LTAT.02.004 Machine Learning II

Sequence models

Sven Laur
University of Tartu

How to write a good touchscreen keyboard?

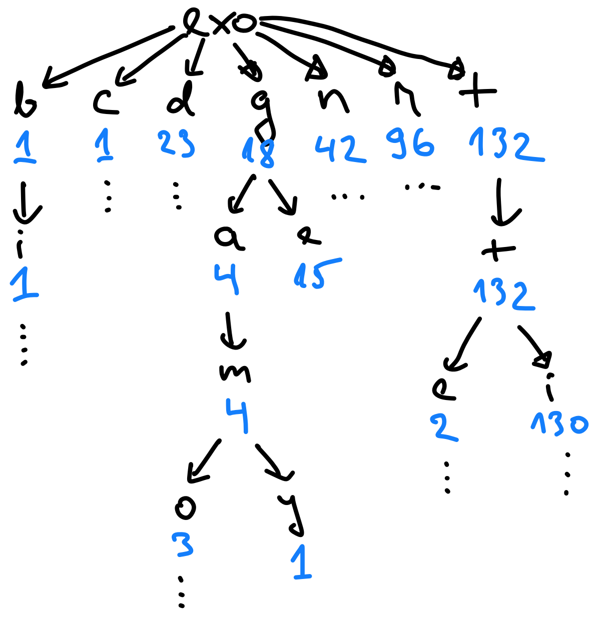
Possibilities

exobiology
exocarp
exodus
exogamous
exogamy
exogenous
exonerate
exoneration
exorbitant
exorbitance
exorcism
exorcist
exorcize
exordium
exoteric
exotic

Likelihoods

exobiology	1
exocarp	1
exodus	23
exogamous	3
exogamy	1
exogenous	15
exonerate	30
exoneration	12
exorbitant	43
exorbitance	5
exorcism	16
exorcist	19
exorcize	12
exordium	1
exoteric	2
exotic	130

Trie of possibilities



Discrete random variables

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

$$\begin{tikzcd} \bullet \arrow[r, "M_A"] & A \arrow[r, "M_{A\to B}"] & B \arrow[r, "M_{B\to C}"] & C \arrow[r, "M_{C\to D}"] & D \arrow[r, "M_{D\to E}"] & E \end{tikzcd}$$

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 $\Pr\left[x_1\right]$ given as vectors
- state transition probabilities $\Pr\left[x_{i+1}|x_i\right]$ given as matrices

What questions can we ask?

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

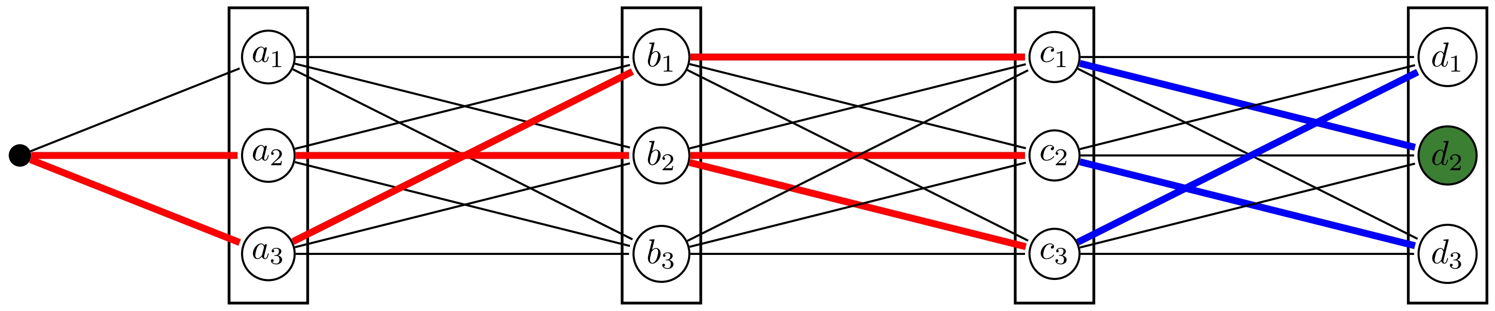
Stationary distribution: What happens if we run the chain infinitely long?
▷ Getting samples from an unnormalised posterior, optimisation tasks.

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.

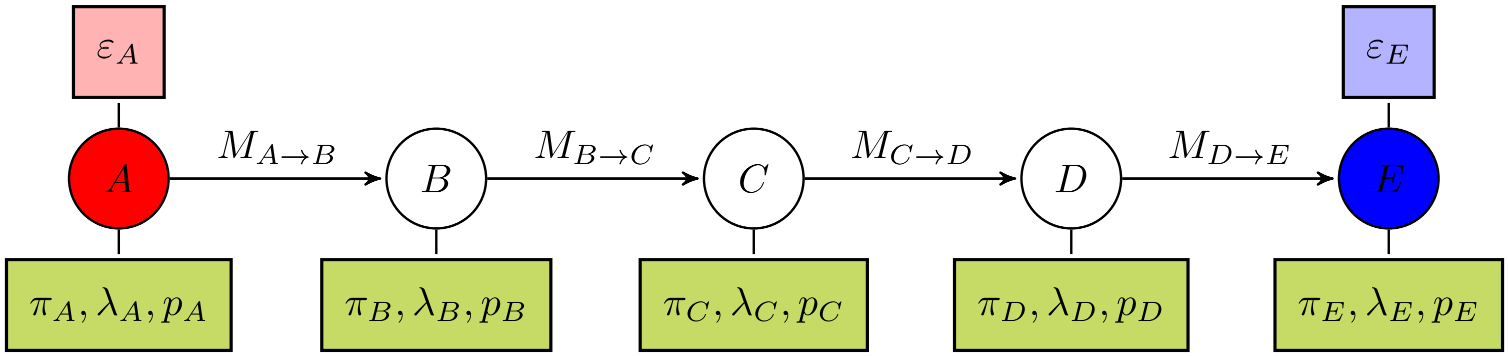
Posterior maximisation in a chain



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

- ▷ The log-posterior $\log \Pr[\boldsymbol{x}|\text{evidence}]$ 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.

