Distributed Nonlinear System Identification in α-Stable Noise

Lu Lu, *Student Member, IEEE*, Haiquan Zhao, *Senior Member, IEEE*, and Benoit Champagne, *Senior Member, IEEE*

***Abstract*—In this letter, a novel diffusion Volterra (DV) algorithm is proposed for distributed in-network system identification in the presence of α-stable noise. The proposed algorithm is based on the logarithmic least mean pth-power criterion, which makes it robust against impulsive interferences, at the price of increased complexity. To overcome this shortcoming, we further develop the diffusion interpolated Volterra algorithm, which provides computational savings and good performance in comparison with the DV algorithm. Simulations results show that the proposed adaptive algorithms achieve better performance than the state-of-the-art approaches for distributed nonlinear system identification in impulsive noise.**

***Index Terms*—Distributed nonlinear network, interpolation, sparse filter, α-stable noise, Volterra filter.**

I. Introduction

DISTRIBUTED processing is a major tenet for sensor networks, where a set of interconnected nodes cooperate to estimate a system's parameter vector from noisy measurements [1]. Compared with its centralized counterpart, distributed estimation offers a more robust strategy that can be efficiently applied to large networks [2]. Many linear diffusion algorithms have been developed for distributed processing based on the adapt-then-combine policy, which yield good estimation under simplified conditions of operations [3]–[6].

In practical environments, each node in the network may introduce nonlinear distortion, in the form a nonlinear relationship between the system's input and output. For these so-called *distributed nonlinear systems*, the linear diffusion algorithms tend to perform poorly [7]. Nevertheless, there is little literature addressing the problem of distributed nonlinear system identification. By making use of the kernel method, some nonlinear distributed algorithms have been proposed in [7]–[9]. Particularly, in [9], an interesting approach was presented for nonlinear adaptive learning by employing the kernel LMS adaptive filter. However, the complexity of this method grows with the number of processed data, which hinders its practical application.

Manuscript received March 14, 2018; revised May 7, 2018; accepted May 8, 2018. Date of publication May 11, 2018; date of current version May 29, 2018. This work was supported by the National Science Foundation of P.R. China under Grant 61571374, Grant 61271340, and Grant 61433011. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Magno T. M. Silva. *(Corresponding author: Haiquan Zhao.)*

L. Lu is with the College of Electronics and Information Engineering, Sichuan University, Chengdu 610065, China (e-mail: lulu19900303@126.com).

H. Zhao is with the School of Electrical Engineering, Southwest Jiaotong University, Chengdu 610031, China (e-mail: hqzhao_swjtu@126.com).

B. Champagne is with the Department of Electrical and Computer Engineering, McGill University, Montreal QC H3A0E9, Canada (e-mail: benoit.champagne@mcgill.ca).

Color versions of one or more of the figures in this letter are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/LSP.2018.2835763

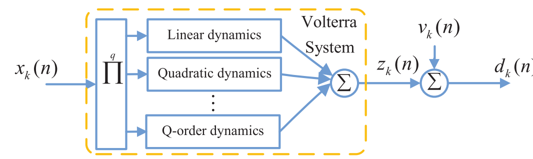

Fig. 1. Block diagram of adaptive Volterra system at node k.

The Volterra filter has been used as a nonlinear system modeling tool with considerable success [10]–[12]. However, this type of filter becomes computationally expensive when a large number of coefficients are required. The simplified second-order Volterra (SOV) filter was developed to cope with the large amount of computations needed to obtain acceptable error level in nonlinear modeling [13], [14]. Sparse-interpolated structures offer an efficient alternative to reduce the computational complexity [15]–[17]. The main idea of the interpolated Volterra filter (IVF) is to use a filter cascade composed of a sparse filter, with a reduced number of coefficients, and an interpolating filter whose purpose is to recreate the missing coefficients in the sparse filter [15]. In this way, the IVF achieves good nonlinear modeling capability.

Motivated by these considerations, in this letter, we focus on the diffusion Volterra (DV) algorithm for distributed nonlinear system identification in the presence of impulsive noise with α-stable distribution. The SOV and IVF concepts are exploited to achieve nonlinear modeling capability with reduced computational complexity. Development of the adaptation rule is based on the minimization of the pth-power logarithmic error criterion, instead of the conventional pth-power error. Consequently, the new logarithmic least mean pth-power (LLMP) algorithm achieves robust performance against outliers, as demonstrated by extensive simulations.

