Simplified Wiener LMS Tracking With Automatic Tuning of the Step-Size

Jonas Rutström

March 2005

UPPSALA UNIVERSITET

Dissertation for the degree of Technical Licentiate
in Signal Processing at Uppsala University, 2005.

ABSTRACT

This work considers tracking of time-varying parameters and automatic tuning of the step-size for the Simplified Wiener LMS algorithm (SWLMS). When tracking time-varying parameters in applications where the rate of change of the time-varying parameters and the noise level may change frequently, it is of interest to adjust the adaptation gain, or the step-size, on-line. The reason for this is that proper manual tuning of the step-size in these cases often is very time consuming, or maybe even impossible. The purpose of this work has been to find a promising step-size updating algorithm to be used in combination with the SWLMS algorithm in order to create an almost self-tuning algorithm that can be used only with little help from the system designer. Various step-size candidates are evaluated and compared in different tracking scenarios.

In addition to the comparison of the different step-size algorithms, a small study concerning two other tracking issues is also performed. The first issue deals with the potential performance gain obtained by introducing individual step-size control of the different time-varying parameters. The second issue concerns the use of specific information, available to the designer, about the time-varying parameters and the characteristics of signals passed through the time-varying system, that maybe can be applied to improve the overall tracking performance.

In the end, a small case study is performed. Here the most promising algorithms are implemented in a realistic communication scenario. It is shown that the proposed methods are widely superior compared to the traditional constant gain algorithms.

Keywords: Adaptive filters, adaptive signal processing, channel estimation, time-varying environments, tracking, variable step-size algorithms, LMS, WLMS.

This thesis has been prepared using LaTeX.

Printed in Sweden by Eklundshof Grafiska AB, Uppsala, March 2005.
Distributed by Signals and Systems, Department of Engineering Sciences,
Uppsala University, Uppsala, Sweden.

To my Parents

Contents

Preface

During the last year of my undergraduate education I was totally confident not to end up at the university. I was going to move from Uppsala like most of my friends, and start a carrier at one of the hi-tech companies in Stockholm. That was the plan! However, since you are reading this, something must have happened somewhere along the track. No one really knows what made me change my mind, but it has something to do with the course *Adaptive Signal Processing*, given at the Signals and Systems Group at Uppsala University. For some reason, me and four other students, out of a class of sixteen, decided to participate in this course although we did not have the slightest clue what it was about, but it sounded interesting.

Signal processing... most people I talk to refer to trains and traffic lights when they here those words - it turned out to be something else! However, due to the low number of students on the course, i.e. three (one gave up!), we developed a nice relationship with the different teachers giving the course. (To be honest, there were actually more teachers than students!). Anyway, this, I guess, was a turning point where I realized that I had found a subject that I really liked. It was a combination of creative thinking, mathematics, programming, and maybe most important of all - it was useful and easy to apply in real applications.

Then one day about a year after finishing the course, when me and my friend Mathias Johansson were looking for a Master thesis, we returned to the Signals and Systems Group asking for suitable projects. We were then asked to investigate the possibility to enhance the quality of sound reproduction systems with the help of signal processing. This, finally, turned out so well that we were able to start a company, *Dirac Research AB*, partly based on the results from our thesis. The company was started together with some colleagues, and our supervisors, Anders Ahlén and Mikael Sternad. However, after finishing the Master thesis we were offered to continue our studies at the university as graduate students, enhancing our knowledge in the art of signal processing. This work is the result of my studies.

Finishing this work would not have been possible without the following persons: Prof. Anders Ahlén, Prof. Mikael Sternad and Dr. Lars Lindbom. These people have thoroughly supported me during the writing of this thesis in order to end up with a nice product.

I also acknowledge all the people working at the Signal and Systems Group.

Special thanks to: Dr. Mathias Johansson, Daniel Aronsson, Erik Björnemo, Dr. Nilo Casimiro Ericsson, Dr. Fredrik Lingvall, Dr. Sorour Falahati, Dr. Tomas Olofsson, Erik Öjefors, Dr. Mattias Wennström, Pär Dahlman, Lars-Johan Brännmark, Peter Lindberg, Marcus Jonsson and Erik Wennerström.

Jonas Rutström
Uppsala, March 2005.

Chapter 1

Introduction

To be able to *track* or *follow* different things has always been important for people . In fact, it may actually start already in the beginning of the day. In order to get ready for work we need to; wake up, hopefully get our breakfast, then, maybe listen to the news to follow what is happening in the world. After that we might take the car and drive to our work, which involves keeping track of the road when driving. Then, when we finally arrive at work we need to follow up upon what to do in order to make sure that our work is on the right track. We probably also follow the progress of our savings day by day by listening to the daily economical reports. These ”tracking problems” are very simple to understand and solve, in fact, we probably don’t even think about them as ”tracking problems” because the they are part of our daily routines.

In this work we will consider another type of tracking problems - *parameter tracking in time variable systems*. This means that our goal is to follow the variations of certain parameters in a system that changes its behavior over time. Examples of such systems might be

- Communication systems
- Process control systems
- Economical systems

We will in this work regard the first one, *Communication systems* as our main target for the results produced hereafter.

Over the years, many different methods of how to track or follow time-varying parameters have been developed see eg. [1],[2],[3]. These methods are called *adaptive algorithms* and are frequently used in many applications. One of the most famous adaptive algorithms is the so called *Least Mean Square* or *LMS* algorithm [3] introduced by Widrow and Hoff in the sixties. The fame of the LMS algorithm is explained by its simplicity. However, due to this simplicity the algorithm suffers from some drawbacks which might restrict its use in some applications. One of these drawbacks is the *slow convergence* of the algorithm. This problem has generated interest among researchers to modify the LMS algorithm such that the problem is alleviated. The slow convergence problem originates from the fact that the LMS algorithm, as well as many other algorithms of low complexity, uses a constant adaptation gain, or step-size[1]. Then in order to solve this problem it is common to introduce some sort of automatic step-size mechanism that controls the adaptation gain. These mechanisms have been shown to work well in situations where the *constant* gain LMS would do less well.

In [4], a framework of how to design tracking algorithms based on Wiener filtering theory was developed. One of the resulting algorithms in this work is called *The Simplified Wiener LMS* algorithm, or SWLMS. This is an adaptive algorithm characterized by its good tracking performance, low complexity and ease of use. It is related to the well known LMS algorithm. Unfortunately, like many other adaptive algorithms the SWLMS algorithm uses a constant adaptation gain which may lead to suboptimal tracking in some time-varying environments. This, together with some of the the different automatic step-size schemes developed for the LMS algorithm [5],[6],[7] has motivated us to investigate the possibility to equip the *Simplified Wiener LMS* algorithm with an automatic step-size adjustment in order to enhance the tracking performance and to make it less sensitive to changes in the tracking environment. Although the original motivation for the automatic step-size algorithms was to improve the convergence properties of the LMS algorithm, our motivation to use them together with the SWLMS is somewhat different. The driving force in our case has been to simplify the work of the system designer, since tuning of adaptive algorithm in real applications might be a time consuming process. Automatic tuning is therefore of great interest.

In this thesis we will investigate some of the step-size algorithms designed for the LMS algorithm, as well as develop a completely new algorithm based on the SWLMS algorithm in order to find the best candidate to be used in combination with the SWLMS algorithm.

[1]The adaptation gain or the step-size will be explained in more detail in Chapter 2.

In addition to the comparison of the different step-size algorithms, a small study concerning two other tracking issues is also performed. The first issue deals with the potential performance gain obtained by introducing individual step-size control of the different time-varying parameters. This means that, instead of using the same step-size parameter for all time-varying parameters an individual time-varying step-size parameter is assigned to each one of the parameters that are to be followed. The second issue concerns the use of specific information, available to the designer, about the time-varying parameters and the characteristics of signals passed through the time-varying system, that maybe can be applied to improve the overall tracking performance.

1.1 Outline

Chapter 2

In this chapter the tracking problem is introduced. Here suitable notation is introduced and the problem is approached mathematically. An introduction to adaptive parameter estimation is first given, then the focus is changed towards the tracking problem. Common adaptive tracking algorithms are introduced as well as the performance measures used in tracking. This chapter serves as the base in order to understand the rest of the work.

Chapter 3

This chapter starts with an overview describing why automatic tuning of the SWLMS algorithm is of interest. Then, common step-size methods used for automatic tuning of the LMS algorithm is introduced, these are then combined with the SWLMS algorithm in order to create a number of candidates used for evaluation in Chapter 4. Finally, a completely new step-size scheme based on the SWLMS algorithm is developed. The complexity for the different algorithms are finally presented in the end of the chapter.

Chapter 4

Here, the different algorithms from Chapter 3 are evaluated in different tracking scenarios. The aim is here to find the best candidate to be used in combination with the SWLMS algorithm. In the end, recommendations based on the different simulations are given in order to provide a potential user with information of how to use an automatic step-size algorithm together with the SWLMS algorithm. In this chapter the two other tracking issues described above are also investigated.

Chapter 5

In this chapter, the most promising algorithms from Chapter 4 are evaluated as parts of a real communication system (EDGE). The algorithms are implemented in an adaptive equalizer as a channel tracker in order to compensate for negative effects imposed by the wireless channel.

Chapter 6

Here we discuss further studies.

Chapter 2

The Tracking Problem

2.1 The system and the linear regression model

Consider a relation between a set of variables. Let us assume that a scalar output signal z_t is produced as

$$z_t = g(\varphi_t, h_t),\ t = 0, 1, 2 \ldots . \tag{2.1}$$

Here, $g(\cdot)$ is a function depending on the variables φ_t and h_t, where φ_t represents input signals and h_t the model parameters. The index t indicates that we are working with signals sampled in discrete time. This can be considered as a general discrete-time model with no restrictions on the system parameters or the input signals. Therefore, the model (2.1) covers both nonlinear and linear systems, dynamic systems and static relations, as well as real and complex signals. In this work the discussion is restricted to a system that is linear in the parameter h_t, described by the scalar linear *regression*

$$z_t = \varphi_t^* h_t,\ t = 0, 1, 2 \ldots \tag{2.2}$$

in which the time varying parameter vector h_t is multiplied on the input variable vector φ_t^* producing the scalar output signal z_t. Both signals and parameters may be complex valued. In this model the components of φ_t^* are called regressors, and constitute input signals which are known at time t. Here both h_t and φ_t have dimension $n_h|1$. The superscript "*" denotes complex conjugate transpose. We will in this model refer to the parameters h_t as the *true parameters* describing the characteristics of the linear system. The signal z_t is therefore assumed to be generated by some mechanism based on these parameters. The actual values of

h_t are only available to us in situations where we analyze different methods or algorithms by computer simulation, not in real situations. Therefore, if we want to measure, and maybe later model z_t by *estimating* the assumed ”true” hidden system parameters, we have to use some apparatus designed to obtain data samples from the signal z_t. However, no matter how good this apparatus is, there will always exist some limitations that prevents us from obtaining an error free measurement of the signal z_t. Thus, in the description of the measured version of the signal z_t we will have to add an additional term representing this imperfection. We therefore replace (2.2) by

$$y_t = \varphi_t^* h_t + v_t. \tag{2.3}$$

The noise term v_t represents everything in the signal y_t that can not be explained by the linear regression model (2.2). We will further assume that the noise v_t has zero mean and is statistically independent of both the vector h_t and of the regressors φ_t^*. Equation (2.3) is illustrated by Figure 2.1.

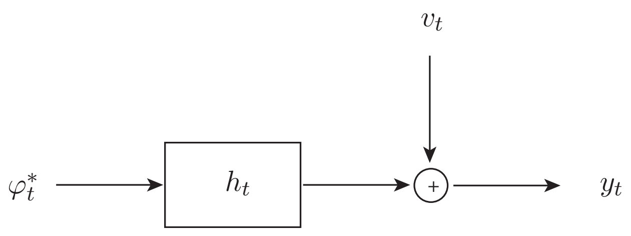

Figure 2.1: A linear regression with additive noise describing the relationship between an input and an output signal disturbed by noise. This representation will be used for the true system that is assumed to generate the scalar measurement time-series y_t.

Let us now introduce the following *linear regression model*

$$\begin{aligned} y_t &= \hat{y}_t + \varepsilon_t \qquad (2.4)\\ \hat{y}_t &= \varphi_t^* \hat{h}_t. \qquad (2.5)\end{aligned}$$

Here $\hat{y}_t$ represents an estimate of the measured signal y_t. The column vector $\hat{h}_t$ of dimension $n_h|1$ is an estimate of the true time varying h_t. The *estimation error* is

$$\varepsilon_t = y_t - \hat{y}_t \quad . \tag{2.6}$$

Furthermore, we also define the *parameter tracking error*

$$\tilde{h}_t = h_t - \hat{h}_t \tag{2.7}$$

as the difference between the true parameter vector h_t and the estimated parameter vector $\hat{h}_t$. Our goal is to adjust the time varying parameter (the elements of $\hat{h}_t$), so that either the estimation error (2.6) or the parameter tracking error (2.7), is minimized in some sense. The adjustment of the parameters $\hat{h}_t$ is to be performed

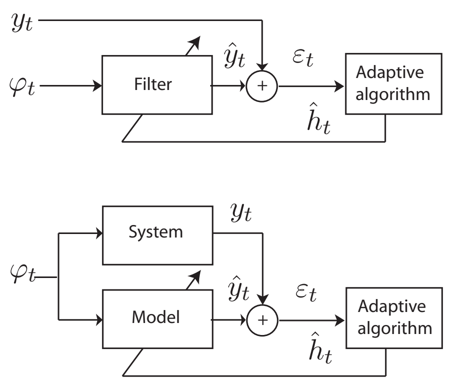

Figure 2.2: The regression model (2.4) illustrated as direct adaptation (upper diagram) and system identification (lower diagram).

recursively by a suitable adaptation algorithm, based on the information contained in the error signal ε_t. The regression model (2.4) is of use in two types of problems, either *direct adaptation* where $\hat{y}_t$ is the output of a filter that is to be tuned so that it follows y_t, or problems where (2.5) is a linear regression model that describes a measured signal y_t. See Figure 2.2.

EXAMPLE 2.1

An example of the linear regression model (2.4) is the second order finite impulse response (FIR) model

$$\begin{aligned} y_t &= \hat{h}_t^0 u_t + \hat{h}_t^1 u_{t-1} + \hat{h}_t^2 u_{t-2} + \varepsilon_t && (2.8)\\ &= \varphi_t^* \, \hat{h}_t + \varepsilon_t, && (2.9) \end{aligned}$$

where

$$\varphi_t^* = (u_t \; u_{t-1} \; u_{t-2}) \quad ; \quad \hat{h}_t = (\hat{h}_t^0 \; \hat{h}_t^1 \; \hat{h}_t^2)^T, \qquad (2.10)$$

where the regression vector in (2.10) consists of delayed known input signals u_t.

2.2 Performance measures and adaptive parameter estimation

A common criterion used in the design of adaptive algorithms is the mean-square value of the estimation error. Substituting (2.5) into (2.6) yields

$$\varepsilon_t = y_t - \varphi_t^* \hat{h}_t \quad . \tag{2.11}$$

By taking the absolute value and squaring (2.11), we obtain

$$|\varepsilon_t|^2 = |y_t|^2 + \hat{h}_t^* \varphi_t \varphi_t^* \hat{h}_t - 2Re\{\varphi_t^* y_t^* \hat{h}_t\} \quad . \tag{2.12}$$

Then, the mean square estimation error (MSE) is expressed as the expected value of (2.12)

$$MSE = E|\varepsilon_t|^2 = \left(E|y_t|^2 + \hat{h}_t^* \mathbf{R} \hat{h}_t - 2\Re\{p^* \hat{h}_t\} \right) \tag{2.13}$$

where the average $E(\cdot)$ is taken with respect to the properties of all regressor variables and v_t in (2.3). The criterion based on the MSE can then be defined as

$$J = \frac{1}{2} \, MSE \quad . \tag{2.14}$$

The regressor vector φ_t is assumed to have zero mean and covariance matrix

$$\mathbf{R} = E\{\varphi_t \varphi_t^*\} \quad . \tag{2.15}$$

The use of (2.12) and introduction of the cross correlation vector between the regressors and the output,

$$p_t = E\{\varphi_t y_t\} \quad , \tag{2.16}$$

gives the last equation of (2.14). The parameter values $\hat{h}_t$ that minimize the criterion J are obtained by differentiating (2.14) with respect to $\hat{h}_t$. We then obtain the well known least square solution

$$\hat{h}_t = \mathbf{R}^{-1} p_t . \tag{2.17}$$

The solution to this equation can be found either by direct calculation or recursively, by means of adaptive algorithms. Direct calculation works well for time-invariant systems whereas adaptive recursive solutions are more suited for time-varying scenarios. In addition to the criterion (2.14), another performance measure

$$\mathrm{E}\,|\tilde{h}_{t+k}|^2 = \mathrm{E}\,|h_{t+k} - \hat{h}_{t+k|t}|^2 \quad , \tag{2.18}$$

will be used in the algorithm design later in this chapter. Here $\hat{h}_{t+k|t}$ is an estimate of h_{t+k} at time t representing filtering ($k = 0$), prediction ($k > 0$), or fixed lag smoothing ($k < 0$). This measure differs from (2.14), since the goal is here to minimize the mean-square of the *parameter tracking error*, instead of the measurement signal *estimation error* (2.6).

Now, let us introduce a class of adaptation algorithms described by

$$\hat{h}_{t+1} = \hat{h}_t + \mu f(\varphi_t, \varepsilon_t). \tag{2.19}$$

Here, μ is denoted the *step-size* of the algorithm, and $f(\varphi_t, \varepsilon_t)$ is a vector function operating on the regressor φ_t and the error signal ε_t. It is assumed to have the property that the norm of $f(\cdot)$ vanishes for small ε_t. The step-size μ controls the adjustment of the parameters h_t from one time-step to the next by appropriate scaling of $f(\varphi_t, \varepsilon_t)$. When $\hat{h}_t$ is close to h_t i.e. when $f(\varphi_t, \varepsilon_t)$ is small, due to a small error ε_t, the adjustment will be small and vice versa. Specific such algorithms and in particular the step-size μ will be further discussed in subsequent sections. For now we will only use (2.19) when we need to refer to an adaptive algorithm in general.

Regarding the properties of the adaptive algorithm, we are here mostly interested in the convergence rate and the excess mean square error (EMSE).

- The *convergence rate* is a measure of the rate for the adaptive algorithm to approach the optimal parameters values h_t, starting at an initial position $\hat{h}_{t,init}$, usually set to zero.
- The *excess mean square error* is defined as the difference between the mean square error J produced by the adaptive algorithm and the *minimum* MSE value, J_{min}, produced by inserting the optimal parameter values h_t from the LS solution (2.17) into equation (2.14) . This can be expressed as

  $$J_{ex} = J - J_{min}. \tag{2.20}$$

  Since the adaptation of the parameters $\hat{h}_t$ is controlled by the *step-size* μ of an adaptive algorithm of the form (2.19), the steady state EMSE is to a large extent determined by the value of this parameter.

Remark: When we in this thesis discuss *convergence*, we mean *convergence in the mean square*, see [1]

Consider a two-dimensional parameter vector $\hat{h}_t = [\hat{h}_t^1 \;\; \hat{h}_t^2]^T$, that is to be adjusted by the adaptive algorithm (2.19) such that the criterion (2.14) is minimized. The minimum is attained at the solution to equation (2.17). To start with, we will assume that the system is time-invariant, i.e, the true parameter vector h_t is not changing over time. This is a *parameter estimation* problem that we are about to solve with an adaptive algorithm[1]

By evaluating equation (2.14) for a range of values of the parameters h_t^1 and h_t^2 we obtain a quadratic error surface. Its minimum point, located at $[h_t^1, h_t^2]$ is denoted the MMSE point. The mission of our adaptive algorithm is to adjust the estimated parameter vector $\hat{h}_t = [\hat{h}_t^1 \ \hat{h}_t^2]^T$ such that the MMSE point is reached.

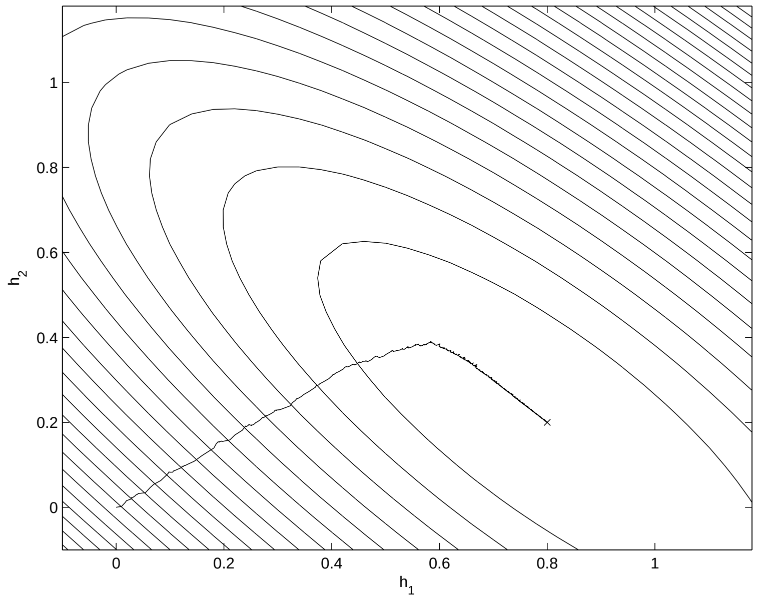

Figure 2.3: This figure illustrates the adaptation process of two parameters towards the MMSE point of a performance surface generated by correlated regressors. The coefficients $\hat{h}_1$ and $\hat{h}_2$ were initiated to zero at the beginning of the adaptation. The LMS adaptation algorithm is used.

It is well known [1],[2],[3] that the shape of the performance surface is very important when its comes to the performance of the adaptive algorithm used to find the minimum point. Since the shape of the performance surface is uniquely determined by the eigenvalues and eigenvectors[2] of the auto correlation matrix $\mathbf{R}$,

[1]Parameter estimation can be performed in a number of different ways. If the system to be estimated is not changing over time, within a considered time interval, then it is possible to use off-line methods (batch methods) in order to find the estimates of the parameters.

[2]The eigenvectors of the autocorrelation matrix $\mathbf{R}$ define the principal axes of the performance

the adaptation of the parameters vector $[\hat{h}_t^1 \; \hat{h}_t^2]^T$ towards the MMSE point will be affected by the properties of the regressors φ_t. Figure 2.3 illustrates the trajectory towards the minimum point for a well known adaptive algorithm called the LMS algorithm[3] in the case of correlated, or colored regressors. Here it can be noted that the trajectory does not progress directly towards the minimum point. This is a result of the combination between *colored* regressors φ_t, and the fact that the LMS algorithm does not include information about this in its structure. In the case of uncorrelated, or *white* regressors the trajectory would have progressed directly towards the MMSE point.

However, use of knowledge of the structure of the adaptation algorithm improves the convergence rate and enables the average trajectory of $\hat{h}_t$ to be directed directly towards the MMSE point. The shape of the error surface might also change over time. This will happen if the regressor statistics change during the adaptation process and thus change the corresponding autocorrelation matrix $\mathbf{R}$. In order to deal with this problem, methods for updating the autocorrelation matrix, or its inverse, need to be included in the structure of the adaptive algorithm. This increases the complexity of the algorithm.

2.3 Tracking time-varying parameters

So far we have discussed recursive parameter estimation for *time-invariant* systems, i.e. where the parameters h_t do not change over time. We are now faced with another type of parameter estimation problem, in which the parameters h_t change.

Adaptive parameter estimators can be viewed as a collection of different methods used to find estimates of the unknown system parameters h_t, that may behave in the following ways:

- Never changing, i.e static parameters
- Rarely changing, maybe by large amplitude
- Continuously changing

Tracking can be seen as a subset of parameter estimation where the true parameters are continuously changing.

surface. The eigenvalues of the autocorrelation matrix $\mathbf{R}$ define the slope of the performance surface along the principal axes. See [3] for more information.

[3]This algorithm will be introduced later in this chapter.

Since the ”true” parameters are changing with time, the position of the MMSE point in the coordinate system will also change. Therefore, we are not only interested in finding the bottom of the bowl, our aim is also to track the changes of the MMSE point caused by the continuously changing parameter vector h_t. This is illustrated by Figure 2.4.

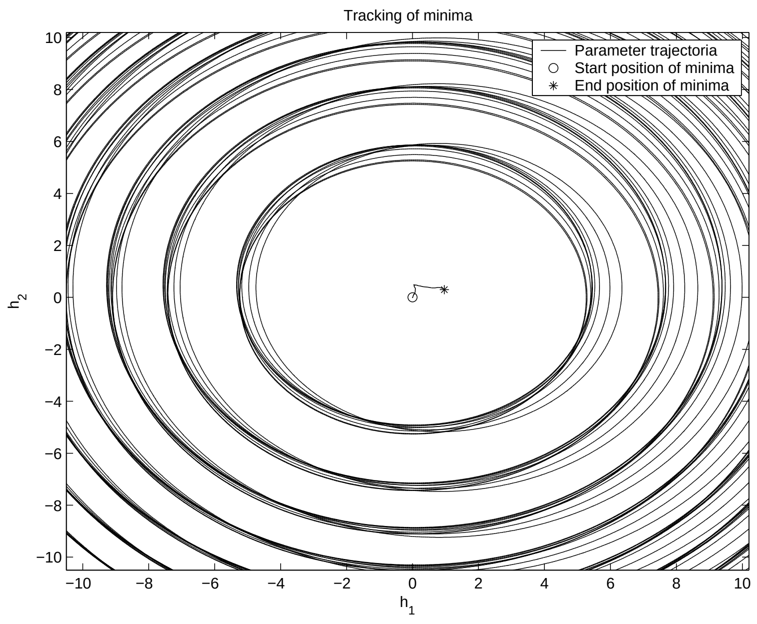

Figure 2.4: Tracking of two time varying parameters. The tracking process starts at one point (o), then continues in the direction of the moving MMSE minimum point until the end of the adaptation (*).

Regarding the tracking properties of adaptive algorithms, we will be mostly concerned with the steady-state performance, i.e. how the algorithm behaves when tracking the MMSE point after the initial transients have decayed. However, since abrupt changes may also occur in tracking problems, fast initial convergence is also an important property.

As a prerequisite for subsequent discussions we will regard abrupt changes and smooth changes of the time varying parameter as two different situations that results in two different modes of the adaptation. The reason for this is that abrupt changes of the parameter vector h_t might change the appropriate tuning of the adaptive algorithm completely.

Important to note here is that these ”modes” are only used to describe different phases of the adaptation, they are not properties of the adaptive algorithm itself or clearly distinguishable in real tracking problems[4].

2.3.1 The transient mode

Assume that the system is time-varying and suddenly may change its behavior. The MMSE point will then move to a new position in the coordinate system due to the new values of the true parameters[5]. This forces the algorithm to start search for the new position of the MMSE point.

The algorithm is said to operate in the transient mode from when the true parameter values h_t jump to the new position and thereby moves the MMSE minimum point to a new position until the algorithm has adapted to the new true parameter values h_t.

In *transient* mode, we would like the tracking algorithm to use a large step-size for fast adaptation to the sudden changes of the parameter values, this is illustrated in Figure 2.5. This statement is based on algorithm structure (2.19) and that the size of $f(\varphi_t, \varepsilon_t)$ in some sense represents the closeness to the MMSE point. If the value of $f(\varphi_t, \varepsilon_t)$ is large, then the algorithm is far from the minimum and therefore needs the large step-size to quickly approach the minimum. However, a large step-size results in a large EMSE in steady state. This is clearly observed in the lower plot of Figure 2.5 where the trajectory is seen to fluctuate around the MMSE point. This in turn means that the variance of estimated parameters $\hat{h}_t$ is large (Figure 2.5, upper diagram). Therefore, we would like the algorithm to use a smaller step-size as soon as the transient phase is over, since this will decrease the EMSE and the variance of the parameters $\hat{h}_t$ in steady state. This, however, results in a slower convergence rate, see Figure 2.6. The smaller fluctuations around the MMSE point due to the smaller step-size is depicted in the lower plot of Figure 2.6.

2.3.2 The tracking mode

When the tracking algorithm has passed through the transient phase and entered a *steady-state* behavior it is said to operate in *tracking mode*. This is typically

[4]Different adaptation modes can, of course, be included in the design of the adaptive algorithm such that the algorithm switches between the modes depending on the current situation. See [8].

[5]The variations of the parameters h_t are here assumed to normally be fairly smooth such that an abrupt change of the parameter values clearly deviates from their past behavior.

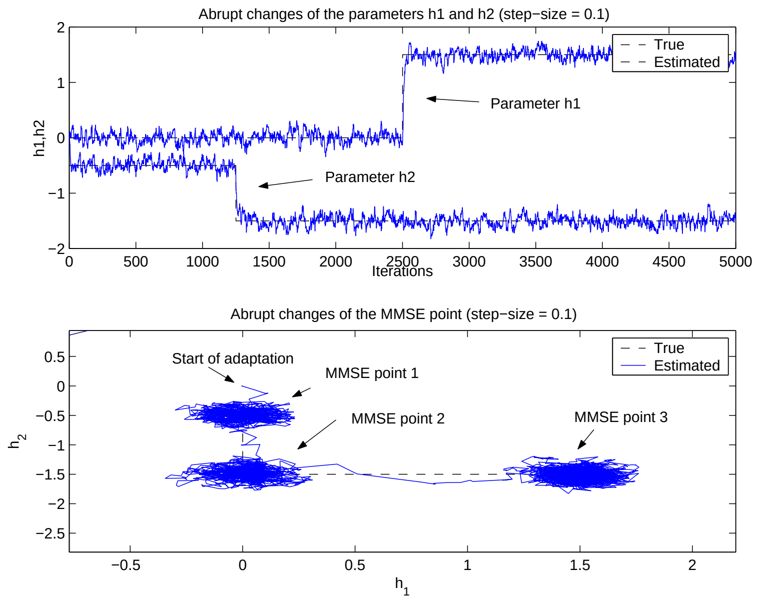

Figure 2.5: Upper diagram: Abrupt changes of the parameter vector h_t. First parameter h_2 changes its value from -0.5 to -1.5 then parameter h_1 changes its value from 0 to 1.5. The goal of the adaptive algorithm is to reach the vicinity of the new parameter values as fast as fast as possible. Here, fast adaptation is obtained by a large step-size ($\mu = 0.1$) at the price of a large EMSE. Lower diagram: The corresponding "jump" in the MMSE minimum point caused by the parameter changes in the upper figure. The adaptation is initiated at the origin and we can clearly see the behavior of the algorithm in the beginning of the adaptation and after the abrupt parameter change where the estimated parameter trajectory aims for the new MMSE minimum.

the "normal" operating mode of the algorithm. In this case the parameters are undergoing small changes between the sampling instants, and no abrupt changes or changes of underlying dynamics governing the parameter variations are assumed. The parameters are assumed to change as in Figure 2.7 or in Figure 2.8.

In *tracking* problems, things tend to be more complicated as compared to static parameter estimation since we are dealing with a parameter vector that is continuously changing. The value of the step-size that produces the smallest error[6], is now depending on the rate of change of the time-varying parameters. For analy-

[6]At this point we only say "error", since we have not yet specified which one of the different errors: the estimation error ε_t, or the parameter tracking error $\tilde{h}_t$, that we strive to minimize.

