Institute of
Communication Systems
Prof. Dr.-Ing. Peter Jax

RWTH AACHEN UNIVERSITY

Investigation on Autoencoder Models for Online System Identification

Master's Thesis by Maximilian Tillmann

Submission date: February 24, 2023

Communicated by Univ.-Prof. Dr.-Ing. Peter Jax

Abstract

Speech communication devices are indispensable in our daily work and personal lives. Using them in hands free mode can create an echo signal, which, if no action is taken, would disturb the speaker. However, the echo signal can be predicted, when the impulse response between loudspeaker and microphone is known. For this task, system identification algorithms exist, such as the Least-Mean-Square (LMS) algorithm, the Normalized-Least-Mean-Square (NLMS) algorithm, and the Kalman filter. They work well in general, but face difficulties when confronted with high correlation input signals, high noise levels, or rapidly changing impulse responses over time.

This thesis aims to explore whether prior knowledge about the impulse response can improve system identification. The key approach is to utilize the manifold hypothesis, which has shown promising results in previous works in mapping acoustic room impulse responses to a lower dimensional subspace. These approaches require training data of impulse responses. This thesis investigates how well affine subspace models can represent impulse response with a limited number of subspace components compared to the same number of components in the time domain. One well known way to find an optimal affine subspace is by Principal-Component-Analysis (PCA). It is shown that the affine subspace model can have the same achievable system mismatch with significantly less number of subspace components, when the loudspeaker and the microphone are constrained in their positions.

The manifold LMS algorithm, the manifold NLMS algorithm and the manifold Kalman filter are proposed in this thesis, which can utilise general non linear manifolds for the acoustic echo compensation task. For the manifold LMS and NLMS algorithm in the case of white noise excitation and an affine manifold, the expected convergence speed and the expected steady state system mismatch are derived theoretically and are shown to accurately describe the algorithms behaviour in simulations. For scenarios with constrained loudspeaker and microphone positions it is shown that the manifold NLMS algorithm significantly outperforms the time domain NLMS algorithm. The manifold Kalman filter is compared to the time domain Kalman filter and another subspace approach from literature. The manifold Kalman filter shows faster initial convergence speed in simulations when the achievable steady state system mismatch is set to be the same for all approaches, which can be explained by its larger step size compared to the reference approaches.

Further research could include the evaluation of the proposed algorithms for non linear manifolds, which can be obtained by neural autoencoders or locally affine subspaces. The latter approach includes the search for an optimal distance measure to select the nearest neighbours of an impulse response in the training data.

Contents

Chapter 1

Introduction

Speech communication devices are indispensable in our daily work and personal lives. It is of great importance that the speech audio quality is on a high level, also in acoustically challenging situations, such as fast changing acoustic environments and high levels of interfering noise. In recent years, many people use their communication devices increasingly often in hands free mode, meaning that the loudspeaker of a device is not directly placed on the ear and no headphones are used, and the microphone is not directly near the mouth. This brings significantly more comfort, but has the disadvantage that the loudspeaker needs to be louder, which means that the microphone picks up the undesirable echo from the far end speaker, which is then sent back to the far end listener. It is very disturbing for the far end speaker to constantly hear his own voice with a delay. Therefore, the echo signal needs to be identified and compensated, ideally without distorting the near end speech signal.

To deal with such an echo problem in a phone call, an online system identification algorithm can be used to identify the acoustic impulse response between the microphone and the loudspeaker in real time. If the impulse response between the microphone and the loudspeaker is estimated sufficiently well, the echo signal can be computed and is then subtracted from the microphone signal, which, under ideal conditions and a perfect estimation, would remove the echo entirely, such that only the near end speech signal is transmitted.

For the task of online identification of acoustic systems, several algorithms have been proposed in the literature, for example the well studied Least-Mean-Square (LMS) algorithm, the Normalized-Least-Mean-Square (NLMS) algorithm, and the Kalman filter [35] with active research going on in this area to improve existing algorithms. For certain acoustic scenarios, the LMS algorithm, the NLMS algorithm, and the Kalman filter do well in the system identification task. However, in other scenarios, for example, using excitation signals with high correlation (e.g. speech signals), or with high noise levels on the near end, or when the impulse response is changing over time at a fast rate by movements of the loudspeaker and the microphone, the performance of the existing algorithms is severely degraded.

The main aim of this thesis is to investigate whether prior knowledge about the impulse response can be exploited to improve the performance of the system identification task. The manifold hypothesis, which is a basic assumption in many machine learning systems, states that all high dimensional data, which carries meaningful information, lies on a lower dimensional subspace. This subspace is usually referred to as a manifold. Applying this manifold hypothesis in previous studies yielded generally good results to map acoustic room impulse responses to a manifold with minor restrictions. All such manifold approaches assume that training data, which generalises well to other unseen scenarios, is available. This data is used to learn a manifold to represent acoustic impulse responses with a smaller number of components compared to true length of the impulse response in the time domain. We propose modified versions of the previously mentioned algorithms, namely the manifold LMS algorithm, the manifold NLMS algorithm, and the manifold Kalman filter, and it is analysed how the exploitation of a learned manifold can improve the performance of the system identification task.

In the following, first the theoretical background is explained and the recent relevant literature is reviewed in Chapter 2. In Chapter 3, the proposed algorithms are derived and analysed theoretically for their convergence speed and their steady state performance. In Chapter 4, the proposed algorithms are tested in simulations for various scenarios and are compared to the standard LMS algorithm, NLMS algorithm and the Kalman filter. Lastly, the results are summarised and an outlook to new research directions is given in Chapter 5.

Chapter 2

Fundamentals

In telephone communication in hands free mode, where the far end listener uses a loudspeaker and a microphone, the microphone can pick up the echo from the loudspeaker, which degrades the perceptual quality significantly. One approach to solve this problem is to estimate the impulse response of the near end room, which allows to compensate the echo that is picked up by the microphone. In the literature there are several algorithms described to perform echo compensation, and the optimization of such algorithms is an active research field. This thesis uses the LMS algorithm, the NLMS algorithms, and the Kalman filter, which are explained in the following.

2.1 Online System Identification and Acoustic Echo Compensation

To do the Acoustic-Echo-Compensation (AEC), the impulse response between the loudspeaker and the microphone at the near end needs to be estimated. Several algorithms exist to estimate this impulse response, such as the LMS algorithm, the NLMS algorithm, and the Kalman filter. In the following section, first the signal model is introduced and then the LMS algorithm, the NLMS, the Kalman filter and the extended Kalman filter are explained in the context of AEC.

2.1.1 The system identification task

In the application of echo compensation, the system identification task is identifying the true impulse response $\boldsymbol{w}_\star$ between the loudspeaker and the microphone at the near end. The signal graph for the identification of the impulse response $\boldsymbol{w}_\star$ is shown in Figure 2.1.

The signal coming from the far end $\boldsymbol{x}$ seen in Figure 2.1 is synonymously referred to as the input signal, the excitation signal, or the far end signal. At each discrete time step k, the vector $\boldsymbol{x}(k)$ contains the last l instances from $\boldsymbol{x}$ starting at time index k, where l is the length of the true impulse response between the near end speaker and microphone $\boldsymbol{w}_\star$ with

$$\boldsymbol{x}(k) = \begin{bmatrix} x_k & x_{k+1} & \dots & x_{k+l-1} \end{bmatrix}^T, \tag{2.1}$$

where the lowercase bold letter denote a vector, and the lowercase non bold letters denote scalars. The far end signal $\boldsymbol{x}(k)$ is emitted by the loudspeaker, propagates trough the room, and is picked up by the microphone. Possible noise present in the room and the inherent microphone noise is modelled with one noise term n.

Since a linear system is assumed, the echo signal is computed by $\boldsymbol{w}_\star^T \boldsymbol{x}(k)$ and the microphone signal is given as

$$y(k) = \boldsymbol{w}_\star^T \boldsymbol{x}(k) + n(k). \tag{2.2}$$

The impulse response estimate $\boldsymbol{w}(k)$ is used to compute an estimate of the echo signal $\boldsymbol{w}(k)^T \boldsymbol{x}(k)$. This estimated echo signal is subtracted from the microphone signal $y(k)$ to cancel out the echo resulting in

$$e(k) = y(k) - \boldsymbol{w}(k)^T \boldsymbol{x}(k), \tag{2.3}$$

which is called the error signal or the residual echo signal. The estimate of the impulse response $\boldsymbol{w}(k)$ is adaptively updated to minimize the error signal $e(k)$.

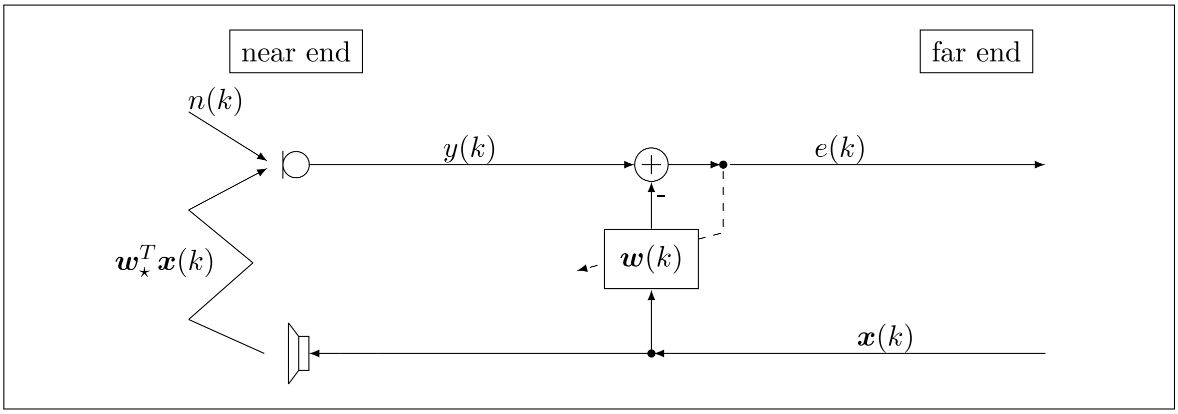


Figure 2.1: The signal model of the acoustic echo compensation task at time step k is shown.

To measure and evaluate how much of the echo is compensated, an objective evaluation criterion is needed. One suitable criterion is the system mismatch, which is the normalised squared difference between the estimated impulse response and the true impulse response between the microphone and the loudspeaker. The system mismatch in dB is defined as

$$\text{system mismatch} = 10 \log_{10} \left(\frac{\|\boldsymbol{w}_\star - \boldsymbol{w}(k)\|_2^2}{\|\boldsymbol{w}(k)\|_2^2} \right), \tag{2.4}$$

which describes the difference of the estimated impulse response $\boldsymbol{w}(k)$ with the true impulse response $\boldsymbol{w}_\star$. The Echo-to-Noise-Ratio (ENR) in dB is defined as

$$\text{ENR} = 10 \log_{10} \left(\frac{\mathrm{E}\left\{ \left(\boldsymbol{w}_\star^T \boldsymbol{x}(k) \right)^2 \right\}}{\mathrm{E}\left\{ n(k)^2 \right\}} \right), \tag{2.5}$$

which describes the noise level by computing the ratio of the energy of the echo signal $\boldsymbol{w}_\star^T \boldsymbol{x}(k)$ and the energy of the near end noise signal $n(k)$.

The LMS algorithm and the related NLMS algorithm which are discussed in the following sections, are recursive algorithms that converge towards the Wiener solution [28]. The Wiener solution is the solution to the mean square minimization of the echo cancellation problem (Wiener filter) for stationary input signals $\boldsymbol{x}(k)$ [35]. In the following, this convergence towards the Wiener solution is referred to convergence towards the optimal point. Since the input signal can have varying statistical properties and the true impulse response can change over time, the Wiener filter has limited practical applicability and adaptive filters like the LMS or the NLMS algorithm are needed to solve the echo cancellation problem [35].

2.1.2 LMS algorithm

The LMS algorithm is an iterative adaptation algorithm to estimate the impulse response $\boldsymbol{w}_\star$ by minimising the squared error e^2 over time. For this minimisation, the estimation of the impulse response $\boldsymbol{w}(k)$ is updated in the direction of the negative gradient of the expected squared error e^2, hence the name least mean square (LMS) algorithm. The gradient is computed to [35]

$$\nabla = \frac{\partial \mathrm{E}\left\{e(k)^2\right\}}{\partial \boldsymbol{w}(k)} \tag{2.6}$$

$$= \frac{\partial \mathrm{E}\left\{\left(\boldsymbol{w}_\star^T \boldsymbol{x}(k) - \boldsymbol{w}^T \boldsymbol{x}(k)\right)^2\right\}}{\partial \boldsymbol{w}(k)} \tag{2.7}$$

$$= -2\mathrm{E}\left\{e(k)\boldsymbol{x}(k)\right\}. \tag{2.8}$$

In practice, the gradient needs to be evaluated at each time step, and therefore the instantaneous gradient is used

$$\hat{\nabla} = -2e(k)\boldsymbol{x}(k), \tag{2.9}$$

which gives the adaptation rule

$$\boldsymbol{w}(k+1) = \boldsymbol{w}(k) + \beta(k)e(k)\boldsymbol{x}(k), \tag{2.10}$$

where β is a step size parameter, which controls the adaptation.

2.1.3 NLMS algorithm

The NLMS algorithm normalizes the step size in each update step by the instantaneous power estimate of the input $\boldsymbol{x}(k)$. The NLMS algorithm is based on the minimization of the Euclidean norm of the change in $\boldsymbol{w}(k)$, subject to the constraint that the error $e(k)$ is zero [36] with

$$\min_{\boldsymbol{w}(k+1)\in\mathbb{R}^l} \|\boldsymbol{w}(k+1) - \boldsymbol{w}(k)\|_2^2 \tag{2.11}$$

subject to

$$\boldsymbol{w}_\star^T \boldsymbol{x}(k) = \boldsymbol{w}(k+1)^T \boldsymbol{x}(k). \tag{2.12}$$

This minimization problem can be solved using the Lagrange multipliers method. The Lagrangian function $\mathcal{L}$ is defined as

$$\mathcal{L}(\boldsymbol{w}(k+1), \lambda) = \|\boldsymbol{w}(k+1) - \boldsymbol{w}(k)\|_2^2 + \lambda\left(\boldsymbol{w}(k+1)^T \boldsymbol{x}(k) - \boldsymbol{w}_\star^T \boldsymbol{x}(k)\right). \tag{2.13}$$

To obtain the solution of 2.11, the derivatives of the Lagrangian function with respect to $\boldsymbol{w}(k+1)$ and λ are set to zero

$$\frac{\partial \mathcal{L}(\boldsymbol{w}(k+1),\lambda)}{\partial \boldsymbol{w}(k+1)} = 2\left(\boldsymbol{w}(k+1)-\boldsymbol{w}(k)\right)^T - \lambda \boldsymbol{x}(k)^T \overset{!}{=} 0 \tag{2.14}$$

$$\frac{\partial \mathcal{L}(\boldsymbol{w}(k+1),\lambda)}{\partial \lambda} = \boldsymbol{w}(k+1)^T \boldsymbol{x}(k) - \boldsymbol{w}_\star^T \boldsymbol{x}(k) \overset{!}{=} 0 \tag{2.15}$$

Equation 2.14 gives us

$$\boldsymbol{w}(k+1)^T = \boldsymbol{w}(k)^T + \frac{\lambda}{2}\boldsymbol{x}(k)^T. \tag{2.16}$$

Multiplying both sides of 2.16 by $\boldsymbol{x}(k)$ from the right side, and using 2.15 we have

$$\boldsymbol{w}_\star^T \boldsymbol{x}(k) = \boldsymbol{w}(k)^T \boldsymbol{x}(k) + \frac{\lambda}{2}\boldsymbol{x}(k)^T \boldsymbol{x}(k), \tag{2.17}$$

which can be solved for lambda and simplified by substituting

$$e(k) = \boldsymbol{w}_\star^T \boldsymbol{x}(k) - \boldsymbol{w}(k)^T \boldsymbol{x}(k). \tag{2.18}$$

Then we have

$$\lambda = \frac{2e(k)}{\|\boldsymbol{x}(k)\|_2^2}. \tag{2.19}$$

The NLMS equation is then given as 2.16 transposed and with λ from 2.19

$$\boldsymbol{w}(k+1) = \boldsymbol{w}(k) + \frac{e(k)\boldsymbol{x}(k)}{\|\boldsymbol{x}(k)\|_2^2}. \tag{2.20}$$

To further steer the step size of the NLMS algorithm, the step size parameter is added to give the NLMS update equation

$$\boldsymbol{w}(k+1) = \boldsymbol{w}(k) + \frac{\beta e(k)\boldsymbol{x}(k)}{\|\boldsymbol{x}(k)\|_2^2}. \tag{2.21}$$

Formulas for the convergence speed and the steady state system mismatch of the LMS and NLMS algorithm in case of a white noise excitation signal are derived in the literature [35, 14].

2.1.4 Kalman filter

Since its original publication [20], the Kalman filter has been studied extensively and has been applied to many different applications. The Kalman filter is the optimal Bayesian estimator for system identification of linear systems [20]. The Kalman filter can be interpreted as a generalisation of the NLMS algorithm, where the step size is adaptively changed based on an estimate of the error covariance [28]. A comprehensive description of the Kalman filter for the application of acoustic echo compensation can be found in [6].

The assumed signal model for the Kalman filter is

$$\boldsymbol{w}_\star(k+1) = \gamma \boldsymbol{w}_\star(k) + \Delta(k), \tag{2.22}$$

where $\boldsymbol{w}_{\star}(k+1)$ is the time varying true impulse response. The process noise $\Delta(k)$ is zero mean Gaussian distributed with covariance $\Psi_{\Delta\Delta} = \sigma_{\Delta}^2 \boldsymbol{I}$, where $\boldsymbol{I}$ is the identity matrix. The fading factor γ is has typical values of $0 \ll \gamma \leq 1$ for acoustical systems [7]. If the system is non time varying, γ is set to 1 and σ_{Δ}^2 to 0, and $\boldsymbol{w}_{\star}(k)$ is written as $\boldsymbol{w}_{\star}$.

The microphone measurement $y(k)$ is given as

$$y(k) = \boldsymbol{w}_{\star}(k)^T \boldsymbol{x}(k) + n(k), \tag{2.23}$$

with the far end signal $\boldsymbol{x}(k)$ and the near end noise $n(k)$. Furthermore, the estimate of the error covariance $\boldsymbol{P}(k)$ is defined as

$$\boldsymbol{P}(k) = \mathrm{E}\left\{(\boldsymbol{w}_{\star}(k) - \boldsymbol{w}(k))(\boldsymbol{w}_{\star}(k) - \boldsymbol{w}(k))^T\right\}. \tag{2.24}$$

The a priori estimates of the impulse response and the error covariance are denoted by $\boldsymbol{w}^{+}(k)$ and $\boldsymbol{P}^{+}(k)$, respectively. The a posteriori estimates of the impulse response and the error covariance, i.e. the estimates after the update from a new measurement, are denoted by $\boldsymbol{w}(k)$ and $\boldsymbol{P}(k)$, respectively.

The Kalman filter predicts the new estimate from the assumed signal model and updates the estimation with information from new measurement in each step. The prediction step of the estimate and the error covariance are given with

$$\boldsymbol{w}^{+}(k) = \gamma \boldsymbol{w}(k), \tag{2.25}$$

and

$$\boldsymbol{P}^{+}(k) = \gamma^2 \boldsymbol{P}(k) + \Psi_{\Delta\Delta}. \tag{2.26}$$

The update equations using the new measurement are given with

$$\boldsymbol{K}(k) = \frac{\boldsymbol{P}^{+}(k)\boldsymbol{x}(k)}{\boldsymbol{x}(k)^T \boldsymbol{P}^{+}(k)\boldsymbol{x}(k) + \sigma_n^2}, \tag{2.27}$$

$$\boldsymbol{w}(k+1) = \boldsymbol{w}^{+}(k) + e(k)\boldsymbol{K}(k), \tag{2.28}$$

and

$$\boldsymbol{P}(k) = \boldsymbol{P}^{+}(k) - \boldsymbol{K}(k)\boldsymbol{x}(k)^T \boldsymbol{P}^{+}(k), \tag{2.29}$$

with the residual echo signal

$$e(k) = y(k) - \boldsymbol{w}^{+}(k)^T \boldsymbol{x}(k). \tag{2.30}$$

where $\boldsymbol{K}(k)$ is the Kalman gain, and σ_n^2 is the noise power.

There are different versions of the Kalman filter, for example the frequency domain Kalman filter and the time domain Kalman filter. They only differ slightly, and for the case of a frame shift of one (meaning that the block length is one), the frequency domain Kalman filter and the time domain Kalman filter are shown to be equivalent [19]. For simplicity, this thesis will only focus on the time domain Kalman filter.

2.1.5 Extended Kalman filter

The extended Kalman filter is a generalisation of the Kalman filter, but unlike the Kalman filter it is not optimal. This is because the extended Kalman filter approximates the Kalman filters states and observations with a linearisation [29, 21]. The extended Kalman filter is not guaranteed to converge in general, however under some conditions, convergence can be guaranteed towards a local minimum [23, 34].

The extended Kalman filter is presented here for the case of a linear signal model with only a non linear observation model, since only a non linear observation model is necessary for the later derived manifold Kalman filter in Chapter 3. The assumed signal model for the extended Kalman filter is

$$\boldsymbol{w}_\star(k+1) = \gamma \boldsymbol{w}_\star(k) + \Delta(k), \tag{2.31}$$

where $\boldsymbol{w}_\star(k+1)$ is the time varying true impulse response. The process noise $\Delta(k)$ is zero mean Gaussian distributed with covariance $\Psi_{\Delta\Delta} = \sigma_\Delta^2 \boldsymbol{I}$, where $\boldsymbol{I}$ is the identity matrix. The fading factor γ is has typical values of $0 \ll \gamma \leq 1$ for acoustical systems [7]. If the system is non time varying, γ is set to 1 and σ_Δ^2 to 0, and $\boldsymbol{w}_\star(k)$ is written as $\boldsymbol{w}_\star$.

The microphone measurement $y(k)$ is given as

$$y(k) = h\left(\boldsymbol{w}_\star(k)\right) + n(k), \tag{2.32}$$

with the non linear observation function h and the near end noise $n(k)$. Furthermore, the estimate of the error covariance $\boldsymbol{P}(k)$ is defined as

$$\boldsymbol{P}(k) = \mathrm{E}\left\{(\boldsymbol{w}_\star(k) - \boldsymbol{w}(k))(\boldsymbol{w}_\star(k) - \boldsymbol{w}(k))^T\right\}. \tag{2.33}$$

The a priori estimates of the impulse response and the error covariance are denoted by $\boldsymbol{w}^+(k)$ and $\boldsymbol{P}^+(k)$, respectively. The a posteriori estimates of the impulse response and the error covariance, i.e. the estimates after the update from a new measurement, are denoted by $\boldsymbol{w}(k)$ and $\boldsymbol{P}(k)$, respectively.

The Kalman filter predicts the new estimate from the assumed signal model and updates the estimation with information from new measurement in each step. The prediction step of the estimate and the error covariance are given with

$$\boldsymbol{w}^+(k) = \gamma \boldsymbol{w}(k), \tag{2.34}$$

and

$$\boldsymbol{P}^+(k) = \gamma^2 \boldsymbol{P}(k) + \Psi_{\Delta\Delta}. \tag{2.35}$$

The update equations using the new measurement are given with

$$\boldsymbol{K}(k) = \frac{\boldsymbol{P}^+(k)\boldsymbol{H}(k)^T}{\boldsymbol{H}(k)\boldsymbol{P}^+(k)\boldsymbol{H}(k)^T + \sigma_n^2}, \tag{2.36}$$

$$\boldsymbol{w}(k+1) = \boldsymbol{w}^+(k) + e(k)\boldsymbol{K}(k), \tag{2.37}$$

and

$$\boldsymbol{P}(k) = \boldsymbol{P}^{+}(k) - \boldsymbol{K}(k)\boldsymbol{x}(k)^T\boldsymbol{P}^{+}(k), \tag{2.38}$$

where $\boldsymbol{K}(k)$ is the Kalman gain, σ_n^2 is the noise power, and

$$\boldsymbol{H}(k) = \frac{\partial h\,(\omega)}{\partial \omega}\Big|_{\omega=\boldsymbol{w}^{+}(k)} \tag{2.39}$$

is the Jacobian of the non linear observation function h, and the residual echo signal $e(k)$ is given as

$$e(k) = y(k) - h\left(\boldsymbol{w}^{+}(k)\right). \tag{2.40}$$

2.2 Acoustic Manifolds

In the context of machine learning, when some training data can be mapped to a lower dimensional subspace this subspace is usually referred to as a manifold [9]. In a more mathematical description, a manifold is a connected region, which is locally Euclidean [9]. A simple example of a two dimensional manifold in the three dimensional space is the surface of a sphere. The underlying assumption to use manifolds is that all high dimensional data that contain meaningful information can be represented by some generally non linear mapping with a smaller number of variables. This is commonly referred to as the manifold hypotheses [9].

One justification to use lower dimensional manifolds is given by analysing the time independent wave equation (also called the Helmholtz equation) of the sound field of a room. There it can be shown that an impulse responses between a loudspeaker and a microphone inside a room can be represented in good approximation by a low rank model, i.e. a lower dimensional model [17].

By modelling room impulse responses with a kernel Principal-Component-Analysis (PCA) model, diffusion maps can be created, which show that impulse responses in a room lie locally on a manifold [26]. These diffusion maps provide one possible way to construct a non linear manifold [26].

Several other approaches on how to construct a manifold have been proposed in the literature. One of the most general approach uses a deep neural network with an autoencoder to learn a non linear encoder function $\boldsymbol{f}_{\text{enc}}$ and a non linear decoder function $\boldsymbol{f}_{\text{dec}}$ to map from the original representation of the impulse response in the time domain to the manifold with the encoder and back with the decoder to the time domain [9].

A more simple class of manifolds are the affine or piecewise affine manifolds. Whereas the optimal affine manifold can be straightforwardly computed by a PCA (global PCA manifold), the piecewise affine manifold needs to group the training data into local subgroups and compute the PCA for each of them (local PCA manifold). The PCA gives the optimal affine manifold for a l' dimensional linear manifold [16]. This means that the PCA gives the best affine predictor of the random variable $\boldsymbol{x}$ by its first l' principal components in the sense that the sum of the residual variances of each element of $\boldsymbol{x}$ is minimized [16]. Furthermore, the mapping from the variables to the principal components is defined to be an orthogonal mapping, which means that the covariance matrix of the principal components is diagonalized [16].

2.3 Related Approaches

In the following section, related approaches in the literature for the acoustic echo compensation with acoustic manifolds are described. Most approaches use global or local subspaces found by PCA, where global PCA means that all available training data is used to compute a PCA, which resembles an affine manifold. Local PCA means that the available training data is split up into local neighbourhoods to compute a PCA for each of them, which resembles a piecewise affine (i.e. a non linear) manifold. In this thesis, a PCA manifold always means a global affine subspace found by PCA, unless stated otherwise.

Some examples of using these manifolds include the use of a local and global subspace found by PCA to increase the performance of general online system identification algorithms by doing the normal update step of the algorithm, which is then projected onto the manifold [12]. This manifold projection method is further developed and applied to a Kalman filter to significantly increase the echo compensation capabilities compared to the classical Kalman filter [13]. In a different application of supervised system identification, local and global subspace models are used to improve the mean square error of the estimation of a loudspeaker position [22]. Another approach involves using a PCA subspace model as basis vectors for the NLMS algorithm, with the convergence speed of the NLMS algorithm being increased by weighting each principal component by the magnitude of the current estimate of each principal component to increase the system identification performance [10]. For multichannel echo compensation problems, one challenge is the non uniqueness problem, which makes it challenging to compensate the echo, especially for correlated input signals. A global PCA is used to learn the principle components of the possible echo paths, which is demonstrated to improve the performance of the echo compensation especially for the case of correlated input signals [8].

Other methods include promoting a low rank structure of the estimate of an impulse response, i.e. promoting a smaller number of components to model the impulse responses by a regularization term, to increase the accuracy of the estimate of the impulse response [18]. In the so called common acoustical pole and zero model, which is a modified pole zero model, the number of parameters is halved, which is shown to increase the convergence speed of the system identification task compared to previous methods [11]. For the case that periodic perfect sequences are used for the system identification task, a change of basis vectors, which consider the use of a periodic perfect sequences, can increase the convergence speed of the NLMS algorithm [4]. Finally, for the estimation of non linear systems, kernel methods have been suggested as a potential solution. One of such methods is the kernel Kalman filter proposed for the application of non linear system identification for echo compensation, which uses a kernel, based on a dictionary learned from training data of impulse responses [30].

Chapter 3

Proposed Methods

This chapter presents the proposed manifold LMS algorithm, the manifold NLMS and the manifold Kalman filter. First, the manifold LMS algorithm and the LMS algorithm are formulated to do the estimation on the latent variables of a manifold with an encoder and decoder function. For the manifold NLMS algorithm the optimal step size is derived by using the analogous minimization problem formulation as it is done for the NLMS algorithm. The manifold Kalman filter is shown to be a variant of the extended Kalman filter, where the time domain is the non linear observation space and the parameter updates are done on the latent variables of the manifold. Finally, the manifold LMS algorithm is analysed theoretically for its convergence speed and its theoretically steady state value for the case of white noise excitation. The derived values of the convergence speed and the steady state system mismatch are compared to simulations of the algorithms in Chapter 4.

3.1 System Identification on Manifolds

The system model for the system identification was fully described in Chapter 2, and is briefly recapped here. The input signal from the far end $\boldsymbol{x}(k)$ is emitted by the loudspeaker and the microphone signal $y(k)$ is given as

$$y(k) = \boldsymbol{w}_\star^T \boldsymbol{x}(k) + n(k), \tag{3.1}$$

where $\boldsymbol{w}_\star$ is the true room impulse response and $n(k)$ is an additive noise term. The estimated echo signal is subtracted from the microphone signal to get the residual echo (or error) signal $e(k)$ with

$$e(k) = y(k) - \boldsymbol{w}(k)^T \boldsymbol{x}(k), \tag{3.2}$$

where $\boldsymbol{w}(k)$ is the estimated impulse response at time step k.

For the manifold system identification, the latent variables of the manifold $\boldsymbol{z}(k) \in \mathbb{R}^{l'}$ are tracked and estimated instead of the impulse response in the time domain $\boldsymbol{w}(k) \in \mathbb{R}^{l}$, where l is the number of time domain components of the impulse response and l' is the number of latent variables. Since the main idea is to speed up the convergence by reducing the number of variables, it is always assumed that $l \geq l'$. In order to map between these two representations, an encoder function to map from the time domain to the manifold domain and a decoder function to map from the manifold domain to the time domain are defined. The encoder is defined as

$$\boldsymbol{f}_{\text{enc}} : \mathbb{R}^{l} \to \mathbb{R}^{l'} : \boldsymbol{x} \mapsto \boldsymbol{f}_{\text{enc}}(\boldsymbol{x}), \tag{3.3}$$

and the decoder defined as

$$\boldsymbol{f}_{\text{dec}} : \mathbb{R}^{l'} \to \mathcal{M} \in \mathbb{R}^{l} : \boldsymbol{x} \mapsto \boldsymbol{f}_{\text{dec}}(\boldsymbol{x}), \tag{3.4}$$

where $\mathcal{M}$ is the manifold, which lies in $\mathbb{R}^l$. This means that the relationship between $\boldsymbol{z}(k)$ and $\boldsymbol{w}(k)$ can be written as

$$\boldsymbol{z}(k) = \boldsymbol{f}_{\text{enc}}(\boldsymbol{w}(k)), \tag{3.5}$$

and

$$\boldsymbol{w}(k) = \boldsymbol{f}_{\text{dec}}(\boldsymbol{z}(k)). \tag{3.6}$$

With these definitions of the manifold, the system model of the residual echo signal $e(k)$ is given as

$$e(k) = y(k) - \boldsymbol{f}_{\text{dec}}^{T}(\boldsymbol{z}(k))\boldsymbol{x}(k), \tag{3.7}$$

where the microphone signal $y(k)$ is the same as before in equation (3.1) and $\boldsymbol{z}(k)$ is the vector of latent variables of the manifold to estimate the impulse response.

3.2 Manifold LMS Algorithm

For deriving the manifold LMS algorithm, first a version is derived, where the latent variables of the manifold are updated, which is called the manifold LMS algorithm. A second version is derived, where the manifold LMS algorithm is linearised at the current estimate of the manifold, to do the estimation in the time domain, which is called the linearised manifold LMS algorithm.

3.2.1 Derivation of the manifold LMS algorithm

The following algorithm derivation follows the same steps as the derivation of the classical LMS algorithm in Section (2.1.2). The difference is that the update of the estimate of the impulse response is done on the latent variable $\boldsymbol{z}(k)$ instead of variable in the full space $\boldsymbol{w}(k)$. We therefore call this algorithm the manifold LMS algorithm. This means that the gradient needs to be calculated with respect to $\boldsymbol{z}_k$. Furthermore, to calculate the residual echo signal $e(k)$, the latent variable $\boldsymbol{z}(k)$ at time step k is mapped back to the full space $\boldsymbol{w}(k) = \boldsymbol{f}_{\text{dec}}(\boldsymbol{z}(k))$ after each update step. The error signal was given in equation 3.7, and the gradient of the squared error signal $e(k)$ is given as

$$\nabla = \frac{\partial \mathrm{E}\left\{e(k)^2\right\}}{\partial \boldsymbol{z}(k)} \tag{3.8}$$

$$= \frac{\partial \mathrm{E}\left\{\left(y(k) - \boldsymbol{f}_{\text{dec}}(\boldsymbol{z}(k))^{T}\boldsymbol{x}(k)\right)^2\right\}}{\partial \boldsymbol{z}(k)} \tag{3.9}$$

$$= -2\mathrm{E}\left\{\left.\frac{\partial \boldsymbol{f}_{\text{dec}}(\boldsymbol{\zeta})}{\partial \boldsymbol{\zeta}}\right|_{\boldsymbol{\zeta}=\boldsymbol{z}(k)}^{T} \boldsymbol{x}(k)e(k)\right\} \tag{3.10}$$

In practice the derivative needs to be evaluated at each time step, and therefore the instantaneous derivative is used. For simpler notation, the Jacobian $\boldsymbol{J}(k)$ at time step is used with

$$\left.\frac{\partial \boldsymbol{f}_{\mathrm{dec}}(\boldsymbol{\zeta})}{\partial \boldsymbol{\zeta}}\right|_{\boldsymbol{\zeta}=\boldsymbol{z}(k)} = \boldsymbol{J}(k), \tag{3.11}$$

and

$$\hat{\nabla} = -2\boldsymbol{J}(k)^T \boldsymbol{x}(k)e(k) \tag{3.12}$$

is obtained, which gives the adaptation rule

$$\boldsymbol{z}(k+1) = \boldsymbol{z}(k) + \beta \boldsymbol{J}(k)\boldsymbol{x}(k)e(k). \tag{3.13}$$

The step size parameter β, which is called the full step size in the following, is constant over time in case of the LMS algorithm. Upper and lower bounds for a guaranteed convergence of the manifold LMS algorithm are discussed in Section 3.5.3

The impulse response $\boldsymbol{w}$ at each time step is calculated by

$$\boldsymbol{w}(k+1) = \boldsymbol{f}_{\mathrm{dec}}(\boldsymbol{z}(k+1)) \tag{3.14}$$

3.2.2 Linearised manifold LMS algorithm

In the previous section the manifold LMS algorithm was derived, where the update of the impulse response is made in the latent space. However, the update equations can also directly be formulated in the full space by linearising the manifold LMS algorithm at the current point $\boldsymbol{z}(k)$ in the latent space. This linearised version will be called the linearised manifold LMS algorithm. Linearising the decoder at $\boldsymbol{z}(k)$ gives the equation

$$\boldsymbol{w}(k+1) = \boldsymbol{f}_{\mathrm{dec}}(\boldsymbol{z}(k+1)) \approx \boldsymbol{f}_{\mathrm{dec}}(\boldsymbol{z}(k)) + \boldsymbol{J}(k)^T \left(\boldsymbol{z}(k+1) - \boldsymbol{z}(k)\right). \tag{3.15}$$

From (3.13) we have

$$\boldsymbol{z}(k+1) - \boldsymbol{z}(k) = \beta \boldsymbol{J}(k)^T \boldsymbol{x}(k)e(k), \tag{3.16}$$

and with $\boldsymbol{w}(k) = \boldsymbol{f}_{\mathrm{dec}}(\boldsymbol{z}(k))$, the update equation can be calculated by

$$\boldsymbol{w}(k+1) = \boldsymbol{w}(k) + \beta \boldsymbol{J}(k)\boldsymbol{J}(k)^T \boldsymbol{x}(k)e(k), \tag{3.17}$$

which is called the linearised manifold LMS algorithm. Since the update is done in the full space, $e(k)$ is obtained by

$$e(k) = y(k) - \boldsymbol{w}(k)^T \boldsymbol{x}(k). \tag{3.18}$$

3.3 Manifold NLMS Algorithm

In this section, the manifold NLMS algorithm is derived, and the update equations of the linearised manifold NLMS algorithm are given, which is analogous to the linearised manifold LMS algorithm, for which the estimate of the impulse response is done in the time domain.

