TTK4250

Week 11

Exploiting structure: Rao-Blackwellization and Graphical Models

Edmund Førland Brekke

31. October 2024

Outline

Outline

Conditionally linear structures

Consider a stochastic system whose state vector can be decomposed as

$$\mathbf{x}_k = \begin{bmatrix} \mathbf{x}_k^n \\ \mathbf{x}_k^l \end{bmatrix}$$

with process and measurement models given by

$$\begin{aligned}
\mathbf{x}_k^n &= \mathbf{f}_k(\mathbf{x}_{k-1}^n) + \mathbf{v}_k^n & \mathbf{v}_k^n &\sim \mathcal{N}(\mathbf{0}, \mathbf{Q}^n) \\
\mathbf{x}_k^l &= \mathbf{A}_k(\mathbf{x}_{k-1}^n)\mathbf{x}_{k-1}^l + \mathbf{v}_k^l & \mathbf{v}_k^l &\sim \mathcal{N}(\mathbf{0}, \mathbf{Q}^l) \\
\mathbf{z}_k &= \mathbf{h}(\mathbf{x}_k^n) + \mathbf{C}_k(\mathbf{x}_k^n)\mathbf{x}_k^l + \mathbf{w}_k & \mathbf{w}_k &\sim \mathcal{N}(\mathbf{0}, \mathbf{R})
\end{aligned}$$

Conditional on $\mathbf{x}_k^n$ we see that $\mathbf{x}_k^l$ is given by entirely linear models.

Perspective 1: The Rao-Blackwell theorem

Let $\theta(\mathbf{z}_{1:k})$ be an estimator of $\mathbf{x}_k^l$. The conditional expectation of $\theta(\mathbf{z}_{1:k})$ given $\mathbf{x}_{1:k}^n$ is then a better estimator of $\mathbf{x}_k^l$.

Perspective 2: Dimensionality reduction

If we are going to estimate $\mathbf{x}_k$ by means of a particle filter, then we don't really want to sample in the space of $\mathbf{x}_k$ if we can get away with only sampling in the space of $\mathbf{x}_k^n$.

Exploiting the structure

To separate the linear part from the nonlinear part, we make use of the following factorization of the posterior:

$$p(\mathbf{x}_k^l, \mathbf{x}_{1:k}^n \mid \mathbf{z}_{1:k}) = p(\mathbf{x}_k^l \mid \mathbf{x}_{1:k}^n, \mathbf{z}_{1:k}) p(\mathbf{x}_{1:k}^n \mid \mathbf{z}_{1:k}).$$

Assume that

$$p(\mathbf{x}_{k-1}^l \mid \mathbf{x}_{1:k-1}^n, \mathbf{z}_{1:k-1}) = \mathcal{N}(\mathbf{x}_{k-1}^l \,;\, \hat{\mathbf{x}}_{k-1}^l(\mathbf{x}_{1:k-1}^n), \mathbf{P}_{k-1}^l(\mathbf{x}_{1:k-1}^n).$$

The Rao-Blackwellized particle filter: The linear part

For the linear part of the state vector, the predicted and posterior densities are

$$\begin{aligned} p(\mathbf{x}_k^l \mid \mathbf{x}_{1:k}^n, \mathbf{z}_{1:k-1}) &= \mathcal{N}(\mathbf{x}_k^l \,;\, \hat{\mathbf{x}}_{k|k-1}^l, \mathbf{P}_{k|k-1}^l) \\ p(\mathbf{x}_k^l \mid \mathbf{x}_{1:k}^n, \mathbf{z}_{1:k}) &= \mathcal{N}(\mathbf{x}_k^l \,;\, \hat{\mathbf{x}}_k^l, \mathbf{P}_k^l) \end{aligned}$$

where expectations and covariances are given by

$$[\hat{\mathbf{x}}_k^l,\ \mathbf{P}_k^l,\ \hat{\mathbf{x}}_{k|k-1}^l,\ \mathbf{P}_{k|k-1}^l, \mathbf{S}_k^l] = \mathrm{KF}(\hat{\mathbf{x}}_{k-1}^l,\ \mathbf{P}_{k-1}^l,\ \mathbf{z}_k - \mathbf{h}_k).$$

with $\mathbf{A}_k$, $\mathbf{C}_k$, $\mathbf{Q}^l$ and $\mathbf{R}$ playing the role as transition matrix, measurement matrix, process noise matrix and measurement noise matrix, respectively.

Exploiting the structure

The Rao-Blackwellized particle filter: The nonlinear part

For the nonlinear part of the state vector, the predicted and posterior densities are given according to

$$p(\mathbf{x}_k^n \mid \mathbf{x}_{1:k-1}^n, \mathbf{z}_{1:k-1}) = \mathcal{N}(\mathbf{x}_k^n \,;\, \mathbf{f}_k^n, \mathbf{Q}_k^n) \tag{1}$$

$$p(\mathbf{z}_k \mid \mathbf{x}_{1:k}^n, \mathbf{z}_{1:k-1}) = \mathcal{N}(\mathbf{z}_k \,;\, \mathbf{h}_k + \mathbf{C}_k \,;\, \hat{\mathbf{x}}_{k|k-1}^l, \mathbf{S}_k^l). \tag{2}$$

We can use these expressions to the define the sampling and weight update of a (Rao-Blackwellized) SIR filter, or any more advanced particle filter that exploits the likelihood above to improve the proposal.

Rao-Blackwellized SIR filter

- For an old particle $\mathbf{x}_{1:k-1}^{n,[p]}$, sample a new particle $\mathbf{x}_{1:k}^{n,[p]}$ using (1).
- We calculate the weight of the new particle by means of (2) according to

$$w_k^{[p]} \propto p(\mathbf{z}_k \mid \mathbf{x}_{1:k}^{n,[p]}, \mathbf{z}_{1:k-1}) w_{k-1}^{[p]}.$$

- For each particle, we use a Kalman filter with measurement $\mathbf{z}_k - \mathbf{h}_k^{[p]}$ to calculate the posterior of the linear state $\mathbf{x}_k^l$.

Example of Rao-Blackwellization

Example: The CT model

The coordinated turn model has a structure that lends itself to Rao-Blackwellization. Recall that its state vector was $\mathbf{x} = [x, y, v_x, v_y, \omega]^\mathsf{T}$. Let

$$\mathbf{x}^n = \omega \quad \text{and} \quad \mathbf{x}^l = [x, y, v_x, v_y]^\mathsf{T}.$$

The CT model from Example 5.1 in the book fits the conditionally linear model if

$$\mathbf{A}_k = \begin{bmatrix} 1 & 0 & (\sin T\omega_{k-1})/\omega_{k-1} & (-1 + \cos T\omega_{k-1})/\omega_{k-1} \\ 0 & 1 & (1 - \cos T\omega_{k-1})/\omega_{k-1} & (\sin T\omega_{k-1})/\omega_{k-1} \\ 0 & 0 & \cos T\omega_{k-1} & -\sin T\omega_{k-1} \\ 0 & 0 & \sin T\omega_{k-1} & \cos T\omega_{k-1} \end{bmatrix}$$

$$\begin{aligned} \mathbf{f}_k(\omega_{k-1}) &= \omega_{k-1} \\ \mathbf{h}_k(\omega_{k-1}) &= 0 \\ \mathbf{C}_k(\omega^n) &= \begin{bmatrix} \mathbf{I}_2 & \mathbf{0}_{2\times 2} \end{bmatrix}. \end{aligned}$$

We see that we can make a particle filter for the CT model by only sampling a one-dimensional state (the turnrate). Each particle then represents a unique turnrate history, and everything else is Gaussian-linear conditional on that history.

Outline

FastSLAM

Can particle filters be used to solve SLAM?

Motivation for investigating particle filters in SLAM

- Combine the solutions to the **online SLAM problem**, $p(\mathbf{x}_k, \mathbf{m} \mid \mathbf{z}_{1:k})$, and **the full SLAM problem**, $p(\mathbf{x}_{1:k}, \mathbf{m} \mid \mathbf{z}_{1:k})$, in one method.
- Exploit structure inherent in the SLAM problem to avoid having to deal with large covariance matrices.

We have to reduce the dimension of the state vector if particle filters are going to have any chance at SLAM.

