

On Markov-Structured Summary Propagation and LFSR Synchronization

Justin Dauwels, Hans-Andrea Loeliger, Patrick Merkli, and Maja Ostojic

Signal & Information Proc. Lab. (ISI-DITET)

ETH Zentrum

CH-8092 Zürich, Switzerland

Abstract

Sum-product message passing (belief propagation) was recently extended to messages/summaries with some nontrivial Markov structure. In this paper, a general update rule for Markov-structured messages is proposed, and further experimental results are presented for the synchronization of noisy linear-feedback shift register (LFSR) sequences.

1 Introduction

In [1], we considered the general idea of message passing with messages that have some nontrivial Markov structure, and we applied this idea to the problem of synchronization (or state estimation) of noisy linear-feedback shift register (LFSR) sequences. In this paper, we further elaborate this topic. First, we will present new experimental results for LFSR synchronization by non-iterative forward-only message passing. Second, we propose a general message update rule for Markov-structured messages that can also be used in iterative algorithms. The proposed new update rule, which has some similarity to generalized belief propagation (GBP) proposed by Yedidia et al., [2], [3], [4], does not work well for iterative LFSR synchronization but will perhaps work well in other applications.

2 LFSR Synchronization with Forward-Only Message Passing

We recall the setup of [1]. For fixed integers ℓ and m , $1 \leq \ell < m$, let

$$X \triangleq X_{-m+1}, \dots, X_{-1}, X_0, X_1, X_2, \dots \quad (1)$$

be a sequence of binary random variables X_k with

$$X_k = X_{k-\ell} \oplus X_{k-m} \quad (2)$$

for $k > 0$ and where “ \oplus ” denotes addition modulo 2. The *state* of X at time k , $k \geq 0$, is the m -tuple $[X]_k \triangleq (X_k, X_{k-1}, \dots, X_{k-m+1})$. The whole sequence X is fully determined by its state $[X]_k$ at any time $k \geq 0$.

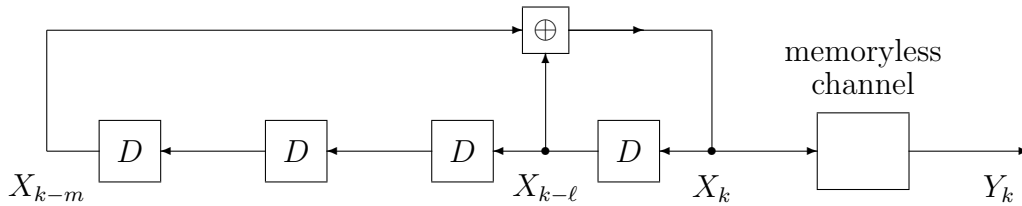


Figure 1: Linear-feedback shift register (LFSR) sequence observed via a noisy channel.

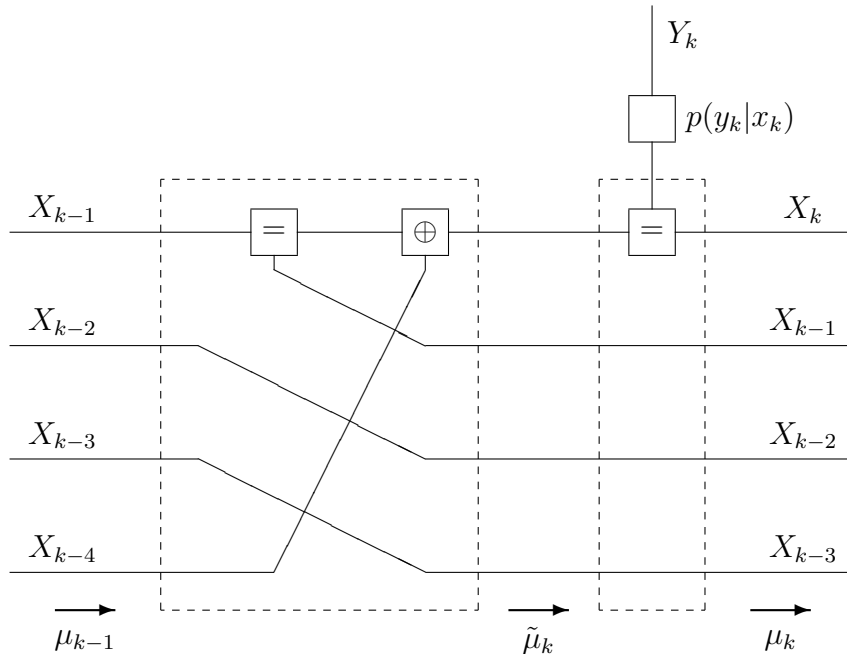


Figure 2: A factor graph (FFG) corresponding to Fig. 1 (with $m = 4$ and $\ell = 1$).

