

# Gaussian Message Passing on Linear Models: An Update

Hans-Andrea Loeliger<sup>1</sup>, Junli Hu<sup>1</sup>, Sascha Korl<sup>2</sup>, Qinghua Guo<sup>3</sup>, and Li Ping<sup>3</sup>

<sup>1</sup> Dept. of Information Technology and Electrical Engineering, ETH Zurich, CH-8092 Zurich, Switzerland.

<sup>2</sup> Phonak AG, Laubisrütistr. 28, CH-8712 Stäfa, Switzerland.

<sup>3</sup> Dept. of Electronic Engineering, City University of Hong Kong, Kowloon, Hong Kong.

## Abstract

This semi-tutorial paper considers message passing algorithms on factor graphs of linear Gaussian models. Freshly polished tables of message computation rules are given and their use is demonstrated for soft-in soft-out equalization.

## 1 Introduction

In this paper, we consider Gaussian message passing in factor graphs of linear models. We present message computation tables for linear building blocks that allow to compose a variety of efficient algorithms without extra computations or derivations. The essence of this approach was presented in [1] and [2]; in the present paper, we review this approach with numerous refinements and demonstrate its application to soft-in soft-out equalization.

For the sake of exposition, we focus on linear state space models with input  $U_k$ , output  $Y_k$ , and state  $X_k$ ,  $k = 1, 2, 3, \dots, N$ , given by

$$X_k = A_k X_{k-1} + B_k U_k \quad (1)$$

$$Y_k = C_k X_k \quad (2)$$

with known (real or complex) matrices  $A_k$ ,  $B_k$ , and  $C_k$ . We assume that such a state space model is part of some larger model that defines a priori probabilities for  $X_0$  and  $U_k$  and connects outputs  $Y_k$  (and/or inputs  $U_k$ ) with known observations. A specific example—equalization—will be described in Section 3.

We will assume either that all variables in such a model are Gaussians or that we choose to treat them as Gaussians for the sake of computational efficiency. In this context, we recall the following facts (see the Appendix for details):

- For Gaussians, MAP estimation coincides with LMMSE (linear minimum mean squared error) estimation.
- MAP estimation with *assumed* Gaussians coincides with *true* LMMSE estimation.
- For Gaussian factor graphs, the sum-product algorithm and the max-product algorithm coincide.

Many problems besides equalization can also be formulated as message passing on linear Gaussian models. In particular, multi-user separation, LPC (linear predictive coding) analysis, and RLS (recursive least squares) adaptive filters lead to special cases of the general state space model (1), (2), cf. [5]. In all these problems, the

message computation tables given in this paper allow to write down complete message passing algorithms without additional computations or derivations.

This paper is structured as follows. Section 2 introduces the factor graph of (1) and (2) that will be used throughout this paper. In Section 3, the general setup as outlined above is described in detail for soft-in soft-out equalization. Section 4 introduces some notation and presents some useful relations among the parameters of Gaussian messages; this section also introduces Tables I–IV, which form the core of this paper. The use of these tables is illustrated in Section 5, which gives a number of complete message passing algorithms. Some basic facts about Gaussian distributions and LMMSE estimation are reviewed in the Appendix.

This paper is not an introduction to factor graphs. For such an introduction, see [2] or [3].

The following notation will be used. The transpose of a matrix (or vector)  $A$  is denoted by  $A^T$ ;  $A^H$  denotes the complex conjugate of  $A^T$ ;  $A^\#$  denotes the Moore-Penrose pseudo-inverse of  $A$ ; and “ $\propto$ ” denotes equality of functions up to a scale factor.

## 2 Factor Graph of Linear State Space Model

We will use Forney-style factor graphs as in [2] where edges represent variables and nodes/boxes represent factors. (Such graphs were introduced in [4], but we deviate in some details from the notation of [4].)

A factor graph corresponding to (1) and (2) is shown in Fig. 1. All nodes in Fig. 1 represent deterministic constraints, which are formally expressed by factors involving Dirac deltas. For example, the node/box connecting  $X_k$ ,  $X'_k$ , and  $X''_k$  represents the factor  $\delta(X_k - X'_k)\delta(X_k - X''_k)$ , which enforces  $X_k = X'_k = X''_k$  for all valid configurations; the node/box connecting  $U_k$  and  $U'_k$  represents the factor  $\delta(U'_k - B_k U_k)$ , which enforces  $U'_k = B_k U_k$  for all valid configurations.

