Poster Abstract: Sparse Bayesian Distributed Least-Squares Strategies for Adaptive Networks

Thomas Buchgraber
Signal Processing and Speech Comm. Lab.
Graz University of Technology
Inffeldgasse 12, 8010 Graz, Austria
Email: thomas.buchgraber@tugraz.at

Dmitriy Shutin
Department of Electrical Engineering
Princeton University
Equad-B311, Olden Street, Princeton, NJ, 08544
Email: dshutin@princeton.edu

***Abstract*—In this work we present a sparse distributed recursive least-squares (dRLS) algorithm for adaptive sensor networks. The algorithm allows to prune weights by exploiting sparse Bayesian learning, while at the same time adding automatic regularization to improve numerical stability. With this reduction of computational complexity, the amount of energy consumption and data to be transmitted between neighboring nodes will decrease. This makes it suitable for the use in wireless sensor networks (WSNs), where energy and bandwidth constraints are typically of a great concern. Furthermore, the proposed algorithm works totally decentralized with no need for a fusion center.**

I. Introduction

The standard RLS algorithm is well known in the field of adaptive signal processing. However, without proper regularization, it suffers from numerical instabilities in case of poor excitation signals. To overcome this problem, a sparse Bayesian method for automatic regularization adjustment and weight pruning was developed in [1]. In this contribution, following the approach of the relevance vector machine [2], the authors start from a linear-in-parameters model and define independent zero-mean Gaussian priors over all the weights. Solving for the precision parameters (inverse variances) of the priors using the Bayesian evidence procedure leads to a method to obtain these coefficients. In case of a sparse weight vector, some precision parameters, also called hyper-parameters, tend to very large numbers, which means that the corresponding weights are unimportant since their distributions are highly peaked at zero. By pruning these weights (and as the consequence, the corresponding entries in the input auto-correlation matrix) sparse estimation is implemented. Since WSNs are typically characterized by limited communication capabilities and energy constraints, pruning of irrelevant weights would lead to a reduced communication load and lower energy requirements.

In [3], a distributed RLS scheme is presented, where each node in a sensor network has an access to input measurement data and a desired signal. The proposed algorithm works decentralized with a collaboration path through the network, shown in Figure 1. The aim is to estimate a global weight vector that is least-squares optimal over the errors of all nodes.

This work was supported by the FWF under Award S10604-N13 within the National Research Network SISE.

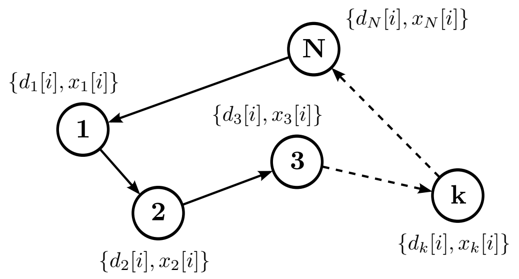

Fig. 1. A collaboration path through the network with N nodes.

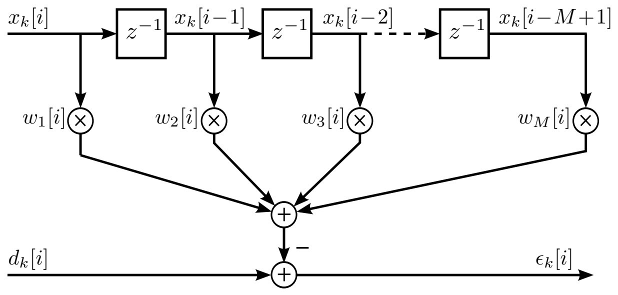

Fig. 2. An M-th order FIR filter at the node k with the input $x_k[i]$, the desired signal $d_k[i]$, and the node-error $\epsilon_k[i]$.

The error at each node is defined as the difference between the desired signal and a linear combination of the weights with the input data (see Fig. 2).

In our work, we present an algorithm that implements the sparse Bayesian estimation method in a distributed fashion. We show how to add regularization by assuming a prior distribution over the weights. We also show how to incorporate the decentralized computation of the hyper parameters into the spatio-temporal dRLS recursion. With this method it is possible to prune irrelevant parameters and reduce the computational complexity for the whole network.

