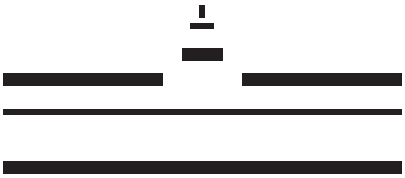

Westfälische
Wilhelms-Universität
Münster

Implementation and Analysis of Dynamic Causal Modeling for EEG/MEG Data

Diplomarbeit

vorgelegt von

Christian Himpe

Betreuer: Prof. Dr. Mario Ohlberger

Institut für Numerische und Angewandte Mathematik

Münster, Februar 2011

Contents

1 Introduction

1.1 Motivation

Analyzing connectivity of brain regions through data reflecting neuronal activity is an inverse problem that can be approached with a two component model, whose parameter distribution is estimated with Bayesian inference, based upon [22], utilizing an Expectation Maximization (EM) algorithm to estimate the parameter distribution under the given data. Dynamic Causal Modeling (DCM) is a procedure that allows to make these deductions about the interaction between neuronal populations as found in the brain, by testing hypothesis on the coupling beween these regions of the brain. Given an experiment that encourages the activity of the brain regions in question, DCM estimates the parameters of the inverse model that simulates the neuronal activity and thus can verify or falsify this hypothesis.

Dynamic Causal Modeling came into being in 2003 through a research paper by Karl Friston (see [17]). This was an extension of former research in the analysis of fMRI time series. Before, in a series of articles from 2000 to 2001 (see [14],[16],[15], [13]), the analysis of a single brain region with the hemodynamic Balloon model by the means of parameter estimation was established. Dynamic Causal Modeling combined this single region model with a multi region model that negotiates the coupling between the considered regions, while the hemodynamic (single region) model then converts the output of the multi region model to a measured response. In 2006 the concept of Dynamic Causal Modeling was extended from fMRI to EEG/MEG with a new mathematical model (see [6]). This model was initially developed by Jansen ([28]). In 2003 and 2005 ([8],[5]) this Jansen model was refined from a single region model to a multi region model by Oliver David and Karl Friston, and was named the neural mass model, which was then extended with various improvements (for example see [47],[31]).

1.2 Overview and Goal

The goal of this work is to summarize the inverse models of DCM for two methods of data acquisition (fMRI and EEG/MEG) and outline the composition of the employed EM-algorithm for the parameter distribution estimation procedure, as well as improving this algorithm in terms of performance. Furthermore, this work is accompanied by a modular implementation of the presented methods, that is tested with synthetic and real-life data (provided by the WWU Institute of Physiology I in Münster) and benchmarked against the original EM-algorithm.

My work consists of three parts. The first part presents the mathematical foundations of the discussed models and methods. The second part describes the implementation of the developed program as well as notes on the algorithmic methods that were used. The third part summarizes the tests with artificial and real data time series.

2 Inverse Modeling for Neuronal Activity

2.1 Concept

Dynamic Causal Modeling (DCM) is based on a parametrized, inverse, two component model. The neuronal activity and the coupling between the considered neuronal population is described by the dynamic submodel. The forward submodel converts the output of the dynamic submodel to a response that emulates measured signals. The parameters of these (sub-)models (dynamic and forward) are then estimated to fit the recorded data with a bayesian statistical inference approach, which approximates the distribution of the parameters utilizing an EM-algorithm.

The two data acquisition methods considered are functional Magnetic Resonance Imaging (fMRI) and Electroencephalography (EEG). The EEG model can also be applied to Magnetoencephalography (MEG). For further information about the data acquisition methods see [29].

2.1.1 Prior Probability Distribution

The parameter estimation is based on a bayesian approach that approximates the distribution of parameters under the given data, which was recorded during the associated experiments. The bayesian statistical inference requires a prior probability distribution on the parameters that are to be estimated. The prior probability distribution, or short prior, on the parameters is assumed to be gaussian und thus will be specified in terms of expectation and variance. Each of the presented submodels has a table of prior expectation and prior variance included, which are incorporated into the parameter distribution estimation process (see 2.3.1.2).

2.2 The Inverse Model

The inverse model consists of two submodels. The first submodel, the dynamic submodel, simulates the coupling of brain regions under deterministic input, which is provided by the experiment. The second submodel, a forward system, converts the simulated neuronal activity of the dynamic submodel to a measured response. The dynamic and forward submodel for fMRI and EEG differ quite significantly. The more simple fMRI model could be used for EEG with an adapted forward submodel, but since the EEG data conveys more information it can consequently be simulated with a more complex model.

2.2.1 The Dynamic Submodel

The dynamic submodel is a dynamic system that represents the change in the neuronal states of the observed brain regions over time. External control, in this case the input, $u(t)$ is given in example by block pulses as applied during the corresponding experiments. The number of considered brain regions is denominated by l, while the number of exogenous inputs is referred to by m.

2.2.1.1 The fMRI Dynamic Submodel[1]

The dynamic submodel, or neuronal state equation, for fMRI represents the change in neuronal activity over time. This model can handle two types of input. First, the direct input into a region, which changes the neuronal state itself. Second, the latent input that scales the coupling between brain regions. Under the assumption that the underlying neuronal network is a deterministic dynamic system, with l neuronal states $z_{i=1\ldots l}$, m inputs $u_{k=1\ldots m}$ and parameters Θ, the corresponding dynamic system is given by:

$$\dot{z} = F(z, u, \Theta) \quad z \in \mathbb{R}^l, u \in \mathbb{R}^m \tag{2.1}$$

The vector z consists of the l neuronal states. The vector u holds the m input sources and Θ represents the parameters of the model. F is an unknown nonlinear function describing the change of neuronal activity over time. The function F can be approximated by its Taylor series using the first order terms as well as the bilinear second order term.

$$F(z, u, \Theta) \approx F(0, 0, \Theta) + \frac{\delta F}{\delta z} z + \frac{\delta F}{\delta u} u + \frac{\delta^2 F}{\delta z \delta u} zu \tag{2.2}$$

This approximation can be reparametrized to a directed graph with l nodes, as presented in [17] and [49]. Each node represents the neuronal state of a brain region.

$$\begin{aligned} \dot{z} &\approx \frac{\delta F}{\delta z} z + \frac{\delta F}{\delta u} u + \sum_k u_k \frac{\delta^2 F}{\delta z \delta u_k} z \\ &= Az + Cu + \sum_k u_k B^k z \\ A &\in \mathbb{R}^{l \times l} \\ C &\in \mathbb{R}^{l \times m} \\ B^k &\in \mathbb{R}^{l \times l}, k = 1 \ldots m \end{aligned} \tag{2.3}$$

The matrix A symbolizes the connectivity between brain regions, while matrices B^k describe for each input source u_k the input induced change in connectivity. Matrix C depicts the direct influence of input on the neuronal states. Now, a connection from the

[1] established in [17]

i-th region to the j-th region is given by the element a_{ji} of matrix A. A connection that is strengthened or weakened by input source k from region i to j is defined by b^k_{ji} of matrix B^k. An input of input source k into region i is declared by the component c_{ik} of matrix C.

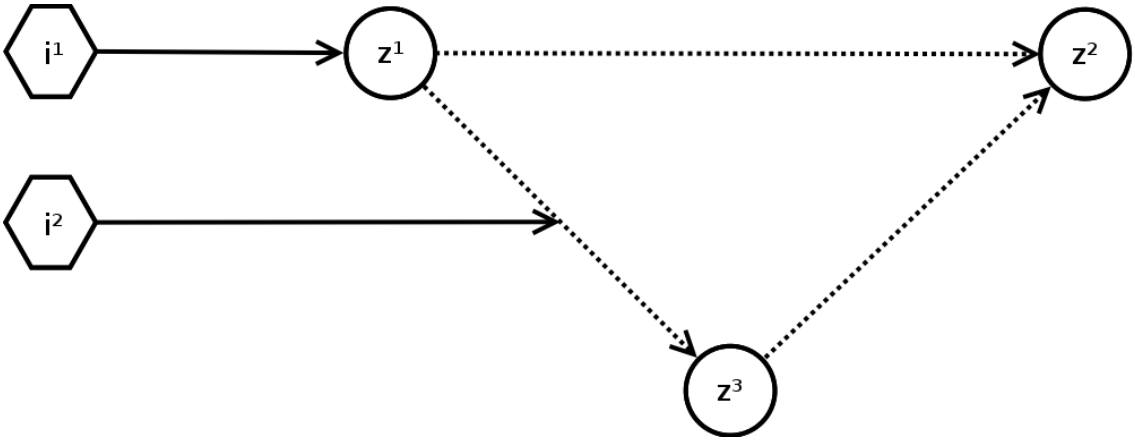

Figure 2.1: Sample fMRI dynamic submodel configuration

Stability

Without the controlling input u, this bilinear system can be considered uncontrolled and thus the indicator for stability, the maximal Lyapunov Exponent, reduces to the real part of the eigenvalues of A, as described in [23]. If the real part of the largest eigenvalue of A is negative then the system remains stable.

$$\max_i(\mathrm{Re}(\lambda_i(A))) < 0 \tag{2.4}$$

Under the assumption that all self-connections, described by the diagonal elements of the matrices A (and B), in the neuronal network are equal-valued (see [17]), the negativity of the eigenvalues can be ensured by normalizing A through multiplication with a scalar value that leaves the diagonal values of matrix A at $a_{ii} = -1$. If the off-diagonal elements of A are constraint this can easily be seen by the Gerschgorin Circle theorem.

$$\hat{A} = \sigma A = \begin{pmatrix} -1 & \hat{a}_{12} & \dots \\ \hat{a}_{21} & -1 & \\ \vdots & & \ddots \end{pmatrix} \tag{2.5}$$

$$\hat{B}^k = \sigma B^k = \begin{pmatrix} \hat{b}^k_{11} & \hat{b}^k_{12} & \dots \\ \hat{b}^k_{21} & \hat{b}^k_{22} & \\ \vdots & & \ddots \end{pmatrix}$$

The input induced connectivity matrices B^k are also scaled by σ as they are closely related to A by representing the same connections, only increased or decreased by input.

Prior Distribution

The priors on the coupling parameters are presumed to be distributed uniformly and independently. Consequently, the prior values can be expressed through their expectation and variance. The means of the parameters $a_{ij,i\neq j}$, b^k_{ij} and c_{ik} are all set to zero. The variances of $a_{ij,i\neq j}$ and b^k_{ij} are chosen, depending on the number of observed regions, to make negative values for the coupling parameters unlikely. As explained in the appendix of [17] the variance $v_{a,b}$ is determined by:

$$v_{a,b} = \frac{l(l-1)}{\Phi^{-1}_{\chi}(p_{a,b})} \tag{2.6}$$

with Φ^{-1}_{χ} being the inverse cumulative $\chi^2_{l(l-1)}$ distribution, and $p_{a,b}$ being the probability that the sum of squared off-diagonal components ($\sum_{i\neq j} a^2_{ij}$) of A remains less than $\frac{l}{l-1}$. The variance of input parameters c_{ik} is set to one. The scaling parameter σ has a mean of one and also a variance selected to make it impropable to become negative. Again following [17] the variance v_σ is computed by:

$$v_\sigma = \Big(\frac{1}{\Phi^{-1}_{N}(p_\sigma)}\Big)^2 \tag{2.7}$$

with Φ^{-1}_{N} being the inverse cumulative normal distribution and p_σ being the probability for σ to remain positive.

Parameters

The parameters of the fMRI dynamic submodel are the matrix components of the connectivity matrix A (except the diagonal entries), of the input induced connectivity matrices B^k, and the direct input matrix C, as well as the scalar σ normalizing A and B.

$$\Theta^{D,MRI} = \{\sigma, a_{i,j,i\neq j}, b^k_{i,j}, c_{i,k}\} \tag{2.8}$$

All these parameters can be concatenated to a parameter vector.

$$\theta^{D,MRI} = \big(\sigma, a_{12}, \ldots, a_{ll-1}, b^1_{11}, \ldots, b^l_{ll}, c_{11}, \ldots, c_{lm}\big) \in \mathbb{R}^{l(l-1)+mll+ml} \tag{2.9}$$

Parameter	Description	Prior Mean	Prior Variance
a_{ij}	Connectivity	0	see (2.6)
b_{ij}	Induced Connectivity	0	see (2.6)
c_{ik}	Input	0	1
σ	Scaling	1	see (2.7)

Figure 2.2: Table of parameters with corresponding priors for the fMRI dynamic submodel

Constant	Description	Value
a_{ii}	Self-Connectivity	-1

Figure 2.3: Table of constants for the fMRI dynamic submodel

2.2.1.2 The EEG Dynamic Submodel[2]

The dynamic submodel for EEG, named neural mass model, was originally based on the Jansen model as described in [28] and refined by Oliver David and Karl Friston in [8], [5] and [6]. This neuronal state equation of the EEG dynamic submodel are built upon the tripartitioning of a considered brain region (supragranular-layer, infragranular-layer and granular-layer-4). This tripartitioning models three neuronal subpopulations, labeled an excitatory subpopulation of interneurons, an inhibitory subpopulation of interneurons and an excitatory output subpopulation of pyramidal cells.

excitatory subpopulation granular-layer-4	← Forward Coupling / Extrinsic Input ← Lateral Coupling
↕	(Intrinsic Coupling)
excitatory pyramidal supopulation supragranular layer	← Backward Coupling ← Lateral Coupling
↕	(Intrinsic Coupling)
inhibitory subpopulation infragranular layer	← Backward Coupling ← Lateral Coupling

Figure 2.4: Tripartitioned cortex region with arriving input

The neural mass model is a combination of two operators. The first of these operators is a convolution of the arriving input $u(t)$ with an impulse response $h_{e,i}(t)$, the second is a sigmoid function $S(x)$.

[2] established in [5], [6] and extended in [47]

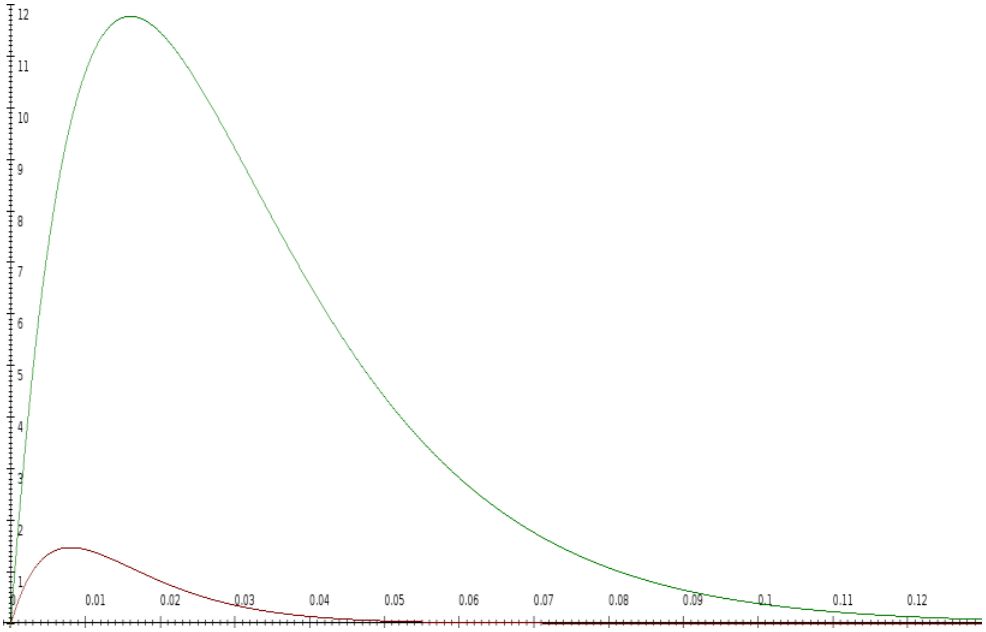

Figure 2.5: The impulse responses $h_i(t)$ and $h_e(t)$

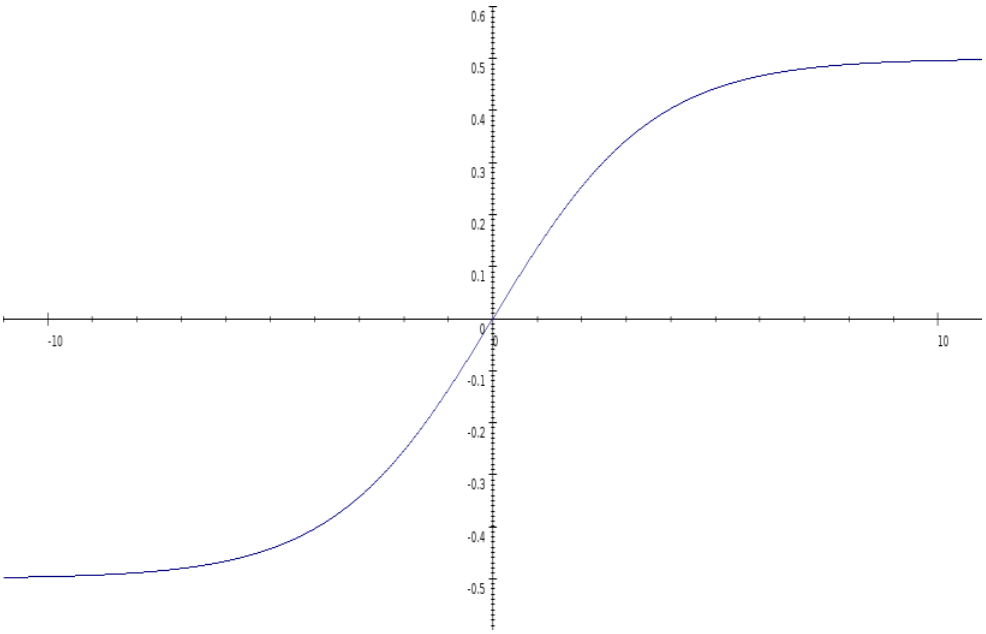

Figure 2.6: The sigmoid function $S(x)$

$$h_{e,i}(t) = \begin{cases} \frac{H_{e,i}}{\tau_{e,i}} t \exp\left(-\frac{t}{\tau_{e,i}}\right) & (t \geq 0) \\ 0 & (t < 0) \end{cases} \tag{2.10}$$

$$S(x) = \frac{2e_0}{1 + \exp\left(-rx\right)} - e_0 \tag{2.11}$$

The index of the impulse response marks it as excitatory ($h_e(t)$) or inhibitory ($h_i(t)$) with their corresponding synaptic parameters H_e,τ_e and H_i,τ_i respectively. The parameters $H_{e,i}$ represent the maximum post-synaptic potential, while the parameters $\tau_{e,i}$ are lumped time constants embodying various temporal delays as described in [5].

