PROBABILISTIC KERNEL LEAST MEAN SQUARES ALGORITHMS

Il Memming Park[⋆], *Sohan Seth*[†], *Steven Van Vaerenbergh*[*]

[⋆] Institute for Neuroscience, University of Texas at Austin, USA
[†] Helsinki Institute for Information Technology HIIT, Aalto University, Finland
[*] Department of Communications Engineering, University of Cantabria, Spain

ABSTRACT

The kernel least mean squares (KLMS) algorithm is a computationally efficient nonlinear adaptive filtering method that "kernelizes" the celebrated (linear) least mean squares algorithm. We demonstrate that the least mean squares algorithm is closely related to the Kalman filtering, and thus, the KLMS can be interpreted as an approximate Bayesian filtering method. This allows us to systematically develop extensions of the KLMS by modifying the underlying state-space and observation models. The resulting extensions introduce many desirable properties such as "forgetting", and the ability to learn from discrete data, while retaining the computational simplicity and time complexity of the original algorithm.

Index Terms— kernel adaptive filtering, KLMS, sequential Bayesian learning, state-space model

1. INTRODUCTION

Adaptive filtering algorithms deal with real-time learning scenarios, in which the environment is often nonstationary. In general, these algorithms need to fulfill three basic requirements: 1) to sequentially learn from each observation; 2) to be adaptive to changing environments; and 3) to be computationally efficient. Among many existing algorithms that fulfill these requirements, one that has stood the test of time is the celebrated least mean squares (LMS) algorithm. This algorithm has several interesting properties, in particular its inherent computational simplicity, and its implicit tracking ability despite its assumption of stationarity.

Inspired by the success of the LMS algorithm, a "kernelization" has been recently proposed under the name kernel least mean squares (KLMS) algorithm [1]. The KLMS inherits many desirable properties of LMS and extends it to a large class of nonlinear filtering algorithms. Nevertheless, it has certain limitations that arise from its formulation as an adaptive filter in a possibly infinite dimensional feature space. Specifically, if implemented naively, the representation of the filter grows linearly with the number of data samples processed. Moreover, both the LMS and the KLMS explicitly minimize the squared error between the desired and the estimated observation values, hence, they cannot be naturally applied to problems with discrete observations, e.g., class labels or neural spike counts.

In this paper, we show that LMS (and KLMS) can indeed be derived as an approximation of a state-space based Bayesian filtering (section 3). In order to achieve a low computational complexity, only the mode of the posterior distribution can be estimated and retained for each sample. This new interpretation allows us to derive extensions of the KLMS algorithm by tweaking the underlying state-space and observation models (section 4.1 and 4.2). Here, we extend the KLMS algorithm to integer-valued observations, and also introduce a forgetting factor in order to improve its tracking ability[1].

2. LMS DERIVATION REVISITED

Both Widrow & Hoff's LMS, and KLMS are derived from mean squared error cost function [2, 1, 3] which is prevalent in traditional signal processing. The filtering setting assumes a linear model

$$f(\mathbf{x};\mathbf{w}) = \mathbf{w}^\top \phi(\mathbf{x})$$

in the *feature space* where $\phi(\mathbf{x}) \in \mathcal{H}$ is the feature vector associated with the input vector $\mathbf{x}$, and $\mathbf{w} \in \mathcal{H}$ is the vector representation of the filter in a Hilbert space $\mathcal{H}$. The following derivation holds for both LMS and KLMS, taking into account that for LMS the feature space is the (Euclidean) input space itself, i.e., $\phi(\mathbf{x}) = \mathbf{x} \in \mathbb{R}^d$, while KLMS uses a (potentially) infinite dimensional (reproducing kernel) Hilbert space induced by a positive definite kernel $k : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ where $k(\mathbf{x}, \mathbf{y}) = \phi(\mathbf{x})^\top \phi(\mathbf{y})$ [4]. The mean squared error is,

$$\mathcal{L}(\mathbf{w}) = \frac{1}{2}\,\mathrm{E}[(f(\mathbf{x};\mathbf{w}) - y)^2],$$

where $\mathbf{x}$ and y are the random vector and variable for the input signal and the desired output, respectively. The basic steepest descent learning rule has the form

$$\Delta\mathbf{w} \leftarrow -\eta \frac{\partial \mathcal{L}(\mathbf{w})}{\partial \mathbf{w}} = -\eta\,\mathrm{E}\left[(\mathbf{w}^\top \phi(\mathbf{x}) - y)\phi(\mathbf{x})\right]. \qquad (1)$$

