On Structured-Summary Propagation, LFSR Synchronization, and Low-Complexity Trellis Decoding

Justin Dauwels, Hans-Andrea Loeliger, Patrick Merkli, and Maja Ostojic
Signal & Information Proc. Lab. (ISI-DITET)
ETH Zentrum
CH-8092 Zürich, Switzerland

Abstract

A general idea—message passing with messages that have some nontrivial Markov structure—is outlined. This general idea is worked out for one particular application, viz., the synchronization (state estimation) of "noisy" linear-feedback shift register sequences. For this application, the flexible tradeoff between performance and complexity is demonstrated by simulation results. Generalizations to low-complexity approximations of the BCJR algorithm are outlined.

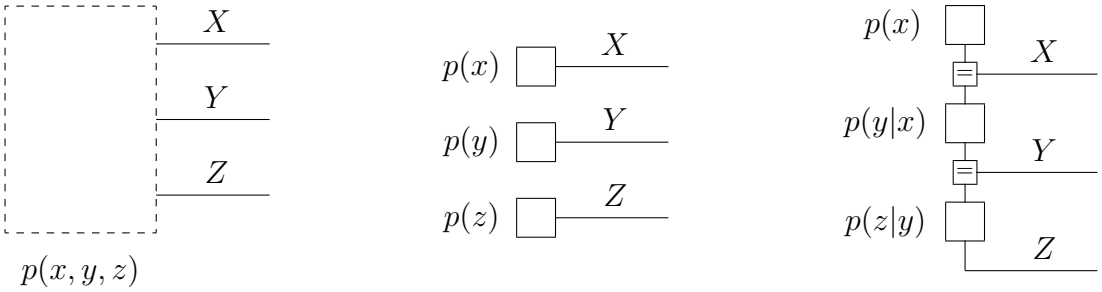
1 Introduction

This paper is both about a general idea and about a specific application. The general idea is to consider message passing algorithms with messages that have some internal Markov structure and thus are themselves nontrivial graphs; the specific application considered in this paper is the synchronization of "noisy" LFSR (linear-feedback shift register) sequences, as will be described below.

The general idea is illustrated in Fig. 1. These figures are Forney-style factor graphs (FFGs) [1], [2], [3], where boxes correspond to factors and edges correspond to variables. Fig. 1a shows a general probability mass function or density (pdf) $p(x, y, z)$ of three variables X, Y, and Z; the box in the figure may have an arbitrary internal structure. According to the standard sum-product (belief propagation) algorithm [4], [3], the messages out of that box are the marginals $p(x)$, $p(y)$, and $p(z)$; this amounts to approximating $p(x, y, z)$ by $p(x)p(y)p(z)$, which is shown in Fig. 1b. However, a better approximation of $p(x, y, z)$ is $p(x)p(y|x)p(z|y)$, which is shown in Fig. 1c. The general idea of this paper is to consider message passing with messages (summaries) that have some Markov structure between the full joint pdf $p(x, y, z)$ and the product of marginals $p(x)p(y)p(z)$.

Related earlier work includes "generalized belief propagation" [5], [6], [7] by Yedidia et al. as well as the more recent "survey propagation" [8]; the relation of our approach to this earlier work remains to be clarified.

The specific application considered in this paper is the synchronization of noisy LFSR sequences, as will be described in Section 2. For this example, we derive explicit (non-iterative) message passing algorithms (in Section 3) and give some preliminary simulation results (in Section 4). A special feature of this example is that forward-only message



a) Some joint pdf. b) Marginals. c) Markov chain.

Figure 1: Models/summaries with Markov structure.

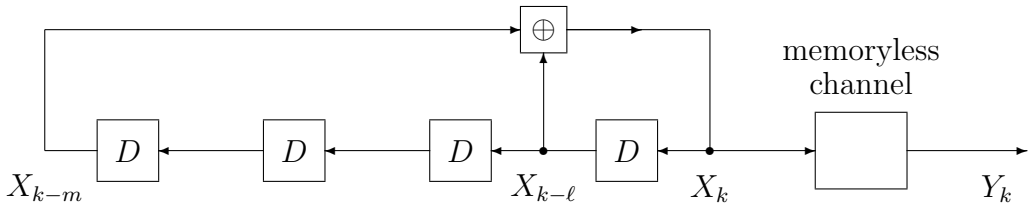


Figure 2: Linear-feedback shift register (LFSR) sequence observed via a noisy channel.

passing—i.e., (nonlinear) filtering—is optimal on the full trellis and attractive for low-complexity algorithms.

