LTAT.02.004 Machine Learning II

Sequence models

Sven Laur
University of Tartu

Discrete random variables

▷ A *random variable* X with possible *outcomes* $x \in \mathrm{supp}(X)$

▷ Compact notation for probabilities

$$\Pr[x_1] := \Pr[\xi \leftarrow X_1 : \xi = x_1]$$

$$\Pr[x_1 \wedge x_2] := \Pr[\xi_1 \leftarrow X_1, \xi_2 \leftarrow X_2 : \xi_1 = x_1 \wedge \xi_2 = x_2]$$

▷ Bayes formula

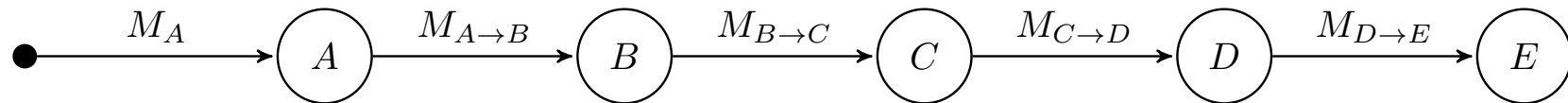$$\Pr[a|b] = \frac{\Pr[a \wedge b]}{\Pr[b]} = \frac{\Pr[b|a]\Pr[a]}{\Pr[b]}$$

▷ Independence of random variables $X_1 \ldots X_m \perp Y_1, \ldots Y_n$:

$$\Pr[x_1 \wedge \ldots \wedge x_m \wedge y_1 \wedge \ldots \wedge y_n] = \Pr[x_1 \wedge \ldots \wedge x_m] \cdot \Pr[y_1 \wedge \ldots \wedge y_n]$$

▷ Marginalisation over variables $Y_1, \ldots, Y_n$:

$$\Pr[x_1 \wedge \ldots \wedge x_m] = \sum_{y_1,\ldots,y_n} \Pr[x_1 \wedge \ldots \wedge x_m \wedge y_1 \wedge \ldots \wedge y_n]$$

Markov chain

$$\bullet \xrightarrow{M_A} \boxed{A} \xrightarrow{M_{A \to B}} \boxed{B} \xrightarrow{M_{B \to C}} \boxed{C} \xrightarrow{M_{C \to D}} \boxed{D} \xrightarrow{M_{D \to E}} \boxed{E}$$

Definition. Let $X_1, X_2, \ldots$ be correlated random variables such that the probability of the observation x_{i+1} depends only on the observation x_i. Then the entire process is known as Markov chain.

Parametrisation. Markov chain is determined by specifying

▷ state spaces $\mathcal{S}_1 \ldots, \mathcal{S}_n$

▷ initial probabilities $\Pr[x_1]$ given as vectors

▷ state transition probabilities $\Pr[x_{i+1}|x_i]$ given as matrices

What questions can we ask?

Sampling: What are typical outcomes of the chain?
▷ Synthesis of time-series, textures, sounds, games movements.

Stationary distribution: What happens if we run the chain infinitely long?
▷ Getting samples from an unnormalised posterior, optimisation tasks.

Likelihood estimation: What is a probability of an observation $x_1, \ldots, x_n$?
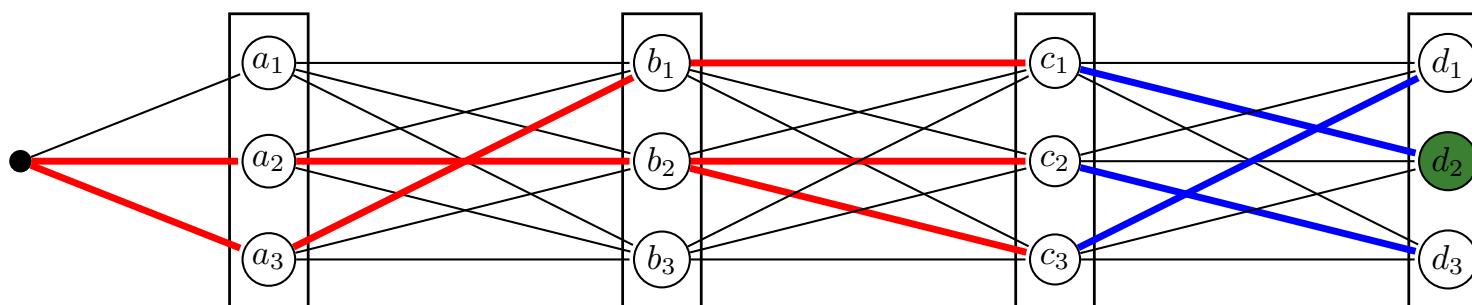▷ Reasoning about probabilities and clustering sequences.

Decoding: What is the most probable outcome $x_1, \ldots, x_n$?
▷ Imputing missing values. Rudimentary logical reasoning.

Parameter estimation: What is are the model parameters?
▷ Machine learning – finding parameters based on observations.

Posterior maximisation in a chain

Inference goal. Given evidence at the ends of the chain find the sequence of states x that maximise the posterior probability $\Pr[x|\text{evidence}]$.

▷ The log-posterior $\log \Pr[x|\text{evidence}]$ decomposes into a sum.

▷ We must find a sequence with maximal weight.

▷ The task can be split into subtask as all subpaths of the path with maximal weight must have maximal weight.

▷ The corresponding iterative algorithm is known as Viterbi algorithm.

Belief propagation in a chain
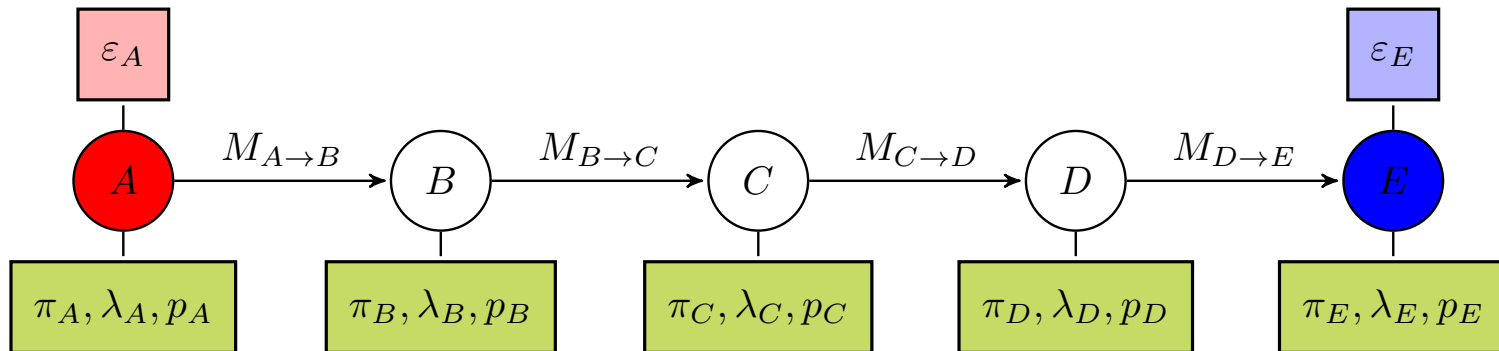
Inference goal. Given evidence at the ends of the chain find marginal posterior probabilities for each node in the chain.