$$x = h_{e,i}(t) * u(t) \tag{2.12}$$

The convolution of the arriving input $u(t)$ with the impulse response $h_{e,i}(t)$ results in the average membrane potential, that equates to the potential that would be measured directly at the cortex region. This transformation leads to a second-order ordinary differential equation, which can be transformed into a system of two first order differential equations as follows:

$$\ddot{x}(t) = \frac{H_e}{\tau_e} u(t) - \frac{2}{\tau_e} \dot{x}(t) - \frac{1}{\tau_e^2} x(t) \tag{2.13}$$

$$\Rightarrow \begin{cases} \dot{x}_0(t) = x_1(t) \\ \dot{x}_1(t) = \frac{H_e}{\tau_e} u(t) - \frac{2}{\tau_e} x_0(t) - \frac{1}{\tau_e^2} x_1(t) \end{cases}$$

The sigmoid $S(x)$ converts the average membrane potential (back) to an average firing rate, that causes the neurons to emit an action potential. The two (constant) parameters e_0 and r control the shape of the sigmoid, at which the constant e_0 represents the maximum firing rate and r determines the slope of the sigmoid.
Applying this representation to the tripartitioning of a neuronal population results in the neural mass model in its state-space representation for a single region:

	$\dot{x}_1$	
	$\dot{x}_4$	
$\gamma_1 \uparrow$		$\downarrow \gamma_2$
	$\dot{x}_2$	
	$\dot{x}_5$	
	$\dot{x}_0$	
	$\dot{x}_3$	
	$\dot{x}_6$	
$\gamma_3 \uparrow$		$\downarrow \gamma_4$
	$\dot{x}_7$	
	$\dot{x}_8$	

excitatory subpopulation

excitatory output subpopulation

inhibitory subpopulation

Figure 2.7: Illustrated tripartitioning by equations

$$
\begin{aligned}
\dot{x_1} &= x_4 \\
\dot{x_4} &= \frac{H_e}{\tau_e}(u(t) + \gamma_1 S(x_0)) - \frac{2}{\tau_e}x_4 - \frac{1}{\tau_e^2}x_1 \\
\hline
\dot{x_2} &= x_5 \\
\dot{x_5} &= \frac{H_e}{\tau_e}\gamma_2 S(x_1) - \frac{2}{\tau_e}x_5 - \frac{1}{\tau_e^2}x_2 \\
\dot{x_0} &= x_5 - x_6 \\
\dot{x_3} &= x_6 \\
\dot{x_6} &= \frac{H_i}{\tau_i}\gamma_4 S(x_7) - \frac{2}{\tau_i}x_6 - \frac{1}{\tau_i^2}x_3 \\
\hline
\dot{x_7} &= x_8 \\
\dot{x_8} &= \frac{H_e}{\tau_e}\gamma_3 S(x_0) - \frac{2}{\tau_e}x_8 - \frac{1}{\tau_e^2}x_7
\end{aligned}
\tag{2.14}
$$

x_0 depicts the pyramidal cell depolarization of membrane potentials, which can be considered the output of the system reflecting a signal which is proportional to measured potentials (EEG signals).

The intrinsic coupling between subpopulations and the relation of the intrinsic coupling among each other has been determined experimentally (see [28]) and set to $\gamma_2 = \frac{4}{5}\gamma_1$ and $\gamma_3 = \gamma_4 = \frac{1}{4}\gamma_1$. The governing intrinsic connection parameter γ_1 can differ significantly under various constraints to the model.

In order to extend this single region model to multiple regions, the extrinsic input into a single region has to be split into input from other regions and experimental exogenous input. The coupling of two regions is assumed to follow a set of connection rules, as described in [5] which comprises three types of connections.

These three kinds of extrinsic coupling types of the neural mass model are defined as forward, backward and lateral connections. All connections originate in the agranular layers (infra- and supergranular) and pass through the granular layer-4. The forward (or bottom-up) connections end in the target regions granular-layer-4. The backward (or top-down) connections connnect to the target regions infra- and supergranular layers. Finally, the lateral connections terminate in all three subpopulations of the target region. Exogenous input is treated like a forward connection, arriving in granular-layer-4.

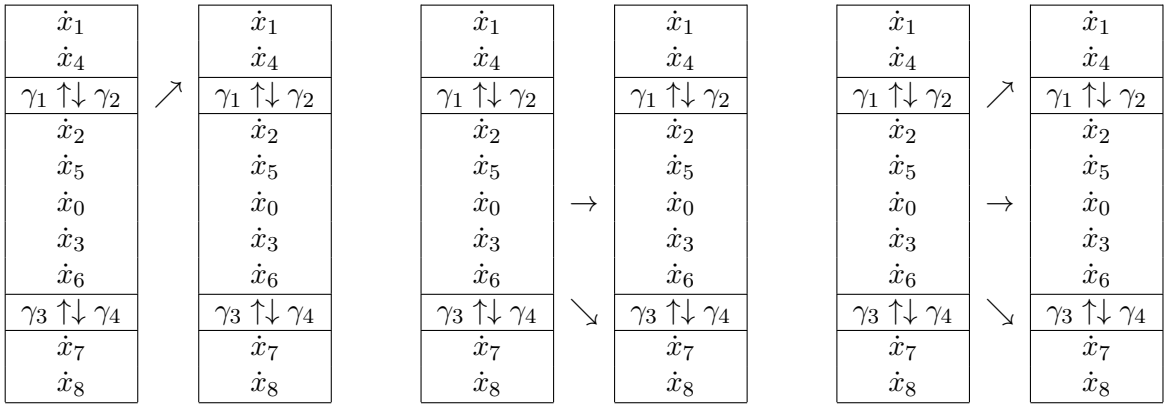

Figure 2.8: Coupling rules for the Neural Mass Model. From left to right: forward, backward and lateral connections.

$$
\begin{aligned}
\dot{x_1} &= x_4 \\
\dot{x_4} &= \frac{H_e}{\tau_e}((C^F + C^L + \gamma_1 I)S(x_0) + C^U u) - \frac{2}{\tau_e}x_4 - \frac{1}{\tau_e^2}x_1 \\
\hline
\dot{x_2} &= x_5 \\
\dot{x_5} &= \frac{H_e}{\tau_e}((C^B + C^L)S(x_0) + \gamma_2 S(x_1)) - \frac{2}{\tau_e}x_5 - \frac{1}{\tau_e^2}x_2 \\
\dot{x_0} &= x_5 - x_6 \\
\dot{x_3} &= x_6 \\
\dot{x_6} &= \frac{H_i}{\tau_i}\gamma_4 S(x_7) - \frac{2}{\tau_i}x_6 - \frac{1}{\tau_i^2}x_3 \\
\hline
\dot{x_7} &= x_8 \\
\dot{x_8} &= \frac{H_e}{\tau_e}((C^B + C^L + \gamma_3 I)S(x_0)) - \frac{2}{\tau_e}x_8 - \frac{1}{\tau_e^2}x_7
\end{aligned}
\tag{2.15}
$$

Note that now each x_i is a vector with l components, while the $l \times l$-matrices C^F, C^B and C^L represent the forward, backward and lateral connections, similar to the fMRI dynamic submodel. As the self-connectivity of each population will be controlled by the intrinsic coupling parameters $\gamma_{1,2,3,4}$, the diagonals of the matrices C^F, C^B, C^L remain zero.

Next, the gain and the transmission delays are included. The gain is represented by a set of matrices G_k, one for each of the k events during the experiment. The diagonal elements g_{kii} scale the excitatory maximum postsynaptic potentials $H_e^{(i)}$ as suggested in [47]. The off-diagonal elements $g_{kij,i\neq j}$ scale the contributions of the coupling matrices by multiplication through the Hadamard-product, in example $G_k \circ C^F, G_k \circ C^B, G_k \circ C^L$. The gain matrix associated with the first event has constant elements of one; by this the

subsequent events are scaled relative to the first. The gain matrices are similar to the matrices B^k of the fMRI dynamic submodel; they represent experiment related changes which can scale the intrinsic and extrinsic connectivity. Also the intrinsic and extrinsic transmission delays are included. These transmission delays, Δ_{ii} for intrinsic delays, Δ_{ij} for extrinsic delays, are applied to the sigmoid arguments, by which the neuronal state equation becomes a system of delay differential equations. Finally, the excitatory synaptic parameters are individualized for each brain region, embodied by matrices T, W and Y.

$$
\begin{aligned}
\dot{x_1} &= x_4(t) \\
\dot{x_4} &= T((C^F + C^L) \circ G_k S(x_0(t - \Delta_{ij})) + \gamma_1 S(x_0(t - \Delta_{ii})) + C^U u(t_p)) - W x_4 - Y x_1 \\
\hline
\dot{x_2} &= x_5(t) \\
\dot{x_5} &= T((C^B + C^L) \circ G_k S(x_0(t - \Delta_{ij})) + \gamma_2 S(x_1(t - \Delta_{ii}))) - W x_5 - Y x_2 \\
\dot{x_0} &= x_5(t) - x_6(t) \\
\dot{x_3} &= x_6(t) \\
\dot{x_6} &= \frac{H_i}{\tau_i}(\gamma_4 S(x_7(t - \Delta_{ii}))) - \frac{2}{\tau_i} x_6 - \frac{1}{\tau_i^2} x_3 \\
\hline
\dot{x_7} &= x_8 \\
\dot{x_8} &= T((C^B + C^L) \circ G_k S(x_0(t - \Delta_{ij})) + \gamma_3 S(x_0(t - \Delta_{ii}))) - W x_8 - Y x_7
\end{aligned}
\tag{2.16}
$$

$$
\begin{aligned}
T &= \operatorname{diag}(\frac{H_e^{(0)}}{\tau_e^{(0)}} g_{k00}, \ldots, \frac{H_e^{(l)}}{\tau_e^{(l)}} g_{kll}) \\
W &= \operatorname{diag}(\frac{2}{\tau_e^{(0)}}, \ldots, \frac{2}{\tau_e^{(l)}}) \\
Y &= \operatorname{diag}(\left(\frac{1}{\tau_e^{(0)}}\right)^2, \ldots, \left(\frac{1}{\tau_e^{(l)}}\right)^2)
\end{aligned}
$$

Input

Experimental input into the system is given by $u(t_p)$ where t_p denotes the peristimulus time. This input, which is the same for each region, is then scaled by the vector C^U. Since exogenous input acts like a forward connection, it influences the same partition as the forward connections, and is given by:

$$
u(t_p) = (\eta_2^{\eta_1} / \Gamma(\eta_1)) t^{\eta_1 - 1} \exp(-\eta_2 t_p) + \sum_r \iota_r \cos(2\pi(r-1)t_p) \tag{2.17}
$$

The first part is a gamma distribution that generates an input delay of $\frac{\eta_1}{\eta_2}$ seconds, the second part a discrete cosine set (see 2.2.3.2), emulating input fluctuations as described in [6]. This is of order eight, but omitting the 0-th order term, in which the contributions

of each term is scaled by the parameter ι_r. This is a major difference to the dynamic submodel for fMRI, where the input as defined by the experiment enters the neuronal state equation, while here the input during the experiment rather defines timespans for the event-related bursts generated by $u(t_p)$.

Stability

Important to the stability of the system is the sigmoid $S(x)$. It ascertains a stable fixed-point during the abscence of input. To prevent oscillatory behaviour of the system, the configuration of the connections has to be chosen carefully. Therefore, during the parameter distribution estimation, this is ensured through the priors which encourage forward over backward and backward over lateral connections.

Parameters

The parameters of the EEG dynamic submodel are the matrix components of the forward, backward and lateral connection matrices (except their diagonals), the input vector C^U, the gain matrices, the excitatory synaptic potentials, lumped rates and extrinsic transmission delays.

$$\Theta^{D,EEG} = \{c^F_{i,j,i\neq j}, c^B_{i,j,i\neq j}, c^L_{i,j,i\neq j}, g_{k,i,j}, c^U_i, H_e, \tau_e, \Delta_{ij}\} \tag{2.18}$$

All these parameters are also concatenated to a parameter vector.

$$\begin{aligned}\theta^{D,EEG} = &\begin{pmatrix} c^F_{i,j,i\neq j} & c^B_{i,j,i\neq j} & c^L_{i,j,i\neq j} & g_{k,i,j,k>1} & c^U_i & H_e^{(i)} & \tau_e^{(i)} & \Delta_{ij} & \eta_{1,2} & \iota_r \end{pmatrix} \\ &\in \mathbb{R}^{l(l-1)+(k-1)ll+3l+11}\end{aligned} \tag{2.19}$$

Prior Distribution

Following [6] the priors for the coupling matrix are chosen to encourage forward over backward and backward over lateral connections, to ensure the stability of the estimated system. Most of the parameters are assumed to have a positive domain; to enforce the parameters positivity, the natural logarithm of the parameters mean is estimated. An example of this is given below. For a parameter Ξ, assuming it is distributed gaussian, with a prior mean of ξ and prior variance σ, a zero centered mean is estimated, then exponentiated and scaled by its actual prior mean ξ.

$$\Xi \approx N(\xi, \sigma) \rightarrow \Xi \approx \xi \exp(N(0, \sigma)) \tag{2.20}$$

This is equivalent to a log-normal prior distribution.

Parameter	Description	Prior Mean	Prior Variance
c_{ij}^F	Forward Extrinsic Coupling*	32 Hz	0.5
c_{ij}^B	Backward Extrinsic Coupling*	16 Hz	0.5
c_{ij}^L	Lateral Extrinsic Coupling*	4 Hz	0.5
g_{ijk}	Coupling Gain*	1	0.5
c_i^U	Extrinsic Input*	1 Hz	0.5
H_e	Excitatory Postsynaptic Potential*	4 mV	0.0625
τ_e	Excitatory Lumped Rate*	0.008 s	0.0625
Δ_{ij}	Extrinsic Conduction Delay*	0.016 s	0.0625
η_1	Input Shape and Scale 1*	1 s	0.0625
η_2	Input Shape and Scale 2*	16 s	0.0625
ι_r	Input Fluctuation	0	1

Figure 2.9: Table of parameters with corresponding priors for the EEG dynamic submodel (taken from [6]). The parameters of which the log-normal distribution is estimated are marked with *.

Constant	Description	Value
e_0	Sigmoid Shift	$0.5\,\mathrm{s}^{-1}$
r	Sigmoid Shape	$0.56\,\mathrm{mV}^{-1}$
γ_1	Intrinsic Coupling 1	128.0 Hz
γ_2	Intrinsic Coupling 2	$\frac{4}{5}\gamma_1 = 102.4$ Hz
γ_3	Intrinsic Coupling 3	$\frac{1}{4}\gamma_1 = 32.0$ Hz
γ_4	Intrinsic Coupling 4	$\frac{1}{4}\gamma_1 = 32.0$ Hz
H_i	Inhibitory Postsynaptic Potential	32.0 mV
τ_i	Inhibitory Lumped Rate	0.016 s
Δ_{ii}	Intrinsic Conduction Delay	0.002 s

Figure 2.10: Table of constants for the EEG dynamic submodel (taken from [6]).

Note

The constants of this model, especially the inhibitory synaptic parameters H_i, τ_i and the intrinsic conduction delay Δ_{ii} can also be included in the set of parameters if the necessity for more parameters arises. In the case of H_i, τ_i and Δ_{ii} the prior means transfer from their constant value and the prior variances are the same as for H_e, τ_e and Δ_{ij} respectively.

2.2.2 The Forward Submodel

The forward submodel converts the neuronal state of the observed brain region, which is the output of the dynamic submodel, to a signal that can be compared to a measured response. In case of fMRI data the forward model is a system of differential equations,

in case of EEG data it is simply a linear transformation. In both model variants (fMRI and EEG) each of the neuronal states has an individual forward system attached.

2.2.2.1 The fMRI Forward Submodel

The fMRI, or hemodynamic, forward submodel is based upon the Balloon-Windkessel-Model as presented in [14], [13] and [17]. This submodel generates a Blood Oxygen Level Dependency (BOLD) response that can be compared to recorded fMRI data. As it is describing change in saturation of oxygen within the blood it is also called hemodynamic model. This model consists of a system of four ordinary differential equations and an output function. The normalized inflow f, which change is given by the vasodilatory signal s, is the input to the system of venomous volume v and deoxyhemoglobin content q, that compose the output of the BOLD signal.

The vasodilatory signal s receives the input z, flattens it with the damped signal itself and the scaled and shifted normalized flow.

$$\dot{s} = z - \kappa s - \gamma(f - 1) \tag{2.21}$$

The signal causes the normalized inflow.

$$\dot{f} = s \tag{2.22}$$

The normalized venomous volume is the difference of (normalized) inflow and outflow.

$$\tau\dot{v} = f - f_{out} \qquad (f_{out} = v^{\frac{1}{\alpha}}) \tag{2.23}$$

The normalized deoxyhemoglobin content intake is also modelled by a difference of inflow and outflow.

$$\tau\dot{q} = f\frac{E(f,\rho)}{\rho} - f_{out}\frac{q}{v} \qquad (E(f,\rho) = 1 - (1-\rho)^{\frac{1}{f}}) \tag{2.24}$$

The output function is a weighted sum of v and q that generate the BOLD signal.

$$y = V_0(\beta_1\rho(1-q) + \beta_2(1 - \frac{q}{v}) + (\beta_3\rho - \beta_4)(1-v)) \tag{2.25}$$

The constant V_0 denotes the Resting Blood Volume Fraction and $\beta_{1\ldots4}$ symbolize the weights for the summands. The values for these constants are given in the table below an were taken from [17]. The following schematic illustrates the hemodynamic system.