Some remarks on other applications are offered in Section 5. In particular, it seems straightforward to develop low-complexity approximations to the BCJR algorithm [9].

2 Synchronization of Noisy LFSR Sequences

As mentioned, we will focus on a specific problem, which was earlier considered in [10]. In this section, we state the problem and summarize the results of [10]; the new algorithms will be described in Section 3.

For fixed integers ℓ and m satisfying $1 \leq \ell < m$, let

$$X \stackrel{\triangle}{=} X_{-m+1}, \ldots, X_{-1}, X_0, X_1, X_2, \ldots \tag{1}$$

be a sequence of variables X_k that take values in $\{0, 1\}$. Let "$\oplus$" denote addition modulo 2 and assume that

$$X_k = X_{k-\ell} \oplus X_{k-m} \tag{2}$$

holds for all $k > 0$. For $k \geq 0$, the m-tuple $[X]_k \stackrel{\triangle}{=} (X_k, X_{k-1}, \ldots, X_{k-m+1})$ will be called the *state* of X at time k. Note that the whole sequence X is fully determined by its state $[X]_k$ at any time $k \geq 0$.

The recursion (2) is illustrated in Fig. 2 for $\ell = 1$ and $m = 4$. The boxes labelled "D" are unit-delay cells; the contents of these boxes determine the state.

The sequence $X_1, X_2, \ldots$ is observed via a memoryless channel with transition probabilities $p(y_k|x_k)$. From the received sequence $Y_1, Y_2, \ldots, Y_n$, we wish to estimate the state

$[X]_n$ of the transmitted sequence. In the numerical examples, we will assume that the channel is defined by

$$Y_k = \tilde{X}_k + Z_k \tag{3}$$

with

$$\tilde{X}_k \triangleq \begin{cases} 1, & \text{if } X_k = 0 \\ -1, & \text{if } X_k = 1 \end{cases} \tag{4}$$

(i.e., binary antipodal signaling) and where $Z_1, Z_2, \ldots$ are independent zero-mean Gaussian random variables with variance σ^2.

As stated, we wish to estimate $[X]_n$ from $Y_1, Y_2, \ldots, Y_n$. The computation of the maximum-likelihood (ML) estimate is straightforward and well known [11]; however, the complexity of this computation is proportional to $n2^m$, which makes it impractical unless m is very small. In [10], a suboptimal algorithm was proposed, the complexity of which is independent of m. The algorithms proposed in the present paper include both the ML estimate and the algorithm of [10] as special cases.

An FFG (Forney-style factor graph) of our system model is shown in Fig. 3. As in [10], we will restrict ourselves to non-iterative *forward-only* (i.e., left-to-right) message passing; all backward (right-to-left) messages may be thought of as carrying a neutral constant factor 1. The forward messages will be denoted by expressions such as $\mu_k(x_k)$ or $\mu_{k-1}(x_{k-1}, \ldots, x_{k-4})$, where the subscript denotes time; all such messages will denote probability mass functions of the indicated variables. As it turns out, it will be advantageous to carry out the update of the messages (from μ_{k-1} to μ_k) in two steps, with intermediate messages $\tilde{\mu}_k$ as indicated in Fig. 3. The first step (resulting in $\tilde{\mu}_k$) is the prediction from the previous state; the second step incorporates the observation Y_k.

In this setup, the low-complexity algorithm of [10] is obtained by straightforward application of the sum-product algorithm [4], [1], [3] with (forward-only) messages as indicated in Fig. 4 (left). In particular, we have

$$\tilde{\mu}_k(x_k) = \sum_{x_{k-1}} \cdots \sum_{x_{k-m}} \delta(x_k \oplus x_{k-\ell} \oplus x_{k-m}) \cdot \mu_{k-1}(x_{k-1}) \cdots \mu_{k-1}(x_{k-m}) \tag{5}$$

$$= \sum_{x_{k-\ell}} \sum_{x_{k-m}} \delta(x_k \oplus x_{k-\ell} \oplus x_{k-m}) \cdot \mu_{k-1}(x_{k-\ell}) \cdot \mu_{k-1}(x_{k-m}) \tag{6}$$