We will assume that all incoming and all outgoing

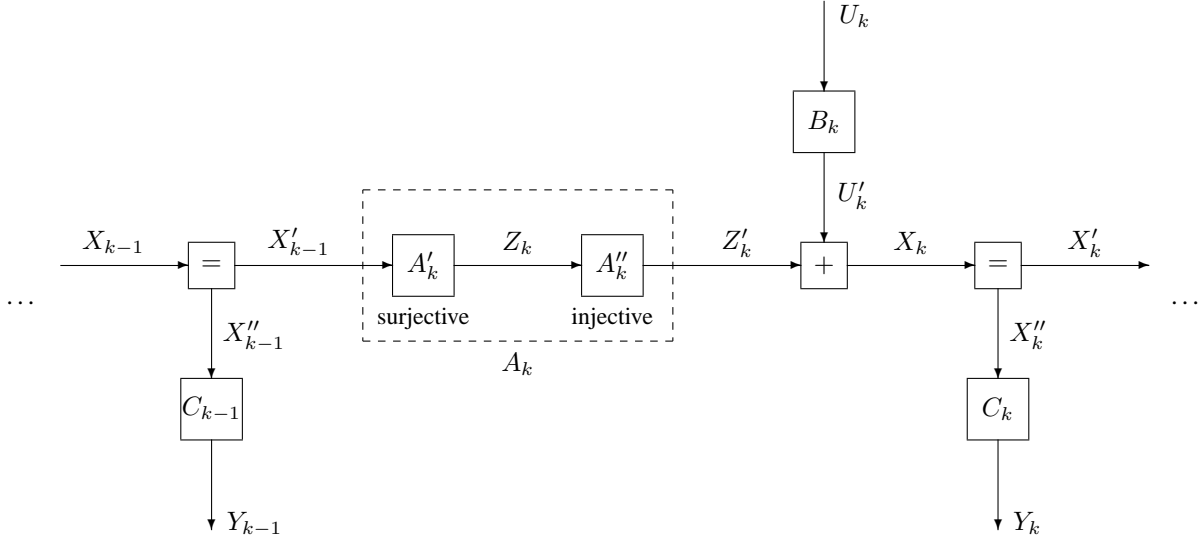


Fig. 1. Factor graph of generic linear state space model.

messages are Gaussians, in which case all internal (sum-product or max-product) messages are also Gaussians.

Also indicated in Fig. 1 is the decomposition of the matrix  $A_k$  of (1) into

$$A_k = A''_k A'_k \quad (3)$$

such that the multiplication by  $A'_k$  is a surjective mapping and the multiplication by  $A''_k$  is an injective mapping. (In other words, the rank of  $A'_k$  equals the number of rows of  $A'_k$  and the rank of  $A''_k$  equals the number of columns of  $A''_k$ .) Such a decomposition is always possible and is sometimes useful if  $A_k$  is singular.

### 3 Example: Equalization

As a specific example, consider the transmission of real symbols  $U_k$  over a discrete-time linear intersymbol interference channel with additive white Gaussian noise. The received real values  $Y'_k$ ,  $k = 1, \dots, N$ , are given by

$$Y'_k = \sum_{\ell=0}^M h_\ell U_{k-\ell} + Z_k, \quad (4)$$

where  $Z_k$ ,  $k = 1, \dots, N$ , are i.i.d. zero-mean Gaussian random variables with variance  $\sigma^2$  and where  $h_0, \dots, h_M$  are known real coefficients. (The initial channel state  $U_0, U_{-1}, \dots, U_{-M+1}$  may be known or unknown, depending of the application.)

We bring (4) into the state space form (1), (2) by defining  $Y_k \triangleq Y'_k - Z_k$  (the noise-free output),  $X_k \triangleq (U_k, \dots, U_{k-M})^T$ , and the matrices

$$A_k \triangleq A \triangleq \begin{pmatrix} 0 & 0 \\ I_M & 0 \end{pmatrix} \quad (5)$$

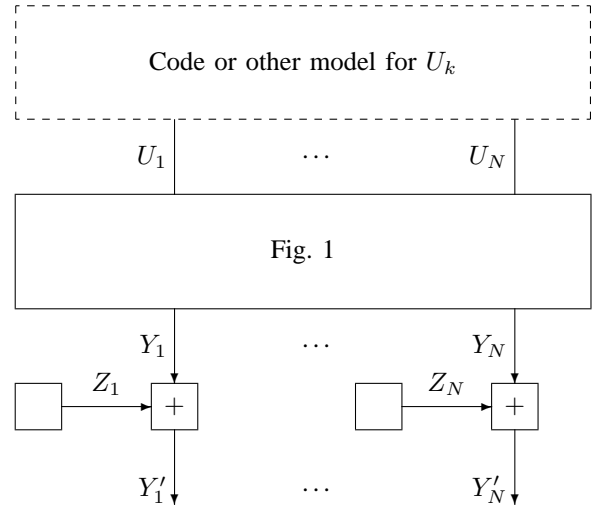


Fig. 2. Factor graph of (4).

(where  $I_M$  denotes the  $M \times M$  identity matrix) and

$$B_k \triangleq B \triangleq (1, 0, \dots, 0)^T \quad (6)$$

$$C_k \triangleq C \triangleq (h_0, h_1, \dots, h_M). \quad (7)$$

We note that the decomposition (3) yields  $A'_k = (I_M, 0)$  and  $A''_k = \begin{pmatrix} 0 \\ I_M \end{pmatrix}$ ; for later use we also note  $(A'_k)^\# = (A'_k)^T$  and  $(A''_k)^\# = (A''_k)^T$ .