▷ Evidence ε_V is an observational data associated with the node V.

▷ Upstream evidence$^-$ is the evidence at the end of chain.

▷ Downstream evidence$^+$ is the evidence at the beginning of chain.

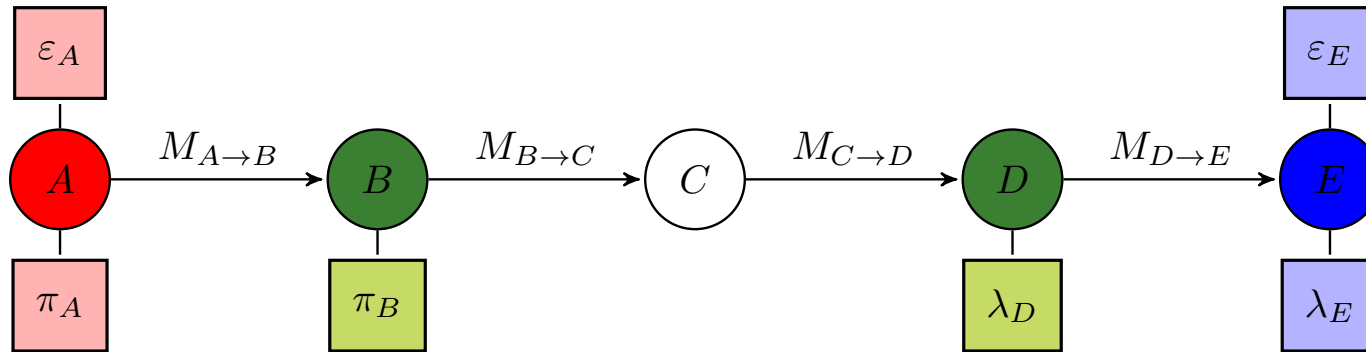▷ Attributes π_V, λ_V, p_V are needed to compute marginal distributions.

Initialisation
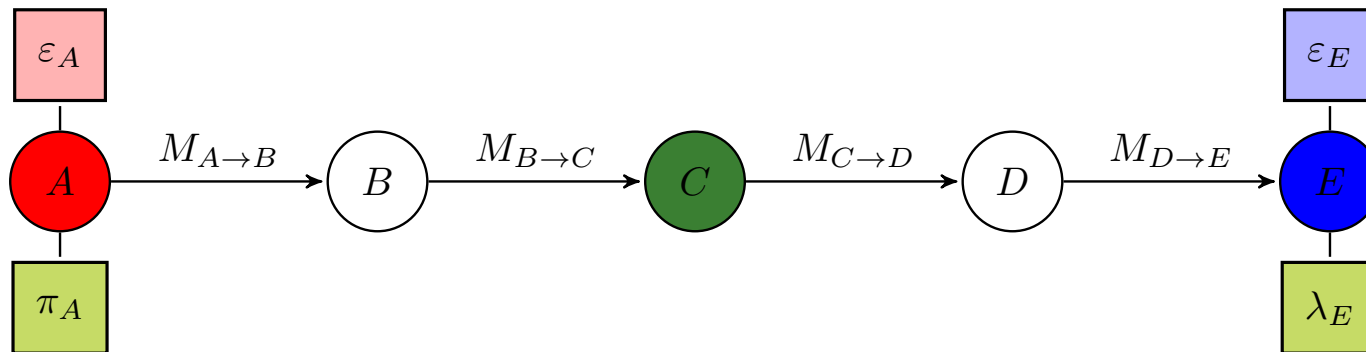
▷ Direct evidence ε_V determines the value of V.

▷ Indirect evidence ε_V determines the value distribution for V.

▷ We can assign the prior for the first and likelihood for the last node

$$\pi_A(a) = \Pr\left[A = a | \text{evidence}^+\right] = \Pr\left[A = a | \varepsilon_A\right]$$

$$\lambda_E(e) = \Pr\left[\text{evidence}^- | E = e\right] = \Pr\left[\varepsilon_E | E = e\right]$$

Belief propagation

Inference goal
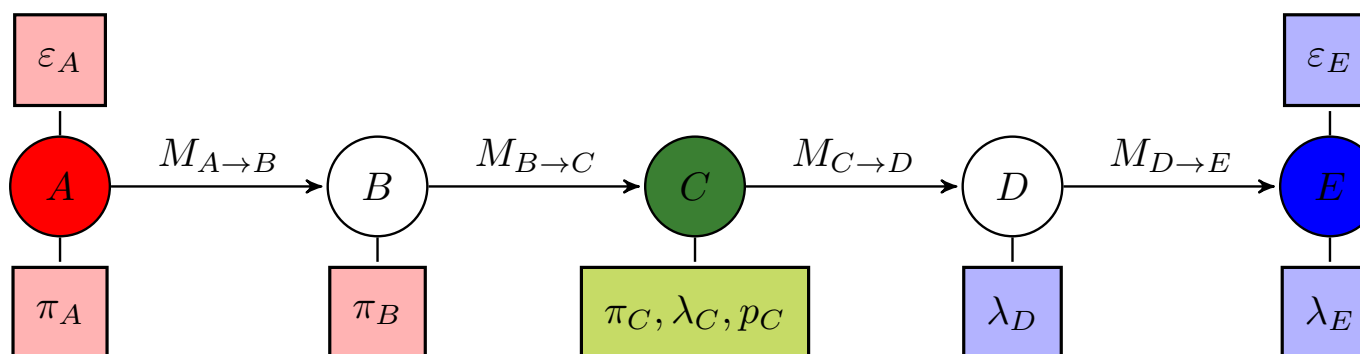
$$\pi_B(b) = \Pr\left[b|\text{evidence}^+\right]$$

$$\lambda_D(d) = \Pr\left[\text{evidence}^-|d\right]$$

Iterative propagation rules

▷ Marginalisation gives an update rule $\lambda_D = M_{D\to E}\lambda_E$.

▷ Marginalisation gives an update rule $\pi_B \propto \pi_A M_{A\to B}$.
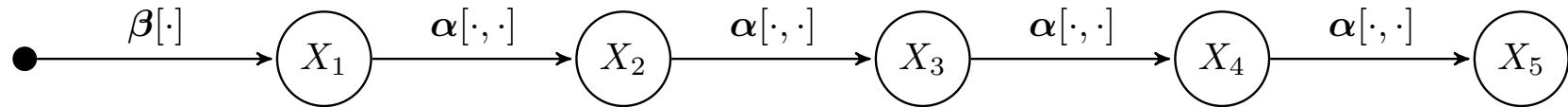
Belief propagation

Inference goal

$$p_C(c) = \Pr\left[c | \text{evidence}^+, \text{evidence}^-\right]$$

Iterative update rule

▷ Bayes formula gives $p_C \propto \pi_C \otimes \lambda_C$.