II. Signal Model

With the filter order M and the number of sensor nodes N, we first define the global weight vector $\mathbf{w}[i] = [w_1[i], w_2[i], \ldots, w_M[i]]^T$ and the corresponding input data vectors $\mathbf{x}_k[i] = [x_k[i], x_k[i-1], \ldots, x_k[i-M+1]]^T$ at each of the nodes $k = 1, \ldots, N$. Furthermore, we introduce a linear-in-parameters model for the desired signal $d_k[i]$ at each node with additive Gaussian perturbation $\epsilon_k[i]$ according to Figure 2,

$$d_k[i] = \mathbf{w}^T[i]\mathbf{x}_k[i] + \epsilon_k[i]. \tag{1}$$

The whole network accesses the snapshot matrices

$$\tilde{\mathbf{D}}_i = [d_1[i], d_2[i], \dots, d_N[i]]^T \qquad (N \times 1)$$

and

$$\tilde{\mathbf{X}}_i = [\mathbf{x}_1[i], \mathbf{x}_2[i], \dots, \mathbf{x}_N[i]]^T \qquad (N \times M)$$

at time i. To collect all the data up to time instant i we define global matrices $\mathbf{D}_i$ and $\mathbf{X}_i$ with

$$\mathbf{D}_i = \left[\tilde{\mathbf{D}}_1^T, \tilde{\mathbf{D}}_2^T, \dots, \tilde{\mathbf{D}}_i^T\right]^T \qquad (i \cdot N \times 1)$$

and

$$\mathbf{X}_i = \left[\tilde{\mathbf{X}}_1^T, \tilde{\mathbf{X}}_2^T, \dots, \tilde{\mathbf{X}}_i^T\right]^T \qquad (i \cdot N \times M).$$

With omitted time index in $\mathbf{w}[i]$ for an uncluttered notation, we obtain the likelihood function

$$p(\mathbf{D}_i|\mathbf{w}) = (2\pi)^{-\frac{i \cdot N}{2}} |\mathbf{\Lambda}_i|^{\frac{1}{2}} \exp\left\{-\frac{1}{2}(\mathbf{D}_i - \mathbf{X}_i\mathbf{w})^T \mathbf{\Lambda}_i (\mathbf{D}_i - \mathbf{X}_i\mathbf{w})\right\}$$

over the weights based on the time-exponentially and spatially weighted model (1). The spatio-temporal diagonal weighting matrix is defined as

$$\mathbf{\Lambda}_i = \sigma^{-2} \operatorname{diag}\left([\lambda^{i-1}\mathbf{\Gamma}, \lambda^{i-2}\mathbf{\Gamma}, \dots, \lambda\mathbf{\Gamma}, \mathbf{\Gamma}]\right), \tag{2}$$

with forgetting factor $0 \ll \lambda \leq 1$, a diagonal spatial weighting matrix $\mathbf{\Gamma} = \operatorname{diag}(\gamma_1, \gamma_2, \dots, \gamma_N)$, the spatial weighting factors $\gamma_k \geq 0$ and σ^2 as the known noise variance.
To introduce regularization, we define a zero mean Gaussian prior over the weights

$$p(\mathbf{w}|\mathbf{A}) = (2\pi)^{-\frac{M}{2}} |\mathbf{A}|^{\frac{1}{2}} \exp\left\{-\frac{1}{2}\mathbf{w}^T\mathbf{A}\mathbf{w}\right\},$$

with diagonal matrix $\mathbf{A} = \operatorname{diag}([\alpha_1, \alpha_2, \dots, \alpha_M])$ that contains precision (inverse variance) hyperparameters α_j with $j = 1, \dots, M$ for each weight w_j.