A detailed derivation of the manifold NLMS algorithm can be found in Appendix A, where the optimal full step size of the manifold NLMS algorithm is shown to be

$$\beta = \frac{\alpha}{\|\boldsymbol{J}(k)^T \boldsymbol{x}(k)\|_2^2}, \tag{3.19}$$

and the update equation of the manifold NLMS algorithm is shown to be

$$\boldsymbol{z}(k+1) = \boldsymbol{z}(k) + \alpha \frac{\boldsymbol{J}(k)^T \boldsymbol{x}(k) e(k)}{\|\boldsymbol{J}(k)^T \boldsymbol{x}(k)\|_2^2}. \tag{3.20}$$

Analogous to the linearised manifold LMS algorithm, the update equation for the manifold NLMS algorithm can also be formulated in the time domain by linearising the decoder at $\boldsymbol{z}(k)$. The derivation steps are skipped here, since they are exactly analogous to derivation the linearised manifold LMS algorithm from Section 3.2.2. The update equation for the linearised manifold NLMS algorithm is given as

$$\boldsymbol{w}(k+1) = \boldsymbol{w}(k) + \alpha \frac{\boldsymbol{J}(k)\boldsymbol{J}(k)^T \boldsymbol{x}(k) e(k)}{\|\boldsymbol{J}(k)^T \boldsymbol{x}(k)\|_2^2}. \tag{3.21}$$

3.4 Manifold Kalman Filter

The Kalman filter and the extended Kalman filter were described in Chapter 2. Using the Kalman filter to update the estimation of an impulse response on a latent variable on a manifold can be formulated with the update equations of the extended Kalman filter.

3.4.1 The manifold Kalman filter as an extended Kalman filter

For the manifold Kalman filter, the extended Kalman filter update equations are used with a non linear observation model and a linear prediction model. The non linear observation model is needed to be able to map from the observation of the residual echo signal in the time domain to the non linear manifold. The prediction step of the latent variables of the manifold Kalman filter can be defined in many ways, but in order to be able to compare the manifold Kalman filter to the time domain Kalman filter, the prediction step of the manifold Kalman filter is defined such that the prediction step of the manifold Kalman filter directly corresponds to the prediction step from the time domain Kalman filter.

The manifold Kalman filter defined in the way to directly correspond to the time domain Kalman filter, using the extended Kalman filter is derived in Appendix B. The assumed signal model for the manifold Kalman filter is

$$\boldsymbol{w}_\star(k+1) = \gamma \boldsymbol{w}_\star(k) + \Delta(k), \tag{3.22}$$

where $\boldsymbol{w}_\star(k+1)$ is the time varying true impulse response. The process noise $\Delta(k)$ is zero mean Gaussian distributed with covariance $\Psi_{\Delta\Delta} = \sigma_\Delta^2 \boldsymbol{I}$, where $\boldsymbol{I}$ is the identity matrix. In Appendix B it is shown that the corresponding signal model for the latent variables to the assumed signal model is given as

$$\boldsymbol{z}_\star(k+1) = \boldsymbol{z}_\star(k) + \boldsymbol{J}(k)^T \Delta(k) + (\gamma - 1)\boldsymbol{J}(k)^T \boldsymbol{f}_{\mathrm{dec}}(\boldsymbol{z}_\star(k)), \tag{3.23}$$

where $\boldsymbol{z}_\star(k+1) = \boldsymbol{f}_{\text{enc}}\left(\boldsymbol{w}_\star(k+1)\right)$ are the time varying latent variables of the true impulse response, and $\boldsymbol{J}(k)^T$ is the Jacobian of the decoder.

The microphone measurement $y(k)$ is given as

$$y(k) = \boldsymbol{f}_{\text{dec}}\left(\boldsymbol{z}_\star(k)\right)^T \boldsymbol{x}(k) + n(k), \tag{3.24}$$

with the far end signal $\boldsymbol{x}(k)$ and the near end noise $n(k)$. Furthermore, the estimate of the error covariance $\boldsymbol{P}(k)$ is defined as

$$\boldsymbol{P}(k) = \mathrm{E}\left\{(\boldsymbol{z}_\star(k) - \boldsymbol{z}(k))(\boldsymbol{z}_\star(k) - \boldsymbol{z}(k))^T\right\}. \tag{3.25}$$

The a priori estimates of the impulse response and the error covariance are denoted by $\boldsymbol{z}^+(k)$ and $\boldsymbol{P}^+(k)$, respectively.The a posterior estimates of the impulse response and the error covariance, i.e. the estimates after the update from a new measurement, are denoted by $\boldsymbol{z}(k)$ and $\boldsymbol{P}(k)$, respectively.

The Kalman filter predicts the new estimate from the assumed signal model and updates the estimation with information from new measurement in each step. The prediction step of the estimate and the error covariance are given with

$$\boldsymbol{z}^+(k) = \boldsymbol{z}(k) + (\gamma - 1)\boldsymbol{J}(k)^T \boldsymbol{f}_{\text{dec}}(\boldsymbol{z}(k)), \tag{3.26}$$

and

$$\boldsymbol{P}^+(k) = \gamma^2 \boldsymbol{P}(k) + \boldsymbol{J}(k)^T \Psi_{\Delta\Delta} \boldsymbol{J}(k), \tag{3.27}$$

The update equations using the new measurement are given with

$$\boldsymbol{K}(k) = \frac{\boldsymbol{P}^+(k)\boldsymbol{H}(k)^T}{\boldsymbol{H}(k)\boldsymbol{P}^+(k)\boldsymbol{H}(k)^T + \sigma_n^2}, \tag{3.28}$$

$$\boldsymbol{z}(k+1) = \boldsymbol{z}^+(k) + e(k)\boldsymbol{K}(k), \tag{3.29}$$

and

$$\boldsymbol{P}(k) = \boldsymbol{P}^+(k) - \boldsymbol{K}(k)\boldsymbol{H}(k)\boldsymbol{P}^+(k), \tag{3.30}$$

with

$$\boldsymbol{H}(k) = \frac{\partial \boldsymbol{f}_{\text{dec}}(\boldsymbol{\zeta})^T \boldsymbol{x}(k)}{\partial \boldsymbol{\zeta}}\bigg|_{\boldsymbol{\zeta}=\boldsymbol{z}(k)} = \left(\boldsymbol{J}(k)^T \boldsymbol{x}(k)\right)^T = \boldsymbol{x}(k)^T \boldsymbol{J}(k), \tag{3.31}$$

where $\boldsymbol{K}(k)$ is the Kalman gain, σ_n^2 is the noise power, and the residual echo signal $e(k)$ is given as

$$e(k) = y(k) - \boldsymbol{f}_{\text{dec}}\left(\boldsymbol{z}^+(k)\right)^T \boldsymbol{x}(k). \tag{3.32}$$

Finally, to get the estimate of the impulse response in the time domain, its representation in latent space needs to be decoded with

$$\boldsymbol{w}(k+1) = \boldsymbol{f}_{\text{dec}}(\boldsymbol{z}(k+1)). \tag{3.33}$$

3.4.2 Linearised manifold Kalman filter

As done before for the manifold NLMS algorithm, the update equation of the manifold Kalman filter can be done in the full space by linearising the decoder at $\boldsymbol{z}^{+}(k)$ with

$$\boldsymbol{w}(k+1) = \boldsymbol{f}_{\text{dec}}\left(\boldsymbol{z}(k+1)\right) \approx \boldsymbol{f}_{\text{dec}}\left(\boldsymbol{z}^{+}(k)\right) + \boldsymbol{J}(k)\left(\boldsymbol{z}(k+1) - \boldsymbol{z}^{+}(k)\right) \tag{3.34}$$

$$= \boldsymbol{w}^{+}(k) + e(k)\boldsymbol{J}(k)\boldsymbol{K}(k), \tag{3.35}$$

where (3.29) is used to calculate (3.35). For the linearised manifold Kalman filter the error covariance $\boldsymbol{P}(k)$ is in the latent space, meaning that only the prediction and update step of the state are changed to the time domain. The signal model of the residual echo signal and the prediction and update equations of the linearised manifold Kalman filter are given as follows

The assumed signal model is given for the full space with

$$\boldsymbol{w}_{\star}(k+1) = \gamma\boldsymbol{w}_{\star}(k) + \Delta(k), \tag{3.36}$$

with the microphone measurement

$$y(k) = \boldsymbol{w}_{\star}(k)^{T}\boldsymbol{x}(k) + n(k). \tag{3.37}$$

The prediction step for the linearised manifold Kalman filter is then

$$\boldsymbol{w}^{+}(k) = \gamma\boldsymbol{w}(k) + \boldsymbol{u}(k), \tag{3.38}$$

and the update step for the estimate of the impulse response is given with

$$\boldsymbol{w}(k+1) = \boldsymbol{w}^{+}(k) + e(k)\boldsymbol{J}(k)\boldsymbol{K}(k), \tag{3.39}$$

with the residual echo signal

$$e(k) = y(k) - \boldsymbol{w}^{+}(k)^{T}\boldsymbol{x}(k). \tag{3.40}$$

For the linearised manifold Kalman filter the same equations for the prediction step of $\boldsymbol{P}^{+}(k)$ (3.27), the update steps of $\boldsymbol{P}(k)$ (3.30), and the Kalman gain $\boldsymbol{K}(k)$ (3.28) are used, meaning that the matrix $\boldsymbol{P}(k)$ is in the dimensions of the latent space.

3.5 Theoretical Analysis of the manifold LMS and NLMS algorithm

In this section, first the advantages of orthogonal affine manifold are discussed and it is explained why this thesis focuses in the simulation part on orthogonal affine manifold. For the class of orthogonal affine manifolds then several analysis are carried out. It is shown that the manifold LMS algorithm and the linearised manifold LMS algorithm are equivalent in the case that an affine manifold is used. Furthermore, the convergence conditions of the manifold LMS and NLMS algorithms, the derivation of the theoretically convergence speed of the manifold LMS algorithm, and the theoretically steady state system mismatch of the manifold LMS algorithm are given.

3.5.1 Orthogonal affine manifolds

This section first discusses the advantages and disadvantages of affine manifolds in detail, defines the mapping of affine manifolds, and discusses the PCA as optimal affine manifold, which is orthogonal. Given the orthogonality, some analysis is done regarding the projection onto the manifold, and at last it is shown that the manifold LMS and NLMS algorithm, and the linearised manifold LMS and NLMS algorithm are each equivalent for the case of an affine manifold.

The acoustic manifold was defined by an encoder function that maps the time domain onto the manifold, and a decoder function that maps the manifold back to the time domain. An autoencoder trained by a neural network can learn any mapping function for the encoder and the decoder, meaning that in general, the learned encoder and decoder are non linear. The LMS algorithm, the NLMS algorithm and the Kalman filter are gradient based algorithms that converge towards the global minimum of the mean square error function, which is a convex function, if certain step size conditions are met [35, 14, 25]. As mentioned, the encoder function and the decoder function are non linear in general, meaning that the objective function to minimize for the manifold LMS algorithm, manifold NLMS algorithm, and the manifold Kalman filter is not convex in general. This means that the convergence towards the global minimum cannot be guaranteed. The condition to ensure a convex manifold loss function, which is the composition of the mean square loss and the decoder, is given if the decoder is either convex and non decreasing or concave and non increasing, assuming that the decoder is twice differentiable [3]. One simple example to fulfil the condition of the convexity of the manifold loss function is given with an affine encoder and decoder function [3, 2].

By choosing the encoder and decoder function, a trade off between the amount of information that the manifold can encode with each latent variable, and the ability to efficiently minimize the manifold loss function can be made. An infinitely powerful encoder and decoder can in theory encode all information in only one variable, but is nearly impossible to minimise except by brute force [9]. On the other hand, an affine manifold is a rather simple manifold, which is limited in its capabilities to encode a large amount of information in each latent variable, but an affine manifold guarantees convergence towards the global optimum because its manifold loss function is guaranteed to be convex.

This work focuses on the class of affine manifolds to ensure the convergence towards the global optimum and to make a theoretical analysis of the manifold LMS algorithm feasible. Given the fact that affine manifolds are analysed, the optimal affine manifold can be easily learned by the PCA, which is furthermore orthogonal, as already stated in Section 2.2. It is left open for future work to fully investigate the general case of non linear manifolds.

For the theoretical analysis only affine orthogonal manifolds learned by a PCA are considered as discussed above. This brings the simplification, that the encoder and decoder functions are constant for all $\boldsymbol{z}(k)$ and $\boldsymbol{w}(k)$, and the relationship between $\boldsymbol{z}(k)$ and $\boldsymbol{w}(k)$ for an orthogonal affine manifold with the encoder and decoder can be written as

$$\boldsymbol{z}(k) = \boldsymbol{f}_{\text{enc}}(\boldsymbol{w}(k)) = \boldsymbol{J}^T \left(\boldsymbol{w}(k) - \bar{\boldsymbol{w}}\right), \tag{3.41}$$

and

$$\boldsymbol{w}(k) = \boldsymbol{f}_{\text{dec}}(\boldsymbol{z}(k)) = \boldsymbol{J}\boldsymbol{z}(k) + \bar{\boldsymbol{w}}, \tag{3.42}$$

with $l' \leq l$, and $\boldsymbol{J} \in \mathbb{R}^{l' \times l}$, which means that $\boldsymbol{J}$ has a rank of l', also meaning that the decoder spans the subspace range $\left(\boldsymbol{J}\boldsymbol{J}^T\right) \in \mathbb{R}^l$.

Furthermore, the derivative of the manifold $\frac{\partial \boldsymbol{f}_{\text{dec}}(\boldsymbol{\zeta})}{\partial \boldsymbol{\zeta}}\Big|_{\boldsymbol{\zeta}=\boldsymbol{z}(k)}$ is also constant for all $\boldsymbol{z}(k)$ with

$$\frac{\partial \boldsymbol{f}_{\text{dec}}(\boldsymbol{\zeta})}{\partial \boldsymbol{\zeta}}\Big|_{\boldsymbol{\zeta}=\boldsymbol{z}(k)} = \begin{bmatrix} \frac{\partial \boldsymbol{f}_{\text{dec}1}}{\partial z_1} & \dots & \frac{\partial \boldsymbol{f}_{\text{dec}1}}{\partial z_{l'}} \\ \vdots & \ddots & \vdots \\ \frac{\partial \boldsymbol{f}_{\text{dec}l}}{\partial z_1} & \dots & \frac{\partial \boldsymbol{f}_{\text{dec}l}}{\partial z_{l'}} \end{bmatrix} = \boldsymbol{J} \tag{3.43}$$

with

$$\boldsymbol{J} = \begin{bmatrix} \boldsymbol{j}_1 & \dots & \boldsymbol{j}_{l'} \end{bmatrix} = \begin{bmatrix} j_{11} & \dots & j_{1l'} \\ \vdots & \ddots & \vdots \\ j_{l1} & \dots & j_{ll'} \end{bmatrix} \tag{3.44}$$

Lastly, as discussed before, the PCA gives us the optimal affine manifold, which has the further benefit that $\boldsymbol{J}$ is orthogonal, meaning that

$$\boldsymbol{j}_m^T \boldsymbol{j}_n = \begin{cases} 0 \text{ for } m \neq n \\ 1 \text{ for } m = n \end{cases} \tag{3.45}$$

The orthogonality brings the advantage that the inverse of the decoder with the orthogonal matrix $\boldsymbol{J}$ for the encoder is just the transpose $\boldsymbol{J}^T$, whereas in the general case of an affine manifold, a pseudo inverse would have to be defined. Furthermore, orthogonality of $\boldsymbol{J}$ brings the advantage that $\boldsymbol{J}\boldsymbol{J}^T$ is an orthogonal projection matrix. This means that any point, after removing the mean of the affine subspace, is projected onto the subspace by a multiplication of $\boldsymbol{J}\boldsymbol{J}^T$. The conditions that $\boldsymbol{J}\boldsymbol{J}^T$ is a projection matrix can be verified by

$$\boldsymbol{J}\boldsymbol{J}^T\boldsymbol{J}\boldsymbol{J}^T = \boldsymbol{J}\boldsymbol{I}_{l'}\boldsymbol{J}^T = \boldsymbol{J}\boldsymbol{J}^T, \tag{3.46}$$

meaning that $\boldsymbol{J}\boldsymbol{J}^T$ is idempotent, and

$$\left(\boldsymbol{J}\boldsymbol{J}^T\boldsymbol{x}\right)^T \boldsymbol{x} = \boldsymbol{x}^T \left(\boldsymbol{J}\boldsymbol{J}^T\right)^T \boldsymbol{x} = \boldsymbol{x}^T \left(\boldsymbol{J}^T\right)^T (\boldsymbol{J})^T \boldsymbol{x} = \boldsymbol{x}^T\boldsymbol{J}\boldsymbol{J}^T\boldsymbol{x}, \tag{3.47}$$

meaning that $\boldsymbol{J}\boldsymbol{J}^T$ is self-adjoint [35]. The identity matrix with l' rows and columns is denoted here by $\boldsymbol{I}_{l'}$.

3.5.2 Comparison of the manifold LMS algorithm and the linearised manifold LMS algorithm

In general, the update rules of the manifold LMS algorithm (3.13) and the linearised manifold LMS algorithm (3.17) yield different results. However, in the case of an affine manifold, the manifold LMS algorithm and the linearised manifold LMS algorithm are equivalent, which is shown in the following. For an affine manifold, equation (3.42) applies, meaning that $\boldsymbol{J}(k)$, given by

$$\frac{\partial \boldsymbol{f}_{\text{dec}}(\boldsymbol{\zeta})}{\partial \boldsymbol{\zeta}}\Big|_{\boldsymbol{\zeta}=\boldsymbol{z}(k)} = \boldsymbol{J}, \tag{3.48}$$

is constant and independent of $\boldsymbol{z}(k)$. This simplifies the update equation (3.13) to

$$\boldsymbol{z}(k+1) = \boldsymbol{z}(k) + \beta\boldsymbol{J}^T\boldsymbol{x}(k)e(k). \tag{3.49}$$

By using (3.42) to write (3.49) in terms of $\boldsymbol{w}(k+1)$ with the decoder, it follows that

$$\begin{aligned}\boldsymbol{w}(k+1) &= \boldsymbol{f}_{\mathrm{dec}}(\boldsymbol{z}(k+1)) && (3.50)\\ &= \boldsymbol{J}\boldsymbol{z}(k+1)+\boldsymbol{z}_0 && (3.51)\\ &= \boldsymbol{J}\boldsymbol{z}(k)+\boldsymbol{z}_0+\beta\boldsymbol{J}\boldsymbol{J}^T\boldsymbol{x}(k)e(k) && (3.52)\\ &= \boldsymbol{w}(k)+\beta\boldsymbol{J}\boldsymbol{J}^T\boldsymbol{x}(k)e(k), && (3.53)\end{aligned}$$

which is equivalent to the update formula of the linearised manifold LMS algorithm update equation (3.17) for a constant $\boldsymbol{J}$.

3.5.3 Convergence conditions of the manifold LMS and NLMS algorithms

First, the criteria for the convergence of the manifold LMS algorithm is discussed and at the end it is explained why the same reasoning can be applied to the manifold NLMS algorithm

Regarding the convergence of the manifold LMS algorithm in the general case, no statement can be made about the convergence to or near the global optimum. The convergence criteria for the manifold LMS algorithm can be derived by applying the corresponding convergence criteria from the LMS algorithm to the manifold LMS algorithm [35]. The LMS algorithm is formulated as a minimization of the convex function

$$\mathrm{E}\left\{e(k)^2\right\} = \mathrm{E}\left\{\left(y(k)-\boldsymbol{w}(k)^T\boldsymbol{x}(k)\right)^2\right\}, \qquad (3.54)$$

which is guaranteed to converge towards the global optimum if the step size of the LMS algorithm is sufficiently small [14]. Recall from Section 2.1.2 that the LMS algorithm converges for

$$0<\beta<\frac{2}{\lambda_{\max}\left(R_{\boldsymbol{xx}}\right)}, \qquad (3.55)$$

where $\lambda_i\left(R_{\boldsymbol{xx}}\right)$ denotes the i-th eigenvalue, and $\lambda_{\max}\left(R_{\boldsymbol{xx}}\right)$ denotes the maximum eigenvalue of the input correlation matrix $R_{\boldsymbol{xx}}$ [35].

However, the manifold LMS algorithm is formulated as the minimization of the function

$$\mathrm{E}\left\{e(k)^2\right\} = \mathrm{E}\left\{\left(y(k)-\boldsymbol{f}_{\mathrm{dec}}(\boldsymbol{z}(k))^T\boldsymbol{x}(k)\right)^2\right\}, \qquad (3.56)$$

which is not convex in general, meaning that there is no guarantee for convergence near the global optimum [3]. If $\boldsymbol{f}_{\mathrm{dec}}$ and $\boldsymbol{f}_{\mathrm{enc}}$ are affine functions, (3.56) is convex, because the composition of a convex function and an affine function is convex [3]. It is reminded here that just requiring $\boldsymbol{f}_{\mathrm{dec}}$ to be convex is not sufficient to make (3.56) convex, as the composition of two convex functions is not necessarily convex [3]. For these reasons, only the case of an affine manifold will be considered in the following section to analyse the convergence and to give a condition for the convergence.

Since we have shown in Section 3.5.2 that the manifold LMS and the linearised manifold LMS are identical in the case of an affine manifold, we will analyse the convergence properties only for the linearised manifold LMS. For an affine manifold the function to minimize (3.56) is convex, and thus the manifold LMS algorithm will converge to the global optimum under the same conditions and assumptions as the LMS algorithm given in [14]. Therefore, only the upper bound for the step size of the manifold LMS algorithm needs to be adjusted to ensure convergence. Looking at the update equation (3.17) of the linearised manifold LMS algorithm for an affine manifold, and writing $\tilde{\boldsymbol{x}} = \boldsymbol{J}\boldsymbol{J}^T\boldsymbol{x}$, we can formulate the update equation as

$$\boldsymbol{w}(k+1) = \boldsymbol{w}(k) + \beta\tilde{\boldsymbol{x}}(k)e(k). \tag{3.57}$$

This means that the excitation signal for the manifold LMS algorithm is now $\tilde{\boldsymbol{x}}$ instead of $\boldsymbol{x}$ for the LMS algorithm. Therefore it can be followed that the manifold algorithm converges for

$$0 < \beta < \frac{2}{\lambda_{\max}\left(R_{\tilde{\boldsymbol{x}}\tilde{\boldsymbol{x}}}\right)}, \tag{3.58}$$

where $\lambda_i\left(R_{\tilde{\boldsymbol{x}}\tilde{\boldsymbol{x}}}\right)$ denotes the i-th eigenvalue and $\lambda_{\max}\left(R_{\tilde{\boldsymbol{x}}\tilde{\boldsymbol{x}}}\right)$ denotes the maximum eigenvalue of the input correlation matrix $R_{\tilde{\boldsymbol{x}}\tilde{\boldsymbol{x}}}$ [35]. As described in [35], the value for the upper bound for β can be lowered with the following inequality

$$\lambda_{\max}\left(R_{\tilde{\boldsymbol{x}}\tilde{\boldsymbol{x}}}\right) \geq \operatorname{trace}\left(R_{\tilde{\boldsymbol{x}}\tilde{\boldsymbol{x}}}\right), \tag{3.59}$$

where $\operatorname{trace}(\cdot)$ denotes the trace of a matrix, which is easier to evaluate then the maximum eigenvalue. Equation (3.59) holds, because the trace is equal to the sum of all eigenvalues [32]

$$\sum_{i=1}^{l} \lambda_i\left(R_{\tilde{\boldsymbol{x}}\tilde{\boldsymbol{x}}}\right) = \operatorname{trace}\left(R_{\tilde{\boldsymbol{x}}\tilde{\boldsymbol{x}}}\right) \tag{3.60}$$

and $\lambda_i\left(R_{\tilde{\boldsymbol{x}}\tilde{\boldsymbol{x}}}\right) \geq 0\ \forall i = 1, \ldots, l$, because

$$R_{\tilde{\boldsymbol{x}}\tilde{\boldsymbol{x}}} = \mathrm{E}\left\{\tilde{\boldsymbol{x}}\tilde{\boldsymbol{x}}^T\right\} \tag{3.61}$$

is a non-negative definite matrix [32]. Furthermore, it can be shown that the correlation matrix for a discrete time stochastic process is almost always positive definite [35]. Now we have

$$\begin{aligned}
\operatorname{trace}\left(R_{\tilde{\boldsymbol{x}}\tilde{\boldsymbol{x}}}\right) &= \operatorname{trace}\left(\mathrm{E}\left\{\boldsymbol{J}\boldsymbol{J}^T\boldsymbol{x}(\boldsymbol{J}\boldsymbol{J}^T\boldsymbol{x})^T\right\}\right) &(3.62)\\
&= \operatorname{trace}\left(\mathrm{E}\left\{(\boldsymbol{J}\boldsymbol{J}^T\boldsymbol{x})^T\boldsymbol{J}\boldsymbol{J}^T\boldsymbol{x}\right\}\right) &(3.63)\\
&= \mathrm{E}\left\{\left\|\boldsymbol{J}\boldsymbol{J}^T\boldsymbol{x}\right\|_2^2\right\}, &(3.64)
\end{aligned}$$

where equation (3.63) uses that $\operatorname{trace}(\boldsymbol{AB}) = \operatorname{trace}(\boldsymbol{BA})$, with two matrices $\boldsymbol{A} \in \mathbb{R}^{m\times n}$ and $\boldsymbol{B} \in \mathbb{R}^{n\times m}$ [32]. With the following proposition, we show that the derived upper bound for β in equation (3.64) is the same for the linearised manifold LMS algorithm with $\tilde{\boldsymbol{x}} = \boldsymbol{J}\boldsymbol{J}^T\boldsymbol{x}$ and the manifold LMS algorithm with $\tilde{\boldsymbol{x}} = \boldsymbol{J}^T\boldsymbol{x}$.

Proposition 1. *Let $\boldsymbol{J} \in \mathbb{R}^{m\times n}$ with $m \geq n$ with the columns $\boldsymbol{j}_n$. Assume that $\boldsymbol{J}$ is orthogonal, meaning that $\boldsymbol{j}_n^T\boldsymbol{j}_k = 0\ \forall n \neq k$ and $\boldsymbol{j}_n^T\boldsymbol{j}_n = 1\ \forall n = 1, \ldots, m$, then for any $\boldsymbol{x} \in \mathbb{R}^m$ the following equality holds.*

$$\left\|\boldsymbol{J}\boldsymbol{J}^T\boldsymbol{x}\right\|_2^2 = \left\|\boldsymbol{J}^T\boldsymbol{x}\right\|_2^2 \tag{3.65}$$

Proof:

$$\left\|\boldsymbol{J}\boldsymbol{J}^T\boldsymbol{x}\right\|_2^2 = (\boldsymbol{J}\boldsymbol{J}^T\boldsymbol{x})^T\boldsymbol{J}\boldsymbol{J}^T\boldsymbol{x} \tag{3.66}$$

$$= \boldsymbol{x}^T\boldsymbol{J}\boldsymbol{J}^T\boldsymbol{J}\boldsymbol{J}^T\boldsymbol{x} \tag{3.67}$$

$$= \boldsymbol{x}^T\boldsymbol{J}\boldsymbol{I}_{l'}\boldsymbol{J}^T\boldsymbol{x} \tag{3.68}$$

$$= (\boldsymbol{J}^T\boldsymbol{x})^T\boldsymbol{J}^T\boldsymbol{x} \tag{3.69}$$

$$= \left\|\boldsymbol{J}^T\boldsymbol{x}\right\|_2^2, \tag{3.70}$$

where $\boldsymbol{I}_{l'}$ denotes the identity matrix with l' rows and columns. ■

The full step size β for both the manifold LMS and the linearised manifold LMS is given in terms of the scaled step size α

$$\beta = \frac{\alpha}{\mathrm{E}\left\{\|\boldsymbol{J}^T\boldsymbol{x}\|_2^2\right\}} \tag{3.71}$$

with $0 < \alpha < 2$ to ensure convergence in the mean square [35].

The following proposition gives a result to simplify the term $\mathrm{E}\left\{\left\|\boldsymbol{J}^T\boldsymbol{x}\right\|_2^2\right\}$ of the full step size β.

Proposition 2. *Assuming that $\boldsymbol{x} \in \mathbb{R}^m$ is a white noise process random variable, and that $\boldsymbol{J}$ is orthogonal, meaning that $\boldsymbol{j}_n^T\boldsymbol{j}_k = 0\ \forall n \neq k$ and $\boldsymbol{j}_n^T\boldsymbol{j}_n = 1\ \forall n = 1, \ldots, m$, then the following equality holds.*

$$\mathrm{E}\left\{\left\|\boldsymbol{J}^T\boldsymbol{x}\right\|_2^2\right\} = l'\sigma_x^2 \tag{3.72}$$

Proof:

$$\mathrm{E}\left\{\left\|\boldsymbol{J}^T\boldsymbol{x}\right\|_2^2\right\} = \mathrm{E}\left\{\boldsymbol{x}^T\boldsymbol{J}\boldsymbol{J}^T\boldsymbol{x}\right\} \tag{3.73}$$

$$= \mathrm{E}\left\{\boldsymbol{x}^T\sum_{i=1}^{l'}\boldsymbol{j}_i\boldsymbol{j}_i^T\boldsymbol{x}\right\} \tag{3.74}$$

$$= \sum_{i=1}^{l'}\mathrm{E}\left\{\left\|\boldsymbol{j}_i^T\boldsymbol{x}\right\|_2^2\right\} \tag{3.75}$$

$$= \sum_{i=1}^{l'}\|\boldsymbol{j}_i\|_2^2\,\mathrm{E}\left\{x_i^2\right\} \tag{3.76}$$

$$= l'\sigma_x^2, \tag{3.77}$$

since $\|\boldsymbol{j}_i\|_2^2 = 1$, and

$$\sigma_x^2 = \mathrm{E}\left\{x_i(k)^2\right\}, \tag{3.78}$$

where $x_i(k)$ is the i-th component of the random vector $\boldsymbol{x}(k)$ with $\mathrm{E}\{x_i(k)\} = 0$. ■

Using Proposition 2, equation (3.71) can be simplified to

$$\beta = \frac{\alpha}{l'\sigma_x^2}. \tag{3.79}$$

Convergence of the manifold NLMS algorithm

The normal NLMS algorithm converges for $0 < \alpha < 2$, as it was discussed in Chapter 2. The convergence condition of the manifold NLMS algorithm can be given in an analogous way as the convergence condition of the manifold LMS algorithm, and it follows that the manifold NLMS algorithm converges for $0 < \alpha < 2$ for an affine orthogonal manifold.

3.5.4 Theoretical convergence speed of the manifold LMS algorithm

This section analyses the average convergence speed of the manifold LMS algorithm. The convergence speed is the rate at which the system mismatch gets smaller for every algorithm iteration step.

Comparing the speed of convergence of the LMS algorithm and the NLMS algorithms, the NLMS algorithm has a potentially faster rate of convergence for both correlated and uncorrelated excitation signals compared to the LMS algorithm [14, 33, 15]. One reason is that the NLMS algorithm is stable in each iteration and not only in the mean, as it is for the LMS algorithm [35]. Furthermore, for both the LMS and NLMS algorithm the rate of convergence can be significantly slower for highly correlated input signals [35, 14]. In particular, the convergence speed and the steady state performance of the LMS and NLMS algorithms depends on the kurtosis of the input signal [38]. When the step size α is large, the NLMS algorithm has a faster convergence rate than the LMS algorithm, but for small step sizes $\alpha \ll 1$, the convergence rate of the LMS and NLMS are approximately the same, since

$$\|x\|_2^2 \approx \sigma_x^2 l \tag{3.80}$$

for large enough l [38, 35]. This means that the derived convergence rate of the manifold LMS algorithm should also approximately hold for the manifold NLMS algorithm, since

$$\left\|\boldsymbol{J}^T x\right\|_2^2 \approx \sigma_x^2 l'. \tag{3.81}$$

Another advantage of the NLMS algorithm is that σ_x^2 does not need to be known to guarantee convergence in comparison to the LMS algorithm, where σ_x^2 needs to be known beforehand to set the step size [38].

To analyse the convergence speed of the manifold LMS algorithm, we will assume the absence of any interference, meaning that the near end noise is zero, i.e. $n(k) = 0$, and that the true impulse response lies perfectly on the manifold. This means that the manifold vector rejection $\boldsymbol{\Delta}_{\mathcal{M}}$, which is defined as the difference between the true impulse response $\boldsymbol{w}_\star$ and the true impulse response mapped onto the manifold is assumed to be zero with

$$\boldsymbol{\Delta}_{\mathcal{M}} = \boldsymbol{w}_\star - \boldsymbol{f}_{\text{dec}}(\boldsymbol{f}_{\text{enc}}(\boldsymbol{w}_\star)) = 0. \tag{3.82}$$

This assumption of not having any interferences is generally infeasible in practice, but at the start of the initial convergence phase, the error introduced by $\|\boldsymbol{w}(k) - \boldsymbol{w}_\star\|_2^2$ is usually dominant over the error from the near end noise for large enough ENR. This means that a good approximation of the initial convergence speed even with some interference can be expected. The input signal $\boldsymbol{x}(k)$ is assumed to be a stationary white noise signal. Furthermore, it is assumed that the signals $\boldsymbol{d}(k)$ and $\boldsymbol{x}(k)$ are statistically independent. The validity of this assumption is discussed in Section 4.2.1.

It is calculated how much the mean system mismatch $\boldsymbol{D}(k)$ decreases in one discrete time step for a single realisation of $\boldsymbol{w}_\star$ to give a number of the convergence speed. The following analysis of the convergence speed is based on the same steps taken in [35], but modified for the manifold LMS algorithm.