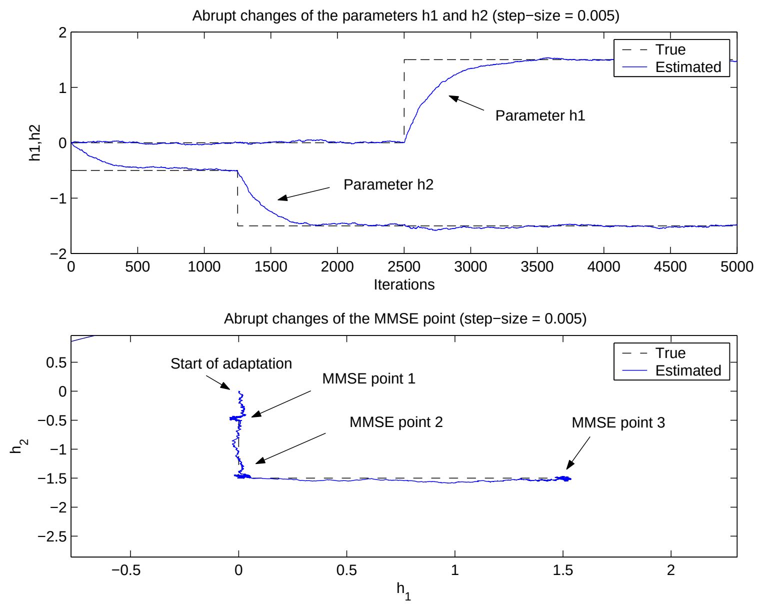

Figure 2.6: Upper diagram: Abrupt changes of the parameter vector h_t. First parameter h_2 changes its value from -0.5 to -1.5, then parameter h_1 changes its value from 0 to 1.5. The goal of the adaptive algorithm is to reach the vicinity of the new parameter values as fast as fast as possible. Here, a small EMSE is obtained by a small step-size ($\mu = 0.005$) at the price of slow adaptation. Lower diagram: The corresponding "jump" in the MMSE minimum point caused by the parameter changes in the upper figure. The adaptation is initiated at the origin and we can clearly see the behavior of the algorithm in the beginning of the adaptation and after the abrupt parameter change where the estimated parameter trajectory aims for the new MMSE minimum.

sis purposes, the total parameter error $\tilde{h}_t$ produced by the adaptive algorithm is sometimes divided into two parts,

$$\tilde{h}_t = \tilde{h}_{t,lag} + \tilde{h}_{t,noise}$$

1. *The weight vector noise* $\tilde{h}_{t,noise}$: the difference between the estimated parameter value $\hat{h}_t$ and the expected value of the estimated parameter value $\hat{h}_t$ due to the adaptation of the parameters, see Figure 2.9.

$$\tilde{h}_{t,noise} = E\{\hat{h}_t\} - \hat{h}_t \tag{2.21}$$

2. *The lag-error* $\tilde{h}_{t,lag}$: defined as the difference between the optimal parameter

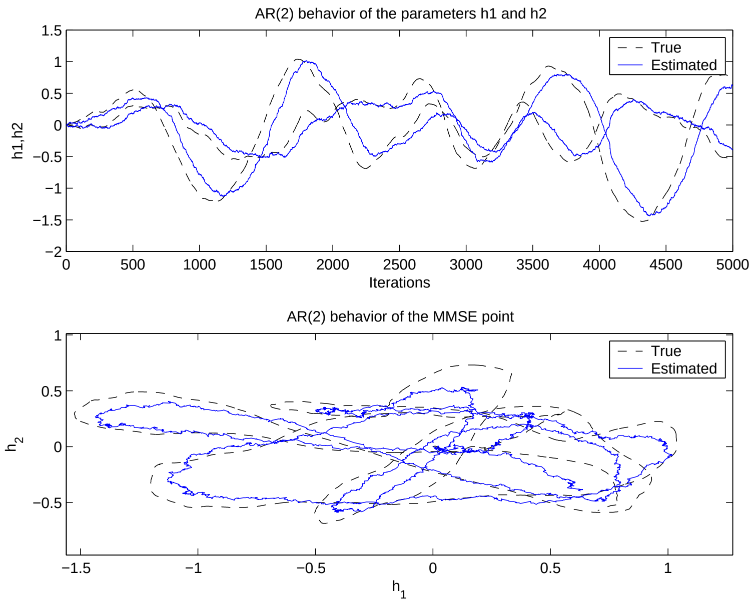

Figure 2.7: Upper diagram: This figure illustrates tracking of two AR(2) parameters. The goal of the adaptive algorithm is to follow the time varying parameters as well as possible. Lower diagram: Illustration of the moving MMSE point caused by the parameter variations in the upper plot.

value h_t and the expected value of estimated parameter value $\hat{h}_t$, see Figure 2.9.

$$\tilde{h}_{t,lag} = h_t - E\{\hat{h}_t\} \quad . \tag{2.22}$$

Here the expected value $E\{\hat{h}_t\}$ is taken over an ensemble of estimates of the parameters $\hat{h}_t$ based on different realizations of the noise v_t and the regressors φ_t^* in (2.3). The weight vector noise is present in both estimation of static parameters and tracking problems, whereas the lag-error only appears in tracking problems due to the time-varying nature of the parameters h_t. The lag-error for tracking problems can be compared to the bias in a system identification problem for static parameters.

A dilemma, regarding the performance of the tracking algorithm is that it is not possible to minimize these two types of errors simultaneously. The reason for this is that a large step-size produces a small lag-error but results in a large weight vector noise and vice versa. Therefore, a trade-off between the lag-error and the weight error noise is necessary when calculating the optimal step-size.

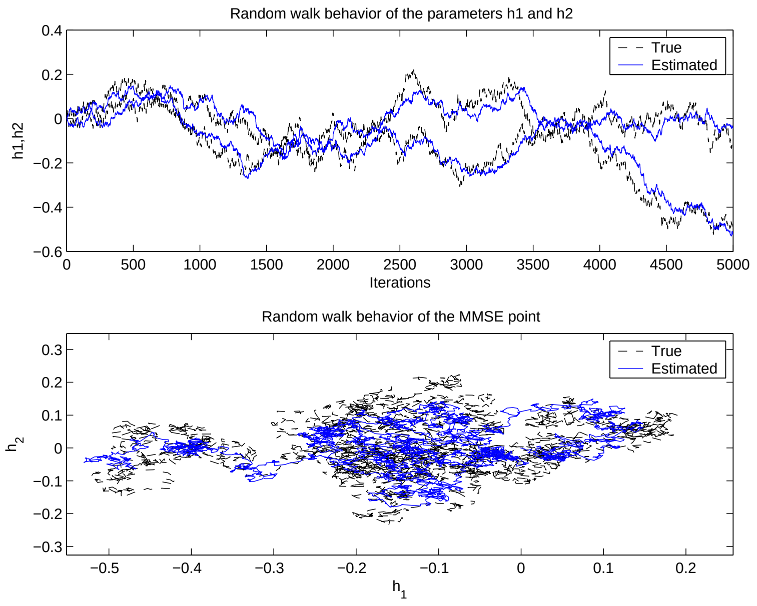

Figure 2.8: Upper diagram: This figure illustrates tracking of two parameters evolving as random walks. Lower diagram: Illustration of the moving MMSE point caused by the parameter variations in the upper plot.

2.4 Search and tracking strategies

Adaptive methods for adjusting the parameter vector $\hat{h}_t$ can be thought of as a numerical search for the true parameter vector h_t that minimizes the criterion J in (2.14). There exist a number of different search strategies that can be used in the adaptation process. The common property for these methods is that they compute adjustment vectors Δh_t to an assumed model

$$\hat{h}_{t+1} = \hat{h}_t + \Delta h_t \tag{2.23}$$

in order to decrease the value of J, on average. Recall the adaptive algorithm (2.19). In that case the adjustment vector Δh_t was expressed as $\mu f(\varphi_t, \varepsilon_t)$, where the step-size μ was used as scaling factor. Among the many different adaptive methods [1],[2],[3] used to adjust the parameter vector $\hat{h}_t$, it is natural to first consider *gradient* algorithms, such as the steepest descent, see [1]. In these methods the update steps Δh_t are on average taken in the direction of the negative gradient of the parameter surface in order to minimize J. The gradient vector is perpendicular to the level curves of the performance function J. For a given possibly

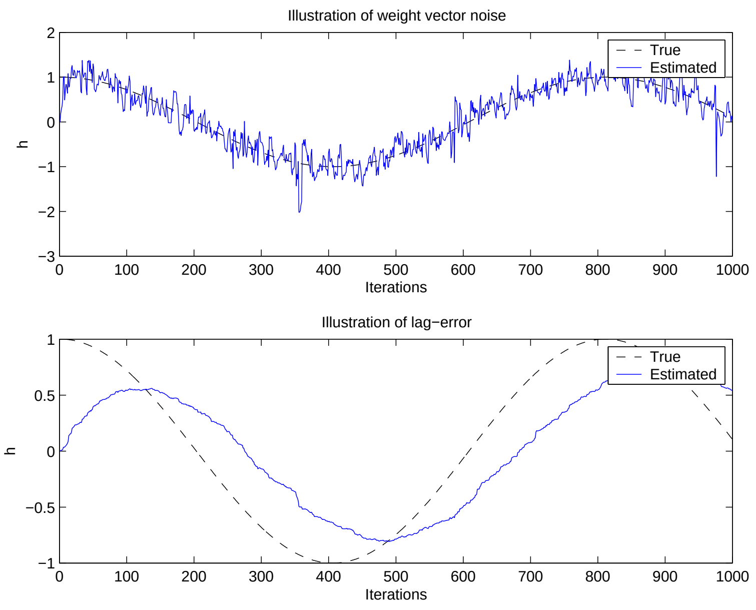

Figure 2.9: Upper diagram: Weight vector noise due to a large step-size. Lower diagram: Lag-error due to a small step-size.

complex-valued fixed parameter vector $h_t = h$, the gradient vector of the criterion (2.14) can be calculated as [1]

$$\begin{aligned}
\nabla_h &= 2\left(\frac{\partial J(h)}{\partial h^*}\right) = 2\left(\frac{\partial}{\partial h^*}\frac{1}{2}[E\{(y_t - \varphi_t^* h)^*(y_t - \varphi_t^* h)\}]\right) && (2.24)\\
&= 2\,\frac{1}{2}\frac{\partial}{\partial h^*}[E\{y_t^* y_t - y_t\varphi_t^* h - h^*\varphi_t y_t + h^*\varphi_t\varphi_t^* h\}] && (2.25)\\
&= E\{[-\varphi_t y_t + \varphi_t\varphi_t^* h]\} && (2.26)\\
&= -E\{\varphi_t[y_t - \varphi_t^* h]\} && (2.27)\\
&= -E\{\varphi_t\varepsilon_t\} && (2.28)
\end{aligned}$$

By using (2.27) the negative gradient can also be written in terms of the auto-correlation matrix $\mathbf{R}$ of the regressors φ_t and the cross-correlation p between the input and output signals as

$$\nabla_h = \mathbf{R}h_t - p_t \quad . \qquad (2.29)$$

If it would have been possible to obtain an exact measure of the gradient (2.28), i.e. if we knew the true parameters h_t, which we do not, we could design a recursive algorithm that decreases the criterion function (2.14) by changing the parameters

$\hat{h}_t$ along the negative gradient direction according to

$$\hat{h}_{t+1} = \hat{h}_t - \mu_t \nabla_h \tag{2.30}$$

Here, we have introduced the possibly time-varying step-size parameter μ_t which scale the steps taken in the direction of the negative gradient. Normally, μ_t is a scalar parameter, however, it is also possible to substitute μ_t with a diagonal matrix with different values along the diagonal in order to individually tune the element in the parameter vector h_t. This increases the flexibility of the algorithm and can be used if the parameters are assumed to vary at different rates.

A drawback with gradient algorithms such as the steepest descent is that they suffer from slow convergence if the performance surface is skew due to correlated regressors. This problem motivates more advanced algorithms such as the *Newton method*. Here, the *Hessian*, i.e. the second derivative at the point $h_t = h$ becomes useful. The Hessian is obtained as follows

$$\frac{\partial}{\partial h}\left(\frac{\partial J(h)}{\partial h^*}\right) = \frac{\partial}{\partial h}[-E\{\varphi_t(y_t - \varphi_t^* h)\}] = E\{\varphi_t \varphi_t^*\} \ . \tag{2.31}$$

We here note that this equals the autocorrelation matrix $\mathbf{R}$ of the regressors for linear regression models with correct structure. By multiplying the negative gradient with the inverse of the Hessian, we obtain Newton's method,

$$\hat{h}_{t+1} = \hat{h}_t + \mu_t \mathbf{R}^{-1} E\{\varphi_t \, \varepsilon_t\} \ . \tag{2.32}$$

By using the inverse of $\mathbf{R}$ in (2.32) the search direction will always be towards the minimum of J which is not the case in general for steepest descent. This can be explained by thinking of the inverse of $\mathbf{R}$ as a tool to transform the elongated performance surface (see Figure 2.3) produced by colored regressors back into a circular performance surface as in the case of uncorrelated regressors.

2.4.1 The LMS algorithm

Since we are not able to measure the exact gradient ∇_h due to the lack of knowledge about the true parameters h_t it is not possible to realize the algorithm (2.30). In order to circumvent this problem an approximation of the exact gradient has to be used. The estimate,

$$\hat{\nabla}_h = \varphi_t \varepsilon_t \ , \tag{2.33}$$

called the *instantaneous gradient of the criterion J with respect to the parameters h*, will on average coincide with the exact gradient (2.28) [1]. In situations with high noise variance, this approximation of the gradient is very uncertain. By using (2.33) in (2.30) we obtain, by using a fixed step-size, the famous LMS algorithm[7] [3]

$$\begin{aligned} \varepsilon_t &= y_t - \varphi_t^* \hat{h}_{t|t-1} && (2.34) \\ \hat{h}_{t+1|t} &= \hat{h}_{t|t-1} + \mu \varphi_t \varepsilon_t, && (2.35) \end{aligned}$$

which was introduced in the 1960's by Widrow and Hoff. The LMS algorithm has since its introduction been frequently used in various applications due to its low complexity. It is often a first choice among designers and it has also become the standard algorithm against which other algorithms are benchmarked.

In a time-invariant case it is well known [1],[3] that the LMS algorithm suffers from slow convergence in situations where the eigenvalue spread of the regressor covariance matrix is large. This has to do with its simple structure which only takes into account an approximation of the exact gradient. In its original form (2.35) the main focus is on simplicity, nothing else. In the literature, many different modifications of the LMS algorithm exist, and common to all these variants is that they in one way or the other strive to improve the performance of the original LMS algorithm at the price of increased complexity. One example is the LMS/Newton method

$$\begin{aligned} \varepsilon_t &= y_t - \varphi_t^* \hat{h}_{t|t-1} && (2.36) \\ \hat{h}_{t+1|t} &= \hat{h}_{t|t-1} + \mu \mathbf{R}^{-1} \varphi_t \varepsilon_t, && (2.37) \end{aligned}$$

which solves the slow convergence problem for correlated input signals. However, it requires that the inverse of $\mathbf{R}$ is known. Another frequently used version of the LMS algorithm is the so called *normalized* LMS algorithm. Here, the step-size μ_t is normalized by the squared norm of the regressors in order to make the algorithm performance independent of the size of the input signal. The normalization is performed as

$$\mu_t = \frac{\mu_0}{\alpha + \varphi_t^* \varphi_t} \quad . \tag{2.38}$$

The parameter α is normally a small number used to prevent division by zero. For a thorough description of LMS-like and other adaptive algorithms, see e.g [1], [2], or [3].

[7] We here introduce the notation $\hat{h}_{t+1|t}$ to emphasize that the estimate of h_{t+1} depends on measured data up to time t.

2.5 Parameter model based tracking

When it comes to tracking of time varying parameters it is of interest to incorporate knowledge about the dynamics of the parameters into the adaptive algorithm to improve the tracking performance. This possibility is of particular interest in applications such as e.g communication over fading radio channels where the characteristics of the channel is well known. This is not possible in the present form of the LMS algorithm. We shall therefore discuss more powerful alternatives to the LMS algorithm next. However, we will first start by introducing the concept of *hypermodels*, i.e, the way we represent prior knowledge about parameter variations.

2.5.1 Hypermodels

A Hypermodel is a mathematical model that is used to describe the dynamics of a time-varying parameter. The use of hypermodels in adaptive algorithms has been discussed previously by e.g. Benveniste et al in [5], Grenier in [9], Kitagawa and Gersch in [10] and Lindbom et al in [4],[11],[12],[13],[14].

In the beginning of this chapter we assumed that the signal y_t was generated by the *true* time varying parameter vector h_t together with the regressors φ_t^*. We will now regard the parameters h_t as time series generated by

$$h_t = \mathcal{H}(q^{-1})e_t \quad . \tag{2.39}$$

where the rational matrix $\mathcal{H}(q^{-1})$ of dimension $n_h|n_h$ denotes the *hypermodel*, and e_t is a noise vector. In most problems, e_t is assumed to have zero mean and to be stationary. It has covariance matrix $\mathbf{R_e}$. Here, q^{-1} is the backward shift operator, i.e. $q^{-1}x_t = x_{t-1}$. The purpose of this model is to represent the designers knowledge, or assumptions, about the second order moments of the time-varying parameters h_t. It can be noted that a more general form of the hypermodel (2.39) is when the matrix $\mathcal{H}$ is time-variable. The matrix $\mathcal{H}(q^{-1})$ describes the second order moments of the variations of h_t as well as the correlation between different elements h_t^i and h_t^j in the parameter vector h_t. A special case of (2.39) is when $\mathcal{H}(q^{-1})$ is diagonal. Each component h_t^i of h_t is then represented by a separate scalar ARMA model.

$$D_i(q^{-1})h_t^i = C_i(q^{-1})e_t^i \quad , i = 1, ... \, n_h \quad , \tag{2.40}$$

where $D_i(q^{-1})$ and $C_i(q^{-1})$ are polynomials in the backward shift operator. By modeling the true parameters h_t like this, i.e. as stochastic processes, it is possible to describe the variations of h_t in a convenient way by the choices of the polynomials $C_i(q^{-1})$ and $D_i(q^{-1})$[8]. The correlation between the components are furthermore described by the covariance matrix of the driving noise

$$\mathbf{R}_e = E[e_t e_t^*] \quad . \tag{2.41}$$

In this work, we will however restrict the class of hypermodels to

$$\mathcal{H}(q^{-1}) = \frac{C(q^{-1})}{D(q^{-1})}\mathbf{I} \, , \tag{2.42}$$

where we have equal transfer functions along the diagonal. This means that the components of h_t are modeled to have the same dynamics. This might at first seem to be a severe restriction. However, it is often a good compromise between generality and complexity of the resulting tracking algorithm.

One of the key issues in the design of tracking algorithms based on parameter models is to obtain information about the parameters h_t such that a hypermodel, $\mathcal{H}(q^{-1})$, can be established and included in the design. In applications where the dynamics of the parameters h_t are assumed to be stationary, off-line estimation is possible. In systems where the dynamics change over time, on-line estimation is required. If estimation of the hypermodel is not possible, then the designer is faced with the problem of choosing a hypermodel that works well on average.

In [4],[14], possible hypermodels representing different parameter variations are discussed. These can be summarized as:

- RW (Random walk): Obtained by choosing

$$C(q^{-1}) = 1, \quad D(q^{-1}) = 1 - q^{-1} \quad . \tag{2.43}$$

  This choice represents that the designer has *no information* about the increments of the parameters h_t. By ”no information” we mean that if this hypermodel is used, then we assume that the parameters are evolving as random walks, i.e. without any correlation between increments $h_{t+1} - h_t$.

- FRW (Filtered Random Walk):

$$C(q^{-1}) = 1, \quad D(q^{-1}) = (1 - q^{-1})(1 - aq^{-1}) \tag{2.44}$$

[8]When considering a time-varying hypermodel $\mathcal{H}_t$, equation (2.40) becomes changed into $D_{i,t}(q^{-1})h_t^i = C_{i,t}(q^{-1})e_t^i$. This form may also include on-line tuning of the polynomials $C_{i,t}(q^{-1})$ and $D_{i,t}(q^{-1})$ in order to adjust for changes of the parameter statistics.

with $|a| \leq 1$. Here, the parameter a controls the assumed correlation between the parameter increment, and the smoothness of the assumed variation of the parameters. Values close to one represents slow variations. In the case of $a = 1$, the integrated random walk (IRW) is obtained. This model with $a > 0$, is appropriate if the parameters are assumed to evolve in the same direction for a short time.

- AR_2 (Autoregressive second order model): If the parameters h_t are oscillating according to some known or estimated frequency it is possible to design the hypermodel to fit these oscillations by choosing

$$C(q^{-1}) = 1, \quad D(q^{-1}) = 1 - 2p\cos(\omega)q^{-1} + p^2q^{-2} \quad . \tag{2.45}$$

  Here, ω, denotes the oscillation frequency and p is the pole radius, which represents the damping.

An exact representation of the *true* hypermodel is of course not possible to obtain in reality. Some degree of uncertainty is always present in the hypermodel. Depending on the size of this uncertainty more or less inaccurate information will be included in the resulting algorithm. The problem of uncertainty in hypermodels is considered in [4],[15] where a design methodology to cope with this problem is developed.

The parameter based adaptive algorithms presented next aim to estimate $\hat{h}_{t+k|t}$ for different choices of the time-lag k by minimizing the criterion (2.18) based on a given hypermodel.

2.5.2 The Kalman estimator

Let

$$\mathcal{H}(q^{-1}) = \mathbf{H}(\mathbf{I} - q^{-1}\mathbf{F})^{-1}\mathbf{G} \tag{2.46}$$

where $\mathbf{H}$, $\mathbf{G}$, and $\mathbf{F}$ are given by the state-space model

$$\begin{aligned} x_{t+1} &= \mathbf{F}x_t + \mathbf{G}e_{t+1} \\ h_t &= \mathbf{H}x_t \end{aligned} \tag{2.47}$$

denote the hypermodel (2.39). Then the optimal linear MMSE estimator, the Kalman estimator, for $k > 0$, can be expressed as

$$\varepsilon_t = y_t - \varphi_t^* \hat{h}_{t|t-1} \tag{2.48}$$

$$\hat{x}_{t|t} = \hat{x}_{t|t-1} + \mathbf{K}_t \varphi_t \varepsilon_t \tag{2.49}$$

$$\hat{h}_{t+k|t} = \mathbf{H}\mathbf{F}^k \hat{x}_{t|t} = \mathbf{H}\hat{x}_{t+k|t} \tag{2.50}$$

where $\mathbf{K}_t$ represents the Kalman prediction gain. In (2.47), x_t is the state vector at time t of dimension $n_x|1$ and **F**, **G**, **H** are matrices of appropriate dimensions. The matrix $\mathbf{K}_t$ is determined via Riccati difference equations, see [4] for details. The design of the Kalman estimator requires $\mathbf{H}$, $\mathbf{G}$, $\mathbf{F}$ as well as knowledge of the covariance matrix $\mathbf{R_e}$ and the variance of the measurement noise, $\sigma_v^2 = E|v_t^2|$. In the case of random walk modeling, i.e. $\mathbf{F} = \mathbf{G} = \mathbf{H} = \mathbf{I}$, and small parameter drift-to-noise ratios $(tr(\mathbf{R}_e)/\sigma_v^2 << 1)$, it is shown in [16] that the Kalman gain is proportional to $\mathbf{R}_e^{1/2}/\sigma_v$ in steady-state. The iterations of the Riccati difference equations require a substantial amount of calculations even when considering low model orders.

2.5.3 The Wiener LMS (WLMS) algorithm

The major drawback with the Kalman estimator is its complexity. This often limits its use in applications where low complexity is important. With this in mind, a framework for designing tracking algorithms based on Wiener filtering theory has been developed in [4]. This design methodology provides a systematic way of utilizing prior information in the form of hypermodels to obtain optimal adjustment of the adaptation under prescribed complexity constraints. Here the tracking problem is formulated as a Wiener filtering problem, where the optimal Wiener estimator is obtained by solving a spectral factorization and a linear diophantine equation[9]. This method of designing tracking algorithms is flexible since the same basic theory can be used to solve problems of varying computational complexity. Different assumptions about the time varying parameters will result in different levels of complexity for the resulting algorithm.

Given the hypermodel

$$\mathcal{H}(q^{-1}) = \frac{C(q^{-1})}{D(q^{-1})}\mathbf{I} \ , \tag{2.51}$$

and measurements described by the linear regression (2.3), the WLMS adaptation algorithm can be expressed as [4]

$$\begin{aligned} \varepsilon_t &= y_t - \varphi_t^* \hat{h}_{t|t-1} && (2.52) \\ \hat{h}_{t|t} &= \hat{h}_{t|t-1} + \mu \mathbf{R}^{-1} \varphi_t \varepsilon_t && (2.53) \\ \hat{h}_{t+k|t} &= \boldsymbol{\mathcal{P}}_k(q^{-1}) \hat{h}_{t|t} \ . && (2.54) \end{aligned}$$

[9]Here the polynomial method described in [17] is used to solve the Wiener filtering problem. Readers not familiar with this way of obtaining the Wiener solution might gain substantial understanding by considering some of the basic examples in [17].

Here, $\boldsymbol{\mathcal{P}}_k(q^{-1})$, also called the *coefficient prediction-smoothing filter*, is constrained to be a diagonal rational matrix with equal stable and causal transfer functions along the diagonal,

$$\boldsymbol{\mathcal{P}}_k(q^{-1}) = \frac{Q_k(q^{-1})}{Q_0(q^{-1})}\mathbf{I} \quad . \tag{2.55}$$

This filter is specified by the polynomials $Q_k(q^{-1})$ and $Q_0(q^{-1})$ in the backward shift operator q^{-1}. Above, $\hat{h}_{t+k|t}$ is an estimate of h_{t+k} at sample time t, which may involve prediction $(k > 0)$, filtering $(k = 0)$ or fixed-lag smoothing $(k < 0)$. The filter polynomials $Q_k(q^{-1})$, $Q_0(q^{-1})$ in (2.55) are given by the solution to the spectral factorization

$$r\beta(q^{-1})\beta_*(q) = \gamma\ C(q^{-1})C_*(q) + D(q^{-1})D_*(q) \quad , \tag{2.56}$$

which provides a polynomial $\beta(q^{-1})$ and a scalar r, followed by solving the diophantine equation

$$q^k\gamma C(q^{-1})C_*(q) = rQ_k(q^{-1})\beta_*(q) + qD(q^{-1})L_{k*}(q) \quad , \tag{2.57}$$

which provides $Q_k(q^{-1})$, together with a polynomial $L_{k*}(q)$.

In (2.57) the parameter γ denotes the *parameter drift-to-noise ratio* and is defined as

$$\gamma = \frac{tr\mathbf{R}_e}{tr\mathbf{R}_\eta} \tag{2.58}$$

where $\mathbf{R}_e$ and $\mathbf{R}_\eta$ are the covariance matrices of the driving noise e_t in (2.39) and of a noise called *the gradient noise* defined as

$$\eta_t = (\varphi_t\varphi_t^* - \mathbf{R})\tilde{h}_{t|t-1} + \varphi_t^* v_t \quad . \tag{2.59}$$

For more information about the gradient noise, see [4],[11],[12],[13]. By introducing the signal η_t and considering it as white and uncorrelated with h_t, adjustment of the WLMS algorithm can be expressed as a Wiener filter design problem [4], see Figure 2.10, where $\mathcal{W}(q^{-1}) = Q_k(q^{-1})/\beta(q^{-1})$.
The scalar drift-to-noise ratio parameter γ and the polynomials $C(q^{-1})$ and $D(q^{-1})$ are the design variables of the WLMS algorithm and are used for tuning the algorithm. They can be adjusted to minimize the MSE of the parameter tracking error (2.18) if the hypermodel (2.51), and the parameter drift-to-noise ratio γ (2.58), is assumed known[10]. If WLMS-Newton is considered, i.e when using the Hessian

[10]Knowledge about the parameter γ is hard to obtain since it is not trivial to estimate the covariance matrix of the parameter η_t. However, in [11] and [13] an iterative design method used to calculate η_t is presented.

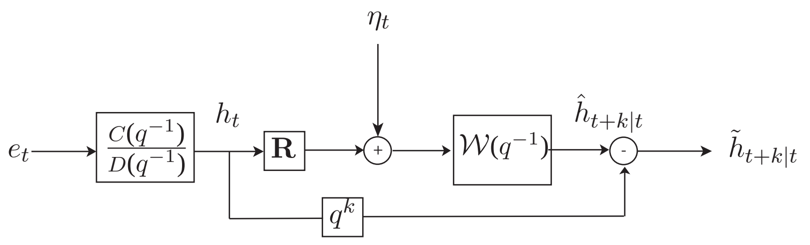

Figure 2.10: The WLMS algorithm expressed as a Wiener filter.

(2.31), in order to improve the convergence rate of the algorithm, then also $\mathbf{R}^{-1}$ need to be known. The optimal step-size μ, in (2.53), is obtained from the first coefficient, Q_0^0, of the polynomial $Q_0(q^{-1})$ according to

$$\mu = Q_0^0 = 1 - \frac{1}{r} \quad , \tag{2.60}$$

where r is given by the solution to the spectral factorization (2.56). The close relationship between the WLMS equations (2.52)-(2.54) and the classical Wiener filter formulation is presented in Appendix A. For more details, and the theory behind the reformulation of the tracking problem to a Wiener filtering problem, see [4],[13].

2.5.4 The Simplified Wiener LMS (SWLMS) algorithm

The name *Simplified Wiener LMS* is related to the initial assumptions about the time-varying parameters h_t and the corresponding hypermodel $\mathcal{H}(q^{-1})$ used in the design of the algorithm. These assumptions can be summarized as

1. The maximum order of the polynomials $C(q^{-1})$ and $D(q^{-1})$ in the hypermodel is restricted to two.

2. All elements of the parameter vector h_t are governed by the same dynamics.

If these assumptions are relaxed, then we have to consider more powerful algorithms such as the *Wiener LMS* algorithm of Section 2.5.3 , the *Generalized Wiener LMS* algorithm, see [13],[4], or the *General Constant Gain* algorithm, see [11], [4]. Due to the design restrictions in the SWLMS algorithm it is possible to calculate the coefficient/prediction filter $\boldsymbol{\mathcal{P}}_k(q^{-1})$ in (2.54) in a much simpler way than for the WLMS algorithm. By reducing the hypermodel to order two, it is possible to obtain a closed form solution for the optimal polynomials $Q_k(q^{-1})$ and $Q_0(q^{-1})$, see Result 3.6 in [4], or Theorem 2 in [13].

Let

$$C(q^{-1}) = 1 \tag{2.61}$$

and

$$D(q^{-1}) = 1 + d_1\, q^{-1} + d_2\, q^{-2} \tag{2.62}$$

define $\mathcal{H}(q^{-1})$ in (2.42). Given the hypermodel $\mathcal{H}(q^{-1})$ (2.42) and a step-size μ we can then calculate $Q_k(q^{-1})$ and $Q_0(q^{-1})$ as

$$Q_k(q^{-1}) = \mu(1\ q^{-1}) \begin{pmatrix} -d_1 & 1 \\ -d_2 & 0 \end{pmatrix}^k \begin{pmatrix} 1 \\ p \end{pmatrix} \tag{2.63}$$

for $k \geq 0$, where the scalar p is defined as

$$p = \frac{d_1 d_2 (1-\mu)}{1 + d_2(1-\mu)} \quad . \tag{2.64}$$

In the WLMS solution, the parameter γ was needed to solve the spectral factorization and the Diophantine equation, cf (2.56) and (2.57), in order to obtain $Q_k(q^{-1})$ and $Q_0(q^{-1})$. The step-size parameter μ serves that purpose in the SWLMS algorithm. However, as can be noted from (2.63) there exist a direct and simple relationship between μ, the design variables d_1 and d_2 and the optimal polynomial $Q_k(q^{-1})$. This makes the SWLMS algorithm tractable from a computational point of view. The step-size parameter μ can therefore be seen as the tuning knob which controls the optimal adjustment of the SWLMS algorithm for a given hypermodel. It should also be noted that this algorithm reduces to the classical LMS algorithm for $d_1 = 1$ and $d_2 = 0$, i.e. when the hypermodel is a random walk model.

2.6 Optimal tuning of tracking algorithms

In order to obtain optimal tracking performance based on the criterion (2.18) it is easy to realize that there is a lot of information that needs to be available for the system designer. In the case of the Kalman estimator, the state-space model, the covariance matrix $\mathbf{R}_e$ and the variance σ_v^2 need to be known. In the WLMS and the SWLMS case, the hypermodel $\mathcal{H}(q^{-1})$, the parameter drift-to-noise ratio γ, and the correlation between η_t and h_t need to be known. When it comes to the classical LMS algorithm it requires the noise variance σ_v^2 and the ratio between the covariance matrices of the regressors φ_t and the driving noise e_t, see [1], [2]. If the Newton direction is to be used in these algorithms, then of course also $\mathbf{R}^{-1}$ is required. Since these important parameters may be hard to obtain, direct on-line tuning of the step-size μ would be of great interest, in algorithms where μ is a main tuning parameter.

By this introduction to the tracking problem and various adaptive algorithms we will now turn our attention towards the SWLMS algorithm and especially the automatic tuning of it. From now on this will be the main focus of this work.

Chapter 3

Step-size adaptation

3.1 Introduction

In the previous chapter we learned that it was possible to achieve optimal tuning of the SWLMS algorithm if the parameter drift-to-noise ratio γ, the correlation between the gradient noise η_t and the parameters h_t were known and perfect agreement between the true parameter variations and the hypermodel was assumed. However, in a real situation it is quite clear that perfect knowledge about these variables is not available since the parameter drift-to-noise ratio γ and the dynamics of the time-varying parameters h_t may change over time. Optimal tracking is therefore difficult to achieve. In such cases, it is up to the system designer to choose appropriate values for the design variables in order to adjust the algorithm to the most likely scenario. However, if the changes in the environment occur very frequently, then the design variables would also need to be updated at the same pace. Unfortunately, as we pointed out in the introduction to this work, it is very common to use a constant step-size together with the LMS- or the SWLMS algorithm when applying them in real applications. This is of course not compatible with the time-varying scenario described above. Our aim in this chapter is therefore to construct a *self-tuning version of the SWLMS algorithm*. By that we mean that the assumptions about the environment that are made in order to tune the algorithm should be reduced to a minimum. The algorithm should be able to adapt to the present situation almost without the help from the system designer. However, if necessary information is available, then it should be used in the best possible way.

The problem of designing self tuning algorithms is not new. Of particular interest in our case is a special class of algorithms aimed for automatic tuning of the step-size in LMS algorithms see e.g. the work in, [18],[19],[6],[20],[21],[22],[23],[24]. These algorithms are called *Variable Step-size algorithms* and are often denoted *VS-algorithms*. The motivation for these algorithms was originally to improve the convergence rate while at the same time decreasing the steady state MSE of the LMS algorithm. This is obtained by using a larger step-size in the beginning of the adaptation and then gradually decreasing the step-size when approaching the minimum[1]. This procedure solves the dilemma mentioned in Chapter 2, when we had to choose between fast adaptation or a small remaining estimation error when choosing the optimal step-size μ.

The common problem when it comes to variable step-size algorithms is *how* to adjust the step-size μ in order to be flexible enough to handle many different tracking situations. This can, of course, be done according to many different strategies (algorithms), some of which are presented next. These algorithms contains one or several design parameters that need to be adjusted. However, the intention is that it should be easier to adjust these parameters than the step-size μ so that the algorithm will work in various tracking scenarios without the help from the designer. In this study we also assume that the step-size μ is real valued and non negative.