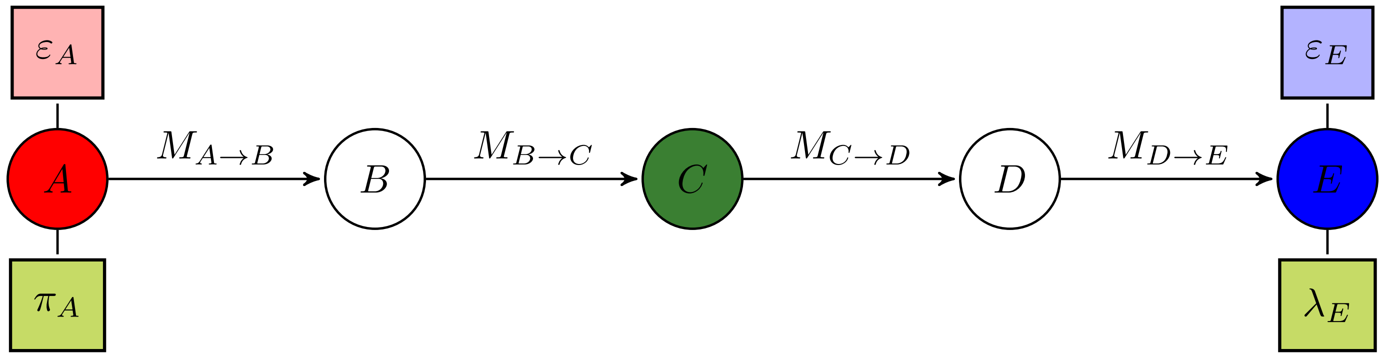
Belief propagation in a chain



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 beginning of chain.
- Downstream evidence$^-$ is the evidence at the end of chain.
- Attributes π_V, λ_V, p_V are needed to compute marginal distributions.

Initialisation

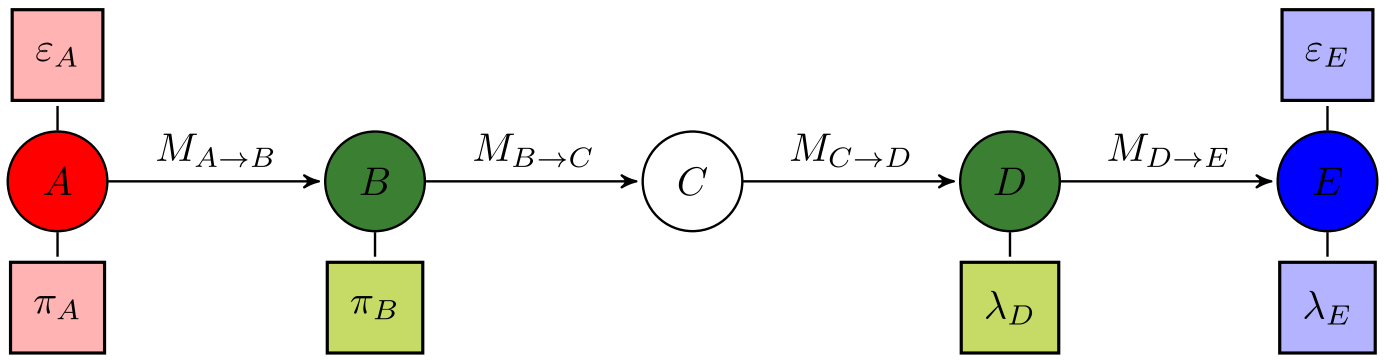


- ▷ Direct evidence ε_V determines the value of V.
- ▷ Indirect evidence ε_V determines the value distribution for V.
- ▷ We can assign the prior for the first and likelihood for the last node

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

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

Belief propagation



Inference goal

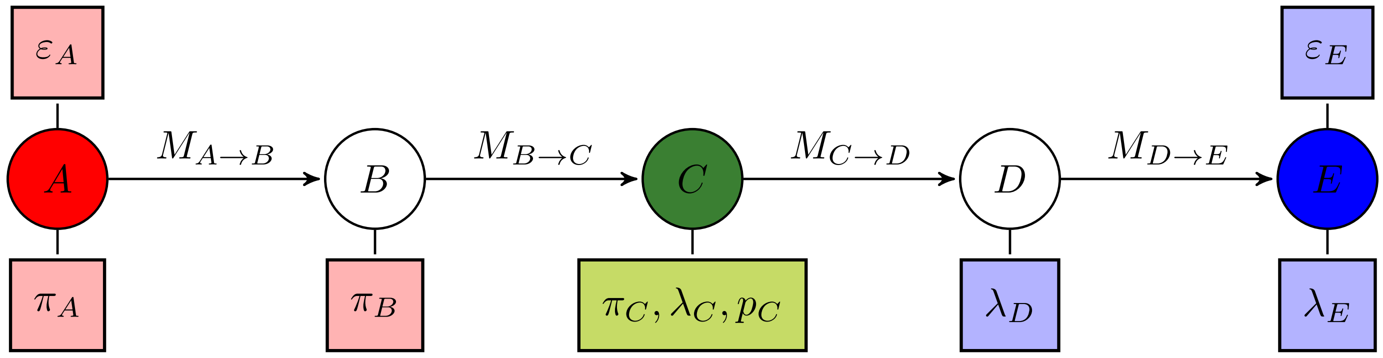
$$\pi_B(b) = \Pr\left[b|\text{evidence}^+\right]$$