II. Problem Formulation

We consider the problem of identifying Volterra coefficients of a nonlinear distributed system using noisy measurements from a spatial network of P sensor nodes, indexed by $k \in \{1, \ldots, P\}$. At each time instant $n \in \mathbb{N}$, node k receives a new observation $\{x_k(n), d_k(n)\}$, where $x_k(n)$ is the input signal, $d_k(n) = z_k(n) + v_k(n)$ is the scalar measurement, $z_k(n)$ is the system output, and $v_k(n)$ is the additive noise (see Fig. 1).

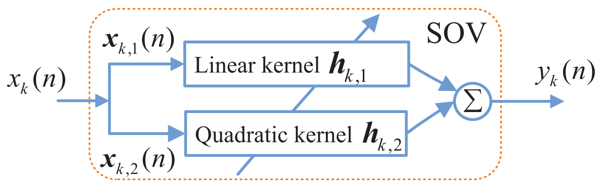

Fig. 2. Block diagram of the DV filter at node k.

The output of the unknown nonlinear system at node k can be represented by the following truncated Volterra series:

$$z_k(n) = \sum_{q=1}^{Q} \sum_{m_1=0}^{M-1} \cdots \sum_{m_q=0}^{M-1} \psi_q(m_1, ..., m_q) \prod_{i=1}^{q} x_k(n - m_i) \tag{1}$$

where Q denotes the order of nonlinearity, M is the memory length, and $\psi_q(\cdot)$ is the qth-order Volterra kernel to be identified. In this letter, we consider a physical scenario where the output of the distributed nonlinear system at each node is contaminated with impulsive noise, modeled by the standard symmetric α-stable distribution. Specifically, the noise samples $v_k(n)$ at node k are statistically independent, with characteristic function

$$\phi_k(t) = \exp\{-|t|^{\alpha_k}\} \tag{2}$$

where $\alpha_k \in (0, 2]$ is the *characteristic exponent*. The values of α_k at the different nodes can be different in practice. In the literature, several approaches have been proposed to address the characteristic exponent estimation problem in practical applications, such as those proposed in [18]–[20] and references therein. In this letter, we assume that prior knowledge of α-stable noise is available, as obtained through the application of such methods.

III. Proposed Algorithms

A. Diffusion Volterra Filter

The diffusion-SOV filter is composed of $Q = 2$ kernels at each node, as shown in Fig. 2. The output $y_k(n)$ of a DV filter at node k relies linearly on the coefficients of the filter itself. Hence, it can be compactly written as

$$\begin{aligned} y_k(n) = \boldsymbol{x}_k^{\mathrm{T}}(n)\boldsymbol{h}_k(n) = &\sum_{m_1=0}^{M-1} h_{k,1}(m_1)x_k(n-m_1) \\ &+ \sum_{m_1=0}^{M-1}\sum_{m_2=m_1}^{M-1} h_{k,2}(m_1, m_2)x_k(n-m_2)x_k(n-m_1) \end{aligned} \tag{3}$$

where the expanded input vector and filter coefficient vector can be expressed as

$$\begin{aligned} \boldsymbol{x}_k(n) = [\boldsymbol{x}_{k,1}^{\mathrm{T}}(n), \boldsymbol{x}_{k,2}^{\mathrm{T}}(n)]^{\mathrm{T}} &= [x_k(n), ..., x_k(n-M+1) \\ & x_k^2(n), x_k(n)x_k(n-1), ..., x_k^2(n-M+1)]^{\mathrm{T}} \end{aligned} \tag{4}$$

$$\begin{aligned} \boldsymbol{h}_k = [\boldsymbol{h}_{k,1}^{\mathrm{T}}, \boldsymbol{h}_{k,2}^{\mathrm{T}}]^{\mathrm{T}} &= [h_{k,1}(0), h_{k,1}(1), ..., h_{k,1}(M-1) \\ & h_{k,2}(0,0), h_{k,2}(0,1), ..., h_{k,2}(M-1, ..., M-1)]^{\mathrm{T}}. \end{aligned} \tag{5}$$

In these expressions, $\boldsymbol{x}_{k,1}$ and $\boldsymbol{x}_{k,2}$, respectively, denote the input vectors of the linear and quadratic kernels, while $\boldsymbol{h}_{k,1}$ and $\boldsymbol{h}_{k,2}$ are the corresponding weight vectors. In the general case