Rao-Blackwellization for SLAM

Recall that the full state vector is $\boldsymbol{\eta} = [\mathbf{x}^\mathsf{T}, \mathbf{m}^\mathsf{T}]^\mathsf{T}$. Partition it into linear and nonlinear states:

$$\mathbf{x}^n = \mathbf{x} \quad \text{and} \quad \mathbf{x}^l = \mathbf{m}$$

Will this give us a conditionally linear model?

Matching SLAM with the conditionally linear model

The process model

From the standard planar SLAM model and the chosen partitioning of $\boldsymbol{\eta}$ we get

$$\mathbf{A}_k = \mathbf{I}$$
$$\mathbf{Q}^l = \mathbf{0}$$
$$\mathbf{f}_k(\mathbf{x}_{k-1}) = \mathbf{x}_{k-1} \oplus \mathbf{u}_k.$$

The measurement model

- The measurement model is nonlinear and does not conform to the conditionally linear model:

$$\mathbf{h}(\mathbf{x}_k, \mathbf{m}^i) = \mathrm{c2p}\left(\mathrm{R}(-\psi)(\mathbf{m}^i - \boldsymbol{\rho}_k)\right)$$

- However, if we linearize it we can still obtain analytical formulas for $p(\mathbf{z}_k^i \mid \mathbf{m}, \mathbf{x}_{1:k})$ and its marginalization over $\mathbf{m}$,
- ... and we can implement an EKF to estimate $\mathbf{m}^i$ conditional on $\mathbf{x}_{1:k}$.

FActored Solution To SLAM

Not only can we treat the landmarks as Gaussian-linear conditional on $\mathbf{x}_{1:k}$, but we can also treat them as conditionally independent.

Main result

The joint posterior of pose-trajectories and landmarks can for all time steps be factorized as

$$p(\mathbf{x}_{1:k}, \mathbf{m} \mid \mathbf{z}_{1:k}, \mathbf{u}_{1:k}) = p(\mathbf{x}_{1:k} \mid \mathbf{z}_{1:k}, \mathbf{u}_{1:k}) \prod_{i=1}^{n} p(\mathbf{m}^i \mid \mathbf{x}_{1:k}, \mathbf{z}_{1:k}, \mathbf{u}_{1:k}).$$

Sketch of proof

- It makes sense to assume independence between all landmarks and the pose to begin with.
- If we have independence conditional on $\mathbf{x}_{1:k-1}$, then simply appending $\mathbf{x}_k$ has no implications on the landmarks.
- The likelihood is a product of individual factors for each landmark, each conditionally dependent on $\mathbf{x}_k$.

FastSLAM: Structure of the particle filter

Each particle p contains:

- A pose trajectory $\mathbf{x}_{1:k-1}^{[p]}$.
- A particle weight $w_{k-1}^{[p]}$.
- For each landmark i the particle carries a Gaussian distribution

$$p(\mathbf{m}^i \mid \mathbf{x}_{1:k-1}^{[p]}, \mathbf{z}_{1:k-1}) = \mathcal{N}(\mathbf{m}^i ; \hat{\mathbf{m}}_{k-1}^{i,[p]}, \mathbf{P}_{k-1}^{i,[p]})$$

  whose mean and covariance contain all that is known about the landmark conditional on the trajectory $\mathbf{x}_{1:k-1}^{[p]}$.

During the estimation cycle, the particles are subject to manipulations as part of

- Landmark prediction,
- Landmark update,
- New pose proposal,
- Particle weight update.

FastSLAM: Landmark prediction and update

The landmark prediction

Nothing happens to the landmarks during the prediction step:

$$p(\mathbf{m}^i \mid \mathbf{x}_{1:k}^{[\mathrm{p}]}, \mathbf{z}_{1:k-1}) = p(\mathbf{m}^i \mid \mathbf{x}_{1:k-1}^{[\mathrm{p}]}, \mathbf{z}_{1:k-1}) = \mathcal{N}\left(\mathbf{m}^i\, ;\, \hat{\mathbf{m}}_{k-1}^{i,[\mathrm{p}]}, \mathbf{P}_{k-1}^{i,[\mathrm{p}]}\right)$$

The landmark update

To use the Rao-Blackwell machinery we linearize the measurement model:

$$\mathbf{z}_k^i \approx \mathbf{h}(\mathbf{x}_k^{[\mathrm{p}]}, \hat{\mathbf{m}}_{k-1}^{i,[\mathrm{p}]}) + \mathbf{H}_{\mathbf{m}}^{i,[\mathrm{p}]} \Delta\mathbf{m}^i + \mathbf{w}_k^i.$$

Here $\Delta\mathbf{m}^i = \mathbf{m}^i - \hat{\mathbf{m}}_{k-1}^{i,[\mathrm{p}]}$ and the Jacobian $\mathbf{H}_{\mathbf{m}}^{i,[\mathrm{p}]}$ is of the same form as in EKF-SLAM.

Thanks to the linearization we can also express the posterior landmark pdf as a Gaussian

$$p(\mathbf{m}^i \mid \mathbf{x}_{1:k}^{[\mathrm{p}]}, \mathbf{z}_{1:k}) = \mathcal{N}\left(\mathbf{m}^i\, ;\, \hat{\mathbf{m}}_{k}^{i,[\mathrm{p}]}, \mathbf{P}_{k}^{i,[\mathrm{p}]}\right)$$

where $\hat{\mathbf{m}}_{k}^{i,[\mathrm{p}]}$ and $\mathbf{P}_{k}^{i,[\mathrm{p}]}$ are output from an EKF (details in the book).

FastSLAM: Pose proposal and weight update

The pose proposal

In FastSLAM 1.0 the new pose $\mathbf{x}_k^{[p]}$ of particle p is proposed from the process model, conditional on $\mathbf{x}_{1:k-1}^{[p]}$.

The weight update

It can be shown that $p(\mathbf{x}_{1:k} \mid \mathbf{z}_{1:k}) \propto p(\mathbf{z}_k \mid \mathbf{x}_{1:k}, \mathbf{z}_{1:k-1}) p(\mathbf{x}_{1:k} \mid \mathbf{z}_{1:k-1})$ where

$$\begin{aligned} p(\mathbf{z}_k \mid \mathbf{x}_{1:k}, \mathbf{z}_{1:k-1}) &= \prod_{i=1}^{m} \int p(\mathbf{z}_k^i \mid \mathbf{m}^i, \mathbf{x}_k) p(\mathbf{m}^i \mid \mathbf{x}_{1:k-1}, \mathbf{z}_{1:k-1}) \mathrm{d}\mathbf{m}^i \\ &\approx \prod_{i=1}^{n} \mathcal{N}(\mathbf{z}_k^i ; \mathbf{h}(\mathbf{x}_k^{[\mathrm{p}]}, \hat{\mathbf{m}}_{k-1}^{i,[\mathrm{p}]}), \mathbf{S}_k^{i,[\mathrm{p}]}) \end{aligned}$$

where $\mathbf{S}_k^{i,[\mathrm{p}]}$ is given by the obvious EKF formula. Consequently, we update the particle weights according to

$$w_k^{[\mathrm{p}]} \propto p(\mathbf{z}_k \mid \mathbf{x}_k^{[\mathrm{p}]}, \mathbf{z}_{1:k-1}) w_{k-1}^{[\mathrm{p}]}.$$

This concludes our development of a basic FastSLAM filter.

FastSLAM with unknown data association

Per-particle data association

- The particles can be used to explore the space of possible association hypotheses.
- We include the association hypothesis a_k as part of the particles. **The likelihood of particle p will be large if a_k fits the data well and small if it does not.**
- ☹ There will inevitably be a need to use different numbers of measurement-to-landmark assignments for different particles. **Are the likelihood products of different particles then comparable?**

Deterministic approach to data association

Greedy mutual exclusion: Arrange the observations sequentially, and then pair each observation with the landmark most likely to have generated it, among landmarks which not yet have been matched.

Probabilistic approach to data association

For each observation, sample landmark associations with probabilities proportional to likelihoods (again utilizing some form of Greedy mutual exclusion).

Outline

Probabilistic graphical models: Motivation

Automating inference

Enable rational inference and decision making under uncertainty.

- Medical diagnostics.
- Decoding in communication systems.

Utilizing structure

Complexity inference problems often have useful sparsity structures.

- Markov assumption in standard Bayesian filtering.
- SLAM in terms of information matrix and not covariance matrix.

Generalization / Approximation

KF is a special case of a methodology known as belief propagation.