[1] Extended version of this paper is available as `http://arxiv.org/abs/1310.5347` [stat.ML]

To make an online learning rule, a stochastic gradient descent is used in practice. In particular, the learning rule of the LMS algorithm is obtained by dropping the expectation from (1), which yields

$$\mathbf{w}_{i+1} \leftarrow \mathbf{w}_i - \eta_i({\mathbf{w}_i}^\top \phi(\mathbf{x}_i) - y_i)\phi(\mathbf{x}_i). \tag{2}$$

Hence, after processing i samples, the prediction for the next sample y_{i+1} is given by,

$$\hat{y}_{i+1} = {\mathbf{w}_i}^\top \phi(\mathbf{x}_{i+1}) = \sum_{k=1}^{i} \eta e_k \phi(\mathbf{x}_k)^\top \phi(\mathbf{x}_{i+1})$$

where $e_k = y_k - {\mathbf{w}_k}^\top \phi(\mathbf{x}_k)$ is the error for each sample. For KLMS, the prediction can be directly computed from the samples despite $f \in \mathcal{H}$, since $\phi(\mathbf{x}_i)^\top \phi(\mathbf{x}_k) = k(\mathbf{x}_i, \mathbf{x}_k)$.

The stochastic gradient descent algorithm is guaranteed to convergence (almost surely) to the global optimal solution under stationary and ergodic observations if a proper step size scheduling is used (e.g., $\sum_{i=1}^{\infty} \eta_i^2 < \infty$ and $\sum_{i=1}^{\infty} \eta_i = \infty$). However, the tracking capability of LMS/KLMS is dependent on the step size; if the step-size were annealed, it would be tracking less efficiently as more samples are seen. Therefore, to have constant tracking, step size is not annealed in practice, that is, $\forall i\ \eta_i = \eta$. Then, for the price of non-zero misadjustment, the algorithms can surprisingly learn continuously from new samples, and overwrite what was learned before. Note that this "hack" disconnects the algorithm from the graphical model Fig. 1A which inherently assumes a stationary data generation process. In the following section, we show how this tracking ability can be derived from first principles.

3. BAYESIAN INTERPRETATION

A slowly changing system can be explicitly described by a probabilistic model with latent dynamics on the parameter. In such a model, each parameter associated with each sample or time is considered as an interdependent random (latent) variable, as illustrated in Fig. 1B. Our goal is to show that the KLMS is an approximate sequential inference for $\mathbf{w}_i$. We start with a diffusion process as a reasonable model for nonstationarity:

$$P(\mathbf{w}_{k+1} | \mathbf{w}_k) = \mathcal{N}(\mathbf{w}_{k+1}; \mathbf{w}_k, \sigma_d^2 \mathbf{I}), \tag{3}$$

where $\mathcal{N}$ denotes a Gaussian distribution in the feature space, and σ_d^2 is the variance of diffusion on each direction. The likelihood model is assumed to be a linear–Gaussian model, similar to the stationary case,

$$P(y_k | \mathbf{x}_k, \mathbf{w}_k) = \mathcal{N}(y_k; \mathbf{w}_k^\top \phi(\mathbf{x}_k), \sigma_n^2) \tag{4}$$

where σ_n^2 is the observation noise variance. We remark that the conditional distributions (3) and (4) for the finite dimensional feature space is a special case of the Kalman filter model with linear dynamics.