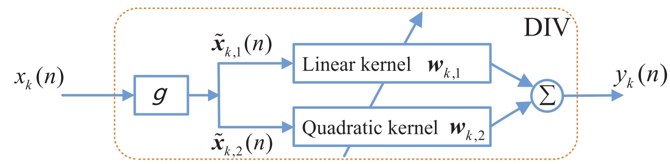

Fig. 3. Block diagram of the DIV filter at node k.

with memory size M and nonlinearity order Q, the number of coefficients for each node can be computed as [15]

$$T_1 = \frac{(M+Q)!}{M!Q!} - 1. \tag{6}$$

Remark 1: The main challenge with the DV filter is that the number of model coefficients grows in M^Q, resulting in high computational costs and memory requirements, especially for large networks. Both the discrete cosine transform method [21] and the sparse interpolated structure [15] can be employed to reduce the computational complexity. However, our preliminary results indicate that the sparse interpolated structure exhibits good performance and low computational cost in this context.

B. Diffusion Interpolated Volterra Filter

The main idea behind the proposed DIV is strongly related to the concept of interpolated sparse adaptive filtering, see e.g., [15], [16], [22], [23]. Fig. 3 shows the diagram of the DIV filter at node k, where $\boldsymbol{g} = [g_0, g_1, ..., g_{N-1}]^{\mathrm{T}}$ is the fixed interpolation filter with length N, $\boldsymbol{w}_{k,q}$ for $q \in \{1, 2\}$ is the qth-order sparse weight vector, and $\boldsymbol{W}_k = [\boldsymbol{w}_{k,1}^{\mathrm{T}}, \boldsymbol{w}_{k,2}^{\mathrm{T}}]^{\mathrm{T}}$. The output of the interpolation filter is expressed as $\widetilde{x}_k(n) = \boldsymbol{u}_k^{\mathrm{T}}(n)\boldsymbol{g}$ with $\boldsymbol{u}_k(n) = [x_k(n), x_k(n-1), ..., x_k(n-N+1)]^{\mathrm{T}}$.

The linear kernel of the DIV filter $\boldsymbol{w}_{k,1}$ can be obtained by setting to zero $L-1$ coefficients of each group of L adjacent coefficients of the basic linear Volterra kernel, i.e.,

$$\begin{aligned} \boldsymbol{w}_{k,1} = &[h_{k,1}(0), 0, ..., h_{k,1}(L) \\ & 0..., h_{k,1}((N_z-1)L), 0, ..., 0]^{\mathrm{T}} \end{aligned} \tag{7}$$

where $N_z = \lfloor (M-1)/L \rfloor + 1$ stands for the number of nonzero coefficients and the operator $\lfloor \cdot \rfloor$ denotes the floor function. The input vector of the linear kernel is given by

$$\widetilde{\boldsymbol{x}}_{k,1}(n) = [\widetilde{x}_k(n), \widetilde{x}_k(n-1), ..., \widetilde{x}_k(n-M+1)]^{\mathrm{T}}. \tag{8}$$

Accordingly, (8) can be expressed as

$$\widetilde{\boldsymbol{x}}_{k,1}(n) = \boldsymbol{X}_{k,1}^{\mathrm{T}}(n)\boldsymbol{g} \tag{9}$$

where $\boldsymbol{X}_{k,1}(n) = [\boldsymbol{u}_k(n), \boldsymbol{u}_k(n-1), ..., \boldsymbol{u}_k(n-M+1)]$. Hence, the output of the linear kernel is given by

$$y_{k,1}(n) = \widetilde{\boldsymbol{x}}_{k,1}^{\mathrm{T}}(n)\boldsymbol{w}_{k,1} = \boldsymbol{g}^{\mathrm{T}}\boldsymbol{X}_{k,1}(n)\boldsymbol{w}_{k,1}. \tag{10}$$

