

On Factor Graphs

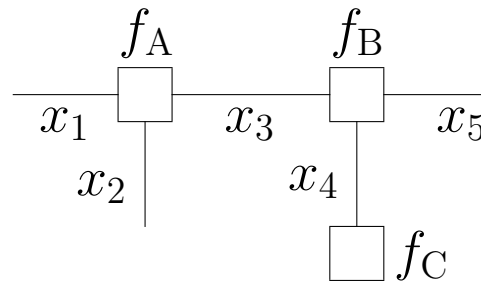
1. Basics
2. Fourier transform and duality
3. Factor graphs and error correcting codes
4. Kalman filtering and recursive least squares
5. Signal processing with factor graphs

Factor Graphs

represent the **factorization** of a function of several variables.
We use **Forney-style** factor graphs. (Forney 2001)

Example:

$$f(x_1, x_2, x_3, x_4, x_5) = f_A(x_1, x_2, x_3) \cdot f_B(x_3, x_4, x_5) \cdot f_C(x_4).$$



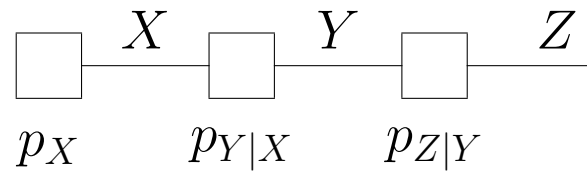
Rules:

- A **node** for every **factor**.
- An **edge** or **half-edge** for every **variable**.
- Node g is connected to edge x iff variable x appears in factor g .

Example:

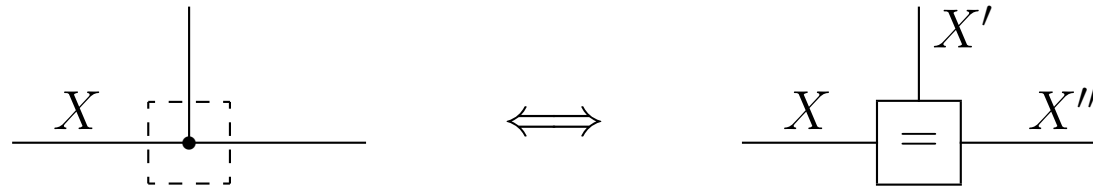
Markov Chain

$$p_{XYZ}(x, y, z) = p_X(x) p_{Y|X}(y|x) p_{Z|Y}(z|y).$$



Branching Points

are formally treated as equality constraint nodes:



There thus arise new variables X' and X'' and a new factor

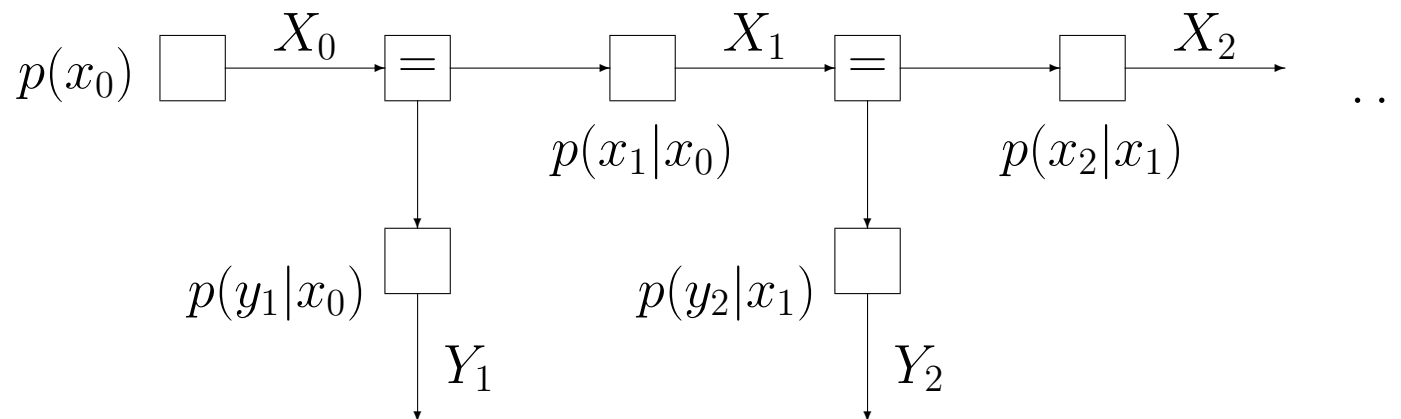
$$f_{=} (x, x', x'') \triangleq \delta(x - x') \delta(x - x'').$$

For every **valid** configuration, $X = X' = X''$ holds.

Example:

Hidden-Markov Model

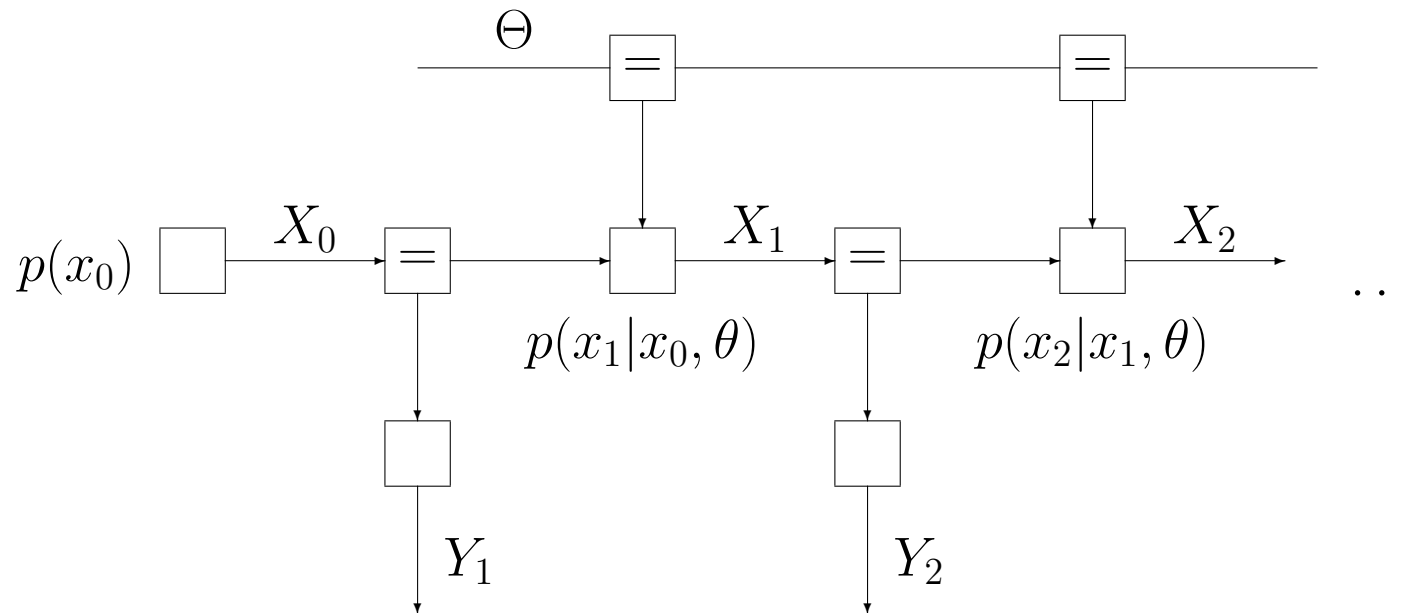
$$p(x_0, x_1, x_2, \dots, x_n, y_1, y_2, \dots, y_n) = p(x_0) \prod_{k=1}^n p(x_k | x_{k-1}) p(y_k | x_{k-1})$$



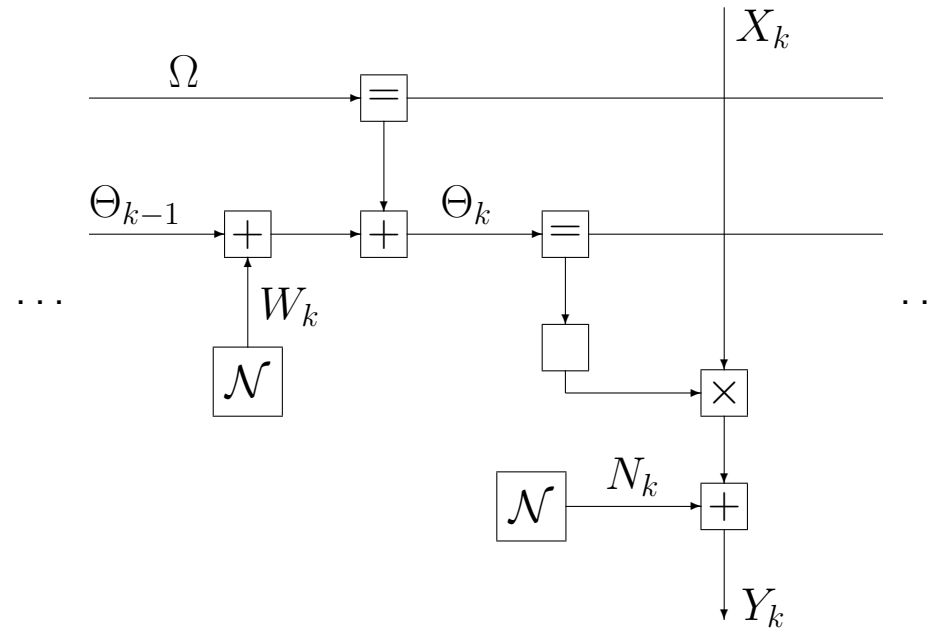
Example:

Hidden-Markov Model with parameter(s) Θ

$$p(x_0, x_1, x_2, \dots, x_n, y_1, y_2, \dots, y_n \mid \theta) = p(x_0) \prod_{k=1}^n p(x_k \mid x_{k-1}, \theta) p(y_k \mid x_{k-1})$$

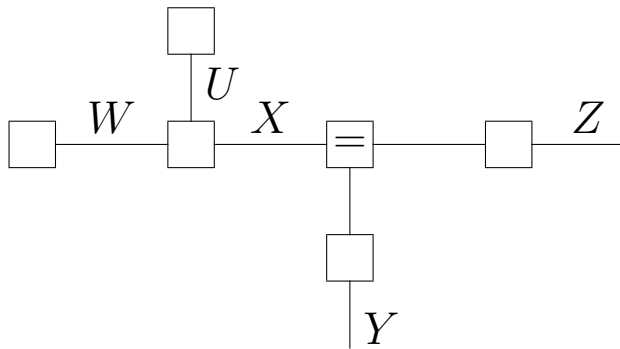


A Real-World Example (to be explained later)

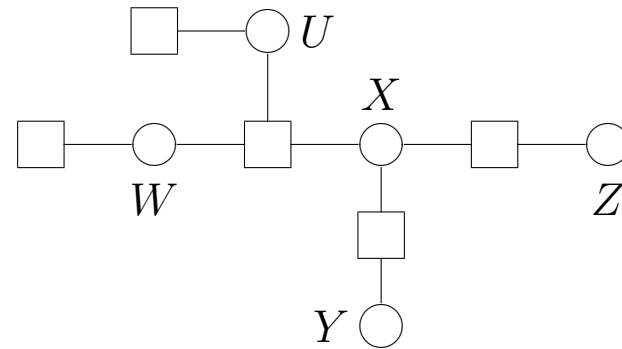


Different Notations for Graphical Models

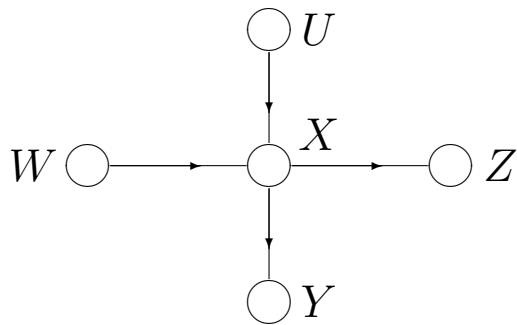
Example: $p(u, w, x, y, z) = p(u)p(w)p(x|u, w)p(y|x)p(z|x)$.



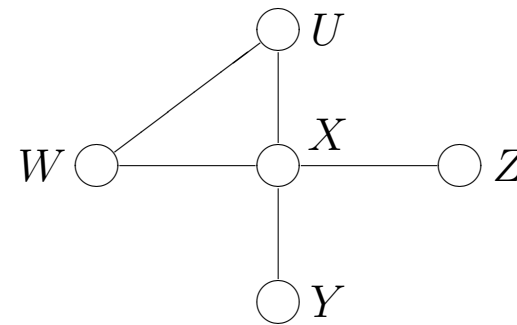
Forney-style factor graph.



Original factor graph.



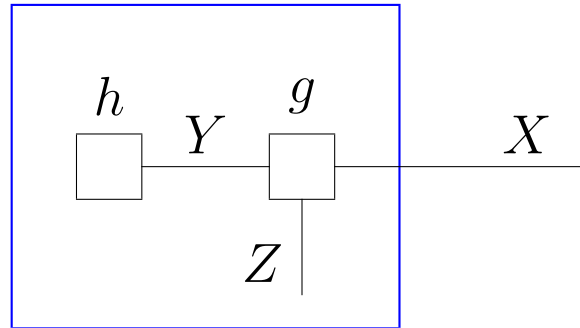
Bayesian network.



Markov random field (MRF).