For simpler notation, the distance $\boldsymbol{d}(k)$ between the true impulse response $\boldsymbol{w}_\star$ to the current estimate $\boldsymbol{w}(k)$ is defined with

$$\boldsymbol{d}(k) = \boldsymbol{w}_\star - \boldsymbol{w}(k). \tag{3.83}$$

The system mismatch $\boldsymbol{D}(k)$ is the squared Euclidean norm of $\boldsymbol{d}(k)$ normalised by the squared Euclidean norm of the true impulse response and is given with

$$\boldsymbol{D}(k) = \mathrm{E}\left\{\frac{\|\boldsymbol{d}(k+1)\|_2^2}{\|\boldsymbol{w}_\star\|_2^2}\right\} = \frac{\mathrm{E}\left\{\|\boldsymbol{d}(k+1)\|_2^2\right\}}{\|\boldsymbol{w}_\star\|_2^2}. \tag{3.84}$$

Since the decay rate of the system mismatch of the LMS algorithm is exponential, it makes most sense not to look at the difference, but at the ratio of the system mismatch at time step $k+1$ compared to the time step k to quantify the convergence speed, which gives the ratio

$$\frac{\mathrm{E}\left\{\frac{\|\boldsymbol{d}(k+1)\|_2^2}{\|\boldsymbol{w}_\star\|_2^2}\right\}}{\mathrm{E}\left\{\frac{\|\boldsymbol{d}(k)\|_2^2}{\|\boldsymbol{w}_\star\|_2^2}\right\}} = \frac{\mathrm{E}\left\{\|\boldsymbol{d}(k+1)\|_2^2\right\}}{\mathrm{E}\left\{\|\boldsymbol{d}(k)\|_2^2\right\}}. \tag{3.85}$$

In order to give the convergence speed s in dB/sec, the formula

$$s = 10\log_{10}\left(\frac{\mathrm{E}\left\{\|\boldsymbol{d}(k+1)\|_2^2\right\}}{\mathrm{E}\left\{\|\boldsymbol{d}(k)\|_2^2\right\}}\right) f_s \left[\frac{\mathrm{dB}}{\mathrm{sec}}\right] \tag{3.86}$$

is used, where f_s is the sampling rate in 1/sec.

The derivation of the convergence speed of the manifold LMS algorithm is listed in Appendix C, where under the given assumptions the convergence speed is show to be calculated to

$$s = 10\log_{10}\left(1 - \frac{\alpha}{l'}(2-\alpha)\right) f_s \left[\frac{\mathrm{dB}}{\mathrm{sec}}\right], \tag{3.87}$$

with the step size α and the number of manifold components l'.

3.5.5 Theoretical steady state system mismatch of the manifold LMS algorithm

The steady state performance of the LMS algorithm is analysed in [35], which in this section is extended to the steady state analysis of the manifold LMS algorithm. The steady state performance mainly depends on the step size α and the near end noise $n(k)$, which is assumed to be a stationary white noise process and to be statistically independent of $\boldsymbol{x}(k)$. Furthermore, two cases are distinguished, where first the case is analysed when the true impulse response lies perfectly on the manifold, i.e. the manifold vector rejection is zero ($\boldsymbol{\Delta}_\mathcal{M} = \boldsymbol{0}$). The second more general case is that the true impulse response does not lie perfectly on the manifold, i.e. the manifold vector rejection is not zero. The input signal $\boldsymbol{x}(k)$ is assumed to be a stationary white noise signal, and it is assumed that the signals $\boldsymbol{d}(k)$ and $\boldsymbol{x}(k)$ are uncorrelated for large k.

The steady state system mismatch $\boldsymbol{D}_\infty$ for one realisation of a true impulse response $\boldsymbol{w}_\star$ is given as

$$\boldsymbol{D}_\infty = \frac{\mathrm{E}\left\{\|\boldsymbol{d}_\infty\|_2^2\right\}}{\|\boldsymbol{w}_\star\|_2^2} = \frac{\mathrm{E}\left\{\|\boldsymbol{w}_\infty - \boldsymbol{w}_\star\|_2^2\right\}}{\|\boldsymbol{w}_\star\|_2^2}, \tag{3.88}$$

where $\boldsymbol{d}_\infty$ and $\boldsymbol{w}_\infty$ denote $\boldsymbol{d}(k)$ and $\boldsymbol{w}(k)$ for $k \to \infty$, respectively. The following analysis of the steady state behaviour of the LMS algorithm in based on [35], and modified for the manifold LMS algorithm. The derivation of the expected steady state system mismatch is conducted in Appendix D.

For the case of a zero manifold vector rejection ($\boldsymbol{\Delta}_\mathcal{M} = \boldsymbol{0}$) and assumptions named above, the steady state system mismatch $\boldsymbol{D}_\infty$ is shown to be given as

$$\boldsymbol{D}_\infty = \frac{\alpha}{2-\alpha} \frac{\sigma_n^2}{\sigma_x^2 \|\boldsymbol{w}_\star\|_2^2}, \tag{3.89}$$

for the step size α, the noise power σ_n^2, and the input signal power σ_x^2. The steady state system mismatch for the manifold LMS algorithm is therefore equal to the steady state system mismatch of the LMS algorithm for $\boldsymbol{\Delta}_\mathcal{M} = \boldsymbol{0}$ [35].

For the more general case with a manifold vector rejection, meaning that the true impulse response does not lie perfectly on the manifold, together with the previously listed assumptions, the steady state system mismatch $\boldsymbol{D}_\infty$ is shown to be

$$\boldsymbol{D}_\infty = \frac{\|\boldsymbol{\Delta}_\mathcal{M}\|_2^2}{\|\boldsymbol{w}_\star\|_2^2} + \frac{\alpha}{2-\alpha}\left(\frac{\|\boldsymbol{\Delta}_\mathcal{M}\|_2^2}{\|\boldsymbol{w}_\star\|_2^2} + \frac{\sigma_n^2}{\sigma_x^2 \|\boldsymbol{w}_\star\|_2^2}\right). \tag{3.90}$$

Chapter 4

Experimental Results

In the following chapter, simulations of different acoustics scenarios are used to compare the manifold approach to the common Kalman filter in the time domain. In Section 4.1 it is analysed how the PCA manifold compares to the time domain in mapping the true impulse response with only a limited number of variables. In the sections thereafter, the manifold LMS algorithm, the manifold NLMS algorithm, and the manifold Kalman filter are evaluated and compared to their counterparts in the time domain. The simulation results of the manifold LMS and manifold NLMS algorithms are also compared to the theoretically derived convergence speed and steady state value of the system mismatch.

4.1 Investigation of the existence of an acoustic manifold

This section investigates whether the use of an acoustic manifold compared to the time domain has an advantage in mapping impulse responses in different acoustic scenarios. Four acoustics scenarios are analysed, where for each scenario four levels of acoustic complexity were considered. In the first scenario, the loudspeaker and the microphone are randomly positioned inside a room. For the second scenario, the loudspeaker and the microphone have a fixed distance, as it is the case for a smart device for example. The third and fourth scenario are simpler cases, where the loudspeaker is at a fixed position, and the microphone is randomly positioned on a line (third scenario) or randomly positioned on a circle around the loudspeaker (fourth scenario).

For all 16 settings, 4500 setups of the loudspeaker and microphone position inside the room are randomly generated, and for each setup the impulse responses between the loudspeaker and the microphone is simulated. These 4500 impulse responses are used to train a PCA. In the experiments it will be investigated how well the impulse responses can be represented by the subspace spanned by the first l' principal components. As references to the principal components, a truncation of the impulse response to l' time domain components and a truncation of the impulse response to l' time domain components with delay compensation and consideration of the mean value. The case with delay compensation and consideration of the mean value uses the first l' time domain components starting from the first non zero time domain component of the average impulse response, which consider the time it takes the sound to travel from the loudspeaker to the microphone. Furthermore, the mean value of the training data is considered for all remaining time domain components. This allows for a more fair comparison to the PCA, as the PCA learns to ignore the time domain components that are always zero, and considers the mean of the training data to represent the impulse responses.

The manifold vector rejection $\boldsymbol{\Delta}_{\mathcal{M}}$ was defined in Chapter 3 (equation (3.82)) and is now explicitly written as a function of the number of variables l' with

$$\boldsymbol{\Delta}_{\mathcal{M}}\left(l'\right) = \boldsymbol{w}_{\star} - \boldsymbol{f}_{\mathrm{dec}}^{(l')}\left(\boldsymbol{f}_{\mathrm{enc}}^{(l')}\left(\boldsymbol{w}_{\star}\right)\right), \tag{4.1}$$

where $\boldsymbol{f}_{\mathrm{dec}}^{(l')}$ and $\boldsymbol{f}_{\mathrm{enc}}^{(l')}$ denote the decoder and encoder of the manifold with l' components, respectively, and $\boldsymbol{w}_{\star}$ is the true impulse response. For the sake of readability, the subspace that is spanned by the first principal components is called PCA manifold in the following. Regarding the time domain representation, it can be stated that all impulse responses with only l' non zero components also populate a linear subspace, where the Jacobian matrix of the encoder and decoder is a truncated identity matrix. The time domain representation with l' components is therefore a special case of a manifold, namely the identity manifold.

The figures in this section show the achievable system mismatch in dB over the number of components l' from 0 to 2000. The number of time domain components of the true impulse response is $l = 4000$ at a sampling frequency of 8000 Hz. The achievable system mismatch is averaged over 20 independently simulated impulse responses. The room has the dimensions of 4 m in the x direction, 6 m in the y direction, and 3 m in the z direction for all impulse response simulations. The impulse responses are simulated using the image source method, where the walls are considered as perfect reflectors [31]. In Section 3.5.5 it was shown that the achievable system mismatch is in fact a lower bound to the system mismatch of the LMS algorithm. The achievable system mismatch $\boldsymbol{D}_{\mathrm{a}}$ is the ratio of the squared distance of the vector rejection $\boldsymbol{\Delta}_{\mathcal{M}}$ and the true impulse response with

$$\boldsymbol{D}_{\mathrm{a}} = 10\log_{10}\left(\frac{\left\|\boldsymbol{\Delta}_{\mathcal{M}}\left(l'\right)\right\|_2^2}{\left\|\boldsymbol{w}_{\star}\right\|_2^2}\right)\ \text{[dB]}. \tag{4.2}$$

This way, if a specific system mismatch needs to be achieved, the number of components l' in each scenario can be found in the figures of this section.

4.1.1 Loudspeaker and microphone at random positions inside the room

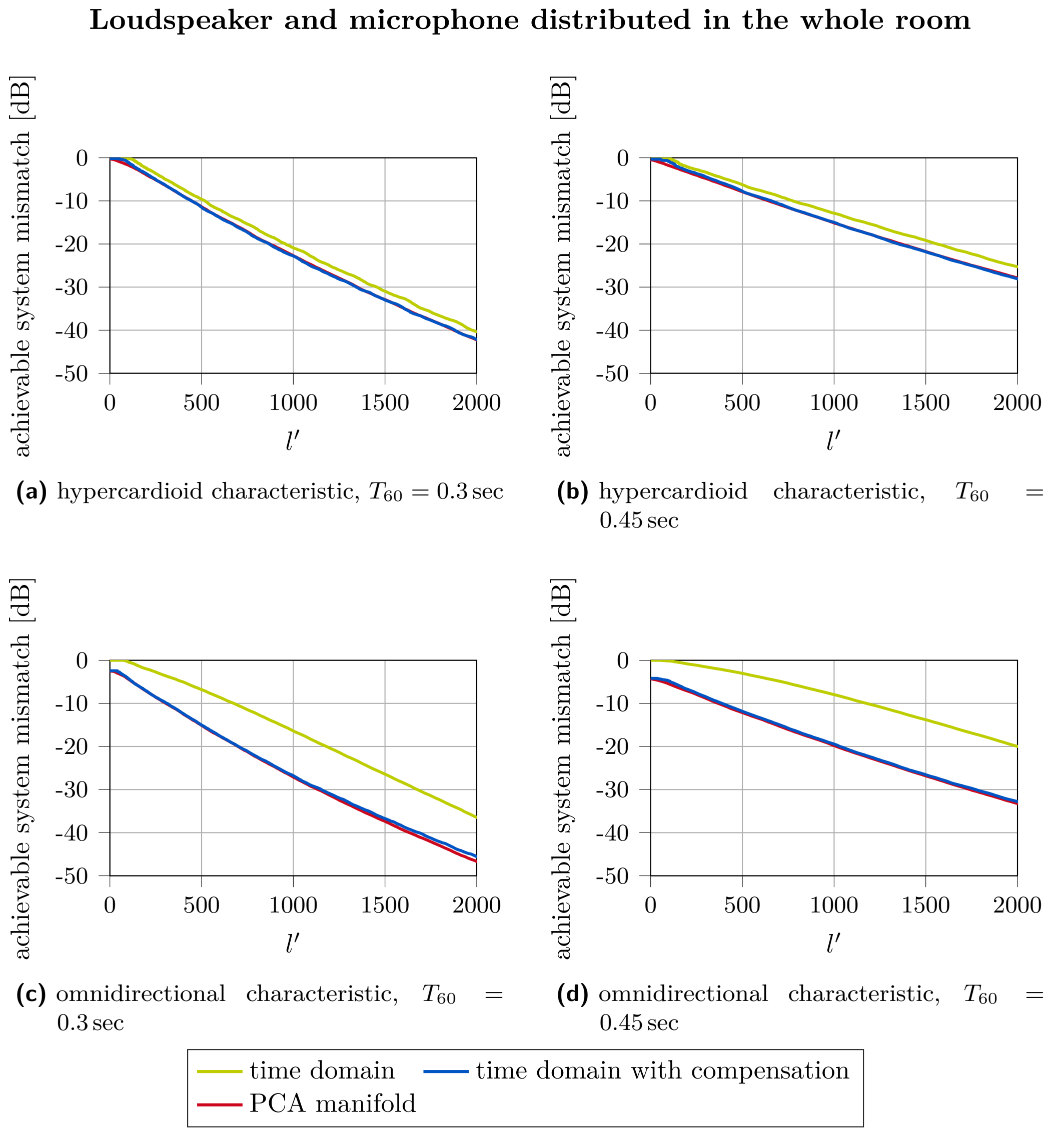
To test the influence of the the directivity of the loudspeaker and the microphone, two cases are analysed. First, the case of the hypercardioid characteristic is analysed, which has the highest focus in the forward direction of the first order cardioid characteristics [5]. The hypercardioid characteristic has a directional response of -12 dB at 90° and -6 dB at 180° [5]. Second, the more simple case, where both the loudspeaker and the microphone are omnidirectional, is analysed, which means that the loudspeaker emits and the microphone picks up the sound equally in all directions [5]. This way, a simple case without directivity and a more complex case with high directivity can be compared to see in which case the PCA can better map the impulse responses. For all simulations where the hypercardioid characteristic is used, it is oriented in the x-y plane, and the directivity is frequency independent [31].

One import characterisation of a room is the reverberation time. The reverberation time T_{60} [sec] is defined as the time, until the energy of the sound field in a room decays by 60 dB from its initial level after the sound source is turned off [24]. A longer reverberation time of a room means that an impulse responses decays slower, and has more components that are above a certain level. To investigate the effect of different reverberation times, two cases are simulated with $T_{60} = 0.3$ seconds (Figures 4.1a and 4.1c) and $T_{60} = 0.45$ seconds (Figures 4.1b and 4.1d), which are typical values for the reverberation time of living area rooms [24].

For the simulation of the impulse response the loudspeaker and the microphone are independent and uniformly distributed anywhere inside the room for the reverberation times of $T_{60} = 0.45$ seconds and $T_{60} = 0.45$ seconds, where the loudspeaker and the microphone both have a hypercardioid characteristic or both have an omnidirectional characteristic. When the hypercardioid characteristic is used, the loudspeaker and the microphone are oriented independently in random directions. In Figure 4.1, the manifold mismatch is shown for the PCA manifold, the time domain, and the time domain with delay compensation and consideration of the mean value. The difference of the time domain and the time domain with delay compensation is that in the case of the time domain with delay compensation, the first components of the time domain, where the impulse response is always zero, are not counted. In the presented scenario, the distance between the loudspeaker and the microphone is random and can be arbitrarily small, meaning that there is no guaranteed delay. In prior experiments it was confirmed that the effect of the delay compensation is small. However, the effect of taking the mean of the training data into account is significant, as it is observed that the achievable system mismatch is significantly larger for the time domain compared to the time domain with delay compensation and consideration of the mean value as seen in Figures 4.1c and 4.1d. For easier notation the time domain with delay compensation and consideration of the mean value of the training data will be referred to as time domain with compensation.

For Figures 4.1a and 4.1b, the loudspeaker and the microphone have a hypercardioid characteristic with an independent random uniformly distributed orientation in the x-y plane. The Figures 4.1c and 4.1d show the case where the loudspeaker and the microphone have an omnidirectional characteristic. For the case where the loudspeaker and the microphone are randomly oriented in different directions (Figures 4.1a and 4.1b), the curves of the achievable system mismatch for the time domain and the PCA manifold are close together, meaning that the PCA can not extract significantly more information per component compared to an equivalent truncation of the impulse response in the time domain. For the case of an omnidirectional loudspeaker and microphone (Figures 4.1c and 4.1d), the difference between the approaches is significantly larger. For example, for $l' = 1000$ components in Figure 4.1c, the PCA has an achievable system mismatch of -27.1 dB and the time domain has an system mismatch of -16.4 dB, i.e. a significant difference of 10.7 dB. It can be concluded that the acoustically more complex case, where the loudspeaker and the microphone have orientations in different directions, the PCA manifold components do not explain significantly more variance than the time domain components.

For the comparison of the different reverberation times, only Figures 4.1c and 4.1d are compared to each other, as in Figures 4.1a and 4.1b, the difference of the PCA manifold and the time domain is small. When the reverberation time increases from $T_{60} = 0.3$ sec (Figure 4.1c) to $T_{60} = 0.45$ sec (Figure 4.1d), the overall achievable system mismatch increases for the same number of variables l'. For $T_{60} = 0.45$ sec (Figure 4.1d) and $l' = 1000$ components for example, the PCA has an achievable system mismatch of -19.8 dB and the time domain has an achievable system mismatch of -8.1 dB, i.e. a difference of 11.7 dB. When the difference between the achievable system mismatch of the time domain and the PCA manifold is analysed,



(a) hypercardioid characteristic, $T_{60} = 0.3$ sec

(b) hypercardioid characteristic, $T_{60} = 0.45$ sec

(c) omnidirectional characteristic, $T_{60} = 0.3$ sec

(d) omnidirectional characteristic, $T_{60} = 0.45$ sec

Figure 4.1: The achievable system mismatch over the number of components l' is shown for the time domain, for the time domain with delay compensation and consideration of the mean of the training data (time domain with compensation), and for the PCA manifold. The positions of the loudspeaker and the microphone are anywhere inside the room, with either a hypercardioid characteristic or an omnidirectional characteristic. The reverberation time is set to $T_{60} = 0.3$ sec or $T_{60} = 0.45$ sec.

this difference is slightly larger for larger reverberation times. For the case of $T_{60} = 0.3$ sec, this distance was 10.7 dB as calculated above for $l' = 1000$, compared to a difference of 11.7 dB for $T_{60} = 0.45$ sec.

Since the the achievable system mismatch of the PCA manifold and the time domain with delay compensation is nearly identical it can be concluded that in a situation where the training data comes from loudspeaker and microphone positions distributed over a whole room, there is nearly no correlation in the training data which could be exploited by the PCA. However, the achievable system mismatch for the PCA manifold compared to the time domain is smaller, indicating that the mean of the training data carries a significant amount of information. From these findings that a higher reverberation time leads to a greater difference in the respectively achievable system mismatch between the time domain and the PCA manifold, the case of $T_{60} = 0.45\,\mathrm{sec}$ will be considered in the following experiments. Furthermore, it was shown that a scenario with higher variability in the setup leads to a smaller difference in the achievable system mismatch between the time domain and the PCA manifold. Therefore, in the following sections, scenarios with less variability in the setup to simulate the impulse responses will be investigated.

4.1.2 Loudspeaker and microphone with fixed relative position

Another scenario is the case where the loudspeaker and the microphone are in a fixed relative position to each other, which is typical for a smart device like a smartphone or a smart speaker. Therefore, the loudspeaker together with the microphone will be called smart device in this section. Compared to the previous case, here the variability in the impulse responses is lower, because the distance between the loudspeaker and the microphone is fixed to be 10 cm in the z direction. In Figures 4.2a and 4.2c, the scenario of the full room is used, where the x and y coordinate of the position of the smart device is distributed in the entire room of the x-y plane with the position of the loudspeaker $[x_{\mathrm{ls}}, y_{\mathrm{ls}}, z_{\mathrm{ls}}]$ uniformly distributed in the range of $x_{\mathrm{ls}} \in [0\,\mathrm{m}, 4\,\mathrm{m}]$, $y_{\mathrm{ls}} \in [0\,\mathrm{m}, 6\,\mathrm{m}]$, and $z_{\mathrm{ls}} \in [1.5\,\mathrm{m}, 1.6\,\mathrm{m}]$. In Figures 4.2b and 4.2d, the scenario of one square meter is used, where the position of the smart device is even more restricted to be uniformly distributed in the range of $x_{\mathrm{ls}} \in [1\,\mathrm{m}, 2\,\mathrm{m}]$, $y_{\mathrm{ls}} \in [2\,\mathrm{m}, 3\,\mathrm{m}]$, and $z_{\mathrm{ls}} \in [1.5\,\mathrm{m}, 1.6\,\mathrm{m}]$, which spans one square meter in the x-y plane. In both scenarios the position of the microphone is fixed in relation to the position of the loudspeaker with $x_{\mathrm{mic}} = x_{\mathrm{ls}}$, $y_{\mathrm{mic}} = y_{\mathrm{ls}}$, $z_{\mathrm{mic}} = z_{\mathrm{ls}} + 0.1\,\mathrm{m}$. For the hypercardioid characteristic, the loudspeaker and microphone are both oriented towards the positive y axis, and T_{60} is 0.45 seconds.

By comparing Figures 4.2a and 4.2c to Figures 4.2b and 4.2d it can be clearly seen that restricting the position of the smart device to a smaller space, the difference of the achievable system mismatch of the PCA manifold compared to the time domain with compensation with the same number of components is much larger. Both the PCA manifold and the time domain with compensation start at an achievable system mismatch of around $-21\,\mathrm{dB}$ for $l' = 0$ in Figure 4.2d, which is due to the large influence of the mean value of the training data in the estimation of the impulse response. However, when more components l' are used, the PCA manifold has a lower achievable system mismatch, since in the case where the smart device can only be positioned in a small space, the PCA is able to exploit correlation in the training data. The achievable system mismatch for the time domain is significantly higher compared to the time domain with compensation because the mean of the training data carries a significant amount of information.

For the case of an omnidirectional loudspeaker and microphone (Figures 4.2c and 4.2d), the difference of the achievable system mismatch of the PCA manifold compared to the time domain with compensation is in the same range as the case of a loudspeaker and a microphone with a hypercardioid characteristic (Figures 4.2a and 4.2b). The difference of the achievable system mismatch of the PCA manifold compared to the time domain is larger for the case of an omnidirectional loudspeaker and microphone.

A comparison of Figure 4.2c and Figure 4.1d from the previous section reveals that the the loudspeaker and the microphone having a fixed position to each other, like it is in the smart device, results only in a slightly lower achievable system mismatch of the PCA manifold

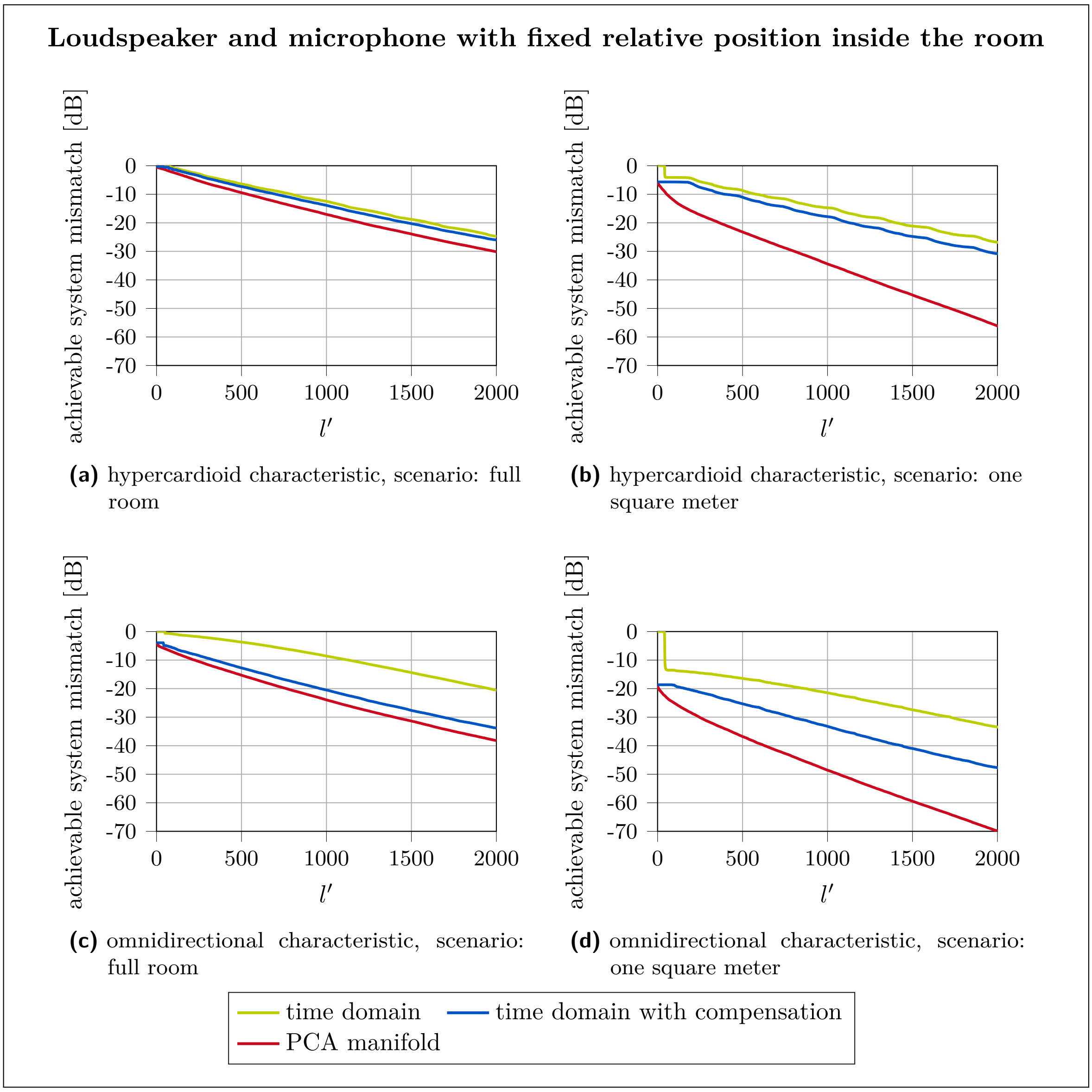


Figure 4.2: The achievable system mismatch over the number of components l' is shown for the time domain, for the time domain with delay compensation and consideration of the mean of the training data (time domain with compensation), and for the PCA manifold. The loudspeaker and the microphone have a fixed relative position, with either a hypercardioid characteristic or an omnidirectional characteristic for two scenarios of different restrictiveness in the positions of the loudspeaker and the microphone.

compared to the references. Only when the positions of the loudspeaker and the microphone is restricted to a smaller space in the room, the difference of the achievable system mismatch of the PCA manifold and the references increases significantly, as seen by comparing Figure 4.2d to Figure 4.2c.

4.1.3 Loudspeaker fixed and microphone randomly positioned on a line

It was observed in the previous section that the difference of the achievable system mismatch for the PCA manifold and the time domain with compensation are the largest for settings, where the positions of the loudspeaker and the microphone are restricted to a small space. This section analyses whether the substantial difference in achievable system mismatch between the PCA manifold and the time domain with compensation in the previous section mainly arises from the fixed relative position of the loudspeaker and microphone. To accomplish this, it is evaluated whether the difference persists or increases when the loudspeaker and microphone positions are confined to a smaller space, but this time without a fixed relative position. Furthermore, the loudspeaker and microphone characteristic is omnidirectional in this experiment, since it was observed in the previous section that the difference in the achievable system mismatch between the PCA manifold and the time domain is larger in that case compared to the case of a hypercardioid characteristic.

In this section, the loudspeaker is kept at a fixed position with an x coordinate of $1\,\mathrm{m}$, a y coordinate of $1.5\,\mathrm{m}$, and a z coordinate of $1.6\,\mathrm{m}$, with the room having the same size as in the previous section. The microphone is positioned on a line in the y direction, where the distance of the loudspeaker and the microphone is uniformly distributed between $0\,\mathrm{m}$ and $4\,\mathrm{m}$. The loudspeaker is positioned on one end of this line. This way, compared to the previous section, the positions of the loudspeaker and the microphone are way more restricted. Four cases are analysed, where the position of the microphone in the x and z direction is varied to different degrees. The position of the microphone is being restricted to a long rectangular cuboid along the $4\,\mathrm{m}$ line in the y direction, with a length of x_Δ in the x direction, and a length of z_Δ in the z direction. The position of the microphone in the x and z direction are varied with a uniform distribution in the range of x_Δ and z_Δ, and four cases are analysed from $x_\Delta = z_\Delta = 0\,\mathrm{m}$ in Figure 4.3a to $x_\Delta = z_\Delta = 0.4\,\mathrm{m}$ in Figure 4.3d. The exact coordinate ranges of the microphone positions are then given with $x_{\mathrm{mic}} \in [1\,\mathrm{m} - \frac{x_\Delta}{2}, 1\,\mathrm{m} + \frac{x_\Delta}{2}]$, $y_{\mathrm{mic}} \in [1.5\,\mathrm{m}, 5.5\,\mathrm{m}]$, and $z_{\mathrm{mic}} \in [1.6\,\mathrm{m} - \frac{z_\Delta}{2}, 1.6\,\mathrm{m} + \frac{z_\Delta}{2}]$. The x and z positions are uniformly distributed inside this rectangular cuboid in each setting with exemplarily $x_\Delta = z_\Delta = 0\,\mathrm{m}$ in Figure 4.3a, and $x_\Delta = z_\Delta = 0.4\,\mathrm{m}$ in Figure 4.3d. The reverberation time T_{60} is set to 0.45 seconds.

The PCA maps the true impulse response very well with only 100 PCA manifold components for $x_\Delta = z_\Delta = 0$ m (Figure 4.3a), as the achievable system mismatch is $-70\,\mathrm{dB}$ in this case. For the time domain with compensation an achievable system mismatch of $-20\,\mathrm{dB}$ is reached with $l' = 1000$ components, and the time domain has an achievable system mismatch of $-20\,\mathrm{dB}$ with $l' = 1800$ components. This means that for $x_\Delta = z_\Delta = 0$ m, there is a significant amount of correlation in the training data that can be exploited by the PCA. Because the difference of the time domain with compensation and the time domain stays the same when x_Δ and z_Δ are increased it can be followed that the significance of the mean and the delay to represent the impulse responses does not get larger when the loudspeaker and microphone positions are restricted to smaller spaces. The scale of the manifold mismatch is restricted from $0\,\mathrm{dB}$ to $-70\,\mathrm{dB}$, because further improvements in the achievable system mismatch are of minor relevance for practical usage, since microphone noise and fluctuations in the acoustic path limit the performance to smaller system mismatches.

When the position of the microphone is less restricted, meaning that the position of the x and z coordinate is randomly varied, the achievable system mismatch decreases for the PCA manifold. This is observed when comparing the cases of $x_\Delta = z_\Delta = 0\,\mathrm{m}$ in Figure 4.3a, $x_\Delta = z_\Delta = 0.1\,\mathrm{m}$ in Figure 4.3b, $x_\Delta = z_\Delta = 0.2\,\mathrm{m}$ in Figure 4.3c, and $x_\Delta = z_\Delta = 0.4\,\mathrm{m}$ in

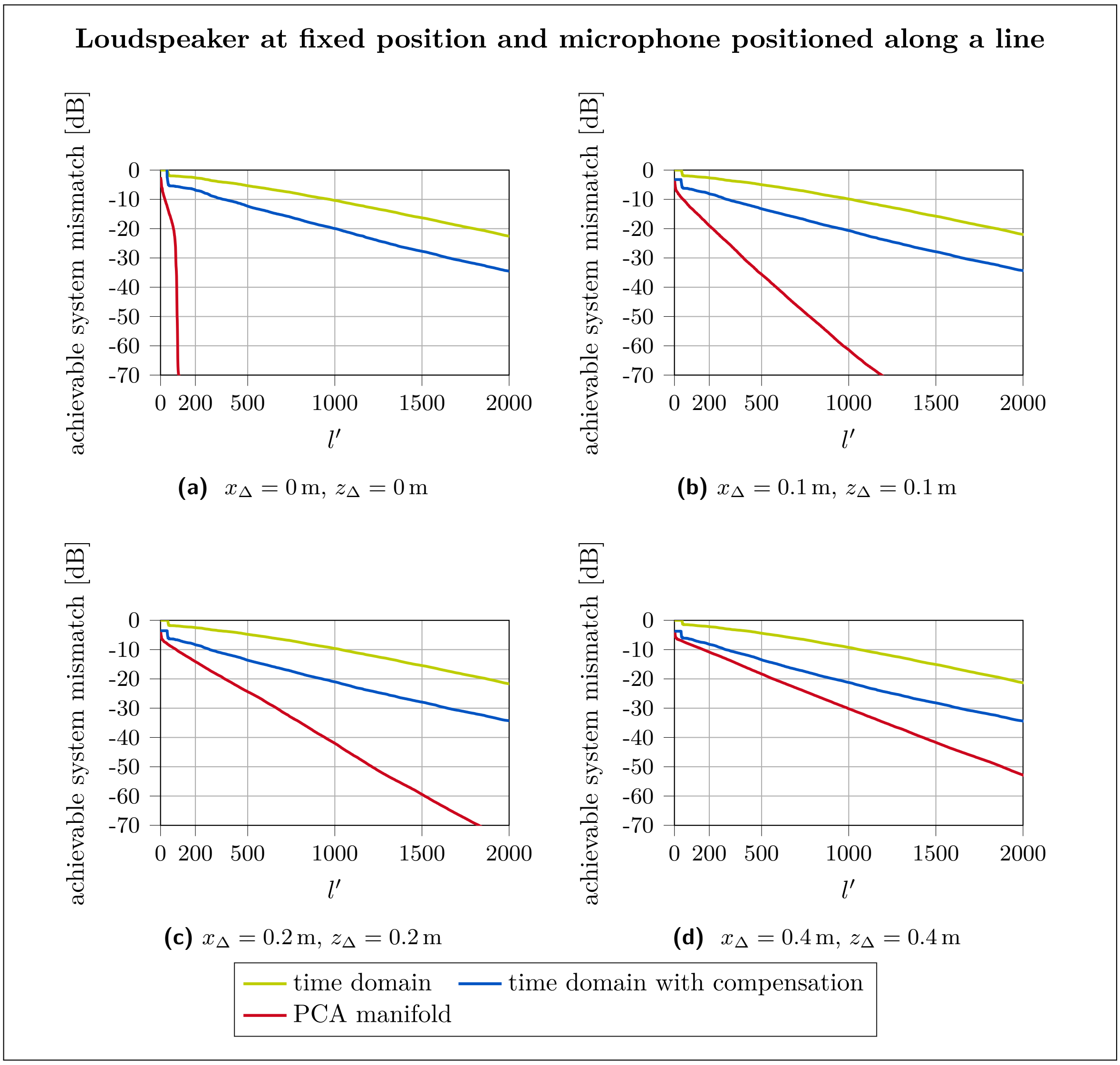


Figure 4.3: The achievable system mismatch over the number of components l' is shown for the time domain, for the time domain with delay compensation and consideration of the mean of the training data (time domain with compensation), and for the PCA manifold. The loudspeaker is at a fixed position and the microphone is randomly positioned on a line with four cases of varying restrictiveness of the microphone position.