3.2 Overview of variable step-size (VS) LMS algorithms

Continuously Decreasing Step-Size (CDSS): Here the step-size is initiated to a large value at the beginning of the adaptation, it is then decreased according to a predefined function

$$\mu_t = f(t)\mu_0 \tag{3.1}$$

where $f(t)$ is some decreasing function of the time index t. For stochastic gradient algorithms it is common to use $\mu_t = \mu_0/t$ since this fulfills the necessary conditions for stability outlined in [25]. This method works well for estimation of time-invariant parameters since the resulting estimation error becomes very small if the step-size is decreasing. However, this method is not aimed for tracking problems since the lag-part of the parameter tracking error $\tilde{h}_t$ will increase as the step-size decreases.

Design variable: $f(t)$

Error based Variable Step-Size (VSS): In this method, proposed in [20], the idea is to connect the adjustment of the step-size μ directly to the size of the in-

[1]The gain of the Kalman filter behaves in this way

stantaneous estimation error ε_t. The equation

$$\mu_{t+1} = \alpha\mu_t + \delta|\varepsilon_t|^2 \quad , \tag{3.2}$$

where the design parameters $0 < \alpha < 1$ and $\delta < 1$ are used for the step-size adjustment. This method is often referred to as the *variable step-size* algorithm or simply VSS. This updating scheme has been criticized for its sensitivity to noise [18].

Design variables: α, δ

Remark: This method can be interpreted as a low-pass filter with a pole in $\alpha < 1$ driven by the squared error signal $|\varepsilon_t|^2$. If the the error signal $|\varepsilon_t|^2 \to 0$ then the step-size $\mu_t \to 0$ as $t \to \infty$. The sensitivity to noise can be explained by writing the instantaneous estimation error, ε_t, as

$$\varepsilon_t = y_t - \hat{y}_t = \varphi_t^* h_t + v_t - \varphi_t^* \hat{h}_t = \varphi_t^*(h_t - \hat{h}_t) + v_t \quad . \tag{3.3}$$

The error can be divided in two parts. One part, $\varphi_t^*(h_t - \hat{h}_t)$, depending on the tracking error $\tilde{h}_t = h_t - \hat{h}_t$ and one part v_t, represented by the measurement noise. Therefore, a large $|\varepsilon_t|^2$ might give the illusion that the parameter estimation error $h_t - \hat{h}_t$ is large, when the real reason for the large error is due to a dominating noise v_t.

Modified (Correlation-based) Variable Step-Size (MVSS): Due to the sensitivity to noise in the VSS algorithm the following updating scheme, often denoted the Modified Variable Step-Size (MVSS) algorithm, was proposed in [18]:

$$\begin{aligned} \mu_{t+1} &= \alpha\mu_t + \delta|p_t|^2 & (3.4)\\ p_t &= \beta p_{t-1} + (1-\beta)\varepsilon_t\varepsilon_{t-1}^* & (3.5) \end{aligned}$$

This algorithm is based on the assumption that the steady-state estimation error ε_t at adjacent time-steps ε_t and ε_{t-1} are uncorrelated, i.e. white, while they are correlated during the convergence phase. Therefore, it is possible to use the time-averaged estimate (3.5) of the autocorrelation at adjacent time-steps in order to adjust the step-size. This method has been shown to work well in situations where the optimal step-size is small i.e. where the time variations of the parameters h_t are slow, or maybe static. However, the tuning of the parameters in this algorithm has turned out to be rather sensitive to the present tracking conditions.

Design variables: α, β,δ

Remark: This method is useful in the case of uncorrelated measurement noise. However, in situations where the measurement noise may be correlated, e.g. echo cancelation problems, see [26] , $E\{\varepsilon_t \varepsilon_{t-1}\}$ is not sufficient as an indicator of the closeness to the minimum of the criterion J.

Robust Modified (correlation-based) Variable Step-Size (RMVSS): Due to the problem of correlated measurement noise v_t in the MVSS algorithm the following algorithm was proposed in [26]

$$\mu_{t+1} = \alpha\mu_t + \delta|p_t|^2 \tag{3.6}$$

$$p_t = \beta p_{t-1} + (1-\beta)\varepsilon_t(\varepsilon_{t-1} + \varepsilon_t)^* \tag{3.7}$$

Here, not only the correlation of the estimation error ε_t at adjacent time-lags is included in the adjustment equation, but also the squared magnitude, $|\varepsilon_t|^2$, of the instantaneous error is used to sense the closeness to the minimum. This improvement robustifies the performance of the algorithm in the presence of correlated measurement noise v_t.

Design variables: α,β,δ

Remark: It should be emphasized that this step-size update, as well as the MVSS algorithm using only $E\{\varepsilon_t \varepsilon^*_{t-1}\}$ in the adjustment of the step-size, suffer from the fact that the noise v_t still may mislead the adaptation.

We will from now on refer to the algorithms above as the *Error Based* (EB) methods. These algorithms are simple in their structure and base the updating of the step-size μ mainly on the instantaneous error ε_t or the correlation of this error.

3.3 Gradient based VS-LMS Algorithms

These methods [6],[21],[27],[28] are based on a recursive gradient descent technique (similar to the LMS algorithm) to adjust the step-size parameter μ in order to minimize the error criterium (2.14). Compared to the methods described above we have found this step-size adjustment technique superior in our search for a suitable candidate to be used together with the SWLMS algorithm. The concept behind this method will now be explained more in detail.

In order to avoid the problem with sensitivity to the noise term v_t in the algorithms described above we need to base the adjustment of the step-size on something more than just the instantaneous estimation error ε_t. Here, the adjustment of the step-size μ is obtained by estimating the gradient of the error criterion J with respect to the step-size parameter μ. As we will see in the next chapter, this will significantly change the behavior of the algorithm as compared to the algorithms presented above. In order to fully understand the concept behind this step-size method we will start by giving a detailed description of a gradient based method first proposed by Benveniste and co-workers in [5]. This step-size updating method, based on the LMS algorithm, will then serve as the base when we later in Section 3.5 develop a new adjustment method for the SWLMS algorithm.

3.3.1 VS-LMS-Benveniste

The idea with this gain adjustment method is to recursively find an estimate of the step-size value μ that minimizes the error (2.14).

Consider the WLMS algorithm (2.52)-(2.54), for the special case $\boldsymbol{\mathcal{P}}_1(q^{-1}) = I$, i.e. the LMS algorithm. Let

$$\nabla_\mu \triangleq \frac{\partial J}{\partial \mu} \tag{3.8}$$

denote the scalar gradient of the criterion J in (2.14) with respect to the step-size parameter μ. In order to recursively adjust the step-size parameter μ in (2.53) we introduce the gradient search

$$\mu_{t+1} = \mu_t - \rho \nabla_\mu \ \ , \tag{3.9}$$

where ρ is the parameter that controls the rate of change for μ_t. Here, the scalar gradient ∇_μ of the criterion J with respect to the *step-size* parameter μ consists of the gradient of the criterion J with respect to the *parameter* h, $\nabla_h = \varphi_t \varepsilon_t$, multiplied by the gradient of the parameter vector $\hat{h}_{t|t-1}$ with respect to *the step-size* μ. As shown in Appendix C, this can be expressed as

$$\nabla_\mu = -\Re\{\psi_t^* \varphi_t \varepsilon_t\} \ \ . \tag{3.10}$$

Here $\Re$ represents the real part and the vector of dimension $n_h|1$, ψ_t, is the gradient of the parameter vector $\hat{h}_{t|t-1}$ with respect to the step-size μ, defined as

$$\psi_t = \frac{\partial \hat{h}_{t|t-1}}{\partial \mu} \quad . \tag{3.11}$$

In Appendix C it is shown that ψ_t is obtained from measured signals via the recursion

$$\psi_t = [I - \mu\varphi_{t-1}\varphi^*_{t-1}]\psi_{t-1} + \varphi_{t-1}\varepsilon_{t-1} \quad . \tag{3.12}$$

In order to obtain a self tuning algorithm, we combine (2.52)-(2.53), (3.9), (3.10) and (3.12) for $\mathbf{R} = \mathbf{I}$ into one scheme

$$\begin{aligned}
\varepsilon_t &= y_t - \varphi_t \hat{h}^*_{t|t-1} && (3.13)\\
\hat{\mu}_{t+1} &= \hat{\mu}_t + \rho\Re\{\hat{\psi}^*_t \varphi_t \varepsilon_t\} && (3.14)\\
\hat{\psi}_{t+1} &= [I - \hat{\mu}_{t+1}\varphi_t\varphi^*_t]\hat{\psi}_t + \varphi_t\varepsilon_t && (3.15)\\
\hat{h}_{t+1|t} &= \hat{h}_{t|t-1} + \hat{\mu}_{t+1}\varphi_t\varepsilon_t \quad . && (3.16)
\end{aligned}$$

Remark: Observe the order of updating the equations. In the literature, e.g. [7] the value $\hat{\mu}_t$ is often used to update $\hat{\psi}_{t+1}$. This, however, is not the best we can do since we can use the latest estimation of the step-size, $\hat{\mu}_{t+1}$ to update $\hat{\psi}_{t+1}$. Furthermore, note also that ψ_{t+1} is replaced by $\hat{\psi}_{t+1}$. The reason for this is that the instantaneous value of the gradient ∇_μ, i.e. $\hat{\nabla}_\mu(t)$ is used in (3.9). For details, see Appendix C.

Design variable: ρ

The main reason why the gradient descent method is considered more interesting than the other step-size schemes mentioned above, is that the updating of the step-size in this case is based on a minimization of a design criterion, rather than an ad-hoc strategy. Also, since our goal is to design an algorithm that is to be tuned almost without any help from the system designer, the number of design variables should be kept to a minimum. For this particular class of adjustment schemes we only have to consider the value of the parameter ρ.

This might however seem strange to some readers. We first start with an algorithm with only one design variable μ. Then, we replace this variable, by another design variable ρ! However, as mentioned before, the whole idea with this is that it should be easier to choose an appropriate value of the parameter ρ than to select an appropriate μ. This will be investigated in the following chapter.

3.3.2 Other VS methods based on the gradient descent technique

In [7] a nice summary of different gradient descent methods is presented. Beside the method presented above a few other interesting variants are mentioned.

1. A multiplicative version of Benveniste's algorithm. This adjustment scheme is written as

$$\hat{\mu}_{t+1} = \hat{\mu}_t[I + \rho \Re\{\hat{\psi}_t^* \varphi_t \varepsilon_t\}] \tag{3.17}$$
$$\hat{\psi}_{t+1} = [I - \hat{\mu}_{t+1}\varphi_t\varphi_t^*]\hat{\psi}_t + \varphi_t\varepsilon_t \quad . \tag{3.18}$$

   This method is, according to the authors of [7], superior in convergence compared to the linear updating scheme (3.14),(3.15) above. This statement was also verified in preliminary simulations by ourself, and we will therefore use the multiplicative version in the following simulations instead of the linear one.

   Design variable: ρ

2. A simplification of Benveniste's multiplicative algorithm, first proposed in [7], can be written as

$$\hat{\mu}_{t+1} = \hat{\mu}_t[I + \rho \Re\{\hat{\psi}_t^* \varphi_t \varepsilon_t\}] \tag{3.19}$$
$$\hat{\psi}_{t+1} = \alpha\hat{\psi}_t + \varphi_t\varepsilon_t \quad . \tag{3.20}$$

   Here, the parameter $\hat{\psi}$ is obtained by low-pass filtering of the signal $\varphi_t\varepsilon_t$. The filtering is performed by a first order low-pass filter with a pole in $a < 1$. The degree of low-pass filtering is here controlled by the value of the parameter α. This value is normally set close to 1.

   Design variables: ρ, α

   Remark: By using α instead of $[I - \hat{\mu}_{t+1}\varphi_t\varphi_t^*]$, c.f (3.18), the explicit dependence on μ_t and the time variability caused by the regressors φ_t in $\hat{\psi}_{t+1}$ are excluded. This may work well in situations where the step-size μ is assumed to change slowly, or in cases where we have enough information about the approximate value of the optimal step-size so that an appropriate value of α could be used. In situations where the step-size μ is large, and the chosen value of α is close to one, a deviation from the optimal steady state step-size μ will be introduced. This phenomenon will be investigated in the following chapter.

3.4 Simplified Wiener LMS with automatic tuning of the step-size

The first step towards a self tuning SWLMS algorithm is to combine the variable step-size schemes above together with the SWLMS equations from Chapter 2. This results in the following algorithms

VS-SWLMS-Benveniste (Multiplicative):

$$\begin{aligned}
\varepsilon_t &= y_t - \varphi_t^* \hat{h}_{t|t-1} & (3.21)\\
\hat{\mu}_{t+1} &= \hat{\mu}_t [I + \rho \Re\{\hat{\psi}_t^* \varphi_t \varepsilon_t\}] & (3.22)\\
\hat{\psi}_{t+1} &= [I - \hat{\mu}_{t+1} \varphi_t \varphi_t^*] \hat{\psi}_t + \varphi_t \varepsilon_t & (3.23)\\
\hat{h}_{t|t} &= \hat{h}_{t|t-1} + \hat{\mu}_{t+1} \mathbf{R}^{-1} \varphi_t \varepsilon_t & (3.24)\\
\hat{h}_{t+k|t} &= \boldsymbol{\mathcal{P}}_k(q^{-1}) \hat{h}_{t|t} & (3.25)
\end{aligned}$$

VS-SWLMS-Simplified Benveniste (Multiplicative):

$$\begin{aligned}
\varepsilon_t &= y_t - \varphi_t^* \hat{h}_{t|t-1} & (3.26)\\
\hat{\mu}_{t+1} &= \hat{\mu}_t [I + \rho \Re\{\hat{\psi}_t^* \varphi_t \varepsilon_t\}] & (3.27)\\
\hat{\psi}_{t+1} &= \alpha \hat{\psi}_t + \varphi_t \varepsilon_t & (3.28)\\
\hat{h}_{t|t} &= \hat{h}_{t|t-1} + \hat{\mu}_{t+1} \mathbf{R}^{-1} \varphi_t \varepsilon_t & (3.29)\\
\hat{h}_{t+k|t} &= \boldsymbol{\mathcal{P}}_k(q^{-1}) \hat{h}_{t|t} & (3.30)
\end{aligned}$$

Here, only the GB algorithms are presented. The other conceivable algorithms with LMS-based step-size adjustments are summarized in Appendix D.

The performance of these algorithms will be investigated in the next chapter. Furthermore, in Appendix F the flexibility of the automatically tuned SWLMS algorithm is illustrated by solving the Wiener filtering problem from Appendix A without any knowledge about the parameter drift to-noise ratio γ.

3.4.1 VS-SWLMS

With Section 3.3 in mind we will now present a new updating scheme based on the SWLMS algorithm. The algorithms above are based on the structure of the LMS algorithm. An algorithm derived from the SWLMS equations will look somewhat different. Of special interest is in what way the coefficient prediction-smoothing filter $\mathcal{P}_k(q^{-1})$ from (2.54) will propagate into the adjustment equation of the step-size and what effect this will have on the tracking performance. Compared to the VS-LMS-Benveniste algorithm described above, the major difference in the new algorithm due to the filter $\mathcal{P}_k(q^{-1})$ will show up in the calculation of the gradient ∇_μ.

The starting point is the SWLMS adaptation algorithm (2.52)-(2.55) for the AR2-hypermodel (2.61), (2.62). Minimizing the criterion J in (2.14) with respect to the step-size parameter μ yields the VS-SWLMS algorithm[2]:

$$\begin{aligned}
\varepsilon_t &= y_t - \varphi_t^* \hat{h}_{t|t-1} && (3.31)\\
\hat{\mu}_{t+1} &= \hat{\mu}_t + \rho \Re\{\hat{\psi}_t^* \varphi_t \varepsilon_t\} && (3.32)\\
p &= 1/(1 + d_2(1 - \hat{\mu}_{t+1})) && (3.33)\\
\hat{b}_0 &= -d_1 p && (3.34)\\
\hat{b}_1 &= -d_2 && (3.35)\\
\hat{a}_1 &= -d_2(1 - \hat{\mu}_{t+1})\hat{b}_0 && (3.36)\\
\hat{c} &= d_2 \hat{b}_0 p && (3.37)\\
Z_t &= \mathbf{R}^{-1} \varphi_t \varepsilon_t && (3.38)\\
\mathbf{X}_t &= \mathbf{R}^{-1} \varphi_t \varphi_t^* && (3.39)\\
P_1 &= (\hat{b}_0\ (\mathbf{I} - \hat{\mu}_{t+1}\mathbf{X}_t) - \hat{a}_1) && (3.40)\\
P_2 &= \hat{b}_1\ (\mathbf{I} - \hat{\mu}_{t+1}\mathbf{X}_{t-1}) && (3.41)\\
\hat{\psi}_{t+1} &= P_1\ \hat{\psi}_t + P_2\ \hat{\psi}_{t-1} + (\hat{c}\ \hat{\mu}_{t+1} + \hat{b}_0) Z_t \\
&+ \hat{b}_1\ Z_{t-1} && (3.42)\\
\hat{h}_{t|t} &= \hat{h}_{t|t-1} + \hat{\mu}_{t+1} Z_t && (3.43)\\
\hat{h}_{t+k|t} &= \mathcal{P}_k(q^{-1}) \hat{h}_{t|t} && (3.44)
\end{aligned}$$

See Appendix E for a complete derivation of equations (3.31) to (3.44).

Remark: In order for the algorithm to work properly, stability of (3.42) is required. Since P_1 and P_2 are time varying the stability analysis is far from trivial. The stability conditions are not easily obtained and the issue is subject to further studies.

Now, comparing the equations (3.15) and (3.42) we observe the additional dynamics introduced by the coefficient prediction filter $\mathcal{P}_k(q^{-1})$. This raises an interesting

[2]The derivation of this step-size scheme is based on the additive updating model to facilitate the simple comparison with the derivation of the VS-LMS-Benveniste algorithm.

question:

In what way will this affect the overall performance of the algorithm?

The following issues are interesting to study for different tracking scenarios:

- The convergence rate
- The steady state tracking performance
- Sensitivity to noise
- Sensitivity to abrupt changes in the parameter vector

If it turns out that the performance of the VS-SWLMS is superior to the SWLMS algorithm used in combination with the step-size scheme derived for the LMS algorithm, is it then worth the increased complexity? This question will be answered in the next chapter where simulations corresponding to different tracking scenarios will be conducted.

3.5 Comments about stability, convergence, and steady-state behavior

In the literature e.g. [1],[2],[25] much can be read about the stability of the LMS algorithm and the different versions of the self tuning, VS-LMS algorithms. The problem of obtaining exact expressions and bounds for the stability has turned out to be an extremely complicated task. Normally, when investigating the stability of the LMS algorithm the so called *independence assumption* is applied [1],[2]. This can be summarized as:

- The regressors φ_t, φ_{t+1}... φ_{t+k} are assumed independent
- At time t, the regressors φ_t are assumed to be independent with all previous samples of the signal y_t.
- At time t, the signal y_t is dependent only on the regressor φ_t and independent of all previous samples of the signal y_t.
- The regressor φ_t and the signal y_t are jointly Gaussian

These assumptions are, of course, unrealistic in virtually every practical situation, but they have shown useful, in situations where the rate of change of the time-varying parameters is slow, i.e. when the optimal step-size is very small. However, when dealing with fast varying parameters these assumptions are simply not appropriate. Signals may not be Gaussian, dependence among the signals may occur, the signal statistics may change over time and so forth. This, together with a large step-size severely complicates the stability and convergence analysis. Furthermore, in the case of VS algorithms we also have an additional adaptation loop which further complicates the analysis. A common way used in the literature, to ensure the stability of the VS algorithms is to limit the step-size parameter μ_t by lower and upper bounds, μ_{min} and μ_{max}, satisfying the stability criterion for the LMS algorithm [1],[2],[7],[18],[20],[29]. If consecutive regressor vectors φ_t are independent then it is appropriate to set these bounds to, 0 and $2/(3tr\mathbf{R})$ [29] respectively, i.e.

$$0 < \mu < \frac{2}{3tr\mathbf{R}}. \tag{3.45}$$

Normally μ_{min} is set to a small value close to zero. This way of ensuring the stability works well in most situations. However, in [29][3] the authors demonstrate that the above assumption in general is *false*, since the independence assumption does not normally hold, and that the stability issue for VS algorithms is something much more complicated than that of the classical LMS algorithm and therefore requires careful analysis. The problem is that the step-size is data dependent and therefore does not fit into the structure normally used for stability analysis. In [29] the authors also show that the stability regions for VS algorithms are smaller compared to that of the traditional LMS algorithms. While caution is thus recommended, our experience from simulations regarding the stability of the VS-LMS algorithms and the new VS-SWLMS algorithm is that the stability bounds suggested in (3.45) are valid in most cases.

When it comes to the *convergence* issue of the step-size μ_t of the VS algorithms, different studies have been performed, see e.g. [18],[20],[21],[29],[30],[31] in order to show that the adaptive step-size converges under certain given assumptions. However, as we see it, often these studies, [18],[21], neglect one very important aspect - *Is the step-size really converging to the optimal step-size predicted by the theory?* This is of course a fundamental issue when it comes to automatic adjustment of the step-size, and can not be neglected. It is common to prove that the VS algorithm as a whole converges in a mean square sense. This, however does not guarantee that the step-size converges to a proper value which results in a good tracking performance. In the literature, when it comes to comparative studies between different adaptive step-size schemes, the estimation MSE after convergence of the parameter vector h_t is often used to tune the different design parameters

[3]Here, exact expressions for the stability region of one-tap filters have been derived. In the multi-tap case, bounds on the stability have been obtained in [29].

in the competing algorithms [18],[20],[7],[26]. Then, the convergence of the estimation MSE (also called the learning curve) is used as a measure to compare the convergence rate of the algorithms. However, since the resulting estimation MSE is very much depending on the time variability of the parameters h_t as well as on the SNR, the process of finding appropriate values of the design parameters of the different algorithms[4] can be quite time consuming. This is of course not tractable since we aim at an almost self-tuning algorithm.

When the steady-state tracking performance is evaluated among different algorithms, it is often performed as follows. Starting with a certain parameter variation, e.g. RW or AR, and a disturbing noise, the steady-state performance is measured from the time the algorithms enter the tracking mode, i.e. when the initial transients are gone. Then the estimation MSE of the algorithms is plotted versus the number of recursions, and finally a winner is chosen. Normally, slow parameter variation is assumed, and the performance is evaluated at different SNR's for various input signals. However, environmental changes might require re-tuning of the algorithms in order to cope with the new situation. Then, again, repeated tests have to be performed in order to find suitable parameter values. This procedure is not acceptable if these algorithms are to be used in real applications where the environment might change. Then the whole idea with adaptive step-size algorithms is violated.

3.6 Complexity

Since the complexity of adaptive algorithms is very important when it comes to implementation in real applications we will here (in Table 3.1) summarize the number of calculations (real multiplications) needed to update the step-sizes μ_t for each one of the presented VS algorithms. In this calculation, multiplication between complex numbers is counted as four real multiplications, whereas multiplications or divisions between a real and a complex number are counted as two multiplications. The polynomials $D(q^{-1})$ in VS-SWLMS is assumed to have real-valued coefficients.

Note that in Table 3.1, only the calculations for updating the step-size is presented. In addition to these calculations the complexity for the constant gain algorithm, such as LMS or SWLMS, needs to be added in order to get the total number of multiplications. The complexity for the LMS algorithm and the SWLMS is presented in Table 3.2. Here we assume a complex linear regression with scalar out-

[4]This is especially true for the error based and the correlation based algorithms.

VS-algorithm	# of real mult.
VS-Benveniste	$26\ n_h$
VS-Benveniste-Simpl	$18\ n_h$
VS-SWLMS	$54\ n_h$
VSS	$7\ n_h$
MVSS	$7\ n_h + 8$
RMVSS	$7\ n_h + 8$
CDSS	$2\ n_h$

Table 3.1: The number of real multiplications required to update the VS algorithms. Here, n_h is the number of parameters in the vector h_t.

put, complex-valued regressors, n_h parameters, and a real-valued $D(q^{-1})$. These complexity calculations concerns one-step predictors and are later to be considered when the performance of the different algorithms is evaluated. For a complexity comparison with the Kalman estimator, we refer to [11].

Algorithm	# of real mult.
LMS	$10\ n_h$
SWLMS	$16\ n_h$

Table 3.2: The number of real multiplications required to update the LMS and SWLMS algorithms. Here, n_h is the number of parameters in the vector h_t.

3.7 Conclusions

In this chapter we introduced the concept of automatic tuning of the simplified WLMS algorithm. Different updating schemes were presented and a new adjustment scheme was derived. The stability issue of variable step-size algorithms were also discussed and it turned out that this is a much more complex problem than when dealing with the original LMS algorithm. Finally, the complexity for the different adjustment methods were compared.

Chapter 4

Simulation and performance evaluation

4.1 Introduction

In Chapter 3 an overview of different step-size methods was given. We will in this chapter study the performance of these step-size algorithms in various tracking scenarios in order to find the best candidate to be used together with the SWLMS algorithm. Recall that the LMS algorithm is obtained as a special case of the SWLMS algorithm by selecting the hypermodel as $\mathcal{H}(q^{-1}) = 1/(1 - q^{-1})$, i.e. a random walk model. Thus only SWLMS algorithms will be considered. Since no analytical results are presented for the algorithms, the following study will be based on simulations. The algorithms that are subjects for the investigation are, the *gradient based methods* (GB):

- Alg 1: VS-SWLMS-Benvenistes, (3.21)-(3.25)
- Alg 2: VS-SWLMS-Simplified Benvenistes, (3.26)-(3.30)
- Alg 3: VS-SWLMS, (3.31)-(3.44)

and, the *error based methods* (EB):

- Alg 4: VSS-SWLMS, (Appendix D)
- Alg 5: MVSS-SWLMS, (Appendix D)
- Alg 6: RMVSS-SWLMS, (Appendix D)

- Alg 7: CDSS-SWLMS, (Appendix D) .

Note that in Alg 4 to Alg 7 the SWLMS algorithm (2.52)-(2.54) is combined with the step-size $\mu = \hat{\mu}_t$ with $\hat{\mu}_t$ given by (3.2), (3.4)-(3.5), (3.6)-(3.7), and (3.1), respectively. To start with, we will investigate the differences between the EB methods and GB methods. Then, we will focus on the differences between the GB algorithms. Of particular interest regarding the GB algorithms is the influence of the parameters ρ and α, and whether or not the newly derived VS-SWLMS (Alg 3) will perform better than the original VS-SWLMS-Benveniste (Alg 1) algorithm and its simplification (Alg 2). We will start with simple adaptation problems and then gradually move on to more realistic scenarios. The algorithms above will be evaluated in the following scenarios:

- Estimation of static parameters.

- Tracking of sinusoidal parameters (slow and fast), at SNR, 5 dB and 15 dB, respectively.

- Tracking of Rayleigh fading parameters.

The comparative study will mainly focus on the behavior of the step-size adaptation of the different algorithms. Here, one of the important questions to answer is if the adaptive step-size will converge to the optimal value of μ predicted by the theory, i.e. the value of the step-size that solves the Wiener filtering problem in Figure 2.10. This, also implies that the algorithm will produce the smallest parameter tracking error.

Apart from the comparative study of the different algorithms, we will also investigate three other issues regarding tracking.

- The *first* issue deals with individual step-size control for each parameter in the vector h_t. The parameters, i.e. the elements of the vector h_t, are here assumed to vary at different rates. Here we illustrate the flexibility of the VS algorithm to individually adjust the step-size μ_t for each parameter h_t^i according to their rate of change of h_t^i.

- The *second* issue deals with the problem of wether to use the Newton direction in tracking problems or not. As mentioned before, see Chapter 2, it is well known that the Newton direction improves the convergence rate in static parameter estimation problems compared to using the gradient direction. Here we will investigate if this also is the case in tracking problems.

- At the end of the chapter, we demonstrate the performance gain obtained by using different hypermodels when tracking time varying parameters of different kinds.

In most of the simulations aimed to describe the behavior of the adaptive step-size algorithms, tracking of only one time-varying parameter will be considered. The observed behavior is however valid also for higher order models. In the comparisons between the algorithms we will focus either on the adaptation of the step-size or the squared parameter tracking error $\tilde{h}_t^2$. In the simulations below, if nothing else is said, the SWLMS algorithm is used in combination with the step-size algorithms (1-7) presented above together with an integrated random walk hypermodel (IRW)[1].

4.2 Comparison between the EB and GB step-size algorithms

Since the structure of the EB and GB algorithms is quite different we expect these algorithms to also show different tracking properties. We start this comparison by studying the convergence properties of the step-size $\hat{\mu}_t$ for Alg 1- Alg 7 when estimating an unknown static parameter.

4.2.1 Simulation 1: Estimation of a static parameter:

Since the parameter h_t is time-invariant we expect the step-size $\hat{\mu}_t$ to decrease until the lower limit of the step-size, μ_{min}, here set to 0.005, is reached. Figure 4.1 illustrates the convergence of the step-size parameter $\hat{\mu}_t$ for the different algorithms. The differences between the algorithm classes are obvious: All the EB algorithms outperform the GB algorithms. Here, the design parameters in the different algorithms are chosen to typical values normally used in order to produce good tracking performance[2]. The rate of convergence for the GB algorithms is much slower than that of the EB algorithms. The reason for this big difference is due to the fact that the instantaneous gradient $\hat{\nabla}_\mu(t)$, used in the step-size updating equation for the GB methods, is on average much smaller than the size of the squared instantaneous

[1]In order to completely study the behavior of the different step-size methods presented in the previous chapter a hypermodel other than the Random Walk (RW) model has to be used. The reason for this is that the SWLMS algorithm in the RW case reduces to the classical LMS algorithm. Some of the interesting features of the automatically tuned SWLMS algorithm would then vanish, and the study would not be of any interest. We therefore applied the IRW model.

[2]Regarding the parameter values for the EB methods we approximately use the values recommended by the different authors.

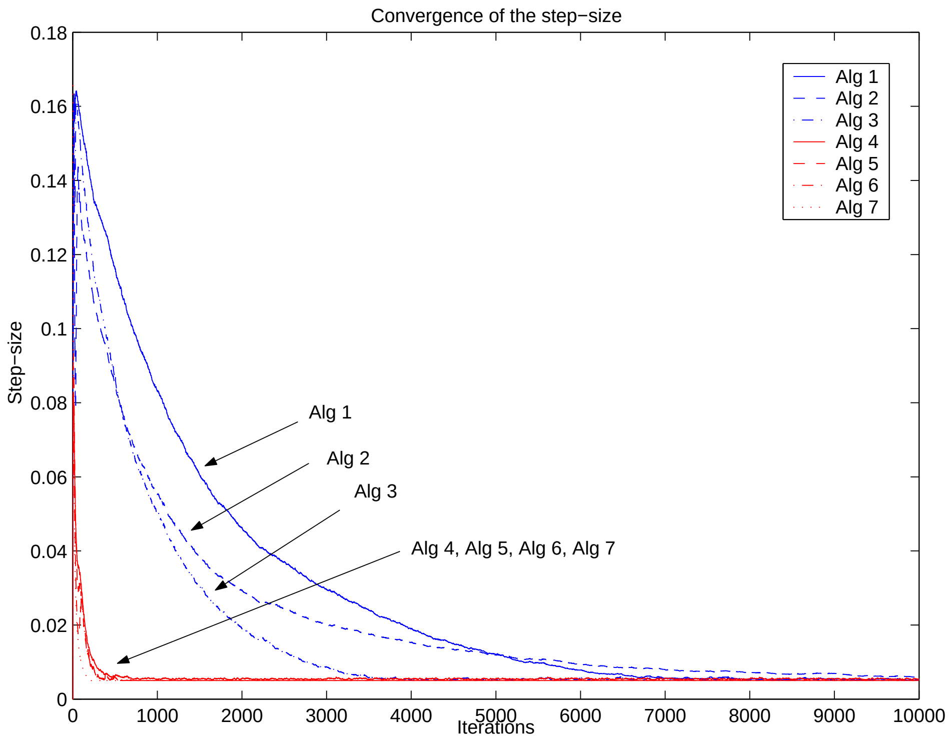

Figure 4.1: The convergence of the step-size $\hat{\mu}_t$ when estimating a static parameter. The convergence rates for the EB methods are much faster than for the GB methods.

error ε_t^2 used in the EB methods. It would of course be possible to speed up the convergence rate of these algorithms by increasing the value of ρ. However, this would increase the parameter tracking error in steady-state. Furthermore, among the GB methods, we observe that the convergence rate of Alg 3 is faster than that of Alg 1 and its simplified version Alg 2. This is interesting since the same value of the convergence-rate parameter ρ in the multiplicative step-size updating equation

$$\hat{\mu}_{t+1|t} = \hat{\mu}_t[I + \rho\Re\{\hat{\psi}_t^* \varphi_t \varepsilon_t\}] \tag{4.1}$$

was used. The estimated gradient $\hat{\psi}_t^*$ in the equation above is calculated according to the following equations, previously presented in Chapter 3.

$$\text{Alg 1)}\ \ \hat{\psi}_{t+1} = [I - \hat{\mu}_{t+1}\varphi_t\varphi_t^*]\hat{\psi}_t + \varphi_t\varepsilon_t \tag{4.2}$$

$$\text{Alg 2)}\ \ \hat{\psi}_{t+1} = \alpha\hat{\psi}_t + \varphi_t\varepsilon_t \tag{4.3}$$

$$\text{Alg 3)}\ \ \hat{\psi}_{t+1} = P_1\ \hat{\psi}_t + P_2\ \hat{\psi}_{t-1} + (\hat{c}\ \hat{\mu}_{t+1} + \hat{b}_0)Z_t + \hat{b}_1\ Z_{t-1} \tag{4.4}$$

Considering the above equations, the magnitude of $\hat{\psi}_t$ will differ among the algorithms. According to Figure 4.1 we can conclude that the extra dynamics resulting from the prediction filter in (4.4) clearly increased the magnitude of $\hat{\psi}_t$ compared to Alg 1, and therefore we obtained an improved convergence rate. We can also see, due to the faster convergence rate in Figure 4.1, that the simplification by using the parameter α[3] in (4.3) also changed the convergence properties compared to the original Alg 1.