Parameter inference for homogenous case

For a sequence of observations $\boldsymbol{x} = (x_1, \ldots, x_n)$ the log-likelihood is

$$\ell[\boldsymbol{x}] = \log \underbrace{\Pr[x_1]}_{\beta[x_1]} + \sum_{i=1}^{n-1} \log \underbrace{\Pr[x_{i+1}|x_i]}_{\alpha[x_i, x_{i+1}]}$$

$$= \log \beta[x_1] + \sum_{u_1, u_2} k(u_1, u_2) \log \alpha[u_1, u_2]$$

where $k(u_1, u_2)$ is the count of bigrams u_1, u_2 in the sequence $\boldsymbol{x}$.

Posterior decomposition

As a result the log-likelihood of unnormalised posterior decomposes into the sum of independent terms

$$\log p[\boldsymbol{\alpha}, \boldsymbol{\beta}|\boldsymbol{x}] = \sum_{u_1} k(u_1) \log \beta[u_1] + \log p(\boldsymbol{\beta})$$

$$+ \sum_{u_1, u_2} k(u_1, u_2) \log \alpha[u_1, u_2] + \sum_{u_1} \log p(\boldsymbol{\alpha}[u_1, \cdot])$$

where

▷ $k(u_1)$ is the count u_1 at the beginning of the observed sequences
▷ $k(u_1, u_2)$ is the count of bigrams u_1, u_2 in the observed sequences.
▷ $p(\boldsymbol{\beta})$ is the prior for an entire vector of initial probabilities
▷ $p(\boldsymbol{\alpha}[u_1, \cdot])$ is the prior for the transition probabilities from u_1

Reduction to the dice throwing experiment

Posterior decomposition leads to many independent optimisation tasks

$$\sum_{u_1} k(u_1) \log \beta[u_1] + \log p(\boldsymbol{\beta}) \to \max$$

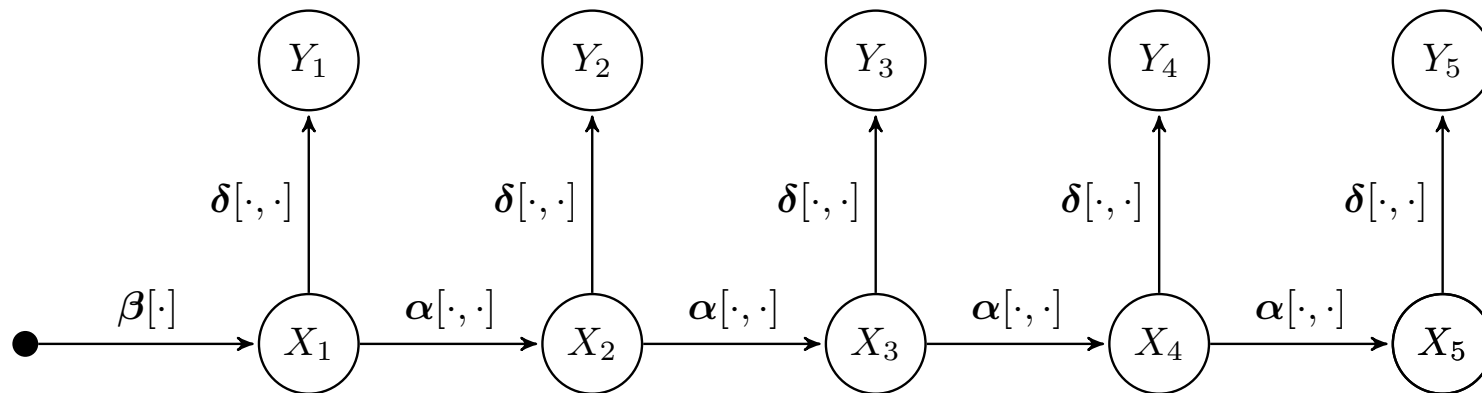$$\sum_{u_2} k(u_1, u_2) \log \alpha[u_1, u_2] + \log p(\boldsymbol{\alpha}[u_1, \cdot]) \to \max$$

where each of these is equivalent to optimisation of dice throwing posterior.
Thus Maximum Aposteriori estimates for parameters are

$$\beta[u_1] = \frac{k(u_1) + c}{k(*) + mc} \qquad \alpha[u_1, u_2] = \frac{k(u_1, u_2) + c}{k(u_1, *) + mc}$$

where

▷ $*$ is a wildcard symbol in the count queries

▷ m is the number of states and c is a constant for Laplacian smoothing.

Hidden Markov Model

Definition. Let $X_1, X_2, \ldots$ be hidden states that form a Markov chain and let $Y_1, Y_2, \ldots$ be observations that the probability of y_i depends only on the state x_i. Then the entire process is known as Hidden Markov Model.