Figure 4.3d. For example, only about $l' = 100$ PCA manifold components are needed for an achievable system mismatch of $-70\,\mathrm{dB}$ in Figure 4.3a, whereas in Figure 4.3d $l' = 1000$ PCA manifold components give an achievable system mismatch of about only $-30\,\mathrm{dB}$. This is in contrast to the corresponding behaviour of the curves of the achievable system mismatch for the time domain components and the time domain with compensation components, which do not change much when $x_\triangle$ and $z_\triangle$ are varied.

4.1.4 Loudspeaker fixed and microphone randomly positioned on a circle

In this section, a setting is used with a similar variation in the loudspeaker and microphone positions as in the previous section. The focus is now to investigate if the observation of the previous section are not just true for specific settings, but can be confirmed for a different acoustic scenario.

Therefore, the loudspeaker is kept again at a fixed position at the x coordinate of 1.5 m, the y coordinate of 2 m, and the z coordinate of 1.5 m, but the microphone is randomly positioned on a circle in the x-y plane around the loudspeaker instead of being positioned along a line as in the previous section. The average radius is given with $r_{\mathrm{mic}} = 1\,\mathrm{m}$, and the average z coordinate of the microphone is 1.5 m, which is the same as for the loudspeaker. For the four cases from Figure 4.4a to Figure 4.4d, the radius r and the z coordinate of the microphone position are varied from $r_\Delta = z_\Delta = 0\,\mathrm{m}$ in Figure 4.4a to $r_\Delta = z_\Delta = 0.4\,\mathrm{m}$ in Figure 4.4d. The microphone is positioned uniformly distributed on the circle in the x-y plane with radius $r_{\mathrm{mic}} \in [1\,\mathrm{m} - \frac{r_\Delta}{2}, 1\,\mathrm{m} + \frac{r_\Delta}{2}]$ with the z coordinate z_{mic} being uniformly distributed in the range of $z_{\mathrm{mic}} \in [1.5\,\mathrm{m} - \frac{z_\Delta}{2}, 1.5\,\mathrm{m} + \frac{z_\Delta}{2}]$. The loudspeaker and microphone characteristic is omnidirectional, and T_{60} is 0.45 seconds.

Comparing Figures 4.4 and 4.3, it can be observed that they are quite similar. This indicates that with a fixed loudspeaker position it does not matter, if the distance from the loudspeaker to the microphone is kept constant (Figure 4.4a), but it matters how restricted the microphone position is in general. Comparing the numbers of Figures 4.4a and 4.3a, for the most restricted case of Figure 4.4a, $l' = 160$ components are needed to achieve a manifold mismatch of $-70\,\mathrm{dB}$ with the PCA manifold with the microphone randomly positioned on a circle, whereas 100 PCA manifold components were needed in Figure 4.3a for the microphone along the line. When there is more variability in the positions (Figure 4.4c), the number of components is slightly less with $l' = 1750$ compared to $l' = 1830$ in Figure 4.3c to achieve a manifold mismatch of $-70\,\mathrm{dB}$. When the radius can vary with a uniform distribution from 0.8 m to 1.2 m and the z coordinate of the microphone from 1.3 m to 1.7 m (Figure 4.4d), the PCA manifold needs approximately 580 components, which is quite similar to Figure 4.3d, where $l' = 570$ components are needed to achieve a manifold mismatch of $-20\,\mathrm{dB}$. The same similarity of Figures 4.4d and 4.3d also applies for the time domain with compensation, where the time domain with compensation needs approximately 1000 components (Figure 4.4d), which is the same number of components as seen Figure 4.3d to have an achievable system mismatch of $-20\,\mathrm{dB}$.

In all following simulations of the manifold LMS algorithm, the manifold NLMS algorithm and the manifold Kalman filter, two cases of this section with different variability in the microphone positions will be used. The first case of Figure 4.4a is a simple case, where it was shown in this section that only a few hundred PCA manifold components are needed to almost fully map the true impulse response without loss of information. The second case of Figure 4.4d is a more realistic scenario, where the microphone can move around in a larger space, but the PCA can still have the same achievable system mismatch with considerable less number of variables compared to the time domain and the time domain with compensation. In the following, the impulse responses from the scenario of Figure 4.4a will be referred to as impulse response dataset $\mathcal{A}$, and the impulse responses from the scenario of Figure 4.4d will be referred to as impulse response dataset $\mathcal{B}$.

Figure 4.5 visualises the PCA manifold by showing the projection matrix $\boldsymbol{J}\boldsymbol{J}^T$ of the manifold for the cases of Figures 4.3a, 4.3d, 4.4a, and 4.4d for $l' = 100$ components. The matrix $\boldsymbol{J}(k)$ is

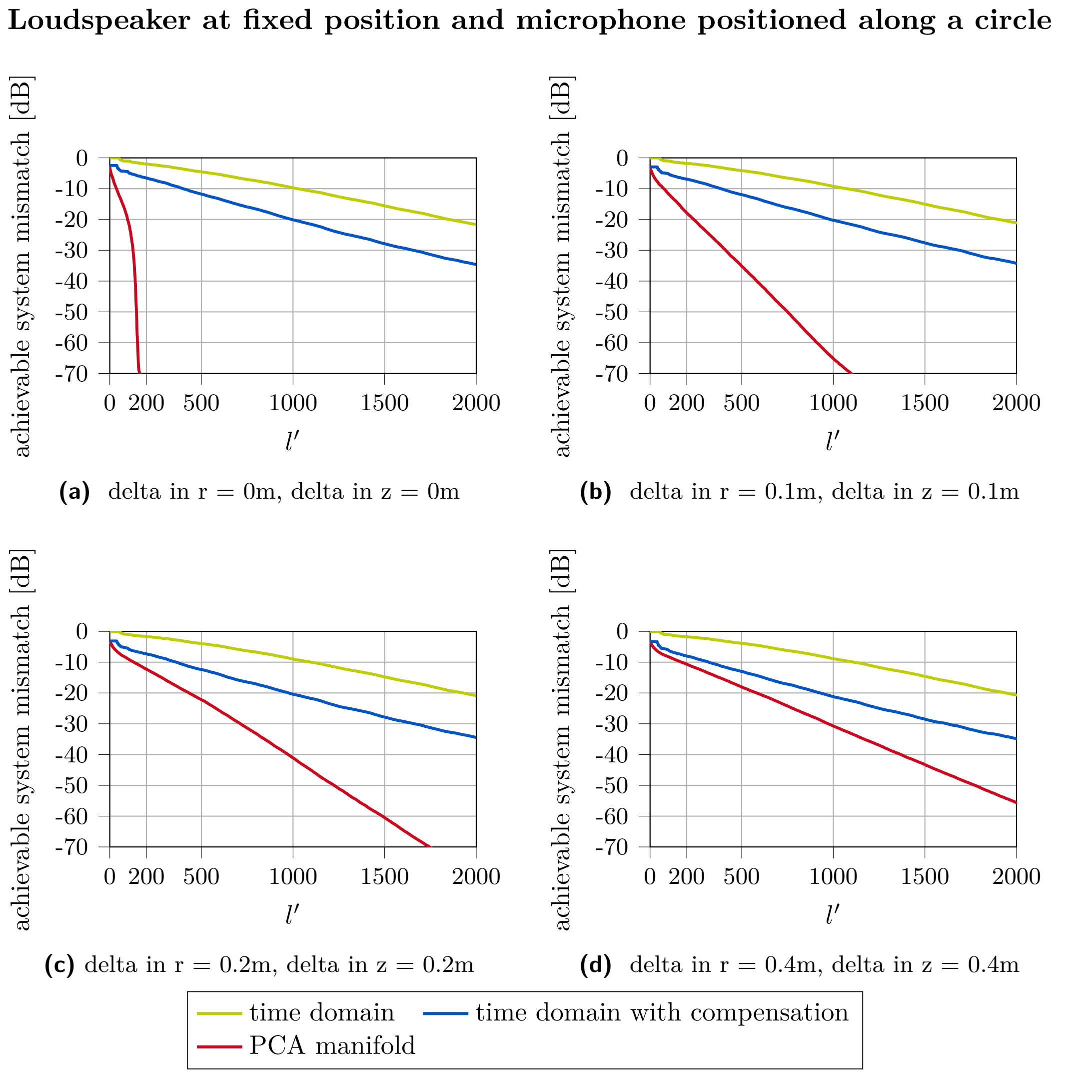
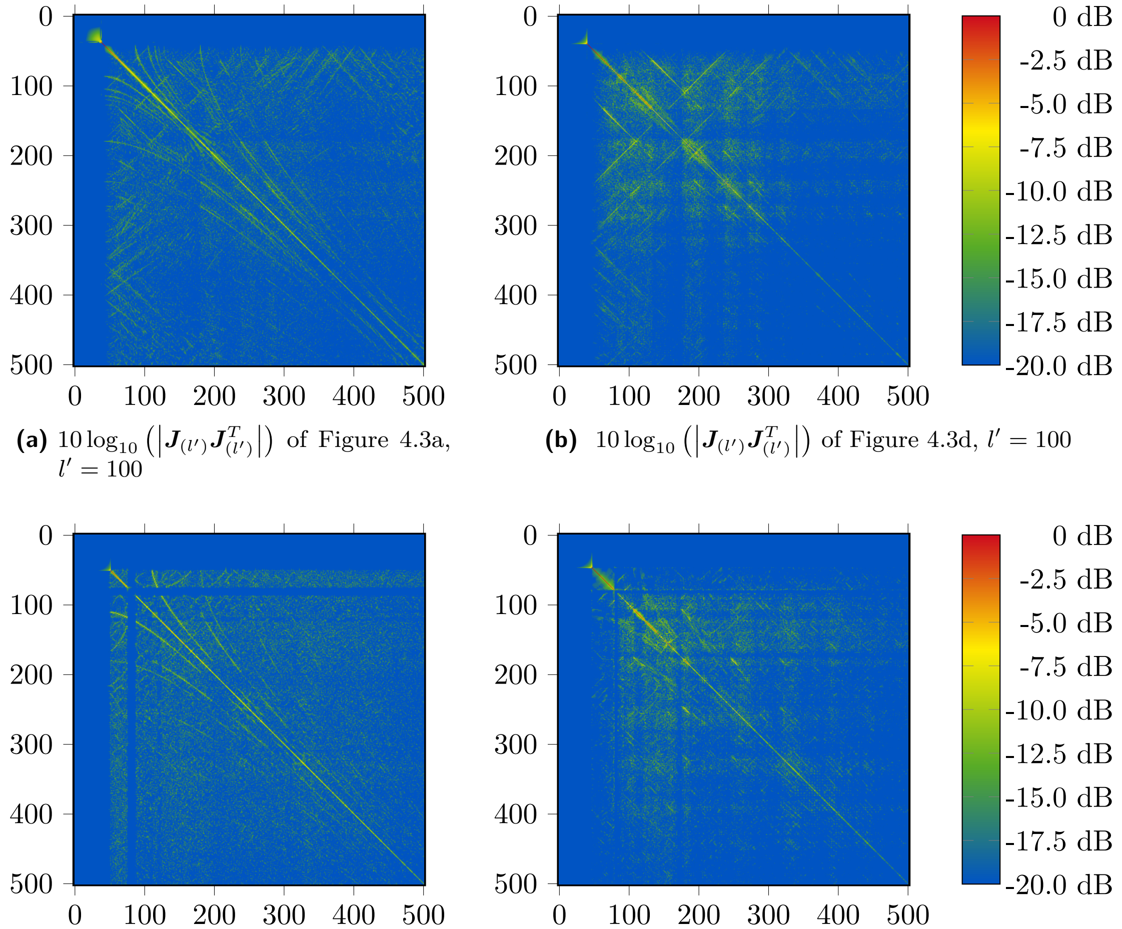


Figure 4.4: The achievable system mismatch over the number of components l' is shown for the time domain, for the time domain with delay compensation and consideration of the mean of the training data (time domain with compensation), and for the PCA manifold. The loudspeaker is at a fixed position and the microphone is randomly positioned on a circle around the loudspeaker with four cases of varying restrictiveness of the microphone position.

the Jacobian of the decoder (defined in equation (3.11)), which is constant over time for affine manifolds, and is therefore just denoted $\boldsymbol{J}$ here. For the visualization, the absolute value of each matrix element is taken, the logarithm is computed and is then coloured according to the colour bar. Values of plus or minus one are in the absolute logarithmic scale 0 dB and are coloured red. Numbers with a small absolute value of -20 dB and below are coloured blue. The matrix $\boldsymbol{J}_{(l')}\boldsymbol{J}_{(l')}^T$ has a size of 4000×4000, but for better visibility it is zoomed in to show only the matrix entries from the row and column with index 0 to 500.

The first observation is that there are more off diagonal entries that are above -20 dB for the case of Figure 4.5a and Figure 4.5c compared to Figure 4.5b and Figure 4.5d. There are



(a) $10\log_{10}\left(\left|\boldsymbol{J}_{(l')}\boldsymbol{J}_{(l')}^T\right|\right)$ of Figure 4.3a, $l' = 100$

(b) $10\log_{10}\left(\left|\boldsymbol{J}_{(l')}\boldsymbol{J}_{(l')}^T\right|\right)$ of Figure 4.3d, $l' = 100$

(c) $10\log_{10}\left(\left|\boldsymbol{J}_{(l')}\boldsymbol{J}_{(l')}^T\right|\right)$ of Figure 4.4a, $l' = 100$

(d) $10\log_{10}\left(\left|\boldsymbol{J}_{(l')}\boldsymbol{J}_{(l')}^T\right|\right)$ of Figure 4.4d, $l' = 100$

Figure 4.5: The matrix $\boldsymbol{J}_{(l')}\boldsymbol{J}_{(l')}^T$, which is the projection matrix of the PCA manifold, is shown in its absolute value and scaled logarithmically. The expression $10\log_{10}\left(\left|\boldsymbol{J}_{(l')}\boldsymbol{J}_{(l')}^T\right|\right)$ means here that the absolute value of each entry of the matrix $\boldsymbol{J}_{(l')}\boldsymbol{J}_{(l')}^T$ is taken, and then the logarithm is calculated and its value in dB is depicted according to the colour bar shown on the right. The matrix entries from 0 to 500 in each row and column are shown. The case for $l' = 100$ is shown with the same settings as in Figure 4.4 and Figure 4.3 for the cases (a) and (d), each as referenced.

lines that start parallel to the main diagonal and then curve outward by 45 degrees to the main diagonal, meaning they form a right angle to the outer edge of the matrix. Furthermore, there are lines that are parallel or perpendicular to the main diagonal. The outward curving lines are more pronounced for the case when the used scenario with $l' = 100$ components had a lower achievable system mismatch, as seen by comparing Figures 4.5a ($-67.0\,\text{dB}$ achievable system mismatch from the scenario of Figure 4.3a) and 4.5c ($-20.6\,\text{dB}$ from the scenario of Figure 4.4a) to Figures 4.5b ($-8.5\,\text{dB}$ from the scenario of Figure 4.3d) and 4.5d ($-8.3\,\text{dB}$ from the scenario of Figure 4.4d). In Figures 4.5a and 4.5b, there are slightly more of the parallel lines and the perpendicular lines compared to Figures 4.5a and 4.5b. These lines show for which time domain components some correlation is exploited by the PCA. Overall, Figures 4.5a and 4.5c show more lines and structures in the matrix compared to Figures 4.5b and 4.5a. This is expected since the manifold mismatch is smaller for the case in Figures 4.5a and 4.5c compared to Figures 4.5b and 4.5d, meaning that in the case of a smaller achievable system mismatch for some fixed number of principal components, the PCA exploits more correlation in the training data.

It it also observed that the values of the main diagonal line decay slower for larger matrix entry indices in Figures 4.5a and 4.5c compared to Figures 4.5b and 4.5d. Furthermore, in all figures the first entries of the matrices are all zero or below $-20\,\mathrm{dB}$. For Figures 4.5c and 4.5d, the first samples of the impulse response are guaranteed to be zero due to the delay between the loudspeaker and the microphone in the circle configuration. This also explains why Figure 4.5c has the largest section of entries that are zero or nearly zero starting from the upper left corner along the main diagonal line, because the minimum distance between the loudspeaker and microphone to generate the impulse responses is $1\,\mathrm{m}$ in Figure 4.5c and only $0.8\,\mathrm{m}$ in Figure 4.5c. However, in Figures 4.5a and 4.5b there is no guaranteed minimal distance between the loudspeaker and the microphone, but it seems that on average, there is at least some distance between the loudspeaker and the microphone indicated by the observation that the first 100 time domain components are set to zero by the PCA and are thus not relevant.

4.2 Manifold LMS algorithm

In this section, the manifold LMS algorithm is analysed for its convergence speed. The experiments in this section are done under the ideal condition that the true impulse responses perfectly lies on the manifold to be be able to examine the convergence speed in isolation, meaning that

$$\boldsymbol{w}_\star = \boldsymbol{f}_{\text{dec}}\left(\boldsymbol{f}_{\text{enc}}\left(\boldsymbol{w}_\star\right)\right), \tag{4.3}$$

which in the following is called, that the manifold vector rejection is zero.

A theoretical value for the convergence speed of the manifold LMS algorithm was derived in Section 3.5 under the assumption that the difference of the current estimate of the impulse response and the true impulse response $\boldsymbol{d}(k)$, and the input vector $\boldsymbol{x}(k)$ are statistically independent. The first experiment in this section will analyse in which cases this assumption is justified. Thereafter, the manifold LMS algorithm is tested for various parameter settings.

All simulations in this and the following sections are averaged over 10 independent runs, and the curves are slightly smoothed for better visibility.

4.2.1 General manifold

For the calculation of the convergence speed of the manifold LMS algorithm, some assumptions were made in the derivation. One assumption was that $\boldsymbol{d}(k)$ and $\boldsymbol{x}(k)$ are statistically independent. The time domain LMS algorithm is a special case of the manifold LMS algorithm with $\boldsymbol{J}$ being an identity matrix. It was shown for the LMS algorithm in [35], that the assumption of a statistical independence of $\boldsymbol{d}(k)$ and $\boldsymbol{x}(k)$ holds in good approximation, by showing that the theoretically derived convergence speed matches the convergence speed of the LMS algorithm well in a simulation. However, at this point it remains unclear if this assumption also holds true in the general case of the manifold LMS algorithm, which will be investigated in this section.

In a first step to investigate if the assumption of $\boldsymbol{d}(k)$ and $\boldsymbol{x}(k)$ being statistically independent is a valid assumption in the general case, the update formula of $\boldsymbol{d}(k)$ is analysed with an illustrative example. In a second step, simulations with different matrices $\boldsymbol{J}\boldsymbol{J}^T$ are conducted to analyse in which cases the convergence speed of a simulation of the manifold LMS algorithm matches the theoretically computed convergence speed.

The update equation of the error vector $\boldsymbol{d}$ for the linearised LMS algorithm is given as

$$\boldsymbol{d}(k+1) = \boldsymbol{d}(k) + \beta \boldsymbol{J}\boldsymbol{J}^T \boldsymbol{x}(k)e. \tag{4.4}$$

For $k = 2$, $\boldsymbol{d}(k)$ is

$$\begin{aligned}\boldsymbol{d}(2) =& \boldsymbol{d}(1) + \beta \boldsymbol{J}\boldsymbol{J}^T \boldsymbol{x}(1)e(1) && (4.5)\\ =& \boldsymbol{d}(0) + \beta \boldsymbol{J}\boldsymbol{J}^T \boldsymbol{x}(0)e(0) + \beta \boldsymbol{J}\boldsymbol{J}^T \boldsymbol{x}(1)e(1), && (4.6)\end{aligned}$$

with

$$\boldsymbol{x}(k) = \begin{bmatrix} x_k & x_{k+1} & \dots & x_{k+l'-1} \end{bmatrix}^T. \tag{4.7}$$

For the simple case of $l' = 3$, $\boldsymbol{d}(2)$ can be written as

$$\boldsymbol{d}(2) = \boldsymbol{d}(0) + \beta \boldsymbol{J}\boldsymbol{J}^T \begin{bmatrix} e(0)x_0 + e(1)x_1 \\ e(0)x_1 + e(1)x_2 \\ e(0)x_2 + e(1)x_3 \end{bmatrix}. \tag{4.8}$$

For easier notation, $\boldsymbol{J}\boldsymbol{J}^T$ is written as

$$\boldsymbol{J}\boldsymbol{J}^T = \boldsymbol{T} = \begin{bmatrix} t_{11} & t_{12} & t_{13} \\ t_{21} & t_{22} & t_{23} \\ t_{31} & t_{32} & t_{33} \end{bmatrix}, \tag{4.9}$$

which makes it possible to write $\boldsymbol{d}(2)$ as

$$\boldsymbol{d}(2) = \boldsymbol{d}(0) + \beta \begin{bmatrix} t_{11}\left(e(0)x_0 + e(1)x_1\right) + t_{12}\left(e(0)x_1 + e(1)x_2\right) + t_{13}\left(e(0)x_2 + e(1)x_3\right) \\ t_{21}\left(e(0)x_0 + e(1)x_1\right) + t_{22}\left(e(0)x_1 + e(1)x_2\right) + t_{23}\left(e(0)x_2 + e(1)x_3\right) \\ t_{31}\left(e(0)x_0 + e(1)x_1\right) + t_{32}\left(e(0)x_1 + e(1)x_2\right) + t_{33}\left(e(0)x_2 + e(1)x_3\right) \end{bmatrix}. \tag{4.10}$$

For simpler notation, $\boldsymbol{d}(2)$ is written as

$$\boldsymbol{d}(2) = \begin{bmatrix} d_0 & d_1 & d_2 \end{bmatrix}^T, \tag{4.11}$$

and $\boldsymbol{x}(2)$ is

$$\boldsymbol{x}(2) = \begin{bmatrix} x_2 & x_3 & x_4 \end{bmatrix}^T, \tag{4.12}$$

meaning that

$$\boldsymbol{d}(2)^T \boldsymbol{x}(2) = d_0 x_2 + d_1 x_3 + d_2 x_4. \tag{4.13}$$

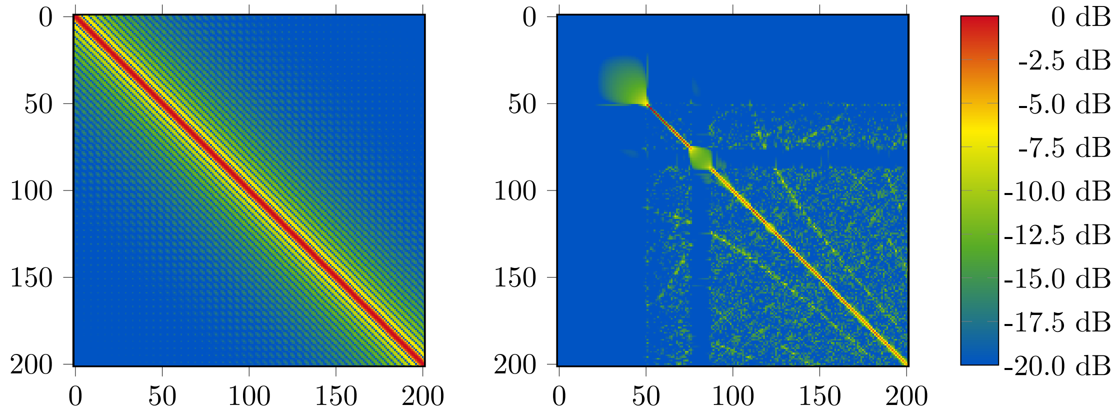
The expression $\boldsymbol{d}(2)^T \boldsymbol{x}(2)$ is compared for the case that $\boldsymbol{T}$ is an identity matrix and the general case of $\boldsymbol{T}$. To compute the expression $\boldsymbol{d}(2)^T \boldsymbol{x}(2)$, the terms $d_0 x_2$, $d_1 x_x$, and $d_2 x_4$

need to be evaluated. For the case that $\boldsymbol{T}$ is an identity matrix, i.e. $t_{11} = t_{22} = t_{33} = 1$ and all other entries of $\boldsymbol{T}$ are zero, d_0 is independent of x_2, d_1 is independent of x_3, and d_2 is independent of x_4. However, when $\boldsymbol{T}$ is a general matrix, d_0 contains the terms $t_{12}e(1)x_2$ and $t_{13}e(0)x_2$, making it dependent of x_2 in the calculation of d_0x_2 in (4.13). Similarly, d_1 also contains the term $t_{23}e(1)x_3$, making it dependent of x_3 in the calculation of d_1x_3 in (4.13). Furthermore, it is observed that the terms which make $\boldsymbol{d}(2)$ and $\boldsymbol{x}(2)$ not independent of each other are multiplied by the full step size β. Looking at this example, a hypothesis can be formulated that the assumption that $\boldsymbol{d}(k)$ and $\boldsymbol{x}(k)$ are statistically independent holds in good approximation when the off diagonal entries of $\boldsymbol{T}$ are small or when the full step size β is small, and that this assumption does not hold when the off diagonal entries of $\boldsymbol{T}$ are large and the full step size β is large. In the following, a simulation is done for two cases, first where $\boldsymbol{J}\boldsymbol{J}^T = \boldsymbol{T}$ has large values in its off diagonal entries, and second where $\boldsymbol{J}\boldsymbol{J}^T = \boldsymbol{T}$ has small values in its off diagonal entries. This allows to test the hypothesis by investigating if the theoretically calculated value of the convergence speed matches the convergence speed of the simulation in the two cases.

An example of an orthogonal affine manifold, where $\boldsymbol{J}\boldsymbol{J}^T = \boldsymbol{T}$ has large values in its off diagonal entries, is given when the matrix rows of $\boldsymbol{J}^T$ consist of sine and cosine oscillations. The row vector $\boldsymbol{j}_m^T$ is the m^{th} row of $\boldsymbol{J}^T$, and j_{mn} is the n^{th} entry of $\boldsymbol{j}_m^T$. The values of each row are constructed with $j_{mn} = \cos((m-1)/2\ 2\pi n/l)$ for the odd rows m, and $j_{mn} = \sin(m/2\ 2\pi n/l)$ for the even rows m, with $m = 1, \ldots, l'$ and $n = 1, \ldots, l$. A matrix constructed this way is orthogonal, and the rows of $\boldsymbol{J}$ are scaled to make all rows orthonormal. In the following, this example will be called sine manifold. For the second case an example of an orthogonal affine manifold, where $\boldsymbol{J}\boldsymbol{J}^T = \boldsymbol{T}$ has small values in its off diagonal entries, is when $\boldsymbol{J}\boldsymbol{J}^T$ is the projection matrix of the PCA manifold from the impulse response dataset $\mathcal{A}$.

Figure 4.6 shows the logarithm of the absolute value of each matrix entry of $\boldsymbol{J}\boldsymbol{J}^T$ from the sine manifold (Figure 4.6a) and from the PCA manifold (Figure 4.6b), scaled in a way that the maximum value is shown as 0 dB in colour red and small values of ≤ -20 dB are shown in blue. In the following experiment, $l = 1000$ and $l' = 250$ is used, meaning that $\boldsymbol{J}\boldsymbol{J}^T$ is a 1000×1000 matrix. The matrix entries from index 0 to 200 are shown for easier visibility. For the sine manifold in Figure 4.6a, it can be seen that an area around the main diagonal line has large values close to one, and that the magnitude of the values getting only slowly smaller for entries further away from the main diagonal line. For the PCA manifold in Figure 4.6b, only the entries directly on the main diagonal are close to one, whereas all off diagonal entries have small values. For the PCA manifold there are many points and lines further away from the main diagonal, but they are fairly small with about -7 dB and below.

The following experiments of this chapter evaluate different algorithms in the acoustic echo compensation task using simulations. The fundamentals of the acoustic echo compensation were described in Chapter 2, and are briefly recapped here. The input signal $\boldsymbol{x}(k)$, also called the far end signal, is emitted by the loudspeaker and is convoluted with the true impulse response from the loudspeaker to the microphone, which is then called the echo signal. A near end noise term $n(k)$ is added to the echo signal, where the noise power is set by the ENR. The echo signal together with the noise signal is the signal picked up by the microphone, and the estimated echo signal is subtracted from the microphone signal to minimise the residual echo signal by adaptively estimating the impulse response. The performance of the echo compensation is measured with the system mismatch, i.e. the normalised distance of the estimated impulse response to the true impulse response.



(a) $10\log_{10}\left(\left|\boldsymbol{J}_{(l')}\boldsymbol{J}_{(l')}^T\right|\right)$ of sine manifold

(b) $10\log_{10}\left(\left|\boldsymbol{J}_{(l')}\boldsymbol{J}_{(l')}^T\right|\right)$ of PCA manifold

Figure 4.6: The matrix $\boldsymbol{J}_{(l')}\boldsymbol{J}_{(l')}^T$, which is the projection matrix of the PCA manifold, is shown in its absolute value and scaled logarithmically. The expression $10\log_{10}\left(\left|\boldsymbol{J}_{(l')}\boldsymbol{J}_{(l')}^T\right|\right)$ means here that the absolute value of each entry of the matrix $\boldsymbol{J}_{(l')}\boldsymbol{J}_{(l')}^T$ is taken, and then the logarithm is calculated and its value in dB is depicted according to the colour bar shown on the right.

For the current simulation the manifold LMS algorithm and the time domain LMS algorithm are used to estimate the impulse response, and the results of the simulation are shown in Figure 4.7. Both the sine manifold and the PCA manifold are used with the step sizes of $\alpha = 1$ and $\alpha = 0.1$ each. The ENR is set to 10 dB. The number of components of the true impulse response is set to $l = 1000$ and the number of components for the manifold is set to $l = 250$. For the sine manifold, the true impulse response $\boldsymbol{w}_\star$ is created by mapping a standard Gaussian distributed random signal on the sine manifold. For the PCA manifold, the true impulse response $\boldsymbol{w}_\star$ is created by mapping an impulse response from the impulse response dataset $\mathcal{A}$ on the PCA manifold. The theoretical convergence speed and steady state system mismatch is also shown for each simulation.

In each panel of Figure 4.7, the theoretic convergence speed is shown as dashed lines. The formula for the convergence speed s of the manifold LMS algorithm for white noise excitation was derived in Section 3.5.4 with

$$s = 10\log_{10}\left(1 - \frac{\alpha}{l'}(2-\alpha)\right) f_s \left[\frac{\text{dB}}{\text{sec}}\right]. \tag{4.14}$$

The convergence speed of the manifold algorithm (4.14) only depends on the number of manifold components l' and the step size α of the LMS algorithm, making it essentially the same formula for the convergence speed as the time domain LMS algorithm, which also only depends on the number of time domain components l' and the step size α.

For the sine manifold, the achieved convergence speed of the simulation of the manifold LMS algorithm is way slower compared to the theoretically calculated convergence speed (Figure 4.7a). As discussed before, this can be explained because the assumption that $\boldsymbol{d}(k)$ and $\boldsymbol{x}(k)$ are uncorrelated does not hold for the sine manifold, and therefore the theoretically derived

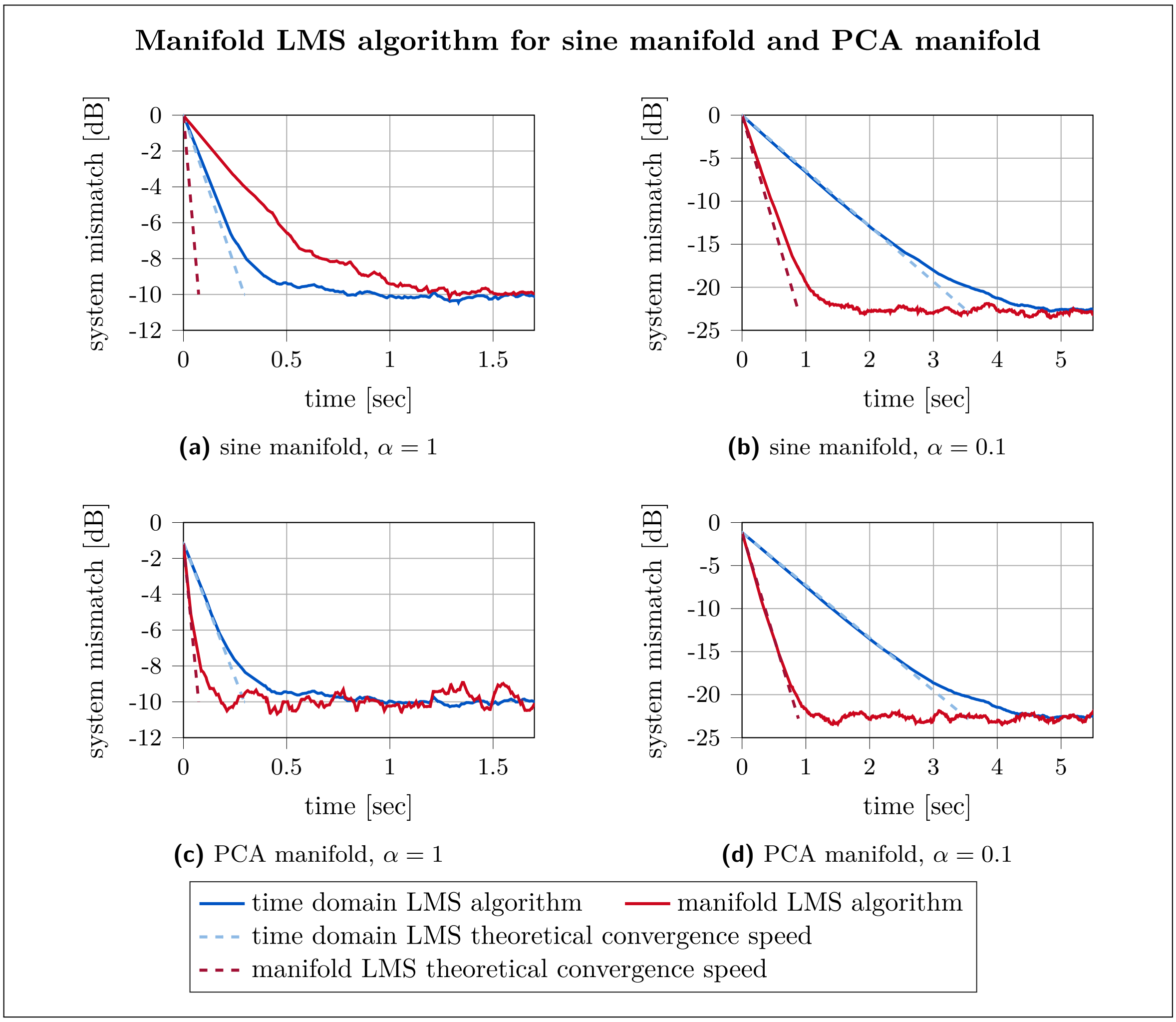


Figure 4.7: The system mismatch over the time for the manifold LMS algorithm and the time domain LMS algorithm is shown for the sine manifold and the PCA manifold. The input signal $\boldsymbol{x}(k)$ is a white noise signal, and the manifold vector rejection is zero. The ENR is 10 dB, and the step sizes are $\alpha = 1$ and $\alpha = 0.1$.

convergence speed does not necessarily align with the convergence speed of the simulation. In Figure 4.7b it is observed that for a smaller step size of $\alpha = 0.1$, the effect visible in Figure 4.7a, that the convergence speed of the simulation of the manifold LMS algorithm differs from the theoretically calculated convergence speed, is mostly vanished. Therefore, it can be concluded that the formula of the theoretically computed convergence speed holds for sufficiently small step sizes.