As we suspected in the beginning of this chapter, the step-sizes of the different algorithms converged into the the region of μ_{min}. However, large differences in the convergence rate was observed. Therefore, if our goal is to find the step-size updating algorithm with the fastest convergence rate, then, we would probably choose one of the EB algorithms. In this case Alg 7 would be the obvious choice since it reached the optimum step-size faster than the other algorithms. However, Figure 4.1 does not say much about the step-size convergence properties of these algorithms in a tracking scenario. Therefore, we will now move on to tracking of time varying parameters.

4.2.2 Tracking a sinusoidal parameter

Tracking a sinusoidal parameter

$$h_t = e^{j\omega t} \tag{4.5}$$

will now be considered. In these examples we will study the behavior of the algorithms for different values of the frequency ω as well as for different signal-to-noise ratios. Tracking of sinusoids will be used in many of the following simulations in order to compare the different algorithms. Other more realistic parameter variations will be used at the end of this chapter.

Simulation 2: Slow parameter variation, SNR = 15dB

Figure 4.2 illustrates tracking of a low frequency sinusoid with $\omega = 0.00039$, at a medium SNR level (15 dB). The frequency is normalized to the sampling rate[4]. In this situation we also expect the adaptive step-size $\hat{\mu}_t$ to converge to a rather small value, since the time variability of the parameter h_t is rather slow. Here we can clearly see that all of the algorithms converges into the region of the optimal step-size 0.007. The optimal step-size was calculated iteratively by evaluating

[3]In this simulation $\alpha = 0.95$ was used.

[4]Sampling rate 270000 Hz will be used in all of the following simulations.

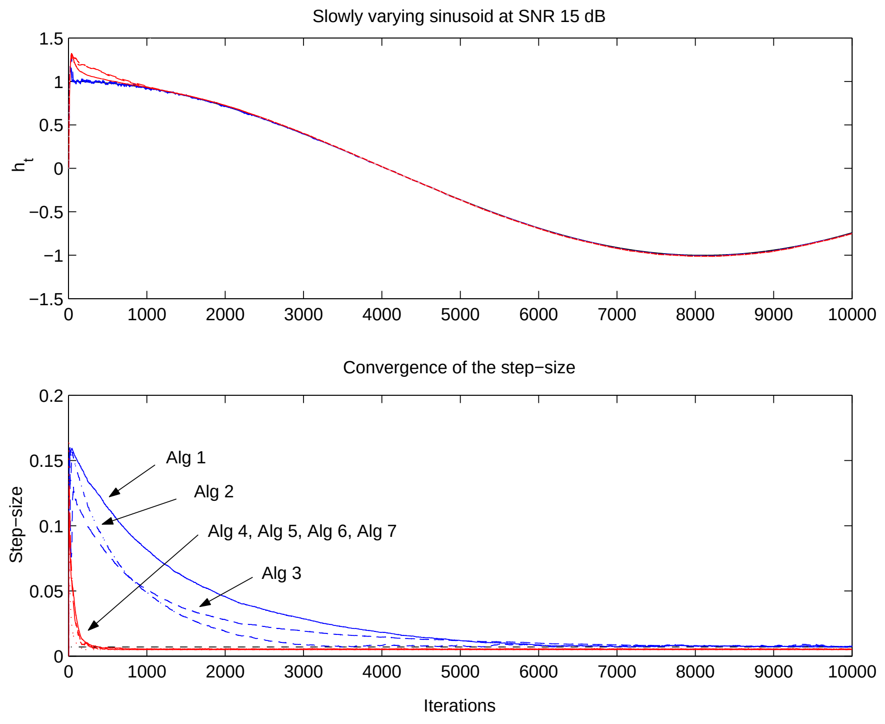

Figure 4.2: Tracking of a slowly varying sinusoid with frequency $\omega = 0.00039$. The upper diagram illustrates the optimal parameter variation of h_t and the estimates $\hat{h}_t$ for the different algorithms as a function of the number of iterations. Here, it can be seen that the GB algorithms achieves faster convergence towards the true parameters although the convergence of μ_t for the EB algorithms is faster. The lower diagram illustrates the adaptive step-size plotted versus the number of iterations.

the parameter tracking MSE for different step-sizes, then the value of μ that produced the lowest MSE was chosen as the optimal step-size. The differences, in terms of the resulting step-size between the algorithms are shown to be very small. However, as noticed in the previous example, the step-size of the GB algorithms suffer from much slower convergence than that of the EB algorithms. Figure 4.3 illustrates the parameter tracking MSE of the different algorithms. The resulting error is about the same due to the almost identical final step-size for all of the algorithms. The differences between the GB algorithms are shown to be very small. A closer study of these algorithms will be performed in a subsequent section.

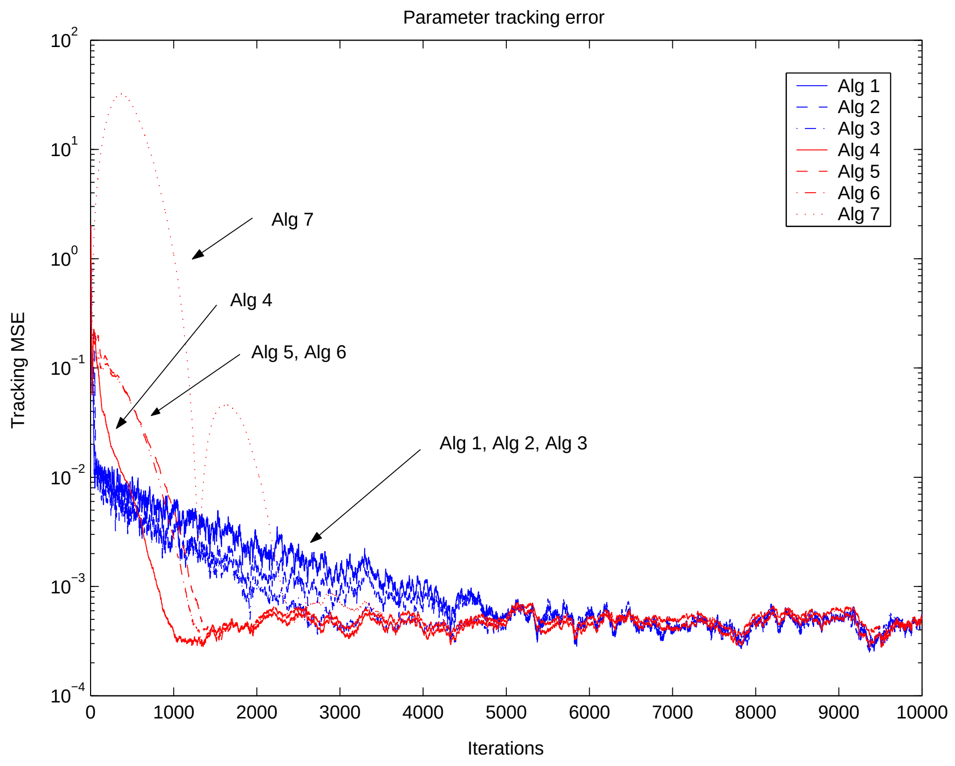

Figure 4.3: The parameter tracking error MSE for the different algorithms. Since the resulting step-size in this case equals the lower bound of the allowed step-size we can see that the MSE at the end of the adaptation is about the same for all algorithms.

Simulation 3: Fast parameter variation, SNR = 15dB

In order to observe the behavior of the algorithms in a different situation where the environment has changed, let the values of the design parameters for the algorithms remain exactly the same as for the slower sinusoid and then increase the frequency ω. This will, according to the theory [1],[2],[4], also increase the optimal step-size μ since the algorithms must be more alert to changes in the parameter vector h_t. Then, what will be interesting to see, is if the algorithms will converge to the step-size predicted by the theory, or if there will be a significant difference between them. The results are shown in Figure 4.4. Here, we observe that there exists a fundamental difference between the algorithms. The GB algorithms cope with these faster parameter variations in a much better way than the EB algorithms.

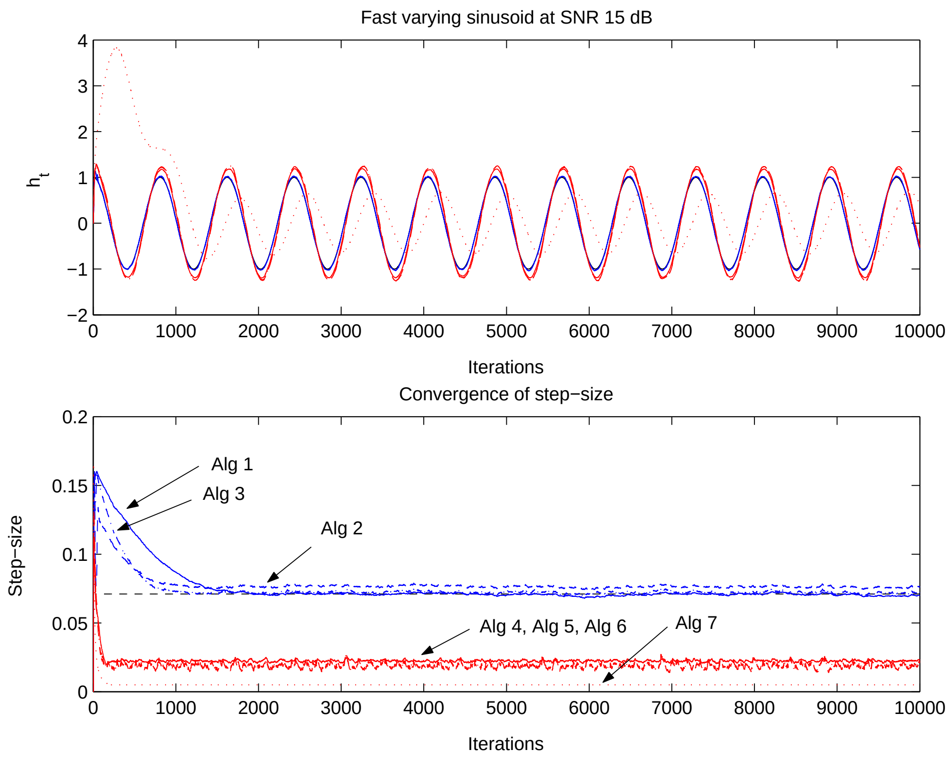

Figure 4.4: Tracking of a fast varying sinusoid, $\omega = 0.0078$, SNR 15 dB. The upper plot illustrates the optimal parameter variation of h_t and the estimates $\hat{h}_t$ for the different algorithms. In the lower plot the adaptive step-size is plotted versus the number of iterations. Here we can clearly see that the gradient based methods converge approximately into the optimal step-size 0.071. A small deviation from the optimal step-size can be noticed for the simplified version of Benvenistes method. This has to do with the value of the parameter α. The reason for this will be explained in later sections

Figure 4.5 shows the parameter tracking MSE of the algorithms when tracking the faster sinusoid. In this case we notice a remarkable difference between the algorithms as compared to the previous example. This effect is explained by the lower plot in Figure 4.4 where the adaptation of the step-sizes for the different algorithms is illustrated. In this example the optimal step-size was calculated to 0.071. As we see in Figure 4.4, the GB methods Alg 1 - Alg 3 manage to converge towards the optimal step-size predicted by the theory. This is however, not true for the EB algorithms Alg 4 - Alg 7. Their step-sizes continue to decrease to a level that is much smaller than the GB algorithms. The final value of these step-sizes, which

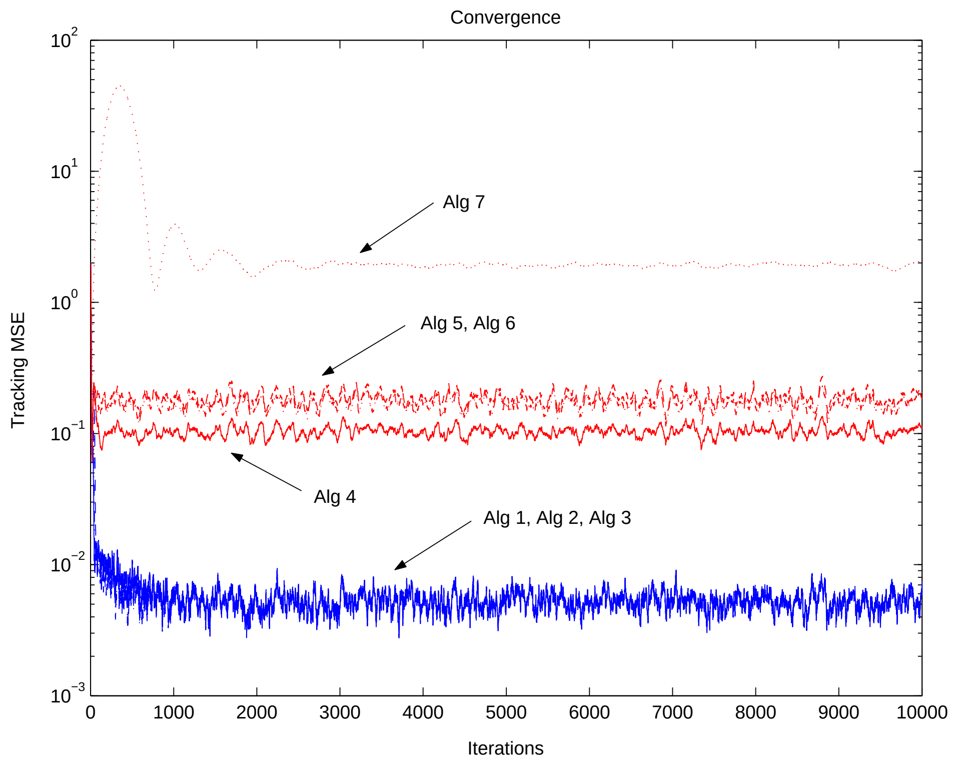

Figure 4.5: The parameter tracking MSE of the algorithms when tracking a fast varying sinusoid with $\omega = 0.0078$ and SNR 15 dB. Due to the large differences of the step-sizes between the EB and GB algorithms their resulting errors will differ significantly. The MSE of the GB methods are much smaller than the MSE of the EB methods since their step-sizes converge into the region of the optimal step-size predicted by the theory. The individual differences between these algorithm are however not noticeable in this case.

to a large extent is influenced by the choices of the parameters γ and α, see (3.2), (3.4), and (3.6), may not correspond to the step-size predicted by the theory.

These differences between the algorithms are not unexpected. By analyzing the design of the different step-size schemes, it becomes quite clear why they behave as they do. In the case of the GB methods, the design is based on the minimization of the criterium J_t in Chapter 2. This, makes the adaptive adjustment scheme search for the particular value of the step-size $\hat{\mu}_t$ that minimizes this criterium. This important feature is not present in the EB methods where the adjustment of the step-size is performed without involving the minimization of a suitable criteron.

Simulation 4: Slow parameter variation, SNR = 5dB

One thing that might change the behavior of the algorithms drastically is the measurement noise. In real applications it is very important that the adaptive methods do not deteriorate to much if the noise increases. Therefore, we will now turn our attention towards the signal-to-noise ratio. Recalling Simulation 1, we will now perform exactly the same experiment but increase the noise level. These results are shown in Figure 4.6. It is now possible to see the effect of the noise in the adapta-

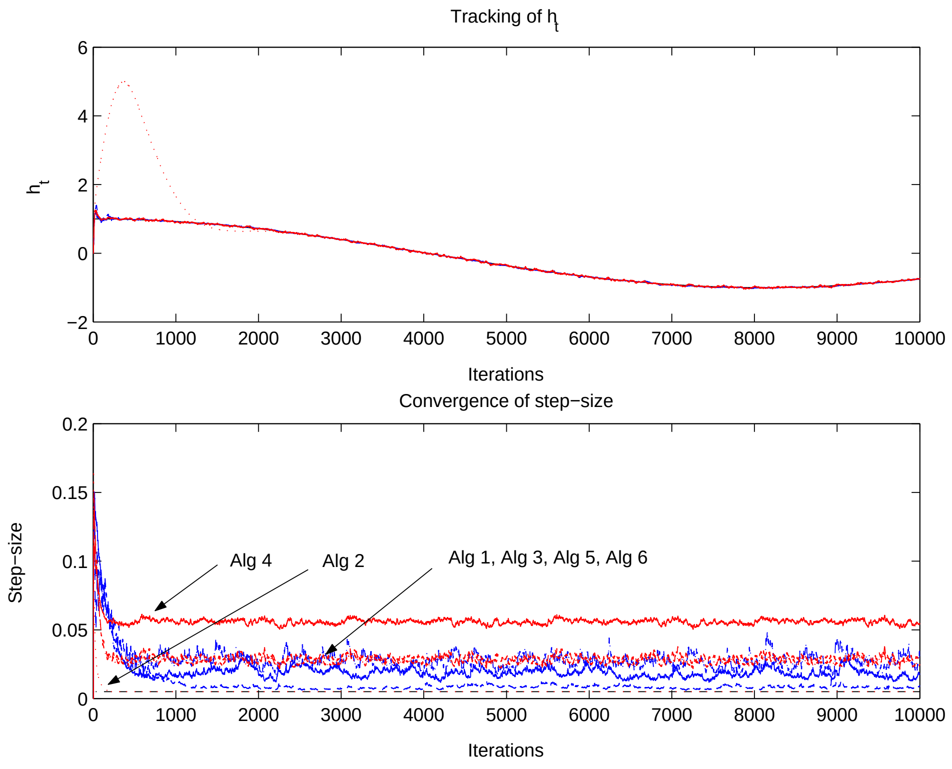

Figure 4.6: Tracking of a slowly varying sinusoid with $\omega = 0.00039$ and SNR 5 dB. In this case the theoretical step-size is quite small due to the large influence of noise. We therefore expect that the different algorithms to converge into this small step-size. However, this is only true for Alg 2, the other GB algorithms suffers from a bias of varying size. The reason for this will be discussed in later sections. Regarding the EB methods, we see that they do not manage to find the optimal step-size either. Also noticeable is that the convergence rate of the GB methods significantly increased due to the higher noise level.

tion of the step-size. Since the noise increases the size of the gradient $\nabla_h = \varphi_t \varepsilon_t$, the convergence rate of the step-size will also increase due to the close connection to the size of the instantaneous gradient $\nabla_\mu(t)$. This is of course attractive, but, unfortunately, together with this increasing convergence rate of the step-size $\hat{\mu}_t$, the fluctuations of the adaptive step-size becomes severe. In the case of the GB methods, these fluctuations can be controlled by the parameter ρ. A small value of this parameter produces a small adjustment of the step-size in each iteration which keeps the variance of $\hat{\mu}_t$ at a low level. However, a small value of the parameter ρ also severely decreases the convergence rate of the step-size, therefore a tradeoff between the fluctuations and the convergence rate must be considered.

What can also be noted in Figure 4.6 is that none of the GB algorithms seem to converge exactly to the optimal step-size, i.e 0.005. A clearly noticeable bias is present, especially for Alg 3 and Alg 1. The reason for this bias is not completely known at present and requires further studies. However, when it comes to Alg 2 it almost manages to find the theoretical step-size. This is explained by the time-invariant low-pass filtering of $\hat{\psi}_t$ used in Alg 2[5]. Regarding the EB methods we notice that their resulting step-sizes increased compared to Simulation 2, and also that they failed to approach the optimal step-size 0.005.

Figure 4.7 shows the corresponding parameter tracking MSE for the different algorithms. Since the optimal step-size in this simulation equals the lower bound $\mu_{min} = 0.005$, the smallest parameter tracking MSE is produced by Alg 7, which adjusts μ_t to be closest to this bound, among the considered algorithms.

Simulation 5: Fast parameter variation, SNR = 5 dB

Here, we will keep the noise level from the previous experiment but increase the frequency of the sinusoid. Now, by comparing Figure 4.8 and Figure 4.4 we notice that the optimal step-size has decreased from 0.071 to 0.046 due to the influence of the noise. However, looking at the step-sizes of the EB methods we observe an *increase* compared to Figure 4.4. This behavior is not desirable. The reason for this is that we in a noisy environment, would like an adaptive step-size algorithm to be more cautious, i.e. to use a smaller step-size, than in a high SNR scenario. On the other hand, looking at the step-sizes for the GB methods we observe a *decrease* of $\hat{\mu}_t$, which is in correspondence with the theory [1],[2]. The variance of the adaptive step-size is also shown to increase due to the higher noise level. Furthermore, we also notice that the bias in the step-size for the GB methods present in Figure 4.6

[5]In this example the value of the parameter α was chosen to 0.97. This assumes an optimal step-size around 0.03 and constant modulus regressors with amplitude one.

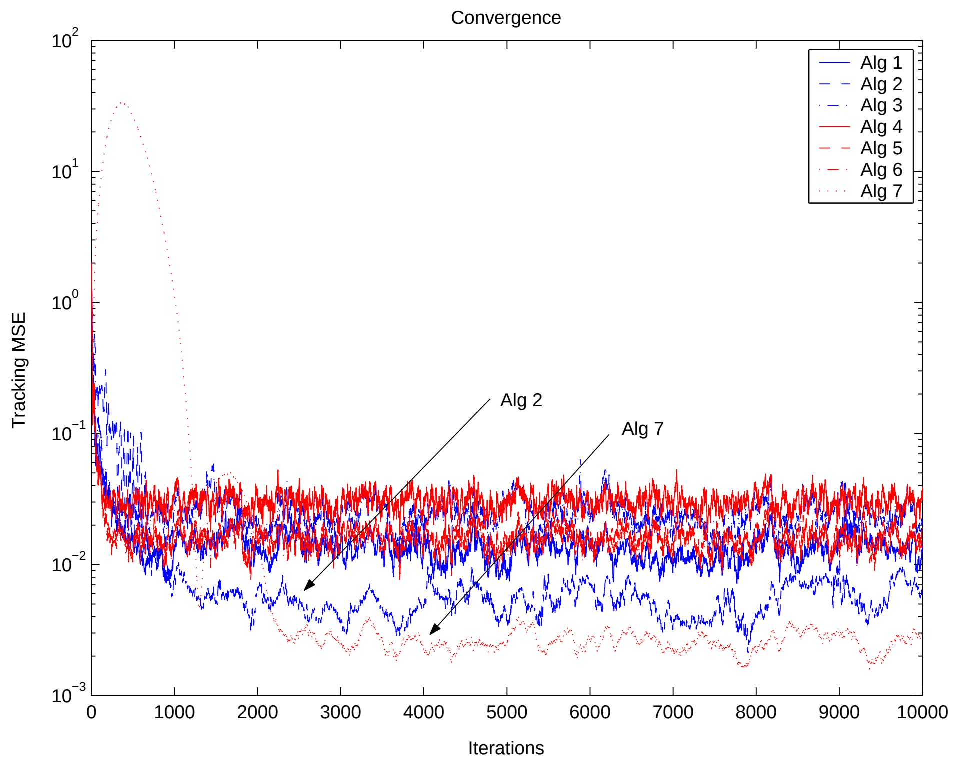

Figure 4.7: The parameter tracking MSE for algorithms Alg 1 - Alg 7 when tracking the slow sinusoid in Figure 4.6 with an SNR of 5 dB.

is still present. This indicates that these methods overestimates the step-size as the noise increases. This will be further discussed in later sections.

Figure 4.9 illustrates the parameter tracking MSE of Simulation 5. Due to the similar step-sizes shown in Figure 4.8 the resulting tracking MSE of the different algorithms are about the same, except for Alg 7 which suffers from larger tracking MSE due to a severe lag error (2.22).

We have so far seen that differences between the GB and EB methods *are* present, however, at this moment we do not exactly know when the differences becomes significant. This will be investigated next.

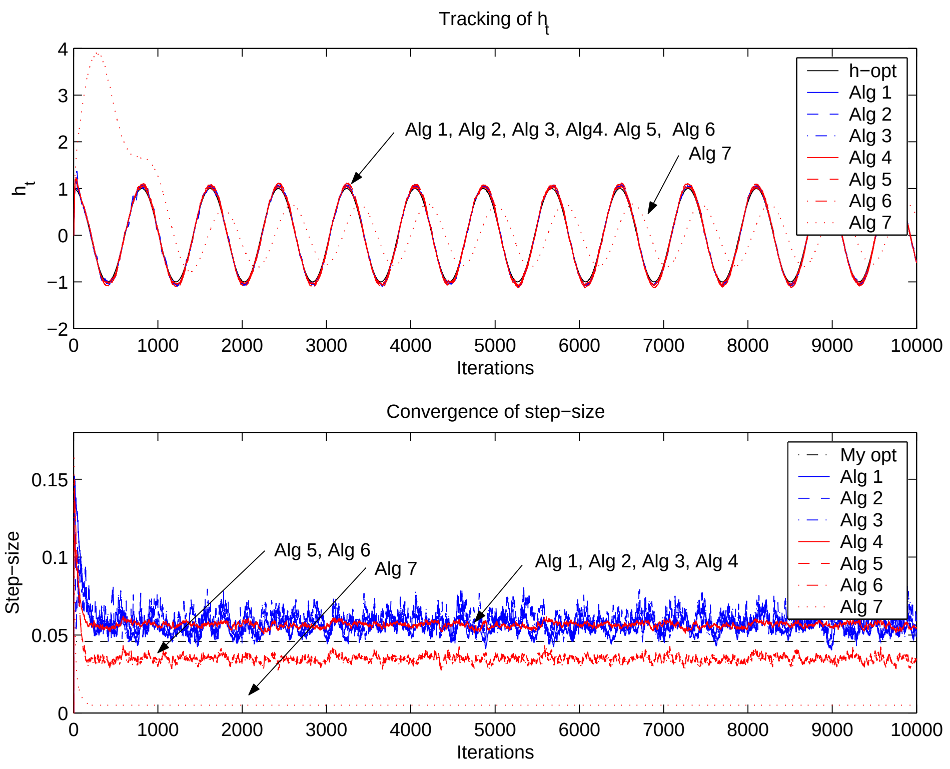

Figure 4.8: . Upper diagram: Tracking of a fast varying sinusoid with frequency $\omega = 0.0078$ and SNR 5 dB. Lower diagram: The convergence of the step-size μ_t for the different algorithms.

Simulation 6: Fast parameter variation, SNR = 25 dB

Since the optimal step-size is depending on the noise level and the rate of the time varying parameter we will now illustrate the behavior of the algorithms under fast variations and low noise conditions. In this example the frequency of the sinusoid is the same as in Simulation 5, i.e. $\omega = 0.0078$. However, the SNR is chosen to 25 dB and the optimal step-size is calculated to 0.1080. Figure 4.10 illustrates the sinusoidal parameter and the convergence of the step-size. The GB methods is seen to converge into the region of the optimal step-size 0.1080. However, the bias of Alg 2 is here larger than in Figure 4.4. The reason for this will be discussed later in Section 4.4. Also seen is that the step-sizes of the EB methods converge into totally erroneous values not representable for tracking fast of varying parameters. From this example we conclude that fast variations together with low noise cannot

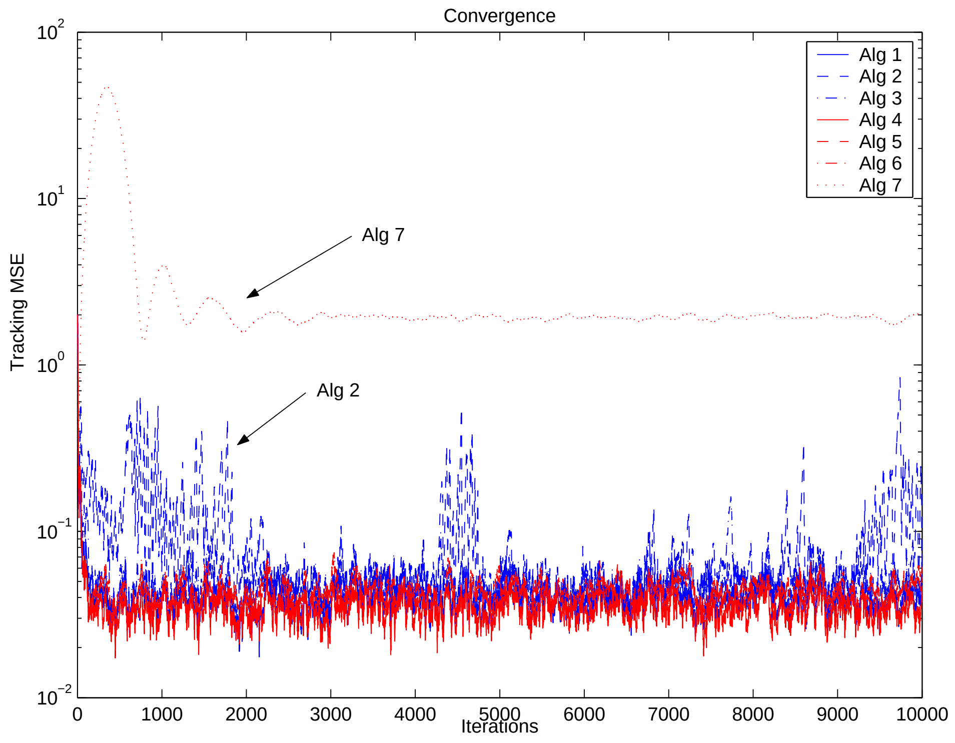

Figure 4.9: The parameter tracking MSE for the different algorithms in Figure 4.8 when tracking a fast sinusoid at SNR 5 dB.

be handled by the EB methods. The number of iterations is, in this simulation, compared to the previous, increased from 10000 to 20000 due to the slow convergence rate of the GB methods at this noise level.

This scenario clearly illustrates the difference between the EB and GB algorithms. By considering the lower plot of Figure 4.10 we conclude the following:

- In situations where μ_{opt} is large due to rapidly varying parameters and/or a high SNR, the EB algorithms seems incapable of finding the optimum value of the step-size. Tracking fast varying parameters/sinusoids using these algorithms in environments where the SNR is high is therefore not recommended.

- The GB methods Alg 1 and Alg 3 manage to converge to the step-size predicted by the theory[6]. With the exeption of generating a bias at low SNR,

[6]By theory, we mean the specific value of the step-size that produces the lowest tracking MSE

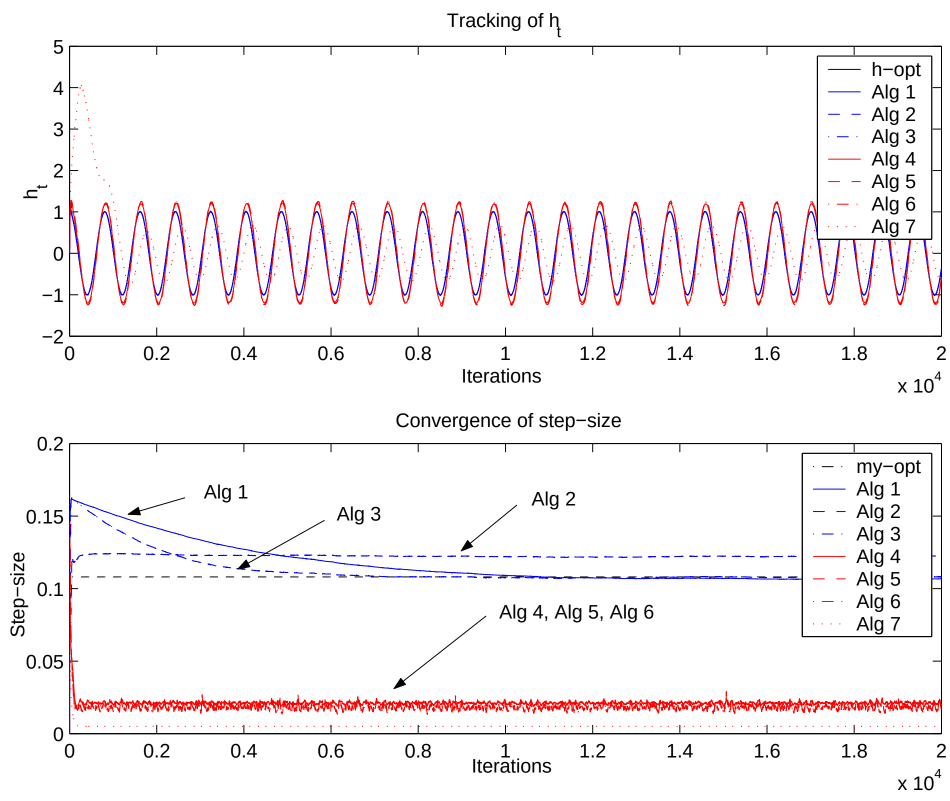

Figure 4.10: Upper diagram: Tracking of a fast varying sinusoid with frequency $\omega = 0.0078$ and SNR 25 dB. Lower diagram: The convergence of the step-size μ_t for the different algorithms.

these methods cope with fast varying parameters in a satisfactory way.

In order to further study the behavior of the different algorithms in the presence of noise we will conclude this section with a final example, where tracking of a sinusoid with high frequency is considered. Here, the simulation interval is divided into four segments in which the noise level is increased segment by segment. In this simulation, see Figure 4.12, the effect of the noise becomes obvious. The GB methods successfully decrease their step-size according to the increasing noise level. This is a result of a decreasing correlation among the components in the instantaneous gradient $\hat{\nabla}_\mu(t)$ used in the step-size updating equations for the GB algorithms. The EB methods, on the other hand, erroneously *increase* their step-

under the conditions given in the simulation, i.e. the rate of the time-varying parameter and the SNR.

