A Fixed-Lag Particle Smoother for Blind SISO Equalization of Time-Varying Channels

Alberto Guimarães, Boujemaa Ait-El-Fquih and François Desbouvries

***Abstract*—We introduce a new sequential importance sampling (SIS) algorithm which propagates in time a Monte Carlo approximation of the posterior fixed-lag smoothing distribution of the symbols under doubly-selective channels. We perform an exact evaluation of the optimal importance distribution, at a reduced computational cost when compared to other optimal solutions proposed for the same state-space model. The method is applied as a soft input–soft output (SISO) blind equalizer in a turbo receiver framework and simulation results are obtained to show its outstanding BER performance.**

***Index Terms*—Fixed-lag smoothing, particle filter, mixture Kalman filter, SISO equalization.**

I. Introduction

BAYESIAN restoration in Conditionally Gaussian Linear State-Space Models (CGLSSM) has received much attention recently [1], particularly in the context of blind equalization problems. Among other solutions, Mixture Kalman Filter (MKF) techniques embed Kalman Filter (KF) recursions in a SIS framework [2]. A smoothing solution for the case of blind SISO equalization with static channels has been proposed in [3]. For time-variant channels, a particular fixed-lag smoothing MKF was studied in [4] and [5]; however, although being a near-optimal solution, the algorithm was not implemented because of a computational cost exponential in the smoothing lag and other parameters[1]. The aforementioned works then overcome this constraint by introducing further Monte Carlo sampling steps, but at the expense of algorithm performance.

By contrast, our method presents a novel solution for the “optimal” approach, which under typical application scenarios is remarkably less complex than the alternative proposed so far. Although this approach is unavoidably constrained by the exponential growth of complexity, it presents superior performance than the suboptimal solutions.

We next apply our method to SISO blind equalization of doubly-selective channels in a turbo equalization setup. Turbo equalizers [6] have already been proposed for doubly-selective channels with blind algorithms performing channel estimation jointly with equalization [7][8]. Computational complexity is a severe issue for such approach, however the remarkable tracking performance of our optimal Bayesian estimator, even in the presence of fast channel variations, enables us to implement a solution in which our SIS-based equalizer is used only in the very first iterations of the turbo loop, and then substituted in subsequent iterations by an equalizer with a reduced computational load. This hybrid solution is validated by simulation results. The rest of the letter is organized as follows. In section II we first describe our model. Our fixed-lag particle smoothing (FLPS) algorithm is developed in section III. In section IV we show the computational load of our method and we evaluate through simulations the Bit-Error Rate performance of a turbo receiver including our algorithm.

II. State-space Model

Let $\mathbf{x}^T$ (resp. $\mathbf{x}^H$) denote the transpose (resp. Hermitian transpose) of vector $\mathbf{x}$. We assume that the receiver input y_n is related to the transmitted complex symbols by

$$y_n = \mathbf{x}_n^T \mathbf{h}_n + \omega_n \tag{1}$$

where $\mathbf{h}_n = [h_{0,n}, \ldots, h_{L-1,n}]^T$ represents the baseband channel impulse response of finite length L, and $\mathbf{x}_n = [x_n, x_{n-1}, \ldots, x_{n-L+1}]^T$ gathers the transmitted symbols from time $n-L+1$ up to time n. The sequence x_n are differentially modulated to resolve phase ambiguity, hence $\{x_n\}$ is a Markov Chain (MC) and from (1) we get[2] $p(x_n|\mathbf{x}_{1:n-1}, \mathbf{y}_{1:n-1}) = p(x_n|x_{n-1})$, and we assume that $p(x_n|x_{n-1})$ is known for all n. The noise variables $\{\omega_n\}$ are independent circularly symmetric complex Gaussian variables with zero mean and known variance Λ_n^{ω}, and are independent of $\{x_n\}$. For notational convenience we consider in this letter the case where y_n is scalar, but the extension of our algorithm to Multiple Input Multiple Output (MIMO) systems is analytically straightforward, although the computational complexity increases exponentially with the number of input streams. Finally $\mathbf{h}_n$ propagates according to

$$\mathbf{h}_{n+1} = \mathbf{F}_n \mathbf{h}_n + \mathbf{v}_n \ , \tag{2}$$

where $\{\mathbf{v}_n\}$ are complex independent variables, independent of $\{x_n\}$, of $\{\omega_n\}$ and of $\mathbf{h}_0$. It is also assumed that $\mathbf{v}_n \sim \mathcal{N}(\mathbf{0}, \mathbf{\Lambda}_n^{\mathbf{v}})$, $\mathbf{h}_0 \sim \mathcal{N}(\mathbf{0}, \mathbf{\Lambda}_0^{\mathbf{h}})$, and that $\mathbf{F}_n$, $\mathbf{\Lambda}_n^{\mathbf{v}}$ and $\mathbf{\Lambda}_0^{\mathbf{h}}$ are known.

III. An FLPS Algorithm

In this section we focus on the computation of the probability mass function (pmf) $p(x_n|\mathbf{y}_{1:n+M})$, where $M > 0$ is some fixed delay. We resort to an SIS approximation (see e.g. [9][10] and references therein). So the a posteriori joint pmf of symbols $\mathbf{x}_{1:n-1}$ at time $n-1$ is approximated by $p(\mathbf{x}_{1:n-1}|\mathbf{y}_{1:n+M-1}) \approx \sum_{i=1}^{N} \lambda_{n-1}^{i} \delta(\mathbf{x}_{1:n-1} -$

Manuscript received December 24, 2008; revised July 7, 2009 and October 27, 2009; accepted November 13, 2009. The associate editor coordinating the review of this letter and approving it for publication was Dr. G. Vitetta.

A. Guimarães is with Instituto Militar de Engenharia, Rio de Janeiro, Brazil (e-mail: agaspar@ime.eb.br).

B. Ait-El-Fquih and F. Desbouvries are with Institut Telecom, Telecom SudParis, CITI dpt. and CNRS UMR 5157, 91011, Evry, France (e-mail: b_aitelfquih@yahoo.fr, Francois.Desbouvries@it-sudparis.eu).