The factor graph of the model (4) is shown in Fig. 2. The unlabeled nodes at the bottom of Fig. 2 represent Gaussian distributions with mean zero and with variance  $\sigma^2$ . The dashed box at the top of Fig. 2 represents a code constraint (e.g., the graph of a low-density parity check code) or another model for  $U_k$ , if available.

## 4 Gaussian Messages and their Computation Rules

All messages and marginal functions (a posteriori probabilities) will be Gaussians, which we will describe either by the mean vector  $m$  and the covariance matrix  $V$  or by the weight matrix  $W \triangleq V^{-1}$  and the transformed mean  $Wm$  (cf. the Appendix).

Each edge of the factor graph carries two messages, one in each direction. We will often refer to these two messages by calling one of them “incoming” and the other “outgoing”, or by calling one of them “forward” and the other “backward”. If some edge represents the variable  $X$ , the incoming message has mean  $m_{\text{in}X}$ , covariance matrix  $V_{\text{in}X}$ , and inverse covariance matrix  $W_{\text{in}X} = V_{\text{in}X}^{-1}$ ; the outgoing message has mean  $m_{\text{out}X}$ , covariance matrix  $V_{\text{out}X}$ , and inverse covariance matrix  $W_{\text{out}X} = V_{\text{out}X}^{-1}$ . The product of these two messages—the marginal of the global function if the factor graph has no cycles—is the Gaussian with mean  $m_X$  and covariance matrix  $V_X = W_X^{-1}$  given by

$$W_X = W_{\text{in}X} + W_{\text{out}X} \quad (8)$$

and

$$W_X m_X = W_{\text{in}X} m_{\text{in}X} + W_{\text{out}X} m_{\text{out}X}. \quad (9)$$

An open half edge without an incoming message may be treated as carrying the neutral factor 1 as incoming message. Such a message may be viewed as the limit of a Gaussian with  $W_{\text{in}X} = 0$  and arbitrary finite  $m_{\text{in}X}$ .

We will also use the auxiliary quantity

$$\tilde{W}_X \triangleq (V_{\text{in}X} + V_{\text{out}X})^{-1}. \quad (10)$$

The following relations among these quantities are often useful:

$$\tilde{W}_X = W_{\text{in}X} V_X W_{\text{out}X} \quad (11)$$

$$= W_{\text{in}X} - W_{\text{in}X} V_X W_{\text{in}X} \quad (12)$$

$$V_X = V_{\text{in}X} \tilde{W}_X V_{\text{out}X} \quad (13)$$

$$= V_{\text{in}X} - V_{\text{in}X} \tilde{W}_X V_{\text{in}X} \quad (14)$$

$$m_X = V_X W_{\text{in}X} m_{\text{in}X} + V_X W_{\text{out}X} m_{\text{out}X} \quad (15)$$

$$= m_{\text{in}X} - V_{\text{in}X} \tilde{W}_X m_{\text{in}X} + V_X W_{\text{out}X} m_{\text{out}X} \quad (16)$$

Note that “in” and “out” are arbitrary labels and may be interchanged in all these relations.

Computation rules for messages (such as  $m_{\text{in}X}$ ,  $V_{\text{out}X}$ , etc.) and marginals (such as  $m_X$ ,  $V_X$ , etc.) are listed in Tables I, II, and IV. In principle, Table I suffices to compute all messages in Fig. 1. However, using only the rules of Table I leads to frequent transformations of  $W$  and  $Wm$  into  $V = W^{-1}$  and  $m$ , and vice versa; if  $V$  and  $W$  are large matrices, such conversions are costly.

(Rules (19)–(30) follow from elementary probability theory. The remaining rules are most easily proved by recognizing them as Fourier transforms of other rules in the spirit of [4].)

The inversion of big matrices can often be avoided by using the message computation rules given in Table II (which follow from the Matrix Inversion Lemma [6]). The point of these rules is that the dimension of  $Y$  may be much smaller than the dimension of  $X$  and  $Z$ ; in particular,  $Y$  may be a scalar. The signs in (38) depend on the direction of the arrows, cf. (23) and (24).

The decompositions shown in Table III often allow the propagation of the “wrong” parameters through a matrix multiplication node without inverting a big matrix. Table III (top) together with Table II (bottom) allows the propagation of  $W$  and  $Wm$  forward through a matrix multiplication node; Table III (bottom) together with Table II (top) allows the propagation of  $V$  and  $m$  backward through a matrix multiplication node. The new “internal” open input in Table III (top) may be viewed as carrying as incoming message a degenerate Gaussian with  $W_{\text{in}} = 0$  and arbitrary finite  $m_{\text{in}}$ ; the new “internal” output in Table III (bottom) may be viewed as carrying as incoming message a degenerate Gaussian with  $m_{\text{in}} = 0$  and  $V_{\text{in}} = 0$ .

