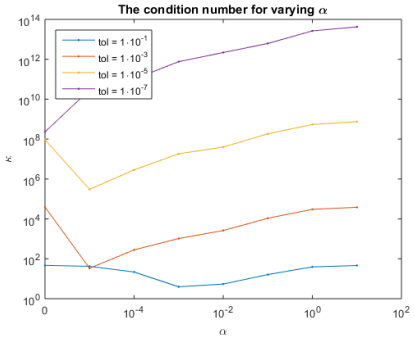


$\kappa(\mathbf{R}_\alpha)$ with $\alpha = 10^{-1}$ is $5.0 \cdot 10^{17}$

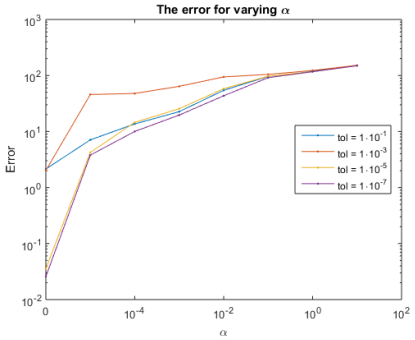
Figure 3.33: Error and condition number. Each color corresponds to a given tolerance as indicated by the legend. The top figure refers to initial condition 3.7, the middle figure to initial condition 3.8 and the bottom figure to initial condition 3.9.



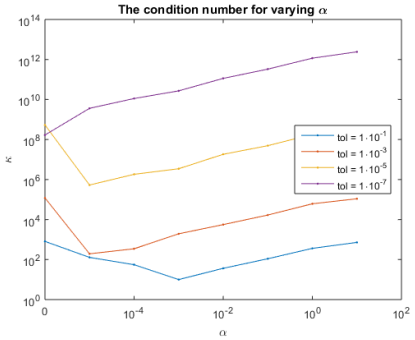
The error using $\mathbf{R}_\alpha$ with $\alpha = 10^{-1}$ is $1.2 \cdot 10^2$



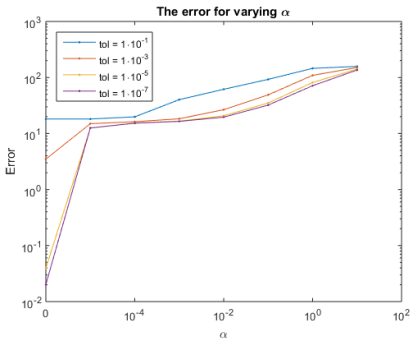
κ using $\mathbf{R}_\alpha$ with $\alpha = 10^{-1}$ is $4.3 \cdot 10^{16}$



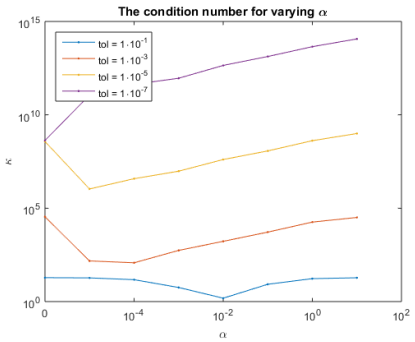
The error using $\mathbf{R}_\alpha$ with $\alpha = 10^{-1}$ is $3.1 \cdot 10^1$