Digital Object Identifier 10.1109/TWC.2009.081694.

[1]In [5] we refer to the *DSIS* method with optimal importance point mass function.

[2]Depending on the context, let $\mathbf{u}_{i:j}$ denote either $\{u_i, \cdots, u_j\}$ or the vector with components $u_k, i \leq k \leq j$.

$\mathbf{x}^i_{1:n-1})$, in which the samples $\mathbf{x}^i_{1:n-1}$ are generated from an importance distribution $q(\mathbf{x}_{1:n-1}|\mathbf{y}_{1:n+M-1})$, and the importance weight λ^i_{n-1} associated to the i-th trajectory $\mathbf{x}^i_{1:n-1}$ is given by $\lambda^i_{n-1} \propto \frac{p(\mathbf{x}^i_{1:n-1}|\mathbf{y}_{1:n+M-1})}{q(\mathbf{x}^i_{1:n-1}|\mathbf{y}_{1:n+M-1})}$, $\sum_{i=1}^N \lambda^i_{n-1} = 1$. If we assume that $q(\mathbf{x}_{1:n}|\mathbf{y}_{1:n+M}) = q(x_n|\mathbf{x}_{1:n-1}, \mathbf{y}_{1:n+M})q(\mathbf{x}_{1:n-1}|\mathbf{y}_{1:n+M-1})$, then the weights can be computed recursively as

$$\lambda^i_n \propto \frac{p(x^i_n|\mathbf{x}^i_{1:n-1}, \mathbf{y}_{1:n+M})p(y_{n+M}|\mathbf{x}^i_{1:n-1}, \mathbf{y}_{1:n+M-1})}{q(x^i_n|\mathbf{x}^i_{1:n-1}, \mathbf{y}_{1:n+M})} \times \underbrace{\frac{p(\mathbf{x}^i_{1:n-1}|\mathbf{y}_{1:n+M-1})}{q(\mathbf{x}^i_{1:n-1}|\mathbf{y}_{1:n+M-1})}}_{\propto \lambda^i_{n-1}}, \tag{3}$$

where $\mathbf{x}^i_{1:n} = [\mathbf{x}^i_{1:n-1}, x^i_n]$ and each particle x^i_n is drawn from the conditional importance distribution (CID) $q(x_n|\mathbf{x}^i_{1:n-1}, \mathbf{y}_{1:n+M})$. Finally $\sum_{i=1}^N \lambda^i_n \delta(\mathbf{x}_{1:n} - \mathbf{x}^i_{1:n})$ approximates $p(\mathbf{x}_{1:n}|\mathbf{y}_{1:n+M})$, and therefore $\sum_{i=1}^N \lambda^i_n \delta(x_n - x^i_n)$ approximates $p(x_n|\mathbf{y}_{1:n+M})$. Now, SIS algorithms are well known to suffer from weights degeneracy. Classical rescues consist in resampling from $\sum_{i=1}^N \lambda^i_n \delta(x_n - x^i_n)$ (either systematically or according to some strategy) and in choosing CID $q(x_n|\mathbf{x}^i_{1:n-1}, \mathbf{y}_{1:n+M})$ carefully. To that respect, one good choice is to sample particles from the "optimal" CID (see [10]), i.e. the distribution which minimizes the variance of the importance weights, conditionally on the observations and past samples. In our case, the optimal distribution reads

$$q^{opt}(x_n|\mathbf{x}^i_{1:n-1}, \mathbf{y}_{1:n+M}) = p(x_n|\mathbf{x}^i_{1:n-1}, \mathbf{y}_{1:n+M}), \tag{4}$$

and under that choice (3) becomes

$$\lambda^i_n \propto \underbrace{p(y_{n+M}|\mathbf{x}^i_{1:n-1}, \mathbf{y}_{1:n+M-1})}_{\tilde{\lambda}^i_n} \lambda^i_{n-1}. \tag{5}$$

From now on, we thus focus on the computation of (4) and of factor $\tilde{\lambda}^i_n$ in (5).

A. Computing the Optimal CID

Let us address (4). For each n and i, we should sample a new particle x^i_n according to

$$p(x_n|\mathbf{x}^i_{1:n-1}, \mathbf{y}_{1:n+M}) = \frac{p(x_n|x^i_{n-1})p(\mathbf{y}_{n:n+M}|x_n, \mathbf{x}^i_{1:n-1}, \mathbf{y}_{1:n-1})}{\sum_{x_n} p(x_n|x^i_{n-1})p(\mathbf{y}_{n:n+M}|x_n, \mathbf{x}^i_{1:n-1}, \mathbf{y}_{1:n-1})}; \tag{6}$$

$p(x_n|x^i_{n-1})$ is known, so it remains to compute $p(\mathbf{y}_{n:n+M}|x_n, \mathbf{x}^i_{1:n-1}, \mathbf{y}_{1:n-1})$, which can be written as

$$p(\mathbf{y}_{n:n+M}|x_n, \mathbf{x}^i_{1:n-1}, \mathbf{y}_{1:n-1}) = \sum_{x_{n+1}} .. \sum_{x_{n+M}} p(\mathbf{y}_{n:n+M}|\boldsymbol{\theta}^i_{\mathbf{x}_{n:n+M}}) \prod_{k=1}^{M} p(x_{n+k}|x_{n+k-1}) \tag{7}$$