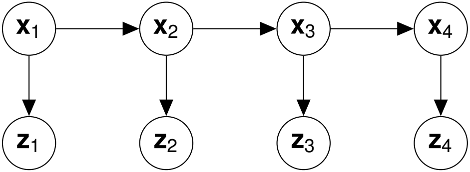
- For tree-structured graphs belief propagation is exact.
- If the graph contains cycles we may still use belief propagation as a variational inference technique.

Transparency in machine learning

- Graphical models as alternatives to neural networks when useful structures and prior knowledge are available.
- Graphical models as pre- and post-processing of neural network techniques.

Bayes nets - the dynamical system example

Consider the linear dynamical system



- Every horizontal edge in the graph represents a pdf of the form

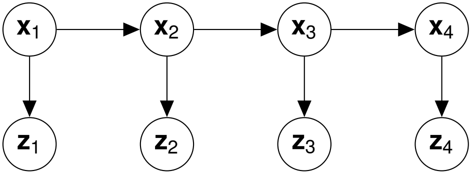
$$f_{\mathbf{x}}(\mathbf{x}_k|\mathbf{x}_{k-1}) = \mathcal{N}(\mathbf{x}_k\,;\,\mathbf{F}\mathbf{x}_{k-1}, \mathbf{Q})$$

- Every vertical edge in the graph represents a pdf of the form

$$f_{\mathbf{z}}(\mathbf{z}_k|\mathbf{x}_k) = \mathcal{N}(\mathbf{z}_k\,;\,\mathbf{F}\mathbf{x}_k, \mathbf{R})$$

Bayes nets

Consider the linear dynamical system

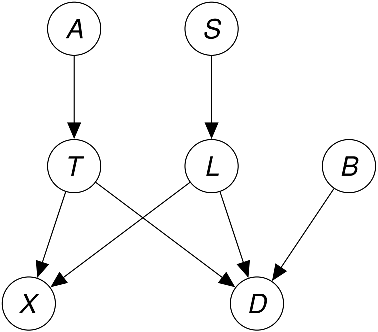


The Markov properties of the system are encoded by the graph: Any two nodes are independent if the path between them is blocked by the conditioning set of nodes.[1]

- Conditionally on $\mathbf{x}_2$ and $\mathbf{x}_3$, the nodes $\mathbf{z}_2$ and $\mathbf{z}_3$ are independent.
- Conditionally on $\mathbf{x}_1$ the nodes $\mathbf{z}_2$ and $\mathbf{z}_3$ are not independent.

[1] We shall on a subsequent slide study the concept of “blocking” and its relationship with independence.

Bayes nets - the Chest Clinic example

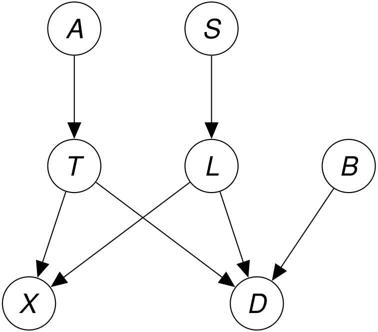


- A patient at the chest clinic returns from Asia with difficulties breathing (D).
- The doctor also asks whether the patient is smoking (S).
- A chest X-ray (X) is taken.
- To evaluate the probabilities of different diagnoses (tuberculosis T, lung cancer L or bronchitis B) we establish the causal relationships in a graph.

A Bayes net provides a factorization of the pdf of the form

$$p(x_1, \ldots, x_N) = \prod_{i=1}^{N} p(x_i \mid \mathrm{Par}(x_i))$$

Independence in Bayesian networks



T and *L* block the path between *X* and *D*.

L blocks the path between *S* and *D*.

X does not block the path between *T* and *L*.

X and *D* do not block the path between *T* and *L*.

Two *x* and *y* nodes are independent conditionally on the set *Z*, if all paths between *x* and *y* are blocked according to any of the following criteria:

1. The path is unidirectional with a node from *Z* on the path.
2. The path involves node from *Z* as a common ancestor.

Factor graphs and marginalization

Factor graphs

A factor graph expresses the factorization properties of a joint pdf:

$$p(\mathbf{x}) = \frac{1}{Z} \prod_{s \in S} \psi_s(\mathbf{x}_s)$$

Marginalization - general formula for tree-structured factor graphs

$$\begin{aligned} p(\mathbf{x}_i) &\propto \sum_{\sim \mathbf{x}_i} \prod_{s \in S} \psi_s(\mathbf{x}_s) \\ &= \prod_{s \in \mathrm{ne}(i)} \sum_{\mathbf{x}_{T(is)}} \Psi_s(\mathbf{x}_i, \mathbf{x}_{T(is)}) \\ &\propto \prod_{s \in \mathrm{ne}(i)} \sum_{\mathbf{x}_{\mathrm{ne}(s) \setminus i}} \psi_s(\mathbf{x}_i, \mathbf{x}_{\mathrm{ne}(s) \setminus i}) \prod_{m \in \mathrm{ne}(s) \setminus i} \prod_{t \in \mathrm{ne}(m) \setminus s} \sum_{\mathbf{x}_{T(mt)}} \Psi_t(\mathbf{x}_m, \mathbf{x}_{T(mt)}) \end{aligned}$$

- $T(is)$ = variable nodes in subtree below node i through factor s.
- $\Psi_s(\mathbf{x}_i, \mathbf{x}_{T(is)})$ is the product of all factors in that sub-tree.

Sums are replaced by integrals if x is continuous-valued.

Outline

Message passing a.k.a. belief propagation

The marginalization formula has a recursive structure that can be decomposed in messages passed between factor nodes and variable node.

Factor-to-variable messages

$$\mu_{\psi_s \to \mathbf{x}_i}(\mathbf{x}_i) = \sum_{x_{\mathrm{ne}(s)\setminus i}} \psi_s(\mathbf{x}_i, \mathbf{x}_{\mathrm{ne}(s)\setminus i}) \prod_{m \in \mathrm{ne}(s)\setminus i} \mu_{\mathbf{x}_m \to \psi_s}(\mathbf{x}_m)$$

Variable-to-factor messages

$$\mu_{\mathbf{x}_m \to \psi_s}(\mathbf{x}_m) = \prod_{t \in \mathrm{ne}(m)\setminus s} \mu_{\psi_t \to \mathbf{x}_m}(\mathbf{x}_m)$$

The marginalization can then be written

$$p(\mathbf{x}_i) \propto \prod_{s \in \mathrm{ne}(i)} \mu_{\psi_s \to \mathbf{x}_i}(\mathbf{x}_i)$$

The sum-product algorithm

Typically we want to find the marginals for all the variable nodes simultaneously.

No particular node is root node. Just pass messages between all nodes.

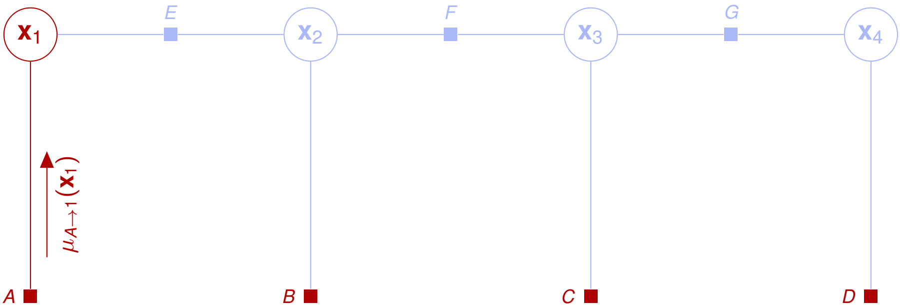
The algorithm

1. Initiate message passing at the leaves.
2. Node v remains idle until messages have arrived on all but one of the edges incident on v.
3. Once these messages have arrived, node v sends a message to its one remaining neighbor w.
4. Node v returns to idle state until it receives a message back from w.
5. Then node v sends messsages to all its other neighbors.
6. The algorithm terminates when two messages have been passed over every edge, one in each direction.

The product of incoming messages to any variable node is then proportional to its marginal density, if the factor graph has no cycles.