where "$\delta(.)$" is the Kronecker delta, and for $n = 1, \ldots, m-1$, we have

$$\tilde{\mu}_k(x_{k-n}) = \mu_{k-1}(x_{k-n}). \tag{7}$$

Note that the parity check (i.e., the factor $\delta(x_k \oplus x_{k-\ell} \oplus x_{k-m})$) does not appear in (7) since the neutral backwards message along X_k effectively removes the parity check. In the second step, we have

$$\mu_k(x_k) \propto p(y_k|x_k) \cdot \tilde{\mu}_k(x_k), \tag{8}$$

where "$\propto$" denotes equality up to a scale factor, and

$$\mu_k(x_{k-n}) = \tilde{\mu}_k(x_{k-n}) \tag{9}$$

for $n = 1, \ldots, m-1$. As pointed out in [10], these computations may be viewed as filtering the received sequence Y by a "soft" version of the LFSR that generates the sequence X.

The maximum likelihood estimate may be obtained by the forward recursion of the BCJR algorithm; see [10] for details. Within the setup of this paper, this algorithm

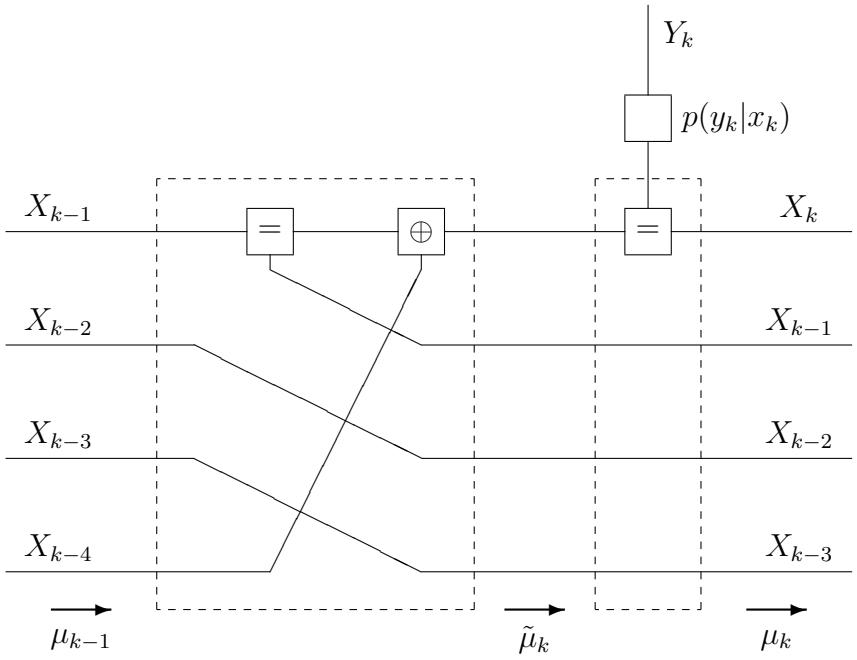


Figure 3: FFG corresponding to Fig. 2 (with $m = 4$ and $\ell = 1$).

Figure 4: Forward-only message passing for marginals (left) and for full joint pdf (right).

Figure 5: Messages as first-order Markov chain (left) and second-order Markov chain (right).

amounts to the (forward-only) sum-product algorithm with messages as indicated in Fig. 4 (right):

$$\tilde{\mu}_k(x_k, \ldots, x_{k-m+1}) = \sum_{x_{k-m}} \delta(x_k \oplus x_{k-\ell} \oplus x_{k-m}) \cdot \mu_{k-1}(x_{k-1}, \ldots, x_{k-m}) \tag{10}$$

and

$$\mu_k(x_k, \ldots, x_{k-m+1}) \propto p(y_k|x_k) \cdot \tilde{\mu}_k(x_k, \ldots, x_{k-m+1}). \tag{11}$$

From these messages, an obvious state estimate $[\hat{X}]_k = (\hat{X}_k, \ldots, \hat{X}_{k-m+1})$ is

$$[\hat{X}]_k = \text{argmax}(\mu_k(x_k, \ldots, x_{k-m+1})). \tag{12}$$

However, in our simulations (also for the algorithms of Section 3), we used the simpler estimate

$$[\hat{X}]_k = \big(\text{argmax}(\mu_k(x_k)), \ldots, \text{argmax}(\mu_k(x_{k-m+1}))\big), \tag{13}$$

the componentwise hard-decision estimate from the marginals; from our limited experience, it seems that the difference in performance between these estimates is insignificant.