κ using $\mathbf{R}_\alpha$ with $\alpha = 10^{-1}$ is $1.2 \cdot 10^{15}$

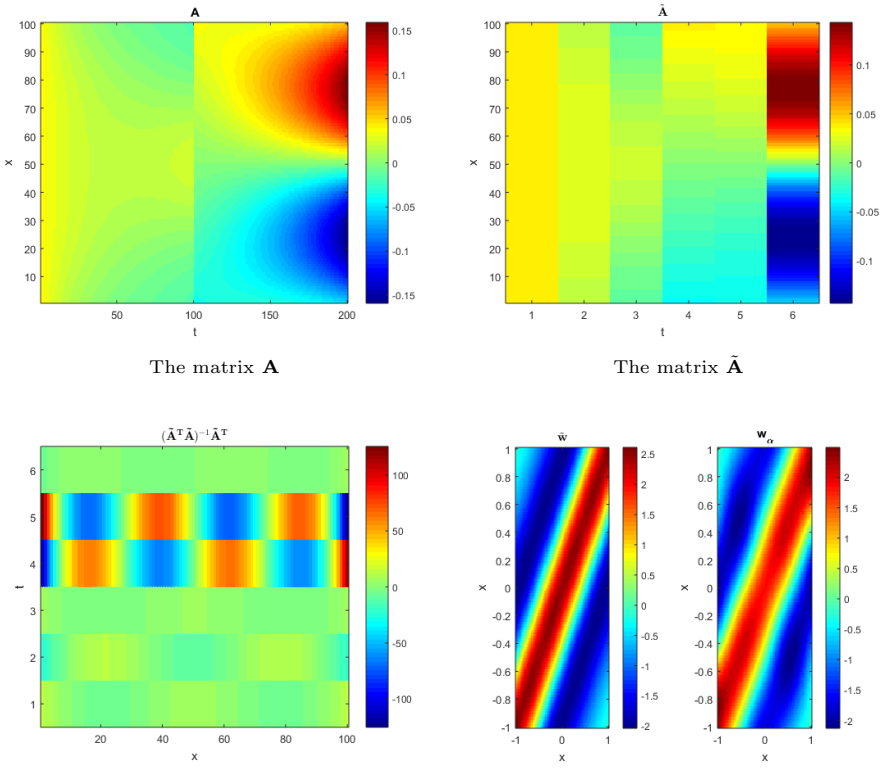


The error using $\mathbf{R}_\alpha$ with $\alpha = 10^{-1}$ is $2.5 \cdot 10^1$



κ using $\mathbf{R}_\alpha$ with $\alpha = 10^{-1}$ is $1.0 \cdot 10^{15}$

Figure 3.34: Error and condition number. The top figure corresponds to combining initial conditions 3.7 and 3.9, the middle figure to combining initial conditions 3.7 and 3.8 and the bottom figure to combining initial condition 3.8 and 3.9.



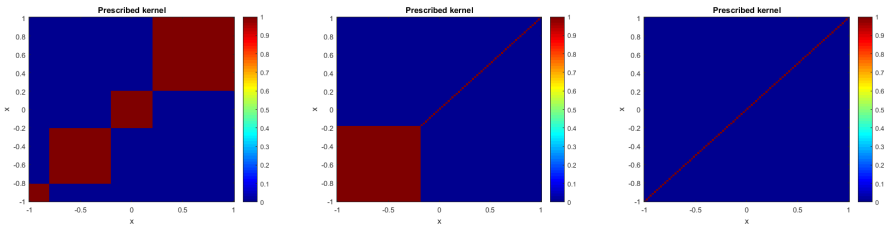
The matrix $\mathbf{A}$ | The matrix $\tilde{\mathbf{A}}$

The matrix $(\tilde{\mathbf{A}}^{\mathbf{T}}\tilde{\mathbf{A}})^{-1}\tilde{\mathbf{A}}^{\mathbf{T}}$ | Subsampling versus Tikhonov regularization

Figure 3.35: Relevant matrices for reconstruction using tol $= 10^{-2}$ for combining initial conditions 3.7 and 3.8.

3.7 Other kernels

We now show some results on kernels as described in [8]. We work with simplified versions, but preserve the overall structure of these kernels. In this section we will not show the results for different tolerances for the RREF algorithm, but we will find the tolerance yielding the best results and present it. The three prescribed kernels we use are shown in Figure 3.36. The names for the kernels are taken from [8] and in this work we will refer to them as such.



Complexity Integration Entropy

Figure 3.36: Prescribed kernels we try to reconstruct.

We reconstruct these kernels using the following initial conditions:

$$u(x,0) = 0.05 \tag{3.10}$$
$$u(x,0) = 0.05x \tag{3.11}$$
$$u(x,0) = \sin\left(\frac{2}{\pi}\right) \tag{3.12}$$

The images for the reconstructed kernels are shown in the appendix in figures A.1, A.2 and A.3. Here, we show the error and condition number in the Tables 3.3, 3.4 and 3.5. We see in Table 3.3 that for the initial condition $u(x,0) = 0.05$ we can reconstruct the kernel nicely when using subsampling. For the other initial conditions this is not the case. We furthermore see from Tables 3.4 and 3.5 that we are not able to nicely reconstruct the Entropy and Integration kernel when using subsampling or Tikhonov. The condition number 1 in Table 3.5 is due to the fact that through subsampling only 1 column is left. This means that the corresponding sequence is terribly non-informative.