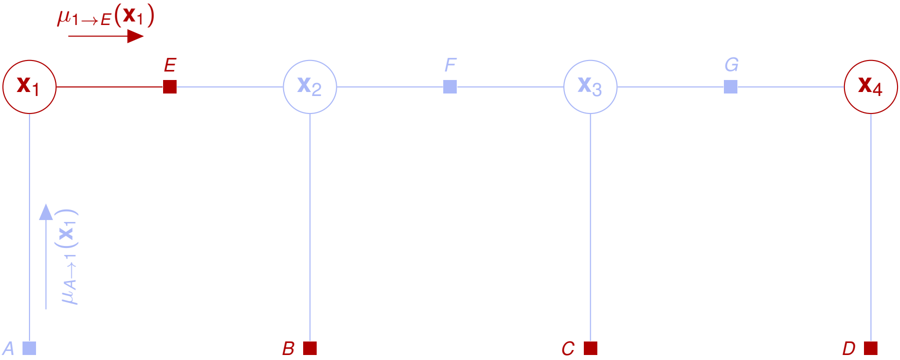
Messages in the dynamical system

We have deliberately ordered the messages in a slightly un-natural ordering.



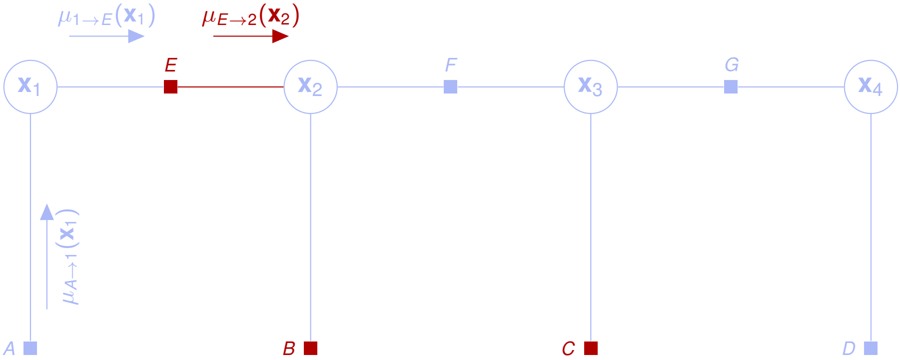
Messages in the dynamical system

We have deliberately ordered the messages in a slightly un-natural ordering.



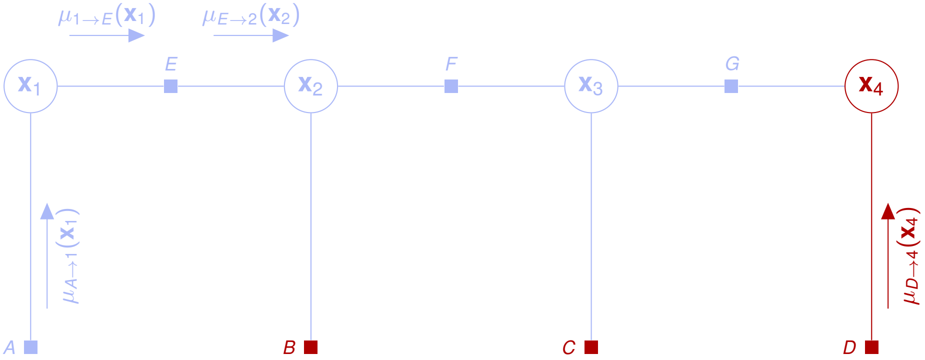
Messages in the dynamical system

We have deliberately ordered the messages in a slightly un-natural ordering.



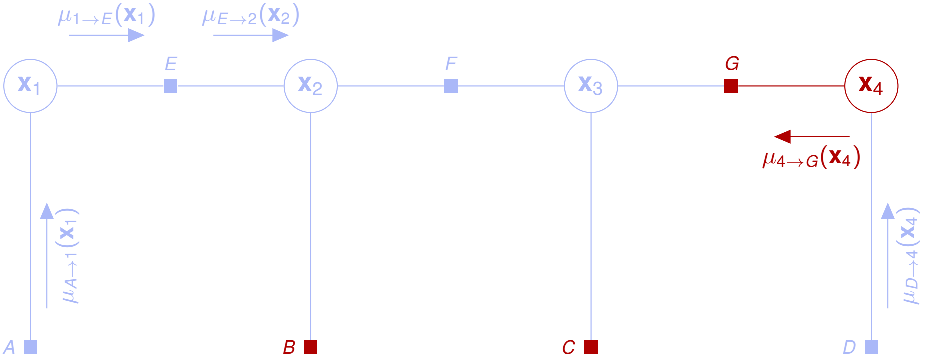
Messages in the dynamical system

We have deliberately ordered the messages in a slightly un-natural ordering.



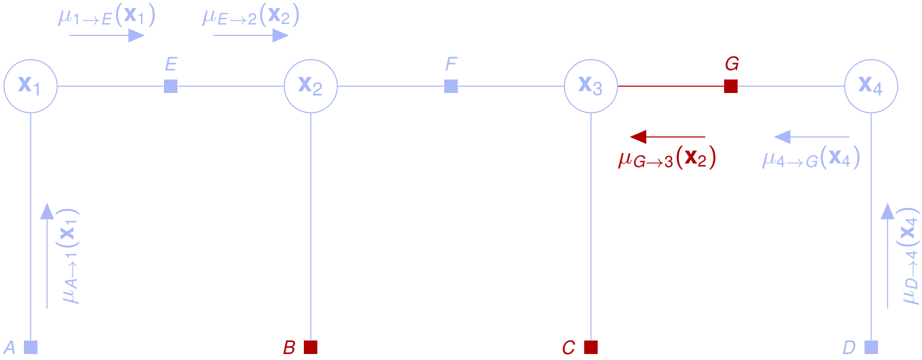
Messages in the dynamical system

We have deliberately ordered the messages in a slightly un-natural ordering.



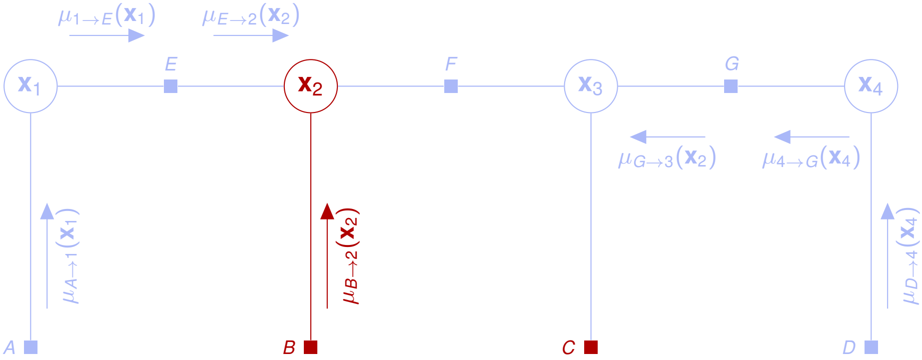
Messages in the dynamical system

We have deliberately ordered the messages in a slightly un-natural ordering.



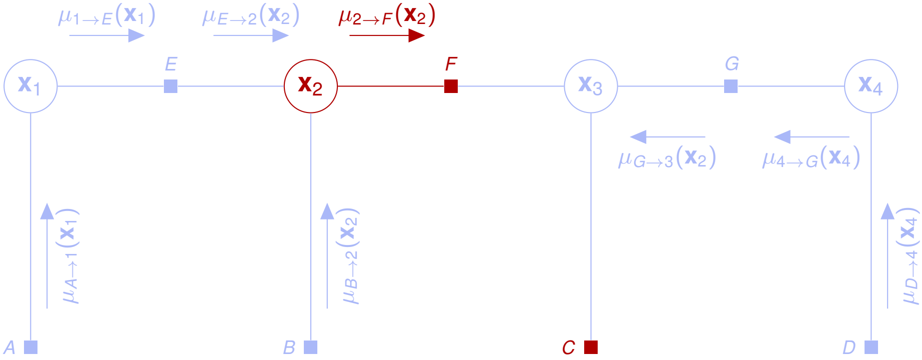
Messages in the dynamical system

We have deliberately ordered the messages in a slightly un-natural ordering.



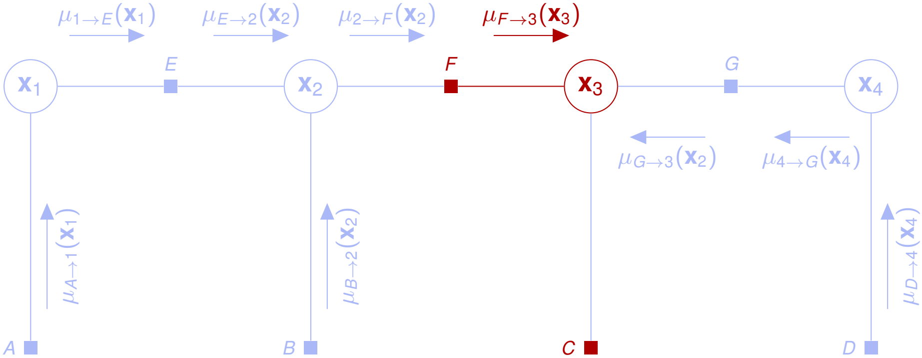
Messages in the dynamical system

We have deliberately ordered the messages in a slightly un-natural ordering.