III. Parameter Learning

A. Bayesian Estimation

By using Bayes' theorem we obtain the posterior over the weights given the data and the hyperparameters

$$p(\mathbf{w}|\mathbf{D}_i, \mathbf{A}) = \frac{p(\mathbf{D}_i|\mathbf{w})p(\mathbf{w}|\mathbf{A})}{\int p(\mathbf{D}_i|\mathbf{w})p(\mathbf{w}|\mathbf{A})d\mathbf{w}}. \tag{3}$$

The maximum a posteriori estimate (MAP) of $\mathbf{w}$ is obtained by minimizing the objective function

$$L(\mathbf{w}) = \frac{1}{2}(\mathbf{D}_i - \mathbf{X}_i\mathbf{w})^T \mathbf{\Lambda}_i (\mathbf{D}_i - \mathbf{X}_i\mathbf{w}) + \frac{1}{2}\mathbf{w}^T\mathbf{A}\mathbf{w},$$

which is equivalent to maximizing the numerator of (3) and results in

$$\hat{\mathbf{w}} = (\mathbf{X}_i^T\mathbf{\Lambda}_i\mathbf{X}_i + \mathbf{A})^{-1}\mathbf{X}_i^T\mathbf{\Lambda}_i\mathbf{D}_i\ , \tag{4}$$

where we define the regularized input auto-correlation matrix estimate with

$$\tilde{\mathbf{\Phi}} = \mathbf{X}_i^T\mathbf{\Lambda}_i\mathbf{X}_i + \mathbf{A} \tag{5}$$

and its inverse as $\tilde{\mathbf{P}} = \tilde{\mathbf{\Phi}}^{-1}$. Furthermore, the density of (3) can be computed and reads

$$p(\mathbf{w}|\mathbf{D}_i, \mathbf{A}) = (2\pi)^{-\frac{M}{2}} |\tilde{\mathbf{\Phi}}|^{\frac{1}{2}} \exp\left\{-\frac{1}{2}(\mathbf{w} - \hat{\mathbf{w}})^T \tilde{\mathbf{\Phi}} (\mathbf{w} - \hat{\mathbf{w}})\right\}.$$

B. Evidence Procedure

By ignoring the normalizing integral of Bayes' theorem, the posterior over $\mathbf{A}$ given the desired signals $\mathbf{D}_i$ is proportional to the marginal likelihood times a prior over $\mathbf{A}$, i.e.

$$p(\mathbf{A}|\mathbf{D}_i) \propto p(\mathbf{D}_i|\mathbf{A})p(\mathbf{A}). \tag{6}$$

Assuming $p(\mathbf{A})$ to be flat over a logarithmic scale (cf. [2]), the maximization of the posterior (6) is equivalent to maximizing the marginal-likelihood

$$p(\mathbf{D}_i|\mathbf{A}) = \int p(\mathbf{D}_i|\mathbf{w})p(\mathbf{w}|\mathbf{A})d\mathbf{w}. \tag{7}$$

Equation (7) is the normalizing integral in the weight posterior (3) and is often referred to as the marginal likelihood, since it is obtained by marginalizing over the weights. It is computable and given by

$$p(\mathbf{D}_i|\mathbf{A}) = \frac{|\mathbf{\Lambda}_i|^{\frac{1}{2}} |\mathbf{A}|^{\frac{1}{2}}}{(2\pi)^{\frac{i \cdot N}{2}} |\tilde{\mathbf{\Phi}}|^{\frac{1}{2}}} \exp\left\{-\frac{1}{2}\mathbf{D}_i^T\mathbf{\Lambda}_i\mathbf{D}_i + \frac{1}{2}\hat{\mathbf{w}}^T\tilde{\mathbf{\Phi}}\hat{\mathbf{w}}\right\}. \tag{8}$$

To maximize the marginal likelihood (8), we define a cost-function $L_e(\mathbf{A}) = -\ln p(\mathbf{D}_i|\mathbf{A})$, that is

$$L_e(\mathbf{A}) = \frac{1}{2}\Big(iN\ln(2\pi) - \ln|\mathbf{\Lambda}_i| - \ln|\mathbf{A}| + \ln|\tilde{\mathbf{\Phi}}| + \mathbf{D}_i^T\mathbf{\Lambda}_i\mathbf{D}_i - \hat{\mathbf{w}}^T\tilde{\mathbf{\Phi}}\hat{\mathbf{w}}\Big), \tag{9}$$

which we would like to minimize. To do so (cf. [1]), we make use of the identity $\frac{\partial}{\partial \mathbf{x}} \ln|\mathbf{B}| = \operatorname{Tr}(\mathbf{B}^{-1}\frac{\partial \mathbf{B}}{\partial \mathbf{x}})$ and the optimization condition $\frac{\partial L_e(\mathbf{A})}{\partial \alpha_j} = 0$, i.e.