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

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

$$\begin{tikzcd} \bullet \arrow[r, "\boldsymbol{\beta}[\cdot]"] & X_1 \arrow[r, "\boldsymbol{\alpha}[\cdot,\cdot]"] & X_2 \arrow[r, "\boldsymbol{\alpha}[\cdot,\cdot]"] & X_3 \arrow[r, "\boldsymbol{\alpha}[\cdot,\cdot]"] & X_4 \arrow[r, "\boldsymbol{\alpha}[\cdot,\cdot]"] & X_5 \end{tikzcd}$$

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

$$\begin{aligned}\ell[\boldsymbol{x}] &= \log \underbrace{\Pr\left[x_1\right]}_{\beta[x_1]} + \sum_{i=1}^{n-1} \log \underbrace{\Pr\left[x_{i+1}|x_i\right]}_{\alpha[x_i, x_{i+1}]} \\ &= \log \beta[x_1] + \sum_{u_1, u_2} k(u_1, u_2) \log \alpha[u_1, u_2]\end{aligned}$$

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

$$\begin{aligned}\log p[\boldsymbol{\alpha}, \boldsymbol{\beta}|\boldsymbol{x}] = &\sum_{u_1} k(u_1) \log \beta[u_1] + \log p(\boldsymbol{\beta}) \\ &+ \sum_{u_1, u_2} k(u_1, u_2) \log \alpha[u_1, u_2] + \sum_{u_1} \log p(\boldsymbol{\alpha}[u_1, \cdot])\end{aligned}$$

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

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

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

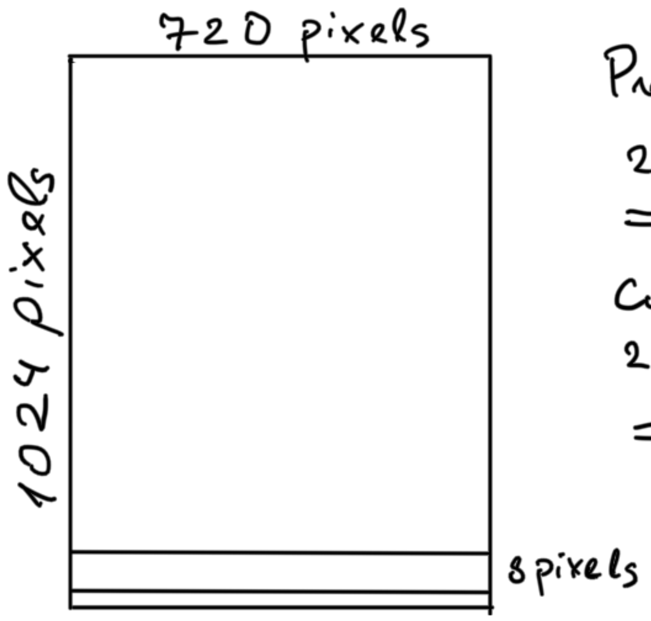
where each of these is equivalent to optimisation of dice throwing posterior. Thus Maximum Aposteriori estimates for parameters are

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

where

- ▷ $*$ is a wildcard symbol in the count queries
- ▷ m is the number of states and c is a constant for Laplacian smoothing.

Why discrete Markov chains fail in practice?



Previous state

256 colors × 720 pixels × 8 pixels

$\Rightarrow 256^{720\times 8}$

Current state

256 colors × 720 pixels

$\Rightarrow 256^{720}$

The number of possible observation is to big already for 8×8 patch:

$$256^{8\times 8} \times 256^{8} \times 2^{10} = 2^{8\times 8\times 8+8\times 8+10} = 2^{586}$$

8×9 pathces are needed to estimate probabilities within ± 3 percent points.

Two ways to build continious Markov chains

- ▷ Replace a list of discrete states with continous variable.
  - ◇ We get 8×8 input features and 8 output features.
  - ◇ We need 8 functions of type $f_i : \mathbb{R}^{64} \to \mathbb{R}$ to fix expectation.
  - ◇ We need 8 functions of type $g_i : \mathbb{R}^{64} \to \mathbb{R}$ to fix variance.
  - ◇ If we use linear functions then we need $8 \times 65 \times 2$ parameters.
- ▷ Embed discrete states into lower-dimensional feature space.
  - ◇ Ideally, these features are have semantical meaning.
  - ◇ In practice, features are fixed up to affine transformations.
  - ◇ Thus, features do not have clear interpretation.

Hidden Markov Model

$$\begin{tikzcd}
 & Y_1 & Y_2 & Y_3 & Y_4 & Y_5 \\
\bullet \arrow[r, "\boldsymbol{\beta}{[\cdot]}"] & X_1 \arrow[u, "\boldsymbol{\delta}{[\cdot,\cdot]}"] \arrow[r, "\boldsymbol{\alpha}{[\cdot,\cdot]}"] & X_2 \arrow[u, "\boldsymbol{\delta}{[\cdot,\cdot]}"] \arrow[r, "\boldsymbol{\alpha}{[\cdot,\cdot]}"] & X_3 \arrow[u, "\boldsymbol{\delta}{[\cdot,\cdot]}"] \arrow[r, "\boldsymbol{\alpha}{[\cdot,\cdot]}"] & X_4 \arrow[u, "\boldsymbol{\delta}{[\cdot,\cdot]}"] \arrow[r, "\boldsymbol{\alpha}{[\cdot,\cdot]}"] & X_5 \arrow[u, "\boldsymbol{\delta}{[\cdot,\cdot]}"]
\end{tikzcd}$$