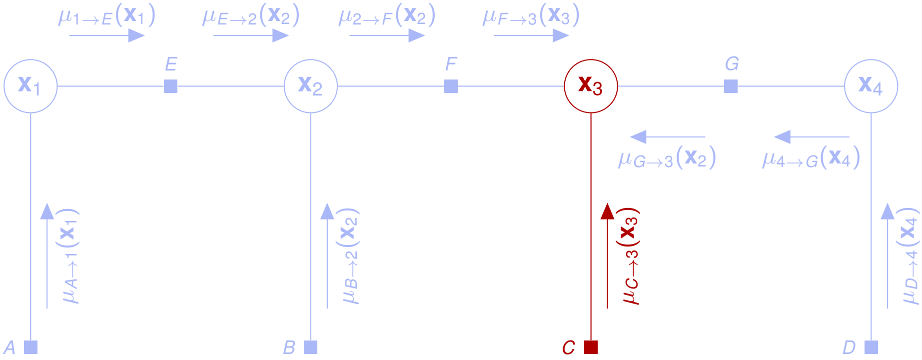
Messages in the dynamical system

We have deliberately ordered the messages in a slightly un-natural ordering.



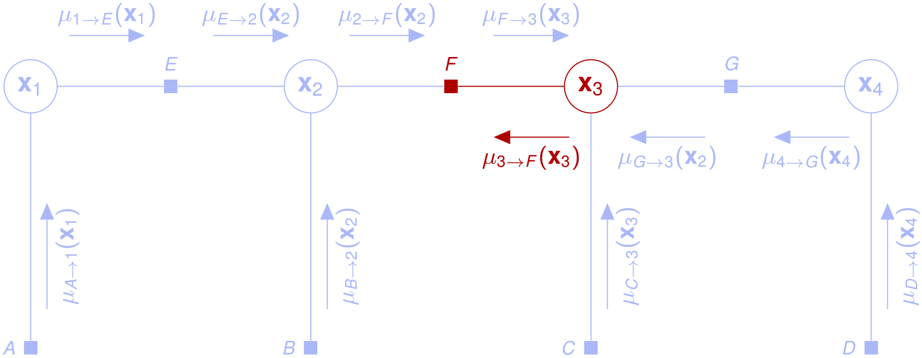
Messages in the dynamical system

We have deliberately ordered the messages in a slightly un-natural ordering.



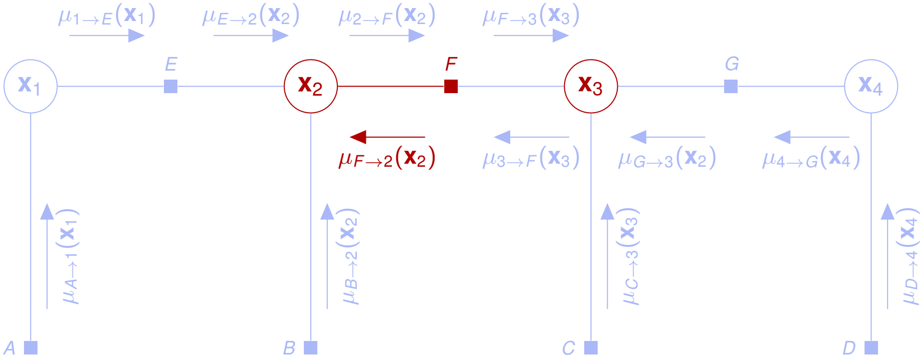
Messages in the dynamical system

We have deliberately ordered the messages in a slightly un-natural ordering.



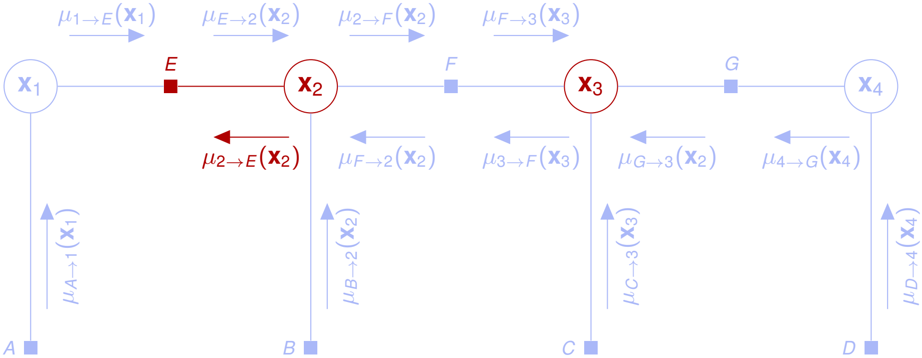
Messages in the dynamical system

We have deliberately ordered the messages in a slightly un-natural ordering.



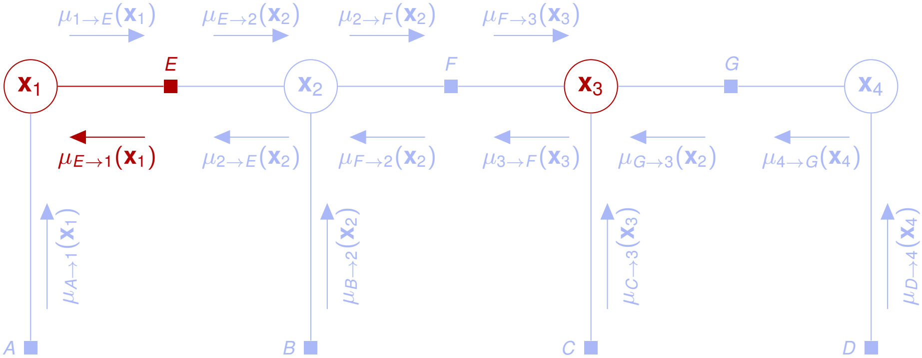
Messages in the dynamical system

We have deliberately ordered the messages in a slightly un-natural ordering.



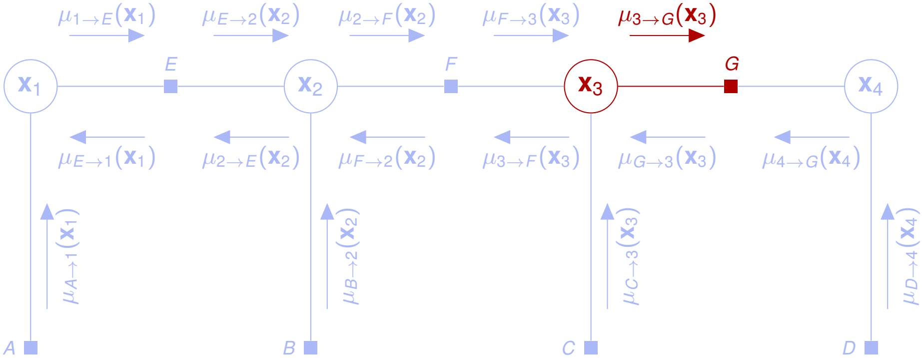
Messages in the dynamical system

We have deliberately ordered the messages in a slightly un-natural ordering.



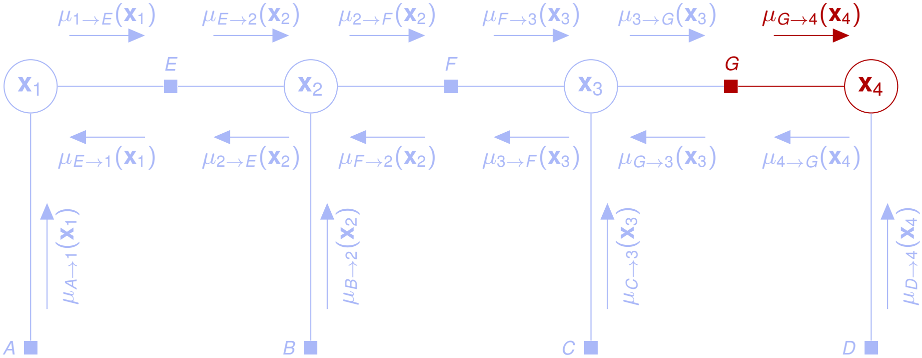
Messages in the dynamical system

We have deliberately ordered the messages in a slightly un-natural ordering.