Localization

To improve the quality of the reconstruction of the Entropy kernel we use localization. For the Entropy kernel we show the results for different ρ and $u(x,0) = 0.05$ in Figure 3.37.

We see that in our case the localization is strongly influenced by the radius we use. Radii of $0.01 \leq \rho < 0.02$ yield a good reconstruction, but for larger ρ

$u(x,0)$	Error	Condition number
0.05	$2.07 \cdot 10^{-9}$	$1.45 \cdot 10^{3}$
$0.05x$	20.31	$5.7 \cdot 10^{3}$
$\sin\left(\frac{2}{\pi}\right)$	38.83	$5.2 \cdot 10^{4}$

$u(x,0)$	Error	Condition number
0.05	24.96	$1.28 \cdot 10^{16}$
$0.05x$	55.88	$1.36 \cdot 10^{16}$
$\sin\left(\frac{2}{\pi}\right)$	40.23	$5.85 \cdot 10^{15}$

Table 3.3: Tables showing the error and the condition number for a given initial condition for the Complexity kernel. The left corresponds to subsampling, the right one to Tikhonov regularization.

$u(x,0)$	Error	Condition number
0.05	9.95	1
$0.05x$	9.95	1
$\sin\left(\frac{2}{\pi}\right)$	9.85	$1.58 \cdot 10^{3}$

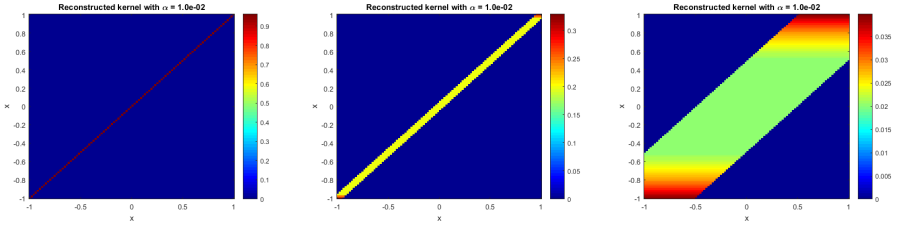
$u(x,0)$	Error	Condition number
0.05	9.95	$1.17 \cdot 10^{16}$
$0.05x$	9.95	$1.99 \cdot 10^{15}$
$\sin\left(\frac{2}{\pi}\right)$	10.93	$5.32 \cdot 10^{15}$

Table 3.4: Tables showing the error and the condition number for a given initial condition for the Entropy kernel. The left corresponds to subsampling, the right one to Tikhonov regularization.

$u(x,0)$	Error	Condition number
0.05	7.62	$8.54 \cdot 10^{1}$
$0.05x$	7.42	$1.08 \cdot 10^{3}$
$\sin\left(\frac{2}{\pi}\right)$	7.61	$2.99 \cdot 10^{3}$

$u(x,0)$	Error	Condition number
0.05	7.65	$2.87 \cdot 10^{16}$
$0.05x$	7.93	$9.53 \cdot 10^{16}$
$\sin\left(\frac{2}{\pi}\right)$	8.05	$9.02 \cdot 10^{15}$

Table 3.5: Tables showing the error and the condition number for a given initial condition for the Integration kernel. The left corresponds to subsampling, the right one to Tikhonov regularization.