Closing Boxes = Elimination of Variables

Example:



From

$$f(x, y, z) = g(x, y, z)h(y)$$

to

$$f(x) \triangleq \sum_{y,z} f(x, y, z)$$

or to

$$f(x) \triangleq \max_{y,z} f(x, y, z)$$

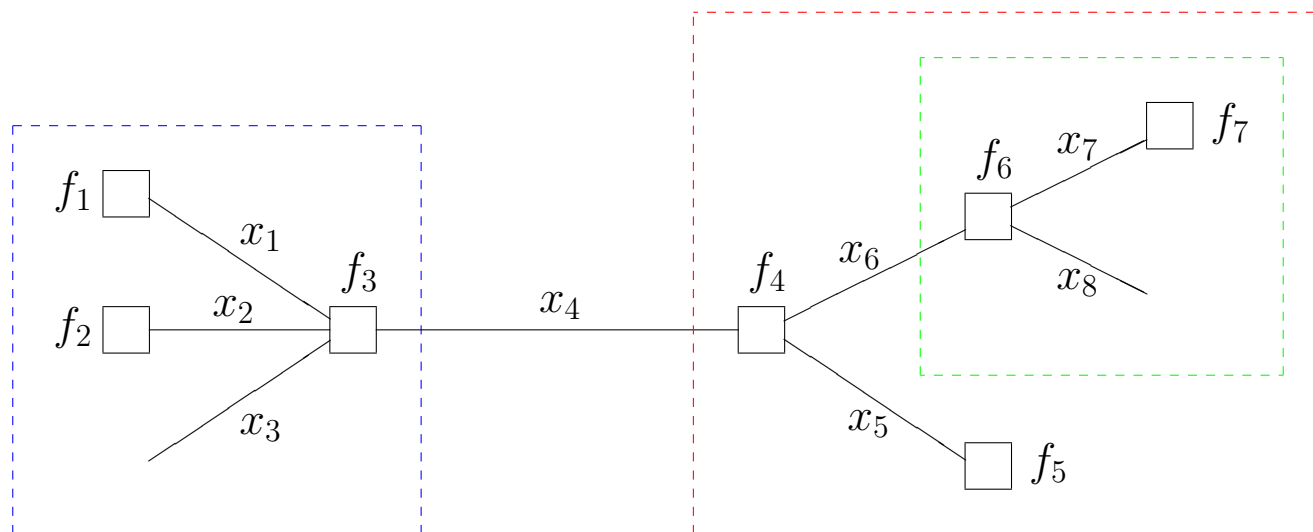
Elimination of Variables (cont'd)

- by **summation** (or integration) over the internal variables:
marginalization of probabilities; Bayesian estimation
- by **maximization** over the internal variables:
set theoretic (or logical) modeling; maximum-likelihood estimation

In either case, only the **valid configurations** (configurations (x, y, z) with $f(x, y, z) > 0$) contribute.

Recursive Elimination of Variables

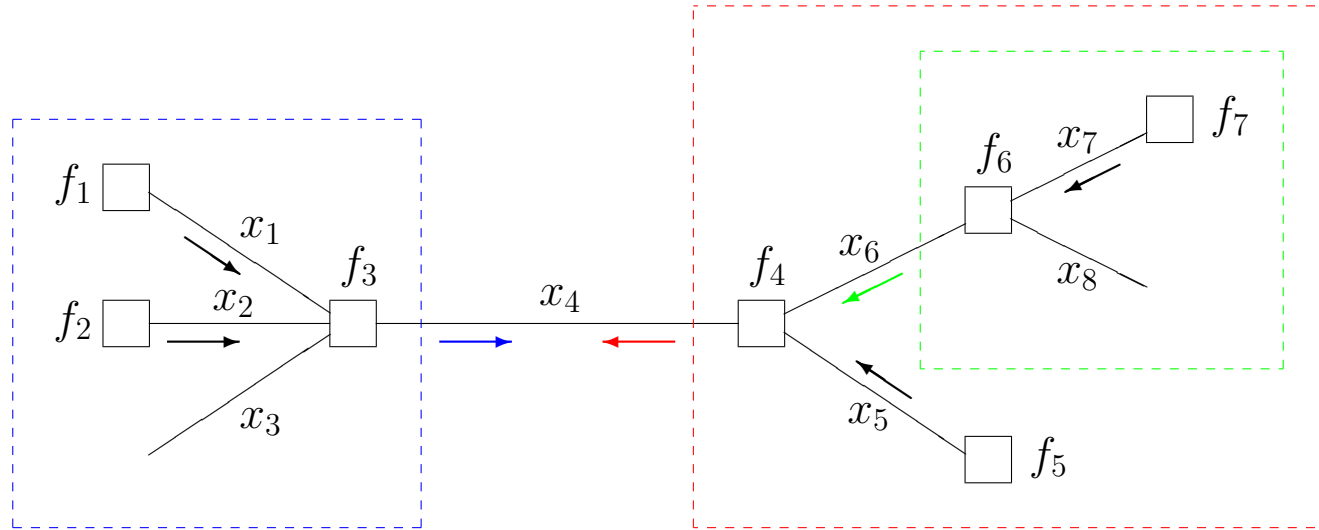
Example:



$$f(x_1, \dots, x_8) = \left(f_1(x_1) f_2(x_2) f_3(x_1, x_2, x_3, x_4) \right) \cdot \left(f_4(x_4, x_5, x_6) f_5(x_5) \left(f_6(x_6, x_7, x_8) f_7(x_7) \right) \right)$$

$$f(x_4) \triangleq \sum_{\text{all except } x_4} f(x_1, \dots, x_8) = ?$$

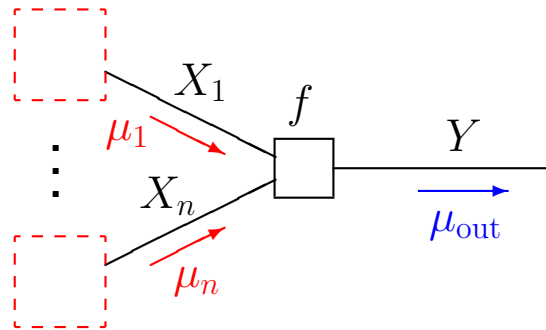
“Closing Boxes” as Message Passing



$$f(x_4) = \left(\sum_{x_1} \sum_{x_2} \sum_{x_3} f_3(x_1, x_2, x_3, x_4) f_1(x_1) f_2(x_2) \right) \cdot \left(\sum_{x_5} \sum_{x_6} f_4(x_4, x_5, x_6) f_5(x_5) \left(\sum_{x_7} \sum_{x_8} f_6(x_6, x_7, x_8) f_7(x_7) \right) \right)$$

Summary-Product Message Update Rule

Each message is a **function** (usually a scaled pmf/pdf) of the variable associated with the corresponding edge. This function is a **summary** of everything “behind” this edge.



$$\mu_{\text{out}}(y) \triangleq \sum_{x_1} \cdots \sum_{x_n} f(x_1, \dots, x_n, y) \mu_1(x_1) \cdots \mu_n(x_n)$$

or (e.g.)

$$\mu_{\text{out}}(y) \triangleq \max_{x_1} \cdots \max_{x_n} f(x_1, \dots, x_n, y) \mu_1(x_1) \cdots \mu_n(x_n)$$

Summary Propagation

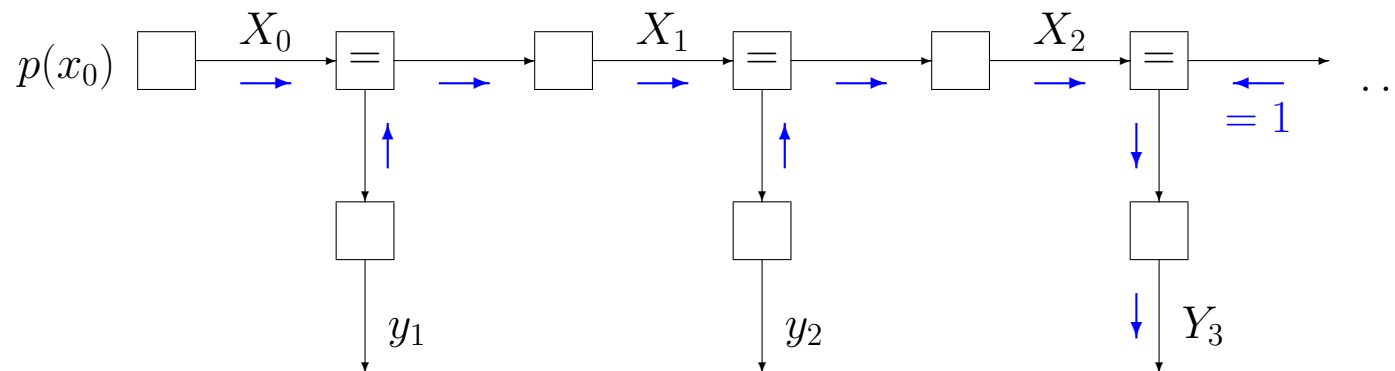
- Messages are **summaries** of the graph “behind” them.
- “Summation” may be replaced by **other “summary operators”** (e.g., “max”)
- Message computation is **local**.
- Computes **exact** marginals/summaries only for **graphs without cycles** ...
- ...but it may be applied also to **graphs with cycles**. We then obtain **iterative** algorithms, which compute some **approximation** of the true marginals. The approximation may be poor! But **this is how capacity approaching codes are decoded**.

Example of Message Passing:

Hidden-Markov Model: Prediction

$$\begin{aligned}
 p(y_n \mid y_1, \dots, y_{n-1}) &= \frac{p(y_1, \dots, y_n)}{p(y_1, \dots, y_{n-1})} \\
 &\propto p(y_1, \dots, y_n) \quad (\text{for fixed } y_1, \dots, y_{n-1}) \\
 &= \sum_{x_0, x_1, \dots} \sum_{y_{n+1}, y_{n+2}, \dots} p(x_0, x_1, \dots, y_1, y_2, \dots)
 \end{aligned}$$

For $n = 3$:

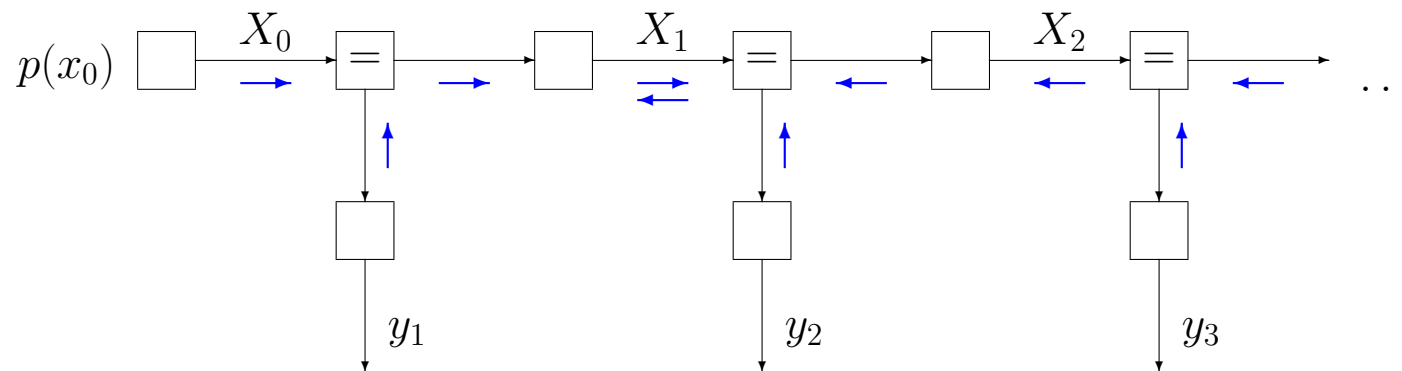


Example of Message Passing:

Hidden-Markov Model: State Estimation

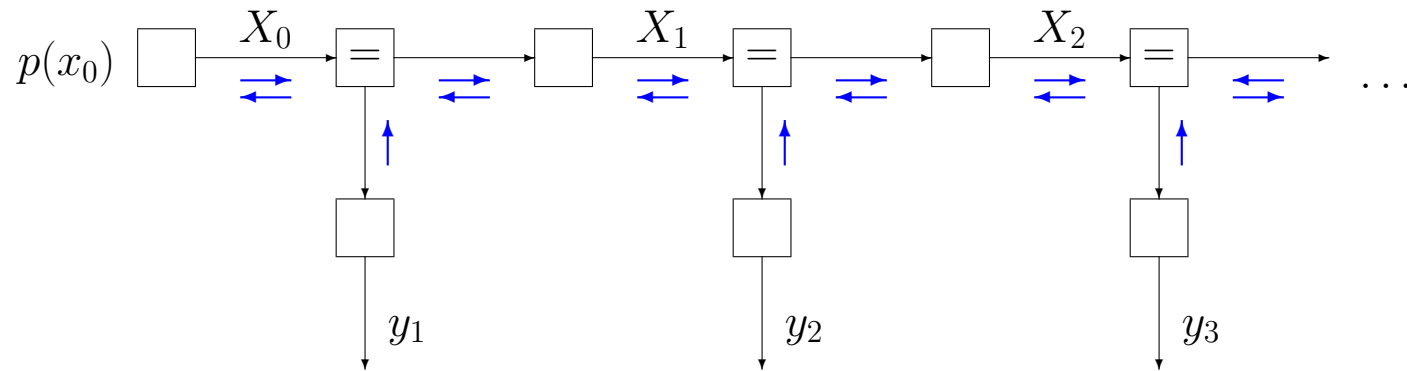
$$\begin{aligned} p(x_m \mid y_1, y_2, \dots) &= \frac{p(x_m, y_1, y_2, \dots)}{p(y_1, y_2, \dots)} \\ &\propto p(x_m, y_1, y_2, \dots) \quad (\text{for fixed } y_1, y_2, \dots) \\ &= \sum_{\substack{x_0, x_1, \dots \\ \text{except } x_m}} p(x_0, x_1, \dots, y_1, y_2, \dots) \end{aligned}$$

For $m = 1$:



Example of Message Passing:

State Estimation for All States Simultaneously



Computation by forward sweep and (independent) backward sweep.