with $\boldsymbol{\theta}^i_{\mathbf{x}_{n:n+M}} \overset{def}{=} (\mathbf{x}^i_{1:n-1}, \mathbf{x}_{n:n+M}, \mathbf{y}_{1:n-1}) \overset{def}{=} (\tilde{\mathbf{x}}_{1:n+M}, \mathbf{y}_{1:n-1})$, i.e., we set $\tilde{x}_k = x^i_k$ if $k \leq n-1$ and $\tilde{x}_k = x_k$ if $n \leq k \leq n+M$. Let also $\tilde{\mathbf{x}}_n = [\tilde{x}_n, \tilde{x}_{n-1}, \ldots, \tilde{x}_{n-L+1}]^T$. One can show that $p(\mathbf{y}_{n:n+M}|\boldsymbol{\theta}^i_{\mathbf{x}_{n:n+M}})$ in (7) is the value at point $\mathbf{y}_{n:n+M}$ of a Gaussian density. Let us thus set $p(\mathbf{y}_{n:n+M}|\boldsymbol{\theta}^i_{\mathbf{x}_{n:n+M}}) = \mathcal{N}(\mathbf{y}_{n:n+M}; \boldsymbol{\mu}^{yi}_{M+1}, \boldsymbol{\Sigma}^{yi}_{M+1})$, where $\mathcal{N}(\ \cdot\ ; \boldsymbol{\mu}, \boldsymbol{\Sigma})$ stands for a circular complex Gaussian probability density function with parameters $(\boldsymbol{\mu}, \boldsymbol{\Sigma})$. In practice, parameters $\boldsymbol{\mu}^{yi}_{M+1}$ and $\boldsymbol{\Sigma}^{yi}_{M+1}$ can be computed recursively, and it has proven advantageous (as far as computational cost is concerned) to compute $p(\mathbf{y}_{n:n+M}|\boldsymbol{\theta}^i_{\mathbf{x}_{n:n+M}})$ recursively as well. This makes the difference between the calculus routine developed here and those considered in [4] and [5]. Let us set $p(\mathbf{y}_{n:n+k-1}, \mathbf{h}_{n+k}|\boldsymbol{\theta}^i_{\mathbf{x}_{n:n+M}}) = \mathcal{N}((\mathbf{y}_{n:n+k-1}, \mathbf{h}_{n+k}); \boldsymbol{\mu}^i_k, \boldsymbol{\Sigma}^i_k)$ with $\boldsymbol{\Sigma}^i_k = \begin{bmatrix} \boldsymbol{\Sigma}^{yi}_k & \boldsymbol{\Sigma}^{yhi}_k \\ \boldsymbol{\Sigma}^{hyi}_k & \boldsymbol{\Sigma}^{hi}_k \end{bmatrix}$ and $\boldsymbol{\mu}^i_k = \begin{bmatrix} \boldsymbol{\mu}^{yi}_k \\ \boldsymbol{\mu}^{hi}_k \end{bmatrix}$, and let us denote the quadratic form in $p(\mathbf{y}_{n:n+k-1}|\boldsymbol{\theta}^i_{\mathbf{x}_{n:n+M}})$ by $QF_{(k)} = (\mathbf{y}_{n:n+k-1} - \boldsymbol{\mu}^{yi}_k)^H(\boldsymbol{\Sigma}^{yi}_k)^{-1}(\mathbf{y}_{n:n+k-1} - \boldsymbol{\mu}^{yi}_k)$. Let $diag(\ )$ denote a block-diagonal matrix and $\mathbf{I}_{k-1}$ the $(k-1) \times (k-1)$ identity matrix. Finally our algorithm (see the Appendix) is as follows:

1) *Recursive computation of $\boldsymbol{\mu}^i_M$ and $\boldsymbol{\Sigma}^i_M$.*
   - Compute parameters of $p(\mathbf{h}_n|\mathbf{x}^i_{1:n-1}, \mathbf{y}_{1:n-1}) = \mathcal{N}(\mathbf{h}_n; \widehat{\mathbf{h}}^i_{n|n-1}, \boldsymbol{\Lambda}^{\mathbf{h}^i}_{n|n-1})$ by the KF;
   - Compute

$$\begin{aligned} \boldsymbol{\mu}^i_1 &= \begin{bmatrix} \tilde{\mathbf{x}}^T_n \\ \mathbf{F}_n \end{bmatrix} \widehat{\mathbf{h}}^i_{n|n-1} \\ \boldsymbol{\Sigma}^i_1 &= \begin{bmatrix} \tilde{\mathbf{x}}^T_n \\ \mathbf{F}_n \end{bmatrix} \boldsymbol{\Lambda}^{\mathbf{h}^i}_{n|n-1} \begin{bmatrix} \tilde{\mathbf{x}}^T_n \\ \mathbf{F}_n \end{bmatrix}^H + \begin{bmatrix} \Lambda^\omega_n & \mathbf{0} \\ \mathbf{0} & \boldsymbol{\Lambda}^{\mathbf{v}}_n \end{bmatrix}; \end{aligned} \tag{8}$$

   - *Recursion $(k-1) \rightarrow k$, for all $k = 2, \cdots, M$.* Compute

$$\begin{aligned} \boldsymbol{\mu}^i_k &= \mathbf{A}^i_{k-1}\boldsymbol{\mu}^i_{k-1} \\ \boldsymbol{\Sigma}^i_k &= \mathbf{A}^i_{k-1}\boldsymbol{\Sigma}^i_{k-1}(\mathbf{A}^i_{k-1})^H + \\ &\quad diag(\mathbf{0}_{k-1}, \Lambda^w_{n+k-1}, \boldsymbol{\Lambda}^{\mathbf{v}}_{n+k-1}) \end{aligned} \tag{9}$$

   with

$$\mathbf{A}^i_{k-1} = \begin{bmatrix} \mathbf{I}_{k-1} & \mathbf{0} \\ \mathbf{0} & \tilde{\mathbf{x}}^T_{n+k-1} \\ \mathbf{0} & \mathbf{F}_{n+k-1} \end{bmatrix}.$$