The grouping of nodes shown in Table IV (top) is used in Algorithm D below. The grouping of nodes shown in Table IV (bottom) is used in Algorithms C and F below.

## 5 Complete Algorithms

We describe sequences of message computations that allow to compute essentially any message and any marginal in the factor graph of Fig. 1. If all the inputs  $U_k$  and all the output  $Y_k$  are scalars (i.e., if the matrices  $B_k$  are column vectors and the matrices  $C_k$  are row vectors), then none of these computations involves a matrix inversion. This applies, in particular, to the example of Section 3.

In these algorithms, forward (left-to-right) messages will be denoted by  $m_{\text{fw}X}$ ,  $V_{\text{fw}X}$ ,  $W_{\text{fw}X}$ , etc., and backward (right-to-left) messages will be denoted by  $m_{\text{bw}X}$ ,  $V_{\text{bw}X}$ ,  $W_{\text{bw}X}$ , etc.

(If the rank of the matrix  $A_k$  in Fig. 1 is smaller than the number of rows of  $A_k$ , then  $W_{\text{fw}Z'_k}$  is undefined; if the rank of  $A_k$  is smaller than the number of columns, then  $V_{\text{bw}X'_{k-1}}$  is undefined.)

We begin with the two basic Kalman filter recursions for the computation of messages in Fig. 1.

**Algorithm A: forward with  $m_{\text{fw}}$  and  $V_{\text{fw}}$ .** (This algorithm is known as “covariance matrix Kalman filter” [6].) From the forward message at  $X_{k-1}$  and the incoming (upward) message at  $Y_{k-1}$ , the forward message at  $X'_k$  is obtained by (35) and (36). ( $X''_{k-1}$  is skipped). The forward message at  $Z'_k$  is obtained by (27) and (28). ( $Z_k$  may be skipped). The incoming

(downward) message at  $U'_k$  is obtained from the incoming message at  $U_k$  by (27) and (28). The forward message at  $X_k$  is then obtained by (21) and (23).

**Algorithm B: backward with  $W_{\text{bw}}$  and  $W_{\text{bw}}m_{\text{bw}}$ .** (This algorithm is known as “information matrix Kalman filter” [6].) From the incoming (upward) message at  $Y_k$ , the incoming message at  $X''_k$  is obtained by (31) and (32). From this and from the backward message at  $X'_k$ , the backward message at  $X_k$  is obtained by (17) and (18). The backward message at  $Z'_k$  is then obtained by (38) and (39). ( $U'_k$  is skipped.) The backward message at  $X'_{k-1}$  is obtained by (31) and (32). ( $Z_k$  may be skipped.)

It is also possible to backward propagate  $m_{\text{bw}}$  and  $V_{\text{bw}}$  and to forward propagate  $W_{\text{fw}}$  and  $W_{\text{fw}}m_{\text{fw}}$ :

**Algorithm C: forward with  $W_{\text{fw}}$  and  $W_{\text{fw}}m_{\text{fw}}$ .** From the incoming (upwards) message at  $Y_{k-1}$ , the incoming (upwards) message at  $X''_{k-1}$  is obtained by (31) and (32). From this and from the forward message at  $X_{k-1}$ , the forward message at  $X'_{k-1}$  is obtained by (17) and (18). The forward message at  $Z'_k$  is obtained by the decomposition of  $A'_k$  as in Table III (top) and using first (38) and (39) and then (31) and (32). The forward message at  $X_k$  is obtained by grouping  $A'_k$  with  $B_k$  as in Table IV (bottom) and using (43) and (44). ( $Z'_k$  is skipped.)

**Algorithm D: backward with  $m_{\text{bw}}$  and  $V_{\text{bw}}$ .** The incoming message at  $U'_k$  is obtained from the incoming (downward) message at  $U_k$  by (27) and (28). From this and from the backward message at  $X_k$ , the backward message at  $Z'_k$  is obtained by (22) and (24). The backward message at  $Z_k$  is obtained by the decomposition of  $A''_k$  as in Table III (bottom) and using first (35) and (36) and then (27) and (28). The backward message at  $X_{k-1}$  is obtained by grouping  $A'_k$  with  $C_{k-1}$  as in Table IV (top) and using (41) and (42). ( $X'_{k-1}$  is skipped.)

Marginals of the global function (usually a posteriori probabilities) as well as output messages upwards out of  $U_k$  and/or downwards out of  $Y_k$  may be obtained by the following algorithms.

