Hindawi Publishing Corporation
EURASIP Journal on Advances in Signal Processing
Volume 2008, Article ID 459586, 13 pages
doi:10.1155/2008/459586

Research Article

Sequential and Adaptive Learning Algorithms for M-Estimation

Guang Deng

Department of Electronic Engineering, Faculty of Science, Technology and Engineering, La Trobe University, Bundoora, VIC 3086, Australia

Correspondence should be addressed to Guang Deng, d.deng@latrobe.edu.au

Received 1 October 2007; Revised 9 January 2008; Accepted 1 April 2008

Recommended by Sergios Theodoridis

The M-estimate of a linear observation model has many important engineering applications such as identifying a linear system under non-Gaussian noise. Batch algorithms based on the EM algorithm or the iterative reweighted least squares algorithm have been widely adopted. In recent years, several sequential algorithms have been proposed. In this paper, we propose a family of sequential algorithms based on the Bayesian formulation of the problem. The basic idea is that in each step we use a Gaussian approximation for the posterior and a quadratic approximation for the log-likelihood function. The maximum a posteriori (MAP) estimation leads naturally to algorithms similar to the recursive least squares (RLSs) algorithm. We discuss the quality of the estimate, issues related to the initialization and estimation of parameters, and robustness of the proposed algorithm. We then develop LMS-type algorithms by replacing the covariance matrix with a scaled identity matrix under the constraint that the determinant of the covariance matrix is preserved. We have proposed two LMS-type algorithms which are effective and low-cost replacement of RLS-type of algorithms working under Gaussian and impulsive noise, respectively. Numerical examples show that the performance of the proposed algorithms are very competitive to that of other recently published algorithms.

1. INTRODUCTION

We consider a robust estimation problem for a linear observation model:

$$y = \mathbf{x}^T\mathbf{w} + r, \tag{1}$$

where $\mathbf{w}$ is the impulse response to be estimated, $\{y, \mathbf{x}\}$ is the known training data and the noise r follows an independent and identical distribution (i.i.d.). Given a set of training data $\{y_k, \mathbf{x}_k\}_{k=1:n}$, the maximum likelihood estimation (MLE) of $\mathbf{w}$ leads to the following problem:

$$\mathbf{w}_n = \arg\min_{\mathbf{w}} \sum_{k=1}^{n} \rho(r_k), \tag{2}$$

where $\rho(r_k) = -\log p(y_k \mid \mathbf{w})$ is the negative log likelihood function. The M-estimate of a linear model can also be expressed as the above MLE problem when those well-developed penalty functions [1, 2] are regarded as generalized negative log-likelihood function. This is a robust regression problem. The solution not only is an essential data analysis tool [3, 4], but also has many practical engineering applications such as in system identification, where the noise model is heavy tailed [5].

The batch algorithms and the sequential algorithms are two basic approaches to solve the problem of (2). The batch algorithms include the EM algorithm for a family of heavy-tailed distributions [3, 4] and iterative reweighted least squares (IRLSs) algorithm for the M-estimate [2, 6]. In signal processing applications, a major disadvantage of a batch algorithm is that when a new set of training data is available the same algorithm must be run again on the whole data. A sequential algorithm, in contrast to a batch algorithm, updates the estimate as a new set of training data is received. In recent years, several sequential algorithms [7–9] have been proposed for the M-estimate of a linear model. These algorithms are based on factorizing the IRLS solution [7] and factorizing the so-called M-estimate normal equation [8, 9]. These sequential algorithms can be regarded as a generalization of recursive least squares (RLSs)

algorithm[10]. Other published works include robust LMS-type algorithms [11–13].

Bayesian learning has been a powerful tool for developing sequential learning algorithms. The problem is formulated as a maximum a posteriori (MAP) estimate problem.The basic idea is to break the sequential learning problem into two major steps [14]. In the update step, an approximate of the posterior at time $n-1$ is used to obtain the new posterior at time n. In the approximation step, this new posterior is approximated by using a particular parametric distribution family. There are many well-documented techniques such as Laplace method [15] and Fisher scoring [16]. The variational Bayesian method has also been studied [17, 18].

In a recent paper [19], we address this problem from a Bayesian perspective and develop RLS-type and LMS-type of sequential learning algorithms. The development is based on using a Laplace approximation of the posterior and solving the maximum a posteriori (MAP) estimate problem by using the MM algorithm [20]. The development of the algorithm is quite complicated. The RLS-type of algorithm is further simplified as an LMS-type algorithm by treating the covariance matrix as being fixed. This has significantly reduced the computational complexity at the cost of degraded performance.

There are two major motivations of this work which is clearly an extension of our previous work [19]. Our first motivation is to follow the same problem formulation as in [19] and to explore an alternative and simpler approach to develop sequential M-estimate algorithms. More specifically, at each iteration, we use Gaussian approximation for the likelihood and the prior. As such, we can determine a close form solution of an MAP estimate sequentially when a set of new training data is available. This MAP estimate is in the similar form as that of an RLS algorithm. Our second motivation is to extend the RLS-type algorithm to the LMS-type algorithm with an adaptive step size. It is well established that a learning algorithm with adaptive step size usually outperforms those with fixed step size in terms of faster initial learning rate and lower steady state [21]. Therefore, instead of treating the covariance as being fixed, as in our previous work, we propose to use a scaled identity matrix to approximate the covariance matrix. The approximation is subject to preserving the determinant of the covariance matrix. As such, instead of updating the covariance, the scaling factor is updated. The update of the impulse response and the scaling factor thus constitute an LMS-type algorithm with an adaptive step size. A major contribution of this work is thus the development of new sequential and adaptive learning algorithms. Another major contribution is that performance of proposed LMS-type of algorithms is very close to that of the RLS-type counterpart.

Since this work is an extension of our previous work in which a survey of related works and Bayesian sequential learning have already been briefly discussed, in this paper, for brevity purpose, we have omitted the presentation of an extensive literature survey. Interested readers can refer to [19] and references therein for more information. The rest of this paper is organized as follows. In Section 2, we present the development of the proposed algorithm including a suboptimal solution. We show that the proposed algorithm consists of an approximation step and a minimization step which lead to the update of the covariance matrix and impulse response, respectively. We also discuss the quality of the estimate, issues related to the initialization and estimation of parameters, and the relationship of the proposed algorithms with those of our previous work. In Section 3, we first develop the general LMS-type of algorithm. We then present three specific algorithms, discuss their stability conditions and parameter initiation. In Section 4, we present three numerical examples. The first one evaluates the performance of the proposed RLS-type of algorithms, while the second and the third evaluate the performance of the proposed LMS-type of algorithms under Gaussian and impulsive noise conditions, respectively. A summary of this paper is presented in Section 5.

2. DEVELOPMENT OF THE ALGORITHM

2.1. Problem formulation

From the Bayesian perspective, after receiving n sets of training data $D_n = \{y_k, \mathbf{x}_k\}|_{k=1:n,}$ the log posterior for the linear observation model (1) is given by

$$\log p(\mathbf{w} \mid D_n) = \sum_{k=1}^{n} \log p(r_k) + \log p(\mathbf{w} \mid \mathcal{H}) + c, \tag{3}$$

where $p(\mathbf{w} \mid \mathcal{H})$ is the prior before receiving any training data and $\mathcal{H}$ represents the model assumption. Throughout this paper, we use "c" to represent a constant. The MAP estimate of $\mathbf{w}$ is given by

$$\mathbf{w}_n = \arg\min_{\mathbf{w}} [-\log p(\mathbf{w} \mid D_n)]. \tag{4}$$

Since the original M-estimation problem (2) can be regarded as a maximum likelihood estimation problem, in order to apply the above Bayesian approach, in this paper we attempt to solve the following problem:

$$\mathbf{w}_n = \arg\min_{\mathbf{w}} \left[\sum_{k=1}^{n} \rho(r_k) + \frac{1}{2}\lambda \mathbf{w}^T\mathbf{w} \right]. \tag{5}$$

This is essentially a constrained MLE problem:

$$\mathbf{w}_n = \arg\min_{\mathbf{w}} \sum_{k=1}^{n} \rho(r_k), \quad \text{subject to } \frac{1}{2}\mathbf{w}^T\mathbf{w} \le d. \tag{6}$$

Using the Lagrange multiplier method, the constrained MLE problem can be recasted as (5), where λ is the Lagrange multiplier and is related to the constant d. We can see that both d and λ can be regarded as regularization parameters which are used to control the model complexity. Bayesian [22] and non-Bayesian [23] approaches have been developed to determine regularization parameters.

We can see that the constrained MLE problem is equivalent to the MAP problem when we set $\log p(r_k) = -\rho(r_k)$ and $\log p(\mathbf{w} \mid \mathcal{H}) = -(1/2)\lambda \mathbf{w}^T\mathbf{w}$. This is equivalent to regarding the penalty function as the negative log likelihood and setting a zero mean Gaussian prior for $\mathbf{w}$ with covariance matrix

$\mathbf{A}_0 = \lambda^{-1}\mathbf{I}$ where $\mathbf{I}$ is an identity matrix. Therefore, in this paper we develop a sequential M-estimation algorithm by solving an MAP problem which is equivalent to a constrained MLE problem.

Since we frequently use the three variables r_n, $\bar{e}_n$, and $\hat{e}_n$, we define them as follows: $r_n = y_n - \mathbf{x}_n^T\mathbf{w}$, $\bar{e}_n = y_n - \mathbf{x}_n^T\mathbf{w}_{n-1}$, and $\hat{e}_n = y_n - \mathbf{x}_n^T\mathbf{w}_n$, where $\mathbf{w}_{n-1}$ and $\mathbf{w}_n$ are the estimates of $\mathbf{w}$ at time $n-1$ and n, respectively. We can see that r_n is the additive noise at time n, and $\bar{e}_n$ and $\hat{e}_n$ are the modelling errors due to using $\mathbf{w}_{n-1}$ and $\mathbf{w}_n$ as the impulse response at time n, respectively.