Topics

1. Basics
2. Fourier transform
3. Factor graphs and error correcting codes
4. Kalman filtering and recursive least squares
5. Signal processing with factor graphs

Fourier Transform

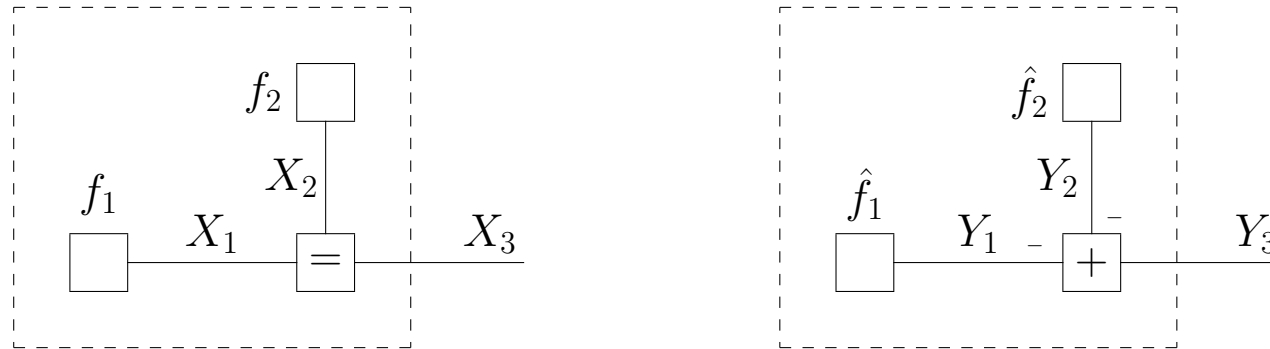
(Forney 2001; Kschischang / Mao)

The Fourier transform of a multi-variable function can be carried out directly in the (Forney-style) factor graph (which may have cycles!):

- Replace each variable by its dual (“frequency”) variable.
- Replace each factor by its Fourier transform. If some factor is the membership indicator function $\delta_V(\cdot)$ of a vector space V , its “Fourier transform” is the membership indicator function $\delta_{V^\perp}(\cdot)$ of the orthogonal complement V^\perp .
- For each edge, introduce a minus sign into one of the two adjacent factors.

For this recipe to work, all variables of interest must be external, i.e., represented by half edges.

Fourier Transform: Example



The Fourier transform of the pointwise multiplication

$$\begin{aligned} f(x_3) &= \sum_{x_1, x_2} f_1(x_1) f_2(x_2) \delta(x_1 - x_3) \delta(x_2 - x_3) \\ &= f_1(x_3) f_2(x_3) \end{aligned}$$

is the convolution

$$\begin{aligned} \hat{f}(y_3) &= \sum_{y_1, y_2} \hat{f}_1(y_1) \hat{f}_2(y_2) \delta(y_3 - y_2 - y_1) \\ &= \sum_{y_2} \hat{f}_1(y_3 - y_2) \hat{f}_2(y_2). \end{aligned}$$

Topics

1. Basics
2. Fourier transform
3. Factor graphs and error correcting codes
4. Kalman filtering and recursive least squares
5. Signal processing with factor graphs

Factor Graph from Parity Check Matrix

Example: $(7, 4, 3)$ binary Hamming code. ($F \triangleq \text{GF}(2)$.)

$$C = \{x \in F^n : Hx^T = 0\}$$

with

$$H = \begin{pmatrix} 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 1 & 0 & 1 \end{pmatrix}$$

The membership indicator function

$$I_C : F^n \rightarrow \{0, 1\} : x \mapsto \begin{cases} 1, & \text{if } x \in C \\ 0, & \text{else} \end{cases}$$

of this code may be written as

$$I_C(x_1, \dots, x_n) = \delta(x_1 \oplus x_2 \oplus x_3 \oplus x_5) \cdot \delta(x_2 \oplus x_3 \oplus x_4 \oplus x_6) \cdot \delta(x_3 \oplus x_4 \oplus x_5 \oplus x_7)$$

where \oplus denotes addition modulo 2. Each factor corresponds to one row of the parity check matrix.

Factor Graph from Parity Check Matrix (cont'd)

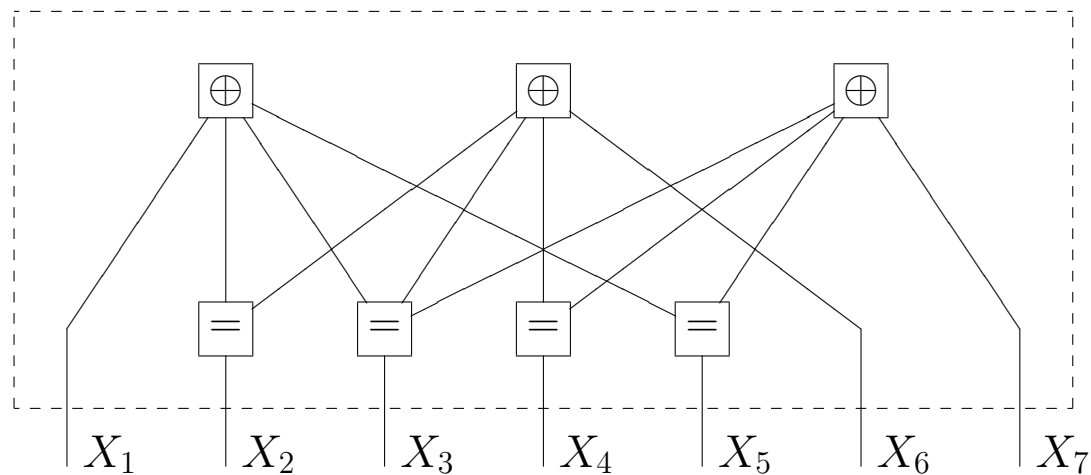
Essentially the same as a [Tanner graph](#)

Example: $(7, 4, 3)$ binary Hamming code.

$$C = \{x \in F^n : Hx^T = 0\}$$

with

$$H = \begin{pmatrix} 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 1 & 0 & 1 \end{pmatrix}$$



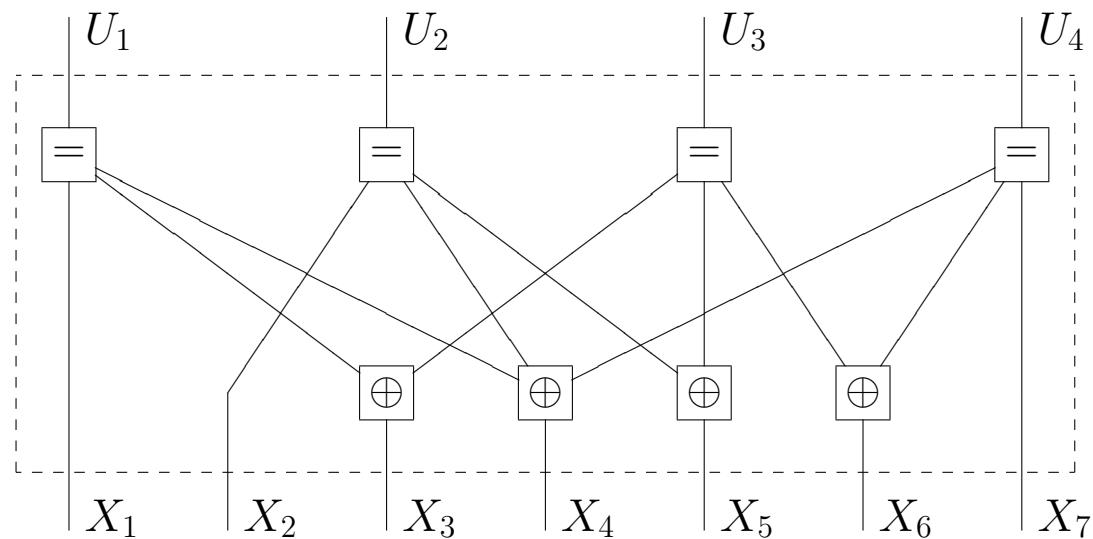
Factor Graph from Generator Matrix

Example: $(7, 4, 3)$ binary Hamming code is the image of

$$F^k \rightarrow F^n : u \mapsto uG$$

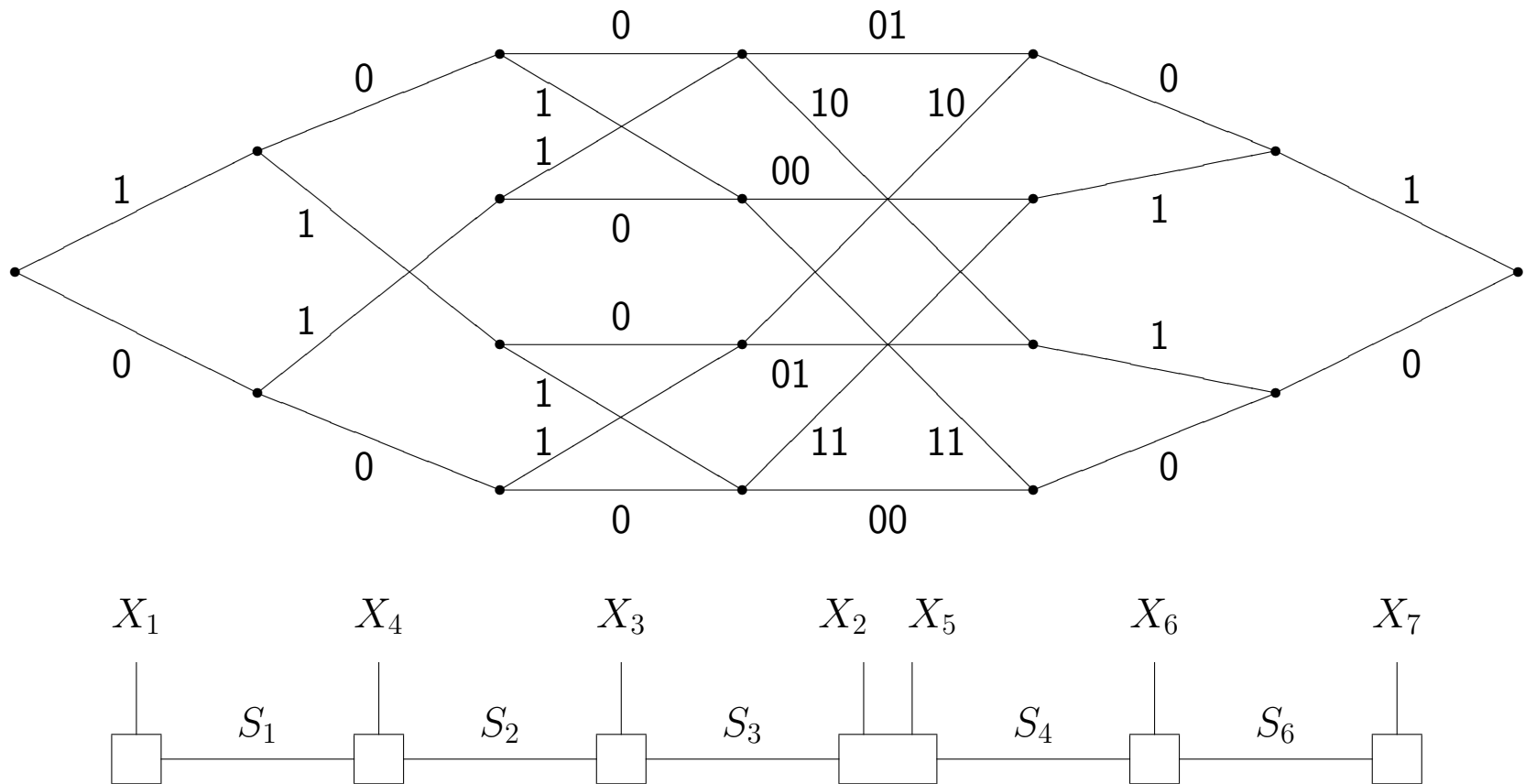
with

$$G = \begin{pmatrix} 1 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 \end{pmatrix}.$$