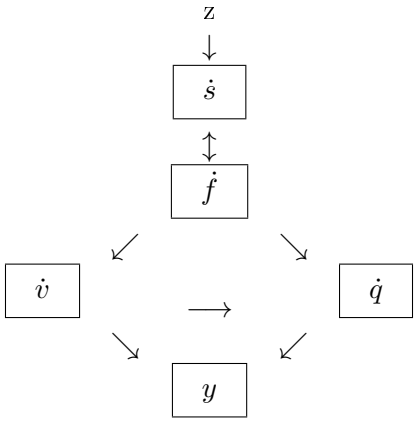

Figure 2.11: The hemodynamic fMRI forward submodel

Together these equations compose the hemodynamic forward system for the i-th region

$$\begin{aligned}
\dot{s}_i &= z_i - \kappa s_i - \gamma(f_i - 1) \\
\dot{f}_i &= s_i \\
\tau \dot{v}_i &= f_i - f_{out} \qquad (f_{out} = v_i^{\frac{1}{\alpha}}) \\
\tau \dot{q}_i &= f_i \frac{E(f_i, \rho)}{\rho} - f_{out} \frac{q}{v} \qquad (E(f_i, \rho) = 1 - (1-\rho)^{\frac{1}{f_i}}) \\
y_i &= V_0(\beta_1 \rho(1 - q_i) + \beta_2(1 - \frac{q_i}{v_i}) + (\beta_3 \rho - \beta_4)(1 - v_i))
\end{aligned} \tag{2.26}$$

For a more expedient computation this can be slightly simplified:

$$\begin{aligned}
\dot{s}_i &= z_i - \kappa s_i - \gamma(f_i - 1) \\
\dot{f}_i &= s_i \\
\dot{v}_i &= \frac{1}{\tau}(f_i - v_i^{\frac{1}{\alpha}}) \\
\dot{q}_i &= \frac{1}{\tau}\frac{f_i}{\rho}(1 - (1-\rho)^{\frac{1}{f_i}}) - v_i^{\frac{1}{\alpha}-1} q_i \\
y_i &= V_0(\beta_1 \rho(1 - q_i) + \beta_2(1 - \frac{q_i}{v_i}) + (\beta_3 \rho - \beta_4)(1 - v_i))
\end{aligned} \tag{2.27}$$

Parameters

The hemodynamic forward model of a single region i has five parameters:

$$\theta_i^{F,MRI} = \{\kappa_i, \gamma_i, \tau_i, \alpha_i, \rho_i\} \tag{2.28}$$

Now combining these hemodynamic parameters of each region into one parameter vector results in:

$$\theta^{F,MRI} = \left(\kappa_i, \gamma_i, \tau_i, \alpha_i, \rho_i\right) \in \mathbb{R}^{5l} \tag{2.29}$$

Prior Distribution

The priors and constants for the hemodynamic forward model are given in the table below.

Parameter	Description	Prior Mean	Prior Variance
κ	Rate of Signal Decay	$0.65\,\mathrm{s}^{-1}$	0.015
γ	Rate of Flow Dependent Elimination	$0.41\,\mathrm{s}^{-1}$	0.002
τ	Hemodynamic Transit Time	0.98 s	0.0568
α	Grubbs Exponent	0.32	0.0015
ρ	Resting Oxygen Extraction Fraction	0.34	0.0024

Figure 2.12: Table of parameters with corresponding priors for the fMRI forward submodel (taken from [17]).

Constant	Description	Value
V_0	Blood Volume	0.02
β_1	Volume Weight 1	7.0
β_2	Volume Weight 2	2.0
β_3	Volume Weight 3	2.0
β_4	Volume Weight 4	0.2

Figure 2.13: Table of constants for the fMRI forward submodel (taken from [17]).

2.2.2.2 The EEG Forward Submodel

The EEG forward submodel is a simple linear transformation of the dynamic submodels' output as opposed to the hemodynamic system that consists of a system of ordinary differential equations. Only an invasive method will be considered. Here the electrodes are implanted on top of the cortex regions and measure the potentials directly from the cortex surface. In this case the forward submodel reduces to a mere scaling of the depolarization output $x_0^{(i)}$. This transformation is given by:

$$y = k_i x_0^{(i)} \tag{2.30}$$

for a region i.

Parameters

The forward parameters are just the set of scaling parameters, one for each of the l states.

$$\Theta_i^{F,EEG} = \{k_i\} \tag{2.31}$$

For all regions, these parameters are also conjoined into a parameter vector.

$$\theta^{F,EEG} = \begin{pmatrix} k_0 & \dots & k_l \end{pmatrix} \in \mathbb{R}^l \tag{2.32}$$

Prior Distribution

The prior expectation and variance for the contribution parameter is given in the table below.

Parameter	Description	Prior Mean	Prior Variance
k_i	contribution	1.0	1.0

Figure 2.14: Table of parameters with corresponding priors for the EEG forward submodel

2.2.3 The Combined Model

Combining the dynamic and forward submodels leads to the full inverse model. This combined model consists of the dynamic submodel mediating the coupling among the brain regions while the l forward submodels will convert each neuronal states' output to a signal that can be compared to a measured signal.

2.2.3.1 The Data Model

The combined model, dependent on the combined parameters $\tilde{\theta} = \theta^{D+F}$ and the input u, can then be described by a function $y = h(\tilde{\theta}, u)$ generating an output signal y. The error in data acquisition is modelled by an additive Gaussian error (vector) ϵ with zero mean. Then the data model is given by:

$$y = h(\tilde{\theta}, u) + \epsilon \tag{2.33}$$

The parameters are assumed to have (almost) Gaussian distribution, meaning the parameters will be treated as independent random variables. For the estimation some prior knowledge about these parameters will be useful and is specified through prior mean $\eta_{\tilde{\theta}}$ and prior covariance $C_{\tilde{\theta}}$ and is listed in the respective tables of the submodels.

2.2.3.2 Drift

As an extension to the data model, a drift component is added (see [17]), simulating low frequency drifts as well as a constant term. The drift is given by a discrete cosine set (similar to the one used for the input fluctuations in the dynamic submodel for EEG in 2.17).

Discrete Cosine Set

A r-th order discrete cosine set as described in [12] essentially stretches the first $\frac{r}{2}$ periods of a cosine to a given length $T = tN$ and generates N discrete cosine values at positions t. A discrete cosine set of order r for N time steps and a time step width of t is given by:

$$f_r(t) = \sqrt{\frac{2}{N}} cos(r\pi \frac{t}{N}) \quad t = 1 \dots N \tag{2.34}$$

whereat the 0-th order term is only a constant.

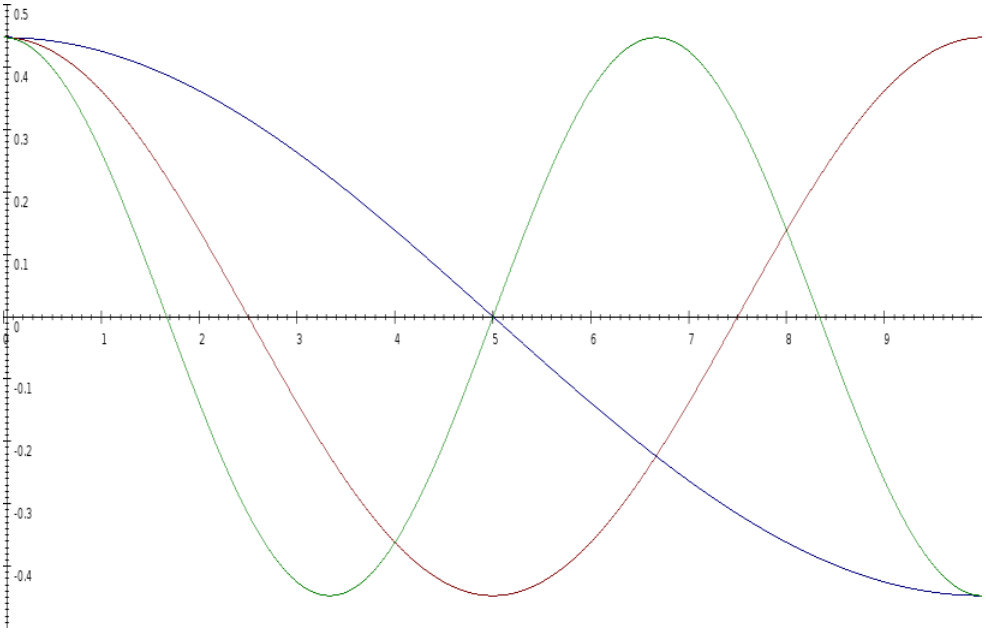

Figure 2.15: Discrete Cosine Sets of Order 1 To 3 ($N = 10$)

Drift Design Matrix

Taking one discrete cosine set for each neuronal state up to a chosen order r, which

is stretched over the whole time series, with discrete components at each timestep t leads to the following matrix X with the corresponding scaling parameters β:

$$X = \begin{pmatrix} f_0^1(0) & \dots & f_r^1(0) & & & 0 & \\ & 0 & & \ddots & & \vdots & \\ & \vdots & & & f_0^l(0) & \dots & f_r^l(0) \\ & & & \vdots & & & \\ & & & \vdots & & & \\ f_0^1(N) & \dots & f_r^1(N) & & & 0 & \\ & 0 & & \ddots & & \vdots & \\ & \vdots & & & f_0^l(N) & \dots & f_r^l(N) \end{pmatrix}, \beta = \begin{pmatrix} \beta_0^1 \\ \vdots \\ \beta_r^1 \\ \vdots \\ \beta_0^l \\ \vdots \\ \beta_r^l \end{pmatrix} \tag{2.35}$$

This drift matrix will be included in the parameter estimation algorithm (2.3.1.2), while the parameters β will be included in the set of estimated parameters. Since each brain region gets its own set of drift parameters this leads to an additional rl parameters which are conjoined to the drift parameter vector θ^β.

2.2.3.3 Extended Data Model

The drift will be included in the data model as follows:

$$y = h(\theta, u) + X\beta + \epsilon \tag{2.36}$$

This extended data model will be the basis for the estimation process.

2.2.3.4 Parameter Overview

The sets of parameters from the dynamic submodel, the forward submodel, and the drift are combined to an overall parameter vector.

$$\tilde{\theta}^{D+F+\beta} = \begin{pmatrix} \theta^D \\ \theta^F \\ \theta^\beta \end{pmatrix} \tag{2.37}$$

All these parameters are to be estimated to fit the data. The following section will explain how this can be achieved.

2.3 Parameter Estimation and EM-Algorithm

Since a statistical approach is used to make inferences about the parameters of the model, not the parameters, but their distribution is estimated. This parameter distribution is

approximated to fit the experimental data. The resulting distribution specified in terms of its first two moments, is called posterior or conditional probability distribution. The aim of the estimation procedure is to determine conditional means and (co-)variances for all parameters of the model.

2.3.1 Mean and Covariance Estimation

Under the assumption that the parameters are distributed uniformly and independently (gaussian), they can be described through mean and covariance. Following [16], the estimation of posterior mean and posterior covariance will be presented.

2.3.1.1 The Bayesian Approach

Starting with the Bayes theorem:

$$p(\theta \mid y) = \frac{p(y \mid \theta)p(\theta)}{p(y)} \tag{2.38}$$

As the evidence $p(y)$ is a normalizing factor the following holds:

$$p(\theta|y) \propto p(y|\theta)p(\theta) \tag{2.39}$$

This means that the posterior distribution is proportional to the likelihood times the prior distribution. The above formula will be the basis for the algorithm estimating the conditional probability or posteriors of the parameters θ through the likelihood under the given data y and the prior distribution.

2.3.1.2 Mean Estimation

An estimation of the full or combined model is given by:

$$h(\theta, u) = h(\eta^{(k)}_{\theta|y}) + J(\theta - \eta^{(k)}_{\theta|y}) \tag{2.40}$$
$$J = \frac{\delta h(\eta^{(k)}_{\theta|y})}{\delta \theta}$$

This local linear approximation estimates the full model through the derivation by the parameters furthermore refered to as parameter-jacobian. Applying Bayes theorem, under Gaussian assumption for the parameters, and utilizing a multivariate normal distribution[3],

$$p_\theta(x) = \exp(\frac{1}{2}(x-\eta)^T C^{-1}(x-\eta)) \tag{2.41}$$

[3] omitting the normalizing term $((2\pi)^{\frac{\dim(\theta)}{2}}|C|^{\frac{1}{2}})^{-1}$

the above, as described in [16], leads to:

$$p(y|\theta) \propto \exp(-\frac{1}{2}((y - h(\eta_{\theta|y}^{(k)})) - J(\theta - \eta_{\theta|y}^{(k)}))^T C_\epsilon^{-1}((y - h(\eta_{\theta|y}^{(k)})) - J(\theta - \eta_{\theta|y}^{(k)})) \tag{2.42}$$

$$p(\theta) \propto \exp(-\frac{1}{2}(\theta - \eta_\theta)^T C_\theta^{-1}(\theta - \eta_\theta))$$

$$\Rightarrow p(\theta|y) \propto \exp(-\frac{1}{2}(\theta - \eta_{\theta|y}^{(k+1)})^T C_{\theta|y}^{-1}(\theta - \eta_{\theta|y}^{(k+1)}))$$

With the prior expectation η_θ, the prior covariance C_θ, the error covariance C_ϵ, the conditional mean estimates

$$\eta_{\theta|y}^{(k+1)} = \eta_{\theta|y}^{(k)} + C_{\theta|y}(J^T C_\epsilon^{-1}(y - h(\eta_{\theta|y}^{(k)})) + C_\theta^{-1}(\eta_\theta - \eta_{\theta|y}^{(k)})) \tag{2.43}$$

and the conditional covariances

$$C_{\theta|y} = (J^T C_\epsilon^{-1} J + C_\theta^{-1})^{-1} \tag{2.44}$$

This can be used to set up an iterative algorithm based upon the Gauss-Newton method (ordinary-least-squares-estimator). The additionally available prior information needs to be incorporated into the algorithm, and also the drift, which parameters are supposed to be estimated as well.

Starting with ordinary-least-squares-estimator:

$$\begin{aligned} \bar{y} &= y - h(\eta_{\theta|y}^{(k)}) \\ J &= \frac{\delta h(\eta_{\theta|y}^{(k)})}{\delta \theta} \\ \eta_{\theta|y}^{(k+1)} &= \eta_{\theta|y}^{(k)} + (JJ^T)^{-1} Jy \end{aligned} \tag{2.45}$$

Including the covariance into the above algorithm, it becomes the weighted-least-squares-estimator:

$$\begin{aligned} \bar{y} &= y - h(\eta_{\theta|y}^{(k)}) \\ J &= \frac{\delta h(\eta_{\theta|y}^{(k)})}{\delta \theta} \\ C_{\theta|y} &= (J^T C_\epsilon^{-1} J)^{-1} \\ \eta_{\theta|y}^{(k+1)} &= \eta_{\theta|y}^{(k)} + C_{\theta|y}(J^T C_\epsilon^{-1} \bar{y}) \end{aligned} \tag{2.46}$$

Following [22] (p.259f) the prior expectation and the prior variance can be included into the algorithm. This leads to the augmented[4] weighted-least-squares-estimator.

[4] as named in [16]

$$
\begin{aligned}
\bar{y} &= \begin{pmatrix} y - h(\eta_{\theta|y}^{(k)}) \\ \eta_\theta - \eta_{\theta|y}^{(k)} \end{pmatrix} \\
J &= \frac{\delta h(\eta_{\theta|y}^{(k)})}{\delta \theta} \\
\bar{J} &= \begin{pmatrix} J \\ 1 \end{pmatrix} \\
C_\epsilon &= \begin{pmatrix} C_\epsilon & 0 \\ 0 & C_\theta \end{pmatrix} \\
C_{\theta|y} &= (\bar{J}^T \bar{C}_\epsilon^{-1} \bar{J})^{-1} \\
\eta_{\theta|y}^{(k+1)} &= \eta_{\theta|y}^{(k)} + C_{\theta|y} (\bar{J}^T \bar{C}_\epsilon{}^{-1} \bar{y})
\end{aligned}
\tag{2.47}
$$

Finally the drift is added into the the design matrix:

$$
\begin{aligned}
\bar{y} &= \begin{pmatrix} y - h(\eta_{\theta|y}^{(k)}) \\ \eta_\theta - \eta_{\theta|y}^{(k)} \end{pmatrix} \\
J &= \frac{\delta h(\eta_{\theta|y}^{(k)})}{\delta \theta} \\
\bar{J} &= \begin{pmatrix} J & X \\ 1 & 0 \end{pmatrix} \\
C_\epsilon &= \begin{pmatrix} C_\epsilon & 0 \\ 0 & C_\theta \end{pmatrix} \\
C_{\theta|y} &= (\bar{J}^T \bar{C}_\epsilon^{-1} \bar{J})^{-1} \\
\eta_{\theta|y}^{(k+1)} &= \eta_{\theta|y}^{(k)} + C_{\theta|y} (\bar{J}^T \bar{C}_\epsilon{}^{-1} \bar{y})
\end{aligned}
\tag{2.48}
$$

This is essentially a Bayesian regression with a known covariance matrix as described in [22] (p.255) and an extended design matrix (see [17]).