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

- ▷ parameter estimation
- ▷ filtering, smoothing, prediction

Applications

Modelling and prediction

▷ stock prices

▷ linear control algorithms

Sequence annotation

▷ fraud detection

▷ change detection

▷ functional motifs of DNA sequences

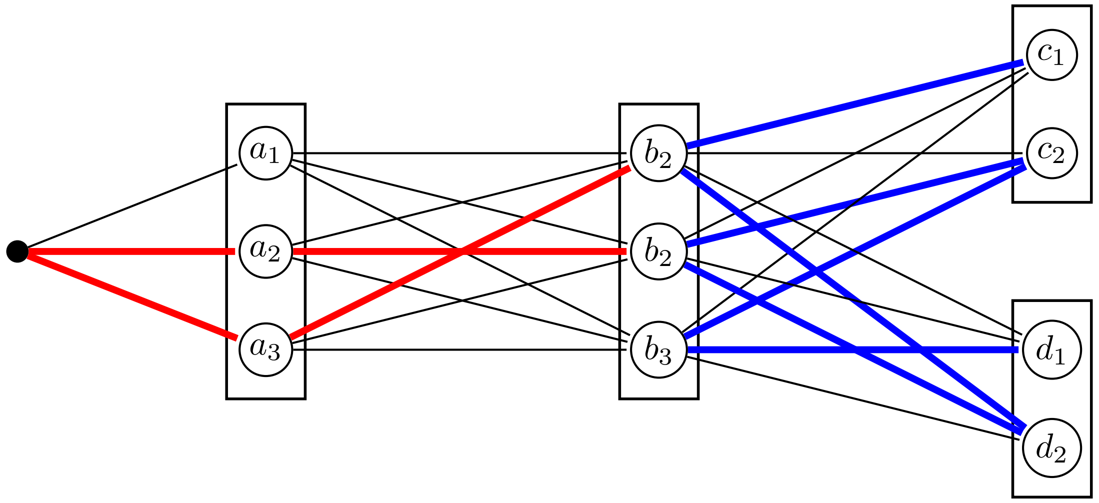
Decoding

▷ speech recognition

▷ communication over a nosy channels

▷ object tracking and data fusion

Posterior maximisation in a tree

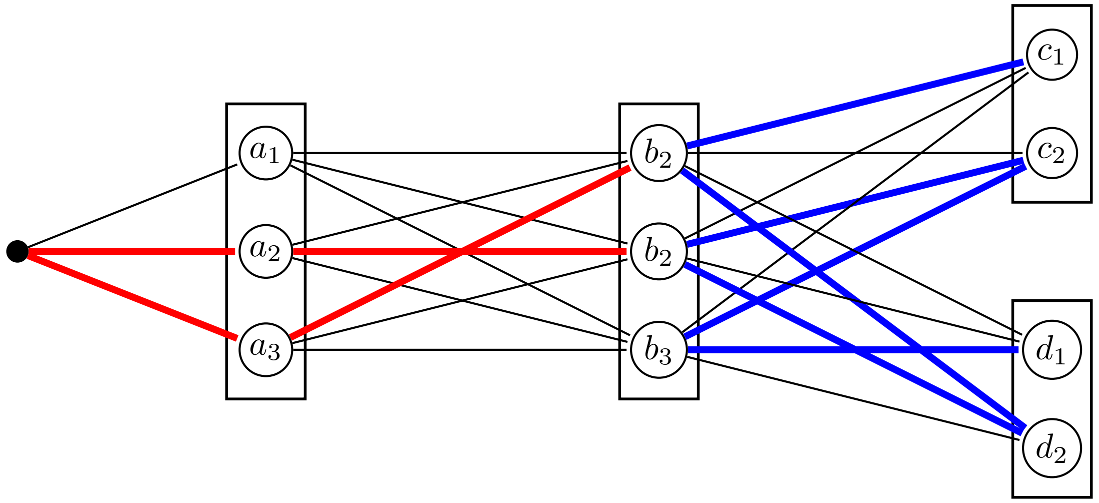


Inference goal. Given evidence at the ends of the chain find the sequence of states $\boldsymbol{x}$ that maximise the posterior probability $\Pr[\boldsymbol{x}|\text{evidence}]$.

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

▷ We must find a tree with maximal weight.

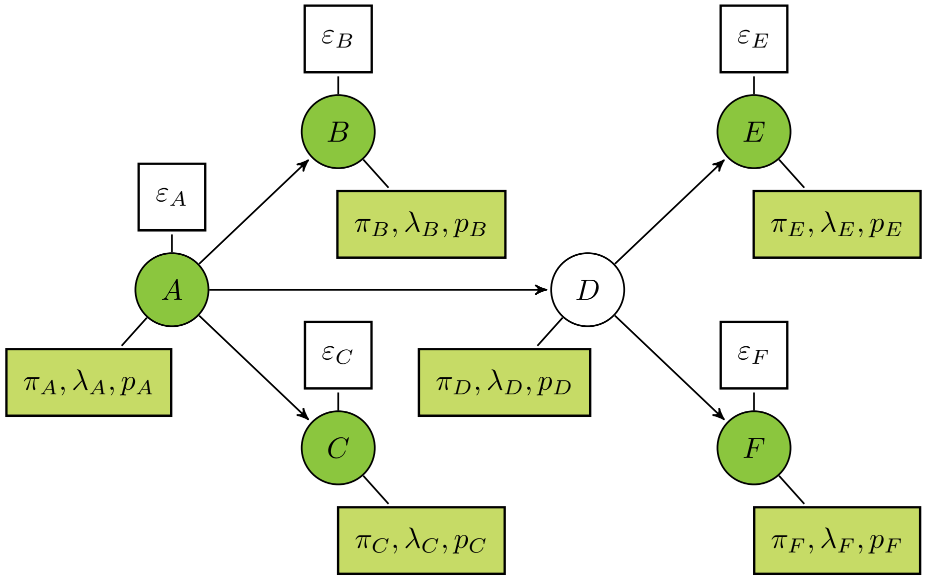
Decomposition into subtasks



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.

