

Markov Chains and Hidden Markov Models

Aayush Bajaj

August 31, 2026

time is just another dimension to factorise over. a bayesian network whose variables are indexed by time steps, with the same local structure copied at every step, is a *dynamic bayesian network* — and its two simplest members, the markov chain and the hidden markov model, carry a startling share of applied probability: language models before transformers, speech recognition, robot localisation, pagerank, and the mcmc machinery behind approximate inference.

the modern neural take on the same sequence problem is the rnn — same graph, learned transition function

Contents

1	markov chains	1
1.1	where is the chain going? the mini-forward algorithm	2
1.2	stationary distributions	2
2	hidden markov models	2
2.1	filtering: the forward algorithm	3
2.2	decoding: the viterbi algorithm	3
3	particle filtering	4
4	dynamic bayesian networks and the kalman filter	4
5	results	4
6	see also	5

1 markov chains

a **markov chain** is a sequence of variables $X_1, X_2, \dots$ over a common state space, forming the world's simplest bayesian network — a path:

$$P(X_1, \dots, X_n) = P(X_1) \prod_{t=2}^n P(X_t \mid X_{t-1}). \quad (1)$$

two assumptions do all the compressing. the **markov assumption** (first-order): the future is independent of the past given the present — each X_t has a single parent X_{t-1} . the **stationarity assumption**: the transition model $P(X_t \mid X_{t-1})$ is the same table at every step, the *dynamics*. so the whole process is two objects — an initial distribution $P(X_1)$ and a $|X| \times |X|$ transition matrix — instead of $O(|X|^n)$ joint entries.

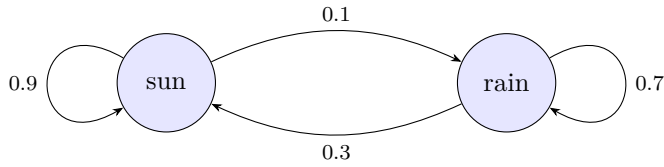


Figure 1: a two-state weather chain, drawn as a state-transition diagram. the same object, drawn as a bayesian network, is the path $X_1 \rightarrow X_2 \rightarrow X_3 \rightarrow \dots$ with this table shared along every edge.

1.1 where is the chain going? the mini-forward algorithm

the state distribution rolls forward by one matrix-vector product per step — variable elimination specialised to a path:

$$P(x_t) = \sum_{x_{t-1}} P(x_t | x_{t-1}) P(x_{t-1}), \quad O(n |X|^2) \text{ for } n \text{ steps.} \quad (2)$$

1.2 stationary distributions

for many chains the state distribution forgets its start: it converges to a **stationary distribution** π , the fixed point

$$\pi(x) = \sum_{x'} P(x | x') \pi(x'), \quad \sum_x \pi(x) = 1. \quad (3)$$

for the weather chain: $\pi(\text{sun}) = 0.9 \pi(\text{sun}) + 0.3 \pi(\text{rain})$ with normalisation gives $\pi = (0.75, 0.25)$ — three sunny days in four, no matter how it started.

when does this work? two structural conditions:

- **irreducible** — every state reachable from every other (the transition diagram is strongly connected). guarantees a *unique* stationary distribution.
- **aperiodic** — the chain is not trapped in deterministic cycles (returns to a state are possible at irregular times; a single self-loop anywhere in an irreducible chain suffices).

an irreducible, aperiodic (together: **ergodic**) chain converges to its unique π from any initial distribution — though possibly slowly; **mixing time** is the practical worry.

the two celebrated applications: pagerank is the stationary distribution of a random web surfer, and mcmc runs a chain engineered so that its stationary distribution is the posterior you want

2 hidden markov models

usually the state itself is not observable — only noisy **emissions** of it. an **hmm** attaches an observed child E_t to each hidden X_t :

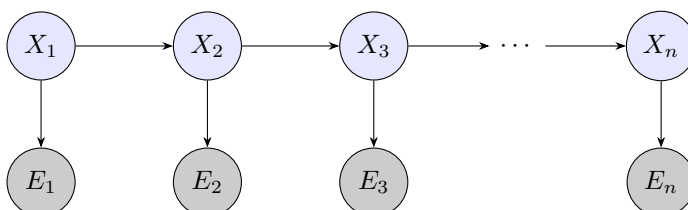


Figure 2: an hmm unrolled. shaded emissions are observed; the hidden chain is recovered only through them. parameters: $P(X_1)$, transition $P(X_t | X_{t-1})$, emission $P(E_t | X_t)$ — all shared across t .

the factorisation, straight off the graph:

$$P(X_{1:n}, E_{1:n}) = P(X_1) P(E_1 | X_1) \prod_{t=2}^n P(X_t | X_{t-1}) P(E_t | X_t), \quad (4)$$

and the d-separations that make inference cheap: $X_t \perp X_{1:t-2}, E_{1:t-2} | X_{t-1}$ and $E_t \perp$ everything earlier $| X_t$. the canonical toy: a phd student in a windowless lab infers the weather (hidden) from whether the supervisor carries an umbrella (emission); the serious versions are speech recognition (phonemes from acoustics), part-of-speech tagging, and robot localisation (position from sensor readings).