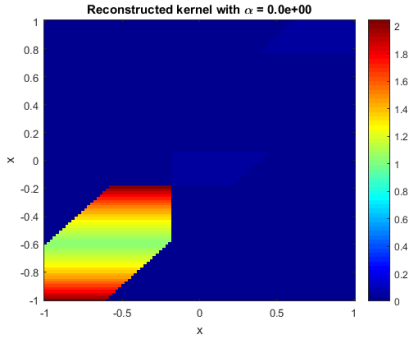


Result for $\rho = 0.01$ Result for $\rho = 0.05$ Result for $\rho = 0.5$

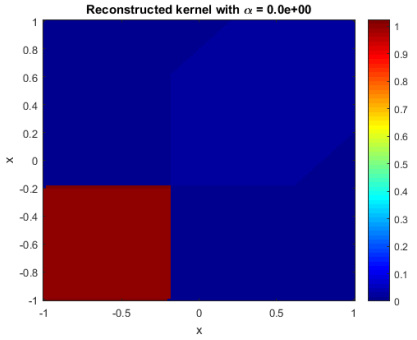
Figure 3.37: Results for varying ρ and $\nu = 0.05$.

we see that the kernel shows connections between all points within the given radius. Combining localization with subsampling produces better results than

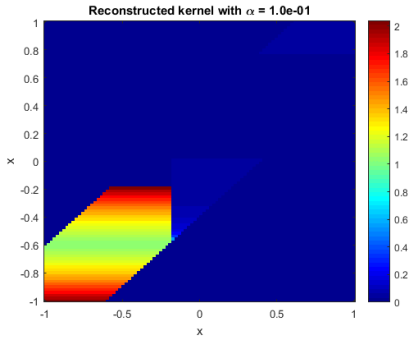
localization alone and we only need one measurement in time in this case. For $\rho = 0.01$ the error for the method without subsampling it is $3.53 \cdot 10^{-1}$ and the error for the method with subsampling is $3.16 \cdot 10^{-13}$. This shows that in this example, localization only works when we very precisely know the scale of the local connections.
We have also tried to improve the quality of the reconstruction for the Integration kernel. The results are shown in Figure 3.38. This is however unsuccessful and the only way we are able to reconstruct this kernel is via δ-pulses.



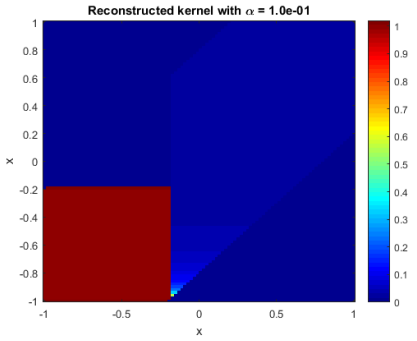
$\tilde{\mathbf{W}}$ with $\rho = 0.4$.



$\tilde{\mathbf{W}}$ with $\rho = 0.8$.



$\mathbf{W}_\alpha$ with $\rho = 0.4$.



$\mathbf{W}_\alpha$ with $\rho = 0.8$.

Figure 3.38: Reconstructed kernels for $u(x, 0) = 0.05$.

Chapter 4

Conclusion and outlook

4.1 Conclusion

In this work we have built upon the work of Potthast and beim Graben [1]. We have used their methods to solve the problem and extended it using three new solution techniques, namely:

1. Parameter optimization
2. Subsampling
3. Combining data

We have shown in sections 3.2, 3.3 and 3.4 that for prescribed sequences we strongly benefit from optimizing over the parameters of the firing function. For prescribed kernels, the choice of the parameters of the firing function is of less influence.
In section 2.1 we have recited the results in [1] and derived the subsampling as a tool for reducing the ill-posedness. Throughout this work, we have shown that this method of regularization is easily applicable and outperforms Tikhonov regularization. We have also found that we benefit from combining the two methods in terms of stability via a lower condition number. For the error however, subsampling alone is the best method. We have also shown that the tolerance in algorithm 2 influences the error and the condition number as well. Increasing the tolerance reduces the condition number, but the error increases. This is a classic trade-off between accuracy and stability that arises when using regularization.
In sections 3.5 and 3.6 we have shown that we can strongly benefit from combining data to reconstruct the kernel. In section 3.5 we have shown that combining sequences can overcome gaps in the reconstructed kernel. These gaps arise when the sequence has parts where it shows no activity, i.e. it is zero. When combining sequences that are complementary in terms of these gaps, we can reduce the error and the condition number. In section 3.6 we have shown that we can

reduce the error and condition number by combining sequences, when we deal with sequences that are not informative enough to reconstruct the kernel by themselves. The key in combining sequences is that they are qualitatively different. We have shown that combining sequences that look qualitatively similar does not yield better results. Furthermore, we have shown that we can always reconstruct a kernel using δ-pulses.
In section 3.7 we have shown in certain cases the ability to reconstruct the kernel depends entirely on the initial condition. We have shown an example where we were able to reconstruct the kernel using only one initial condition. For two other initial conditions we could not, and combining them did not benefit. We have applied localization in section 3.7 to reconstruct a kernel that has only local connections. This technique has shown to work only in this case when we precisely know the parameter ρ, and thus requires a great amount of prior knowledge.
We have furthermore shown an example, namely the Integration kernel, which we can only reconstruct using δ-pulses.