Based on [24], the input vector of the quadratic kernel is

$$\widetilde{\boldsymbol{x}}_{k,2}(n) = \widetilde{\boldsymbol{x}}_{k,1}(n) \otimes \widetilde{\boldsymbol{x}}_{k,1}(n) \tag{11}$$

where $\otimes$ denotes the Kronecker product. For $L = 2$, the sparse weight vector of the quadratic kernel is given by

$$\begin{aligned} \boldsymbol{w}_{k,2} = &[h_{k,2}(0,0), 0, h_{k,2}(0,2), 0, 0, 0 \\ & h_{k,2}(2,0), 0, h_{k,2}(2,2)]^{\mathrm{T}} \end{aligned} \tag{12}$$

while for the general case L arbitrary, the reader is referred to [16]. Using properties of the Kronecker mixed products and considering (9) and (11), the output of the quadratic kernel at node k can be expressed as

$$\begin{aligned} y_{k,2}(n) &= \widetilde{\boldsymbol{x}}_{k,2}^{\mathrm{T}}(n)\boldsymbol{w}_{k,2} = \left[\boldsymbol{g}^{\mathrm{T}}\boldsymbol{X}_{k,1}(n)\otimes\boldsymbol{g}^{\mathrm{T}}\boldsymbol{X}_{k,1}(n)\right]\boldsymbol{w}_{k,2} \\ &= \left(\boldsymbol{g}^{\mathrm{T}}\otimes\boldsymbol{g}^{\mathrm{T}}\right)\left[\boldsymbol{X}_{k,1}(n)\otimes\boldsymbol{X}_{k,1}(n)\right]\boldsymbol{w}_{k,2} \\ &= \left(\boldsymbol{g}^{\mathrm{T}}\otimes\boldsymbol{g}^{\mathrm{T}}\right)\boldsymbol{X}_{k,2}(n)\boldsymbol{w}_{k,2} \end{aligned} \tag{13}$$

where $\boldsymbol{X}_{k,2}(n) = \boldsymbol{X}_{k,1}(n)\otimes\boldsymbol{X}_{k,1}(n)$. Consequently, for the SOV structure, the output of the DIV filter is given by

$$y_k(n) = \sum_{q=1}^{2} y_{k,q}(n) = \boldsymbol{U}_k^{\mathrm{T}}(n)\boldsymbol{W}_k \tag{14}$$

where $\boldsymbol{U}_k(n) = [\boldsymbol{g}^{\mathrm{T}}\boldsymbol{X}_{k,1}(n), (\boldsymbol{g}^{\mathrm{T}}\otimes\boldsymbol{g}^{\mathrm{T}})\boldsymbol{X}_{k,2}(n)]^{\mathrm{T}} = [\boldsymbol{U}_{k,1}^{\mathrm{T}}(n), \boldsymbol{U}_{k,2}^{\mathrm{T}}(n)]^{\mathrm{T}}$, $\boldsymbol{U}_{k,1}^{\mathrm{T}}(n)$ is the input vector of the linear kernel and $\boldsymbol{U}_{k,2}^{\mathrm{T}}(n)$ is the input vector of the quadratic kernel.

From (6) and (7), the number of model coefficients of the DIV filter for each node is now given by

$$T_2 = \frac{(N_z+Q)!}{N_z!Q!} - 1. \tag{15}$$

C. Adaptation Rule: LLMP Algorithm

To identify the parameters of the Volterra system in α-stable noise, we hereby propose a novel cost function. Define the error signal at node k as

$$e_k(n) \triangleq d_k(n) - y_k(n) \tag{16}$$

where we may employ (3) and (14) to compute output signal $y_k(n)$ for the DV and DIV filter, respectively. In [25], a logarithmic cost function is proposed for performance improvement. However, such cost function does not consider using fractional lower order moments of the error signal for combating highly impulsive interference. Motivated by [11], a new local cost function that includes logarithmic dependence on the error and allows for noninteger exponent in the error magnitudes is defined for the DV filter as

$$J_k^{\mathrm{loc}}(\boldsymbol{h}) \triangleq \sum_{l\in\mathcal{N}_k} a_{l,k}\mathrm{E}\left[F_l(e_l(n)) - \ln\{1+F_l(e_l(n))\}\right] \tag{17}$$

where $\mathcal{N}_k$ is the set of nodes with which node k shares information (including k itself), $\{a_{l,k}\}$ are real nonnegative weighting coefficients, and $\mathrm{E}[\cdot]$ denotes statistical expectation. We define the function $F_l(e_l(n)) = |e_l(n)|^{p_l}$ to represent the pth-power error signal, where the exponent p_l is node dependent in general. For the DIV filter, we replace $\boldsymbol{h}$ in (17) by $\boldsymbol{W}$. Using the method of steepest descent, the DV and DIV filters can be updated as

$$\boldsymbol{\zeta}_{k,q}(n) = \boldsymbol{h}_{k,q}(n-1) + \mu_q\frac{\boldsymbol{x}_{k,q}(n)|e_k(n)|^{2p_k-1}\mathrm{sign}(e_k(n))}{1+|e_k(n)|^{p_k}} \tag{18}$$