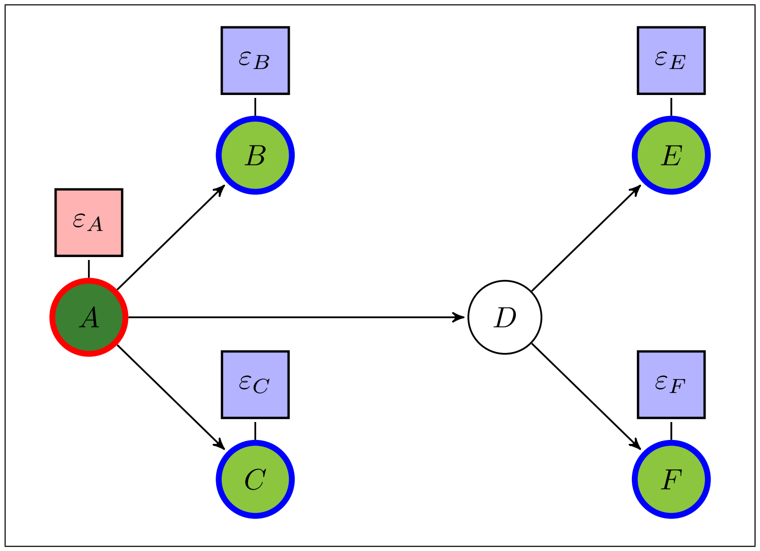
Belief propagation in a tree

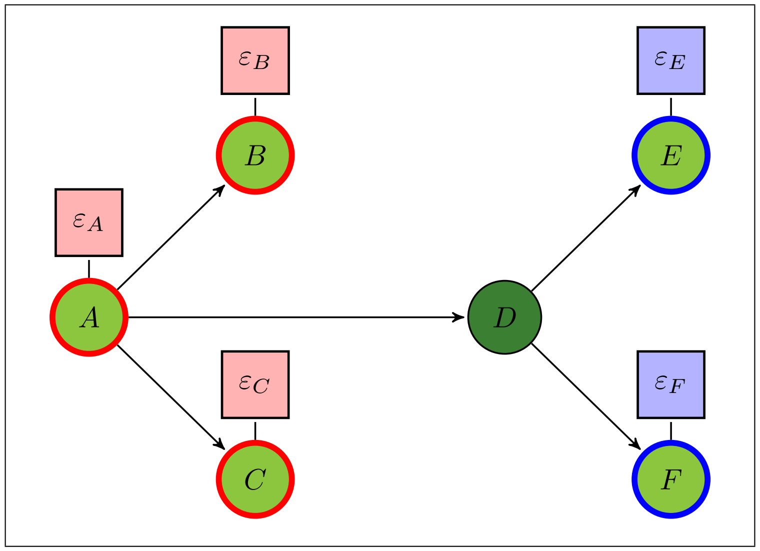


Inference goal. Given evidence at the ends of the leafs and the root of tree find marginal posterior probabilities for each node in the tree.

- 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.

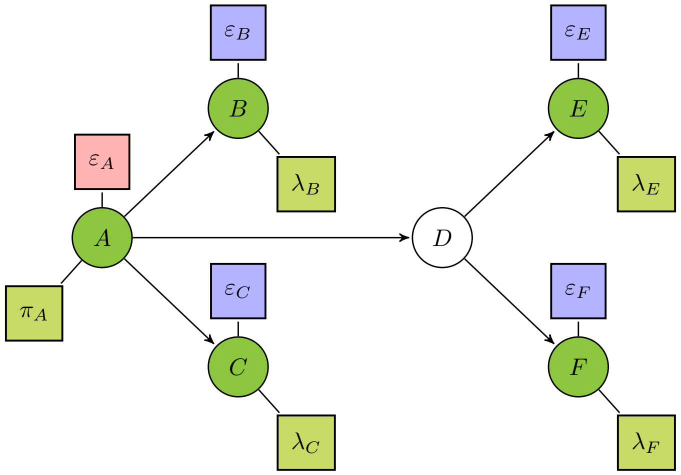
Evidence decomposition





- ▷ 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.