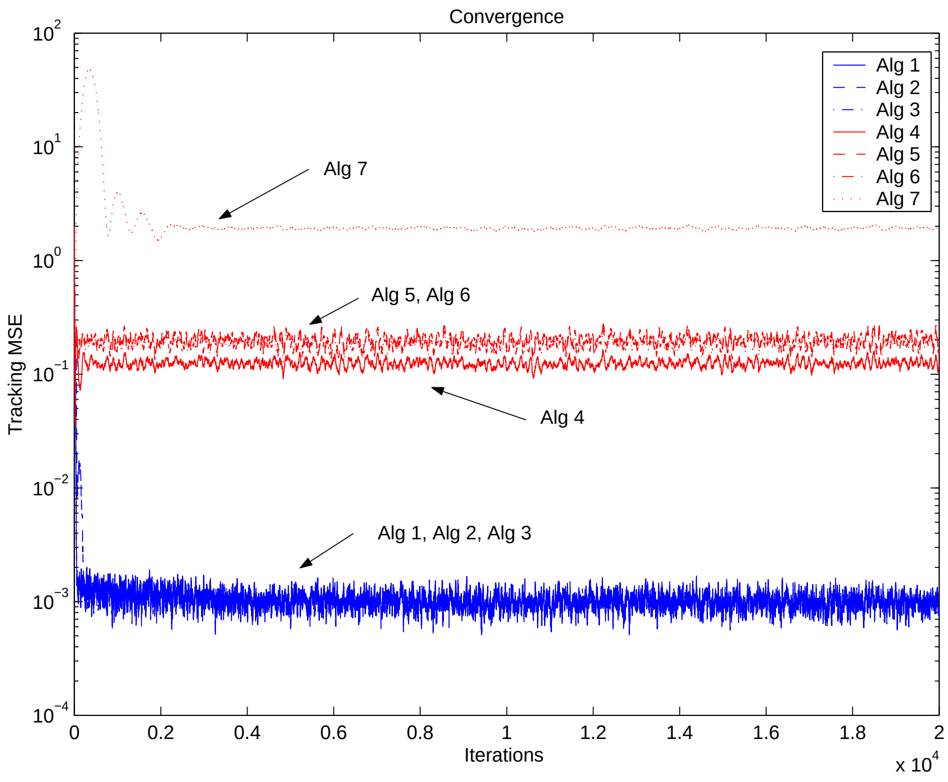

Figure 4.11: The tracking MSE for the EB and GB algorithms when tracking a sinusoid with frequency $\omega = 0.0078$ *at SNR 25 dB. The GB methods are here shown to outperform the EB methods since the tracking MSE is at least 100 times smaller.*

sizes when the noise becomes more dominant. The optimal step-size is plotted according to the noise level in the first segment. The noise level in the different segments is 1.5, 2 and 2.5 times the level in the first segment, which is 15 dB. What is interesting in Figure 4.13 is the development of the parameter tracking MSE for the EB methods. *Due to the fact that the noise level actually increases the parameter tracking error is shown to decrease*. This is explained by the fact that their step-sizes by coincidence increase towards the optimal step-size, see Figure 4.12.

So far we have investigated the convergence properties of the step-size for the different algorithms when tracking a sinusoid and a static parameter. In order to illustrate the behavior of the different algorithms when the environment is chang-

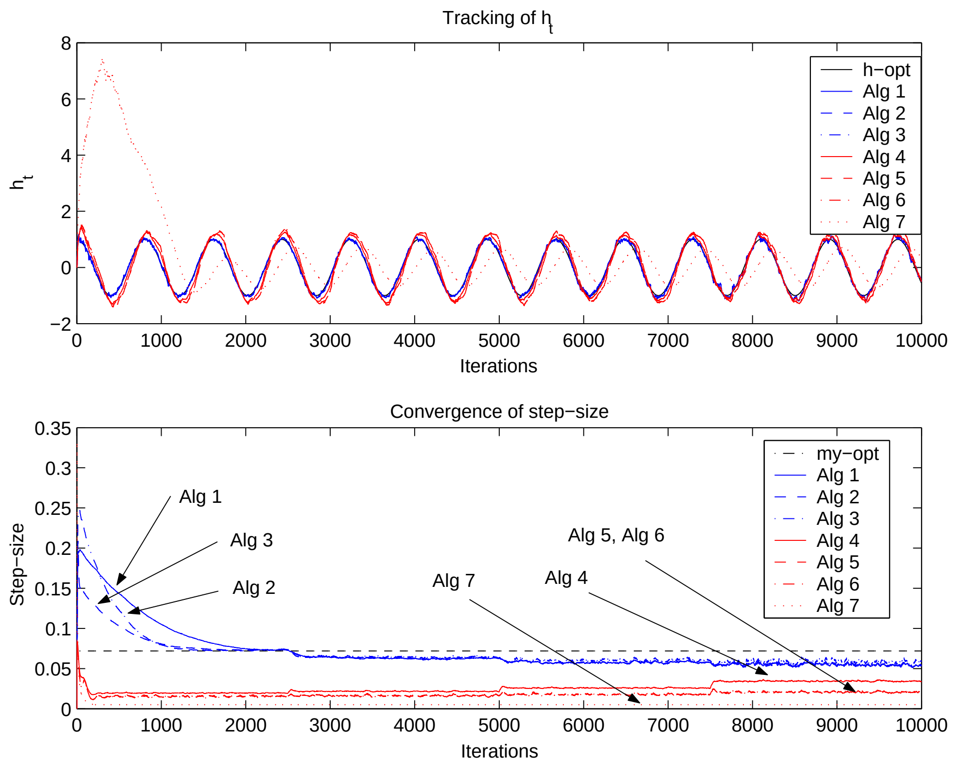

Figure 4.12: Tracking of a fast varying sinusoid in increasing noise. The noise level in the different segments is 1.5, 2 and 2.5 times the level in the first segment, which is 15 dB. In correspondence with the theory, i.e. when the measurement noise increases, then the step-size of the GB algorithms are seen to gradually decrease. The EB algorithms on the other hand, increases their step-size when the noise increases since they in this case, erroneously 'believe', that the tracking of the parameter h_t is bad due to the fact that the instantaneous error ε_t becomes larger.

ing, the design parameters of the algorithms have been the same in all simulations. It would of course have been possible to improve the performance by changing the design parameters for each simulation, however, this totally contradicts the automatic tuning concept we are about to investigate. Our intention by letting the design parameters remain the same during the different simulations is to find out how sensitive the different algorithms are based on a specific parameter setting. Based on the above simulations, the performance of the GB methods seems very promising. They have all shown a higher level of robustness to changes in the environment than the EB algorithms. Due to this superiority we will now leave the

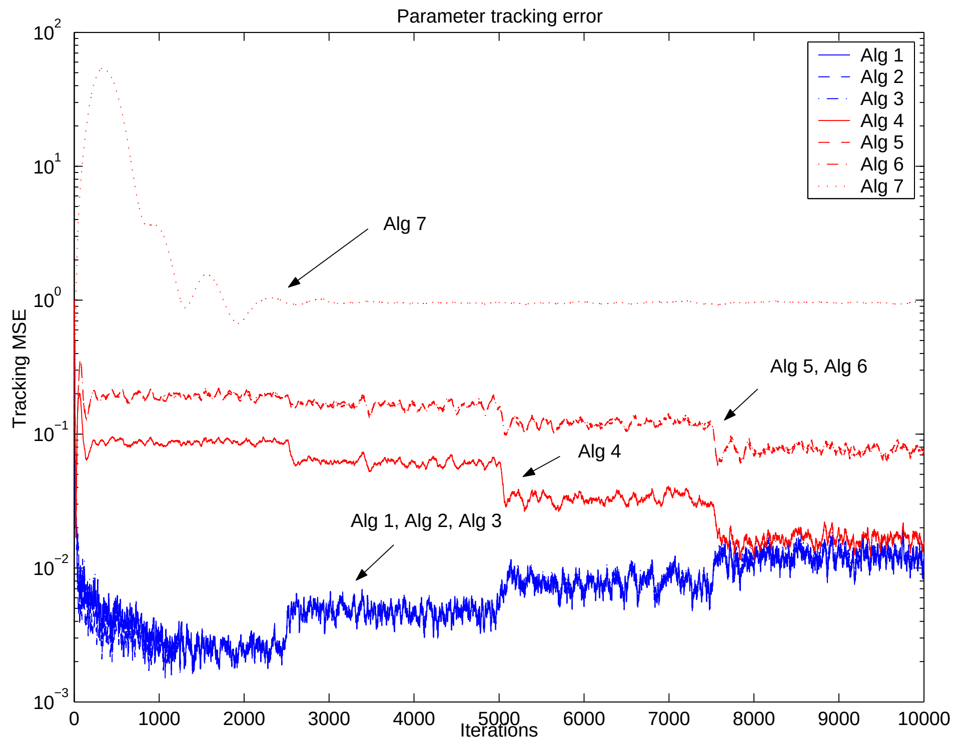

Figure 4.13: The corresponding tracking MSE when tracking a fast varying sinusoid in noise with stepwise increasing variance. The interesting part in this figure is that the MSE for the EB methods actually is shown to decrease although the noise is gradually increasing. We also note that Alg 5 and Alg 6 method is more robust to noise than Alg 4.

EB methods and focus only on the GB algorithms.

4.3 GB algorithms and the parameter ρ

To start with, we recall the equations (4.1)-(4.4). These equations represent the core of the step-size adaptation for the GB methods. Common to these three methods is the parameter ρ in (4.1), which controls the convergence rate of the step-size as well as the resulting tracking MSE in steady-state. Our aim in this thesis is to find a robust automatic step-size algorithm that does not need to be re-calibrated for each new tracking scenario. Therefore, in order to work well in real applications, the resulting automatically tuned SWLMS algorithm cannot be too sensitive to the choice of the parameter ρ. In the following examples we will study the behavior of the GB algorithms for different values of the parameter ρ in different tracking scenarios[7]. To start with, we will focus only on one of the GB algorithms in order to illustrate the main behavior. Figure 4.14 shows the convergence of the step-size for Alg 1 for four different values of the parameter ρ when tracking a sinusoidal parameter at the frequency $\omega = 0.0078$ at two different signal-to-noise ratios, 5 and 15 dB, respectively. As mentioned in a previous section, the convergence rate

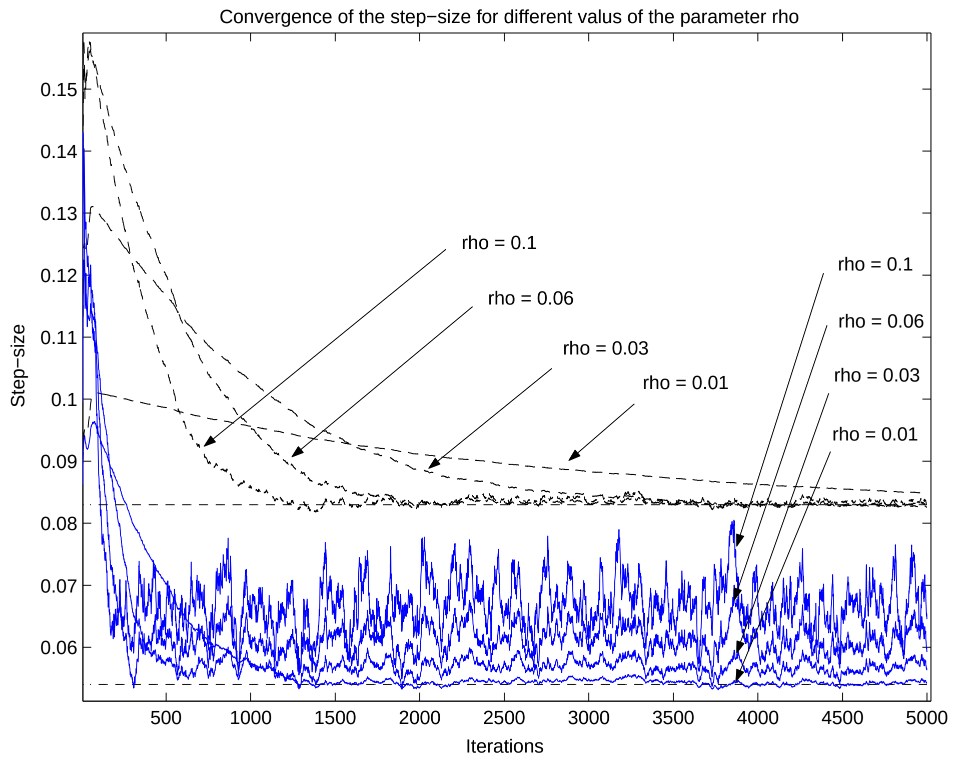

Figure 4.14: The convergence of the step-size $\hat{\mu}_t$ for Alg 1 for different values of the parameter ρ when tracking a sinusoidal parameter at the frequency $\omega = 0.0078$. In the case of SNR 15 dB (dashed) the different algorithms manage to converge to the step-size predicted by the theory. However, in the case of SNR 5 dB (solid) a bias becomes noticeable if the value of ρ increases.

of the step-size becomes faster if the value of the parameter ρ is large or if the SNR-

[7]In order to capture the initial behavior of the algorithm we here study the tracking scenario during the first 5000 iterations

level is low. The reason why the convergence rate increases with an increased noise level is that the driving force behind the adaptation, i.e. the norm of the instantaneous gradient $\hat{\nabla}_\mu(t)$, becomes larger. This phenomenon also becomes visible if we *decrease* the frequency ω. Then, it will be easier for the algorithms to follow the variations of the parameter h_t, thus, the parameter error $\tilde{h}_t$ at each time-step will be smaller than for a fast varying sinusoid. The driving force ∇_μ behind the adaptation therefore becomes smaller. This results in a slower convergence rate as well as smaller fluctuations of the step-size after convergence. These results are depicted in Figure 4.15. Here we can clearly see that the convergence rates for the step-sizes are much slower compared to those in Figure 4.14, where the frequency of the sinusoid was higher. What also can be noted in Figure 4.14 is that in the case of higher SNR, the different algorithms manage to converge to the step-size predicted by the theory. This, however, is not true in the case of lower SNR where the bias becomes noticeable. Furthermore, in the low SNR case, the bias seem to increase as the size of ρ increases.

So far, we have only focused on one of the three GB algorithms in order to illustrate the typical behavior for different values of the convergence rate parameter. We will now proceed with two final examples where we compare the three GB algorithms: Alg 1, Alg 2 and Alg 3. In Figure 4.16 the convergence rates for the GB algorithms are illustrated for four values of the parameter ρ when tracking a sinusoidal parameter at SNR 5 dB at two different frequencies. We can note that an increased value of the parameter ρ increase the convergence rate as well as the variance of the step-size $\hat{\mu}_t$. We also notice that Alg 1 (upper diagram) is most tolerant among the three algorithms against changes of the parameter ρ. This is observed by inspecting the variance of the step-size for the different algorithms in Figure 4.16. When it comes to Alg 2 (middle), we notice that the variance of the step-size is severely increased when tracking the faster parameter (blue curve). However, we also observe that the bias present among the red curves for Alg 1 and Alg 3 is vanished since it converges towards the correct step-size. This is a result of the time-invariant low-pass filtering of the gradient $\varphi_t\varepsilon_t$ performed by the parameter α in Alg 2. However, in the case of the rapidly varying sinusoid (blue curve), where the step-size is larger we notice that the low-pass filtering of $\varphi_t\varepsilon_t$ does not decrease the variance of $\hat{\mu}_t$. This is an effect of the design parameter α. The influence of the parameter α will be discussed in the next section. For the Alg 3 (lower) we observe both an increased variance of the step-size and an increased bias compared to that of Alg 1. Common to Alg 1 and Alg 3 in this simulation is that the variance of $\hat{\mu}_t$ increases if the frequency of the sinusoid increases and vice versa. This is observed by comparing variance of the red and blue curves. However, it is also noticeable that the bias seems to *decrease* with higher frequency.

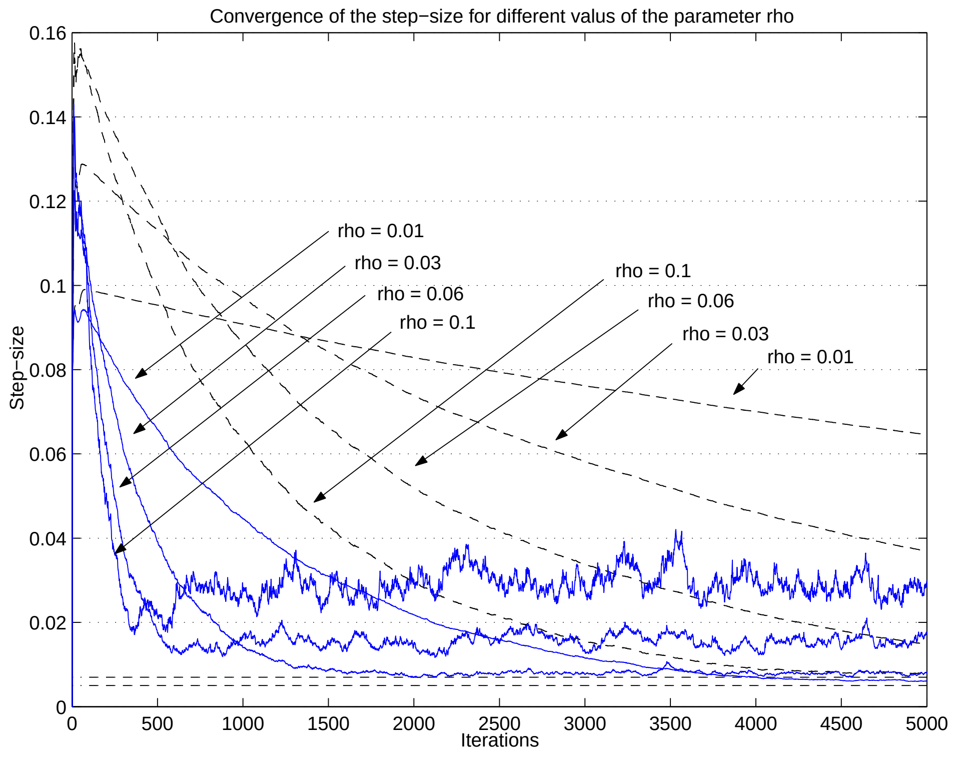

Figure 4.15: The convergence of the step-size $\hat{\mu}_t$ for Alg 1 for different values of the parameter ρ when tracking a sinusoidal parameter at the frequency $\omega = 0.00039$ SNR = 15 dB dashed, 5 dB solid. The convergence of the step-size is here slower than in Figure 4.14 due the lower frequency of the sinusoid. The bias is also in this case clearly visible at the lower SNR.

This phenomenon is explained by Figure 4.17 where the parameter variation and the instantaneous gradient $\hat{\nabla}_\mu(t)$ for Alg 1 when tracking a sinusoidal parameter at two different frequencies, $\omega = 0.00039$ and $\omega = 0.0078$ at SNR 5 dB is illustrated. Here we observe that the variance of the instantaneous gradient $\hat{\nabla}_\mu(t)$ for the fast sinusoid is much smaller than for the slower sinusoid. The bias is believed to originate from this difference. The reason for this difference in the variance of $\hat{\nabla}_\mu(t)$ is explained as follows. We know from Chapter 3 that the instantaneous gradient with respect to the step-size μ_t is defined as

$$\nabla_\mu = -\Re[\psi_t^* \varphi_t \varepsilon_t] \quad . \tag{4.6}$$

The size of these three components will depend on the variation of the parameter h_t and on the SNR level. Small variations of the parameter h_t, e.g. slowly varying

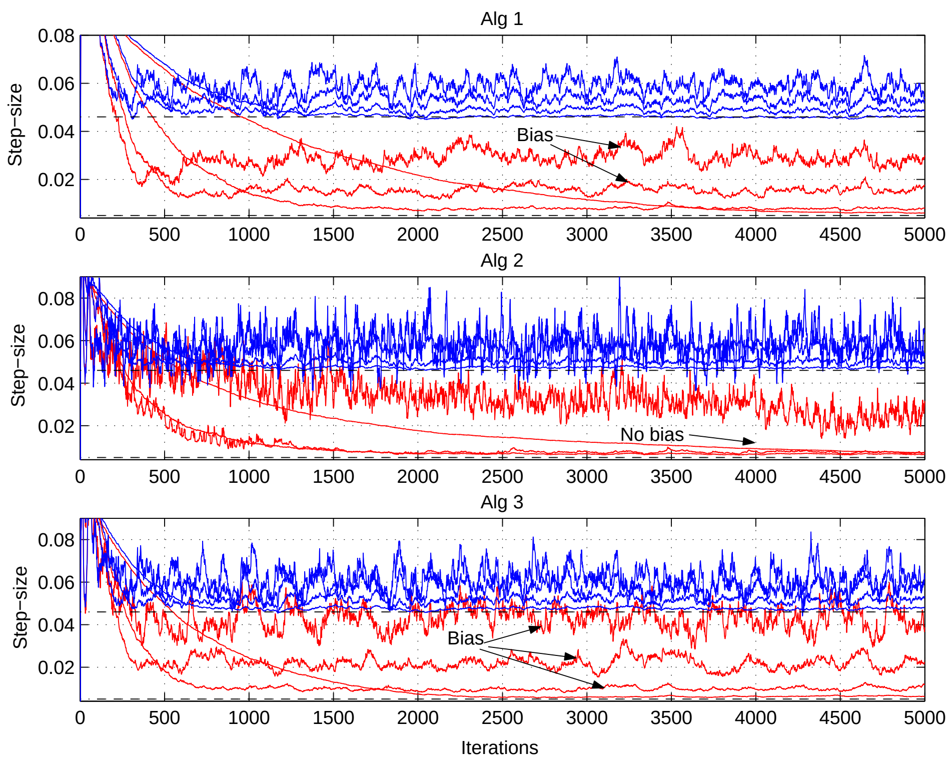

Figure 4.16: Convergence of the step-size $\hat{\mu}_t$ for the GB algorithms for different values $(0.1, 0.06, 0.03, 0.01)$ of the parameter ρ when tracking a sinusoidal parameter at two different frequencies, $\omega = 0.00039$ (blue) and $\omega = 0.0078$ (red) at SNR 5 dB. It is noted that the step-sizes using a larger value of the parameter ρ results in a larger bias since the deviation from the optimal step-size after convergence becomes larger. Observe that the bias present among the red curves for Alg 1 and Alg 3 has vanished for Alg 2.

sinusoids, results in a dominant part represented by $\hat{\psi}_t^*$ due to the similar values of the parameters h_t at two adjacent time-steps. This, creates a larger variance of $\hat{\nabla}_\mu(t)$. Conversely, less similarity between rapidly varying h_t and h_{t-1} results in a smaller value of $\hat{\psi}_t^*$ which in turn results in a smaller variance of $\hat{\nabla}_\mu(t)$.

Figure 4.18 illustrates the same tracking problem as above but at an SNR of 15 dB. Here, the effects of the different values of ρ are not as visible as in the low SNR scenario. This is partly due to that the convergence rate is slower for low values of ρ. There are two noticeable effects. First, there are tendencies of unstable behav-

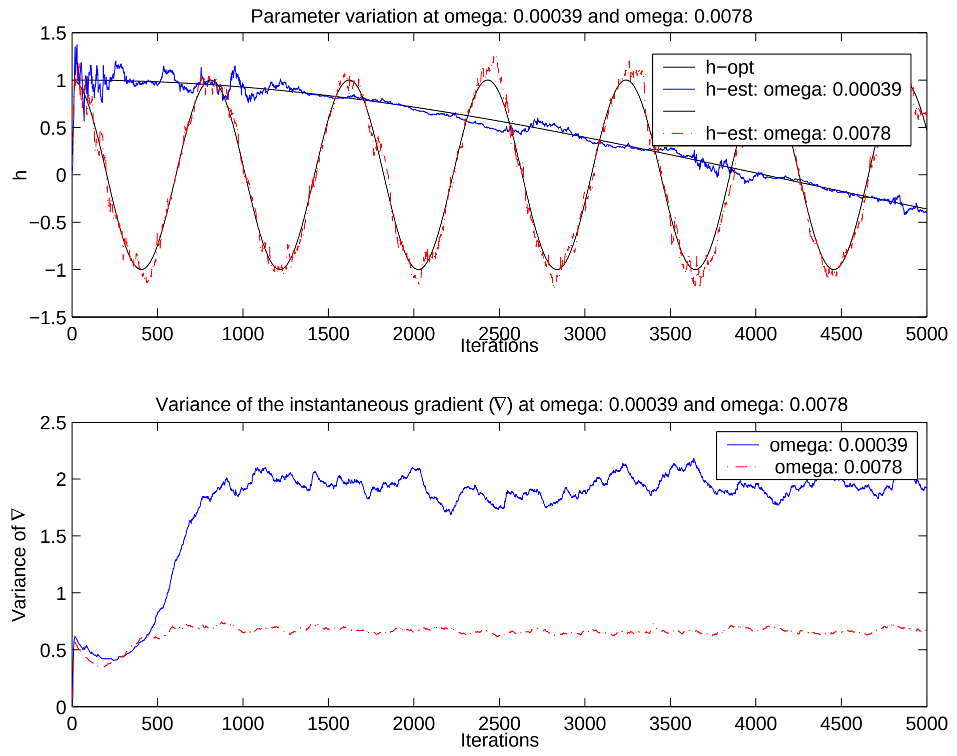

Figure 4.17: The parameter variation and the variance of the instantaneous gradient $\hat{\nabla}_\mu$ when tracking a sinusoidal parameter at two different frequencies, $\omega = 0.00039$ and $\omega = 0.0078$ at SNR 5 dB. The variance of the instantaneous gradient is here shown to be smaller when tracking a rapidly varying parameter.

ior in the beginning of the adaptation in the middle plot for the largest value of ρ. Second, there is a bias of the step-size for Alg 2, due to an improper value of the parameter α. Regarding Alg 1 and Alg 3, we observe that the convergence rate is faster for the Alg 3 than for Alg 1 due to the increased norm of the instantaneous gradient $\hat{\nabla}_\mu(t)$ introduced by the coefficient prediction filter through the gradient $\tilde{\psi}_t$, see (4.4).

We may now summarize our findings about the parameter ρ as follows:

- The parameter ρ controls the convergence rate of the step-size parameter $\hat{\mu}_t$ and its variance after convergence.

- In noisy environments, the value of the parameter ρ should not be chosen too

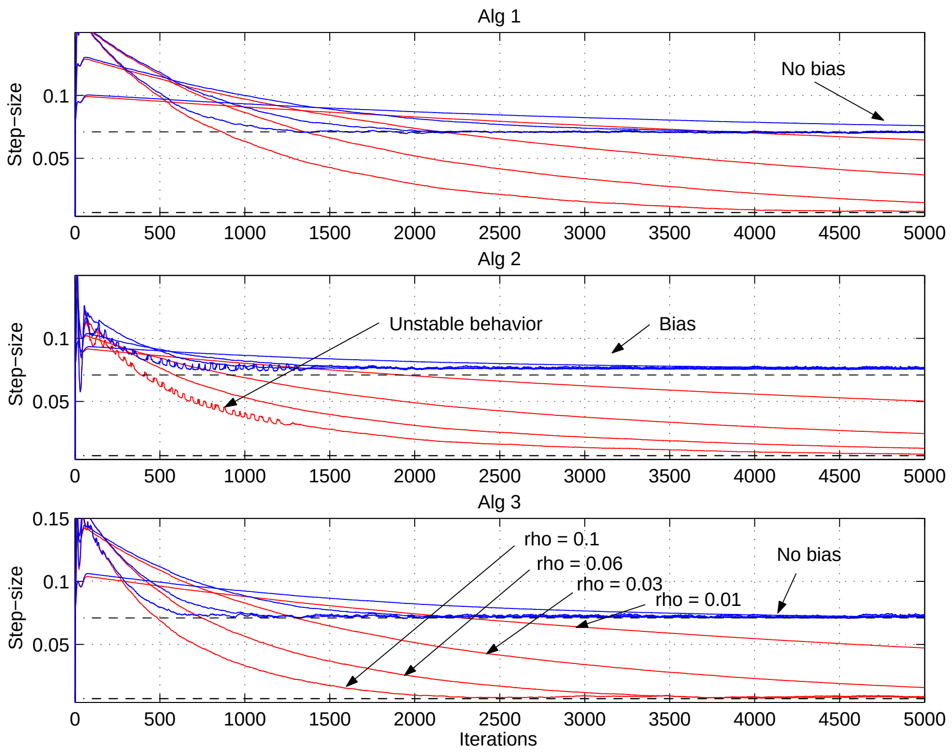

Figure 4.18: The convergence of the step-size for different values (0.1*,* 0.06*,* 0.03*,* 0.01*) of the parameter* ρ *when tracking a sinusoidal parameter at two different frequencies,* $\omega = 0.00039$ *(blue) and* $\omega = 0.0078$ *at SNR* 15 *dB (red). Step-sizes with a low value of* ρ *results in a slower convergence rate compared to step-sizes with a high value of* ρ*. Here it is noted that the convergence rate of Alg 3 is somewhat higher than for Alg 1 and Alg 2. Alg 2 also show signs of unstable behavior in the beginning of the simulation.*

large due to the increased variance of the resulting step-size and the corresponding bias.

- Slower parameter variations requires a larger value of the parameter ρ than faster varying parameters in order to obtain fast convergence.

- Among the algorithms tested here, Alg 1 seems to be the one that is most robust to different values of the parameter ρ.

4.4 Benveniste's simplified multiplicative method and the parameter α

We will now investigate the influence of the parameter α in Alg 2. Recall the step-size equation of Alg 1 before and after the simplification, cf. (3.18) and (3.20), that is,

$$\hat{\psi}_{t+1} = [I - \hat{\mu}_{t+1}\varphi_t\varphi_t^*]\hat{\psi}_t + \varphi_t\varepsilon_t \quad (4.7)$$
$$\hat{\psi}_{t+1} = \alpha\hat{\psi}_t + \varphi_t\varepsilon_t \ . \quad (4.8)$$

The simplification is performed by substituting the time-varying expression $[I - \hat{\mu}_{t+1}\varphi_t\varphi_t^*]$ with a constant gain matrix $\alpha\ I$. As mentioned in Chapter 3, this leads to a time-invariant filtering of the gradient $\varphi_t\varepsilon_t$ instead of a time-dependent filtering. The consequences of this simplification will now be investigated in a few tracking scenarios. Figure 4.19 illustrates the 'true' value of α, i.e. $[I - \hat{\mu}_{t+1}\varphi_t\varphi_t^*]$ for a one-tap parameter vector h_t when tracking sinusoids of different frequencies[8] at two values of the SNR. Since this, 'true' value, is directly related to the step-size $\hat{\mu}_t$ we notice that it will change according to the time variability of the parameter h_t and the SNR. Thus, by analyzing Figure 4.19 it becomes obvious that a fixed value of the parameter α may lead to unwanted effects in the step-size adaptation. Faster parameter variations require a smaller value of α compared to slower variations due to the larger step-size. This means in terms of time-dependent filtering that the cut-off frequency of the instantaneous low-pass filter (4.7) that controls the frequency components of $\varphi_t\varepsilon_t$ will change according to the parameter variations. Therefore, as we can see in Figure 4.16 (middle diagram), if the value of α is not tuned according to the noise level and the variations of the parameter h_t, i.e. to the correct step-size $\hat{\mu}_t$, then the filtering of $\varphi_t\varepsilon_t$ does not work properly. In Figure 4.16 this can be seen as an increased variance of the parameter $\hat{\mu}_t$ due to a noisy gradient and a bias. Furthermore, care must be taken when combining α and ρ. Our simulations indicates that a large value of α, i.e. $0.97 - 0.99$ combined with a large value of ρ, i.e. ~ 0.1 may lead to an unstable behavior of the adaptive step-size $\hat{\mu}_t$. An example of this phenomenon can be observed in the middle plot of Figure 4.18. Here the convergence of the step-size $\hat{\mu}_t$ for the simplified version shows signs of unstable behavior in the beginning of the adaptation when using $\rho = 0.1$. Another example of this behavior is observed in Figure 4.9 where the parameter tracking MSE of the simplified algorithm (Alg 2) is clearly affected.

We summarize our findings about the parameter α as follows:

[8]Here the frequencies 0.00039, 0.0039, 0.0078 and 0.0155 are used.

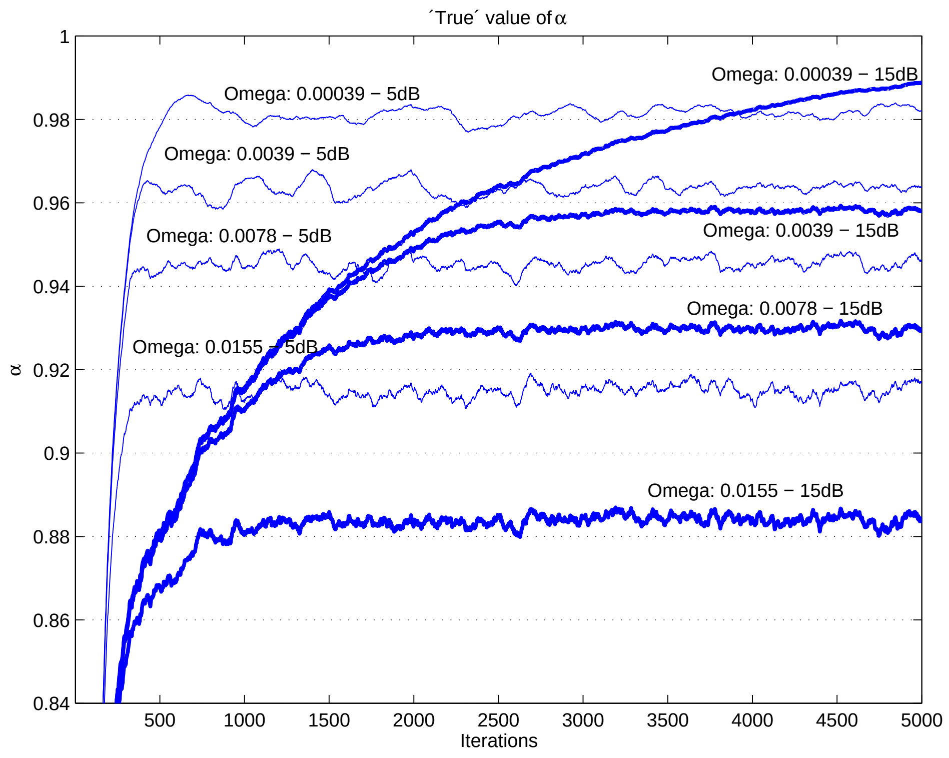

Figure 4.19: The 'true' value of the expression $[I - \hat{\mu}_{t+1}\varphi_t\varphi_t^*]$ at SNR, 5 dB and 15 dB and at different frequencies of the sinusoid. The thicker curves represents the value of α at SNR 15 dB. As seen, the value of the 'true' alpha varies significantly with the rate of the parameter variation or the SNR.