$$\boldsymbol{\xi}_{k,q}(n) = \boldsymbol{w}_{k,q}(n-1) + \mu_q\frac{\boldsymbol{U}_{k,q}(n)|e_k(n)|^{2p_k-1}\mathrm{sign}(e_k(n))}{1+|e_k(n)|^{p_k}} \tag{19}$$

where $q\in\{1,2\}$, sign $(\cdot)$ denotes the sign function, and μ_1 and μ_2 are the step sizes for the linear and quadratic kernels, respectively. Considering that the eigenvalue spread of the correlation matrix of the quadratic kernel is larger than that of the linear kernel, these two kernels for both DV–LLMP and DIV–LLMP algorithms are adapted by using different step sizes, i.e., $\mu_1 > \mu_2$. This approach can help achieve a compromise between the different convergence behaviors of the two kernels [12]. Finally, the updated weights are linearly combined to obtain the estimate of the unknown system, i.e.,

$$\boldsymbol{h}_{k,q}(n) = \sum_{l\in\mathcal{N}_k} a_{l,k}\boldsymbol{\zeta}_{l,q}(n) \tag{20}$$

$$\boldsymbol{w}_{k,q}(n) = \sum_{l\in\mathcal{N}_k} a_{l,k}\boldsymbol{\xi}_{l,q}(n). \tag{21}$$

Remark 2: When $p_k = 1$, the LLMP algorithm reduces to the diffusion least logarithmic absolute difference (LLAD) algorithm [25]. When $p_k = 2$, it becomes the diffusion least mean logarithmic square (LMLS) algorithm [26]. The LMLS algorithm involves the fourth-order statistics of the error signal, and its performance may degrade in the presence of impulsive noise. The LLAD algorithm intrinsically combines the l_2-norm and l_1-norm. In impulsive noise environments, the LLAD algorithm is therefore expected to achieve better performance than the LMLS algorithm.

D. Complexity and Memory Requirements

TABLE I
SUMMARY OF THE COMPUTATIONAL COMPLEXITY AND RUN TIME OF DIV-BASED ALGORITHMS (PER ITERATION PER NODE)

Algorithm	$\times$	$+$	$\div$	sign	pth-power	Run time
LMP	$2T_s+1$	$2T_s$	0	1	1	$2.78\times10^{-4}s$
LMLS	$2T_s+3$	$2T_s+1$	1	0	—	$2.69\times10^{-4}s$
LLAD	$2T_s+1$	$2T_s+1$	1	1	—	$2.54\times10^{-4}s$
LLMP	$2T_s+1$	$2T_s+1$	1	1	4	$3.20\times10^{-4}s$

Table I summarizes the computational complexity and the run time (per iteration per node) of the proposed DIV–LLMP algorithm, along with that of the DIV-least mean pth power (LMP) [10], DIV–LMLS, and DIV–LLAD, where T_s for $s = 1, 2$ is the number of filter coefficients as per (6) of (15). The column labeled pth power, gives the integer count of fractional power operations required. The rightmost column provides the average run time per iteration per node of the DIV-based algorithms, when implemented in Matlab R2013a.[1] It is seen that the computational cost of the proposed DIV–LLMP only slightly exceeds that of the LMP, LMLS, or LLAD algorithms.

Next, we investigate the memory requirements of the proposed algorithms. The DV–LLMP algorithm requires $(T_1+1)N_k + 2T_1 + 6$ memory locations for node k, where T_1 is the number of coefficients for the DV filter [see (6)] and $N_k = |\mathcal{N}_k|$ is the number of elements in the neighborhood set $\mathcal{N}_k$. The DIV–LLMP algorithm requires $(T_2+1)N_k + 2T_2 + 6 + N$ memory locations for node k, where T_2 denotes the number of coefficients of the DIV filter [see (15)]. Fig. 4 illustrates the memory requirements of the proposed algorithms as a function of N_k and M. As can be seen, the DIV–LLMP algorithm significantly reduces the memory requirements as compared with the DV–LLMP algorithm.

[1] The run times were obtained by implementing the algorithms on a 2.1 GHz AMD processor with 4 GB RAM and averaging over 100 runs. These figures may vary depending on specific implementation, type of memory access, multitasking, etc., but the values are representative of the relative complexity of the various algorithms under evaluation.