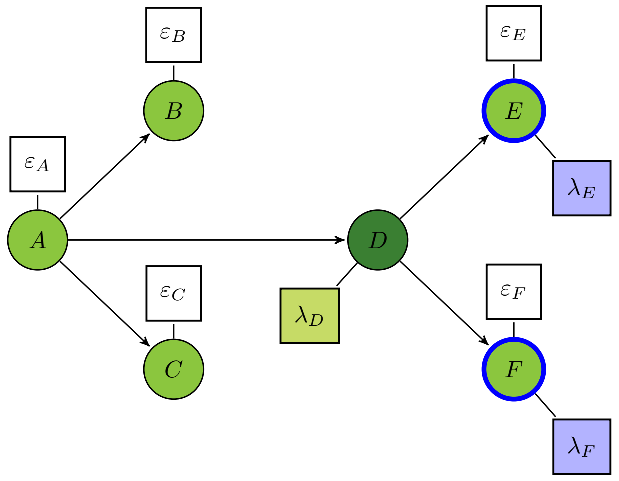
$$\pi_A(a) = \Pr\left[A = a | \text{evidence}^{+}(A)\right] = \Pr\left[A = a | \varepsilon_A\right]$$

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

$$\dots$$

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

Likelihood propagation



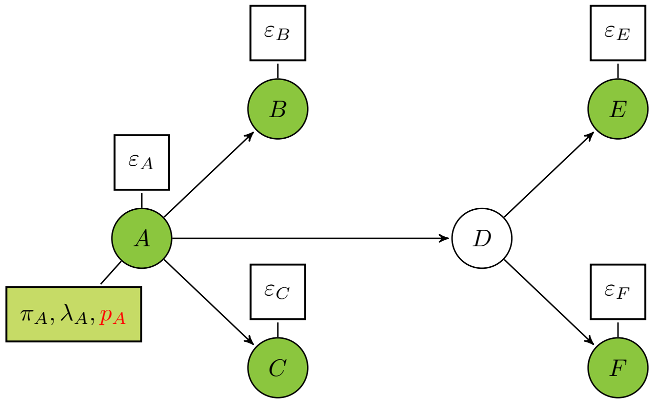
Inference goal

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

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



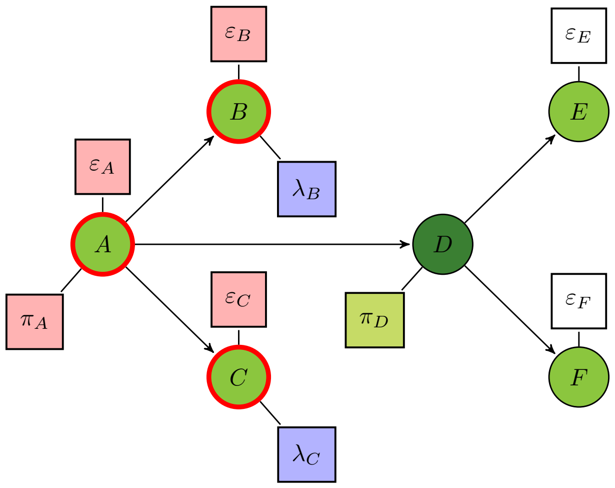
Inference goal

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

Iterative propagation rule

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

Prior propagation



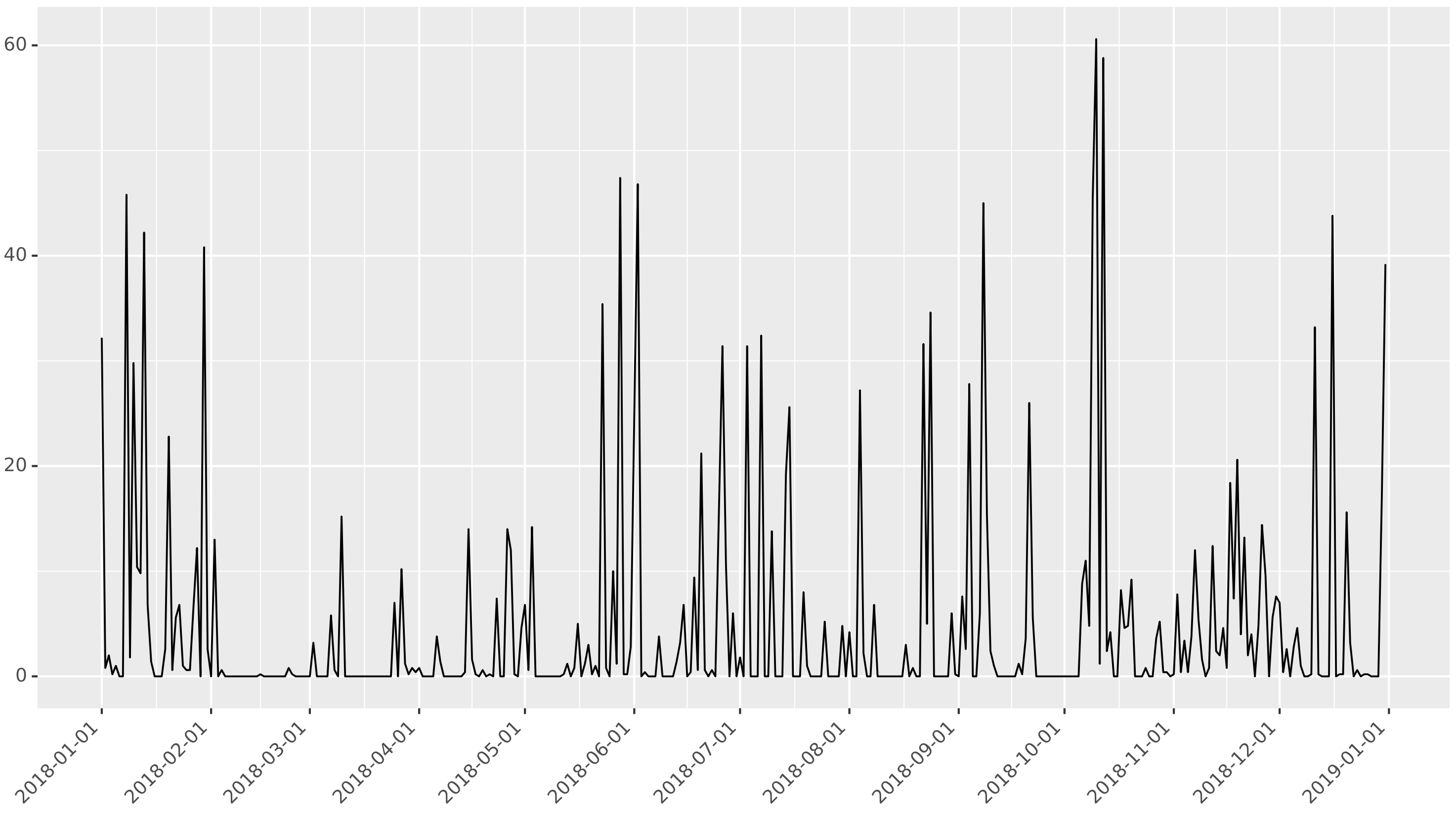
Inference goal

$$\begin{aligned}\pi_D(d) &= \Pr\left[D = d | \text{evidence}^+(D)\right]\\ &= \Pr\left[D = d | \text{evidence}^+(A), \text{evidence}^-(B), \text{evidence}^-(C)\right]\end{aligned}$$

Iterative propagation rule

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

Application on rainfall data



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

Modelling with Hidden Markov Model