Factor Graph Corresponding to Trellis

Example: $(7, 4, 3)$ binary Hamming code

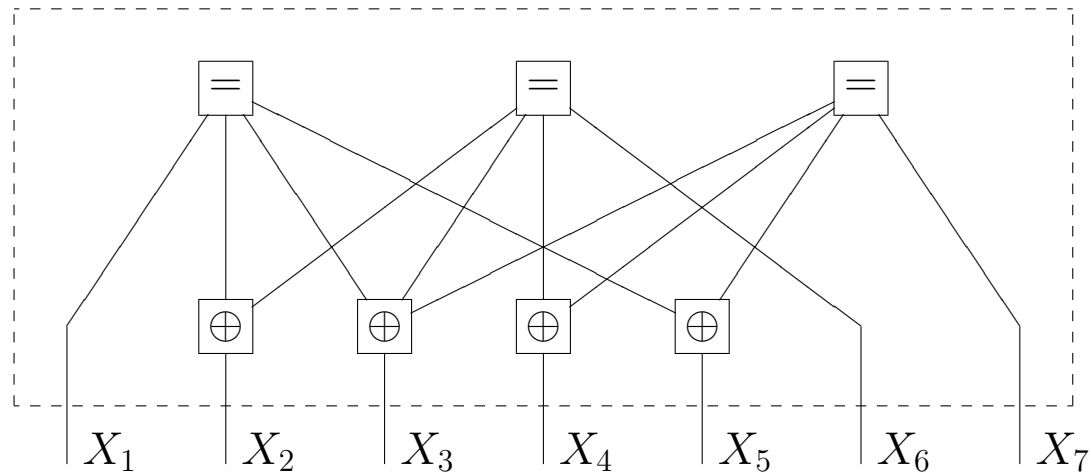


Factor Graph of Dual Code

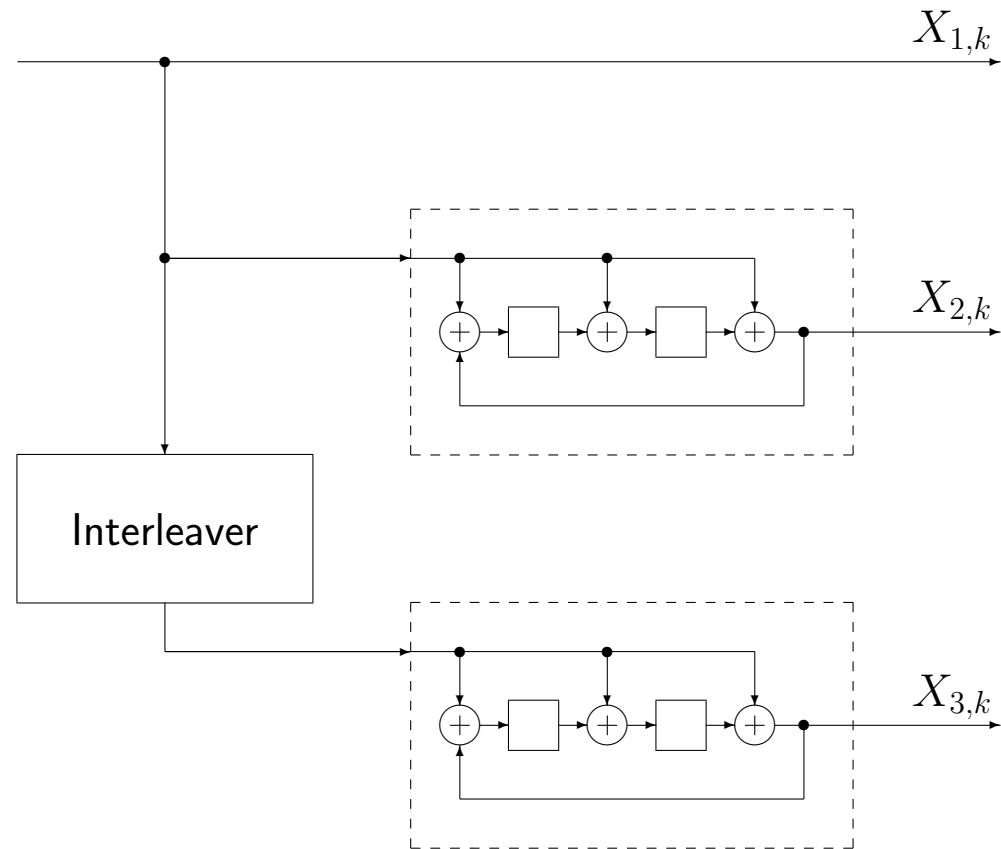
is obtained by **interchanging parity check nodes and equality check nodes** (Kschischang, Forney).

Works only for Forney-style factor graphs where all code symbols are external (half-edge) variables.

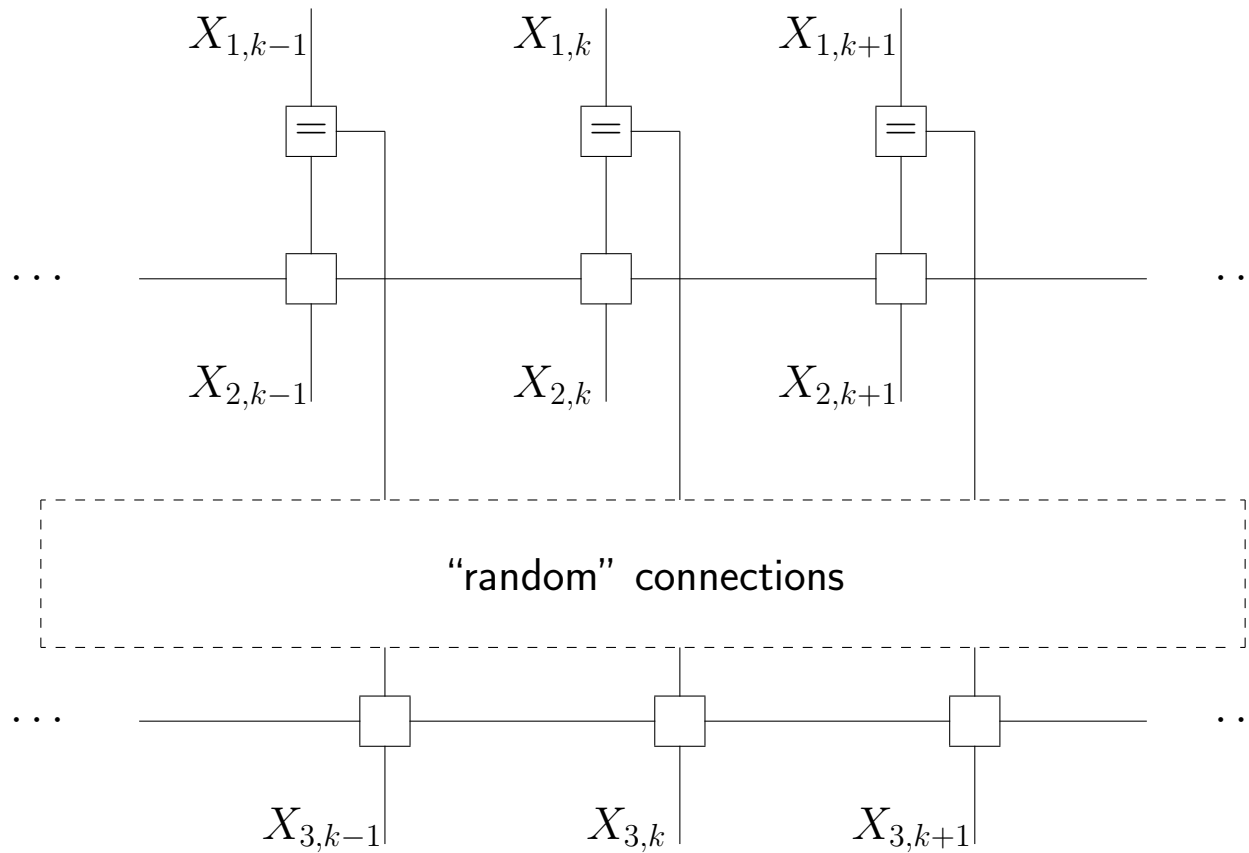
Example: dual of $(7, 4, 3)$ binary Hamming code



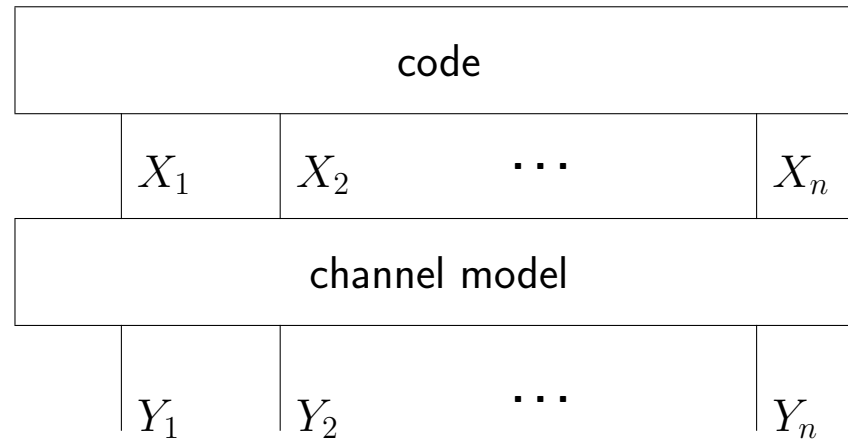
Encoder of a Turbo Code



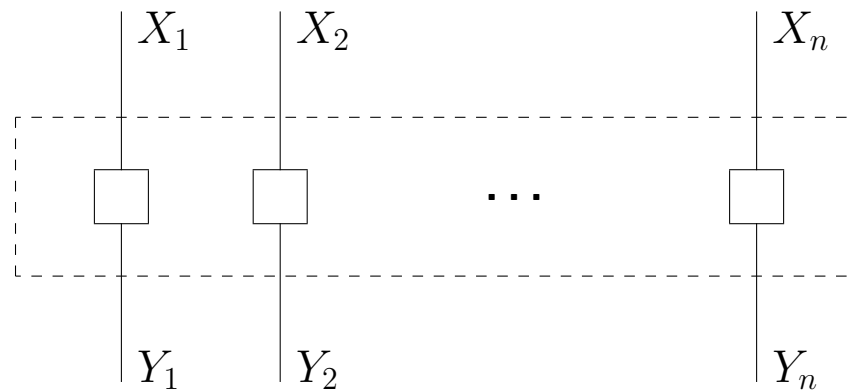
Factor Graph of a Turbo Code



Factor Graph for Joint Code / Channel Model



Example: memoryless channel:



Topics

1. Basics
2. Fourier transform and duality
3. Factor graphs and error correcting codes
4. Kalman filtering and recursive least squares
5. Signal processing with factor graphs

Linear State Space Model

$$X_k = A_k X_{k-1} + B_k U_k$$

$$Y_k = C_k X_k$$

$$k = 1, 2, 3, \dots,$$

with

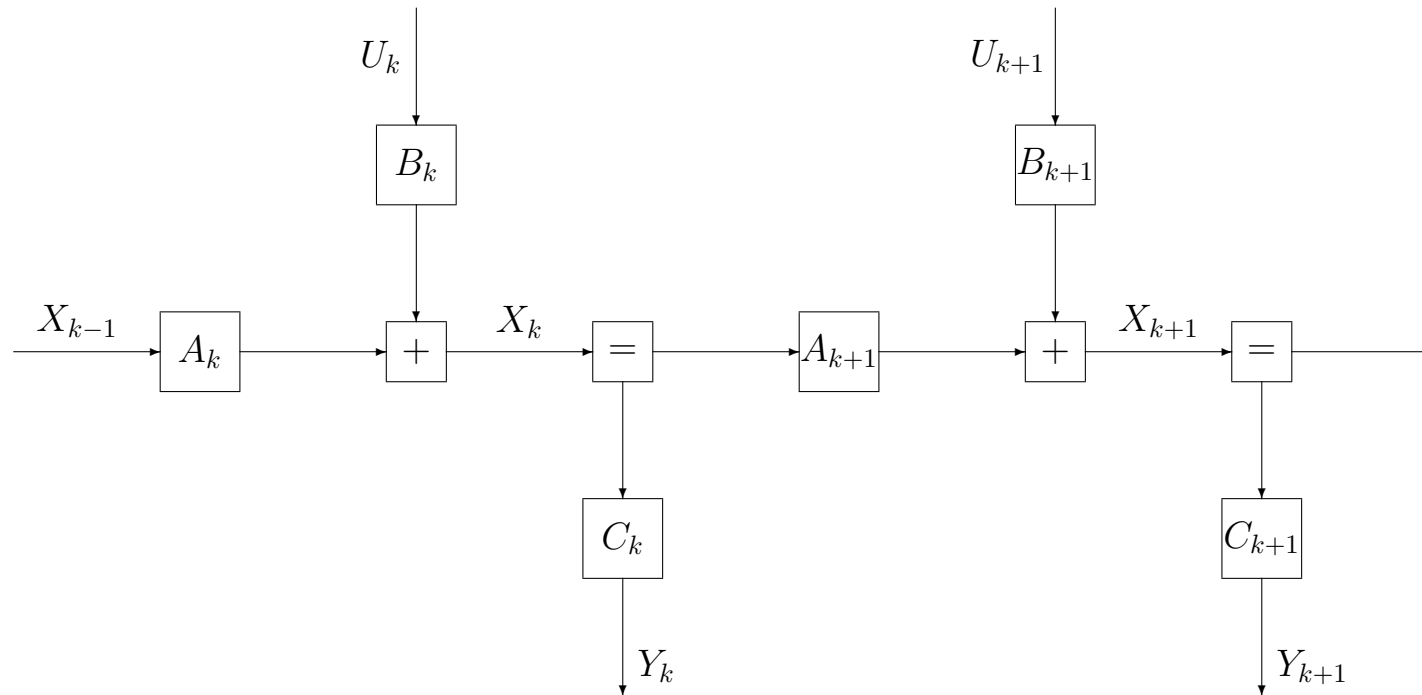
X_k state vector at discrete time k

U_k input (vector or scalar) at time k

Y_k output (vector or scalar) at time k

A_k, B_k, C_k (known) matrices of suitable dimensions.

Factor Graph of Linear State Space Model



We will use only (multidimensional) **Gaussian messages**.