For the case of the PCA manifold, the manifold LMS algorithm in the simulation matches quite well with the theoretically calculated convergence speed for both $\alpha = 1$ (Figure 4.7c) and $\alpha = 0.1$ (Figure 4.7d). This shows that all assumptions for the calculation of the theoretical convergence speed are fulfilled in good approximation for the PCA manifold. This in turn means that for the application of a PCA manifold learned from simulated room impulse responses, it is reasonable to assume that $\boldsymbol{d}(k)$ and $\boldsymbol{x}(k)$ are statistically independent.

4.2.2 Manifold learned from impulse response dataset

In the previous section it was observed that the theoretically computed convergence speed of the manifold LMS algorithm is in good agreement with the simulations, when the manifold is a PCA manifold learned from an impulse response dataset. In this section, the manifold LMS algorithm is tested for three different step sizes α and different ENR.

In the simulations of this section, the impulse response dataset $\mathcal{A}$ is used for the PCA manifold, the number of components of the true impulse response is set to $l = 1000$, the number of PCA manifold components is $l' = 250$, and the manifold vector rejection is set to zero.

In each panel of Figure 4.8, the theoretic convergence speed and the theoretic steady state system mismatch is shown as dashed lines. Since the manifold vector rejection is zero, the theoretic steady state system mismatch can be calculated by the formula of the time domain LMS algorithm from [35] with

$$\frac{\mathrm{E}\left\{\|\boldsymbol{d}_\infty\|_2^2\right\}}{\|\boldsymbol{w}_\star\|_2^2} = \frac{\alpha}{2-\alpha}\left(\frac{\sigma_n^2}{\|\boldsymbol{w}_\star\|_2^2\,\sigma_x^2}\right), \tag{4.15}$$

which depends for white noise excitation only on the ENR and the step size α, because

$$\mathrm{ENR} = 10\log_{10}\left(\frac{\mathrm{E}\left\{\left(\boldsymbol{w}_\star^T\boldsymbol{x}(k)\right)^2\right\}}{\mathrm{E}\left\{n(k)^2\right\}}\right) = \frac{\|\boldsymbol{w}_\star\|_2^2\,\sigma_x^2}{\sigma_n^2}, \tag{4.16}$$

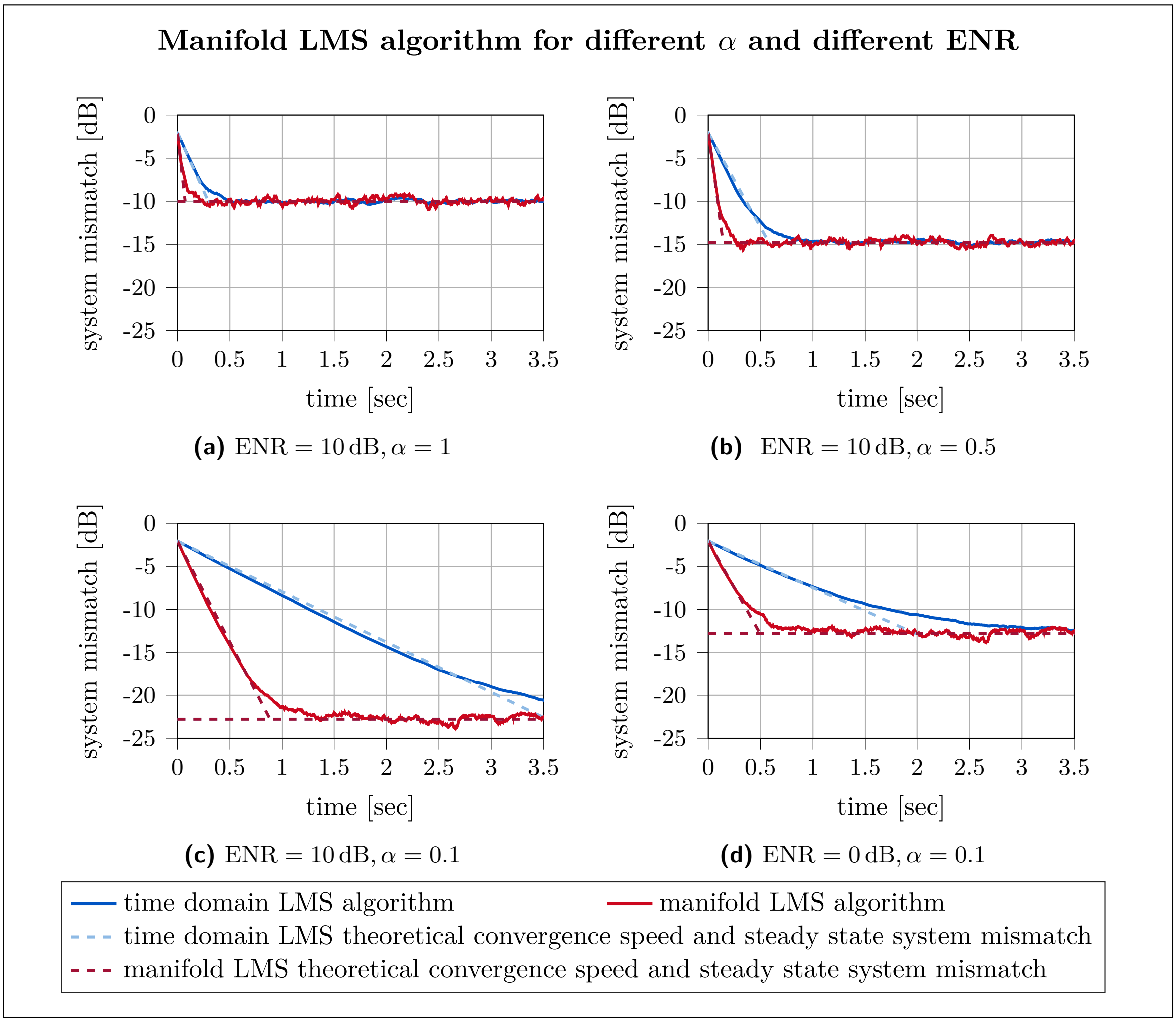
when $\boldsymbol{x}(k)$ and $n(k)$ are a stationary white noise processes and σ_x^2 and σ_n^2 is the signal energy of $\boldsymbol{x}(k)$ and $n(k)$, respectively.

For a fair comparison of the manifold LMS algorithm and the time domain LMS algorithm, the latter is initialised with the mean value of the training data, which corresponds to the zero state $\boldsymbol{z} = \mathbf{0}$ on the PCA manifold. Therefore, the manifold LMS algorithm and the time domain LMS algorithm start at a system mismatch of less than 0 dB on average. This initialisation will be used for this and the following simulations.

The general influence of the step size α for the NLMS algorithm can be evaluated when comparing Figures 4.8a, 4.8b, and 4.8c, where the step size is varied with $\alpha = 1$, $\alpha = 0.5$, and $\alpha = 0.1$. For larger step size α values up to 1, the convergence speed of the manifold LMS algorithm is faster. For even larger values of α, the convergence is slower compared to the case of $\alpha = 1$ (not show here). Furthermore, a reduction in step size α causes the steady state system mismatch to become smaller. When comparing Figures 4.8a, 4.8b, and 4.8c, the system mismatch is the smallest for $\alpha = 0.1$ (Figure 4.8c). In conclusion, the step size of the LMS algorithm gives a trade off between a faster initial convergence speed, and a lower steady state system mismatch. The same holds for the time domain LMS algorithm.

When the level of the near end noise compared to the echo signal is larger, i.e. when the ENR is smaller, the convergence speed does not change, but the steady state system mismatch becomes worse, as seen in the comparison of the simulation with ENR = 10 dB in Figure 4.8c with ENR = 0 dB in Figure 4.8d.

Generally, in all shown cases in Figure 4.8, the theoretically computed convergence speed matches the convergence speed of the manifold LMS algorithm quite well. The next step in the following section is to analyse the manifold NLMS algorithm to investigate, if the theoretically computed convergence speed and steady state system mismatch also aligns with the simulations of the manifold NLMS algorithm.



(a) $\mathrm{ENR} = 10\,\mathrm{dB}, \alpha = 1$ **(b)** $\mathrm{ENR} = 10\,\mathrm{dB}, \alpha = 0.5$

(c) $\mathrm{ENR} = 10\,\mathrm{dB}, \alpha = 0.1$ **(d)** $\mathrm{ENR} = 0\,\mathrm{dB}, \alpha = 0.1$

Figure 4.8: The system mismatch over the time for the manifold LMS algorithm and the time domain LMS algorithm is shown for different step sizes α and ENR. The input signal $\boldsymbol{x}(k)$ is a white noise signal, and the manifold vector rejection is zero. The number of components of the true impulse response is $l = 1000$, the number of time domain components is $l' = 1000$, and the number of PCA manifold components is $l' = 250$.

4.3 Manifold NLMS algorithm

This section analyses the manifold NLMS algorithm for various scenarios. In Section 4.3.1, the same parameter settings are chosen as in Section 4.2.2 to allow a direct comparison of the manifold NLMS algorithm and the manifold LMS algorithm. Thereafter, the convergence speed for different numbers of PCA manifold components l' and the steady state system mismatch is investigated for the case that the manifold vector rejection is not zero. Furthermore, the convergence behaviour is simulated in the case when a speech input signal is used.

4.3.1 White noise excitation with zero manifold vector rejection

Two simulation experiments are conducted in this section, both with the impulse response dataset $\mathcal{A}$ with white noise input signals and with zero manifold vector rejection. The number of components is varied for the PCA manifold from $l' = 100$ to $l' = 750$, and the number

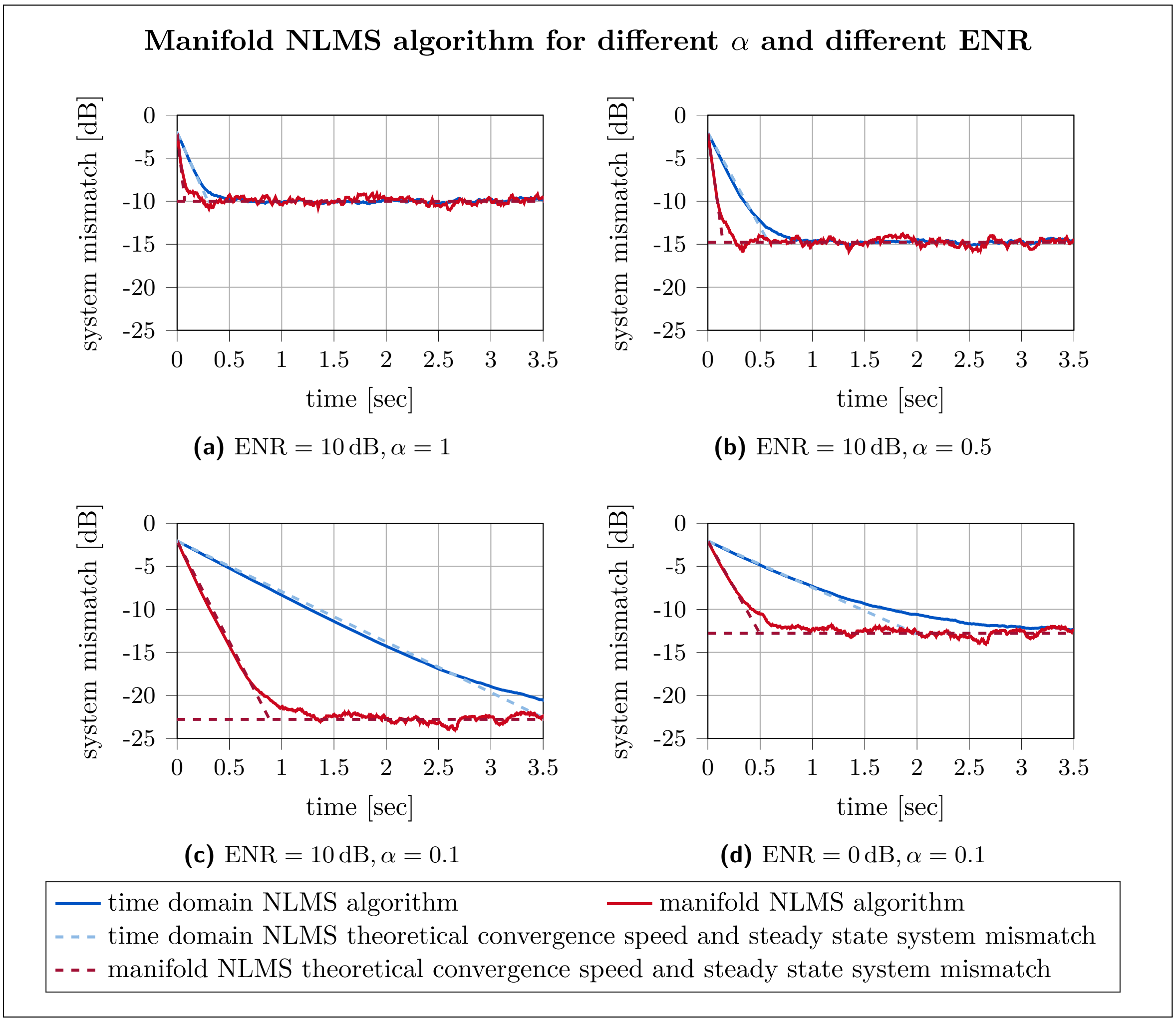


Figure 4.9: The system mismatch over the time for the manifold NLMS algorithm and the time domain NLMS algorithm is shown for different step sizes α and ENR. The input signal $\boldsymbol{x}(k)$ is a white noise signal, and the manifold vector rejection is zero. The number of components of the true impulse response is $l = 1000$, the number of time domain components is $l' = 1000$, and the number of PCA manifold components is $l' = 250$.

of time domain components is the same as the number of components of the true impulse response, i.e. $l' = l = 1000$.

In Section 4.2.2, the manifold LMS algorithm was analysed for different step sizes and different ENR. The same simulation setup is repeated for the manifold NLMS algorithm and the time domain NLMS algorithm. Since the convergence speed of time domain NLMS and time domain LMS algorithm are quite similar for white noise input [38, 35], it is expected that this also holds for the manifold NLMS algorithm and the manifold LMS algorithm.

As in the previous section, the convergence speed and the steady state system mismatch in the simulations are compared to their theoretically predicted values. Figure 4.9 shows the system mismatch of the manifold NLMS algorithm and the time domain NLMS algorithm for the same parameter settings of the step sizes α and the ENR as in the previous experiment. In comparison, Figures 4.9 and 4.8 look nearly identical. This verifies the expected behaviour

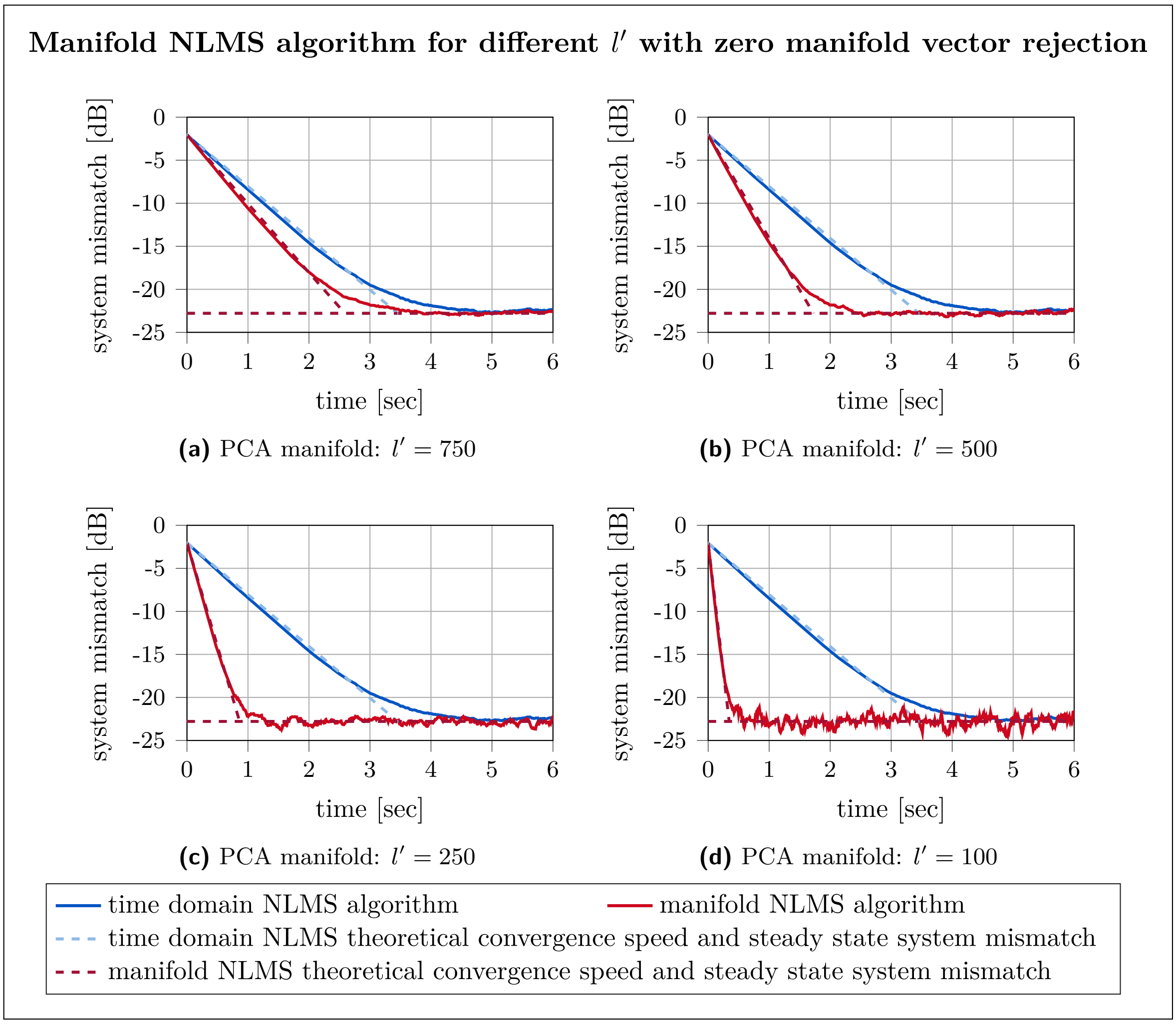


Figure 4.10: The system mismatch over the time for the manifold NLMS algorithm and the time domain NLMS algorithm is shown for different number of PCA manifold components l'. The input signal $\boldsymbol{x}(k)$ is a white noise signal, and the manifold vector rejection is zero. The number of components of the true impulse response is $l = 1000$, the number of time domain components is $l' = 1000$, and the number of PCA manifold components is varied with $l' = 750$, $l' = 500$, $l' = 250$, and $l' = 100$. The step size is $\alpha = 0.1$, and the ENR is 10 dB.

that the convergence speed of the manifold LMS and manifold NLMS algorithm are similar for white noise input. For input signals that are correlated, like speech signals, both the LMS and NLMS algorithm are known to converge slower [35]. It is also known from literature that the NLMS algorithm is less sensitive to correlated input signals compared to the LMS algorithm [1]. Therefore, only the manifold NLMS algorithm will be used in the next simulations.

Since the convergence speed (4.14) of the manifold NLMS algorithm mostly depends on the number of PCA manifold components, the next simulation compares the manifold NLMS for different number of PCA manifold components l'. The results are depicted in Figure 4.10. When l' is small (Figure 4.10d), the steady state system mismatch fluctuates, because the full step size $\beta = \alpha / \|x\|_2^2$ depends on the number of components. When the number of components is small, the full step size β is large, meaning that the fluctuation in the steady state system mismatch will be large as well.

From the observation of Figure 4.10 it can be concluded that using the manifold NLMS algorithm instead of the time domain NLMS algorithm with the aim to achieve a faster convergence speed, it makes most sense when the number of PCA manifold components is significantly lower compared to the number of time domain components.

4.3.2 White noise excitation in the general case

When the manifold NLMS algorithm or the NLMS algorithm is used in practice, the number of components of the algorithm needs to be chosen as a compromise between computational complexity, the convergence speed, and the achievable system mismatch [35]. For more components l', the manifold NLMS algorithm and the time domain NLMS algorithm need to do more computations to update the larger numbers of components. In the formula of the convergence speed for both the manifold and time domain NLMS algorithm in equation (4.14) it can be seen that less components increase the convergence speed. The achievable system mismatch decreases with a higher number of components, which was shown in Section 4.1 for different acoustic scenarios. However, the achievable system mismatch is only a lower bound for the system mismatch of the algorithm, meaning that near end noise or a larger step size α lead to a higher system mismatch than the achievable system mismatch in the simulation. In general, this makes choosing an appropriate number of components challenging. For a fair comparison of the manifold NLMS algorithm and the time domain NLMS algorithm, the latter uses the time domain with delay compensation and consideration of the mean of the training data. This means that both the manifold NLMS algorithm and the time domain NLMS algorithm use the same initialisation and therefore start with the same system mismatch, which is below 0 dB on average. This fair comparison of the algorithms is done for all following simulations. The achievable system mismatch for the manifold NLMS algorithm and the time domain NLMS algorithm can be read off Figure 4.1.4.

In order to investigate the change in convergence speed and steady state system mismatch for different numbers of components of the time domain NLMS algorithm and the manifold NLMS algorithm, a simulation is run for two different impulse response datasets for varying numbers of components.

In order to analyse the influence of different acoustic scenarios, impulse response dataset $\mathcal{A}$ (circle without variation, see Section 4.1.4) and dataset $\mathcal{B}$ (circle with variation, see Section 4.1.4) is used in the simulations. The true impulse response is set to a length of $l = 4000$, and the number of components for the time domain and the PCA manifold is varied to include a small and a large number of components each for dataset $\mathcal{A}$ and dataset $\mathcal{B}$. Furthermore, the input signal $\boldsymbol{x}(k)$ is a white noise signal, the step size is $\alpha = 0.1$, and $\mathrm{ENR} = 10\,\mathrm{dB}$. As showed in Section 4.1.4, dataset $\mathcal{A}$ yields a large increase for the achievable system mismatch for the PCA manifold compared to the time domain, whereas the effect is less significant for dataset $\mathcal{B}$. Therefore, the number of components in the experiment are set to $l' = 1000$ (Figure 4.11a) and $l' = 250$ (Figure 4.11b) for dataset $\mathcal{A}$, and $l' = 1000$ (Figure 4.11c) and $l' = 500$ (Figure 4.11d) for dataset $\mathcal{B}$.

The formula for the steady state distance from the true impulse response was derived in Section 3.5.5 and given in equation (3.90). The expected value of the steady state system mismatch of the manifold NLMS algorithm for white noise excitation is given as

$$\frac{\mathrm{E}\left\{\|\boldsymbol{d}_\infty\|_2^2\right\}}{\|\boldsymbol{w}_\star\|_2^2} = \frac{\|\boldsymbol{\Delta}_\mathcal{M}\|_2^2}{\|\boldsymbol{w}_\star\|_2^2} + \frac{\alpha}{2-\alpha}\left(\frac{\|\boldsymbol{\Delta}_\mathcal{M}\|_2^2}{\|\boldsymbol{w}_\star\|_2^2} + \frac{\sigma_n^2}{\sigma_x^2\,\|\boldsymbol{w}_\star\|_2^2}\right) \tag{4.17}$$

$$= \frac{2}{2-\alpha}\frac{\|\boldsymbol{\Delta}_\mathcal{M}\|_2^2}{\|\boldsymbol{w}_\star\|_2^2} + \frac{\alpha}{2-\alpha}\frac{\sigma_n^2}{\sigma_x^2\,\|\boldsymbol{w}_\star\|_2^2}. \tag{4.18}$$

The term $\frac{\|\boldsymbol{\Delta}_\mathcal{M}\|_2^2}{\|\boldsymbol{w}_\star\|_2^2}$ is the achievable system mismatch, which was discussed in Section 3.5.5. In equation (4.17) it can be seen that the achievable system mismatch is in fact a lower bound for the steady state system mismatch. Furthermore, equation (4.18) depends on the ENR and the step size α. For example, with the simulation setting of ENR $= 10\,\mathrm{dB}$ and $\alpha = 0.1$ (Figure 4.11) the second term of equation (4.18) computes to

$$10\log_{10}\left(\frac{\alpha}{2-\alpha}\frac{\sigma_n^2}{\sigma_x^2\,\|\boldsymbol{w}_\star\|_2^2}\right) \approx -22.8\,\mathrm{dB}. \tag{4.19}$$

When the number of components is rather large with $l' = 1000$ (Figures 4.11a an 4.11c), the achievable system mismatch $\frac{\|\boldsymbol{\Delta}_\mathcal{M}\|_2^2}{\|\boldsymbol{w}_\star\|_2^2}$ is small, as already shown and discussed in Section 4.1. For example, in Figure 4.4 it was observed that for both the impulse response dataset $\mathcal{A}$ and dataset $\mathcal{B}$, the achievable system mismatch is about $-20\,\mathrm{dB}$ for $l' = 1000$ time domain components and below $-50\,\mathrm{dB}$ for $l' = 1000$ PCA manifold components.

In Figure 4.11, the expected steady state system mismatch (4.17) is shown with a dashed horizontal line for both the time domain and manifold NLMS algorithm. Analysing equation (4.11) with the value of (4.19) and $\frac{\|\boldsymbol{\Delta}_\mathcal{M}\|_2^2}{\|\boldsymbol{w}_\star\|_2^2} \approx -20\,\mathrm{dB}$, the expected steady state system mismatch for the time domain NLMS algorithm is about $-18.5\,\mathrm{dB}$ for both Figures 4.11a an 4.11c. For the PCA manifold the achievable system mismatch is much smaller than the computed value of (4.19), meaning that the expected steady state system mismatch of the manifold NLMS algorithm is about $-22.8\,\mathrm{dB}$ for both Figures 4.11a and 4.11c. As a general observation, the steady state system mismatch of the simulations matches the expected steady state system mismatch.

Figure 4.11b shows the results for $l' = 250$ components and impulse response dataset $\mathcal{A}$. Compared to Figure 4.11a, the convergence speed is about eight times faster, which corresponds to the expectation, since the number of components is eight times smaller. The relationship of the ratio of the convergence speeds to the ratio of the number of components can be seen in equation (4.14). The ratio of the convergence speed with l'_a and l'_b components can be written in an approximation as

$$\frac{10\log_{10}\left(1-\frac{\alpha}{l'_a}(2-\alpha)\right)f_s}{10\log_{10}\left(1-\frac{\alpha}{l'_b}(2-\alpha)\right)f_s} = \frac{\log_{10}\left(1-\frac{\alpha}{l'_a}(2-\alpha)\right)}{\log_{10}\left(1-\frac{\alpha}{l'_b}(2-\alpha)\right)} \approx \frac{-\frac{\alpha}{l'_a}(2-\alpha)}{-\frac{\alpha}{l'_b}(2-\alpha)} = \frac{l'_b}{l'_a}, \tag{4.20}$$

which is simply the ratio of the number of components. The approximation in (4.20) uses that $\log_{10}(1+x) \approx x$ for small x in the absolute value, which is applicable since $\frac{\alpha}{l'}(2-\alpha)$ is usually small in the absolute value.

Even in the case of only $l' = 250$ PCA manifold components, the achievable system mismatch is much smaller than the computed value of (4.19), meaning that the expected steady state

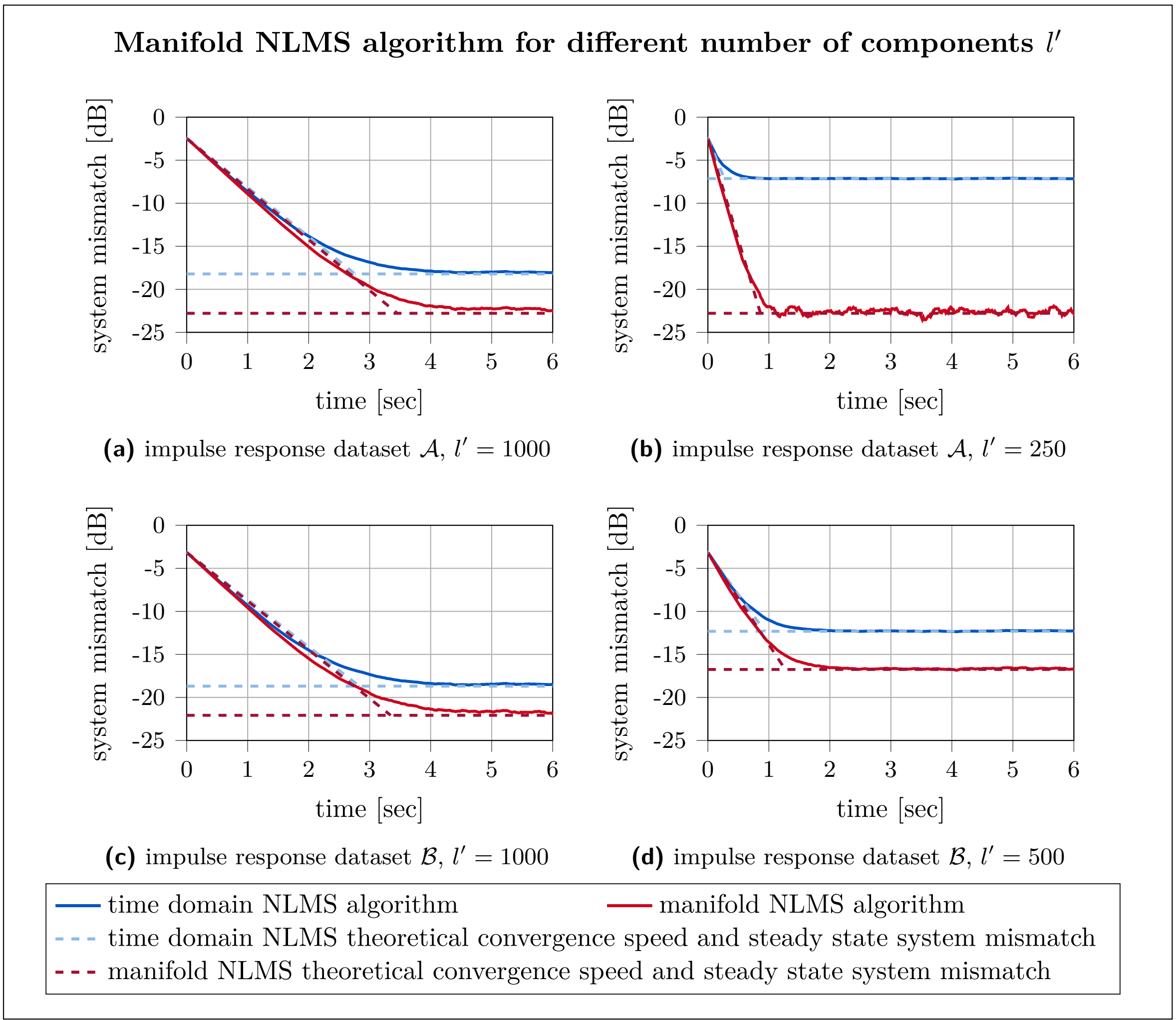


Figure 4.11: The system mismatch over the time for the manifold NLMS algorithm and the time domain NLMS algorithm is shown for the PCA manifold of dataset $\mathcal{A}$ (circle with no variation) and dataset $\mathcal{B}$ (circle with variation). The input signal $\boldsymbol{x}(k)$ is a white noise signal, and the number of components of the true impulse response is $l = 4000$. The number of time domain components and PCA manifold components is $l' = 1000$ and $l' = 250$ for dataset $\mathcal{A}$, and $l' = 1000$ and $l' = 500$ for dataset $\mathcal{B}$.

system mismatch of the manifold NLMS algorithm is the same as for $l' = 1000$ components, as observed in Figures 4.11b and 4.11a. However, for the time domain the achievable steady state system mismatch is significantly larger for $l' = 250$, resulting in an expected steady state system mismatch of only $-7.1\,\mathrm{dB}$.

Figure 4.11d shows the case for $l' = 500$ and impulse response dataset $\mathcal{B}$, where the microphone had a higher variability compared to dataset $\mathcal{A}$ leading to a larger achievable system mismatch for the same number of PCA manifold components, as discussed in Section 4.1.4. Comparing Figures 4.11c and 4.11b, the convergence speed of the manifold NLMS algorithm is about four times faster, because the number of components is four times smaller. The achievable system mismatch for $l' = 500$ PCA manifold components (Figure 4.11d) is higher than the lower bound calculated in equation (4.19), and the expected steady state system mismatch is about $-16.7\,\mathrm{dB}$. For the time domain, the steady state system mismatch is larger compared to the PCA manifold which is about $-12.3\,\mathrm{dB}$ for $l' = 500$ time domain components.

4.3.3 Speech excitation

In practical applications of voice communication systems, primarily speech signals are transmitted, meaning that the estimation of the impulse response needs to use a speech signal as an excitation signal $\boldsymbol{x}(k)$. For this reason the manifold NLMS algorithm is simulated with speech input signals in this section. For the manifold and time domain NLMS algorithm, two small modification are made. The formula of the full step size β is altered and a block processing version of the NLMS algorithm is used. In all simulations in this chapter, the used speech signals are from the CSTR VCTK Corpus [37]. In contrast to the case when the excitation signal is a white noise signal, speech signals as excitation signals make it not possible to calculate convergence speed and steady state system mismatch in a closed form solution.

For the NLMS algorithm, the full step size β is computed as

$$\beta = \frac{\alpha}{\|\boldsymbol{x}(k)\|_2^2}, \tag{4.21}$$

where $\boldsymbol{x}(k)$ is the excitation signal. For speech signals there can be periods of silence between the spoken words, which means that $\|\boldsymbol{x}(k)\|_2^2$ can potentially be very small, meaning that β can become very large and severely disturb the estimation of the impulse response. To prevent this problem of speech signals, the manifold and time domain NLMS algorithm compute the full step size with

$$\beta = \frac{\alpha}{\sigma_n^2 + \|\boldsymbol{x}(k)\|_2^2}, \tag{4.22}$$

where σ_n^2 is the near end noise power.

As already discussed, the computational complexity is higher for a larger number of components l'. One way to reduce the computational complexity is to use block processing [35]. In this case, multiple updates for the estimation of the impulse response are computed without waiting for a new measurement of the echo signal from the microphone. This block of updates of the estimation can be written as a convolution, which can be efficiently computed in the frequency domain using the fast Fourier transform. For the simulation with speech signals, the manifold NLMS and the time domain NLMS algorithm are simulated using a frame shift of 64, meaning that a block of 64 updates is done at the same time. The block processing NLMS algorithm compared to the NLMS algorithm without block processing achieves very similar results in general with a slightly lower steady state system mismatch [35]. For these discussed reasons of the reduced computational complexity and the only slightly worse performance, the block processing is often used in practice, and is also used in all following simulations. Note that in the previous sections the simulations were compared to the theoretical steady state system mismatch of the algorithms, which assumed that no block processing is used, meaning that the simulations were run without block processing for a fair comparison.