$$\begin{tikzcd}
 & Y_1 & Y_2 & Y_3 & Y_4 & Y_5 \\
\bullet \arrow[r, "\boldsymbol{\beta}[\cdot]"] & X_1 \arrow[u, "\boldsymbol{\delta}[\cdot,\cdot]"] \arrow[r, "\boldsymbol{\alpha}[\cdot,\cdot]"] & X_2 \arrow[u, "\boldsymbol{\delta}[\cdot,\cdot]"] \arrow[r, "\boldsymbol{\alpha}[\cdot,\cdot]"] & X_3 \arrow[u, "\boldsymbol{\delta}[\cdot,\cdot]"] \arrow[r, "\boldsymbol{\alpha}[\cdot,\cdot]"] & X_4 \arrow[u, "\boldsymbol{\delta}[\cdot,\cdot]"] \arrow[r, "\boldsymbol{\alpha}[\cdot,\cdot]"] & X_5 \arrow[u, "\boldsymbol{\delta}[\cdot,\cdot]"]
\end{tikzcd}$$

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

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

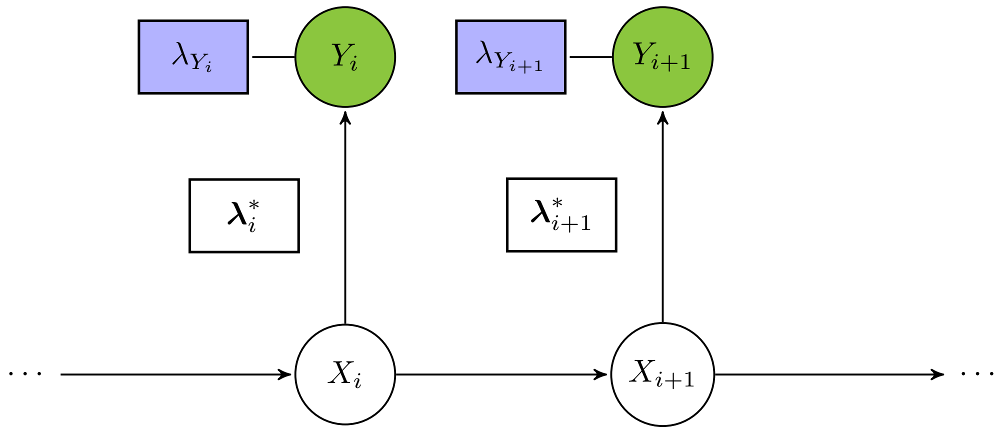
$$\boldsymbol{\alpha} = \begin{pmatrix} 0.95 & 0.05 \\ 0.05 & 0.95 \end{pmatrix}$$

Emission distributions

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

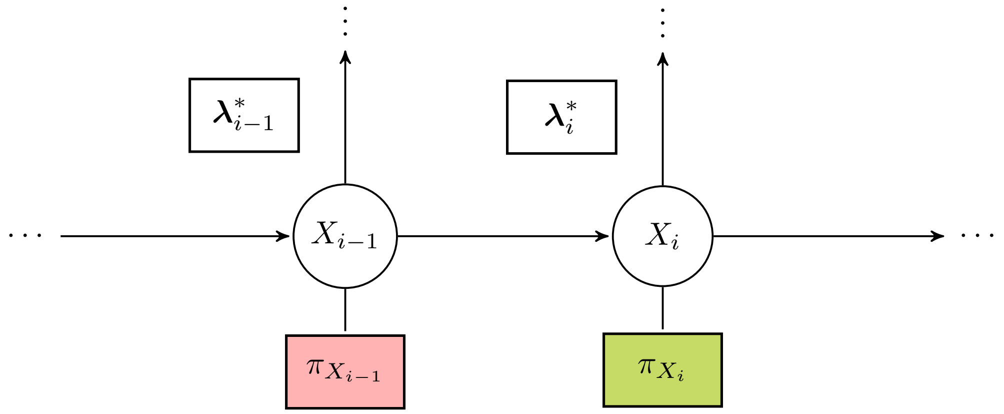
$$Y_i | X_i = 1 \sim \mathcal{N}(\mu_1, \sigma_1)$$

Belief propagation. Initialisation



- ▷ 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 $\lambda_i^*(x_i) = \Pr\left[Y_i = y_i | x_i\right]$ is a finite vector.