Messages in the dynamical system

We have deliberately ordered the messages in a slightly un-natural ordering.



Forward-backward algorithm for a hidden Markov model (HMM)

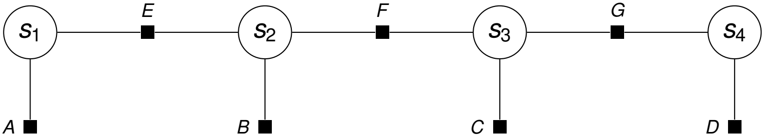
The standard sum-product algorithm is most popular for discrete-valued systems.

A general hidden Markov model

We have a Markov model, a measurement mode and an initial prior:

$$f_{\mathbf{x}}(s_k|s_{k-1}) = a^{s_{k-1}s_k} \qquad f_{\mathbf{z}}(z_k|s_k) = b^{s_k z_k} \qquad p(s_1) = \pi^{s_1}$$

The matrices $\mathbf{A} = [a^{ij}]$, $\mathbf{B} = [b^{ij}]$ and $\boldsymbol{\pi} = [\pi^i]$ specity the HMM.



Forward-backward for HMM

The sum-product algorithm for HMMs is also known as the forward-backward algorithm.

Let us look at some of the messages:

$$
\begin{aligned}
\mu_{E\to 2}(s_2) &= \sum_{s_1} a^{s_1 s_2} \underbrace{\pi^{s_1} b^{s_1 z_1}}_{\mu_{A\to 1}(s_1)} && = p(s_2, z_1) \\
&\vdots \\
\mu_{B\to 2}(s_2) &= b^{s_2 z_2} && = p(z_2|s_2) \\
&\vdots \\
\mu_{F\to 2}(s_2) &= \sum_{s_3} a^{s_2 s_3} \underbrace{b^{s_3 z_3} \underbrace{\sum_{s_4} a^{s_3 s_4} b^{s_4 z_4}}_{\mu_{G\to 3}(s_3)}}_{\mu_{3\to F}(s_3)} && = p(z_3, z_4|s_2). \\
&\vdots
\end{aligned}
$$

Forward-backward for HMM continued

- We can obtain all marginal probabilities as proportional to products of incoming messages.

The marginal probability $p(s_2|z_{1:4})$

$$
\begin{aligned}
p(s_2|z_{1:4}) &\propto \mu_{E\to 2}(s_2)\mu_{B\to 2}(s_2)\mu_{F\to 2}(s_2) \\
&= p(s_2, z_1)\, p(z_2|s_2)\, p(z_3, z_4|s_2) \\
&\propto p(s_2)p(z_{1:4} \mid s_2) \\
&\propto p(s_2, z_{1:4}) \\
&\propto p(s_2 \mid z_{1:4})
\end{aligned}
$$

- We can also evaluate the “two-slice probabilities”, which for $s_{2:3}$ are given by

$$
p(s_2, s_3) = \left(\prod_{s\in\{B,E\}} \mu_{s\to 2}(s_2)\right)\psi_F(s_2, s_3)\left(\prod_{s\in\{C,D\}} \mu_{s\to 3}(s_3)\right)
$$

Belief propagation for the linear dynamical system: The information filter

Consider once again the standard Gaussian-linear dynamical system. What comes out of the forward messages?

- Clearly it must be something equivalent to the Kalman filter.
- The multiplication of messages is more naturally expressed as addition of information states and information matrices.
- This leads a version of the Kalman filter known as the **information filter**.

$k-1$	Update	$\boldsymbol{\eta}_{k-1}$		$\boldsymbol{\Lambda}_{k-1}$	
		$\downarrow$		$\downarrow$	
k	Prediction	$\boldsymbol{\eta}_{k\|k-1}$	$= \boldsymbol{\Lambda}_{k\|k-1}\mathbf{F}\boldsymbol{\Lambda}_{k-1}^{-1}\boldsymbol{\eta}_{k-1}$	$\boldsymbol{\Lambda}_{k\|k-1}$	$= (\mathbf{Q} + \mathbf{F}\boldsymbol{\Lambda}_{k-1}^{-1}\mathbf{F}^{\mathsf{T}})^{-1}$
		$\downarrow$		$\downarrow$	
k	Update	$\boldsymbol{\eta}_{k}$	$= \boldsymbol{\eta}_{k\|k-1} + \mathbf{H}^{\mathsf{T}}\mathbf{R}^{-1}\mathbf{z}_k$	$\boldsymbol{\Lambda}_{k}$	$= \boldsymbol{\Lambda}_{k\|k-1} + \mathbf{H}^{\mathsf{T}}\mathbf{R}^{-1}\mathbf{H}$
		$\downarrow$		$\downarrow$	

The information filter can be worth considering if the measurement update in moment form is expensive, and the inversions involved in $(\mathbf{Q} + \mathbf{F}\boldsymbol{\Lambda}_{k-1}^{-1}\mathbf{F}^{\mathsf{T}})^{-1}$ are cheap.

Belief propagation for the linear dynamical system: Two-filter smoother

With the backward messages included we get the two-filter smoother.

The forward filter

Consists of the left-to-right messages as elaborated on the previous slide.

The backward filter

Consists of the right-to-left messages. In the Gaussian-linear case, these are given by the recursive expressions

$$\begin{aligned}\boldsymbol{\eta}_k^b =& \mathbf{F}^\mathsf{T}\mathbf{Q}^{-1}(\mathbf{Q}^{-1} + \boldsymbol{\Lambda}_{k+1}^b)^{-1}\boldsymbol{\eta}_{k+1}^b + \mathbf{H}^\mathsf{T}\mathbf{R}^{-1}\mathbf{z}_k \\ \boldsymbol{\Lambda}_k^b =& \mathbf{F}\mathbf{Q}^{-1}(\mathbf{Q} - (\mathbf{Q}^{-1} + \boldsymbol{\Lambda}_{k+1}^b)^{-1})\mathbf{F}^\mathsf{T}\mathbf{Q}^{-1} + \mathbf{H}^\mathsf{T}\mathbf{R}^{-1}\mathbf{H}.\end{aligned}$$

Canonical parametrization is necessary because Λ_n may not be invertible.

The smoother

The smoothed density at time step k is then given by

$$\begin{aligned}\boldsymbol{\eta}_k^s =& \boldsymbol{\eta}_{k|k-1} + \boldsymbol{\eta}_k^b \\ \boldsymbol{\Lambda}_k^s =& \boldsymbol{\Lambda}_{k|k-1} + \boldsymbol{\Lambda}_k^b.\end{aligned}$$

Belief propagation for the linear dynamical system: Two-filter smoother

$$
\begin{tikzcd}
\cdots \arrow[d] & \cdots & \cdots \\
p(\mathbf{x}_{k-1}|\mathbf{z}_{1:k-1}) \arrow[d] & & p(\mathbf{z}_{k-1:n}|\mathbf{x}_{k-1}) \arrow[u] \\
p(\mathbf{x}_{k}|\mathbf{z}_{1:k-1}) \arrow[d] \arrow[dr] & & \\
p(\mathbf{x}_{k}|\mathbf{z}_{1:k}) \arrow[d] & p(\mathbf{x}_{k}|\mathbf{z}_{1:n}) & p(\mathbf{z}_{k:n}|\mathbf{x}_{k}) \arrow[l] \arrow[uu] \\
p(\mathbf{x}_{k+1}|\mathbf{z}_{1:k}) \arrow[d] & & \\
p(\mathbf{x}_{k+1}|\mathbf{z}_{1:k+1}) \arrow[d] & & p(\mathbf{z}_{k+1:n}|\mathbf{x}_{k+1}) \arrow[uu] \\
\cdots & \cdots & \cdots \arrow[u]
\end{tikzcd}
$$

We use prediction and not update from the forward filter because we must avoid double-counting information.

Outline

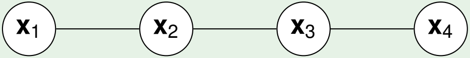
Markov random fields

Pairwise Markov random fields

A pairwise MRF provides a factorization of the pdf of the form

$$p(x_1, \dots, x_N) = \frac{1}{Z} \prod_i \psi(x_i) \prod_{ij} \psi(x_i, x_j)$$

Example: The dynamical system



Cliques

- A clique is a subset of nodes so that every two distinct nodes in that subset are adjacent.
- A maximal clique is a clique that cannot be expanded by adding more nodes.