Common tasks

$\triangleright$ parameter estimation

$\triangleright$ filtering, smoothing, prediction

Applications

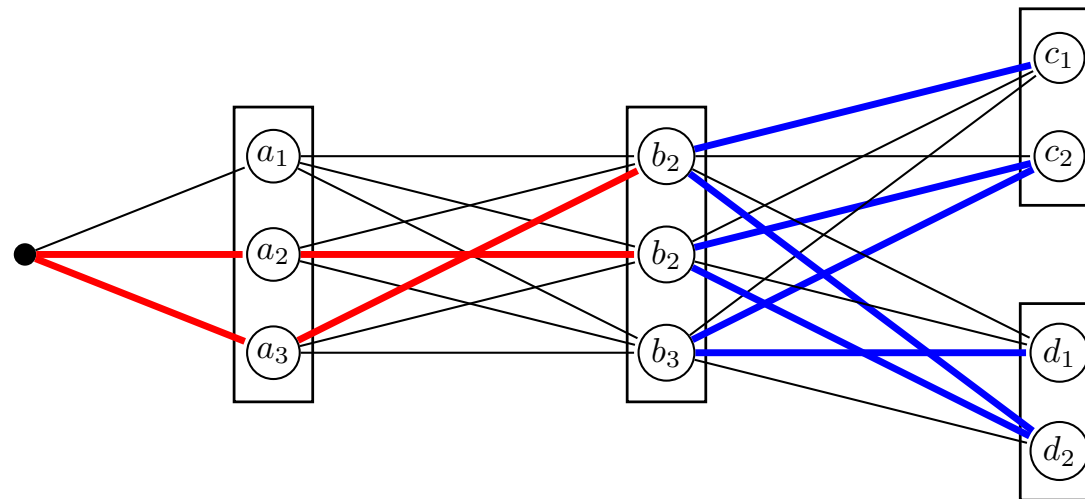## Modelling and prediction

▷ stock prices

▷ linear control algorithms

Sequence annotation

▷ fraud detection

▷ change detection

▷ functional motifs of DNA sequences

Decoding

▷ speech recognition

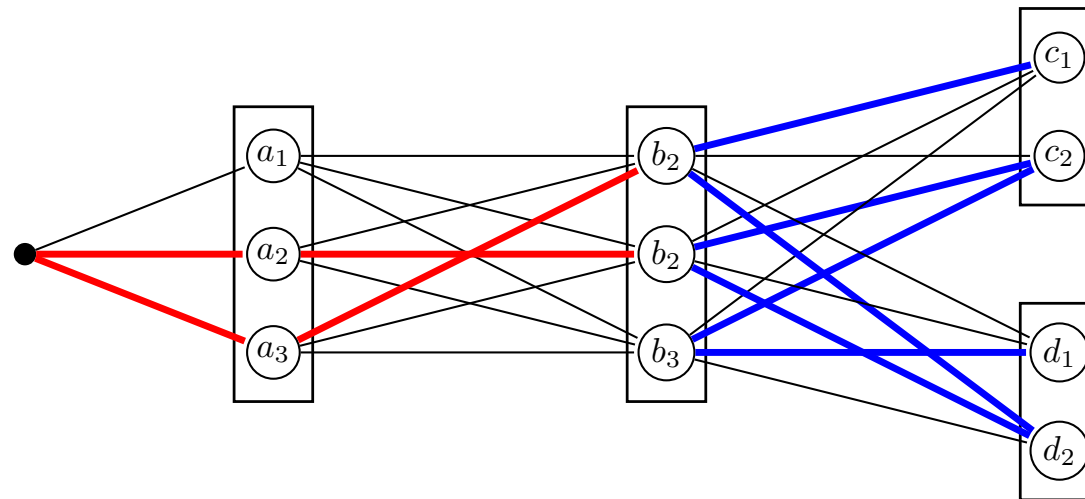▷ communication over a nosy channels

▷ object tracking and data fusion

Posterior maximisation in a tree

Inference goal. Given evidence at the ends of the chain find the sequence of states x that maximise the posterior probability $\Pr\left[x|\text{evidence}\right]$.

▷ The log-posterior $\log \Pr\left[x|\text{evidence}\right]$ decomposes into a sum.

▷ We must find a tree with maximal weight.

Decomposition into subtasks

All subtrees of the tree with maximal weight must have maximal weight.

▷ We can build chains with maximum weight form leafs

▷ We can merge subtrees with maximum weight to maximise the weight.

▷ The algorithm works from leafs to the root node.

▷ The corresponding iterative algorithm is known as Viterbi algorithm.
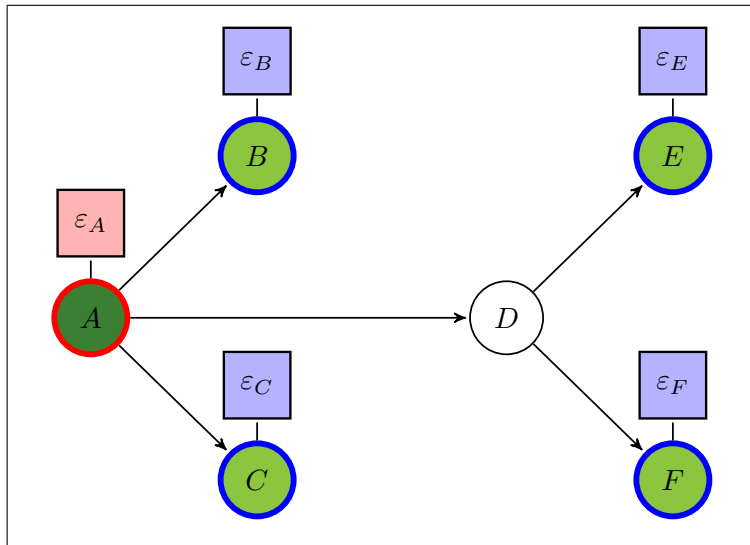
Belief propagation in a tree
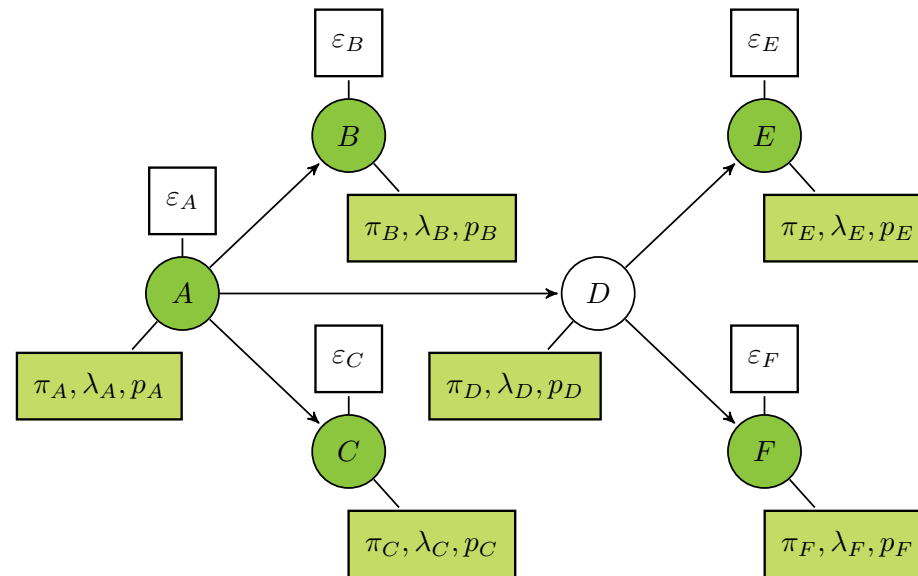
Inference goal. Given evidence at the ends of the leafs and the root of tree find marginal posterior probabilities for each node in the tree.

▷ Evidence ε_V is an observational data associated with the node V.

▷ Attributes π_V, λ_V, p_V are needed to compute marginal distributions.

Evidence decomposition
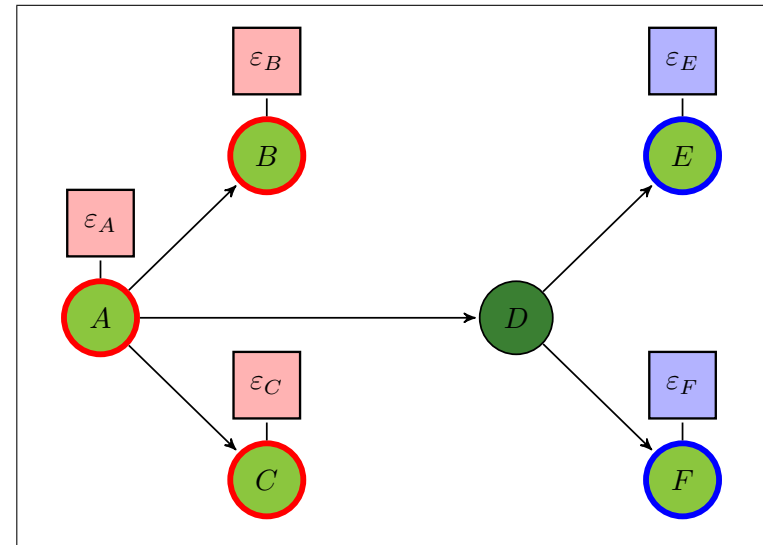
▷ Evidence decomposes into up- and downstream evidence

▷ Downstream evidence$^-(V)$ is reachable through child nodes.