$$\frac{1}{2}\operatorname{Tr}(\tilde{\mathbf{\Phi}}^{-1}\frac{\partial \tilde{\mathbf{\Phi}}}{\partial \alpha_j}) - \frac{1}{2}\operatorname{Tr}(\mathbf{A}^{-1}\frac{\partial \mathbf{A}}{\partial \alpha_j}) + \frac{1}{2}\hat{w}_j^2 = 0,$$

which for all $j = 1, \dots, M$ gets an explicit equation

$$\alpha_j = \frac{1}{\hat{w}_j^2 + \tilde{\mathbf{\Phi}}_{jj}^{-1}}$$

that could be iteratively reestimated at time instant i by using the following formulas during one estimation loop with index l until the objective-function (9) reaches a local minimum:

$$\left.\begin{aligned} \tilde{\mathbf{P}}^{(l)} &= (\mathbf{X}_i^T\mathbf{\Lambda}_i\mathbf{X}_i + \mathbf{A}^{(l)})^{-1} \\ \hat{\mathbf{w}}^{(l)} &= \tilde{\mathbf{P}}^{(l)}\mathbf{X}_i^T\mathbf{\Lambda}_i\mathbf{D}_i \\ \alpha_j^{(l+1)} &= \frac{1}{(\hat{w}_j^{(l)})^2 + \tilde{\mathbf{P}}_{jj}^{(l)}} \end{aligned}\right\} \forall\, j = 1, \dots, M. \tag{10}$$

C. Time and Space Recursion

To distribute (10) over time and space in an efficient way, we start by defining $\mathbf{\Phi} = \sigma^2 \tilde{\mathbf{\Phi}}$ for the input auto-correlation estimate update and reintroducing the time index i in our notation. From (5) we obtain

$$\mathbf{\Phi}[i] = \lambda \mathbf{\Phi}[i-1] + \tilde{\mathbf{X}}_i^T \mathbf{\Gamma} \tilde{\mathbf{X}}_i + \sigma^2 \mathbf{A}(1-\lambda), \tag{11}$$

where the last term in (11) corresponds to a full-rank update that does not allow for a fast recursive computation of the inverse $\mathbf{P}[i] = \sigma^{-2} \tilde{\mathbf{\Phi}}^{-1}[i]$. According to [4], it is enough to apply a rank-one update at time instant i and so we define pivot vectors $\mathbf{v}[i] = [0, 0, \ldots, 1, 0, \ldots, 0]^T$ with only one nonzero entry at position $q = 1 + i \bmod M$. With $\kappa[i] = \sigma^2(1-\lambda)\alpha_q M$ we can rewrite (11) with two equations

$$\mathbf{\Phi}[i] = \breve{\mathbf{\Phi}}[i] + \tilde{\mathbf{X}}_i^T \mathbf{\Gamma} \tilde{\mathbf{X}}_i \tag{12}$$

and

$$\breve{\mathbf{\Phi}}[i] = \lambda \mathbf{\Phi}[i-1] + \kappa[i] \mathbf{v}[i] \mathbf{v}^T[i]. \tag{13}$$

Equation (12) still has no fast recursive structure for the computation of its inverse, but since the term $\tilde{\mathbf{X}}_i^T \mathbf{\Gamma} \tilde{\mathbf{X}}_i$ represents the spatially weighted inputs of the whole network at time i, it can be separated over the network, which leads to a rank-one update at each node. The covariance matrix estimate $\mathbf{P}_k[i]$ at node k can be computed as follows:

$$\begin{aligned}
\breve{\mathbf{P}}[i] &= (\lambda \mathbf{P}^{-1}[i-1] + \kappa[i]\mathbf{v}[i]\mathbf{v}^T[i])^{-1} \\
\mathbf{P}_0[i] &\leftarrow \breve{\mathbf{P}}[i] \\
\mathbf{P}_1[i] &= (\mathbf{P}_0^{-1}[i] + \gamma_1 \mathbf{x}_1[i]\mathbf{x}_1^T[i])^{-1} \\
\mathbf{P}_2[i] &= (\mathbf{P}_1^{-1}[i] + \gamma_2 \mathbf{x}_2[i]\mathbf{x}_2^T[i])^{-1} \\
&\vdots \\
\mathbf{P}_N[i] &= (\mathbf{P}_{N-1}^{-1}[i] + \gamma_N \mathbf{x}_N[i]\mathbf{x}_N^T[i])^{-1} \\
\mathbf{P}[i] &\leftarrow \mathbf{P}_N[i]
\end{aligned} \tag{14}$$

The first node in the serial network has to perform extra computations according to the first line in (14). Using the matrix inversion lemma for rank-one updates, finally gives

$$\begin{cases}
\breve{\mathbf{P}}[i] = \left[\mathbf{I} - \frac{\kappa[i]\mathbf{P}[i-1]\mathbf{v}[i]\mathbf{v}^T[i]}{\lambda + \kappa[i]\mathbf{v}^T[i]\mathbf{P}[i-1]\mathbf{v}[i]}\right] \lambda^{-1}\mathbf{P}[i-1] \\
\mathbf{P}_0[i] \leftarrow \breve{\mathbf{P}}[i] \\
\text{for } k = 1:N \\
\quad \mathbf{P}_k[i] = \mathbf{P}_{k-1}[i] - \frac{\mathbf{P}_{k-1}[i]\mathbf{x}_k[i]\mathbf{x}_k^T[i]\mathbf{P}_{k-1}[i]}{\gamma_k^{-1} + \mathbf{x}_k^T[i]\mathbf{P}_{k-1}[i]\mathbf{x}_k[i]} \\
\text{end} \\
\mathbf{P}[i] \leftarrow \mathbf{P}_N[i].
\end{cases} \tag{15}$$

We now would like to investigate the update of the weights on each node of the network and define global matrices up to time i and node k as

$$\mathbf{D}_{i,k} = \left[\mathbf{D}_{i-1}^T, d_1[i], d_2[i], \ldots, d_k[i]\right]^T$$

and

$$\mathbf{X}_{i,k} = \left[\mathbf{X}_{i-1}^T, \mathbf{x}_1[i], \mathbf{x}_2[i], \ldots, , \mathbf{x}_k[i]\right]^T.$$

Additionally, we also define the diagonal spatial weighting matrix $\mathbf{\Gamma}_k = \operatorname{diag}([\gamma_1, \gamma_2, \ldots, \gamma_k])$ up to node k and the corresponding spatio-temporal weighting matrix

$$\mathbf{\Lambda}_{i,k} = \operatorname{diag}\left([\lambda \mathbf{\Lambda}_{i-1}, \sigma^{-2}\mathbf{\Gamma}_k]\right), \tag{16}$$

where one should note that (16) can be rewritten $\mathbf{\Lambda}_{i,k} = \operatorname{diag}([\mathbf{\Lambda}_{i,k-1}, \sigma^{-2}\gamma_k])$ and from (2) we know that $\mathbf{\Lambda}_i = \mathbf{\Lambda}_{i,N}$. Starting with the MAP estimate (4), we define the regularized least-squares solution over the weights at node k by

$$\begin{aligned}
\boldsymbol{\psi}_k[i] &= \sigma^2 \mathbf{P}_k[i] \mathbf{X}_{i,k}^T \mathbf{\Lambda}_{i,k} \mathbf{D}_{i,k} \\
&= \sigma^2 \mathbf{P}_k[i] \Big(\mathbf{X}_{i,k-1}^T \mathbf{\Lambda}_{i,k-1} \mathbf{D}_{i,k-1} \\
&\quad + \sigma^{-2}\gamma_k \mathbf{x}_k[i] d_k[i] \Big)
\end{aligned} \tag{17}$$

with $\hat{\mathbf{w}}[i] = \boldsymbol{\psi}_N[i]$. We can rewrite the local update of the covariance matrix estimate defined in (15) as $\mathbf{P}_k[i] = \big(\mathbf{I} - \mathbf{c}_k[i]\mathbf{x}_k^T[i]\big)\mathbf{P}_{k-1}[i]$ where we define a Kalman gain vector

$$\mathbf{c}_k[i] = \frac{\mathbf{P}_{k-1}[i]\mathbf{x}_k[i]}{\gamma_k^{-1} + \mathbf{x}_k^T[i]\mathbf{P}_{k-1}[i]\mathbf{x}_k[i]}. \tag{18}$$