3 Intermediate-Complexity Algorithms

Algorithms with a complexity between the two extremes of Section 2 may be obtained by imposing a nontrival Markov chain structure on the messages. We will work this out in detail for messages that are first-order Markov chains as illustrated in Fig. 5 (left); the generalization to higher-order Markov chains (Fig. 5 (right)) is straightforward. We thus assume that the total message $\mu_k(x_k, \ldots, x_{k-m+1})$ factors as

$$\mu_k(x_k, \ldots, x_{k-m+1}) = \frac{\mu_k(x_k, x_{k-1}) \cdot \mu_k(x_{k-1}, x_{k-2}) \cdots \mu_k(x_{k-m+2}, x_{k-m+1})}{\mu_k(x_{k-1}) \cdot \mu_k(x_{k-2}) \cdots \mu_k(x_{k-m+2})} \tag{14}$$

$$= \mu_k(x_k) \cdot \mu_k(x_{k-1}|x_k) \cdots \mu_k(x_{k-m+1}|x_{k-m+2}), \tag{15}$$

and likewise for $\tilde{\mu}_k$. We will work with the form (14), although the form (15) (or the reversed chain) may be more convenient for practical calculations. Note that marginals such as $\mu_k(x_{k-1})$ may be obtained either from $\mu_k(x_k, x_{k-1})$ or from $\mu_k(x_{k-1}, x_{k-2})$.

For the first step, from μ_{k-1} to $\tilde{\mu}_k$, we obtain the prediction message $\tilde{\mu}_k(x_k, x_{k-1})$ as

$$\tilde{\mu}_k(x_k, x_{k-1}) = \sum_{x_{k-2}} \cdots \sum_{x_{k-m}} \delta(x_k \oplus x_{k-\ell} \oplus x_{k-m}) \cdot \mu_{k-1}(x_{k-1}, \ldots, x_{k-m}) \tag{16}$$

$$= \sum_{x_{k-2}} \frac{\mu_{k-1}(x_{k-1}, x_{k-2})}{\mu_{k-1}(x_{k-2})} \sum_{x_{k-3}} \frac{\mu_{k-1}(x_{k-2}, x_{k-3})}{\mu_{k-1}(x_{k-3})} \cdots$$

$$\sum_{x_{k-m}} \mu_{k-1}(x_{k-m+1}, x_{k-m}) \cdot \delta(x_k \oplus x_{k-\ell} \oplus x_{k-m}), \tag{17}$$

which amounts to a bottom-up sum-product sweep through the Markov chain μ_{k-1}. The remaining intermediate messages are obtained as

$$\tilde{\mu}_k(x_{k-n}, x_{k-n-1}) = \mu_{k-1}(x_{k-n}, x_{k-n-1}) \tag{18}$$

for $n = 1, \ldots, m-2$.

For the second step, from $\tilde{\mu}_k$ to μ_k, we obtain first

$$\mu_k(x_k, x_{k-1}) \propto p(y_k|x_k) \sum_{x_{k-2}} \cdots \sum_{x_{k-m+1}} \tilde{\mu}_k(x_k, \ldots, x_{k-m+1}) \tag{19}$$

$$= p(y_k|x_k) \cdot \frac{\tilde{\mu}_k(x_k, x_{k-1})}{\tilde{\mu}_k(x_{k-1})} \sum_{x_{k-2}} \cdots \sum_{x_{k-m+1}} \tilde{\mu}_k(x_{k-1}, \ldots, x_{k-m+1}) \tag{20}$$

$$= p(y_k|x_k) \cdot \tilde{\mu}_k(x_k, x_{k-1}), \tag{21}$$

and then recursively

$$\mu_k(x_{k-1}, x_{k-2}) \propto \sum_{x_k} \sum_{x_{k-3}} \cdots \sum_{x_{k-m+1}} p(y_k|x_k) \cdot \tilde{\mu}_k(x_k, \ldots, x_{k-m+1}) \tag{22}$$