Example: Equalization

$$\tilde{Y}_k = \sum_{\ell=0}^M h_{\ell} U_{k-\ell} + Z_k,$$

with

U_k real-valued transmitted symbols
 h_0, \dots, h_M known real channel coefficients
 Z_k white Gaussian noise

State space formulation:

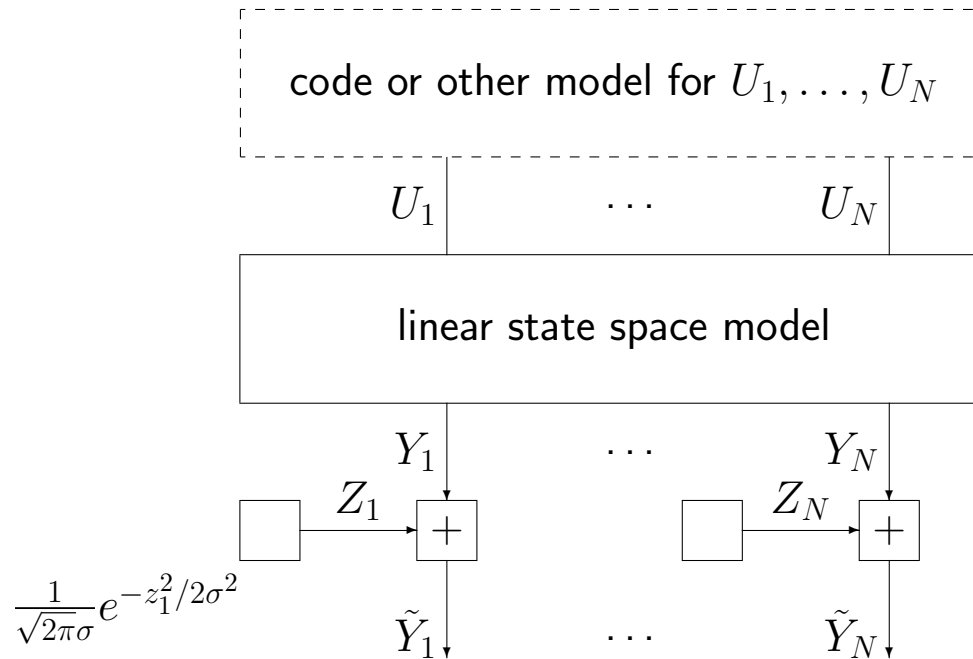
$$\begin{aligned} X_k &= A_k X_{k-1} + B_k U_k \\ Y_k &= C_k X_k \end{aligned}$$

with

$$\begin{aligned} X_k &\triangleq (U_k, \dots, U_{k-M})^T \\ B_k &\triangleq (1, 0, \dots, 0)^T \\ C_k &\triangleq (h_0, h_1, \dots, h_M) \\ Y_k &\triangleq \tilde{Y}_k - Z_k \text{ (noise-free output)} \end{aligned}$$

$$A_k \triangleq \begin{pmatrix} 0 & 0 \\ I_M & 0 \end{pmatrix}$$

Equalization: Factor Graph



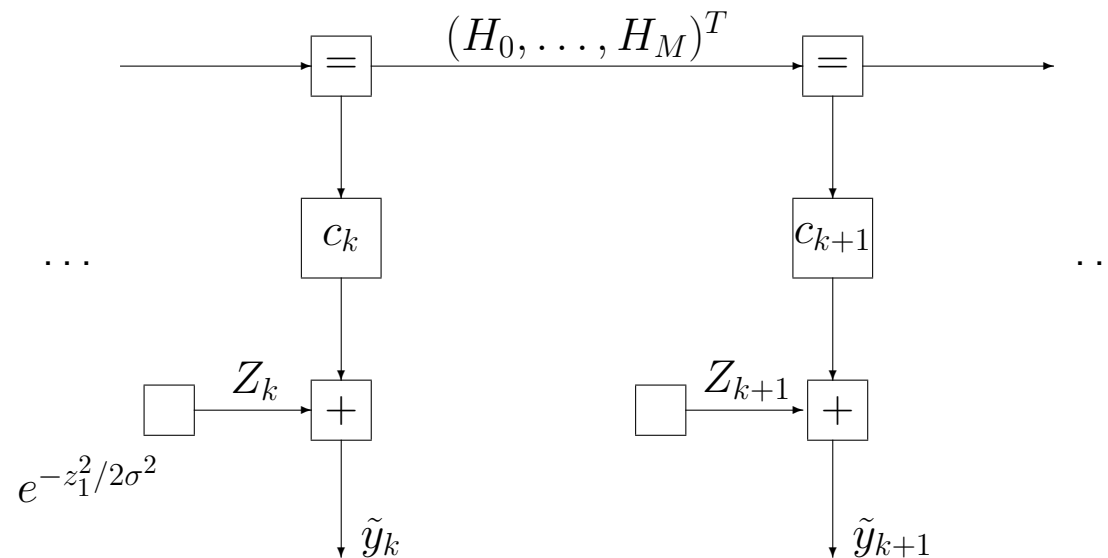
Forward-backward Gaussian message passing
 \implies LMMSE/Kalman equalization (many versions!)

Example: Adaptive FIR Filter—RLS Algorithm

For known inputs u_k and known noisy outputs \tilde{y}_k with

$$\tilde{y}_k = \sum_{\ell=0}^M h_{\ell} u_{k-\ell} + z_k$$

$k = 1, 2, \dots$, determine filter coefficients h_0, \dots, h_M to minimize $\sum_k z_k^2$ or, equivalently, to maximize $\prod_k e^{-z_k^2/2\sigma^2}$.



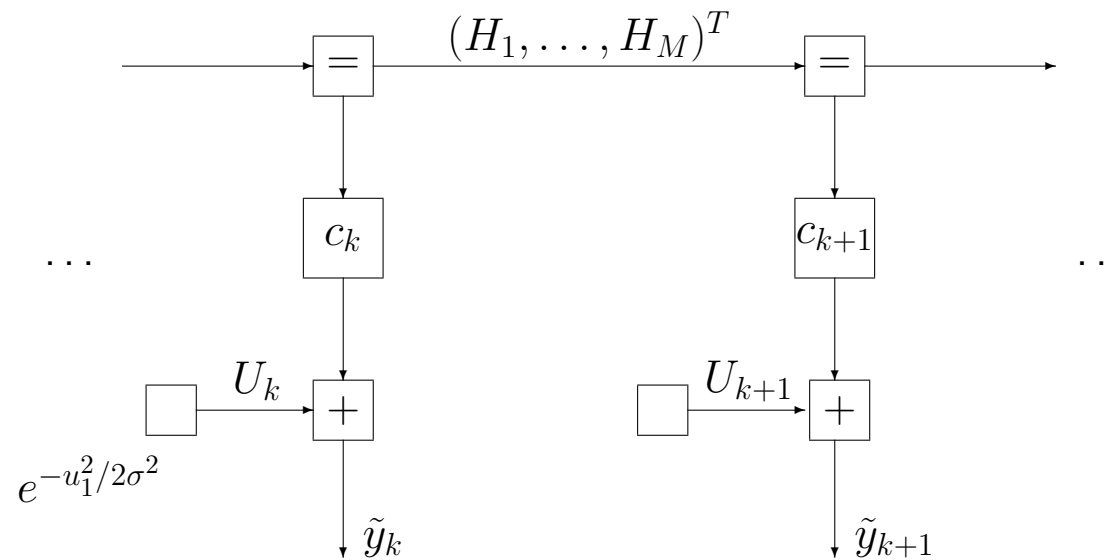
with $c_k \triangleq (u_k, u_{k-1}, \dots, u_{k-M})$.

Example: Adaptive Autoregressive Filter—LPC Analysis

For known signal \tilde{y}_k with

$$\tilde{y}_k = \sum_{\ell=1}^M h_{\ell} \tilde{y}_{k-\ell} + u_k$$

$k = 1, 2, \dots$, determine filter coefficients h_1, \dots, h_M to minimize $\sum_k u_k^2$ or, equivalently, to maximize $\prod_k e^{-u_k^2/2\sigma^2}$.



with $c_k \triangleq (\tilde{y}_{k-1}, \dots, \tilde{y}_{k-M})$.

Example: **Multi-User Communication** (Wang/Poor, Boutros/Caire)

$$\tilde{Y} = HU + Z$$

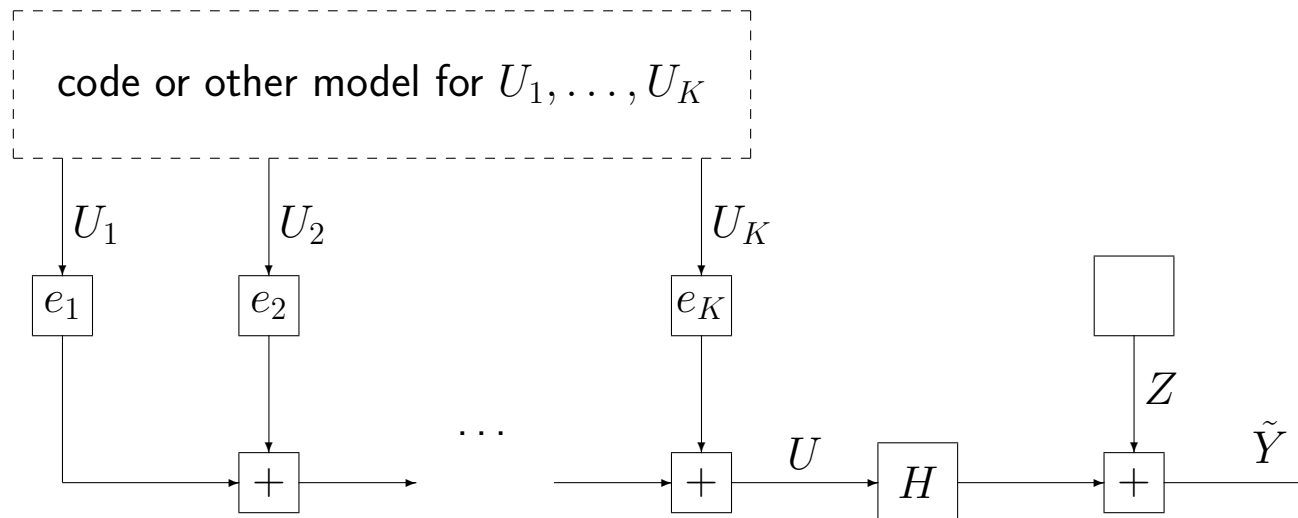
with

$U = (U_1, \dots, U_K)^T$ signals transmitted by K -users

$\tilde{Y} = (\tilde{Y}_1, \dots, \tilde{Y}_N)^T$ received signal

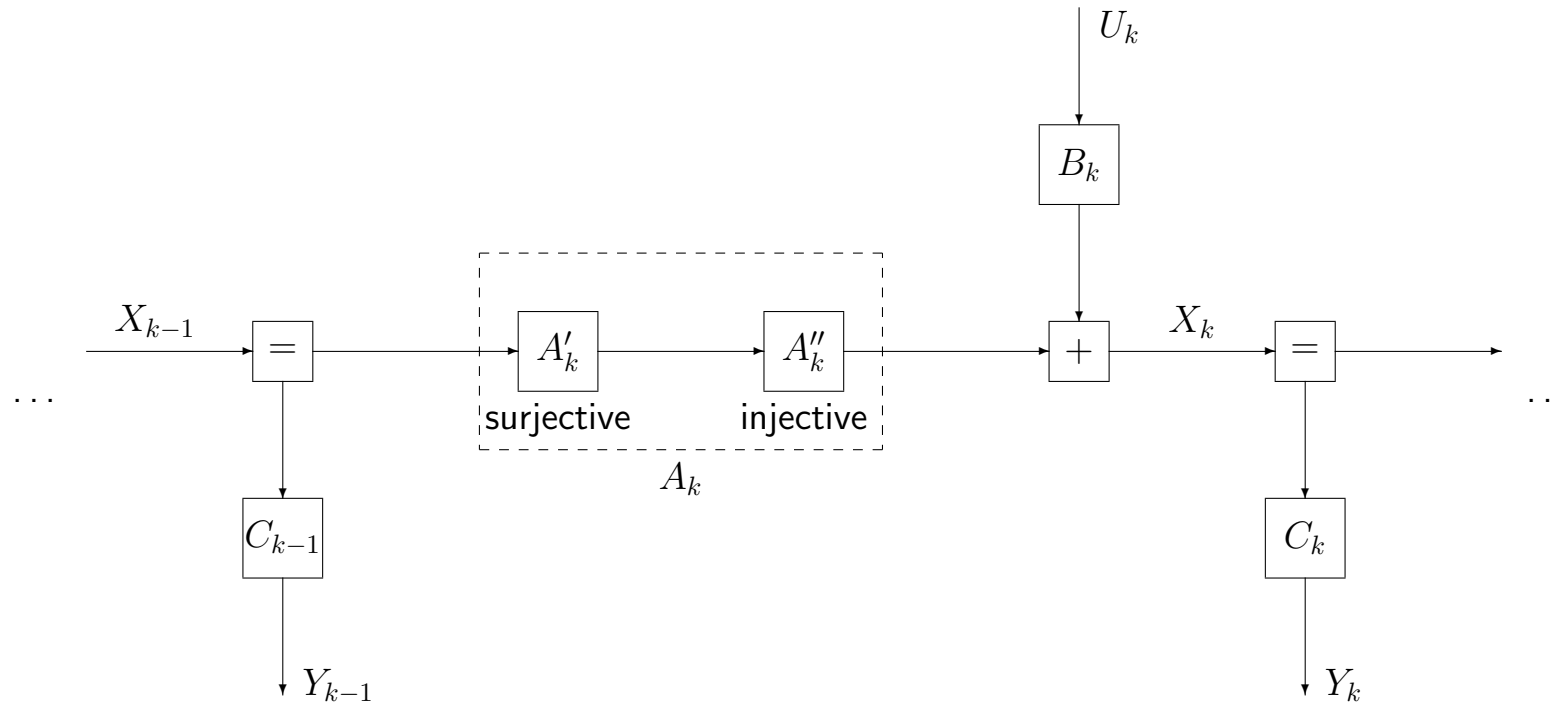
$Z = (Z_1, \dots, Z_N)^T$ Gaussian noise

$H \in \mathbb{R}^{N \times K}$



with $e_1 \triangleq (1, 0, \dots, 0)^T$, $e_2 \triangleq (0, 1, 0, \dots, 0)^T$, etc.