As in [1], we assume that the sequence X_1, X_2, \dots is observed via a memoryless channel with transition probabilities $p(y_k|x_k)$. From the received sequence Y_1, Y_2, \dots, Y_n , we wish to estimate the state $[X]_n$ (or, equivalently, $[X]_0$) of the transmitted sequence. In the numerical examples, we will assume a memoryless Gaussian channel, i.e., $Y_k = \tilde{X}_k + Z_k$ with $\tilde{X}_k = (-1)^{X_k}$ and where Z_1, Z_2, \dots are independent zero-mean Gaussian random variables with variance σ^2 .

An example with $\ell = 1$ and $m = 4$ is shown in Fig. 1, where the boxes labelled “D” are unit-delay cells. A factor graph corresponding to Fig. 1 is shown in Fig. 2. We use Forney-style factor graphs (FFGs) where variables are represented by edges [5].

As pointed out in [1], the maximum-likelihood estimate of $[X]_n$ may be obtained by non-iterative forward-only message passing with messages $\mu_k(x_k, \dots, x_{k-m+1})$ that represent a full joint probability mass function over the state variables, cf. Fig. 3 (right). A much simpler algorithm is obtained by forward-only sum-product message passing (belief propagation) with scalar messages as indicated in Fig. 3 (left). This algorithm may be viewed as using the approximation

$$\mu_k(x_k, \dots, x_{k-m+1}) \approx \mu_k(x_k) \cdots \mu_k(x_{k-m+1}) \quad (3)$$

in every step (and likewise for $\tilde{\mu}_k$ in Fig. 2).

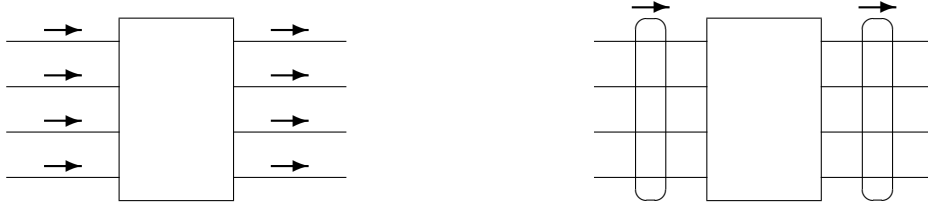


Figure 3: Forward-only message passing for marginals (left) and for full joint pmf (right).

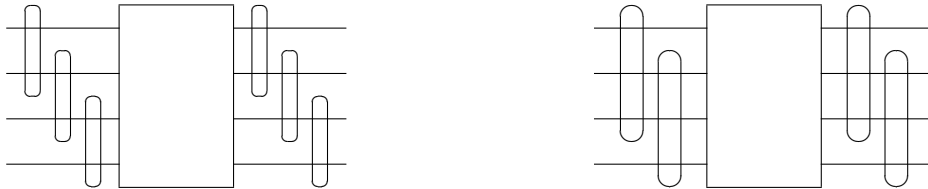


Figure 4: Messages as first-order Markov chain (left) and second-order Markov chain (right).

As in [1], we are interested in algorithms with a complexity in between these two extreme cases. The example worked out in [1] uses the Markov-chain approximation

$$\mu_k(x_k, \dots, x_{k-m+1}) \approx \frac{\mu_k(x_k, x_{k-1}) \cdot \mu_k(x_{k-1}, x_{k-2}) \cdots \mu_k(x_{k-m+2}, x_{k-m+1})}{\mu_k(x_{k-1}) \cdot \mu_k(x_{k-2}) \cdots \mu_k(x_{k-m+2})} \quad (4)$$

(and likewise for $\tilde{\mu}_k$) in every step. As in (3), the factors on the right hand side of (4) are defined as marginals of the left-hand side. In each step, these marginals ($\mu_k(x_k, x_{k-1})$ etc.) are computed by the standard sum-product rule (cf. [5]) applied to the big dashed boxes in Fig. 2; see [1] for a worked-out example. Note that the right hand side of (4) is the maximum-entropy distribution with these marginals (cf. the appendix), and this property extends to more general cycle-free Markov structures.