**Algorithm E: all marginals and output messages by forward-backward propagation.** Forward pass: with  $m_{\text{fw}}$  and  $V_{\text{fw}}$  according to Algorithm A. Backward pass: with  $W_{\text{bw}}$  and  $W_{\text{bw}}m_{\text{bw}}$  according to Algorithm B, augmented by the simultaneous computation of the marginals  $V$ ,  $m$ , and the auxiliary quantity  $\tilde{W}$  (10) as follows. From  $m$  and  $V$  at  $X'_k$ , we obtain  $m$  and  $V$  both at  $X_k$  and at  $X''_k$  by (19) and (20). We then obtain  $\tilde{W}$  at  $X_k$  (from  $V$  and  $W_{\text{bw}}$  at  $X_k$ ) by (12) and  $m$  at  $X_k$  (from  $m_{\text{fw}}$ ,  $V_{\text{fw}}$ , and  $W_{\text{bw}}m_{\text{bw}}$ ) by (16).  $\tilde{W}$  both at  $Z'_k$  and at  $U'_k$  is obtained by (26), and  $V$  at  $Z'_k$  is obtained by (14).  $m$  at  $Z'_k$  is obtained again by (16) and  $m$  at  $U'_k$  is obtained by (25).  $\tilde{W}$  and  $\tilde{W}m$  at  $X'_{k-1}$  are obtained by (34) and (33), respectively.

Finally,  $V$  and  $m$  at  $X'_{k-1}$  are obtained by (14) and (16), respectively.

The outgoing message at  $Y_k$  may be obtained as follows. From  $m$  and  $V$  at  $X''_k$ , we obtain  $m$  and  $V$  at  $Y_k$  by (29) and (30). Inverting  $V$  at  $Y_k$  yields  $W$  at  $Y_k$ , from which the outgoing message at  $Y_k$  can be extracted by (8) and (9).

The outgoing message at  $U_k$  may be obtained as follows. From  $m$  and  $\tilde{W}$  at  $U'_k$ , we obtain  $\tilde{W}$  and  $\tilde{W}m$  at  $U_k$  by (34) and (33), from which the outgoing message can be extracted by (10) and (9).

The following algorithm is a variation of Algorithm E which is often simpler.

**Algorithm F: all marginals and output messages by forward-backward propagation.** Forward pass: with  $m_{\text{fw}}$  and  $V_{\text{fw}}$  according to Algorithm A. Backward pass: with  $W_{\text{bw}}$  and  $W_{\text{bw}}m_{\text{bw}}$  according to Algorithm B, augmented by the simultaneous computation of the marginals  $V$ ,  $m$ , and the auxiliary quantity  $\tilde{W}$  (10) as follows. From  $m$  and  $V$  at  $X'_k$ , we obtain  $m$  and  $V$  both at  $X_k$  and at  $X''_k$  by (19) and (20). By grouping  $A''_k$  with  $B_k$  as in Table IV (bottom), we simultaneously obtain  $m$  and  $V$  at  $Z_k$  and at  $U_k$ . We then obtain  $\tilde{W}$  at  $Z_k$  (from  $V$  and  $W_{\text{bw}}$  at  $Z_k$ ) by (12) and  $m$  at  $Z_k$  (from  $m_{\text{fw}}$ ,  $V_{\text{fw}}$ , and  $W_{\text{bw}}m_{\text{bw}}$ ) by (16).  $\tilde{W}$  and  $\tilde{W}m$  at  $X'_{k-1}$  are obtained by (34) and (33), respectively. Finally,  $V$  and  $m$  at  $X'_{k-1}$  are obtained by (14) and (16), respectively.

The outgoing message at  $U_k$  is obtained by extracting it from  $m$  and  $V$  at  $U_k$ . The outgoing message at  $Y_k$  may be obtained as in Algorithm E.

**Algorithm G: all marginals and output messages by backward-forward propagation.** Backward pass: with  $W_{\text{bw}}$  and  $W_{\text{bw}}m_{\text{bw}}$  according to Algorithm B. Forward pass: with  $m_{\text{fw}}$  and  $V_{\text{fw}}$  according to Algorithm A, augmented by the simultaneous computation of the marginals  $V$ ,  $m$ , and the auxiliary quantity  $\tilde{W}$  (10) as follows. From  $m$  and  $V$  at  $X_{k-1}$ , we obtain  $m$  and  $V$  at both  $X'_{k-1}$  and  $X''_{k-1}$  by (19) and (20). We then obtain  $m$  and  $V$  at  $Z'_k$  by (29) and (30), from which we obtain  $\tilde{W}$  at  $Z'_k$  (from  $V$  and  $W_{\text{bw}}$ ) by (12) and  $m$  at  $Z'_k$  (from  $m_{\text{fw}}$ ,  $V_{\text{fw}}$ , and  $W_{\text{bw}}m_{\text{bw}}$ ) by (16).  $\tilde{W}$  at both  $U'_k$  and  $X_k$  are obtained from (26), and  $V$  at  $X_k$  is obtained by (14).  $m$  at  $X_k$  is then obtained by (16) and  $m$  at  $U'_k$  is obtained by (25).

The outgoing messages at  $U_k$  and  $Y_k$  are computed as in Algorithm E.

For the example of Section 3, Algorithm F is particularly attractive since, in this case, the required inverse of the matrix  $(A''_k, B_k)$  is a trivial permutation matrix.