Using (17), the local weight estimate recursion can be found by

$$\begin{aligned}
\boldsymbol{\psi}_k[i] = &\underbrace{\sigma^2 \mathbf{P}_{k-1}[i] \mathbf{X}_{i,k-1}^T \mathbf{\Lambda}_{i,k-1} \mathbf{D}_{i,k-1}}_{\boldsymbol{\psi}_{k-1}[i]} \\
&- \mathbf{c}_k[i]\mathbf{x}_k^T[i] \underbrace{\sigma^2 \mathbf{P}_{k-1}[i] \mathbf{X}_{i,k-1}^T \mathbf{\Lambda}_{i,k-1} \mathbf{D}_{i,k-1}}_{\boldsymbol{\psi}_{k-1}[i]} \\
&- \mathbf{c}_k[i]\mathbf{x}_k^T[i]\mathbf{P}_{k-1}[i]\gamma_k \mathbf{x}_k[i] d_k[i] \\
&+ \mathbf{P}_{k-1}[i]\gamma_k \mathbf{x}_k[i] d_k[i],
\end{aligned} \tag{19}$$

from which follows that

$$\boldsymbol{\psi}_k[i] = \boldsymbol{\psi}_{k-1}[i] + \mathbf{c}_k[i] e_k[i], \tag{20}$$

where we have plugged in a reordered version of (18) for the last term of (19) and have introduced the local a-priori output error $e_k[i] = \big(d_k[i] - \mathbf{x}_k^T[i]\boldsymbol{\psi}_{k-1}[i]\big)$.

To obtain the recursion for the weights between different time-points i we note that $\mathbf{X}_{i,0}^T \mathbf{\Lambda}_{i,0} \mathbf{D}_{i,0} = \lambda \mathbf{X}_{i-1}^T \mathbf{\Lambda}_{i-1} \mathbf{D}_{i-1}$ and make use of Equation (17) at the first node $k = 1$. By defining another Kalman gain vector as

$$\breve{\mathbf{c}}[i] = \frac{\kappa[i]\mathbf{P}[i-1]\mathbf{v}[i]}{\lambda + \kappa[i]\mathbf{v}^T[i]\mathbf{P}[i-1]\mathbf{v}[i]}$$

and using the first line in (15) together with Equation (4) leads to an update of the form (20)

$$\begin{aligned}
\boldsymbol{\psi}_1[i] = &\underbrace{(\mathbf{I} - \breve{\mathbf{c}}[i]\mathbf{v}^T[i])\hat{\mathbf{w}}[i-1]}_{\boldsymbol{\psi}_0[i]} \\
&+ \mathbf{c}_1[i] \left(d_1[i] - \mathbf{x}_1^T[i] \underbrace{(\mathbf{I} - \breve{\mathbf{c}}[i]\mathbf{v}^T[i])\hat{\mathbf{w}}[i-1]}_{\boldsymbol{\psi}_0[i]} \right).
\end{aligned}$$