$$= \sum_{x_k} p(y_k|x_k) \frac{\tilde{\mu}_k(x_k, x_{k-1}) \cdot \tilde{\mu}_k(x_{k-1}, x_{k-2})}{\tilde{\mu}_k(x_{k-1}) \cdot \tilde{\mu}_k(x_{k-2})} \sum_{x_{k-3}} \cdots \sum_{x_{k-m+1}} \tilde{\mu}_k(x_{k-2}, \ldots, x_{k-m+1}) \tag{23}$$

$$= \sum_{x_k} p(y_k|x_k) \frac{\tilde{\mu}_k(x_k, x_{k-1}) \cdot \tilde{\mu}_k(x_{k-1}, x_{k-2})}{\tilde{\mu}_k(x_{k-1})} \tag{24}$$

$$\propto \frac{\tilde{\mu}_k(x_{k-1}, x_{k-2})}{\tilde{\mu}_k(x_{k-1})} \sum_{x_k} \mu_k(x_k, x_{k-1}) \tag{25}$$

and

$$\mu_k(x_{k-2}, x_{k-3}) \propto \sum_{x_k} \sum_{x_{k-1}} \sum_{x_{k-4}} \cdots \sum_{x_{k-m+1}} p(y_k|x_k) \cdot \tilde{\mu}_k(x_k, \ldots, x_{k-m+1}) \tag{26}$$

$$= \sum_{x_k} \sum_{x_{k-1}} p(y_k|x_k) \frac{\tilde{\mu}_k(x_k, x_{k-1}) \cdot \tilde{\mu}_k(x_{k-1}, x_{k-2}) \cdot \tilde{\mu}_k(x_{k-2}, x_{k-3})}{\tilde{\mu}_k(x_{k-1}) \cdot \tilde{\mu}_k(x_{k-2}) \cdot \tilde{\mu}_k(x_{k-3})} \sum_{x_{k-4}} \cdots \sum_{x_{k-m+1}} \tilde{\mu}_k(x_{k-3}, \ldots, x_{k-m+1}) \tag{27}$$

$$= \sum_{x_k} \sum_{x_{k-1}} p(y_k|x_k) \frac{\tilde{\mu}_k(x_k, x_{k-1}) \cdot \tilde{\mu}_k(x_{k-1}, x_{k-2}) \cdot \tilde{\mu}_k(x_{k-2}, x_{k-3})}{\tilde{\mu}_k(x_{k-1}) \cdot \tilde{\mu}_k(x_{k-2})} \tag{28}$$

$$= \frac{\tilde{\mu}_k(x_{k-2}, x_{k-3})}{\tilde{\mu}_k(x_{k-2})} \sum_{x_{k-1}} \frac{\tilde{\mu}_k(x_{k-1}, x_{k-2})}{\tilde{\mu}_k(x_{k-1})} \sum_{x_k} p(y_k|x_k) \cdot \tilde{\mu}_k(x_k, x_{k-1}) \tag{29}$$

$$\propto \frac{\tilde{\mu}_k(x_{k-2}, x_{k-3})}{\tilde{\mu}_k(x_{k-2})} \sum_{x_{k-1}} \mu_k(x_{k-1}, x_{k-2}), \tag{30}$$

etc., which amounts to a top-down sum-product sweep through the Markov chain $\tilde{\mu}_k$.

4 Simulation Results

Some first simulation results are shown in Figures 6–11. All these figures show plots of the probability of synchronization

$$\mathrm{P}_{synch}(k) \triangleq P\left([\hat{X}]_k = [X]_k\right), \tag{31}$$

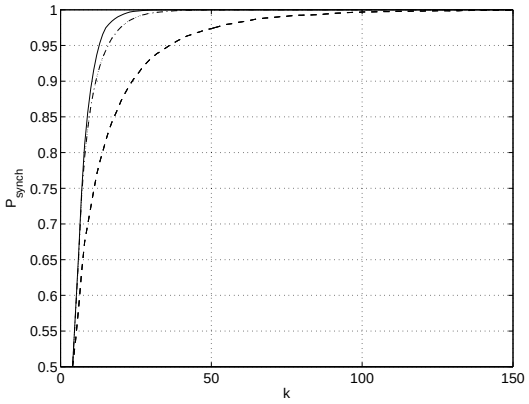


Figure 6: $m = 4$; P_{synch} vs. k at 0 dB.