2.2. The proposed RLS-type algorithms

To develop a sequential algorithm, we rewrite (3) as follows:

$$\log p(\mathbf{w} \mid D_n) = \log p(r_n) + \log p(\mathbf{w} \mid D_{n-1}) + c, \quad (7)$$

where the term $\log p(\mathbf{w} \mid D_{n-1})$ is the log posterior at time $(n-1)$ and is also the log prior at time n. The term $\log p(r_n) = \log p(y_n \mid \mathbf{w})$ is the log-likelihood function. The basic idea of the proposed sequential algorithm is that an approximated log posterior is formed by replacing the log prior $\log p(\mathbf{w} \mid D_{n-1})$ with its quadratic approximation. The negative of the approximated log posterior is then minimized to obtain a new estimate.

To illustrate the idea, we start our development from the beginning stage of the learning process. Since the exact prior distribution for $\mathbf{w}$ is usually unknown, we use a Gaussian distribution with zero mean $\mathbf{w}_0 = 0$ and covariance $\mathbf{A}_0 = \lambda^{-1}\mathbf{I}$ as an approximation. The negative log prior $-\log p(\mathbf{w} \mid \mathcal{H})$ is approximated by $J_0(\mathbf{w})$

$$J_0(\mathbf{w}) = \frac{1}{2}(\mathbf{w} - \mathbf{w}_0)^T\mathbf{A}_0^{-1}(\mathbf{w} - \mathbf{w}_0) + c. \quad (8)$$

When the first set of training data $D_1 = \{y_1, \mathbf{x}_1\}$ is received, the negative log likelihood is $-\log p(y_1 \mid \mathbf{w}) = \rho(r_1)$ and the negative log posterior with the approximated prior, denoted by $\mathcal{P}_1(\mathbf{w}) = -\log p(\mathbf{w} \mid D_1)$, can be written as

$$\mathcal{P}_1(\mathbf{w}) = \rho(r_1) + J_0(\mathbf{w}) + c. \quad (9)$$

This is the approximation step. In the minimization step, we determine the minimizer of $\mathcal{P}_1(\mathbf{w})$, denoted by $\mathbf{w}_1$, by solving the equation $\nabla\mathcal{P}_1(\mathbf{w}_1) = 0$.

We then determine a quadratic approximation of $\mathcal{P}_1(\mathbf{w})$ around $\mathbf{w}_1$ through the Taylor-series expansion:

$$\mathcal{P}_1(\mathbf{w}) = \mathcal{P}_1(\mathbf{w}_1) + \frac{1}{2}(\mathbf{w} - \mathbf{w}_1)^T\mathbf{A}_1^{-1}(\mathbf{w} - \mathbf{w}_1) + \cdots, \quad (10)$$

where $\mathcal{P}_1(\mathbf{w}_1)$ is a constant, $\mathbf{A}_1^{-1} = \nabla\nabla\mathcal{P}_1(\mathbf{w})\,|_{\mathbf{w}=\mathbf{w}_1}$ is the Hessian evaluated at $\mathbf{w} = \mathbf{w}_1$, and the linear term $[\nabla\mathcal{P}_1(\mathbf{w}_1)]^T(\mathbf{w} - \mathbf{w}_1)$ is zero since $\nabla\mathcal{P}_1(\mathbf{w}_1) = 0$. Ignoring higher-order terms, we have the quadratic approximation for $\mathcal{P}_1(\mathbf{w})$ as follows:

$$J_1(\mathbf{w}) = \frac{1}{2}(\mathbf{w} - \mathbf{w}_1)^T\mathbf{A}_1^{-1}(\mathbf{w} - \mathbf{w}_1) + c. \quad (11)$$

This is equivalent to using a Gaussian distribution to approximate the posterior distribution $p(\mathbf{w} \mid D_1)$ with mean $\mathbf{w}_1$ and covariance $\mathbf{A}_1$. In Bayesian learning, this is well-known technique called Laplace approximation [15]. In optimization theory [24], a local quadratic approximation of the objective function is frequently used.

When we receive the second set of training data, we form the negative log posterior, denoted $\mathcal{P}_2(\mathbf{w}) = -\log p(\mathbf{w} \mid D_2)$, by replacing $\mathcal{P}_1(\mathbf{w})$ with $J_1(\mathbf{w})$ as follows:

$$\mathcal{P}_2(\mathbf{w}) = \rho(r_2) + J_1(\mathbf{w}) + c. \quad (12)$$

The minimization step results in an optimal estimate $\mathbf{w}_2$.

Continuing this process and following the same procedure, at time n, we use a quadratic approximation for $\mathcal{P}_{n-1}(\mathbf{w})$ and form an approximation of the negative log posterior as

$$\mathcal{P}_n(\mathbf{w}) = \rho(r_n) + \frac{1}{2}(\mathbf{w} - \mathbf{w}_{n-1})^T\mathbf{A}_{n-1}^{-1}(\mathbf{w} - \mathbf{w}_{n-1}) + c, \quad (13)$$

where $\mathbf{w}_{n-1}$ is optimal estimate at time $n-1$ and is the minimizer of $\mathcal{P}_{n-1}(\mathbf{w})$. The MAP estimate at time n, denoted by $\mathbf{w}_n$, satisfies the following equation:

$$\nabla\mathcal{P}_n(\mathbf{w}_n) = -\psi(\hat{e}_n)\mathbf{x}_n + \mathbf{A}_{n-1}^{-1}(\mathbf{w}_n - \mathbf{w}_{n-1}) = 0, \quad (14)$$

where $\psi(t) = \rho'(t)$ and $\hat{e}_n = y_n - \mathbf{x}_n^T\mathbf{w}_n$. Note that, r_n in (13) is replaced by $\hat{e}_n$ in (14) because $\mathbf{w}$ is replaced by $\mathbf{w}_n$. From (14), it is easy to show that

$$\mathbf{w}_n = \mathbf{w}_{n-1} + \psi(\hat{e}_n)\mathbf{A}_{n-1}\mathbf{x}_{n.} \quad (15)$$

Since $\mathbf{w}_n$ depends on $\psi(\hat{e}_n)$, we need to determine $\hat{e}_n$. Left-multiplying (15) by $\mathbf{x}_n^T$, then using the definition of $\hat{e}_n$, we can show that

$$\hat{e}_n = \bar{e}_n - \psi(\hat{e}_n)\mathbf{x}_n^T\mathbf{A}_{n-1}\mathbf{x}_{n,} \quad (16)$$

where $\bar{e}_n = y_n - \mathbf{x}_n^T\mathbf{w}_{n-1}$. Once we have determined $\hat{e}_n$ from (16), we can calculate $\psi(\hat{e}_n)$ and substitute it into (15). We show in Appendix A that the solution of (16) has the following properties: when $\bar{e}_n = 0$, $\hat{e}_n = 0$, when $\bar{e}_n \neq 0$, $|\hat{e}_n| < |\bar{e}_n|$ and $\text{sign}(\bar{e}_n) = \text{sign}(\hat{e}_n)$.

Next, we determine a quadratic approximation for $\mathcal{P}_n(\mathbf{w})$ around $\mathbf{w}_n$. This is equivalent to approximating the posterior $p(\mathbf{w} \mid D_n)$ by a Gaussian distribution with mean $\mathbf{w}_n$ and the covariance matrix $\mathbf{A}_{n:}$

$$\begin{aligned}\mathbf{A}_n^{-1} &= \nabla\nabla\mathcal{P}_n(\mathbf{w})\,|_{\mathbf{w}=\mathbf{w}_n} \\ &= \varphi(\hat{e}_n)\mathbf{x}_n\mathbf{x}_n^T + \mathbf{A}_{n-1}^{-1},\end{aligned} \quad (17)$$

where $\varphi(t) = \rho''(t)$. Using a matrix inverse formula, we have the update of the covariance matrix for $\varphi(\hat{e}_n) > 0$ as follows:

$$\mathbf{A}_n = \mathbf{A}_{n-1} - \frac{\mathbf{A}_{n-1}\mathbf{x}_n\mathbf{x}_n^T\mathbf{A}_{n-1}}{1/\varphi(\hat{e}_n) + \mathbf{x}_n^T\mathbf{A}_{n-1}\mathbf{x}_n}. \quad (18)$$

If $\varphi(\hat{e}_n) = 0$, then we have $\mathbf{A}_n = \mathbf{A}_{n-1}$.

If there is no closed form solution for (16), then we must use a numerical algorithm [25] such as Newton's method or a

Table 1: A list of some commonly used penalty functions and their first and second derivatives, denoted by $\rho(x)$, $\psi(x) = \rho'(x)$ and $\varphi(x) = \rho''(x)$, respectively.