2) *Recursive computation of $p(\mathbf{y}_{n:n+M}|\boldsymbol{\theta}^i_{\mathbf{x}_{n:n+M}}) = \mathcal{N}(\mathbf{y}_{n:n+M}; \boldsymbol{\mu}^{yi}_{M+1}, \boldsymbol{\Sigma}^{yi}_{M+1})$.*
   - Compute $(\Sigma^{yi}_1)^{-1} = ([1 \quad 0]\boldsymbol{\Sigma}^i_1[1 \quad 0]^T)^{-1}$ and $QF_{(1)} = (y_n - \mu^{yi}_1)^T(\Sigma^{yi}_1)^{-1}(y_n - \mu^{yi}_1)$;
   - *Recursion $(k-1) \rightarrow k$, for $k = 2, \cdots, M+1$.* Compute

$$(\boldsymbol{\Sigma}^{yi}_k)^{-1} = \begin{bmatrix} \boldsymbol{\alpha} & \boldsymbol{\beta} \\ \boldsymbol{\beta}^T & \gamma \end{bmatrix}, \tag{10}$$

$$\det \boldsymbol{\Sigma}^{yi}_k = \gamma^{-1} \times \det \boldsymbol{\Sigma}^{yi}_{k-1}, \tag{11}$$

$$QF_{(k)} = QF_{(k-1)} + \mathcal{Y}^2\gamma, \tag{12}$$

where $\boldsymbol{\alpha} = (\boldsymbol{\Sigma}^{yi}_{k-1})^{-1} - (\boldsymbol{\Sigma}^{yi}_{k-1})^{-1}\mathbf{b}[\mathbf{b}^T(\boldsymbol{\Sigma}^{yi}_{k-1})^{-1}\mathbf{b} - D]^{-1}\mathbf{b}^T(\boldsymbol{\Sigma}^{yi}_{k-1})^{-1}$; $\boldsymbol{\beta} = -\boldsymbol{\alpha}\mathbf{b}D^{-1}$ and $\gamma = [D - \mathbf{b}^T(\Sigma^{yi}_{k-1})^{-1}\mathbf{b}]^{-1}$, with $\mathbf{b} = \boldsymbol{\Sigma}^{yhi}_{k-1}\mathbf{x}_{n+k-1}$ and $D = \mathbf{x}^T_{n+k-1}(\boldsymbol{\Sigma}^{hi}_{k-1})\mathbf{x}_{n+k-1} + \Lambda^w_{n+k-1}$; and $\mathcal{Y} = (y_{n+k} - \mathbf{x}^T_{n+k-1}\boldsymbol{\mu}^{hi}_{k-1}) - (\mathbf{y}_{n:n+k-1} - \boldsymbol{\mu}^{yi}_{k-1})^T(\boldsymbol{\Sigma}^{yi}_{k-1})^{-1}\mathbf{b}$.

B. Updating the Importance Weights

We now address the computation of $\tilde{\lambda}_n^i$ in (5). One can see easily that

$$\tilde{\lambda}_n^i = \frac{p(\mathbf{y}_{n:n+M}|\mathbf{x}_{1:n-1}^i, \mathbf{y}_{1:n-1})}{p(\mathbf{y}_{n:n+M-1}|\mathbf{x}_{1:n-1}^i, \mathbf{y}_{1:n-1})}. \quad (13)$$

The numerator of (13) is equal to the denominator of (6) which has been computed before. The denominator of (13) is the value of a Gaussian density $\mathcal{N}(\ \cdot\ ; \boldsymbol{\mu}_M^{yi}\ , \boldsymbol{\Sigma}_M^{yi})$ at point $\mathbf{y}_{n:n+M-1}$. Its determinant $\det \boldsymbol{\Sigma}_M^{yi}$ and quadratic form $QF_{(M)}$ have been computed at step $(k = M)$ of (11) and (12) . So the computation of $\tilde{\lambda}_n^i$ follows directly from the evaluation of the CID in (6).

IV. Performance Results

A. Computational Load

We first compare our FLPS algorithm with that of the algorithm in [5] with optimal importance pmf (*DSIS Optimal*). The main difference relies in the computation of (7) : in [5] one writes $p(\mathbf{y}_{n:n+M}|\mathbf{x}_{1:n+M}, \mathbf{y}_{1:n-1}) = \prod_{k=0}^{M} p(y_{n+k}|\mathbf{x}_{1:n+M}, \mathbf{y}_{1:n+k-1})$, so if $\mathcal{X}$ denotes symbol alphabet size one needs to implement $(M+1)\mathcal{X}^M$ KF. The number of floating operations (flops) required by our method to evaluate (4) at time n, for a given trajectory i, is approximately

$$N_1 = \frac{3}{2}ML^3 + \left(\frac{M^2 - M}{2} + M\right)L^2 + \mathcal{X}^M M\left[\frac{M+3}{2}L^2\left(\frac{M}{2}+1\right)L + \frac{5}{6}M^2 + 2M + \frac{53}{12}\right].$$

On the other hand, the number of flops required by the method of [5] is approximately

$$N_2 = \mathcal{X}^M \times (M+1)\left[\frac{3}{2}L^3 + 3L^2 + \frac{7}{2}L + 3\right]. \quad (14)$$

In Fig. 1 N_1 and N_2 are plotted for $M = 3$, $\mathcal{X} = 2$ or 4, and $L = 1$ to 5. We see above that the computational complexity of both proposals increases exponentially with M, but in all scenarios $N_1 < N_2$ and the difference significantly increases as L increases[3].

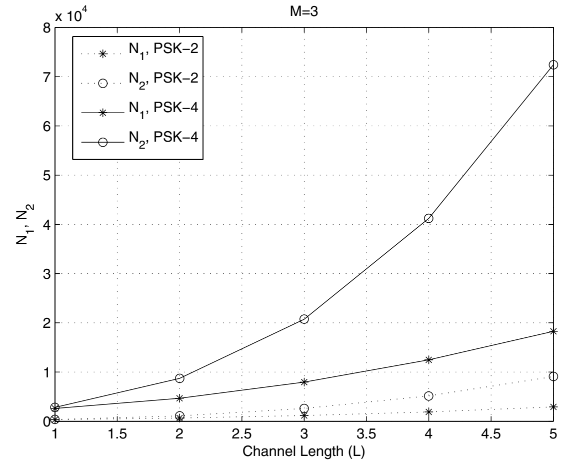

Fig. 1. Comparison in terms of number of floating operations between our proposal and the method of [4],[5].