2.3.1.3 Covariance Estimation

As the covariance matrix C_ϵ is usually unknown, it has to be estimated as well. First it is parametrized by hyperparameters λ_i, which scale their respective associated basis components Q_i. Then the covariance matrix C_ϵ is a sum of the matrices Q_i scaled by the covariance hyperparameters λ_i. Together the hyperparameters λ_i form a hyperparameter vector $\lambda \in \mathbb{R}^l$.

$$
C_\epsilon = \sum_i^l \lambda_i Q_i \tag{2.49}
$$

The matrices Q_i, representing a basis set for the hyperparameters λ_i, are a Kronecker product of the unit matrix V, of dimension $n \times n$ (n being the number of datapoints of the time series) with the matrices $(\delta_{ij})_{i=j=l}$. The latter are matrices that have only one non-zero element, which is positioned on the i-th diagonal element.

$$Q_i = V \otimes (\delta_{ij})_{i=j=l} \in \mathbb{R}^{ln \times ln} \tag{2.50}$$

Adding the priors to the covariance matrix as it was done in [22] with the residual vector and design matrix in the mean estimation, results in the extended covariance matrix:

$$\bar{C}_\epsilon = \begin{pmatrix} \sum_i^l \lambda_i Q_i & 0 \\ 0 & C_\theta \end{pmatrix} \tag{2.51}$$

The hyperparameters λ_i are esitmated with a Scoring algorithm. This is essentially a Newton's method applied to find maximum values.

$$\begin{aligned} P &= C_\epsilon^{-1} - C_\epsilon^{-1} \bar{J} C_{\theta|y} \bar{J}^T C_\epsilon^{-1} \\ g_i &= \frac{\delta F}{\delta \lambda_{(k)}} = \operatorname{tr}(-\frac{\delta F}{\delta C_\epsilon^{-1}} C_\epsilon^{-1} Q_i C_\epsilon^{-1}) = -\frac{1}{2}\operatorname{tr}(PQ_i) + \frac{1}{2}\bar{y}^T P^T Q_i P \bar{y} \\ H_{ij} &= \frac{\delta^2 F}{\delta \lambda_{ij}^2} = \frac{1}{2}\operatorname{tr}(PQ_iPQ_j) - \bar{y}^T PQ_iPQ_jP\bar{y} \\ \lambda^{(k+1)} &= \lambda^{(k)} + H^{-1} g \end{aligned} \tag{2.52}$$

The function F, of which the derivatives are taken, is the negative free energy ([16]) or variational free energy ([36]) of the system given by:

$$F(\tilde{p}(\theta), \lambda) = E_{\tilde{p}}[\ln p(\theta, y|\lambda)] - E_{\tilde{p}}[\ln \tilde{p}(\theta)]. \tag{2.53}$$

The use of Fisher information replaces the second derivative with its expectation. This variant of the Scoring algorithm reduces the necessary computations and is called Fisher-Scoring algorithm.

$$\begin{aligned} P &= C_\epsilon^{-1} - C_\epsilon^{-1} \bar{X} C_{\theta|y} \bar{X}^T C_\epsilon^{-1} \\ g_i &= \frac{\delta F}{\delta \lambda_i} = \operatorname{tr}(-\frac{\delta F}{\delta C_\epsilon^{-1}} C_\epsilon^{-1} Q_i C_\epsilon^{-1}) = -\frac{1}{2}\operatorname{tr}(PQ_i) + \frac{1}{2}\bar{y}^T P^T Q_i P \bar{y} \\ \langle H_{ij} \rangle &= \Big\langle -\frac{\delta^2 F}{\delta \lambda_{ij}^2} \Big\rangle = -\frac{1}{2}\operatorname{tr}(PQ_iPQ_j) \\ \lambda^{(k+1)} &= \lambda^{(k)} - \langle H \rangle^{-1} g \end{aligned} \tag{2.54}$$

This algorithm is derived in full detail in [16].

2.3.2 The EM-Algorithm

The aim of the parameter estimation process is to maximize the (log-) likelihood of the estimated posteriors. As the covariance is unknown, the posterior estimation is done with a two step procedure called EM-algorithm (Expectation Maximization). The E-Step (Expectation) estimates the posterior mean $\eta_{\theta|y}$ while holding the hyperparameter vector λ fixed. The M-Step (Maximization) then estimates the hyperparameters λ while keeping the posteriors $\eta_{\theta|y}$ fixed. The EM-algorithm can be summarized (see [36],[16]), with some conditional density $\tilde{p}(\theta)$, as follows:

$$\begin{aligned} \text{E-Step:} \quad & \tilde{p} \leftarrow \max_{\tilde{p}} F(\tilde{p}(\theta), \lambda) \\ \text{M-Step:} \quad & \lambda \leftarrow \max_{\lambda} F(\tilde{p}(\theta), \lambda) \end{aligned} \tag{2.55}$$

Combining the estimation of posterior mean $\eta_{y|\theta}$ and covariance hyperparameters λ leads to the EM-algorithm as it was provided by [16], [13] and [17] which in turn is based on [25], [9] and [36].

$$\begin{aligned} & while(K < c) \\ & \{ \\ & \text{E-Step:} \\ & \bar{y} = \begin{pmatrix} y - h(\eta_{\theta|y}^{(k)}) \\ \eta_\theta - \eta_{\theta|y}^{(k)} \end{pmatrix} \\ & J = \frac{\delta h(\eta_{\theta|y}^{(k)})}{\delta \theta} \\ & \bar{J} = \begin{pmatrix} J & X \\ 1 & 0 \end{pmatrix} \\ & \bar{C}_\epsilon = \begin{pmatrix} \sum_i^l \lambda_i^{(k)} Q_i & 0 \\ 0 & C_\theta \end{pmatrix} \\ & C_{\theta|y} = (\bar{J}^T \bar{C}_\epsilon^{-1} \bar{J})^{-1} \\ & \Delta\eta_{\theta|y} = C_{\theta|y} (\bar{J}^T {\bar{C}_\epsilon}^{-1} \bar{y}) \\ & \eta_{\theta|y}^{(k+1)} = \eta_{\theta|y}^{(k)} + \Delta\eta_{\theta|y} \end{aligned} \tag{2.56}$$

M-Step:

$$
\begin{aligned}
P &= \bar{C}_\epsilon^{-1} - \bar{C}_\epsilon^{-1}\bar{J}C_{\theta|y}\bar{J}^T\bar{C}_\epsilon^{-1} \\
g_i &= -\frac{1}{2}\operatorname{tr}(PQ_i) + \frac{1}{2}\bar{y}^T P^T Q_i P\bar{y} \\
H_{ij} &= -\frac{1}{2}\operatorname{tr}(PQ_iPQ_j) \\
\Delta\lambda &= \langle H\rangle^{-1} g \\
\lambda^{(k+1)} &= \lambda^{(k)} - \Delta\lambda \\
&\}
\end{aligned}
$$

The starting values for the parameters are the priors, for the hyperparameters they are selected small enough to make a positive definite ($\bar{J}^T\bar{C}_\epsilon^{-1}\bar{J}$) likely.

$$
\begin{aligned}
\eta_{\theta|y}^{(0)} &= \eta_\theta \\
\lambda_i^{(0)} &= 0.001
\end{aligned}
\tag{2.57}
$$

The convergence condition K was suggested in [13] to be the length of the update to the conditional parameter vector.

$$
K = \|\Delta\eta_{\theta|y}\| \tag{2.58}
$$

2.3.2.1 Improved EM-Algorithm

In case of large sets of recorded data the above algorithm holds two problems. First, the residual forming matrix P in the M-step will not only get very large, but also very dense if not even fully populated. Second, in the above form the algorithm calculates the same matrix products multiple times. All improvements that will be made next are centered around the following three identities. These basic linear algebra identities will be used to rearrange matrix products into a more efficient order.

Lemma 1. *(Matrix Product Identity) Given two matrices $A \in \mathbb{R}^{m\times n}$ and $B \in \mathbb{R}^{n\times o}$ then*

$$
(AB)^T = B^T A^T \tag{2.59}
$$

holds.

Proof. An Element of the Matrix Product is given by:

$$
(ab)_{ij}^T = \sum_k a_{jk}b_{ki} = \sum_k b_{ki}a_{jk} = \sum_k b_{ik}^T a_{kj}^T = (b^T a^T)_{ij} \tag{2.60}
$$

□

Lemma 2. *(Trace of a Product) Given two matrices $A \in \mathbb{R}^{m \times n}$ and $B \in \mathbb{R}^{n \times m}$ then*

$$\mathrm{tr}(AB) = \sum_i \sum_j a_{ij} b_{ji} \tag{2.61}$$

holds.

Proof. An Element of the Matrix Product is given by:

$$(ab)_{ij} = \sum_k a_{ik} b_{kj} \tag{2.62}$$

For the trace only the diagonal elements are required:

$$(ab)_{ii} = \sum_k a_{ik} b_{ki} \tag{2.63}$$

Summing up all diagonal elements completes the proof:

$$\sum_i (ab)_{ii} = \sum_i \sum_k a_{ik} b_{ki} \tag{2.64}$$

□

Lemma 3. *(Cyclic Permutations of Trace Arguments) Given three matrices $A \in \mathbb{R}^{m \times n}, B \in \mathbb{R}^{n \times o}, C \in \mathbb{R}^{o \times m}$ then*

$$\mathrm{tr}(ABC) = \mathrm{tr}(BCA) = \mathrm{tr}(CAB) \tag{2.65}$$

holds.

Proof. First rewriting the argument:

$$D := BC \Rightarrow \mathrm{tr}(ABC) = \mathrm{tr}(AD) \tag{2.66}$$

Applying Lemma 2 leads to:

$$\mathrm{tr}(AD) = \sum_i \sum_j a_{ij} d_{ji} = \sum_j \sum_i a_{ij} d_{ji} = \sum_j \sum_i d_{ji} a_{ij} = \mathrm{tr}(DA) \tag{2.67}$$

□

Next, a step-by-step transformation of the original EM-algorithm, split by E-step and M-step, is presented. Starting with the original E-step:

$$
\begin{aligned}
\bar{y} &= \begin{pmatrix} y - h(\eta_{\theta|y}^{(k)}) \\ \eta_\theta - \eta_{\theta|y}^{(k)} \end{pmatrix} \\
J &= \frac{\delta h(\eta_{\theta|y}^{(k)})}{\delta \theta} \\
\bar{J} &= \begin{pmatrix} J & X \\ 1 & 0 \end{pmatrix} \\
\bar{C}_\epsilon &= \begin{pmatrix} \sum_i^l \lambda_i^{(k)} Q_i & 0 \\ 0 & C_\theta \end{pmatrix} \\
C_{\theta|y} &= (\bar{J}^T C_\epsilon^{-1} \bar{J}^T)^{-1} \\
\Delta\eta_{\theta|y} &= C_\theta \bar{J} C_\epsilon^{-1} \bar{y} \\
\eta_{\theta|y}^{(k+1)} &= \eta_{\theta|y}^{(k)} + \Delta\eta_{\theta|y}
\end{aligned}
\tag{2.68}
$$

Now, the number of transpostions is reduced and intermediate results are saved for later use in the (improved) M-Step. Then the improved E-Step is given by:

$$
\begin{aligned}
\bar{y} &= \begin{pmatrix} y - h(\eta_{\theta|y}^{(k)}) \\ \eta_\theta - \eta_{\theta|y}^{(k)} \end{pmatrix} \\
J &= \frac{\delta h(\eta_{\theta|y}^{(k)})}{\delta \theta} \\
\bar{J} &= \begin{pmatrix} J & X \\ 1 & 0 \end{pmatrix} \\
\bar{C}_\epsilon &= \begin{pmatrix} \sum_i^l \lambda_i^{(k)} Q_i & 0 \\ 0 & C_\theta \end{pmatrix} \\
\bar{J}^A &= C_\epsilon^{-1} \bar{J} \\
\bar{J}^B &= \bar{J^A}^T \\
C_{\theta|y} &= (\bar{J}^B \bar{J})^{-1} \\
\bar{D} &= C_{\theta|y} \bar{J}^B \\
\Delta\eta_{\theta|y} &= \bar{D} \bar{y} \\
\eta_{\theta|y}^{(k+1)} &= \eta_{\theta|y}^{(k)} + \Delta\eta_{\theta|y}
\end{aligned}
\tag{2.69}
$$

Some of these changes might seem inefficient as they actually increase the number of operations, but combined with the following improved M-step they reduce a lot of computations. Even more important than the performance gain of the algorithm is the

prevention of the computation of the product $(C_\epsilon^{-1}\bar{J})^T(C_{\theta|y}\bar{J}^T C_\epsilon^{-1})$, that is necessary for the residual forming matrix P in the M-Step, as this will likely produce a very large and dense matrix.

Again starting with the original M-step:

$$\begin{aligned} P &= C_\epsilon^{-1} - C_\epsilon^{-1}\bar{J}C_{\theta|y}\bar{J}^T C_\epsilon^{-1} \\ g_i &= -\frac{1}{2}\operatorname{tr}(PQ_i) + \frac{1}{2}\bar{y}^T P^T Q_i P\bar{y} \\ H_{ij} &= -\frac{1}{2}\operatorname{tr}(PQ_iPQ_j) \\ \Delta\lambda &= \langle H\rangle^{-1} g \\ \lambda^{(k+1)} &= \lambda^{(k)} - \Delta\lambda \end{aligned} \tag{2.70}$$

Removing the unnecessary multiplications with constants:

$$\begin{aligned} P &= C_\epsilon^{-1} - C_\epsilon^{-1}\bar{J}C_{\theta|y}\bar{J}^T C_\epsilon^{-1} \\ g_i &= \bar{y}^T P^T Q_i P\bar{y} - \operatorname{tr}(PQ_i) \\ H_{ij} &= \operatorname{tr}(PQ_iPQ_j) \\ \Delta\lambda &= \langle H\rangle^{-1} g \\ \lambda^{(k+1)} &= \lambda^{(k)} + \Delta\lambda \end{aligned} \tag{2.71}$$

Cycling the trace arguments by applying Lemma 2 to have the more sparse matrix first:

$$\begin{aligned} P &= C_\epsilon^{-1} - C_\epsilon^{-1}\bar{J}C_{\theta|y}\bar{J}^T C_\epsilon^{-1} \\ g_i &= (P\bar{y})^T Q_i (P\bar{y}) - \operatorname{tr}(Q_i P) \\ H_{ij} &= \operatorname{tr}(Q_iPQ_jP) \\ \Delta\lambda &= \langle H\rangle^{-1} g \\ \lambda^{(k+1)} &= \lambda^{(k)} + \Delta\lambda \end{aligned} \tag{2.72}$$

Sorting by precomputed vectors and matrices from the E-step:

$$\begin{aligned} P &= C_\epsilon^{-1} - (\bar{J}^T C_\epsilon^{-1})^T(C_{\theta|y}\bar{J}^T C_\epsilon^{-1}) \\ g_i &= (P\bar{y})^T Q_i (P\bar{y}) - \operatorname{tr}(Q_i P) \\ H_{ij} &= \operatorname{tr}(Q_iPQ_jP) \\ \Delta\lambda &= \langle H\rangle^{-1} g \\ \lambda^{(k+1)} &= \lambda^{(k)} + \Delta\lambda \end{aligned} \tag{2.73}$$

Rearranging the matrix traces, utilizing the additivity of the matrix trace as well as the distributivity of the matrix multiplication leads to:

$$
\begin{aligned}
p_y &= C_\epsilon^{-1}\bar{y} - (\bar{J}^T C_\epsilon^{-1})^T (C_{\theta|y}\bar{J}^T C_\epsilon^{-1})\bar{y} \\
g_i &= p_y^T Q_i p_y - \mathrm{tr}(Q_i C_\epsilon^{-1}) + \mathrm{tr}((C_{\theta|y}\bar{J}^T C_\epsilon^{-1}) Q_i (\bar{J}^T C_\epsilon^{-1})^T) \\
H_{ij} &= \mathrm{tr}(Q_i C_\epsilon^{-1} Q_j C_\epsilon^{-1}) \\
&\quad - \mathrm{tr}(Q_i C_\epsilon^{-1} Q_j (\bar{J}^T C_\epsilon^{-1})^T (C_{\theta|y}\bar{J}^T C_\epsilon^{-1})) \\
&\quad - \mathrm{tr}(Q_j C_\epsilon^{-1} Q_i (\bar{J}^T C_\epsilon^{-1})^T (C_{\theta|y}\bar{J}^T C_\epsilon^{-1})) \\
&\quad + \mathrm{tr}((C_{\theta|y}\bar{J}^T C_\epsilon^{-1}) Q_i (\bar{J}^T C_\epsilon^{-1})^T)(C_{\theta|y}\bar{J}^T C_\epsilon^{-1}) Q_j (\bar{J}^T C_\epsilon^{-1})^T) \\
\Delta\lambda &= \langle H \rangle^{-1} g \\
\lambda^{(k+1)} &= \lambda^{(k)} + \Delta\lambda
\end{aligned}
\tag{2.74}
$$

Finally preventing double calculations by precomputations:

$$
\begin{aligned}
Q_i^A &= Q_i C_\epsilon^{-1} \\
Q_i^B &= Q_i (\bar{J}^T C_\epsilon^{-1})^T \\
Q_i^C &= (C_{\theta|y}\bar{J}^T C_\epsilon^{-1}) Q_i^B
\end{aligned}
\tag{2.75}
$$

$$
\begin{aligned}
p_y &= C_\epsilon^{-1}\bar{y} - (\bar{J}^T C_\epsilon^{-1})^T (C_{\theta|y}\bar{J}^T C_\epsilon^{-1})\bar{y} \\
g_i &= p_y^T Q_i p_y - \mathrm{tr}(Q_i^A) + \mathrm{tr}(Q_i^C) \\
H_{ij} &= \mathrm{tr}(Q_i^A Q_j^A) \\
&\quad - \mathrm{tr}(Q_j^A Q_i^B (C_{\theta|y}\bar{J}^T C_\epsilon^{-1})) \\
&\quad - \mathrm{tr}(Q_i^A Q_j^B (C_{\theta|y}\bar{J}^T C_\epsilon^{-1})) \\
&\quad + \mathrm{tr}(Q_i^C Q_j^C) \\
\Delta\lambda &= \langle H \rangle^{-1} g \\
\lambda^{(k+1)} &= \lambda^{(k)} + \Delta\lambda
\end{aligned}
$$

Reusing the matrices that were computed in the improved E-step, results in the improved M-step:

$$
\begin{aligned}
Q_i^A &= Q_i C_\epsilon^{-1} \\
Q_i^B &= Q_i \bar{J}^A \\
Q_i^C &= \bar{D} Q_i^B
\end{aligned}
\tag{2.76}
$$

$$
\begin{aligned}
p_y &= C_\epsilon^{-1} \bar{y} - \bar{J}^A \Delta\eta_{\theta|y} \\
g_i &= p_y^T Q_i p_y - tr(Q_i^A) + \mathrm{tr}(Q_i^C) \\
H_{ij} &= \mathrm{tr}(Q_i^A Q_j^A) - \mathrm{tr}((Q_j^A Q_i^B)\bar{D}) - \mathrm{tr}((Q_i^A Q_j^B)\bar{D}) + \mathrm{tr}(Q_i^C Q_j^C) \\
\Delta\lambda &= \langle H \rangle^{-1} g \\
\lambda^{(k+1)} &= \lambda^{(k)} + \Delta\lambda
\end{aligned}
$$

These improvements prevent the very inefficient multiplication $\bar{J}^A \bar{D} = (C_\epsilon^{-1}\bar{J})^T (C_{\theta|y} \bar{J}^T C_\epsilon^{-1})$ as these expressions are only evaluated inside a trace (of a product of matrices) which can be computed without actually calculating the matrix product utilizing Lemma 2.