- The parameter α controls the time-invariant filtering of the gradient $\varphi_t\varepsilon_t$ in (4.8)

- The smoothing of the gradient $\hat{\nabla}_\mu(t)$ works only in cases where the value of the parameter α is correctly tuned, i.e. according to the speed of the parameter variations and the SNR. In our simulations we have noticed that, if the value of α used in the simulations is *higher* than the ´true´ or steady state value of $[\mathbf{I} - \hat{\mu}_{t+1}\varphi_t\varphi_t^*]$, then an increase in variance of $\hat{\mu}_t$ is obtained. This also results in a bias, with a magnitude that depends on the SNR level and the parameter variations of h_t. On the other hand, if the simulated value of α is *lower* than the ´true´ value, a smoothing of the step-size $\hat{\mu}_t$ is obtained. This reduces the bias in noisy conditions, or maybe in the best case, totally eliminates it.

The success of the simplification $[\mathbf{I} - \hat{\mu}_{t+1}\varphi_t\varphi_t^*] \to \alpha$ is therefore depending on the relation between the optimal step-size, i.e, the SNR and the parameter variation, the value of the parameter ρ and the chosen value of the parameter α. By simulations we have found that values in the region $0.92 - 0.97$ are appropriate. The value 0.95 is normally used in our simulations.

4.5 Parameter Tracking MSE of the GB algorithms

So far we have mostly been concerned with the convergence properties of the step-sizes for the different algorithms. We will now turn our attention to the parameter tracking MSE, $E\{|\tilde{h}_t|^2\}$. In the beginning of this chapter we illustrated the parameter tracking MSE for the GB methods together with the EB methods. In that case the aim was to show the significant differences between the EB and GB families. However, in those MSE-figures, Figures 4.3, 4.5, 4.7 and 4.9 the differences between the different GB algorithms were hardly noticeable. Therefore we will now closer investigate the parameter tracking MSE for these algorithms. Based on what we know so far about the GB algorithms and the behavior of their step-size adaptation in different tracking situations we might suspect that they will differ also when it comes to the parameter tracking MSE, since it is directly proportional to the step-size μ_t. Figure 4.20 shows the parameter tracking MSE and the variance of the instantaneous gradient $\hat{\nabla}_\mu(t)$ of the GB algorithms when tracking a rapidly varying sinusoid at two different signal-to-noise ratios. In the case of the lower noise level (SNR 15dB) we note that the algorithms perform about the same, only small differences in the steady-state tracking MSE can be observed. However, if the noise level increases, then, we observe that the MSE for Alg 3 deviates from the other two GB algorithms. This is explained by looking at the lower plot of Figure 4.20 where the variance of the instantaneous gradient $\hat{\nabla}_\mu(t)$ is plotted versus the number of iterations. Looking at the upper curves (5dB), we can clearly see that the variance of $\hat{\nabla}_\mu(t)$ for Alg 3 is larger than that of the other two algorithms. This is, as mentioned before, a result of the dynamics from the coefficient prediction filter. We also observe that the variance of the $\hat{\nabla}_\mu(t)$ for Alg 2 is larger than that of Alg 1. This is explained by a too large value of the parameter α.

Now, continuing our study of the parameter tracking MSE we will *decrease* the frequency of the sinusoid from the previous simulation. This is illustrated in Figure 4.21. Similar to the discussion above we observe that the algorithms perform about the same at high SNR and that the differences show up at low SNR. However, in this case we notice that Alg 2 produces the lowest tracking MSE in both situations. This is explained by the lower diagram of Figure 4.21. Here, the effect of the low-

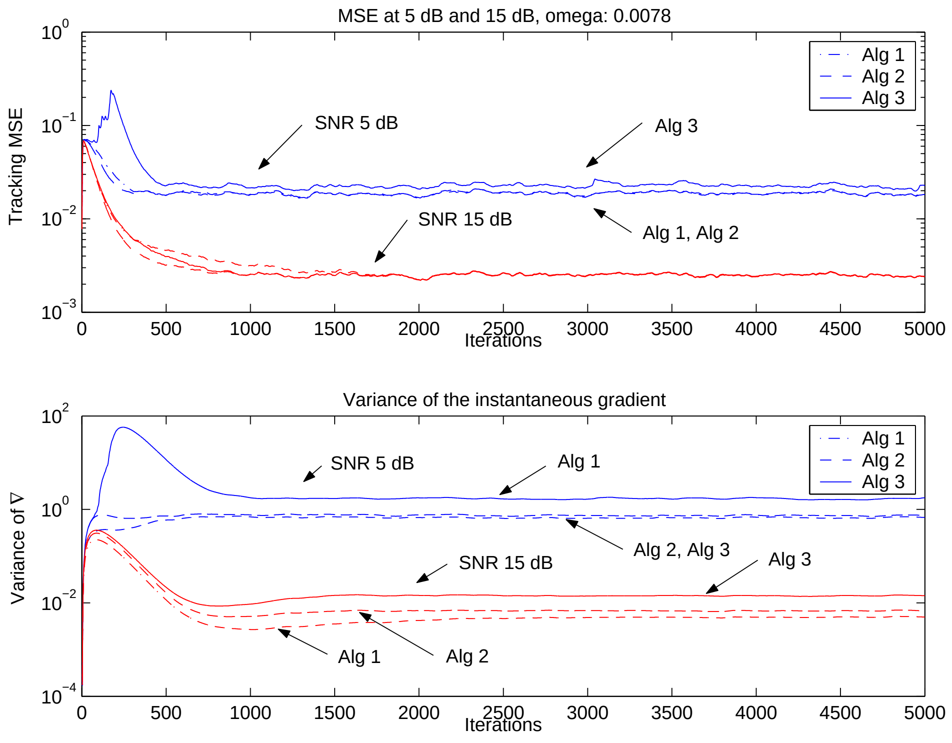

Figure 4.20: Upper diagram: The parameter tracking MSE when tracking a rapidly varying sinusoidal parameter at 5dB and 15dB, respectively. In the case of the higher SNR hardly any differences apart from the convergence rate in the beginning of the adaptation are noticed. All methods converge into the same steady-state tracking MSE. However, at the lower SNR we notice that Alg 3 deviates from the other two. Lower diagram: The variance of the instantaneous gradient $\hat{\nabla}_\mu(t)$.

pass filtering of $\varphi_t \varepsilon_t$ becomes visible. In this case the optimal step-size is quite small due to the slowly varying parameter. This produces, as we know from earlier discussions, a large value of the 'true' α and since we are using $\alpha = 0.95$ which is *lower* than the 'true' value of α, smoothing of $\hat{\mu}_t$ takes place.

4.6 Tracking with individual step-size for each parameter

In the previous simulations we have only used one time varying parameter in order to describe the behavior of the different algorithms. However, tracking of one parameter is of course not very realistic since tracking in real applications normally

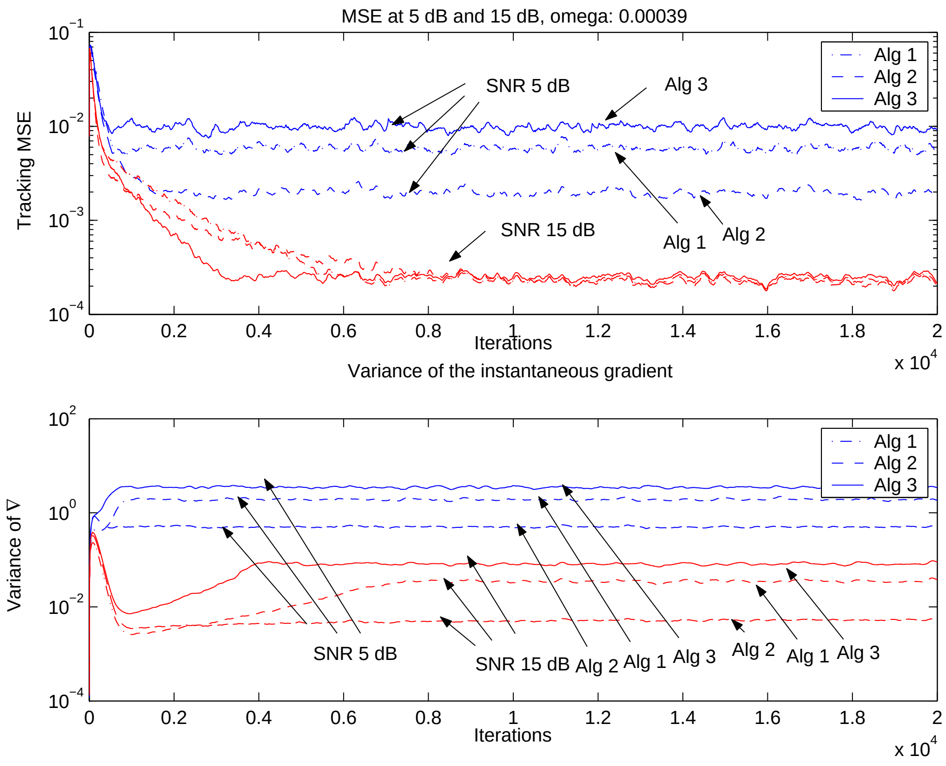

Figure 4.21: *The parameter tracking MSE when tracking a slow sinusoidal parameter at 5 dB and 15 dB, respectively. Compared to the fast varying parameter depicted in Figure 4.20 we here note that the variance of the instantaneous gradient* $\hat{\nabla}_{\mu}(t)$ *(lower diagram) is larger and that Alg 2 produces a much lower tracking MSE than the other algorithms at SNR 5 dB. No significant differences are observed at 15dB.*

involves more than one parameter. When it comes to tracking of several parameters based on the algorithms presented in Chapter 3 we can choose to use the same step-size for all parameters or to use individual step-sizes. The latter is easily obtained by replacing the scalar step-size parameter μ_t with a diagonal matrix containing a variable step-size for each parameter. Tracking of several time-varying parameters will increase the complexity of the adaptive algorithm. However, the increased complexity is traded for better performance. For example, a typical situation where individual adaptation of the step-size is motivated is when the parameters of interest vary at different rates, and if they change their typical behavior over time. In the case of the automatically tuned SWLMS algorithm, individual adaptation adds another degree of freedom not included in the original SWLMS algorithm. This has to do with the restrictions of the SWLMS hypermodel. Since we use a diagonal hypermodel with the *same* transfer functions along the diagonal, i.e., the parameters h_t are assumed to be governed by the same dynamics, individual control of the parameters is not as flexible as in the WLMS algorithm. In the following tracking scenario we will illustrate the performance gain introduced by using individual step-size adaptation when tracking four individually time-varying parameters[9]. Figure 4.22 shows the parameter variations for the different parameters. The

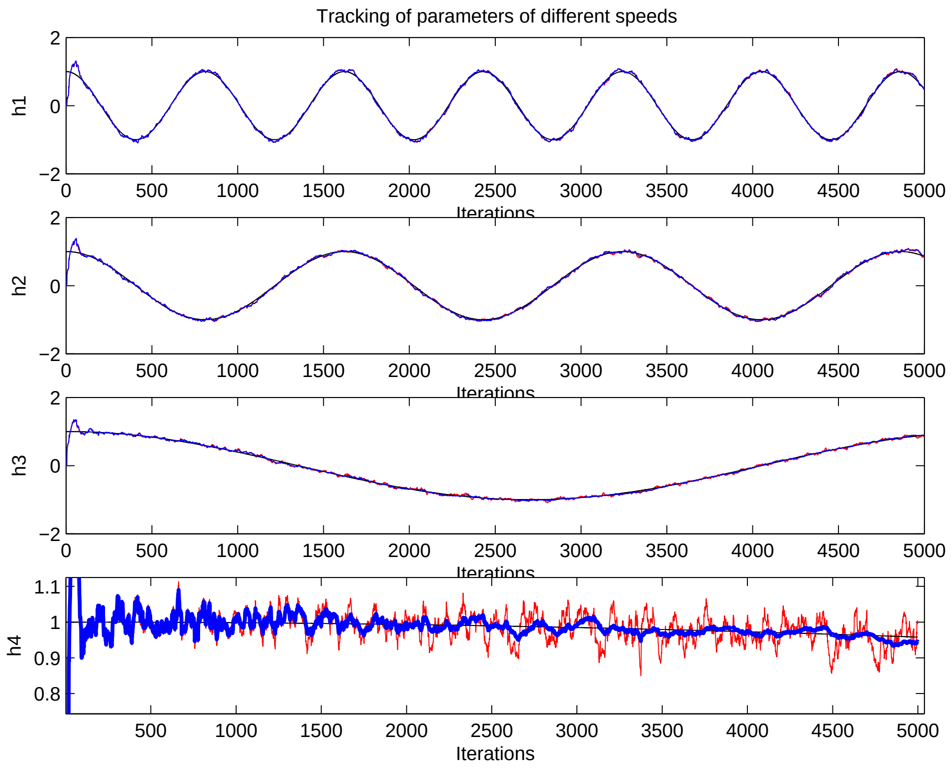

Figure 4.22: Tracking performed by Alg 1 of 4 individually time-varying sinusoidal parameters at SNR 15 *dB. In the diagrams, the red color illustrates the common step-size and blue color illustrates the individual step-sizes. It can be noted in the bottom diagram, i.e. the slowest varying parameter* h_t^4*, that the individual step-size is significantly smaller than the common step-size towards the end of the simulation. This, results in a smaller parameter tracking MSE, see Figure 4.24.*

[9]Hypermodel IRW was used in this simulation.

advantage of letting the step-size adjust itself individually to the time-varying parameters is obvious in the lower plot. In order to cope with the slowly varying fourth parameter the individually tuned step-size converges to a much smaller value than for the other parameters. This, clearly improves the tracking performance due to the smaller variance of the estimated parameter h_t^4. The convergence of the step-sizes is illustrated in Figure 4.23. Here, we can also see that the step-size for the common step-size algorithm is slightly smaller than that of the first parameter in the individual case. Figure 4.24 shows the parameter tracking MSE for the two

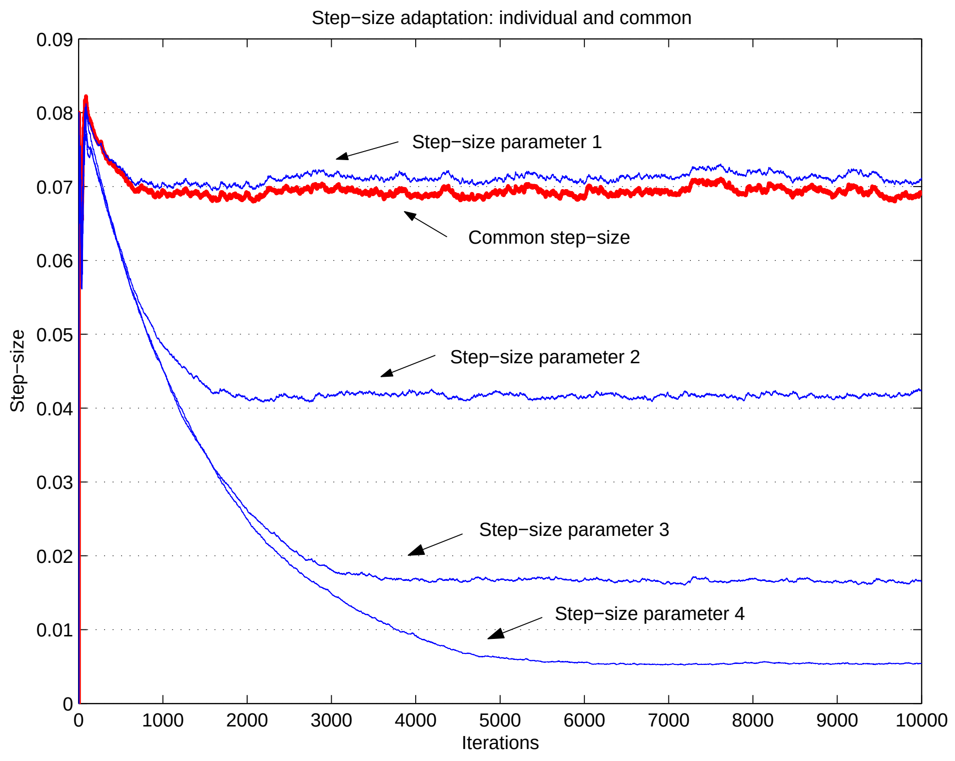

Figure 4.23: The convergence of the step-size for individual and common step-size. It can be noted that the individual step-sizes adjust themselves to the rate of their time-varying parameter. In the case of the common step-size, the resulting step-size adjust itself to the parameter that contribute the most to the resulting tracking error, which in this case is parameter 1,

algorithms. As expected, the performance is significantly improved by individual tuning of the different step-sizes. From our simulations we can conclude that

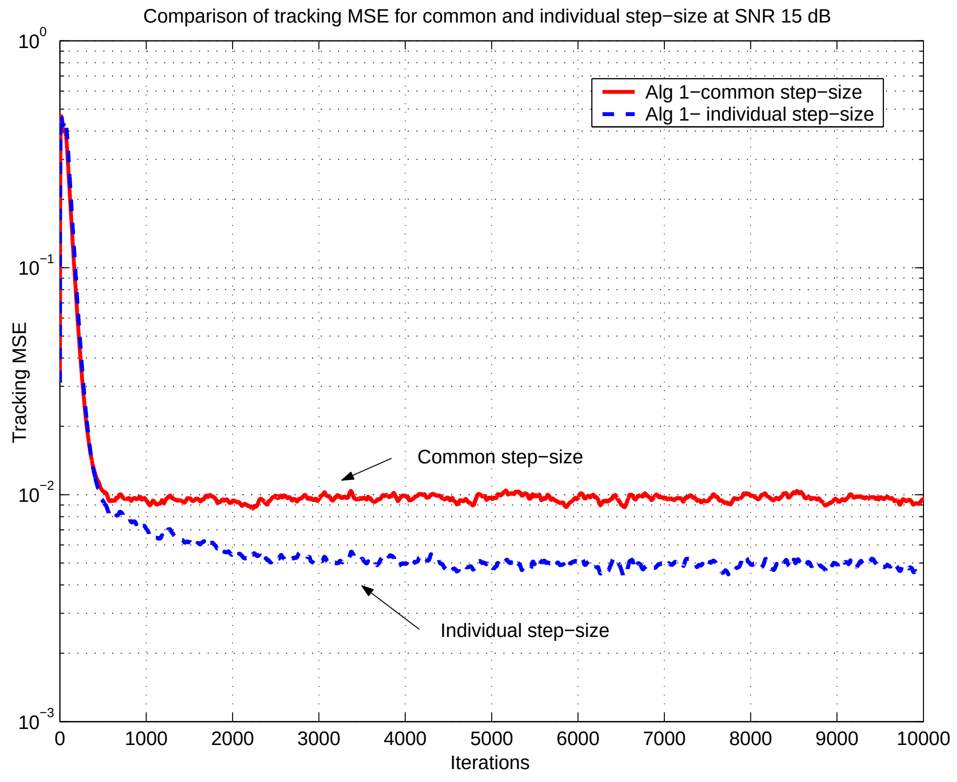

Figure 4.24: Tracking MSE when tracking the four sinusoidal parameters with differing frequencies in Figure 4.22 at SNR 15 *dB. It is here clearly noted that the use of individual step-size adaptation improves the tracking performance compared to that of a common step-size.*

- Individual step-size adaptation enhances the performance of the tracking algorithm if the parameters are varying at significantly different rates.
- Individual step-size adaptation should be used if the extra complexity involved is not a major problem.

4.7 Tracking in the Newton direction using R^{-1}

It is well known that the convergence properties of adaptive algorithms such as LMS is improved by using the Newton direction instead of the gradient direction when estimating parameters in a situation with colored regressors. However, when it comes to tracking, i.e. when we are dealing with a moving performance surface

- Is is also true then? In order to answer this question we recall *why* the Newton direction improves the convergence rate of the adaptation. For a non-moving performance surface, this is simply explained by the fact that the search for the MMSE point always proceeds in the direction of the minimum. However, when it comes to tracking, and a moving surface, the Newton direction at one time-step might be significantly different in the next time-step. To what extent this is true depends on several things such as

- the shape of the performance surface, i.e. the correlation of the regressors.
- the rate of change of the time-varying parameters
- the variation, i.e. smooth or randomly varying, of the time-varying parameters.
- the correlation between the time-varying parameters

First of all, if the performance surface in a static scenario is completely symmetrical in every direction due to uncorrelated regressors, then the gradient direction and the Newton direction is the same and the use of $\mathbf{R^{-1}}$ in the algorithm provides no advantage. It is when we are dealing with an elongated performance surface that the use of the Newton direction might be of interest.

Secondly, when it comes to the rate of change of the time-varying parameters, slowly varying parameters do not change the position of the MMSE point as much as rapidly varying parameters do, therefore, problems with using the Newton direction may only arise in the case of rapidly varying parameters. Thirdly, assuming that we are dealing with rapidly varying parameters, then, the direction of the movement of the performance surface becomes very important.

Thirdly, the moving direction of the performance surface is depending on the nature of the time-varying parameters. For example, assume two uncorrelated random walk parameters. These will change the position of the MMSE point in an unpredictable way, see Figure 2.8. In this case, the Newton direction will change dramatically from one time-step to another. Therefore, little is gained by including the Newton direction. On the other hand, if we assume two uncorrelated IRW parameters, then they will produce fairly smooth changes of the MMSE point, see Figure 2.7. In that case, the probability that the Newton direction will change to an entirely new direction in the next time-step is much lower. Then, we might gain something by using the Newton direction.

Fourthly, since the movement of the performance surface also is affected by the *correlation* between the parameters this has to be taken into account. Now, assume two rapidly varying parameters. If they are positively correlated, then they will move the MMSE point diagonally along a positive slope in the parameter space spanned by h_1 and h_2. If they are anti-correlated, then the slope will be negative. See Figure 4.25 and Figure 4.26. Now to the interesting part. Whether the New-

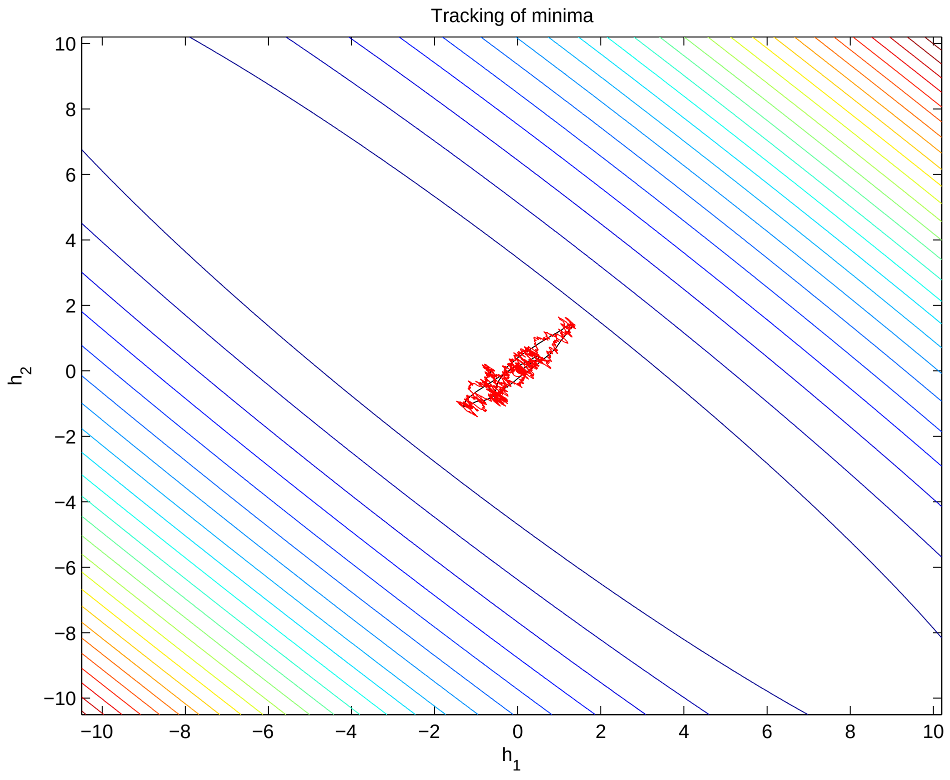

Figure 4.25: The trajectories for fast varying correlated parameters. In this case the direction of the trajectories becomes orthogonal to the elongated direction of the performance surface. The use of the Newton direction would here decrease the tracking performance, i.e. increase the parameter tracking MSE as well as decreasing the convergence rate, as compared to neglecting the factor $\mathbf{R}^{-1}$

.

ton direction should be used or not is to a large extent depending on the direction of the moving performance surface in combination with the shape of the surface, due to the correlation between the regressor elements. In our simulations we have found that, if the elongation of the performance surface agrees with the direction of the moving MMSE point, or, if the performance surface is elongated and the parameters h_t are uncorrelated, then the Newton direction should be used. In other situations, the use of the Newton direction may actually decrease the performance. However, as always, when it comes to real applications the designer might lack

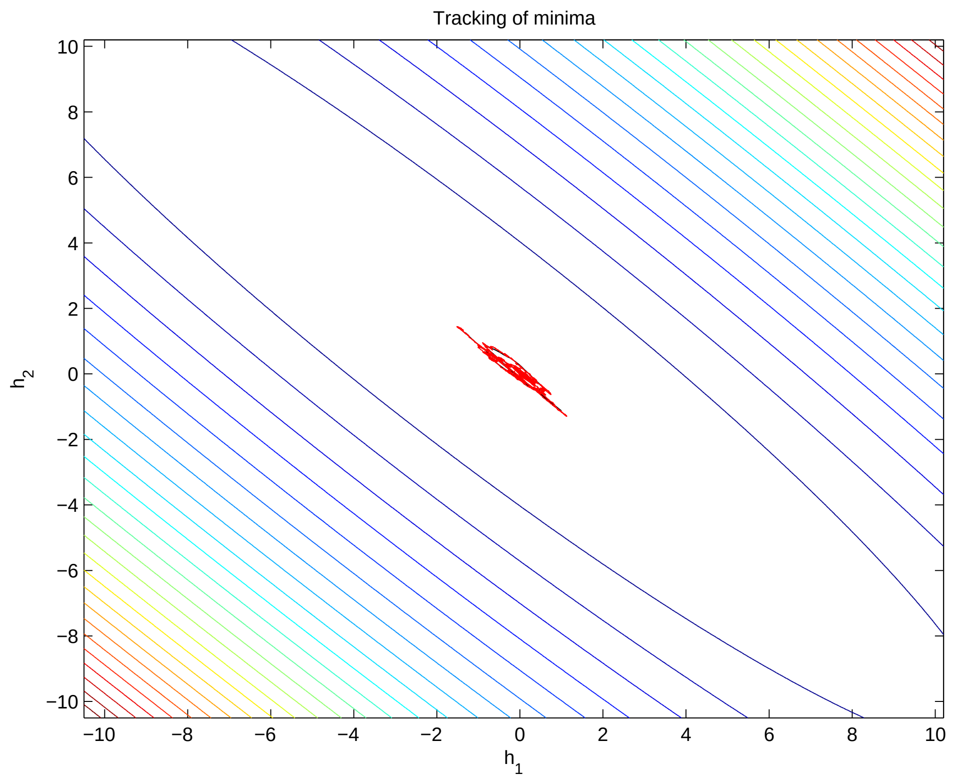

Figure 4.26: The trajectories for fast varying anti-correlated parameters. In this case the direction of the trajectories agree with the elongated direction of the performance surface. Therefore, the convergence rate will be improved by including the factor $\mathbf{R}^{-1}$ *due to the larger steps in the elongated direction.*

important information about the parameters needed to decide wether to use the Newton direction or not. This problem has to be treated from case to case.

4.8 Performance gain obtained by hypermodels and step-size adaptation

So far we have primarily focused on the details and the behavior of the different versions of adaptive step-size methods. We will now turn our attention to the automatically tuned SWLMS algorithm, as a whole, as a tool used to track parameters of time-varying systems. We will here illustrate the performance gain obtained by using automatic tuning of the step-size as well as how different levels of prior information, i.e., different *hypermodels*[10] improve the tracking performance.

4.8.1 Tracking of Rayleigh fading parameters

In this subsection we consider tracking of one Rayleigh fading [32] channel parameter in a wireless communication scenario between a base station and a mobile unit. Here, the time-variability of the channel parameter is caused by the relative motion, i.e. the doppler frequency, between the mobile unit and the base station. Other factors, such as the environment surrounding the mobile, except the SNR, is not taken in to account. We will therefore focus on the doppler frequency as the main factor influencing the time-varying channel. The simulation is performed over 10000 iterations, where the first 7500 iterations represent a fast acceleration[11], then, an abrupt change occur and the acceleration is started again from the beginning. The SNR level in the example is 15 dB. Figure 4.27 illustrates the tracking performance of the following algorithms

1. LMS: SWLMS with hypermodel RW and no adaptation of the step-size. No prior information about the parameters is included in the design. The step-size ($\mu = 0.15$) is optimized for a mobile speed of 100 km/h.

2. VS-LMS: Automatic tuning is now added to the previous LMS algorithm.

3. SWLMS: SWLMS with hypermodel IRW and no adaptation of the step-size. Information about smooth parameter behavior is now included. The fixed step-size ($\mu = 0.08$) is optimized for 100 km/h.

4. VS-SWLMS-IRW (Alg 1 with hypermodel IRW) : Automatic tuning of the

[10] For a thorough description and analysis of the use of hypermodels together with the WLMS and SWLMS algorithms we refer to [4], [11], [12], [13] and [14].

[11] Here the doppler frequency is changed from $0 - 1000$Hz. At the carrier frequency (1800Mhz) in this example, and the sampling rate 270000 Hz, this doppler shift corresponds to an acceleration from $0 - 600$ km/h in 0.0278 seconds. This is of course unrealistic, however, in this case it is used to illustrates the behavior of the algorithms in an extreme situation.

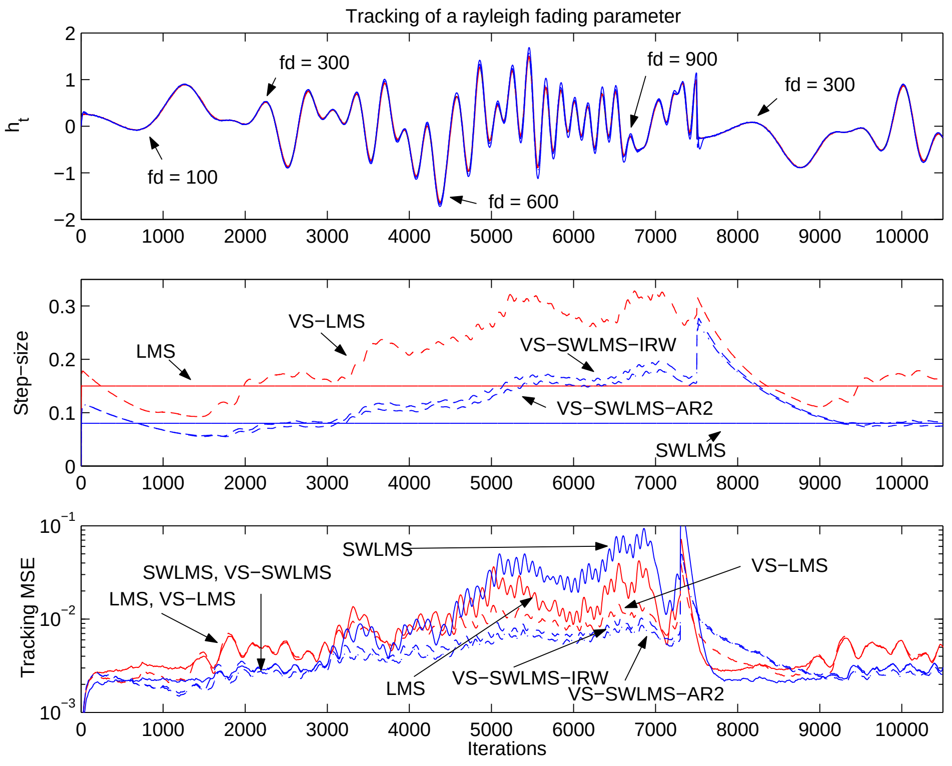

Figure 4.27: Tracking of a Rayleigh fading parameter. The doppler frequency is gradually increasing from $0-1000$Hz, then the parameter is exposed to an abrupt change and the acceleration is started again.

SWLMS algorithm and information about smooth behavior of the parameters is included, by using the IRW hypermodel.