Correlated input signals like speech signals are known to slow down the convergence of the LMS and NLMS algorithm compared to a white noise input signals [35]. This observation also holds for the manifold NLMS algorithm. Compared to the results in the previous section, it can be seen that the system mismatch for the same number of components l' at e.g. 10

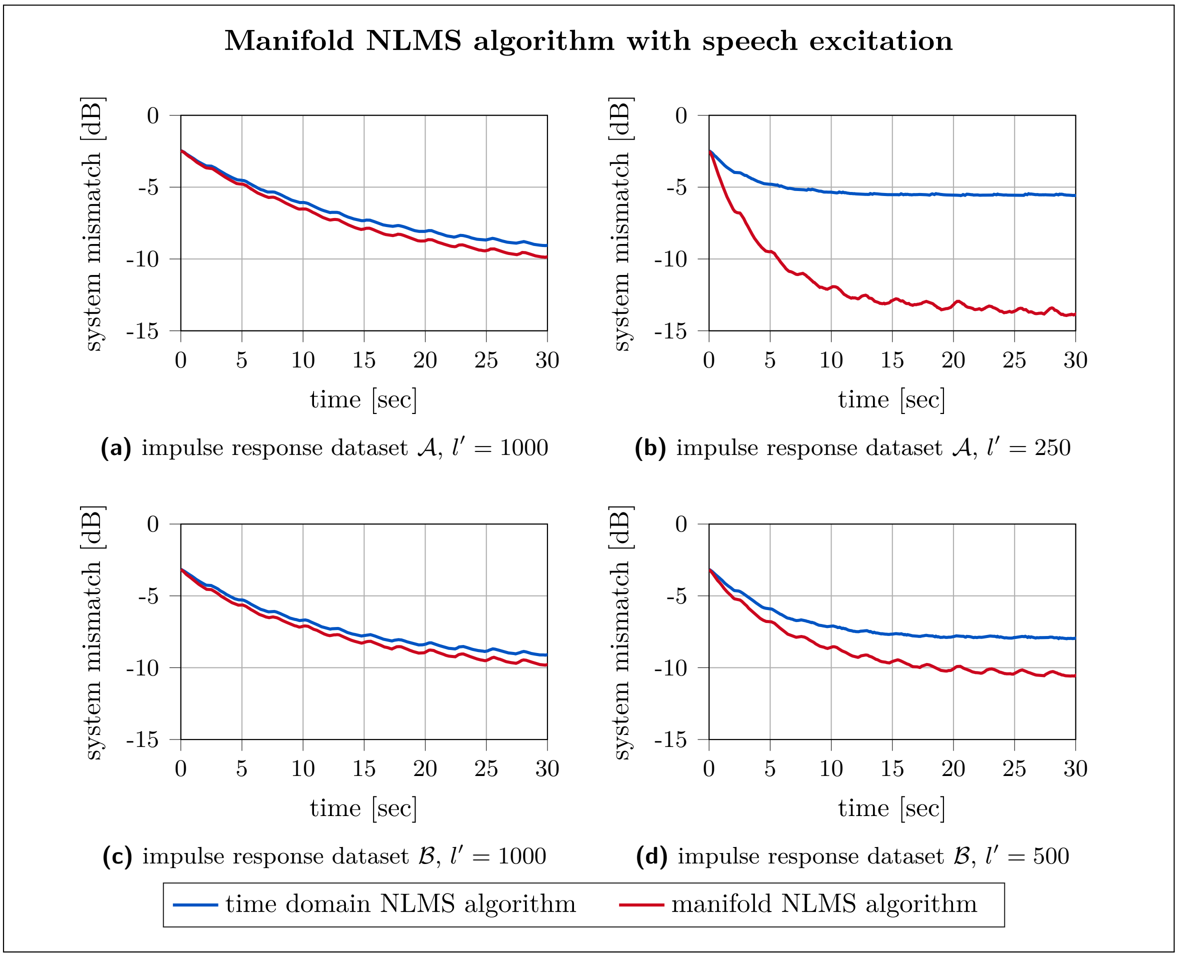


Figure 4.12: The system mismatch over the time for the manifold NLMS algorithm and the time domain NLMS algorithm is shown for the PCA manifold of dataset $\mathcal{A}$ (circle with no variation) and dataset $\mathcal{B}$ (circle with variation). The input signal $\boldsymbol{x}(k)$ is a speech signal, and the number of components of the true impulse response is $l = 4000$. The number of time domain components and PCA manifold components is $l' = 1000$ and $l' = 250$ for dataset $\mathcal{A}$, and $l' = 1000$ and $l' = 500$ for dataset $\mathcal{B}$.

seconds is significantly higher when $\boldsymbol{x}(k)$ is a speech signal compared to the case when $\boldsymbol{x}(k)$ is white noise, as observed in Figures 4.12 and 4.9.

The number of components of the true impulse response is $l = 4000$, the input signals $\boldsymbol{x}(k)$ are speech signals, the step size is $\alpha = 0.1$, and ENR $= 10\,\text{dB}$. Figures 4.12a and 4.12c show the results for $l' = 1000$. Regardless of using dataset $\mathcal{A}$ or dataset $\mathcal{B}$, the manifold and time domain NLMS algorithm achieve a system mismatch of about $-8\,\text{dB}$ after 30 seconds, and the manifold NLMS algorithm has a slightly lower system mismatch compared to the time domain NLMS algorithm. When the number of components is reduced to $l' = 250$ (Figure 4.12b), the convergence speed is vastly improved and about $-9.5\,\text{dB}$ is achieved after about 5 seconds for the manifold NLMS algorithm. The manifold NLMS algorithm in the case of $l' = 500$ components with dataset $\mathcal{B}$ (Figure 4.12d) achieves a system mismatch of about $-10\,\text{dB}$ after about 18 seconds. The time domain NLMS algorithm can not reach a low system mismatch because the achievable system mismatch is high for the case of $l' = 250$ time domain components (Figure 4.12b) and for $l' = 500$ time domain components (Figure 4.12d).

4.4 Manifold Kalman filter

In the following section the manifold Kalman filter and the time domain Kalman filter are compared. The Kalman filter is a generalization of the NLMS algorithm and offers an adaptive step size based on the error covariance estimate $\boldsymbol{P}(k)$, which is an estimate of the uncertainty of the current estimate of the impulse response [28]. In comparison, the NLMS algorithm has a constant step size. By adaptively changing the step size, the Kalman filter can achieve faster convergence speed and a lower steady state system mismatch than the NLMS algorithm.

The manifold Kalman filter is also compared with the projection Kalman filter, which was proposed in [13]. The projection Kalman filter updates the prediction of the impulse response in the time domain and then projects it onto the PCA manifold.

The simulations for the manifold Kalman filter are structured similarly to those in the previous section, with the analysis being performed for three cases, first with white noise input signal and zero manifold vector rejection, second for white noise input with manifold vector rejection, and third for speech input signals with manifold vector rejection.

For all simulations, unless stated otherwise, the true impulse response is not changing over time, meaning that for the Kalman filter state update equations, γ is set to 1, and for $\boldsymbol{\Psi}_{\Delta\Delta} = \sigma_{\Delta}^2 \boldsymbol{I}$, the variance σ_{Δ}^2 is set to 0. With these parameters, the Kalman filter can theoretically achieve a steady state system mismatch of negative infinity if $l' = l$ [7].

4.4.1 White noise excitation for zero manifold vector rejection

The experiment in this section compares two different initialisations of the error covariance $\boldsymbol{P}_{\text{init}}$. The simulation uses $l = 1000$ components for the true impulse response, $l' = 1000$ time domain components, and $l' = 250$ or $l' = 50$ PCA manifold components. The projection Kalman filter runs in the time domain with also $l' = 1000$ components and projects onto the same PCA manifold which is used for the manifold Kalman filter. In each simulation, the true impulse response is mapped onto the manifold to have a zero manifold vector rejection in order to investigate the convergence speed of the manifold Kalman filter in isolation. The input signal $\boldsymbol{x}(k)$ is a white noise signal, the manifold vector rejection is zero, the impulse response dataset $\mathcal{A}$ is used, and an ENR of 10 dB is used.

The initialisation is often set to a constant multiplied by the identity matrix. In the following simulation $\boldsymbol{P}_{\text{init}}$ is set to $0.01\boldsymbol{I}$ (where $\boldsymbol{I}$ denotes the identity matrix), which was found to give good results in preliminary tests. In this case, there is training data of the impulse responses available, meaning that the covariance matrix of the training data with zero mean gives the average distance of each component to an impulse response in the training data. Using this covariance matrix as $\boldsymbol{P}_{\text{init}}$ therefore gives the best possible initialisation on average, and is therefore called the average best initialisation in the following. For the manifold Kalman filter the average best initialisation takes the covariance matrix of the training data mapped onto the manifold as $\boldsymbol{P}_{\text{init}}$. These two different cases of initialisation for $\boldsymbol{P}(k)$ are compared with $\boldsymbol{P}_{\text{init}} = 0.01\boldsymbol{I}$ in Figures 4.13a and 4.13b, and $\boldsymbol{P}_{\text{init}}$ set to the average best initialisation in Figures 4.13c and 4.13d. It is observed that the different initialisations make a significant difference in the convergence speed of all three investigated algorithms with the average best initialisation achieving a smaller system mismatch after significantly less time. It is an important observation that the time domain Kalman filter with $l' = 1000$ time domain components and the manifold Kalman filter with $l' = 250$ components have the same convergence behaviour if $\boldsymbol{P}(k)$ is initialized with the training data covariance. This is observed

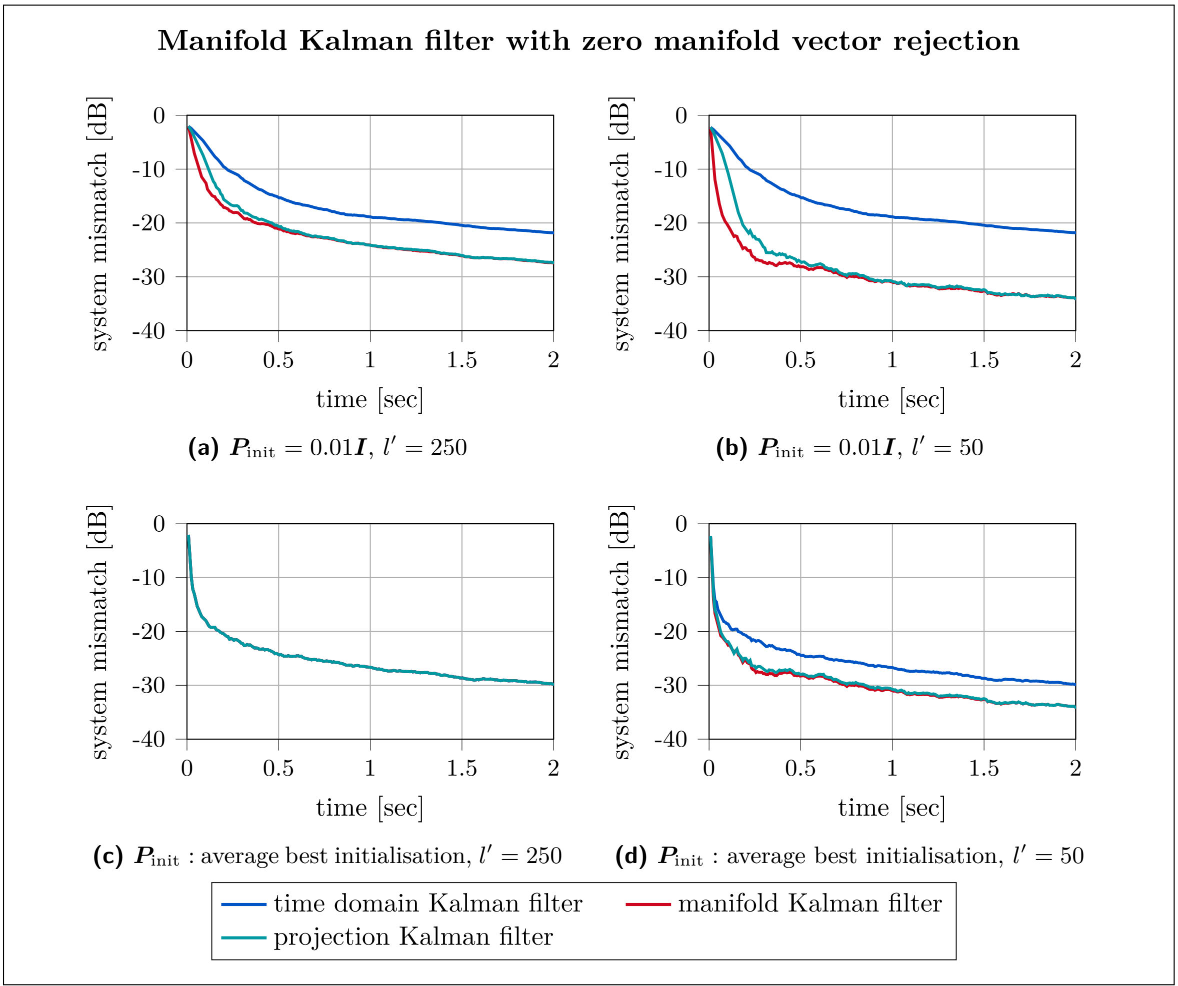


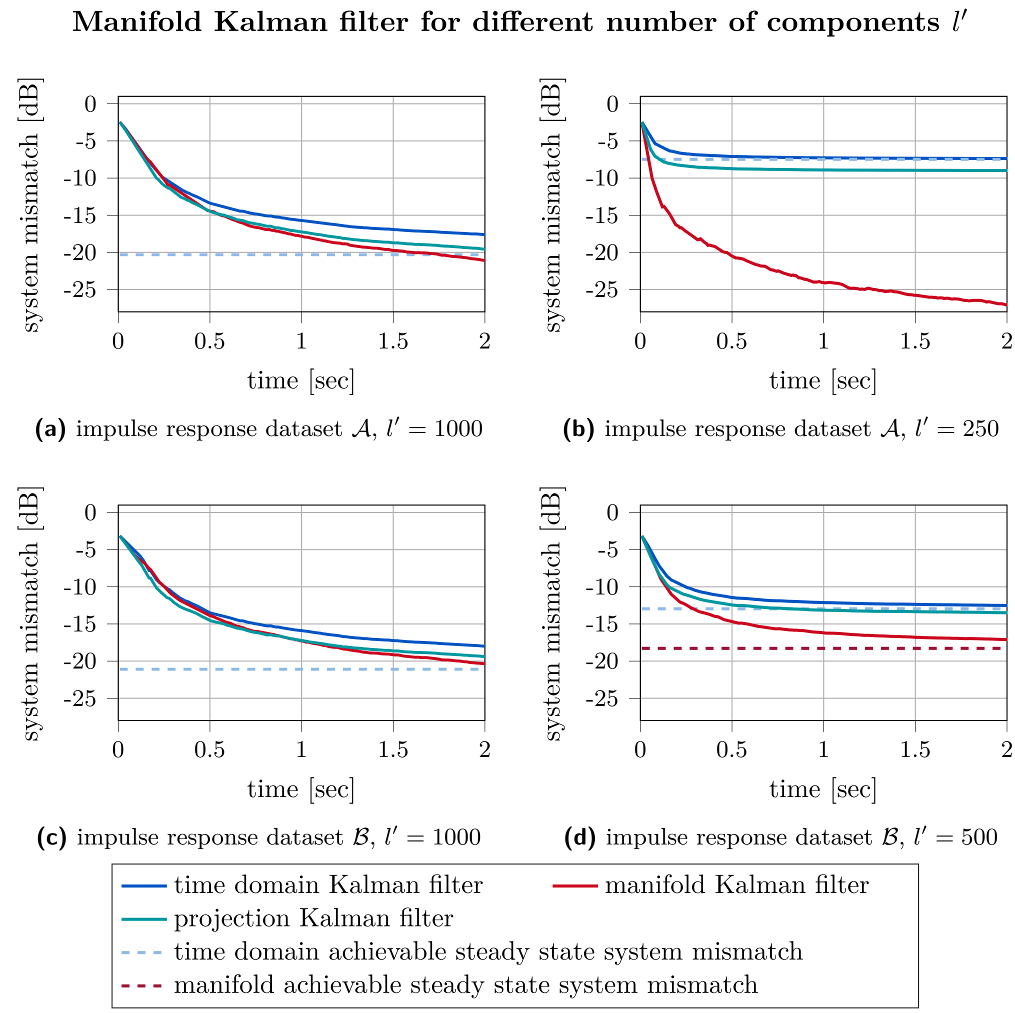
Figure 4.13: The system mismatch over the time for the time domain Kalman filter, the manifold Kalman filter, and the projection Kalman filter is shown for different initialisation of the error covariance estimate $\boldsymbol{P}_{\text{init}}$. The input signal $\boldsymbol{x}(k)$ is a white noise signal, and the manifold vector rejection is zero. The number of components of the true impulse response is $l = 1000$, the number of time domain components is $l' = 1000$, and the number of PCA manifold components is varied with $l' = 250$ and $l' = 50$.

in Figure 4.13c where the curves are identical. This is a major difference between the NLMS algorithm and the Kalman filter, because for the NLMS algorithm the convergence speed is directly related to the number of components l'.

Since in practical applications the true impulse response often changes over time, the first initialisation of $\boldsymbol{P}(k)$ is not significant. Because the initialisation of $\boldsymbol{P}_{\text{init}} = 0.01\boldsymbol{I}$ corresponds better to the case of reconvergence after the true impulse response changes, compared to the average best initialisation, the former will be used in the following simulations.

4.4.2 White noise excitation in the general case

In this section, the time domain Kalman filter, the manifold Kalman filter, and the projection Kalman filter are compared for varying numbers of time domain and manifold components l'. The results are shown in Figure 4.14. In the simulations in this section the impulse response



(a) impulse response dataset $\mathcal{A}$, $l' = 1000$

(b) impulse response dataset $\mathcal{A}$, $l' = 250$

(c) impulse response dataset $\mathcal{B}$, $l' = 1000$

(d) impulse response dataset $\mathcal{B}$, $l' = 500$

Figure 4.14: The system mismatch over the time for the time domain Kalman filter, the manifold Kalman filter, and the projection Kalman filter is shown for the PCA manifold of dataset $\mathcal{A}$ (circle with no variation) and dataset $\mathcal{B}$ (circle with variation) with varying number of components l'. The input signal $\boldsymbol{x}(k)$ is a white noise signal, and the number of components of the true impulse response is $l = 4000$. The number of time domain components and PCA manifold components is $l' = 1000$ and $l' = 250$ for dataset $\mathcal{A}$, and $l' = 1000$ and $l' = 500$ for dataset $\mathcal{B}$.

dataset $\mathcal{A}$ and dataset $\mathcal{B}$, and an impulse response of length $l = 4000$ are used. The number of time domain components and PCA manifold components is varied from $l' = 1000$ to $l' = 250$ for dataset $\mathcal{A}$, and varied from $l' = 1000$ to $l' = 500$ for dataset $\mathcal{B}$, which are the same parameter settings as in the previous section dealing with the manifold NLMS algorithm. The ENR is set to 10 dB. The achievable system mismatch is indicated with dashed lines.

When both the number of time domain components and the number of PCA manifold components is $l' = 1000$ (Figures 4.14a and 4.14c), the convergence speed is rather slow, and the algorithms show a general similarity in the curves of the system mismatch. It can be seen that the convergence speed of the projection Kalman filter is the fastest for $l' = 1000$ time domain components and PCA manifold components, but after 2 seconds, the system mismatch of the manifold Kalman filter is the lowest of the three algorithms. This observations hold for both dataset $\mathcal{A}$ and dataset $\mathcal{B}$.

When the number of components is small with $l' = 250$ (Figure 4.14b), the achievable system mismatch with $l' = 250$ time domain components is only $-7.3\,$dB, whereas the achievable system mismatch for $l' = 250$ PCA manifold components is well below $-30\,$dB. Furthermore, the convergence speed is much faster compared to the case of $l' = 1000$ components. Projecting onto the PCA manifold when using $l' = 250$ time domain components for the projection Kalman filter leads to an achievable system mismatch of about $-9\,$dB, which is only slightly smaller than the achievable system mismatch of the time domain. The projection Kalman filter uses here the same number of time domain components as the time domain Kalman filter. This means that the projection onto the manifold cannot achieve a significantly lower system mismatch than the number of time domain components in itself. Therefore, a comparison of the projection Kalman filter and the manifold Kalman filter is done in the next section for the case when both algorithms have the same achievable system mismatch.

4.4.3 Speech excitation

In this section, the performance of the algorithm is investigated when confronted with speech input signals. Only the PCA manifold learned from impulse response dataset dataset $\mathcal{A}$ (circle with no variation) will be used, since in this case the number of PCA manifold components can be smaller to have the same achievable system mismatch compared to dataset $\mathcal{B}$ (circle with variation). In the previous section it was observed that for a small number of components l' the steady state system mismatch of the time domain Kalman filter was way larger compared to the manifold Kalman filter. Therefore, a first simulation is conducted to compare the convergence behaviour of the algorithms with either a constant number of components l' for all algorithms or a constant achievable system mismatch for all algorithms. The second simulation in this section is conducted to compare the convergence speed of the Kalman filter for two different ENR. For the third experiment, the position of the microphone is changed abruptly at 10 seconds to analyse the algorithms behaviour during reconvergence. The initialisation of the error covariance matrix is set to $\boldsymbol{P}_{\text{init}} = 0.01\boldsymbol{I}$ in all simulations in this section, and the number of components of the true impulse response is $l = 4000$.

In Figure 4.15 the results of the simulation are shown for the two cases when the number of time domain components and PCA manifold components is $l' = 97$ (Figure 4.15a), and when the achievable system mismatch for all algorithms is $-20\,$dB (Figure 4.15b). For the simulation of Figure 4.15a the projection Kalman filter uses $l' = 97$ time domain components to run the time domain Kalman filter, and the impulse response is projected on the PCA manifold with $l' = 97$ components. For the simulation of Figure 4.15b the number of time domain and PCA manifold components is selected according to Figure 4.1.4 to ensure an achievable system mismatch of $-20\,$dB. For this, $l' = 97$ PCA manifold components are needed, and $l' = 1000$ time domain components. The projection Kalman filter uses 1000 time domain components and $l' = 97$ PCA manifold components to also have an achievable system mismatch of about $-20\,$dB. For the simulation an ENR of $10\,$dB is used. In Figure 4.15a it is observed that the achievable system mismatch is severely limited for the time domain Kalman filter and the projection Kalman filter. When the number of time domain components is $l' = 97$, the system mismatch of the time domain Kalman filter does not go below $-4\,$dB, and the system mismatch of the projection Kalman filter does not go below $-6\,$dB, whereas the system mismatch of the manifold Kalman filter goes below $-15\,$dB after

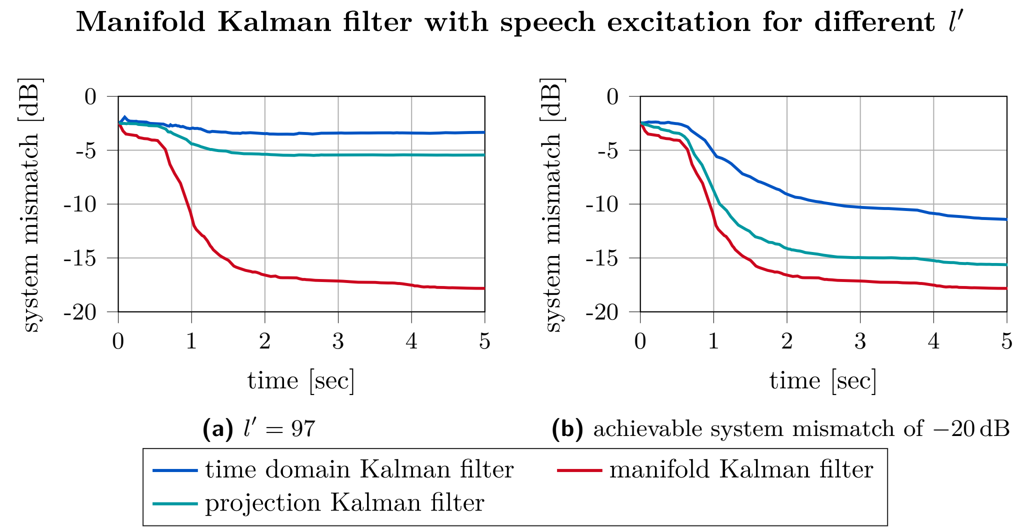


Figure 4.15: The system mismatch over the time for the time domain Kalman filter, the manifold Kalman filter, and the projection Kalman filter is shown for the case of $l' = 97$ time domain and PCA manifold components (a), or for the case when the achievable system mismatch of $-20\,$dB for all algorithms (b). The input signal $\boldsymbol{x}(k)$ is a speech signal, the number of components of the true impulse response is $l = 4000$.

about 1.5 seconds. In Figure 4.15b it can be seen that the convergence speed for the manifold Kalman filter is the fastest compared to the other algorithms. Because the number of time domain components is increased to $l' = 1000$ components for the projection Kalman filter to ensure an achievable system mismatch of $-20\,$dB, the step size (the denominator of the Kalman gain) of the projection Kalman filter is smaller compared to the manifold Kalman filter. This explains the faster convergence speed of the manifold Kalman filter compared to the projection Kalman filter.

Figure 4.16 shows the comparison of the convergence behaviour of the Kalman filter algorithms for different ENR values. To have a comparison of the algorithms with the same achievable system mismatch, the number of PCA manifold components and time domain components l' is selected from Figure 4.1.4 in to have an achievable system mismatch of $-20\,$dB, as described above. Comparing Figures 4.16a and 4.16b, the convergence speed is faster when the ENR is larger with 10 dB in Figure 4.16a compared to 0 dB in Figure 4.16b. This is true for the time domain, the manifold, and the projection Kalman filter. This is a different behaviour compared to the NLMS and LMS algorithm, where the convergence speed is independent of the ENR, as discussed in the previous sections.

In practical applications, it is not uncommon that the position of the loudspeaker or the microphone changes over time. To analyse a simplified case of a changing impulse response over time, a simulation is conducted where the true impulse response changes instantaneously after 10 seconds, with its results shown in Figure 4.17. To have a comparison of the algorithms with the same achievable system mismatch, the number of PCA manifold components and time domain components l' is selected from Figure 4.1.4 in to have an achievable system mismatch of $-20\,$dB, as described above, and an ENR of 10 dB is used in the simulation. To allow for reconvergence, the system update equations of the Kalman filter must allow the estimation of the error covariance to become larger over time. This can be achieved by setting

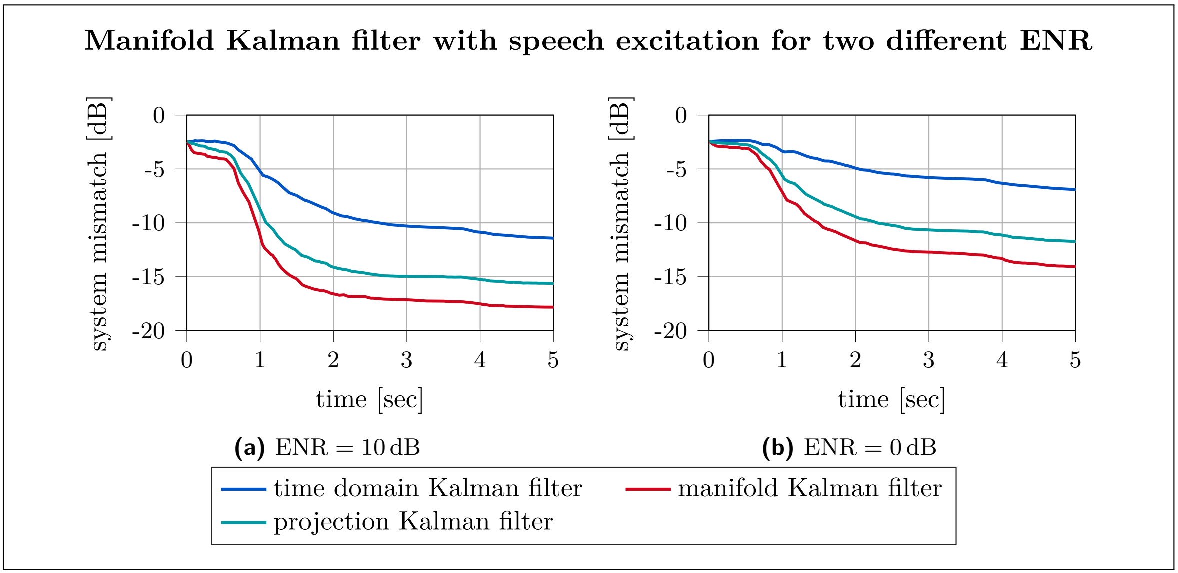


Figure 4.16: The system mismatch over the time for the time domain Kalman filter, the manifold Kalman filter, and the projection Kalman filter is shown for varying ENR. The number of time domain components and PCA manifold components are adjusted to have an achievable system mismatch of $-20\,\text{dB}$. The input signal $\boldsymbol{x}(k)$ is a speech signal, and the number of components of the true impulse response is $l = 4000$.

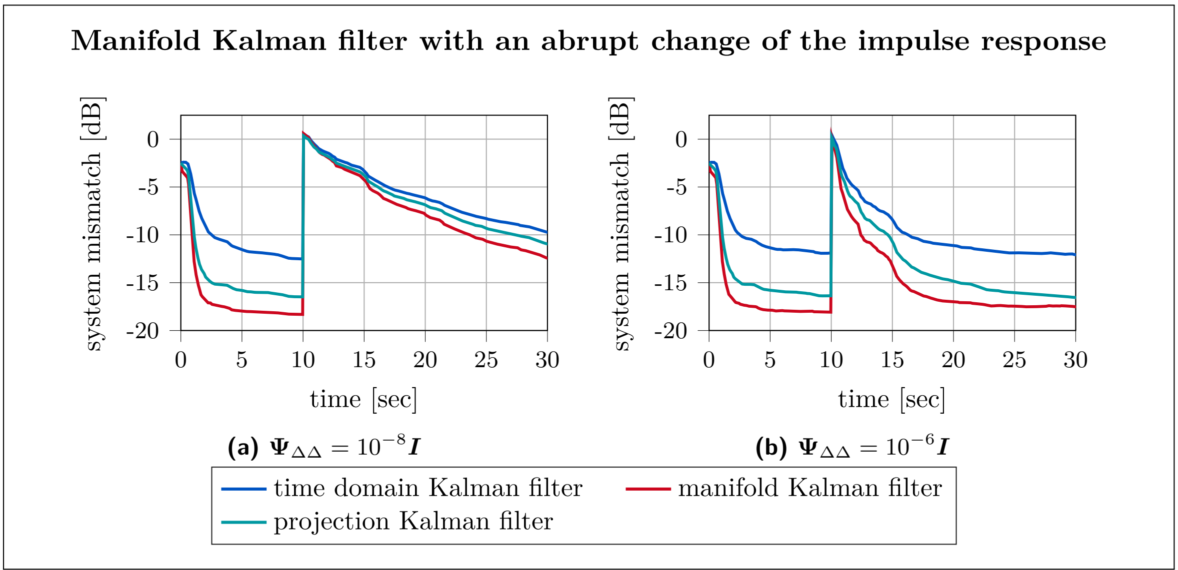
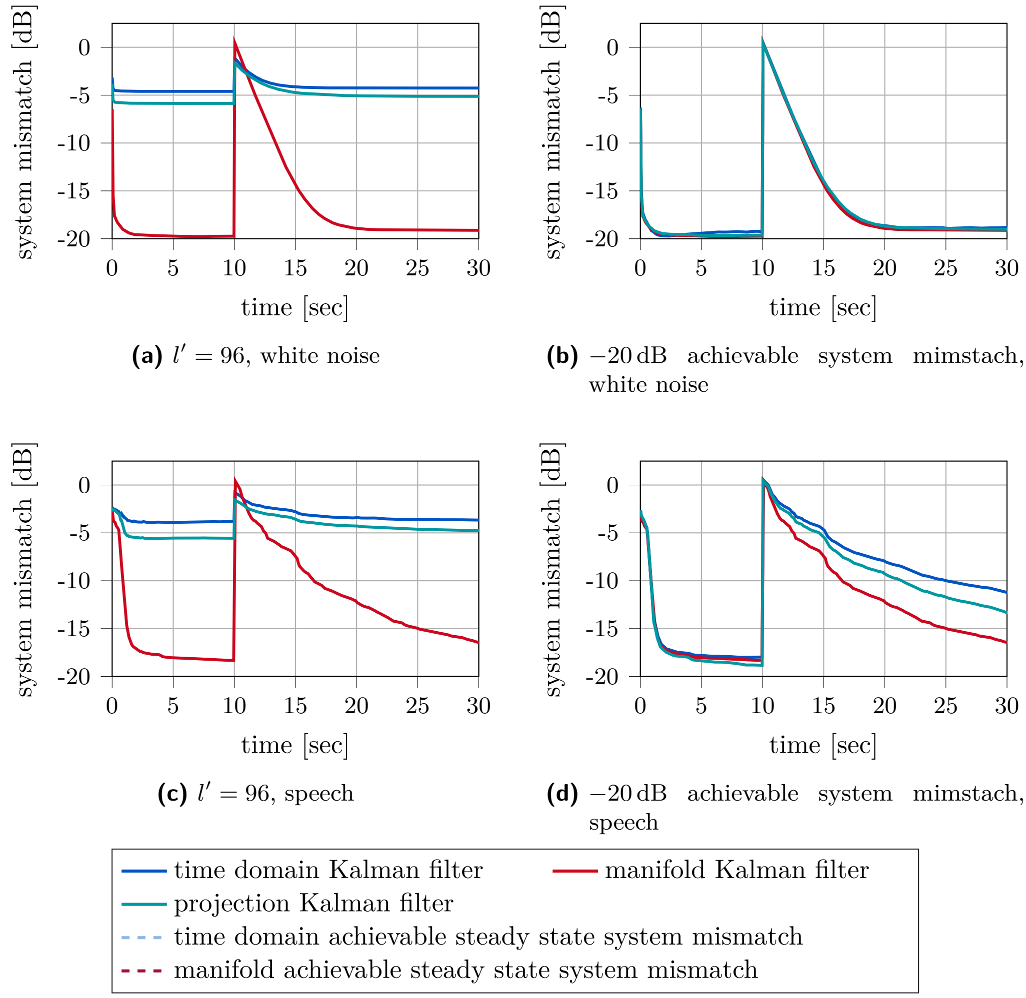


Figure 4.17: The system mismatch over the time for the time domain Kalman filter, the manifold Kalman filter, and the projection Kalman filter is shown for varying $\boldsymbol{\Psi}_{\Delta\Delta}$. The number of time domain components and PCA manifold components are adjusted to have an achievable system mismatch of $-20\,\text{dB}$. The input signal $\boldsymbol{x}(k)$ is a speech signal, the number of components of the true impulse response is $l = 4000$.

σ_Δ^2 to a value that is larger than zero. In the simulations $\sigma_\Delta^2 = 10^{-8}$ (Figure 4.17a), and $\sigma_\Delta^2 = 10^{-6}$ (Figure 4.17b) is used for $\boldsymbol{\Psi}_{\Delta\Delta} = \sigma_\Delta^2 \boldsymbol{I}$. A larger σ_Δ^2 allows for faster reconvergence after system changes, at the expense of a higher steady state system mismatch [7]. Comparing Figures 4.17a and 4.17b, it can be seen that the manifold Kalman filter achieves the fastest convergence speed for a larger σ_Δ^2 (Figure 4.17b), and the reconvergence is faster compared to the time domain Kalman filter and the projection Kalman filter after the true impulse response changes at 10 seconds.

NEW plot



(a) $l' = 96$, white noise

(b) $-20\,\mathrm{dB}$ achievable system mimstach, white noise

(c) $l' = 96$, speech

(d) $-20\,\mathrm{dB}$ achievable system mimstach, speech

Figure 4.18: The system mismatch over the time is shown.