▷ Upstream evidence$^+(V)$ is reachable through the predessesor node.

▷ Different nodes have totally different decompositions.

Initialisation
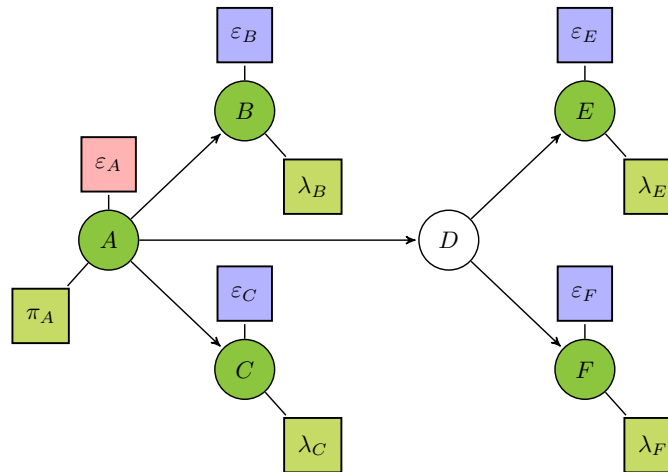
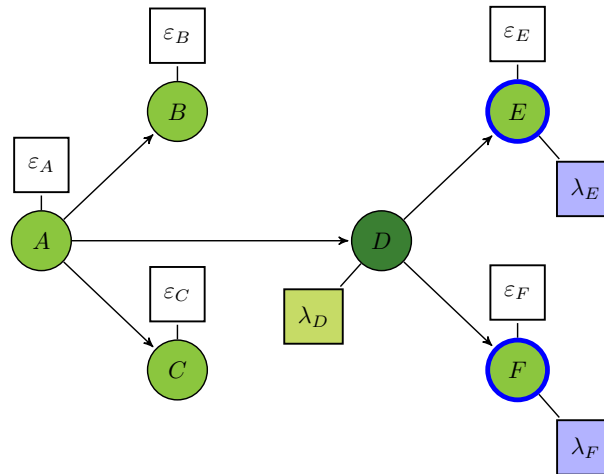**Goal.** Assign prior to the root node and likelihood to the leaf nodes.

$$\pi_A(a) = \Pr\left[A = a | \text{evidence}^+(A)\right] = \Pr\left[A = a | \varepsilon_A\right]$$

$$\lambda_B(b) = \Pr\left[\text{evidence}^-(B) | F = f\right] = \Pr\left[\varepsilon_B | B = b\right]$$

$$\cdots$$

$$\lambda_F(f) = \Pr\left[\text{evidence}^-(F) | F = f\right] = \Pr\left[\varepsilon_F | F = f\right]$$
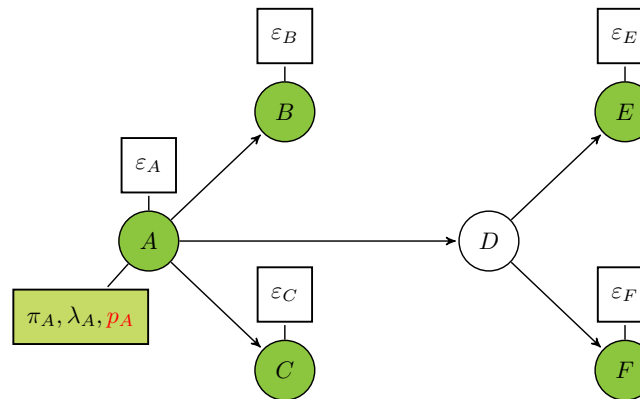
Likelihood propagation

Inference goal

$$\lambda_D(d) = \Pr\left[\mathsf{evidence}^-(D)|D = d\right]$$

Iterative propagation rules

▷ Independence gives a pooling rule $\lambda_D = \lambda_1 \otimes \lambda_2$

▷ Marginalisation gives rules $\lambda_1 = M_{D \to E}\lambda_E$ and $\lambda_2 = M_{D \to F}\lambda_F$.
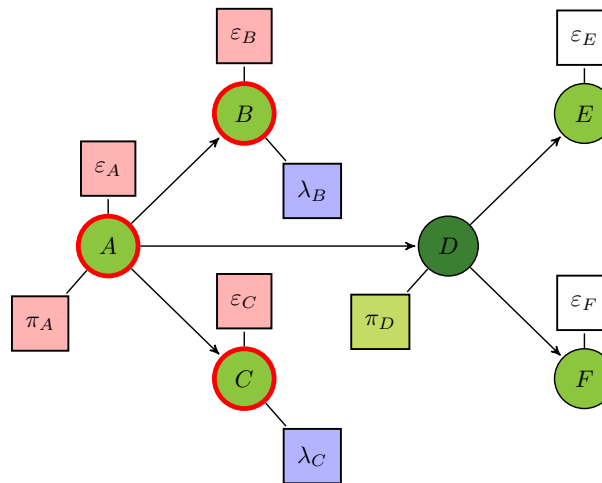
Posterior propagation