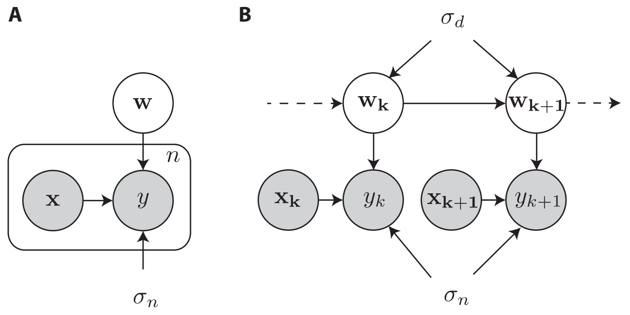

Fig. 1. Graphical models illustrating the contrast between stationary and nonstationary generative processes. The arrows signify the conditional dependence between variables. Gray shaded circles denote observed variables, and the box denotes repetition. (A) Stationary model. Each observation pair (x_i, y_i) is assumed to have the same relation $\mathbf{w}$ as in the classic regression setting. Original derivation of LMS is given in this context. (B) Non-stationary model. The weight $\mathbf{w}$ evolves over time—hence the relation between (x_i, y_i)—through diffusion with parameter σ_d. The Kalman filter is derived under this model. We show the KLMS can also be derived from the nonstationary model.

The posterior weight distribution given data $p(\mathbf{w}_k | \mathcal{D}_k)$ is fully described by its mean and covariance, since we assume Gaussianity in this model [5]. Let $P(\mathbf{w}_{k-1} | \mathcal{D}_{k-1}) = \mathcal{N}(\boldsymbol{\mu}_{k-1}, \boldsymbol{\Sigma}_{k-1})$, then a single linear Gaussian observation results in a one step evolution of the posterior as another Gaussian $P(\mathbf{w}_k | \mathcal{D}_k) = \mathcal{N}(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$, with

$$\boldsymbol{\Sigma}_k^{-1} = \boldsymbol{\Sigma}_{k-1}^{-1} + \frac{1}{\sigma_n^2} \phi(\mathbf{x}_k)\phi(\mathbf{x}_k)^\top$$

$$\boldsymbol{\mu}_k = \boldsymbol{\Sigma}_k \left[\frac{1}{\sigma_n^2} y_k \phi(\mathbf{x}_k) + \boldsymbol{\Sigma}_{k-1}^{-1} \boldsymbol{\mu}_{k-1} \right].$$

This recursion can be solved efficiently, and the solution is known as the extended recursive least squares algorithm [6]. However, it requires a quadratic number of operations in terms of the dimension of the feature vector for updating the (inverse) covariance matrix. In case of an infinite-dimensional feature space, the feature vector dimension grows linearly with the number of observations, rendering this approach prohibitive. Therefore, in order to obtain a linear time complexity algorithm, we assume the posterior to be concentrated around the maximum. In other words, we approximate the posterior as a delta function at the maximum a posteriori (MAP) estimate $P(\mathbf{w}_k | \mathcal{D}_k) \simeq \delta_{\mathbf{w}_k^{\text{MAP}}}$ before inferring $P(\mathbf{w}_{k+1} | \mathcal{D}_k)$. Below, we show the steps for online inference rules using this approximation.

First, note that the approximation is equivalent to assuming an isotropic Gaussian around the MAP estimate for the previous sample.

$$P(\mathbf{w}_{k+1} | \mathcal{D}_k) = \int P(\mathbf{w}_{k+1} | \mathbf{w}_k) P(\mathbf{w}_k | \mathcal{D}_k) \mathrm{d}\mathbf{w}_k$$