New simulation results for an LFSR with $m = 15$ and $\ell = 1$ are shown in Figures 5–7. Fig. 5 shows $P_{\text{synch}} \triangleq P([\hat{X}]_k = [X]_k)$ vs. the time index k at a signal-to-noise ratio (SNR) of -4 dB. Fig. 6 shows $1 - P_{\text{synch}}$ in logarithmic scale. Fig. 7 shows $1 - P_{\text{synch}}$ vs. SNR at $k = 130$. All these plots show both the ML estimate (the best) and scalar message passing (the worst; in Fig. 5 and 6, it does not work at all).

Between these two extremes, the plots show the performance of messages with non-trivial Markov-structures. The worst-performing (but simplest) of these is the first-order Markov chain (4). Then come Markov chains of order 3, 5, and 7 (which work with joint marginals of 4, 6, and 8 variables, respectively). Contrary to our expectation (expressed in [1]), these higher order Markov chains offer very little improvement over first-order Markov chains; in Fig. 7, they lie all essentially on top of each other.

Essentially the same performance was obtained also with the Markov structure given by (the right-hand side of) the following approximation:

$$\mu_k(x_k, \dots, x_{k-m+1}) \approx \frac{\mu_k(x_k, x_{k-1}) \cdot \mu_k(x_k, x_{k-2}) \cdots \mu_k(x_k, x_{k-m+1})}{\mu_k(x_k)^{m-2}} \quad (5)$$

A distinct improvement was obtained, however, by the Markov structure given by (the

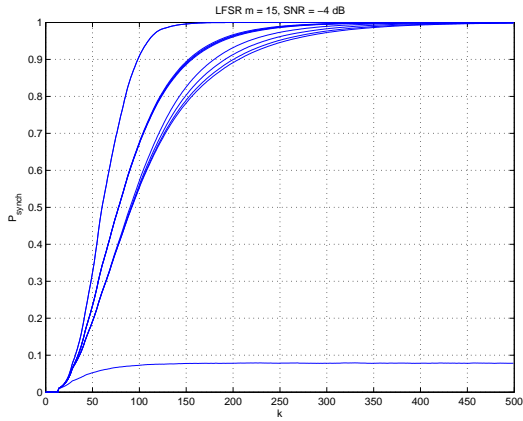


Figure 5: P_{synch} vs. k at -4 dB. From bottom to top: scalar BP; Markov chains of order 1, 3, 5, 7; structure (6) and generalizations to higher order; and ML.

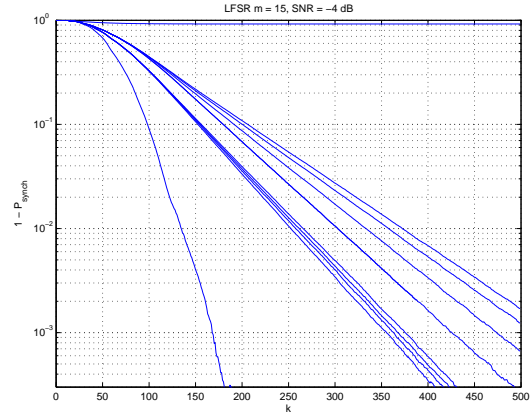


Figure 6: $1 - P_{\text{synch}}$ vs. k at -4 dB. From top to bottom: scalar BP; Markov chains of order 1, 3, 5, 7; structure (6) and generalizations to higher order; and ML.

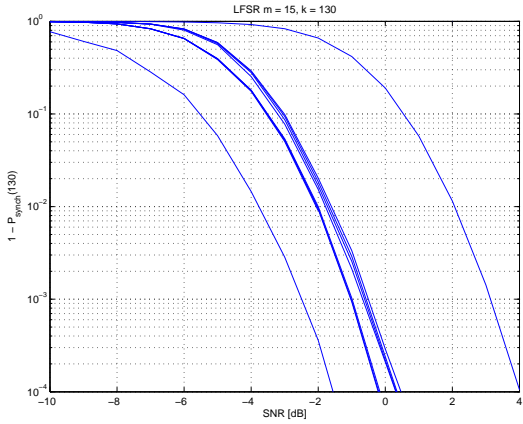


Figure 7: $1 - P_{\text{synch}}$ vs. SNR at $k = 130$. From top to bottom: scalar BP; Markov chains of order 1, 3, 5, 7; structure (6) and generalizations to higher order; and ML.