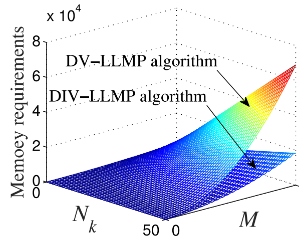

Fig. 4. Memory requirements of the proposed algorithms ($L = 2, N = 3$).

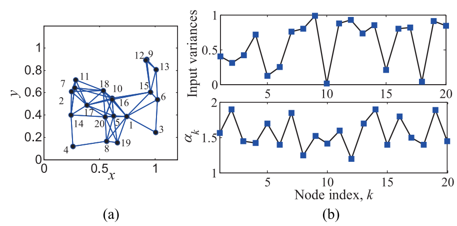

Fig. 5. (a) Network topology. (b) Variances of the input signals and α-stable noises at the different nodes in the nonlinear distributed network.

IV. Simulation Results

We present simulation results to verify the effectiveness of the proposed algorithms in comparison with the DV–LMLS, DIV–LMLS, DV–LLAD, DIV–LLAD, DV–LMP, and DIV–LMP algorithms.[2] The nonlinear distributed network is composed of $P = 20$ nodes, where every pair of nodes has a 30% probability of being connected [see Fig. 5(a)]. The linear combination coefficients $\{a_{l,k}\}$ in (17) and (21) are selected based on the uniform rule [3]. The performance of the various algorithms is evaluated in terms of the network excess mean-square error (EMSE), defined as

$$\text{network EMSE} = \frac{1}{P}\sum_{k=1}^{P}(z_k(n) - y_k(n))^2. \tag{22}$$

All simulations are averaged over 200 independent trials. The interpolation factor is set to $L = 2$ and the coefficients of the fixed interpolator are given by $\boldsymbol{g} = [0.5, 1, 0.5]^{\mathrm{T}}$. The memory size of the nonlinear system is set to $M = 11$ [15]. The variances of the Gaussian input signals and impulsive noises at the different nodes in the network are illustrated in Fig. 5(b). The values of the characteristic exponents α_k at the different nodes are randomly generated between 1 and 2. It is now well established that a Banach space framework can be used for the geometrical treatment of α-stable processes, where only the existence of pth-power moment for $p_k < \alpha_k$ is required [28]. Besides, the choice of p_k closest to α_k tends to yield the best results in system identification problems [10]. Based on such considerations, here we simply set $p_k = \alpha_k - 0.1$ for convenience.

Fig. 6 illustrates the network EMSE learning curves for the different algorithms under evaluation, for a scenario where drastic changes occur at regular intervals in the true nonlinear system parameters. We can see that in all cases, the DV-based algorithms achieves better performance than their DIV-based counterparts, but at the cost of increased complexity (for DV, $T_1 = 1720$, while for DIV, $T_2 = 623$). Among the DIV-based algorithms, the proposed DIV–LLMP generally achieves faster convergence

[2]The DV–LMLS, DIV–LMLS, DV–LLAD, DIV–LLAD, DV–LMP, and DIV–LMP algorithms can be derived by integrating algorithms [26], [27] in the DV and DIV filter structures.

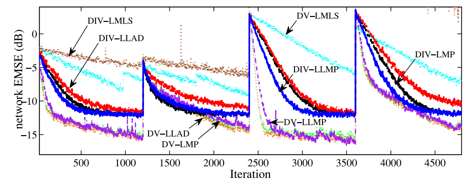

Fig. 6. Network EMSE versus iteration number for various algorithms under study. At $n = 1200$, the quadratic kernel changes abruptly while the linear one remains invariant, and at $n = 2400$, the opposite situation occurs. Finally, at $n = 3600$, both kernels are simultaneously changed. DV–LMP: $\mu_1 = 0.05$, $\mu_2 = 0.005$, DV–LMLS: $\mu_1 = 0.002$, $\mu_2 = 0.0002$, DV–LLAD: $\mu_1 = 0.1$, $\mu_2 = 0.01$, DV–LLMP: $\mu_1 = 0.05$, $\mu_2 = 0.005$, DIV–LMP: $\mu_1 = 0.005$, $\mu_2 = 0.003$, DIV–LMLS: $\mu_1 = 0.0005$, $\mu_2 = 0.00005$, DIV–LLAD: $\mu_1 = 0.01$, $\mu_2 = 0.001$, DIV–LLMP: $\mu_1 = 0.01$, $\mu_2 = 0.003$.