Algorithm 1 Pseudo-code for the sparse distributed RLS algorithm with hyperparameter-update using evidence procedure

```
if $i \bmod M_A = 0$
    $\alpha_j[i] = \frac{1}{\hat{w}_j[i-1]^2 + \sigma^2 \mathbf{P}_{jj}[i-1]}$, $\forall j$
    $(\hat{\mathbf{w}}, \mathbf{P}, \mathbf{A}, M) = \mathrm{prune}(\hat{\mathbf{w}}, \mathbf{P}, \mathbf{A}, B)$
else
    $\alpha_j[i] = \alpha_j[i-1]$, $\forall j = 1, \ldots, M$
end
$q = 1 + i \bmod M$
Generate $\mathbf{v}[i]$ with '1' at q
$\kappa[i] = \sigma^2 (1-\lambda) \alpha_q[i] M$
$\breve{\mathbf{c}}[i] = \frac{\kappa[i]\mathbf{P}[i-1]\mathbf{v}[i]}{\lambda + \kappa[i]\mathbf{v}^T[i]\mathbf{P}[i-1]\mathbf{v}[i]}$
$\boldsymbol{\psi}_0[i] \leftarrow \left(\mathbf{I} - \breve{\mathbf{c}}[i]\mathbf{v}^T[i]\right)\hat{\mathbf{w}}[i-1]$
$\mathbf{P}_0[i] \leftarrow \left(\mathbf{I} - \breve{\mathbf{c}}[i]\mathbf{v}^T[i]\right)\lambda^{-1}\mathbf{P}[i-1]$
for $k = 1 : N$
    $e_k[i] = d_k[i] - \mathbf{x}_k^T[i]\boldsymbol{\psi}_{k-1}[i]$
    $\mathbf{c}_k[i] = \frac{\mathbf{P}_{k-1}[i]\mathbf{x}_k[i]}{\gamma_k^{-1} + \mathbf{x}_k^T[i]\mathbf{P}_{k-1}[i]\mathbf{x}_k[i]}$
    $\boldsymbol{\psi}_k[i] = \boldsymbol{\psi}_{k-1}[i] + \mathbf{c}_k[i]e_k[i]$
    $\mathbf{P}_k[i] = \mathbf{P}_{k-1}[i] - \mathbf{c}_k[i]\mathbf{x}_k^T[i]\mathbf{P}_{k-1}[i]$
end
$\hat{\mathbf{w}}[i] \leftarrow \boldsymbol{\psi}_N[i]$;   $\mathbf{P}[i] \leftarrow \mathbf{P}_N[i]$.
```

Based on the previous result for $\hat{\mathbf{w}}$ at time $i-1$, we have defined another extra computation to be performed by the first node in the serial network, namely

$$\boldsymbol{\psi}_0[i] = (\mathbf{I} - \breve{\mathbf{c}}[i]\mathbf{v}^T[i])\hat{\mathbf{w}}[i-1].$$

D. Pseudo-Code of the Algorithm

Algorithm 1 shows the pseudo-code for the proposed sparse distributed RLS scheme with integrated weight pruning according to [1]. The function $\mathrm{prune}(.)$ reduces the number of coefficients in the vector $\hat{\mathbf{w}}$ if the corresponding α_j (diagonal elements of $\mathbf{A}$) exceed a given threshold B. According to the weights, also the appropriate elements in $\mathbf{A}$ and $\boldsymbol{\Phi}$ have to be pruned. Note that in the efficient recursive implementation we do not calculate $\boldsymbol{\Phi}$ directly, but instead use its inverse $\mathbf{P}$. To further avoid the use of the computationally intensive matrix-invertion, [1] described an efficient method that only accesses the covariance matrix estimate $\mathbf{P}$.
Because we incorporated a rank-1 instead of a full-rank update in (13), we have to update the hyperparameters at multiples of M cycles and therefore define $M_A = mM$ with $m \in \mathbb{N}^+$.

IV. Conclusion

We proposed an automatic regularization method for a distributed RLS algorithm. By using Bayesian evidence procedure, we presented a method to obtain the regularization parameters which contain relevance information for the corresponding weights. Furthermore, we showed a distributed algorithm that prunes irrelevant weights from the network and thus reduces the communication load between neighboring nodes.

References

[1] H. Koeppl, G. Kubin and G. Paoli, *Bayesian Methods for Sparse RLS Adaptive Filters*, 2003.

[2] Michael E. Tipping, *Sparse Bayesian Learning and the Relevance Vector Machine*, 2001.

[3] Ali H. Sayed and Cassio G. Lopes, *Distributed Recursive Least-Squares Strategies over Adaptive Networks*, 2006.

[4] S. L. Gay, *Dynamically regularized fast RLS with application to echo cancellation*, 1996.