Chapter 5

Conclusion and Outlook

5.1 Summary

This thesis investigated the ability of affine subspace models to map impulse responses in the context of acoustic echo compensation and system identification. It was found that system identification algorithms operating on carefully chosen subspaces require less coefficients than common approaches without degrading the performance of the algorithms.

Furthermore, three novel algorithms, namely the manifold LMS algorithm, the manifold NLMS algorithm and the manifold Kalman filter were proposed and tested under various parameter settings. They provide a new approach by estimating the latent variables of an acoustic manifold to solve the system identification problem instead of estimating the impulse response directly. The algorithms were derived for the general case of a non linear manifold for two versions each, where the update of the estimate of the room impulse response can be done either on the latent variables or in the time domain. For affine manifolds, these two versions were shown to be equivalent. For the manifold LMS algorithm the expected convergence speed and steady state system mismatch for white noise excitation was theoretically derived. With this it was shown that the theoretic values of the convergence speed and the steady state system mismatch were in good agreement with simulations for the manifold LMS algorithm, and also for the manifold NLMS algorithm, which had a very similar behaviour for white noise excitation.

For PCA manifolds learned from impulse response training data, the manifold LMS and the manifold NLMS algorithm had a convergence speed that only depended on the step size and the number of PCA components. Furthermore, when comparing the manifold NLMS algorithm and the time domain NLMS algorithm, a significantly faster convergence speed to a similar or better steady state system mismatch was achieved for scenarios, where the loudspeaker and microphone positions inside a room are restricted to a smaller space.

The manifold Kalman filter was compared to the time domain Kalman filter and to a reference model from literature that projects onto the manifold after each update step, which was recently proposed as an effective way to increase the performance of the online system identification task. It was shown that for a global PCA manifold, the manifold Kalman filter had a faster initial convergence speed compared to the time domain Kalman filter and the projection Kalman filter, when the estimate of the error covariance is not perfect. This is the case in practical applications, where the impulse response can change over time. Compared to the projection Kalman filter, the manifold Kalman filter enabled the use of a larger step size for each update step in cases where the number of PCA components is smaller than the number of time domain components. A further advantage of the manifold Kalman filter was the reduction of computational complexity when the number of PCA components was smaller compared to the number of time domain components for the same achievable system mismatch.

5.2 Discussion and future work

As summarised in the previous section, this thesis examined the manifold NLMS algorithm and the manifold Kalman filter, which showed promising results in the simulated scenarios. In future works it should be investigated if the results can be replicated for more general scenarios, most importantly for a local PCA manifold. It was shown that a global PCA manifold works well when the training data originates from spatially restricted loudspeaker and microphone positions. If the training data comes from more general settings, e.g. where the loudspeaker and microphone are positioned randomly anywhere in the room, it should be investigated if a local PCA model achieve the same performance than the global PCA for a restricted training data set. There are in particular three points that should be investigated with more detail for a local PCA model.

First, the distance measure to select the nearest neighbours of the current estimate to select the impulse responses for the local PCA is an important design criterion. In previous works mainly the Euclidean distance or the Mahalanobis distance (which takes the covariance of the estimated impulse response into account) of impulse response in the time domain are used. It may be worth investigating if an improved distance measure could be defined. In this work, it was found that impulse response training data sets, where the loudspeaker and microphone positions were fairly restricted in their positions, e.g. the microphone being only positioned on a line, lead to affine subspace models that could achieve a small steady state system mismatch with a small number of PCA components. For the scenario when the loudspeaker has a fixed position and the microphone is moved inside a room, a promising approach could be to use a distance measure that takes the spatial position of the microphone into account. A reason why such a distance measure might be beneficial compared to the Euclidean distance is given by the fact that a small shift in the position of a microphone can cause a small shift in the time domain of a signal, which can lead to a large Euclidean distance of the impulse response. Approaches to estimating the spatial position of the microphone by only having an estimate of the impulse response are given by [26, 27]. Another approach to get a distance measure to determine the local neighbours to calculate a local PCA might be found using existing machine learning methods to train a neural network to learn a distance measure by minimizing the achievable system mismatch of a local PCA for some number of local PCA components.

The second main point is the task of determining the optimal number of local training data points for the local PCA. For this question not only the lowest system mismatch, but also the computational complexity should be taken into account, as more local training data samples increase the computational complexity significantly, because the local PCA is often computed during the runtime of the algorithm. Regarding this issue, it could also be investigated if the local PCA manifolds can be computed in advance without a loss in system identification performance.

Third, for the change of one local subspace to the next local subspace, it needs to be investigated what the best way of transitioning between two local subspaces is, especially for the estimate of the error covariance matrix, which is needed for the Kalman filter. A simple approach could be to keep track of the error covariance in the time domain and convert it to the subspace.

While this thesis investigated affine manifolds in the simulations, it could be analysed how the use of general manifolds learned by neural autoencoders compares to affine manifolds. In particular, it needs to be investigated in which scenarios non linear manifolds are able to have lower achievable system mismatches. Furthermore, since the convergence towards the global minimum for the manifold NLMS algorithm and the manifold Kalman filter can not be guaranteed for non linear manifolds, it needs to be tested if this is practically relevant for the convergence behaviours of the algorithms on average.

A related idea, which is not restricted to the manifold LMS algorithm, the manifold NLMS algorithm, or the manifold Kalman filter, is the adaptive change of the number of PCA components (or time domain components) during runtime to achieve faster initial convergence speed with a low steady state system mismatch at the same time. This can be achieved by starting with a small number of PCA components when the system mismatch is still large, and increasing the number of PCA components when the system mismatch gets smaller over time. However, there are several challenges, including that an estimate of the current system mismatch is needed to select the number of PCA components, and an initialisation for each newly added PCA component is required. For the NLMS algorithm this is not a problem, since setting the new latent variables to zero proved to be effective. However, for the Kalman filter the error covariance also needs to be initialised. This is a problem, since for the estimate of the error covariance to be near the true error covariance, the algorithm needs to run for a few iterations. When a latent variable is not updated from the start, but is added after some time, the error covariance estimate is less accurate, which slows down the convergence speed significantly.

In this thesis the case was simulated where the loudspeaker or the microphone changes its position instantaneously after some time, corresponding to an instantaneous change in the impulse response. The scenario of a changing impulse response over time could be extended to the more complex and more realistic case of the microphone and the loudspeaker moving along a continuous path over time. It could be expected that the proposed manifold Kalman filter can outperform other methods in the scenario of continuously moving sources, as it was here shown that especially the initial convergence speed of the manifold Kalman filter is improved compared to the projection Kalman filter and the time domain Kalman filter.

Appendix A

Derivation of the manifold NLMS algorithm

The following derivation uses the same objective function to formulate a minimization problem as the NLMS algorithm from [36], with the difference that the latent variable $\boldsymbol{z}$ is estimated instead of the variable $\boldsymbol{w}$ in the full space.

$$\min_{\boldsymbol{z}(k+1)\in\mathbb{R}^{l'}} \|\boldsymbol{z}(k+1)-\boldsymbol{z}(k)\|_2^2 \tag{A.1}$$

subject to

$$\boldsymbol{w}_\star^T\boldsymbol{x}(k) = \boldsymbol{f}_{\text{dec}}(\boldsymbol{z}(k+1))^T\boldsymbol{x}(k). \tag{A.2}$$

This minimization problem can be solved using the Lagrange multipliers method. The Lagrangian function $\mathcal{L}$ is defined as

$$\mathcal{L}(\boldsymbol{z}(k+1),\lambda) = \|\boldsymbol{z}(k+1)-\boldsymbol{z}(k)\|_2^2 + \lambda\left(\boldsymbol{w}_\star^T\boldsymbol{x}(k) - \boldsymbol{f}_{\text{dec}}(\boldsymbol{z}(k+1))^T\boldsymbol{x}(k)\right). \tag{A.3}$$

To obtain the solution of (A.1), the derivatives of the Lagrangian function with respect to $\boldsymbol{z}(k+1)$ and λ are set to zero with

$$\frac{\partial\mathcal{L}(\boldsymbol{z}(k+1),\lambda)}{\partial\boldsymbol{z}(k+1)} = 2\left(\boldsymbol{z}(k+1)-\boldsymbol{z}(k)\right)^T - \lambda\boldsymbol{x}(k)^T\frac{\partial\boldsymbol{f}_{\text{dec}}(\boldsymbol{\zeta})}{\partial\boldsymbol{\zeta}}\Big|_{\boldsymbol{\zeta}=\boldsymbol{z}(k+1)} \stackrel{!}{=} 0, \tag{A.4}$$

and

$$\frac{\partial\mathcal{L}(\boldsymbol{z}(k+1),\lambda)}{\partial\lambda}\boldsymbol{f}_{\text{dec}}(\boldsymbol{z}(k+1))^T\boldsymbol{x}(k) - \boldsymbol{w}_\star^T\boldsymbol{x}(k) \stackrel{!}{=} 0. \tag{A.5}$$

Equation (A.4) with the notation

$$\frac{\partial\boldsymbol{f}_{\text{dec}}(\boldsymbol{\zeta})}{\partial\boldsymbol{\zeta}}\Big|_{\boldsymbol{\zeta}=\boldsymbol{z}(k+1)} = \boldsymbol{J}(k+1) \tag{A.6}$$

and

$$\frac{\partial\boldsymbol{f}_{\text{dec}}(\boldsymbol{\zeta})}{\partial\boldsymbol{\zeta}}\Big|_{\boldsymbol{\zeta}=\boldsymbol{z}(k)} = \boldsymbol{J}(k) \tag{A.7}$$

gives us

$$\boldsymbol{z}(k+1)^T - \boldsymbol{z}(k)^T = \frac{\lambda}{2}\boldsymbol{x}(k)^T \boldsymbol{J}(k+1). \tag{A.8}$$

Multiplying both sides of (A.8) by $\boldsymbol{J}(k)^T\boldsymbol{x}(k)$ from the right side gives us

$$\left(\boldsymbol{z}(k+1) - \boldsymbol{z}(k)\right)^T \boldsymbol{J}(k)^T\boldsymbol{x}(k) = \frac{\lambda}{2}\left(\boldsymbol{J}(k+1)^T\boldsymbol{x}(k)\right)^T \boldsymbol{J}(k)^T\boldsymbol{x}(k). \tag{A.9}$$

The decoder $\boldsymbol{f}_{\text{dec}}(\boldsymbol{z}(k+1))$ is linearised at $\boldsymbol{z}(k)$:

$$\boldsymbol{f}_{\text{dec}}(\boldsymbol{z}(k+1)) \approx \boldsymbol{f}_{\text{dec}}(\boldsymbol{z}(k)) + \boldsymbol{J}(k)\left(\boldsymbol{z}(k+1) - \boldsymbol{z}(k)\right), \tag{A.10}$$

which can be rearranged to:

$$\left(\boldsymbol{z}(k+1) - \boldsymbol{z}(k)\right)^T \boldsymbol{J}(k)^T = \boldsymbol{f}_{\text{dec}}(\boldsymbol{z}(k+1))^T - \boldsymbol{f}_{\text{dec}}(\boldsymbol{z}(k))^T. \tag{A.11}$$

Substituting (A.11) into (A.9) gives us:

$$\boldsymbol{f}_{\text{dec}}(\boldsymbol{z}(k+1))^T\boldsymbol{x}(k) = \boldsymbol{f}_{\text{dec}}(\boldsymbol{z}(k))^T\boldsymbol{x}(k) + \frac{\lambda}{2}\left(\boldsymbol{J}(k+1)^T\boldsymbol{x}(k)\right)^T \boldsymbol{J}(k)^T\boldsymbol{x}(k), \tag{A.12}$$

and using (A.5), we have

$$\boldsymbol{w}_\star^T\boldsymbol{x}(k) = \boldsymbol{f}_{\text{dec}}(\boldsymbol{z}(k))^T\boldsymbol{x}(k) + \frac{\lambda}{2}\left(\boldsymbol{J}(k+1)^T\boldsymbol{x}(k)\right)^T \boldsymbol{J}(k)^T\boldsymbol{x}(k), \tag{A.13}$$

which can be solved for lambda and further simplified by substituting in $e(k) = \boldsymbol{w}_\star^T\boldsymbol{x}(k) - \boldsymbol{f}_{\text{dec}}(\boldsymbol{z}(k))^T\boldsymbol{x}(k)$. Then we have

$$\lambda = \frac{2e(k)}{\left(\boldsymbol{J}(k+1)^T\boldsymbol{x}(k)\right)^T \boldsymbol{J}(k)^T\boldsymbol{x}(k)}. \tag{A.14}$$

Transposing equation (A.9), and using the value for λ from equation (A.14), the NLMS update equation is given with

$$\boldsymbol{z}(k+1) = \boldsymbol{z}(k) + \frac{\boldsymbol{x}(k)e(k)}{\left(\boldsymbol{J}(k+1)^T\boldsymbol{x}(k)\right)^T \boldsymbol{J}(k)^T\boldsymbol{x}(k)}. \tag{A.15}$$

As $\boldsymbol{J}(k+1)$ is not available at the update step to calculate $\boldsymbol{w}(k+1)$, $\boldsymbol{J}(k+1)$ needs to be approximated by $\boldsymbol{J}(k)$, which gives the update equation:

$$\boldsymbol{z}(k+1) = \boldsymbol{z}(k) + \frac{\boldsymbol{x}(k)e(k)}{\|\boldsymbol{J}(k)^T\boldsymbol{x}(k)\|_2^2}. \tag{A.16}$$

To control the step size of the NLMS algorithm to change the convergence speed and the steady state value, a step size parameter α is added to give the update equation

$$\boldsymbol{z}(k+1) = \boldsymbol{z}(k) + \alpha\frac{\boldsymbol{J}(k)^T\boldsymbol{x}(k)e(k)}{\|\boldsymbol{J}(k)^T\boldsymbol{x}(k)\|_2^2}. \tag{A.17}$$

Appendix B

Derivation of the manifold Kalman filter

In the following, we derive the manifold Kalman filter in a manner that allows its prediction step to correspond directly with that of the time domain Kalman filter, although the prediction step of the manifold Kalman filter could be defined in multiple ways.

The prediction step of the time domain Kalman filter is given as

$$\boldsymbol{w}^{+}(k) = \gamma \boldsymbol{w}(k) + \Delta(k), \tag{B.1}$$

which can be mapped to the latent variable $\boldsymbol{z}^{+}(k)$ of the manifold with

$$\begin{aligned}
\boldsymbol{z}^{+}(k) &= \boldsymbol{f}_{\mathrm{enc}}(\boldsymbol{w}^{+}(k)) && \text{(B.2)}\\
&= \boldsymbol{f}_{\mathrm{enc}}(\gamma \boldsymbol{w}(k) + \Delta(k)) && \text{(B.3)}\\
&\approx \boldsymbol{f}_{\mathrm{enc}}(\boldsymbol{w}(k)) + \boldsymbol{J}(k)^{T}(\gamma \boldsymbol{w}(k) + \Delta(k) - \boldsymbol{w}(k)) && \text{(B.4)}\\
&= \boldsymbol{z}(k) + \boldsymbol{J}(k)^{T}\Delta(k) + (\gamma - 1)\boldsymbol{J}(k)^{T}\boldsymbol{f}_{\mathrm{dec}}(\boldsymbol{z}(k)), && \text{(B.5)}
\end{aligned}$$

where the encoder is linearised at $\boldsymbol{w}(k)$ to give the update rule of the prediction step of the manifold Kalman filter. The microphone measurement $y(k)$ is given as

$$y(k) = \boldsymbol{f}_{\mathrm{dec}}\left(\boldsymbol{z}_{\star}(k)\right)^{T}\boldsymbol{x}(k) + n(k), \tag{B.6}$$

with the far end signal $\boldsymbol{x}(k)$ and the near end noise $n(k)$.

The covariance matrix in the full space is denoted here by

$$\boldsymbol{P}_{w}^{+}(k) = \mathrm{E}\left\{(\boldsymbol{w}_{\star}^{\mathcal{M}} - \boldsymbol{w}^{+}(k))(\boldsymbol{w}_{\star}^{\mathcal{M}} - \boldsymbol{w}^{+}(k))^{T}\right\}, \tag{B.7}$$

where $\boldsymbol{w}_{\star}^{\mathcal{M}}$ is the true impulse response mapped onto the manifold, and the prediction step equation in the time domain was given with

$$\boldsymbol{P}_{w}^{+}(k) = \gamma^{2}\boldsymbol{P}_{w}(k) + \boldsymbol{\Psi}_{\Delta\Delta}. \tag{B.8}$$

The covariance matrix of the latent space is given as

$$\boldsymbol{P}^{+}(k) = \mathrm{E}\left\{(\boldsymbol{z}_{\star} - \boldsymbol{z}^{+}(k))(\boldsymbol{z}_{\star} - \boldsymbol{z}^{+}(k))^{T}\right\}, \tag{B.9}$$

where $\boldsymbol{z}_\star = \boldsymbol{f}_{\text{enc}}(\boldsymbol{w}_\star)$. The encoder is linearised at $\boldsymbol{w}^+(k)$ with

$$\boldsymbol{f}_{\text{enc}}(\boldsymbol{w}_\star^{\mathcal{M}}) \approx \boldsymbol{f}_{\text{enc}}(\boldsymbol{w}^+(k)) + \boldsymbol{J}(k)^T(\boldsymbol{w}_\star^{\mathcal{M}} - \boldsymbol{w}^+(k)), \tag{B.10}$$

which computes $\boldsymbol{P}^+(k)$ to

$$\begin{aligned}
\boldsymbol{P}^+(k) &= \mathrm{E}\left\{\left(\boldsymbol{f}_{\text{enc}}(\boldsymbol{w}_\star^{\mathcal{M}}) - \boldsymbol{f}_{\text{enc}}(\boldsymbol{w}^+(k))\right)\left(\boldsymbol{f}_{\text{enc}}(\boldsymbol{w}_\star^{\mathcal{M}}) - \boldsymbol{f}_{\text{enc}}(\boldsymbol{w}^+(k))\right)^T\right\} && \text{(B.11)}\\
&\approx \mathrm{E}\left\{\left(\boldsymbol{J}(k)^T(\boldsymbol{w}_\star^{\mathcal{M}} - \boldsymbol{w}^+(k))\right)\left(\boldsymbol{J}(k)^T(\boldsymbol{w}_\star^{\mathcal{M}} - \boldsymbol{w}^+(k))\right)^T\right\} && \text{(B.12)}\\
&= \boldsymbol{J}(k)^T\mathrm{E}\left\{\left(\boldsymbol{w}_\star^{\mathcal{M}} - \boldsymbol{w}^+(k)\right)\left(\boldsymbol{w}_\star^{\mathcal{M}} - \boldsymbol{w}^+(k)\right)^T\right\}\boldsymbol{J}(k) && \text{(B.13)}\\
&= \boldsymbol{J}(k)^T\boldsymbol{P}_w^+(k)\boldsymbol{J}(k) && \text{(B.14)}\\
&= \boldsymbol{J}(k)^T\gamma^2\boldsymbol{P}_w(k) + \boldsymbol{\Psi}_{\Delta\Delta}\boldsymbol{J}(k) && \text{(B.15)}\\
&= \gamma^2\boldsymbol{J}(k)^T\boldsymbol{P}_w(k)\boldsymbol{J}(k) + \boldsymbol{J}(k)^T\boldsymbol{\Psi}_{\Delta\Delta}\boldsymbol{J}(k). && \text{(B.16)}
\end{aligned}$$

Furthermore, the decoder can be linearised at $\boldsymbol{z}^+(k)$ with $\boldsymbol{f}_{\text{dec}}(\boldsymbol{z}_\star) \approx \boldsymbol{f}_{\text{dec}}(\boldsymbol{z}^+(k)) + \boldsymbol{J}(k)^T(\boldsymbol{z}_\star - \boldsymbol{z}^+(k))$ to show that

$$\begin{aligned}
\boldsymbol{P}_w(k) =& \mathrm{E}\left\{\left(\boldsymbol{w}_\star^{\mathcal{M}} - \boldsymbol{w}(k)\right)\left(\boldsymbol{w}_\star^{\mathcal{M}} - \boldsymbol{w}(k)\right)^T\right\} && \text{(B.17)}\\
=& \mathrm{E}\left\{\left(\boldsymbol{f}_{\text{dec}}(\boldsymbol{z}_\star) - \boldsymbol{f}_{\text{dec}}(\boldsymbol{z}(k))\right)\left(\boldsymbol{f}_{\text{dec}}(\boldsymbol{z}_\star) - \boldsymbol{f}_{\text{dec}}(\boldsymbol{z}(k))\right)^T\right\} && \text{(B.18)}\\
\approx& \mathrm{E}\left\{\left(\boldsymbol{J}(k)(\boldsymbol{z}_\star - \boldsymbol{z}(k))\right)\left(\boldsymbol{J}(k)(\boldsymbol{z}_\star - \boldsymbol{z}(k))\right)^T\right\} && \text{(B.19)}\\
=& \boldsymbol{J}(k)\mathrm{E}\left\{\left(\boldsymbol{z}_\star - \boldsymbol{z}(k)\right)\left(\boldsymbol{z}_\star - \boldsymbol{z}(k)\right)^T\right\}\boldsymbol{J}(k)^T && \text{(B.20)}\\
=& \boldsymbol{J}(k)\boldsymbol{P}(k)\boldsymbol{J}(k)^T. && \text{(B.21)}
\end{aligned}$$

Using equations (B.16) and (B.21), the prediction step equation for $\boldsymbol{P}^+(k)$ can be formulated as

$$\begin{aligned}
\boldsymbol{P}^+(k) &= \gamma^2\boldsymbol{J}(k)^T\boldsymbol{J}(k)\boldsymbol{P}(k)\boldsymbol{J}(k)^T\boldsymbol{J}(k) + \boldsymbol{J}(k)^T\boldsymbol{\Psi}_{\Delta\Delta}\boldsymbol{J}(k) && \text{(B.22)}\\
&= \gamma^2\boldsymbol{P}(k) + \boldsymbol{J}(k)^T\boldsymbol{\Psi}_{\Delta\Delta}\boldsymbol{J}(k). && \text{(B.23)}
\end{aligned}$$

Finally, the manifold Kalman filter prediction and update equations can be formulated. The prediction step of the estimate and the error covariance are given with

$$\boldsymbol{z}^+(k) = \boldsymbol{z}(k) + (\gamma - 1)\boldsymbol{J}(k)^T\boldsymbol{f}_{\text{dec}}(\boldsymbol{z}(k)), \tag{B.24}$$

and

$$\boldsymbol{P}^+(k) = \gamma^2\boldsymbol{P}(k) + \boldsymbol{J}(k)^T\boldsymbol{\Psi}_{\Delta\Delta}\boldsymbol{J}(k), \tag{B.25}$$

The update equations using the new measurement are given with

$$\boldsymbol{K}(k) = \frac{\boldsymbol{P}^+(k)\boldsymbol{H}(k)^T}{\boldsymbol{H}(k)\boldsymbol{P}^+(k)\boldsymbol{H}(k)^T + \sigma_n^2}, \tag{B.26}$$

$$\boldsymbol{z}(k+1) = \boldsymbol{z}^+(k) + e(k)\boldsymbol{K}(k), \tag{B.27}$$

and

$$\boldsymbol{P}(k) = \boldsymbol{P}^{+}(k) - \boldsymbol{K}(k)\boldsymbol{H}(k)\boldsymbol{P}^{+}(k), \tag{B.28}$$

with

$$\boldsymbol{H}(k) = \frac{\partial \boldsymbol{f}_{\mathrm{dec}}(\boldsymbol{\zeta})^T \boldsymbol{x}(k)}{\partial \boldsymbol{\zeta}}\bigg|_{\boldsymbol{\zeta}=\boldsymbol{z}(k)} = \left(\boldsymbol{J}(k)^T \boldsymbol{x}(k)\right)^T = \boldsymbol{x}(k)^T \boldsymbol{J}(k), \tag{B.29}$$

where $\boldsymbol{K}(k)$ is the Kalman gain, and σ_n^2 is the noise power, and the residual echo signal $e(k)$ is given as

$$e(k) = y(k) - \boldsymbol{f}_{\mathrm{dec}}\left(\boldsymbol{z}^{+}(k)\right)^T \boldsymbol{x}(k). \tag{B.30}$$

To get the estimate of the impulse response in the time domain, its representation in latent space needs to be decoded with

$$\boldsymbol{w}(k+1) = \boldsymbol{f}_{\mathrm{dec}}(\boldsymbol{z}(k+1)). \tag{B.31}$$

Appendix C

Convergence speed of the manifold LMS algorithm

In the following analysis, we examine the convergence speed of the manifold LMS algorithm under several assumptions. Specifically, we assume that the near end noise $n(k)$ and manifold vector rejection $\boldsymbol{\Delta}_{\mathcal{M}}$ are zero, the input signal $\boldsymbol{x}(k)$ is a white noise signal, and that the signals $\boldsymbol{d}(k)$ and $\boldsymbol{x}(k)$ are statistically independent.

For simpler notation, the distance $\boldsymbol{d}(k)$ between the true impulse response $\boldsymbol{w}_\star$ to the current estimate $\boldsymbol{w}(k)$ is defined with

$$\boldsymbol{d}(k) = \boldsymbol{w}_\star - \boldsymbol{w}(k), \tag{C.1}$$

which is analysed further with

$$\begin{aligned}
\mathrm{E}\left\{\|\boldsymbol{d}(k+1)\|_2^2\right\} =& \mathrm{E}\left\{\|\boldsymbol{w}_\star - \boldsymbol{w}(k+1)\|_2^2\right\} && \text{(C.2)}\\
=& \mathrm{E}\left\{\left\|\boldsymbol{w}_\star - \boldsymbol{w}(k) - \beta \boldsymbol{J}\boldsymbol{J}^T \boldsymbol{x}(k) e(k)\right\|_2^2\right\} && \text{(C.3)}\\
=& \mathrm{E}\left\{\left\|\boldsymbol{d}(k) - \beta \boldsymbol{J}\boldsymbol{J}^T \boldsymbol{x}(k) e(k)\right\|_2^2\right\} && \text{(C.4)}\\
=& \mathrm{E}\left\{\|\boldsymbol{d}(k)\|_2^2\right\} - 2\mathrm{E}\left\{\boldsymbol{d}(k)^T \beta \boldsymbol{J}\boldsymbol{J}^T \boldsymbol{x}(k) e(k)\right\} \\
&+ \beta^2 \mathrm{E}\left\{\left\|\boldsymbol{J}\boldsymbol{J}^T \boldsymbol{x}(k)\right\|_2^2 e(k)^2\right\} && \text{(C.5)}\\
=& \mathrm{E}\left\{\|\boldsymbol{d}(k)\|_2^2\right\} - 2\beta \mathrm{E}\left\{\boldsymbol{d}(k)^T \boldsymbol{J}\boldsymbol{J}^T \boldsymbol{x}(k) \boldsymbol{d}(k)^T \boldsymbol{x}(k)\right\} \\
&+ \beta^2 \mathrm{E}\left\{\left\|\boldsymbol{J}\boldsymbol{J}^T \boldsymbol{x}(k)\right\|_2^2 (\boldsymbol{d}(k)^T \boldsymbol{x}(k))^2\right\}. && \text{(C.6)}
\end{aligned}$$

The following proposition gives a result to simplify equation (C.12), and allows to easily use the result of the proposition later again.

Proposition 3. *Assuming an orthogonal affine manifold for the manifold LMS algorithm, the following equations holds.*

$$\boldsymbol{f}_{dec}\left(\boldsymbol{f}_{enc}\left(\boldsymbol{w}_\star\right)\right) - \boldsymbol{w}(k) = \boldsymbol{J}\boldsymbol{J}^T\left(\boldsymbol{f}_{dec}\left(\boldsymbol{f}_{enc}\left(\boldsymbol{w}_\star\right)\right) - \boldsymbol{w}(k)\right) \tag{C.7}$$

Proof: The encoder and decoder functions of an affine manifold are given as $\boldsymbol{f}_{\text{enc}}(\boldsymbol{w}(k)) = \boldsymbol{J}^T (\boldsymbol{w}(k) - \bar{\boldsymbol{w}})$, and $\boldsymbol{f}_{\text{dec}}(\boldsymbol{z}(k)) = \boldsymbol{J}\boldsymbol{z}(k) + \bar{\boldsymbol{w}}$. Since $\boldsymbol{w}(k) = \boldsymbol{f}_{\text{dec}}(\boldsymbol{z}(k))$ for the manifold LMS algorithm, $\boldsymbol{w}(k)$ lies always on the manifold with $\boldsymbol{w}(k) = \boldsymbol{f}_{\text{dec}}(\boldsymbol{f}_{\text{enc}}(\boldsymbol{w}(k)))$. This way it can be followed that

$$\begin{aligned}\boldsymbol{f}_{\text{dec}}(\boldsymbol{f}_{\text{enc}}(\boldsymbol{w}_\star)) - \boldsymbol{w}(k) &= \boldsymbol{f}_{\text{dec}}(\boldsymbol{f}_{\text{enc}}(\boldsymbol{w}_\star)) - \boldsymbol{f}_{\text{dec}}(\boldsymbol{f}_{\text{enc}}(\boldsymbol{w}(k))) && \text{(C.8)}\\ &= \boldsymbol{J}\boldsymbol{J}^T(\boldsymbol{w}_\star - \bar{\boldsymbol{w}}) + \bar{\boldsymbol{w}} - \left(\boldsymbol{J}\boldsymbol{J}^T(\boldsymbol{w}(k) - \bar{\boldsymbol{w}}) + \bar{\boldsymbol{w}}\right) && \text{(C.9)}\\ &= \boldsymbol{J}\boldsymbol{J}^T(\boldsymbol{f}_{\text{dec}}(\boldsymbol{f}_{\text{enc}}(\boldsymbol{w}_\star)) - \boldsymbol{w}(k)). && \text{(C.10)}\end{aligned}$$

■

Using Proposition 3, together with the assumptions that $\boldsymbol{\Delta}_{\mathcal{M}} = \boldsymbol{0}$, it can be followed that

$$\boldsymbol{d}(k)^T\boldsymbol{J}\boldsymbol{J}^T = \left(\boldsymbol{J}\boldsymbol{J}^T\boldsymbol{w}_\star - \boldsymbol{J}\boldsymbol{J}^T\boldsymbol{w}(k)\right)^T = (\boldsymbol{w}_\star - \boldsymbol{w}(k))^T = \boldsymbol{d}(k)^T. \tag{C.11}$$

Using this result leads to

$$\mathrm{E}\left\{\boldsymbol{d}(k)^T\boldsymbol{J}\boldsymbol{J}^T\boldsymbol{x}(k)\boldsymbol{d}(k)^T\boldsymbol{x}(k)\right\} = \mathrm{E}\left\{\left(\boldsymbol{d}(k)^T\boldsymbol{x}(k)\right)^2\right\} = \mathrm{E}\left\{\|\boldsymbol{d}(k)\|_2^2\right\}\sigma_x^2, \tag{C.12}$$

where the assumption that $\boldsymbol{d}(k)$ and $\boldsymbol{x}(k)$ are statistically independent is used.

Since $\mathrm{E}\{x_i\} = 0$ and $\mathrm{E}\{x_i^3\} = 0$ for white noise and with the assumption that $\boldsymbol{d}(k)$ and $\boldsymbol{x}(k)$ are statistically independent, we can write

$$\begin{aligned}&\mathrm{E}\left\{\left\|\boldsymbol{J}\boldsymbol{J}^T\boldsymbol{x}(k)\right\|_2^2(\boldsymbol{d}(k)^T\boldsymbol{x}(k))^2\right\}\\ =&\mathrm{E}\left\{\left\|\boldsymbol{J}^T\boldsymbol{x}(k)\right\|_2^2(\boldsymbol{d}(k)^T\boldsymbol{x}(k))^2\right\} && \text{(C.13)}\\ =&\mathrm{E}\left\{\left(x_k^2j_{11}^2 + \dots + x_{k+l-1}^2j_{l1}^2 + \dots + x_k^2j_{1l'}^2 + \dots + x_{k+l-1}^2j_{ll'}^2\right)\left(d_1^2x_k^2 + \dots d_l^2x_{k+l-1}^2\right)\right\}\\ & && \text{(C.14)}\\ =&\mathrm{E}\left\{d_1^2\left(x_k^4C_1 + x_k^2x_2^2C_2 + \dots + x_k^2x_{k+l-1}^2C_l\right) + \dots\right. && \text{(C.15)}\\ &\left.+d_l^2\left(x_{k+l-1}^2x_1^2C_1 + \dots + x_{k+l-1}^2x_{k+l-2}C_{l-1} + x_{k+l-1}^4C_l\right)\right\}\\ =&\mathrm{E}\left\{d_1^2\left(x_1^4C_1 + \sigma_x^4(l' - C_1)\right) + \dots + d_l^2\left(x_{k+l-1}^4C_l + \sigma_x^4(l' - C_l)\right)\right\}, && \text{(C.16)}\end{aligned}$$

where

$$C_n = \sum_{i=1}^{l'} j_{ni}^2, \tag{C.17}$$

and

$$\sum_{i=1}^{l} C_i = l'. \tag{C.18}$$

When the kurtosis of $\boldsymbol{x}(k)$ ($\mathrm{E}\{x_i^4\}$) is not too large, which is for example the case for a Gaussian distribution or a uniform distribution, a reasonable approximation of (C.16) is given as

$$\mathrm{E}\left\{d_1^2\sigma_x^4l' + \dots + d_l^2\sigma_x^4l'\right\} = \mathrm{E}\left\{\|\boldsymbol{d}(k)\|_2^2\right\}\sigma_x^4l' \tag{C.19}$$

Now, equation (C.6) combined with (C.12), (C.19) and with $\beta = \frac{\alpha}{l'\sigma_x^2}$ from (3.79), we have

$$\mathrm{E}\left\{\|\boldsymbol{d}(k+1)\|_2^2\right\} \approx \mathrm{E}\left\{\|\boldsymbol{d}(k)\|_2^2\right\} - \frac{2\alpha}{l'\sigma_x^2}\mathrm{E}\left\{\|\boldsymbol{d}(k)\|_2^2\right\}\sigma_x^2 + \left(\frac{\alpha}{l'\sigma_x^2}\right)^2 \mathrm{E}\left\{\|\boldsymbol{d}(k)\|_2^2\right\}\sigma_x^4 l' \tag{C.20}$$

$$= \mathrm{E}\left\{\|\boldsymbol{d}(k)\|_2^2\right\}\left(1 - \frac{2\alpha}{l'} + \frac{\alpha^2}{l'}\right) \tag{C.21}$$

$$= \mathrm{E}\left\{\|\boldsymbol{d}(k)\|_2^2\right\}\left(1 - \frac{\alpha}{l'}(2-\alpha)\right). \tag{C.22}$$

Dividing by $\mathrm{E}\left\{\|\boldsymbol{d}(k)\|_2^2\right\}$, the convergence speed s is concluded to be

$$s = 10\log_{10}\left(1 - \frac{\alpha}{l'}(2-\alpha)\right) f_s \left[\frac{\mathrm{dB}}{\mathrm{sec}}\right]. \tag{C.23}$$

Appendix D

Steady state system mismatch of the manifold LMS algorithm

In the following, we explore the steady state system mismatch of the manifold LMS algorithm, which assumes that the near end noise $n(k)$ is a stationary white noise process that is statistically independent of $\boldsymbol{x}(k)$. The analysis considers two cases: first, when the true impulse response perfectly aligns with the manifold ($\boldsymbol{\Delta}_{\mathcal{M}} = \boldsymbol{0}$), and second, when the true impulse response deviates from the manifold. The input signal $\boldsymbol{x}(k)$ is a stationary white noise signal, and we assume that the signals $\boldsymbol{d}(k)$ and $\boldsymbol{x}(k)$ are uncorrelated for large k.