4.2 Outlook

For future work we should look into a more sophisticated method for selecting useful data. We now use the Reduced Row Echelon Form, via algorithm 2, but we would like a method that has a better measure for determining whether a measurement is useful or not. We have shown some difficulties in determining the right tolerance and would like a method that is more stable. The RREF algorithm furthermore selects on a "'left-to-right"' basis and does not distinguish between elements of a linearly dependent class. This becomes important when the tolerance of the algorithm starts playing a role. Furthermore, the RREF algorithm takes up a significant amount of computation time that hopefully can be reduced with different methods. The methods we are looking for are closely related to compressive sensing, where subsampling methods are widely used [14], [15], [16]. Recently it has been shown that subsampling can have a regularizing effect [17].
Another aspect that can be improved is the understanding of the informativeness of a sequence. We have argued that the rank is a good indication, but we have shown in 3.5 that this is not a sufficient measure for informativeness. We have shown that for two sequences with almost the same rank the reconstruction of the kernel using the one sequence was considerably better than using the other. For further research we would also like to look into different prescribed sequences and prescribed kernels known from Neural Field Theory, or possibly real data. Furthermore, we would like to show the effects that noise in the data has on the inverse problem.

Bibliography

[1] R. Potthast and P. beim Graben, *Inverse Problems in Neural Field Theory*, SIAM J. Applied Dynamical Systems, Vol. 8, No. 4, pp. 1405-1433

[2] P. beim Graben and R. Potthast, *Inverse Problems in Dynamic Cognitive Modeling*, Chaos, 19(1):015103, 2008

[3] S. Amari, *Homogeneous nets of neuron-like elements* Biological Cybernetics, 17:211220, 1975.

[4] S. Amari, *Dynamics of pattern formation in lateral-inhibition type neural fields*, Biolog. Cybernet., 27 77-87, 1977

[5] D. Colton and R. Kress, *Inverse acoustic and Electromagnetic Scattering Theory*, Springer Verlag, New York, Berlin, 1998.

[6] G. Faye and O. Faugeras, *Some theoretical and numerical results for delayed neural field equations*, Physica D 239(9)(2010) 561-578

[7] K. Dijkstra, S.A. van Gils, S.G. Janssens, Yu.A. Kuznetsov, S.Visser *Pitchfork-Hopf bifurcations in 1D neural field models with transmission delays*, Physica D 297(2015) 88-101

[8] O. Sporns and G. Tononi, *Classes of Network Connectivity and Dynamics* Complexity 7(2002), 28 38.

[9] R. L. Beurle, *Properties of a mass of cells capable of regenerating pulses*, Philosophical Transactions of the Royal Society London B, 240:5594, 1956

[10] J. S. Griffith, *A field theory of neural nets: I: Derivation of field equations*, Bulletin of Mathematical Biophysics, 25:111120, 1963.

[11] J. S. Griffith, *A field theory of neural nets: II: Properties of field equations*, Bulletin of Mathematical Biophysics, 27:187195, 1965.

[12] H.R. Wilson and J.D. Cowan, *Excitatory and inhibitory interactions in localized populations of model neurons*, Biophysical Journal, 12:124, 1972.

[13] H.R. Wilson and J.D. Cowan, *A mathematical theory of the functional dynamics of cortical and thalamic nervous tissue* , Kybernetik, 13:5580, 1973.

[14] S. Foucart, H. Rauhut, *A Mathematical Introduction to Compressive Sensing*, Springer, New York, Heidelberg, Dordrecht, London, 2013.