Combining the improved E-step and improved M-step leads to the improved EM-algorithm. This variant of the algorithm prevents (relatively) large and dense matrices and as a side-effect reduces the number of overall operations significantly, since the matrices Q_i are diagonal matrices without fully populated diagonals.

2.3.3 Posterior Probabilities

Having computed the conditional expectation and covariance, the posterior probability of the parameters can be determined with:

$$
p_p = \Phi_N\Big(\frac{y - c^T \eta_{\theta|y}}{\sqrt{c^T C_{\theta|y} c}}\Big). \tag{2.78}
$$

The vector c specifies contrasts for the conditional parameters $\eta_{\theta|y}$. γ is a threshold to which the probability of it being exceeded is calculated as described in [13] and [17]. Φ_N is the cumulative normal distribution. The contrast c, in example, could be selected to give the average over all parameters, exclude or highlight certain parameters.

$$
\begin{aligned}
&while(K < c) && (2.77)\\
&\{\\
&\text{E-Step:}\\
&\bar{y} = \begin{pmatrix} y - h(\eta_{\theta|y}^{(k)}) \\ \eta_\theta - \eta_{\theta|y}^{(k)} \end{pmatrix}\\
&J = \frac{\delta h(\eta_{\theta|y}^{(k)})}{\delta \theta}\\
&\bar{J} = \begin{pmatrix} J & X \\ 1 & 0 \end{pmatrix}\\
&\bar{C_\epsilon} = \begin{pmatrix} \sum_i^l \lambda_i^{(k)} Q_i & 0 \\ 0 & C_\theta \end{pmatrix}\\
&\bar{J}^A = C_\epsilon^{-1} \bar{J}\\
&\bar{J}^B = \bar{J^A}^T\\
&C_{\theta|y} = (\bar{J}^B \bar{J})^{-1}\\
&\bar{D} = C_{\theta|y} \bar{J}^B\\
&\Delta\eta_{\theta|y} = \bar{D}\bar{y}\\
&\eta_{\theta|y}^{(k+1)} = \eta_{\theta|y}^{(k)} + \Delta\eta_{\theta|y}\\
&\text{M-Step:}\\
&Q_i^A = Q_i C_\epsilon^{-1}\\
&Q_i^B = Q_i \bar{J}^A\\
&Q_i^C = \bar{D} Q_i^B\\
&p_y = C_\epsilon^{-1}\bar{y} - \bar{J}^A \Delta\eta_{\theta|y}\\
&g_i = p_y^T Q_i p_y - tr(Q_i^A) + tr(Q_i^C)\\
&H_{ij} = tr(Q_i^A Q_j^A) - tr(Q_j^A Q_i^B \bar{D}) - tr(Q_i^A Q_j^B \bar{D}) + tr(Q_i^C Q_j^C)\\
&\Delta\lambda = \langle H \rangle^{-1} g\\
&\lambda^{(k+1)} = \lambda^{(k)} + \Delta\lambda\\
&\}
\end{aligned}
$$

Figure 2.16: Improved EM-algorithm

3 Implementation

3.1 Paradigms

This implementation of Dynamic Causal Modeling is realized in C++ to achieve a good overall performance and utilize parallelization to further improve the execution time. Furthermore the submodel system classes are implemented modularly to allow easy extensions of the mathematical models and also free combinations of different dynamic and forward submodels.

3.1.1 Parallelization

Three parts of the program code are particulary well suited for parallelization. First, the matrix multiplication, with its general (for sparse matrices worst case) complexity of n^3. Second, the computation of the parameter-jacobian in the E-step of the EM-algorithm and finally the calculation of the elements of $\frac{\delta F}{\delta \lambda_i}$ and $\frac{\delta^2 F}{\delta \lambda_{ij}^2}$ in the M-step of the EM-algorithm. In all three cases the parallelization involves exclusively spreading passes of "for-loops" to different threads. Further parallelization of blocks of the EM-algorithms did not proof to speed up the overall time consumption. The parallelization itself was implemented with the OpenMP library.

3.1.2 Modularization

One important point of this implementation is the modularization of the submodels. This not only allows an easy extension with further dynamic and forward submodels, but also the possibility to estimate data with different submodel combinations by a mere change of the file extension of the dataset.

3.1.3 The Layout

The DCM Implementation consists mainly of seven major classes. They are, in the order they are instantiated:

- `Import` - interprets the given parameter or data file and configuration file
- `Feed` - controls and distributes the input for the dynamic system
- `System` - represents the combined dynamic and forward system
- `Drift` - generates a drift matrix and handles the associated parameters

- `Output` - writes the different output files
- `Solver` - computes the systems of (ordinary) differential equations
- `Bayes` - estimates parameters from data with EM-algorithm

Furthermore two minor classes:

- `Vector` - dense vector
- `Matrix` - sparse matrix

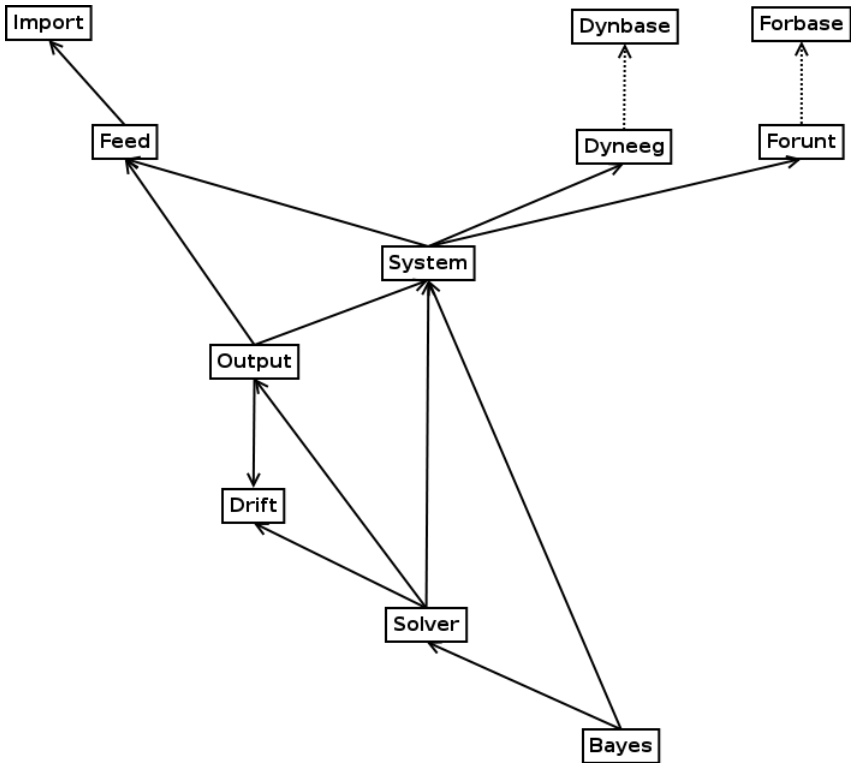

Figure 3.1: Class diagram of the DCM implementation

3.2 Description of Implemented Classes

3.2.1 Specification and Data Import Class[5]

This class processes the raw data or simulation files as well as the configuration file. In the case of a simulation file, the system specification, the parameters, and the input time series are read. For a raw data file, the system specification is determined by the

[5]The source code file of the import class is src/import.hh

file header (first row), the data and input time series are sorted and read. Finally, the series step size and time series length is computed. The `Import` class decides by the file extension whether it should interpret it as a simulation file for a simulation (`.dcm`) or as a datafile for an estimation (`.eeu`). The configuration file is named `dcm.ini`.

3.2.2 Data Feed Provider Class[6]

The `Feed` class preprocesses the input and prerecorded data, manages the input distribution to the dynamic system and data distribution to the EM-algorithm. The constructor expects a reference to the system specification structure and a pointer to the input and data array; both are provided by the `Import` class. The datapoints can be scaled to fit the correct magnitude, in example milli Volt (mV) in case of EEG data.

3.2.3 Drift Generator Class[7]

This class provides the drift generated by discrete cosine sets grouped into a global drift matrix and the associated parameters. The global drift matrix is assembled by the constructor. Upon request also a local drift matrix (by timestep) can be returned, in example for use in the `Solver` class.

3.2.4 Combined System Class[8]

The combined `system` class holds both dynamic and forward systems. This bundles the access for the solver to a single call. The use of modular dynamic and forward systems is enabled by this class, as both these member classes have to be derived from an interface defining base class.

3.2.5 Modularized Dynamic System Classes

As this Implementation accepts different types of dynamic systems, a common virtual base class is implemented to force each actual dynamic system class derived from this base class to provide a certain interface.

3.2.5.1 Dynamic Base Class[9]

The dynamic base class `Dynbase` provides an abstract interface for all derived dynamic systems. This interface comprises all necessary methods that are requested by the governing `System` class.

[6] The source code file of the feed class is src/feed.hh
[7] The source code file of the drift class is src/drift.hh
[8] The source code file of the system class is src/system.hh
[9] The source code file of the dynamic base class is src/dynbase.hh

3.2.5.2 Dynamic EEG System Class[10]

Derived from the dynamic base class `Dynbase`, this class implements the dynamic system as described in 2.2.1.2. The righthand-side is implemented as shown in (2.16). The initial value as well as the initial back states for the delays are set to zero vectors.

3.2.6 Modularized Forward System Classes

As for the dynamic system, the forward system classes are also derived from a common virtual base class for the same reasons as above.

3.2.6.1 Forward Base Class[11]

The forward base class `Forbase` provides an abstract interface for all derived forward systems. Similar to the dynamic base class `Dynbase` this class holds all methods that are requested by the `System` class.

3.2.6.2 Forward EEG Class[12]

Derived from the forward base class `Forbase`, it implements the forward system as described in 2.2.2.2. As this is not a system of differential equations, most of the methods are dummies.

3.2.7 Solver Class[13]

The `Solver` class offers methods to integrate the dynamic and forward systems through the interface of the `System` class. The drift, as generated by the `Drift` class, is applied from this class. The employed Runge-Kutta integration technique is described next.

3.2.7.1 Runge-Kutta-Fehlberg

For the integration of systems of ordinary differential equations, the Runge-Kutta-Fehlberg (RKF5) method is used. This is an explicit Runge-Kutta method of order five as decribed in [26]. Below the Butcher's Tableau corresponding to the used RKF5 method is given:

[10] The source code file of the dynamic eeg class is src/dyneeg.hh
[11] The source code file of the forward base class is src/forbase.hh
[12] The source code file of the forward eeg class is src/foreeg.hh
[13] The source code file of the solver class is src/solver.hh

0					
$\frac{1}{4}$	$\frac{1}{4}$				
$\frac{3}{8}$	$\frac{2}{32}$	$\frac{9}{32}$			
$\frac{12}{13}$	$\frac{1932}{2197}$	$-\frac{7200}{2197}$	$\frac{7296}{2197}$		
1	$\frac{439}{216}$	-8	$\frac{3680}{513}$	$-\frac{845}{4104}$	
$\frac{1}{2}$	$-\frac{8}{27}$	2	$-\frac{3544}{2565}$	$\frac{1859}{4104}$	$-\frac{11}{40}$
	$\frac{16}{135}$	$\frac{6656}{12825}$	$\frac{28561}{56430}$	$-\frac{9}{50}$	$\frac{2}{55}$

Figure 3.2: Runge-Kutta-Fehlberg (5th order) Butcher's Tableau

3.2.8 Output Class[14]

This class holds the output data of the dynamic and forward systems after the last integration of the system. It provides functionality to generate three types of output:

- Data Output, columnwise sorted input, error, output from dynamic and forward system, as well as the original data
- Plot Output, GnuPlot Script allowing to visualize the datapoints in the data output
- Parameter Output, a plain textfile listing the parameters and their estimated posterior values

3.2.9 Bayes Class[15]

The `Bayes` class implements the improved EM-algorithm, as described in (2.77) and can compute the posterior probabilities to a given contrast. Some implementation details to the EM-algorithm will be given below.

3.2.9.1 The Parameter-Jacobian

To calculate the parameter-jacobian matrix J for the EM-algorithm, a straightforward first-order (forward) finite difference approach is used.

$$J = \frac{\delta h(z,u,\theta)}{\delta\theta} \approx \frac{h(z,u,\theta+k) - h(z,u,\theta)}{k} \tag{3.1}$$

[14] The source code file of the output class is src/output.hh
[15] The source code file of the bayes class is src/bayes.hh

Though a second-order finite difference scheme would be significantly more accurate, due to the long assembly time of the parameter-jacobian with

$$J = \frac{\delta h(z, u, \theta)}{\delta \theta} \approx \frac{h(z, u, \theta + k) - h(z, u, \theta - k)}{2k} \tag{3.2}$$

this would be very inefficient, since the model would have to be integrated twice, for $h(z, u, \theta + k)$ and $h(z, u, \theta - k)$. The computation of the parameter-jacobian was parallelized by segmenting J columnwise and solving for each perturbed parameter.

3.2.9.2 The EM-Algorithm

In the EM-algorithm the variance matrix $\bar{C}_\epsilon$ is only needed in its inverted form. As $\bar{C}_\epsilon$ is a diagonal matrix, it can naturally be inverted very swiftly by inverting the diagonal entries. Also the assembly of $\bar{C}_\epsilon$ can be sped up with combined scalar multiplication and matrix addition (SAXPY) operations. The inversion of $\bar{J}^B\bar{J}$ (improved E-step) and the solving of $\Delta\lambda = H^{-1}g \Leftrightarrow H\Delta\lambda = g$ (M-step) are both accomplished with the Cholesky decomposition.

4 Numerical Experiments and Validation with Real and Artificial EEG Data

4.1 Preliminary Remark

The numerical experiments that were conducted to test the validity of the implemented parameter estimation, the required integration scheme and the combined dynamic and forward model includes two test series. First, a series of artificial EEG data, which was simulated with the implemented program. Second, a series of real life (EEG) data. The data, the associated hypothesis and background information was provided by Prof. Dr. Hans-Christian Pape, Dr. Thomas Seidenbecher and Dr. Jörg Lesting of the WWU Institute of Physiology I in Münster. This data was gathered in the course of fear experiments on conditioned mice conducted by Rajeevan Narayanan.

4.2 Artificial EEG Data

The artificial data experiments are grouped in two sets. First, simple 2-Region networks testing the connection types. Second, 3-Region networks testing the interaction of different connection types. All simulated datasets were downsampled to 8 ms. The rather less interesting parameters are kept at the prior value. Input was restricted to the first region only.

4.2.1 2-Region Tests

Initially three simple networks consisting of two regions were simulated and estimated to test the three connection types (forward, backward and lateral). For these tests of a duration of 800 ms with two input events that lasted for 300 ms was chosen, which was restricted to the first region. Next, these three cases will be presented with their schematic and the (simulated) EEG curves.

4.2.1.1 Forward Coupling Test[16]

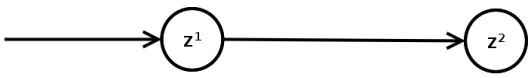

Figure 4.1: Forward Coupling Test Schematic

[16] see `syn_2f.dcm` and `syn_2f.eeu`

The first test includes a single forward connection[17] from the first to the second region. Below are the graphs of the simulated EEG curve and the difference between the simulated curve and the estimated curve.