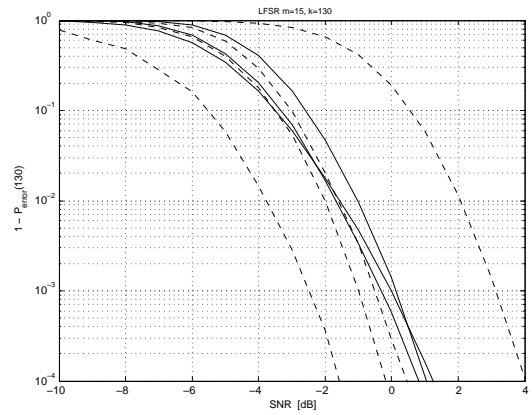


Figure 8: $1 - P_{\text{synch}}$ vs. SNR at $k = 130$. Solid lines: iterative scalar BP, with and without damping. Dashed lines: forward only; from top to bottom: scalar BP; Markov chain of order 1; structure (6); and ML.

right-hand side of) the approximation

$$\mu_k(x_k, \dots, x_{k-m+1}) \approx \frac{\mu_k(x_k, x_{k-1}, x_{k-2}) \cdot \mu_k(x_k, x_{k-2}, x_{k-3}) \cdots \mu_k(x_k, x_{k-m+2}, x_{k-m+1})}{\mu_k(x_k, x_{k-2}) \mu_k(x_k, x_{k-3}) \cdots \mu_k(x_k, x_{k-m+2})} \quad (6)$$

This Markov structure yields the best performance that we could achieve so far with moderate complexity. Interestingly, the obvious generalizations of (6) to higher order marginals yield only minor improvements; in Fig. 7, they lie all on top of each other.

A comparison with standard iterative message passing (BP) is given in Fig. 8. At high SNR (low error probability), the performance of BP can be improved by damping the messages, but it is still outperformed by the forward-only algorithms.

3 A General Update Rule for Markov-Structured Messages

As mentioned, the computation of the Markov-structured messages in the previous section amounts to the computation of “overlapping” multi-variable marginals, which in turn are computed by applying the elementary sum-product rule to the dashed boxes in Fig. 2. However, for iterative message passing, it is not obvious how the Markov-structured messages should be computed. In this section, we propose a general update rule for Markov structured messages in an arbitrary factor graph.

3.1 Problem Statement

The setup is shown in Fig. 9. The three variables/edges X , Y , and Z may be “vectors”, i.e., they may consist of many elementary variables/edges. The solid box labeled f is a generic function/node in some “big” (Forney-style) factor graph (cf. [5]). The other solid box in Fig. 9 is the whole rest of the graph, which is summarized by the message $\overleftarrow{\mu}(x, y, z)$. By standard factor graph syntax, the graph of Fig. 9 (ignoring the dashed box) represents the global function

$$g(x, y, z) \triangleq f(x, y, z) \overleftarrow{\mu}(x, y, z). \quad (7)$$

We will propose a general rule for the computation of messages out of f , which subsumes the standard sum-product rule and generalizes it to Markov-structured multi-variable messages.

In the setup of Fig. 9, standard scalar sum-product message passing assumes that the message $\overleftarrow{\mu}(x, y, z)$ arriving at f factors as

$$\overleftarrow{\mu}(x, y, z) = \overleftarrow{\mu}(x) \overleftarrow{\mu}(y) \overleftarrow{\mu}(z) \quad (8)$$

(which is often called the “independence assumption”). The outgoing message is then

$$\overrightarrow{\mu}(x, y, z) = \overrightarrow{\mu}(x) \overrightarrow{\mu}(y) \overrightarrow{\mu}(z) \quad (9)$$

with

$$\overrightarrow{\mu}(x) = \sum_y \sum_z f(x, y, z) \overleftarrow{\mu}(y) \overleftarrow{\mu}(z) \quad (10)$$