Linear State Space Model (again)



More background:

Facts on Gaussian Estimation

1. If vectors X and Y are jointly Gaussian, then:
 - (a) The MAP estimate of X from $Y = y$ is $E[X|Y = y]$, which is an affine (linear + offset) function of y .
 - (b) Therefore, MAP estimation = MMSE estimation = LMMSE estimation.
2. Even if X and Y are not jointly Gaussian, the LMMSE estimate of X from $Y = y$ 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.
3. For Gaussian messages, the sum-product (integral-product) algorithm coincides with the max-product algorithm.

Gaussian Messages and Marginals

General Gaussian distribution:

$$e^{-\frac{1}{2}(x-m)^T W (x-m)}$$

with mean vector m and covariance matrix $V = W^{-1}$.

Parameterizations of messages and marginal at some edge X :

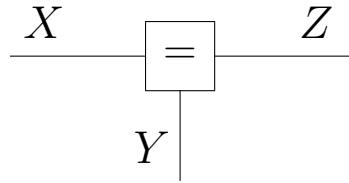
messages: $m_{\text{in}X}$ and $V_{\text{in}X}$ ($m_{\text{out}X}$ and $V_{\text{out}X}$)
 $W_{\text{in}X} m_{\text{in}X}$ and $W_{\text{in}X} \triangleq V_{\text{in}X}^{-1}$ ($W_{\text{out}X} m_{\text{out}X}$ and $W_{\text{out}X}$)

marginals: m_X and $V_X = (W_{\text{in}X} + W_{\text{out}X})^{-1} = W_X^{-1}$
 $\tilde{W}_X m_X$ and $\tilde{W}_X \triangleq (V_{\text{in}X} + V_{\text{out}X})^{-1} \neq V_X^{-1}$

Main concern:

avoid frequent matrix inversions from V to W and vice versa.

Computation Rules for Gaussian Messages I



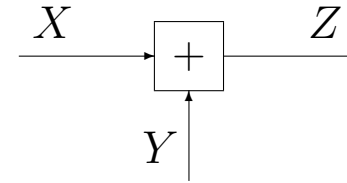
$$W_{\text{out}Z} = W_{\text{in}X} + W_{\text{in}Y}$$

$$W_{\text{out}Z} m_{\text{out}Z} = W_{\text{in}X} m_{\text{in}X} + W_{\text{in}Y} m_{\text{in}Y}$$

$$m_X = m_Y = m_Z$$

$$V_X = V_Y = V_Z$$

Assuming Gaussians.



$$V_{\text{out}Z} = V_{\text{in}X} + V_{\text{in}Y}$$

$$V_{\text{out}X} = V_{\text{in}Y} + V_{\text{in}Z}$$

$$m_{\text{out}Z} = m_{\text{in}X} + m_{\text{in}Y}$$

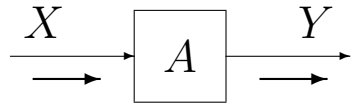
$$m_{\text{out}X} = m_{\text{in}Z} - m_{\text{in}Y}$$

$$m_X + m_Y - m_Z = 0$$

$$\tilde{W}_X = \tilde{W}_Y = \tilde{W}_Z$$

Valid for any distribution.

Computation Rules for Gaussian Messages II



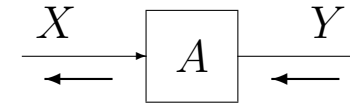
$$V_{\text{out}Y} = A V_{\text{in}X} A^H$$

$$m_{\text{out}Y} = A m_{\text{in}X}$$

$$m_Y = A m_X$$

$$V_Y = A V_X A^H$$

Valid for any distribution.



$$W_{\text{out}X} = A^H W_{\text{in}Y} A$$

$$W_{\text{out}X} m_{\text{out}X} = A^H W_{\text{in}Y} m_{\text{in}Y}$$

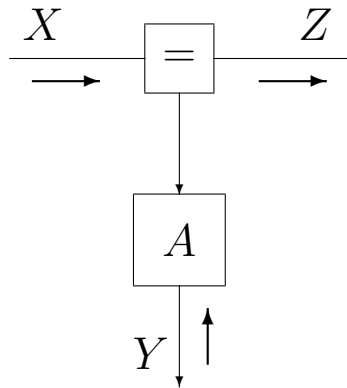
$$\tilde{W}_X m_X = A^H \tilde{W}_Y m_Y$$

$$\tilde{W}_X = A^H \tilde{W}_Y A$$

Assuming Gaussians.

(Valid for any distribution if the rank of A equals the number of rows.)

Computation Rules for Gaussian Messages III

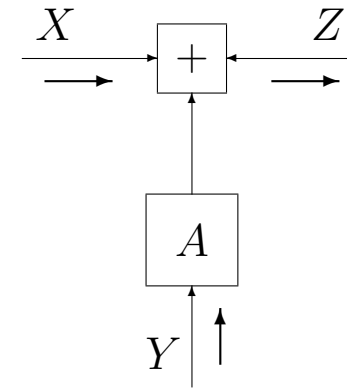


$$m_{\text{out}Z} = m_{\text{in}X} + V_{\text{in}X} A^H G (m_{\text{in}Y} - A m_{\text{in}X})$$

$$V_{\text{out}Z} = V_{\text{in}X} - V_{\text{in}X} A^H G A V_{\text{in}X}$$

with $G \triangleq (V_{\text{in}Y} + A V_{\text{in}X} A^H)^{-1}$

Assuming Gaussians.



$$m_{\text{out}Z} = -m_{\text{in}X} - A m_{\text{in}Y}$$

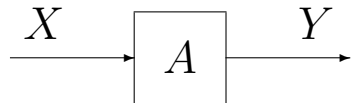
$$W_{\text{out}Z} = W_{\text{in}X} - W_{\text{in}X} A H A^H W_{\text{in}X}$$

with $H \triangleq (W_{\text{in}Y} + A^H W_{\text{in}X} A)^{-1}$

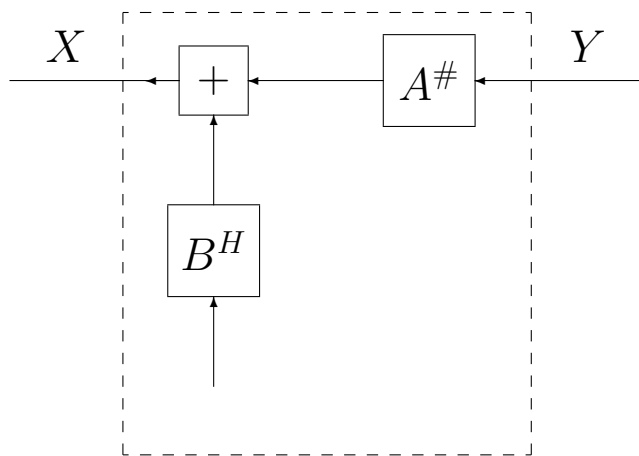
Valid for any distribution.

Reversing Matrix Multiplication

If $\text{rank } A = \text{num. rows} < \text{num. columns}$:

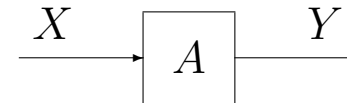


is equivalent to:

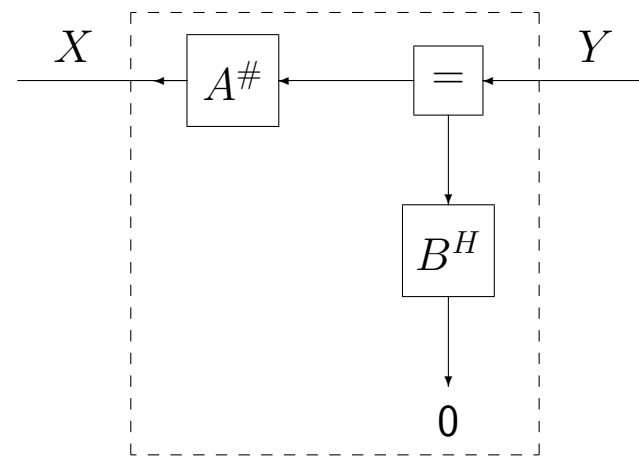


where $A^\# = A^H(AA^H)^{-1}$ and where B is a matrix such that $\begin{pmatrix} A \\ B \end{pmatrix}$ is a nonsingular square matrix and $AB^H = 0$.

If $\text{rank } A = \text{num. columns} < \text{num. rows}$:

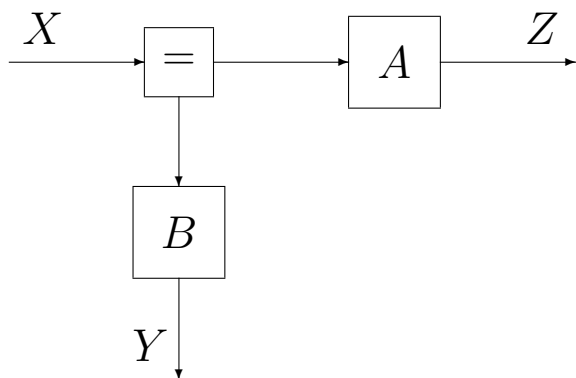


is equivalent to:



where $A^\# = (A^H A)^{-1} A^H$ and where B is a matrix such that (A, B) is a nonsingular square matrix and $B^H A = 0$.

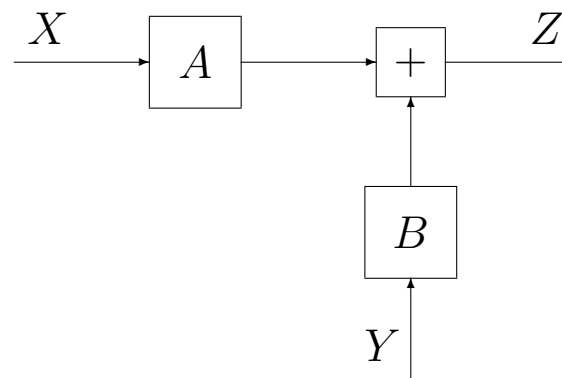
Computation Rules for Gaussian Messages IV



If $\begin{pmatrix} A \\ B \end{pmatrix}$ is a nonsingular square matrix:

$$m_{\text{out}X} = \begin{pmatrix} A \\ B \end{pmatrix}^{-1} \begin{pmatrix} m_{\text{in}Z} \\ m_{\text{in}Y} \end{pmatrix}$$

$$V_{\text{out}X} = \begin{pmatrix} A \\ B \end{pmatrix}^{-1} \begin{pmatrix} V_{\text{in}Z} & 0 \\ 0 & V_{\text{in}Y} \end{pmatrix} (A^H, B^H)^{-1}$$



If (A, B) is a nonsingular square matrix:

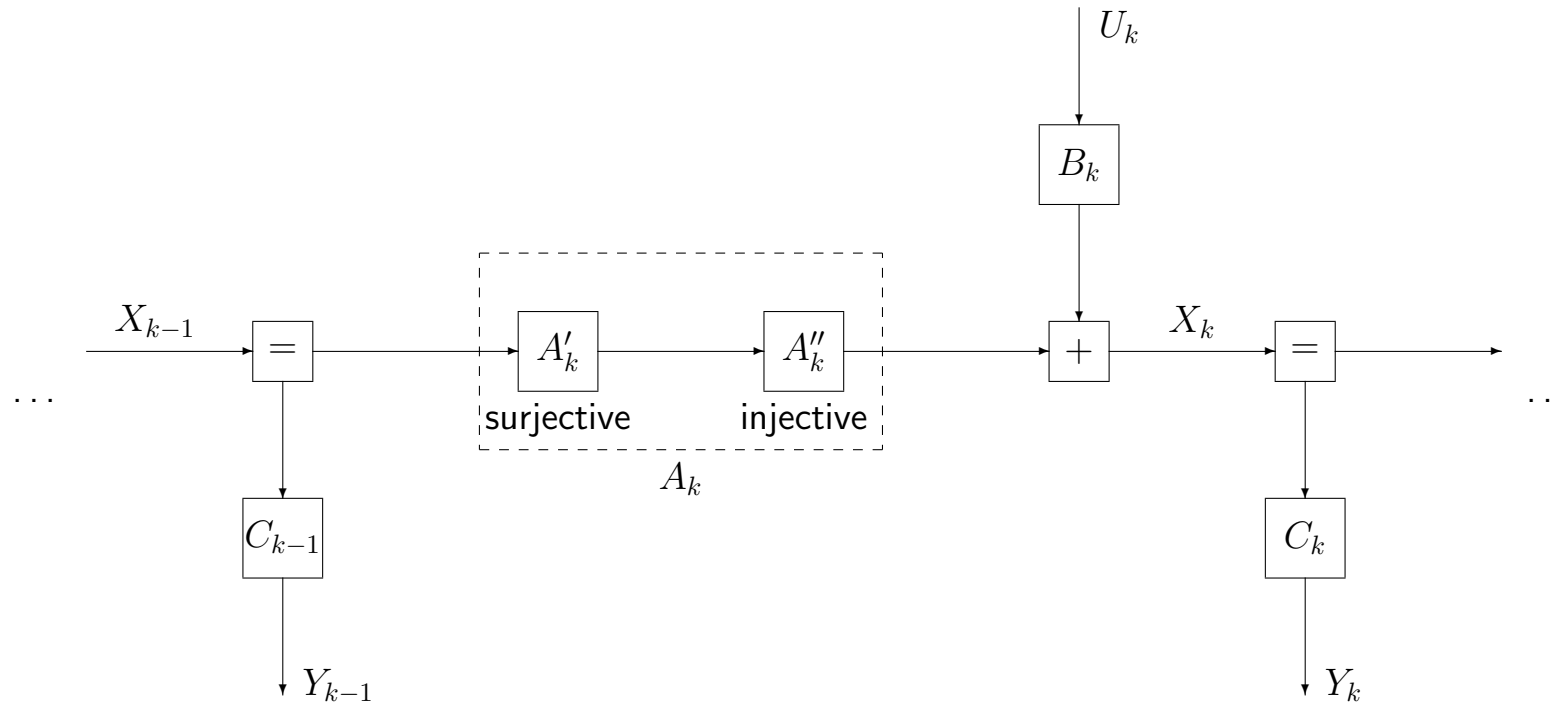
$$W_{\text{out}Z} m_{\text{out}Z} = \begin{pmatrix} A^H \\ B^H \end{pmatrix}^{-1} \begin{pmatrix} W_{\text{in}X} m_{\text{in}X} \\ W_{\text{in}Y} m_{\text{in}Y} \end{pmatrix}$$

$$W_{\text{out}Z} = \begin{pmatrix} A^H \\ B^H \end{pmatrix}^{-1} \begin{pmatrix} W_{\text{in}X} & 0 \\ 0 & W_{\text{in}Y} \end{pmatrix} (A, B)^{-1}$$

$$\begin{pmatrix} m_X \\ m_Y \end{pmatrix} = (A, B)^{-1} m_Z$$

$$\begin{pmatrix} V_X & V_{XY} \\ V_{YX} & V_Y \end{pmatrix} = (A, B)^{-1} V_Z \begin{pmatrix} A^H \\ B^H \end{pmatrix}^{-1}$$

Linear State Space Model (again)



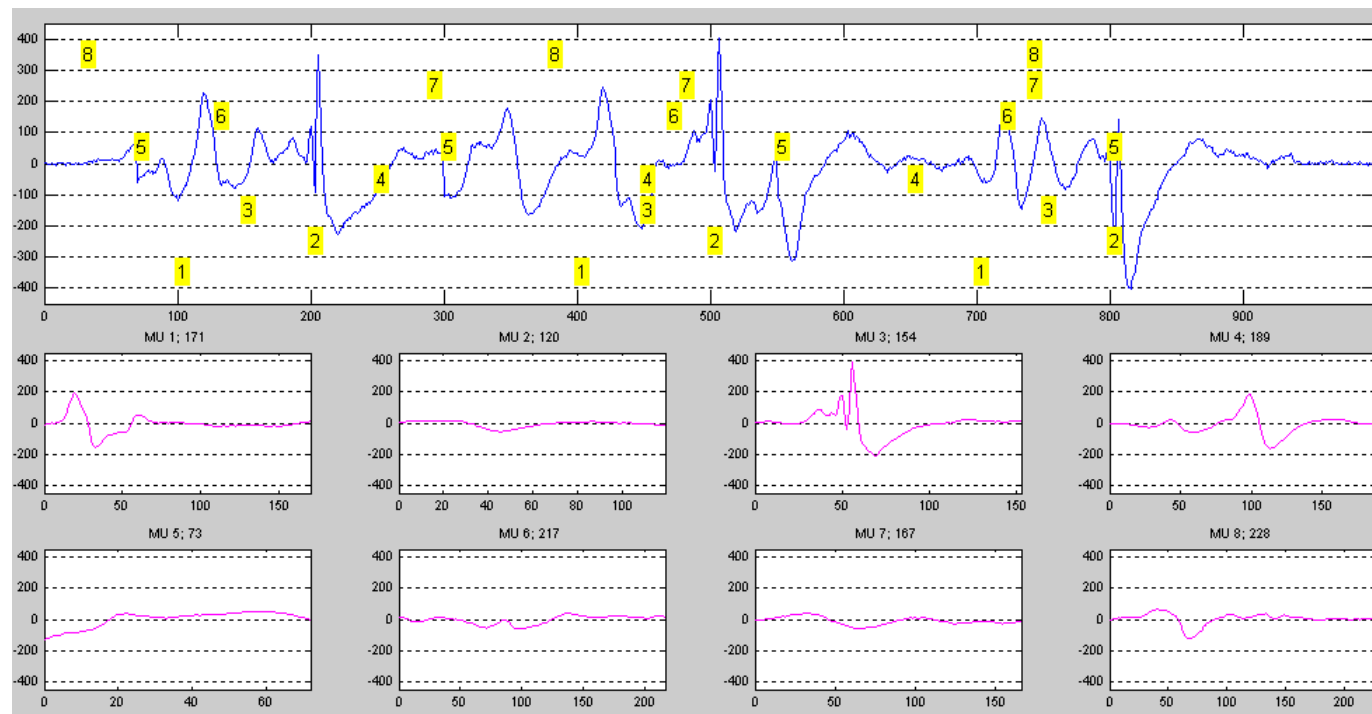
Topics

1. Basics
2. Fourier transform and duality
3. Factor graphs and error correcting codes
4. Kalman filtering and recursive least squares
5. Signal processing with factor graphs

Example 1: Electromyography (EMG)

(with V. Koch)

Muscular activity in human bodies is accompanied by electrical signals inside the muscle fibers. Such signals can be measured by electrodes (either on the skin or inside the muscle). The art of measuring and analyzing such signals is called electromyography (EMG).

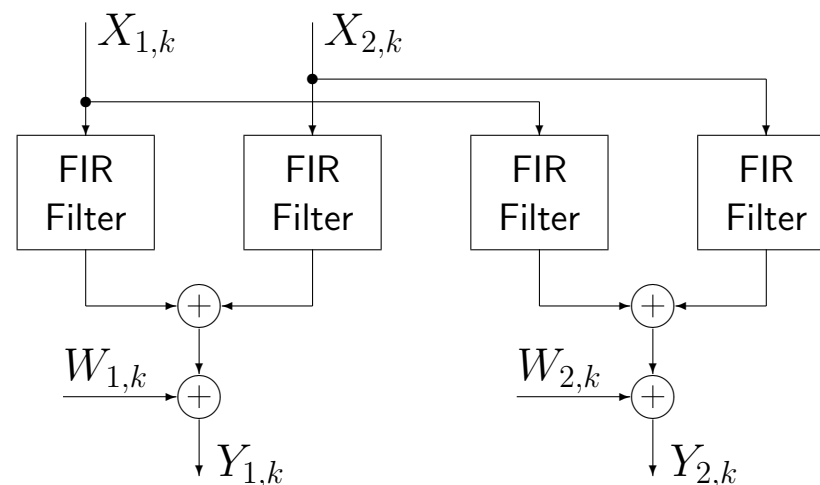


Discrete-Time Model of EMG Signals

- Many (< 50) independent sources;
Source i emits **sparse** signal $X_{i,1}, X_{i,2}, \dots$ with $X_{i,k} \in \{0, 1\}$.
- Each electrode (of $1 \dots 128$) picks up a **noisy superposition of heavily filtered sources**:

$$Y_{j,k} = \sum_i \sum_{\ell=0}^M X_{i,k-\ell} \cdot h_{i,j,\ell} + W_{j,k}$$

with **known** $h_{i,j,\ell} \in \mathbb{R}$ and white Gaussian noise $W_{j,1}, W_{j,2}, \dots$



Source Model

$S_{i,k}$ is the state of the FIR filters fed by source i , i.e., the vector $(X_{i,k}, X_{i,k-1}, \dots, X_{i,k-M})$, which we assume to contain at most one “1”.

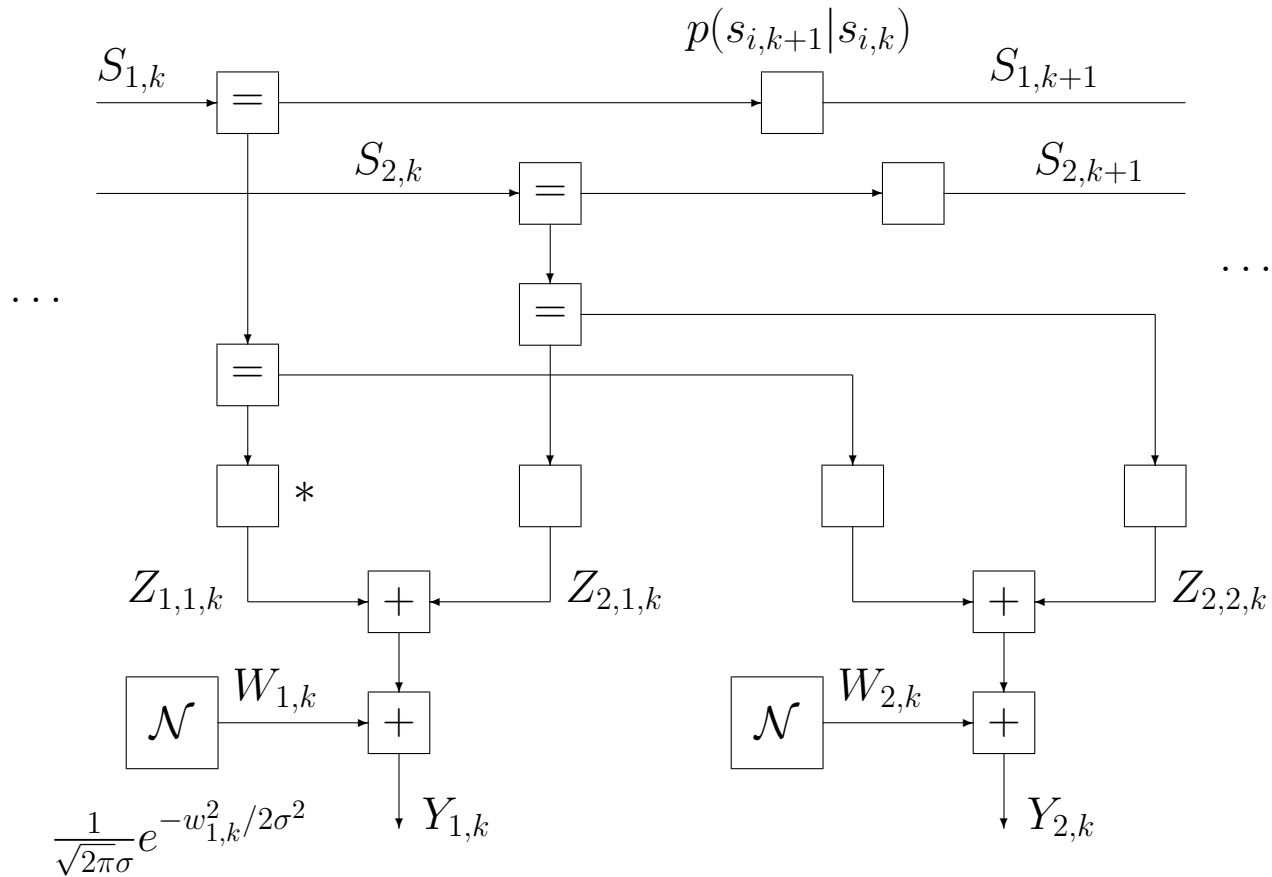
Assumed state transition probabilities $p(s_{i,k+1}|s_{i,k})$:

$s_{i,k}$	$s_{i,k+1}$	$p(s_{i,k+1} s_{i,k})$
all zeros	all zeros	$1 - \varepsilon$
all zeros	$(1, 0, \dots, 0)$	ε
“1” at position n	“1” at position $n+1$	1
$(0, \dots, 0, 1)$	all zeros	1
everything else		0

with $\varepsilon \approx 10^{-2}$.

Factor Graph of EMG Signal Model

represents a factorization of the joint pdf/pmf of all variables



$$(*) \quad Z_{i,j,k} \triangleq \sum_{\ell=0}^M X_{i,k-\ell} \cdot h_{i,j,\ell} = f(S_{i,k})$$

Example 2: Channels with Phase Noise

(Colavolpe/Barbieri/Caire; Dauwels/L.)

$$Y_k = X_k e^{i\Theta_k} + N_k$$

where N_1, N_2, \dots is white Gaussian noise with known variance σ_N^2 .

Three different models for the phase Θ_k are considered:

Constant Phase: $\Theta_k = \Theta$, an unknown constant.

Random Walk:

$$\Theta_k = \Theta_{k-1} + W_k,$$

where W_k is white Gaussian noise with known variance σ_W^2 .