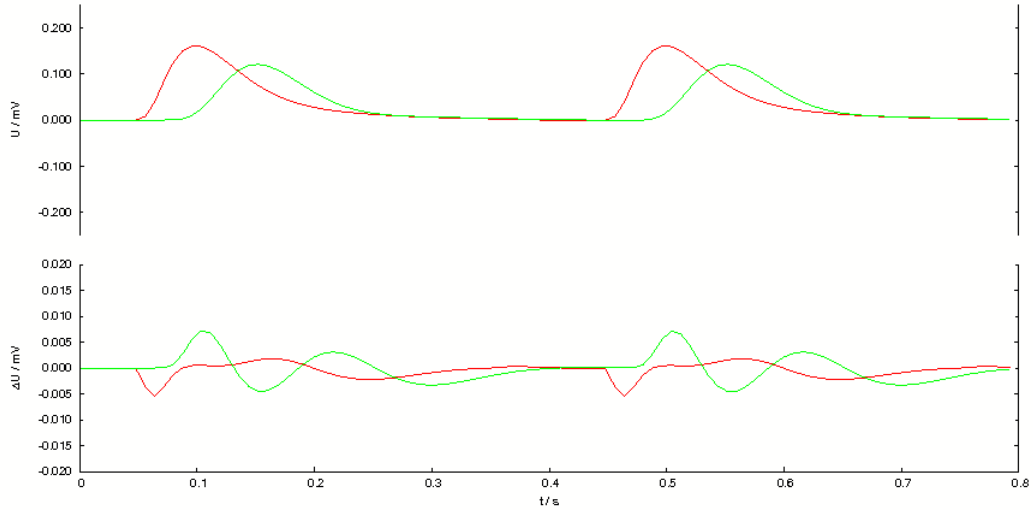

Figure 4.2: Top: EEG curve for the forward coupling test of the first (red) and second (green) region. Bottom: Difference between the simulated and estimated EEG curves.

The maximum error in the estimated curve is 7%, the average error is 1.5%.

4.2.1.2 Backward Coupling Test[18]

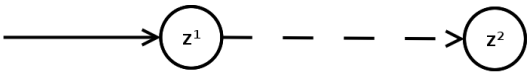

Figure 4.3: Backward Coupling Test Schematic

The second test includes a single backward connection[19] from the first to the second region. Below are the graphs of the simulated EEG curve and the difference between the simulated curve and the estimated curve.

[17] solid arrows
[18] see `syn_2b.dcm` and `syn_2b.eeu`
[19] dashed arrow

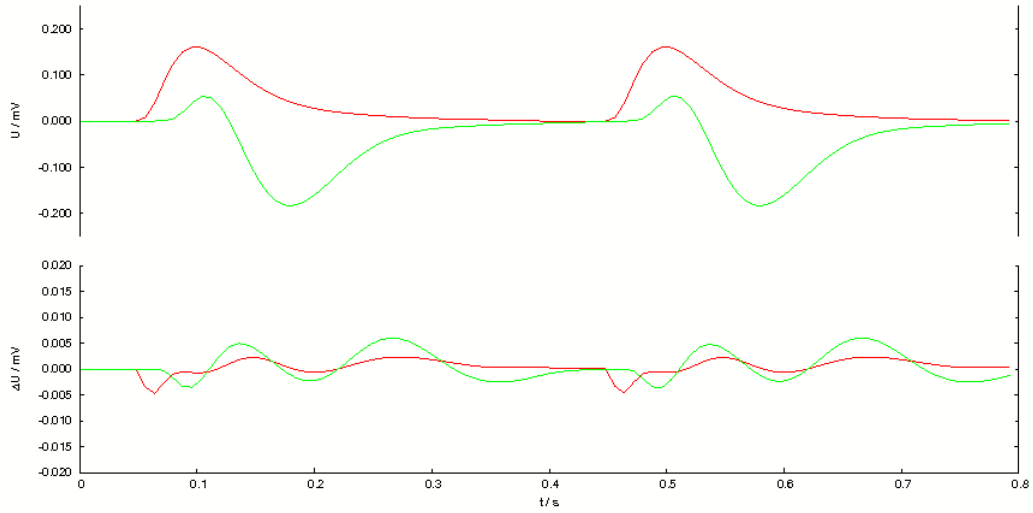

Figure 4.4: Top: EEG curve for the backward coupling test of the first (red) and second (green) second. Bottom: Difference between the simulated and estimated EEG curves.

The maximum error in the estimated curve is 6%, the average error is 0.7%.

4.2.1.3 Lateral Coupling Test[20]

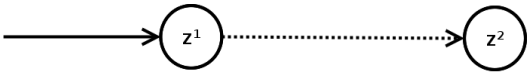

Figure 4.5: Lateral Coupling Test Schematic

The third test includes a single lateral connection[21] from the first to the second region. Below are the graphs of the simulated EEG curve and the difference between the simulated curve and the estimated curve.

[20] see `syn_2l.dcm` and `syn_2l.eeu`
[21] dotted arrow

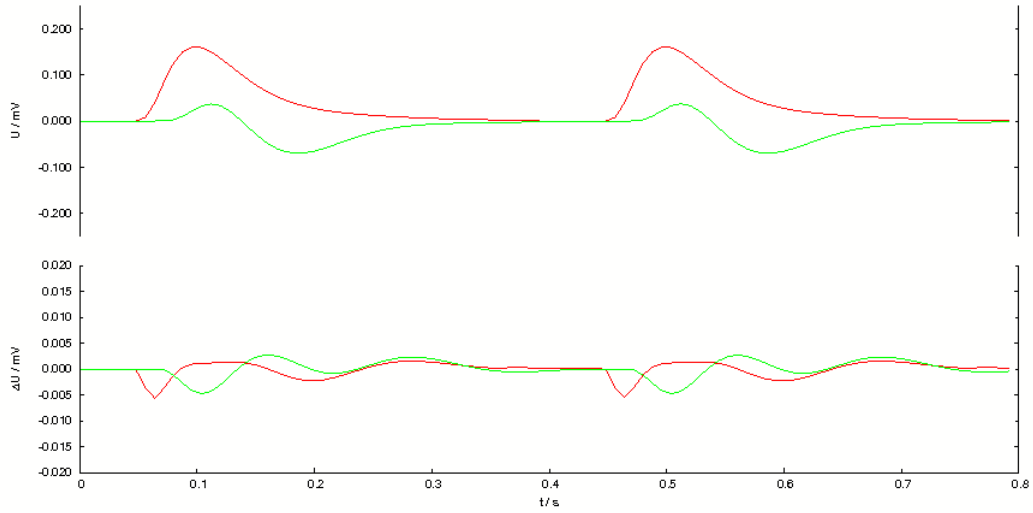

Figure 4.6: Top: EEG curve for the lateral coupling test of the first (red) and second (green) region. Bottom: Difference between the simulated and estimated EEG curves.

The maximum error in the estimated curve is 6%, the average error is 0.8%.

4.2.2 3-Region Tests

To test combinations of different connections five networks consisting of three regions were simulated. Similar to the two region tests a duration of 800 ms and two input events that lasted for 300 ms were chosen. Again input was restricted to the first region.

4.2.2.1 Forward-Lateral Test[22]

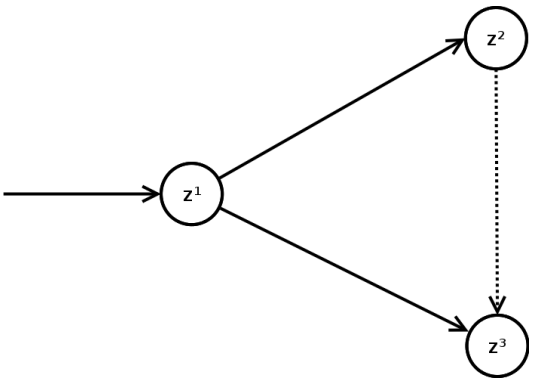

Figure 4.7: Forward-Lateral Test Schematic

[22] see `syn_3a.dcm` and `syn_3a.eeu`

For the first test the region that receives input distributes it evenly via forward connections[17] to the other two regions. The receiving two regions are connecting with a uni-directional lateral connection[21]. Below are the graphs of the simulated EEG curve and the difference between the simulated curve and the estimated curve.

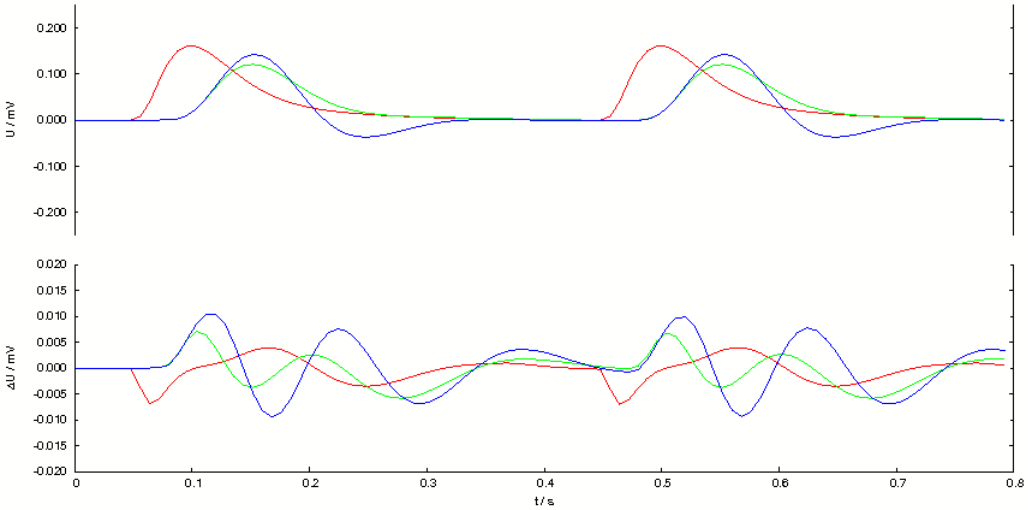

Figure 4.8: Top: EEG curve for the forward-lateral test of the first (red), the second (green) and the third (blue) region. Bottom: Difference between the simulated and estimated EEG curves.

The maximum error in the estimated curve is 10%, the average error is 2.7%.

4.2.2.2 Chained Forward Test[23]

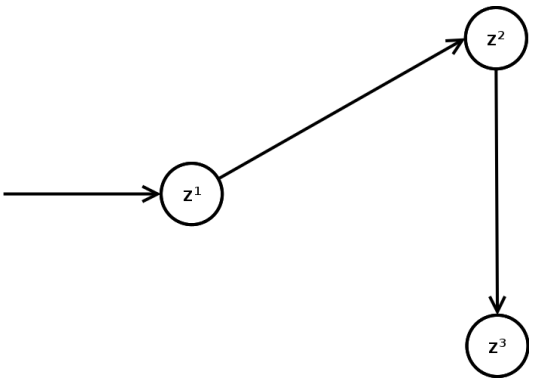

Figure 4.9: Chained Forward Test Schematic

[23] see `syn_3b.dcm` and `syn_3b.eeu`

For the second test the region that receives input relays it via a forward connection[17] to the second region, which in turn transmits it to the third region through a forward connection[17]. Below are the graphs of the simulated EEG curve and the difference between the simulated curve and the estimated curve.

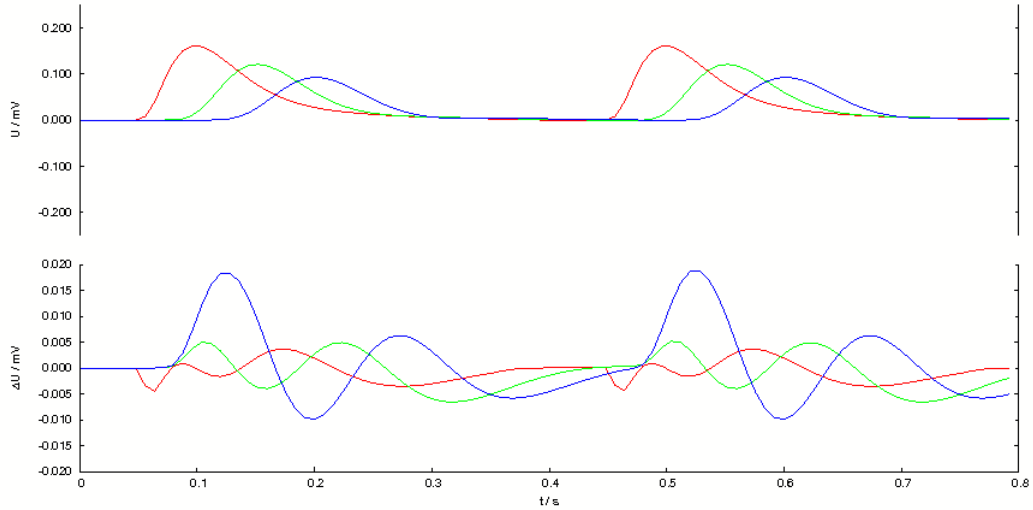

Figure 4.10: Top: EEG curve for the chained forward test of the first (red), the second green) and the third (blue) region. Bottom: Difference between the simulated and estimated EEG curves.

The maximum error in the estimated curve is 18%, the average error is 4.4%.

4.2.2.3 Forward-Lateral and Backward-Lateral Test[24]

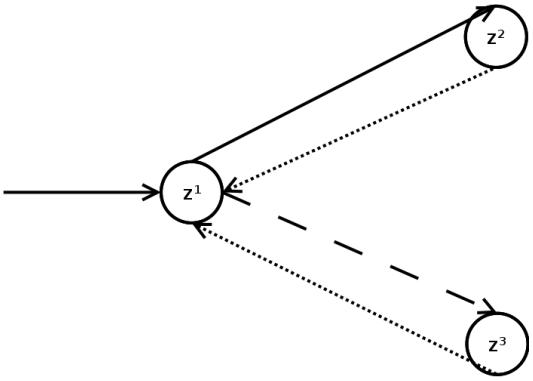

Figure 4.11: Forward-Lateral and Backward-Lateral Test Schematic

[24] see `syn_3c.dcm` and `syn_3c.eeu`

For the third test the region that receives input distributes it via a forward connection[17] to the second and via a backward[19] connection to the third. Both receiving regions are in turn coupled through a lateral connection[21] with the first region. Below are the graphs of the simulated EEG curve and the difference between the simulated curve and the estimated curve.

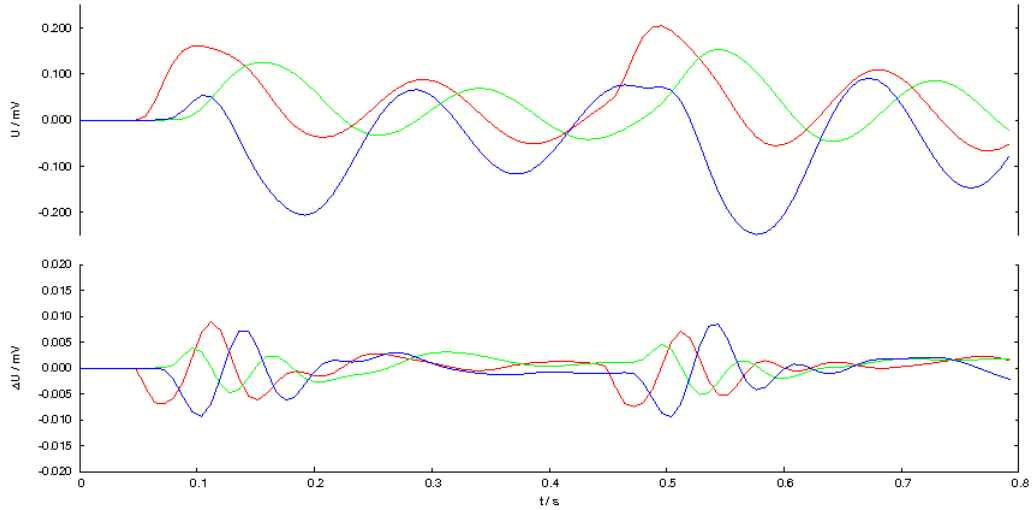

Figure 4.12: Top: EEG curve for the forward-lateral and backward-lateral test of the first (red), the second (green) and the third (blue) region. Bottom: Difference between the simulated and estimated EEG curves.

The maximum error in the estimated curve is 9%, the average error is 0.9%.

4.2.2.4 Forward-Backward and Lateral-Lateral Test[25]

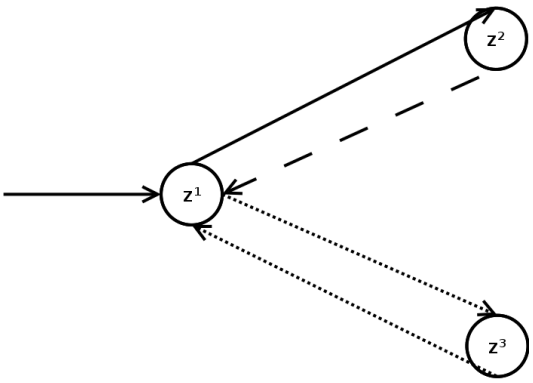

Figure 4.13: Forward-Backward and Lateral-Lateral Test Schematic

[25] see `syn_3d.dcm` and `syn_3d.eeu`

For the fourth test the region that receives input distributes it via a forward connection[17] to the second, which connects back to the first with a backward connection[19]. The third region is connected through a bidirectional lateral connection[21] with the first. Below are the graphs of the simulated EEG curve and the difference between the simulated curve and the estimated curve.

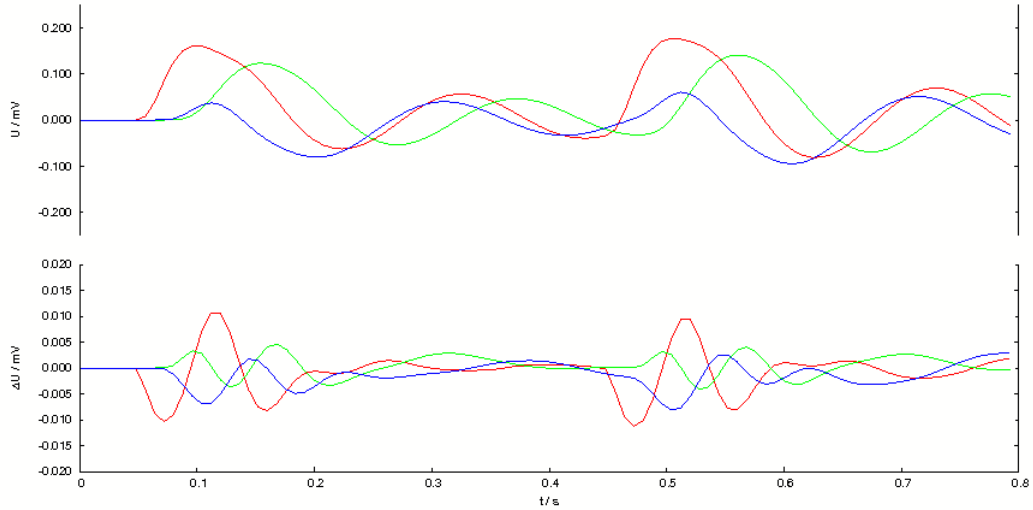

Figure 4.14: Top: EEG curve for the forward-backward and lateral-lateral test of the first (red), the second (green) and the third (blue) region. Bottom: Difference between the simulated and estimated EEG curves.