B. BER Performance

The following simulation setup is considered to evaluate the performance of our algorithm used as a SISO equalizer embedded in a turbo equalization receiver. A set of 80 random data bits is encoded using a 1/2-rate $(1+D^2, 1+D+D^2)$ convolutional encoder. Next the coded bits $\{c_n\}$ are interleaved, mapped to ±1 symbols (BPSK), and transmitted over a channel with dynamics given by (2) with $\mathbf{F}_n = \kappa^{1/2}\mathbf{I}_2$, $\boldsymbol{\Lambda}_n^{\mathbf{v}} = 0.5(1-\kappa)\mathbf{I}_2$ and $\mathbf{h}_0 \sim \mathcal{N}(\mathbf{0}, 0.5\mathbf{I})$. Parameter κ is set to 0.999 or 0.992 corresponding respectively to the slow and fast-fading scenarios[4], and $\Lambda_n^{\omega} = N_0/2$. At the receiver the BCJR algorithm is used as the SISO decoder. Our algorithm is implemented with $M = 3$ and $N = 30$ samples. We resample from $p(x_n|\mathbf{y}_{1:n+M}) \approx \sum_{i=1}^{N} \lambda_n^i \delta(x_n - x_n^i)$ whenever the effective sample size $N_{eff} \approx (\sum_{i=1}^{N}(\lambda_n^i)^2)^{-1} < N/3$.

We first illustrate the performance of the blind turbo receiver where $\mathbf{h}_{0|-1}^i \sim \mathcal{N}(\mathbf{0}, 0.5\mathbf{I})$ $(i = 1, \ldots, N)$. Differential encoding is employed to combat phase ambiguity and the soft information outputted from the FLPS equalizer is computed as in [8]: $P(c_n = C_j|y_{1:n+M}) = \sum_{i=1}^{N} \lambda_n^i \delta(x_n^i \times x_{n-1}^i - X_j)$, where X_j is the BPSK mapping of bit C_j. The plots in Fig. 2 correspond to the first and fourth iterations, averaged over 1000 channel realizations under a slow fading scenario. We observe a noticeable performance gain (2 dB for $BER < 5 \times 10^{-3}$) over the iterations of our FLPS-based receiver, which shows its efficiency as a SISO processing block. Even though a thorough investigation of robustness is out of the scope of this note, in Fig. 2 we considered the effect of channel order overestimation by setting $\mathbf{F}_n = 0.999^{1/2}\mathbf{I}_3$ and $\boldsymbol{\Lambda}_n^{\mathbf{v}} = 10^{-3} \times diag(0.43, 0.43, 0.13)$. For the first iteration this mismatched model causes a 2 dB performance loss at $BER < 10^{-2}$, but the degradation for the fourth iteration is only 1 dB.

We next show the performance of our FLPS turbo receiver when the equalizer has some information about $\mathbf{h_0}$, i.e. this algorithm is initialized with $\mathbf{h}_{0|-1}^i \sim \mathcal{N}(\mathbf{h}_0, 0.1\mathbf{I})$ $(i = 1, \ldots, N)$. In this case differential encoding is not used so $\{x_n\}$ are independent. Consequently, $p(x_k|x_{k-1}) = p(x_k)$ in eqs. (6) and (7) and $P(c_n = C_j|y_{1:n+M}) = \sum_{i=1}^{N} \lambda_n^i \delta(x_n^i - X_j)$. We compare this proposal against the turbo equalizer presented in [11], which performs an iterative channel estimation from second-order statistics of each transmitted symbol, fed back from the decoder in the previous iteration. This turbo scheme with soft input channel estimator (SICE) is implemented with a BCJR algorithm as the SISO equalizer, initialized by a preamble with 40 pilot symbols.

The BER performances of the turbo receivers with the SICE and FLPS based equalizers are displayed in Figs. 3 and 4. The figures also include the performance achieved by a turbo scheme after the fourth iteration using a clairvoyant

[3]If $\mathbf{h}_n$ is constant the algorithm presented in [3] requires less computations than ours when $M > 3$ and $M \approx L$. Nevertheless, the solution proposed therein cannot be extended to scenarios involving time-variant channels.

[4]Defining the normalized fading rate f_d from $\int_0^{f_d} S(f)df \approx 0.98\int_0^{\infty} S(f)df$, where $S(f)$ is the power spectrum density of the channel coefficients, $\kappa = 0.999$ corresponds to a fading rate of $f_d \approx 10^{-3}$, and $\kappa = 0.992$ yields $f_d \approx 10^{-2}$.

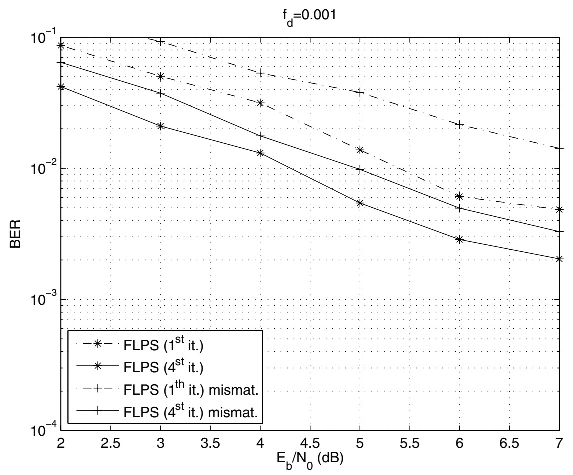

Fig. 2. BER performance of blind FLPS-based turbo receiver for $f_d = 10^{-3}$ ($\kappa = 0.999$).