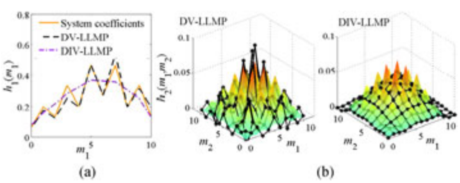

Fig. 7. (a) Linear kernel of the algorithms in steady state. (b) Quadratic kernel of the algorithms in steady state (Key: solid surface, system coefficients; black grid, coefficients of the proposed structure).

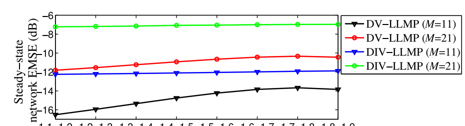

Fig. 8. Steady-state network EMSE for different α_k.

than the existing algorithms while reaching a similar level of steady-state error. To further illustrate the performance of the proposed algorithms, in Fig. 7, the estimated system coefficients in steady state are shown along with the true system coefficients. One can see that the DV–LLMP algorithm achieves better performance in terms of modeling capability than the DIV–LLMP algorithm. Finally, the steady-state network EMSE performance of the proposed algorithms is illustrated in Fig. 8. The same parameter settings and input signal are taken for simulation study as Fig. 6. We observed that the proposed algorithms achieve the best performance under the highly impulsive noise.

V. Conclusion

Novel DV–LLMP and a DIV–LLMP algorithms have been proposed for distributed nonlinear system identification and investigated through simulations. The DV–LLMP algorithm is based on the pth-power logarithmic cost and enjoys small kernel misadjustment. The DIV–LLMP algorithm incorporates the interpolated Volterra structure along with the LLMP adaptive algorithm, which can reduce computational complexity and maintain robustness against impulsive interference. Simulation results demonstrate that the proposed algorithms can achieve excellent performance for distributed nonlinear identification in the presence of α-stable noise. The proposed algorithms have potential applications in distributed wireless sensor networks, as previous studies in this area show that locally, i.e., at the node level, the communication process is often plagued by nonlinear distortions and impulsive interferences. Our future work will focus on applying the newly proposed diffusion-based algorithms to the identification of key system parameters in such practical settings.

REFERENCES

[1] S. Kanna and D. P. Mandic, “Steady-state behavior of general complex-valued diffusion LMS strategies,” *IEEE Signal Process. Lett.*, vol. 23, no. 5, pp. 722–726, May 2016.

[2] R. Abdolee and B. Champagne, “Diffusion LMS strategies in sensor networks with noisy input data,” *IEEE/ACM Trans. Netw.*, vol. 24, no. 1, pp. 3–14, Feb. 2016.

[3] C. Li, P. Shen, Y. Liu, and Z. Zhang, “Diffusion information theoretic learning for distributed estimation over network,” *IEEE Trans. Signal Process.*, vol. 61, no. 16, pp. 4011–4024, Aug. 2013.

[4] F. S. Cattivelli and A. H. Sayed, “Diffusion LMS strategies for distributed estimation,” *IEEE Trans. Signal Process.*, vol. 58, no. 3, pp. 1035–1048, Mar. 2010.

[5] Y. Zhang, C. Wang, L. Zhao, and J. A. Chambers, “A spatial diffusion strategy for tap-length estimation over adaptive networks,” *IEEE Trans. Signal Process.*, vol. 63, no. 17, pp. 4487–4501, Sep. 2015.

[6] X. Zhao and A. H. Sayed, “Performance limits for distributed estimation over LMS adaptive networks,” *IEEE Trans. Signal Process.*, vol. 60, no. 10, pp. 5107–5124, Oct. 2012.

[7] R. Mitra and V. Bhatia, “Diffusion-KLMS algorithm and its performance analysis for non-linear distributed networks,” arXiv:1509.01352, 2015.

[8] B.-S. Shin, H. Paul, and A. Dekorsy, “Distributed kernel least squares for nonlinear regression applied to sensor networks,” in *Proc. 24th Eur. Signal Process. Conf.*, Aug. 29–Sep. 2, 2016, pp. 1588–1592.

[9] S. Chouvardas and M. Draief, “A diffusion kernel LMS algorithm for nonlinear adaptive networks,” in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, Mar. 20–25, 2016, pp. 4164–4168.