The maximum error in the estimated curve is 11%, the average error is 1.5%.

4.2.2.5 Full Connectivity Test[26]

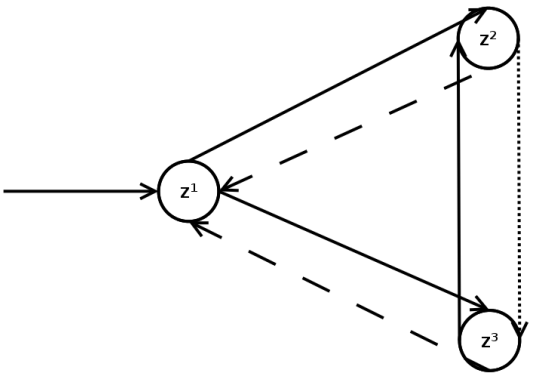

Figure 4.15: Full Connectivity Test Schematic

[26] see `syn_3e.dcm` and `syn_3e.eeu`

For the fifth test the region that receives input is connected with forward connections[17] with the other two and both connect back to the first via backward connections[19]. The second region is furthermore connected to the third through a lateral connection[21] and and third with the second via a forward connection[17]. Below are the graphs of the simulated EEG curve and the difference between the simulated curve and the estimated curve.

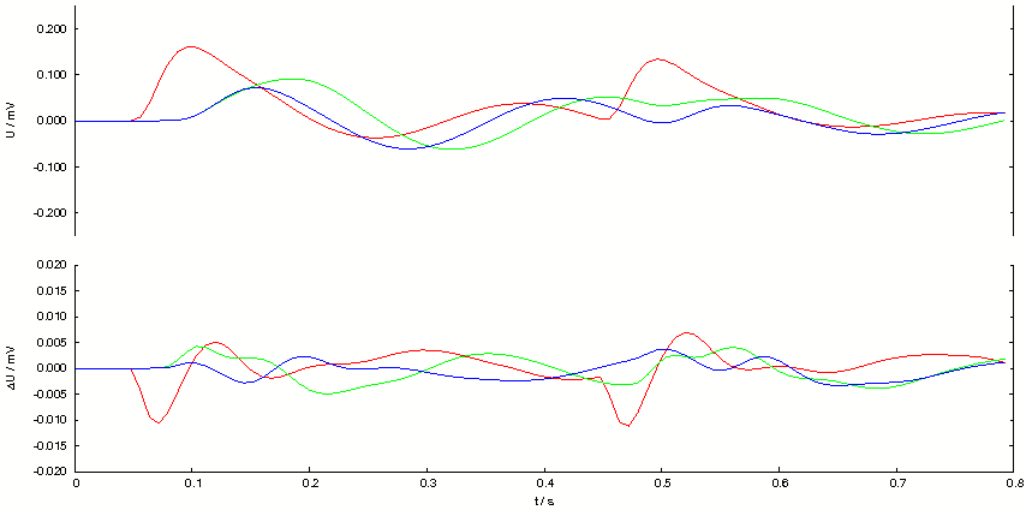

Figure 4.16: Top: EEG curve for the full connectivity test of the first (red), the second (green) and the third (blue) region. Bottom: Difference between the simulated and estimated EEG curves.

The maximum error in the estimated curve is 11%, the average error is 1.8%.

4.2.3 Conclusion

The first negative peak, which occurs at the beginning of an input stimulus, is related to the downsampling, which might displace the precise point in time where the input stimulus begins, and also to the sudden change of input and the consequential rise of energy in the dynamic system underlying the (dynamic sub-) model. The subsequent sinuous differences, due to the change in energy in the other regions, seem to be a damped oscillation.

4.3 Real EEG Data

The real life EEG data was recorded during fear experiments as described in [45], [30] and [43]. These were conducted with freely behaving mice, that were conditioned to experience fear when an auditory stimulus is presented. The complete experiment for one subject consisted of eleven sessions. The first three sessions are used to adapt (first

day) and condition (second day) the subject to the auditory stimulus with a sensoric stimulus (electric foot shock). A presentation of the auditory stimulus combined with a sensoric stimulus is marked CS^+, without a sensoric stimulus is marked CS^-. A day after the conditioning, six trials labeled $R1$ to $R6$ were conducted each 30 min apart. Those lasted for 360 s with eight stimulus presentation (four CS^- and four CS^+) for a duration of 10 s with a inter-stimulus period of 20 s. A day later two more sessions labeled $E1$ and $E2$ are conducted, structured equal to the session $R1$ to $R6$. For an overview of this experiment see the supplementary data to [30]. Analyzed was the first stimulus block marked CS^+ of the sessions $R1$ and $R6$.

4.3.1 Hypothesis

Three brain regions are considered, the lateral amygdala (AMY), the first cornu ammonis region (CA1) of the hippocampus and the prefrontal cortex (PFC). The constraints on the connectivity of these three considered brain regions is given by full connectivity between all regions as well as all regions being able to receive the experimental input. The hypothesis on this network is given by the lateral amygdala being the source of activitiy relayed to the CA1 which propagates to the PFC for the session $R1$, and a similar lower connectivity between these three regions for the session $R6$.

4.3.2 Data Preprocessing

During a single session the amplitudes of interest could be hidden beneath other signals generated by other concurrent processes in the same or a nearby cortex region. To emphasize the actual Event-Related Potentials (ERP), a set of data is generated by averaging over all available datasets of the same experimental session (see [2]), in this case over all sessions $R1$ during the first CS^+ to generate the first dataset, and over all sessions $R6$ also during the CS^+ for all ten testsubjects. This lessens the representation of random or unrelated effects of the individual trials. Furthermore the data was low-pass filtered to reflect the frequencies of interest, as well as downsampled to 8 ms.

4.3.2.1 Full CS^+ Block Analysis

This first estimation is applied to the first CS^+ block with a duration of ten seconds of the averaged $R1$ and $R6$ sessions.

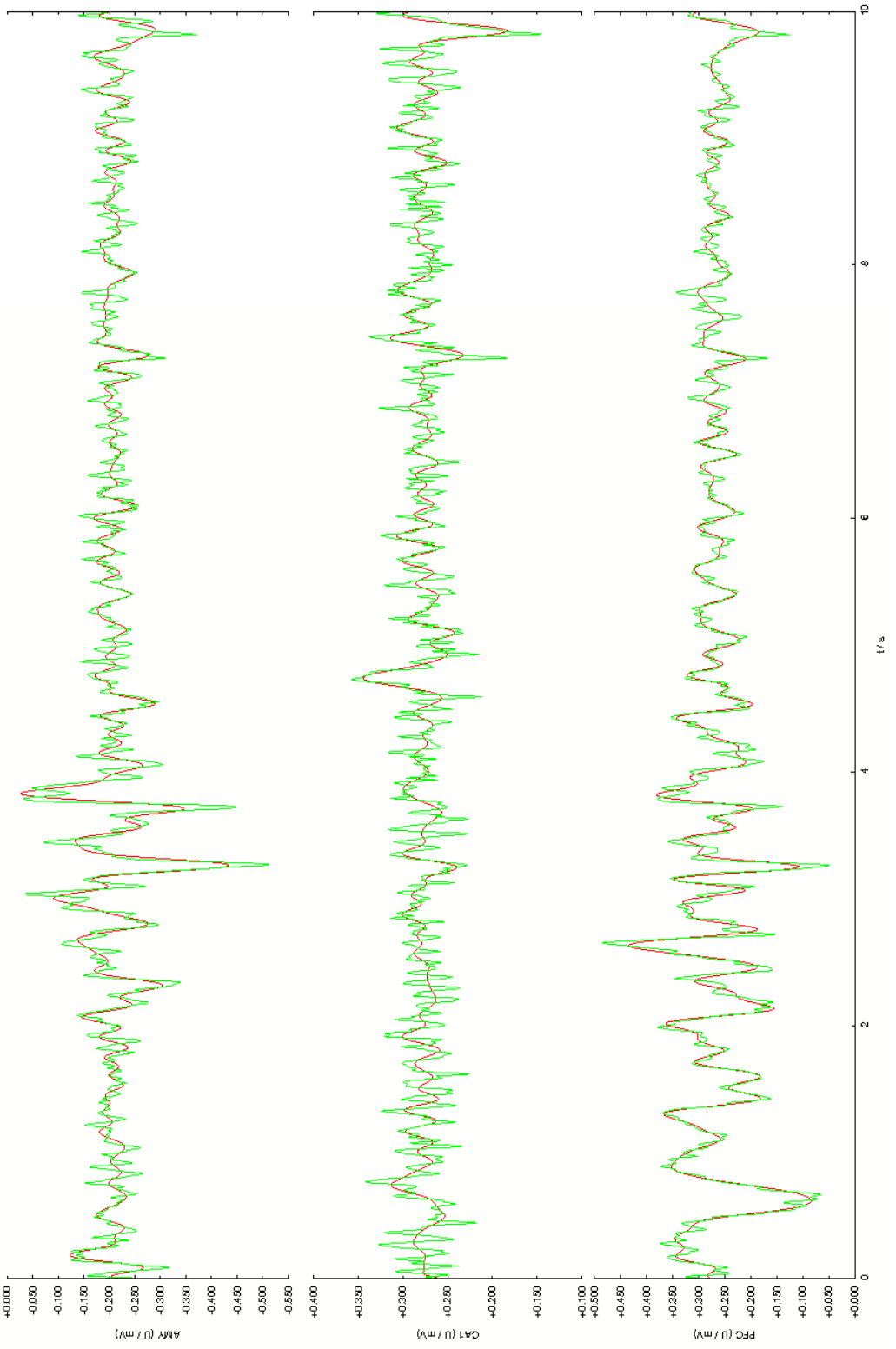

Figure 4.17: Estimation (green) of the complete first CS^+ block of the ERP (red) over all $R1$ sessions.

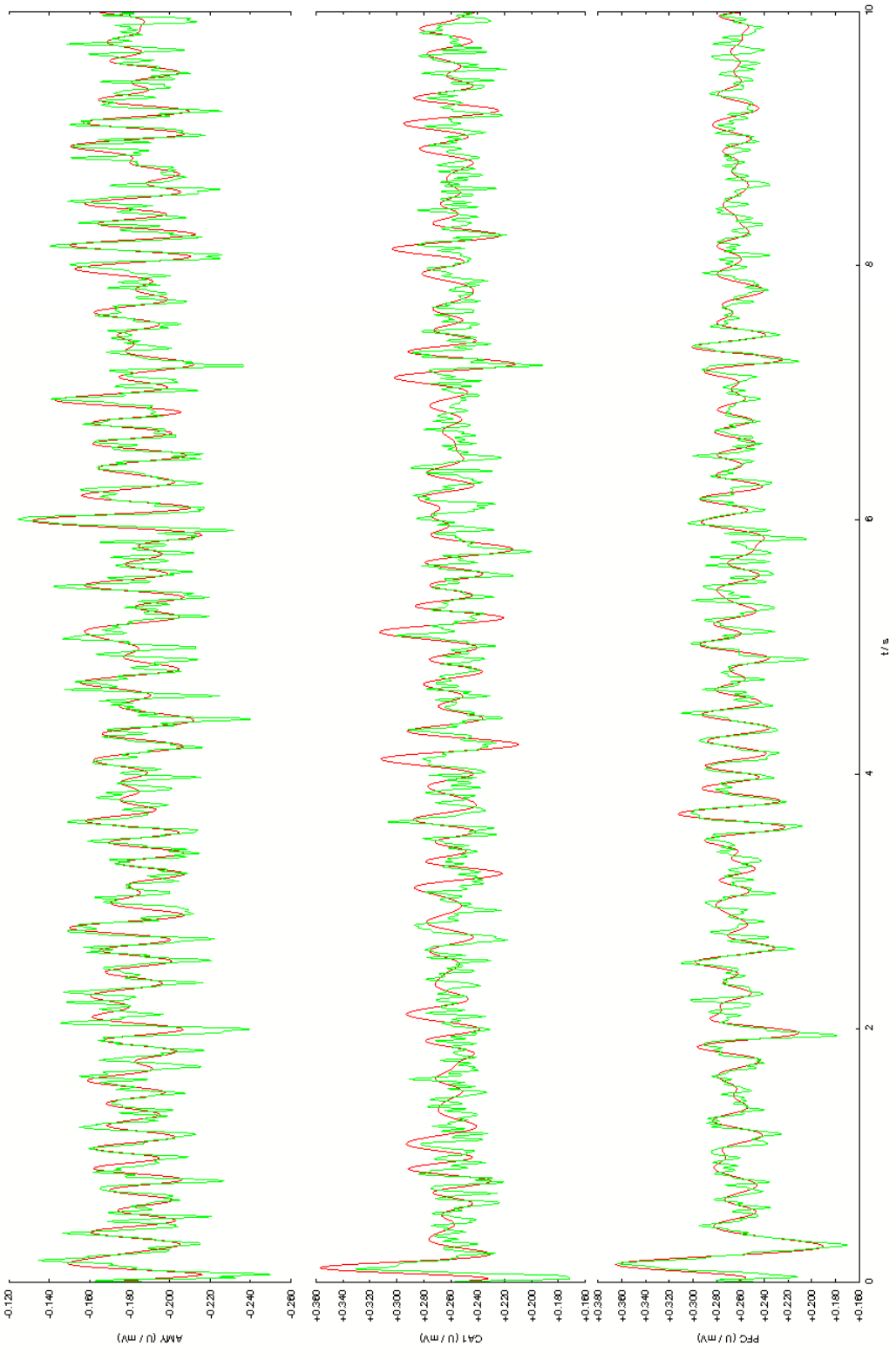

Figure 4.18: Estimation (green) of the complete first CS^+ block of the ERP (red) over all $R6$ sessions.

To fit the estimated curve to the data, a rather high order of the drift had to be chosen. To obtain a more accurate estimation of the coupling, a window of a single second is estimated next.

4.3.2.2 Partial CS^+ Block Analysis

For this estimation of the first CS^+ block, the time series is truncated to the second second of the full ten second block of the averaged $R1$ and $R6$ sessions. For the averaged $R1$ session the estimation leads to the following curve:

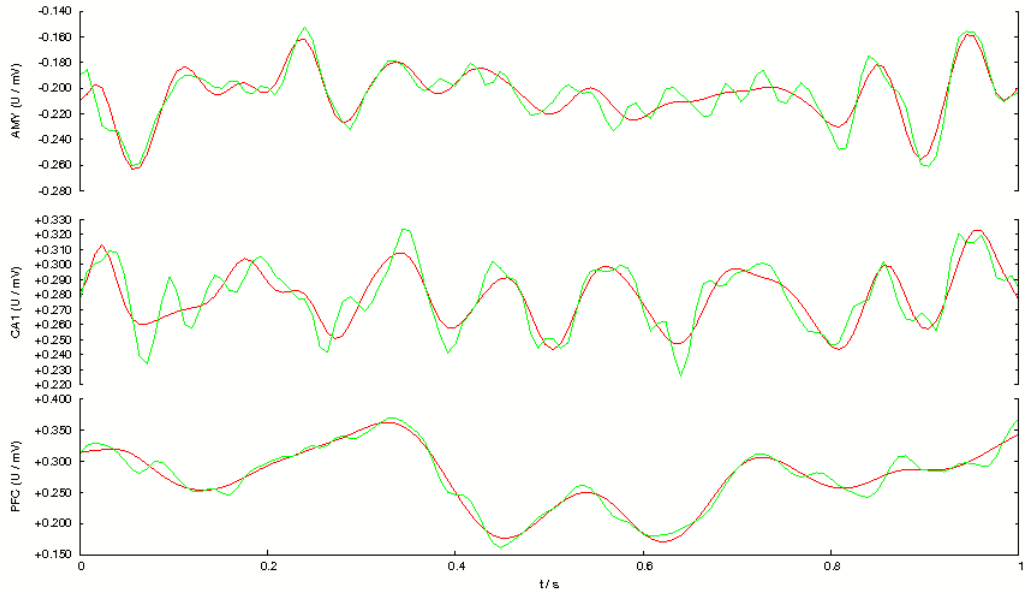

Figure 4.19: Measured EEG curve (green) and estimated EEG curve (red) of the second second of the first CS^+ block of the ERP.

The dominating estimated connections are presented next in the following schematic.

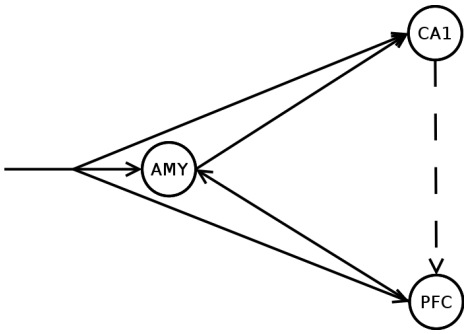

Figure 4.20: Coupling schematic of the estimated data, presenting the dominant coupling.