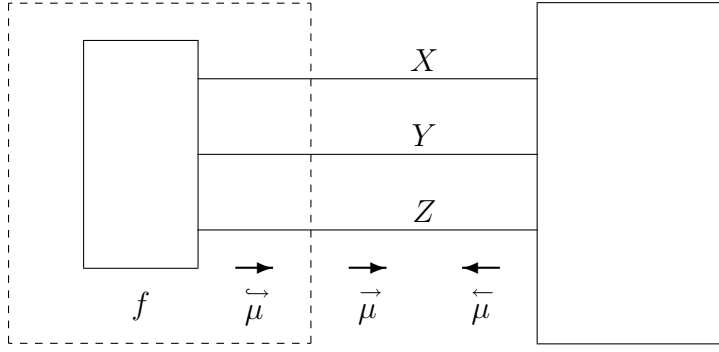


Figure 9: Message passing out of a generic node.

and with $\vec{\mu}(y)$ and $\vec{\mu}(z)$ defined analogously. Note that (10) implies

$$\vec{\mu}(x)\overleftarrow{\mu}(x) = g(x) \quad (11)$$

with

$$g(x) \triangleq \sum_y \sum_z g(x, y, z), \quad (12)$$

and analogously we have $\vec{\mu}(y)\overleftarrow{\mu}(y) = g(y)$ and $\vec{\mu}(z)\overleftarrow{\mu}(z) = g(z)$. Replacing $f(x, y, z)$ by the “lossy summary” $\vec{\mu}(x, y, z)$ (the dashed box in Fig. 9) changes the global function from $g(x, y, z)$ to

$$\tilde{g}(x, y, z) \triangleq \vec{\mu}(x, y, z)\overleftarrow{\mu}(x, y, z) \quad (13)$$

$$= \vec{\mu}(x)\vec{\mu}(y)\vec{\mu}(z)\overleftarrow{\mu}(x)\overleftarrow{\mu}(y)\overleftarrow{\mu}(z) \quad (14)$$

$$= g(x)g(y)g(z), \quad (15)$$

which is the maximum-entropy distribution with the marginals $g(x)$, $g(y)$, and $g(z)$.

We seek to generalize (8)–(15) and the maximum-entropy property of $\tilde{g}(x, y, z)$ to messages with some nontrivial Markov structure such as, e.g.,

$$\overleftarrow{\mu}(x, y, z) = \frac{\overleftarrow{\mu}(x, y)\overleftarrow{\mu}(y, z)}{\overleftarrow{\mu}(y)} \quad (16)$$

3.2 Auxiliary Quantities

As in Section 2, we will define “marginals” of messages as if they were probability mass functions (which they usually are); for example:

$$\vec{\mu}(x, y) = \sum_z \vec{\mu}(x, y, z) \quad (17)$$

$$\vec{\mu}(x) = \sum_y \sum_z \vec{\mu}(x, y, z) \quad (18)$$

$$= \sum_y \vec{\mu}(x, y). \quad (19)$$

The same rules will apply also to marginals of $\overleftarrow{\mu}(x, y, z)$ and of $f(x, y, z)$ and $g(x, y, z)$.

We will use auxiliary quantities (*Merkli messages*) $\overleftarrow{\mu}(x, y)$, $\overleftarrow{\mu}(x)$, etc., which (in the setup of Fig. 9) are defined as follows:

$$\overleftarrow{\mu}(x, y, z) \triangleq \frac{g(x, y, z)}{\overleftarrow{\mu}(x, y, z)} \quad (20)$$

$$= f(x, y, z) \quad (21)$$

$$\overleftarrow{\mu}(x, y) \triangleq \frac{g(x, y)}{\overleftarrow{\mu}(x, y)} \quad (22)$$

$$\overleftarrow{\mu}(x) \triangleq \frac{g(x)}{\overleftarrow{\mu}(x)} \quad (23)$$

etc. These quantities do *not* obey the standard rules for marginalization. Instead, we have the following rules (*Merkli marginalization*):

$$\overleftarrow{\mu}(x, y) = \sum_z \frac{g(x, y, z)}{\overleftarrow{\mu}(x, y, z)} \quad (24)$$

$$= \sum_z f(x, y, z) \frac{\overleftarrow{\mu}(x, y, z)}{\overleftarrow{\mu}(x, y, z)} \quad (25)$$

$$= \sum_z \overleftarrow{\mu}(x, y, z) \frac{\overleftarrow{\mu}(x, y, z)}{\overleftarrow{\mu}(x, y, z)}; \quad (26)$$

$$\overleftarrow{\mu}(x) = \sum_y \sum_z \frac{g(x, y, z)}{\overleftarrow{\mu}(x, y, z)} \quad (27)$$

$$= \sum_y \sum_z f(x, y, z) \frac{\overleftarrow{\mu}(x, y, z)}{\overleftarrow{\mu}(x, y, z)} \quad (28)$$

$$= \sum_y \overleftarrow{\mu}(x, y) \frac{\overleftarrow{\mu}(x, y)}{\overleftarrow{\mu}(x, y)} \quad (29)$$

etc.