	$\rho(x)$	$\psi(x) = \rho'(x)$	$\varphi(x) = \rho''(x)$
L_2	$\rho(x) = \frac{x^2}{2\sigma^2}$	$\psi(x) = \frac{x}{\sigma^2}$	$\varphi(x) = \frac{1}{\sigma^2}$
Huber	$\rho(x) = \begin{cases} \frac{1}{2}\frac{x^2}{\sigma^2}, & \lvert\frac{x}{\sigma}\rvert \le \nu \\ \nu\lvert\frac{x}{\sigma}\rvert - \frac{1}{2}\nu^2, & \lvert\frac{x}{\sigma}\rvert \ge \nu \end{cases}$	$\varphi(x) = \begin{cases} \frac{x}{\sigma^2}, & \lvert\frac{x}{\sigma}\rvert \le \nu \\ \frac{\nu}{\sigma}\text{sign}(x), & \lvert\frac{x}{\sigma}\rvert \ge \nu \end{cases}$	$\varphi(x) = \begin{cases} \frac{1}{\sigma^2}, & \lvert\frac{x}{\sigma}\rvert \le \nu \\ 0, & \lvert\frac{x}{\sigma}\rvert \ge \nu \end{cases}$
Fair	$\rho(x) = \sigma^2\left[\left\lvert\frac{x}{\sigma}\right\rvert - \log\left(1 + \left\lvert\frac{x}{\sigma}\right\rvert\right)\right]$	$\psi(x) = \frac{x}{1 + \lvert x/\sigma\rvert}$	$\varphi(x) = \left(1 + \left\lvert\frac{x}{\sigma}\right\rvert\right)^{-2}$

fixed-point iteration algorithm to find a solution. This would add a significant computational cost to proposed algorithm. An alternative way is to seek a closed form solution by using a quadratic approximation of the penalty function $\rho(r_n)$ as follows:

$$\hat{\rho}(r_n) = \rho(\bar{e}_n) + \psi(\bar{e}_n)(r_n - \bar{e}_n) + \frac{1}{2}\varphi(\bar{e}_n)(r - \bar{e}_n)^2. \quad (19)$$

As such, the cost function $\mathcal{P}_n(\mathbf{w})$ is approximated by

$$\widehat{\mathcal{P}}_n(\mathbf{w}) = \hat{\rho}(r_n) + \frac{1}{2}(\mathbf{w} - \mathbf{w}_{n-1})^T \mathbf{A}_{n-1}^{-1}(\mathbf{w} - \mathbf{w}_{n-1}). \quad (20)$$

In Appendix B, we show that the optimal estimate and the update of the covariance matrix are given by

$$\mathbf{w}_n = \mathbf{w}_{n-1} + \frac{\psi(\bar{e}_n)\mathbf{A}_{n-1}\mathbf{x}_n}{1 + \varphi(\bar{e}_n)\mathbf{x}_n^T\mathbf{A}_{n-1}\mathbf{x}_n}, \quad (21)$$

$$\mathbf{A}_n = \mathbf{A}_{n-1} - \frac{\mathbf{A}_{n-1}\mathbf{x}_n\mathbf{x}_n^T\mathbf{A}_{n-1}}{1/\varphi(\bar{e}_n) + \mathbf{x}_n^T\mathbf{A}_{n-1}\mathbf{x}_n}, \quad (22)$$

respectively. Comparing (15) with (21), we can see that using the quadratic approximation for $\rho(r_n)$ results in an approximation of $\psi(\hat{e}_n)$ by $\psi(\bar{e}_n)/(1 + \varphi(\bar{e}_n)\mathbf{x}_n^T\mathbf{A}_{n-1}\mathbf{x}_n)$. Comparing (18) with (22), we can see that the only change due to the approximation is replacing $\varphi(\hat{e}_n)$ by $\varphi(\bar{e}_n)$.

In summary, the proposed sequential algorithm for a particular penalty function can be developed as follows. Suppose at time n, we have $\mathbf{w}_{n-1}$, $\mathbf{A}_{n-1}$ and the training data. We have two approaches here. If we can solve (16) for $\hat{e}_n$, then we can calculate $\mathbf{w}_n$ using (15) and update $\mathbf{A}_n$ using (18). On the other hand, if there is no close form solution for $\hat{e}_n$ or the solution is very complicated, then we can use (21) and (22).

2.3. Specific algorithms

In this section, we present three examples of the proposed algorithm using three commonly used penalty functions. These penalty functions and their first and second derivatives are listed in Table 1. These functions are shown in Figure 1. We also discuss the robustness of these algorithms. To simplify discussion, we use (21) and (22) for the algorithm development.

2.3.1. The L_2 penalty function

We can easily see that by substituting $\psi(x) = x/\sigma^2$ and $\varphi(x) = 1/\sigma^2$ into (21) and (22), we have an RLS-type of algorithm [19]:

$$\mathbf{w}_n = \mathbf{w}_{n-1} + \frac{\bar{e}_n\mathbf{A}_{n-1}\mathbf{x}_n}{\sigma^2 + \mathbf{x}_n^T\mathbf{A}_{n-1}\mathbf{x}_n}, \quad (23)$$

$$\mathbf{A}_n = \mathbf{A}_{n-1} - \frac{\mathbf{A}_{n-1}\mathbf{x}_n\mathbf{x}_n^T\mathbf{A}_{n-1}}{\sigma^2 + \mathbf{x}_n^T\mathbf{A}_{n-1}\mathbf{x}_n}. \quad (24)$$

When $\sigma^2 = 1$, this reduced to a recursive least squares algorithm [27]. One can easily see that the update of the impulse response is proportional to $|\bar{e}_n|$. As such, it is not robust against impulsive noise which leads to a large value of $|\bar{e}_n|$ and thus a large unnecessary adjustment.

We note that we have used an approximate approach to derive (23) and (24). This is only used for the simplification of the presentation. In fact, for an L_2 penalty function (23) and (24) can be directly derive from (15) and (18), respectively. The results are exactly the same as (23) and (24).

2.3.2. Huber's penalty function

By substituting the respective terms of $\varphi(\bar{e}_n)$ and $\psi(\bar{e}_n)$ into (21) and (22), we have the following:

$$\mathbf{w}_n = \begin{cases} \mathbf{w}_{n-1} + \dfrac{\bar{e}_n\mathbf{A}_{n-1}\mathbf{x}_n}{\sigma^2 + \mathbf{x}_n^T\mathbf{A}_{n-1}\mathbf{x}_n}, & |\bar{e}_n| \le \lambda_H \\ \mathbf{w}_{n-1} + \dfrac{\gamma}{\sigma}\,\text{sign}(\bar{e}_n)\mathbf{A}_{n-1}\mathbf{x}_n, & |\bar{e}_n| > \lambda_H, \end{cases} \quad (25)$$

$$\mathbf{A}_n = \begin{cases} \mathbf{A}_{n-1} - \dfrac{\mathbf{A}_{n-1}\mathbf{x}_n\mathbf{x}_n^T\mathbf{A}_{n-1}}{\sigma^2 + \mathbf{x}_n^T\mathbf{A}_{n-1}\mathbf{x}_n}, & |\bar{e}_n| \le \lambda_H \\ \mathbf{A}_{n-1}, & |\bar{e}_n| > \lambda_H, \end{cases} \quad (26)$$

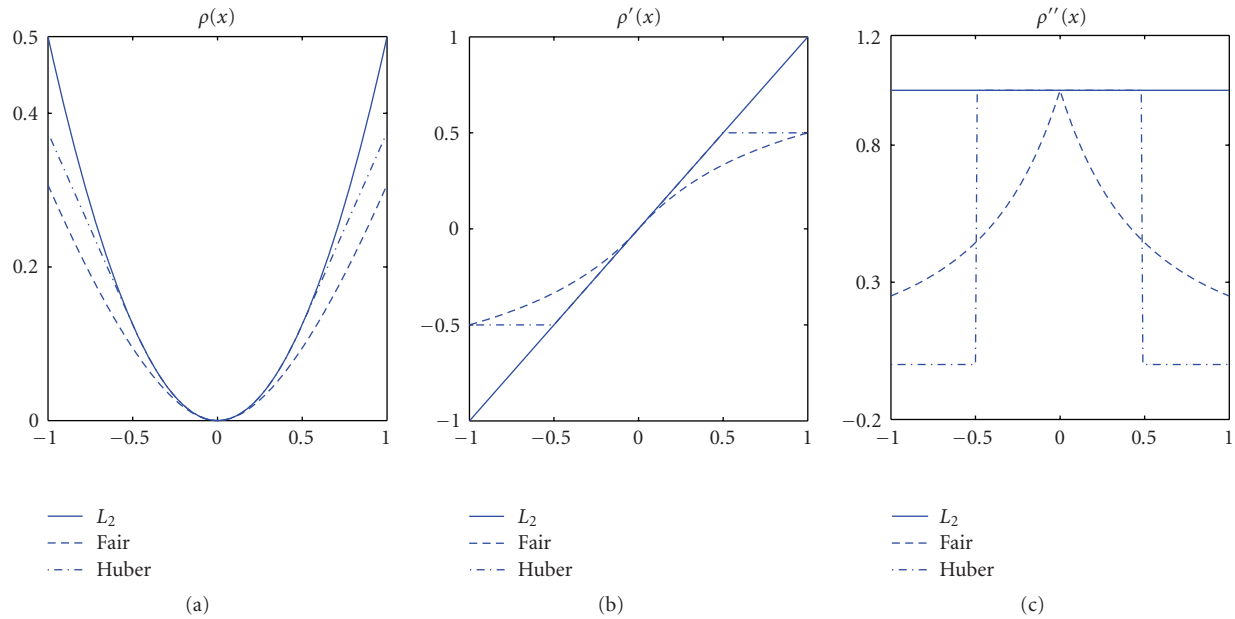

Figure 1: The three penalty functions and their first and second derivatives. We set $\sigma = 1$ and $\nu = 0.5$ when plotting these functions.

where $\lambda_H = \nu\sigma$. Comparing (25) with (23), we can see that when $|\bar{e}_n| \le \lambda_H$ they are the same. However, when $|\bar{e}_n| > \lambda_H$, indicating a possible case of outlier, (25) only uses the sign information to avoid making large misadjustment. For the update of the covariance matrix, when $|\bar{e}_n| \le \lambda_H$, it is the same as (24). However, when $|\bar{e}_n| > \lambda_H$, no update is performed.