It should be noted that all these algorithms may have numerical problems (depending on the application) due to the subtractions in (35), (36), (39), (12), (14), (16). In such cases, it may help to revert, at least occasionally, to the elementary rules of Table I and accept the necessity of matrix inversions.

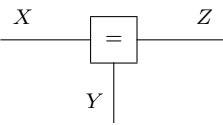
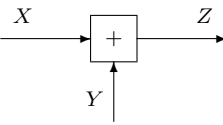
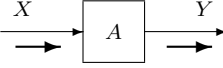
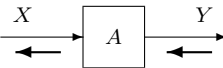

$W_{outZ} = W_{inX} + W_{inY} \quad (17)$ $W_{outZ} m_{outZ} = W_{inX} m_{inX} + W_{inY} m_{inY} \quad (18)$ $m_X = m_Y = m_Z \quad (19)$ $V_X = V_Y = V_Z \quad (20)$
Assuming Gaussians.

$V_{outZ} = V_{inX} + V_{inY} \quad (21)$ $V_{outX} = V_{inY} + V_{inZ} \quad (22)$ $m_{outZ} = m_{inX} + m_{inY} \quad (23)$ $m_{outX} = m_{inZ} - m_{inY} \quad (24)$ $m_X + m_Y - m_Z = 0 \quad (25)$ $\tilde{W}_X = \tilde{W}_Y = \tilde{W}_Z \quad (26)$
Valid for any distribution.

$V_{outY} = AV_{inX}A^H \quad (27)$ $m_{outY} = Am_{inX} \quad (28)$ $m_Y = Am_X \quad (29)$ $V_Y = AV_XA^H \quad (30)$
Valid for any distribution. $W_{outY}$ may be obtained via Table III or Table IV.

$W_{outX} = A^H W_{inY} A \quad (31)$ $W_{outX} m_{outX} = A^H W_{inY} m_{inY} \quad (32)$ $\tilde{W}_X m_X = A^H \tilde{W}_Y m_Y \quad (33)$ $\tilde{W}_X = A^H \tilde{W}_Y A \quad (34)$
Assuming Gaussians. (Valid for any distribution if the rank of $A$ equals the number of rows.) $V_{outX}$ may be obtained via Table III or Table IV.

TABLE I

MESSAGE COMPUTATION RULES FOR LINEAR BUILDING BLOCKS.

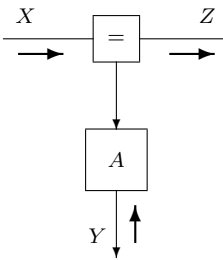
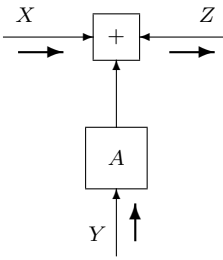

$m_{outZ} = m_{inX} + V_{inX} A^H G (m_{inY} - A m_{inX}) \quad (35)$ $V_{outZ} = V_{inX} - V_{inX} A^H G A V_{inX} \quad (36)$ $\text{with } G \triangleq (V_{inY} + A V_{inX} A^H)^{-1} \quad (37)$
Assuming Gaussians.

$m_{outZ} = -m_{inX} - A m_{inY} \quad (38)$ $W_{outZ} = W_{inX} - W_{inX} A H A^H W_{inX} \quad (39)$ $\text{with } H \triangleq (W_{inY} + A^H W_{inX} A)^{-1} \quad (40)$
Valid for any distribution.

TABLE II

MESSAGE COMPUTATION RULES FOR COMPOSITE NODES.

## 6 Conclusions

We have presented enhanced message computation tables for Gaussian messages in (the factor graph of) linear models. These tables allow to write down a variety of algorithms without additional computations or derivations.

## Acknowledgement

The first author's view on the topics of this paper has been much influenced by G. D. Forney, Jr., and P. O. Vontobel.