Example: Cliques in the dynamical system

- The cliques are the sets $\{1\}$, $\{2\}$, $\{3\}$, $\{4\}$, $\{1, 2\}$, $\{2, 3\}$ and $\{3, 4\}$.
- The maximal cliques are the sets $\{1, 2\}$, $\{2, 3\}$ and $\{3, 4\}$.

More about Markov random fields

General Markov random fields

A Markov random field describes a factorization of a pdf over cliques of the graph:

$$p(\mathbf{x}) = \frac{1}{Z} \prod_{c \in \mathcal{C}} \psi_c(\mathbf{x}_c)$$

Is the MRF unique?

- We can have situations where different collections of maximal cliques are possible.
- We can also include potentials over non-maximal cliques.

We can also have several Bayes nets for a given factor graph.

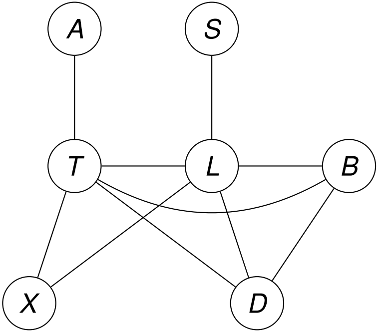
MRFs and belief propagation

Belief propagation for an MRF is similar to belief propagation for a factor graph.

- Instead of products over both factors and variables we only have products over variables.
- An MRF may have cycles which could have been avoided in an equivalent factor graph.

Outline

From Bayes net to Markov random field



To convert a Bayes net to an MRF we must **moralize** all **colliders**.

Colliders (V-structures)

Two nodes that have the same child without being connected is called a collider.

Moralization

Parents of the same child must marry.

The MRF is likely to be more dense than the underlying Bayes net.

For the Chest Clinic network, the **conditional probabilities**

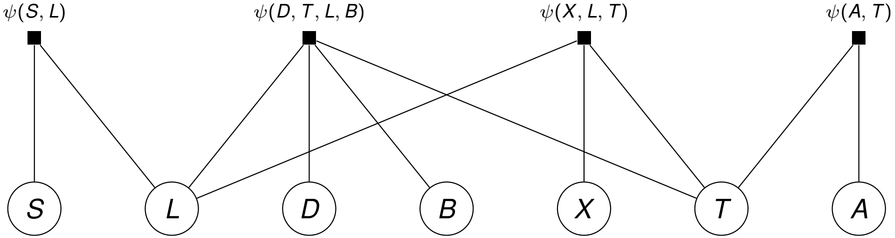
$$p(A) \qquad p(S) \qquad p(B) \qquad p(T|A) \qquad p(L|S) \qquad p(X|T,L) \qquad p(D|T,L,B)$$

are turned into the **potential functions**

$$\psi(T,A) \qquad \psi(L,S) \qquad \psi(X,T,L) \qquad \psi(D,T,L,B).$$

From Bayes net to factor graph

- We can convert all conditional probability functions of the Bayes net to factors.
- . . . or we can make the corresponding MRF first, and convert all its potential functions to factors.



Factor graph of the chest clinic example displayed to emphasize its **bipartite** naturer.

From factor graph to Bayes net (The variable elimination algorithm)

for *each node* $\mathbf{x}_i$ **do**

$\mathcal{S}(i) \leftarrow$ all nodes involved in factors adjacent to i except i;

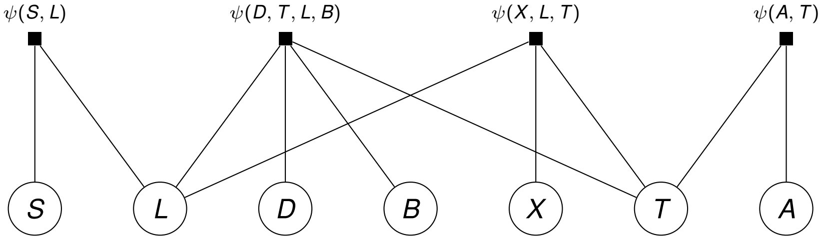
$\phi(\mathbf{x}_{\mathcal{S}(i)}) \leftarrow \prod_{c \in \mathrm{ne}(i)} \psi_c(\mathbf{x}_c)$;

$q(\mathbf{x}_i | \mathbf{x}_{\mathcal{S}(i)}) \tau(\mathbf{x}_{\mathcal{S}(i)}) \leftarrow \phi(\mathbf{x}_{\mathcal{S}(i)})$;

Replace factors in $\mathrm{ne}(i)$ with $\tau(\mathbf{x}_{\mathcal{S}(i)})$;

Insert $q(\mathbf{x}_i | \mathbf{x}_{\mathcal{S}(i)})$ in Bayes net ;

end



From factor graph to Bayes net (The variable elimination algorithm)

for *each node* $\mathbf{x}_i$ **do**

$\mathcal{S}(i) \leftarrow$ all nodes involved in factors adjacent to i except i;

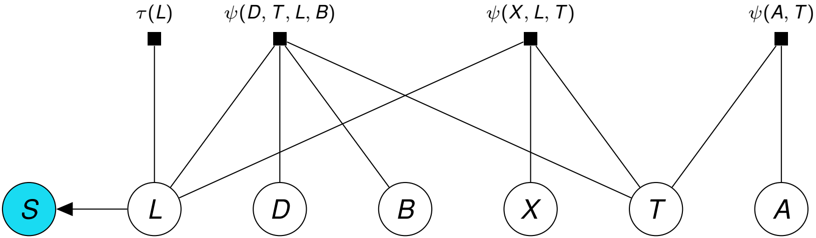
$\phi(\mathbf{x}_{\mathcal{S}(i)}) \leftarrow \prod_{c \in \mathrm{ne}(i)} \psi_c(\mathbf{x}_c)$;

$q(\mathbf{x}_i | \mathbf{x}_{\mathcal{S}(i)}) \tau(\mathbf{x}_{\mathcal{S}(i)}) \leftarrow \phi(\mathbf{x}_{\mathcal{S}(i)})$;

Replace factors in $\mathrm{ne}(i)$ with $\tau(\mathbf{x}_{\mathcal{S}(i)})$;

Insert $q(\mathbf{x}_i | \mathbf{x}_{\mathcal{S}(i)})$ in Bayes net ;

end



From factor graph to Bayes net (The variable elimination algorithm)

for *each node* $\mathbf{x}_i$ **do**

$\mathcal{S}(i) \leftarrow$ all nodes involved in factors adjacent to i except i;

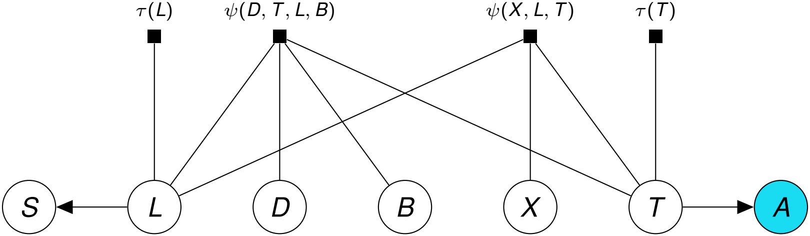
$\phi(\mathbf{x}_{\mathcal{S}(i)}) \leftarrow \prod_{c \in \mathrm{ne}(i)} \psi_c(\mathbf{x}_c)$;

$q(\mathbf{x}_i | \mathbf{x}_{\mathcal{S}(i)}) \tau(\mathbf{x}_{\mathcal{S}(i)}) \leftarrow \phi(\mathbf{x}_{\mathcal{S}(i)})$;

Replace factors in $\mathrm{ne}(i)$ with $\tau(\mathbf{x}_{\mathcal{S}(i)})$;

Insert $q(\mathbf{x}_i | \mathbf{x}_{\mathcal{S}(i)})$ in Bayes net ;

end



From factor graph to Bayes net (The variable elimination algorithm)

for *each node* $\mathbf{x}_i$ **do**

$\mathcal{S}(i) \leftarrow$ all nodes involved in factors adjacent to i except i;