Inference goal

$$p_A(a) = \Pr\left[A = a | \text{evidence}^+(A), \text{evidence}^-(A)\right]$$

Iterative propagation rule

▷ Marginal conditional probability $p_A \propto \pi_A \otimes \lambda_A$
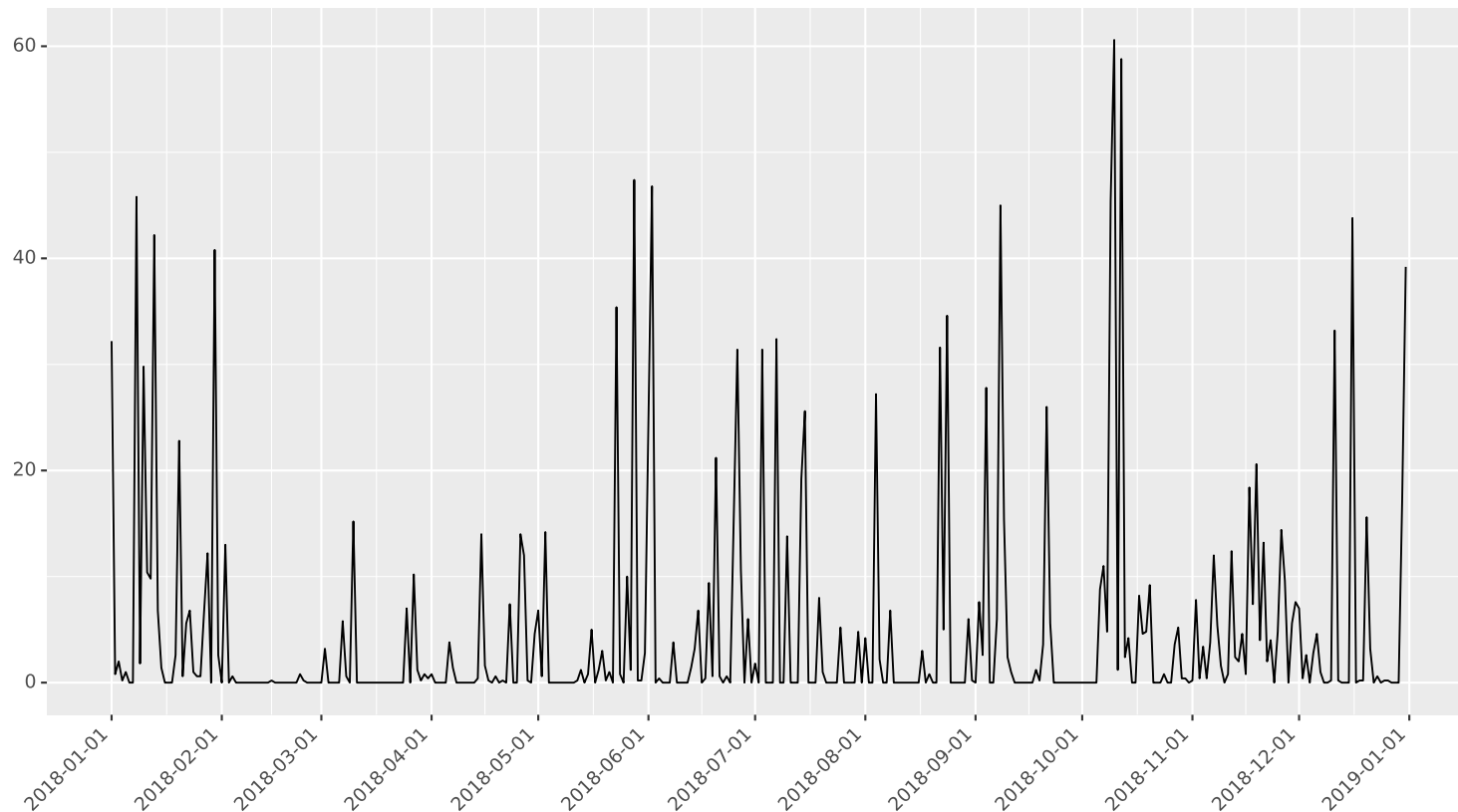
Prior propagation

Inference goal

$$\pi_D(d) = \Pr\left[D = d | \text{evidence}^+(D)\right]$$

$$= \Pr\left[D = d | \text{evidence}^+(A), \text{evidence}^-(B), \text{evidence}^-(C)\right]$$
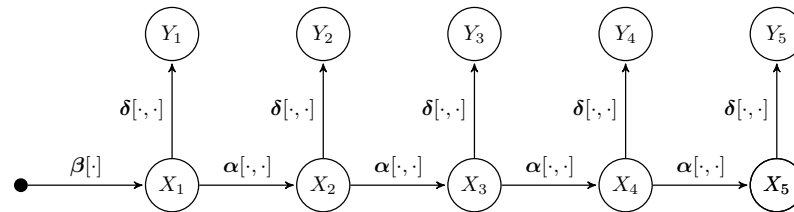
Iterative propagation rule

▷ Prior can be computed as $\pi_D \propto \pi_A M_{A \rightarrow D} \otimes M_{A \rightarrow B} \lambda_B \otimes M_{A \rightarrow C} \lambda_C$.

Application on rainfall data

There are two monsoon seasons in Singapore: dry and wet phase.

Modelling with Hidden Markov Model

Markov chain with states $\mathcal{S} = \{0, 1\}$ and parameters

$$\boldsymbol{\beta} = (0.5, 0.5)$$

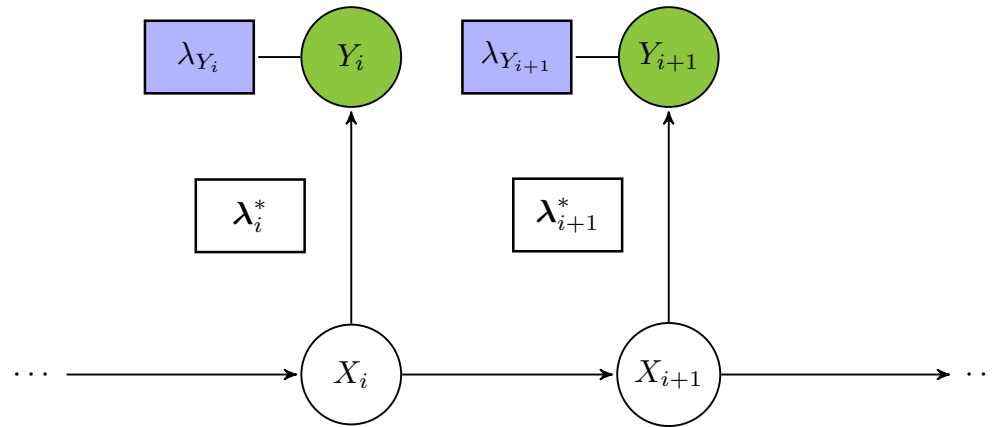$$\boldsymbol{\alpha} = \begin{pmatrix} 0.95 & 0.05 \\ 0.05 & 0.95 \end{pmatrix}$$

Emission distributions

$$Y_i | X_i = 0 \sim \mathcal{N}(\mu_0, \sigma_0)$$

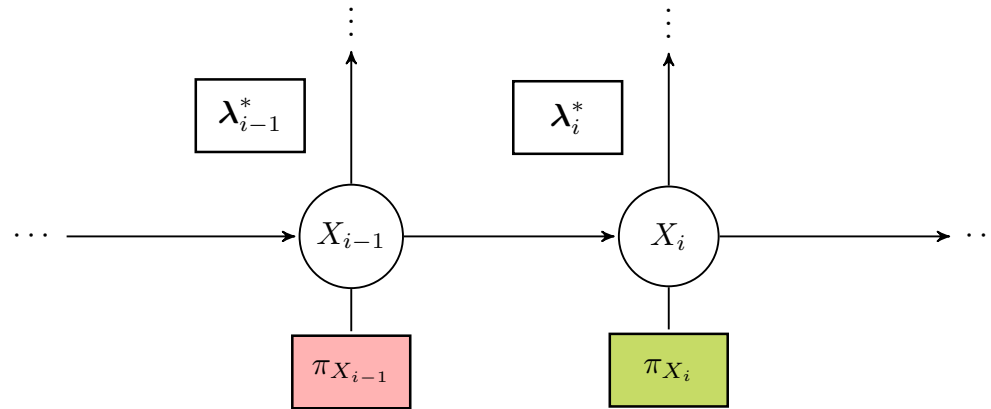$$Y_i | X_i = 1 \sim \mathcal{N}(\mu_1, \sigma_1)$$

Belief propagation. Initialisation

$\triangleright$ We have a direct evidence $Y_i = y_i$ for each node Y_i.