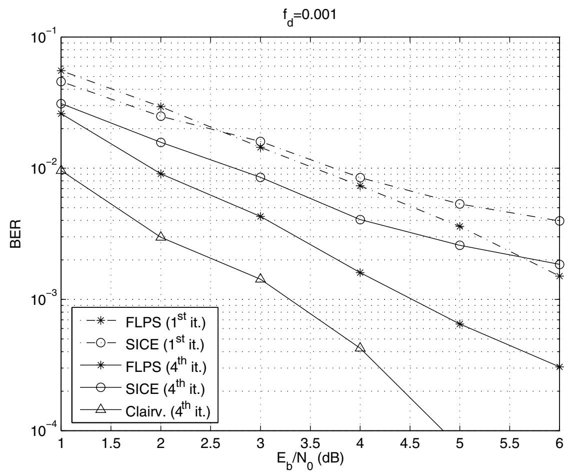

Fig. 3. BER performance of FLPS-based turbo receiver compared (in a non-blind context) with the SICE-based and clairvoyant BCJR-based turbo receivers for $f_d = 10^{-3}$ ($\kappa = 0.999$).

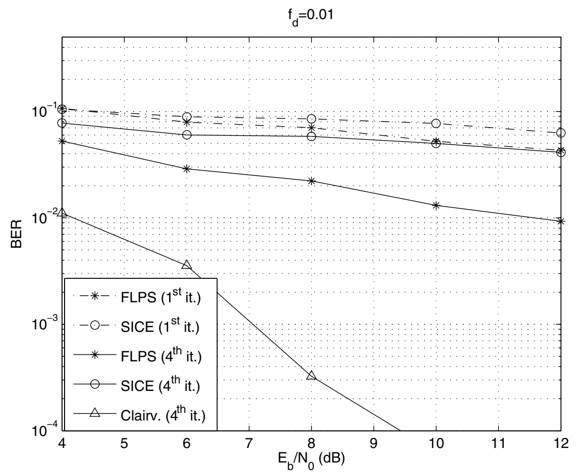

Fig. 4. BER performance of FLPS-based turbo receiver compared (in a non-blind context) with the SICE-based and clairvoyant BCJR-based turbo receivers for $f_d = 10^{-2}$ ($\kappa = 0.992$).

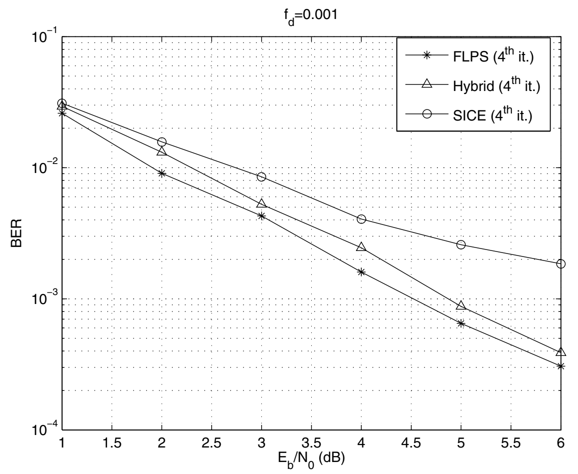

Fig. 5. BER performance of turbo receiver with the hybrid scheme (FLPS and BCJR) compared with the turbo receivers with the FLPS and SICE equalizers for $f_d = 10^{-3}$ ($\kappa = 0.999$).

BCJR equalizer. In Fig. 3 there is only a 1.5–2 dB gap between the FLPS receiver performance and the upper limit shown by the receiver with the clairvoyant BCJR equalizer. Our method clearly outperforms the SICE based receiver, mainly when SNR increases. In Fig. 4 we observe that the performance of the FLPS receiver is far superior under such scenario after four iterations. Accordingly, the BER plot has a noticeably decreasing slope, whereas the performance of the SICE receiver does not improve significantly when the observations samples become less noisy, even after some turbo iterations.

Finally, we have run a turbo processing experiment (again under partial knowledge of $\mathbf{h}_0$) where we use the FLPS SISO equalizer for the initial iterations, and for the subsequent iterations a BCJR-based equalizer is used based on the channel estimate $\hat{\mathbf{h}}_n = \sum_i \lambda_n^i \hat{\mathbf{h}}_{n|n-1}^i$ and on the soft information produced by the FLPS algorithm in the previous iterations. Results are shown in Figs. 5 and 6, respectively for the slow and fast fading channels, under the same transmission conditions. For the slow fading case we used the FLPS equalizer only in the first iteration, and in the fast fading case until the second loop. Both figures show that the performance degradation is indeed small. So this hybrid solution still outperforms the SICE-based receiver, with a computational time which is comparable to that proposal.

V. Conclusions

In this paper we propose a novel SIS-based Bayesian restoration method for CGLSSM using a FLPS methodology. We sample particles from the optimal importance function exactly evaluated by using an algorithm with reduced computational cost when compared to existing alternatives. Simulations show the performance of our technique as a SISO blind equalizer embedded in a turbo equalization receiver. Due to the optimality of the equalizer here developed, our turbo receiver shows outstanding performance when compared (in a non-blind context) to another turbo scheme designed for doubly-selective channels, specially under a fast fading scenario. Simulations also demonstrate the robustness of the

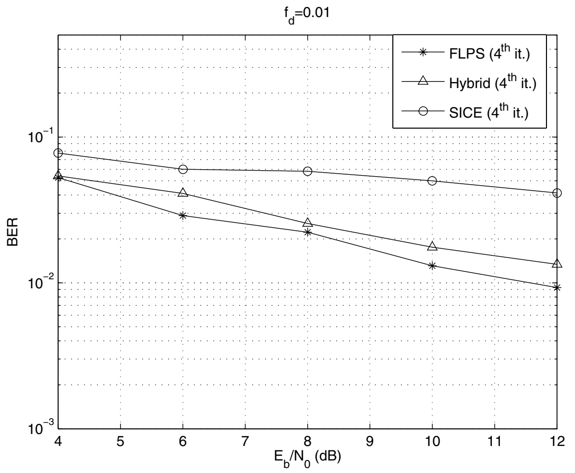

Fig. 6. BER performance of turbo receiver with the hybrid scheme (FLPS and BCJR) compared with the turbo receivers with the FLPS and SICE equalizers for $f_d = 10^{-2}$ ($\kappa = 0.992$).

equalizer and the validity of a hybrid proposal, in which our algorithm initializes a BCJR-based turbo equalizer.