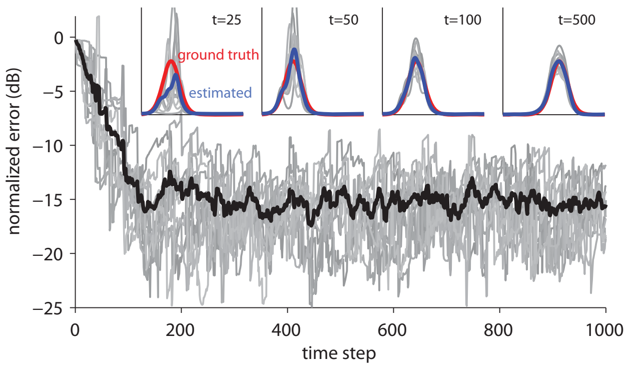

Fig. 2. Tracking example of the Poisson extension of KLMS algorithm. The observations model a slowly drifting tuning curve of a simple cell in V1. The tuning curve is modeled as an exponentiated cosine $\lambda(x) = \exp(4\cos(x-\mu) - 0.1)$ where μ constantly drifted 100 degrees during the 1000 iterations. We measure the normalized estimation error between the true tuning curve and the estimated curve. Insets show the actual function estimate at 25, 50, 100, 500 time steps. Gray lines show 11 repeats of the experiment, and the dark curves correspond to their average. The kernel was $k(x, y) = \exp(-(x-y)^2/100)$ and $\sigma_d^2 = 0.1$.

$$\begin{aligned} &\simeq \int P(\mathbf{w}_{k+1}|\mathbf{w}_k)\delta(\mathbf{w}_k^{\text{MAP}})\mathrm{d}\mathbf{w}_k \\ &= P(\mathbf{w}_{k+1}|\mathbf{w}_k^{\text{MAP}}) = \mathcal{N}(\mathbf{w}_k^{\text{MAP}}, \sigma_d^2\mathbf{I}). \qquad (5) \end{aligned}$$

Using Bayes' rule, the posterior weight distribution is,

$$\begin{aligned} &P(\mathbf{w}_{k+1}|\mathcal{D}_{k+1}) \propto P(y_{k+1}|\mathbf{x}_{k+1}, \mathbf{w}_{k+1})P(\mathbf{w}_{k+1}|\mathcal{D}_k) \\ &= \mathcal{N}(y_{k+1}; \mathbf{w}_{k+1}^\top \phi(\mathbf{x}_{k+1}), \sigma_n^2) \cdot \mathcal{N}(\mathbf{w}_{k+1}; \mathbf{w}_k^{\text{MAP}}, \sigma_d^2\mathbf{I}) \\ &= \mathcal{N}(\mathbf{w}_{k+1}; \mathbf{w}_{k+1}^{\text{MAP}}, \mathbf{\Sigma}_{k+1}) \end{aligned}$$

where the parameters for the posterior are,

$$\begin{aligned} \mathbf{\Sigma}_{k+1}^{-1} &= \frac{1}{\sigma_d^2}\mathbf{I} + \frac{1}{\sigma_n^2}\phi(\mathbf{x}_{k+1})\phi(\mathbf{x}_{k+1})^\top \\ \mathbf{w}_{k+1}^{\text{MAP}} &= \mathbf{\Sigma}_{k+1}\left[\frac{\mathbf{w}_k^{\text{MAP}}}{\sigma_d^2} + \frac{y_{k+1}\phi(\mathbf{x}_{k+1})}{\sigma_n^2}\right]. \end{aligned}$$

This can be simplified using the matrix inversion lemma,

$$\mathbf{w}_{k+1}^{\text{MAP}} = \mathbf{w}_k^{\text{MAP}} + \frac{\eta'(y_{k+1} - \mathbf{w}_k^{\text{MAP}\top}\phi(\mathbf{x}_{k+1}))\phi(\mathbf{x}_{k+1})}{1 + \eta'\phi(\mathbf{x}_{k+1})^\top\phi(\mathbf{x}_{k+1})} \qquad (6)$$

where the learning rate is determined by the diffusion-to-noise ratio $\eta' = \sigma_d^2/\sigma_n^2$. This is very similar to the normalized LMS (NLMS) update rule, although not identical. If the kernel is normalized, such that $k(\mathbf{x}, \mathbf{x}) = 1$, then it can be simply rewritten as,

$$\mathbf{w}_{k+1}^{\text{MAP}} = \mathbf{w}_k^{\text{MAP}} + \eta e_k \phi(\mathbf{x}_k) \qquad (7)$$

where $\eta = \eta'/(1+\eta')$. Note that the stochastic gradient derivation (2) is identical to the approximate Bayesian learning rule (7); we have rederived KLMS with a state-space model. Also, note that $0 < \eta < 1$, thus we have a frequentist convergence guarantee of the weight vector to the optimal weight vector $\mathbf{w}^*$ in mean in a stationary environment, i.e., $\lim_{k\to\infty} \mathrm{E}[\mathbf{w}_k] \to \mathbf{w}^*$ [1].