The estimation of the averaged $R6$ session is given by the following curve:

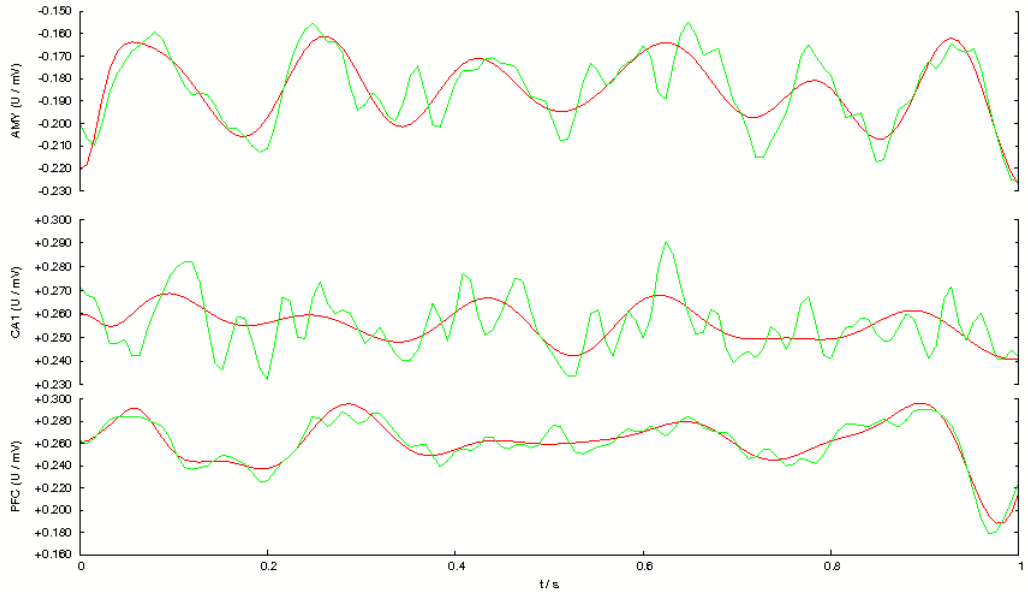

Figure 4.21: Measured EEG curve (green) and estimated EEG curve (red) of the second second of the first CS^+ block of the ERP.

The dominant estimated connections are presented next in the following schematic.

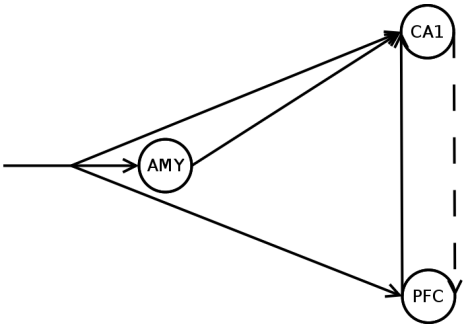

Figure 4.22: Coupling schematic of the estimated data, presenting the dominant coupling.

4.3.2.3 Experimental Discussion

The estimated coupling verifies the hypothesis on the connectivity of the considered brain regions. Additionally, a connection from the prefrontal cortex to the amygdalar is estimated for the averaged $R1$ session, and a connection from the prefrontal cortex to the first cornu ammonis region for the averaged $R6$ region. The dominating connections of the averaged $R6$ session are about $\frac{9}{20}$ in strength compared to the averaged $R1$ session.

4.3.2.4 Mathematical Discussion

The truncated time series is estimated more accurate, matching most of the peaks of the data with a relative low order drift. The current model seems incapable of estimating the progression of a measured EEG curve for long time series, especially if this time series comprises a single stimulus spanning the whole time series. The concept of adaptation, which might be necessary to estimate time series of this length, is not taken into account for this model, in particular the dynamic submodel for EEG. The measured curve can only be followed, if a very high order drift is used, which compensates for these shortcomings. Though this roughly approximates the measured EEG curve, it does not reflect the connectivity of the brain regions, since the peaks of the data are primarily matched by the drift. For shorter time series, the current model matches the data with a low order drift.

4.4 Performance Analysis

To test the performance gain of the parallelization and the improved EM-algorithm the artificial EEG data was evaluated in four variants with a preselected number of ten iterations of the EM-algorithm. First, the original EM-algorithm as presented in [16], [13] and [17]. Second, the original EM-algorithm with parallelization[27]. Third, the improved EM-algorithm as described in (2.77). Fourth, the improved EM-algorithm with parallelization[27].

Desktop System[28]	original EM (1 Thread)	original EM (3 Threads)	improved EM (1 Thread)	improved EM (3 Threads)
Syn_2f (see 4.2.1.1)	270.79 s	135.90 s	55.29 s	36.83 s
Syn_2b (see 4.2.1.2)	269.28 s	139.77 s	55.84 s	36.46 s
Syn_2l (see 4.2.1.3)	269.87 s	137.51 s	55.28 s	30.81 s
Syn_3a (see 4.2.2.1)	1565.94 s	618.06 s	135.05 s	79.67 s
Syn_3b (see 4.2.2.2)	1583.37 s	626.23 s	135.25 s	78.99 s
Syn_3c (see 4.2.2.3)	1581.86 s	625.29 s	133.61 s	78.09 s
Syn_3d (see 4.2.2.4)	1564.45 s	641.86 s	136.53 s	80.09 s
Syn_3e (see 4.2.2.5)	1587.07 s	627.42 s	136.77 s	80.52 s

Figure 4.23: Comparision of the original and improved EM-algorithm variants with and without parallelization on a desktop computer

[27] As described in (3.1.1)

[28] Intel Atom 330 - a Dual-Core (In-Order) Processor with Hyperthreading (logical Quad-Core) and 2GB of RAM

Workstation System[29]	original EM (1 Thread)	original EM (22 Threads)	improved EM (1 Thread)	improved EM (22 Threads)
Syn_2f (see 4.2.1.1)	74.68 s	33.43 s	8.38 s	2.73 s
Syn_2b (see 4.2.1.2)	74.61 s	32.92 s	8.44 s	2.66 s
Syn_2l (see 4.2.1.3)	74.40 s	33.12 s	9.97 s	2.41 s
Syn_3a (see 4.2.2.1)	498.78 s	169.14 s	21.57 s	5.39 s
Syn_3b (see 4.2.2.2)	495.31 s	167.75 s	21.66 s	5.35 s
Syn_3c (see 4.2.2.3)	499.30 s	166.36 s	21.50 s	5.52 s
Syn_3d (see 4.2.2.4)	500.64 s	168.26 s	21.68 s	5.44 s
Syn_3e (see 4.2.2.5)	496.84 s	168.27 s	21.56 s	5.33 s

Figure 4.24: Comparision of the original and improved EM-algorithm variants with and without parallelization on a workstation computer

Workstation System[29]	improved EM Syn_3e (see 4.2.2.5)
1 Thread	21.56 s
2 Threads	13.46 s
3 Threads	12.39 s
4 Threads	9.37 s
5 Threads	8.02 s
6 Threads	6.79 s
7 Threads	5.59 s
8 Threads	5.08 s
9 Threads	8.26 s
10 Threads	4.50 s
11 Threads	4.64 s
12 Threads	4.65 s
13 Threads	5.32 s
14 Threads	5.54 s
15 Threads	5.42 s
16 Threads	6.97 s
17 Threads	5.38 s
18 Threads	5.36 s
19 Threads	5.52 s
20 Threads	4.83 s
21 Threads	4.92 s
22 Threads	5.33 s

Figure 4.25: Comparison of runtime with varying numbers of threads on a workstation computer

[29]Intel Xeon 5660 - a 12-Core (Out-Of-Order) Processor with Hyperthreading (logical 24-Core) and 96GB of RAM

Both measures, either parallelization or the improved EM-algorithm, enhance the overall performance of the parameter distribution estimation independent from each other. Especially noteworthy is that the improvements to the EM-algorithm have a bigger impact on the runtime than the parallelization of the original EM-algorithm. This is due to the massive amount of floating-point operations that are not computed. On a desktop computer the runtime can be reduced to almost a twentieth with the parallelized improved EM-algorithm compared to the not parallelized original EM-algorithm. On a workstation this reduction can be even almost the factor hundred. The parallelization seems to be optimal with ten threads. For systems with more neuronal states, significantly longer time series or other measures that change the size of the design matrix $\bar{J}$, in example a higher order of drift or more parameters, the number of threads that results in the lowest runtime can be higher.

4.5 Outlook

Besides the modifications of the EEG dynamic submodel as presented in [35] and [34], which would include an adaptation to a stimulus, a new upgrade from this (bi-)linear model to a nonlinear model could be made. This can conceptionally be based on the nonlinear extension of the dynamic submodel for fMRI as developed in [49], encompassing a mechanism by which the neuronal state of a region can influence the coupling of others. Developing a nonlinear dynamic submodel for EEG could allow the modeling of faster changes in connectivity. Additionally or alternatively the gain matrices G_k can be parametrized through polynomials allowing more complex input induced changes in connectivity. This could, especially for time series with long stimulus durations, estimate the change in coupling better than the current constant gain matrices.

Moreover the comparison of different models could be incorporated to determine the evidence of an estimated model like described in [39]. With the model comparison, in example the drift order or intrinsic connectivity fitting the data best can be determined. A major acceleration of the EM-algorithm can be made utilizing ordered subsets as described in [27]. A problem of the current variant of the EM-algorithm is, that the hyperparameter estimation can produce negative weights for the parameter estimation. This can lead to non positive definite design matrices that consequently cannot be decomposed by the Cholesky decomposition. To prevent this, a modification to the covariance component estimation similar to the hyperparametrization described in the appendix of [19] can be made. This would require a reformulation of the hyperparameter estimation. To further reduce the runtime of the program, the Cholesky decomposition could be parallelized. Finally, the solver could be parallelized as well and equipped with adaptive timesteps, for example with RKF45 or RKF56.

5 Appendix

5.1 Program Usage

The program can be started from the linux command line via `./dcm dat/simu.dcm` for simulations or `./dcm dat/data.eeu`. Additionally it can be chosen which states are enabled to receive extrinsic input by adding a binary as second argument. For example: `./dcm /dat/data.eeu 110`, assuming data.eeu is a dataset of three states, the first two states will be able receive input the third state is excluded from the set of parameters and set to zero. Omitting this second argument is equivalent to all regions being able to receive extrinsic input.

5.2 Dependencies

Required packages are:

- `libgomp1` available in most repositories,
- `gcc` (Version: >=4.2) available in most repositories,
- `gnuplot` (Version: >=4.4) available in most repositories (only for data visualization)

5.3 Program and Source Code License

The source code is licensed under the zlib-license. Following is the license text:

the product documentation would be appreciated but is not required.

2. Altered source versions must be plainly marked as such, and must not be misrepresented as being the original software.

3. This notice may not be removed or altered from any source distribution.

5.4 Abbreviations

DCM	Dynamic Causal Modeling
fMRI	functional Magnetic Resonance Imaging
EEG	Electroencephalography
ERP	Event Related Potential
MEG	Magnetoencephalography
BOLD	Blood Oxygen Level Dependency
EM	Expectation Maximization
RKF5	5th-Order Runge-Kutta-Fehlberg
SAXPY	Scalar Alpha times X Plus Y
AMY	Amygdala
CA1	Cornu Ammonis 1st Region
PFC	Prefrontal Cortex

5.5 Symbol Index

Kronecker Product	$\otimes$
Hadamard Product	$\circ$
Convolution Operator	$*$
Matrix Trace	tr
Diagonal Matrix	diag

Bibliography

[1] M. Abromowitz and I. A. Stegun. *Handbook of Mathematical Functions.* National Bureau of Standards, 1972.

[2] M. G. H. Coles and M. D. Rugg. *Event-related brain potentials: an introduction.* 1996.

[3] J. Daunizeau et al. Dynamic causal modelling: A critical review of the biophysical and statistical foundations. *NeuroImage*, 2009.

[4] J. Daunizeau et al. Dynamic causal modelling of distributed electromagnetic responses. *NeuroImage*, 47:590–601, 2009.

[5] O. David et al. Modelling event-related responses in the brain. *NeuroImage*, 25:756–770, 2005.

[6] O. David et al. Dynamic causal modeling of evoked responses in EEG and MEG. *NeuroImage*, 30:1255–1272, 2006.

[7] O. David et al. Mechanisms of evoked and induced responses in MEG/EEG. *NeuroImage*, 31:1580–1591, 2006.

[8] O. David and K. J. Friston. A neural mass model for MEG/EEG coupling and neuronal dynamics. *NeuroImage*, 20:1743–1755, 2003.

[9] A. P. Dempster and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of Royal Statistical Society*, 39:1–38, 1977.

[10] L. Fahrmeier and G. Tutz. *Multivariate Statistical Modelling Based on Generalized Linear Models.* Springer, 2001.

[11] M. Fastenrath et al. Dynamical causal modelling for M/EEG: Spatial and temporal symmetry constraints. *NeuroImage*, 44:154–163, 2009.

[12] R. S. J. Frackowiak. *Human Brain Function Second Edition.* Academic Press, 2004.

[13] K. J. Friston. Bayesian Estimation of Dynamical Systems: An Application to fMRI. *NeuroImage*, 16:513–530, 2002.

[14] K. J. Friston et al. Nonlinear Responses in fMRI: The Balloon Model, Volterra Kernels, and Other Hemodynamics. *NeuroImage*, 12:466–477, 2000.

[15] K. J. Friston et al. Classical and Bayesian Inference in Neuroimaging: Applications. *NeuroImage*, 16:484–512, 2002.

[16] K. J. Friston et al. Classical and Bayesian Inference in Neuroimaging: Theory. *NeuroImage*, 16:465–483, 2002.

[17] K. J. Friston et al. Dynamic causal modelling. *NeuroImage*, 19:1273–1302, 2003.

[18] K. J. Friston et al. Bayesian Estimation of Evoked and Induced Responses. *Human Brain Mapping*, pages 722–735, 2006.

[19] K. J. Friston et al. Variational free energy and the Laplace approximation. *NeuroImage*, 34:220–234, 2006.

[20] K. J. Friston and W. D. Penny. Posterior probability maps and SPMs. *NeuroImage*, 19:1240–1249, 2003.

[21] M. Garrido et al. Dynamic causal modelling of evoked potentials: A reproducibility study. *NeuroImage*, 36:571–580, 2007.

[22] A. Gelman. *Bayesian Data Analysis*. Chapman and Hall/CRC, 2004.

[23] L. Grüne. Numerical Stabilization of Bilinear Control Systems. *SIAM Journal on Control and Optimization*, 34, 1996.

[24] L. Grüne and O. Junge. *Gewöhnliche Differentialgleichungen*. Vieweg+Teubner, 2009.

[25] D. A. Harville. Maximum Likelihood Approaches to Variance Component Estimation and to Related Problems. *Journal of American Statistical Association*, 72:320–338, 1977.

[26] M. Hermann. *Numerik Gewöhnlicher Differentialgleichungen*. Oldenbourg, 2004.

[27] H. M. Hudson and R. S. Larkin. Accelerated Image Reconstruction using Ordered Subsets of Projection Data. *IEEE Transactions on Medical Imaging*, pages 601–609, 1994.

[28] B. H. Jansen and V. G. Rit. Electroencephalogram and visual evoked potential generation in a mathematical model of coupled cortical coloumns. *Biological Cybernetics*, 73:357–366, 1995.

[29] V. K. Jirsa and A. R. McIntosh. *Handbook of Brain Connectivity*. Springer, 2007.

[30] K. Jüngling et al. Neuropeptide S-Mediated Control of Fear Expression and Extinction: Role of Intercalated GABAergic Neurons in the Amygdala. *Neuron*, 59:298–310, 2008.

[31] S. J. Kiebel et al. Dynamic causal modelling of evoked responses in EEG/MEG with lead field parametrization. *NeuroImage*, 30:1273–1284, 2006.

[32] J. T. Linn. Approximating the cumulative chi-square distribution and its inverse. *Journal of Royal Statistical Society*, 37:3–5, 1988.

[33] R. J. Moran et al. A neural mass model of spectral responses in electrophysiology. *NeuroImage*, 37:706–720, 2007.

[34] R. J. Moran et al. Bayesian estimation of synaptic physiology from spectral responses of neural masses. *NeuroImage*, 42:272–284, 2008.

[35] R. J. Moran et al. Dynamic causal models of steady-state responses. *NeuroImage*, 44:796–811, 2009.

[36] R. M. Neal and G. E. Hinton. A View of the EM Algorithm that Justifies Incremental, Sparse, and Other Variants. pages 355–368, 1998.

[37] J. Nocedal and S. Wright. *Numerical Optimization*. Springer, 2000.

[38] V. Pan and R. Schreiber. An Improved Newton Iteration for the Generalized Inverse of a Matrix, with Applications. *SIAM*, 1990.

[39] W. D. Penny et al. Comparing dynamic causal models. *NeuroImage*, 22:1157–1172, 2004.

[40] K. B. Petersen and M. S. Pedersen. *The Matrix Cookbook*. Technical University of Denmark, October 2008. Version 20081110.

[41] W. H. Press. *Numerical Recipes in C++ Second Edition*. Camebridge University Press, 2002.

[42] A. Quarteroni and F. Saleri R. Sacco. *Numerical Mathematics*. Springer, 2007.

[43] S. Sangha et al. Deficiency of the 65 kDa Isoform of Glutamic Acid Decarboxylase Impairs Extinction of Cued But Not Contextual Fear Memory. *Journal of Neuroscience*, 29:15714–15720, 2009.

[44] G. A. F. Seber. *A matrix handbook for statisticians*. Wiley, 2008.

[45] T. Seidenbecher et al. Amygdalar and Hippocampal Theta Rhythm Synchronization During Fear Memory Retrieval. *Science*, 301:846–850, 2003.

[46] K. E. Stephan et al. Comparing hemodynamic models with DCM. *NeuroImage*, 38:387–401, 2007.

[47] K. E. Stephan et al. Dynamic causal modelling of evoked responses: The role of intrinsic connections. *NeuroImage*, 36:332–345, 2007.

[48] K. E. Stephan et al. Dynamic causal models of neural system dynamics: current state and future extensions. *Journal of Biosciences*, 32:129–144, 2007.

[49] K. E. Stephan et al. Nonlinear dynamic causal models for fMRI. *NeuroImage*, 42:649–662, 2008.