Appendix - Computing $p(\mathbf{y}_{n:n+M}|\boldsymbol{\theta}^i_{\mathbf{x}_{n:n+M}})$ in (7).

From (1)(2), conditionally on $\boldsymbol{\theta}^i_{\mathbf{x}_{n:n+M}}$, $\{(\mathbf{h}_{n+k}, y_{n+k-1})\}_{k=0}^M$ is a (vector) MC. So

$$
\begin{aligned}
& p(\mathbf{y}_{n:n+M}|\boldsymbol{\theta}^i_{\mathbf{x}_{n:n+M}}) = \\
& \int p(\mathbf{h}_{n:n+M}, \mathbf{y}_{n:n+M}|\boldsymbol{\theta}^i_{\mathbf{x}_{n:n+M}}) d\mathbf{h}_{n:n+M} = \\
& \int p(\mathbf{h}_n|\boldsymbol{\theta}^i_{\mathbf{x}_{n:n+M}}) p(\mathbf{h}_{n+1}, y_n|\mathbf{h}_n, \boldsymbol{\theta}^i_{\mathbf{x}_{n:n+M}}) \cdots \times \\
& p(\mathbf{h}_{n+M}, y_{n+M-1}|\mathbf{h}_{n+M-1}, \mathbf{y}_{n:n+M-2}, \boldsymbol{\theta}^i_{\mathbf{x}_{n:n+M}}) \times \\
& p(y_{n+M}|\mathbf{h}_{n+M}, \mathbf{y}_{n:n+M-1}, \boldsymbol{\theta}^i_{\mathbf{x}_{n:n+M}}) d\mathbf{h}_{n:n+M}.
\end{aligned}
$$

So $p(\mathbf{y}_{n:n+M}|\boldsymbol{\theta}^i_{\mathbf{x}_{n:n+M}})$ can be computed recursively, starting from $p(\mathbf{h}_n|\boldsymbol{\theta}^i_{\mathbf{x}_{n:n+M}})$ and integrating w.r.t. $\mathbf{h}_n$, $\cdots$, $\mathbf{h}_{n+M}$. Other simplifications result from (1) (2) : $p(\mathbf{h}_n|\boldsymbol{\theta}^i_{\mathbf{x}_{n:n+M}}) = p(\mathbf{h}_n|\mathbf{x}^i_{1:n-1}, \mathbf{y}_{1:n-1})$; and $p(y_{n+k-1}, \mathbf{h}_{n+k}|\mathbf{y}_{n:n+k-2}, \mathbf{h}_{n+k-1}, \boldsymbol{\theta}^i_{\mathbf{x}_{n:n+M}})$ factorizes as $p(y_{n+k-1}|\mathbf{h}_{n+k-1}, \tilde{\mathbf{x}}_{n+k-1}) p(\mathbf{h}_{n+k}|\ \mathbf{h}_{n+k-1})$. Let us summarize the computation of $p(\mathbf{y}_{n:n+M}|\boldsymbol{\theta}^i_{\mathbf{x}_{n:n+M}})$ in the following recursion :

1) *Initialization.*
Compute $p(\mathbf{h}_n|\boldsymbol{\theta}^i_{\mathbf{x}_{n:n+M}}) = p(\mathbf{h}_n|\mathbf{x}^i_{1:n-1}, \mathbf{y}_{1:n-1})$, and next $p(y_n, \mathbf{h}_{n+1}|\boldsymbol{\theta}^i_{\mathbf{x}_{n:n+M}})$ from $p(\mathbf{h}_n|\boldsymbol{\theta}^i_{\mathbf{x}_{n:n+M}})$ as

$$
p(y_n, \mathbf{h}_{n+1}|\boldsymbol{\theta}^i_{\mathbf{x}_{n:n+M}}) = \int p(\mathbf{h}_n|\boldsymbol{\theta}^i_{\mathbf{x}_{n:n+M}}) \times \underbrace{p(y_n|\mathbf{h}_n, \tilde{\mathbf{x}}_n) p(\mathbf{h}_{n+1}|\mathbf{h}_n)}_{p(y_n, \mathbf{h}_{n+1}|\mathbf{h}_n, \boldsymbol{\theta}^i_{\mathbf{x}_{n:n+M}})} d\mathbf{h}_n \quad \text{(A.1)}
$$

2) *Recursion* $(k-1) \rightarrow k$, *for all* $k = 2, \cdots, M$. Compute $p(\mathbf{y}_{n:n+k-1}, \mathbf{h}_{n+k}|\boldsymbol{\theta}^i_{\mathbf{x}_{n:n+M}})$ from $p(\mathbf{y}_{n:n+k-2}, \mathbf{h}_{n+k-1}|\boldsymbol{\theta}^i_{\mathbf{x}_{n:n+M}})$ as

$$
\begin{aligned}
& p(\mathbf{y}_{n:n+k-1}, \mathbf{h}_{n+k}|\boldsymbol{\theta}^i_{\mathbf{x}_{n:n+M}}) = \\
& \int p(\mathbf{y}_{n:n+k-2}, \mathbf{h}_{n+k-1}|\boldsymbol{\theta}^i_{\mathbf{x}_{n:n+M}}) \times \\
& \underbrace{p(y_{n+k-1}|\mathbf{h}_{n+k-1}, \tilde{\mathbf{x}}_{n+k-1}) p(\mathbf{h}_{n+k}|\mathbf{h}_{n+k-1})}_{p(y_{n+k-1}, \mathbf{h}_{n+k}|\mathbf{y}_{n:n+k-2}, \mathbf{h}_{n+k-1}, \boldsymbol{\theta}^i_{\mathbf{x}_{n:n+M}})} \times \\
& d\mathbf{h}_{n+k-1};
\end{aligned}
\quad \text{(A.2)}
$$