4. EXTENSTIONS

4.1. Forgetful dynamics for KLMS

Instead of a pure random walk dynamics (3), we add a leakage towards the origin, thus effectively forgetting the past exponentially. The resulting diffusion is a discrete-time analogue of the Ornstein-Uhlenbeck process (equivalently, a first order auto-regressive process).

$$p(\mathbf{w}_{k+1}|\mathbf{w}_k) = \mathcal{N}(\lambda\mathbf{w}_k, \sigma_d^2). \qquad (8)$$

The asymptotic marginal distribution of the prior dynamics is $\mathcal{N}(0, \sigma_d^2/(1-\lambda^2)\mathbf{I})$, hence the weights are isotropically distributed around the origin in the absence of observation. If $k(\mathbf{x}, \mathbf{x})$ is constant, the norm in the Hilbert space is proportional to the function norm, and the norm of the corresponding functions follows a Gaussian distribution centered around the origin. As a result the learning rule (6) becomes

$$\mathbf{w}_k^{\text{MAP}} = \lambda\mathbf{w}_{k-1}^{\text{MAP}} + \frac{\eta(y_k - \lambda\mathbf{w}_{k-1}^{\text{MAP}\top}\phi(\mathbf{x}_k))}{1 + \eta\|\phi(\mathbf{x}_k)\|^2}\phi(\mathbf{x}_k). \qquad (9)$$

This learning rule (9) is very similar to that of NORMA [2]. Note that the learning rule (9) can be rewritten as,

$$\mathbf{w}_k^{\text{MAP}} = \sum_{i=1}^{k} \lambda^{k-i}\beta_i\phi(\mathbf{x}_i), \qquad (10)$$

where β_i is a scalar corresponding to the coefficient at the learning step. We can see that each effective coefficient $\lambda^{k-i}\beta_i$ for each $\phi(\mathbf{x}_i)$ shrinks geometrically over time. Thus, the effect of older observation to the current weight estimate is small in general. Note, however, that the algorithm forgets not by making the covariance larger as in the Kalman filter, but by changing the mean, as discussed in [7, 8].

4.2. Novel observations models for KLMS

Poisson likelihood is widely used when the observations are natural numbers: $0, 1, 2, \cdots$. For example, in neuroscience, neural response is often quantified by the number of spikes, and tracking how the neural code changes during experiment is of great importance [9]. We use the canonical inverse link function (exponential) for the Poisson distribution to map the linear (or nonlinear) function from the input to the non-negative rate parameter, i.e.,

$$P(y_k|\mathbf{x}_k, \mathbf{w}_k) = \text{Poisson}(y_k; \exp(\mathbf{w}_k^\top\phi(\mathbf{x}_k))). \qquad (11)$$