5. VS-SWLMS-AR2 (Alg 1 with hypermodel AR2): Same as in 4, but, in this case we also assume exact knowledge of the doppler frequency.

Remark: Note that Alg 1 is chosen in 4. and 5. in order to represent a suitable VS-SWLMS algorithm. This choice is based on the results in the previous simulations where Alg 1 has shown to be superior compared to the other candidates.

In the early stage of the adaptation (iteration $0-3000$), the gain from the hypermodelling is visible in the parameter tracking MSE plot (lower diagram). The parameter tracking MSE for the algorithms using IRW and AR2 hypermodels is significantly smaller than that of the LMS and the VS-LMS algorithms. However, as the variation of the parameter increases, the effects of the variable step-size becomes more significant. The fixed step-size algorithms (LMS and SWLMS) clearly suffers from a severely increasing parameter tracking MSE compared to the VS algorithms. The SWLMS algorithm is also shown to perform worse than the LMS from iteration 4500 until 7500. This is explained by the large lag-error due to its smaller step-size. However, when the parameter variation slows down again, at iteration 7500, the SWLMS recovers its superiority. From Figure 4.27 we notice that the hypermodelling improves the performance compared to the classical LMS algorithm and its variable step-size companion VS-LMS at doppler frequencies up to approximately 400Hz. After that, the improvements obtained by the VS algo-

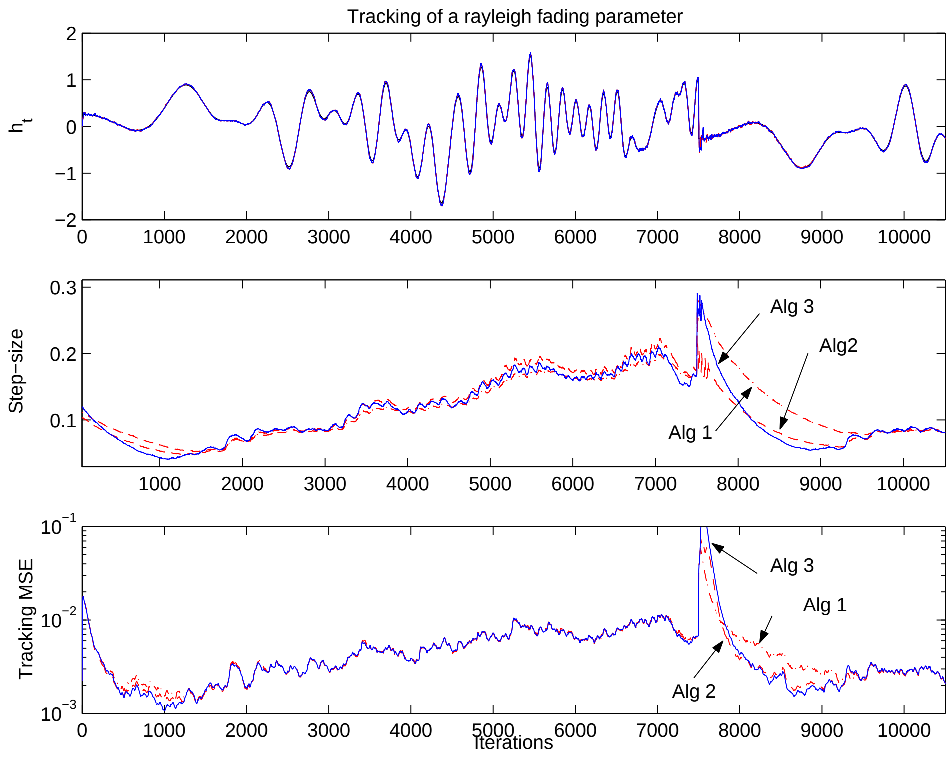

Figure 4.28: Tracking of a Rayleigh fading parameter with the GB algorithms. The tracking performance is about the same for all of the algorithms except for the convergence rates and the unstable behavior of Alg 3 and Alg 2 directly after the abrupt parameter change.

rithms improves even further. Since the fixed step-sizes of LMS and SWLMS are optimized for the speed 100 km/h, i.e. a doppler frequency of ~ 170 Hz, they cannot cope with the increasing variation of the parameter in the same way as the VS algorithms do. When it comes to the information about the doppler frequency used in the AR2 hypermodel compared to the IRW hypermodel, it clearly becomes visible at $f_d \sim 200$ Hz that this enhances the performance. However, since the difference is fairly small, we conclude that the IRW hypermodel works well even for very fast variations without this information. Furthermore, it is also known, see [14], that, for parameter vectors of dimension > 1, the SWLMS algorithm is significantly better than the classical LMS algorithm. Then, also different choices of the hypermodel becomes more visible. In Figure 4.28, the same tracking scenario as above is performed with the different GB algorithms (Alg 1 - Alg 3). In accordance to what we have seen in the previous simulations, no significant differences between these algorithms are observed, except from the faster convergence rate of the Alg 3 algorithm. The parameter tracking MSE of the algorithms are almost identical except for the convergence phase in the beginning of the adaptation and after the abrupt change. However, after the abrupt parameter change, signs of unstable behavior is present in both Alg 3 and Alg 2.

4.9 Concluding remarks

In this chapter a comprehensive simulation study of the step-size algorithms from Chapter 3 has been performed. The algorithms have been investigated in various scenarios designed to bring out their characteristic behavior. The purpose of this study was to find the best candidate of the presented algorithms to be used together with the SWLMS algorithm in order to obtain a self tuning tracking algorithm. We will now, based on the simulations performed in this chapter, summarize and conclude our findings in order to give a potential user guidelines on how to apply a suitable automatic step-size scheme in combination with the SWLMS algorithm.

In the beginning of this chapter a comparison between the error based (EB) algorithms and the gradient based (GB) algorithms was performed. It was demonstrated that the EB algorithms were working properly in situations where the parameter vector h_t was static or slowly varying. However, in cases where the rate of change of the parameter vector h_t was faster, limitations in the structure of the EB algorithms precludes their use in such situations. This clearly favors the use of the GB methods.

The different algorithms were also investigated under various noise conditions. It was shown that the EB algorithms were more sensitive to noise than the GB methods. In tracking situations involving a rapidly varying parameter vector h_t and a noise that may have varying power, we have through the performed simulations seen that the GB algorithms are superior compared to the EB algorithms. Furthermore, it should be noted that the EB algorithms never outperformed the GB algorithms except from simulation 1, i.e. in the estimation of a static parameter. From this we conclude that the GB algorithms can be used in situations where the parameter vector is changing both slowly and rapidly.

Among the GB algorithms, we investigated three different step-size methods which we called Alg 1, Alg 2, and Alg 3 (cf page 43). The first algorithm was based on the classical LMS algorithm. The second algorithm was a simplification of Alg 1 in which the complexity was slightly reduced. Alg 3, was a completely new step-size algorithm derived directly from the SWLMS equations. One of the primary goals with this thesis was to answer the question, whether or not improved tracking performance was possible to attain by deriving a new step-size adjustment method based on the SWLMS equations, instead of using one of the step-size methods previously derived for the LMS algorithm. It was shown that the new step-size method possessed almost the same tracking performance as the other two step-size methods. A small improvement of the convergence rate was discovered, but, this came at the price of an increase of the complexity which could hardly be motivated in a practical application.

Our recommendations when it comes to using an automatic step-size scheme together with the SWLMS algorithms are as follows:

- A simple gradient based method, based on the LMS algorithm works perfectly fine in order to automatically adjust the step-size μ on-line. In our case, *Benvenistes* algorithm is chosen as the winner among the investigated algorithms. It is simple and has shown to be more robust against noise, as compared to the other algorithms. However, its simplification possessed an interesting property in terms of low-pass filtering of the instantaneous gradient, $\hat{\nabla}_\mu(t)$, given that a suitable value of the parameter α was used. This filtering reduced the deviation from the optimally calculated step-size otherwise generated by a noisy gradient $\hat{\nabla}_\mu(t)$.

- If using the simplified version of Benvenistes algorithm, care must be taken when deciding on the value of the parameter α. Approximate knowledge about the rate of the variations of the parameter vector h_t and the SNR is recommended before setting the value of α. In order to give one numerical value that has worked well in most of our simulation, we would recommend

$\alpha = 0.95$. However, values in the interval $0.92 - 0.97$ are all possible. A general rule is that if the parameter is rapidly varying, then a lower value of α should be used.

- When it comes to the numerical value of the parameter ρ we recommend the region $0.01 - 0.08$. In most of the simulations we have used the value 0.07.

At the end of this chapter we also investigated whether or not to use the Newton direction in tracking problem as well as the gain obtained by using individual step-size adaptation for each parameter in the parameter vector h_t. The recommendations concerning the the Newton direction are as follows:

- When tracking rapidly varying parameters, *do not use the Newton direction* if not enough information about the parameter variation, the correlation between the regressors, and the correlation between the parameters is available.
- When tracking slowly varying parameters, information about the correlation of the regressors is the main factor influencing the choice of using $\mathbf{R}^{-1}$. If this information is available or is easy to estimate, then the use of $\mathbf{R}^{-1}$ is motivated.

When it comes to individual step-size adaptation our findings and recommendations are the following:

- If the available computing power allows for the extra complexity introduced by the individual tuning, then it should definitely be used.
- The performance gain of individual tuning of the step-size μ increases with the size of the parameters vector h_t. Naturally, it also increases with the difference in drift rate between different elements of h_t. Therefore, if complexity is an issue, then study the difference in the assumed drift rate between the parameters and take into account the size of h_t.

With these conclusions we will now move on to a real application in which the automatically tuned SWLMS algorithm is applied and compared with other adaptive algorithms.

Chapter 5

Adaptive Channel Equalization in EDGE - A Case Study

5.1 Introduction

In this chapter we will investigate the performance of some of the previously presented algorithms in a realistic simulation of a specific application. We will here consider a communication scenario where information is sent over a wireless channel. The task of the adaptive algorithm is to compensate for negative effects in transmission originating from the wireless channel.

5.2 EDGE

We shall evaluate the block error rate (BLER) performance of a delayed decision-feedback sequence estimator (DDFSE) [33] in conjunction with the proposed algorithms on multipath fading channels associated with the EDGE [34] air interface. This radio interface is based on GSM, which implies that the symbol rate and the slot format (in terms of symbols) are the same, as illustrated in Figure 5.1. In EDGE, the bit rate is adapted to the long-term channel conditions by selecting one of nine different modulation and coding schemes (MCS). For MCS five and higher, the modulation is 8PSK with linearized GMSK pulse shaping. In this study we shall focus on a Rural Area (RA) scenario since in such cases differences in tracking accuracy will be of particular interest.

An adequate single receiver antenna transmission model for short EDGE/GSM

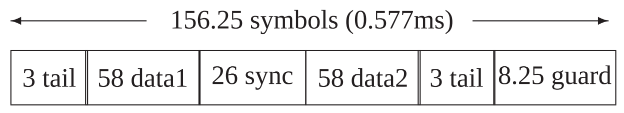

Figure 5.1: The Edge slot format.

channels, such as Rural Area propagations [35], can be expressed as

$$y_t = \varphi_t^* h_t + v_t, \tag{5.1}$$

where v_t represents noise and interference, h_t is a scalar time varying complex-valued gain and φ_t^* represents a scalar regressor, expressed as

$$\varphi_t^* = \sum_{k=0}^{M} g_k s_{t-k} \quad . \tag{5.2}$$

Above, g_k and s_t denote channel taps and transmitted symbols, respectively. It should be noted that the correct channel model normally is described by an M-tap FIR-filter with time-varying taps and that the above model (5.1), (5.2) only serves as a good approximation over an EDGE-slot. In the case of flat fading channels the approximation error will vanish. For short EDGE channels, the intersymbol interference modeled by (5.2) is mainly caused by the pulse shaping and the receiver (RX) filtering, whereas h_t models flat fading as well as impacts caused by frequency offsets. In the case of a pure frequency offset, ω_o, originating from a deviation of the carrier frequency from the receiver oscillator frequency, the gain h_t is given by

$$h_t = e^{j\omega_o t} \quad . \tag{5.3}$$

For RA channels, the spectrum of h_t is approximately described by a Rice model [35],

$$\Phi_h(\omega) = A\delta(\omega - 0.7\omega_D) + B\Phi_J(\omega, \omega_D) \quad , \tag{5.4}$$

for some constants A and B, where $\Phi_J(\omega, \omega_D)$ represents the spectrum of Jakes model [32] with a Doppler frequency ω_D. In real scenarios, a combination of the models (5.3) and (5.4) is normally used since both offset and fading need to be modeled.

The taps g_k are in this presentation estimated by Least Squares (LS) over the sync data interval (Figure 5.1), under the assumption that $h_t = 1$, and are then held constant over the entire slot. The estimated channel taps relate to the slot synchronization position that yield the smallest cumulative squared LS residuals. Tracking of the time varying tap h_t begins in the middle of the sync interval. Decision-directed mode, where known sync symbols are replaced by detected symbols $\bar{s}_t$, is used within the data blocks. The data2 block in Figure 5.1 is first detected in the forward direction, and the data1 block is then detected in the backward direction (time-reversal detection). Hence, the equalization will be performed on two half slots. The considered detector structure is depicted in Figure 5.2. The tracker has to perform d-step prediction, since it works on d-step delayed data. The prefilter is designed, for each slot half, as a feedforward filter of an MMSE decision-feedback equalizer (DFE).[1] In the simulations below, a square root raised cosine RX filter

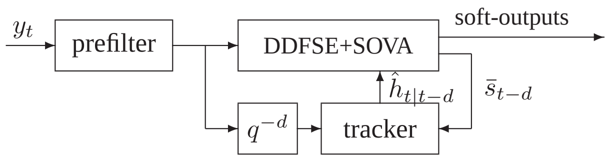

Figure 5.2: Soft-output delayed decision feedback sequence estimator (DDFSE) with d-step prediction tracking of a flat fading channel. The tracker operates on pre-filtered data and delayed symbol decisions $\bar{s}_{t-d}$. Soft sample values for convolutional decoding are here delivered by the soft-output Viterbi algorithm (SOVA) [37].

with single-side bandwidth of $150\,\mathrm{kHz}$ and roll-off factor of 0.15 is used. Figure 5.3 shows the block error rate (BLER) when transmitting over a rather severe fading channel in the 1800 MHz band. In this example, the prefilter consists of 15 taps, the DDFSE consists of two maximum likelihood sequence estimate (MLSE) taps and two feedback taps, $M = 3$ in (5.2), and a decision delay d of three symbols is used. Ideal frequency hopping is assumed. For the tracking, a d-step ahead VS-SWLMS prediction with adaptive and fixed step-size is used. The predictors are designed for an *integrated* random walk model (IRW)[2]. In the upper plot of Figure 5.3 no frequency offset is assumed. The sender and the receiver are in this case perfectly synchronized. Therefore the BLER for the different algorithms are smaller than in the lower figure where an offset of 200 Hz is assumed. Comparing the two cases we also notice that the differences between the algorithms using adaptive step-size and the ones using fixed step-sizes are larger in the 200 Hz offset case.

[1]This receiver structure is similar to the one used in [36] (single branch), but where we have included one-tap tracking and use another soft-output scheme.

[2]The reason for this choice is a compromise between performance and complexity. The use of a hypermodel that includes more information about the assumed parameter behavior would probably result in better performance. However, the IRW model captures the significant behavior of the parameter and is therefore considered as a reasonable choice.

The performance of the gradient-based (GB) VS-SWLMS algorithms from Chapter 4 (Alg 1 -Alg 3) is here compared to algorithms without adaptive step-size, here denoted SWLMS and NLMS[3]. Also the VS-LMS algorithm is evaluated. The

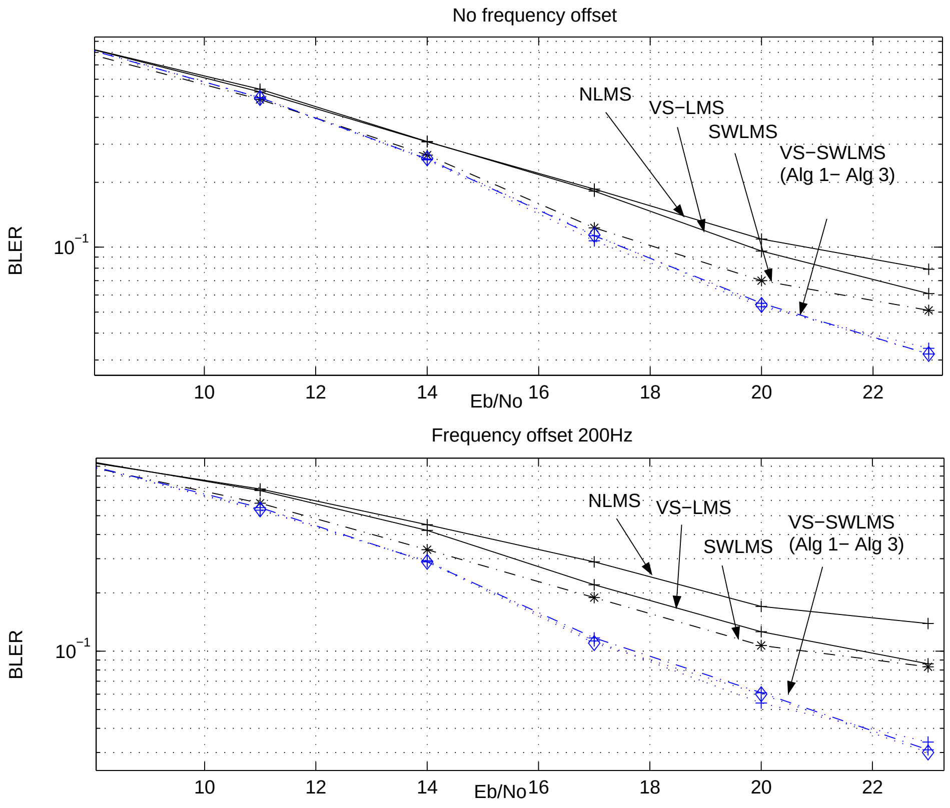

Figure 5.3: Block error rate for modulation and coding scheme nr 7 (MCS-7) and an RA channel at 1800MHz as a function of Eb/No, at a mobile speed of 200 *km/h with and without frequency offset. Comparison between VS-SWLMS (Alg 1 - Alg 3), SWLMS with fixed step size, Variable Step-size LMS (VS-LMS) and (normalized) LMS with fixed step size (NLMS).*

design parameter ρ in the GB algorithms is set to 0.07, and the variable step-size μ_t is adapted over the slots. The normalized LMS (NLMS) algorithm with fixed gain here corresponds to a SWLMS with $d_1 = -1$ and $d_2 = 0$ (μ is normalized by σ_φ^2 in order to make the algorithm independent of the magnitude of the regressors).

[3]The normalized version of the LMS algorithm is used in order to make the algorithm independent of the magnitude of the regressors. The initial step-size $\mu_i nit$ is set to 0.15.

The fixed step-sizes were optimized for zero frequency offset and a mobile speed of 100 km/h and an SNR of 15 dB, which provides good performance on average over vehicle speeds between zero and 200 km/h. These values were also used as initial values for $\mu = \bar{\mu}\sigma_{\varphi}^{2}$ in the algorithms with adaptive step-size.

The VS-SWLMS algorithms overall provides superior performance compared to that of the VS-LMS algorithm. However, no significant differences are observed between the different GB algorithms (Alg 1 - Alg 3). At 200 km/h and 10% BLER in the upper plot of Figure 5.3, we obtain an improvement of about 3.6 dB in comparison with the VS-LMS algorithm. Note also that the SWLMS with incorrectly

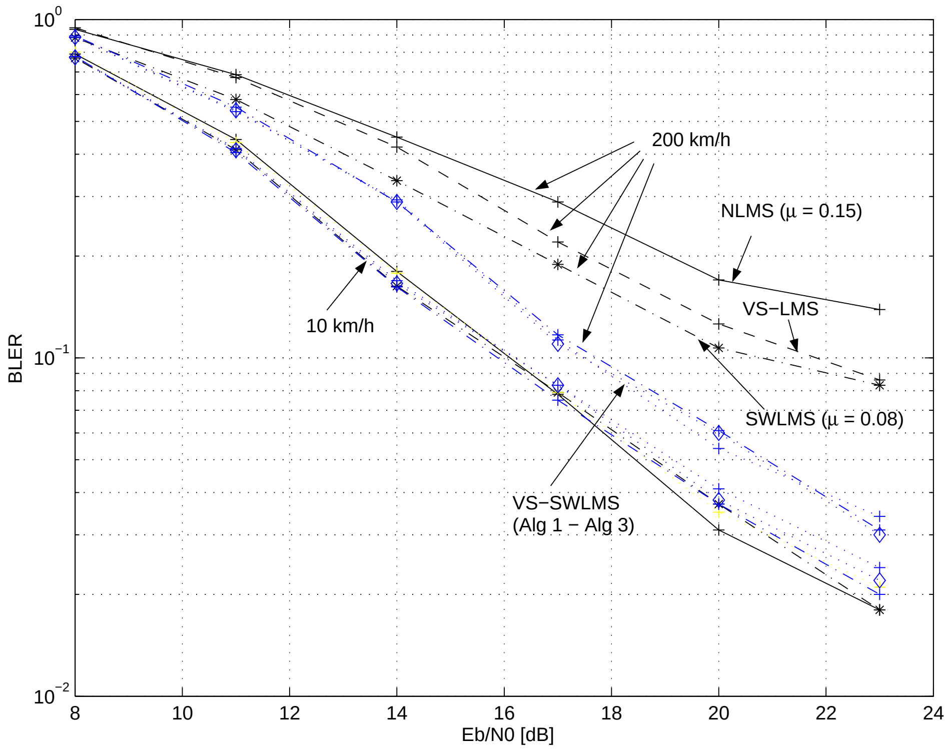

Figure 5.4: Block error rate for modulation and coding scheme nr 7 (MCS-7) and an RA channel at 1800MHz, at a mobile speed of 10 *and* 200 *km/h with frequency offset 200Hz. This clearly shows that adaptation of* h_t *improves the performance in severely fading channels.*

tuned (fixed) step-size will provide better performance than the VS-LMS algorithm. This is a result of a the hypermodel. The use of an automatic step-size scheme comes best to its right in situations where the environment may change frequently. In a communication scenarios this can be simulated by different speeds of the mobile unit. Figure 5.4 illustrates the BLER of the different algorithms for two different velocities of the mobile, 10 km/h and 200 km/h. The tuning of the algorithms are exactly the same as in the previous figure. Here it becomes obvious that a fixed step-size tracker in the adaptive equalizer severely degrades the performance of the communication system compared to the automatically tuned equalizer. For slowly moving mobiles the differences are hardly noticeable. However, when the speed of the mobile increases, the fading becomes so fast that the step-size need to be changed to a totally new value. This is also performed by the VS algorithms and explains the large difference in BLER.

5.3 Conclusions

Our case study clearly shows the superiority of the variable step-size algorithm in combination with the SWLMS algorithm. Even though a simple IRW hypermodel was used to model the fading parameter, the VS-SWLMS algorithm was shown to outperform both the NLMS algorithm and the VS-LMS algorithm. By using more informative hypermodels, such as AR2-models tuned according to the doppler spectrum of the fading parameter, the performance may be improved further. However, this will be a topic for further studies.

Chapter 6

Further work

This thesis has investigated tracking of time varying parameters. The main focus has been on investigating the possibility to equip the Simplified Wiener LMS algorithm with a suitable automatic step-size scheme. The motivation to study this has been to simplify the work of the designer in a real tracking scenario, where the choice of a proper step-size μ might be a time-consuming process if the algorithm is to be used in a time-varying environment where the noise level may change frequently. We have studied the behavior and the performance of a few known variable step-size algorithms as well as an entirely new step-size scheme based on the SWLMS structure. In this work we have focused on some of the basic features regarding the automatic tuning of the SWLMS algorithm. However, a lot of interesting research regarding this problem remains do be done before a complete picture about the behavior of the VS-SWLMS algorithm can be given. The following tasks are subject for further studies:

- Theoretical studies of convergence and stability
- The validity between the automatically tuned step-size and the optimal step-size for different prediction horizons and different number of parameters in the vector h_t.
- Reducing the complexity of the present algorithms by updating the step-size less often; not at each iteration.
- Applying the concept of automatically tuned step-size in other applications such as e.g. echo cancelation and active noise control.
- On-line tuning of the parameters d_1 and d_2 in the hypermodel.

Appendix A

The relation between the classical Wiener filter and the WLMS algorithm

Here we will illustrate the relation between the classical Wiener filter and the structure of the WLMS algorithm for the particular case of estimating a signal in noise. It will be shown that the WLMS algorithm in this case is just a different realization of the Wiener solution, in the form of a feedback loop.

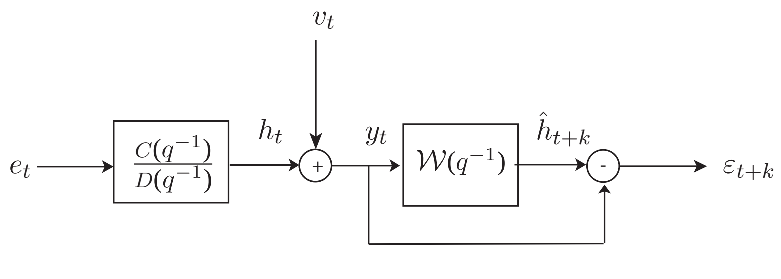

Figure A.1: The Wiener filtering problem of estimating a signal in noise. The signal y_t is produced by the signal h_t disturbed by the noise v_t. Here, the goal of the Wiener filter is to produce an estimate of the signal h_t, given the noisy signal y_t.

Consider the signals

$$\begin{aligned} h_t &= \frac{C(q^{-1})}{D(q^{-1})} e_t & \text{(A.1)} \\ y_t &= \varphi_t^* h_t + v_t = h_t + v_t \quad . & \text{(A.2)} \end{aligned}$$

Here, the signal y_t is a noisy measurement of the signal h_t, disturbed by the white noise v_t. Note also that the regressor φ_t^* present in (A.2) is set to 1 in order to obtain a direct relationship with a time-invariant Wiener filter. Given (A.1) and (A.2), it can be shown [4] that the Wiener filter $\mathcal{W}(q^{-1})$ in Figure A.1, estimates h_{t+k} based on the measurement y_t

$$\hat{h}_{t+k|t} = \mathcal{W}(q^{-1}) y_t = \frac{Q_k(q^{-1})}{\beta(q^{-1})} y_t \quad . \tag{A.3}$$

This solution to the Wiener filter is obtained by solving the spectral factorization

$$r\beta\beta_* = \gamma\, CC_* + DD_* \tag{A.4}$$

and the diophantine equation

$$q^k \gamma CC_* = rQ_k\beta_* + qDL_{k*} \quad . \tag{A.5}$$

where, γ, is the relationship between the process noise e_t and the measurement noise v_t, expressed as

$$\gamma = \frac{E\{\|e_t\|_2^2\}}{E\{\|v_t\|_2^2\}} = \frac{R_e}{R_v} \quad , \tag{A.6}$$

where R_e and R_v are the autocorrelation matrices of the process noise e_t and the measurement noise v_t, respectively. This method of calculating the optimal Wiener filter is described more in detail in [38]. Now, in order to create the connection between the WLMS and the Wiener filter we rewrite the steady-state solution (A.3) as

$$\begin{aligned} \hat{h}_{t|t} &= \frac{Q_0(q^{-1})}{\beta(q^{-1})} y_t & \text{(A.7)} \\ \hat{h}_{t+k|t} &= \frac{Q_k(q^{-1})}{Q_0(q^{-1})} \hat{h}_{t|t} & \text{(A.8)} \end{aligned}$$

where it can be shown[1] that $Q_0(z^{-1})$ has all its zeros in $|z| < 1$. We also express the connection between the spectral factor $\beta(q^{-1})$ and the step-size μ in the following way

$$\beta(q^{-1}) = \frac{(Q_0(q^{-1}) - q^{-1}(1-\mu)Q_1(q^{-1}))}{\mu} \quad . \tag{A.9}$$

[1]See Lemma 2 and Appendix B in [13]

For a derivation of (A.9), see Appendix B. Then, using the result (A.9) in (A.7) gives us

$$(Q_0(q^{-1}) - q^{-1}(1-\mu)Q_1(q^{-1}))\hat{h}_{t|t} = \mu Q_0(q^{-1})y_t \tag{A.10}$$

which can be rewritten as

$$Q_0(q^{-1})\hat{h}_{t|t} - q^{-1}(1-\mu)Q_0(q^{-1})\hat{h}_{t+1|t} = \mu Q_0(q^{-1})y_t \tag{A.11}$$

by using the relation

$$Q_0(q^{-1})\hat{h}_{t+1|t} = Q_1(q^{-1})\hat{h}_{t|t} \tag{A.12}$$

from (A.8). Now, given that the polynomial Q_0 is time-invariant we obtain

$$q^{-1}(1-\mu)Q_0(q^{-1})\hat{h}_{t+1|t} = (1-\mu)Q_0(q^{-1})\hat{h}_{t|t-1} \ . \tag{A.13}$$

Since Q_0 is stable it is possible to express the equation (A.11) as

$$\hat{h}_{t|t} = (1-\mu)\hat{h}_{t|t-1} + \mu y_t \tag{A.14}$$

Finally, by introducing $\varepsilon_t = y_t - \hat{h}_{t|t-1}$ we can express the Wiener filter as

$$\begin{aligned} \varepsilon_t &= y_t - \hat{h}_{t|t-1} && \text{(A.15)} \\ \hat{h}_{t|t} &= \hat{h}_{t|t-1} + \mu\varepsilon_t && \text{(A.16)} \\ \hat{h}_{t+k|t} &= \frac{Q_k(q^{-1})}{Q_0(q^{-1})}\hat{h}_{t|t} && \text{(A.17)} \end{aligned}$$

i.e. the WLMS structure. The WLMS structure is in this case a realization of the Wiener filter $\mathcal{W}(q^{-1})$ in a form that uses the one-step prediction error ε_t and the one-step prediction $\hat{h}_{t|t-1}$ as intermediate results in a feedback loop. The step-size μ to be used in the WLMS algorithm in order to produce the same result as the classical Wiener filter is then obtained as

$$\mu = 1 - \frac{1}{r} \ . \tag{A.18}$$

where r is given by the solution to the spectral factorization (A.4). The purpose with this example has been to illustrate the relation between the Wiener filter and the WLMS algorithm. However, what is also important to realize is that the optimal performance of the WLMS algorithm is based on the knowledge of the signal model (A.1) and the parameter γ. In Chapter 3 the Wiener filtering problem is further discussed in order to illustrate an alternative solution based on less information about the environment.