Steady state value with zero manifold vector rejection

The true impulse response $\boldsymbol{w}_\star$ is assumed to lie perfectly on the manifold, meaning that

$$\boldsymbol{w}_\star = \boldsymbol{f}_{\mathrm{dec}}(\boldsymbol{f}_{\mathrm{enc}}(\boldsymbol{w}_\star)). \tag{D.1}$$

Similar to the derivation of the convergence speed in Section 3.5.4, we write

$$\begin{aligned}
\mathrm{E}\left\{\|\boldsymbol{d}(k+1)\|_2^2\right\} =&\mathrm{E}\left\{\|\boldsymbol{d}(k)\|_2^2\right\} - 2\mathrm{E}\left\{\boldsymbol{d}(k)^T\beta\boldsymbol{J}\boldsymbol{J}^T\boldsymbol{x}(k)e(k)\right\} \\
&+ \beta^2\mathrm{E}\left\{\left\|\boldsymbol{J}\boldsymbol{J}^T\boldsymbol{x}(k)\right\|_2^2 e(k)^2\right\} \qquad \text{(D.2)}\\
=&\mathrm{E}\left\{\|\boldsymbol{d}(k)\|_2^2\right\} - 2\beta\mathrm{E}\left\{\boldsymbol{d}(k)^T\boldsymbol{J}\boldsymbol{J}^T\boldsymbol{x}(k)\left(\boldsymbol{d}(k)^T\boldsymbol{x}(k)+n(k)\right)\right\} \\
&+ \beta^2\mathrm{E}\left\{\left\|\boldsymbol{J}\boldsymbol{J}^T\boldsymbol{x}(k)\right\|_2^2 (\boldsymbol{d}(k)^T\boldsymbol{x}(k)+n(k))^2\right\}. \qquad \text{(D.3)}
\end{aligned}$$

With $\boldsymbol{\Delta}_{\mathcal{M}} = 0$, and $\boldsymbol{d}(k)^T\boldsymbol{J}\boldsymbol{J}^T = \boldsymbol{d}(k)^T$ (shown in equation (C.11) using Proposition 3), the second term is simplified to

$$\begin{aligned}
&\mathrm{E}\left\{\boldsymbol{d}(k)^T\boldsymbol{J}\boldsymbol{J}^T\boldsymbol{x}(k)\left(\boldsymbol{d}(k)^T\boldsymbol{x}(k)+n(k)\right)\right\}\\
=&\mathrm{E}\left\{\boldsymbol{d}(k)^T\boldsymbol{x}(k)\left(\boldsymbol{d}(k)^T\boldsymbol{x}(k)+n(k)\right)\right\} \qquad \text{(D.4)}\\
=&\mathrm{E}\left\{\boldsymbol{d}(k)^T\boldsymbol{x}(k)\boldsymbol{d}(k)^T\boldsymbol{x}(k)\right\} + \mathrm{E}\left\{\boldsymbol{d}(k)^T\boldsymbol{x}(k)n(k)\right\} \qquad \text{(D.5)}\\
=&\mathrm{E}\left\{\boldsymbol{d}(k)^T\boldsymbol{x}(k)\boldsymbol{d}(k)^T\boldsymbol{x}(k)\right\}, \qquad \text{(D.6)}
\end{aligned}$$

since $n(k)$ is assumed to be independent from $\boldsymbol{x}(k)$ and $\boldsymbol{d}(k)$ and $\mathrm{E}\{n(k)\} = 0$. Because the steady state is being examined here, it only has to be assumed that $\boldsymbol{d}(k)$ is uncorrelated with $\boldsymbol{x}(k)$ for $k \to \infty$, which is a reasonable assumption regardless of $\boldsymbol{J}$. With this assumption, (D.6) is simplified to

$$\mathrm{E}\left\{\boldsymbol{d}(k)^T\boldsymbol{x}(k)\boldsymbol{d}(k)^T\boldsymbol{x}(k)\right\} = \mathrm{E}\left\{\|\boldsymbol{d}(k)\|_2^2\right\}\sigma_x^2 \tag{D.7}$$

The third term can be rewritten to

$$\begin{aligned}
&\mathrm{E}\left\{\left\|\boldsymbol{J}\boldsymbol{J}^T\boldsymbol{x}(k)\right\|_2^2\left(\boldsymbol{d}(k)^T\boldsymbol{x}(k)+n(k)\right)^2\right\}\\
=&\mathrm{E}\left\{\left\|\boldsymbol{J}\boldsymbol{J}^T\boldsymbol{x}(k)\right\|_2^2\left((\boldsymbol{d}(k)^T\boldsymbol{x}(k))^2+2\boldsymbol{d}(k)^T\boldsymbol{x}(k)n(k)+n(k)^2\right)\right\} &\text{(D.8)}\\
=&\mathrm{E}\left\{\left\|\boldsymbol{J}\boldsymbol{J}^T\boldsymbol{x}(k)\right\|_2^2(\boldsymbol{d}(k)^T\boldsymbol{x}(k))^2\right\}+\mathrm{E}\left\{\left\|\boldsymbol{J}\boldsymbol{J}^T\boldsymbol{x}(k)\right\|_2^2\right\}\mathrm{E}\left\{n(k)^2\right\} &\text{(D.9)}\\
=&\mathrm{E}\left\{\left\|\boldsymbol{J}\boldsymbol{J}^T\boldsymbol{x}(k)\right\|_2^2(\boldsymbol{d}(k)^T\boldsymbol{x}(k))^2\right\}+l'\sigma_x^2\mathrm{E}\left\{n(k)^2\right\} &\text{(D.10)}\\
\approx&\mathrm{E}\left\{\|\boldsymbol{d}(k)\|_2^2\right\}l'\sigma_x^4+l'\sigma_x^2\mathrm{E}\left\{n(k)^2\right\}, &\text{(D.11)}
\end{aligned}$$

where (D.9) uses that $n(k)$ is independent from $\boldsymbol{x}(k)$ and $\boldsymbol{d}(k)$, and that $\mathrm{E}\{n(k)\} = 0$, (D.10) uses Propositions 1 and 2 (from Section 3.5.3), and equation (D.11) uses the same approximation as (C.19) with the assumption that $\boldsymbol{d}(k)$ is uncorrelated from $\boldsymbol{x}(k)$ for $k \to \infty$.

The noise power is written as $\mathrm{E}\{n(k)^2\} = \sigma_n^2$. With (D.6) and (D.11) equation (D.3) can be simplified to

$$\mathrm{E}\left\{\|\boldsymbol{d}(k+1)\|_2^2\right\} = \mathrm{E}\left\{\|\boldsymbol{d}(k)\|_2^2\right\}\left(1-\frac{\alpha}{l'}(2-\alpha)\right)+\left(\frac{\alpha}{l'\sigma_x^2}\right)^2 l'\sigma_x^2\sigma_n^2. \tag{D.12}$$

For $k \to \infty$, $\mathrm{E}\left\{\|\boldsymbol{d}(k+1)\|_2^2\right\} = \mathrm{E}\left\{\|\boldsymbol{d}(k)\|_2^2\right\} = \mathrm{E}\left\{\|\boldsymbol{d}_\infty\|_2^2\right\}$, which leads to

$$\mathrm{E}\left\{\|\boldsymbol{d}_\infty\|_2^2\right\} = \mathrm{E}\left\{\|\boldsymbol{d}_\infty\|_2^2\right\}\left(1-\frac{\alpha}{l'}(2-\alpha)\right)+\frac{\alpha^2}{l'}\frac{\sigma_n^2}{\sigma_x^2}. \tag{D.13}$$

This can be further rearranged to

$$\mathrm{E}\left\{\|\boldsymbol{d}_\infty\|_2^2\right\}\left(\frac{\alpha}{l'}(2-\alpha)\right) = \frac{\alpha^2}{l'}\frac{\sigma_n^2}{\sigma_x^2}, \tag{D.14}$$

and finally to

$$\mathrm{E}\left\{\|\boldsymbol{d}_\infty\|_2^2\right\} = \frac{\alpha}{2-\alpha}\frac{\sigma_n^2}{\sigma_x^2}. \tag{D.15}$$

The steady state system mismatch $\boldsymbol{D}_\infty$ is then given as

$$\boldsymbol{D}_\infty = \frac{\alpha}{2-\alpha}\frac{\sigma_n^2}{\sigma_x^2\|\boldsymbol{w}_\star\|_2^2}. \tag{D.16}$$

Steady state system mismatch with manifold vector rejection

Now the case is considered where the true impulse response does not lie on the manifold with the manifold vector rejection

$$\boldsymbol{\Delta}_{\mathcal{M}} = \boldsymbol{w}_{\star} - \boldsymbol{f}_{\text{dec}}(\boldsymbol{f}_{\text{enc}}(\boldsymbol{w}_{\star})). \tag{D.17}$$

The true impulse response mapped onto the manifold $\boldsymbol{w}_{\star}^{\mathcal{M}}$ is defined here with

$$\boldsymbol{w}_{\star}^{\mathcal{M}} = \boldsymbol{f}_{\text{dec}}\left(\boldsymbol{f}_{\text{enc}}(\boldsymbol{w}_{\star})\right), \tag{D.18}$$

which lets us write (3.88) as

$$\begin{aligned}
\mathrm{E}\left\{\|\boldsymbol{d}_{\infty}\|_2^2\right\} =& \mathrm{E}\left\{\left\|\boldsymbol{w}_{\infty} - \boldsymbol{w}_{\star}^{\mathcal{M}} - \boldsymbol{\Delta}_{\mathcal{M}}\right\|_2^2\right\} && \text{(D.19)}\\
=& \mathrm{E}\left\{\left\|\boldsymbol{w}_{\infty} - \boldsymbol{w}_{\star}^{\mathcal{M}}\right\|_2^2\right\} - 2\mathrm{E}\left\{(\boldsymbol{w}_{\infty} - \boldsymbol{w}_{\star}^{\mathcal{M}})^T\boldsymbol{\Delta}_{\mathcal{M}}\right\} + \mathrm{E}\left\{\|\boldsymbol{\Delta}_{\mathcal{M}}\|_2^2\right\} && \text{(D.20)}\\
=& \mathrm{E}\left\{\left\|\boldsymbol{w}_{\infty} - \boldsymbol{w}_{\star}^{\mathcal{M}}\right\|_2^2\right\} + \|\boldsymbol{\Delta}_{\mathcal{M}}\|_2^2, && \text{(D.21)}
\end{aligned}$$

because $(\boldsymbol{w}_{\infty} - \boldsymbol{w}_{\star}^{\mathcal{M}})$ and $\boldsymbol{\Delta}_{\mathcal{M}}$ are uncorrelated and $(\boldsymbol{w}_{\infty} - \boldsymbol{w}_{\star}^{\mathcal{M}})$ has zero mean.

The term $\mathrm{E}\left\{\left\|\boldsymbol{w}_{\infty} - \boldsymbol{w}_{\star}^{\mathcal{M}}\right\|_2^2\right\}$ is analysed further and it is shown that it is equivalent to $\mathrm{E}\left\{\|\boldsymbol{z}_{\infty} - \boldsymbol{z}_{\star}\|_2^2\right\}$ for affine manifolds in the following. The term $\|\boldsymbol{z}_{\infty} - \boldsymbol{z}_{\star}\|_2^2$ describes the steady state distance on the manifold with $\boldsymbol{z}_{\star} = \boldsymbol{f}_{\text{enc}}(\boldsymbol{w}_{\star})$ and $\boldsymbol{z}_{\infty} = \boldsymbol{f}_{\text{enc}}(\boldsymbol{w}_{\infty})$.

Proposition 3 gives the result that

$$\boldsymbol{w}_{\star}^{\mathcal{M}} - \boldsymbol{w}(k) = \boldsymbol{J}\boldsymbol{J}^T\left(\boldsymbol{w}_{\star}^{\mathcal{M}} - \boldsymbol{w}(k)\right), \tag{D.22}$$

and by using the squared norm and the expected value on both sides we have

$$\mathrm{E}\left\{\left\|\boldsymbol{w}(k) - \boldsymbol{w}_{\star}^{\mathcal{M}}\right\|_2^2\right\} = \mathrm{E}\left\{\left\|\boldsymbol{J}\boldsymbol{J}^T\left(\boldsymbol{w}(k) - \boldsymbol{w}_{\star}^{\mathcal{M}}\right)\right\|_2^2\right\}. \tag{D.23}$$

Furthermore, using Proposition 1 gives the result that

$$\mathrm{E}\left\{\left\|\boldsymbol{J}\boldsymbol{J}^T\left(\boldsymbol{w}(k) - \boldsymbol{w}_{\star}^{\mathcal{M}}\right)\right\|_2^2\right\} = \mathrm{E}\left\{\left\|\boldsymbol{J}^T\left(\boldsymbol{w}(k) - \boldsymbol{w}_{\star}^{\mathcal{M}}\right)\right\|_2^2\right\}, \tag{D.24}$$

and since $\boldsymbol{f}_{\text{enc}}(\boldsymbol{w}(k)) = \boldsymbol{J}^T(\boldsymbol{w}(k) - \bar{\boldsymbol{w}})$, it follows that

$$\begin{aligned}
\boldsymbol{z}_{\star} - \boldsymbol{z}(k) =& \boldsymbol{f}_{\text{enc}}(\boldsymbol{w}_{\star}) - \boldsymbol{f}_{\text{enc}}(zk) && \text{(D.25)}\\
=& \boldsymbol{J}^T(\boldsymbol{w}_{\star} - \bar{\boldsymbol{w}}) - \boldsymbol{J}^T(\boldsymbol{w}(k) - \bar{\boldsymbol{w}}) && \text{(D.26)}\\
=& \boldsymbol{J}^T(\boldsymbol{w}_{\star} - \boldsymbol{w}(k)). && \text{(D.27)}
\end{aligned}$$

With equations (D.24) and (D.27) it can be concluded that

$$\mathrm{E}\left\{\left\|\boldsymbol{w}(k) - \boldsymbol{w}_{\star}^{\mathcal{M}}\right\|_2^2\right\} = \mathrm{E}\left\{\|\boldsymbol{z}(k) - \boldsymbol{z}_{\star}\|_2^2\right\}, \tag{D.28}$$

showing that the system mismatch of the true impulse response mapped to the manifold is equivalent to the system mismatch evaluated on the latent variables of the manifold.

The steady state distance of the true impulse response mapped onto the manifold is computed to

$$\mathrm{E}\left\{\left\|\boldsymbol{d}^{\mathcal{M}}(k+1)\right\|_2^2\right\} \tag{D.29}$$

$$=\mathrm{E}\left\{\left\|\boldsymbol{w}_\star^{\mathcal{M}}-\boldsymbol{w}(k+1)\right\|_2^2\right\} \tag{D.30}$$

$$=\mathrm{E}\left\{\left\|\boldsymbol{w}_\star^{\mathcal{M}}-\boldsymbol{w}(k)-\beta\boldsymbol{J}\boldsymbol{J}^T\boldsymbol{x}(k)e(k)\right\|_2^2\right\} \tag{D.31}$$

$$\begin{aligned}=&\mathrm{E}\left\{\left\|\boldsymbol{d}^{\mathcal{M}}(k)\right\|_2^2\right\}-2\mathrm{E}\left\{\boldsymbol{d}^{\mathcal{M}}(k)^T\beta\boldsymbol{J}\boldsymbol{J}^T\boldsymbol{x}(k)e(k)\right\}\\&+\beta^2\mathrm{E}\left\{\left\|\boldsymbol{J}\boldsymbol{J}^T\boldsymbol{x}(k)\right\|_2^2e(k)^2\right\}\end{aligned} \tag{D.32}$$

$$\begin{aligned}=&\mathrm{E}\left\{\left\|\boldsymbol{d}^{\mathcal{M}}(k)\right\|_2^2\right\}-2\beta\mathrm{E}\left\{\boldsymbol{d}^{\mathcal{M}}(k)^T\boldsymbol{J}\boldsymbol{J}^T\boldsymbol{x}(k)\left(\boldsymbol{d}^{\mathcal{M}}(k)^T\boldsymbol{x}(k)+\boldsymbol{\Delta}_{\mathcal{M}}^T\boldsymbol{x}(k)+n(k)\right)\right\}\\&+\beta^2\mathrm{E}\left\{\left\|\boldsymbol{J}(k)\boldsymbol{J}(k)^T\boldsymbol{x}(k)\right\|_2^2(\boldsymbol{d}^{\mathcal{M}}(k)^T\boldsymbol{x}(k)+\boldsymbol{\Delta}_{\mathcal{M}}^T\boldsymbol{x}(k)+n(k))^2\right\}.\end{aligned} \tag{D.33}$$

The second term can be simplified to

$$\mathrm{E}\left\{\boldsymbol{d}^{\mathcal{M}}(k)^T\boldsymbol{J}(k)\boldsymbol{J}(k)^T\boldsymbol{x}(k)\left(\boldsymbol{d}^{\mathcal{M}}(k)^T\boldsymbol{x}(k)+\boldsymbol{\Delta}_{\mathcal{M}}^T\boldsymbol{x}(k)+n(k)\right)\right\} \tag{D.34}$$

$$\begin{aligned}=&\mathrm{E}\left\{\boldsymbol{d}^{\mathcal{M}}(k)^T\boldsymbol{J}(k)\boldsymbol{J}(k)^T\boldsymbol{x}(k)\boldsymbol{d}^{\mathcal{M}}(k)^T\boldsymbol{x}(k)\right\}+\mathrm{E}\left\{\boldsymbol{d}^{\mathcal{M}}(k)^T\boldsymbol{J}(k)\boldsymbol{J}(k)^T\boldsymbol{x}(k)\boldsymbol{\Delta}_{\mathcal{M}}^T\boldsymbol{x}(k)\right\}\\&+\mathrm{E}\left\{\boldsymbol{d}^{\mathcal{M}}(k)^T\boldsymbol{J}(k)\boldsymbol{J}(k)^T\boldsymbol{x}(k)n(k)\right\}\end{aligned} \tag{D.35}$$

$$=\mathrm{E}\left\{\left(\boldsymbol{d}^{\mathcal{M}}(k)^T\boldsymbol{x}(k)\right)^2\right\}+\mathrm{E}\left\{\boldsymbol{d}^{\mathcal{M}}(k)^T\boldsymbol{x}(k)\boldsymbol{\Delta}_{\mathcal{M}}^T\boldsymbol{x}(k)\right\}+\mathrm{E}\left\{\boldsymbol{d}^{\mathcal{M}}(k)^T\boldsymbol{x}(k)\right\}\mathrm{E}\left\{n(k)\right\}, \tag{D.36}$$

because

$$\boldsymbol{J}(k)\boldsymbol{J}(k)^T\boldsymbol{d}^{\mathcal{M}}(k)=\boldsymbol{d}^{\mathcal{M}}(k), \tag{D.37}$$

and $n(k)$ is independent of $\boldsymbol{d}^{\mathcal{M}}(k)$ and $\boldsymbol{x}(k)$. The case of $k\to\infty$ is analysed for the steady state system mismatch, meaning that it can be safely assumed that $\boldsymbol{d}^{\mathcal{M}}(k)$ is uncorrelated from $\boldsymbol{\Delta}_{\mathcal{M}}$. Furthermore, $\mathrm{E}\left\{d_i^M(k)\right\}=0$, which leads to

$$\mathrm{E}\left\{\boldsymbol{d}^{\mathcal{M}}(k)^T\boldsymbol{x}(k)\boldsymbol{\Delta}_{\mathcal{M}}^T\boldsymbol{x}(k)\right\}=0. \tag{D.38}$$

This simplifies (D.36) to

$$\begin{aligned}&\mathrm{E}\left\{\left(\boldsymbol{d}^{\mathcal{M}}(k)^T\boldsymbol{x}(k)\right)^2\right\}+\mathrm{E}\left\{\boldsymbol{d}^{\mathcal{M}}(k)^T\boldsymbol{x}(k)\boldsymbol{\Delta}_{\mathcal{M}}^T\boldsymbol{x}(k)\right\}+\mathrm{E}\left\{\boldsymbol{d}^{\mathcal{M}}(k)^T\boldsymbol{x}(k)\right\}\mathrm{E}\left\{n(k)\right\}\\=&\mathrm{E}\left\{\left\|\boldsymbol{d}^{\mathcal{M}}(k)^T\right\|_2^2\right\}\sigma_x^2.\end{aligned} \tag{D.39}$$

The third term can be calculated to

$$\begin{aligned}&\mathrm{E}\left\{\left\|\boldsymbol{J}(k)\boldsymbol{J}(k)^T\boldsymbol{x}(k)\right\|_2^2(\boldsymbol{d}^{\mathcal{M}}(k)^T\boldsymbol{x}(k)+\boldsymbol{\Delta}_{\mathcal{M}}^T\boldsymbol{x}(k)+n(k))^2\right\}\\=&\mathrm{E}\left\{\left\|\boldsymbol{J}(k)\boldsymbol{J}(k)^T\boldsymbol{x}(k)\right\|_2^2\left(\left(\boldsymbol{d}^{\mathcal{M}}(k)^T\boldsymbol{x}(k)\right)^2+\left(\boldsymbol{\Delta}_{\mathcal{M}}^T\boldsymbol{x}(k)\right)^2+n(k)^2\right.\right.\\&\left.\left.+2\boldsymbol{d}^{\mathcal{M}}(k)^T\boldsymbol{x}(k)\boldsymbol{\Delta}_{\mathcal{M}}^T\boldsymbol{x}(k)+2\boldsymbol{d}^{\mathcal{M}}(k)^T\boldsymbol{x}(k)n(k)+2\boldsymbol{\Delta}_{\mathcal{M}}^T\boldsymbol{x}(k)n(k)\right)\right\},\end{aligned} \tag{D.40}$$

where the expectation of the mixed term containing simply $n(k)$ becomes zero. Following the same argumentation as above, the mixed therm $\mathrm{E}\left\{\left\|\boldsymbol{J}(k)\boldsymbol{J}(k)^T\boldsymbol{x}(k)\right\|_2^2 \boldsymbol{d}^{\mathcal{M}}(k)^T\boldsymbol{x}(k)\boldsymbol{\Delta}_{\mathcal{M}}^T\boldsymbol{x}(k)\right\}$ also becomes zero, simplifying (D.40) to

$$\mathrm{E}\left\{\left\|\boldsymbol{J}(k)\boldsymbol{J}(k)^T\boldsymbol{x}(k)\right\|_2^2\left(\left(\boldsymbol{d}^{\mathcal{M}}(k)^T\boldsymbol{x}(k)\right)^2+\left(\boldsymbol{\Delta}_{\mathcal{M}}^T\boldsymbol{x}(k)\right)^2+n(k)^2\right)\right\} \tag{D.41}$$

$$=\mathrm{E}\left\{\left\|\boldsymbol{J}(k)\boldsymbol{J}(k)^T\boldsymbol{x}(k)\right\|_2^2\left(\left(\boldsymbol{d}^{\mathcal{M}}(k)^T\boldsymbol{x}(k)\right)^2+\left(\boldsymbol{\Delta}_{\mathcal{M}}^T\boldsymbol{x}(k)\right)^2\right)\right\}+l'\sigma_x^2\sigma_n^2 \tag{D.42}$$

$$\approx l'\sigma_x^4\left\|\boldsymbol{d}^{\mathcal{M}}(k)\right\|_2^2+l'\sigma_x^4\left\|\boldsymbol{\Delta}_{\mathcal{M}}\right\|_2^2+l'\sigma_x^2\sigma_n^2, \tag{D.43}$$

where the same approximation steps are used as in (C.19).

Finally, using (D.39) and (D.43), equation (D.33) can be simplified to

$$\begin{aligned}\mathrm{E}\left\{\left\|\boldsymbol{d}^{\mathcal{M}}(k+1)\right\|_2^2\right\} =&\mathrm{E}\left\{\left\|\boldsymbol{d}^{\mathcal{M}}(k)\right\|_2^2-2\beta\mathrm{E}\left\{\left\|\boldsymbol{d}^{\mathcal{M}}(k)\right\|_2^2\right\}\sigma_x^2\right.\\ &\left.+\beta^2\left(l'\sigma_x^4\left\|\boldsymbol{d}^{\mathcal{M}}(k)\right\|_2^2+l'\sigma_x^4\left\|\boldsymbol{\Delta}_{\mathcal{M}}\right\|_2^2+l'\sigma_x^2\sigma_n^2\right)\right\}.\end{aligned} \tag{D.44}$$

The steady state solution $\mathrm{E}\left\{\left\|\boldsymbol{d}_\infty^{\mathcal{M}}\right\|_2^2\right\}$ is defined as

$$\mathrm{E}\left\{\left\|\boldsymbol{d}^{\mathcal{M}}(k+1)\right\|_2^2\right\}=\mathrm{E}\left\{\left\|\boldsymbol{d}^{\mathcal{M}}(k)\right\|_2^2\right\}=\mathrm{E}\left\{\left\|\boldsymbol{d}_\infty^{\mathcal{M}}\right\|_2^2\right\}. \tag{D.45}$$

To evaluate the steady state solution of (D.44), the equation changes to

$$\begin{aligned}\mathrm{E}\left\{\left\|\boldsymbol{d}_\infty^{\mathcal{M}}\right\|_2^2\right\} =&\mathrm{E}\left\{\left\|\boldsymbol{d}_\infty^{\mathcal{M}}\right\|_2^2-2\beta\mathrm{E}\left\{\left\|\boldsymbol{d}_\infty^{\mathcal{M}}\right\|_2^2\right\}\sigma_x^2\right.\\ &\left.+\beta^2\left(l'\sigma_x^4\left\|\boldsymbol{d}_\infty^{\mathcal{M}}\right\|_2^2+l'\sigma_x^4\left\|\boldsymbol{\Delta}_{\mathcal{M}}\right\|_2^2+l'\sigma_x^2\sigma_n^2\right)\right\},\end{aligned} \tag{D.46}$$

which is rearranged to be solved for $\mathrm{E}\left\{\left\|\boldsymbol{d}_\infty^{\mathcal{M}}\right\|_2^2\right\}$ with

$$\mathrm{E}\left\{\left\|\boldsymbol{d}_\infty^{\mathcal{M}}\right\|_2^2\right\}\left(2\beta\sigma_x^2-\beta^2l'\sigma_x^4\right)=\beta^2l'\sigma_x^4\left\|\boldsymbol{\Delta}_{\mathcal{M}}\right\|_2^2+\beta^2l'\sigma_x^2\sigma_n^2. \tag{D.47}$$

Putting in the step size $\beta=\alpha/(l'\sigma_x^2)$ simplifies the equation to

$$\mathrm{E}\left\{\left\|\boldsymbol{d}_\infty^{\mathcal{M}}\right\|_2^2\right\}\left(2\frac{\alpha}{l'}-\frac{\alpha^2}{l'}\right)=\frac{\alpha^2}{l'}\left\|\boldsymbol{\Delta}_{\mathcal{M}}\right\|_2^2+\frac{\alpha^2}{l'\sigma_x^2}\sigma_n^2, \tag{D.48}$$

dividing by l' and α, (D.48) is solved for $\mathrm{E}\left\{\left\|\boldsymbol{d}_\infty^{\mathcal{M}}\right\|_2^2\right\}$

$$\mathrm{E}\left\{\left\|\boldsymbol{d}_\infty^{\mathcal{M}}\right\|_2^2\right\}=\frac{\alpha}{2-\alpha}\left(\left\|\boldsymbol{\Delta}_{\mathcal{M}}\right\|_2^2+\frac{\sigma_n^2}{\sigma_x^2}\right). \tag{D.49}$$

To calculate the steady state distance of $\mathrm{E}\left\{\|\boldsymbol{d}_\infty\|_2^2\right\}$, equation (D.21) with (D.49) is used to conclude that

$$\mathrm{E}\left\{\|\boldsymbol{d}_\infty\|_2^2\right\}=\frac{\alpha}{2-\alpha}\left(\left\|\boldsymbol{\Delta}_{\mathcal{M}}\right\|_2^2+\frac{\sigma_n^2}{\sigma_x^2}\right)+\left\|\boldsymbol{\Delta}_{\mathcal{M}}\right\|_2^2. \tag{D.50}$$

The steady state system mismatch $\boldsymbol{D}_\infty$ is then given as

$$\boldsymbol{D}_\infty = \frac{\alpha}{2-\alpha}\left(\frac{\|\boldsymbol{\Delta}_{\mathcal{M}}\|_2^2}{\|\boldsymbol{w}_\star\|_2^2} + \frac{\sigma_n^2}{\sigma_x^2\,\|\boldsymbol{w}_\star\|_2^2}\right) + \frac{\|\boldsymbol{\Delta}_{\mathcal{M}}\|_2^2}{\|\boldsymbol{w}_\star\|_2^2}. \tag{D.51}$$

List of abbreviations

ENR Echo-to-Noise-Ratio

AEC Acoustic-Echo-Compensation

LMS Least-Mean-Square

NLMS Normalized-Least-Mean-Square

PCA Principal-Component-Analysis

List of Figures

Bibliography

[1] A. Ali, M. Moinuddin, and T. Y. Al-Naffouri. Nlms is more robust to input-correlation than lms: A proof. *IEEE Signal Processing Letters*, 29:279–283, 2021.

[2] M. Berger. Convexity. *The American Mathematical Monthly*, 97(8):650–678, 1990.

[3] S. Boyd and L. Vandenberghe. *Convex optimization.* Cambridge University Press, 2004.

[4] A. Carini. Efficient nlms and rls algorithms for perfect and imperfect periodic sequences. *IEEE transactions on signal processing*, 58(4):2048–2059, 2009.

[5] J. Eargle. *The Microphone Book: From mono to stereo to surround-a guide to microphone design and application.* Routledge, 2012.

[6] G. Enzner. Kalman filtering in acoustic echo control: A smooth ride on a rocky road. *Advances in Digital Speech Transmission*, pages 79–106, 2008.

[7] J. Fabry, S. Kühl, and P. Jax. On the steady state performance of the kalman filter applied to acoustical systems. *IEEE Signal Processing Letters*, 27:1854–1858, 2020.

[8] M. Fozunbal, T. Kalker, and R. W. Schafer. Multi-channel echo control by model learning. In *Proc. 11th IEEE International Workshop on Acoustic Echo and Noise Control*, 2008.

[9] I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning.* MIT Press, 2016. `http://www.deeplearningbook.org`.

[10] S. Grant and S. Gay. A multiple principal components based adaptive filter. In *Conference Record of the Thirty-Eighth Asilomar Conference on Signals, Systems and Computers*, volume 1, pages 945–949 Vol.1, 2004.

[11] Y. Haneda, S. Makino, and Y. Kaneda. Common acoustical pole and zero modeling of room transfer functions. *IEEE Transactions on Speech and Audio Processing*, 2(2):320–328, 1994.

[12] T. Haubner, A. Brendel, and W. Kellermann. Online supervised acoustic system identification exploiting prelearned local affine subspace models. In *2020 IEEE 30th International Workshop on Machine Learning for Signal Processing (MLSP)*, pages 1–6. IEEE, 2020.

[13] T. Haubner, A. Brendel, and W. Kellermann. Online acoustic system identification exploiting kalman filtering and an adaptive impulse response subspace model. *Journal of Signal Processing Systems*, 94(2):147–160, 2022.

[14] S. Haykin. *Adaptive Filter Theory, International Edition, 5th edition.* Pearson, 2013.

[15] T. Hsia. Convergence analysis of lms and nlms adaptive algorithms. In *ICASSP'83. IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 8, pages 667–670. IEEE, 1983.

[16] I. T. Jolliffe. *Principal component analysis for special types of data.* Springer, 2002.

[17] M. Jälmby, F. Elvander, and T. van Waterschoot. Low-rank tensor modeling of room impulse responses. In *2021 29th European Signal Processing Conference (EUSIPCO)*, pages 111–115, 2021.

[18] M. Jälmby, F. Elvander, and T. van Waterschoot. Low-rank room impulse response estimation. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, pages 1–13, 2023.

[19] T. Kabzinski and P. Jax. A unified perspective on time-domain and frequency-domain kalman filters for acoustic system identification. In *2022 30th European Signal Processing Conference (EUSIPCO)*, pages 90–94. IEEE, 2022.

[20] R. E. Kalman. A new approach to linear filtering and prediction problems. *Transactions of the ASME–Journal of Basic Engineering*, 82(Series D):35–45, 1960.

[21] R. E. Kalman and R. S. Bucy. New results in linear filtering and prediction theory. *ME Transaction, SeriesD*, 83(1):95–108, 1961.

[22] T. Koren, R. Talmon, and I. Cohen. Supervised system identification based on local pca models. In *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 541–544. IEEE, 2012.

[23] A. J. Krener. The convergence of the extended kalman filter. In *Directions in Mathematical Systems Theory and Optimization*, pages 173–182. Springer Berlin Heidelberg, 2003.

[24] H. Kuttruff. *Room acoustics.* Crc Press, 2016.

[25] T. Lacey. Tutorial: the kalman filter 11.1 introduction 11.2 mean squared error. 1998. `https://web.mit.edu/kirtley/kirtley/binlustuff/literature/control/Kalman%20filter.pdf`.

[26] B. Laufer-Goldshtein, R. Talmon, and S. Gannot. A study on manifolds of acoustic responses. In *Latent Variable Analysis and Signal Separation*, pages 203–210. Springer International Publishing, 2015.

[27] B. Laufer-Goldshtein, R. Talmon, and S. Gannot. Semi-supervised sound source localization based on manifold regularization. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 24(8):1393–1407, 2016.

[28] D. P. Mandic, S. Kanna, and A. G. Constantinides. On the intrinsic relationship between the least mean square and kalman filters [lecture notes]. *IEEE Signal Processing Magazine*, 32(6):117–122, 2015.

[29] M. I. Ribeiro. Kalman and extended kalman filters: Concept, derivation and properties. *Institute for Systems and Robotics*, 43:46, 2004.

[30] J. B. Rosinha, S. J. M. de Almeida, and J. C. M. Bermudez. A new kernel kalman filter algorithm for estimating time-varying nonlinear systems. In *2017 IEEE International Symposium on Circuits and Systems (ISCAS)*, pages 1–4, 2017.

[31] R. Scheibler, E. Bezzam, and I. Dokmanić. Pyroomacoustics: A python package for audio room simulation and array processing algorithms. In *2018 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pages 351–355. IEEE, 2018.

[32] K. Schmidt and G. Trenkler. *Masszahlen von Matrizen*, pages 59–83. Springer Berlin Heidelberg, 2015.

[33] D. T. Slock. On the convergence behavior of the lms and the normalized lms algorithms. *IEEE Transactions on Signal processing*, 41(9):2811–2825, 1993.

[34] Y. Song and J. W. Grizzle. The extended kalman filter as a local asymptotic observer for nonlinear discrete-time systems. In *1992 American control conference*, pages 3365–3369. IEEE, 1992.

[35] P. Vary and R. Martin. *Digital speech transmission: Enhancement, coding and error concealment*. John Wiley & Sons, 2006.

[36] S. V. Vaseghi. *Multimedia signal processing: theory and applications in speech, music and communications*. John Wiley & Sons, 2007.

[37] C. Veaux, J. Yamagishi, and S. King. The voice bank corpus: Design, collection and data analysis of a large regional accent speech database. In *2013 International Conference Oriental COCOSDA held jointly with 2013 Conference on Asian Spoken Language Research and Evaluation (O-COCOSDA/CASLRE)*, pages 1–4, 2013.

[38] S. Zhang, J. Zhang, and H. C. So. Mean square deviation analysis of lms and nlms algorithms with white reference inputs. *Signal Processing*, 131:20–26, 2017.