[15] J. Romberg, *Compressive Sensing by Random Convolution*, SIAM J. Imaging Sciences, Vol. 2, No. 4, pp. 1098-1128

[16] D. Baron, S. Sarvotham and R.G. Baraniuk, *Bayesian Compressive Sensing Via Belief Propagation*, IEEE Transactions on Signal Processing, Vol. 58 No. 1, January 2010

[17] A. Rudi, R. Camoriano, L. Rosasco *Less is More: Nyström Computational Regularization*, CoRR, Vol. 1507.04717, 2015

[18] H.W. Engl, M. Hanke and A. Neubauer, *Regularization of Inverse Problems*, Springer, Berlin, 1996

[19] R. Kress *Linear Integral Equations*, Springer, Berlin, 1989

[20] G. Nakamura and R. Potthast, *Inverse Modeling*, IOP Publishing, 2015, Chapter 7

[21] G.H. Golub and C.F. van Loan, *Matrix Computations*, The Johns Hopkins University Press, Fourth Edition, 2013, p.87

Appendix A

Reconstruction for the Complexity, Entropy and Integration kernel

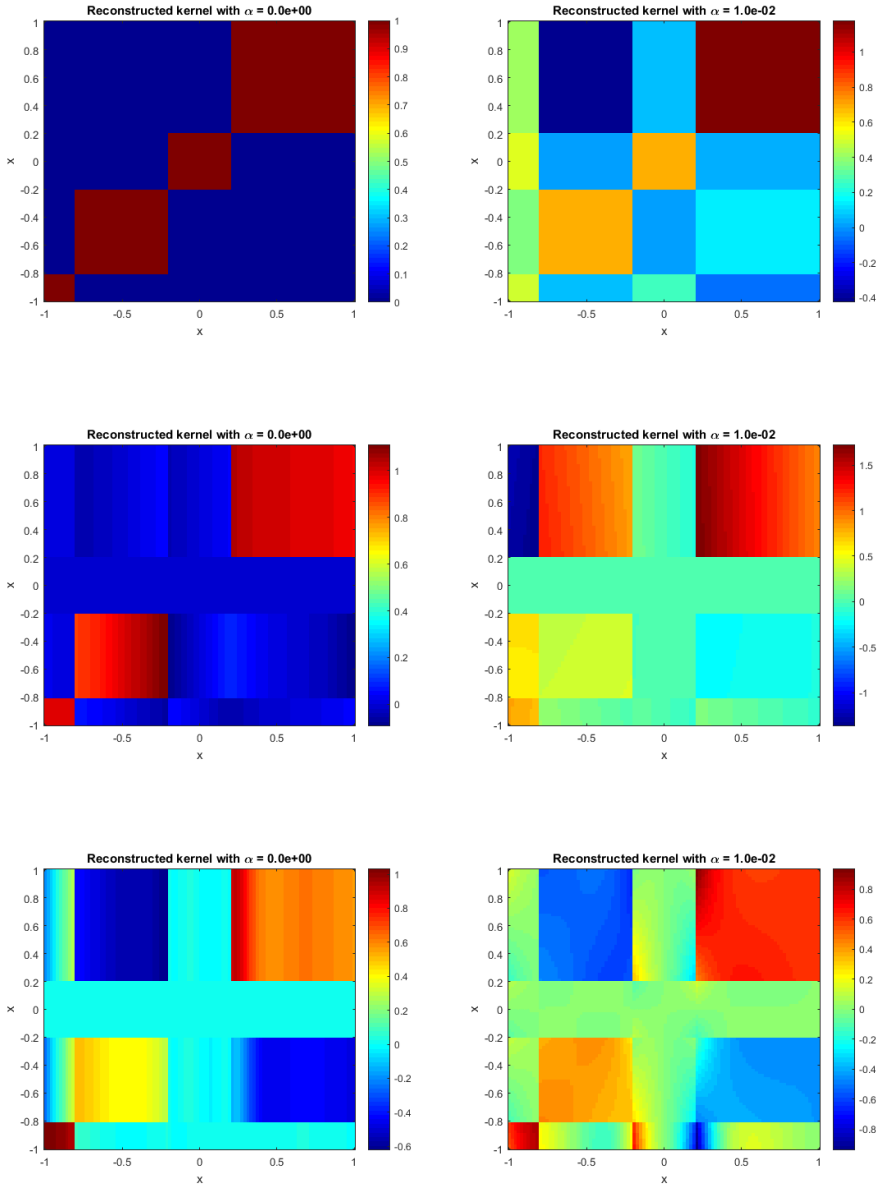


Figure A.1: Reconstructed kernels for the Complexity kernel. The left column corresponds to subsampling and the right column to Tikhonov regularization. The top figure is for $u(x,0) = 0.05$, the middle figure is for $u(x,0) = 0.05x$ and the bottom figure is for $u(x,0) = \sin\left(\frac{2}{\pi}\right)$.

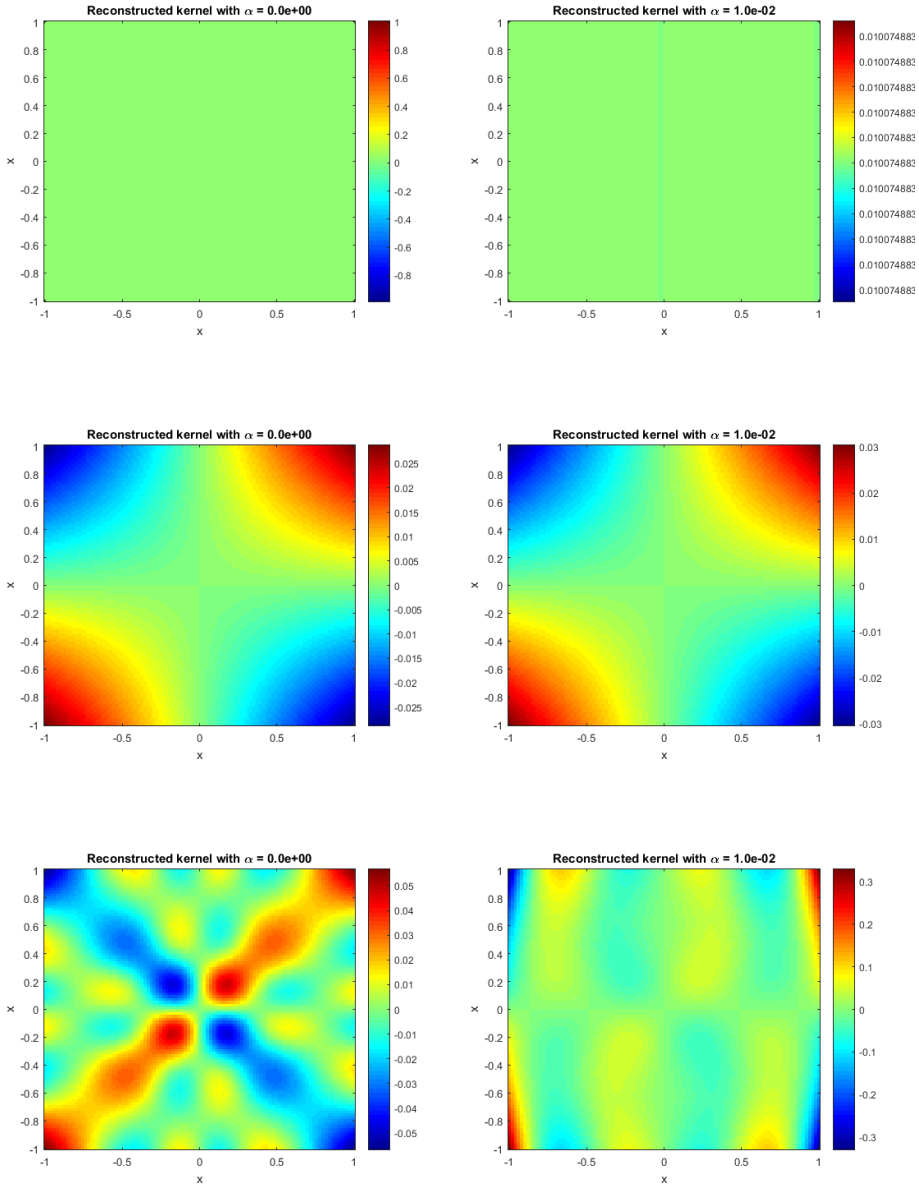


Figure A.2: Reconstructed kernels for the Entropy kernel. The left column corresponds to subsampling and the right column to Tikhonov regularization. The top figure is for $u(x,0) = 0.05$, the middle figure is for $u(x,0) = 0.05x$ and the bottom figure is for $u(x,0) = \sin\left(\frac{2}{\pi}\right)$.