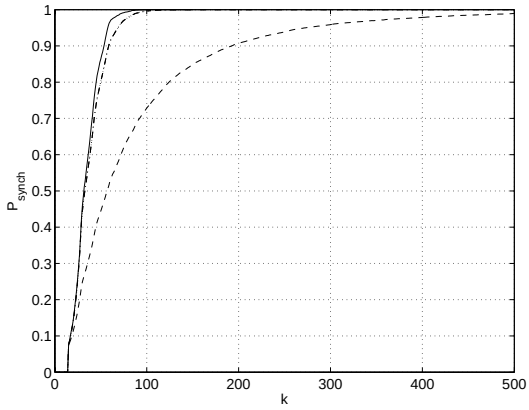


Figure 7: $m = 15$; P_{synch} vs. k at 0 dB.

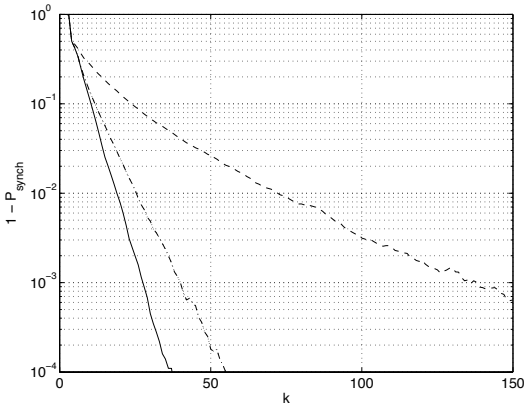


Figure 8: $m = 4$; $1 - \mathrm{P}_{synch}$ vs. k at 0 dB.

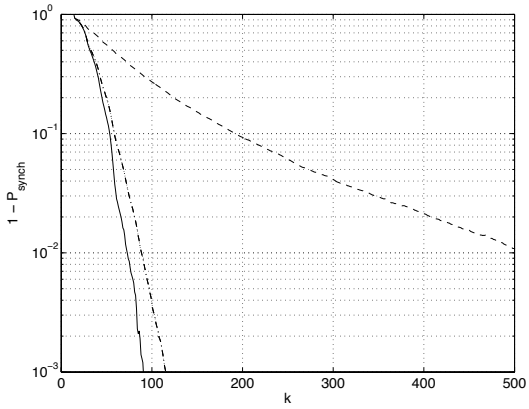


Figure 9: $m = 15$; $1 - \mathrm{P}_{synch}$ vs. k at 0 dB.

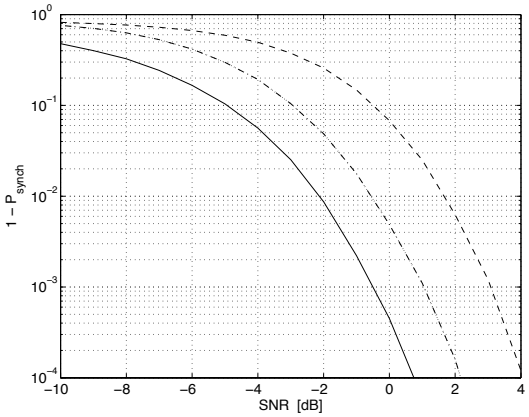


Figure 10: $m = 4$; $1 - \mathrm{P}_{synch}$ vs. SNR at $k = 30$.

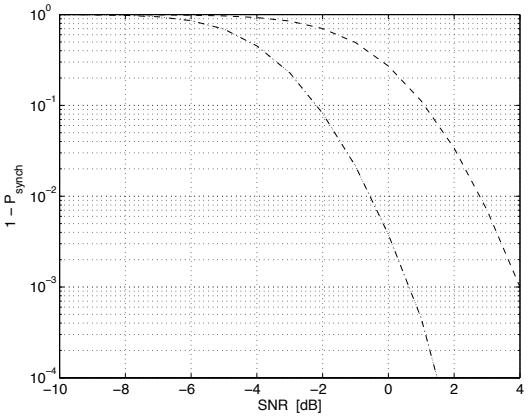


Figure 11: $m = 15$; $1 - \mathrm{P}_{synch}$ vs. SNR at $k = 100$.

either versus the time index k or versus the signal-to-noise ratio $\mathrm{SNR}_{\mathrm{dB}} \triangleq -10 \cdot \log_{10} \sigma^2$, where σ^2 is the noise variance.