Prior propagation. Filtering



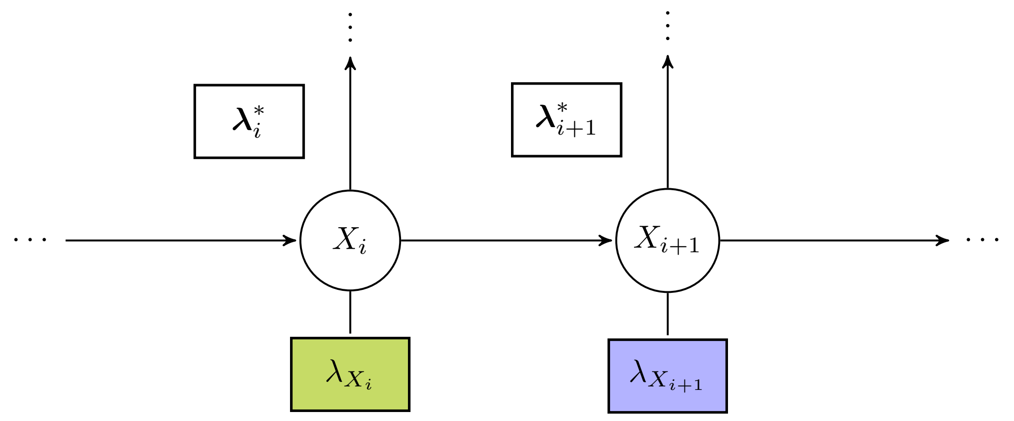
Prior propagation rule yields

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

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



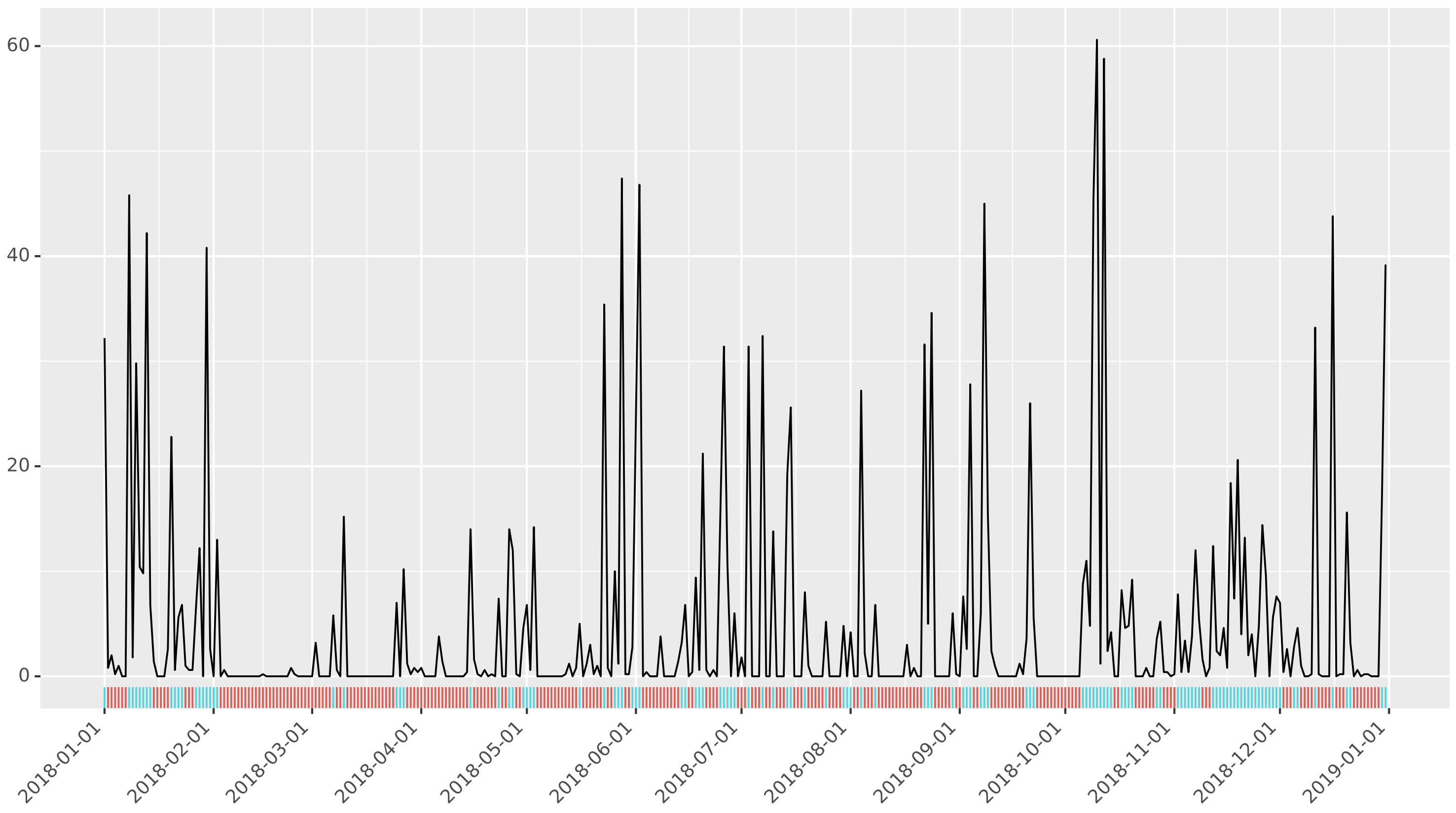
Likelihood propagation rule yields

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

Now we can do smoothing

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

Annotated rainfall data


60
40
20
0
2018-01-01
2018-02-01
2018-03-01
2018-04-01
2018-05-01
2018-06-01
2018-07-01
2018-08-01
2018-09-01
2018-10-01
2018-11-01
2018-12-01
2019-01-01