2.3.3. The fair penalty function

We note that for the Fair penalty function, we have $\psi(\bar{e}_n) = \psi(|\bar{e}_n|)\text{sign}(\bar{e}_n)$ and $\varphi(|\bar{e}_n|) = \varphi(\bar{e}_n)$. Substituting the respective values of $\psi(\bar{e}_n)$ and $\varphi(\bar{e}_n)$ into (21) and (22), we have the following two update equations:

$$\begin{aligned}\mathbf{w}_n &= \mathbf{w}_{n-1} + \Phi(|\bar{e}_n|)\,\text{sign}(\bar{e}_n)\mathbf{A}_{n-1}\mathbf{x}_n,\\ \mathbf{A}_n &= \mathbf{A}_{n-1} - \frac{\mathbf{A}_{n-1}\mathbf{x}_n\mathbf{x}_n^T\mathbf{A}_{n-1}}{1/\varphi(|\bar{e}_n|) + \mathbf{x}_n^T\mathbf{A}_{n-1}\mathbf{x}_n},\end{aligned} \tag{27}$$

where

$$\Phi(|\bar{e}_n|) = \frac{\psi(|\bar{e}_n|)}{1 + \varphi(|\bar{e}_n|)\mathbf{x}_n^T\mathbf{A}_{n-1}\mathbf{x}_n}. \tag{28}$$

It is easy to show that for the Fair penalty function, we have

$$\Phi'(|\bar{e}_n|) = \frac{d\Phi(|\bar{e}_n|)}{d|\bar{e}_n|} > 0, \tag{29}$$

$$\lim_{|\bar{e}_n|\to\infty} \Phi(|\bar{e}_n|) = \sigma \tag{30}$$

Therefore, the value of $\Phi(|\bar{e}_n|)$ is increasing in $|\bar{e}_n|$ and is bounded by σ. As a result, the learning algorithm avoids making large misadjustment when $|\bar{e}_n|$ is large. In addition, the update for the covariance is controlled by the term $1/\varphi(|\bar{e}_n|)$ which is increasing in $|\bar{e}_n|$. Thus the amount of adjustment decreases as $|\bar{e}_n|$ increases.

2.4. Discussions

2.4.1. Properties of the estimate

Since in each step a Gaussian approximation is used for the posterior, it is an essential requirement that $\mathbf{A}_n^{-1}$ must be positive definite. We show that this requirement is indeed satisfied. Referring to (17) and using the fact that $\varphi(r_n)$ is nonnegative for the penalty functions considered [see Table 1] and that $\mathbf{A}_0^{-1}$ is positive definite, we can see that the inverse of the covariance matrix $\mathbf{A}_1^{-1} = \nabla\nabla\mathcal{P}_1(\mathbf{w})\,|_{\mathbf{w}=\mathbf{w}_1}$ is positive definite. Using mathematical induction, it is easy to prove that $\mathbf{A}_n^{-1} = \nabla\nabla\mathcal{P}_n(\mathbf{w})\,|_{\mathbf{w}=\mathbf{w}_n}$ is positive definite.

In the same way, we can prove that the Hessian of the objective function given by

$$\nabla\nabla\mathcal{P}_n(\mathbf{w}) = \varphi(r_n)\mathbf{x}_n\mathbf{x}_n^T + \mathbf{A}_{n-1}^{-1} \tag{31}$$

is also positive definite. Thus the objective function is strictly convex and the solution $\mathbf{w}_n$ is a global minimum.

Another interesting question is: does the estimate improve due to the new data $\{y_n, \mathbf{x}_n\}$? To answer this question, we can study the determinant of the precision matrix which is defined as $|\mathbf{B}_n| = |\mathbf{A}_n^{-1}|$. The basic idea is that for a univariate Gaussian, the precision is the inverse of the variance. A smaller variance is equivalent to a larger precision which implies a better estimate. From (17), we can write

$$\begin{aligned}|\mathbf{B}_n| &= |\mathbf{A}_n^{-1}|\\ &= |\varphi(\bar{e}_n)\mathbf{x}_n\mathbf{x}_n^T + \mathbf{A}_{n-1}^{-1}|\\ &= |\mathbf{B}_{n-1}|\,(1 + \varphi(\bar{e}_n)\mathbf{x}_n^T\mathbf{A}_{n-1}\mathbf{x}_n),\end{aligned} \tag{32}$$

where we have used the substitution $|\mathbf{B}_{n-1}| = |\mathbf{A}_{n-1}^{-1}|$. In deriving the above results, we have used a matrix identity:

Table 2: The update equations of three RLS-type algorithms.

Proposed	$\mathbf{w}_n = \mathbf{w}_{n-1} + \frac{\psi(\bar{e}_n)\mathbf{A}_{n-1}\mathbf{x}_n}{1+\varphi(\bar{e}_n)\mathbf{x}_n^T\mathbf{A}_{n-1}\mathbf{x}_n}$	$\mathbf{A}_n^{-1} = \mathbf{A}_{n-1}^{-1} + \varphi(\hat{e}_n)\mathbf{x}_n\mathbf{x}_n^T$
H^∞ [26]	$\mathbf{w}_n = \mathbf{w}_{n-1} + \frac{\mathbf{A}_{n-1}\mathbf{x}_n}{1+\mathbf{x}_n^T\mathbf{A}_{n-1}\mathbf{x}_n}$	$\mathbf{A}_n^{-1} = \mathbf{A}_{n-1}^{-1} + \mathbf{x}_n\mathbf{x}_n^T - \gamma_s^2\mathbf{I}$
RLS [10]	$\mathbf{w}_n = \mathbf{w}_{n-1} + \frac{\mathbf{A}_{n-1}\mathbf{x}_n}{\lambda+\mathbf{x}_n^T\mathbf{A}_{n-1}\mathbf{x}_n}$	$\mathbf{A}_n^{-1} = \lambda\mathbf{A}_{n-1}^{-1} + \mathbf{x}_n\mathbf{x}_n^T(\lambda \le 1)$

$|\mathbf{A} + \mathbf{x}\mathbf{y}^T| = |\mathbf{A}|(1 + \mathbf{y}^T\mathbf{A}^{-1}\mathbf{x})$. Since $\mathbf{x}_n^T\mathbf{A}_{n-1}\mathbf{x}_n > 0$ and $\varphi(\bar{e}_n) \ge 0$ [see Table 1], we have $|\mathbf{B}_n| \ge |\mathbf{B}_{n-1}|$. It means that the precision of the current estimate due to the new training data is better than or at least as good as that of the previous estimate. We note that when we use the update (18) for the covariance matrix, the above discussion is still valid.

2.4.2. Parameter initialization and estimation

The proposed algorithm starts with a Gaussian approximation of the prior. We can simply set the prior mean as zero $\mathbf{w}_0 = \mathbf{0}$ and set the prior covariance as $\mathbf{A}_0 = \lambda^{-1}\mathbf{I}$, where $\mathbf{I}$ is an identity matrix and λ is set to a small value to reflect the uncertainty about the true prior distribution. In our simulations, we set $\lambda = 0.01$. For the robust penalty functions listed in Table 1, σ is a scaling parameter. We propose a simple online algorithm to estimate σ as follows:

$$\sigma_n = \beta\sigma_{n-1} + (1-\beta)\min\left[3\sigma_{n-1}, |\bar{e}_n|\right], \tag{33}$$

where $\beta = 0.95$ in our simulations. The function $\min[a, b]$ takes the smaller value of the two inputs as the output. It makes the estimate of σ_n robust to outliers.

It should be noted that for a 0.95 asymptotic efficiency on the standard normal distribution, the optimal value for σ can be found in [2]. In addition, for Huber's penalty function, the additional parameter ν is set to $\nu = 2.69\sigma$ for a 0.95 asymptotic efficiency on the normal distribution [2].

2.4.3. Connection with the one-step MM algorithm [19]

Since the RLS-type of algorithm [see (21) and (22)] is derived from the same problem formulation as that in our previous work [19] and is based on different approximations, it is interesting to compare the results. For easy reference, we recall that in [19] we defined $\rho(x) = -f(t)$ where $t = x^2/2\sigma^2$. It is easy to show that

$$\psi(x) = \rho'(x) = -\frac{x}{\sigma^2}f'(t), \tag{34}$$

$$\varphi(x) = \rho''(x) = -\frac{1}{\sigma^2}\left[2tf''(t) + f'(t)\right]. \tag{35}$$

For easy reference, we reproduce (40) and (44) in [19] as follows:

$$w_n = \mathbf{w}_{n-1} + \frac{\bar{e}_n\mathbf{A}_{n-1}\mathbf{x}_n}{\bar{\tau} + \mathbf{x}_n^T\mathbf{A}_{n-1}\mathbf{x}_n}, \tag{36}$$

$$A_n = A_{n-1} - \frac{A_{n-1}x_nx_n^TA_{n-1}}{\kappa\hat{\tau} + x_n^TA_{n-1}x_n}, \tag{37}$$

where $\bar{\tau} = -\sigma^2/f'(t_n)$, $\kappa\hat{\tau} = -\sigma^2/[f'(t_n) + 2t_nf''(t_n)]$, and $t_n = \bar{e}_n^2/(2\sigma^2)$. Substituting (34) into (36), we have the RLS-type algorithm which is the one-step MM algorithm in terms of $\psi(\bar{e}_n)$ as the following:

$$\mathbf{w}_n = \mathbf{w}_{n-1} + \frac{\bar{e}_n\mathbf{A}_{n-1}\mathbf{x}_n}{\bar{e}_n/\psi(\bar{e}_n) + \mathbf{x}_n^T\mathbf{A}_{n-1}\mathbf{x}_n}, \tag{38}$$

$$\mathbf{A}_n = \mathbf{A}_{n-1} - \frac{\mathbf{A}_{n-1}\mathbf{x}_n\mathbf{x}_n^T\mathbf{A}_{n-1}}{1/\psi(\bar{e}_n) + \mathbf{x}_n^T\mathbf{A}_{n-1}\mathbf{x}_n}. \tag{39}$$

We can easily see that (39) is exactly the same as (22). To compare (38) with (21), we rewrite (21) as follows:

$$\mathbf{w}_n = \mathbf{w}_{n-1} + \frac{\bar{e}_n\mathbf{A}_{n-1}\mathbf{x}_n}{\bar{e}_n/\psi(\bar{e}_n) + (\bar{e}_n\varphi(\bar{e}_n)/\psi(\bar{e}_n))\mathbf{x}_n^T\mathbf{A}_{n-1}\mathbf{x}_n}. \tag{40}$$

It is clear that (40) has an extra term $\bar{e}_n\varphi(\bar{e}_n)/\psi(\bar{e}_n)$ compared to (38). The value of this term depends on the penalty function. For the L_2 penalty function, this term equals to one.