3.3 An Update Rule for Markov Structured Messages

We propose a general update rule for Markov structured messages. We will state it explicitly for three special cases, all within the setup of Fig. 9; the general case will then be obvious.

Standard Scalar Messages

Assumed form of incoming message:

$$\overleftarrow{\mu}(x, y, z) \propto \overleftarrow{\mu}(x) \overleftarrow{\mu}(y) \overleftarrow{\mu}(z). \quad (30)$$

Update rule for outgoing message:

$$\overrightarrow{\mu}(x, y, z) \propto \overrightarrow{\mu}(x) \overrightarrow{\mu}(y) \overrightarrow{\mu}(z). \quad (31)$$

It follows that the outgoing message can be written as

$$\vec{\mu}(x, y, z) \propto \vec{\mu}(x) \vec{\mu}(y) \vec{\mu}(z) \quad (32)$$

which has the same Markov structure as the incoming message $\overleftarrow{\mu}(x, y, z)$.

From (28), the quantities $\vec{\mu}(x)$, $\vec{\mu}(y)$, and $\vec{\mu}(z)$ may be computed as, e.g.,

$$\vec{\mu}(x) = \sum_y \sum_z f(x, y, z) \overleftarrow{\mu}(y) \overleftarrow{\mu}(z), \quad (33)$$

which is the standard sum-product rule (10). Approximating $f(x, y, z)$ by the message $\vec{\mu}(x, y, z)$ amounts to approximating $g(x, y, z)$ by

$$\tilde{g}(x, y, z) \triangleq \vec{\mu}(x, y, z) \overleftarrow{\mu}(x, y, z) \quad (34)$$

$$\propto \vec{\mu}(x) \vec{\mu}(y) \vec{\mu}(z) \overleftarrow{\mu}(x) \overleftarrow{\mu}(y) \overleftarrow{\mu}(z) \quad (35)$$

$$= g(x)g(y)g(z). \quad (36)$$

Markov Chain Messages

Assumed form of incoming message:

$$\overleftarrow{\mu}(x, y, z) \propto \frac{\overleftarrow{\mu}(x, y) \overleftarrow{\mu}(y, z)}{\overleftarrow{\mu}(y)}. \quad (37)$$

Update rule for outgoing message:

$$\vec{\mu}(x, y, z) \propto \frac{\vec{\mu}(x, y) \vec{\mu}(y, z)}{\vec{\mu}(y)}. \quad (38)$$

It follows that the outgoing message can be written as

$$\vec{\mu}(x, y, z) \propto \frac{\vec{\mu}(x, y) \vec{\mu}(y, z)}{\vec{\mu}(y)} \quad (39)$$

which has the same Markov structure as the incoming message $\overleftarrow{\mu}(x, y, z)$. (However, in general, $\vec{\mu}(x, y) \neq \overleftarrow{\mu}(x, y)$, etc.)

From (25), the quantity $\vec{\mu}(x, y)$ may be computed as

$$\vec{\mu}(x, y) = \sum_z f(x, y, z) \frac{\overleftarrow{\mu}(y, z)}{\overleftarrow{\mu}(y)} \quad (40)$$

and the quantity $\overleftarrow{\mu}(y)$ may be computed according to (29). Approximating $f(x, y, z)$ by the message $\vec{\mu}(x, y, z)$ amounts to approximating $g(x, y, z)$ by

$$\tilde{g}(x, y, z) \triangleq \vec{\mu}(x, y, z) \overleftarrow{\mu}(x, y, z) \quad (41)$$

$$\propto \frac{\vec{\mu}(x, y) \vec{\mu}(y, z)}{\vec{\mu}(y)} \frac{\overleftarrow{\mu}(x, y) \overleftarrow{\mu}(y, z)}{\overleftarrow{\mu}(y)} \quad (42)$$

$$= \frac{g(x, y)g(y, z)}{g(y)} \quad (43)$$

Note that (43) is the maximum-entropy distribution with the marginals $g(x, y)$ and $g(y, z)$, cf. the appendix.