Appendix B

The relation between the step-size μ and the spectral factor $\beta(q^{-1})$

The solutions to the diophantine equation (2.57) for different time-lags k and white noise v_t can be obtained recursively from $Q_0(q^{-1})$ and $Q_1(q^{-1})$ according to

$$\begin{aligned} Q_1(q^{-1}) &= q(\beta(q^{-1}) - D(q^{-1})) && \text{(B.1)} \\ Q_{k+1}(q^{-1}) &= q(Q_k(q^{-1}) - D(q^{-1})Q_0^k(q^{-1})) && \text{(B.2)} \\ Q_0(q^{-1}) &= \beta(q^{-1}) - D(q^{-1})\frac{1}{r} \ , && \text{(B.3)} \end{aligned}$$

for details see Corollary 1 in [4]. Then, the connection between the polynomials $Q_1(q^{-1})$, $Q_{k+1}(q^{-1})$, $\beta(q^{-1})$, and the step size μ is calculated as follows. Set $k = 0$ and $Q_0^0(q^{-1}) = \mu$ in (B.2). This gives

$$Q_1(q^{-1}) = q(Q_0(q^{-1}) - D(q^{-1})\mu) \ . \tag{B.4}$$

Then, solving for $D(q^{-1})$ in (B.1) results in

$$D(q^{-1}) = \beta(q^{-1}) - q^{-1}Q_1(q^{-1}) \tag{B.5}$$

Now, by inserting (B.5) into (B.4) and solving for $\beta(q^{-1})$, we obtain (A.9),

$$\begin{aligned} Q_1(q^{-1}) &= q(Q_0(q^{-1}) - (\beta(q^{-1}) - q^{-1}Q_1(q^{-1}))\mu) && \text{(B.6)} \\ q^{-1}Q_1(q^{-1}) &= Q_0(q^{-1}) - \beta(q^{-1})\mu + q^{-1}Q_1(q^{-1})\mu && \text{(B.7)} \\ \beta(q^{-1})\mu &= Q_0(q^{-1}) - q^{-1}Q_1(q^{-1}) + q^{-1}Q_1(q^{-1})\mu && \text{(B.8)} \\ \beta(q^{-1}) &= \frac{Q_0(q^{-1}) - q^{-1}Q_1(q^{-1}) + q^{-1}Q_1(q^{-1})\mu}{\mu} \ . && \text{(B.9)} \end{aligned}$$

Appendix C

Derivation of Benvenistes VS-LMS algorithm

The following derivation is performed in a similar way as in the book [1].
Let the error function be denoted

$$J = \frac{1}{2}\mathrm{E}\,|\varepsilon_t|^2, \tag{C.1}$$

where ε_t is the estimation error,

$$\varepsilon_t = y_t - \varphi_t^* \hat{h}_{t|t-1}. \tag{C.2}$$

By differentiating the criterion function J with respect to the step-size parameter μ, the scalar gradient $\nabla_\mu(t)$ becomes

$$\nabla_\mu = \frac{\partial J}{\partial \mu} = \frac{1}{2}\mathrm{E}\left[\frac{\partial \varepsilon_t}{\partial \mu}\varepsilon_t^* + \frac{\partial \varepsilon_t^*}{\partial \mu}\varepsilon_t\right] = \mathrm{E}\left[\Re\{\varepsilon_t \frac{\partial \varepsilon_t^*}{\partial \mu}\}\right] \tag{C.3}$$

Then, the use of equation $(C.2)$ gives

$$\frac{\partial \varepsilon_t^*}{\partial \mu} = \frac{\partial}{\partial \mu}\left(y_t - \varphi_t^* \hat{h}_{t|t-1}\right)^* = -\frac{\partial}{\partial \mu}(\hat{h}_{t|t-1}^* \varphi_t) = -\psi_t^* \varphi_t \tag{C.4}$$

where

$$\psi_t \quad \triangleq \quad \frac{\partial \hat{h}_{t|t-1}}{\partial \mu}. \tag{C.5}$$

The gradient ∇_μ can now be expressed as

$$\nabla_\mu = -\mathrm{E}\left[\Re\{\varepsilon_t \psi_t^* \varphi_t\}\right] = -\mathrm{E}\left[\Re\{\psi_t^* \varphi_t \varepsilon_t\}\right] \ . \tag{C.6}$$

In order to calculate (C.6), the derivative ψ^* is needed. From (2.54) we obtain

$$\hat{h}_{t+1|t} = \boldsymbol{\mathcal{P}}_1(q^{-1})\hat{h}_{t|t} = \hat{h}_{t|t} = \hat{h}_{t|t-1} + \mu\mathbf{R}^{-1}\varphi_t\varepsilon_t \ . \tag{C.7}$$

Note that we have used the fact that $\boldsymbol{\mathcal{P}}_k(q^{-1}) = I$ for $k = 1$ in the LMS case, cf. (2.54), (2.55), (2.63), and (2.64). The derivative of (C.7) with respect to the step-size μ is

$$\begin{aligned}\frac{\partial \hat{h}_{t+1|t}}{\partial \mu} = \frac{\partial \hat{h}_{t|t}}{\partial \mu} &= \frac{\partial}{\partial \mu}(\hat{h}_{t|t-1} + \mu\mathbf{R}^{-1}\varphi_t\varepsilon_t) \\ &= \psi_t + \mathbf{R}^{-1}\varphi_t\varepsilon_t + \mu\mathbf{R}^{-1}\varphi_t\frac{\partial \varepsilon_t}{\partial \mu} \\ &= \psi_t + \mathbf{R}^{-1}\varphi_t\varepsilon_t - \mu\mathbf{R}^{-1}\varphi_t\varphi_t^*\psi_t \\ &= [I - \mu\mathbf{R}^{-1}\varphi_t\varphi_t^*]\psi_t + \mathbf{R}^{-1}\varphi_t\varepsilon_t \ . \end{aligned} \tag{C.8}$$

Equation (C.8) can thus be expressed as

$$\psi_{t+1} = [I - \mu\mathbf{R}^{-1}\varphi_t\varphi_t^*]\psi_t + \mathbf{R}^{-1}\varphi_t\varepsilon_t \ . \tag{C.9}$$

The optimal μ could be obtained by setting ∇_μ in (C.6) to zero and solving for μ. However, this is not feasible in an online situation. An iterative search of the performance surface would be more appropriate. Thus introduce a gradient search for μ as

$$\mu_{t+1} = \mu_t - \rho\hat{\nabla}_\mu(t) \ . \tag{C.10}$$

where $\hat{\nabla}_\mu(t)$ is an approximation of (C.6) at time instant t. Here, $\hat{\nabla}_\mu(t)$ is obtained by taking the instantaneous value of (C.6) while replacing μ with $\hat{\mu}_t$ in the calculation of ψ_t. We thus obtain

$$\hat{\mu}_{t+1} = \hat{\mu}_t + \rho\Re\{\hat{\psi}_t^*\varphi_t\varepsilon_t\} \tag{C.11}$$

where

$$\hat{\psi}_{t+1} = [I - \hat{\mu}_{t+1}\mathbf{R}^{-1}\varphi_t\varphi_t^*]\hat{\psi}_t + \mathbf{R}^{-1}\varphi_t\varepsilon_t \ . \tag{C.12}$$

which is obtained by replacing μ with $\hat{\mu}_{t+1}$ in (C.9).

We can now summarize the VS-LMS algorithm for the case $\mathbf{R} = \mathbf{I}$ as follows

$$\begin{aligned}
\varepsilon_t &= y_t - \varphi_t \hat{h}^*_{t|t-1} && \text{(C.13)}\\
\hat{\mu}_{t+1} &= \hat{\mu}_t + \rho \Re\{\hat{\psi}^*_t \varphi_t \varepsilon_t\} && \text{(C.14)}\\
\hat{\psi}_{t+1} &= [I - \hat{\mu}_{t+1}\varphi_t \varphi^*_t]\hat{\psi}_t + \varphi_t \varepsilon_t && \text{(C.15)}\\
\hat{h}_{t+1|t} &= \hat{h}_{t|t-1} + \hat{\mu}_{t+1}\varphi_t \varepsilon_t \quad . && \text{(C.16)}
\end{aligned}$$

Note the order of updating the equations. Since we are able to update the step-size $\hat{\mu}_{t+1}$ with the last piece of information from the error ε_t, $\hat{\mu}_{t+1}$ is used to update the derivative $\hat{\psi}_{t+1}$.

Appendix D

VS-SWLMS algorithm candidates

VSS-SWLMS:

$$\begin{aligned}
\varepsilon_t &= y_t - \varphi_t^* \hat{h}_{t|t-1} && \text{(D.1)}\\
\hat{\mu}_{t+1} &= \alpha\hat{\mu}_t + \delta|\varepsilon_t|^2 && \text{(D.2)}\\
\hat{h}_{t|t} &= \hat{h}_{t|t-1} + \hat{\mu}_{t+1}\mathbf{R}^{-1}\varphi_t\varepsilon_t && \text{(D.3)}\\
\hat{h}_{t+k|t} &= \boldsymbol{\mathcal{P}}_k(q^{-1})\hat{h}_{t|t} && \text{(D.4)}
\end{aligned}$$

MVSS-SWLMS:

$$\begin{aligned}
\varepsilon_t &= y_t - \varphi_t^* \hat{h}_{t|t-1} && \text{(D.5)}\\
p_t &= \beta p_{t-1} + (1-\beta)|\varepsilon_t\varepsilon_{t-1}^*| && \text{(D.6)}\\
\hat{\mu}_{t+1} &= \alpha\hat{\mu}_t + \delta p_t^2 && \text{(D.7)}\\
\hat{h}_{t|t} &= \hat{h}_{t|t-1} + \hat{\mu}_{t+1}\mathbf{R}^{-1}\varphi_t\varepsilon_t && \text{(D.8)}\\
\hat{h}_{t+k|t} &= \boldsymbol{\mathcal{P}}_k(q^{-1})\hat{h}_{t|t} && \text{(D.9)}\\
& && \text{(D.10)}
\end{aligned}$$

RMVSS-SWLMS:

$$\begin{aligned}
\varepsilon_t &= y_t - \varphi_t^* \hat{h}_{t|t-1} && \text{(D.11)}\\
p_t &= \beta p_{t-1} + (1-\beta)|\varepsilon_t(\varepsilon_t+\varepsilon_{t-1})^*| && \text{(D.12)}\\
\hat{\mu}_{t+1} &= \alpha\hat{\mu}_t + \delta p_t^2 && \text{(D.13)}\\
\hat{h}_{t|t} &= \hat{h}_{t|t-1} + \hat{\mu}_{t+1}\mathbf{R}^{-1}\varphi_t\varepsilon_t && \text{(D.14)}\\
\hat{h}_{t+k|t} &= \boldsymbol{\mathcal{P}}_k(q^{-1})\hat{h}_{t|t} && \text{(D.15)}
\end{aligned}$$

CDSS-SWLMS:

$$
\begin{aligned}
\varepsilon_t &= y_t - \varphi_t^* \hat{h}_{t|t-1} && \text{(D.16)}\\
\hat{\mu}_{t+1} &= \frac{\mu_0}{t} && \text{(D.17)}\\
\hat{h}_{t|t} &= \hat{h}_{t|t-1} + \hat{\mu}_{t+1} \mathbf{R}^{-1} \varphi_t \varepsilon_t && \text{(D.18)}\\
\hat{h}_{t+k|t} &= \boldsymbol{\mathcal{P}}_k(q^{-1}) \hat{h}_{t|t} && \text{(D.19)}
\end{aligned}
$$

Appendix E

Derivation of the VS-SWLMS algorithm

The VS-SWLMS algorithm is here derived in a similar way as the variable step-size LMS algorithm is derived in Appendix C. This will make it easy to compare and understand the differences between the two algorithms.

Let the error function be denoted

$$J = \frac{1}{2}\mathrm{E}\,|\varepsilon_t|^2, \tag{E.1}$$

where ε_t is the estimation error,

$$\varepsilon_t = y_t - \varphi_t^* \hat{h}_{t|t-1}. \tag{E.2}$$

By differentiating the criterion function J with respect to the step-size parameter μ, the scalar gradient $\nabla_\mu(t)$

$$\nabla_\mu = \frac{\partial J}{\partial \mu} = \frac{1}{2}\mathrm{E}\,[\frac{\partial \varepsilon_t}{\partial \mu}\varepsilon_t^* + \frac{\partial \varepsilon_t^*}{\partial \mu}\varepsilon_t] = \mathrm{E}\,[\Re\{\varepsilon_t \frac{\partial \varepsilon_t^*}{\partial \mu}\}] \tag{E.3}$$

is determined. Then, the use of equation $(E.2)$ gives

$$\frac{\partial \varepsilon_t^*}{\partial \mu} = \frac{\partial}{\partial \mu}\left(y_t - \varphi_t^* \hat{h}_{t|t-1}\right)^* = -\frac{\partial}{\partial \mu}(\hat{h}_{t|t-1}^* \varphi_t) = -\psi_t^* \varphi_t \tag{E.4}$$

where

$$\psi_t \quad \triangleq \quad \frac{\partial \hat{h}_{t|t-1}}{\partial \mu}. \tag{E.5}$$

The gradient ∇_μ can therefore be expressed as

$$\nabla_\mu = -\mathrm{E}\left[\Re\{\varepsilon_t \psi_t^* \varphi_t\}\right] = -\mathrm{E}\left[\Re\{\psi_t^* \varphi_t \varepsilon_t\}\right] \ . \tag{E.6}$$

In order to calculate (E.6), the derivative ψ_t^* is needed. From (2.54) we obtain

$$\hat{h}_{t|t-1} = \boldsymbol{\mathcal{P}}_1(q^{-1})\hat{h}_{t-1|t-1} \ , \tag{E.7}$$

where

$$\boldsymbol{\mathcal{P}}_1(q^{-1}) = \frac{Q_1(q^{-1})}{Q_0(q^{-1})} = \frac{b_0 + b_1 q^{-1}}{a_0 + a_1 q^{-1}} \ , \tag{E.8}$$

and the coefficients b_0,b_1,a_0,a_1 are obtained from (2.63) and (2.64) as

$$b_0 = \frac{-d_1}{1 + d_2(1-\mu)} \tag{E.9}$$

$$b_1 = -d_2 \tag{E.10}$$

$$a_0 = 1 \tag{E.11}$$

$$a_1 = -b_0 d_2 (1-\mu) \ . \tag{E.12}$$

The coefficients d_1, d_2 in (E.9)-(E.12) are given by the hypermodel

$$\mathcal{H}(q^{-1}) = 1/(1 + d_1 q^{-1} + d_2 q^{-1}) \ . \tag{E.13}$$

Equation (E.7) can now be rewritten as

$$\hat{h}_{t|t-1} = -a_1 \hat{h}_{t-1|t-2} + b_0 \hat{h}_{t-1|t-1} + b_1 \hat{h}_{t-2|t-2} \ . \tag{E.14}$$

The derivative of (E.14) with respect to the step-size μ is calculated as

$$\begin{aligned}
\frac{\partial \hat{h}_{t|t-1}}{\partial \mu} &= -\frac{\partial}{\partial \mu}(a_1 \hat{h}_{t-1|t-2}) + \frac{\partial}{\partial \mu}(b_0 \hat{h}_{t-1|t-1}) \\
&+ \frac{\partial}{\partial \mu}(b_1 \hat{h}_{t-2|t-2}) \\
&= -(\frac{\partial a_1}{\partial \mu}\hat{h}_{t-1|t-2} + a_1 \frac{\partial \hat{h}_{t-1|t-2}}{\partial \mu}) \\
&+ (\frac{\partial b_0}{\partial \mu}\hat{h}_{t-1|t-1} + b_0 \frac{\partial \hat{h}_{t-1|t-1}}{\partial \mu}) \\
&+ (\frac{\partial b_1}{\partial \mu}\hat{h}_{t-2|t-2} + b_1 \frac{\partial \hat{h}_{t-2|t-2}}{\partial \mu}) \ ,
\end{aligned} \tag{E.15}$$

where the derivative of the parameters a_1 and b_0 with respect to the step-size parameter is given by

$$\frac{\partial a_1}{\partial \mu} = \frac{\partial b_0}{\partial \mu} = \frac{-d_1 d_2}{[1+d_2(1-\mu)]^2} \triangleq c \quad . \tag{E.16}$$

Furthermore, from (2.53) we obtain

$$\hat{h}_{t-1|t-1} = \hat{h}_{t-1|t-2} + \mu \mathbf{R}^{-1} \varphi_{t-1} \varepsilon_{t-1} \quad , \tag{E.17}$$

which gives us

$$\begin{aligned}\frac{\partial \hat{h}_{t-1|t-1}}{\partial \mu} &= \frac{\partial}{\partial \mu}(\hat{h}_{t-1|t-2} + \mu \mathbf{R}^{-1} \varphi_{t-1} \varepsilon_{t-1}) \\ &= \psi_{t-1} + \mathbf{R}^{-1} \varphi_{t-1} \varepsilon_{t-1} + \mu \mathbf{R}^{-1} \varphi_{t-1} (-\varphi^*_{t-1} \psi_{t-1}) \ .\end{aligned} \tag{E.18}$$

In order to simplify the expression we define

$$Z_t \triangleq R^{-1} \varphi_t \varepsilon_t \tag{E.19}$$

$$\mathbf{X}_t \triangleq R^{-1} \varphi_t \varphi_t^* \ . \tag{E.20}$$

We can now express equation (E.15) as

$$\begin{aligned}\frac{\partial \hat{h}_{t|t-1}}{\partial \mu} &= -c\, \hat{h}_{t-1|t-2} - a_1 \psi_{t-1} \\ &+ c\, \hat{h}_{t-1|t-1} + b_0(\psi_{t-1} + Z_{t-1} - \mu \mathbf{X}_{t-1} \psi_{t-1}) \\ &+ 0 \qquad\qquad + b_1(\psi_{t-2} + Z_{t-2} - \mu \mathbf{X}_{t-2} \psi_{t-2}) \\ &= c\, \mu Z_{t-1} - a_1\ \psi_{t-1} \\ &+ b_0\ ((I - \mu \mathbf{X}_{t-1})\psi_{t-1} + Z_{t-1}) \\ &+ b_1\ ((I - \mu \mathbf{X}_{t-2})\psi_{t-2} + Z_{t-2}) \\ &= (b_0\ (I - \mu \mathbf{X}_{t-1}) - a_1)\psi_{t-1} \\ &+ b_1\ (I - \mu \mathbf{X}_{t-2})\psi_{t-2} \\ &+ (c\ \mu + b_0) Z_{t-1} + b_1\ Z_{t-2} \ .\end{aligned} \tag{E.21}$$

This expression can further be simplified by defining

$$P_1 = (b_0(I - \mu \mathbf{X}_{t-1}) - a_1) \tag{E.22}$$

$$P_2 = b_1(I - \mu \mathbf{X}_{t-2}) \ . \tag{E.23}$$

Equation (E.21) can then be rewritten according to

$$\psi_t = P_1 \psi_{t-1} + P_2 \psi_{t-2} + (c\mu + b_0) Z_{t-1} + b_1 Z_{t-2} \ . \tag{E.24}$$

The optimal μ could be obtained by setting ∇_μ in (E.6) to zero and solving for μ. However, this is not feasible in an online situation. An iterative search of the performance surface would be more appropriate. Thus introduce a gradient search for μ as

$$\mu_{t+1} = \mu_t - \rho\hat{\nabla}_\mu(t) \ . \tag{E.25}$$

where $\hat{\nabla}_\mu(t)$ is an approximation of (E.6) at time instant t. Here, $\hat{\nabla}_\mu(t)$ is obtained by taking the instantaneous value of (E.6) while replacing μ with $\hat{\mu}_t$ in the calculation of ψ_t. We thus obtain

$$\hat{\mu}_{t+1} = \hat{\mu}_t + \rho\Re\{\hat{\psi}_t^*\varphi_t\varepsilon_t\} \tag{E.26}$$

where

$$\hat{\psi}_t = P_1\ \hat{\psi}_{t-1} + P_2\ \hat{\psi}_{t-2} + (c\ \hat{\mu}_t + b_0)Z_{t-1} + b_1\ Z_{t-2}\ . \tag{E.27}$$

In (E.27) $\hat{c}$ is obtained by exchanging μ for $\hat{\mu}_t$ in (E.16).

We can now summarize the SWLMS algorithm with automatically updated step-size (VS-SWLMS) as follows

$$\begin{aligned}
\varepsilon_t &= y_t - \varphi_t^*\hat{h}_{t|t-1} && \text{(E.28)}\\
\hat{\mu}_{t+1} &= \hat{\mu}_t + \rho\Re\{\hat{\psi}_t^*\varphi_t\varepsilon_t\} && \text{(E.29)}\\
p &= 1/(1 + d_2(1 - \hat{\mu}_{t+1})) && \text{(E.30)}\\
\hat{b}_0 &= -d_1 p && \text{(E.31)}\\
\hat{b}_1 &= -d_2 && \text{(E.32)}\\
\hat{a}_1 &= -d_2(1 - \hat{\mu}_{t+1})\hat{b}_0 && \text{(E.33)}\\
\hat{c} &= d_2\hat{b}_0 p && \text{(E.34)}\\
Z_t &= \mathbf{R}^{-1}\varphi_t\varepsilon_t && \text{(E.35)}\\
\mathbf{X}_t &= \mathbf{R}^{-1}\varphi_t\varphi_t^* && \text{(E.36)}\\
P_1 &= (\hat{b}_0\ (\mathbf{I} - \hat{\mu}_{t+1}\mathbf{X}_t) - \hat{a}_1) && \text{(E.37)}\\
P_2 &= \hat{b}_1\ (\mathbf{I} - \hat{\mu}_{t+1}\mathbf{X}_{t-1}) && \text{(E.38)}\\
\hat{\psi}_{t+1} &= P_1\ \hat{\psi}_t + P_2\ \hat{\psi}_{t-1} + (\hat{c}\ \hat{\mu}_{t+1} + \hat{b}_0)Z_t \\
&+ \hat{b}_1\ Z_{t-1} && \text{(E.39)}\\
\hat{h}_{t|t} &= \hat{h}_{t|t-1} + \hat{\mu}_{t+1}Z_t && \text{(E.40)}\\
\hat{h}_{t+k|t} &= \boldsymbol{\mathcal{P}}_k(q^{-1})\hat{h}_{t|t} && \text{(E.41)}
\end{aligned}$$

Appendix F

Solving a Wiener filtering problem with the automatically tuned SWLMS algorithm

In order to illustrate the flexibility of automatic tuning of the step-size we shall get back to the Wiener filtering problem from Appendix A. There, we explained the relation between the Wiener filter and the WLMS algorithm. This was illustrated by verifying that the classical Wiener filter used to clean the signal y_t from the disturbing noise v_t also was obtained by the recursive WLMS algorithm. We learned that in order to calculate the optimal step-size μ for the recursive WLMS algorithm, knowledge about: the hypermodel, the process noise and the measurement noise, etc was needed. However, we will in this example show that the adaptive step-size μ_t produced by Benveniste's step-size method in combination with the SWLMS algorithm actually converges[1] into the optimal step-size predicted by the theory *without knowledge about the process noise and the measurement noise*. We will also show, by simulation, that this adaptive step-size method produces an recursive solution the diophantine equation normally needed to calculate the optimal step-size.

Recall the the signals

$$h_t = \frac{C(q^{-1})}{D(q^{-1})}e_t \tag{F.1}$$

$$y_t = h_t + v_t \ . \tag{F.2}$$

[1]Here we mean that $E\mu_t \to \mu_{opt}$ as $t \to \infty$

In this numerical example we will use $C(q^{-1}) = 0.009$, $D(q^{-1}) = [1 \ \ -2 \ 1]$, the driving noise e_t is white with variance 0.000006 and the variance of the measurement noise v_t is 0.006. Then, the solution to the spectral factorization

$$r\beta\beta_* = \gamma \, CC_* + DD_* \tag{F.3}$$

where

$$\gamma = \frac{E\{\|e_t\|_2^2\}}{E\{\|v_t\|_2^2\}} = \frac{R_e}{R_v} = 1.67 * 10^{-4} \quad , \tag{F.4}$$

is given by

$$r \quad = \quad 1.0154 \tag{F.5}$$

$$\beta \quad = \quad 1 - 1.9848q^{-1} - 0.9849q^{-2} \quad . \tag{F.6}$$

The optimal step-size for the SWLMS algorithm is then calculated to 0.0151 by the relation

$$\mu_{opt} = 1 - \frac{1}{r} \quad . \tag{F.7}$$

Now, suppose that we *do not know the variances of the process noise or the measurement noise*, this means that we cannot calculate γ, and therefore, we will not be able to solve the diophantine equation. However we circumvent this problem by applying Benveniste's step-size scheme. This will produce a recursive updating of the step-size μ that hopefully converges into the same optimal value as above. The result is illustrated in Figure F.1. Then, based on the recursive step-size above we can also present the recursive solution to the diophantine equation using the relation (A.9), i.e.,

$$\beta_t(q^{-1}) = \frac{(Q_0^t(q^{-1}) - q^{-1}(1-\mu_t)Q_1^t(q^{-1}))}{\mu_t} \quad . \tag{F.8}$$

As we see in Figure F.2 the recursive solution manages to find the correct parameter values of the polynomial β. This example is interesting in that sense the the adaptive step-size algorithm, in this case, Benveniste's, correctly estimates the parameters r and β without knowledge of the process noise or the measurement noise. In other words, the automatically tuned SWLMS algorithm is able to estimate the relationship between the process noise and the measurement noise *indirectly* by the relation between the instantaneous error ε_t and the closed form expression for the coefficient/prediction filter given in Chapter 2. This Wiener filtering problem is very trivial in the sense that no regressors are used, and we are only estimating the parameter h_t. However, we have seen through many different simulations that this method of estimating the optimal step-size works well for non-trivial tracking problems as well.

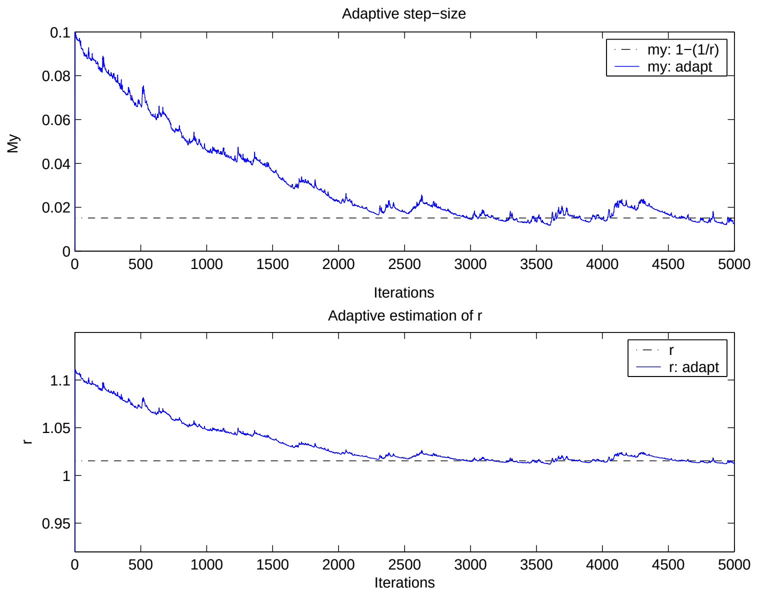

Figure F.1: The convergence of the step-size and the corresponding value of the parameter r when tracking an integrated random walk parameter disturbed by noise. We can here clearly see that the iterative solution approaches the optimal solution

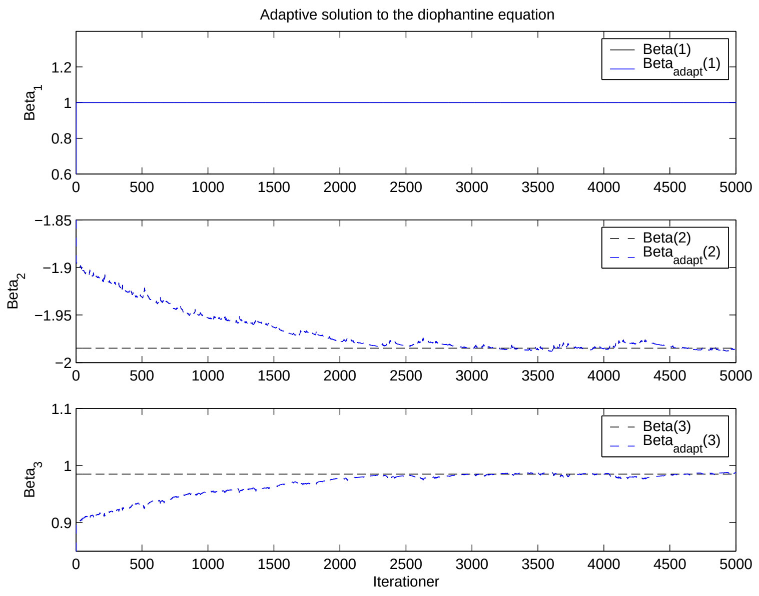

Figure F.2: Illustration of the adaptation towards the ”true” parameter values of the polynomial β.

Bibliography

[1] S. Haykin, *Adaptive Filter Theory, 3rd Edition*. New Jersey: Prentice Hall, 1996.

[2] A.H Sayed, *Fundamentals of Adaptive Filtering*. New Jersey: John Wiley and Sons, 2003.

[3] B. Widrow and S.D. Stearns, *Adaptive Signal Processing*. New Jersey: Prentice Hall, 1985.

[4] L. Lindbom. *A Wiener Filtering Approach to the Design of Tracking Algorithms*, PhD thesis, Signals and Systems Group, Uppsala University, Uppsala, Sweden, 1995.

[5] M. Métivier A. Benveniste and P. Priouret, *Adaptive Algoriths and Stochastic Approximations*. Berlin, Heidelberg: Springer-Verlag, 1990.

[6] S.C Douglas, "Generalized gradient adaptive step sizes for stochastic gradient adaptive filters," in *Proceedings of International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, vol. 2, 9-12 May 1995.

[7] W. P Ang and B. Farhang-Boroujeny, "A new class of gradient adaptive step-size LMS algorithms," *SP*, vol. 49, no. 4, pp. 805–810, Apr. 2001.

[8] P. Kuosmanen R. Bilcu and C. Rusu, "A noise constrained VS-LMS algorithm," *IEEE/AFCEA Eurocomm 2000, Information Systems for Enhanced Public Safety and Security*, pp. 29–33, 2000.

[9] Y. Grenier, "Time-dependent ARMA modeling of nonstationary signals," *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 31, pp. 899–911, 1983.

[10] W. Gersch G. Kitagawa, "A smoothness priors time-varying ar coeficient modeling of nonstationary covariance time series," *Proceedings of International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, vol. 30, pp. 48–56, 1985.

[11] M. Sternad, L. Lindbom, and A. Ahlén, "Wiener design of adaptation algorithms with time-invariant gains," *IEEE Transactions on Signal Processing*, vol. 50, no. 8, pp. 1895–1907, Aug. 2002.

[12] A. Ahlén, L. Lindbom, and M. Sternad, "Analysis of stability and performance of adaptation algorithms with time-invariant gains," *IEEE Transactions on Signal Processing*, vol. 52, pp. 103–116, Jan. 2004.

[13] L. Lindbom, M. Sternad, and A. Ahlén, "Tracking of time-varying mobile radio channels part I: The Wiener LMS algorithm," *IEEE Transactions on Communications*, vol. 49, no. 12, pp. 2207–2217, Dec. 2001.

[14] L. Lindbom, A. Ahlén, M. Sternad, and M. Falkenström, "Tracking of time-varying mobile radio channels part II: A case study," *IEEE Transactions on Communications*, vol. 50, no. 1, pp. 156–167, Jan. 2002.

[15] M. Sternad, L. Lindbom, and A. Ahlén, "Robust Wiener design of adaptation laws with constant gains," in *Proceedings in IFAC Workshop on Adaptation and Learning in Control and Signal Processing*, Como, Italy, Aug. 2001.

[16] A. Benveniste and G. Ruget, "A measure of the tracking capability of recursive stochastic algortihms with constant gains," *IEEE Transactions on Automatic Control*, vol. 27, no. 3, June 1982.

[17] A. Ahlén and M. Sternad, "Wiener filter design using polynomial equations," *IEEE Transactions on Signal Processing*, vol. 39, pp. 2387–2399, 1991.

[18] T. Aboulnasr and K. Mayyas, "A robust variable step-size LMS-type algorithm: analysis and simulations," *IEEE Transactions on Signal Processing*, vol. 45, no. 3, pp. 631–639, Mar. 1997.

[19] Ang Wee-Peng and B. Farhang-Boroujeny, "Gradient adaptive step-size LMS algorithms: past results and new developments," in *Proceedings of IEEE Adaptive Systems for Signal Processing, Communications, and Control Symposium*, 1-4 Oct. 2000.

[20] R.H. Kwong and E.W. Johnston, "A variable step size LMS algorithm,", vol. 40, no. 7, pp. 1633–1642, July 1992.

[21] V.J. Mathews and Z. Xie, “A stochastic gradient adaptive filter with gradient adaptive step size,” *IEEE Transactions on Signal Processing*, vol. 41, no. 6, pp. 2075–2087, Jun. 1993.

[22] J. Okello, Y. Itoh, Y. Fukui, I. Nakanishi, and M. Kobayashi, “A new modified variable step size for the LMS algorithm,” in *Proceedings of the 1998 IEEE International Symposium on Circuits and Systems*, vol. 5, 31 May-3 June 1998.

[23] D.I. Pazaitis and A.G. Constantinides, “A kurtosis-driven variable step-size LMS algorithm,” in *Proceedings of International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, vol. 3, 7-10 May 1996.

[24] W.Y. Chen and R.A Haddad, “A variable step size LMS algorithm,” in *Proceedings of the 33rd Midwest Symposium on Circuits and Systems*, vol. 1, 12-14 Aug. 1990.

[25] L. Ljung and T. Söderström, *Theory and practice of recursive identification*. Cambridge, MA: MIT Press, 1983.

[26] P. Sristi, W.-S. Lu, and A. Antoniou, “A new variable-step-size LMS algorithm and its application in subband adaptive filtering for echo cancellation,” in *Proceedings of IEEE International Symposium on Circuits and Systems*, vol. 2, 6-9 May 2001.

[27] Y.K Shin and J.G Lee, “A study on the fast convergence algorithm for the LMS adaptive filter design,” *KIEE*, vol. 19, no. 5, pp. 12–19, Oct. 1985.

[28] A. Sugiyama, “An interference-robust stochastic gradient algorithm with a gradient-adaptive step-size,” in *Proceedings of International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, vol. 3, 27-30 April 1993.

[29] S.B. Gelfand, W. Yongbin, and J.V. Krogmeier, “The stability of variable step-size LMS algorithms,” *IEEE Transactions on Signal Processing*, vol. 47, no. 12, pp. 3277–3288, Dec. 1999.

[30] P. Xue J.B. Evans and B. Liu, “Analyses and implemantation of variable step size adaptive algorithms,” *IEEE Transactions on Signal Processing*, vol. 41, no. 8, pp. 2517–2535, Aug. 1993.

[31] A.I. Sulyman and A. Zerguine, “Convergence and steady-state analysis of a variable step-size normalized LMS algorithm,” in *Proceedings of Seventh International Symposium on Signal Processing and Its Applications*, vol. 2, July 1-4, 2003.

[32] W.C. Jakes, *Microwave Mobile Communications*. New York: Wiley, 1974.

[33] A. Duel-Hallen and C. Heegard, “Delayed decision-feedback sequence estimation,” *IEEE Transactions on Communications*, vol. 37, pp. 428–436, May 1989.

[34] A. Furuskär, S. Mazur, F. Muller, and H. Olofsson, “EDGE: Enhanced data rates for GSM and TDMA/136 evolution,” *IEEE Transactions on Personal Communication*, vol. 6, pp. 55–66, June 1999.

[35] “GSM 05.05, radio transmission and reception, European Telecom. Standards Institute ETSI,”.

[36] S. Ariyavisitakul, J. Winters, and N. Sollenberger, “Joint equalization and interference suppression for high data rate wireless systems,” *IEEE Journal on Selected Areas in Communications*, vol. 18, pp. 1214–1220, July 2000.

[37] J. Hagenauer, “Source-controlled channel decoding,” *IEEE Transactions on Communications*, vol. 43, pp. 2449–2457, Sept 1995.

[38] A. Ahlén and M. Sternad, “Derivation and design of Wiener filters using polynomial equations,” *in Control and Dynamic Systems, C T Leondes, Academic Press, New York, NY*, vol. 64, Stochastic Techniques in Digital Signal Processing Systems, pp. 353–418, 1994.