[10] B. Weng and K. E. Barner, “Nonlinear system identification in impulsive environments,” *IEEE Trans. Signal Process.*, vol. 53, no. 7, pp. 2588–2594, Jul. 2005.

[11] L. Lu, H. Zhao, and B. Chen, “Improved-variable-forgetting-factor recursive algorithm based on the logarithmic cost for Volterra system identification,” *IEEE Trans. Circuits Syst. II*, vol. 63, no. 6, pp. 588–592, Jun. 2016.

[12] L. Lu, H. Zhao, and B. Chen, “Collaborative adaptive Volterra filters for nonlinear system identification in α-stable noise environments,” *J. Franklin Inst.*, vol. 353, no. 17, pp. 4500–4525, Nov. 2016.

[13] A. Fermo, A. Carini, and G. L. Sicuranza, “Simplified Volterra filters for acoustic echo cancellation in GSM receivers,” in *Proc. 10th Eur. Signal Process. Conf.*, Sep. 4–8, 2000, pp. 1–4.

[14] M. Zeller, L. A. A. Ruiz, J. A. Garcia, and W. Kellermann, “Adaptive Volterra filters with evolutionary quadratic kernels using a combination scheme for memory control,” *IEEE Trans. Signal Process.*, vol. 59, no. 4, pp. 1449–1464, Apr. 2011.

[15] E. L. Batista, O. J. Tobias, and R. Seara, “A fully LMS adaptive interpolated Volterra structure,” in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, Mar. 31–Apr. 4, 2008, pp. 3613–3616.

[16] E. L. O. Batista, O. J. Tobias, and R. Seara, “A sparse-interpolated scheme for implementing adaptive Volterra filters,” *IEEE Trans. Signal Process.*, vol. 58, no. 4, pp. 2022–2035, Apr. 2010.

[17] E. L. O. Batista and R. Seara, “On the performance of adaptive pruned Volterra filters,” *Signal Process.*, vol. 93, no. 7, pp. 1909–1920, Jul. 2013.

[18] K. Kokkinakis and A. K. Nandi, “Exponent parameter estimation for generalized Gaussian probability density functions with application to speech modeling,” *Signal Process.*, vol. 85, no. 9, pp. 1852–1858, Sep. 2005.

[19] M. Pi, “Improve maximum likelihood estimation for subband GGD parameters,” *Pattern Recognit. Lett.*, vol. 27, no. 14, pp. 1710–1713, Oct. 2006.

[20] Y. Chen and N. C. Beaulieu, “Novel low-complexity estimators for the shape parameter of the generalized Gaussian distribution,” *IEEE Trans. Veh. Technol.*, vol. 58, no. 4, pp. 2067–2071, May 2009.

[21] T. I. Haweel, “A new square wave transform based on the DCT,” *Signal Process.*, vol. 81, no. 11, pp. 2309–2319, Nov. 2001.

[22] R. C. Lamare and R. Sampaio-Neto, “Adaptive reduced-rank MMSE filtering with interpolated FIR filters and adaptive interpolators,” *IEEE Signal Process. Lett.*, vol. 12, no. 3, pp. 177–180, Mar. 2005.

[23] R. C. Lamare and R. Sampaio-Neto, “Adaptive reduced-rank processing based on joint and iterative interpolation, decimation and filtering,” *IEEE Trans. Signal Process.*, vol. 57, no. 7, pp. 2503–2514, Jul. 2009.

[24] V. J. Mathews and G. L. Sicuranza, *Polynomial Signal Processing*, vol. 27. Hoboken, NJ, USA: Wiley-Interscience, 2000.

[25] M. O. Sayin, N. D. Vanli, and S. S. Kozat, “A novel family of adaptive filtering algorithms based on the logarithmic cost,” *IEEE Trans. Signal Process.*, vol. 62, no. 17, pp. 4411–4424, Sep. 2014.

[26] Z. Zheng, Z. Liu, and M. Huang, “Diffusion least mean square/fourth algorithm for distributed estimation,” *Signal Process.*, vol. 134, pp. 268–274, May 2017.

[27] F. Wen, “Diffusion least-mean p-power algorithms for distributed estimation in alpha-stable noise environments,” *Electron. Lett.*, vol. 49, no. 21, pp. 1355–1356, Oct. 2013.

[28] M. Shao and C. L. Nikias, “Signal processing with fractional lower order moments: Stable processes and their applications,” *Proc. IEEE*, vol. 81, no. 7, pp. 986–1010, Jul. 1993.