2.1 filtering: the forward algorithm

filtering maintains the **belief state** $B(X_t) = P(X_t | e_{1:t})$ online, alternating two steps:

- **passage of time** — push the belief through the dynamics (exactly the mini-forward update):

$$P(X_{t+1} | e_{1:t}) = \sum_{x_t} P(X_{t+1} | x_t) B(x_t); \quad (5)$$

- **observation** — multiply in the evidence likelihood and renormalise:

$$B(X_{t+1}) \propto P(e_{t+1} | X_{t+1}) P(X_{t+1} | e_{1:t}). \quad (6)$$

this is the **forward algorithm**: $O(n|X|^2)$ time, $O(|X|)$ memory — variable elimination along the temporal order, with the normalisation constants multiplying up to the probability of evidence $P(e_{1:n})$.

2.2 decoding: the viterbi algorithm

the most probable *trajectory* — the mpe of the chain — is not the sequence of individually most probable states. picture the **state trellis**: states stacked vertically, time running horizontally, edges weighted by transition $\times$ emission probabilities. a path through the trellis is a trajectory, its weight the product along it; the forward algorithm *sums* path weights into each node, viterbi *maximises*:

$$s_t[x_t] = P(e_t | x_t) \sum_{x_{t-1}} P(x_t | x_{t-1}) s_{t-1}[x_{t-1}] \quad (\text{forward}) \quad (7)$$

$$m_t[x_t] = P(e_t | x_t) \max_{x_{t-1}} P(x_t | x_{t-1}) m_{t-1}[x_{t-1}] \quad (\text{viterbi}) \quad (8)$$

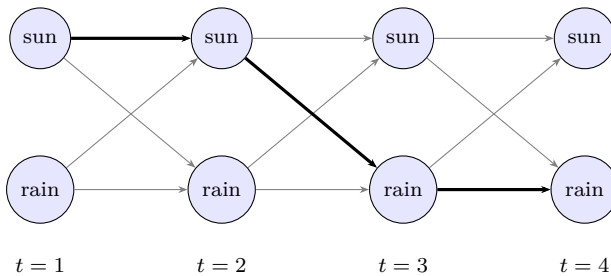


Figure 3: the state trellis for a two-state hmm over four steps. forward sums over the incoming edges of each node; viterbi keeps the single best (bold path) and backtracks from the best final node.

each node also records *which* predecessor achieved the max; **backtracking** from the best final state reads the trajectory off in reverse. two numerical realities: products of many probabilities vanish below floating-point range, so implementations either renormalise every step or — cleaner — work with **log probabilities**, replacing multiplications with additions; and ties in the max must be broken consistently or backtracking wanders. both algorithms are $O(n|X|^2)$, linear in sequence length.

viterbi with a beam — keep only the top-k trellis nodes per step — was state of the art in speech for decades

3 particle filtering

when the state space is huge or continuous ($|X|^2$ unaffordable — a robot’s pose in a warehouse), replace the belief *table* with a population of N samples (“particles”) and push the samples through the same two-step loop:

1. **elapse time** — move each particle by sampling its successor from the transition model;
2. **observe** — do *not* sample the evidence: weight each particle by $P(e_t | x)$;
3. **resample** — draw N fresh particles proportional to weight, returning to an unweighted population (equivalent to renormalising, and concentrating particles where the evidence points).

this is sequential importance resampling — likelihood weighting made to work over time by pruning hopeless samples before their weights collapse. accuracy grows with N ; the failure mode is *particle deprivation*, all particles missing the true state.

4 dynamic bayesian networks and the kalman filter

nothing above needed the hidden state to be a single variable. a **dynamic bayesian network** copies an arbitrary “slice” network at each time step, with edges within a slice and from slice $t - 1$ to t — an hmm whose state factorises. exact filtering must in general drag around the joint over the full slice (the factorisation does *not* survive conditioning on evidence), so dbn inference typically falls back to the jointree algorithm per window, or to particle methods.

the continuous cousin: make the hmm’s transition and emission models **linear gaussian** — $X_{t+1} = AX_t + \varepsilon$, $E_t = HX_t + \delta$ — and the belief state stays exactly gaussian forever; propagating its mean and covariance is the **kalman filter**. same graph as the hmm, closed-form updates, the workhorse of tracking and control.

5 results

- **markov chain parameters** — initial distribution + shared transition matrix: $O(|X|^2)$ numbers for arbitrarily long sequences.
- **stationary distribution** — an irreducible chain has a unique π solving $\pi = \pi P$; adding aperiodicity, $P(X_t) \rightarrow \pi$ from every start.
- **forward algorithm** — filtering in $O(n|X|^2)$ time, $O(|X|)$ space; normalisers accumulate $P(e_{1:n})$.

- **viterbi algorithm** — mpe trajectory in $O(n|X|^2)$ by max-product on the trellis + backtracking; run in log-space.
- **forward = ve, viterbi = max-out ve** — both are variable elimination along the temporal order on the hmm's chain network.
- **particle filter** — sample-based filtering: propagate, weight by evidence, resample; cost $O(nN)$, exact as $N \rightarrow \infty$.
- **kalman filter** — exact filtering for linear gaussian dynamics; gaussian belief state propagated in closed form.

6 see also

- bayesian networks — hmms are just chain-structured ones
- exact inference — the general algorithms these specialise
- approximate inference — mcmc, built on ergodic markov chains
- recurrent neural networks — the neural sequence model on the same unrolled graph