2.4.4. Connections with other RLS-type algorithms

We briefly comment on the connections of the proposed algorithm with that based on the H^∞ framework (see [26, Problem 2]) and the classical RLS algorithm with a forgetting factor [10]. For easy reference, the update equations for these algorithms are listed in Table 2. Comparing these algorithms, we can see that a major difference is in the way $\mathbf{A}_n^{-1}$ is updated. The robustness of the proposed algorithm is provided by the scaling factor $\varphi(\hat{e}_n)$ which controls the "amount" of update. Please refer to Figure 1 for a graphical representation of this function. For the H^∞-based algorithm, an adaptively calculated quantity $\gamma_s^2\mathbf{I}$ (see [26, equation (9)]) is subtracted from the update. This is another way of controlling the "amount" update. For the RLS algorithm, the forgetting factor plays the role of exponential-weighted sum of squared errors. The update is not controlled based on the

current modelling error. It is now clear that the term $\varphi(\hat{e}_n)$ and the term λ play a very different role in their respective algorithms.

It should be noted that by using the Bayesian approach, it is quite easy to introduce the forgetting factor into the proposed algorithm. Using the forgetting factor, the tracking performance of the proposed algorithm can be controlled. Since the development has been reported in our previous work [19], we do not discuss it in detail in this paper. A further interesting point is the interpretation of the matrix $\mathbf{A}_n$. For the L_2 penalty function, $\mathbf{A}_n$ can be called the covariance matrix. But for the Huber and fair penalty function, its interpretation is less clear. However, when we use a Gaussian distribution to approximate the posterior, we can still regard it as a covariance matrix of the Gaussian.

3. EXTENSION TO LMS-TYPE OF ALGORITHMS

3.1. General algorithm

For the RLS-type algorithms, a major contribution to the computational cost is the update of the covariance matrix. To reduce the cost, a key idea is to approximate the covariance matrix $\mathbf{A}_n$ in each iteration by $\hat{\mathbf{A}}_n = \alpha_n \mathbf{I}$, where α_n is a positive scalar and $\mathbf{I}$ is an identity matrix of suitable dimension. In this paper, we propose an approximation under the constraint of preserving the determinant, that is, $|\mathbf{A}_n| = |\hat{\mathbf{A}}_n|$. Since the determinant of the covariance matrix is an indication of the precision of the estimate, preserving the determinant thus permits passing on information about the quality of the estimate at time n to the next iteration. As such, we have $|\mathbf{A}_n| = \alpha_n^M$ where M is the length of the impulse response. The task of updating $\mathbf{A}_n$ becomes updating α_n.

From (17) and using a matrix identity $|\mathbf{A}+\mathbf{x}\mathbf{y}^T| = |\mathbf{A}|(1+\mathbf{y}^T\mathbf{A}^{-1}\mathbf{x})$, we can see that

$$\left|\mathbf{A}_n^{-1}\right| = \left|\mathbf{A}_{n-1}^{-1}\right|\left(1+\varphi(\bar{e}_n)\mathbf{x}_n^T\mathbf{A}_{n-1}\mathbf{x}_n\right). \tag{41}$$

[Here we assume that the size of the matrix $\mathbf{A}$ and the sizes of the two vectors $\mathbf{x}$ and $\mathbf{y}$ are properly defined]. Suppose, at time $n-1$,we have the approximation $\hat{\mathbf{A}}_{n-1} = \alpha_{n-1}\mathbf{I}$. Substituting this approximation into the left-hand side of (41), we have

$$\begin{aligned}\left|\mathbf{A}_n^{-1}\right| &\approx \left|\hat{\mathbf{A}}_{n-1}^{-1}\right|\left(1+\varphi(\bar{e}_n)\mathbf{x}_n^T\hat{\mathbf{A}}_{n-1}\mathbf{x}_n\right)\\ &= \alpha_{n-1}^{-M}\left(1+\alpha_{n-1}\varphi(\bar{e}_n)\mathbf{x}_n^T\mathbf{x}_n\right).\end{aligned} \tag{42}$$

Substituting $|\mathbf{A}_n^{-1}| = \alpha_n^{-M}$ into (42), we have the following:

$$\frac{1}{\alpha_n} \approx \frac{1}{\alpha_{n-1}}\left(1+\alpha_{n-1}\varphi(\bar{e}_n)\mathbf{x}_n^T\mathbf{x}_n\right)^{1/M}. \tag{43}$$

Using a further approximation $(1+x)^{1/M} \approx 1 + x/M$ to simply (43), we derive the update rule for α_n as follows:

$$\frac{1}{\alpha_n} = \frac{1}{\alpha_{n-1}} + \varphi(\bar{e}_n)\frac{\mathbf{x}_n^T\mathbf{x}_n}{M}. \tag{44}$$

Replacing $\mathbf{A}_{n-1}$ in (21) by $\alpha_{n-1}\mathbf{I}$, we have the update of the estimate

$$\mathbf{w}_n = \mathbf{w}_{n-1} + \frac{\psi(\bar{e}_n)\mathbf{x}_n}{1/\alpha_{n-1} + \varphi(\bar{e}_n)\mathbf{x}_n^T\mathbf{x}_n}. \tag{45}$$

Equations (44) and (45) can be regarded as the LMS-type of algorithm with an adaptive step size.

In [28], a stability condition for a class of LMS-type of algorithm is established as follows. The system is stable when $|\hat{e}_n| < \theta|\bar{e}_n|$ $(0 < \theta < 1)$ is satisfied. We will use this condition to discuss the stability of the proposed algorithms in Section 3.2.

We point out that in developing the above update scheme for $1/\alpha_n$, we have assumed that $\mathbf{w}$ is fixed. As such, the update rule cannot cope with a sudden change of $\mathbf{w}$ since $1/\alpha_n$ is increasing with n. This is inherent problem with the problem formulation. A systematic way to deal with it is to reformulate the problem to allow a time varying $\mathbf{w}$ by using a state space model. Another way is to detect the change of $\mathbf{w}$ and reset $1/\alpha_n$ to its default value accordingly.

3.2. Specific algorithms

Specific algorithms for the three penalty functions can be developed by substituting $\psi(\bar{e}_n)$ and $\varphi(\bar{e}_n)$ into (44) and (45). We note that the L_2 penalty function can be regarded a special case of the penalty functions used in the M-estimate. The discussion of robustness is very similar to that presented in Section 2.3 and is omitted. Details of the algorithms are described below.

3.2.1. The L_2 penalty function

Substituting $\psi(\bar{e}_n) = \bar{e}_n/\sigma^2$ and $\varphi(\bar{e}_n) = 1/\sigma^2$ into (45), we have

$$\mathbf{w}_n = \mathbf{w}_{n-1} + \frac{\bar{e}_n\mathbf{x}_n}{\mu_{n-1}+\mathbf{x}_n^T\mathbf{x}_n}, \tag{46}$$

where $\mu_{n-1} = \sigma^2/\alpha_{n-1}$. From (44), we have

$$\frac{1}{\alpha_n} = \frac{1}{\alpha_{n-1}} + \frac{\mathbf{x}_n^T\mathbf{x}_n}{\sigma^2 M}, \tag{47}$$

which can be rewritten as follows:

$$\mu_n = \mu_{n-1} + \frac{\mathbf{x}_n^T\mathbf{x}_n}{M}. \tag{48}$$

The proposed algorithm is thus given by (46) and (48). A very attractive property of this algorithm is that it has no parameters. We only need to set the initial value of μ_0 which can be set to zero (i.e., $\alpha_0 \to \infty$) reflecting our assumption that the prior distribution of $\mathbf{w}$ is flat.

The stability of this algorithm can be established by noting that

$$\hat{e}_n = \frac{\mu_{n-1}}{\mu_{n-1}+\mathbf{x}_n^T\mathbf{x}_n}\bar{e}_n. \tag{49}$$

Since $0 < \mu_{n-1}/(\mu_{n-1}+\mathbf{x}_n^T\mathbf{x}_n) < 1$ when $\mathbf{x}_n^T\mathbf{x}_n \neq 0$, the stability condition is satisfied.

3.2.2. Huber's penalty function

In a similar way, we obtain the update for $\mathbf{w}_n$ and μ_n as follows:

$$\mathbf{w}_n = \begin{cases} \mathbf{w}_{n-1} + \dfrac{\overline{e}_n \mathbf{x}_n}{\mu_{n-1} + \mathbf{x}_n^T \mathbf{x}_n}, & |\overline{e}_n| \le \lambda_H \\ \mathbf{w}_{n-1} + \dfrac{\nu\sigma}{\mu_{n-1}} \operatorname{sign}(\overline{e}_n)\mathbf{x}_n, & |\overline{e}_n| > \lambda_H, \end{cases} \tag{50}$$

$$\mu_n = \begin{cases} \mu_{n-1} + \mathbf{x}_n^T \mathbf{x}_n / M, & |\overline{e}_n| \le \lambda_H \\ \mu_{n-1}, & |\overline{e}_n| > \lambda_H \end{cases} \tag{51}$$

where $\lambda_H = \nu\sigma$. The stability of the algorithm can be established by noting that when $|\overline{e}_n| \le \lambda_H$, we have

$$\widehat{e}_n = \frac{\mu_{n-1}}{\mu_{n-1} + \mathbf{x}_n^T \mathbf{x}_n} \overline{e}_n. \tag{52}$$

which is the same as the L_2 case. One the other hand, when $|\overline{e}_n| > \lambda_H$, we can easily show that $\operatorname{sign}(\widehat{e}_n) = \operatorname{sign}(\overline{e}_n)$. As such, from (50) we have for $\overline{e}_n \neq 0$

$$\begin{aligned} \widehat{e}_n &= \overline{e}_n - \frac{\nu\sigma}{\mu_{n-1}} \operatorname{sign}(\overline{e}_n)\mathbf{x}_n^T \mathbf{x}_n \\ &= \overline{e}_n \left(1 - \frac{\nu\sigma}{\mu_{n-1}|\overline{e}_n|} \mathbf{x}_n^T \mathbf{x}_n\right). \end{aligned} \tag{53}$$

Since $\operatorname{sign}(\widehat{e}_n) = \operatorname{sign}(\overline{e}_n)$, we have $0 \le 1 - (\nu\sigma/\mu_{n-1}|\overline{e}_n|)\mathbf{x}_n^T \mathbf{x}_n < 1$. Thus the stability condition is also satisfied.