The figures in the left column are for a small LFSR with $m = 4$ and $\ell = 1$; the figures in the right column are for a bigger LFSR with $m = 15$ and $\ell = 1$. The solid curves in all figures are the maximum-likelihood estimate, which corresponds to propagating the full joint pdf as described in Section 2. The dashed curves result from the algorithm of [10], which is equivalent to propagating only the marginals as described in Section 2. The dot-dashed curves (between the other curves) result from the algorithm of Section 3, where the messages are first-order Markov chains.

As is obvious from these plots, the algorithm proposed in Section 3 provides a significant improvement over the algorithm of [10] even for first-order Markov chains. Using higher-order Markov chains should provide further substantial improvements.

5 Concluding Remarks

We have outlined the general idea of working with messages that have some nontrivial Markov structure. We have worked out this idea for one application, the synchronization of noisy LFSR sequences, for which we have given some simulation results.

The general idea of this paper seems to be related to "generalized belief propagation" by Yedidia et al. [5], [6], [7] as well as to "survey propagation" by Braunstein, Mézard, and Zecchina [8]. A closely related idea is the approximation of general covariance matrices in Kalman filtering and recursive least squares by band-diagonal matrices.

It seems straightforward to generalize the algorithms of Section 3 so that they can serve as low-complexity approximations both to the forward recursion and to the backward recursion of the BCJR algorithm for general (large) trellises. For example, it appears promising to apply such algorithms to (the trellis of) intersymbol-interference channels; it will also be interesting to see how well such algorithms do when applied to (forward-backward) decoding of convolutional codes.

References

[1] G. D. Forney, Jr., "Codes on graphs: normal realizations," *IEEE Trans. Information Theory,* vol. 47, no. 2, pp. 520–548, 2001.

[2] H.-A. Loeliger, "Least squares and Kalman filtering on Forney graphs," in *Codes, Graphs, and Systems,* (festschrift in honour of David Forney on the occasion of his 60^{th} birthday), R. E. Blahut and R. Koetter, eds., Kluwer, 2002, pp. 113–135.

[3] H.-A. Loeliger, "An introduction to factor graphs," *IEEE Signal Proc. Mag.,* to appear.

[4] F. R. Kschischang, B. J. Frey, and H.-A. Loeliger, "Factor graphs and the sum-product algorithm," *IEEE Trans. Information Theory,* vol. 47, pp. 498–519, Feb. 2001.

[5] J. S. Yedidia, W. T. Freeman, and Y. Weiss, "Generalized belief propagation," in *Advances in Neural Information Processing Systems,* vol. 13, pp. 689–695, Dec. 2000.

[6] J. S. Yedidia, W. T. Freeman, and Y. Weiss, "Understanding Belief Propagation and Its Generalizations," in *Exploring Artificial Intelligence in the New Millennium,* ISBN 1558608117, pp. 239–236, Jan. 2003.

[7] R. J. McEliece and M. Yildirim, "Belief propagation on partially ordered sets," in *Mathematical Systems Theory in Biology, Communication, Computation, and Finance,* J. Rosenthal and D. S. Gilliam, eds., IMA Volumes in Math. & Appl., Springer Verlag, pp. 275–299.

[8] A. Braunstein, M. Mézard, and R. Zecchina, "Survey propagation: an algorithm for satisfiability," preprint URL: `http://lanl.arXiv.org/cs.CC/0212002`.

[9] L. R. Bahl, J. Cocke, F. Jelinek, and J. Raviv, "Optimal decoding of linear codes for minimizing symbol error rate," *IEEE Trans. Information Theory,* vol. 20, pp. 284–287, March 1974.

[10] B. Vigoda, J. Dauwels, N. Gershenfeld, and H.-A. Loeliger, "Low-complexity LFSR synchronization by forward-only message passing," submitted to *IEEE Trans. Information Theory.*

[11] M. K. Simon, J. K. Omura, R. A. Scholtz, and B. K. Levitt, *Spread Spectrum Communications,* vol. 3. Rockville, Maryland, USA: Computer Science Press, 1985.