In the special case where $f(x, y, z)$ has already the Markov structure of $\overleftarrow{\mu}(x, y, z)$, we obtain $\overrightarrow{\mu}(x, y, z) \propto f(x, y, z)$. This can be seen as follows. Assume

$$f(x, y, z) = \frac{f(x, y)f(y, z)}{f(y)} \quad (44)$$

and thus

$$g(x, y, z) = f(x, y, z)\overleftarrow{\mu}(x, y, z) \quad (45)$$

$$= \frac{f(x, y)f(y, z)}{f(y)} \frac{\overleftarrow{\mu}(x, y)\overleftarrow{\mu}(y, z)}{\overleftarrow{\mu}(y)} \quad (46)$$

It follows that $g(x, y, z)$ can be written as

$$g(x, y, z) = \frac{g(x, y)g(y, z)}{g(y)} \quad (47)$$

Together with (43), this implies $\tilde{g}(x, y, z) \propto g(x, y, z)$, which in turn implies $\overrightarrow{\mu}(x, y, z) = f(x, y, z)$.

Lossless Messages

Assumed form of incoming message:

$$\overleftarrow{\mu}(x, y, z) \quad (\text{some general function}). \quad (48)$$

Update rule for outgoing message:

$$\overrightarrow{\mu}(x, y, z) \propto \overleftarrow{\mu}(x, y, z) \quad (49)$$

$$= f(x, y, z). \quad (50)$$

3.4 Experimental Results

The forward-only algorithms of Section 2 may all be viewed as trivial applications of the update rule proposed in this section. However, the real test of the proposed update rule are iterative algorithms. Unfortunately, the straightforward application of the proposed update rule to iterative LFSR synchronization (with Markov-structured forward and backward messages) gives poor numerical results. Damping helps, but we have not yet studied this thoroughly. Better results may perhaps be obtained in other applications.

4 Conclusion

We have considered message passing algorithms with Markov-structured messages. However, we now have more questions than answers. We have proposed a new general update rule, which, however, is not backed by convincing experimental results. For LFSR synchronization, we have obtained good experimental results with noniterative forward-only message passing using simple low-order Markov structures. However, the performance

of more complex higher-order Markov structures remains unsatisfactory. The design of good iterative algorithms for this problem seems difficult. No iterative algorithm seems to work without some sort of damping, and all iterative algorithms seem to have difficulties at high SNR. However, a systematic study of all known techniques including GBP has not yet been made.

5 Acknowledgements

This project was supported in part by the Swiss National Science Foundation grant 200021-101955.

Appendix: Max-Entropy Property of Markov Chains

The following theorem is undoubtedly well known, and its generalization to general cycle-free Markov structures is straightforward.

Theorem: Let $p(x, y, z)$ and $q(x, y, z)$ be two probability mass functions such that

$$p(x, y) = q(x, y) \tag{51}$$

$$p(y, z) = q(y, z) \tag{52}$$

$$p(x, y, z) = p(x|y)p(y)p(z|y). \tag{53}$$

Then

$$H_p(X, Y, Z) \geq H_q(X, Y, Z) \tag{54}$$

with $H_p(X, Y, Z) \triangleq -\sum_{x,y,z} p(x, y, z) \log p(x, y, z)$ and analogously for $H_q(X, Y, Z)$.

Proof:

$$H_p(X, Y, Z) = H_p(X, Y) + H_p(Z|X, Y) \tag{55}$$

$$= H_p(X, Y) + H_p(Z|Y) \tag{56}$$

$$= H_q(X, Y) + H_q(Z|Y) \tag{57}$$

$$\geq H_q(X, Y) + H_q(Z|X, Y) \tag{58}$$

$$= H_q(X, Y, Z). \tag{59}$$

□

References

- [1] J. Dauwels, H.-A. Loeliger, P. Merkli, and M. Ostojic, “On structured-summary propagation, LFSR synchronization, and low-complexity trellis decoding” *Proc. 41st Allerton Conf. on Communication, Control, and Computing*, (Allerton House, Monticello, Illinois), Oct. 1–3, 2003, pp. 459–467.
- [2] 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.

- [3] 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.
- [4] S. M. Aji and R. J. McEliece, “The generalized distributive law and free energy minimization,” *Proc. 39st Allerton Conf. on Communication, Control, and Computing*, (Allerton House, Monticello, Illinois), October 2001.
- [5] H.-A. Loeliger, “An introduction to factor graphs,” *IEEE Signal Proc. Mag.*, Jan. 2004, pp. 28–41.