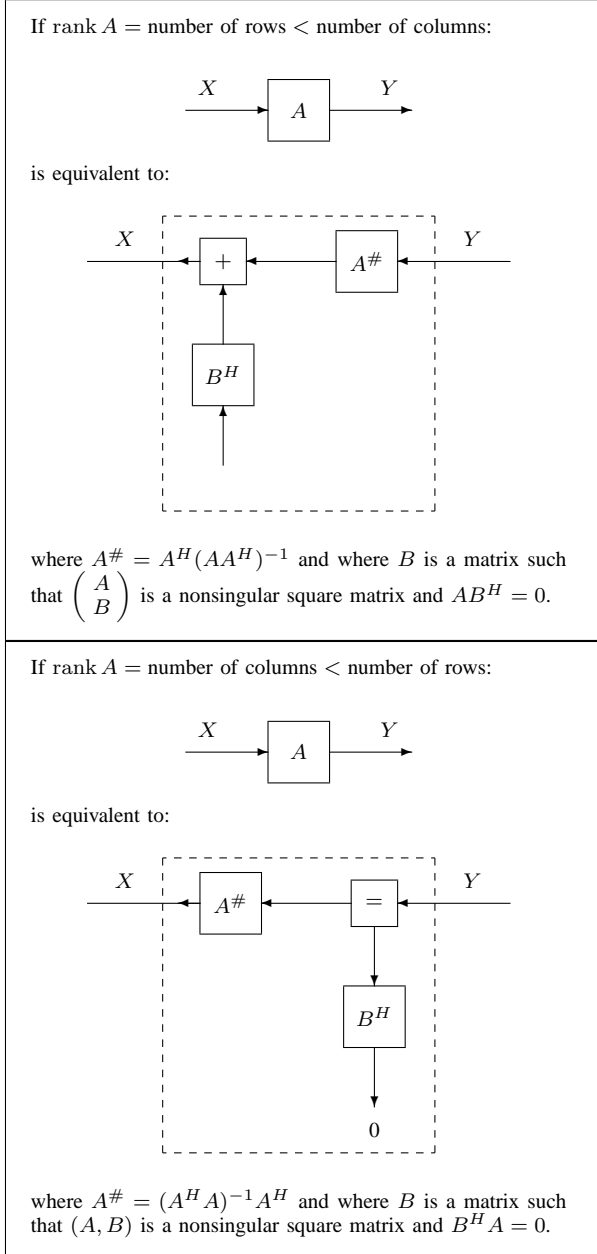


TABLE III

REVERSING A MATRIX MULTIPLICATION.

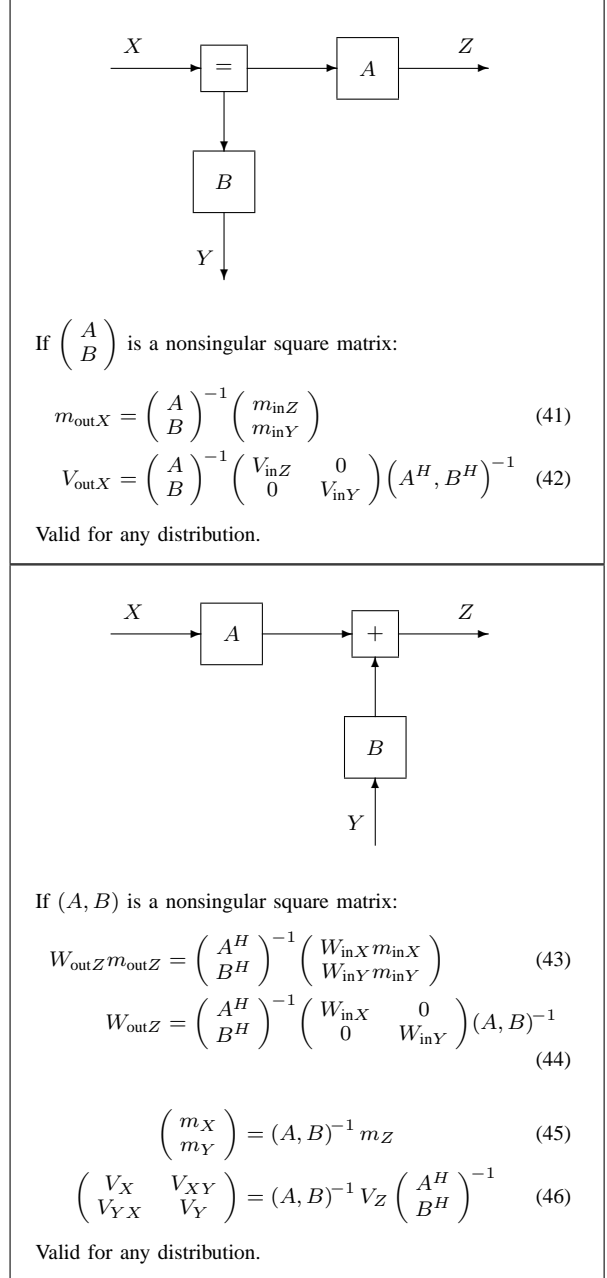


TABLE IV

MESSAGE COMPUTATION RULES FOR COMPOSITE NODES.

## Appendix: On Gaussian Distributions and LMMSE Estimation

We briefly review some basic and well known facts about Gaussian distributions, LMMSE estimation, and the equivalence of the sum-product algorithm and the max-product algorithm for Gaussians.

Let  $F = \mathbb{R}$  or  $F = \mathbb{C}$ . A general Gaussian random (column) vector  $X = (X_1, \dots, X_n)^T$  over  $F$  with mean vector  $m = (m_1, \dots, m_n)^T \in F^n$  can be written as

$$X = AU + m \quad (47)$$

where  $A$  is a nonsingular  $n \times n$  matrix over  $F$  and where  $U = (U_1, \dots, U_n)^T$  consists of independent  $F$ -valued Gaussian random variables  $U_1, \dots, U_n$  with mean zero and variance one. The covariance matrix of  $X$  is  $V = AA^H$ . The probability density of  $X$  is

$$f_X(x) \propto e^{-\beta(x-m)^H W(x-m)} \quad (48)$$

$$\propto e^{-\beta(x^H W x - 2\operatorname{Re}(x^H W m))} \quad (49)$$

for  $W = V^{-1} = (A^{-1})^H A^{-1}$  and with  $\beta = 1/2$  in the real case ( $F = \mathbb{R}$ ) and  $\beta = 1$  in the complex case ( $F = \mathbb{C}$ ). Conversely, any function of the form (48) with positive definite  $W$  may be obtained in this way with some suitable matrix  $A$ .