3.2.3. The fair penalty function

For the Fair penalty function, we define $\phi(t) = 1 + |t|/\sigma$. We have $\psi(t) = t/\phi(t)$ and $\varphi(t) = 1/\phi^2(t)$. Using (45), we can write

$$\mathbf{w}_n = \mathbf{w}_{n-1} + \frac{\overline{e}_n \mathbf{x}_n}{k_F}, \tag{54}$$

where $k_F = \phi(\overline{e}_n)/\alpha_{n-1} + \mathbf{x}_n^T \mathbf{x}_n / \phi(\overline{e}_n)$. The update for the precision is given by

$$\frac{1}{\alpha_n} = \frac{1}{\alpha_{n-1}} + \frac{1}{\phi^2(\overline{e}_n)} \frac{\mathbf{x}_n^T \mathbf{x}_n}{M}. \tag{55}$$

A potential problem is that the algorithm may be unstable in that the stability condition $|\widehat{e}_n| < \theta|\overline{e}_n|$ may not be satisfied. This is because

$$|\widehat{e}_n| = \delta_F |\overline{e}_n|, \tag{56}$$

where $\delta_F = |1 - \mathbf{x}_n^T \mathbf{x}_n / k_F|$. We can easily see that when $\mathbf{x}_n^T \mathbf{x}_n > 2k_F$, we have $\delta_F > 1$ which leads to an unstable system.

To solve the potential instability problem, we propose to replace k_F in (54) by k which is defined as

$$k = \begin{cases} k_F, & k_F > \dfrac{1}{2}\mathbf{x}_n^T \mathbf{x}_n \\ k_G, & \text{otherwise}, \end{cases} \tag{57}$$

where $k_G = 1/\alpha_{n-1} + \mathbf{x}_n^T \mathbf{x}_n$. We note that k_G can be regarded as a special case of k_F when $\phi(\overline{e}_n) = 1$. When $k = k_G$, we can show that $\delta_F = |1 - \mathbf{x}_n^T \mathbf{x}_n / k_G| < 1$. As a result, the system is stable. On the other hand, when $k = k_F$ (implying $k_F > (1/2)\mathbf{x}_n^T \mathbf{x}_n$), we can show that $\delta_F = |1 - \mathbf{x}_n^T \mathbf{x}_n / k_F| < 1$ which also leads to a stable system.

3.3. Initialization and estimation of parameters

In actual implementation, we can set $\mu_0 = 0$ which corresponds to setting $\alpha_0 \to \infty$. In the Bayesian perspective, this sets a uniform prior for $\mathbf{w}$, which represents the uncertainty about $\mathbf{w}$ before receiving any training data. To enhance the learning speed of this algorithm, we shrink the value of μ_n in the first N iterations, that is, $\mu_n = \beta(\mu_{n-1} + (1/\phi^2(\overline{e}_n))(\mathbf{x}_n^T \mathbf{x}_n / M))$, where $0 < \beta < 1$. An intuitive justification is that μ_n is an approximation of the precision of the estimate. In the L_2 penalty function case, μ_n is scaled by the unknown but assumed constant noise variance. Due to the nature of the approximation that ignores the higher order terms, the precision is overly estimated. A natural idea is to scale the estimated precision μ_n. In simulations, we find that $\beta = 0.9$ and $N = 8M$ lead to improved learning speed.

For the Huber and the fair penalty functions, it is necessary to estimate the scaling parameter σ. We use a simple online algorithm to estimate σ as follows:

$$\sigma_n = \gamma\sigma_{n-1} + (1 - \gamma)|\overline{e}_n|, \tag{58}$$

where $\gamma = 0.95$ in our simulations. In addition, for Huber's penalty function, the additional parameter ν is set to $\nu = 2.69\sigma$ for a 0.95 asymptotic efficiency on the normal distribution [2].

4. NUMERICAL EXAMPLES

4.1. General simulation setup

To use the proposed algorithms to identify the linear observation model of (1), at the nth iteration we generate a zero mean Gaussian random vector $\mathbf{x}_n$ of size $(M \times 1)$ as the input vector. The variance of this random vector is 1. We then generate the noise and calculate the output of the system y_n. The performance of an algorithm is measured by $h(n) = \|\mathbf{w} - \mathbf{w}_n\|_2^2$ which is a function of n and is called the learning curve. Each learning curve is the result of averaging 50-run of the program using the same additive noise. The purpose is to average out possible effect of the random input vector $\mathbf{x}_n$. The result is then plotted in the log scale, that is, $10\log_{10}[\overline{h}(n)]$, where $\overline{h}(n)$ is the averaged learning curve.

4.2. Performance of the proposed RLS algorithms

We set up the following simulation experiments. The impulse response to be identified is given by $\mathbf{w} = [0.1, 0.2, 0.3, 0.4, 0.5, 0.4, 0.3, 0.2, 0.1]^T$. In the nth iteration, a random input signal vector $\mathbf{x}_n$ is generated as $\mathbf{x}_n = \text{randn}(9, 1)$ and y_n is calculated using (1). The noise r_n is generated from a mixture of two zero mean Gaussian distributions which

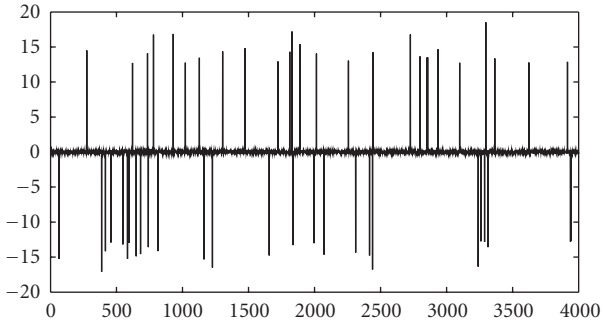

Figure 2: Noise signal used in simulations.

is simulated in Matlab by: $rn = 0.1^{*}\texttt{randn}(4000,1) + 5^{*}\texttt{randn}(4000,1).^{*}(\texttt{abs}(\texttt{randn}(4000,1) > T))$. The threshold T controls the percentage of impulsive noise. In our experiments, we set $T = 2.5$ which correspond to about 1.2% of impulsive noise. A typical case for the noise used in our simulation is shown in Figure 2

Since the proposed algorithms using Huber and fair penalty functions are similar to the RLS algorithm, we compare their learning performance with that of the RLS and a recently published RLM algorithm [8] using suggested values of parameters. Simulation results are shown in Figure 3. We observe from simulation results that the learning curves of proposed algorithms are very close to that of the RLM algorithm and are significantly better than that of the RLS algorithm which is not robust to non-Gaussian noise. The performance of the proposed algorithm in this paper is also very closed to that of our previous work [19] and the comparison results are not presented for brevity.

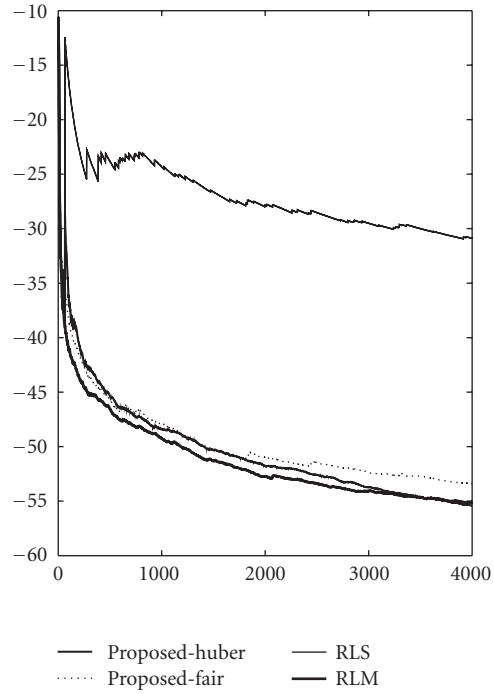

Figure 3: A comparison of learning curves for different RLS-type algorithms.

4.3. Performance of proposed LMS type of algorithms

We first compare the performance of our proposed LMS-type of algorithms using the fair and Huber penalty functions to a recently published robust LMS algorithm (called the CAF algorithm in this paper) using the suggested settings of parameters [13]. The CAF algorithm adaptively combines the NLMS and the signed NLMS algorithms. As a bench mark, we also include simulation results using the RLM algorithm which is computationally more demanding than any LMS type of algorithms. The noise used is similar to that described in Section 4.2. We have tested these algorithms with three different length of impulse responses $M = 10, 100, 512$. In each simulation, the impulse response is generated as a zero-mean Gaussian random $(M \times 1)$ vector with standard deviation of 1. Simulation results are shown in Figure 4.

From this figure, we can see that the performance of the two proposed algorithms is consistently better than that of the CAF algorithm. The performance of the proposed algorithm with the fair penalty function is also better than that with the Huber penalty function. When the length of the impulse response is moderate, the performance of the proposed algorithm with the fair penalty function is very close to that of the RLM algorithm. The latter has a notable faster learning rate than the former when the length is 512. Therefore, the proposed algorithm with the fair penalty function can be a low computational-cost replacement of the RLM algorithm for identifying an unknown linear system with moderate length.