$\triangleright$ The likelihood vector is infinite and captured by $\lambda_{Y_i} = \delta_{y_i}$.

$\triangleright$ The local likelihood $\lambda_i^*(x_i) = \Pr[Y_i = y_i | x_i]$ is a finite vector.
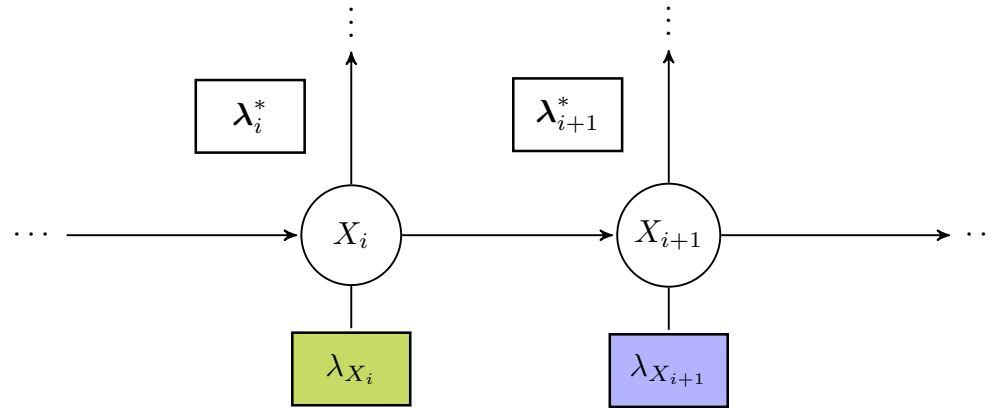
Prior propagation. Filtering

Prior propagation rule yields

$$\pi_{X_i}(x_i) \propto \sum_{x_{i-1}\in\mathcal{S}} \alpha[x_{i-1}, x_i] \cdot \lambda^*_{i-1}(x_{i-1}) \cdot \pi_{X_{i-1}}(x_{i-1})$$

Now we can do filtering

$$\Pr\left[x_i | y_1, \ldots, y_i\right] \propto \pi_{X_i}(x_i) \cdot \lambda^*_i(x_i)$$
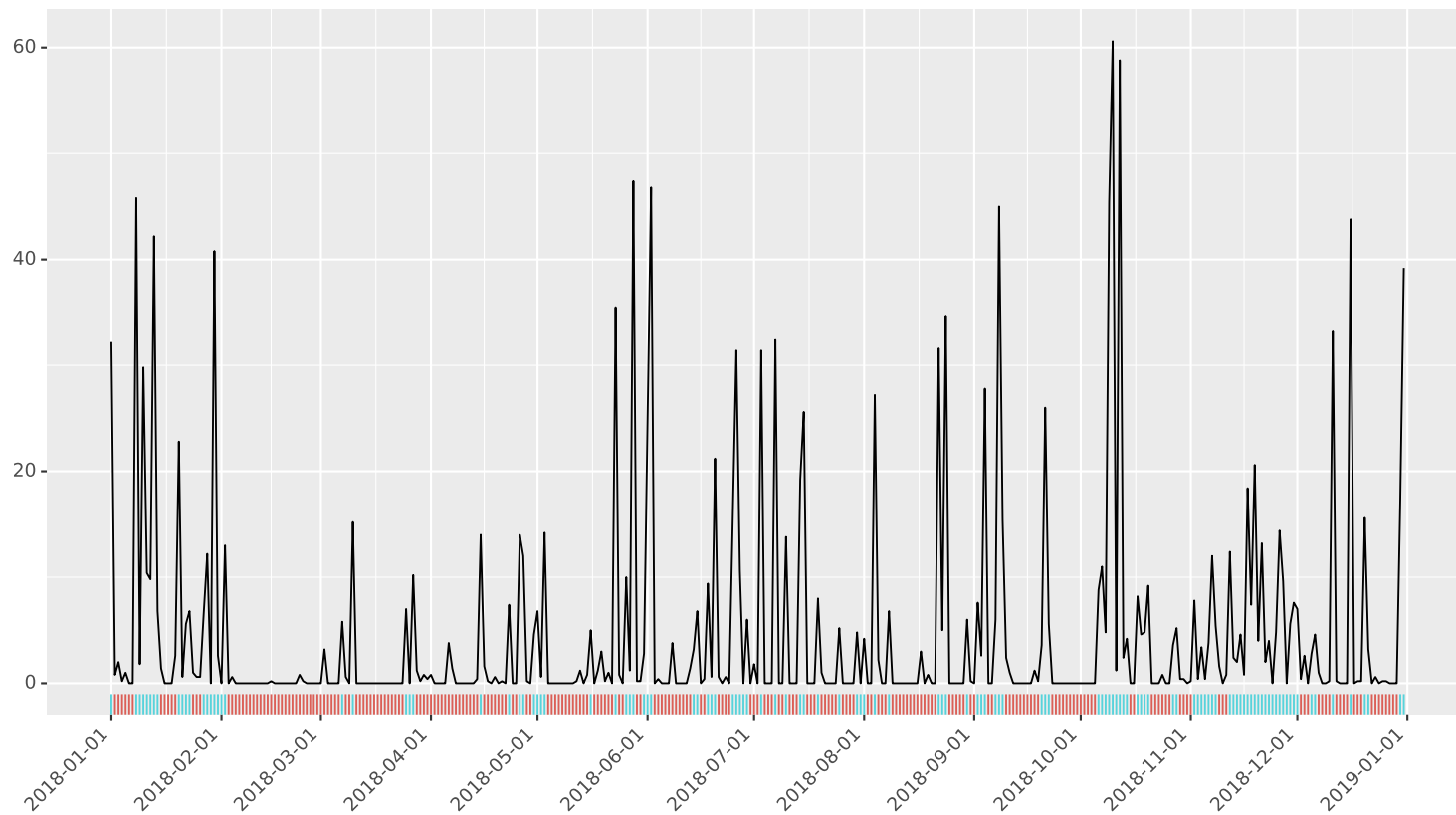
Likelihood propagation. Smoothing

Likelihood propagation rule yields

$$\lambda_{X_i}(x_i) \propto \sum_{x_{i+1} \in \mathcal{S}} \alpha[x_i, x_{i+1}] \cdot \lambda_{X_{i+1}}(x_{i+1}) \cdot \lambda_i^*(x_i)$$

Now we can do smoothing

$$\Pr[x_i | y_1, \ldots, y_n] \propto \pi_{X_i}(x_i) \cdot \lambda_{X_i}(x_i)$$

Annotated rainfall data

Sensor fusion problem. Kalman filter

Several sensors measure a physical system

▷ Measurements are observable as $\boldsymbol{y} \in \mathbb{R}^p$.