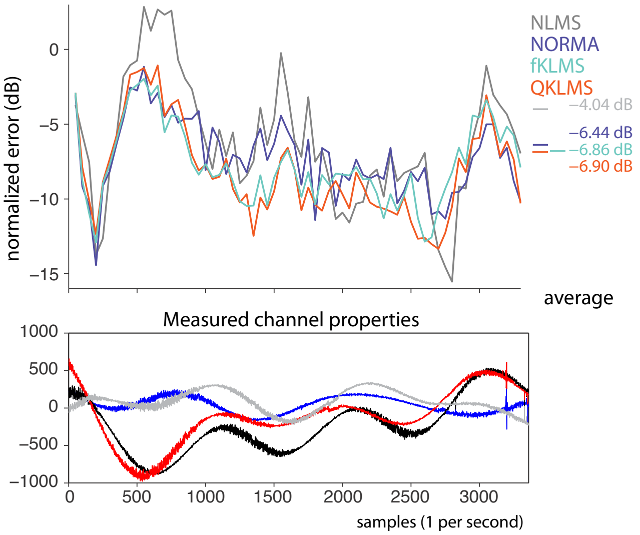

Fig. 3. (Top) Tracking results on a nonlinear Rayleigh fading channel, using data measured on a test bed with fast time-varying channels. One-step ahead prediction error and average over time are compared for 4 algorithms. fKLMS denotes the forgetting extension of KLMS (9) (Bottom) Time varying properties of a 4-channel MIMO system. Real part of the linear stage is measured. When the values are close to 0, the system operates almost linearly.

To derive the adaptive filtering algorithm, once again, we approximate the current state given the previous observations as (5), for which the log prior is,

$$\log P(\mathbf{w}_k|\mathcal{D}_{k-1}) = -\frac{1}{2\sigma_d^2}(\lambda\mathbf{w}_{k-1}^{\text{MAP}} - \mathbf{w}_k)^\top(\lambda\mathbf{w}_{k-1}^{\text{MAP}} - \mathbf{w}_k) + c.$$

Therefore, using Bayes' rule, the posterior at time k is,

$$\begin{aligned}\log P(\mathbf{w}_k|\mathcal{D}_k) &= \log P(y_k|\mathbf{x}_k, \mathbf{w}_k) + \log P(\mathbf{w}_k|\mathcal{D}_{k-1})\\ &= y_k\mathbf{w}_k^\top\phi(\mathbf{x}_k) - \exp(\mathbf{w}_k^\top\phi(\mathbf{x}_k))\\ &\quad - \frac{1}{2\sigma_d^2}(\lambda\mathbf{w}_{k-1}^{\text{MAP}} - \mathbf{w}_k)^\top(\lambda\mathbf{w}_{k-1}^{\text{MAP}} - \mathbf{w}_k),\end{aligned}$$

where irrelevant constants are omitted. We need to maximize this log-posterior over $\mathbf{w}_k$ to estimate $\mathbf{w}_k^{\text{MAP}}$. The stationary condition $\frac{\partial}{\partial\mathbf{w}_k}\log P(\mathbf{w}_k|\mathcal{D}_k) = 0$, implies,

$$(y_k - \exp({\mathbf{w}_k^{\text{MAP}}}^\top\phi(\mathbf{x}_k)))\phi(\mathbf{x}_k) = \frac{\lambda\mathbf{w}_{k-1}^{\text{MAP}} - \mathbf{w}_k^{\text{MAP}}}{2\sigma_d^2} \qquad (12)$$

We observe that the solution of (12) can be expressed as,

$$\mathbf{w}_k^{\text{MAP}} = \lambda\mathbf{w}_{k-1}^{\text{MAP}} + \alpha_k\phi(\mathbf{x}_k),$$

where α_k is a scalar, and, therefore, we can rewrite the log-posterior as,

$$J(\alpha_k) = y_k(\log(\psi_k) + \alpha_k) - \psi_k\exp(\alpha_k) - \frac{\alpha_k^2}{2\sigma_d^2}$$

where $\psi_k = \exp(\lambda{\mathbf{w}_{k-1}^{\text{MAP}}}^\top\phi(\mathbf{x}_k))$, and we have assumed a normalized kernel for simplicity.

Thus, we reduce the problem of finding an infinite-dimensional weight vector to a one-dimensional optimization. Although there is no analytical solution, the cost function $J(\alpha_k)$ is strictly concave and therefore, its maximum can be easily found by existing optimization tools. The complexity of this algorithm is still $\mathcal{O}(n)$, with a constant overhead for solving a concave maximization problem at each step. We demonstrate its performance on a neurally inspired example in Fig. 2. A typical nonlinear response function (tuning curve) is set to shift its center slowly, creating a non-stationary tracking problem. The Poisson-KLMS extension correctly tracks, and maintains a small MSE throughout the experiment.