We now compare the performance of the proposed LMS-type algorithm using the L_2 penalty function with a recently published NLMS algorithm with adaptive parameter estimation [21]. This algorithm (see [21, equation (10)]) is called the VSS-NLMS algorithm in this paper. The VSS-NLMS algorithm is chosen because its performance has been compared to many other LMS-type of algorithms with variable step sizes. We tune the parameter of the VSS-NLMS algorithm such that it reach the lowest possible steady state in each case. As a bench mark, we also include simulation results using the RLS algorithm. We have tested these algorithms with three different length of impulse responses $M = 10, 100, 512$. In each simulation, the impulse response is generated as a zero mean Gaussian random $(M \times 1)$ vector with standard deviation of 1. We have also tested settings with three different noise variances $\sigma_r = 0.1, 0.5$ and 1. We have obtained similar results for all three cases. In Figure 5, we present the steady state and the transient responses for these algorithms under the condition $\sigma_r = 0.5$. We can see that the performance of the proposed algorithm is very close to that of the RLS algorithm for the two cases $M = 10$ and $M = 100$. In fact, these two algorithms converge to almost the same steady state and the learning rate of the RLS algorithm is slightly faster. For the case of $M = 512$, the RLS algorithm, being a lot more computational demanding, has a faster learning rate in the transient response than the

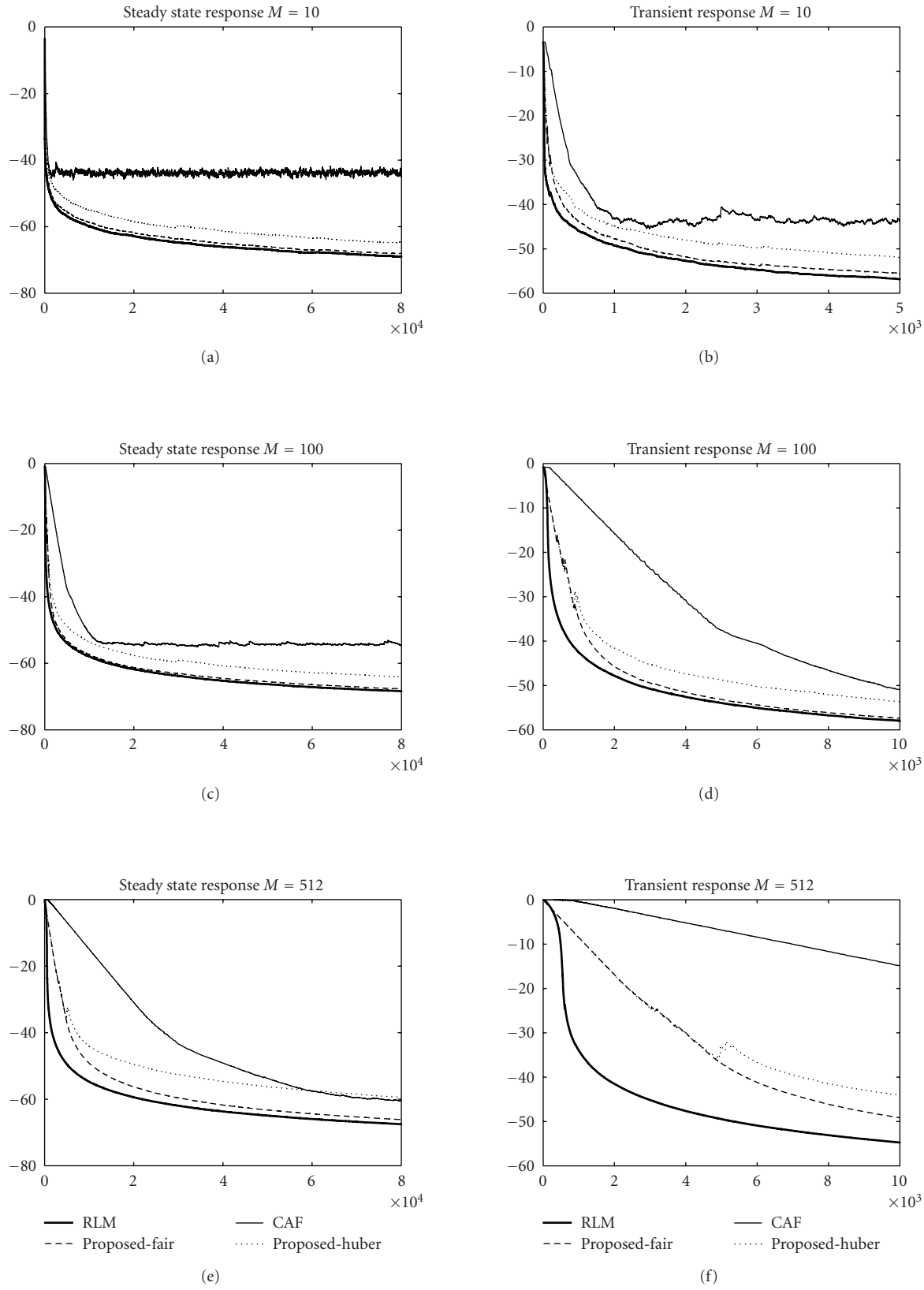

FIGURE 4: A comparison of the learning performance of different algorithms in terms of the transient response (right panel of the figure) and the steady state (left panel of the figure). Subfigures presented from top to bottom are results of testing different length of impulse response $M = 10, 100, 512$. Legends for all subfigures are the same and are included only in the top-right sub-figure.

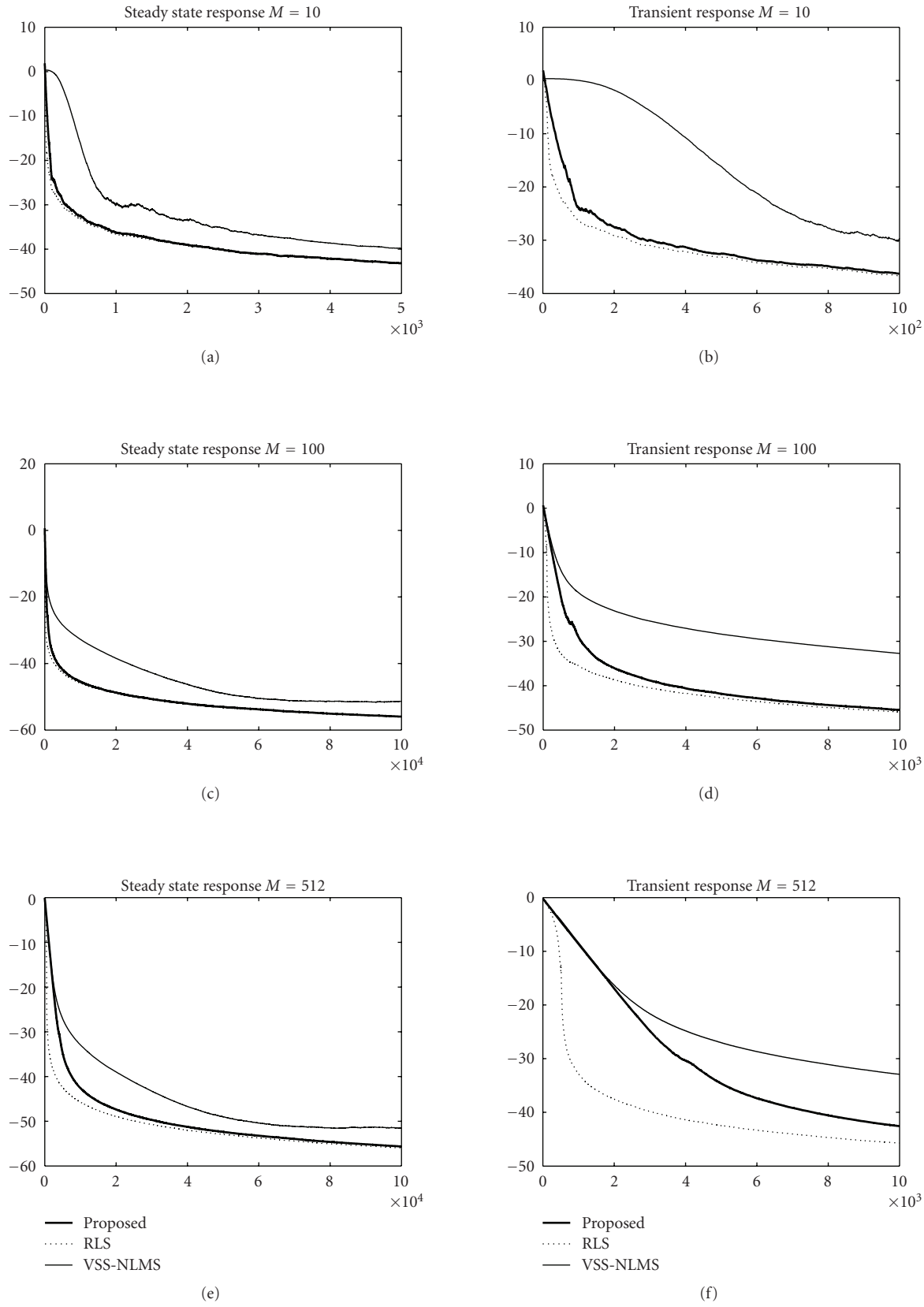

FIGURE 5: A comparison of the learning performance of different algorithms in terms of the transient response (right panel of the figure) and the steady state (left panel of the figure). Subfigures presented from top to bottom are results of testing different length of impulse response $M = 10, 100, 512$. Legends for all subfigures are the same and are included only in the top-right subfigure. We note that for the two cases $M = 10$ and 100, the proposed algorithm converges to the almost the same level of steady state as that of the RLS algorithm.

proposed algorithm does. Comparing with the VSS-NLMS algorithm, the performance of the proposed algorithm is consistently better. Therefore, the proposed algorithm can be a low computational-cost replacement for the RLS algorithm for learning an unknown linear system of moderate length.