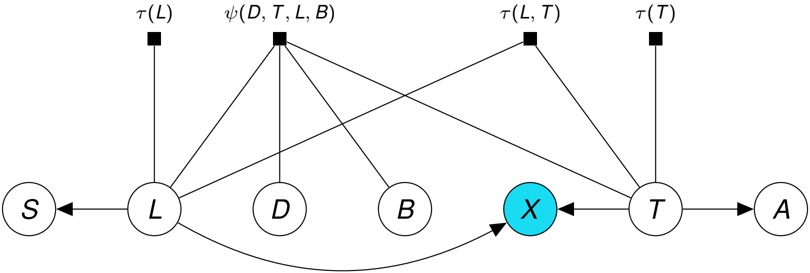
$\phi(\mathbf{x}_{\mathcal{S}(i)}) \leftarrow \prod_{c \in \mathrm{ne}(i)} \psi_c(\mathbf{x}_c)$;

$q(\mathbf{x}_i | \mathbf{x}_{\mathcal{S}(i)}) \tau(\mathbf{x}_{\mathcal{S}(i)}) \leftarrow \phi(\mathbf{x}_{\mathcal{S}(i)})$;

Replace factors in $\mathrm{ne}(i)$ with $\tau(\mathbf{x}_{\mathcal{S}(i)})$;

Insert $q(\mathbf{x}_i | \mathbf{x}_{\mathcal{S}(i)})$ in Bayes net ;

end



From factor graph to Bayes net (The variable elimination algorithm)

for *each node* $\mathbf{x}_i$ **do**

$\mathcal{S}(i) \leftarrow$ all nodes involved in factors adjacent to i except i;

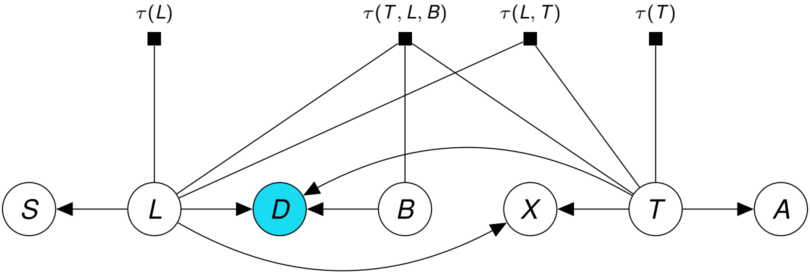
$\phi(\mathbf{x}_{\mathcal{S}(i)}) \leftarrow \prod_{c \in \mathrm{ne}(i)} \psi_c(\mathbf{x}_c)$;

$q(\mathbf{x}_i | \mathbf{x}_{\mathcal{S}(i)}) \tau(\mathbf{x}_{\mathcal{S}(i)}) \leftarrow \phi(\mathbf{x}_{\mathcal{S}(i)})$;

Replace factors in $\mathrm{ne}(i)$ with $\tau(\mathbf{x}_{\mathcal{S}(i)})$;

Insert $q(\mathbf{x}_i | \mathbf{x}_{\mathcal{S}(i)})$ in Bayes net ;

end



From factor graph to Bayes net (The variable elimination algorithm)

for *each node* $\mathbf{x}_i$ **do**

  $\mathcal{S}(i) \leftarrow$ all nodes involved in factors adjacent to i except i;

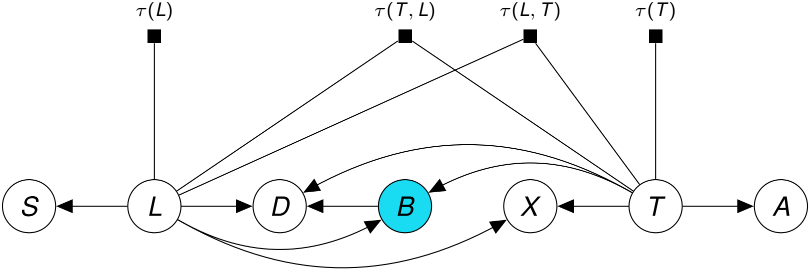
  $\phi(\mathbf{x}_{\mathcal{S}(i)}) \leftarrow \prod_{c \in \mathrm{ne}(i)} \psi_c(\mathbf{x}_c)$;

  $q(\mathbf{x}_i | \mathbf{x}_{\mathcal{S}(i)}) \tau(\mathbf{x}_{\mathcal{S}(i)}) \leftarrow \phi(\mathbf{x}_{\mathcal{S}(i)})$;

  Replace factors in $\mathrm{ne}(i)$ with $\tau(\mathbf{x}_{\mathcal{S}(i)})$;

  Insert $q(\mathbf{x}_i | \mathbf{x}_{\mathcal{S}(i)})$ in Bayes net ;

end



From factor graph to Bayes net (The variable elimination algorithm)

for *each node* $\mathbf{x}_i$ **do**

  $\mathcal{S}(i) \leftarrow$ all nodes involved in factors adjacent to i except i;

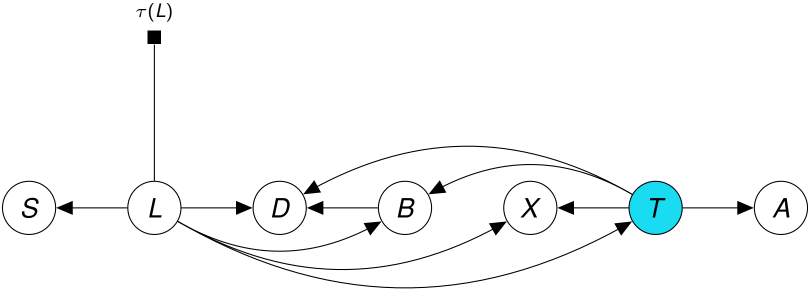
  $\phi(\mathbf{x}_{\mathcal{S}(i)}) \leftarrow \prod_{c \in \mathrm{ne}(i)} \psi_c(\mathbf{x}_c)$;

  $q(\mathbf{x}_i | \mathbf{x}_{\mathcal{S}(i)}) \tau(\mathbf{x}_{\mathcal{S}(i)}) \leftarrow \phi(\mathbf{x}_{\mathcal{S}(i)})$;

  Replace factors in $\mathrm{ne}(i)$ with $\tau(\mathbf{x}_{\mathcal{S}(i)})$;

  Insert $q(\mathbf{x}_i | \mathbf{x}_{\mathcal{S}(i)})$ in Bayes net ;

end



From factor graph to Bayes net (The variable elimination algorithm)

for *each node* $\mathbf{x}_i$ **do**

$\mathcal{S}(i) \leftarrow$ all nodes involved in factors adjacent to i except i;

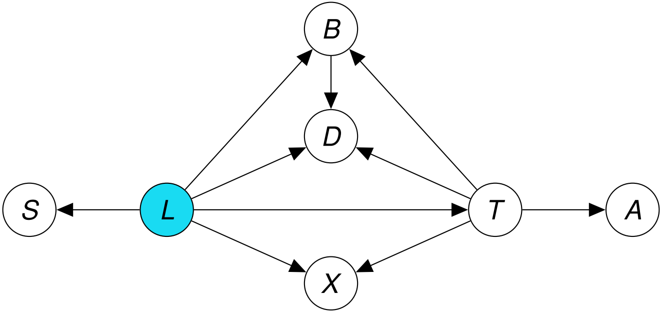
$\phi(\mathbf{x}_{\mathcal{S}(i)}) \leftarrow \prod_{c \in \mathrm{ne}(i)} \psi_c(\mathbf{x}_c)$;

$q(\mathbf{x}_i | \mathbf{x}_{\mathcal{S}(i)}) \tau(\mathbf{x}_{\mathcal{S}(i)}) \leftarrow \phi(\mathbf{x}_{\mathcal{S}(i)})$;

Replace factors in $\mathrm{ne}(i)$ with $\tau(\mathbf{x}_{\mathcal{S}(i)})$;

Insert $q(\mathbf{x}_i | \mathbf{x}_{\mathcal{S}(i)})$ in Bayes net ;

end



The road ahead

The Bayes tree (Junction tree)

- The elimination algorithm is the key tool to calculate marginals in a PGM.
- Organize the calculations in a tree structure so that they can be re-used as much as possible.

The Rauch-Tung-Striebel smoother

Bayes tree for a dynamical system.

- Forward pass (KF): Find the τ's and the q's.
- Backward pass (smoothing): Marginalize along the arrows of the Bayes net.

Triangulating the information matrix

Consider a multivariate Gaussian represented as a factor graph.

- Bayes net is a DAG.

$\Rightarrow$ Can be represented by an upper triangular matrix.

Factor Graph SLAM

- Model available information as factors.
- Estimate the entire trajectory.
- Sparse linear algebra techniques for efficient optimization.