3) *Step* $M \rightarrow (M+1)$. Compute $p(\mathbf{y}_{n:n+M}|\boldsymbol{\theta}^i_{\mathbf{x}_{n:n+M}})$ from $p(\mathbf{y}_{n:n+M-1}, \mathbf{h}_{n+M}|\boldsymbol{\theta}^i_{\mathbf{x}_{n:n+M}})$ as

$$
\begin{aligned}
& p(\mathbf{y}_{n:n+M}|\boldsymbol{\theta}^i_{\mathbf{x}_{n:n+M}}) = \\
& \int p(\mathbf{y}_{n:n+M-1}, \mathbf{h}_{n+M}|\boldsymbol{\theta}^i_{\mathbf{x}_{n:n+M}}) \times \\
& \underbrace{p(y_{n+M}|\mathbf{h}_{n+M}, \tilde{\mathbf{x}}_{n+M})}_{p(y_{n+M}|\mathbf{y}_{n:n+M-1}, \mathbf{h}_{n+M}, \boldsymbol{\theta}^i_{\mathbf{x}_{n:n+M}})} \quad d\mathbf{h}_{n+M}.
\end{aligned}
\quad \text{(A.3)}
$$

In practice, the Gaussian assumption in (1) (2) implies that all densities in (A.1), (A.2) and (A.3) are Gaussian as well. Let $p(\mathbf{h}_n|\mathbf{x}^i_{1:n-1}, \mathbf{y}_{1:n-1}) \sim \mathcal{N}(\widehat{\mathbf{h}}^i_{n|n-1}, \boldsymbol{\Lambda}^{\mathbf{h}^i}_{n|n-1})$, $p(\mathbf{y}_{n:n+k-1}, \mathbf{h}_{n+k}|\boldsymbol{\theta}^i_{\mathbf{x}_{n:n+M}}) \sim \mathcal{N}(\boldsymbol{\mu}^i_k, \boldsymbol{\Sigma}^i_k)$ and $p(\mathbf{y}_{n:n+M}|\boldsymbol{\theta}^i_{\mathbf{x}_{n:n+M}}) \sim \mathcal{N}(\boldsymbol{\mu}^{yi}_{M+1}, \boldsymbol{\Sigma}^{yi}_{M+1})$. From (1) (2) $p(\mathbf{h}_{n+k}|\mathbf{h}_{n+k-1}) \overset{(2)}{\sim} \mathcal{N}(\mathbf{F}_{n+k-1}\mathbf{h}_{n+k-1}\ ,\ \boldsymbol{\Lambda}^{\mathbf{v}}_{n+k-1})$ and $p(y_{n+k}|\mathbf{h}_{n+k}, \boldsymbol{\theta}^i_{\mathbf{x}_{n:n+M}}) \overset{(1)}{\sim} \mathcal{N}(\tilde{\mathbf{x}}^T_{n+k}\mathbf{h}_{n+k},\ \Lambda^w_{n+k})$. Parameters $\widehat{\mathbf{h}}^i_{n|n-1}$ and $\Lambda^{\mathbf{h}^i}_{n|n-1}$ are computed via the KF, and finally the computation of $p(\mathbf{y}_{n:n+M}|\boldsymbol{\theta}^i_{\mathbf{x}_{n:n+M}})$ from $p(\mathbf{h}_n|\ \boldsymbol{\theta}^i_{\mathbf{x}_{n:n+M}})$ reduces to that of $\boldsymbol{\mu}^{yi}_{M+1}$ and $\boldsymbol{\Sigma}^{yi}_{M+1}$ from $\widehat{\mathbf{h}}^i_{n|n-1}$ and $\boldsymbol{\Lambda}^{\mathbf{h}^i}_{n|n-1}$, whence (8) and (9). Equations (10)-(12) are easily verified.

References

[1] M. Lombardi and S.J. Godsill, "On-line Bayesian estimation of AR signals in symmetric alpha-stable noise," *IEEE Trans. Signal Processing*, pp. 775-779, Feb. 2006.

[2] R. Chen, X. Wang, and J. S. Liu, "Adaptive joint detection and decoding in flat-fading channels via Mixture Kalman filtering," *IEEE Trans. Inf. Theory*, vol. 46, no. 6, pp. 2079-2094, Sept. 2000.

[3] J. Miguez and P. M. Djuric, "Blind equalization of frequency-selective channels by sequential importance sampling," *IEEE Trans. Signal Process.*, vol. 52, no. 10, pp. 2738-2748, Oct. 2004.

[4] A. Doucet, N. J. Gordon, and V. Krishnamurthy, "Particle filters for state estimation of jump Markov linear systems," *IEEE Trans. Signal Process.*, vol. 49, no. 3, pp. 613-24, Mar. 2001.

[5] M. A. Vázquez, M. F. Bugallo, and J. Míguez, "Sequential Monte Carlo methods for complexity-constrained MAP equalization of dispersive MIMO channels," *Signal Process. Elsevier*, vol. 88, no. 4, pp. 1017-1034, Apr. 2008.

[6] M. Tuchler, R. Koetter, and A. Singer, "Turbo equalization," *IEEE Signal Processing Mag.*, pp. 67-80, 2004.

[7] L.M. Davis, I.B. Collings, and P. Hoeher, "Joint MAP equalization and channel estimation for frequency-selective and frequency-flat fast-fading channels," *IEEE Trans. Commun.*, vol. 49, no. 12, pp. 2106-2114, Dec. 2001.

[8] D. Guo and X. Wang, "Blind Detection in MIMO Systems via Sequential Monte Carlo," *IEEE J. Select. Areas Commun.*, vol. 21, pp. 464-473, Apr. 2003.

[9] P. Djuric, J. Kotecha, J. Zhang, Y. Huang, T. Ghirmai, M. Bugallo, and J. Miguez, "Particle filtering," *IEEE Signal Process. Mag.*, vol. 50, pp. 281-289, Sept. 2003.

[10] A. Doucet, J. F. G. de Freitas, and N. J. Gordon (editors), *Sequential Monte-Carlo Methods in Practice*, Springer Verlag, NY, May 2001.

[11] S. Song, A. Singer, and K-M. Sung, "Soft input channel estimation for turbo equalization," *IEEE Trans. Signal Process.*, vol. 52, no. 10, pp. 2885-2894, Oct. 2004.