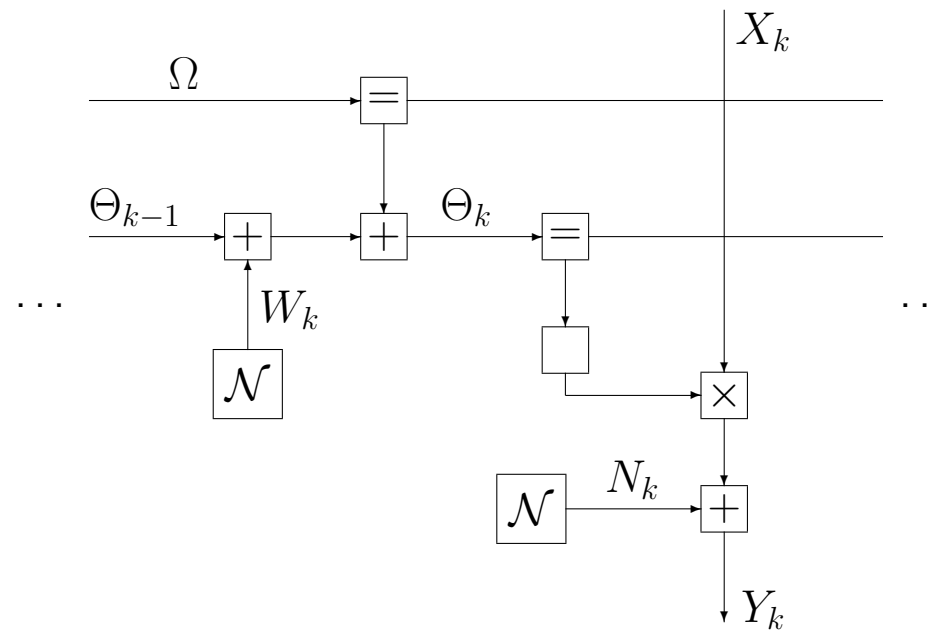
Random Walk with Unknown Drift:

$$\Theta_k = \Theta_{k-1} + W_k + \Omega$$

with an unknown drift parameter Ω and with W_k as above.

Channels with Phase Noise (cont'd)

Factor graph:



Example 3: AR Model Parameter Estimation

(with S. Korl)

X_1, X_2, \dots are real-valued random variables defined by

$$X_k = \sum_{\ell=1}^M X_{k-\ell} \cdot a_\ell + U_k$$

with $a_\ell \in \mathbb{R}$ and where U_1, U_2, \dots is white Gaussian noise with variance σ_U^2 .

We observe the process Y_1, Y_2, \dots with

$$Y_k = X_k + W_k,$$

where W_1, W_2, \dots is white Gaussian noise with variance σ_W^2 , and we wish to estimate the unknown coefficients

$$a \triangleq (a_1, \dots, a_M)^T.$$

In a first version of the problem, the parameters σ_U^2 and σ_W^2 are known; in a second version, they are unknown and need to be estimated as well.

AR Model: State Space Formulation

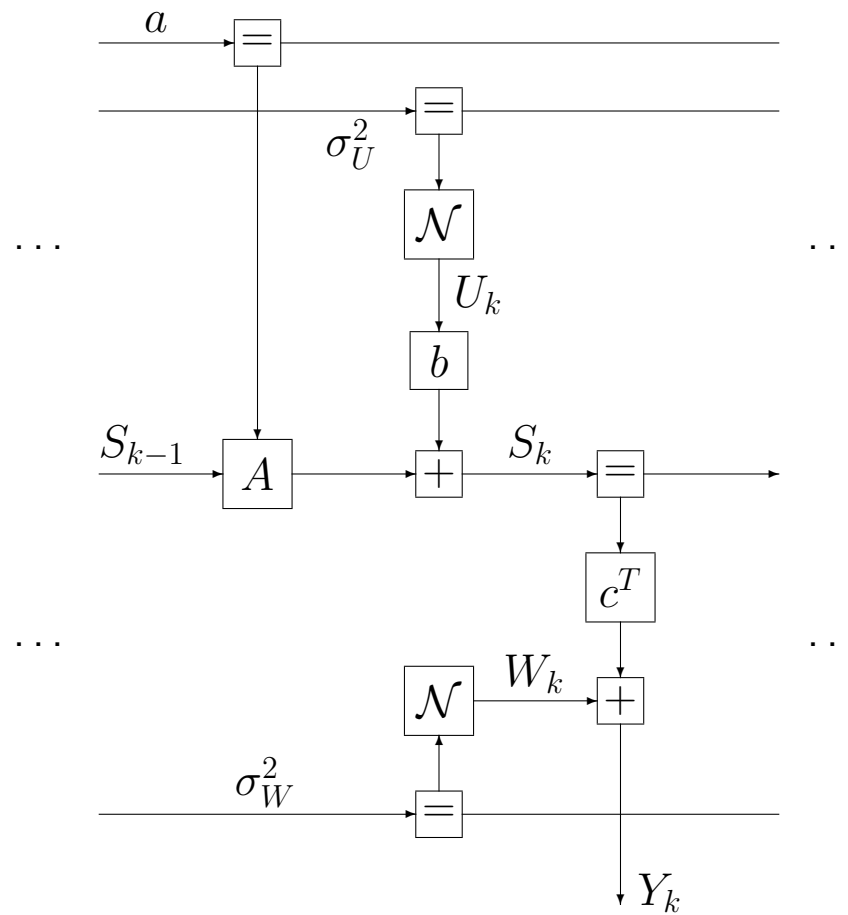
With

$$\begin{aligned} S_k &\triangleq (X_k, \dots, X_{k-M+1})^T \\ A &\triangleq \begin{pmatrix} a^T \\ I & 0 \end{pmatrix} \\ b &\triangleq c \triangleq (1, 0, \dots, 0)^T \end{aligned}$$

we can write

$$\begin{aligned} S_k &= AS_{k-1} + bU_k \\ Y_k &= c^T S_k + W_k \end{aligned}$$

AR Parameter Estimation: Factor Graph



Continuous Variables: Message Types

The following message types are widely applicable.

- **Hard-decision estimate.**
- **Quantization** of variables.
Obvious, but infeasible in higher dimensions.
- **Function value and derivative (or gradient)** at a point selected by the receiving node. This data type allows gradient algorithms (e.g., LMS).
- **Gaussian distributions.** In particular, Kalman filtering and LMMSE estimation.
- **Gaussian mixtures.**
- **List of samples.** A pdf can be represented by a list of samples. This data type is the basis of the **particle filter**, but it can be used as a generic data type in a graphical model.
- **Compound messages.** The “product” of other message types.

The Factor Graph Approach to Signal Processing

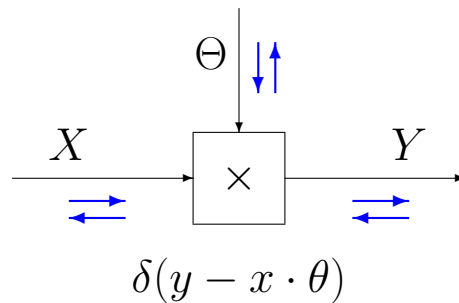
1. Choose a factor graph to represent the system model.
2. Choose the message types (e.g., hard-decisions, gradients, Gaussians, Gaussian mixtures, list of particles, ...) and the suitable update rules for continuous variables.
Use and maintain tables of message computation rules for elementary building blocks.
3. Choose a message update schedule.

Systematic approach to mix and match everything: iterative decoders, Kalman filtering/smoothing, particle methods, gradient methods, expectation maximization,

Beyond Sum/Max-Product

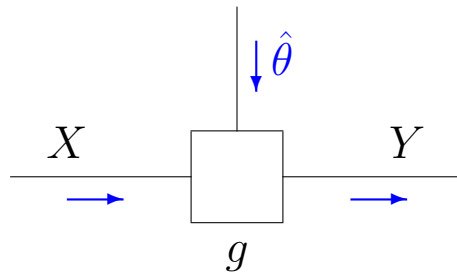
Sum-product rule may give unwieldy messages.

Example: unknown factor/amplification Θ



Destroys Gaussianity!

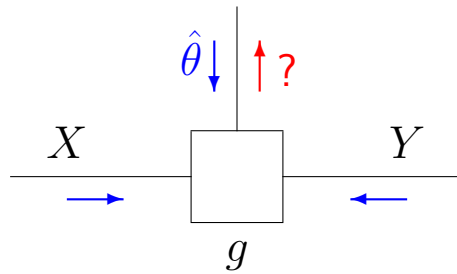
Obvious Idea: Use Tentative Decision $\hat{\theta}$



Sum-product rule simplifies to

$$\mu_{g \rightarrow Y}(y) = \int_x g(x, y, \hat{\theta}) \mu_{X \rightarrow g}(x) dx$$

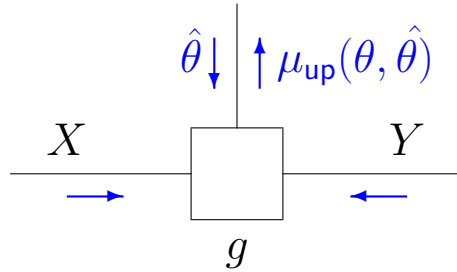
Upwards Message?



Two approaches:

- gradient methods
- expectation maximization (Dempster/Laird/Rubin 1977)

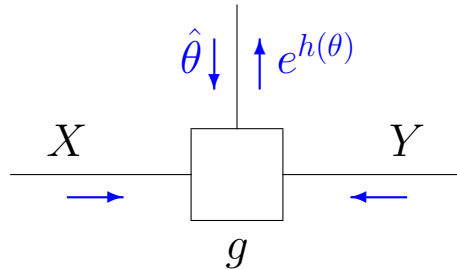
Gradient Messages



Instead of the sum-product message $\mu_{g \rightarrow \theta}(\theta)$, we use

$$\mu_{\text{up}}(\theta, \hat{\theta}) = \frac{d}{d\theta} \log \mu_{g \rightarrow \theta}(\theta) \Big|_{\theta = \hat{\theta}}$$

Local EM Message Computation Rule (ISIT 2005)



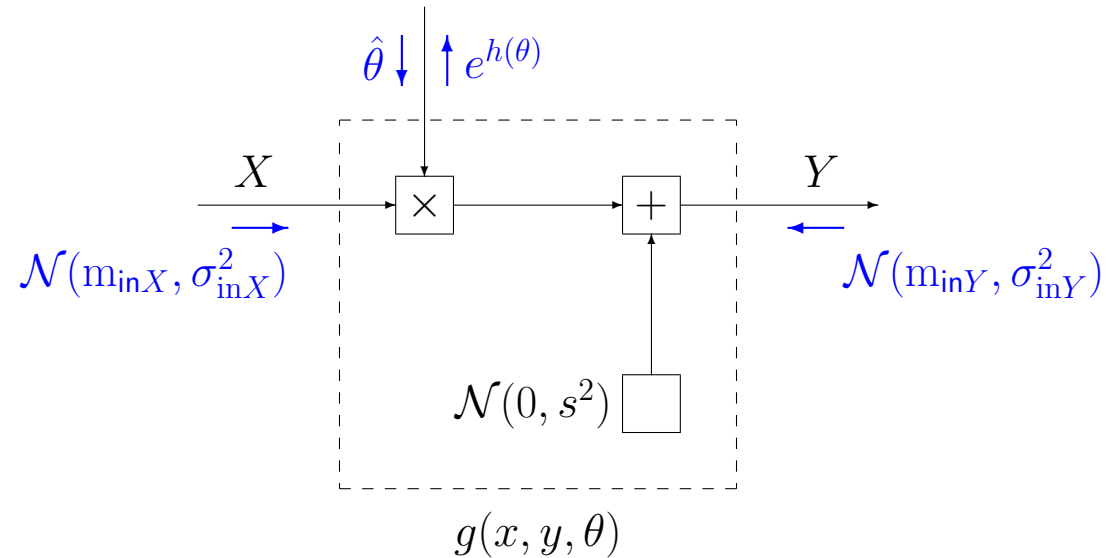
$$h(\theta) = \mathbb{E}_{p_{\text{local}}} [\log g(X, Y, \theta)]$$

with

$$p_{\text{local}}(x, y | \hat{\theta}) \propto g(x, y, \hat{\theta}) \underbrace{\mu_{X \rightarrow g}(x) \mu_{Y \rightarrow g}(y)}_{\text{sum-product messages}}$$

Looks complicated, but it sometimes yields nicer expressions than sum-product rule!

Local EM: Example



Message $e^{h(\theta)} \propto \mathcal{N}(m_{out\theta}, \sigma_{out\theta}^2)$ with

$$m_{out\theta} = \frac{\sigma_{XY} + m_X m_Y}{\sigma_X^2 + m_X^2}$$

$$\sigma_{out\theta}^2 = \frac{s^2}{\sigma_X^2 + m_X^2}$$

We have seen:

1. Basics
2. Fourier transform and duality
3. Factor graphs and error correcting codes
4. Kalman filtering and recursive least squares
5. Signal processing with factor graphs

The Factor Graph Approach to Signal Processing

1. Choose a factor graph to represent the system model.
2. Choose the message types (e.g., hard-decisions, gradients, Gaussians, Gaussian mixtures, list of particles, ...) and the suitable update rules for continuous variables.
Use and maintain tables of message computation rules for elementary building blocks.
3. Choose a message update schedule.

Systematic approach to mix and match everything: iterative decoders, Kalman filtering/smoothing, particle methods, gradient methods, expectation maximization,