5. NONLINEAR AND NONSTATIONARY CHANNEL

To test tracking, we acquired data from a wireless communication test bed that is used to evaluate the performance of digital communication systems in realistic indoor environments. The platform is composed of several transmit and receive nodes, each one including a radio-frequency front-end and baseband hardware for signal generation and acquisition. The front-end also incorporates a programmable variable attenuator that causes signal saturation (see [10] for details). Using the hardware platform, we transmitted clipped orthogonal frequency-division multiplexing (OFDM) signals centered at 5.4 GHz over real frequency-selective and time-varying channels, with normalized Doppler frequency around 10^{-3}. The transmit amplifier was operated close to saturation. In this experiment the transmitted and received signals are used to track the variations of the nonlinear channel. We compare 4 algorithms with hyperparameters tuned using the first 500 samples [2, 1, 11]. Fig. 3 displays the one-step ahead prediction normalized mean squared error (NMSE) of the tracking experiment. Quantized KLMS [11] and KLMS extended with forgetful dynamics have almost identical performance.

6. CONCLUSION

In this paper, we derived a family of linear time and space complexity kernel adaptive filtering algorithms from Bayesian filtering by maintaining only the mode of the posterior at each iteration and discarding the covariance. One of the basic resulting algorithms is the original KLMS. The tracking ability of LMS/KLMS is usually understood by its stochastic nature that allows it to continually adjust itself to the non-stationary environment. We provide an alternate explanation of this mechanism by showing the KLMS can also be seen as an approximation to state-space modeling which possesses explicit tracking abilities. Our framework allows flexibility in the state-space models which can be used to induce forgetting behavior, as well as novel observation noise models, such as Poisson and Bernoulli.

7. REFERENCES

[1] W. Liu, P. P. Pokharel, and J. C. Príncipe, "The kernel least-mean-square algorithm," *IEEE Transactions on Signal Processing*, vol. 56, no. 2, pp. 543–554, 2008.

[2] J. Kivinen, A. J. Smola, and R. C. Williamson, "Online learning with kernels," *IEEE Transactions on Signal Processing*, vol. 52, no. 8, pp. 2165–2176, Aug. 2004.

[3] W. Liu, J. C. Principe, and S. Haykin, *Kernel Adaptive Filtering: A Comprehensive Introduction*, Wiley Publishing, 1st edition, 2010.

[4] B. Schölkopf and A. J. Smola, *Learning with kernels: support vector machines, regularization, optimization, and beyond*, Adaptive computation and machine learning. MIT Press, 2002.

[5] S. Roweis and Z. Ghahramani, "A unifying review of linear gaussian models," *Neural Computation*, vol. 11, no. 2, pp. 305–345, Feb. 1999.

[6] W. Liu, I. Park, Y. Wang, and J. C. Príncipe, "Extended kernel recursive least squares algorithm," *IEEE Transactions on Signal Processing*, vol. 57, no. 10, pp. 3801–3814, Oct. 2009.

[7] A. H. Sayed, *Fundamentals of Adaptive Filtering*, Wiley-IEEE Press, 1 edition, June 2003.

[8] S. Van Vaerenbergh, M. Lázaro-Gredilla, and I. Santamaría, "Kernel recursive least-squares tracker for time-varying regression," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 23, no. 8, pp. 1313–1326, Aug. 2012.

[9] E. N. Brown, D. P. Nguyen, L. M. Frank, M. A. Wilson, and V. Solo, "An analysis of neural receptive field plasticity by point process adaptive filtering," *Proceedings of the National Academy of Sciences*, vol. 98, pp. 12261–12266, 2001.

[10] J. Gutiérrez, Ó. González, J. Pérez, D. Ramírez, L. Vielva, J. Ibáñez, and I. Santamaría, "Frequency-domain methodology for measuring MIMO channels using a generic test bed," *IEEE Trans. on Instrumentation and Measurement*, vol. 60, no. 3, pp. 827–838, Mar. 2011.

[11] B. Chen, S. Zhao, P. Zhu, and J. C. Principe, "Quantized kernel least mean square algorithm," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 23, no. 1, pp. 22–32, Jan. 2012.