▷ Physical system has an hidden state $\boldsymbol{x} \in \mathbb{R}^n$.

▷ Physical system evolves linearly $\boldsymbol{x}_{i+1} = A\boldsymbol{x}_i + \boldsymbol{w}_i$.

▷ Measurements are linear from the state $\boldsymbol{y}_i = C\boldsymbol{x}_i + \boldsymbol{v}_i$.

Unknown quantities in the system

▷ Measurement noise $\boldsymbol{v}_t$ is modelled with a normal distribution.

▷ Unknown control signal $\boldsymbol{w}_i$ is modelled with a normal distribution.

▷ Unknown initial state $\boldsymbol{x}_0$ is modelled with a normal distribution.

▷ Quantities $\boldsymbol{x}_0, \boldsymbol{v}_i, \boldsymbol{w}_i$ are assumed to be independent.

▷ All normal distributions can have complex correlation structure.

Belief propagation. Initialisation
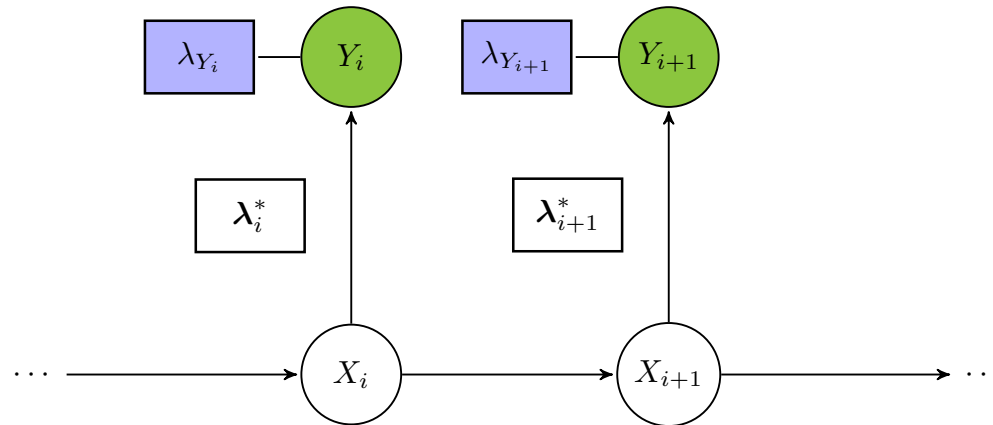
▷ We have a direct evidence $Y_i = y_i$ for each node Y_i.

▷ The likelihood vector is infinite and captured by $\lambda_{Y_i} = \delta_{y_i}$.

▷ The local likelihood $\lambda_i^*(x_i) = p[Y_i = y_i | x_i]$ is an infinite vector.

▷ The form $\boldsymbol{y}_i = C\boldsymbol{x}_i + \boldsymbol{v}_i$ assures that $\boldsymbol{y}_i | \boldsymbol{x}_i$ is normal distribution.

▷ The local likelihood λ_i^* has a finite description.
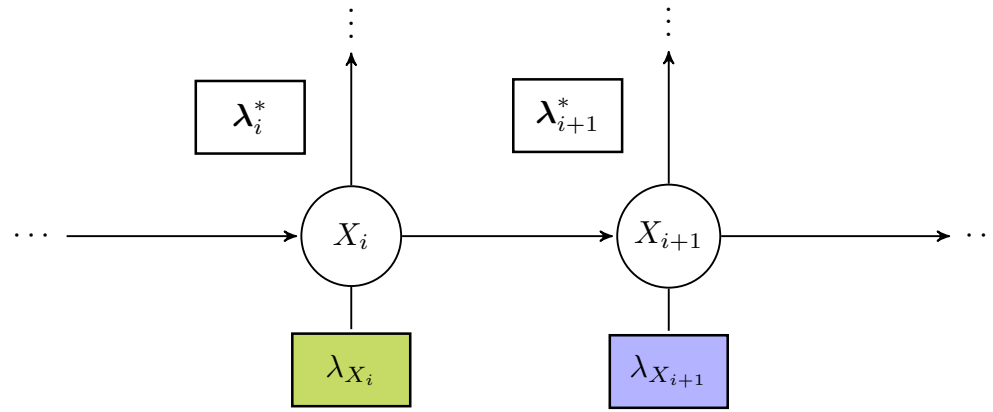
Prior propagation. Filtering
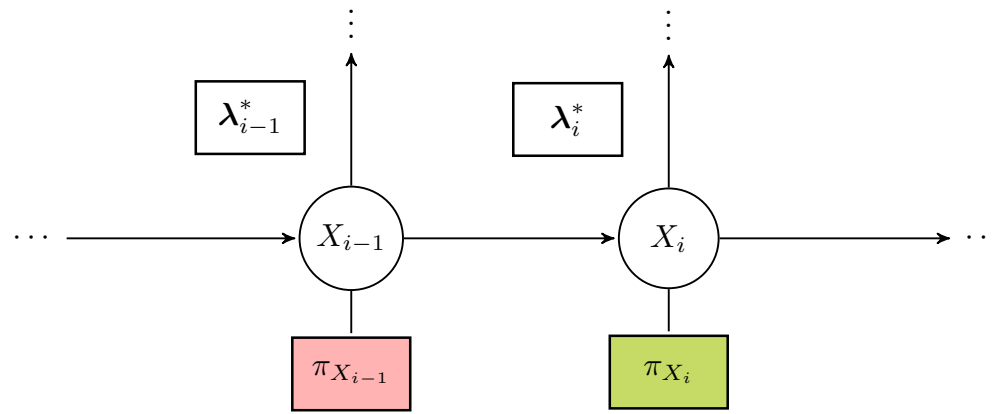
Prior propagation rule

$$\pi_{X_i}(\boldsymbol{x}_i) \propto \int\limits_{\boldsymbol{x}_{i-1}} \alpha[\boldsymbol{x}_{i-1}, \boldsymbol{x}_i] \cdot \lambda^*_{i-1}(\boldsymbol{x}_{i-1}) \cdot \pi_{X_{i-1}}(\boldsymbol{x}_{i-1}) d\boldsymbol{x}_{i-1}$$

leads to a finite description because on the right is a normal distribution.

Likelihood propagation. Smoothing

Likelihood propagation rule

$$\lambda_{X_i}(x_i) \propto \int\limits_{\boldsymbol{x}_{i+1}} \alpha[\boldsymbol{x}_i, \boldsymbol{x}_{i+1}] \cdot \lambda_{X_{i+1}}(\boldsymbol{x}_{i+1}) \cdot \lambda_i^*(\boldsymbol{x}_i) d\boldsymbol{x}_{i+1}$$

leads to a finite description because on the right is a normal distribution.