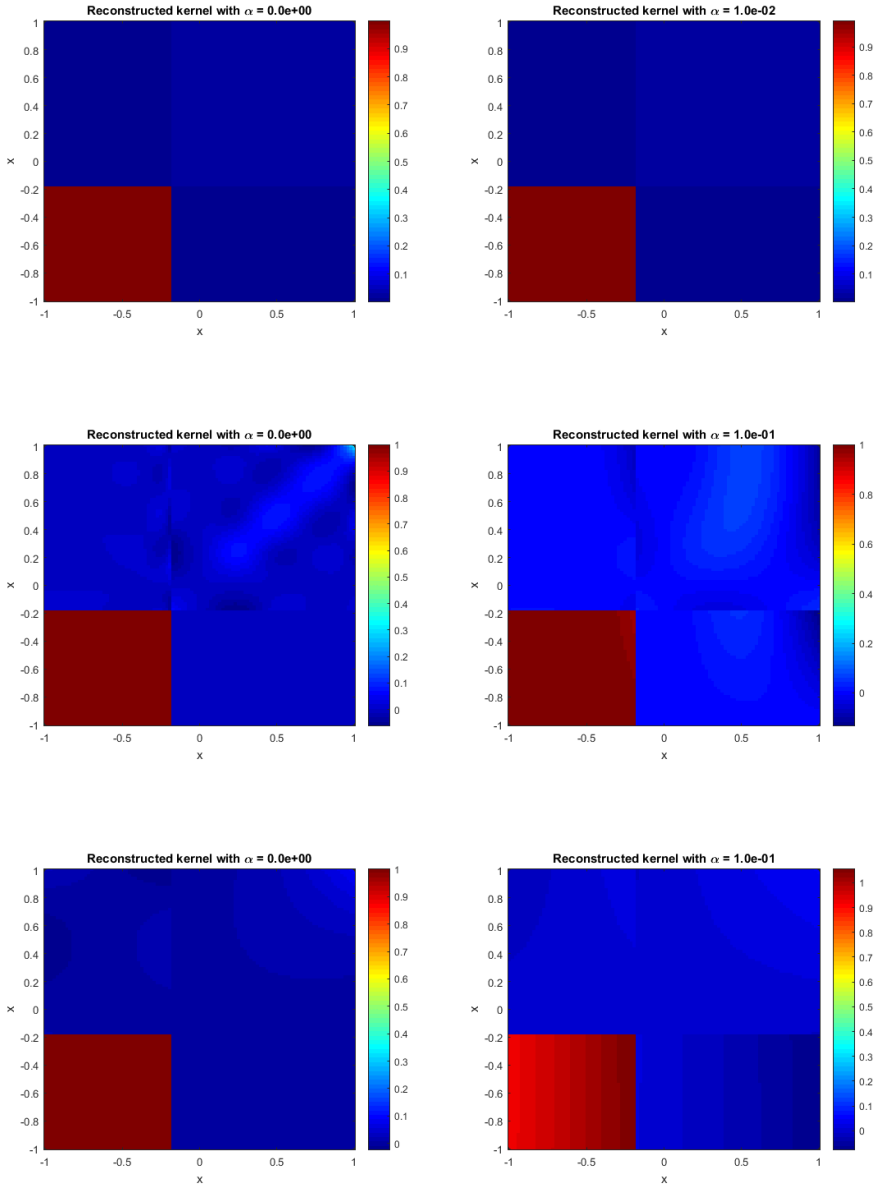


Figure A.3: Reconstructed kernels for the Integration kernel. The left column corresponds to subsampling and the right column to Tikhonov regularization. The top figure is for $u(x, 0) = 0.05$, the middle figure is for $u(x, 0) = 0.05x$ and the bottom figure is for $u(x, 0) = \sin\left(\frac{2}{\pi}\right)$.