5. CONCLUSION

In this paper, we develop a general sequential algorithm for the M-estimate of a linear observation model. Our development is based on formulating the problem from a Bayesian perspective and using a Gaussian approximation for the posterior and likelihood function in each learning step. The sequential algorithm is then developed by determining a maximum a posteriori (MAP) estimate when a new set of training data is received. The Gaussian approximation leads naturally to a quadratic objective function and the MAP estimate is an RLS-type algorithm. We have discussed the quality of the estimate, issues related to the initialization and estimation of parameters, and the relationship of the proposed algorithm with those of previous work. Motivated by reducing computational cost of the RLS-type algorithm, we develop a family of LMS-type algorithms by replacing the covariance matrix with a scaled identity matrix. Instead of updating the covariance matrix, we update the scalar which is set to preserve the determinant of the covariance matrix. Simulation results show that the learning performance of the proposed algorithms is competitive to that of some recently published algorithms. In particular, the performance of proposed LMS-type algorithms has been shown to be very close to that of their respective RLS-type algorithms. Thus they can be replacements for RLS-type of algorithms at a relatively low computational cost.

APPENDICES

A. PROPERTIES OF $\hat{e}$

Let us consider the solution to the following equation:

$$x = a - b\psi(x). \tag{A.1}$$

Comparing it to (16), we can see that $x = \hat{e}_n$, $a = \bar{e}_n$, $b = \mathbf{x}_n^T \mathbf{A}_{n-1}\mathbf{x}_n$ $(b > 0)$ and $\psi(x) = \rho'(x)$. We note that for the penalty functions $\rho(x)$ used for M-estimation, we have the following: $\psi(-x) = -\psi(x)$, $\psi(0) = 0$ and $\psi(|x|) \geq 0$. Let x_0 be a solution of (A.1).We can easily see that when $a = 0$ the solution is $x_0 = 0$. When $a \neq 0$, we can rewrite (A.1) as follows:

$$|x_0| = \frac{\text{sign}(a)}{\text{sign}(x_0)}|a| - b\psi(|x_0|). \tag{A.2}$$

The solution x_0 must satisfy two conditions: $\text{sign}(a) = \text{sign}(x_0)$ and $|a| > b\psi(|x_0|)$. These two conditions imply that $|x_0| < |a|$ which is same as $|\hat{e}_n| < |\bar{e}_n|$.

B. DERIVATION OF EQUATIONS (21) AND (22)

Substituting (19) into (20) and taking the first derivative, we have

$$\nabla\widehat{\mathcal{P}}_n(\mathbf{w}) = \left[\psi(\bar{e}_n) + \varphi(\bar{e}_n)(r_n - \bar{e}_n)\right]\mathbf{x}_n + \mathbf{A}_{n-1}^{-1}(\mathbf{w} - \mathbf{w}_{n-1}). \tag{B.1}$$

The update for $\mathbf{w}_n$ is then determined by solving $\nabla\widehat{\mathcal{P}}_n(\mathbf{w}) = 0$ as follows:

$$\mathbf{w}_n = \mathbf{w}_{n-1} + \left[\psi(\bar{e}_n) + \varphi(\bar{e}_n)(\hat{e}_n - \bar{e}_n)\right]\mathbf{A}_{n-1}\mathbf{x}_n, \tag{B.2}$$

where we have replaced $r_n = y_n - \mathbf{x}_n^T\mathbf{w}$ by $\hat{e}_n = y_n - \mathbf{x}_n^T\mathbf{w}_n$. Left multiplying both sides of the above equation by $\mathbf{x}_n^T$, then subtracting both sides by y_n, we obtain

$$\hat{e}_n = \bar{e}_n - \frac{\psi(\bar{e}_n)\mathbf{x}_n^T\mathbf{A}_{n-1}\mathbf{x}_n}{1 + \varphi(\bar{e}_n)\mathbf{x}_n^T\mathbf{A}_{n-1}\mathbf{x}_n}. \tag{B.3}$$

Substitute $\hat{e}_n$ into (B.2), we have the update for $\mathbf{w}_n$ given by (21). The update of the covariance matrix $\mathbf{A}_n$ given by (22) can be determined by using $\mathbf{A}_n^{-1} = \nabla\nabla\widehat{\mathcal{P}}_n(\mathbf{w})\,|_{\mathbf{w}=\mathbf{w}_n}$, where $\nabla\nabla\widehat{\mathcal{P}}_n(\mathbf{w})$ is given by

$$\nabla\nabla\widehat{\mathcal{P}}_n(\mathbf{w}) = \varphi(\bar{e}_n)\mathbf{x}_n\mathbf{x}_n^T + \mathbf{A}_{n-1.}^{-1} \tag{B.4}$$

REFERENCES

[1] P. J. Huber, *Robust Statistics*, John Wiley & Sons, New York, NY, USA, 1981.

[2] W. J. J. Rey, *Introduction to Robust and Quasi-Robust Statistical Methods*, Springer, Berlin, Germany, 1983.

[3] K. Lange and J. S. Sinsheimer, "Normal/independent distributions and their applications in robust regression," *Journal of Computational and Graphical Statistics*, vol. 2, no. 2, pp. 175–198, 1993.

[4] A. Gelman, H. B. Carlin, H. S. Stern, and D. B. Rubin, *Bayesian Data Analysis*, Chapman & Hall/CRC, Boca Raton, Fla, USA, 2004.

[5] S. A. Kassam and H. V. Poor, "Robust techniques for signal proecssing: a survey," *Proceedings of the IEEE*, vol. 73, no. 3, pp. 433–481, 1985.

[6] P. Petrus, "Robust Huber adaptive filter," *IEEE Transactions on Signal Processing*, vol. 47, no. 4, pp. 1129–1133, 1999.

[7] K. L. Boyer, M. J. Mirza, and G. Ganguly, "Robust sequential estimator: a general approach and its application to surface organization in range data," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 16, no. 10, pp. 987–1001, 1994.

[8] S.-C. Chan and Y.-X. Zou, "A recursive least M-estimate algorithm for robust adaptive filtering in impulsive noise: fast algorithm and convergence performance analysis," *IEEE Transactions on Signal Processing*, vol. 52, no. 4, pp. 975–991, 2004.

[9] D. S. Pham and A. M. Zoubir, "A sequential algorithm for robust parameter estimation," *IEEE Signal Processing Letters*, vol. 12, no. 1, pp. 21–24, 2005.

[10] S. Haykin, *Adaptive Filter Theory*, Prentice-Hall, Englewood Cliffs, NJ, USA, 4th edition, 2002.

[11] W. A. Sethares, "The least mean square family," in *Adaptive System Identification and Signal Processing Algorithms*, N. Kalouptsidis and S. Theodoridis, Eds., pp. 84–122, Prentice-Hall, Englewood Cliffs, NJ, USA, 1993.

[12] J. Chambers and A. Avlonitis, "A robust mixed-norm adaptive filter algorithm," *IEEE Signal Processing Letters*, vol. 4, no. 2, pp. 46–48, 1997.
[13] J. Arenas-García and A. R. Figueiras-Vidal, "Adaptive combination of normalised filters for robust system identification," *Electronics Letters*, vol. 41, no. 15, pp. 874–875, 2005.
[14] M. Opper, "A Bayesian approach to online learning," in *Online Learning in Neural Networks*, D. Saad, Ed., pp. 363–378, Cambridge University Press, Cambridge, UK, 1998.
[15] D. J. C. MacKay, *Information Theory, Inference and Learning Algorithms*, Cambridge University Press, Cambridge, UK, 2003.
[16] T. Briegel and V. Tresp, "Robust neural network regression for offline and online learning," in *Advances in Neural Information Processing Systems 12*, T. K. Leen, K.-R. Muller, and S. A. Solla, Eds., pp. 407–413, MIT Press, Cambridge, Mass, USA, 2000.
[17] Z. Ghahramani and M. J. Beal, "Propagation algorithms for variational Bayesian learning," in *Advances in Neural Information Processing Systems*, T. K. Leen, T. Dietterich, and V. Tresp, Eds., vol. 13, pp. 507–513, MIT Press, Cambridge, Mass, USA, 2001.
[18] A. Honkela and H. Valpola, "Online variational Bayesian learning," in *Proceedings of the 4th International Symposium on Independent Component Analysis and Blind Signal Separation (ICA '03)*, pp. 803–808, Nara, Japan, April 2003.
[19] G. Deng, "Robust sequential learning algorithms for linear observation models," *IEEE Transactions on Signal Processing*, vol. 55, no. 6, pp. 2472–2485, 2007.
[20] D. R. Hunter and K. Lange, "A tutorial on MM algorithms," *American Statistician*, vol. 58, no. 1, pp. 30–37, 2004.
[21] H.-C. Shin, A. H. Sayed, and W.-J. Song, "Variable step-size nlms and affine projection algorithms," *IEEE Signal Processing Letters*, vol. 11, no. 2, pp. 132–135, 2004.
[22] D. J. C. Mackay, "Bayesian interpolation," *Neural Computation*, vol. 4, no. 3, pp. 415–447, 1992.
[23] M. J. L. Orr, "Recent advances in radial basis function networks," 1999, http://www.anc.ed.ac.uk/#mjo/papers/recad.ps.gz.
[24] S. Boyd and L. Vandenberghe, *Convex Optimization*, Cambridge University Press, Cambridge, UK, 2004.
[25] M. T. Heath, *Scientific Computing: An Introductory Survey*, McGraw-Hill, New York, NY, USA, 2nd edition, 2002.
[26] B. Hassibi and T. Kailath, "Adaptive filtering with an h-infinity criterion," in *Proceedings of the 28th Asilomar Conference on Signals, Systems and Computers (ACSSC '94*, pp. 1483–1487, Pacific Grove, Calif, USA, October-November 1994.
[27] S. M. Kay, *Fundamentals of Statistical Signal Processing Estimation Theory*, Prentice-Hall, Englewood Cliffs, NJ, USA, 1993.
[28] S. C. Douglas and M. Rupp, "A posteriori update for adaptive filters," in *Proceedings of the 31st Asilomar Conference on Signals, Systems and Computers (ACSSC '97)*, vol. 2, pp. 1641–1645, Pacific Grove, Calif, USA, November 1997.