Now let  $Z$  be a Gaussian random (column) vector, which we partition as

$$Z = \begin{pmatrix} X \\ Y \end{pmatrix}, \quad (50)$$

where  $X$  and  $Y$  are themselves (column) vectors. The density of  $Z$  is  $f_Z(z) \propto e^{-\beta q(x,y)}$  with

$$q(x,y) = \begin{pmatrix} (x - m_X)^H & (y - m_Y)^H \end{pmatrix} \cdot \begin{pmatrix} W_X & W_{XY} \\ W_{YX} & W_Y \end{pmatrix} \begin{pmatrix} x - m_X \\ y - m_Y \end{pmatrix} \quad (51)$$

with positive definite  $W_X$  and  $W_Y$  and with  $W_{YX} = W_{XY}^H$ .

For fixed  $y$ , considered as a function of  $x$  alone, (51) becomes

$$q(x,y) = x^H W_X x - 2\operatorname{Re}\left(x^H W_X (m_X - W_X^{-1} W_{XY} (y - m_Y))\right) + \text{const.} \quad (52)$$

Comparing this with (49) yields the following theorem:

**Theorem 1 (Gaussian Conditioning Theorem).** If  $X$  and  $Y$  are jointly Gaussian with joint distribution  $\propto e^{-\beta q(x,y)}$  as above, then *conditioned on*  $Y = y$  (for any fixed  $y$ ),  $X$  is Gaussian with mean

$$E[X|Y = y] = m_X - W_X^{-1} W_{XY} (y - m_Y) \quad (53)$$

and covariance matrix  $W_X^{-1}$ .  $\square$

Note that  $E[X|Y = y]$  is both the MAP (maximum a posteriori) estimate and the MMSE (minimum mean

squared error) estimate of  $X$  given the observation  $Y = y$ . According to (53),  $E[X|Y = y]$  is an *affine* (= linear with offset) function of the observation  $y$ . We thus have the following theorem:

**Theorem 2.** For jointly Gaussian random variables or vectors  $X$  and  $Y$ , the MAP estimate of  $X$  from the observation  $Y = y$  is an affine function of  $y$  and coincides both with the MMSE estimate and the LMMSE estimate.  $\square$

Note that, in this theorem as well as in the following theorem, the ‘‘L’’ in LMMSE must be understood as ‘‘affine’’ (= linear with offset).

**Theorem 3 (LMMSE Via Gaussian MAP Theorem).**

Let  $X$  and  $Y$  be random variables (or vectors) with arbitrary distributions but with finite means and with finite second-order moments. Then the LMMSE estimate of  $X$  based on the observation  $Y = y$  may be obtained by pretending that  $X$  and  $Y$  are jointly Gaussian (with their actual means and second-order moments) and forming the corresponding MAP estimate.  $\square$

The proof follows from noting that, according to the orthogonality principle, the LMMSE estimate of  $X$  based on  $Y = y$  depends only on the means and second-order moments.

In a different direction, we also recall the following fact.

**Theorem 4 (Gaussian Max/Int Theorem).** Let  $q(x,y)$  be a quadratic form as in (51) with  $W_X$  positive definite. Then

$$\int_{-\infty}^{\infty} e^{-q(x,y)} dx \propto \max_x e^{-q(x,y)} \quad (54)$$

$$= e^{-\min_x q(x,y)}. \quad (55)$$

$\square$

(A proof may be found in [1].) It follows that, for Gaussian factor graphs, the sum-product algorithm coincides with the max-product algorithm.

## References

- [1] 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<sup>th</sup> birthday), R. E. Blahut and R. Koetter, eds., Kluwer, 2002, pp. 113–135.
- [2] H.-A. Loeliger, ‘‘An introduction to factor graphs,’’ *IEEE Signal Proc. Mag.*, Jan. 2004, pp. 28–41.
- [3] 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.
- [4] G. D. Forney, Jr., ‘‘Codes on graphs: normal realizations,’’ *IEEE Trans. Information Theory*, vol. 47, no. 2, pp. 520–548, 2001.
- [5] S. Kori, *A Factor Graph Approach to Signal Modelling, System Identification and Filtering*. PhD thesis at ETH Zurich No 16170, Hartung Gorre Verlag, Konstanz, 2005.
- [6] S. Haykin, *Adaptive Filter Theory*. 3rd ed., Prentice Hall, 1996.